Đồ án tốt nghiệp

Tính toán chuyển động chương trình và thiết kế robot MMR

Tính toán động học, mô phỏng chuyển động của robot **MMR** (**M**ini **M**obile **R**obot) và thiết kế, chế tạo mẫu robot **MMR**.

Đồ án được chia thành 2 phần :

- **Phần I**: Tính toán động học và mô phỏng chuyển động robot **MMR**.
  - -Chương 1: Cơ sở lý thuyết khảo sát bài toán động học robot.
  - -Chương 2: Áp dụng tính động học cho robot **MMR**.
  - -Chương 3: Phần mềm tính toán và mô phỏng.
- **Phần I** : Thiết kế và chế tạo mẫu robot **MMR**
  - **-**Chương 1: Lựa chọn cấu trúc robot **MMR**
  - **-**Chương 2 : Thiết kế cơ khí robot **MMR**

Em xin chân thành cảm ơn **T.S Phan Bùi Khôi** cùng toàn thể các thầy cô trong bộ môn cơ học ứng dụng đã tận tình hướng dẫn em hoàn thành đồ án này.

Mặc dù rất cố gắng nhưng do kiến thức và thời gian có hạn nên đồ án không tránh khỏi thiếu sót. Em mong nhận được sự chỉ bảo của các thầy, các cô trong bộ môn cũng như các bạn sinh viên, những người quan tâm đến robot.

PHẦN I

TÍNH TOÁN ĐỘNG HỌC VÀ MÔ PHỎNG CHUYỂN ĐỘNG ROBOT **MMR**

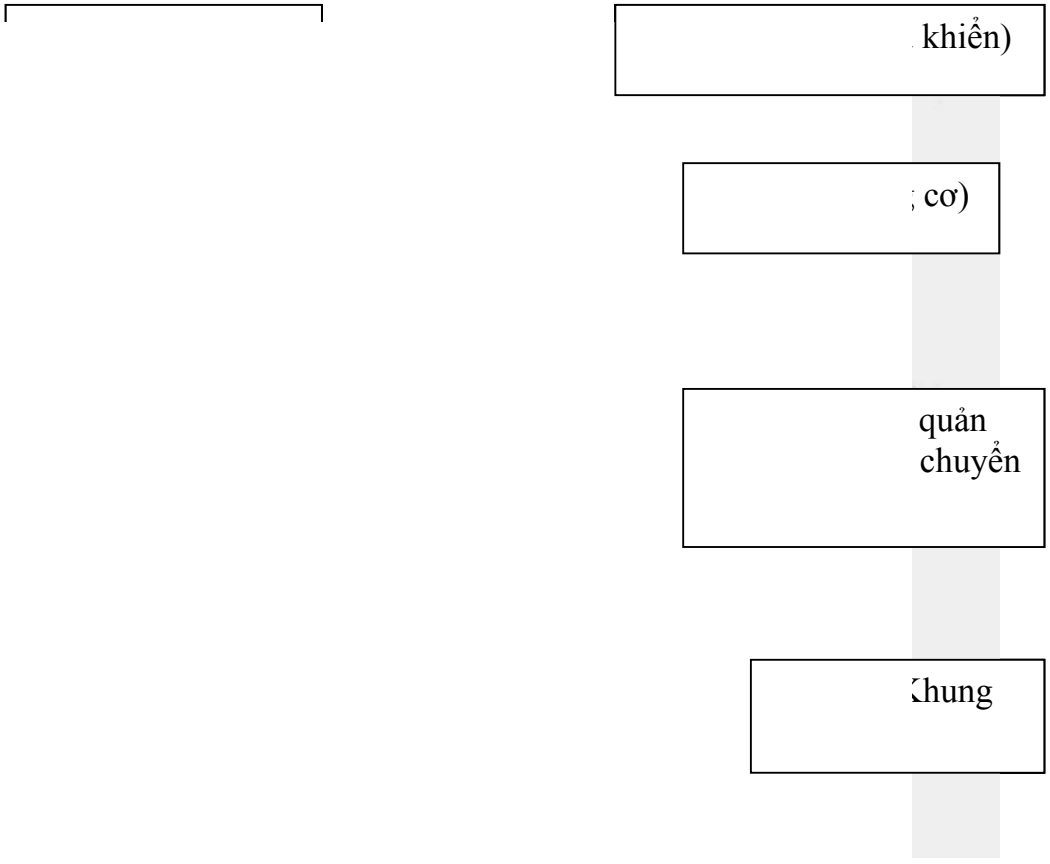
CHƯƠNG 1

CƠ SỞ LÝ THUYẾT KHẢO SÁT ĐỘNG HỌC ROBOT

1.1.Cấu trúc động học robot

1.1.1 Khái quát về robot

Cùng với sự phát triển không ngừng của các nghành khoa học kỹ thuật đặc biệt là lĩnh vực cơ khí, điện tử điều khiển và tin học đã làm cho robot ngày càng có những chức năng gần giống như con người nhiều hơn, trong robot có các bộ phận như cơ cấu chấp hành, hệ dẫn động và hệ thống điều khiển. Cơ cấu chấp hành cũng như cánh tay chân con người, hệ dẫn động chính là các cơ bắp và được trái tim con người tương ứng với động cơ đặt trong robot vận hành, hệ thống điều khiển là bộ não điều khiển mọi hoạt động của robot.



Cuộc sống ngày càng văn minh hiện đại, mức sống của người dân ngày càng được nâng cao, đòi hỏi phải nâng cao năng suất và chất lượng của sản phẩm. Vì vậy càng phải ứng dụng rộng rãi các phương tiện tự động hoá vào sản xuất nên càng tăng nhanh nhu cầu về ứng dụng robot để tạo ra các hệ thống sản xuất tự động và linh hoạt.

Robot có những đặc điểm nổi trội đó là:

- Có thể thực hiện công việc một cách bền bỉ, không biết mệt mỏi nên chất lượng sản phẩm được giữ ổn định. Giá thành sản phẩm hạ do giảm được chi phí cho người lao động. Ở nước ta trong những năm gần đây ở nhiều doanh nghiệp, khoản chi phí về lương bổng cũng chiếm tỷ lệ khá cao trong giá thành sản phẩm, càng cần phải ứng dụng công nghệ robot vào dây chuyền sản xuất.
- Nhất là ở nhiều nơi hiện nay cũng cần ứng dụng công nghệ robot để cải thiện điều kiện lao động vì trong thực tế sản xuất người lao động phải làm việc suốt buổi trong môi trường bụi bặm, ẩm ướt, ồn ào…quá mức cho phép nhiều lần. Thậm chí phải làm việc trong môi trường độc hại, nguy hiểm đến sức khoẻ con người.
- Mặt khác, khi áp dụng công nghệ robot vào sản xuất ta cũng cần lưu ý và phân tích kỹ toàn bộ hệ thống sản xuất sao cho phù hợp với các nguyên công và phù hợp với tình hình sản xuất của nhà máy. Cần xét đến đầy đủ các chi phí phụ và hiệu quả mang lại cho toàn bộ hệ thống. Khi xác định đưa robot vào hệ thống sản xuất thì cũng cần phải xét xem khả năng liệu robot có thay thế được hay không và có hiệu quả hơn không. Vì trong thực tế sản xuất cho thấy xu hướng thay thế hoàn toàn bằng robot nhiều khi không hiệu quả bằng việc giữ lại một số công đoạn mà cần phải có sự khéo léo của con người.
- Kỹ thuật robot có ưu điểm quan trọng nhất là tạo nên khả năng linh hoạt hóa sản xuất. Mà trong đó kĩ thuật robot và máy vi tính đã đóng

vai trò quan trọng trong việc tạo ra các dây chuyền tự động linh hoạt.Vì vậy trong những năm gần đây không những chỉ các nhà khoa học mà cả các nhà sản xuất đã tập trung sự chú ý vào việc hình thành và áp dụng các hệ sản xuất linh hoạt.

So với lúc mới ra đời, ngày nay công nghệ robot đã có những bước phát triển vượt bậc. Đặc biệt là vào những năm 60 của thế kỉ trước, với sự góp mặt của máy tính. Ở giai đoạn đầu người ta rất quan tâm đến việc tạo ra những cơ cấu tay máy nhiều bậc tự do, được trang bị cảm biến để thực hiện những công việc phức tạp. Ngày càng có những cải tiến quan trọng trong kết cấu các bộ phận chấp hành, tăng độ tin cậy của các bộ phận điều khiển, tăng mức thuận tiện và dễ dàng khi lập trình. Tăng cường khả năng nhận biết và xử lý tín hiệu từ môi trường làm việc để mở rộng phạm vi ứng dụng cho robot.

Trong tương lai số lượng lao động được thay thế ngày càng nhiều vì một mặt giá thành robot ngày càng giảm do mặt hàng vi điện tử liên tục giảm giá đồng thời chất lượng liên tục tăng. Mặt khác chi phí về lương và các khoản phụ cấp cho người lao động ngày càng tăng. Robot ngày càng vạn năng hơn để có thể làm được nhiều việc trên các dây chuyền.

Công đoạn lắp ráp thường chiếm tỷ lệ cao so với tổng thời gian sản xuất trên toàn bộ dây chuyền. Công việc lại đòi hỏi phải cẩn thận, nhẹ nhàng tinh tế và chính xác. Nên nếu là công nhân thì cần phải thợ có tay nghề cao và làm việc đơn điệu, căng thẳng. Robot đã có mặt nhiều trên các công đoạn lắp ráp phức tạp do được thừa hưởng kĩ thuật cảm biến, kĩ thuật tin học với những ngôn ngữ lập trình bậc cao.

Robot tự hành cũng sẽ phát triển mạnh trong tương lai, có thể đi được bằng chân để thích hợp với mọi địa hình ví dụ như có thể tự leo bậc thang… Việc tạo ra các cơ cấu chấp hành cơ khí vừa bền vững, nhẹ nhàng chính xác và linh hoạt như chân tay người là đối tượng nghiên cứu chủ yếu.

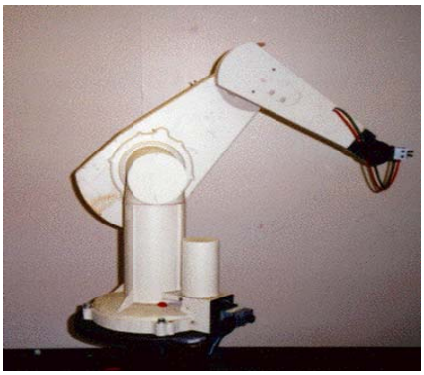
Kỹ thuật robot cũng từng bước áp dụng các kết quả nghiên cứu về trí khôn nhân tạo và đưa vào ứng dụng trong công nghiệp. Cải tiến và bổ xung các *modul* cảm biến và các *modul* phần mềm phù hợp có thể cải tiến và thông minh hoá nhiều loại robot. Điều quan trọng là các cơ cấu chấp hành của robot phải hoạt động chính xác.

1.1.2 Cấu trúc động học robot

Ta có thể khái quát định nghĩa robot theo cách nhìn của cơ học là một chuỗi động, mỗi khâu được ghép với nhau bởi các khớp nối, hoạt động linh hoạt nhờ hệ dẫn động và được điều khiển bằng hệ thống điều khiển.

Dưới đây là một số hình robot liên tục được ứng dụng nhiều trong các lĩnh vực:

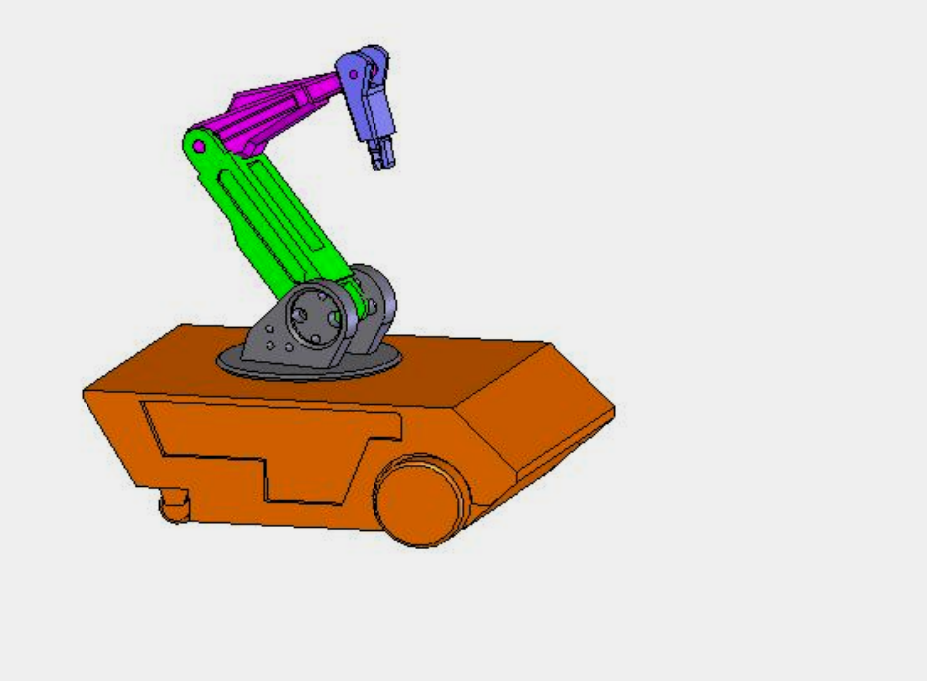
- Trong gia công cơ khí: thường sử dụng trong các máy hàn tự động, máy khoan, trong các dây truyền lắp ráp, v…v…
- Trong dây truyền sản xuất: Tham gia vào một số dây truyền sản xuất như gia công, phun sơn, đóng gói bao bì, v…v…
- Trong vận tải thường dùng để bốc xếp hàng hóa .



Hình 1.2 Robot Hipo

Hình 1.4 robot Kuka Hình 1.5 Laser Robotic.

Trong đồ án này em xin chọn mô hình robot **MMR** khảo sát là một chuỗi động hở, robot gồm 4 khâu và 4 khớp quay có thể thao tác trong không gian cố định (xe không di chuyển)(hình 1.6). Khâu cuối của robot có thể mang dụng cụ cắt, mỏ hàn,bàn kẹp, v…v…



Hình 1.6

1.2 Bậc tự do của robot

Cơ cấu tay của robot phải được cấu tạo sao cho khâu cuối phải có vị trí và theo một hướng nhất định nào đó và dễ dàng di chuyển dễ dàng trong vùng làm việc. Muốn vậy cơ cấu tay của robot phải đạt được một số bậc tự do chuyển động.

Để tính số bậc tự do của robot thì ta có nhiều cách tính dưới đây ta đưa ra cách tính dựa vào định lý *Gruebler*. Theo *Gruebler* thì bậc tự do *f* được tính theo công thức:

$$f = \lambda.(n-1) - \sum_{i=1}^{g} (\lambda - f_i) - f_0$$

(1.1)

Trong đó :

- f : Là số bậc tự do của cơ cấu.

- λ: Bậc tự do của một vật rắn không chụi liên kết trong không gian làm việc của robot (λ = 3 ứng với không gian làm việc trong mặt phẳng, λ = 6 ứng với không gian làm việc trong không gian).
- n : Số khâu (kể cả giá cố định).
- f_i : Số bậc tự do của khớp thứ i.
- g : Tổng số khớp của cơ cấu.
- f_0: Số bậc tự do thừa

Một số ví dụ:

- Số bậc tự do của mô hình robot trong đồ án
  - λ = 6 (Vì không gian làm việc trong không gian).
  - n = 5 (Số khâu của robot kể cả xe).
  - f_i = 1(Vì tất cả các khớp quay trong robot đều có 1 bậc tự do).
  - g = 4 (Tổng số khớp của cơ cấu).
  - f_0 =0 (Không có bậc tự do thừa).

  Bậc tự do của robot là :

$$f = \lambda.(n-1) - \sum_{i=1}^{g}(\lambda - f_i) - f_0$$

$$f = 6.(5-1) - \sum_{1}^{4}(6-1) - 0 = 4$$

- Laser Robotic (Hình 1.5)

  $\lambda = 6$, $n = 7$, $g = 6$, $f_i = 1$, f_0=0

  Bậc tự do của robot là : $f = \lambda.(n-1) - \sum_{i=1}^{g}(\lambda - f_i) - f_0$

$$f = 6.(7-1) - \sum_{1}^{6}(6-1) - 0 = 6$$

1.3 Phương pháp khảo sát bài toán động học

Sử dụng phương pháp ma trận chuyển *Denavit- Hartenberg*

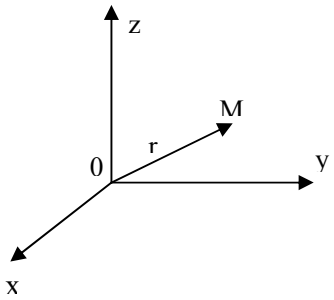
1.3.1 Tọa độ thuần nhất và ma trận biến đổi tọa độ thuần nhất

a) Vector điểm và tọa độ thuần nhất

Vector điểm dùng để mô tả vị trí của điểm trong không gian 3 chiều.

Xét điểm M trong không gian 3 chiều có thể biểu diễn bằng vector r trong hệ tọa độ Oxyz:

$$r = (r_x, r_y, r_z)^T \qquad (1.2)$$



Hình 1.7 Biểu diễn một điểm trong không gian

Vector $r = (r_x, r_y, r_z)^T$ trong không gian ba chiều, được bổ sung thêm một thành phần thứ tư và thể hiện bằng một vector mở rộng:

$$r = (\mu r_x, \mu r_y, \mu r_z, m)^T \qquad (1.3)$$

Đó là cách biểu diễn vector điểm trong không gian tọa độ thuần nhất.

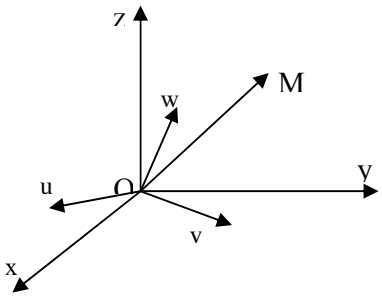
Như vậy có rất nhiều cách biểu diễn tọa độ trong không gian tọa độ thuần nhất, nó phụ thuộc vào giá trị của hệ số tỉ lệ μ. Nếu lấy μ = 1 thì các tọa độ biểu diễn bằng tọa độ có thực, vector mở rộng được viết lại như sau:

$$r = (r_x, r_y, r_z, 1)^T \qquad (1.4)$$

Nếu lấy $\mu \neq 1$ thì các tọa độ biểu diễn gấp μ lần tọa độ thực.

b) Quay hệ tọa độ dùng ma trận 3x3

Trước hết ta thiết lập quan hệ giữa 2 hệ tọa độ xyz và uvw chuyển động quay tương đối với nhau khi gốc O của 2 hệ vẫn trùng nhau.



Hình 1.8 Các hệ tọa độ

Gọi (i_x , j_y , k_z) và (i_u , j_v , k_w) là các vector đơn vị chỉ phương các trục Oxyz và Ouvw tương ứng.

Một điểm M nào đó được biểu diễn trong hệ tọa độ Oxyz bằng vector:

$$r_{xyz} = (r_x, r_y, r_z)^T$$

(1.5)

còn trong hệ tọa độ Ouvw bằng vector :

$$r_{uvw} = (r_u, r_v, r_w)^T$$

(1.6)

Như vậy:

$$\begin{cases} r = r_{xyz} = r_x i_x + r_y j_y + r_z k_z \\ r = r_{uvw} = r_u i_u + r_v j_v + r_w k_w \end{cases}$$

(1.7)

Từ đó ta có:

$$\begin{cases} r_x = i_x r = i_x i_u r_u + i_x j_v r_v + i_x k_w r_w \\ r_y = j_y r = j_x i_u r_u + j_y j_v r_v + k_y k_w r_w \\ r_z = k_z r = k_z i_u r_u + k_z j_v r_v + k_z k_w r_w \end{cases}$$

(1.8)

Hoặc viết dưới dạng ma trận:

$$\begin{bmatrix} r_x \\ r_y \\ r_z \end{bmatrix} = \begin{bmatrix} i_x i_u & i_x j_v & i_x k_w \\ j_y i_u & j_y j_v & j_y k_w \\ k_z i_u & k_z j_v & k_z k_w \end{bmatrix} \cdot \begin{bmatrix} r_u \\ r_v \\ r_w \end{bmatrix}$$

(1.9)

Gọi R là ma trận quay 3x3 với các phần tử là tích vô hướng 2 vector chỉ phương các trục tương ứng của 2 hệ tọa độ Oxyz và Ouvw. Phương trình (1.9) được viết lại:

$$\begin{cases} r_{xyz} = \mathbf{R}.r_{uvw} \\ r_{uvw} = \mathbf{R}^{-1}.r_{xyz} \end{cases} \qquad (1.10)$$

Có thể biểu diễn các phần tử ma trận R và R^{-1} như sau:

$$\mathbf{R} = \left[a_{ij} \right] = \begin{bmatrix} \cos(x,u) & \cos(x,v) & \cos(x,w) \\ \cos(y,u) & \cos(y,v) & \cos(y,w) \\ \cos(z,u) & \cos(z,v) & \cos(z,w) \end{bmatrix}$$

(1.11)

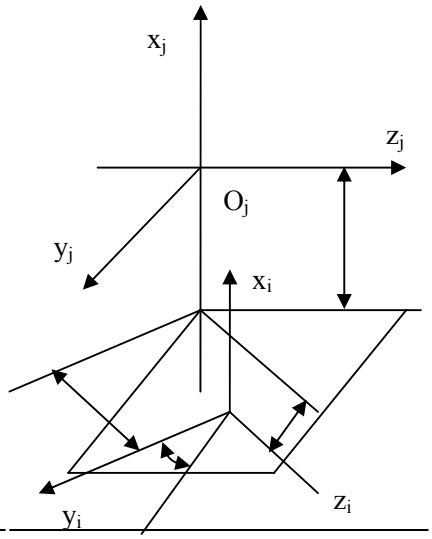
$$\mathbf{R}^{-1} = \left[b_{ij} \right] = \begin{bmatrix} \cos(u,x) & \cos(u,y) & \cos(u,z) \\ \cos(v,x) & \cos(v,y) & \cos(v,z) \\ \cos(w,x) & \cos(w,y) & \cos(w,z) \end{bmatrix}$$

(1.12)

Nhận xét: $\mathbf{R}^{-1} = \mathbf{R}^{T}$

c) Biến đổi tọa độ dùng ma trận thuần nhất

Bây giờ ta thiết lập quan hệ giữa 2 hệ tọa độ: hệ tọa độ $O_j\ x_j\ y_j\ z_j$ sang hệ tọa độ mới $O_i\ x_i\ y_i\ z_i$. Chúng không những quay tương đối với nhau mà tịnh tiến cả gốc tọa độ .



Hình 1.9

Gốc O_j xác định trong hệ tọa độ $O_i\ x_i\ y_i\ z_i$ bằng vector p:

$$p = (a, -b, -c, 1)^T$$

(1.13)

Giả sử vị trí của điểm M trong hệ tọa độ $O_j\ x_j\ y_j\ z_j$ được xác định bằng vector r_j : $r_j = (x_j , y_j , z_j ,1)^T$

(1.14)

và trong hệ tọa độ $O_i\ x_i\ y_i\ z_i$ được xác định bằng vector r_i :

$$r_i = (x_i , y_i , z_i ,1)^T$$

(1.15)

Dễ dàng thiết lập được các tọa độ:

$$\begin{cases} x_i = x_j + at_j \\ y_i = y_j \cos\varphi - z_j \sin\varphi - bt_j \\ t_i = t_j = 1 \end{cases}$$

(1.16)

Sắp xếp các hệ số ứng với x_j , y_j , z_j và t_j thành một ma trận:

$$\mathbf{T}_{ij} = \begin{bmatrix} 1 & 0 & 0 & a \\ 0 & \cos\varphi & -\sin\varphi & -b \\ 0 & \sin\varphi & \cos\varphi & -c \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

(1.17)

Phương trình biến đổi tọa độ được viết lại:

$$\mathbf{r}_i = \mathbf{T}_{ij}\ \mathbf{r}_j$$

(1.18)

Ma trận T_{ij} biểu thị bằng ma trận 4x4 như (1.17) gọi là ma trận thuần nhất. (1.17) được viết lại :

$$\begin{bmatrix} x_i \\ y_i \\ z_i \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & a \\ 0 & \cos\varphi & -\sin\varphi & -b \\ 0 & \sin\varphi & \cos\varphi & -c \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x_j \\ y_j \\ z_j \\ 1 \end{bmatrix}$$

(1.19)

Như vậy ta đã dùng ma trận thuần nhất để biến đổi vector mở rộng từ hệ tọa độ thuần nhất này sang hệ tọa độ thuần nhất kia. Sử dụng ma trận thuần nhất trong phép biến đổi tọa độ tỏ ra có nhiều ưu điểm, bởi vì trong ma trận 4x4 bao gồm cả thông tin về sự quay và về cả dịch chuyển tịnh tiến.

Ma trận thuần nhất T_{ij} được viết rút gọn:

$$\mathbf{T}_{ij} = \begin{bmatrix} R_{ij} & P \\ 0 & 1 \end{bmatrix}$$

(1.20)

Trong đó:

$\mathbf{R}_{ij}$: Ma trận quay 3x3.

P: Ma trận 3x1 biểu thị tọa độ của điểm gốc hệ tọa độ O_j trong hệ tọa độ $O_i\ x_i\ y_i\ z_i$.

Ma trận thuần nhất $\mathbf{T}_{4x4}$ hoàn toàn xác định vị trí (ma trận P) và hướng (ma trận **R**) của hệ tọa độ $O_j\ x_j\ y_j\ z_j$ sang hệ tọa độ $O_i\ x_i\ y_i\ z_i$.

d) Các phép biến đổi cơ bản

- Phép biến đổi tịnh tiến: ta có φ =0, do đó:

$$\mathbf{T} = \begin{bmatrix} 1 & 0 & 0 & p_x \\ 0 & 1 & 0 & p_y \\ 0 & 0 & 1 & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

(1.21)

Tịnh tiến a đơn vị dọc theo trục x, b đơn vị dọc theo trục y, c đơn vị dọc theo trục z, khi đó:

$$\mathbf{T} = \begin{bmatrix} 1 & 0 & 0 & a \\ 0 & 1 & 0 & b \\ 0 & 0 & 1 & c \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

(1.22)

Phép quay quanh các trục tọa độ

- Quay quanh trục x góc θ

$$\mathbf{R}(x,\theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

(1.23)

- Quay quanh trục y góc α

$$\mathbf{R}(y,\alpha) = \begin{bmatrix} \cos\alpha & 0 & \sin\alpha & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\alpha & 0 & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

(1.24)

- Quay quanh trục z góc φ

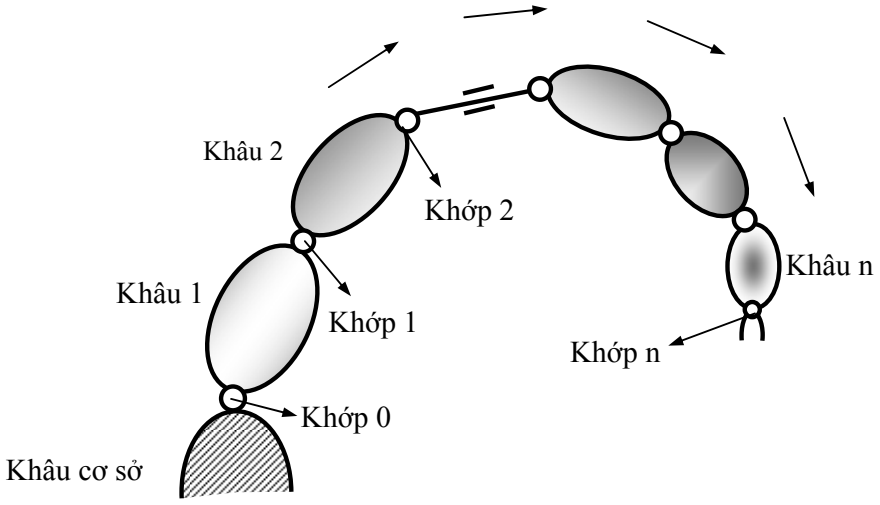
$$\mathbf{R}(z,\varphi)=\begin{bmatrix}\cos\varphi & -\sin\varphi & 0 & 0\\ \sin\varphi & \cos\varphi & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}$$

(1.25)

1.3.2 Ma trận Denavit-Hartenberg

Xét mô hình rôbốt gồm có *n* khâu như hình 1.10. Các khâu được đánh số tăng dần từ khâu cơ sở (khâu 0) cho đến khâu thứ *n*. Khớp thứ *k* nối giữa khâu *k-1* và khâu *k*. Hai loại khớp thường được dùng trong thiết kế rôbốt là khớp quay và khớp tịnh tiến. Mỗi khớp chỉ có một bậc tự do.

Để mô tả mối quan hệ về mặt động học của hai khâu liên tiếp, người ta thường sử dụng các quy ước do *Denavit-Hartenberg* (DH) đề xuất năm 1955. Theo DH, tại mỗi khớp ta gắn một hệ trục toạ độ, quy ước về cách đặt hệ toạ độ này như sau:



Hình 1.10 Robot n khâu

- Trục z_i được liên kết với trục của khớp thứ i+1. Chiều của z_i được chọn tuỳ ý.

- Trục x_i được xác định là đường vuông góc chung giữa trục khớp *i* và khớp *i+1*, hướng từ điểm trục của khớp i tới khớp i+1. Nếu hai trục song song, thì x_i có thể chọn bất kỳ là đường vuông góc chung hai trục khớp. Trong trường hợp hai trục này cắt nhau, x_i được xác định theo chiều của $z_i \times z_{i+1}$ (hoặc quy tắc bàn tay phải).

- Trục y_i được xác định theo x_i và z_i theo quy tắc bàn tay phải.

Bốn thông số DH liên hệ giữa phép biến đổi của hai hệ trục toạ độ liên tiếp được xác định như sau:

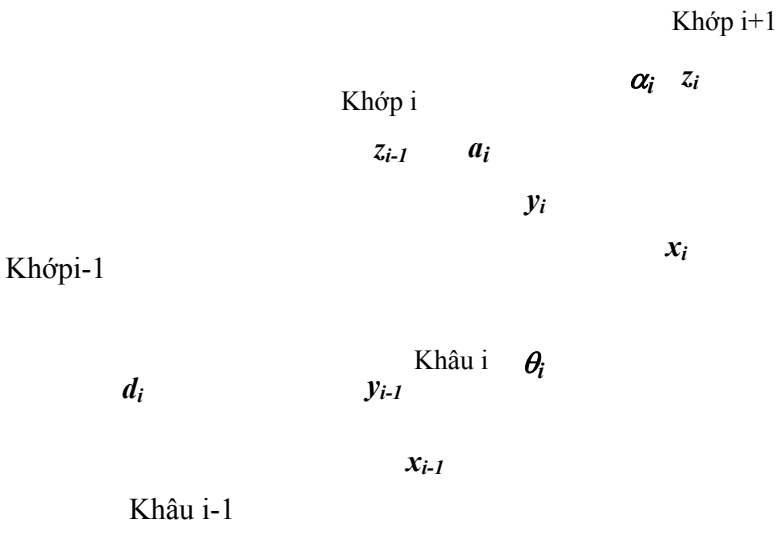
θ_i: Góc xoay đưa trục x_{i-1} về x_i quanh z_{i-1} theo quy tắc bàn tay phải.

d_i: Dịch chuyển dọc trục z_{i-1} đưa gốc toạ độ về nằm trên trục z_i.

α_i: Góc xoay đưa trục z_{i-1} về z_i quanh x_i theo quy tắc bàn tay phải.

a_i: Dịch chuyển dọc trục x_i, đưa gốc toạ độ về nằm trên trục x_i.

Ma trận của phép biến đổi, ký hiệu là $\mathbf{H}_i$, là tích của bốn ma trận biến đổi cơ bản và có dạng như sau



Hình 1.11 Hai khâu liên tiếp

$$ {}^{i-1}\mathbf{H}_i = \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 & 0 \\ \sin\theta & \cos\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha_i & -\sin\alpha_i & 0 \\ 0 & \sin\alpha_i & \cos\alpha_i & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

(1.26)

hay dạng thu gọn:

$$ {}^{i-1}\mathbf{H}_i = \mathbf{H}_i = \begin{bmatrix} \cos\theta_i & -\sin\theta_i\cos\alpha_i & \sin\theta_i\sin\alpha_i & a_i\cos\theta_i \\ \sin\theta_i & \cos\theta_i\cos\alpha_i & -\cos\theta_i\sin\alpha_i & a_i\sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

(1.27)

Do mỗi khớp chỉ có một bậc tự do nên trong bốn thông số trên chỉ có duy nhất một thông số đóng vai trò là ẩn.

Nếu khớp là khớp tịnh tiến thì d_i sẽ là ẩn.

Nếu khớp là khớp quay thì θ_i sẽ là ẩn.

Một cách hình thức có thể biểu diễn ma trận thuần nhất như sau:

$$ \mathbf{H}_i = \begin{bmatrix} {}^{i-1}\mathbf{A}_i & {}^{i-1}\mathbf{p}_i \\ \mathbf{0} & 1 \end{bmatrix} $$

(1.28)

Trong đó

${}^{i-1}\mathbf{A}_i$ (3x3): Ma trận côsin chỉ hướng đưa hệ toạ độ i về $i-1$

${}^{i-1}\mathbf{p}_i$ (3x1): Vị trí gốc toạ độ của hệ toạ độ i đặt trong hệ $i-1$

Nếu thực hiện phép biến đổi liên tiếp, quan hệ giữa hệ toạ độ i so với khâu cơ sở (hệ toạ độ 0) được xác định bởi:

$$\mathbf{T}_i = \mathbf{H}_1\mathbf{H}_2...\mathbf{H}_i = \begin{bmatrix} {}^0\mathbf{A}_i & {}^0\mathbf{p}_i \\ \mathbf{0} & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{A}_i & \mathbf{p}_i \\ \mathbf{0} & 1 \end{bmatrix}$$

(1.29)

Trong đó:

$\mathbf{A}_i$ (3x3): Ma trận côsin chỉ hướng đưa hệ của hệ toạ độ i về hệ toạ độ 0.

$\mathbf{p}_i$ (3x1): Vị trí gốc toạ độ của hệ toạ độ i so với khâu cơ sở.

Phép biến đổi ngược từ hệ toạ độ cơ sở về hệ toạ độ *i* chính là ma trận nghịch đảo của ma trận thuần nhất.

Nếu ký hiệu ma trận nghịch đảo dạng khối:

$$\left(\mathbf{T}_i\right)^{-1} = \begin{bmatrix} \mathbf{B}_i & \mathbf{b}_i \\ \mathbf{0} & 1 \end{bmatrix}$$

(1.30)

ta có

$$\mathbf{T}_i\left(\mathbf{T}_i\right)^{-1} = \begin{bmatrix} \mathbf{A}_i & \mathbf{p}_i \\ \mathbf{0} & 1 \end{bmatrix}\begin{bmatrix} \mathbf{B}_i & \mathbf{b}_i \\ \mathbf{0} & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{E} & \mathbf{0} \\ \mathbf{0} & 1 \end{bmatrix}$$

(1.31)

hay

$$\begin{bmatrix} \mathbf{A}_i\mathbf{B}_i & \mathbf{A}_i\mathbf{b}_i + \mathbf{p}_i \\ \mathbf{0} & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{E} & \mathbf{0} \\ \mathbf{0} & 1 \end{bmatrix}$$

(1.32)

Đồng nhất từng phần tử ma trận khối của (1.31) ta được:

$$\mathbf{A}_i\mathbf{B}_i = \mathbf{E} \Rightarrow \mathbf{B}_i = \left(\mathbf{A}_i\right)^{-1} = \mathbf{A}_i^T$$

(1.33)

$$\mathbf{A}_i\mathbf{b}_i + \mathbf{p}_i = \mathbf{0} \Rightarrow \mathbf{b}_i = -\left(\mathbf{A}_i\right)^{-1}\mathbf{p}_i = -\mathbf{A}_i^T\mathbf{p}_i$$

(1.34)

Vậy:

$$\left(\mathbf{T}_i\right)^{-1} = \begin{bmatrix} \mathbf{A}_i^T & -\mathbf{A}_i^T\mathbf{p}_i \\ \mathbf{0} & 1 \end{bmatrix}$$

(1.35)

Với việc sử dụng ma trận biến đổi thuần nhất 4x4, việc xác định vị trí và hướng của một khâu bất kỳ của rôbốt là hoàn toàn xác định.

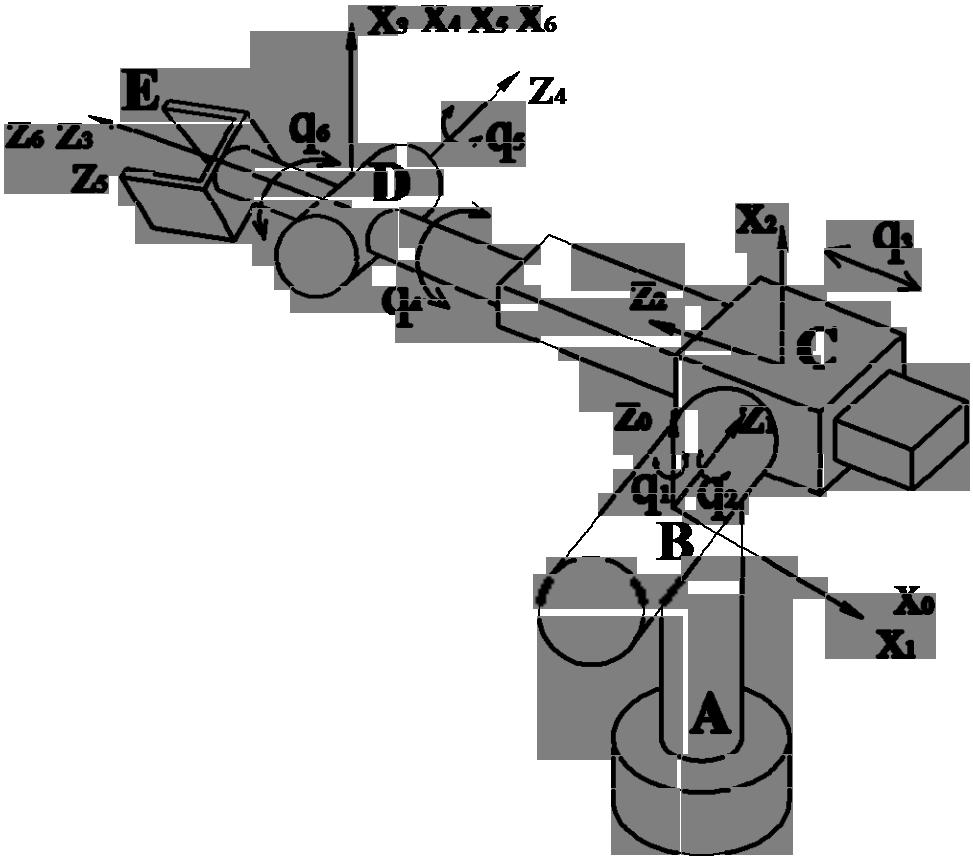
1.4 Chuỗi động học robot

Giả sử khảo sát chuỗi động học của robot *STANFORD* như vẽ (hình 1.12) .

Các hệ tọa độ chọn theo quy tắc *Denavit-Hartenberg*.

Bảng thông số động học DH của robot *STANFORD* như sau:

Khâu	θ	d_i	a_i	α
1	θ_1^*	0	0	-90^0
2	θ_2^*	d_2	0	90^0
3	0	d_3^*	0	0
4	θ_4^*	0	0	-90^0
5	θ_5^*	0	0	90^0
6	θ_6^*	0	0	0



Hình 1.12 Robot *STANFORD*

Các ma trận **H** của robot *STANFORD* được xác định theo công thức (1.27)

$$ {}^{i-1}\mathbf{H}_i = \mathbf{H}_i = \begin{bmatrix} \cos\theta_i & -\sin\theta_i\cos\alpha_i & \sin\theta_i\sin\alpha_i & a_i\cos\theta_i \\ \sin\theta_i & \cos\theta_i\cos\alpha_i & -\cos\theta_i\sin\alpha_i & a_i\sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

- Ma trận mô tả vị trí và hướng của $Bx_1y_1z_1$ đối với $Bx_0y_0z_0$: ${}^0\mathbf{H}_1$

$$ {}^{0}\mathbf{H}_1=\begin{bmatrix} \cos q_1 & 0 & -\sin q_1 & 0 \\ \sin q_1 & 0 & \cos q_1 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

(1.36)

- Ma trận mô tả vị trí và hướng của $Cx_2y_2z_2$ đối với $Bx_1y_1z_1$: ${}^{1}\mathbf{H}_2$

$$ {}^{1}\mathbf{H}_2=\begin{bmatrix} \cos q_2 & \sin q_2 & 0 & 0 \\ \sin q_2 & -\cos q_2 & 0 & 0 \\ 0 & 1 & 0 & d_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

(1.37)

- Ma trận mô tả vị trí và hướng của $Dx_3y_3z_3$ đối với $Cx_2y_2z_2$: ${}^{2}\mathbf{H}_3$

$$ {}^{2}\mathbf{H}_3=\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & q_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

(1.38)

- Ma trận mô tả vị trí và hướng của $Dx_4y_4z_4$ đối với $Dx_3y_3z_3$: ${}^{3}\mathbf{H}_4$

$$ {}^{3}\mathbf{H}_4=\begin{bmatrix} \cos q_4 & 0 & -\sin q_4 & 0 \\ \sin q_4 & 0 & \cos q_4 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

(1.39)

- Ma trận mô tả vị trí và hướng của $Dx_5y_5z_5$ đối với $Dx_4y_4z_4$: ${}^{4}\mathbf{H}_5$

$$ {}^{3}\mathbf{H}_4 = \begin{bmatrix} \cos q_5 & 0 & \sin q_5 & 0 \\ \sin q_5 & 0 & -\cos q_5 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

(1.40)

- Ma trận mô tả vị trí và hướng của $Dx_6y_6z_6$ đối với $Dx_5y_5z_5$: ${}^{5}\mathbf{H}_6$

$$ {}^{5}\mathbf{H}_6 = \begin{bmatrix} \cos q_6 & \sin q_6 & 0 & 0 \\ \sin q_6 & -\cos q_6 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

(1.41)

Từ các ma trận *Denavit-Hartenberg* ta tính được vị trí, hướng của khâu thao tác đối với hệ tọa độ cố định $O_0x_0y_0z_0$ là ma trận $\mathbf{T}_6$.

$$ \mathbf{T}_6 = {}^{0}\mathbf{H}_1 \, {}^{1}\mathbf{H}_2 \, {}^{2}\mathbf{H}_3 \, {}^{3}\mathbf{H}_4 \, {}^{4}\mathbf{H}_5 \, {}^{5}\mathbf{H}_6 $$

(1.42)

Các giá trị ${}^{0}\mathbf{H}_1$, ${}^{1}\mathbf{H}_2$, ${}^{2}\mathbf{H}_3$, ${}^{3}\mathbf{H}_4$, ${}^{4}\mathbf{H}_5$, ${}^{5}\mathbf{H}_6$ được xác định từ công thức (1.36), (1.37),…, (1.41).

Ma trận $\mathbf{T}_6$ cho ta biết hướng và vị trí của khâu thao tác trong hệ tọa độ cố định hay nói cách khác là vị trí của điểm tác động cuối và hướng của hệ tọa độ động gắn vào khâu tại điểm tác động cuối trong hệ tọa độ cố định.

Mặt khác nếu ta gọi [xp yp zp rotxp rotyp rotzp] là vector mô tả trực tiếp vị trí và hướng của $O_6x_6y_6z_6$ trong hệ tọa độ $O_0x_0y_0z_0$. Trong đó [xp yp zp] là tọa độ và [rotxp rotyp rotzp] là các góc quay *Cardan* của $O_6x_6y_6z_6$ đối với $O_0x_0y_0z_0$. Khi đó ta có:

$$ {}^{0}\mathbf{A}_{f} = \begin{bmatrix} {}^{0}\mathbf{C}_{d} & {}^{0}\mathbf{r}_{f} \\ 0 & 1 \end{bmatrix} $$

(1.43)

${}^{0}\mathbf{C}_{d}$: Là ma trận *Cardan* mô tả hướng của $O_6x_6y_6z_6$ đối với $O_0x_0y_0z_0$

${}^{0}\mathbf{r}_{f}$: Vector vị trí của $O_6x_6y_6z_6$ đối với $O_0x_0y_0z_0$

$$ \mathbf{C}_{d} = \begin{bmatrix} C\psi.C\theta & -C\psi.S\theta & S\psi \\ S\varphi.S\psi.C\theta + C\varphi.S\theta & -S\varphi.S\psi.S\theta + C\varphi.C\theta & -S\varphi.C\psi \\ -C\varphi.S\psi.C\theta + S\varphi.S\theta & C\varphi.S\psi.S\theta + S\varphi.C\theta & C\varphi.C\psi \end{bmatrix} $$

(1.44)

Trong đó ký hiệu $C = \cos$, $S = \sin$, $\varphi = r$otxp, ψ=rotyp, θ=rotzp .

Ma trận $\mathbf{T}_6$ là ma trận mô tả vị trí và hướng của khâu thao tác trong hệ tọa độ cố định thông qua các biến khớp q_i. Còn ma trận ${}^{0}\mathbf{A}_{f}$ cũng mô tả vị trí và hướng của khâu thao tác trong hệ tọa độ cố định nhưng trực tiếp qua các góc quay *Cardan* và tọa độ khâu thao tác. Từ đây suy ra:

$$ \mathbf{T}_6 = {}^{0}\mathbf{A}_{f} $$

(1.45)

Từ phương trình (1.35) suy ra hệ 6 phương trình độc lập:

$$ \begin{cases} f_1 = \mathbf{T}_6[1,4] - {}^{0}\mathbf{A}_{\mathrm{f}}[1,4] \\ f_2 = \mathbf{T}_6[2,4] - {}^{0}\mathbf{A}_{\mathrm{f}}[2,4] \\ f_3 = \mathbf{T}_6[3,4] - {}^{0}\mathbf{A}_{\mathrm{f}}[3,4] \\ f_4 = \mathbf{T}_6[1,2] - {}^{0}\mathbf{A}_{\mathrm{f}}[1,2] \\ f_5 = \mathbf{T}_6[2,3] - {}^{0}\mathbf{A}_{\mathrm{f}}[2,3] \\ f_6 = \mathbf{T}_6[3,1] - {}^{0}\mathbf{A}_{\mathrm{f}}[3,1] \end{cases} $$

(1.46)

Viết lại hệ phương trình (1.36) dạng:

$$ \mathbf{f}(\mathbf{x}) = 0 $$

(1.47)

Trong đó:

$$\mathbf{f} = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_6 \end{bmatrix}$$

(1.48)

$$\mathbf{x} = [q_1, q_2, q_3, q_4, q_5, q_6 xp, yp, zp, rotxp, rotyp, rotzp]$$

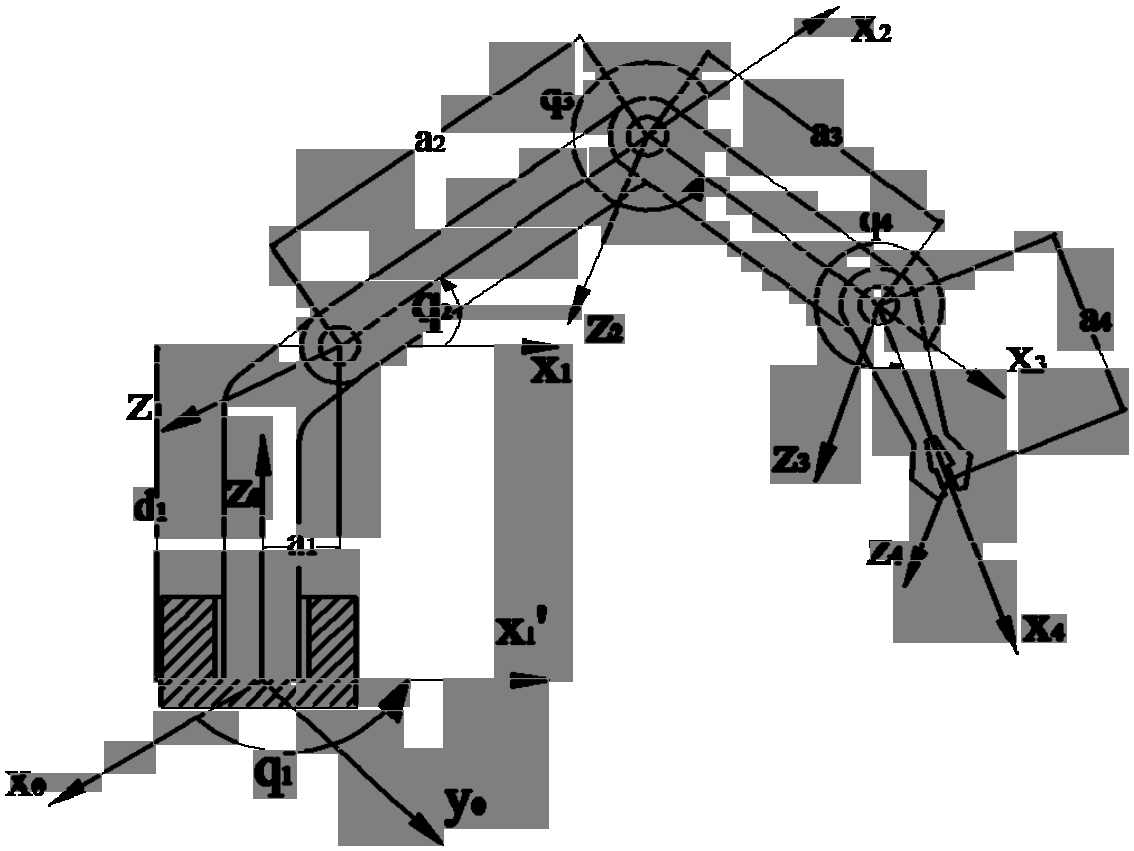
(1.49)

Nếu các tham số biết trước là $xp, yp, zp, rotxp, rotyp, rotzp$ các tham số cần xác định là: $\begin{bmatrix} q_1 & q_2 & q_3 & q_4 & q_5 & q_6 \end{bmatrix}$ và ngược lại.

CHƯƠNG 2
BÀI TOÁN ĐỘNG HỌC ROBOT MMR

2.1 Hệ phương trình động học cơ bản của MMR

Khảo sát chuyển động của robot **MMR** khi xe dừng lại và cánh tay thực hiện thao tác công việc. Chọn hệ tọa độ theo quy tắc của *Denavit-Hartenberg* như hình 2.1.



hình 2.1

- Chọn hệ tọa độ $O_0x_0y_0z_0$ gắn tại khâu 0 và đặt tại khâu 1. Trục z_0 trùng với trục quay của khâu 1, x_0, y_0 được chọn sao cho $O_0x_0y_0z_0$ là hệ quy chiếu thuận.
- Chọn hệ tọa độ $O_1x_1y_1z_1$ gắn tại khâu 1 và đặt tại khâu 2. Trục z_1 trùng với trục quay của khâu 2, x_1 được chọn sao cho là được vuông góc chung của z_0 và z_1, y_1 chọn sao cho $O_1x_1y_1z_1$ là hệ quy chiếu thuận.
- Chọn hệ tọa độ $O_2x_2y_2z_2$ gắn tại khâu 2 và đặt tại khâu 3. Trục z_2 trùng với trục quay của khâu 3, x_2 được chọn sao cho là được vuông góc chung của z_1 và z_2, y_2 chọn sao cho $O_2x_2y_2z_2$ là hệ quy chiếu thuận.

- Chọn hệ tọa độ $O_3x_3y_3z_3$ gắn tại khâu 3 và đặt tại khâu 4. Trục z_3 trùng với trục quay của khâu 3, x_3 được chọn sao cho là được vuông góc chung của z_2 và z_3, y_3 chọn sao cho $O_3x_3y_3z_3$ là hệ quy chiếu thuận.
- Chọn hệ tọa độ $O_4x_4y_4z_4$ đặt ở vị trí thao tác, trục z_4 trùng với trục của khâu 4, x_4 là đường vuông góc chung của z_3 và z_4, y_4 chọn sao cho $O_4x_4y_4z_4$ là hệ quy chiếu thuận.

Từ hệ tọa độ đã chọn ta có bảng động học *Denavit-Hartenberg* như sau:

Khâu	θ_i	d_i	a_i	α_i
1	q_1	d_1	a_1	$\pi/2$
2	q_2	0	a_2	0
3	q_3	0	a_3	0
4	q_4	0	a_4	0

Trong đó:

$$d_1= 90,\ a_1= 45,\ a_2= 283,\ a_3= 263,\ a_4= 130\ (mm)$$

Từ cơ sở lý thuyết đã nêu ở chương 1 ta xác định các ma trận *Denavit-Hartenberg* như sau:

- Ma trận mô tả vị trí và hướng của $O_1x_1y_1z_1$ đối với $O_0x_0y_0z_0$: ${}^0\mathbf{H}_1$

$$ {}^0\mathbf{H}_1=\begin{bmatrix} \cos q_1 & -\sin q_1\cos\alpha_1 & \sin q_1\sin\alpha_1 & a_1\cos q_1 \\ \sin q_1 & \cos q_1\cos\alpha_1 & -\cos q_1\sin\alpha_1 & a_1\sin q_1 \\ 0 & \sin\alpha_1 & \cos\alpha_1 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$ {}^{0}\mathbf{H}_1=\begin{bmatrix} \cos q_1 & 0 & \sin q_1 & a_1\cos q_1 \\ \sin q_1 & 0 & -\cos q_1 & a_1\sin q_1 \\ 0 & 1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

(2.1)

- Ma trận mô tả vị trí và hướng của $O_2x_2y_2z_2$ đối với $O_1x_1y_1z_1$: ${}^{1}\mathbf{H}_2$

$$ {}^{1}\mathbf{H}_2=\begin{bmatrix} \cos q_2 & -\sin q_2\cos\alpha_2 & \sin q_2\sin\alpha_2 & a_2\cos q_2 \\ \sin q_2 & \cos q_2\cos\alpha_2 & -\cos q_2\sin\alpha_2 & a_2\sin q_2 \\ 0 & \sin\alpha_2 & \cos\alpha_2 & d_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^{1}\mathbf{H}_2=\begin{bmatrix} \cos q_2 & -\sin q_2 & 0 & a_2\cos q_2 \\ \sin q_2 & \cos q_2 & 0 & a_2\sin q_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

(2.2)

- Ma trận mô tả vị trí và hướng của $O_3x_3y_3z_3$ đối với $O_2x_2y_2z_2$: ${}^{2}\mathbf{H}_3$

$$ {}^{2}\mathbf{H}_3=\begin{bmatrix} \cos q_3 & -\sin q_3\cos\alpha_3 & \sin q_3\sin\alpha_3 & a_3\cos q_3 \\ \sin q_3 & \cos q_3\cos\alpha_3 & -\cos q_3\sin\alpha_3 & a_3\sin q_3 \\ 0 & \sin\alpha_3 & \cos\alpha_3 & d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^{2}\mathbf{H}_3=\begin{bmatrix} \cos q_3 & -\sin q_3 & 0 & a_3\cos q_3 \\ \sin q_3 & \cos q_3 & 0 & a_3\sin q_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

(2.3)

♦ Ma trận mô tả vị trí và hướng của $O_4x_4y_4z_4$ đối với $O_3x_3y_3z_3$:

$^3\mathbf{H}_4$ $^3\mathbf{H}_4 =$

$$\begin{bmatrix} \cos q_4 & -\sin q_4 \cos\alpha_4 & \sin q_4 \sin\alpha_4 & a_4 \cos q_4 \\ \sin q_4 & \cos q_4 \cos\alpha_4 & -\cos q_4 \sin\alpha_4 & a_4 \sin q_4 \\ 0 & \sin\alpha_4 & \cos\alpha_4 & d_4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$^3\mathbf{H}_4 = \begin{bmatrix} \cos q_4 & -\sin q_4 & 0 & a_4 \cos q_4 \\ \sin q_4 & \cos q_4 & 0 & a_4 \sin q_4 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

(2.4)

Từ các ma trận $^0\mathbf{H}_1$, $^1\mathbf{H}_2$, $^2\mathbf{H}_3$, $^3\mathbf{H}_4$ được xác định theo công thức (2.1), (2.2), (2.3), (2.4) ta tính được ma trận mô tả vị trí và hướng của khâu thao tác trong hệ tọa độ cố định $O_0x_0y_0z_0$ theo công thức:

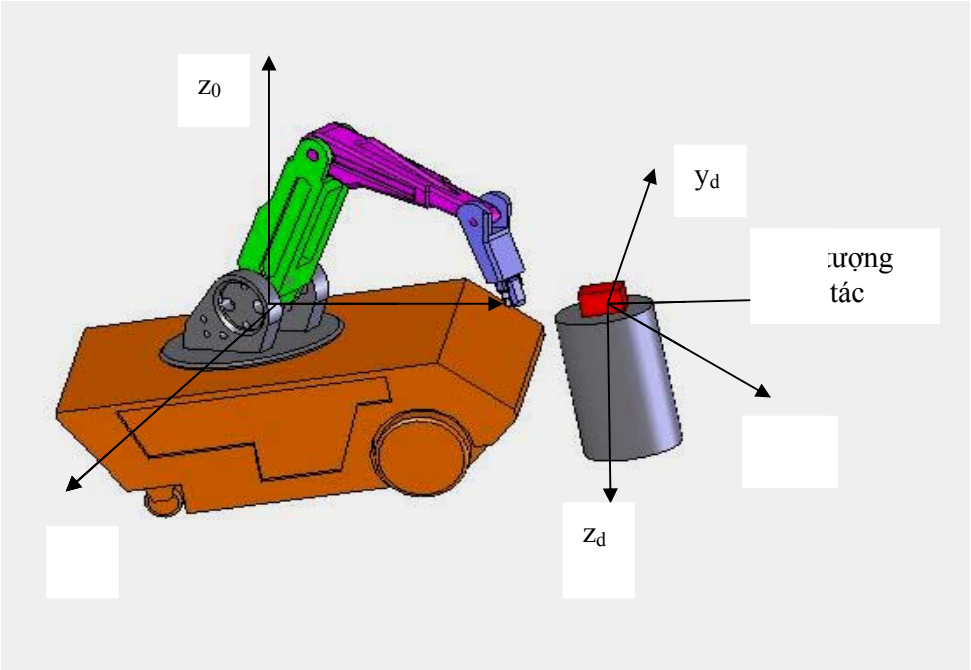
$$\mathbf{T}_4 = {}^0\mathbf{H}_1 . {}^1\mathbf{H}_2 . {}^2\mathbf{H}_3 . {}^3\mathbf{H}_4$$

(2.5)

Giả sử robot cần thực hiện thao tác đối với đối tượng như hình vẽ (hình 2.2). Ta sử dụng hệ tọa độ $O_d x_d y_d z_d$ gắn vào đối tượng. Khi đó ma trận mô tả vị trí và hướng của $O_d x_d y_d z_d$ trong hệ tọa độ cố định $O_0 x_0 y_0 z_0$ là ma trận: $^0\mathbf{A}_d$

Ma trận mô tả vị trí và hướng của khâu thao tác trên đối tượng đối với hệ tọa độ $O_d x_d y_d z_d$ là ma trận: $^d\mathbf{A}_f$.

Vậy ta có $^0\mathbf{A}_f = {}^0\mathbf{A}_d . {}^d\mathbf{A}_f$ chính là ma trận mô vị trí và hướng của khâu thao tác trên vật đối với hệ tọa độ cố định.



Hình 2.2

Theo cơ sở lý thuyết đã trình bày ở chương 1 ta có :

$$\mathbf{T}_4 = {}^0\mathbf{A}_f$$

(2.6)

Trong đó $\mathbf{T}_4 = \begin{bmatrix} \mathbf{A}_4 & \mathbf{p}_4 \\ 0 & 1 \end{bmatrix}$ và ${}^0\mathbf{A}_f = \begin{bmatrix} \mathbf{C}_d & \mathbf{r}_f \\ 0 & 1 \end{bmatrix}$

Từ đây ta rút ra 6 phương trình gồm 10 tham số:

$$f_\alpha(\mathbf{x}) = 0$$

(2.7)

$$\mathbf{x} = [q_1 \;\; q_2 \;\; q_3 \;\; q_4 \;\; xp \;\; yp \;\; zp \;\; rotxp \;\; rotyp \;\; rotzp]$$

(2.8)

Vì robot có 4 bậc tự do ta chỉ thực hiện điều khiển chuyển động của robot với 4 tham số ở đây cho quy luật của điểm tác động:

$xp = xp(t), yp = yp(t), zp = zp(t)$ và một 1 tham số xác định hướng cua khâu thao tác, có thể cho trước $rotyp = 0$. Từ đó giải 6 phương trình 6 ẩn số.

Các phương trình động học của robot **MMR** như sau:

$$f_1 := 0.13\ (\cos(q_1)\cos(q_2)\cos(q_3) - \cos(q_1)\sin(q_2)\sin(q_3))\cos(q_4)$$
$$+ 0.13\ (-\cos(q_1)\cos(q_2)\sin(q_3) - \cos(q_1)\sin(q_2)\cos(q_3))\sin(q_4) + 0.263\cos(q_1)\cos(q_2)\cos(q_3)$$
$$- 0.263\cos(q_1)\sin(q_2)\sin(q_3) + 0.283\cos(q_1)\cos(q_2) + 0.045\cos(q_1) - xp$$

$$f_2 := 0.13\ (\sin(q_1)\cos(q_2)\cos(q_3) - \sin(q_1)\sin(q_2)\sin(q_3))\cos(q_4)$$
$$+ 0.13\ (-\sin(q_1)\cos(q_2)\sin(q_3) - \sin(q_1)\sin(q_2)\cos(q_3))\sin(q_4) + 0.263\sin(q_1)\cos(q_2)\cos(q_3)$$
$$- 0.263\sin(q_1)\sin(q_2)\sin(q_3) + 0.283\sin(q_1)\cos(q_2) + 0.045\sin(q_1) - yp$$

$$f_3 := 0.13\ (\sin(q_2)\cos(q_3) + \cos(q_2)\sin(q_3))\cos(q_4) + 0.13\ (-\sin(q_2)\sin(q_3) + \cos(q_2)\cos(q_3))\sin(q_4) + 0.09$$
$$+ 0.263\sin(q_2)\cos(q_3) + 0.263\cos(q_2)\sin(q_3) + 0.283\sin(q_2) - zp$$

$$f_4 := -(\cos(q_1)\cos(q_2)\cos(q_3) - \cos(q_1)\sin(q_2)\sin(q_3))\sin(q_4)$$
$$+ (-\cos(q_1)\cos(q_2)\sin(q_3) - \cos(q_1)\sin(q_2)\cos(q_3))\cos(q_4) + \cos(rotyp)\sin(rotzp)$$

$$f_5 := -\cos(q_1) + \sin(rotxp)\cos(rotyp)$$

$$f_6 := (\sin(q_2)\cos(q_3) + \cos(q_2)\sin(q_3))\cos(q_4) + (-\sin(q_2)\sin(q_3) + \cos(q_2)\cos(q_3))\sin(q_4)$$
$$+ \cos(rotxp)\sin(rotyp)\cos(rotzp) - \sin(rotxp)\sin(rotzp)$$

(2.9)

2.2 Bài toán vị trí

2.2.1 Bài toán thuận.

- Biết trước giá trị của biến khớp (q_1,q_2,q_3,q_4)
- Yêu cầu tìm các toạ độ của khâu cuối (xp, yp, zp, rotxp, rotyp, rotzp).

Vị trí của điểm tác động cuối lên đối tượng cần thao tác được xác định bởi toạ độ điểm P(xp, yp, zp), hướng của nó được xác định bởi các góc quay (rotxp, rotyp,rotzp).

Theo hệ phương trình (2.6):

Ba phương trình đầu xác định được vị trí của đối tượng:

$$\begin{cases} xp = {}^{0}\mathbf{A}_{f}[1,4]=\mathbf{T}_{4}[1,4] \\ yp = {}^{0}\mathbf{A}_{f}[2,4]=\mathbf{T}_{4}[2,4] \\ zp = {}^{0}\mathbf{A}_{f}[3,4]=\mathbf{T}_{4}[3,4] \end{cases}$$

(2.10)

Ba phương trình sau cho ta bài toán xác định hướng của điểm tác động cuối lên đối tượng:

$$\begin{cases} f_4 = \mathbf{T}_{4}[1,2]\text{-}{}^{0}\mathbf{A}_{\mathrm{f}}[1,2] \\ f_5 = \mathbf{T}_{4}[2,3]\text{-}{}^{0}\mathbf{A}_{\mathrm{f}}[2,3] \\ f_6 = \mathbf{T}_{4}[3,1]\text{-}{}^{0}\mathbf{A}_{\mathrm{f}}[3,1] \end{cases}$$

(2.11)

Ba phương trình f_4, f_5, f_6 đã được tính ở phần trên theo (2.9):

$$f_4 := -(\cos(q_1)\cos(q_2)\cos(q_3) - \cos(q_1)\sin(q_2)\sin(q_3))\sin(q_4) + (-\cos(q_1)\cos(q_2)\sin(q_3) - \cos(q_1)\sin(q_2)\cos(q_3))\cos(q_4) + \cos(rotyp)\sin(rotzp)$$

$$f_5 := -\cos(q_1) + \sin(rotxp)\cos(rotyp)$$

$$f_6 := (\sin(q_2)\cos(q_3) + \cos(q_2)\sin(q_3))\cos(q_4) + (-\sin(q_2)\sin(q_3) + \cos(q_2)\cos(q_3))\sin(q_4) + \cos(rotxp)\sin(rotyp)\cos(rotzp) - \sin(rotxp)\sin(rotzp)$$

Giải các phương trình trên ta sẽ tính được hướng của hệ tọa độ khâu thao tác đối với hệ tọa độ cố định.

2.2.2 Bài toán ngược

Bài toán ngược là bài toán có ý nghĩa rất quan trọng trong thực tế. Khi biết quy luật chuyển động của khâu thao tác và ta phải tìm các giá trị của biến khớp. Việc xác định các giá trị của biến khớp cho phép ta điều khiển robot theo đúng quỹ đạo đã cho.

Từ trên theo (2.7) ta đã có 6 phương trình với 10 tham số:

$$f_{\alpha}(\mathbf{x}) = 0$$

$$\mathbf{x} = [q_1 \;\; q_2 \;\; q_3 \;\; q_4 \;\; xp \;\; yp \;\; zp \;\; rotxp \;\; rotyp \;\; rotzp]$$

Vì vậy ta phải biết trước 4 tham số hay còn gọi là biến điều khiển. Với mô hình robot **MMR** này ta cho biết trước [xp yp zp rotyp]. Các phương trình trên đều là các phương trình đại số phi tuyến do đó để giải các phương trình này ta dùng phương pháp lặp *Newton-Raphson* .

2.3 Bài toán vận tốc

2.3.1 Bài toán thuận

Ta có thể viết lại phương trình (2.7) ở dạng sau:

$$\mathbf{f(p,q)} = \mathbf{0}$$

(2.12)

Trong đó:

p: là vector chứa thông số của điểm tác động cuối:

p = [xp yp zp rotpx rotyp rotzp]

q : Là véctơ có các thành phần là các tọa độ điều khiển:

q = [q_1 q_2 q_3 q_4]

Đạo hàm hai vế của phương trình (2.12) theo thời gian ta được:

$$\sum_{i=1}^{6} \frac{\partial f_\alpha}{\partial p_i} \dot{p}_i = -\sum_{k=1}^{4} \frac{\partial f_\alpha}{\partial q_k} \dot{q}_k \qquad \alpha = 1..6$$

(2.13)

Có thể viết:

$$\begin{bmatrix} \frac{\partial f_1}{\partial p_1} & \frac{\partial f_1}{\partial p_2} & \cdots & \frac{\partial f_1}{\partial p_6} \\ \frac{\partial f_2}{\partial p_1} & \frac{\partial f_2}{\partial p_2} & \cdots & \frac{\partial f_2}{\partial p_6} \\ \vdots & \vdots & \cdots & \vdots \\ \frac{\partial f_6}{\partial p_1} & \frac{\partial f_6}{\partial p_2} & \cdots & \frac{\partial f_6}{\partial p_6} \end{bmatrix} \begin{bmatrix} \dot{p}_1 \\ \dot{p}_2 \\ \vdots \\ \dot{p}_6 \end{bmatrix} = \begin{bmatrix} -\frac{\partial f_1}{\partial q_1} & -\frac{\partial f_1}{\partial q_2} & -\frac{\partial f_1}{\partial q_3} & -\frac{\partial f_1}{\partial q_4} \\ -\frac{\partial f_2}{\partial q_1} & -\frac{\partial f_2}{\partial q_2} & -\frac{\partial f_2}{\partial q_3} & -\frac{\partial f_2}{\partial q_4} \\ \vdots & \vdots & \vdots & \vdots \\ -\frac{\partial f_6}{\partial q_1} & -\frac{\partial f_6}{\partial q_2} & -\frac{\partial f_6}{\partial q_3} & -\frac{\partial f_6}{\partial q_4} \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \\ \dot{q}_4 \end{bmatrix}$$

Đặt

$$\mathbf{J}_p = \begin{bmatrix} \frac{\partial f_1}{\partial p_1} & \frac{\partial f_1}{\partial p_2} & \cdots & \frac{\partial f_1}{\partial p_6} \\ \frac{\partial f_2}{\partial p_1} & \frac{\partial f_2}{\partial p_2} & \cdots & \frac{\partial f_2}{\partial p_6} \\ \vdots & \vdots & \cdots & \vdots \\ \frac{\partial f_6}{\partial p_1} & \frac{\partial f_6}{\partial p_2} & \cdots & \frac{\partial f_6}{\partial p_6} \end{bmatrix}$$

(2.14)

$$\mathbf{J}_q = \begin{bmatrix} -\frac{\partial f_1}{\partial q_1} & -\frac{\partial f_1}{\partial q_2} & -\frac{\partial f_1}{\partial q_3} & -\frac{\partial f_1}{\partial q_4} \\ -\frac{\partial f_2}{\partial q_1} & -\frac{\partial f_2}{\partial q_2} & -\frac{\partial f_2}{\partial q_3} & -\frac{\partial f_2}{\partial q_4} \\ \vdots & \vdots & \vdots & \vdots \\ -\frac{\partial f_6}{\partial q_1} & -\frac{\partial f_6}{\partial q_2} & -\frac{\partial f_6}{\partial q_3} & -\frac{\partial f_6}{\partial q_4} \end{bmatrix}$$

(2.15)

Thế vào được phương trình:

$$\mathbf{J}_p.\dot{\mathbf{p}} = \mathbf{J}_q.\dot{\mathbf{q}}$$

(2.16)

Hay

$$\dot{\mathbf{p}} = \mathbf{J}_p^{-1}.\mathbf{J}_q.\dot{\mathbf{q}} = \mathbf{J}_f.\dot{\mathbf{q}}$$

(2.17)

Trong đó

$$\mathbf{J}_p^{-1}.\mathbf{J}_q = \mathbf{J}_f$$

(2.18)

2.2.2 Bài toán ngược

Ta có thể viết (2.7) dưới dạng sau:

$$\mathbf{f}(\mathbf{p}^*, \mathbf{s}) = \mathbf{0}$$

(2.19)

Trong đó :

$$\mathbf{p}^* = \begin{bmatrix} p_1^* & p_2^* & p_3^* & p_4^* & p_5^* & p_6^* \end{bmatrix}^T$$

$$= \begin{bmatrix} q_1 & q_2 & q_3 & q_4 & r\text{otxp} & \text{rotzp} \end{bmatrix}^T$$

$$\mathbf{s} = \begin{bmatrix} s_1 & s_2 & s_3 & s_4 \end{bmatrix}^T$$

$$= \begin{bmatrix} xp & yp & zp & rotyp \end{bmatrix}^T$$

Đạo hàm phương trình (2.19) theo thời gian ta được:

$$\sum_{i=1}^{6} \frac{\partial f_\alpha}{\partial p_i^*} \dot{p}_i^* = -\sum_{k=1}^{4} \frac{\partial f_\alpha}{\partial s_k} \dot{s}_k \qquad \alpha = 1..6 \qquad (2.20)$$

Có thể viết:

$$\begin{bmatrix} \frac{\partial f_1}{\partial p_1^*} & \frac{\partial f_1}{\partial p_2^*} & \cdots & \frac{\partial f_1}{\partial p_6^*} \\ \frac{\partial f_2}{\partial p_1^*} & \frac{\partial f_2}{\partial p_2^*} & \cdots & \frac{\partial f_2}{\partial p_6^*} \\ \vdots & \vdots & \cdots & \vdots \\ \frac{\partial f_6}{\partial p_1^*} & \frac{\partial f_6}{\partial p_2^*} & \cdots & \frac{\partial f_6}{\partial p_6^*} \end{bmatrix} \begin{bmatrix} \dot{p}_1^* \\ \dot{p}_2^* \\ \vdots \\ \dot{p}_6^* \end{bmatrix} = \begin{bmatrix} -\frac{\partial f_1}{\partial s_1} & -\frac{\partial f_1}{\partial s_2} & -\frac{\partial f_1}{\partial s_3} & -\frac{\partial f_1}{\partial s_4} \\ -\frac{\partial f_2}{\partial s_1} & -\frac{\partial f_2}{\partial s_2} & -\frac{\partial f_2}{\partial s_3} & -\frac{\partial f_2}{\partial s_4} \\ \vdots & \vdots & \vdots & \vdots \\ -\frac{\partial f_6}{\partial s_1} & -\frac{\partial f_6}{\partial s_2} & -\frac{\partial f_6}{\partial s_3} & -\frac{\partial f_6}{\partial s_4} \end{bmatrix} \begin{bmatrix} \dot{s}_1 \\ \dot{s}_2 \\ \dot{s}_3 \\ \dot{s}_4 \end{bmatrix}$$

Đặt:

$$\mathbf{J}_p^* = \begin{bmatrix} \frac{\partial f_1}{\partial p_1^*} & \frac{\partial f_1}{\partial p_2^*} & \cdots & \frac{\partial f_1}{\partial p_6^*} \\ \frac{\partial f_2}{\partial p_1^*} & \frac{\partial f_2}{\partial p_2^*} & \cdots & \frac{\partial f_2}{\partial p_6^*} \\ \vdots & \vdots & \cdots & \vdots \\ \frac{\partial f_6}{\partial p_1^*} & \frac{\partial f_6}{\partial p_2^*} & \cdots & \frac{\partial f_6}{\partial p_6^*} \end{bmatrix}$$

(2.21)

$$\mathbf{J}_s = \begin{bmatrix} -\frac{\partial f_1}{\partial s_1} & -\frac{\partial f_1}{\partial s_2} & -\frac{\partial f_1}{\partial s_3} & -\frac{\partial f_1}{\partial s_4} \\ -\frac{\partial f_2}{\partial s_1} & -\frac{\partial f_2}{\partial s_2} & -\frac{\partial f_2}{\partial s_3} & -\frac{\partial f_2}{\partial s_4} \\ \vdots & \vdots & \vdots & \vdots \\ -\frac{\partial f_6}{\partial s_1} & -\frac{\partial f_6}{\partial s_2} & -\frac{\partial f_6}{\partial s_3} & -\frac{\partial f_6}{\partial s_4} \end{bmatrix}$$

(2.22)

Thế vào ta nhận được phương trình:

$$\mathbf{J}_p^* . \dot{\mathbf{p}}^* = \mathbf{J}_s . \dot{\mathbf{s}}$$

(2.23)

Hay

$$\dot{\mathbf{p}}^* = \mathbf{J}^{*-1}_p . \mathbf{J}_s . \dot{\mathbf{s}} = \mathbf{J}_{inv} . \dot{\mathbf{s}}$$

(2.24)

Trong đó

$$\mathbf{J}^{*-1}_p . \mathbf{J}_s = \mathbf{J}_{inv}$$

(2.25)

2.3Bài toán gia tốc

2.3.1 Bài toán thuận

Đạo hàm phương trình (2.13) theo thời gian ta có:

$$\sum_{i=1}^{6}\frac{\partial f_\alpha}{\partial p_i}\ddot{p}_i+\sum_{j=1}^{6}\sum_{i=1}^{6}\frac{\partial f_\alpha^2}{\partial p_i.\partial p_j}\dot{p}_i.\dot{p}_j=-\sum_{k=1}^{4}\frac{\partial f_\alpha}{\partial q_k}\ddot{q}_k-\sum_{l=1}^{4}\sum_{k=1}^{4}\frac{\partial f_\alpha^2}{\partial q_l.\partial q_k}\dot{q}_l.\dot{q}_k$$

(2.26)

$\alpha=1..6$

Hay có thể viết (2.26) ở dạng:

$$\mathbf{J}_p.\ddot{\mathbf{p}}=\mathbf{g}$$

(2.27)

Với $\mathbf{J}_d$ xác định theo công thức (2.14)

$$\mathbf{g}=[g_1\ \ g_2\ \ g_3\ \ g_4\ \ g_5\ \ g_6]$$

Ở đây:

$$g_\alpha=-\sum_{j=1}^{6}\sum_{i=1}^{6}\frac{\partial f_\alpha^2}{\partial p_i.\partial p_j}\dot{p}_i.\dot{p}_j-\sum_{k=1}^{4}\frac{\partial f_\alpha}{\partial q_k}\ddot{q}_k-\sum_{l=1}^{4}\sum_{k=1}^{4}\frac{\partial f_\alpha^2}{\partial q_l.\partial q_k}\dot{q}_l.\dot{q}_k$$

(2.28)

Từ hệ thức (2.27) ta nhận được:

$$\ddot{\mathbf{p}}=\mathbf{J}_p^{-1}.\mathbf{g}$$

2.3.2 Bài toán ngược

Đạo hàm hệ phương trình (2.20) theo thời gian ta được

$$\sum_{j=1}^{6}\sum_{i=1}^{6}\frac{\partial^2 f_\alpha}{\partial p_i^*\partial p_j^*}\dot{p}_i^*.\dot{p}_j^*+\sum_{i=1}^{6}\frac{\partial f_\alpha}{\partial p_i}\ddot{p}_i^*=-\sum_{l=1}^{4}\sum_{k=1}^{4}\frac{\partial^2 f_\alpha}{\partial s_k\partial s_l}\dot{s}_l\dot{s}_k-\sum_{k=1}^{4}\frac{\partial f_\alpha}{\partial s_k}\ddot{s}_k$$

(2.29)

Với $\alpha=1..6$

Có thể viết (2.29) dưới dạng:

$$\mathbf{J}_p^*.\ddot{\mathbf{p}}^* = \mathbf{g}^* \qquad (2.30)$$

$\mathbf{J}_p^*$ được tính theo công thức (2.21)

$$\mathbf{g} = [g_1 \;\; g_2 \;\; g_3 \;\; g_4 \;\; g_5 \;\; g_6 \;]$$

$$g_\alpha^* = -\sum_{j=1}^{6}\sum_{i=1}^{6}\frac{\partial^2 f_\alpha}{\partial p_i^* \partial p_j^*}\dot{p}_i^*.\dot{p}_j^* - \sum_{l=1}^{4}\sum_{k=1}^{4}\frac{\partial^2 f_\alpha}{\partial s_k \partial s_l}\dot{s}_l\,\dot{s}_k - \sum_{k=1}^{4}\frac{\partial f_\alpha}{\partial s_k}\ddot{s}_k \qquad (2.31)$$

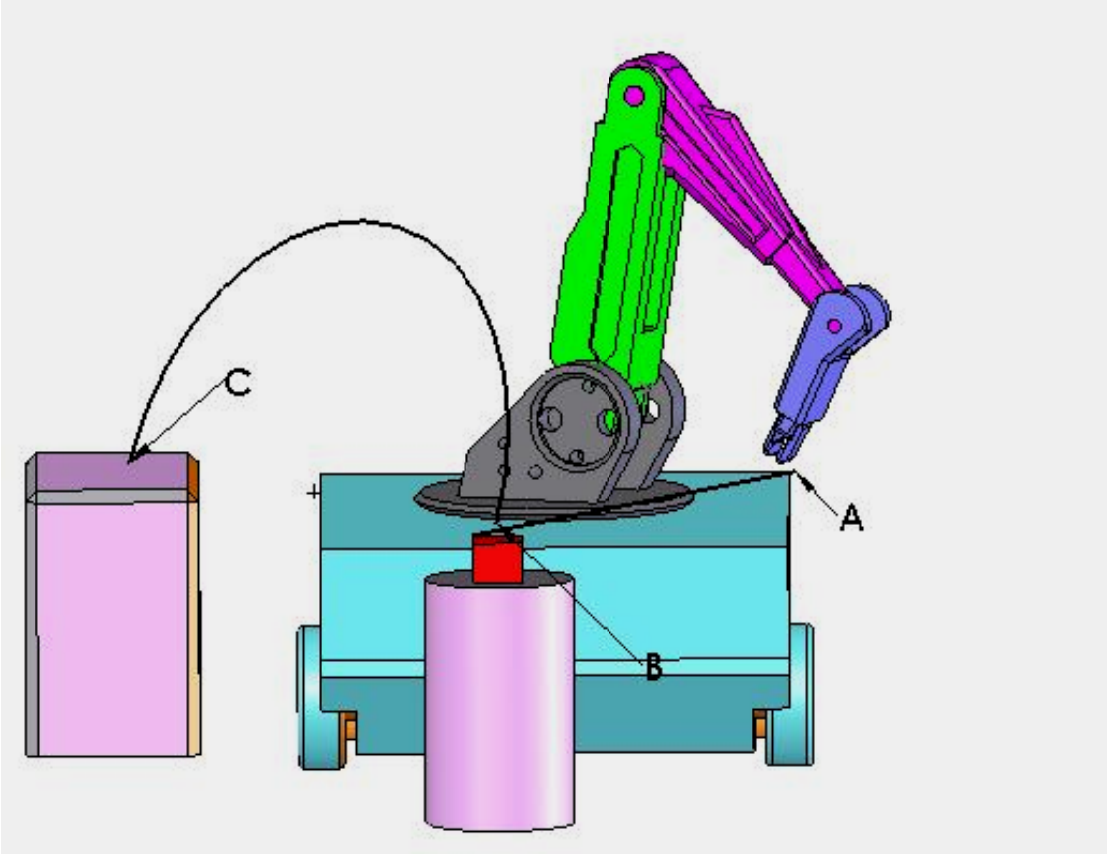
Từ (2.30) ta có :

$$\ddot{\mathbf{p}}^* = \mathbf{J}_p^{*-1}.\mathbf{g}^*$$

2.4 Chuyển động chương trình của robot MMR

2.4.1 Robot thao tác trong quá trình đóng gói sản phẩm

Tính toán cụ thể với robot thực. Tay robot di chuyển từ vị trí $A(x_A,y_A,z_A)$ trong không gian đến vị trí $B(x_B,y_B,z_B)$ để gắp sản phẩm, sau đó mang sản phẩm từ B đến $C(x_c, y_c,z_c)$ cho vào thùng đóng gói như hình 2.3



Hình 2.3

Giả sử robot đi từ A đến B theo một đường thẳng.Ta có phương trình đường thẳng AB có dạng sau:

$$\frac{x - x_A}{x_B - x_A} = \frac{y - y_A}{y_B - y_A} = \frac{z - z_A}{z_B - z_A}$$

(2.32)

Cho robot chuyển động trong thời gian là 20s sẽ đi từ A(80,139,596) đến B(300,250,300). Tại vị trí A các khớp $q_1 = 60^0$, $q_2 = 120^0$, $q_3 = -60^0$, $q_4 = 45^0$.

Từ (2.32) ta có phương trình AB (hình 2.4)

Hình 2.4

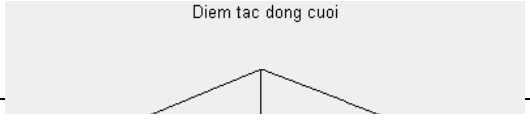
$$\frac{x-80}{300-80}=\frac{y-139}{250-139}=\frac{z-0,132}{300-596}=\frac{t}{20} \qquad (2.33)$$

Rút gọn lại ta được :

$$\Delta 1\begin{cases} x=11.t+80 \\ y=5,55.t+139 \\ z=-14.8*t+596 \end{cases}$$

(2.34)

Sau khi robot đi từ A đến B gắp sản phẩm và tiếp tục đi từ B đến C theo một cung tròn giả sử cùng tròn là nửa đường tròn đường kính BC. Cho tọa độ điểm C(300,50,100). Nhận thấy BC nằm trong mặt phẳng vuông góc với trục Ox (hình 2.5).



Hình 2.5

Ta viết phương trình cung $\overset{\frown}{BC}$ trong mặt phẳng vuông góc với trục Ox có dạng sau :

$$(y-150)^2+(z-200)^2=100\sqrt{2} \quad (2.35)$$

Hay:

$$\left(\frac{y-150}{100\sqrt{2}}\right)^2+\left(\frac{z-200}{100\sqrt{2}}\right)^2=1 \quad (2.36)$$

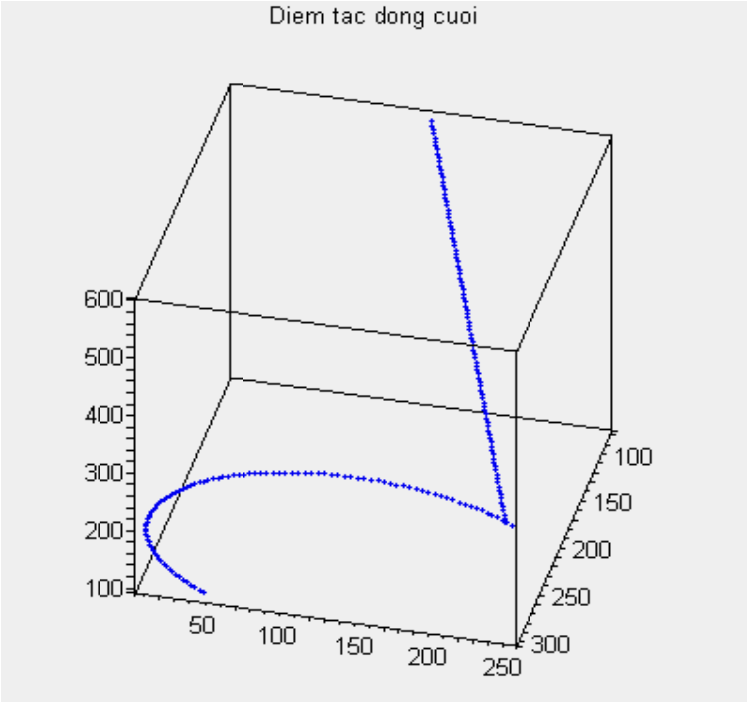
Đặt

$$\frac{y}{100\sqrt{2}}=\sin(at+b)$$
$$\frac{z-200}{100\sqrt{2}}=\cos(at+b) \quad (2.37)$$

Ta cho chuyển động của khâu thao tác đi từ B đến C trong thời gian 20s. Tìm được $\Delta 2$ có dạng như sau:

$$\Delta 2\begin{cases} x = 300 \\ y = 100\sqrt{2}\sin(\frac{\pi.t}{40} + \frac{\pi}{4}) + 150 \\ z = 100\sqrt{2}\cos(\frac{\pi.t}{40} + \frac{\pi}{4}) + 200 \end{cases} \qquad (2.38)$$

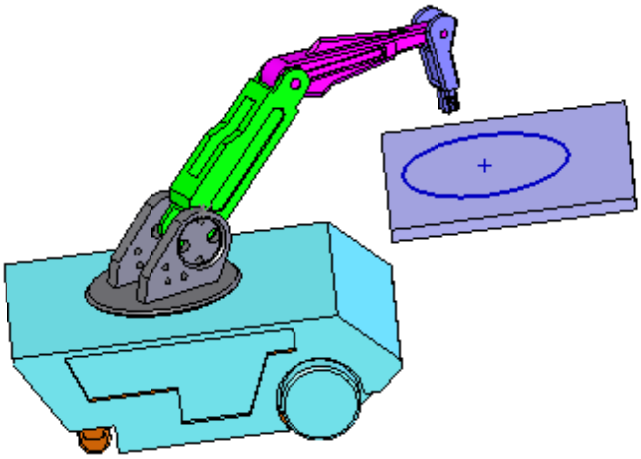
Vậy khi robot di chuyển để thực hiện công việc đi từ A đến B sau đó từ B đến C thì quỹ đạo của điểm tác động cuối có dạng như hình 2.6



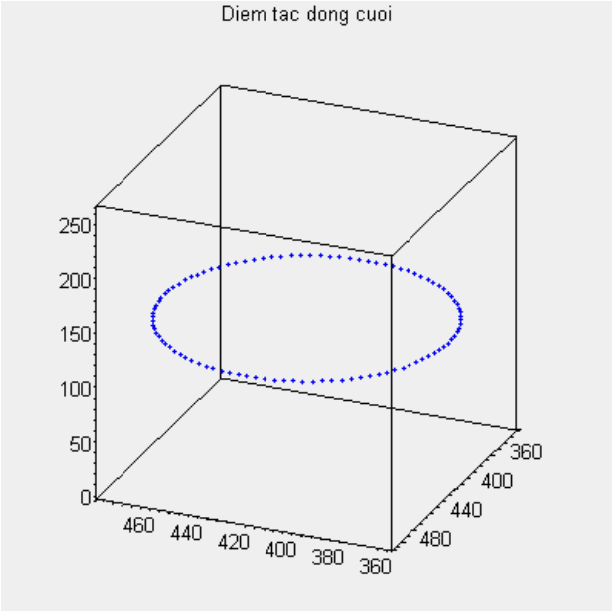
Hình 2.6

2.4.2 Robot thực hiện một công việc trên bề mặt chi tiết

Giả sử robot **MMR** cần phải hàn một bề mặt theo một quỹ đạo hình elip cho trước (hình 2.7)



Hình 2.7



Hình 2.8

Phương trình quỹ đạo khâu thao tác có dạng :

$$\Delta 3 \begin{cases} x = 60\sin(\frac{\pi}{10}t) + 417 \\ y = 80\sin(\frac{\pi}{10}t) + 417 \\ z = 130 \end{cases} \qquad (2.39)$$

CHƯƠNG 3
XÂY DỰNG PHẦN MỀM TÍNH TOÁN VÀ MÔ PHỎNG

Để giải quyết bài toán cơ học ta có thể sử dụng nhiều phần mềm tính toán khác nhau. Trong đồ án này sử dụng *Maple* để tính toán và ghi két quả ra *File* , sau đó dùng Visual C++ để đọc kết quả và mô phỏng.

3.1 Phần mềm ứng dụng tính toán

3.1.1 Giới thiệu về Maple

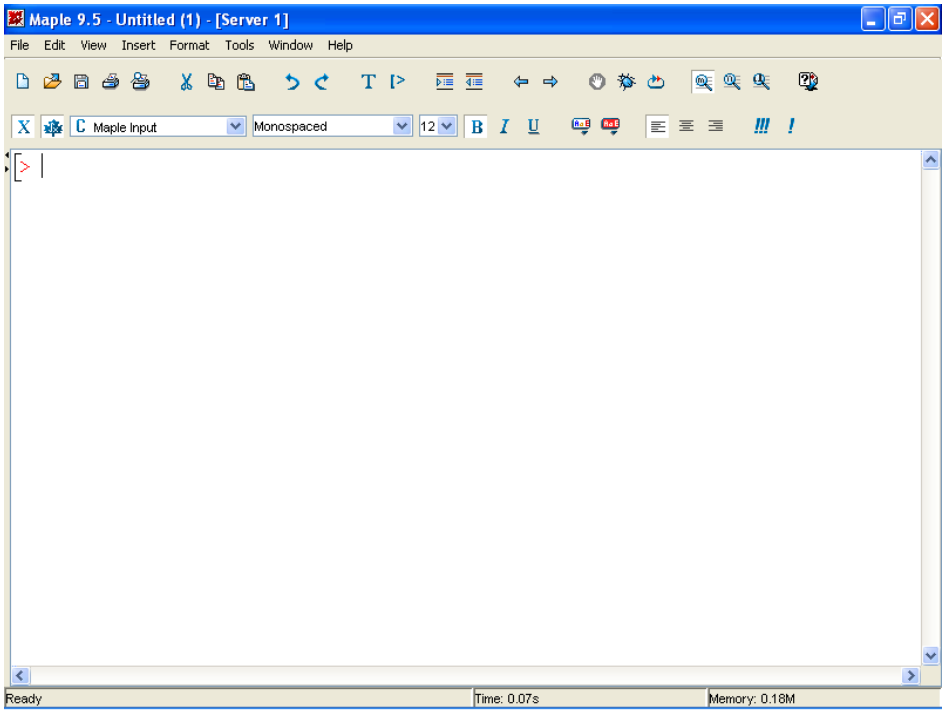
Maple là một phần mềm được phát triển ở trường đại học *Waterloo* ở Canada từ năm 1990 và phát triển tiếp tục. Đây là một phần mềm rất thích hợp dùng cho PC.

Maple là một môi trường tính toán **số** và **chữ** và các ứng dụng đồ hoạ của toán học. Nó không chỉ thuần túy là môi trường tính toán mà còn là một ngôn ngữ lập trình dạng biên dịch. *Maple* cho phép người sử dụng có thể triển khai các ứng dụng một cách nhanh chóng.

Một đặc điểm nổi bật nhất của *Maple* mà hầu như không có một ngôn ngữ nào hiện nay có được chính là khả năng thay thế việc tính toán biến đổi bằng tay bằng việc tính toán biến đổi bằng máy. Khả năng này cực kỳ linh hoạt và phong phú. Điều này làm cho *Maple* trở nên rất hấp dẫn người sử dụng đặc biệt là những người làm về kỹ thuật.

Maple có thể giải quyết rất nhiều vấn đề của toán học như đại số ma trận, vector, giải hệ phương trình phi tuyến, hệ phương trình tuyến tính, tích phân, giải phương trình vi phân thường, phương trình đạo hàm riêng, giải các bài toán về trị riêng, vector riêng, các bài toán đa thức…Nói chung tất cả các lĩnh vực của toán học cổ điển cũng như hiện đại đều có thể tìm thấy trên *Maple*. Điều này giúp chúng ta có thể giảm thiểu thời gian tính toán, dành nhiều thời gian cho việc hoàn thiện mô hình và đánh giá kết quả.

Cửa sổ làm việc chính của *Maple*:



Giao diện của *Maple* cũng giống như giao diện của các chương trình ứng dụng khác trên Windows. Tuy nhiên *Maple* là chương trình thiên về tính toán, nên nó cũng có một số chức năng đặc thù riêng.

Maple là một hệ thống mở, nó cho phép ta tạo ra những ứng dụng riêng mới dựa trên những cái có sẵn rất. Nó cung cấp rất nhiều các thư viện chuẩn.

Cú pháp gọi thư viện:

[> **with**(*library_name*);

library_name : Là tên thư viện cần gọi. Trong *Maple* có rất nhiều thư viện như:

- **Linalg**: Thư viện chương trình đại số tuyến tính.
- **Plots**: Thư viện đồ họa vẽ đồ thị hai chiều và ba chiều.
- **Plottools**: Thư viện đồ họa cung cấp các đối tượng hai chiều và ba chiều như hình trụ, hình cầu, hình nón…

a) Một số lệnh cơ bản

Các lệnh cơ bản như cộng, trừ, nhân, chia ma trận, vector có kí hiệu như toán học thông thường dễ sử dụng.

- [> *restart*: Lệnh này thường dùng khởi đầu một chương trình *Maple*. Sau lệnh này các biến dùng trước nó không còn.
- [>*multiply*(**A,B**): Lệnh nhân hai ma trận. Trong thư viện **linalg**

```
[>restart;
  with(linalg):
> A := array( [[1,2],[3,4]] );
B := array( [[0,1],[1,0]] );
C := array( [[1,2],[4,5]] );
multiply(A, B, C);
```

$$A=\begin{bmatrix}1 & 2\\ 3 & 4\end{bmatrix}$$

$$B=\begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}$$

$$C=\begin{bmatrix}1 & 2\\ 4 & 5\end{bmatrix}$$

$$\begin{bmatrix}6 & 9\\ 16 & 23\end{bmatrix}$$

- [>inverse(**A**): Lệnh tính ma trận nghịch đảo của **A**.

```
> A:=array(1..2,1..2,[[1,2],[3,4]]);
```

$$A=\begin{bmatrix}1 & 2\\ 3 & 4\end{bmatrix}$$

```
> inverse(A);
```

$$\begin{bmatrix}-2 & 1\\ \frac{3}{2} & -\frac{1}{2}\end{bmatrix}$$

- [>diff(**f**(x),x): Đạo hàm của hàm **f**(x) theo x.

```
> diff(x*sin(cos(x)),x);
```

$$\sin(\cos(x)) - x\cos(\cos(x))\sin(x)$$

- Subs(var(1)=rep(1),…,var(n)=rep(n),expr): Lệnh thay thế giá trị vào một biểu thức.

```
> subs( x=2, x^2+x+1 );
```

$$7$$

- [>fsolve(eqns, vars, option): Giải hệ phương trình phi tuyến.
  - eqns: Tập các phương trình của hệ
  - var: Tập các biến của hệ.
  - option: Các lựa chọn cho việc giải

```
> f := sin(x+y) - exp(x)*y = 0;
g := x^2 - y = 2;
fsolve({f,g},{x,y},{x=-1..1,y=-2..0});
```

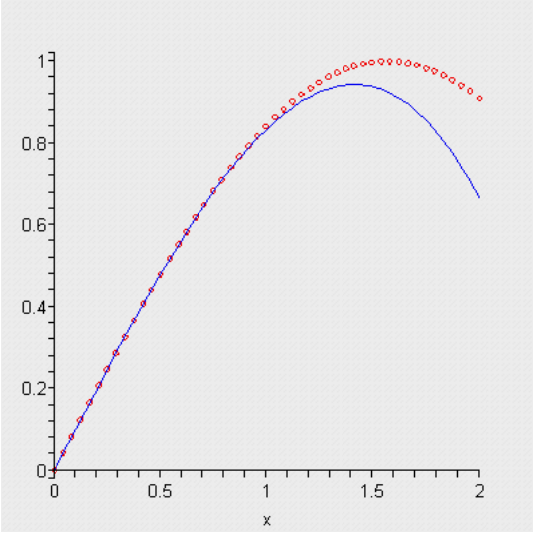
$$f := \sin(x+y) - \mathbf{e}^{x} y = 0$$

$$g := x^{2} - y = 2$$

$$\{x = -0.6687012050,\ y = -1.552838698\}$$

- [plot(): Dùng để vẽ đồ thị

```
[> plot([sin(x), x-x^3/6], x=0..2, color=[red,blue], style=[point,line]);
```



b) Các kiểu dữ liệu cơ bản

Để nắm được *Maple* phải nắm vững các kiểu dữ liệu của nó. Đây cũng là một ưu điểm nổi bật của *Maple* so với ngôn ngữ khác. Các kiểu dữ liệu này giúp cho việc lập trình như tính toán trở lên cực kỳ linh hoạt. Một số kiểu hay dùng:

- Kiểu tuần tự (sequences) đây là một kiểu đơn giản nhất của *Maple*, đó là một nhóm các biểu thức được viết cách nhau bởi dấu phẩy:

```
> seq( i^2, i=1..5 );
```

1, 4, 9, 16, 25

- Kiểu liệt kê (Lists) khác với kiểu tuần tự, các thành phần của danh sách bị bao bởi cặp dấu ngoặc vuông "[" và "]" các phần tử cách nhau bởi dấu phẩy:

```
> L := [1,[2,3],[4,[5,6],7],8,9];
```

$L := [1, [2, 3], [4, [5, 6], 7], 8, 9]$

- Kiểu tập hợp(sets) khác với kiểu tuần tự, các phần tử của tập hợp bị bao bởi cặp các dậu móc nhọn "{" và "}" , cũng cách nhau bởi dấu phẩy. Ý nghĩa của kiểu tập hợp rất giống trong toán học:

```
> S := {v,w,x,y,z};
```

$S := \{x, y, v, w, z\}$

- Kiểu mảng (arrays) đây là kiểu dữ liệu thông dụng như trong các ngôn ngữ lập trình khác.

Cú pháp : name:=array(…)

```
> A:=Array([[1,2,3],[4,5,6]]);
```

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$$

- Kiểu bảng (table) giống kiểu **record** trong *pascal* hoặc kiểu **Struct** trong *C*.

```
> S := table([(2)=45,(4)=61]);
```

$$S := table([2 = 45, 4 = 61])$$

- Kiểu xâu ký tự (String) để khai báo một xâu ký tự ta sử dụng hai dấy phẩy kép:

```
> myname:="Do Viet Hung";
```

$$myname := \text{"Do Viet Hung"}$$

c) Lập trình trong Maple

- Các loại toán tử thường sử dụng khi lập trình:

Toán tử	Ký hiệu
Số học	+, -, *, /, ^, **
Quan hệ	<, >, <=, >=, <>, =
Logic	and, or, not, xor

- Các kiểu dữ liệu đã trình bày ở trên.
- Lệnh rẽ nhánh **if**

if <conditional expression> **then**
<statement sequence>
else
<statement sequence>
end if;

Hoặc sử dụng nhiều lệnh rẽ nhánh

if <conditional expression> **then**
<statement sequence>
elif <conditional expression> **then**
<statement sequence>
else
<statement sequence>

end if;

Ví dụ

> a := 3; b := 5;

$$a := 3$$

$$b := 5$$

> if (a > b) then a else b end if;

$$5$$

- Lệnh **for**

for <name> **from** <expr> **by** <expr> **to** <expr> **while** <expr> **do**

<statement sequence>

end do;

Ví dụ:

[>for i from 6 by 2 to 100 do print(i) end do;

- Lệnh **while**

while <expr> **do**

<statement sequence>

end do;

Ví dụ:

```
[>tot := 0;
for i from 11 by 2 while i < 100 do
  tot := tot + i
end do;
```

- Xây dựng hàm thủ tục: **proc**

Hàm và thủ tục thường hay được sử dụng để cho chương trình sáng sủa và ngắn gọn. Ý nghĩa của nó hoàn toàn giống hàm thủ tục trong bất kỳ ngôn ngữ lập trình nào.

Name:=**proc**(agseq)::type

local var1::type1, var2::type2, …;

global nseq;

options nseq;

desription stringseq;

statseq

return …

End proc;

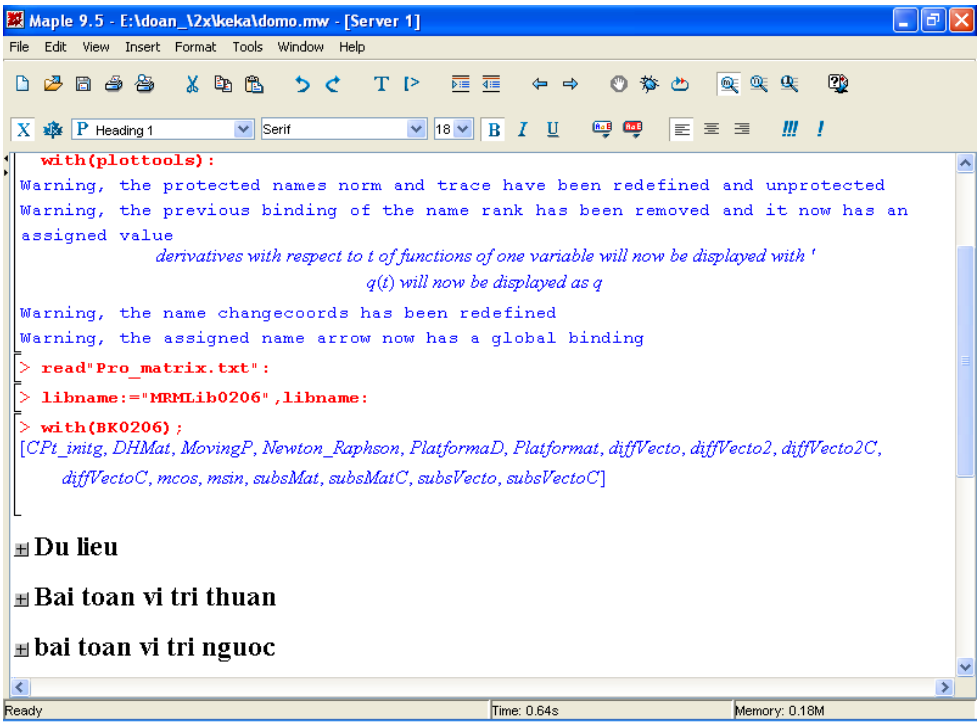
- Thao tác với *File*: **fopen**, **fclose**, **fprintf**, **fscanf**, **readline**, **readdata**.

Đặc điểm của *File* là có thể lưu bất kỳ kiểu dữ liệu với kích thước không hạn chế. Có hai loại *File:* dạng BINARY và dạng TEXT. Việc sử dụng các loại *File* này tùy thuộc vào cách thức tổ chức dữ liệu cũng như yêu cầu của từng bài toán.

fopen	Mở *File*
fclose	Đóng *File*
fprintf	Ghi dữ liệu từ *File*
fscanf	Đọc dữ liệu từ *File*
readline	Đọc một dòng dữ liệu từ *File*
readdata	Đọc một dòng dữ liệu từ *File* có cấu trúc đã định sẵn

3.1.2 Giải bài toán thuận và bài toán ngược

Phần trên ta đã thiết lập cách xây dựng các phương trình của robot. Trong phần này ta trình bày cách giải bằng phần mềm *Maple*.



♦ Mục ***dữ liệu*** dùng lệnh đọc *read* để đọc *File* data và *File* Pro_matrix.

File data.txt

Unknowns := [q[1], q[2], q[3], q[4],rotxp,rotzp]; là các biến cần phải đi xác định trong bài toán ngược của vị trí.

InitrP := [xp, yp, zp,rotxp, rotyp,rotzp]; Vector mô tả vị trí và hướng của điểm tác động cuối trong hệ tọa độ cố định $O_0x_0y_0z_0$.

Init_1:=[1.047197551,2.094395103,-1.047197551, -.7853981635,0.538739,0.13318]: Điều kiện đầu của bài toán, hay chính la giá trị của các biến khớp của robot trước khi làm việc.

#---------- Quy luật chuyển động của khâu thao tác----------

TRAEK_P0:=[11*t+80,5.5*t+139,-14.8*t+596]; Quỹ đạo ta xây dựng trong chương 2

THONG SO DONG HOC DENAVIT-HATENBERG

robot:=table([Init= [0,0,0,0,0,0],

DH =[[theta[1],d[1],a[1],alpha[1]],

[theta[2],d[2],a[2],alpha[2]],

[theta[3],d[3],a[3],alpha[3]],

[theta[4],d[4],a[4],alpha[4]]],

flatform=[xp,yp,zp,rotx,roty,rotz]); bảng động học *Denavit-Hartenberg* của robot.

#----------Khâu1----------

d[1]:=90;

a[1]:= 45;

alpha[1]:= Pi/2;

#----------khâu 2----------

d[2]:=0;

a[2]:=283;

alpha[2]:=0;

#----------khâu 3----------

d[3]:=0;

a[3]:=263;

alpha[3]:=0;

#----------khâu 4----------

d[4]:=0;

a[4]:=130;

alpha[4]:=0;

#----------Các biến khớp----------

theta[1]:=q[1];

theta[2]:=q[2];

theta[3]:=q[3];

theta[4]:=q[4];

#----------

dt:=0.2; Thời gian mỗi bước của robot khi di chuyển.

tg:=0.0;

T_:=20.0; Thời gian robot thực hiện thao tác.

MaxLoop:=100; Số vòng lặp tối đa trong thuật giải *Newton-Raphson*.

AbsErr:=0.0001; Điều kiện để hội tụ của thuật giải *Newton-Raphson.*

` Pro_matrix.txt Gồm các thủ tục tính ma trận *Denavit-Hartenberg, Cardan, ghi File kết quả.*

thủ tục tính ma trận DH

```
Dmat:=proc(theta,d,a,alpha)
local A;
A:=matrix(4,4);
A[1,1]:=cos(theta);
A[1,2]:=-sin(theta)*cos(alpha);
A[1,3]:=sin(theta)*sin(alpha);
A[1,4]:=a*cos(theta);
#---------------------------------------
A[2,1]:=sin(theta);
A[2,2]:=cos(theta)*cos(alpha);
A[2,3]:=-cos(theta)*sin(alpha);
A[2,4]:=a*sin(theta);
#---------------------------------------
A[3,1]:=0;
A[3,2]:=sin(alpha);
A[3,3]:=cos(alpha);
A[3,4]:=d;
#---------------------------------------
A[4,1]:=0;
A[4,2]:=0;
A[4,3]:=0;
A[4,4]:=1;
return (A);
end;
```

thủ tục tính ma trận *Cardan*

```
Cardan:=proc(x,y,z,alpha,Psi,theta)
local A;
A:=matrix(4,4);
```

```
A[1,1]:=cos(Psi)*cos(theta);
A[1,2]:=-cos(Psi)*sin(theta);
A[1,3]:=sin(Psi);
A[1,4]:=x;
#---------------------------------------
A[2,1]:=sin(alpha)*sin(Psi)*cos(theta)+cos(alpha)*sin(theta);
A[2,2]:=-sin(alpha)*sin(Psi)*sin(theta)+cos(alpha)*cos(theta);
A[2,3]:=-sin(alpha)*cos(Psi);
A[2,4]:=y;
#---------------------------------------
A[3,1]:= -cos(alpha)*sin(Psi)*cos(theta) +sin(alpha)*sin(theta);
A[3,2]:=cos(alpha)*sin(Psi)*sin(theta)+sin(alpha)*cos(theta);
A[3,3]:=cos(alpha)*cos(Psi);
A[3,4]:=z;
#---------------------------------------
A[4,1]:=0;
A[4,2]:=0;
A[4,3]:=0;
A[4,4]:=1;
return (A);
end;
```

thủ tục vẽ đồ thị của điểm tác động cuối và các biến khớp.

```
Myplot:=proc(input,time)
local n,m,k,i,j,temp,out;
m:=rowdim(input);
n:=coldim(input);
k:=vectdim(time);
if k<>m then
  print(`ERROR: khong hop le\n`);
  return;
end if;
temp:=vector(n);
```

```
out:=vector(n);
for j from 1 by 1 to n do
   temp[j]:=[seq([time[i],input[i,j]],i=1..m)];
   out[j]:=listplot(temp[j],color=red);
end do;
display([seq(out[i],i=1..n)]);
end:
#----------------------------------------
plot_f:=proc(input,t)
local A,i,n,dt,tg:
n:=101:
dt:=0.2:
A:=matrix(n,3,0):
tg:=0:
for i from 1 by 1 to n do
  for k from 1 by 1 to 3 do
          A[i,k]:=subs(t=tg,input[k]);
  od:
tg:=tg+dt:
od:
return(A):
end:
```

Thủ tục ghi *File* kết quả

```
Write File_in:=proc(A,File_in)
 local N,M,i,j,fd;
N:=rowdim(A);
 M:=coldim(A);
 fd := fopen(File_in, WRITE,TEXT);
 for i from 1 by 1 to N do
  for j from 1 by 1 to M do
  fprintf(fd, "%.9g ",A[i,j]);
  end do;
```

```
        fprintf(fd,"\n");
    end do;
    fclose(fd);
  end;
```

♦ *Mục **bài toán vị trí thuận*** : Đọc *File* main_1.txt.

```
#-------ma tran thuan nhat mo ta vi tri va huong cac khop-
for i from 1 by 1 to 4 do
    A[i]:=Dmat(theta[i],d[i],a[i],alpha[i]):
print(A[i]);
od;
#--- Tinh ma tran mo ta vi tri huong cua khau cuoi ------
T:=multiply(A[1],A[2],A[3],A[4]):
print(T);
#----------------Ma tran mo ta khau cuoi-------
A0:=Cardan(xp, yp, zp,rotxp, rotyp,rotzp):
print(A0);
#------------voi cac bien q[i]-----------------
f[1]:=T[1,4]-A0[1,4];
f[2]:=T[2,4]-A0[2,4];
f[3]:=T[3,4]-A0[3,4];
f[4]:=T[1,2]-A0[1,2];
f[5]:=T[2,3]-A0[2,3];
f[6]:=T[3,1]-A0[3,1];
```

Sau phần tính toán ở mục này thì ta nhận được ma trận $\mathbf{T_4}$ mô tả vị trí và hướng của điểm tác động cuối. Và rút ra 6 phương trình động học như đã trình bày ở chương trước.

♦ Mục ***bài toán vị trí ngược*** đọc *file* main_2.txt.

```
rotyp:=0:
init_:=copy(Init_1):
NZ:=round(T_/dt):
TRAEK_P:=copy(TRAEK_P0):
KQ:=matrix(NZ+1,9,0):
```

```
ttime:=vector(NZ+1,0):
tg:=0.0:
dt:=0.2:
for kz from 1 by 1 to NZ+1 do
f_:=[seq(f[k],k=1..6)]:
Jaco_:=jacobian(f_,Unknowns):
for k from 1 by 1 to 3 do
TRAEK_P_[k]:=evalf(subs(t=tg,TRAEK_P[k])):
end do:
for j from 1 by 1 to 3 do
f_:=subsVectoC([InitrP[j]=TRAEK_P_[j]],f_):
Jaco_:=subsMatC([InitrP[j]=TRAEK_P_[j]],Jaco_):#
end do:
y:=Newton_Raphson(f_,Jaco_,Unknowns,init_,AbsErr,MaxLoop):
for j from 1 by 1 to 6 do
init_[j]:=y[j]:
KQ[kz,j]:=y[j]:
end do:
for i from 1 by 1 to 3 do
KQ[kz,i+6]:=TRAEK_P_[i];
end do:
ttime[kz]:=tg:
tg:=tg+dt:
end do:
WriteFile_in(KQ,"kq.txt");
```

Sau khi đọc *File* main_2.txt sẽ cho ta biết được giá trị của các biến khớp và hướng của điểm tác động cuối để robot có thể đi theo quỹ đạo cho trước. Trong chương trình này đã sử dụng thuật giải *Newton-Raphson* để giải bài toán vị trí ngược. Thủ tục của thuật giải này được đóng gói trong thư viện tiện ích **"MRMLib0206"** và để sử dụng thư viện này ta có thủ tục gọi như sau:

```
> libname:="MRMLib0206",libname:
> with(BK0206);
```

[*CPt_initg, DHMat, MovingP, Newton_Raphson, PlatformaD, Platformat, diffVecto, diffVecto2, diffVecto2C, diffVectoC, mcos, msin, subsMat, subsMatC, subsVecto, subsVectoC*]

3.1.3 Các ví dụ

a) Bài toán thuận

Giả sử ta cho giá trị của các biến khớp

$q_1 = \frac{\pi}{3}, q_2 = \frac{7.\pi}{36}, q_3 = -\frac{\pi}{4}, q_4 = -\frac{5.\pi}{36}$. Áp dụng *Maple* tính được giá trị của vị trí và hướng của điểm tác động cuối.

Khi đó theo công thức (2.10) ta tính được

$$\begin{cases} xp = {}^0\mathbf{A}_f[1,4] = \mathbf{T}_4[1,4] = 321,15 \\ yp = {}^0\mathbf{A}_f[2,4] = \mathbf{T}_4[2,4] = 556,26 \\ zp = {}^0\mathbf{A}_f[3,4] = \mathbf{T}_4[3,4] = 132,08 \end{cases}$$

(3.1)

Từ công thức (2.11) ta có hệ phương trình xác định hướng của vị trí điểm tác động cuối :

$$\begin{cases} f_4 = 0,286788 + \cos(rotyp).\sin(rotzp) \\ f_5 = -0,5 + \sin(rotxp)\cos(rotyp) \\ f_6 = 0,573 + \cos(rotxp)\sin(rotyp)\cos(rotzp) - \sin(rotxp).\sin(rotzp) \end{cases}$$

(3.2)

Giải hệ phương trình trên bằng lệnh **fslove .** Cấu trúc ngữ pháp của câu lệnh này là:

fsolve(tập hợp hệ các phương trình, tập hợp các biến, các tùy chọn điều khiển).

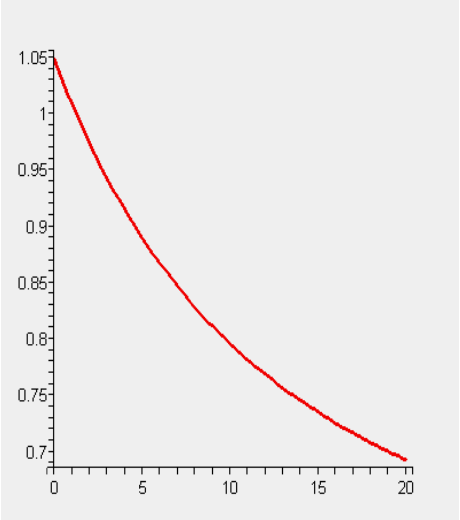
Ta nhận được các giá trị :

$$\begin{cases} rotxp = -10,9955 \\ rotyp = -1,0471 \\ rozp = -8,8139 \end{cases}$$

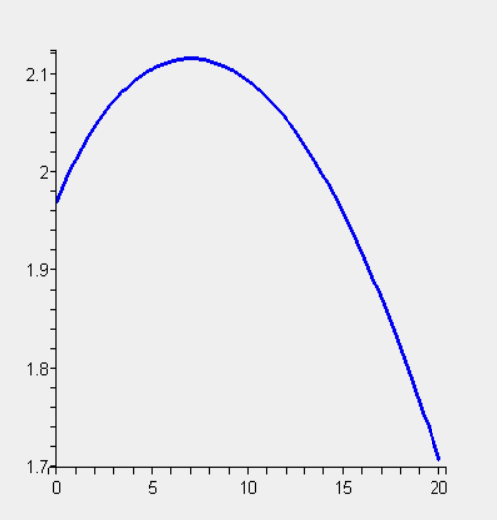
(3.3)

b)Bài toán ngược

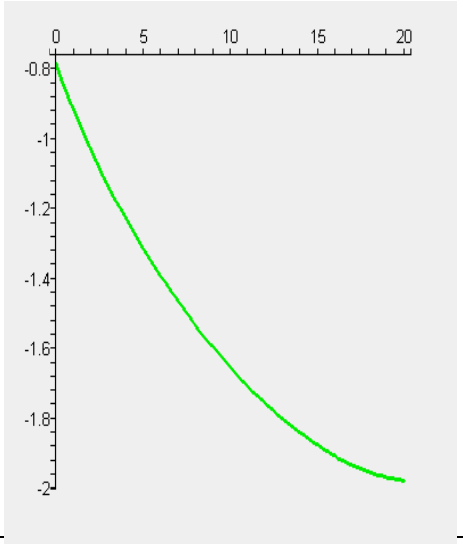
- Khi cho khâu tác động đi theo quỹ đạo Δl là đường thẳng AB(công thức 2.34) ta nhận được các giá trị của q_1, q_2, q_3, q_4 được biểu thị trên đồ thị.

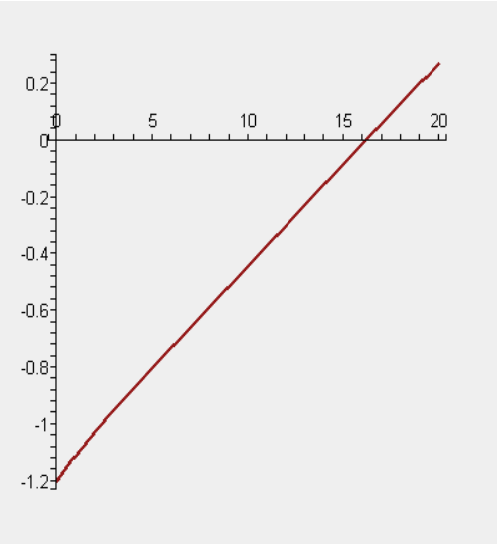


Đồ thị tọa độ suy rộng q_1



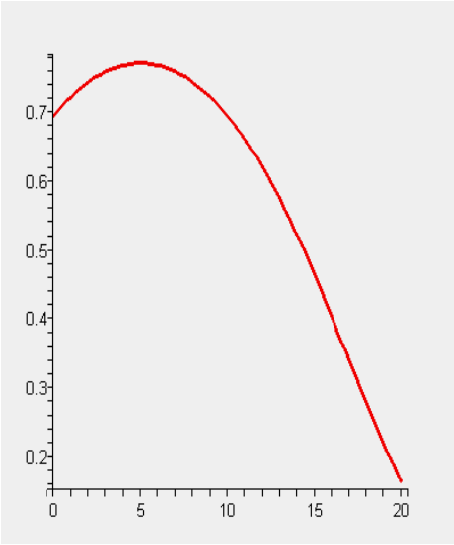
Đồ thị tọa độ suy rộng q_2

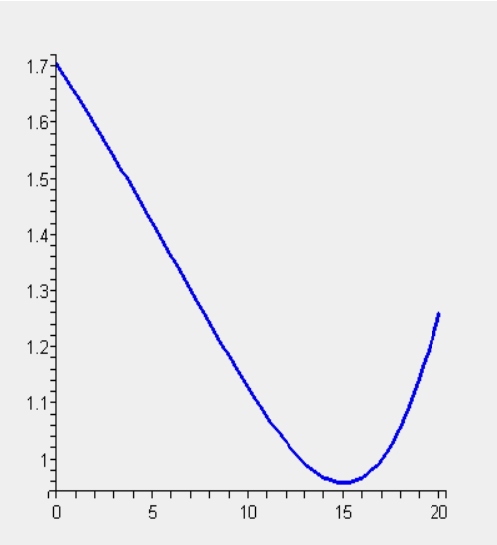




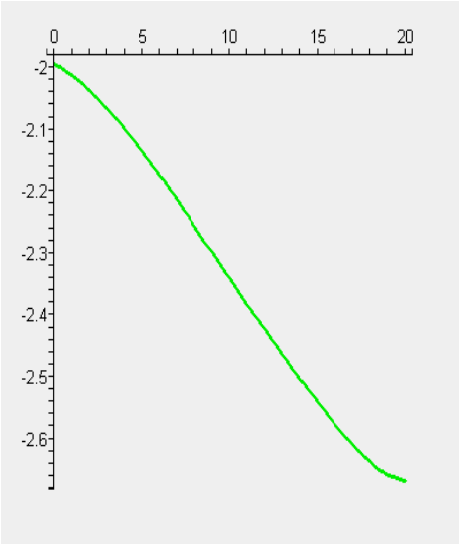
Đồ thị tọa độ suy rộng q_3 Đồ thị tọa độ suy rộng q_4

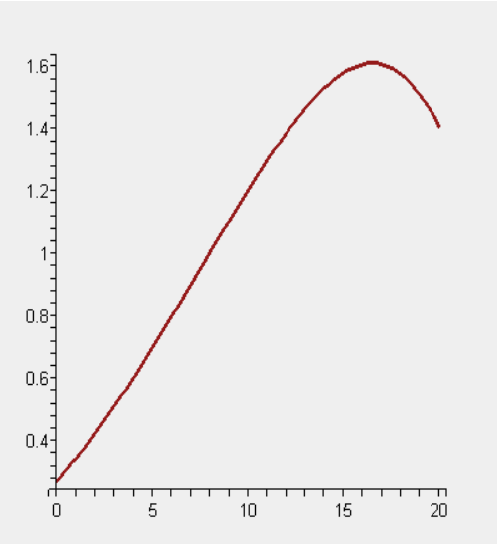
- Khi cho khâu tác động đi theo quỹ đạo $\Delta 2$ là cung tròn $\overset{\frown}{BC}$ (công thức 2.34) ta nhận được các giá trị của $q_1, q_2, q_3, q_4, rotxp, rotzp$ được biểu thị trên đồ thị





Đồ thị tọa độ suy rộng q_1 Đồ thị tọa độ suy rộng q_2





Đồ thị tọa độ suy rộng q_3 Đồ thị tọa độ suy rộng q_4

3.1.3 Giới thiệu về thư viện *OpenGL*

Thư viện đồ họa *OpenGL* là một thiết bị và là một hệ thống các thư viện độc lập sử dụng cho không gian ba chiều. Thư viện *OpenGL* được phát triển bởi tập đoàn *Silicon Graphic Inc* (SGI). Hiện nay *OpenGL* đã trở thành một công cụ được sử dụng rộng rãi trong các hệ điều hành như Windows 9x, Windows NT ..

a)Tổng quan về OpenGL

Mục đích của thư viện *OpenGL* là trả về đối tượng không gian hai chiều và ba chiều vào một một bộ đêm khung (frames buffer) như là điểm nhở của phần cứng đồ họa. Thư viện *OpenGL* về cơ bản là một thủ tục, do đó không cần miêu tả giống đối tượng mà phải chỉ định rõ cách mà đối tượng được vẽ. Đối tượng phức tạp này sẽ được mô tả trong một phần tử đơn giản mà ứng dụng của người dùng định nghĩa. Thư viện *OpenGL* cũng được thực hiện theo mô hình Client- Server.

b)Khái niệm cơ bản về OpenGL

Ở mức độ cơ bản thư viện *OpenGL* giai quyết theo từng đỉnh. Một đỉnh là một điểm, ví dụ như điểm cuối là một đường thẳng, góc của một hình đa giác. Đỉnh có Thể là hai hoặc ba chiều. Ở mức độ tiếp theo sẽ bao gồm một nhóm một hoặc nhiều đỉnh.

c)Khởi tạo OpenGL

Trước khi thư viện *OpenGL* được sử dụng, một số bước khởi tạo cơ bản sẽ được thực hiện.

Hầu hết các ứng dụng Windows sử dụng *OpenGL* phải được kết hợp với một số ngữ cảnh thiết bị. Ngữ cảnh thiết bị phải là một ngữ cảnh hiển thị hoặc là một ngữ cảnh thiết bị nhớ tương ứng với ngữ cảnh thiết bị hiển thị. Để cài một ngữ cảnh trả về, đầu tiên phải sử dụng hàm

SetPixelFormat() để thiết lập dạng của điểm cho thiết bị, tiếp theo gọi hàm *wglCreateContext* nếu thành công hàm này sẽ trả về biến kiểu HGLRC.

d)Vẽ với OpenGL

Hầu hết các chương trình vẽ dùng *OpenGL* sẽ bao gồm một loạt các đỉnh và được đặt trong một cặp *glBegin* và *glEnd* .

3.1.4 Mô phỏng chuyển động của robot MMR

Khả năng mạnh mẽ của thư viện đồ họa *OpenGL* đã được giới thiệu ở trên. Chương trình sử dụng thư việc này trong việc mô phỏng robot dựa trên các giá trị tính toán của bài toán ngược.

Nếu chỉ dựa trên các hàm vẽ cơ bản thì ta khó khăn trong công việc dựng hình 3D cho robot vì việc xác định đỉnh của các khâu rất khó khăn. Nhưng khi ta sử dụng một số *File* định dạng của một số phần mềm đồ họa 3D như: **AutoCAD, Solid Works, 3Dmax**. Các *file* này dễ dàng đọc được bằng phần mềm. Các định dạng *file* có có thể liệt kê ra như:

+ Stereo LithorGaphy(.STL)

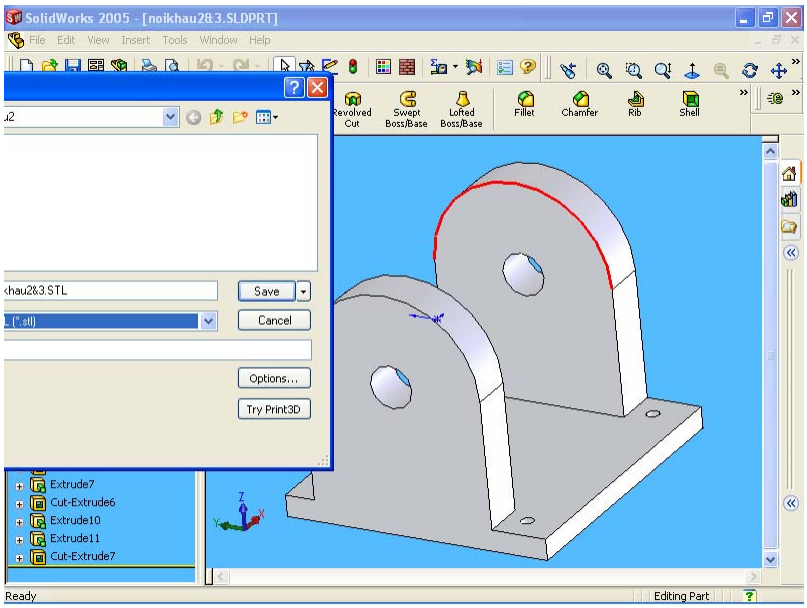
+WRML *File*

+ASC *File*

+SAT *File*

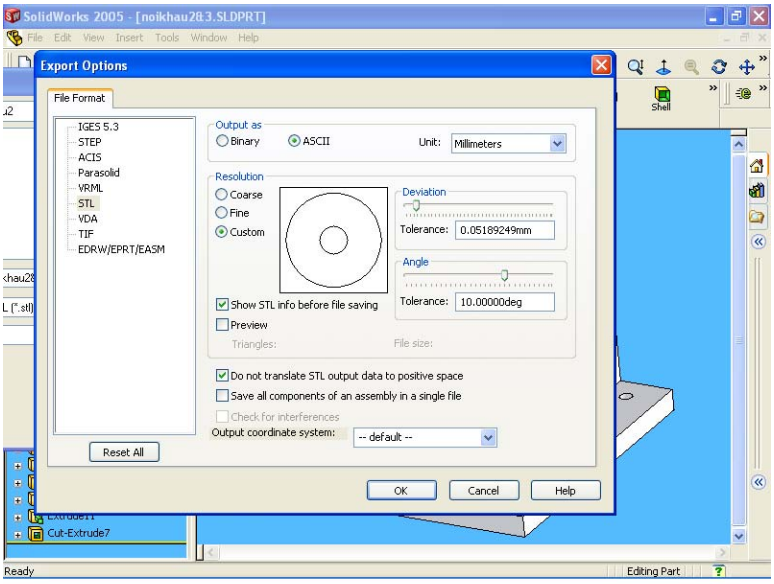
Các *file* này đều có chung đặc điểm là lưu trữ dữ liệu của vật thể 3D dưới dạng các mặt nhỏ (FACE) và các Vector pháp tuyến cùng với một số thông tin khác về vật thể và môi trường như ánh sáng, vật liệu,… Trong đồ án này sử dụng định dạng *.STL, *file* này có thể được xuất ra dưới dạng nhị phân hoặc text.

Khi thiết thế trên phần mềm Solid works xong ta *Save* đối tượng dưới dạng *. STL như hình vẽ (hình 3.1).



Hình 3.1

Chú ý khi *Save* ta chọn *Options* có cửa sổ hiện ra, trong *output as* chọn *ASCII* .



Hình 3.2

Sau khi *save* dưới định dạng .STL ta tiến hành việc mô phỏng chuyển động của robot. Để xuất các khâu của robot vào trong môi trường làm việc của *OpenGL* ra sử dụng hàm *OnlnitialUpdate()*

```
void CAaView::OnInitialUpdate()
    {
        CView::OnInitialUpdate();
        // TODO: Add your specialized code here and/or call the base class
        a.addObjects(CObjects("draw/xe.stl"));
        a.addObjects(CObjects("draw/khau1.stl"));
        a.addObjects(CObjects("draw/khau2.stl"));
        a.addObjects(CObjects("draw/khau3.stl"));
        a.addObjects(CObjects("draw/khau4.stl"));
        CVector scale(_scale, _scale, _scale);
        a.setScale(scale);
    }
```

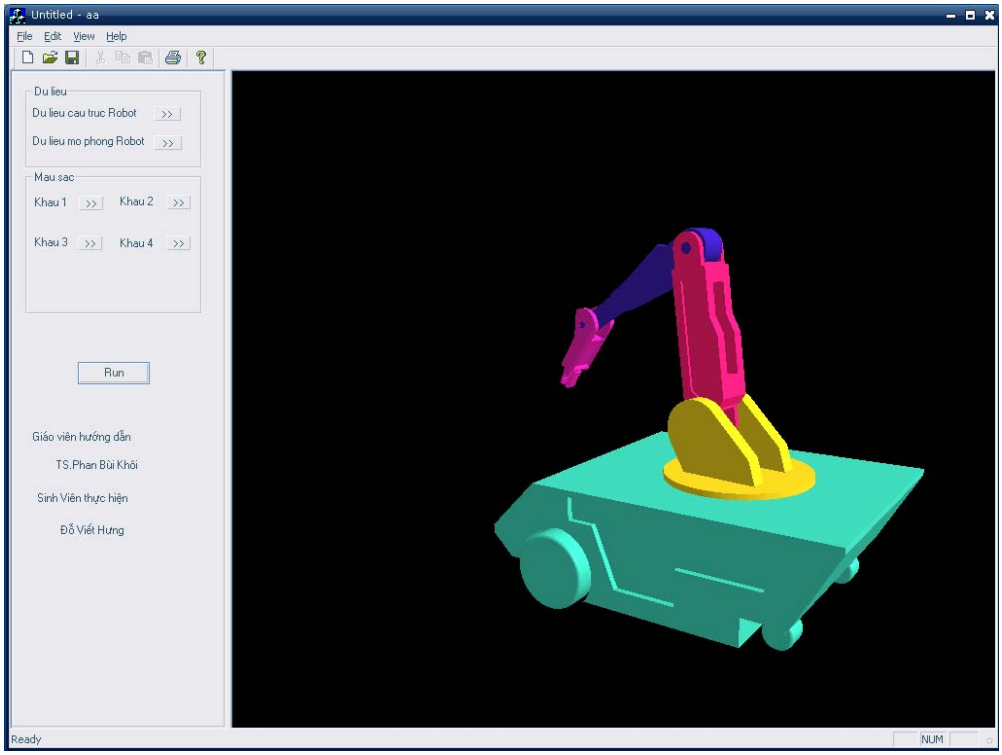
Sau khi insert các khâu vào trong môi trường làm việc xong ta tiến hành đưa các khâu và khớp vào đúng vị trí theo quy tắc *Denavit-Hartenberg* và mô phỏng bằng hàm *Draw()*

```
 void CAaView::Draw()
{
glClearColor(0.0f,0.0f,0.0f,0.0f);
glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT);
//glShadeModel(GL_SMOOTH);
CMainFrame* pMain = (CMainFrame*)AfxGetApp()->GetMainWnd();
CCcommand* pForm = (CCcommand*)pMain->m_wndSplitter.GetPane(0,0);
max = pForm->max;
glColor3f(1.0f,0.0f,0.0f);
glLineWidth(1.0);
if(pForm->run == TRUE)
{
        glPushMatrix();//ngu canh          //1
```

```
// Position / translation / scale
        glTranslated(0,-20,30);
        glRotatef(60, 0.0, 0.0, 1.0);
        glRotatef(3, 1.0, 0.0, 0.0);
        glRotatef(-6, 0.0, 1.0, 0.0);
        //chuot
        glTranslated(m_xTranslation,m_yTranslation,m_zTranslation);
        glRotatef(m_xRotation, 1.0, 0.0, 0.0);
        glRotatef(m_yRotation, 0.0, 1.0, 0.0);
        glRotatef(m_zRotation, 0.0, 0.0, 1.0);
        glScalef(m_xScaling,m_yScaling,m_zScaling);
        //glRotated(30,0.0,1.0,0.0);
        //glRotated(Index,1.0,0.0,0.0);
        glColor3f(0.0f,1.0f,0.0f);
        glLineWidth(1.0);
            glBegin(GL_LINES);
                glVertex3d(0, 0, 0);
                glVertex3d(0, 0, 12);
            glEnd();
            glBegin(GL_LINES);
                glVertex3d(0, 0, 0);
                glVertex3d(0, 40, 0);
            glEnd();
            glBegin(GL_LINES);
                glVertex3d(0, 0, 0);
                glVertex3d(50, 0, 0);
            glEnd();
        glColor3f(0.3,0.5,0.8);
        glLineWidth(1.0);
        glPushMatrix(); // push xe        //2
            glColor3f(0.2,0.2,0.2);
            glColor3f(0.2f,0.7f,0.6f);
            a[0].drawObjects();
```

```
                    //********** khau1 ********************/

                    glPushMatrix();
                            glColor3f(0.8f,0.7f,0.1f);
                            glRotated(90,1,0,0);
                            glTranslated(45*_scale,90*_scale,0);
                            a[1].drawObjects();

                            glRotated(pForm->q1[Index],0,0,1);
                            //******************** khau2 **********/
                glPushMatrix();
                                    glColor3f(0.9f,0.1f,0.4f);
                                    glRotated(pForm->q2[Index],0,0,1);
                                    a[2].drawObjects();
                //***************** khau3******************/
                                    glRotated(45,0.0,0.0,1.0);
                                    glPushMatrix();
                                            glColor3f(0.2f,0.1f,0.6f);
                                    glRotated(-(pForm-
                            >q3[Index]),0.0,1.0,0.0);
                                    a[3].drawObjects();
                //***************** khau4******************/
                                            glRotated(25,0.0,0.0,1.0);
                                            glPushMatrix();

        glColor3f(0.8f,0.1f,0.6f);
                                                    glRotated(pForm-
                                            >q4[Index],0.0,0.0,1.0);
                                            a[4].drawObjects();
                                                    glPopMatrix();
                                            glPopMatrix();
                                    glPopMatrix();
                            glPopMatrix();
```

```
                glPopMatrix();
            glPopMatrix();

        glPopMatrix();
    }
```



Hình 3.3: Giao diện chương trình mô phỏng.

PHẦN II

THIẾT KẾ VÀ CHẾ TẠO MẪU ROBOT MMR

CHƯƠNG 1
GIỚI THIỆU CHUNG

1.1 Đặt vấn đề

Trong điều kiện phát triển như ngày nay thì việc thay thế con người làm việc trong môi trường độc hại hay nhỏ hẹp là rất quan trọng. Các công việc như vào trong lò hạt nhân, thông cống ngầm, làm sạch các khoang tàu,…thì việc thay thế con người bằng robot là một giải pháp rất hữu hiệu và khả thi. Trong đề tài này việc nghiên cứu và chế tạo mẫu robot có kích thước nhỏ, di chuyển dễ dàng và thực hiện các thao tác linh hoạt(**MRM**-**M***ini* **M***obile* **R***obot*), các thao tác này có thể là: hàn, phun sơn, tháo gắp các bộ phận cần sửa chữa, loại bỏ các chi tiết thừa v.v.v…Rôbốt thông qua kết nối với máy tính bằng dây cáp sẽ được điều khiển từ xa, ngoài ra trong thực tế rôbốt phải được lắp các hệ thống camera hay sensor dẫn đường.

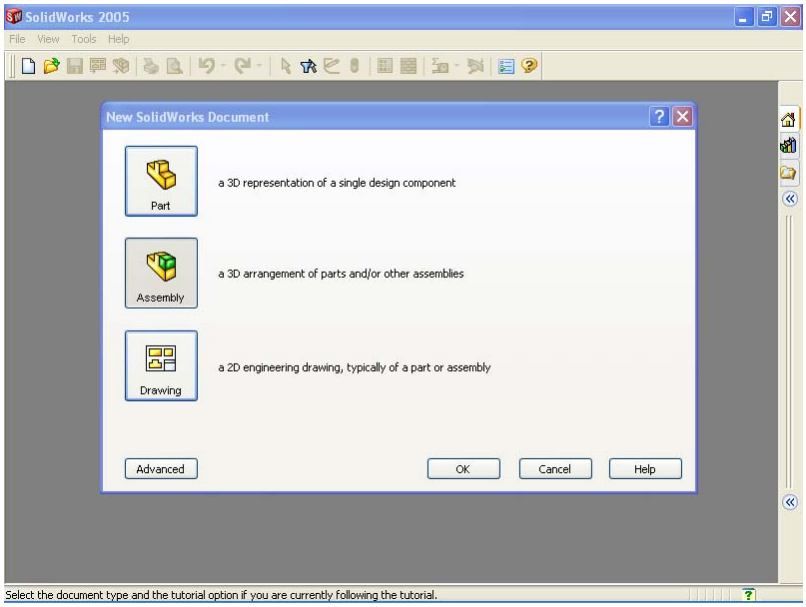
Việc lựa chọn kết cấu sao cho robot nhỏ, di chuyển dễ dàng và thao tác linh hoạt là điều rất quan trọng vì thế trước khi chế tạo cần phải thiết kế mô hình của robot bằng phần mềm đồ họa. Phần mềm đồ họa sử dụng để thiết kế robot **MMR** là *Solid Works* và *Auto CAD*.

Hình ảnh robot **MMR** đã được chế tạo

1.2 Giới thiệu về phần mềm Solid Works

SolidWorks là một phần mềm thiết kế ba chiều được sử dụng rất rộng rãi trong nhiều lĩnh vực khác nhau. Ưu điểm của phần mềm này là rất dễ sử dụng, thân thiện với người dùng, và dễ chỉnh sửa lại bản thiết kế khi cần thay đổi.

Khởi động *Solid Works* và ấn tổ hợp phím Ctrl+N hay vào thanh công cụ ***FILE*/New** hình(1.2)



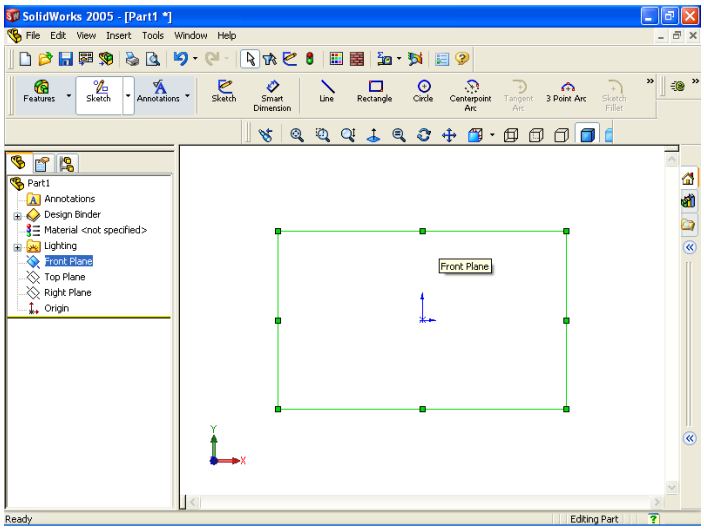
Hình 1.1

Có ba loại bản vẽ,tuỳ theo mục đích sử dụng mà ta mở các bản vẽ khác nhau:

- **Part** (bản vẽ chi tiết): Bản vẽ được sử dụng để tạo các chi tiết riêng lẻ, và trong một bản vẽ chi tiết ta không thể tạo được hai chi tiết.Trong thiết kế cơ khí mỗi một loại máy móc ,một cơ cấu hay một robot… thường có cấu tạo từ nhiều chi tiết khác nhau ghép lại. Mỗi bản vẽ thể hiện từng chi tiết này, và sau đó chúng được lắp ghép trên một bản vẽ khác. Do đó rất thuận tiện khi kiểm tra và thay đổi chi tiết. Các *file* này có phần mở rộng ***.sldprt.**
- **Assembly**(bản vẽ lắp):Bản vẽ này liên kết các chi tiết trong bản vẽ lắp lại với nhau, tạo thành một cụm chi tiết hoặc một sản phẩm hoàn chỉnh. Khi một bản vẽ chi tiết được thay đổi thì bản vẽ lắp tương ứng cũng thay đổi theo. Các *file* này có phần mở rộng ***.sldasm**
- **Drawing** (bản vẽ kĩ thuật): khi đã có bản vẽ chi tiết hay bản vẽ lắp ta chọn Drawing để biểu diễn các hình chiếu các mặt cắt từ bản vẽ chi tiết hay bản vẽ lắp đã có ở trên các *file* này có phần mở rộng là ***.slddrw**.

*a) Bản vẽ chi tiết (**Part**)*

Để thiết kế các chi tiết 3D trước hết phải có các bản vẽ phác thảo, thông thường Solid Works mặc định mặt Front làm bản vẽ phác thỏa, tùy vào kết cấu của chi tiết thiết kế mà ta tạo ra các mặt phác thảo khác nhau. Để bắt đầu vẽ phác thảo phải khởi động thanh menu **Sketch**  trên thanh công cụ. Khi đó giao diện màn hình như sau:



Hình 1.2

Khi đó ta có thể sử dụng các công cụ vẽ trên mặt phẳng của *Solid Works:*

- Công cụ **Line** : Tạo đường thẳng.
- Công cụ **Rectangle** : Tạo hình chữ nhật .
- Công cụ **Centerpoint Arc** :Vẽ cung tròn có tâm xác định.
- Công cụ **Tangent Arc** :Vẽ cung tròn tiếp tuyến.
- Công cụ 3 **Pt Arc** : Vẽ cung tròn bằng ba điểm.
- Công cụ **Circle** : Vẽ đường tròn.
- Công cụ **Ellipse** : Vẽ Ellipse.
- Công cụ **Parabola** **:** Vẽ Parabol.
- Công cụ **Spline** : Vẽ đường cong tự do.
- Công cụ **Centerline** : Vẽ đường tâm.

Ta có thể chỉnh sửa hình phác thảo bằng các công cụ tương tự như trong *Auto CAD*:

- Công cụ **Mirror** : Lấy đối xứng các đối tượng qua đường Centerline.

- Công cụ **Fillet** : Tạo góc lượn.
- Công cụ **Chamfer** : Vát góc.
- Công cụ **Trim** : Được dùng để xén một đoạn của đường thẳng, hoặc đường tròn.
- Công cụ **Offset** : Tạo một đối tương mới có các biên dạng song song và cách đều các biên dạng tương ứng cả đối tượng cũ một khoảng cách cho trước.
- Công cụ **Extend** : Được dùng để kéo dài đối tượng cho tới khi gặp đối tượng khác.
- Công cụ **Linear Step and Repeat** : Công cụ này được dùng để sao chép đối tượng từ đối tượng gốc thành nhiều đối tượng khác và các đối tượng đó được xắp xếp theo hàng hoặc cột.
- Công cụ **Circular Step and Repeat** : Giống như lệnh trên nhưng quỹ đạo sao chép các đối tượng là đường tròn.

Các công cụ tạo mối quan hệ giữa các đối tượng :

- Công cụ **Dimension** : Tạo kích thước cho đối tượng .
- Công cụ **Add Relation** : Tạo quan hệ hình học cho các đối tượng như song song, vuông góc, trùng nhau, tiếp xúc…

Sau khi tạo hình dáng xong rồi ta sử dụng các công cụ tạo khối 3**D** để vẽ:

- Công cụ **Extrude Base/Boss** **:** Nó có chức năng kéo dài đối tượng vẽ phác thành vật thể khối.
- Công cụ **Extrude Cut** **: C**ó chức năng khoét vật thể khối theo biên dạng đã vẽ phác.

- Công cụ **Revolve Base/Boss**  : Có chức năng tạo một khối Base hoặc Boss tròn xoay quanh đường Centerline.
- Công cụ **Sweep**  : Tạo các khối cơ sở, khối dựng đứng ,khoét bằng phương pháp di chuyển biên dạng trên mặt phẳng vẽ phác dọc theo một đường dẫn.
- Công cụ **Linear Pattens**  : Có chức năng sao chép một đặc điểm của mô hình thành nhiều đăc điểm và được xắp xếp theo hàng hoặc cột.
- Công cụ **Circular Pattern**  : Có chức năng sao chép một đặc điểm của mô hình thành nhiều đặc điểm và được xắp xếp theo một đường tròn .

Các công cụ hiệu chỉnh:

- Công cụ **Fillet**  : Chức năng bo tròn các cạnh hoặc các đỉnh của đối tượng.
- Công cụ **Chamfer**  : Vát mép cạnh hoặc đỉnh của đối tượng.

*b) Bản vẽ lắp(**Assembly**)*

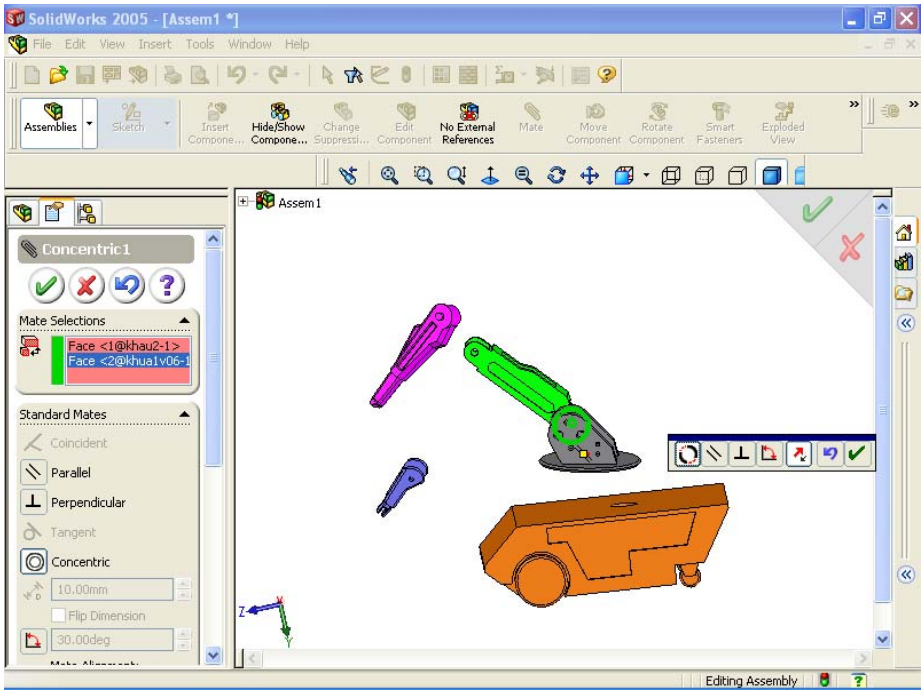
Sau khi đã thiết kế xong tất cả các chi tiết thì cần phải ghép chúng lại với nhau thành một chi tiết hay thành một máy công cụ. Các chi tiết trong bản vẽ lắp cần phải đúng vị trí và có mối quan hệ ràng buộc với nhau.

Công cụ **Mate**  : Cho phép ta tạo các ràng buộc hạn chế một bậc tự do tương đối giữa các chi tiết với nhau tức ghép các chi tiết theo một ràng buộc cụ thể theo cơ cấu và máy cụ thể. Ta có thể chọn các mối ghép như sau:

- **Coincident:** Cho phép ghép hai mặt phẳng tiếp xúc nhau.

- **Concentic**: Cho phép ghép hai mặt trụ, cầu đồng tâm.
- **Parallel**: Cho phép ghép hai mặt phẳng song song và cách nhau một khoảng d.
- **Perpendicular**: Cho phép ghép hai mặt phẳng vuông góc với nhau.
- **Tangent**: Cho phép ghép hai mặt cong, mặt trụ với trụ, mặt cầu với mặt phẳng, mặt trụ và mặt côn với mặt phẳng tiếp xúc với nhau.

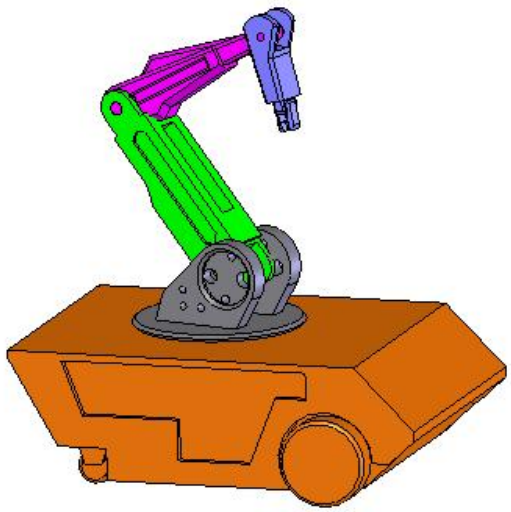
Sau khi thiết kế bước đầu của robot **MMR** ta lắp ghép các khâu vào như sau:



Hình 1.3

CHƯƠNG 2
THIẾT KẾ CƠ KHÍ ROBOT MMR

Mô hình robot **MMR** ban đầu thiết kế có dạng:

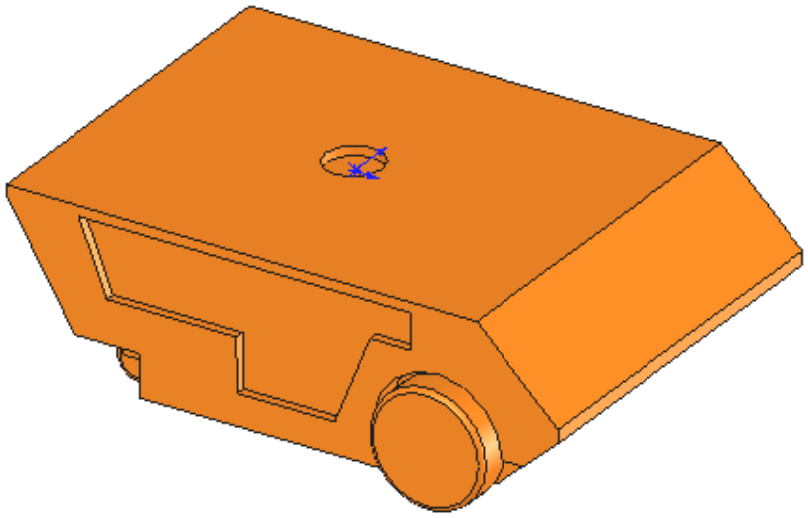


Hình 2.1 Mô hình bề ngoài của **MMR** ban đầu.

Từ mô hình cơ bản trên chúng em đã thiết kế từng khâu, trong quá trình thiết kế cơ khí của robot MMR em đã được giao trách nhiệm quan trọng nhất là thiết kế phần xe . Vì vậy em xin trình bày đầy đủ các bước thiết kế và các bản vẽ chi tiết của xe.

2.1.Đặt vấn đề

Xe là phần vừa di chuyển vừa mang tất cả khối lượng của robot, ngoài ra còn phải chứa các bộ nguồn và mạch điều khiển robot. Trong khi di chuyển xe phải rất chắc chắn, hình dạng phải gọn và phù hợp với điều kiện làm việc. Để đi đến thiết kế cụ thể thì mô hình đầu tiên của xe (hình 2.2) đã được lựa chọn. Robot **MMR** có thể làm việc trong những nơi nhỏ hẹp như là làm sạch khoang tàu, sửa chữa khoang tàu. Vì vậy hình dáng của xe có dạng hình thoi là rất hợp lý, hình dáng này vừa cho khoảng trống trong xe rộng vừa tạo dáng công nghiệp.



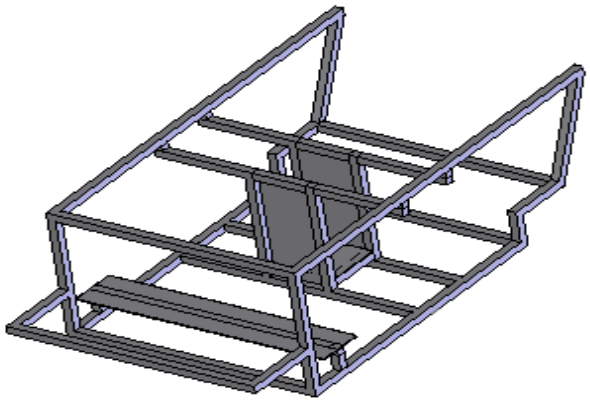
Hình 2.2

Nếu hình dáng xe như vậy mà ta thiết kế theo chi tiết dạng hộp hay các tấm sắt dày ghép lại với nhau thì sẽ không khả thi mà giá cả có thể đắt. Với robot **MMR** này em xin đưa ra phương án thiết kế là: sử dụng các thanh ghép lại thành một khung và sau đó dùng tấm tôn bọc ngoài.

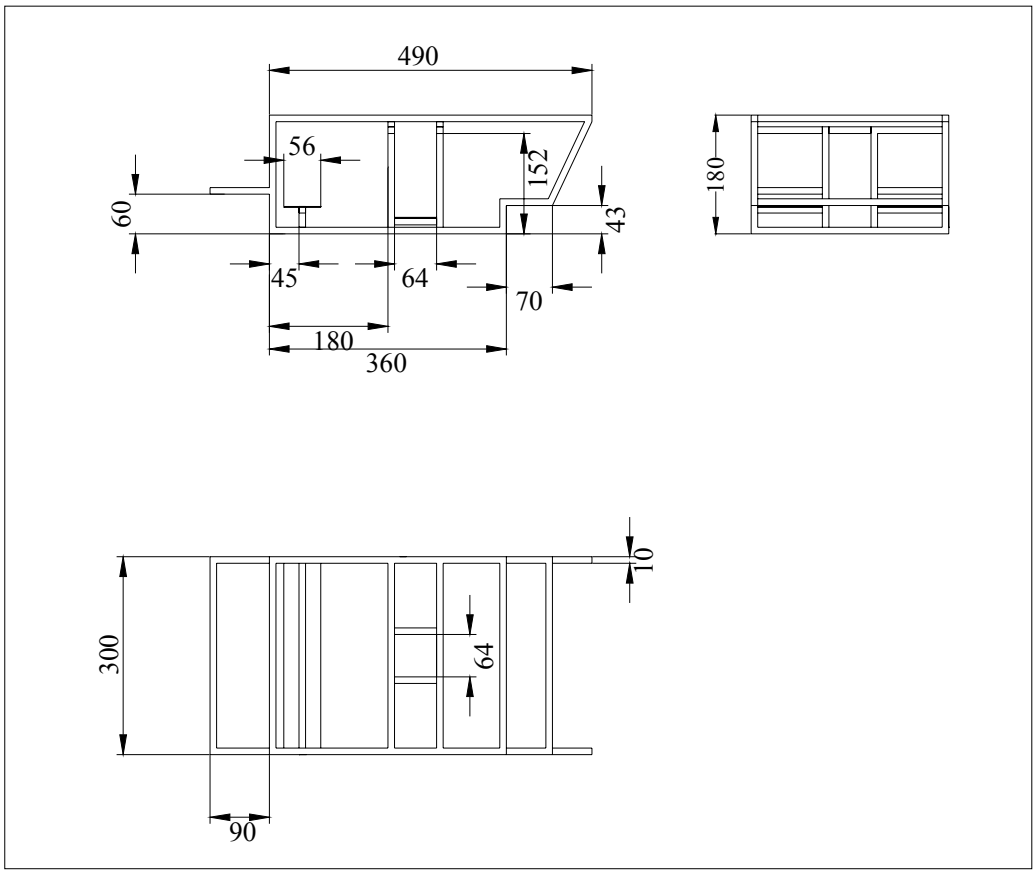
2.2 Thiết kế xe

2.2.1 Khung xe

Khung xe (hình 2.3) được ghép với nhau bởi các thanh nhôm. Để liên kết các thanh nhôm này lại với nhau ta dùng vít hoặc định tán.

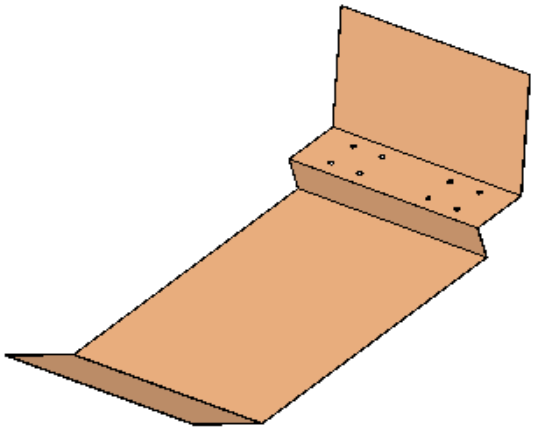


Hình 2.3

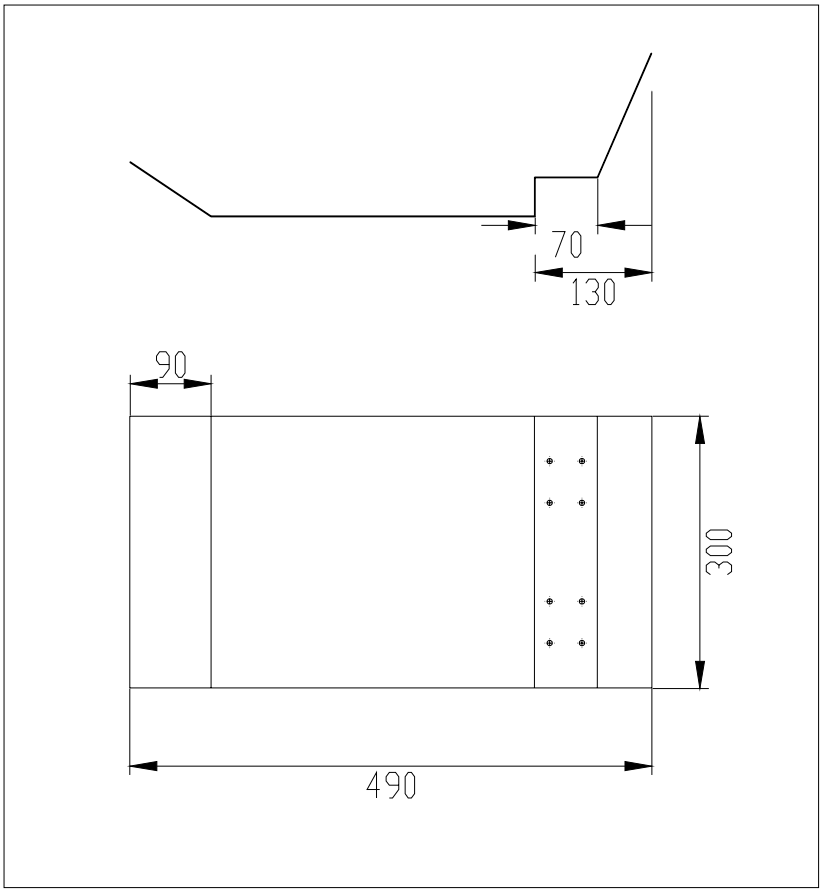


2.2.2 Vỏ bọc của xe

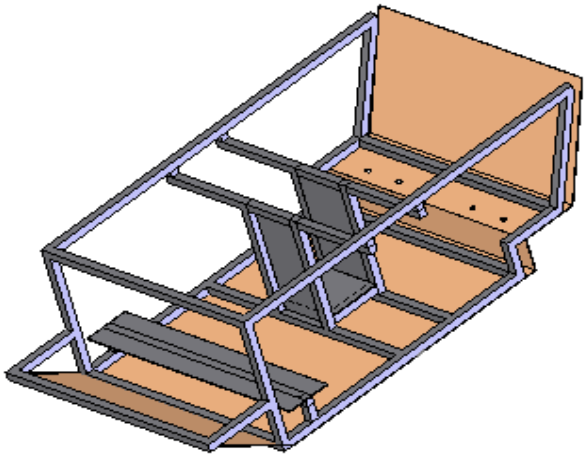
a) Tấm bọc ở gầm xe được làm từ tôn chiều dày của loại tôn đã chọn là 0.3mm



Hình 2.4



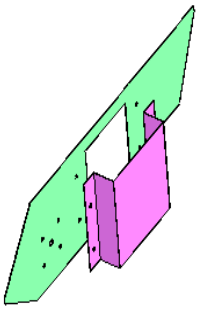
Ta ghép phần khung xe và tấm bọc ngoài này vào như hình 2.5 và được lắp chặt nhờ đinh tán.



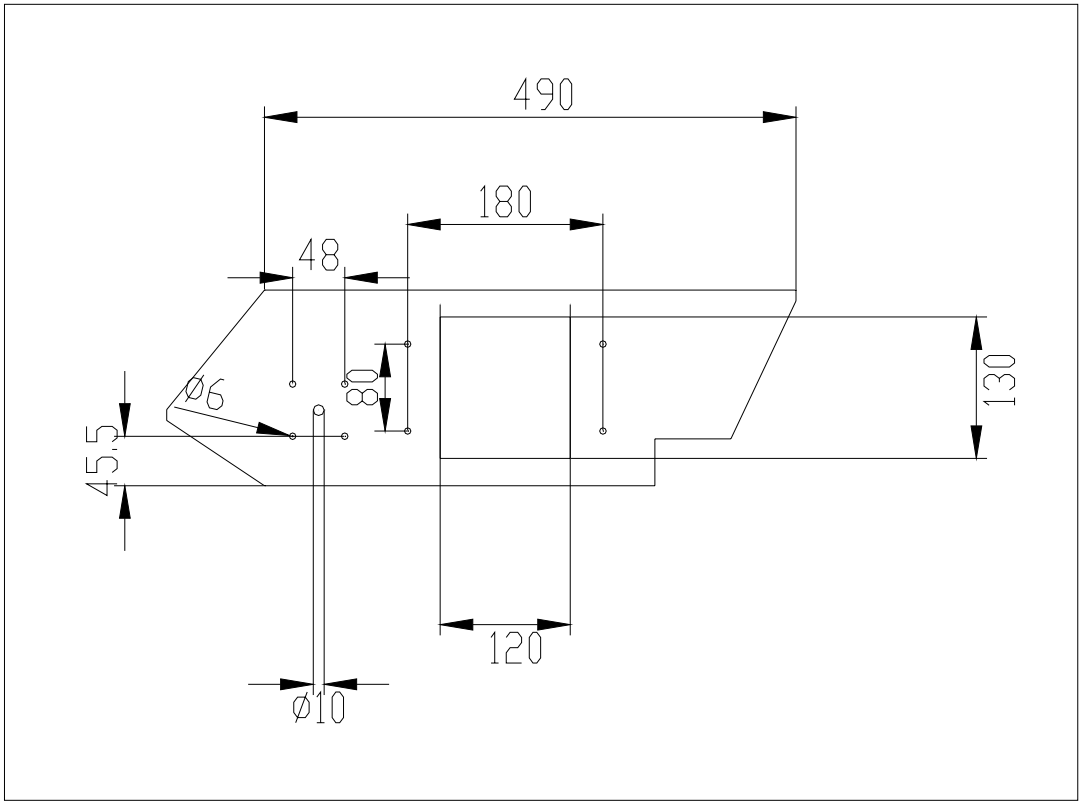
Hình 2.5

b) Tấm bọc 2 bên sườn xe.

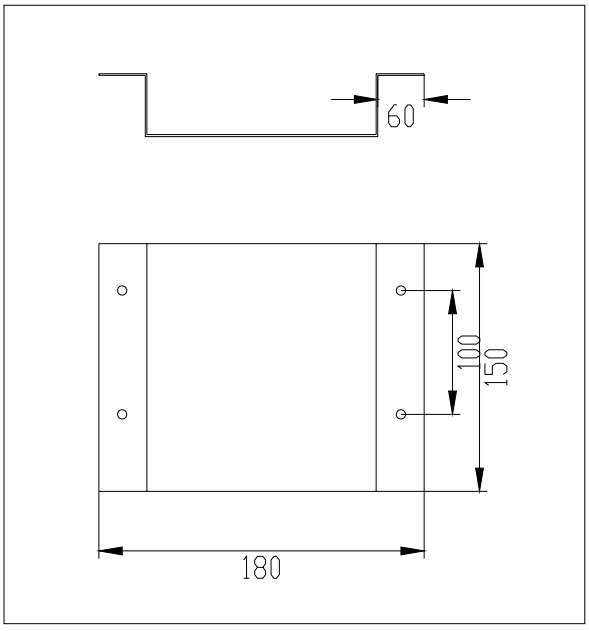
Gồm 4 tấm như hình vẽ (mỗi loại có 2 tấm) Ở 2 tấm to thì có bố trí các lỗ để lắp bánh xe.



Hình 2.6

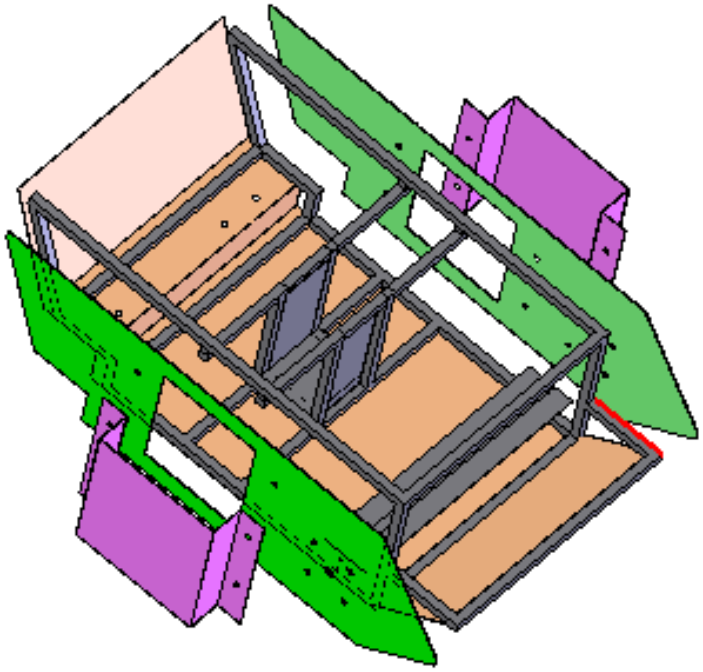


Kích thước của tấm cạnh (gồm 2 tấm)



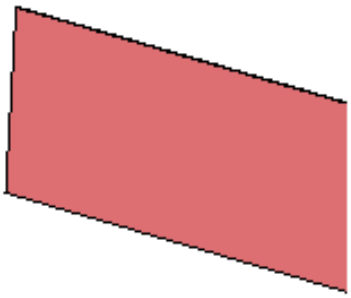
Kích thước tấm ốp 2 cạnh (gồm 2 tấm)

Các tấm được lắp vào khung như (hình 2.7).



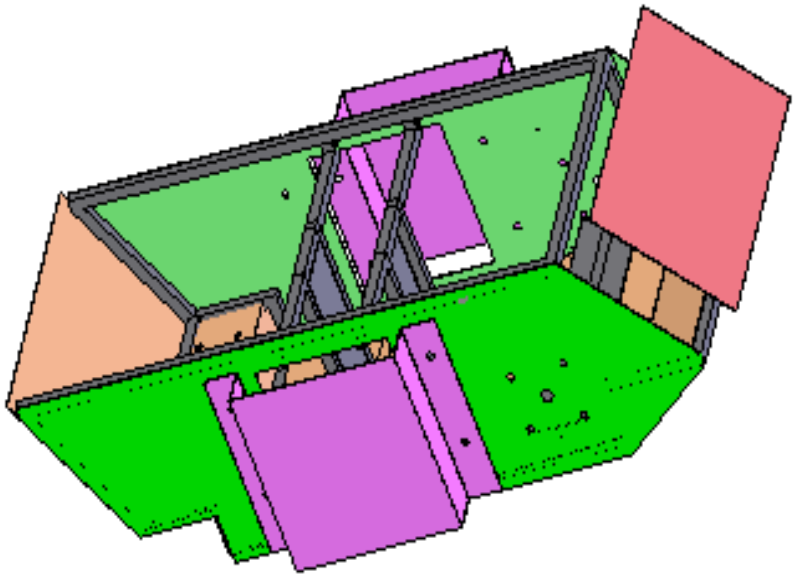
Hình 2.7

c) Tấm phía trước mui xe (hình 2.8) kích thước của tấm là 300 x 150 mm



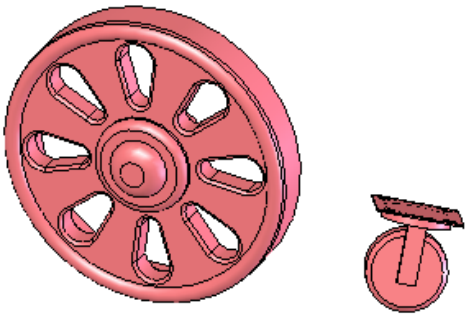
Hình 2.8

Ghép tấm vào xe (hình 2.9)

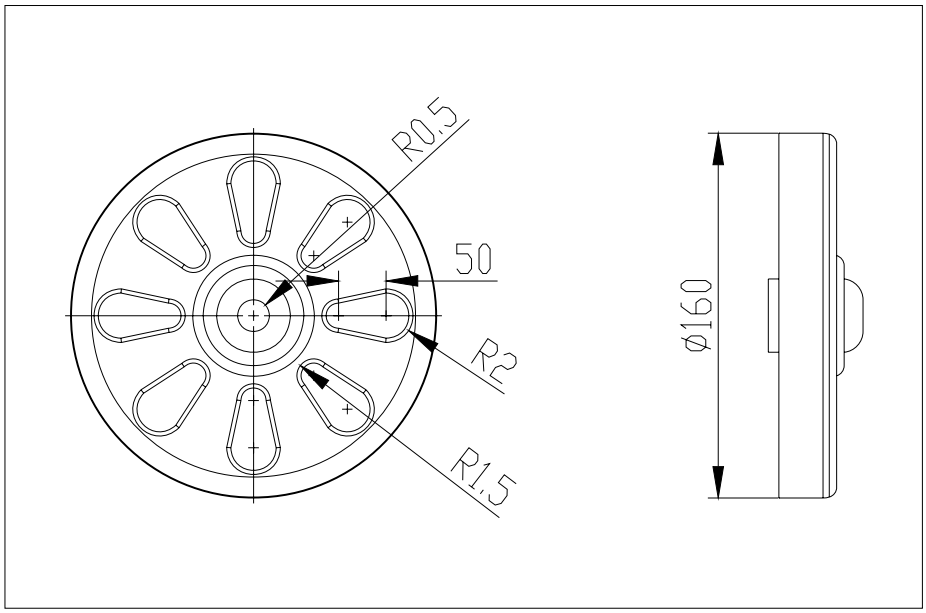


Hình 2.9

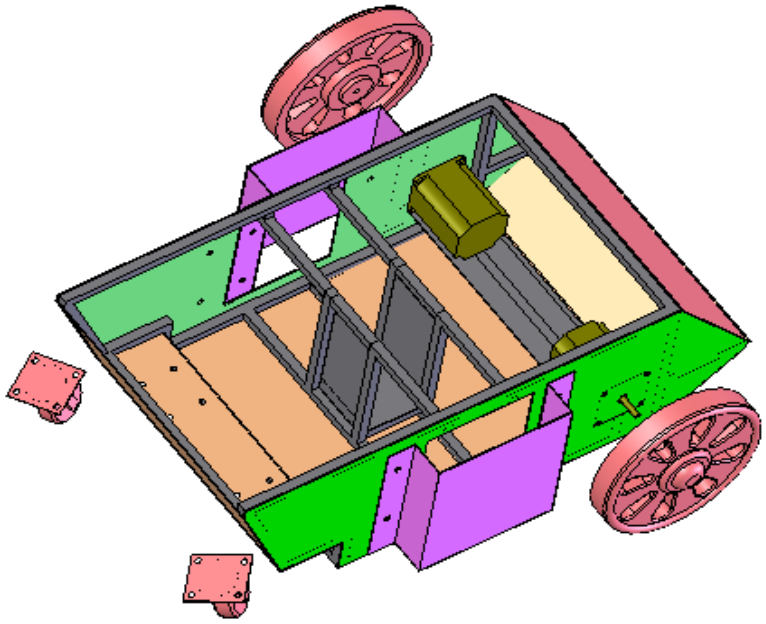
d)Bánh **:** Sau khi đã nắp xong các chi tiết của xe ta phải đi lắp các bánh. Gồm 4 bánh 2 bánh nhỏ và 2 bánh to (hình 2.10).



Hình 2.10

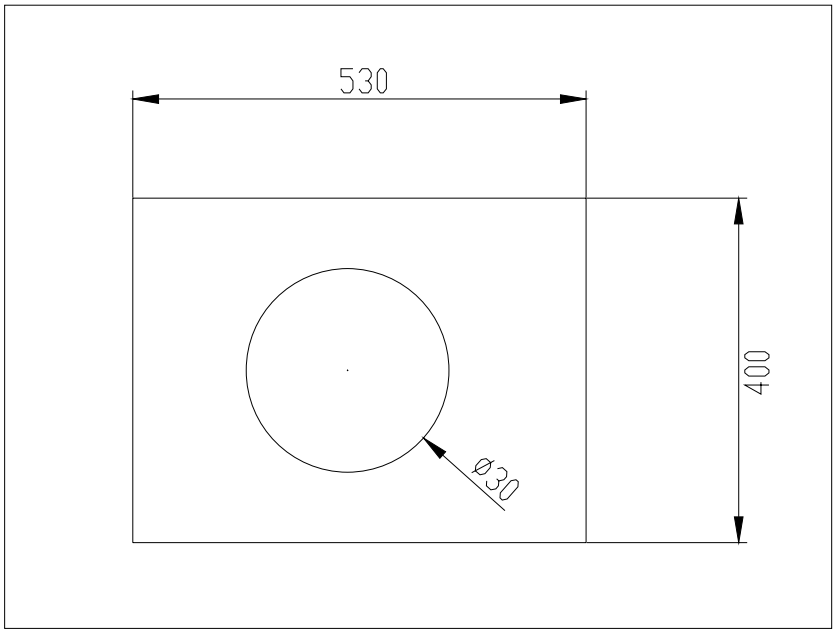


Để điều khiển xe chạy theo các hướng thì ta bố trí 2 động cơ bước gắn trực tiếp vào 2 bánh lớn và lắp vào xe (hình 2.11)

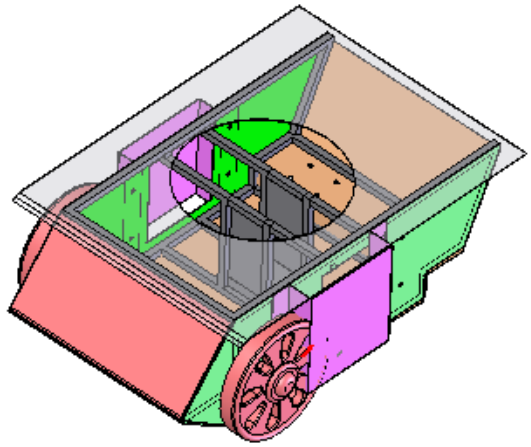


Hình 2.12

Kích thước của nắp trên xe:



Khi các chi tiết của xe đã thiết kế và lắp đặt xong thì phần khoảng trống trong xe là nơi đặt các thiết bị điều khiển (hình 2.13).



Hình 2.13

KẾT LUẬN

❖ Về mặt lý thuyết:

- Đã nắm được cơ sở cơ học robot và các phương pháp nghiên cứu:
  + Phương pháp khảo sát bài toán động học thuận và ngược.
  + Phương pháp xây dựng quỹ đạo cho robot.
- Đã nắm được quy trình tính toán, thiết kế và chế tạo một robot đơn giản.

❖ Về mặt thực hành:

- Đã thiết kế, chế tạo mẫu được mô hình rôbốt mini điều khiển từ xa bằng máy tính. Thông qua mạch điều khiển đã thực hiện được việc kết nối giữa cơ khí và điện tử.Thiết kế được một bộ chương trình gồm các mô đun tính toán, mô phỏng hoạt động, mạch điện tử điều khiển rôbốt.

Việc kết hợp học tập nghiên cứu về lý thuyết gắn liền với thực tế chế tạo đã giúp chúng em hiểu sâu hơn lý thuyết đồng thời bước đầu làm quen với công việc thực tế. Về đề tài này tuy đã có những thành công bước đầu nhưng vẫn còn nhiều hạn chế do đó chúng em đề ra hướng phát triển trong tương lai là:

- Hoàn thiện hơn về kết cấu cơ khí cũng như hệ thống điều khiển để robot có thể hoạt động một cách chính xác.
- Trang bị thêm hệ thống *sensor*, *camera* dẫn đường cho rôbốt để rôbốt có thể hoàn toàn được điều khiển từ xa trong môi trường tách biệt. Đặc biệt có thể lập trình xử lý ảnh giúp rôbốt có khả năng tự động nhận biết và giải quyết thông minh tình huống đặt ra.

TÀI LIỆU THAM KHẢO

[1] Nguyễn Văn Khang: *Cơ sở cơ học kỹ thuật.* Tập I, II. NXB Đại học Quốc gia Hà Nôi 2002.

[2] Nguyễn Văn Khang: *Bài giảng động lực học hệ nhiều vật.* ĐHBK Hà Nội 2002.

[3] Nguyễn Thiện Phúc: *Robot công nghiệp.* NXB Khoa học và kỹ thuật, Hà Nội 2002.

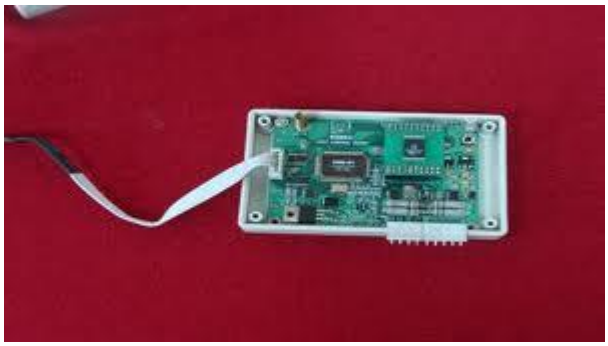
[4] Trịnh Chất- Lê Văn Uyển: *Tính toán thiết kế hệ dẫn động cơ khí*. Tập I, II. NXB giáo dục 2001.

[5] Phạm Huy Điển: *Tính toán, lập trình và giảng dạy trên toán học trên MAPLE.* NXB khoa học kỹ thuật Hà Nội 2002.

[6] Phạm Công Ngô: *Tự học lập trình VISUAL C++6.0*. NXB thống kê 2002.

ĐỒ ÁN TỐT NGHIỆP

VI ĐIỀU KHIỂN ROBOT DÒ ĐƯỜNG



MỤC LỤC

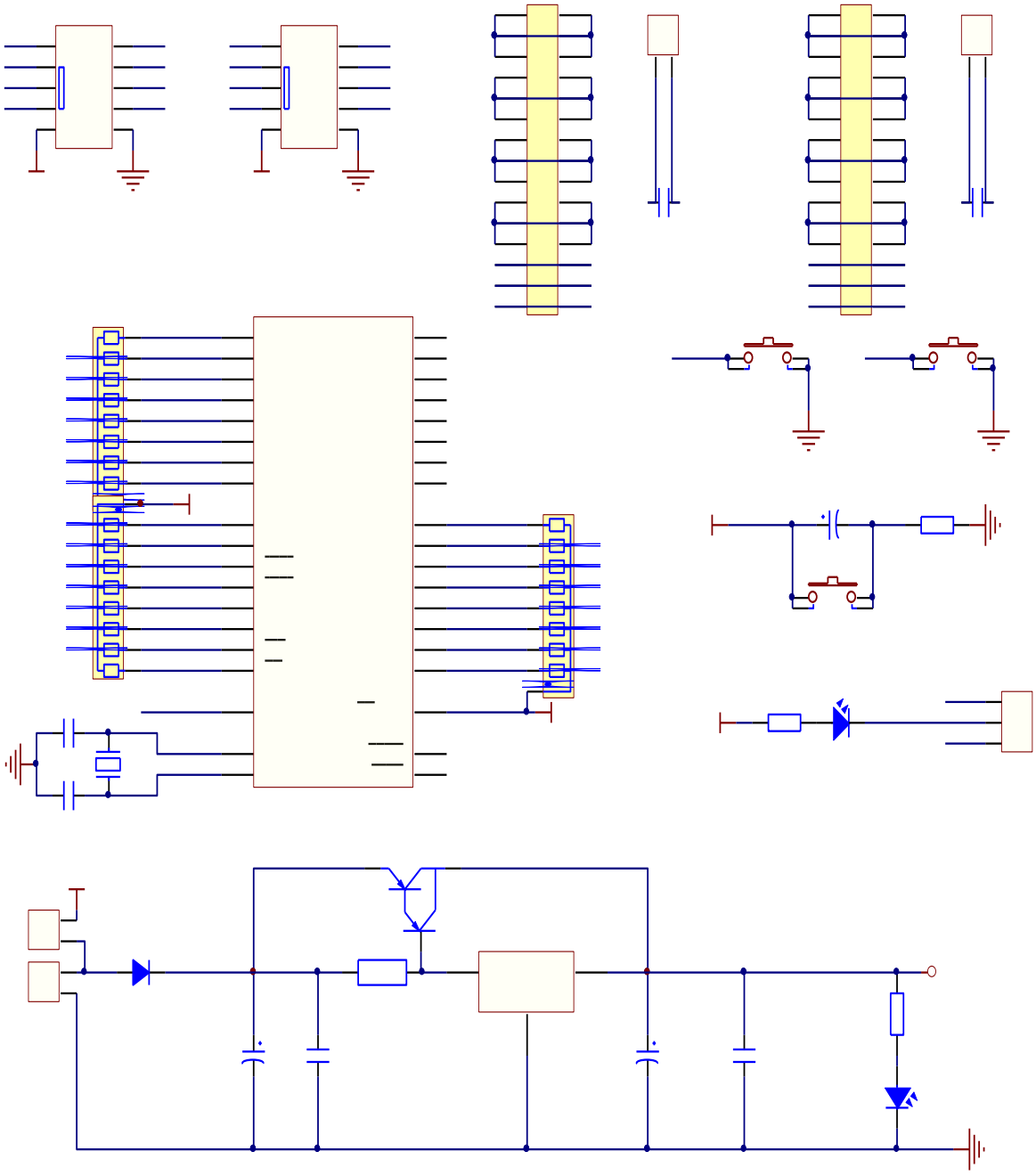
MỤC LỤC 2
Bài báo cáo vi điều khiển Robot dò đường 3
I. Mạch 3
Mạch điều khiển 89S52 3
Mạch nguyên lý 3
Mạch in 4
Mạch điều khiển động cơ " IC MC 33486" 5
Mạch nguyên lý 5
Mạch in 6
Mạch so sánh ("IC LM358") 6
Mắt dò đường 8
Mạch nguyên lý 8
Mạch in 8
II. CODE Robot dò đường 9
TR0=1; 10
TR0=1; 10
TR1=1; 11
TR1=1; 11
TR1=1; 11
TR1=0; 12
TR1=1; 12
TR0=0; 12
TL0=256-tocdo1; 12
TR0=1; 12
TR0=0; 12
TR0=1; 12
TR0=0; 17
TR1=0; 17
IT0=1; 17
IP=0; 17
III. Đồ án Robot dò đường dùng vi điều khiển 89S52 18

Bài báo cáo vi điều khiển Robot do đường

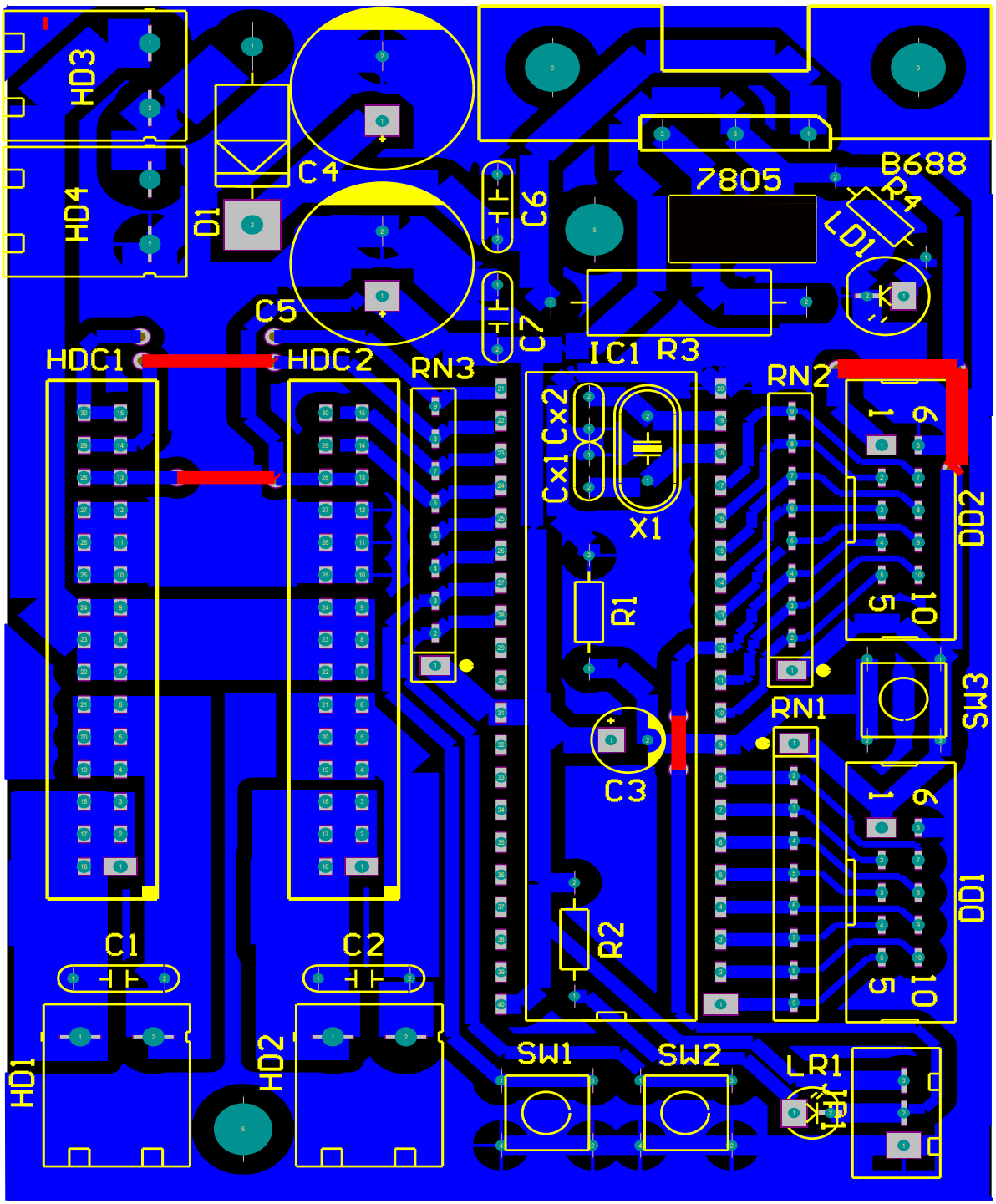
I. Mạch

Mạch điều khiển 89S52

Mạch nguyên lý

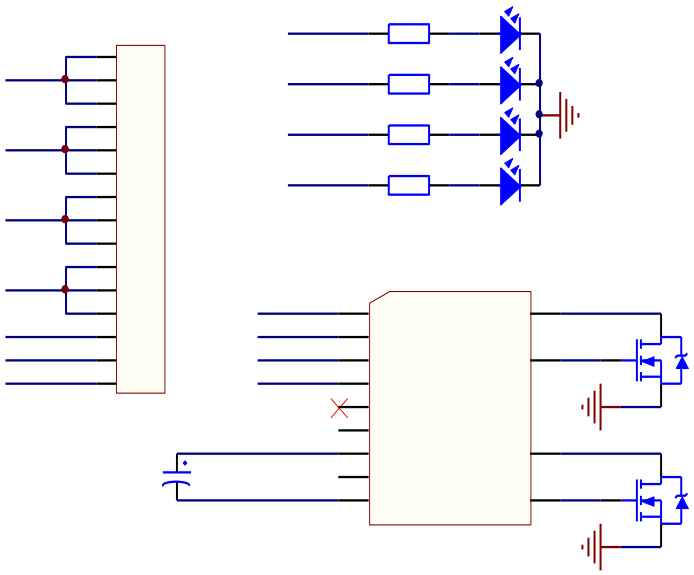


Mạch in

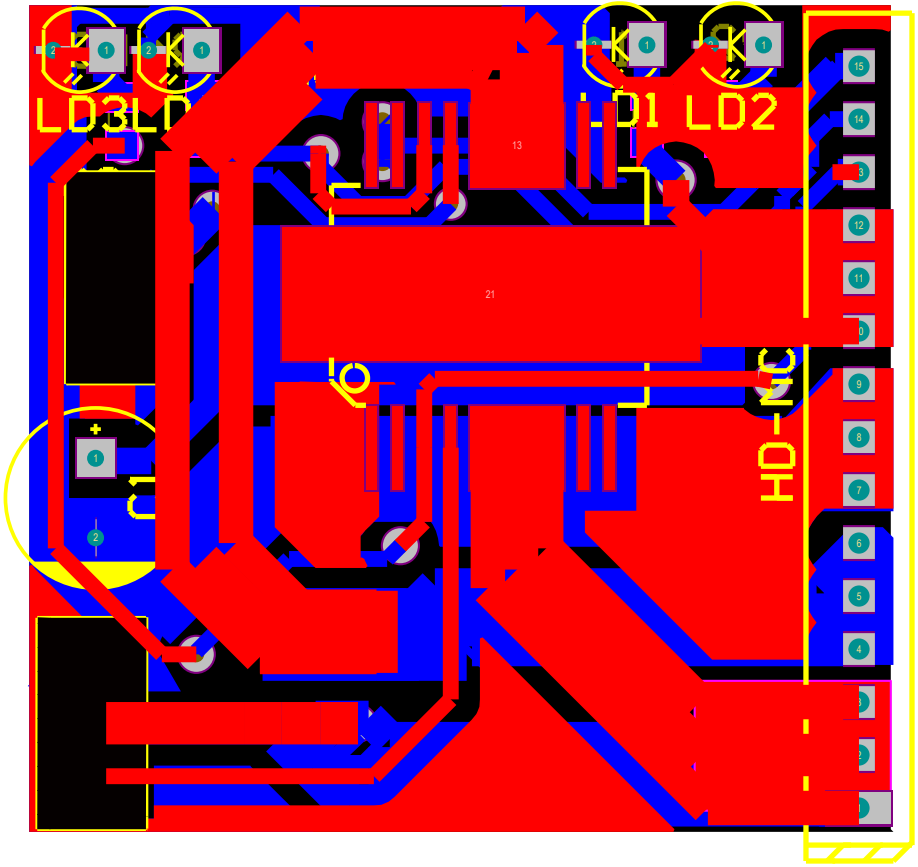

HD3
HD4
D1
C4
C6
7805
B688
R4
LD1
C5
C7
HDC1
HDC2
RN3
IC1
R3
RN2
Cx1
Cx2
X1
R1
DD2
SW3
RN1
C3
DD1
R2
C1
C2
HD1
HD2
SW1
SW2
LR1

Mạch điều khiển động cơ “ IC MC 33486”

Mạch nguyên lý

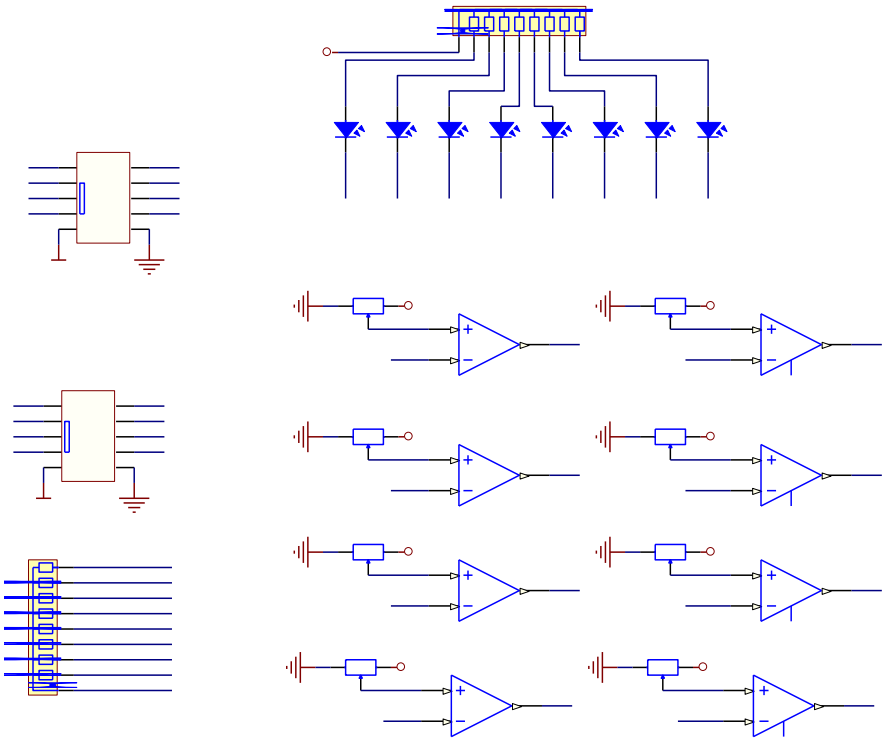


Mạch in

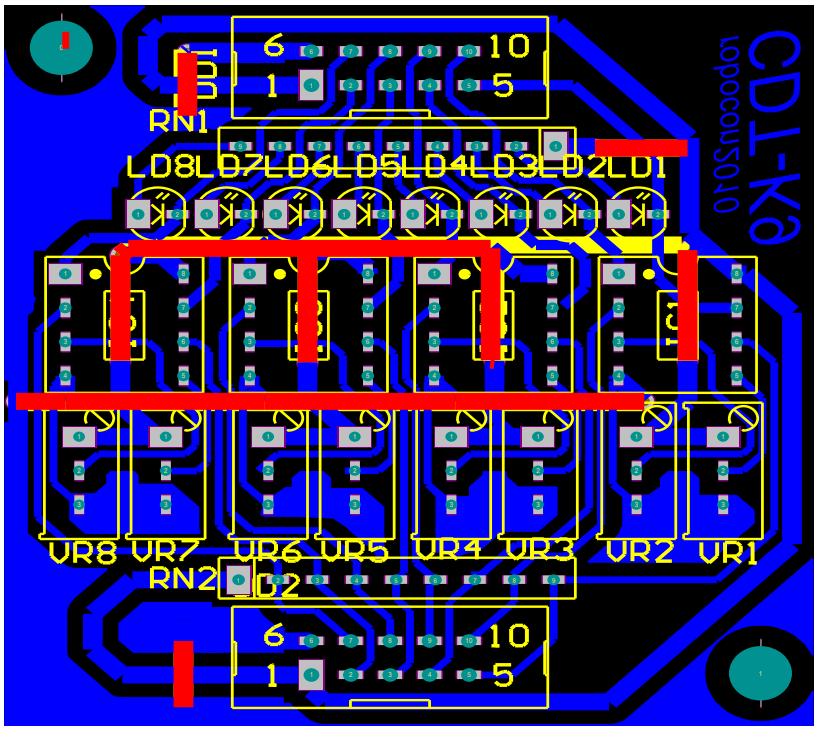


Mạch so sánh (“IC LM358”)

a. Mạch nguyên lý

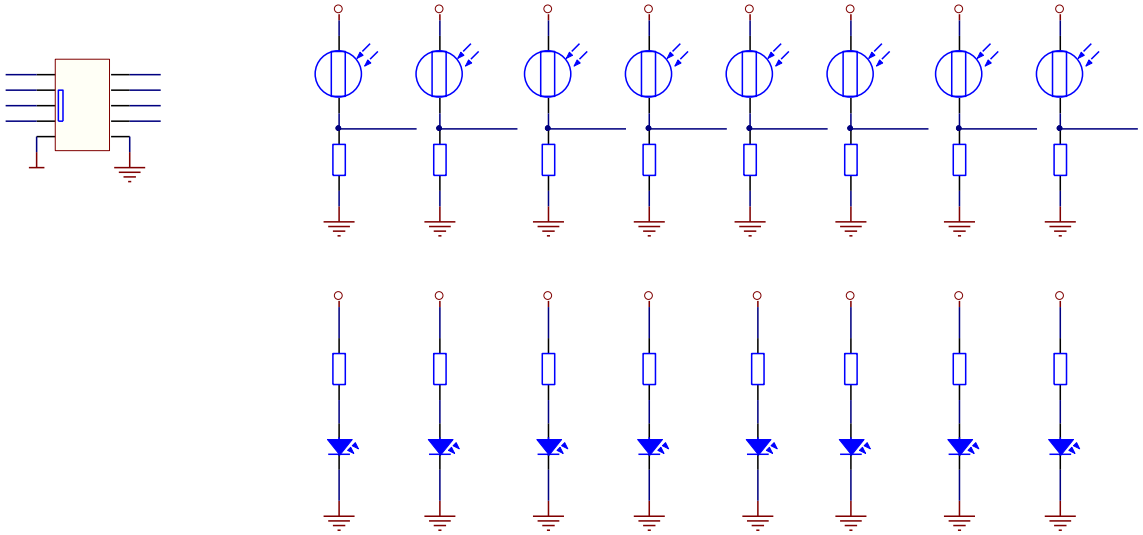


b. Mạch in

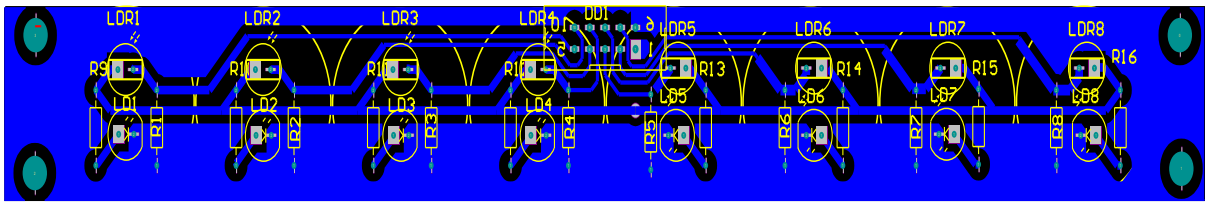

CDT-K9
RN1
LD8LD7LD6LD5LD4LD3LD2LD1
VR8 VR7 VR6 VR5 VR4 VR3 VR2 VR1
RN2

Mắt dò đường

Mạch nguyên lý



Mạch in



II. CODE Robot dò đường

```
#include <REGX51.H>
//========= Định nghĩa chiều quay bánh==========

#define go 1
#define back 0

//-----------------------------------------------------------------

#define   out    0xFF      // 0B11111111

//============= Định nghĩa mắt dò================

#define   R7     0x7F      // 0B01111111
#define   R6     0x3F      // 0B00111111
#define   R5     0xBF      // 0B10111111
#define   R4     0x9F      // 0B10011111
#define   R3     0xDF      // 0B11011111
#define   R2     0xAF      // 0B11001111
#define   R1     0xEF      // 0B11101111
#define   TT     0xE7      // 0B11100111
#define   L1     0xF7      // 0B11110111
#define   L2     0xF3      // 0B11110011
#define   L3     0xFB      // 0B11111011
#define   L4     0xF9      // 0B11111001
#define   L5     0xFD      // 0B11111101
#define   L6     0xFA      // 0B11111100
#define   L7     0xFE      // 0B11111110
//-----------------------------------------------------------------

// ========== Định nghĩa Pin xung PWM, Pin điều khiển=======
sbit xung=P2^0;
sbit dk1=P2^1;
sbit dk2=P2^3;
sbit xung2=P2^2;
//-------------------------------------------------------------------------------
```

```
// khai báo
unsigned char tocdo1,tocdo2 ;
unsigned char x=0,y=0;
unsigned long int counter0=0;
unsigned char INPUT;

// ===============Hàm tạo trễ===============
void delay(unsigned long int a)
 {
   unsigned long int i;
   for (i=0;i<=a;i++);
 }
//-------------------------------------------------------------

//========== Đếm đường ngắt 0=================
 void ngat0(void) interrupt 0
 {
 counter0++;
 }

//-------------------------------------------------------------

//================ Điều khiển bánh trái==================
void banhtrai (unsigned char tocdo0,unsigned char chieu)
 {
   if (chieu==go)
     {
            TR0=1;
            tocdo1=tocdo0;
            dk1=1;
          }
         if (chieu==back)
    {
            TR0=1;
            tocdo1=tocdo0;
            dk1=0;
          }
 }
```

```
//-------------------------------------------------------------------------------

// ================ Điều khiển bánh phải==================

 void banhphai (unsigned char tocdo0,unsigned char chieu)
 {
   if (chieu==go)
     {
            TR1=1;
            tocdo2=tocdo0;
            dk2=1;
           }
         if (chieu==back)
     {
            TR1=1;
            tocdo2=tocdo0;
            dk2=0;
           }
 }
//-------------------------------------------------------------------------------




//============ Tao Xung PWM bánh phải ==============
void ngatt1 (void) interrupt 3
 {

 y++;

 if( y==1)
 {        TR1=0;

          TL1=256-tocdo2;

           xung2=1;

            TR1=1;
          }
         if (y==2)
```

```
          {

            TR1=0;

            TL1=256-(100-tocdo2);
            xung2=0;

            TR1=1;
            y=0;
          }
  }
//---------------------------------------------------------------------

//============ Tao Xung PWM bánh trái ==============

void ngatt0 (void) interrupt 1
 {

   x++;
    if (x==1)
          {
            TR0=0;

            TL0=256-tocdo1;

           xung=1;

            TR0=1;
          }
         if (x==2)
          {

            TR0=0;

            TL0=256-(100-tocdo1);
           xung=0;

            TR0=1;
           x=0;
          }
```

```
  }
 //---------------------------------------------------------------------


//================ Hàm do đường ================
 void doduong(unsigned long int cm)
{
unsigned char nhotrai,nhophai;
unsigned char INPUT;
counter0=0;

while((100*cm)>=(counter0*18))

{

INPUT=P1; // mắt dò đượng nhận vào bằng PORT 1.

switch (INPUT)
{
 case  TT:
  {
  banhphai(50,go);
  banhtrai(50,go);
  nhophai=0;
  nhotrai=0;
  break;
  }
  case L1:
  {
  banhphai(55,go);
  banhtrai(45,go);
  nhophai=0;
  nhotrai=0;
   break;
  }
  case L2:
  {
  banhphai(60,go);
  banhtrai(40,go);
  nhophai=0;
```

```
nhotrai=0;
break;
}
case L3:
{
banhphai(65,go);
banhtrai(35,go);
nhophai=0;
nhotrai=0;
break;
}
case L4:
{
banhphai(70,go);
banhtrai(30,go);
nhophai=0;
nhotrai=0;
break;
}
case L5:
{
banhphai(75,go);
banhtrai(25,go);
nhophai=1;
nhotrai=0;
break;
}
case L6:
{
banhphai(80,go);
banhtrai(20,go);
nhophai=1;
nhotrai=0;
break;
}
case L7:
{
banhphai(85,go);
banhtrai(15,go);
nhophai=1;
```

```
 nhotrai=0;
  break;
 }
 case R1:
 {
 banhphai(45,go);
 banhtrai(55,go);
 nhophai=0;
 nhotrai=0;
 break;
 }
 case R2:
 {
 banhphai(40,go);
  banhtrai(60,go);
 nhophai=0;
 nhotrai=0;
 break;
 }
 case R3:
 {
 banhphai(35,go);
 banhtrai(65,go);
 nhophai=0;
 nhotrai=0;
 break;
 }
  case R4:
 {
 banhphai(30,go);
 banhtrai(70,go);
nhophai=0;
 nhotrai=0;
 break;
 }
 case R5:
 {
 banhphai(25,go);
 banhtrai(75,go);
 nhophai=0;
```

```
 nhotrai=1;
 break;
 }
 case R6:
 {
banhphai(20,go);
 banhtrai(80,go);
 nhophai=0;
 nhotrai=1;
 break;
 }
 case R7:
 {
 banhphai(15,go);
 banhtrai(85,go);
 nhophai=0;
 nhotrai=1;
 break;
 }
 case out:
 {
 if (nhotrai == 1)
  {
  banhphai(90,go);
  banhtrai(10,back);
  }
 if (nhophai == 1)
  {
  banhphai(10,back);
  banhtrai(90,go);
  }
break;
  }

 default :
  {

if(((INPUT!=R1)||(INPUT!=R2)||(INPUT!=R3)||(INPUT!=R4)||(INPUT!=R
5)||(INPUT!=R6)||(INPUT!=R7)||(INPUT!=L1)||(INPUT!=L2)||(INPUT!=L3
```

```
)||(INPUT!=L4)||(INPUT!=L5)||(INPUT!=L6)||(INPUT!=L7)||(INPUT!=out)
));
  {

  banhphai(50,go);
  banhtrai(50,go);
  }
  break;
  }
 }
 }
}

//---------------------------------------------------------------------------

//================ Hàm Stop===============
  void stop(void)
  {
  TR0=0;
  TR1=0;
 xung=0;
  dk1=0;
dk2=0;
 xung2=0;
}
//-----------------------------------------------------------------
// Hàm chính
void main (void)

  {

    TMOD=0x22;
         IE=0x8B;
         IT0=1;
         IP=0;
          while(1)
     {
           doduong(300); // dò đường 3m dừng.
           stop();
           while(1);
```

```
    }
}
```

- Bài sử dụng các tài nguyên của 89S52 như sau:
  - Timer 0,1 chế đỗ 8bit tạo xung dùng ngắt 1,3
  - Ngắt ngoài 0 để đếm đường bằng Encoder
  - PORT 1 để nhận tín hiệu từ mắt dò
  - Và Pin I/O điều khiển

III. Đồ án Robot dò đường dùng vi điều khiển 89S52

Thực hiên đồ án: Phạm Văn Lượng
Lê Xuân Định
Nguyễn Trần Cường

Lớp: CDT- K9

ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA CƠ KHÍ
BỘ MÔN CƠ ĐIỆN TỬ

š š & › ›

LUẬN VĂN TỐT NGHIỆP ĐẠI HỌC

ROBOT DI ĐỘNG THEO DẤU TƯỜNG

Sinh viên thực hiện: DOÃN MINH ĐĂNG
MSSV: P9900012
Cán bộ hướng dẫn : TS. NGUYỄN TẤN TIẾN

CHƯƠNG TRÌNH ĐÀO TẠO KỸ SƯ CHẤT LƯỢNG CAO
KHÓA 1: 1999 - 2004
TP.Hồ Chí Minh, 07/2004

Lời cảm ơn

Để thực hiện đề tài, tác giả đã nhận được rất nhiều sự chỉ dẫn, giúp đỡ và động viên quý báu của nhiều người, thiếu một trong các sự giúp đỡ đó cũng có thể làm cho đề tài không đạt kết quả như hiện nay.

Trước hết, em xin bày tỏ lòng cảm ơn sâu sắc đối với thầy TS. Nguyễn Tấn Tiến, người thầy hướng dẫn đã tận tình chỉ cho em phương pháp nghiên cứu khoa học, thầy cũng đã cung cấp cho em rất nhiều kiến thức chuyên sâu để thực hiện đề tài.

Em cũng vô cùng cảm ơn cô Th.S Trần Thị Ngọc Dung và các thầy cô ở chương trình đào tạo Kỹ sư chất lượng cao, thầy TS. Nguyễn Văn Giáp và các thầy cô ở bộ môn Cơ Điện Tử, Khoa Cơ khí, Trường Đại Học Bách Khoa Tp.HCM đã tham gia quá trình đào tạo và hướng dẫn em trong suốt thời gian học đại học, nhờ các thầy cô mà em có đủ kiến thức và lòng tự tin để thực hiện đề tài nghiên cứu này cũng như các đề tài trong tương lai.

Bên cạnh đó, sự hợp tác và giúp đỡ của bạn bè và các thế hệ đàn anh cũng giúp tôi rất nhiều trong việc thực hiện đề tài này. Em xin được cảm ơn KS. Lưu Tuấn Anh, Khoa Công nghệ Vật Liệu, người đã hướng dẫn em đi vào nghiên cứu vè robot, Th.S Trần Văn Tùng và các bạn ở Phòng thí nghiệm Thiết Kế Máy đã tích cực giúp đỡ em trong thời gian thực hiện đề tài. Tôi cũng xin chân thành cảm ơn các bạn cùng học lớp Cơ Điện Tử – Việt Pháp 99, đặc biệt là các bạn Đoàn Hiệp, Nguyễn Anh Kiệt, Phạm Huỳnh Phong, Nguyễn Minh Trung, những người cùng nghiên cứu về robot di động đã cho tôi các ý kiến đóng góp quý giá!

Con cũng xin cảm ơn gia đình đã luôn chăm sóc và quan tâm đến việc học của con, con vô cùng cảm ơn và luôn tự hào vì có Bố, Mẹ, Chị luôn động viên con trong quá trình học tập.

Và cuối cùng, tôi xin gửi lời cảm ơn tới những người đã tham gia giúp đỡ tôi trong quá trình thực hiện luận văn mà tôi chưa nêu tên ở đây, sự giúp đỡ của họ dù ít hay nhiều cũng đóng góp một phần vào kết quả thực hiện đề tài tốt nghiệp này.

Tp. Hồ Chí Minh, ngày 19 tháng 7 năm 2004

Doãn Minh Đăng

Mục lục

Lời cảm ơn .....
Mục lục i
Danh mục các hình vẽ iii
Danh mục các bảng iii
Tóm tắt đề tài iv
Abstract v
1 Tổng quan và đặt vấn đề 1
1.1 Giới thiệu chung về robot 1
1.2 Tổng quan về các bài toán của robot di động [5] 4
1.3 Bài toán di chuyển theo tường và các nghiên cứu liên quan 5
1.3.1 Giới thiệu bài toán 6
1.3.2 Mô hình toán học 6
1.3.3 Mục tiêu điều khiển 8
1.4 Phương pháp giải quyết vấn đề 8
2 Tóm tắt thuật toán điều khiển 9
2.1 Mô hình bộ điều khiển 9
2.2 Đặc tính bộ điều khiển (theo kết quả chứng minh và mô phỏng) 9
3 Thiết kế và thực hiện phần cứng 10
3.1 Kiến trúc robot 10
3.2 Vi điều khiển PIC 16F877[13] 11
3.3 Thiết kế khung giao tiếp I2C 13
3.3.1 Lý do sử dụng giao tiếp I2C 13
3.3.2 Khung giao tiếp I2C trong robot 13
3.4 Thiết kế đế di chuyển và bộ điều khiển động cơ 14
3.4.1 Thiết kế đế di chuyển 14
3.4.2 Bộ điều khiển PID [15] 14
3.5 Thiết kế cảm biến 16
3.5.1 Mô hình toán học của cảm biến 16
3.5.2 Thực hiện cảm biến 17
3.6 Thiết kế các mạch điện tử 19
3.6.1 Mạch module master 19
3.6.2 Mạch module slave 20
4 Thực hiện bộ điều khiển và kiểm chứng giải thuật 22
4.1 Sơ đồ giải thuật chương trình 22
4.1.1 Giải thuật cho master module 23
4.1.2 Giải thuật cho slave module 24
4.2 Tiến hành thí nghiệm 25
4.3 So sánh các kết quả mô phỏng và thí nghiệm 26
4.3.1 So sánh kết quả mô phỏng bằng Matlab với kết quả thí nghiệm 26
4.3.2 Các nhận xét bổ sung 28
5 Kết luận 33

5.1 Độ thích hợp của giải thuật..... 33
5.2 Những hạn chế của đề tài..... 33
5.2.1 Về việc chế tạo phần cứng..... 33
5.2.2 Những hiện tượng ảnh hưởng đến kết quả và cách khắc phục..... 33
5.3 Hướng nghiên cứu tiếp..... 34
TÀI LIỆU THAM KHẢO.....35
PHỤ LỤC A.....37
PHỤ LỤC B.....39

Danh mục các hình vẽ

Hình 1.1 Một số hình ảnh về robot và các ứng dụng 4
Hình 1.2 Mô hình bài toán robot di động bám tường 7
Hình 3.1 Sơ đồ khối của all-following mobile robot 11
Hình 3.2 Sơ đồ chân PIC 16F877 12
Hình 3.3 Mô hình đế di chuyển lật ngược 14
Hình 3.4 Bộ điều khiển PID vận tốc theo mô hình song song 14
Hình 3.5 Đáp ứng của bộ điều khiển PID với kp=8, ki=1, kd=1 15
Hình 3.6 Đáp ứng của bộ điều khiển PID với kp=8.2, ki=1, kd=0.8 16
Hình 3.7 Mô hình toán học của cảm biến 17
Hình 3.8 Phần đệm tín hiệu từ encoder vào vi điều khiển ở module master 18
Hình 3.9 Hình chụp module cảm biến 18
Hình 3.10 Sơ đồ nguyên lý của mạch module master 19
Hình 3.11 Hình chụp module master 20
Hình 3.12 Sơ đồ nguyên lý khối xử lý chính của module slave 20
Hình 3.13 Sơ đồ nguyên lý khối khuếch đại công suất của module slave 21
Hình 3.14 Hình chụp module slave 21
Hình 4.1 Lưu đồ giải thuật của master module 23
Hình 4.2 Lưu đồ giải thuật của slave module 24
Hình 4.3 Mô hình thí nghiệm 26
Hình 4.4 So sánh đồ thị của vận tốc robot 27
Hình 4.5 So sánh đồ thị của sai số khoảng cách 27
Hình 4.6 So sánh đồ thị của sai số góc 28
Hình 4.7 Giá trị của cảm biến 29
Hình 4.8 Giá trị vận tốc góc của robot và vận tốc góc (ước lượng) của tường 29
Hình 4.9 Biến đổi của các sai lệch trong quá trình hoạt động 30
Hình 4.10 Giá trị vận tốc ra lệnh cho 2 bánh xe 30
Hình 4.11 So sánh các đồ thị e_1 và e_2 của hai thí nghiệm 32

Danh mục các bảng

Bảng 1.1 Tóm tắt lịch sử phát triển của công nghệ robot 2
Bảng 4.1 Thông số thí nghiệm 26
Bảng 4.2 Thông số của 2 thí nghiệm (TN) dùng để so sánh 31
Bảng 5.1 Các hiện tượng ảnh hưởng đến kết quả và cách khắc phục 33

Tóm tắt đề tài

Trong thời đại công nghiệp ngày nay, Robot ngày càng được sử dụng phổ biến trong sản xuất cũng như trong cuộc sống của con người. Robot đã có một vị trí quan trọng khó có thể thay thế được, nó giúp con người để làm việc trong các điều kiện nguy hiểm, khó khăn. Ngoài ra, Robot còn được dùng vào các lĩnh vực thám hiểm không gian, quân sự, giải trí... Lĩnh vực Robot di động đang ngày càng chiếm được sự quan tâm của các nhà nghiên cứu và xã hội. Từ tình hình thực tế đó, việc xây dựng các chương trình hoạt động cho các Robot là điều thiết yếu đặc biệt đối với các Robot di động. Bài toán Robot di động bám tường (wall-following problem) là một trong các bài toán thường gặp của Robot kiểu phản xạ (reactive paradigm), nó đã được giải bằng nhiều cách khác nhau. Trong đề tài "Robot di động theo dấu tường", bài toán Robot di động bám tường được giải quyết bằng một bộ điều khiển hồi tiếp đầy đủ trạng thái mà kết quả đã được chứng minh bằng mô phỏng. Một cảm biến tiếp xúc dùng các encoder được tạo ra để sử dụng cho robot. Mô hình robot được chế tạo để tiến hành thí nghiệm nhằm kiểm chứng giải thuật của bộ điều khiển. Kết quả thí nghiệm là căn cứ để phát triển bộ điều khiển dành cho bài toán wall-following trong các Robot sau này.

Abstract

This project studies on control of a wall-following mobile robot. The wall is assumed unknown. A tactile sensor is constructed to measure the angle and the distance of mobile robot relatively to the wall that the mobile robot must follow. A nonlinear controller is built based on Lyapunov stability. The experiment has been carried out to verify the study. Based on this result, the proposed controller can be used for control of a wall-following mobile robot problem.

1
TỔNG QUAN VÀ ĐẶT VẤN ĐỀ

1.1 Giới thiệu chung về robot

Khái niệm Robot theo nghĩa chung thường được hiểu đồng nghĩa với khái niệm tự động hoá công nghiệp, điều này chỉ đúng một phần bởi vì: thứ nhất, Robot chỉ là một thành phần trong hệ thống tự động hoá, thứ hai là tự thân việc trình bày, miêu tả Robot trong sinh hoạt xã hội ít nhiều phóng đại.

Những Robot xuất hiện lần đầu tiên ở NewYork vào ngày 9/10/1922 trong vở kịch"Rossum's Universal Robot" của nhà soạn kịch người Tiệp Khắc là Karen Chapek, còn từ Robot là một cách gọi khác của từ Robota-theo tiếng Tiệp có nghĩa là công việc lao dịch. Khi đó, Karen Chapek cho rằng Robot là những người máy có khả năng làm việc nhưng không có khả năng suy nghĩ.

Gần một thế kỷ tiếp theo, khái niệm robot đã liên tục được phát triển, đóng góp thêm bởi nhiều nhà nghiên cứu, nhiều công ty chuyên về lĩnh vực robot. Dưới đây là bảng tóm tắt quá trình lịch sử hình thành và phát triển của công nghệ chế tạo robot, và những tác động của khoa học cũng như xã hội đối với từng thời kỳ [4].

Bảng 1.1 Tóm tắt lịch sử phát triển của công nghệ robot

Mốc thời gian	Nghiên cứu và phát triển	Ứng dụng trong công nghiệp	Kỹ thuật hỗ trợ	Các yếu tố ảnh hưởng
1920	Khái niệm robot xuất hiện trong tiểu thuyết			
1940	Phát minh ra cánh tay máy			
1950	Phát sinh khái niệm robot thông minh		Giới thiệu về bộ nhớ vòng	
1960	Giới thiệu về robot được điều khiển bằng máy tính Tăng cường nghiên cứu	Phát triển robot trong công nghiệp Ứng dụng ở NASA và NAVY	Máy tính dùng transitor Giới thiệu vi xử lý	
1970	Robot có trí thông minh nhân tạo	Sự bộc phát lần đầu tiên về robot	Phát triển vi xử lý	Sự hạn chế của nền kinh tế
1980	Robot dùng trong những công việc nguy hiểm (1983)	Robot công nghiệp thực tế và các ứng dụng rộng rãi khác	Kỹ thuật số Kỹ thuật quang	Nhu cầu tăng cường tự động
1990		Giới thiệu về robot thông minh trong sản xuất	Điều khiển logic Nghiên cứu về robot trí thông minh nhân tạo	Robot gây nên thất nghiệp
2000		Robot giống con người	Các tiến bộ về cơ khí	

Trước những năm 1970, người ta chỉ tập trung vào việc phát triển những robot tay máy hoạt động trong các nhà máy công nghiệp. Sau đó mới xuất hiện những khái niệm về robot thông minh, và các nghiên cứu bắt đầu tập trung hơn vào robot di động. Một trong những chuyên gia đầu ngành về robot di động là Hans P. Moravec (bắt đầu nghiên cứu từ năm 1964), và hiện nay, chuyên nghiên cứu về robot di động là Sebastien Thruns.

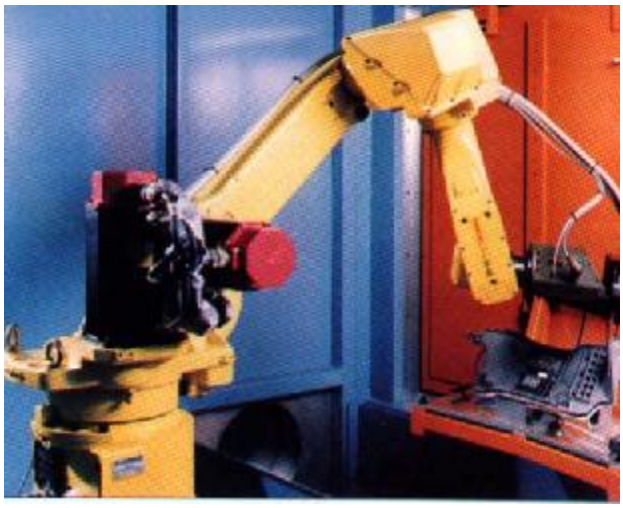
Các robot di động có người điều khiển đã được dùng cho các mục đích quân sự, các nhiệm vụ nguy hiểm như phá mìn, thăm dò đáy đại dương, hầm mỏ, kiểm tra các đường ống ngầm, hay thăm dò sao Hoả...

Sản phẩm robot di động được sản xuất đại trà và đưa vào thị trường lần đầu tiên là robot hút bụi Roomba và Trilobite của hãng Electrolux năm 2003.

Ở hình 1.1 là một số hình ảnh về các Robot và ứng dụng của nó:

a. Robot tự hành Sojourner thám hiểm sao Hoả

b. Robot dò mìn



c. Tay máy dùng trong công nghiệp

d. Robot hút bụi Trilobite

Hình 1.1 Một số hình ảnh về robot và các ứng dụng

1.2 Tổng quan về các bài toán của robot di động [5]

Vấn đề "**navigation**" (tạm dịch là "di chuyển") là vấn đề trọng tâm của robot di động. Để di chuyển được, robot phải thực hiện một loạt các tác vụ, mỗi tác vụ gắn với một bài toán nhỏ trong bài toán "navigation". Các bài toán đó gồm:

Mapping: là công việc lập bản đồ môi trường hoạt động của robot. Nếu không được cung cấp dữ liệu trước thì robot phải có khả năng lập bản đồ.

Positioning: là việc định vị, robot phải có khả năng biết được mình đang ở đâu trong bản đồ toàn cục hoặc địa phương.

Path planning: là việc hoạch định đường đi sắp tới của robot, sau khi nó biết được bản đồ và biết mình đang ở vị trí nào.

Motion control: là việc điều khiển cho robot di động, tức là điều khiển các cơ cấu để robot đi theo con đường thu được từ bài toán "path planning".

Obstacle avoidance: là nhiệm vụ tránh chướng ngại vật khi robot đang di chuyển.

1.3 Bài toán di chuyển theo tường và các nghiên cứu liên quan

Bài toán di chuyển theo tường:

Việc di chuyển theo tường (wall following) là một tác vụ thường thấy ở robot di động, trong các môi trường biết trước hoặc không biết trước. Tác vụ này được dùng với các nhiệm vụ: tránh chướng ngại vật, đi theo tường biết trước, đi theo tường không biết trước.

Một số nghiên cứu đã thực hiện:

Turennout et al., 1992 [6]: một bộ điều khiển hồi tiếp dùng bộ quan sát để ước lượng khoảng cách và góc giữa robot với tường, dùng cảm biến siêu âm.

Medromi et al., 1994 [7]: dùng một bộ quan sát để ước lượng trạng thái của hệ phi tuyến với đầu vào không biết trước.

Urzelai, J. et al., 1997 [8]: sử dụng bộ điều khiển mờ để điều khiển robot VEA-II dùng cảm biến siêu âm.

Bemporad et al., 1997 [9]: đưa ra hướng tiếp cận dùng sự chồng chất cảm biến để ước lượng toạ độ của robot, trong đó dùng một bộ lọc Kalman để kết hợp tín hiệu từ cảm biến siêu âm và cảm biến độ dịch chuyển.

Yata et al., 1998 [10]: đưa ra phương pháp bám tường sử dụng việc đo góc nhờ cảm biến siêu âm.

Chung Tan Lam et al, 2004 [11]: một bộ điều khiển hồi tiếp phi tuyến khi biết khoảng cách và góc giữa robot với tường, một bộ điều khiển dựa trên bộ quan sát khi chỉ biết khoảng cách giữa robot với tường, dùng cảm biến cơ. Các bộ điều khiển ổn định theo tiêu chuẩn Lyapunov, các kết quả mô phỏng và thực nghiệm đã chứng tỏ hiệu quả của các bộ điều khiển.

Nguyễn Viết Hiệp và Phạm Đình Anh Vũ [3]: thiết kế bộ điều khiển hồi tiếp phi tuyến và bộ điều khiển dựa trên bộ quan sát để điều khiển robot đi theo tường, mô phỏng các bộ điều khiển để kiểm chứng tính hội tụ và ổn định.

Các kết quả nghiên cứu trên cho thấy hướng giải quyết bài toán robot di chuyển theo tường bằng các bộ điều khiển hồi tiếp phi tuyến là một hướng đi thích hợp. Ở đề tài của Nguyễn Viết Hiệp và Phạm Đình Anh Vũ, các bộ điều khiển đã được đưa ra và chứng minh bằng lý thuyết và mô phỏng, nhưng chưa được kiểm nghiệm thực tế. Dựa trên thành quả đó, luận văn này có nhiệm vụ là: **Kiểm chứng lý thuyết nghiên cứu về bài toán robot di chuyển theo tường đã được xây dựng trong luận văn tốt nghiệp đi trước.**

Để thực hiện mục tiêu trên, luận văn này tập trung vào các vấn đề sau:

- Nghiên cứu các bộ điều khiển cho robot bám tường.

- Thiết kế và chế tạo một đế di chuyển (mobile platform).

- Thiết kế và chế tạo một cảm biến tiếp xúc gắn lên robot để do sai số giữa robot với tường.

- Thiết kế và thực hiện các mạch điều khiển cho robot.

- Lập trình cho robot để hiện thực các bộ điều khiển.

- Thí nghiệm và so sánh kết quả thí nghiệm với kết quả mô phỏng.

- Nhận xét kết quả và kết luận.

1.3.1 Giới thiệu bài toán

Thiết kế và thực hiện bộ điều khiển dùng để điều khiển Mobile Robot di chuyển dọc theo tường với vận tốc và khoảng cách từ Robot đến tường cho trước.

Giả thiết của bài toán:

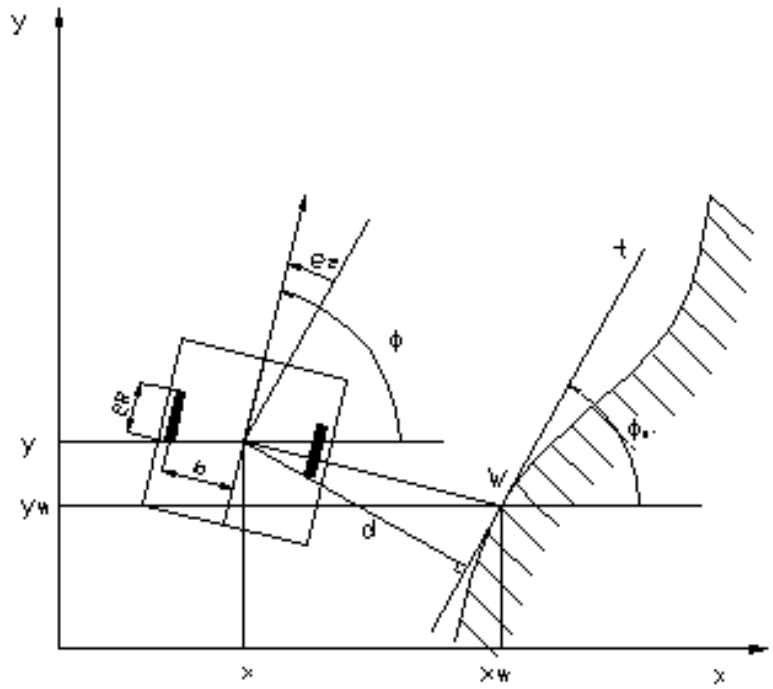
* Đo được khoảng cách d từ tường đến Robot.

* Đo được góc lệch giữa Robot và tường.

* Tường là đường cong trơn bất kỳ với bán kính cong của tường không nhỏ hơn khoảng cách d.

1.3.2 Mô hình toán học

Mô hình robot di động bám tường được cho như hình 1.2



Hình 1.2 Mô hình bài toán robot di động bám tường

Trong đó:

* (x,y): hệ trục toạ độ tuyệt đối.

* R: bán kính bánh xe của Mobile Robot.

* b: khoảng cách từ tâm Mobile Robot đến bánh xe.

* W:giao điểm của trục Y của Mobile Robot với tường.

* t: tiếp tuyến với tường tại điểm W.

* f : góc định hướng của Mobile Robot.

* f_w : góc nghiêng của tường.

* d :khoảng cách từ Mobile Robot đến tiếp tuyến t.

* d0 :khoảng cách yêu cầu từ Mobile Robot đến tường.

* e_2: góc lệch giữa mobile robot với tường.

Phương trình động học của Mobile robot đã được nghiên cứu bởi các công trình trước đây của nhiều tác giả [6]. Vận tốc được biểu diễn như sau:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{f} \end{bmatrix} = \begin{bmatrix} \cos f & 0 \\ \sin f & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ w \end{bmatrix} \quad \textbf{[1-1]}$$

Với v: vận tốc dài của Robot.

w : vận tốc góc của Robot.

Mối quan hệ giữa v , w với vận tốc góc của hai bánh xe như sau:

$$\begin{bmatrix} w_{rw} \\ w_{lw} \end{bmatrix} = \begin{bmatrix} 1/R & b/R \\ 1/R & -b/R \end{bmatrix} \begin{bmatrix} v \\ w \end{bmatrix} \quad \textbf{[1-2]}$$

với w_{lw} , w_{rw} : vận tốc góc của bánh xe trái và bánh xe phải của Mobile Robot.

1.3.3 Mục tiêu điều khiển

Yêu cầu của bài toán là phải điều khiển robot chạy theo biên dạng song song với tường, sao cho các sai số khoảng cách (e_1) và (e_2) hội tụ về 0, robot hoạt động ổn định trong một vùng lân cận điểm hội tụ.

1.4 Phương pháp giải quyết vấn đề

Để giải quyết bài toán robot di động đi theo tường, hai bộ điều khiển đã được trình bày trong đề tài luận văn tốt nghiệp ngành Cơ Điện Tử, trường ĐHBK Tp.HCM của hai sinh viên Nguyễn Viết Hiệp và Phạm Đình Anh Vũ. Một bộ điều khiển là hồi tiếp đầy đủ trạng thái (full-state feedback), một bộ điều khiển là dựa trên bộ quan sát (observer-based) [2]. Trong đề tài này, chúng tôi sử dụng các kết quả đó để thực hiện các bộ điều khiển, nhằm kiểm tra tính chính xác của kết quả mô phỏng.

Theo giả thiết của bài toán, ta biết được giá trị của cả 2 biến trạng thái là e_1 và e_2, như vậy ta có thể sử dụng một bộ hồi tiếp tất cả biến trạng thái (full-state feedback controller). Bộ điều khiển này sẽ được đề cập đến ở chương sau.

2
TÓM TẮT THUẬT TOÁN ĐIỀU KHIỂN

2.1 Mô hình bộ điều khiển

Công thức của bộ điều khiển full-state feedback:

$$\begin{cases} v = v_r \\ w = k_2[v_r - (e_1 + d_0)\hat{w}_w / \cos e_2]e_1 - k_1 \sin e_2 + \hat{w}_w \\ \dot{\hat{w}}_w = e_1(e_1 + d_0)\tan e_2 - \sin e_2 / k_2 \end{cases} \quad \textbf{[2-1]}$$

Trong đó:

v_r: vận tốc yêu cầu

d_0: khoảng cách yêu cầu

$\hat{w}$: ước lượng vận tốc góc của tường

k_1, k_2: các tham số của bộ điều khiển để ổn định Lyapunov

e_1, e_2: sai số về khoảng cách và góc giữa robot với tường

v: vận tốc dài của robot

w: vận tốc góc của robot (vận tốc của vectơ chỉ hướng robot)

2.2 Đặc tính bộ điều khiển (theo kết quả chứng minh và mô phỏng)

Qua kết quả chứng minh và mô phỏng, bộ điều khiển này có các tính chất sau:

- Ổn định theo tiêu chuẩn Lyapunov dạng 2.

- Các sai số e1 và e2 hội tụ về zero sau thời gian 8-10s, với các tham số mô phỏng lấy theo mô hình thực.

3
THIẾT KẾ VÀ THỰC HIỆN PHẦN CỨNG

3.1 Kiến trúc robot

Robot được thiết kế theo kiểu kiến trúc SA (subsumption architect) [12], phân thành 2 lớp:

- Lớp dưới là các bộ điều khiển động cơ, bộ thu tín hiệu cảm biến.

- Lớp trên là bộ điều khiển trung tâm. Bộ điều khiển này không can thiệp vào hoạt động của các bộ phận ở lớp dưới.

Sơ đồ khối điều khiển được cho ở hình 3.1, gồm 01 khối xử lý chính (master module) và 02 khối xử lý phụ (slave module).

Master module có các chức năng:

- Đọc tín hiệu từ cảm biến, tính các sai số về khoảng cách và góc giữa robot với tường.

- Dùng giải thuật điều khiển full-state feedback để tìm các vận tốc dài và vận tốc góc mới cho robot, nhằm giảm các sai số.

- Chuyển vận tốc của robot thành vận tốc của 2 bánh xe, gửi các giá trị đó cho các slave module.

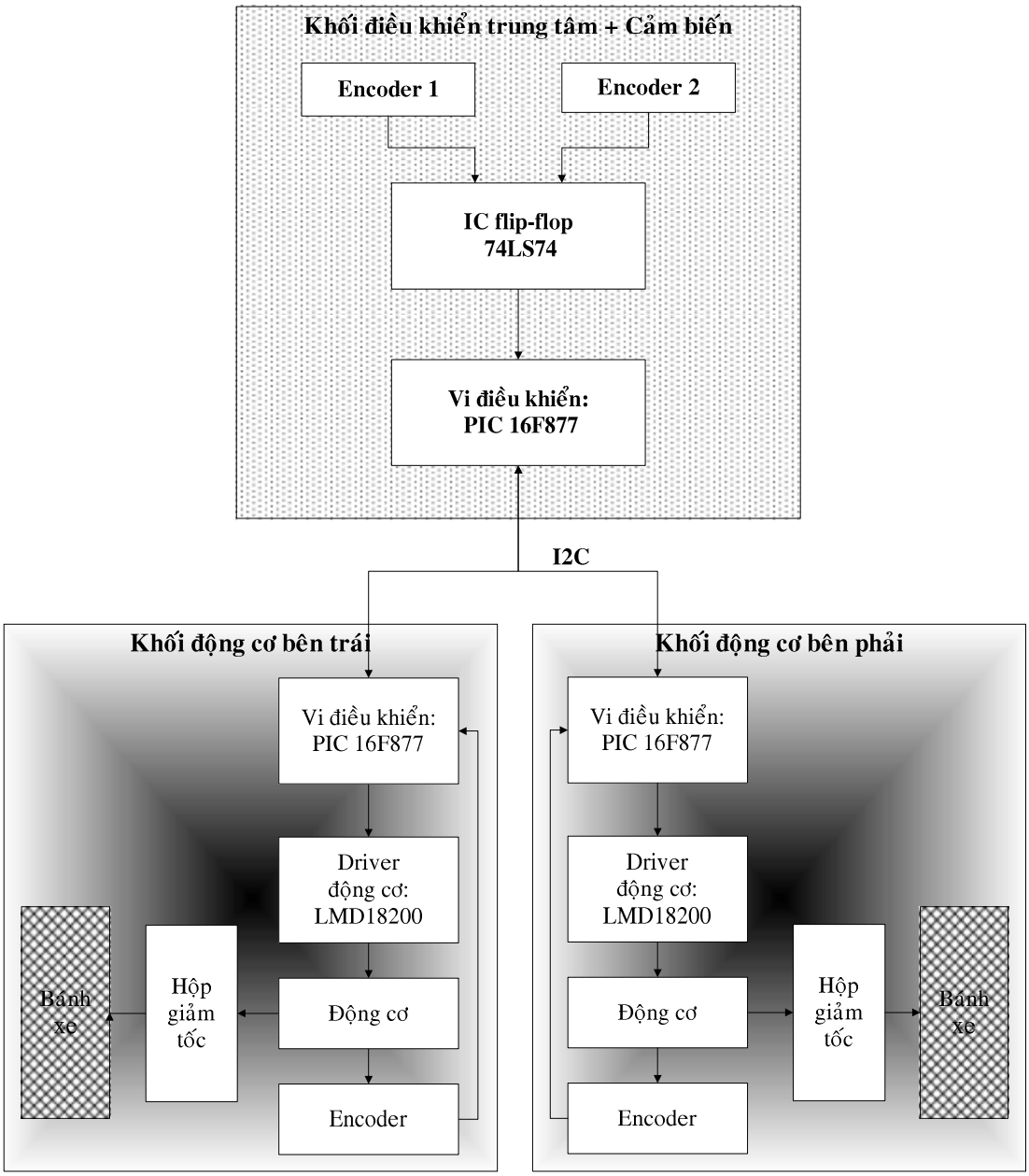
Slave module có các chức năng:

- Nhận lệnh từ master module (vận tốc mong muốn của mỗi bánh xe).

- Tính vận tốc hiện thời của robot qua số xung hồi tiếp trong 1 chu kỳ.

- Dùng giải thuật điều khiển PID để tính giá trị PWM, nhằm điều khiển bánh xe quay với vận tốc được ra lệnh.

Giao tiếp giữa master module và 2 slave module được thực hiện dựa trên chuẩn giao tiếp I2C, chuẩn này sẽ được giải thích rõ hơn ở đoạn dưới.



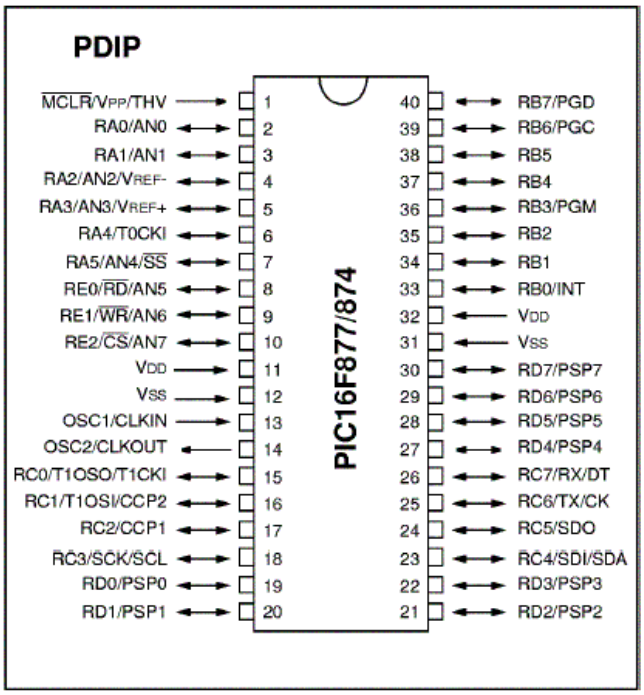
Hình 3.1 Sơ đồ khối của Wall-following mobile robot

3.2 Vi điều khiển PIC 16F877[13]

Vi điều khiển được chọn dùng trong đề tài là loại vi điều khiển PIC 16F877 của công ty Microchip. Sau đây là vài nét chính của vi điều khiển này:

Bộ xử lý chính:

- Loại bộ xử lý: RISC CPU
- Chỉ có tất cả 35 lệnh
- Hầu hết các lệnh là 1 chu kỳ máy, chỉ các lênh rẽ nhánh là 2 chu kỳ
- Tần số tối đa: với thạch anh 20 MHz – chu kỳ máy là 200ns
- Lưu được 8K lệnh trong bộ nhớ chương trình, mỗi lệnh 14 bits.
- Có 368 x 8 bytes bộ nhớ dữ liệu (RAM)
- Có 256 x 8 bytes bộ nhớ EEPROM
- Nhiều loại ngắt (14 loại)
- Power-on Reset (POR)
- Power-up Timer (PWRT) và
- Oscillator Start-up Timer (OST)
- Watchdog Timer (WDT) với bộ dao động tích hợp bên trong
- Bảo mật chương trình
- Có chế độ SLEEP để tiết kiệm năng lượng
- In-Circuit Serial Programming (ICSP - chuẩn ghi bộ nhớ chương trình khi vi xử lý vẫn ở trong mạch)
- Single 5V In-Circuit Serial Programming
- In-Circuit Debugging
- Vi xử lý truy cập được vào bộ nhớ chương trình
- Điện áp hoạt động rộng: 2.0V đến 5.5V
- Chịu được nhiệt độ trong môi trường công nghiệp
- Năng lượng tiêu thụ ít:
  < 0.6 mA ở 3V, 4 MHz
  20 mμA ở 3V, 32 kHz
  < 1 mμA ở chế độ standby



Hình 3.2 Sơ đồ chân PIC 16F877

Thiết bị ngoại vi:

- Timer0: 8-bit timer/counter với bộ chia trước 8-bit
- Timer1: 16-bit timer/counter với bộ chia trước, có thể hoạt động trong chế độ SLEEP
- Timer2: 8-bit timer/counter với bộ chia trước và chia sau 8-bit
- 2 bộ tích hợp Capture, Compare, PWM
  - Capture 16-bit, độ phân giải tối đa 12.5 ns
  - Compare is 16-bit, độ phân giải tối đa 200 ns
  - PWM có độ phân giải tối đa 10-bit
- 8 kênh biến đổi Analog-to-Digital 10-bit.
- Synchronous Serial Port (SSP) với 2 chuẩn: SPI (Master mode) và I2C (Master/Slave)
- Universal Synchronous Asynchronous Receiver Transmitter (USART/SCI)
- Parallel Slave Port (PSP) 8-bits, có các chân điều khiển RD, WR và CS
- Brown-out Reset (BOR)

3.3 Thiết kế khung giao tiếp I2C

3.3.1 Lý do sử dụng giao tiếp I2C

Vi điều khiển PIC 16F877 hỗ trợ nhiều chuẩn giao tiếp, trong đó có chuẩn giao tiếp I2C được sử dụng làm cho giao tiếp giữa các module trong robot này. Chúng tôi chọn sử dụng chuẩn I2C vì một số lý do sau:

- I2C hỗ trợ một mạng nhiều thiết bị kết nối với nhau. Số thiết bị tối đa có thể có trong mạng I2C gồm 1 module master và 127 module slave (hiện tại mới dùng 2 module slave). Như vậy robot còn nhiều khả năng mở rộng về sau.

- Chuẩn I2C là chuẩn thông dụng, được sử dụng nhiều trong các linh kiện dùng chế tạo robot di động (như cảm biến siêu âm SRF08, cảm biến la bàn CMPS03). Nếu sau này chế tạo một robot di động dựa trên cơ sở robot bám theo tường, ta cũng dễ dàng tích hợp các linh kiện khác vào robot.

- Môi trường lập trình đang dùng hỗ trợ tốt cho việc giao tiếp bằng I2C. Chúng tôi đã thử nghiệm và thấy chương trình giao tiếp hoạt động ổn định.

Để hiểu thêm về chuẩn giao tiếp I2C, xin xem phụ lục A.

3.3.2 Khung giao tiếp I2C trong robot

Cách thức truyền tín hiệu qua I2C trong robot được quy định như sau:

- Module master chỉ ghi giá trị vào các module slave, không có chế độ đọc.

- Địa chỉ của module slave điều khiển di chuyển bánh trái là 0xA0, của module điều khiển di chuyển bánh phải là 0xC0. Mỗi khi muốn gửi dữ liệu cho module slave nào, module master gửi 1 byte địa chỉ của slave đó, sau đó gửi tiếp 2 byte dữ liệu.

- Dữ liệu gửi từ module master đến module slave là một biến số nguyên 16 bit. Trước khi gửi nó phải được cắt thành 2 byte, sau khi module slave nhận thì ghép 2 byte đó trở lại thành số nguyên 16 bit ban đầu.

- Ở mỗi chu kỳ hoạt động, module master gửi cho mỗi module slave một dữ liệu, là vận tốc yêu cầu module slave đạt được. 2 dữ liệu được gửi liên tiếp để giảm độ trễ đáp ứng giữa 2 module slave.

3.4 Thiết kế đế di chuyển và bộ điều khiển động cơ

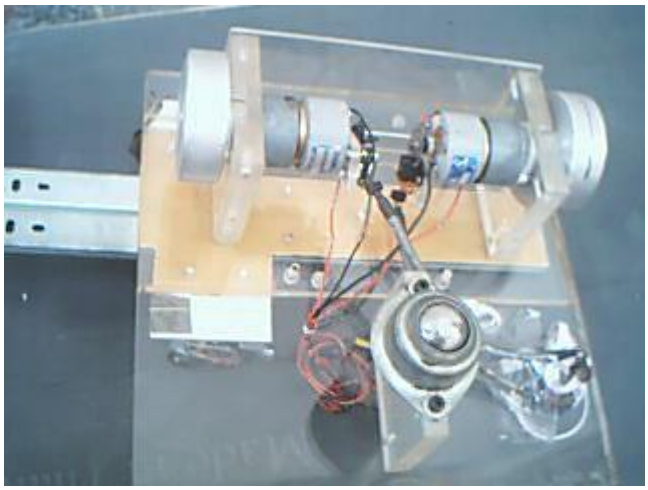
3.4.1 Thiết kế đế di chuyển

Các thành phần của đế di chuyển:

- 2 bánh dẫn động, gắn vào 2 động cơ có hộp giảm tốc, có hồi tiếp bằng encoder quang. Mỗi động cơ được điều khiển bởi 1 mạch điện riêng. Mạch điều khiển, động cơ và bánh xe cấu thành một module di chuyển, robot có 2 module di chuyển trái và phải tách rời nhau.

- Bánh tuỳ động là một bánh cầu, đặt ở sau xe.

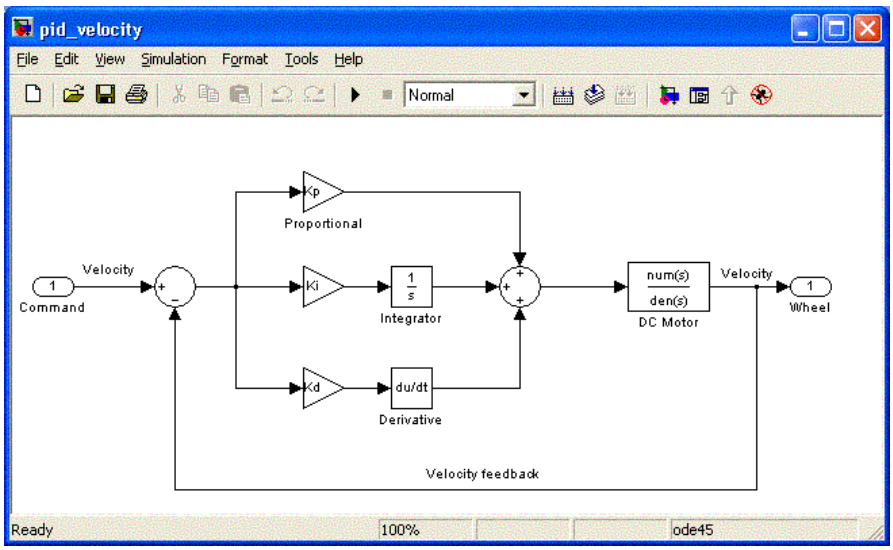
Bản vẽ thiết kế: xem bản vẽ mô hình robot (bản vẽ đính kèm).



Hình 3.3 Mô hình đế di chuyển lật ngược

3.4.2 Bộ điều khiển PID [15]

Mô hình điều khiển PID được sử dụng ở đây là:



Hình 3.4 Bộ điều khiển PID vận tốc theo mô hình song song

Bộ PID được sử dụng để đảm bảo vận tốc quay mà module điều khiển trung tâm ra lệnh cho module di chuyển thực hiện.

- Đầu vào của bộ PID: vận tốc yêu cầu, đơn vị là vòng/phút.

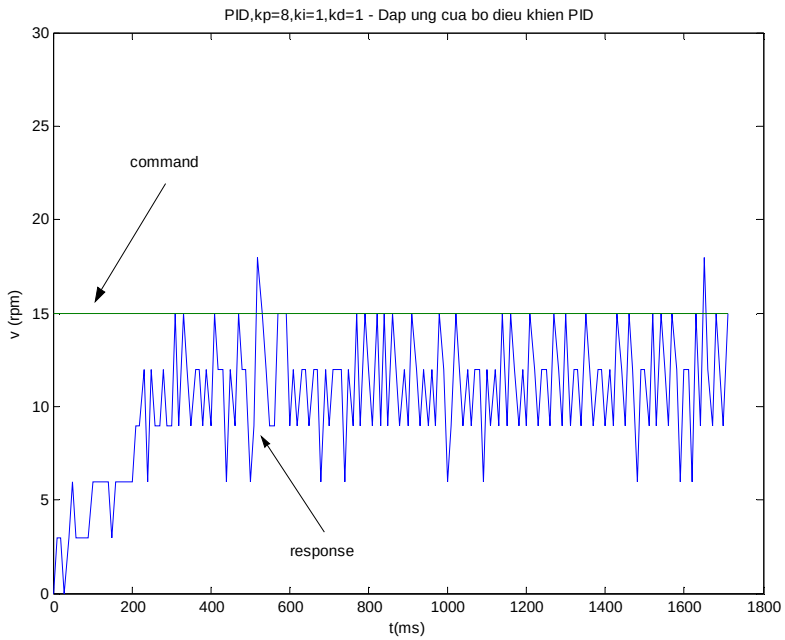
- Tín hiệu hồi tiếp: vận tốc hiện thời, từ số xung encoder đọc được trong 1 chu kỳ, đổi ra vòng/phút.

- Đối tượng điều khiển: vận tốc động cơ.

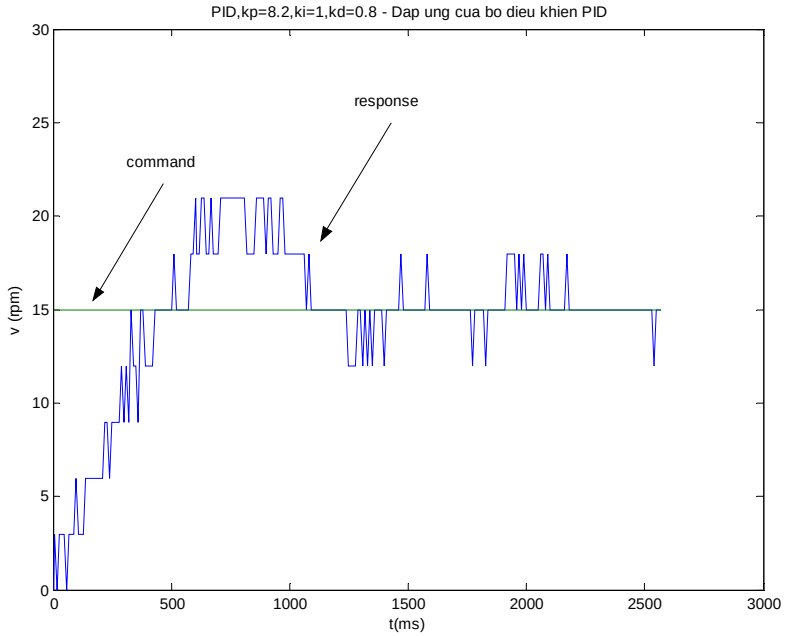
- Đầu ra của bộ PID: giá trị chu kỳ độ rộng xung (PWM duty) của điện áp hai đầu động cơ.

Phương pháp thực hiện bộ điều khiển PID vận tốc: do không có nhiều thời gian để tìm các thông số của động cơ nhằm mô hình hoá động cơ, chúng tôi lập trình bộ điều khiển động cơ trên vi điều khiển PIC với các giá trị kp, ki và kd thay đổi được. Khi hiệu chỉnh từ từ các tham số kp, ki và kd và xem đáp ứng của bộ điều khiển, chúng tôi lựa chọn được bộ tham số thích hợp cho bộ điều khiển.

Ta có thể thấy đáp ứng của bộ PID thay đổi theo sự thay đổi nhỏ của các hệ số điều khiển thông qua các đồ thị ở hình 3.5 và hình 3.6. Bộ PID được trình bày ở hình 4.6 được xem là tốt hơn bộ PID ở hình 3.5 (đáp ứng đạt mức yêu cầu, ít dao động), đó là bộ PID tốt nhất mà chúng tôi tìm được cho các module di chuyển của robot.



Hình 3.5 Đáp ứng của bộ điều khiển PID với kp=8, ki=1, kd=1



Hình 3.6 Đáp ứng của bộ điều khiển PID với kp=8.2, ki=1, kd=0.8

Các chỉ tiêu của bộ điều khiển PID:

- Thời gian đạt mức (rise time): 400ms

- Độ vọt lố (overshoot): 40% (tương ứng với 2 xung encoder/10ms)

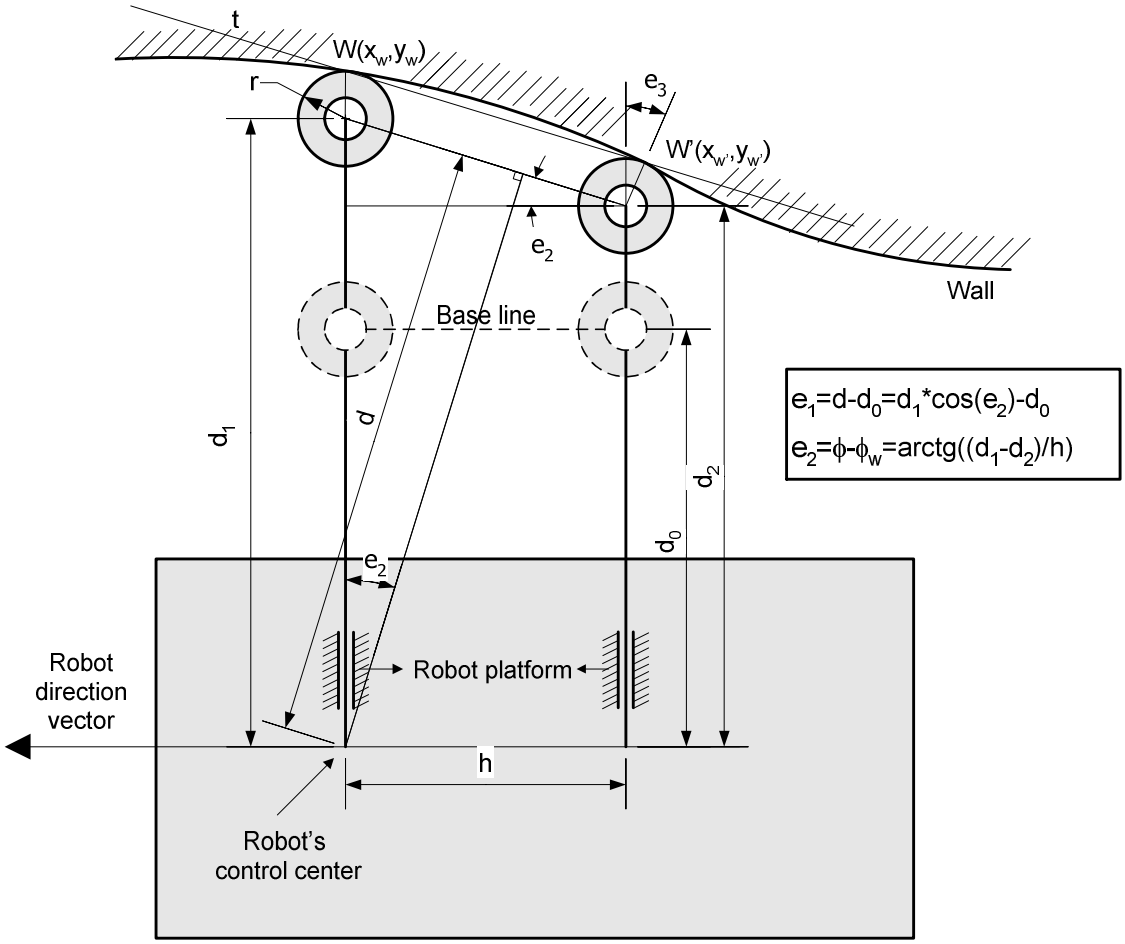
- Thời gian xác lập (settling time): 1000ms

Nhận xét: Do động cơ được sử dụng là một động cơ không tốt (công suất 4W, tốc độ tối đa khoảng 1000 vòng/phút, tỉ số hộp giảm tốc là 10), cộng với encoder có độ phân giải không cao (giá trị vọt lố 40% tương ứng với 2 xung encoder trong 1 chu kỳ lấy mẫu là 10ms), hơn nữa ta lại sử dụng động cơ ở tốc độ quay thấp, nên kết quả của bộ điều khiển PID không được tốt. Tuy nhiên, nếu không có các thành phần I và D thì bộ điều khiển P thông thường sẽ không thể đáp ứng được vận tốc mong muốn trong thời gian ngắn, đó là lý do phải sử dụng bộ điều khiển PID.

3.5 Thiết kế cảm biến

3.5.1 Mô hình toán học của cảm biến

Việc thiết kế một cảm biến tốt có ý nghĩa rất quan trọng trong robot di động này, để làm được điều đó, ta cần một mô hình toán học hợp lý. Một số mô hình toán học của cảm biến đã được nghiên cứu, cuối cùng chúng tôi đưa ra mô hình toán học ở hình 3.7 để thực hiện cảm biến cho robot.



Hình 3.7 Mô hình toán học của cảm biến

Cảm biến được set vị trí ban đầu ứng với e_2=0 và d=d_0. Khi robot chuyển động, 2 thanh trượt tiếp xúc với tường qua 2 con lăn tại các tiếp điểm W và W'. Do bán kính cong của tường lớn, ta xem tường trong đoạn WW' là thẳng. Hơn nữa, nhờ r nhỏ nên e_3 không đáng kể, ta xem e_3=0. Công thức tính e_1 và e_2 là:

$$e_1 = d - d_0 = d_1 * \cos(e_2) - d_0$$
$$e_2 = f - f_w = arctg(\frac{d_1 - d_2}{h}) \quad \textbf{[3-1]}$$

với d_1, d_2: độ dịch chuyển của thanh trượt 1 và 2

h: khoảng cách giữa 2 thanh trượt.

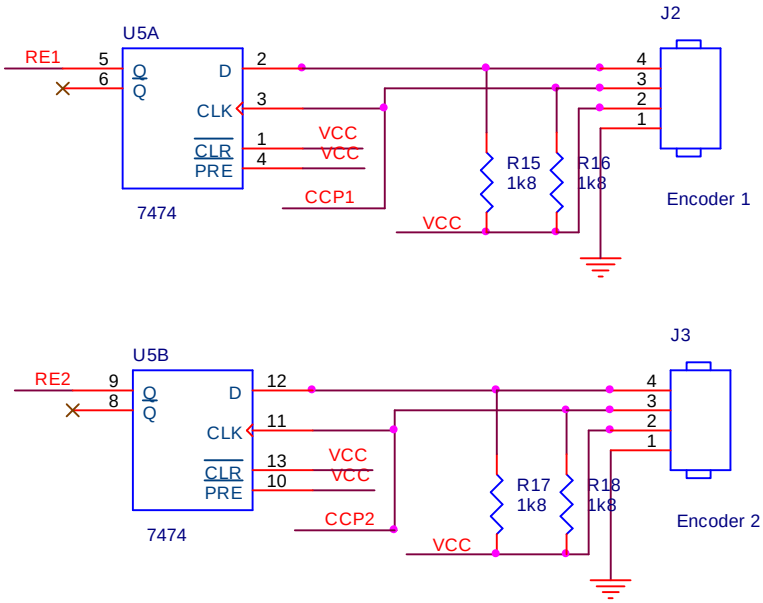
3.5.2 Thực hiện cảm biến

Module cảm biến bao gồm 3 bộ phận chính: 2 thanh trượt có con lăn ở đầu để tiếp xúc với tường, 2 encoder để dò độ dịch chuyển của các thanh trượt và 1 vi điều khiển để đọc độ dịch chuyển.

Thanh trượt được gắn lò xo (ở đây dùng dây thun) đẩy ra để luôn tiếp xúc với tường. Trên mỗi thanh trượt có gắn dây kéo dọc theo thanh, khi thanh trượt chuyển động, dây kéo sẽ kéo con lăn chuyển động. Do con lăn được nối chặt với trục

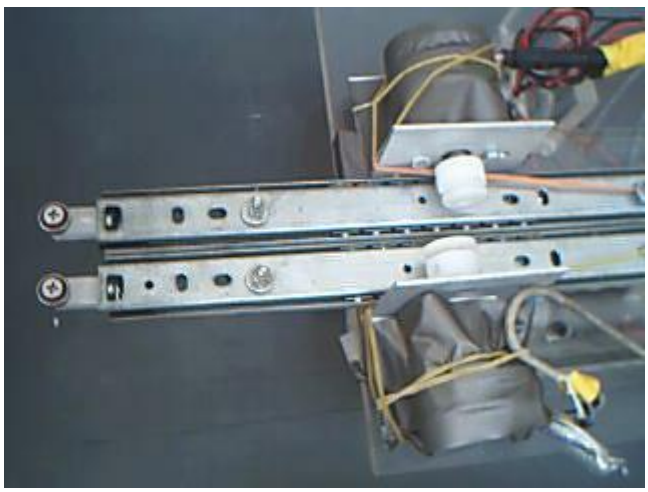
encoder nên chuyển động của con lăn sẽ làm quay đĩa của encoder, với mỗi dịch chuyển nhỏ của đĩa, encoder sẽ gửi xung về cho vi điều khiển để xử lý. Mỗi encoder truyền tín hiệu về cho vi điều khiển qua 2 đường A và B, chúng được cho vào một IC flip-flop để xác định chiều quay của encoder. Vi điều khiển nhận tín hiệu từ các encoder qua 2 cổng CCP (capture) và dùng 2 chân digital input (RE1 và RE2) để nhận biết chiều quay gửi từ IC flip-flop, các xung gửi từ encoder vào cổng CCP sẽ tạo ra ngắt (interrupt) để tiện việc tính toán trên vi điều khiển, tín hiệu ở các chân digital input sẽ cho vi điều khiển biết được xung đó ứng với chuyển động vào hay ra của thanh trượt.

Sơ đồ mạch thu tín hiệu từ encoder:



Hình 3.8 Phần đệm tín hiệu từ encoder vào vi điều khiển ở module master

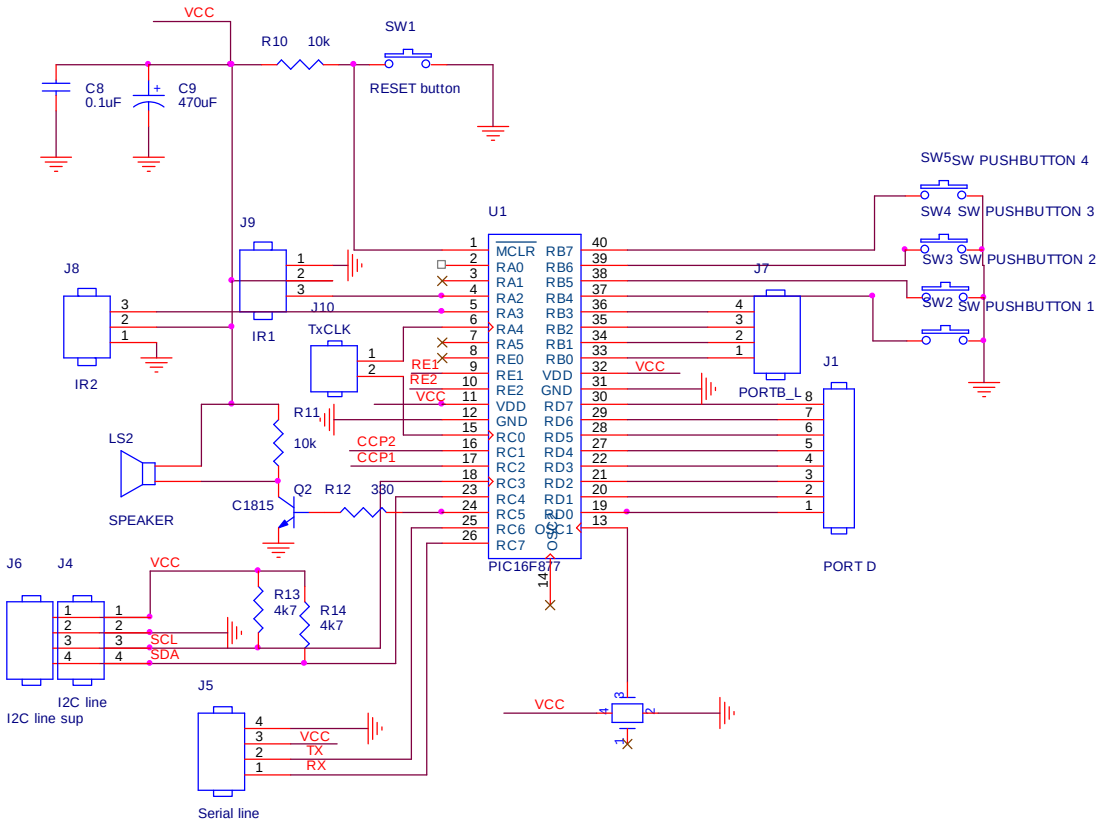
Kết quả thực hiện:



Hình 3.9 Hình chụp module cảm biến

3.6 Thiết kế các mạch điện tử

3.6.1 Mạch module master



Hình 3.10 Sơ đồ nguyên lý của mạch module master

Các thành phần và chức năng của chúng trong mạch master:

- Vi điều khiển PIC 16F877: bộ xử lý của cả mạch.

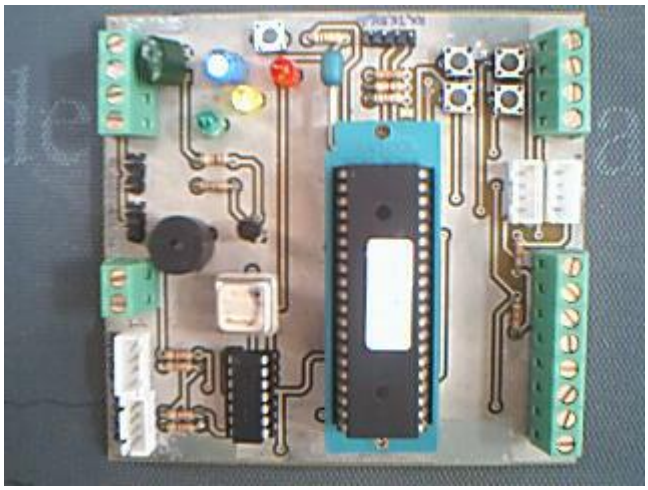
- Ngõ giao tiếp nối tiếp: để giao tiếp với mạch nạp và chương trình kiểm tra trên máy tính.

- Ngõ giao tiếp I2C: để giao tiếp với 2 vi điều khiển của các module slave.

- 2 ngõ nối với encoder để đọc tín hiệu từ cảm biến, có một IC flip-flop để đệm cho tín hiệu từ encoder.

- 4 nút bấm phục vụ việc nhận lệnh từ người sử dụng.

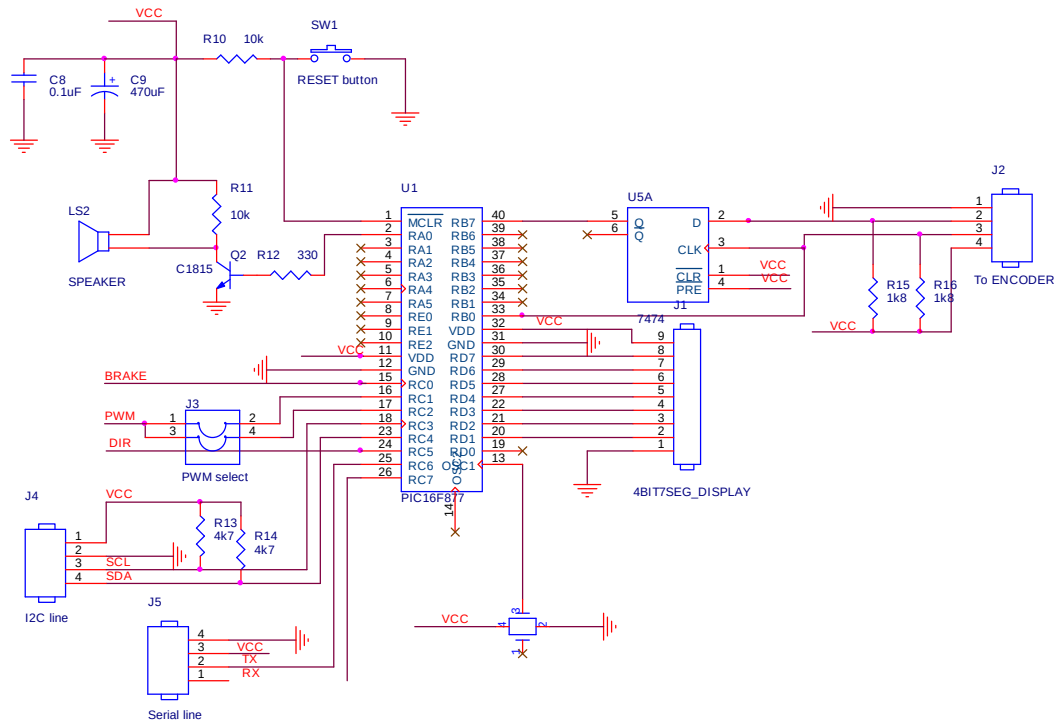
- 1 loa dành để báo hiệu các giai đoạn trong chương trình.



Hình 3.11 Hình chụp module master

3.6.2 Mạch module slave

Gồm 2 khối: khối xử lý chính và khối khuếch đại công suất.



Hình 3.12 Sơ đồ nguyên lý khối xử lý chính của module slave

Các thành phần trong khối xử lý chính:

- Vi điều khiển PIC 16F877: bộ xử lý của module.

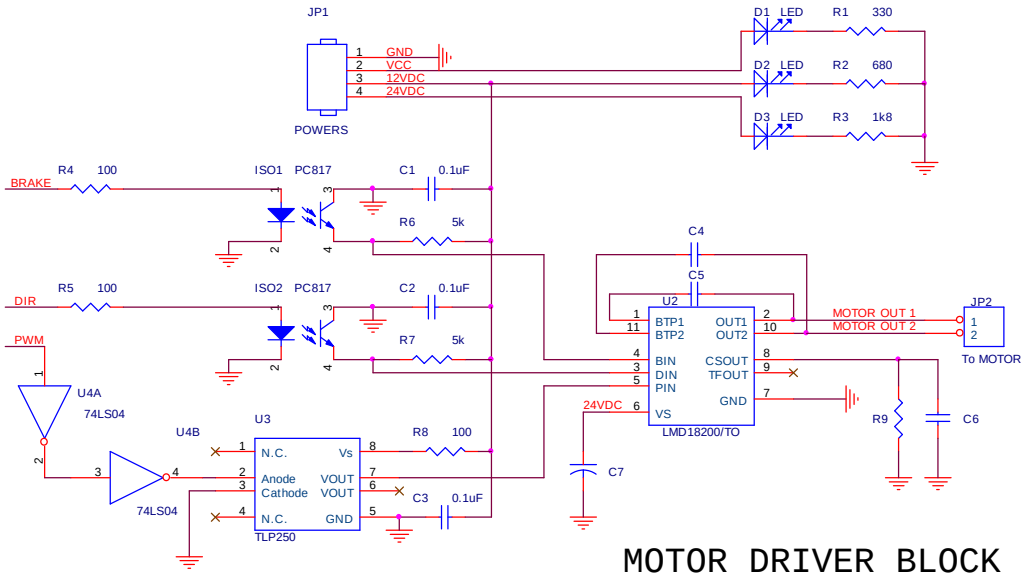
- Ngõ giao tiếp nối tiếp: để giao tiếp với mạch nạp và chương trình kiểm tra trên máy tính.

- Ngõ giao tiếp I2C: để giao tiếp với vi điều khiển của module master.

- Ngõ nối với encoder để đọc tín hiệu hồi tiếp, có một IC flip-flop để đệm cho tín hiệu từ encoder.

- 1 loa dành để báo hiệu.

Khối khuếch đại công suất:



Hình 3.13 Sơ đồ nguyên lý khối khuếch đại công suất của module slave

Các thành phần trong khối khuếch đại công suất:

- Chip điều khiển động cơ LMD18200.

- 3 opto để cách ly các ngõ vào của chip LMD18200, bảo vệ phần mạch phía trước.

- Các cổng logic để nắn tín hiệu cho ngõ vào PWM.

Hình 3.14 Hình chụp module slave

4
THỰC HIỆN BỘ ĐIỀU KHIỂN VÀ KIỂM CHỨNG GIẢI THUẬT

4.1 Sơ đồ giải thuật chương trình

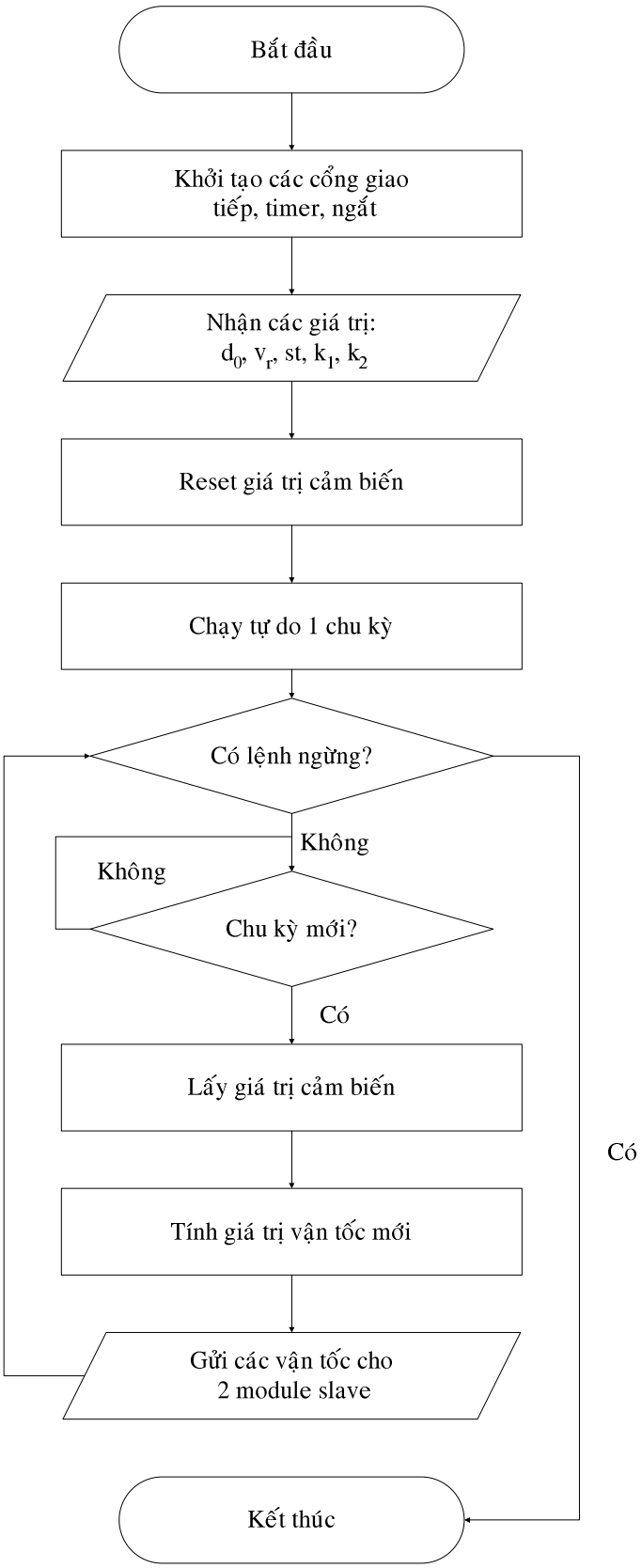
Giải thuật cho robot di động theo tường được thực hiện nhờ 2 chương trình, một chương trình dành cho master module (chương trình chính) và chương trình kia dành cho slave module (chương trình phụ).

Chương trình chính có nhiệm vụ là bộ điều khiển full-state feedback của toàn hệ thống, chương trình phụ có nhiệm vụ là bộ điều khiển PID vận tốc cho mỗi bánh xe. Chức năng cụ thể của từng chương trình được đề cập ở phần 3.1.

Các chương trình được lập trình bằng ngôn ngữ C, biên dịch cho các vi điều khiển PIC bằng trình biên dịch PIC-C, sau đó nạp vào các vi điều khiển (nạp chương trình chính vào vi điều khiển ở master module, nạp chương trình phụ vào 2 vi điều khiển ở 2 slave module).

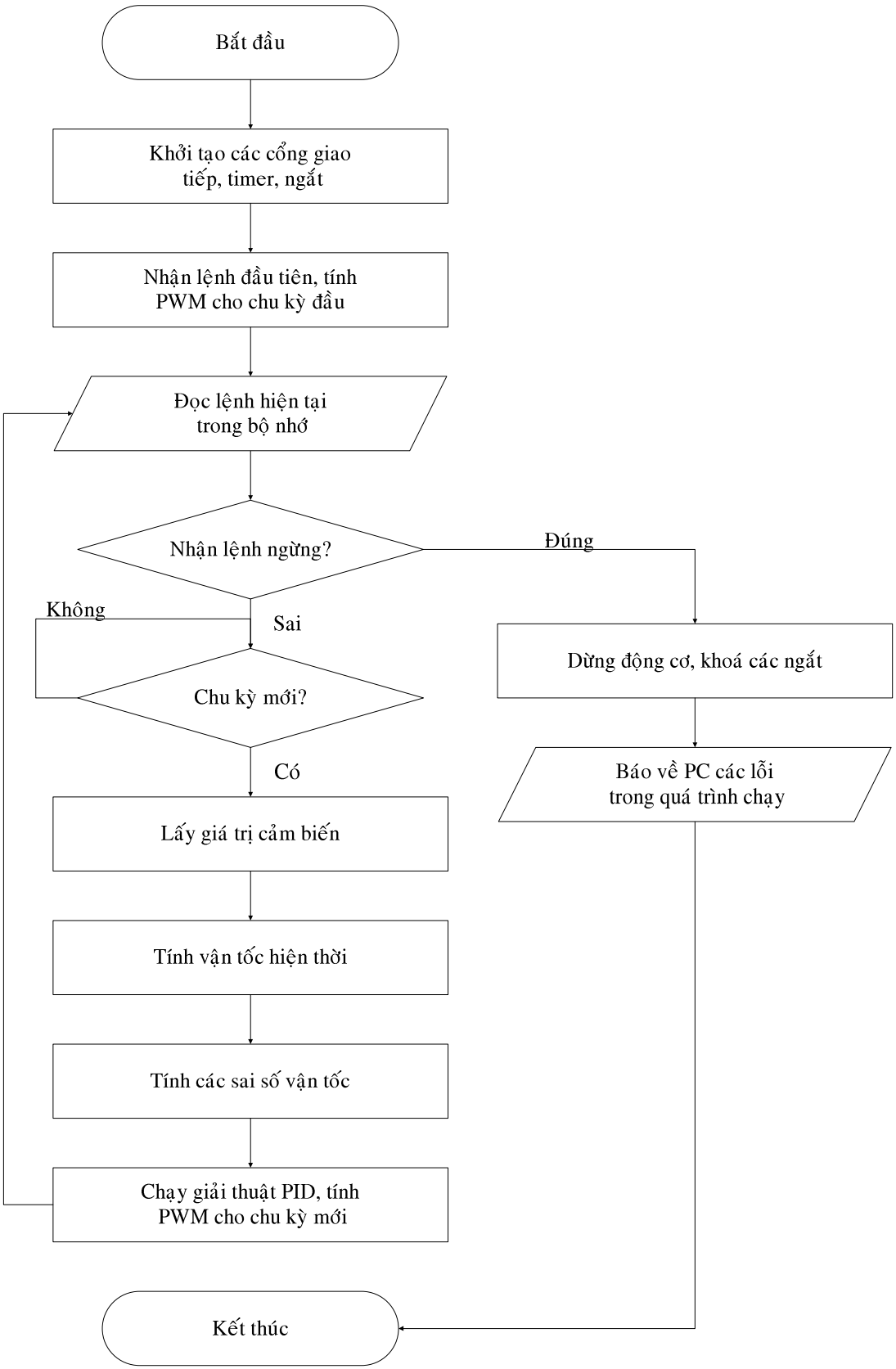
Mã nguồn của các chương trình: xem phụ lục B.

4.1.1 Giải thuật cho master module



Hình 4.1 Lưu đồ giải thuật của master module

4.1.2 Giải thuật cho slave module



Hình 4.2 Lưu đồ giải thuật của slave module

4.2 Tiến hành thí nghiệm

Các thí nghiệm được thực hiện như sau:

* Môi trường hoạt động: gồm tường và sàn nhà. Tường cần được lót một lớp đệm để có bề mặt êm và trơn. Bề mặt trơn để giảm lực ma sát tác động lên robot thông qua cảm biến tiếp xúc, bề mặt cũng cần êm để giảm rung động cho cảm biến, vì theo quan sát, sự rung động ở cảm biến sẽ gây ra sai số đọc encoder. Sàn nhà cũng được lót đệm để tăng hệ số ma sát giữa bánh xe với sàn, và cũng nhằm giảm rung động cho robot.

* Robot di động: robot được nuôi bằng nguồn điện ở ngoài. Thông qua các thiết bị biến thế, ổn áp, từ nguồn điện xoay chiều 220V/50Hz ta có được các nguồn một chiều (5V, 12V và 24V) để nuôi robot. Trên module master của robot có cắm dây nối với máy tính qua cổng giao tiếp nối tiếp, để robot truyền các kết quả đo được về máy tính.

* Máy tính: chúng tôi dùng một máy tính laptop có cổng giao tiếp nối tiếp (chuẩn RS-232) để vừa ra lệnh cho robot, vừa quan sát các trạng thái của robot. Máy tính có một chương trình dạng "terminal" để gửi lệnh và nhận dữ liệu qua cổng nối tiếp, dữ liệu nhận được sẽ được lưu thành file kết quả, file này sẽ được xử lý bằng phần mềm Matlab để thể hiện kết quả thí nghiệm qua các đồ thị.

Tiến trình thực hiện một thí nghiệm gồm các bước:

- Bật nguồn điện, reset các vi điều khiển trên robot, cài đặt thông số cho chương trình terminal trên máy tính. Cài đặt vị trí chuẩn của cảm biến, sau đó dịch chuyển robot để tạo độ lệch ban đầu cho cảm biến.

- Truyền các tham số cho module master của robot, các thông số được truyền theo thứ tự gồm: khoảng cách mong muốn d0, vận tốc mong muốn vr, thời gian lấy mẫu st, các tham số của bộ điều khiển k1 và k2.

- Đợi robot di chuyển đến hết đoạn đường cần thí nghiệm, ngừng robot và lưu kết quả thí nghiệm.

- Chạy phần mềm Matlab để xử lý kết quả thí nghiệm và nhận xét.

Do bị giới hạn về thời gian nghiên cứu, các thí nghiệm trên tường cong chưa được thực hiện. Chúng tôi chỉ xin trình bày ở đây các thí nghiệm robot di chuyển theo tường thẳng.

Hình 4.3 Mô hình thí nghiệm

4.3 So sánh các kết quả mô phỏng và thí nghiệm

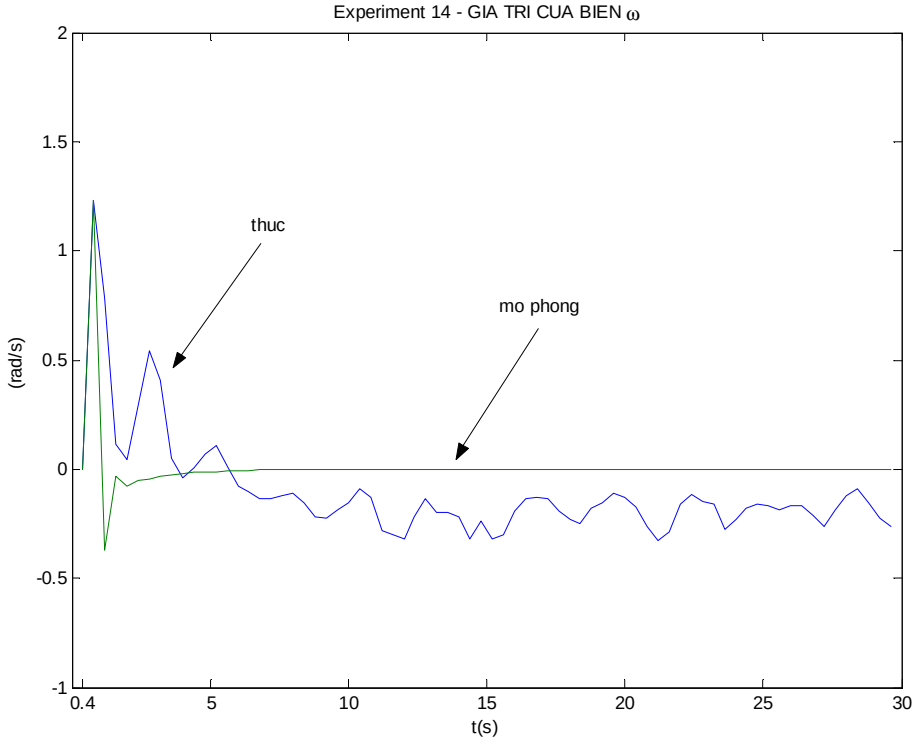
4.3.1 So sánh kết quả mô phỏng bằng Matlab với kết quả thí nghiệm

Chúng tôi tiến hành rất nhiều thí nghiệm với tường thẳng, sau đây là một số thí nghiệm cho kết quả điển hình và các nhận xét.

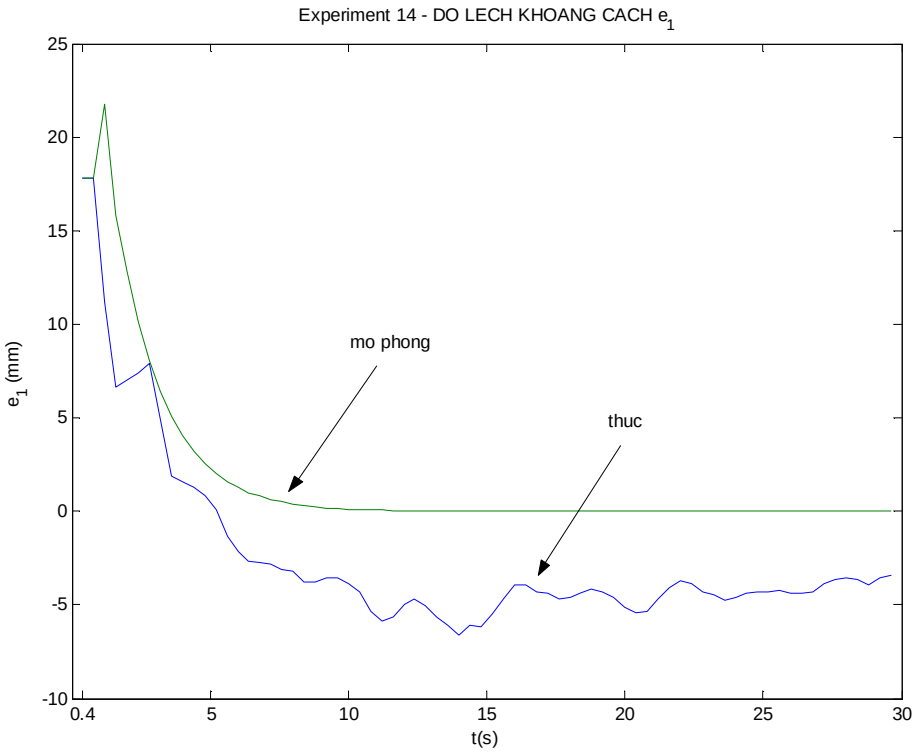
Bảng 4.1 Thông số thí nghiệm

Tham số	*Giá trị*	*Đơn vị*	*Ghi chú*
d_0	0.25	m	d_0 là giống nhau trong tất cả các thí nghiệm
v_r	0.05	m/s	
st	0.4	s	st khá lớn, do bộ điều khiển động cơ đáp ứng chậm
k_1	3.5		
k_2	615		
e_1 ban đầu	0.0178	m	Tính ngược từ giá trị hai encoder ở cảm biến
e_2 ban đầu	-0.1955	rad	Tính ngược từ giá trị hai encoder ở cảm biến

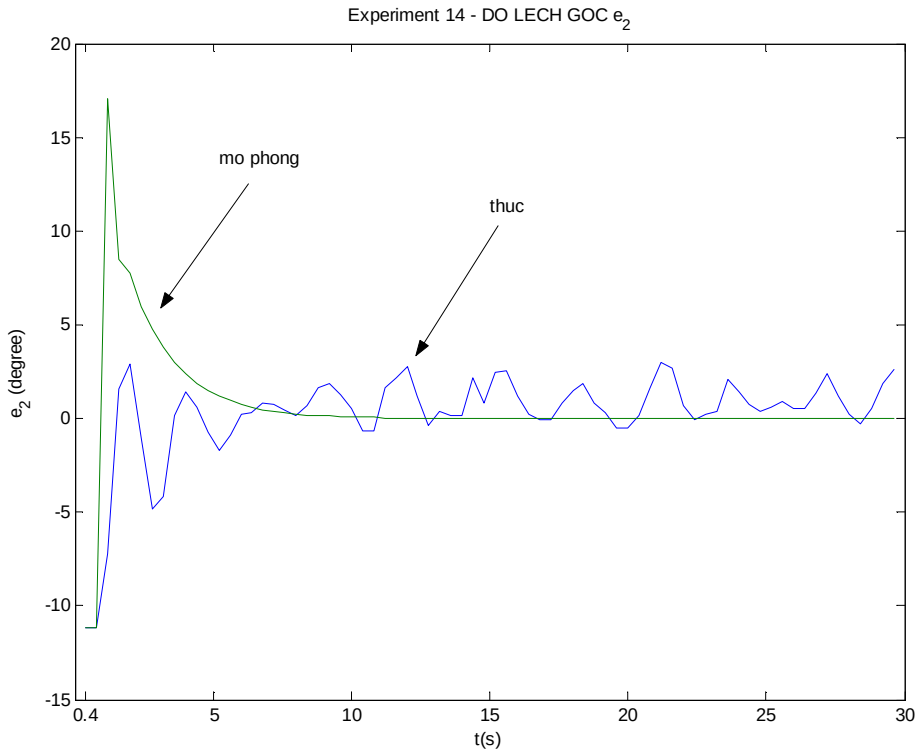
Đồ thị so sánh giữa kết quả mô phỏng và kết quả thí nghiệm:



Hình 4.4 So sánh đồ thị của vận tốc robot



Hình 4.5 So sánh đồ thị của sai số khoảng cách



Hình 4.6 So sánh đồ thị của sai số góc

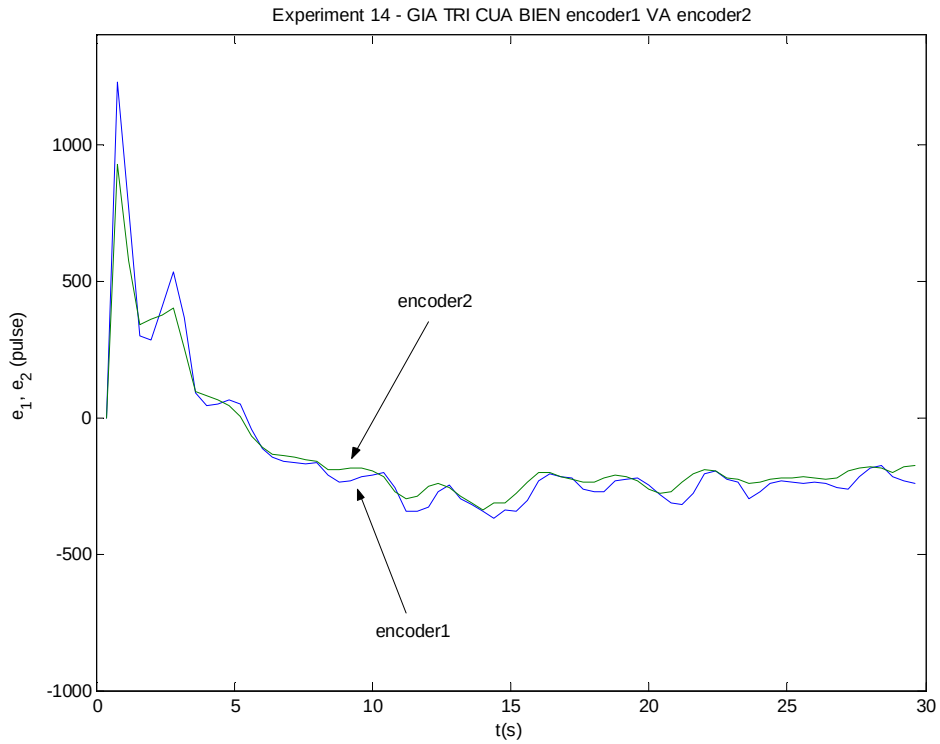
Nhận xét 1: Kết quả thực có đồ thị khá giống dạng đồ thị của kết quả mô phỏng, điều này cho thấy việc mô phỏng đã chỉ đường đúng đắn cho thực nghiệm.

Nhận xét 2: Kết quả thực khá xấu so với kết quả mô phỏng. Điều này cũng dễ hiểu vì robot thực của chúng ta chịu nhiều tác động của các sai số khi chế tạo, các sai số do linh kiện phần cứng và các sai số nhiễu, trong khi việc mô phỏng được thực hiện trong môi trường lý tưởng.

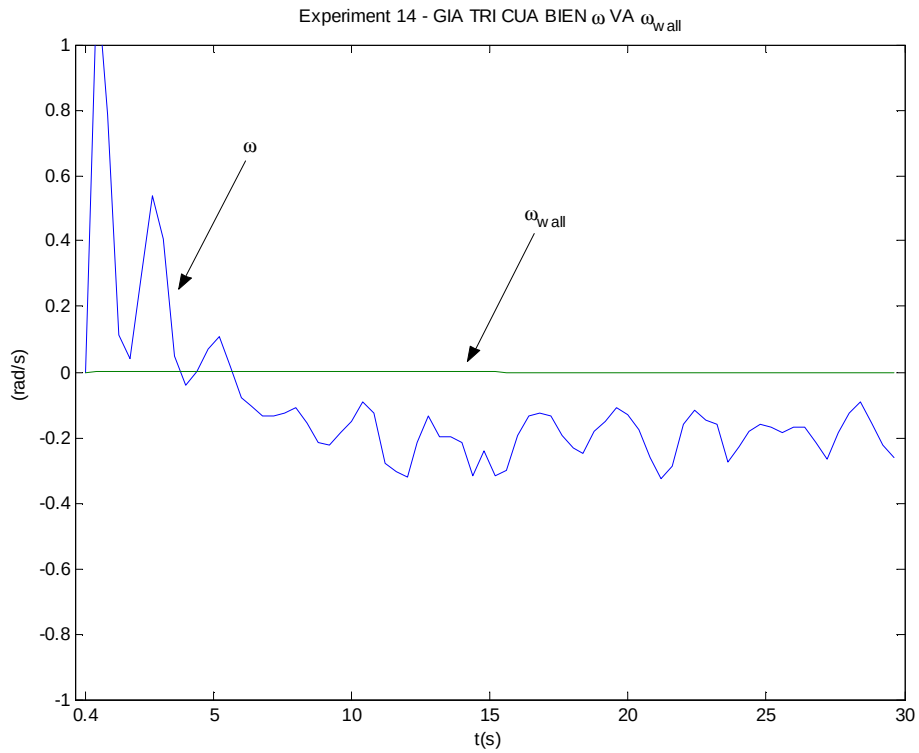
Nhận xét 3: Bộ điều khiển là hội tụ. Thời gian hội tụ là khoảng 10s theo mô phỏng và khoảng 15s theo kết quả thí nghiệm (ta đánh giá giá trị thời gian này là dựa vào nhận xét rằng từ thời điểm 15s, các giá trị e1 và e2 không thay đổi nhiều, chúng dao động ở gần điểm cân bằng).

4.3.2 Các nhận xét bổ sung

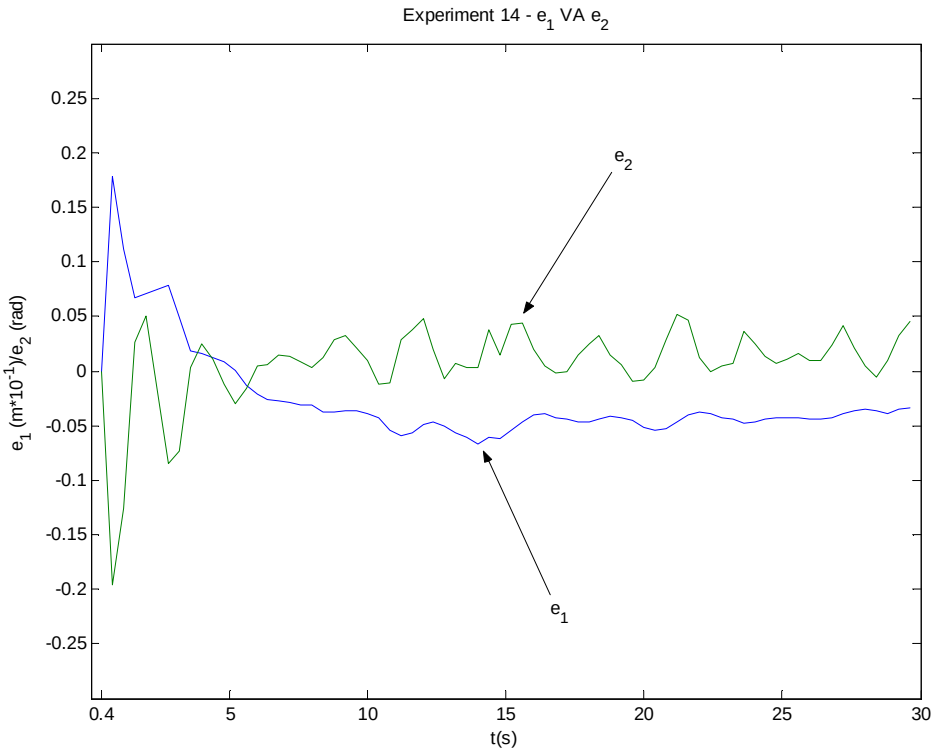
Ta xét thêm các đồ thị kết quả của thí nghiệm đã nêu ở trên:



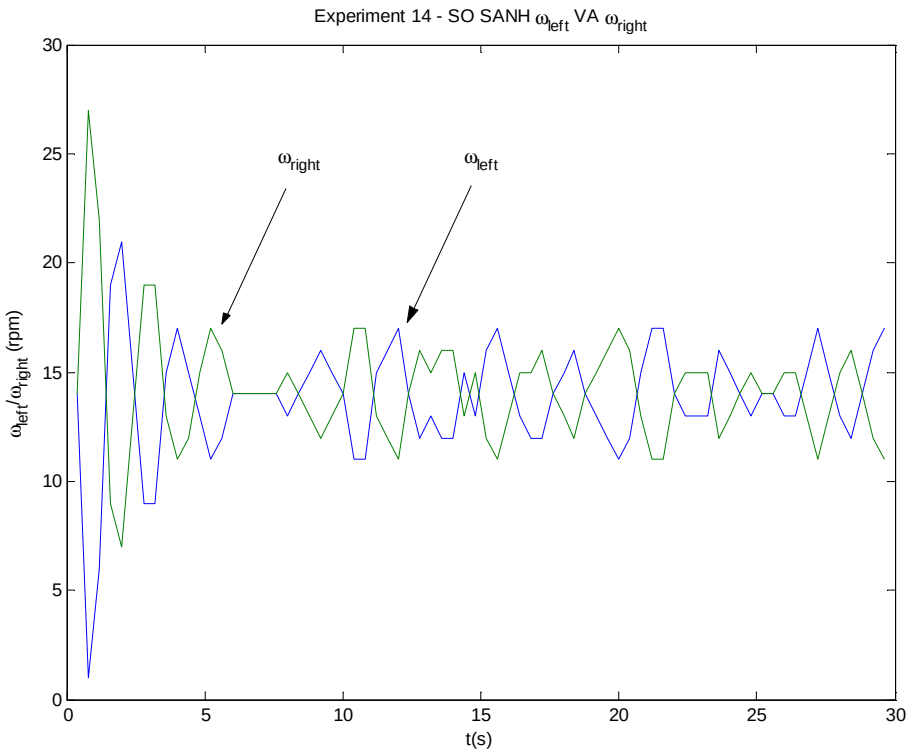
Hình 4.7 Giá trị của cảm biến



Hình 4.8 Giá trị vận tốc góc của robot và vận tốc góc (ước lượng) của tường



Hình 4.9 Biến đổi của các sai lệch trong quá trình hoạt động



Hình 4.10 Giá trị vận tốc ra lệnh cho 2 bánh xe

Nhận xét 4: Bộ điều khiển là ổn định với độ lệch ban đầu nằm trong vùng lân cận điểm cân bằng. (thật ra thí nghiệm được nêu trên đây có độ lệch ban đầu nằm gần biên của vùng lân cận đó).

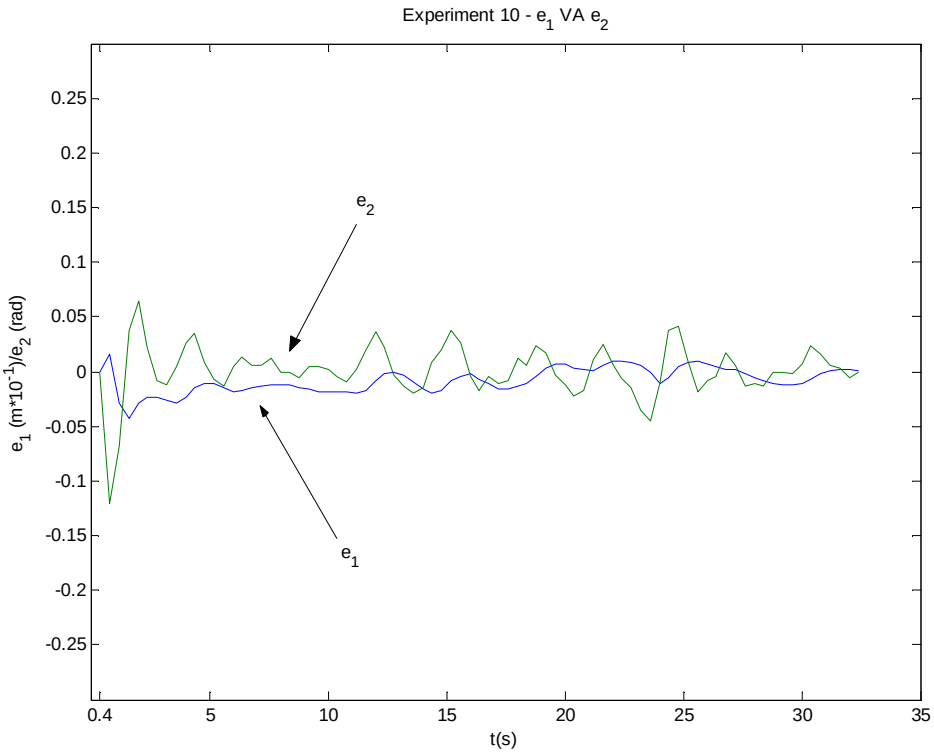
Hai thí nghiệm sau đây sẽ cho thấy tốc độ hội tụ của 2 sai số khoảng cách và góc:

Bảng 4.2 Thông số của 2 thí nghiệm (TN) dùng để so sánh

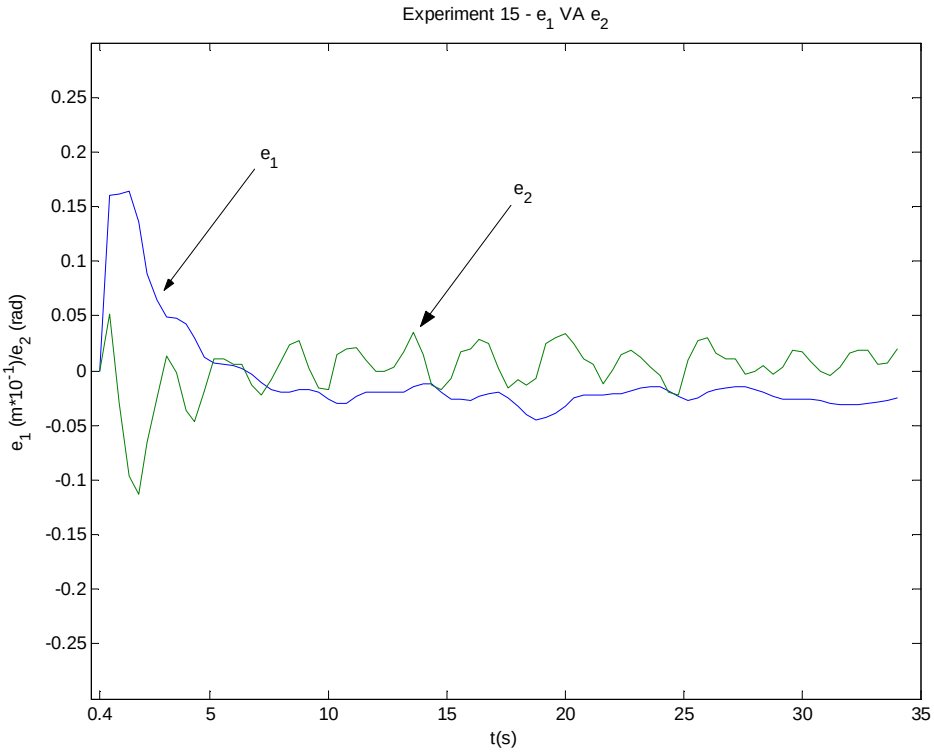
Tham số	*Giá trị*		*Đơn vị*	*Ghi chú*
	TN1	TN2		
d_0	0.25	0.25	m	
v_r	0.05	0.05	m/s	
st	0.4	0.4	s	
k_1	3.5	3.5		
k_2	615	615		
e_1 ban đầu	0.0017	0.0160	m	TH1: e_1 ban đầu rất nhỏ; TH2: e_1 ban đầu lớn
e_2 ban đầu	-0.1203	0.0516	m	TH1: e_2 ban đầu lớn; TH2: e_2 ban đầu rất nhỏ

Các đồ thị kết quả được cho ở hình 4.11.

Nhận xét 5: Sai số về góc (e_2) hội tụ nhanh hơn sai số về khoảng cách (e_1). Khi robot nhận e_1 ban đầu lớn, thời gian hội tụ chung của cả 2 sai số sẽ bị kéo dài hơn so với trường hợp e_2 ban đầu lớn.



a. Đồ thị e_1 và e_2 của TN1



b. Đồ thị e_1 và e_2 của TN2

Hình 4.11 So sánh các đồ thị e_1 và e_2 của hai thí nghiệm

5
KẾT LUẬN

5.1 Độ thích hợp của giải thuật

Luận văn này đã kiểm nghiệm được bộ điều khiển cho robot di động bám tường. Bộ điều khiển được dùng là bộ điều khiển hồi tiếp tất cả trạng thái (full-state feedback controller). Kết quả thí nghiệm đã kiểm chứng tính hội tụ và ổn định Lyapunov của bộ điều khiển.

5.2 Những hạn chế của đề tài

5.2.1 Về việc chế tạo phần cứng

Do chưa có nhiều kinh nghiệm trong việc gia công cơ khí, chúng tôi đã phải mất rất nhiều thời gian cho việc chế tạo phần cơ của robot. Tuy việc chế tạo còn nhiều sai số, robot cũng đạt điều kiện vừa đủ để phục vụ cho việc thí nghiệm. Khuyết điểm lớn nhất ở robot này là sử dụng các động cơ cũ (2 động cơ mua ở chợ động cơ cũ trên đường Vĩnh Viễn, Q.10, Tp.HCM), với các thông số không đảm bảo và mômen tải nhỏ, động cơ có đáp ứng chậm kéo theo thời gian lấy mẫu của hệ thống phải lớn, làm cho kết quả thí nghiệm không tốt như mong muốn.

5.2.2 Những hiện tượng ảnh hưởng đến kết quả và cách khắc phục

Trong quá trình thực hiện đề tài, chúng tôi quan sát thấy một số hiện tượng có thể gây sai lệch ở kết quả. Nhằm giúp các bạn sinh viên đi sau tiết kiệm thời gian, chúng tôi xin trình bày các hiện tượng đó và đề xuất cách khắc phục.

Bảng 5.1 Các hiện tượng ảnh hưởng đến kết quả và cách khắc phục

Hiện tượng	*Cách khắc phục*
Độ rơ của các khớp nối giữa trục động cơ (qua hộp giảm tốc) và bánh xe, bán kính 2 bánh dẫn động không bằng nhau. Hậu quả: bộ điều khiển PID hoạt động không tốt.	Gia công bánh xe bằng kim loại (nên dùng nhôm), cố gắng giảm sai số khi khoan lỗ trục bánh xe.

Rung động ở các thanh trượt và encoder trên cảm biến sẽ gây ra sai số của cảm biến, sai số tích luỹ trong thời gian hoạt động của robot sẽ đủ lớn để dẫn đến kết quả xấu ở bộ điều khiển.	Gá chặt encoder vào cảm biến; sử dụng các thanh trượt chính xác; giảm rung động cho robot bằng việc tạo ra tường và sàn êm.
Chương trình điều khiển lớn, khi biên dịch bằng PICC có thể không hoạt động đúng (do lỗi của trình biên dịch)	Tiết kiệm bộ nhớ chương trình bằng cách hạn chế dùng biến số thực và các lệnh xuất/nhập qua cổng nối tiếp, nên giới hạn chương trình trong khoảng <80% ROM.
Kết quả mô phỏng có thể tốt với nhiều bộ (k1,k2), nhưng khi đưa vào robot thực thì chỉ có một số bộ (k1,k2) thích hợp.	Các tham số k1 và k2 của bộ điều khiển cần phải dò lại trong khi thí nghiệm bằng cách điều chỉnh sau mỗi thí nghiệm.

5.3 Hướng nghiên cứu tiếp

Để tiếp tục đề tài này cho đến hết nhiệm vụ kiểm nghiệm giải thuật, chúng tôi xin đề nghị hướng nghiên cứu tiếp như sau: cải tiến phần cơ của robot để giảm các sai số chế tạo, thay các động cơ hiện có bằng 2 động cơ mới với bộ truyền động có tỉ số truyền lớn, đồng thời gắn encoder có độ phân giải lớn hơn vào động cơ để tăng khả năng điều khiển vận tốc; cố gắng giảm thời gian lấy mẫu của chương trình chính; lập trình bộ điều khiển dùng bộ quan sát để so sánh tốc độ hội tụ giữa việc dùng bộ điều khiển này với bộ điều khiển hồi tiếp tất cả trạng thái đã thực hiện trong đề tài.

TÀI LIỆU THAM KHẢO

Tiếng Việt:

[1] Trần Hữu Quế, Vẽ Kĩ Thuật Cơ Khí, Tập 1&2, Nhà xuất bản Giáo Dục, **2000**

[2] Nguyễn Doãn Phước, Phan Xuân Minh & Hán ThànhTrung, Lý Thuyết Điều *Khiển Phi Tuyến, Nhà xuất bản Khoa Học và Kỹ Thuật, 2003,* trang 53-75.

[3] Nguyễn Viết Hiệp & Phạm Đình Anh Vũ, Mô hình hoá – mô phỏng và điều *khiển Robot theo dấu tường, Luận văn tốt nghiệp Đại học,* Đại học Bách Khoa Tp.HCM, 2004.

Tiếng nước ngoài:

[4] Detriche, Jean-Marie, *Systemes robotiques et Mecatroniques*, Cours d'Ecole Centrale Paris, **1999-2000**.

[5] Lagoudakis, Michail G., *Mobile Robot Local Navigation with a Polar Neural Map*, MSc Thesis, University of Southwestern Louisiana, **1998**.

[6] P. van Turennout, G. Honderd, L.J. van Schelven,*Wall following control of a Mobile Robot,* International Conference on Robotics and Automation, Nice, France, **1992**, p. 280-285.

[7] Medromi, J.Y. Tigli, and M.C. Thomas, *Posture Estimation of a Mobile Robot: Observers-Sensors*, Proceedings of the 1994 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems, p. 661-666, **1994**.

[8] Urzelai J., Uribe J.P., and Ezkerra M., *Fuzzy Controller for Wall-Following With a non-holonomous Mobile Robot*, Proceedings of the 6th IEEE International Conference on Fuzzy System, Vol. 3, p.1361-1368, **1997**.

[9] A. Bemporad, M.D. Marco, and A. Tesi, *Wall-Following Controllers for a Sonar Mobile Robot*, Proceedings of the 36th Conference on Decision & Control, California USA, p. 3063-3068, **1997**.

[10] T. Yata, L. Kleeman, and S. Yuta, *Wall Following Using Angle Information Measured by a Single Ultrasonic Transducer*, Proceedings of the 1998 IEEE International Conference on Robotics and Automation, p1590-1596, **1998**.

[11] Chung Tan Lam, *A nonlinear Feedback Control of Wall-Following Mobile Robot*, Ms.C Thesis, Pukyong National University, Korea, **2004**.

[12] Rodney A. Brooks, A robust layered control system for a mobile robot, *Research report of Massachusette Institute of Technology*, **1985**.

[13] http://www.microchip.com

[14] **http://www.semiconductors.philips.com/buses/i2c/**

[15] John A. Shaw, *The PID Control Algorithm - How it works, how to tune it, and how to use it, 2nd edition*, (ebook), **2003**.

PHỤ LỤC A
SƠ LƯỢC VỀ CHUẨN GIAO TIẾP I2C [14]

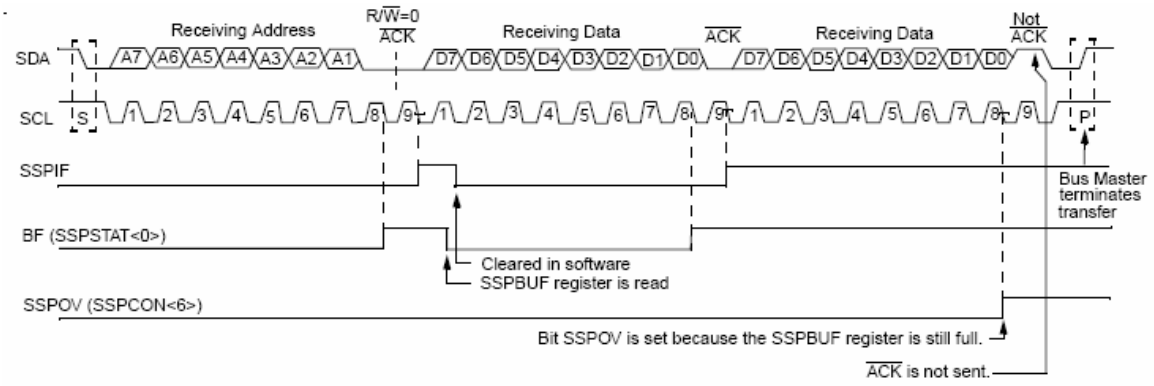
I2C là chuẩn giao tiếp ban đầu được hãng Philips phát triển để phục vụ cho các giao tiếp với Tivi, sau đó, do những lợi thế của chuẩn giao tiếp này nên nó đã trở nên rất phổ biến trong công nghiệp. Các tài liệu về chuẩn I2C được cung cấp rất đầy đủ và chi tiết tại trang web của hãng Philips.

Microchip đã tích hợp chuẩn giao tiếp này vào phần cứng cho một số vi điều khiển, trong đó có PIC16F877. Chuẩn này được biết dưới dạng chuẩn giao tiếp đồng bộ SSP. Trong chuẩn giao tiếp SSP có hai chuẩn con là chuẩn SPI và I2C. SPI là chuẩn giao tiếp nối tiếp dùng 1 dây nối, và I2C là chuẩn giao tiếp nối tiếp 2 dây. Dây SDA là dây để truyền dữ liệu và dây SCL là dây để giữ nhịp cho dữ liệu truyền.

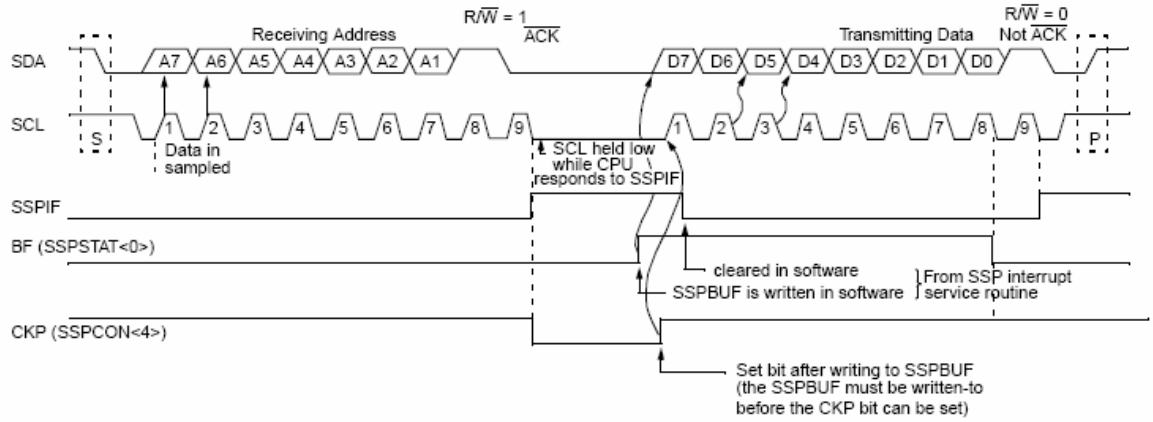
Chuẩn I2C là chuẩn giao tiếp trong đó dùng một thiết bị làm Master, và các thiết bị khác trong mạng là Slave trong một thời điểm nhất định. Tại một thời điểm, Master có quyền đọc và xuất dữ liệu qua tất cả các Slave thông qua địa chỉ của Slave đó.

Master bắt đầu quá trình đọc hoặc ghi dữ liệu vào một Slave bằng cách đặt tín hiệu Start (S) vào đường truyền (hình A.1 và A.2). Byte tiếp theo được gửi đi là byte địa chỉ của Slave cần giao tiếp. Quá trình truyền nhận được kết thúc bằng bit Stop (P).

Sau đây là biểu đồ thời gian truyền và nhận trong chuẩn giao tiếp I2C



Hình A.1: Biểu đồ nhận dữ liệu, địa chỉ 7 bit



Hình A.2: Biểu đồ truyền dữ liệu, địa chỉ 7 bit

I2C phần cứng của PIC 16F877 hỗ trợ hai chế độ định địa chỉ 10 bit và 7 bit. Tuy nhiên, chúng ta vẫn chỉ thường sử dụng địa chỉ 7 bit, hỗ trợ giao tiếp 128 thiết bị. Tuy vậy, chuẩn I2C cũng giống như chuẩn RS232, khi thao tác với các vi điều khiển, có thể được thiết lập bằng phần mềm.

Một lần truyền địa chỉ, sẽ có 8 bit, bit thấp nhất là bit xác định chế độ đọc hoặc ghi (R/W) (hình A.2).

Chuẩn I2C rất dễ sử dụng, có các tốc độ truyền nhận là 100Kbps, 400Kbps và 1Mbps, vi điều khiển PIC 16F877 hỗ trợ tốc độ 100Kbps và 400Kbps, nhanh hơn nhiều lần so với chuẩn RS232. Ngoài ra, không cần dùng bất kỳ thiết bị chuyển đổi nào để chuyển đổi điện áp tín hiệu, do vậy, chuẩn I2C thích hợp nhất cho các giao tiếp trong phạm vi ngắn (dưới 1m) giữa các vi điều khiển.

PHỤ LỤC B
MÃ NGUỒN CÁC CHƯƠNG TRÌNH

Chương trình cho master module:

```
/************************
Description: This sourcecode is for the master module of the thesis
"Study on Control of Wall-Following Mobile Robot"
by Doan Minh Dang - P9900012
Start date: 2004.05.27
End date: 2004.07.03
************************/
#include <16f877.h>
#device PIC16F877 *=16 ADC=10
#include <string.h>
#include <stdlib.h>
#include <math.h>
#use delay(clock=20000000)
#use rs232(baud=9600,parity=N,xmit=PIN_C6, rcv=PIN_C7)
#use I2C(master, sda=PIN_C4, scl=PIN_C3)
#use fast_IO(E)
#priority ccp1,ccp2,rb,timer1
//address
#define address_left 0xa0
#define address_right 0xc0
//constant
float vr=0.05,d0=0,k1=2.5,k2=250;//unit: [vr]=m/s,[d0]=m
```

```
//variables
int1 vr_ok=0;
char s[5];
int1 stop1=0;
int1 stop2=0;
int1 online=0;//operating flag
//int16 temp=0; for testing
signed int16 encoder_1=0;//supplemental encoder in primitive unit, [encoder_1]=pulse
signed int16 encoder_2=0;//main encoder in primitive unit, [encoder_2]=pulse
//signed int16 encoder_1_old,encoder_2_old;for filtering
signed int16 temp_int16=0;
//float encoder_1_m=0;//d2, [encoder_1_m]=m
//float encoder_2_m=0;//d1, [encoder_2_m]=m
float d1,d2;//[d1]=m,[d2]=m
float e1,e2,omega,delta_omega_mu,part1,omega_left,omega_right;
float omega_mu=0;
//float omega_mu_temp=0;
//[e1]=mm
//[e2]=rad
//[omega]=rad
//[omega_mu]=rad
//[delta_omega_mu]=rad/(T*ms) (T=150)
//[omega_left]=[omega_right]=round/minute (rpm)
signed int16 omega_left_new,omega_right_new;
//[omega_left_new]=[omega_right_new]=round/minute
byte last_b;
int1 count=0;
```

```
int cycle_count=0;
int1 new_cycle=0;
int number_cycle=2;
signed int16 omega_left_last=0;
//[omega_left_last]=round/minute
signed int16 omega_right_last=0;
//[omega_right_last]=round/minute

void tilt_c5(int16 speak_time)
{//master board will speak in specified time(ms)
  output_high(PIN_C5);
  delay_ms(speak_time);
  output_low(PIN_C5);
}
#SEPARATE
void send_commands_via_i2c(int first_add,signed int16 first_data,int second_add,signed int16 second_data)
{
int hi,lo;
//send command to first slave
  hi=make8(first_data,1);
  lo=make8(first_data,0);
      i2c_start();
      i2c_write(first_add);
      delay_ms(1);
      i2c_write(lo);//send low byte of velocity
      delay_ms(1);
      i2c_write(hi);//then send high byte
      i2c_stop();
```

```
//send command to second slave
  hi=make8(second_data,1);
  lo=make8(second_data,0);
      delay_ms(1);
      i2c_start();
      i2c_write(second_add);
      delay_ms(1);
      i2c_write(lo);
      delay_ms(1);
      i2c_write(hi);
      i2c_stop();
}

void calculate_slave_velocities()
{
//input: omega_left, omega_right - float, global variables
//output: omega_left_new,omega_right_new - signed int16, global variables
//This part is moved to a function to reduce memory cost
      //calculate new left and right velocities
      omega_left=(vr-omega*0.095)/0.033;
      omega_right=(vr+omega*0.095)/0.033;
      //send new omega to left and right modules
      omega_left_new=(signed int16)(omega_left*9.55);
//convert from rad/s to rpm: (rad/s)*60/2pi=rpm
      omega_right_new=(signed int16)(omega_right*9.55);
}

#INT_RB
```

```
void rb_isr()
{
  byte changes,new_b;
   new_b=input_b();
  changes = last_b ^ new_b;
  last_b = new_b;
    if (bit_test(changes,5))//RB5: start button
    {
      //b5 went low
      stop1=!stop1;//if it is set, it will be clear, and vice versa
    }
    if (bit_test(changes,4))//RB4: stop button
    {
      //b4 went low
      if (stop1) stop2=1;
    }
  delay_ms(200);  //debounce
}
#INT_CCP1
void vantoc1()
{
   if (input(PIN_E1)) encoder_1++;
   else encoder_1--;
}
#INT_CCP2
void vantoc2()
{
    if (input(PIN_E2)) encoder_2--;
    else encoder_2++;
```

```
}
#INT_TIMER1
void timer1_isr()
{
  set_timer1(34286);
  cycle_count++;
   if (cycle_count==number_cycle)
   {
    cycle_count=0;
    new_cycle=1;
   }
}
void main()
{
//INIT
  tilt_c5(500);
  set_tris_e(0x07);
  set_tris_b(0b00110000);
  last_b=input_b();
  setup_timer_1(T1_INTERNAL|T1_DIV_BY_8);
  setup_CCP1(CCP_CAPTURE_FE);
  setup_CCP2(CCP_CAPTURE_FE);
  enable_interrupts(INT_CCP1);
  enable_interrupts(INT_CCP2);
  enable_interrupts(INT_RB);
  enable_interrupts(GLOBAL);
//RECEIVE PARAMETERS
  //vr
  gets(s);
```

```
        vr=atof(s);
        printf("Receive velocity desired: %f \n",vr);
        gets(s);
        number_cycle=atoi(s);
        printf("Sampling time=%u*0.05s \n",number_cycle);
        encoder_1=0;
        encoder_2=0;
        tilt_c5(1000);
        gets(s);
        k1=atof(s);
        printf("k1=%f",k1);
        gets(s);
        k2=atof(s);
        printf("k2=%f",k2);
        //first move: move straight forward with at speed=vr
        omega=0;
        calculate_slave_velocities();
        printf("Go!\n");
        send_commands_via_i2c
(address_left,omega_left_new,address_right,omega_right_new);
        omega_left_last=omega_left_new;
        omega_right_last=omega_right_new;
        printf("l:%ld\nr:%ld\n",omega_left_new,omega_right_new);
        new_cycle=0;
        //up to here, it cost 5146 timer1 counts=8233us
        //it will ring 500-8=492ms
        tilt_c5(492);
        set_timer1(34286);//overflow after 3250 machine
cycles=50ms
```

```
  enable_interrupts(INT_TIMER1);


//MAIN LOOP
   while ((!stop1)&&(!stop2))
   {
      if (new_cycle)
      {
   //start new cycle
//GET d1 AND d2
         d2=(float)encoder_1/51000.0;//scale: 51000 pulse/m
      d1=(float)encoder_2/51000.0;
//SEND ENCODERS' VALUES TO PC
         printf("1:%ld\n2:%ld\n",encoder_1,encoder_2);
//CALCULATE e1 & e2
         e2=atan((d1-d2)/0.030);
      e1=-d1*cos(e2);
         //calculate omega
      part1=vr-(e1+d0)*omega_mu/cos(e2);
      omega=k2*part1*e1-k1*sin(e2)+omega_mu;
         //calculate new omega_mu
      delta_omega_mu=e1*(e1+d0)*tan(e2)-sin(e2)/k2;
      delta_omega_mu=delta_omega_mu*0.05*number_cycle;
      omega_mu +=delta_omega_mu;
         //calculate data to send to slave
     calculate_slave_velocities();
//CHECK FOR EXCEEDED VELOCITIES
         if (omega_left_new>90)//maximum limit
         {
         printf("ERR1");//err1: Left velocity exceed the range
```

```
            omega_left_new=90;
           }
            if (omega_left_new<-90)//maximum limit
            {
            printf("ERR2");//err1: Left velocity exceed the range
            omega_left_new=-90;
            }
            if (omega_right_new>90)//maximum limit
            {
            printf("ERR3");//err2: Right velocity exceed the range
            omega_right_new=90;
            }
            if (omega_right_new<-90)//maximum limit
          {
            printf("ERR4");//err2: Right velocity exceed the range
            omega_right_new=-90;
            }
    //SEND NEW VELOCITIES
        send_commands_via_i2c
(address_left,omega_left_new,address_right,omega_right_new);

printf("l:%ld\nr:%ld\n",omega_left_new,omega_right_new);
        omega_left_last=omega_left_new;
        omega_right_last=omega_right_new;
        //wait until next cycle
        new_cycle=0;
          }
       }
    //SEND STOP COMMAND
```

```
send_commands_via_i2c(address_left,0xffff,address_right,0xffff
);
      tilt_c5(200);//ringing for stop declaring
      delay_ms(200);
      tilt_c5(200);
   }
```

Chương trình cho slave module:

```
/**************
```

Description: This program is used for controlling motor DC velocity with PID method.

There are slight differences between the programs for left and right modules.

Doan Minh Dang - P9900012

Start date: 2004.04.27

End date: 2004.06.20

```
**************/
#include <16f877.h>
#device PIC16F877 *=16 ADC=10
#fuses hs, nowdt, noprotect, nolvp, put, brownout
#use delay(clock=20000000)
#use rs232(baud=9600,parity=N,xmit=PIN_C6, rcv=PIN_C7)
#use
I2C(slave,sda=PIN_C4,scl=PIN_C3,address=0xa0,FORCE_HW)
#include <stdlib.h>
#include <string.h>
#include <control_motor.c>
//the function run_motor(value,condition,direct,frequency) is
used to control motor
```

```
//constant
float kp=8.2;
float ki=1;
float kd=0.8;
//variables
int1 i2c_flag=0;
int1 start_cycle=0;
int1 stop=0;
int i2c_error=0;
int velocity_error=0;
int16 times=0;
int error_command;
signed int count=0;
signed int count_last=0;
signed int delta_count=0;
char s[7];
byte command_listen[3];
int i=0;
signed int16 command_velocity=0; //[rpm]
int1 direction;
float real_velocity=0;
signed int16 real_velocity_print;
signed int16 duty=0;
int16 PWMduty;
float error_rpm_now=0;
float error_rpm_last=0;
float error_rpm_last_last=0;
float p_value;
float i_value;
```

```
    float d_value;
    float control_value;
    ////////////////////////////////////////////////////////////////
////////////////////////////////
    void tilt_a0(int16 speak_time)
    {//slave board will speak in specified time(ms)
      output_high(PIN_A0);
      delay_ms(speak_time);
      output_low(PIN_A0);
    }

    void convert_duty()
    {
        if (duty<0)
        {
          PWMduty=-duty;
          direction=0;//negative speed: rotate backward
          }
        else
          {
          PWMduty=duty;
          direction=1;//positive speed: rotate forward
          }
          //control_motor(dir=0): backward
          //control_motor(dir=1): forward
      if (PWMduty>1023)
      {
            PWMduty=1023;
            velocity_error++;
```

```
      }
      if (64>PWMduty)
      {
            PWMduty=64;
      }
    }

    void PID_calculation()
    {
        p_value=kp*(error_rpm_now-error_rpm_last);
        i_value=ki*error_rpm_now;
        d_value=kd*(error_rpm_now-2*error_rpm_last
+error_rpm_last_last);
        control_value=p_value+i_value+d_value;
        duty+=(signed int16)control_value;
    }

    ////////////////////////////////////////////////////////////
//////////
    #INT_SSP
    void ssp_isr()
    {
      if (i2c_poll()==FALSE)
       {
        i=0;
       }
      else
       {
            command_listen[i++]=i2c_read();
```

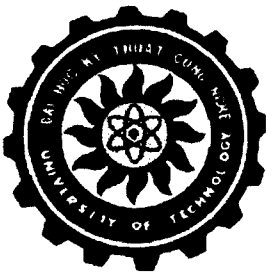
```
//        i=1;
            if (i==3)
          {
                  times++;
          if
((command_listen[1]!=0xa0)&&(command_listen[2]!=0xa0))
    //condition: data must be received correctly
              {
                  i2c_flag=1;
                  i=0;
command_velocity=make16(command_listen[2],command_listen[1]);
                  if (command_velocity==0xffff) //stop command
                  {
                        stop=1;
                  }
              }
              else i2c_error++;
        }
      }
    }
    #INT_EXT
    void count_encoder()
    {
      if (input(PIN_B7)) count--;//backward rotate - left
      else count++;//forward - left
      //reverse for right module
    }
    ////////////////////////////////////////////////////////
/////
```

```
#INT_TIMER1
void update_time()
{
  start_cycle=1;
  set_timer1(15536); //update time = 10ms
  delta_count=count-count_last;
  count_last=count;
//multiplier: 4ms - 7.5; 5ms - 6; 10ms - 3
  real_velocity=(float)delta_count*3.0;
  error_rpm_last_last=error_rpm_last;
  error_rpm_last=error_rpm_now;
  error_rpm_now=(float)command_velocity-real_velocity;
}
//////////////////////////////////////////////////////////////////
//////
void main()
{
  tilt_a0(400);
  set_tris_b(0x81);
  ext_int_edge(0,L_TO_H);
  control_motor(0,0,0,2);
  command_listen[0]=0;
  command_listen[1]=0;
  command_listen[2]=0;
  setup_timer_1(T1_INTERNAL);
  enable_interrupts(GLOBAL);
  enable_interrupts(INT_SSP);
  while (!i2c_flag) {};
  delay_us(3385);
```

```
tilt_a0(500);
set_timer1(15535);
enable_interrupts(INT_EXT);
enable_interrupts(INT_TIMER1);
start_cycle=0;
while (!stop)
{
  if (start_cycle)
  {
    if (command_velocity!=0)
    {
      PID_calculation();
            convert_duty();
      control_motor(PWMduty,1,direction,2);
       }
    else {control_motor(0,1,0,2);}//stop
    start_cycle=0;
  }
}
control_motor(0,0,0,2);
disable_interrupts(GLOBAL);
disable_interrupts(INT_SSP);
disable_interrupts(INT_EXT);
disable_interrupts(INT_TIMER1);
tilt_a0(200);
delay_ms(200);
tilt_a0(200);
delay_ms(200);
if ((i2c_error>0)||(velocity_error>0))
```

```
    {
     tilt_a0(1000);
    }
   while (1)
   {
    gets(s);
     error_command=atoi(s);
    switch (error_command)
    {
            case 1: printf("i2c_error:%u ",i2c_error);
            break;
            case 2: printf("velocity_error:%u",velocity_error);
            break;
            case 3: printf("times:%lu",times);
            break;
    }
   }
}
```

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG KĨ THUẬT CÔNG NGHỆ
KHOA ĐIỆN- ĐIỆN TỬ
NGÀNH ĐIỆN TỬ VIỄN THÔNG



NGUYỄN CHÍ THIỆN
02DHDT186

ROBOT TỰ HÀNH

CHUYÊN NGÀNH ĐIỆN TỬ VIỄN THÔNG

LUẬN VĂN TỐT NGHIỆP

TP HỒ CHÍ MINH- 01/2009

LỜI CÁM ƠN

Để thực hiện đề tài, em đã nhận được rất nhiều sự chỉ dẫn, giúp đỡ và động viên quý báu của nhiều người, thiếu một trong các sự giúp đỡ đó cũng có thể làm cho đề tài không đạt kết quả như hiện nay.

Trước hết, em xin bày tỏ lòng cảm ơn sâu sắc đối với thầy Thạc Sĩ Nguyễn Trọng Hải, người thầy hướng dẫn đã tận tình chỉ cho em phương pháp nghiên cứu khoa học, thầy cũng đã cung cấp cho em rất nhiều kiến thức chuyên sâu để thực hiện đề tài.

Em cũng vô cùng cảm ơn các thầy cô ở bộ môn Điện Tử Viễn Thông Trường Đại Học Kĩ Thuật Công Nghệ Tp.HCM đã tham gia quá trình đào tạo và hướng dẫn em trong suốt thời gian học đại học, nhờ các thầy cô mà em có đủ kiến thức và lòng tự tin để thực hiện đề tài nghiên cứu này cũng như các đề tài trong tương lai.

Bên cạnh đó, sự hợp tác và giúp đỡ của bạn bè và các thế hệ đàn anh cũng giúp em rất nhiều trong việc thực hiện đề tài này.

Con cũng xin cảm ơn gia đình đã luôn chăm sóc và quan tâm đến việc học của Con. Con vô cùng cảm ơn và luôn tự hào vì có Ba, Mẹ luôn động viên con trong quá trình học tập.

Và cuối cùng, tôi xin gửi lời cảm ơn tới những người đã tham gia giúp đỡ tôi trong quá trình thực hiện luận văn mà tôi chưa nêu lên ở đây, sự giúp đỡ của họ dù ít hay nhiều cũng đóng góp một phần vào kết quả thực hiện đề tài tốt nghiệp này.

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐHDL KỸ THUẬT CÔNG NGHỆ
KHOA ĐIỆN – ĐIỆN TỬ
****

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
ĐỘC LẬP – TỰ DO – HẠNH PHÚC
--- oOo ---

NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP

(Sinh viên phải đóng bản nhiệm vụ này vào trang thứ nhất của đồ án)

Họ và tên SV : **Nguyễn Chí Thiện** MSSV : 02DHDT186
Ngành : **Điện tử - Viễn thông** Lớp : 02ĐT02

1. Đầu đề đồ án tốt nghiệp:

ROBOT TỰ HÀNH

2. Nhiệm vụ đồ án tốt nghiệp (Yêu cầu về nội dung và số liệu ban đầu):

- Điều khiển động cơ bước

- Điều khiển tốc độ chuyển động của động cơ

- Điều khiển động cơ bước theo đường đi bất kỳ dùng cảm biến quang

3. Ngày giao nhiệm vụ đồ án : **03/10/2008**

4. Ngày hoàn thành đồ án : **03/01/2009**

5. Họ tên người hướng dẫn :

1/ Ths. Nguyễn Trọng Hải

2/ .....

3/ .....

Phần hướng dẫn

1/ .....

2/ .....

3/ .....

Nội dung và yêu cầu của ĐATN đã được thông qua

Ngày 10 tháng 10 năm 2008

P. TRƯỞNG KHOA

(Ký và ghi rõ họ tên)

ĐẠI HỌC KỸ THUẬT CÔNG NGHỆ
KHOA ĐIỆN - ĐIỆN TỬ

ThS. Ngô Cao Cường

NGƯỜI HƯỚNG DẪN CHÍNH

(Ký và ghi rõ họ tên)

Ths. Nguyễn Trọng Hải

TH.S. Ngô Cao Cường

BẢNG NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

Họ tên sinh viên : **NGUYỄN CHÍ THIỆN** Lớp: **02DT02**
Giáo viên hướng dẫn : **Thầy NGUYỄN TRỌNG HẢI**
Tên đề tài:
ROBOT TỰ HÀNH
Nhận xét của giáo viên hướng dẫn :

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

TP. Hồ Chí Minh, Ngày 3 tháng 1 năm 2008

Giáo viên hướng dẫn

Nguyễn Trọng Hải

BẢNG NHẬN XÉT CỦA GIÁO VIÊN PHẢN BIỆN

Họ tên sinh viên : **NGUYỄN CHÍ THIỆN** Lớp: **02DT02**

Giáo viên phản biện : .....

Tên đề tài:

ROBOT TỰ HÀNH

Nhận xét của giáo viên phản biện :

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

TP. Hồ Chí Minh, Ngày tháng 1 năm 2008

Giáo viên phản biện

LỜI NÓI ĐẦU

Cùng với sự phát triển nhanh chống của các ngành khoa học kỹ thuật, công nghệ chế tạo robot cũng phát triển theo. Robot ngày càng gia tăng về số lượng, lẫn chất lượng nhằm đáp ứng cho nhu cầu tự động hoá các quy trình sản xuất, các nhu cầu trong thám hiểm ... cũng như phục vụ cho đời sống con người.

Trong xu thế sử dụng robot ngày càng nhiều trong sản xuất và sinh hoạt. Ở Việt Nam, ngành công nghiệp robot cũng đang từng bước phát triển. Tuy nhiên các robot này chỉ sản xuất dưới dạng người máy công nghiệp, các tay máy hoặc như khủng long máy phục vụ cho nhu cầu giải trí. Trong khi đó, loại robot di động vẫn chưa được chú ý nhiều, mặc dù đây là loại robot có tiềm năng rất lớn, có thể dùng trong nhiều mục đích khác nhau như : giải trí trong gia đình, phục vụ trong các công sở, trong các mục đích quân sự, và dùng nhiều trong thám hiểm hành tinh. Tiêu biểu trong họ robot này có thể kể đến như : chó robot Aibo, xe tự hành Sojourner thám hiểm sao hoả và các loại robot dò mìn ...

Robot di động là một lĩnh vực có nhiều tiềm năng, nên dù hạn chế về kiến thức và ít ỏi về tài liệu nhưng chúng em cũng cố gắng thực hiện luận án về đề tài này ở mức độ đơn giản, và đây là cố gắng của em trong suốt ba tháng tìm hiểu, thiết kế và thi công. Dù đã cố gắng nhưng cũng còn rất nhiều sai sót, khuyết điểm. Tự đáy lòng,em rất mong được sự chỉ bảo thêm nơi quý thầy cô và bạn bè. Em xin chân thành cảm ơn sự quan tâm của quý thầy cô và bạn bè.

MỤC LỤC

LỜI CẢM ƠN Trang 3
NHIỆM VỤ LUẬN ÁN TỐT NGHIỆP Trang 4
LỜI MỞ ĐẦU Trang 7
TÓM TẮT ĐỀ TÀI Trang 10

CHƯƠNG 1: QUÁ TRÌNH PHÁT TRIỂN ROBOT

I. Giới thiệu sơ lược về Robot Trang 11
1.1. Sơ lược lịch sử phát triển của Robot Trang 11
1.2. Sự tác động của Robot lên xã hội Trang 15
1.3. Tương lai của Robot Trang 16
II. Phương pháp thiết kế Robot di động Trang 16

CHƯƠNG 2: NỘI DUNG ĐIỀU KHIỂN ROBOT

I. Nhiệm vụ đề tài Trang 18
II. Yêu cầu kỹ thuật hoạt động theo hành trình Trang 18
2.2.1. Yêu cầu Trang 18
2.2.2. Phương án giải quyết Trang 18
2.2.2.1. Nguyên tắc hoạt động cảm biến Trang 18
2.2.2.2. Phương pháp dò đường Trang 19
2.2.3. Sơ đồ khối phần dò đường Trang 19

CHƯƠNG 3: TỔNG QUAN ĐIỀU KHIỂN ĐỘNG CƠ

I. Khái quát về động cơ bước Trang 20
3.1.1. Khái niệm chung Trang 20
3.1.2. Nguyên lý hoạt động Trang 21
3.1.3. Moment đồng bộ và trạng thái ổn định Trang 23
3.1.4. Cấu tạo và phân loại động cơ bước Trang 25
II. Cơ sở lý thuyết điều khiển động cơ bước Trang 28
3.2.1. Ba chế độ (mode) điều khiển động cơ bước Trang 28
3.2.2. Các đặc trưng tín hiệu điều khiển động cơ bước Trang 32
3.2.3. Điều khiển tốc độ quay của động cơ bước Trang 33
3.2.4. Điều khiển chiều quay của động cơ bước Trang 34

III. Điều khiển động cơ bước 4 pha Trang 35
3.3.1. Cấu tạo Trang 35
3.3.2. Phương thức hoạt động Trang 35
3.3.3. Xác định các ngõ vào Trang 39
3.3.4. Ứng dụng Trang 40

CHƯƠNG 4: THIẾT KẾ KỸ THUẬT

I. Mô hình thiết kế Trang 41
4.1.1. Sơ đồ khối Trang 41
4.1.2. Sơ đồ nguyên lý Trang 41
II. Phân tích mạch điện chi tiết các khối Trang 42
4.2.1. Thiết kế cụ thể đo đường Trang 42
4.3.2. Nguyên lí mạch RESET 89C51 Trang 43
4.2.3. Mạch tạo dao động Trang 43
4.2.4. Mạch điều khiển động cơ Trang 44
III. Lưu đồ và chương trình giải thuật Trang 46

KẾT LUẬN VÀ HƯỚNG MỞ RỘNG PHÁT TRIỂN ĐỀ TÀI Trang 55
TÀI LIỆU THAM KHẢO Trang 56
PHỤ LỤC

TÓM TẮT ĐỀ TÀI

I. ĐẶT VẤN ĐỀ:

- Với sự phát triển của công nghệ điện tử, kỹ thuật số, các hệ thống điều khiển dần dần được tự động hoá. Nó trở thành một lĩnh vực khoa học, mà ứng dụng của nó không thể thiếu trong dân dụng cũng như trong các ngành công nghiệp, và nó còn là nền tảng cho các ngành khác. Với kỹ thuật tiên tiến như: vi xử lý, vi mạch số ... được ứng dụng vào lĩnh vực điều khiển.
- Trong quá trình sản xuất ở các nhà máy, xí nghiệp. Việc phải sử dụng những cánh tay robot, những máy tự động làm việc theo theo chương trình cài sẳn, những robot di động thực hiện các công việc nguy hiểm là một yêu cầu hết sức cần thiết và quan trọng. Nó giúp thực hiện những thao tác đòi hỏi độ chính xác cao, đảm bảo an toàn cho người lao động và đồng thời kéo theo sự phát triển của nền kinh tế.
- Để đáp ứng yêu cầu trên thì có nhiều phương pháp để thực hiện. Nhưng ở đây, vì kiến thức còn hạn chế, thời gian thực hiện luận án có hạn và cũng thiếu về linh kiện, thiết bị nên chúng em chỉ thực hiện đề tài ở mức độ đơn giản. Đó là thiết kế và thi công xe mô hình tự hành theo đường sơn màu trắng. Đây là lý do em chọn đề tài này.

II. MỤC ĐÍCH NGHIÊN CỨU:

- Mục đích trước hết là hoàn thành đề tài tốt nghiệp.

- Áp dụng những gì đã học lý thuyết vào thực tế, phát huy những thành quả của vi điều khiển vào thực tế nhằm tạo những sản phẩm, những thiết bị tiên tiến hơn và đạt hiệu quả sản xuất cao hơn.

- Tìm hiểu rộng thêm những kiến thức đã được học.

CHƯƠNG I: GIỚI THIỆU ROBOT DI ĐỘNG

I. GIỚI THIỆU SƠ LƯỢC VỀ ROBOT:

Năm 1921, trên sân khấu kịch nói ở Tiệp Khắc, trong tác phẩm nhan đề "R.U.R" hay "Rossuum's Universal Robot's" của Karel Capek, xuất hiện nhân vật "Robotnik" có nghĩa là công nhân. Nhân vật trong vở diễn mô tả một khối cơ khí rập khuôn hình dáng con người. Sau đó những cỗ máy này được dùng trong chiến tranh và cuối cùng đã quay lại chống loài người đã sáng tạo ra chúng.

Tuy nhiên trước đó người Hy lạp đã chế tạo những tượng đá có thể chuyển động được, đó chính là sự bắt đầu của những gì chúng ta gọi là Robot. Trong phần lớn trường hợp danh từ Robot hiện nay hàm ý những cỗ máy nhân tạo có thể thực hiện những công việc hay những hành động khác thường, được thực hiện bởi con người.

Đa số các Robot ngay nay được sử dụng trong các nhà máy để chế tạo những sản phẩm như xe hơi và điện tử, máy móc công nghiệp. Những loại khác được dùng trong các công việc nghiên cứu, thám hiểm mặt đất, đại dương và trên các hành tinh khác.

Robot thường có 3 phần chính:

- Bộ não(Brain) : Thường là một máy tính, vi xử lý hoặc vi điều khiển.
- Các bộ phận cơ khí và chấp hành (Actuator) : Động cơ, piston, cơ cấu kẹp, bánh răng, bánh xe.
- Cảm biến (Sensor):Thị giác, âm thanh, nhiệt độ, chuyển động, ánh sáng, xúc giác...

Với 3 thành phần này, Robot có thể tương tác và tác động đến môi trường của chúng ta trở nên hữu dụng.

1.1 Sơ lược lịch sử phát triển của Robot :

Sự ra đời của Robot gắn liền với sự phát triển của công nghiệp, khi nền công nghiệp phát triển, đòi hỏi một đội ngũ lao động dồi dào để đảm bảo cho việc sản xuất hàng hoá. Tuy nhiên, nền công nghiệp phát triển rất mạnh trong khi đó lực lượng lao động không thể tăng lên mãi, nên không thể đáp ứng đủ cho nhu cầu sản xuất.

Trước thực trạng này, người ta đã nghiên cứu và chế taọ ra các máy móc để thay thế cho lực lượng lao động con người, tuy vậy cũng cần có lực lượng lao động con người để điều khiển máy móc. Có những công việc lặp đi lặp lại rất nhàm chán, nhưng lại đòi hỏi độ chính xác cao. Để khắc phục tình trạng này, máy móc đã được cải tiến dần dần để có thể tự động thực hiện tốt hơn một số công việc, quá trình cải tiến máy móc theo hướng tự động hoá đã dẫn đến sự ra đời của robot công nghiệp.

Các thành tựu khoa học công nghệ đã dần dần làm cho chi phí lao động khi dùng robot được hạ thấp dần, trong khi đó chi phí lao động cho con người không ngừng tăng lên, điều này đã dẫn đến việc Robot được áp dụng rộng rãi trong sản xuất và các ứng dụng khác.

Sự phát triển của các loại máy móc tự động đã dẫn đến sự ra đời và phát triển của các Robot công nghiệp :

Năm 1801 Joseph Jacquard phát minh ra một máy dệt được điều khiển bằng card đục lỗ. Vào thế kỉ 18 "Búp bê" tự động hoá đã biểu diễn nhạc, cầm bút vẽ và viết các nét chữ. "Búp bê" thổi sáo của nhà cơ học người Pháp Jacques Vocanson (1709 - 1782) các ngón tay bấm giúp sáo thổi được 11 giọng khác nhau.

Nhà chế tạo đồng hồ ở Thuỵ Sĩ là Pier Jacquet Droz (1721 - 1790) và con trai của ông là Henri Droz (1752 - 1791) đã tạo ra nhiều búp bê tự động hoá theo bộ máy đồng hồ mang tên Android.

Augustur Huber (người Áo) đã chế tạo nhiều con rối tự động bằng vô tuyến điện có thể đi, nói và thao tác như con người.

Năm 1830 Christopher Spencer (người Mỹ) đã thiết kế máy tiện đưa vào kỹ thuật cam.

Năm 1892 Ở Mỹ, Seward Babbitt thiết kế một cần trục cơ giới hoá có kẹp để di chuyển các thỏi kim loại ra khỏi lò luyện.

Năm 1921 Robot xuất hiện lần đầu tiên trong trò chơi điện tử ở Luân Đôn. Trò chơi được thiết kế bởi Karel Capek (người Tiệp Khắc) giới thiệu từ Robot trong cụm từ Tiệp robota có nghĩa là một nông nô, nô lệ. Từ đó có khái niệm Robot xuất hiện.

Năm 1938 Willard Pollard Harold Roselund (người Mỹ) thiết kế một cơ cấu sơn phun có thể lập trình được cho công ty De Vibiss.

Năm 1946 Reorge Devol được cấp bằng sáng chế về một thiết bị phản hồi, điều khiển máy móc đa chức năng.thiết bị sử dụng là một thiết bị ghi từ tính.

Trong cùng năm này, máy tính được xuất hiện đầu tiên. Nhà khoa học người Mỹ J.Presper Eckert và Lonh Mauchly xây dựng máy tính điện tử lớn đầu tiên gọi là Eniac ở trường Đại học Pennsylvania. Một máy tính thứ hai được đặt tên là Whirlwind, là máy tính số đa năng đầu tiên được thiết kế ở M.I.T.

Năm 1948 Norbert Wiener, một giáo sư ở M.I.T, xuất bản quyển Cybernetics mô tả khái niệm giao tiếp và điều khiển trong điện tử máy móc và hệ thống sinh vật học.

Năm 1951 Một cánh tay có khớp nối trang bị bộ hoạt động từ xa được thiết kế bởi Raymond Goertz cho Atomic Energy Commission.

Năm 1954 Robot có thể lập trình đầu tiên được thiết kế bởi George Devoil, người đã đưa ra thuật ngữ Universal Automation. Sau đó thuật ngữ này được viết ngắn gọn là Unimation, đó chính là tên của công ty Robot đầu tiên.

Năm 1959 Tập đoàn Planet đưa ra thị trường Robot thương mại đầu tiên.

Năm 1960 Unimation được tập đoàn Condec mua lại và sự phát triển của hệ thống Robot Unimate được bắt đầu.

Tập đoàn AMF (American Machine and Foundry) đưa ra thị trường một Robot Versa Tran, được thiết kế bởi Harry Johnson và Veljiko Milenkovic.

Năm 1962 General Motors lắp đặt Robot công nghiệp đầu tiên vào dây chuyền sản xuất. Đó chính là Robot Unimate, dùng trong chế tạo xe hơi.

Năm 1964 Các phòng nghiên cứu trí tuệ nhân tạo được mở ở M.I.T, viện nghiên cứu Stanford (SRI) , Đại học Stanford vàĐại học Edinburgh.

Năm 1968 SRI xây dựng và thử nghiệm Robot di động có khả năng nhìn được gọi là Shakey.

Năm 1970 Đại học Stanford phát triển một cánh tay Robot làm tiêu chuẩn cho các công trình nghiên cứu, cánh tay hoạt động bằng điện năng và được gọi là cánh tay Stanford.

Đầu những năm 1970 xuất hiện người máy công nghiệp ở dạng thử nghiệm, lúc đầu là những cỗ máy cơ khí bắt chước cánh tay người để nâng đỡ, gối lắp các phụ kiện máy sau đó là những cỗ xe thật sự chạy bằng động cơ cỡ lớn đưa chân ra và giang những cánh tay lớn khỏe, thọc sâu những bàn tay để khoan phá, đào bới xúc và múc những gàu, những phiếm quặng lớn hay nhỏ đưa lên mặt đất, đó là những người máy khai thác mỏ thay thế người, người máy lắp ráp ô tô hay những người máy lập trình trên máy tính điện tử được người thật điều khiển từ xa có khả năng cảm nhận, với bộ não là bộ máy vi xử lý.Vào thập niên 70 này các nhà công nghiệp của các nước khác nhau trên thế giới như Anh, Liên Xô(cũ) , Thụy sĩ, Đức, Nhật, Pháp, Italia...thi nhau chế tạo các người máy công nghiệp ngày càng hoàn thiện hơn.

Đến năm 1972, tổng số người máy trên toàn thế giới đã lên tới khoảng 3000.

Năm 1973 Robot thương mại đầu tiên được điều khiển bằng máy tính mini do Richard Hohn thiết kế cho tập đoàn Cincinnati Milacron.Robot được gọi là T3. The Tomorrow Tool.

Năm 1974 Giáo sư Scheiman, người phát triển cánh tay Stanford thành lập liên hợp Vicarm để đưa ra thị trường một phiên bản cánh tay mới cho các

ứng dụng công nghiệp.Cánh tay mới được điều khiển bằng một máy tính mini.

Năm 1976 Các cánh tay Robot được sử dụng trên các tàu vũ trụ không người lái Viking 1 và Viking 2.

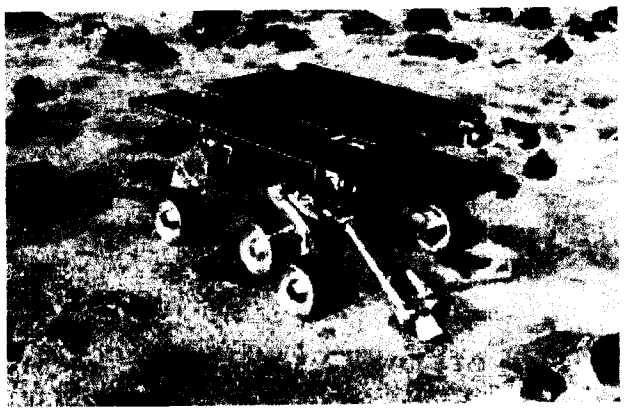
Năm 1977 ASEA, một công ty Robot Châu Âu, đưa ra các Robot công nghiệp dùng điện năng. Chúng sử dụng một máy vi tính để điều khiển, lập trình và hoạt động.

Cùng năm đó, Unimation mua lại liên hợp Vicarmm.

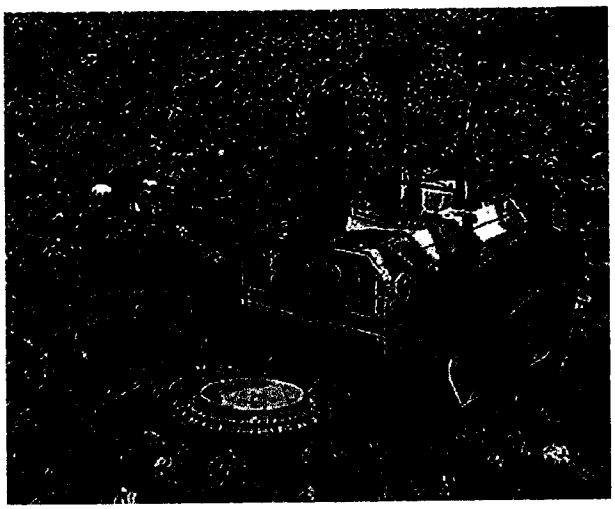
Năm 1978 Robot Puma (Programmable Universal Machine for Assembly) được Unimation phát triển từ kỹ thuật Vicarm.

Năm 1980 Công nghiệp Robot bắt đầu phát triển nhanh với một Robot mới hoặc một công ty Robot mới ra đời vào mỗi tháng. Vào năm này Nhật Bản đứng đầu các nước chế tạo người máy, có khoảng 8000 người máy hoạt động khắp các nhà máy và nhà công nghiệp ở Nhật., đứng thứ hai là Mỹ và Liên Xô (cũ) mỗi nước có khoảng 6000 người máy, các nước Tây âu khi đó có khoảng 4000 người máy.

Một số hình ảnh về robot và các ứng dụng:



H1.1: -Robot tự hành Sojourner thám hiểm sao Hoả



H1.2:-Robot dò mìn

H1.3:-Tay máy dùng trong công nghiệp

1.2 Sự tác động của Robot lên xã hội:

Vì Robot được dùng chủ yếu trong sản xuất, nên chúng ta có thể thấy được sự tác động của chúng lên những sản phẩn chúng ta dùng hằng ngày. Thường thì sự tác động này mang tính tích cực, làm cho sản phẩm trở nên tốt hơn, giá rẻ hơn. Robot cũng được dùng trong những trường hợp mà nó có thể làm tốt hơn con người như giải phẫu, ở đó sự chính xác cao rất cần thiết. Robot còn được dùng trong thám hiểm những nơi nguy hiểm như núi lửa hoặc trong việc dò mìn, điều này cho phép chúng ta hoàn thành công việc mà không gây nguy hiểm cho ta.

Tuy nhiên vẫn có những vấn đề với Robot. Như bất cứ loại máy nào Robot có thể ngưng hoạt động và ngay cả gây tai nạn. Chúng là những cỗ máy mạnh mẽ, con người để cho nó tự điều khiển một vài công việc. Khi có một hành động nào đó đi sai hướng thì những điều nguy hiểm nhất có thể xảy ra. May mắn là điều này ít khi

xảy ra vì hệ thống Robot được thiết kế với nhiều đặc tính an toàn, sẽ giới hạn những thiệt hại có thể.

Ngoài ra, vấn đề sẽ trở nên rất nghiêm trọng nếu Robot được sử dụng vào những mục đích xấu, chống lại con người. Ngày nay điều này đã trở thành hiện thực, với những hình thức khác nhau của kỹ thuật như chế tạo vũ khí và vật liệu sinh học.

1.3 Tương lai của Robot :

Số lượng Robot đang gia tăng nhanh chóng. Dẫn đầu về số lượng Robot là Nhật Bản (gần gấp đôi so với Mỹ). Mọi sự phỏng đoán đều cho rằng Robot sẽ gia tăng không ngừng trong xã hội hiện đại. Chúng sẽ tiếp tục được dùng trong những công việc nguy hiểm, những công việc có tính lặp lại, giảm chi phí hay đòi hỏi sự chính xác mà con người khó thực hiện được. Với mức sống ngày càng cao của xã hội hiện đại, Robot dần dần đã xâm nhập vào tư gia của mỗi người với vai trò của người máy giúp việc, hay các vật nuôi như mèo máy, chó Robot ...do đó nhu cầu về Robot không bao giờ suy giảm mà luôn phát triển đa dạng muôn hình muôn vẻ.

II. PHƯƠNG PHÁP THIẾT KẾ ROBOT DI ĐỘNG:

Robot là một sản phẩm tổng hợp của nhiều ngành kỹ thuật như cơ khí, điện tử, máy tính và trí tuệ nhân tạo. Trong đó phân biệt ra nhiều loại Robot khác nhau, và phương pháp thiết kế mỗi loại Robot cũng khác nhau. Đối với các Robot thuộc dạng tay máy công nghiệp thì sự nhỏ gọn, tính thẩm mĩ không phải là sự quan tâm hàng đầu, bù lại khả năng tính toán của nó rất lớn do được nối kết với các trạm máy tính cố định. Nhưng đối với Robot chuyển động thì khác, thiết kế Robot chuyển động là thiết kế các bộ phận truyền động, các hệ thống điều khiển, hệ thống cảm biến, nguồn cấp điện ổn định, các chương trình phần mềm tin cậy và nhỏ gọn. Tất cả các thành phần này phải được gắn trong một bộ khung gọn nhẹ, thích hợp để Robot vừa có thể chuyển động vừa thực hiện nhiệm vụ của nó một cách hiệu quả.

Cấu trúc của một Robot chuyển động thường gồm hai phần : phần cứng và phần mềm, ngoài ra còn có hệ thống cơ khí làm bộ phận chuyển động và khung để chứa các thành phần bên trong của Robot. Phần cứng của Robot là thuật ngữ chỉ các mạch điện tử trong Robot, quyết định tiềm năng của Robot. Nhưng để khai thác các tiềm năng này, Robot cần phải có phần mềm là các chương trình điều khiển Robot, giao tiếp bộ vi xử lý với các phần tử ngoại vi như cảm biến, các bộ phận chấp hành...phần cứng và phần mềm quan hệ chặt chẽ với nhau để tạo nên một Robot hữu dụng.

Thành phần chính trong phần cứng của Robot là bộ xử lý mà thông dụng là vi xử lí . Tuy nhiên các bộ vi xử lý đa phần có các tập lệnh tập trung vào việc xử lý dữ liệu, rất ít các lệnh vào, ra nên khả năng tương tác với các ngoại vi là rất thấp. Để giao tiếp với ngoại vi cần có thêm các mạch điện phụ, làm cho phần cứng trở nên

cồng kềnh hơn,và hoàn toàn không thích hợp cho các Robot chuyển động nhỏ luôn phải manh máy tính trên mình. Do đó sự ra đời của các họ vi điều khiển tích hợp cao đã đem lại rất nhiều tiện dụng

CHƯƠNG 2 : *NỘI DUNG ĐIỀU KHIỂN ROBOT*

I. NHIỆM VỤ ĐỀ TÀI :

- Nhiều loại Robot hoạt động theo các yêu cầu khác nhau,điều này tuỳ theo nhu cầu của mỗi người.Với đề tài luận án này có nhiệm vụ điều khiển Robot chạy theo hành trình.

II. YÊU CẦU KỸ THUẬT HOẠT ĐỘNG THEO HÀNH TRÌNH :

2.2.1 Yêu cầu:

- Là một Robot hoạt động theo hành trình cho nên điều quan trọng là khả năng chạy đúng lộ trình,từ đó yêu cầu đặt ra cần thiết là phải có phần dò đường cho Robot.Vì chỉ có phương pháp dò đường bám theo lộ trình thì Robot mới không lái sai đường.Đây cũng là phần quan trọng trong yêu cầu thiết kế Robot hành trình.

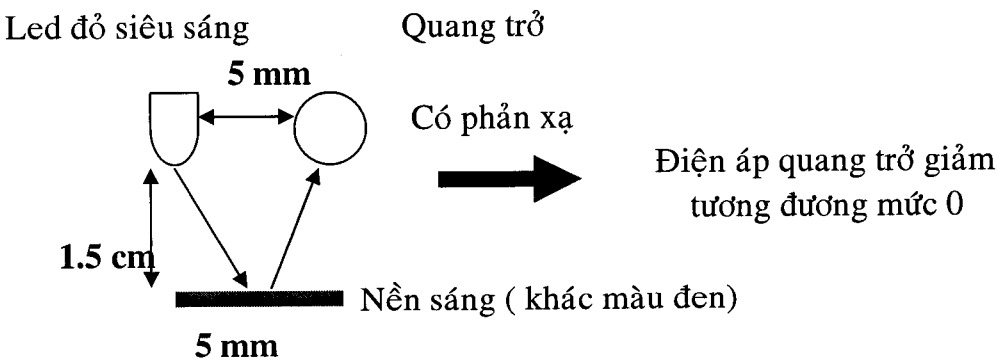
2.2.2 Phương án giải quyết:

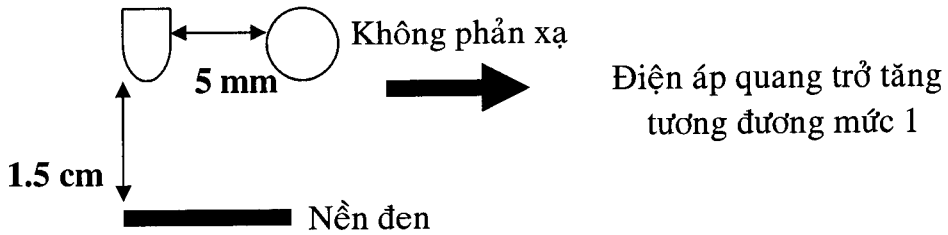
- Với yêu cầu đặt ra cho việc điều khiển hành trình,ta sẽ đưa ra phần dò đường bằng cảm biến như sau:

2.2.2.1 Nguyên tắc hoạt động cảm biến:

- Khi có ánh sáng thì điện áp quang trở giảm, tương đương với mức 0 (điện áp bằng 0V). Khi không có ánh sáng chiếu vào thì điện áp quang trở tăng, tương đương với mức 1 (điện áp bằng 5V).Áp dụng tính chất trên của quang trở nên ta thiết kế phương pháp dò đường như sau:

Sử dụng vạch đường là màu trắng. Khi có ánh sáng chiếu xuống nền đường (nền có màu khác màu đen) thì ánh sáng sẽ phản xạ dẫn đến quang trở nhận được ánh sáng (điện áp bằng 0V). Khi ánh sáng chiếu xuống vạch đường có nền màu đen thì quang trở sẽ không nhận được ánh sáng (điện áp bằng 5V), vì trên nền đen khi có ánh sáng chiếu sẽ không phản xạ.Sơ đồ cụ thể:

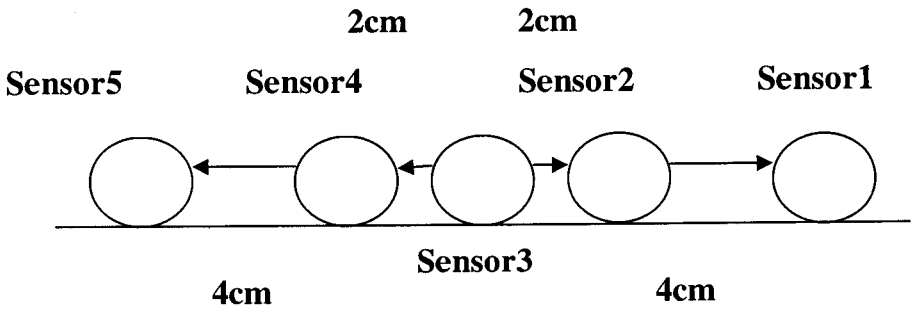




Khi có tín hiệu từ quang trở, mạch sẽ điều chỉnh mức điện áp hoặc là mức 1, hoặc là mức 0. Bằng cách sử dụng các cổng logic như: IC LM339 và 4093. Mạch cảm biến sẽ đưa tín hiệu này về mạch điều khiển để xử lý.

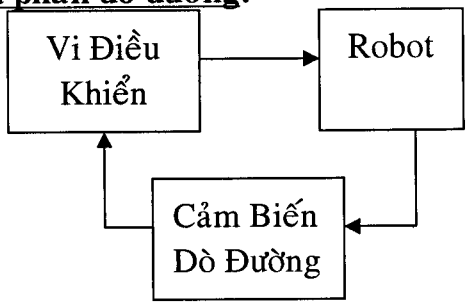
2.2.2.2 Phương pháp dò đường:

Ta sẽ bố trí 5 cặp cảm biến như sau:



- Khi Robot chạy đúng đường thì cảm biến 3 sẽ không phản xạ,các cảm biến còn lại đều bị phản xạ trở lại,vậy cảm biến ba cho biết Robot chạy đúng đường
- Cảm biến hai và một cho biết Robot đang lệch về phía bên phải
- Cảm biến bốn và năm cho biết Robot đang lệch về phía bên trái

2.2.3 Sơ đồ khối của phần dò đường:



- Robot liên tục truyền tín hiệu của cảm biến dò đường về vi điều khiển kiểm tra.Do lúc chạy có thể gặp những nhân tố khách quan mà Robot có thể bị lệch đường,vì thế các cảm biến dò đường được bố trí để kiểm soát liên tục trạng thái của Robot.Như vậy chỉ cần Robot chạy lệch hướng là sẽ được điều chỉnh lại ngay.

CHƯƠNG 3: TỔNG QUAN VỀ LÝ THUYẾT ĐIỀU KHIỂN ĐỘNG CƠ

I.KHÁI QUÁT VỀ ĐỘNG CƠ BƯỚC:

3.1.1.Khái niệm chung:

Các hệ truyền đạt rời rạc thực hiện nhờ động cơ thực hiện theo từng chu trình điều khiển đặc biệt gọi là động cơ bước.

Động cơ bước là một động cơ đồng bộ dùng để biến đổi các tín hiệu điều khiển dưới dạng các xung điện rời rạc kế tiếp nhau thành các chuyển động góc quay hoặc các chuyển động của rôto đến những vị trí cần thiết.

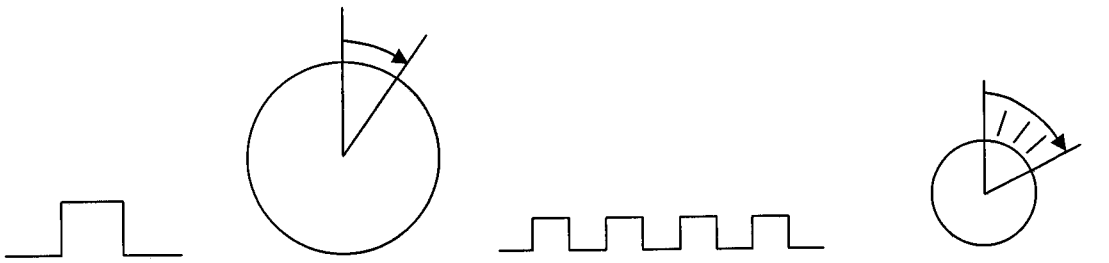
Động cơ bước làm việc được là nhờ có bộ chuyển mạch điện tử đưa các tín hiệu điều khiển vào stato theo một chu kỳ tuần tự với tần số nhất định. Tổng số góc quay của rôto tương ứng với tần số chuyển mạch, cũng như chiều quay và tốc độ quay của rôto, phụ thuộc vào thứ tự chuyển đổi và tần số chuyển đổi. Khi một xung điện áp đặt vào cuộn dây stato (phần cố định) của động cơ sẽ quay đi một góc nhất định, góc ấy là một bước quay của động cơ. Khi các xung điện áp đặt vào các cuộn dây phần ứng thay đổi liên tục thì rôto sẽ quay liên tục.

Về cấu tạo, động cơ bước có thể coi là tổng hợp của hai động cơ: động cơ một chiều không tiếp xúc và động cơ đồng bộ điều tốc công suất nhỏ.

Trong khi động cơ một chiều không tiếp xúc rôto thường là một nam châm vĩnh cửu, số đôi cực 2p = 2 và cần có một cảm biến vị trí rôto để thực hiện chức năng tạo ra tín hiệu điều khiển nhằm xác định thời điểm và thứ tự đổi chiều thì động cơ bước có rôto dạng cực lồi gồm nhiều răng cách đều cấu thành các cặp nam châm N–S xen kẽ nhau để tạo ra số cặp cực 2p lớn hơn và không cần phải có bộ giảm biến vị trí rôto. Nhờ cảm biến vị trí rôto, có thể điều khiển dòng một chiều vào các cuộn dây stato để có từ trường quay liên tục nên động cơ một chiều không tiếp xúc quay liên tục. Đối với động cơ bước, vì từ trường quay không liên tục do các xung điện cấp vào rời rạc nên rôto quay theo các bước.

Cũng giống như động cơ đồng bộ giảm tốc công suất nhỏ, động cơ bước có các cuộn dây tạo thành các pha trên stato, đồng thời trên cả stato và rôto đều có các răng để tạo thành các cặp cực và các nam châm điện. Nhưng động cơ đồng bộ giảm tốc cần có các cuộn kích thích và cần phải có dòng điện kích thích để khởi động, còn động cơ bước không cần yếu tố này. Mặt khác, trên stato của động cơ đồng bộ giảm tốc, ngoài cuộn dây phụ (để kích thích) thì cuộn dây chính thường là ba pha hoặc hai pha được nuôi bằng nguồn xoay chiều tạo ra từ trường quay liên tục với vận tốc góc ω.Vì vậy sau khi hoàn thành việc khởi động, rôto quay với vận tốc đồng bộ (nhỏ hơn vận tốc của từ trường quay). Trong khi đó, stato của động cơ bước chỉ có một loại cuộn dây pha và chúng có vai trò như nhau.

Theo một phương diện khác, có thể coi động cơ bước là linh kiện (hay dụng cụ) số (Digital Device) mà ở đó các thông tin số đã thiết lập sẽ được chuyển thành động cơ quay theo từng bước. Động cơ bước sẽ thực hiện theo các lệnh đã số hóa mà máy tính yêu cầu (xem hình 3.1).



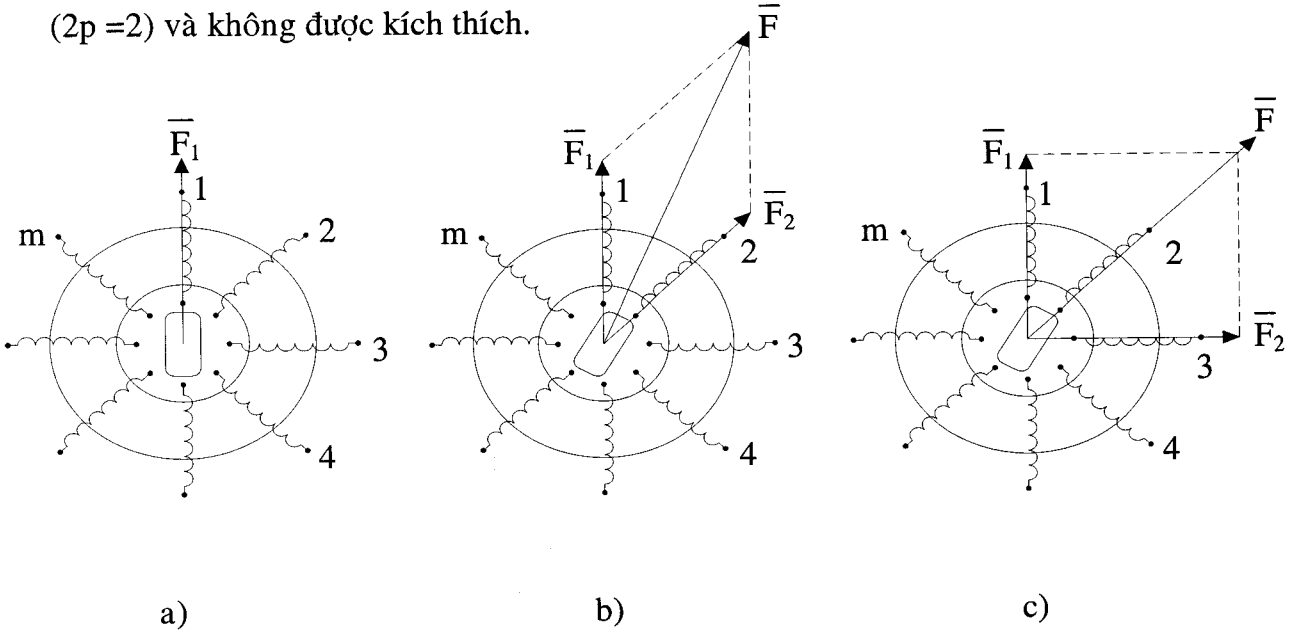
a). Một xung bằng một bước *b).Số xung bằng số bước (4 xung – 4 bước)*

Hình 3.1- Cách thức hoạt động theo xung của động cơ bước.

3.1.2.Nguyên lý hoạt động:

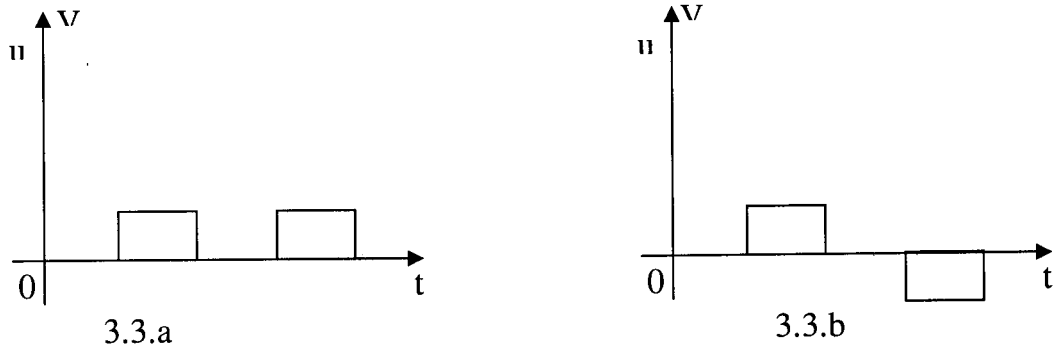
Khác với động cơ đồng bộ bình thường, rôto của động cơ bước không có cuộn dây khởi động mà nó được khởi động bằng phương pháp xung kích với tần số nhất định. Rôto của động cơ bước có thể kích thích (rôto tích cực) hoặc không được kích thích (rôto thụ động).

Hình 3.2 vẽ sơ đồ nguyên lý động cơ bước m pha với rôto có 2 cực (2p =2) và không được kích thích.



Hình 3.2- Sơ đồ nguyên lý động cơ bước m pha với rôto 2 cực và các lực điều khiển bằng xung 1 cực

Xung điện áp cấp cho m cuộn dây stato có thể là xung một cực (hình 3.3.a) hoặc xung 2 cực (hình 3.3.b)



Hình 3.3- Xung điện áp cấp cho cuộn dây stato
3.3.a) Xung một cực 3.3. b) Xung hai cực

Chuyển mạch điện tử có thể cung cấp điện áp điều khiển cho các cuộn dây stato theo từng cuộn dây riêng lẻ hoặc theo từng nhóm các cuộn dây. Trị số và chiều của lực từ tổng F của động cơ và do vị trí của rôto trong không gian hoàn toàn phụ thuộc vào phương pháp cung cấp điện cho các cuộn dây.

Ví dụ, nếu các cuộn dây của động cơ trên hình 3.2 được cấp điện cho từng dây riêng lẻ theo thứ tự 1, 2, 3, ...m, bởi xung 1 cực, thì rôto của động cơ có m vị trí ổn định trung với trục của các cuộn dây (hình 3.2.a).

Trong thực tế để tăng cường lực điện từ tổng của stato và do đó tăng từ thông và mômen đồng bộ, người ta thường cấp điện đồng thời cho hai, ba hoặc nhiều cuộn dây. Lúc đó rôto của động cơ bước sẽ có vị trí cân bằng (ổn định) trùng với vectơ lực điện từ tổng F. Đồng thời lực điện từ tổng F cũng có giá trị lớn hơn lực điện từ thành phần của các cuộn dây stato (hình 3.2.b và 3.2.c).

Trên hình 3.2.b vẽ lực điện từ tổng F khi cung cấp điện đồng thời cho một số chẵn cuộn dây (2 cuộn dây). Lực điện từ tổng F có trị số lớn hơn và nằm ở vị trí chính giữa hai trục của hai cuộn dây. Trên hình 3.2.c vẽ lực điện từ tổng F khi cấp điện đồng thời cho một số lẻ cuộn dây (3 cuộn dây). Lực điện từ tổng F nằm trùng với trục của một cuộn dây nhưng có trị số lớn hơn. Trong cả hai trường hợp (cấp điện cho một số chẵn cuộn dây và cho một số lẻ cuộn dây), rôto của động cơ bước sẽ có m vị trí cân bằng. Góc xê dịch giữa hai vị trí liên tiếp của rôto bằng $2\pi/m$.

Nếu cấp điện theo thứ tự một số chẵn cuộn dây, rồi một số lẻ cuộn dây (ví dụ, kết hợp giữa hình 3.2.b và 3.2.c), có nghĩa là số lượng cuộn dây được điều khiển luôn luôn thay đổi từ chẵn sang lẻ và từ lẻ sang chẵn thì số vị trí cân bằng của rôto sẽ tăng lên gấp đôi là 2m, độ lớn của một bước sẽ giảm đi một nửa bằng $2\pi/m$. Trường hợp này được gọi là điều khiển không đối xứng, hay điều khiển nửa bước (half Step).

Nếu số lượng cuộn dây được điều khiển luôn luôn không đổi (một số chẵn cuộn dây hoặc một số lẽ cuộn dây, ví dụ hình 3.2.b hoặc hình 3.2.c) thì rôto có m vị trí cân bằng và được gọi là điều khiển đối xứng, hay điều khiển cả bước (Full Step).

Một cách tổng quát, số bước quay của rôto trong khoảng $0 \div 360^0$ là:

$$K = m \cdot n_1 \cdot n_2 \cdot p \qquad (1.1)$$

Trong đó:

p: số đôi cực của rôto;

m: số cuộn dây điều khiển trên stato (số pha) ;

n_1: hệ số, $n_1 = 1$ ứng với điều khiển đối xứng;

$n_1 = 2$ ứng với điều khiển không đối xứng;

n_2: hệ số, $n_2 = 1$ điều khiển bằng xung 1 cực;

$n_2 = 2$ điều khiển bằng xung 2 cực.

Bước quay của rôto trong không gian là $\alpha = 360^0/K$.

Trường hợp riêng công thức (1.1) không đúng (sẽ đề cập sau).

3.1.3.Momen đồng bộ và trạng thái ổn định của động cơ bước :

Biểu thức mômen đồng bộ tĩnh của động cơ bước khác với biểu thức mômen đồng bộ của động cơ đồng bộ thường có cùng cấu trúc. Sự khác nhau đó là do động cơ bước được cấp bởi dòng điện một chiều, chứ không phải dòng điện xoay chiều.

Trong động cơ được cấp bởi nguồn một chiều, sau quá trình quá đô, dòng điện trong các cuộn stato là hằng số

$$I = \frac{U}{r_S} = \text{hằng số} \qquad (A)$$

Từ thông do nó sinh ra tác động vào cặp cực của rôto:

$$\Phi = B.S.\cos\alpha = B.S.\sin\theta \qquad (1.2)$$

Trong đó:

B :cường độ từ trường do dòng điện I sinh ra trong cuộn dây có điện cảm L. Đơn vị : Tesla (T)

S :tiết diện vuông góc của cặp cực.

α :góc giữa trục cặp cực với trục của cuộn dây pha.

θ :góc giữa trục cặp cực và đường vuông góc với trục của cuộn dây pha.

Do góc θ thay đổi nên từ dẫn dọc theo đường đi của từ thông thay đổi theo, làm cho từ thông thay đổi trong một giới hạn rộng. Nếu viết biểu thức Φ dưới dạng:

$$\Phi = L \cdot I \text{ (L là điện cảm)} \qquad (T) \qquad (1.3)$$

thì sự thay đổi của góc θ cùng vơi sự thay đổi khe hở không khí giữa các răng của stato và rôto làm cho điện cảm L thay đổi theo dẫn đến từ thông Φ và do đó mômen của động cơ thay đổi.

Mômen đồng bộ tĩnh của động cơ bước khi stato và rôto có răng và được kích thích có thể viết dưới dạng tổng của 3 mômen quay.

$$M(\theta) = M_S + M_R + M_{SR} = C_M.I_S.\frac{dL_S}{d\theta} + C_M.I_R.\frac{dL_R}{d\theta} + 2C_M.I_{SR}.\frac{dL_{SR}}{d\theta} \quad (1.4)$$

Trong đó: I_S, I_R, $L_S(\theta)$ là các trị số xác định xác lập tương ứng của dòng điện, điện cảm và hỗ cảm của stato (S) và rôto (R); CM là hằng số, phụ thuộc vào cấu tạo của từng loại động cơ.

Mômen M_S được hình thành do sự thay đổi từ dẫn của khi hở không khí trên đường đi của từ thông stato được kích thích bởi dòng I_S.

M_S được hình thành do sự thay đổi từ dẫn trên đường đi của từ thông rôto được kích thích bởi dòng điện I_R.

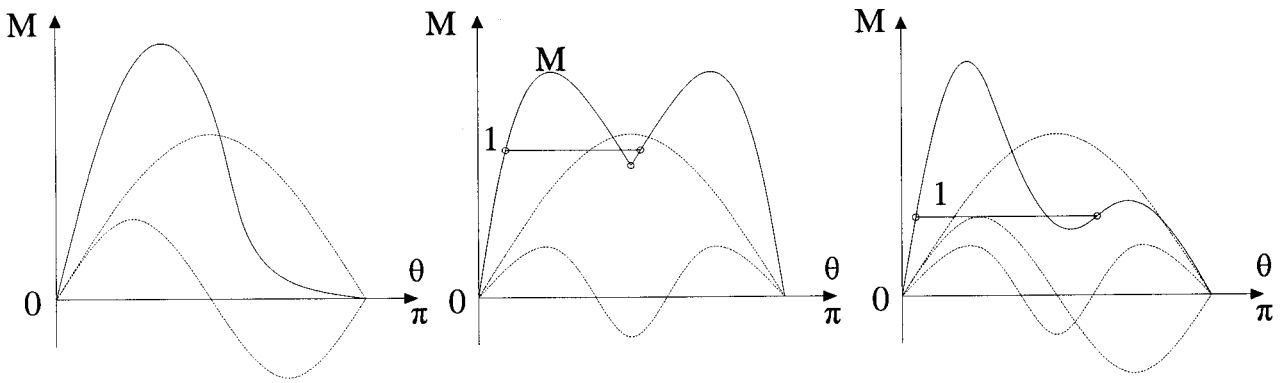
M_{SR} được hình thành do sự thay đổi hổ cảm giữa stato và rôto được kích thích bởi dòng điện I_S và I_R.

Hiện nay, phần lớn các động cơ bước đều có cấu tạo rôto không có cuộn kích thích. Do đó biểu thức mômen có dạng đơn giản sau:

$$M(\theta) = C_M.I_S.\frac{dL_s}{d\theta} \quad (1.5)$$

Quan hệ giữa mômen với góc quay θ thường là không hình sin do ảnh hưởng của cấu trúc răng và cấu trúc cực lồi của stato và của rôto, cũng như do xung điện áp cấp vào có dạng xung vuông là tổng của các thành phần điều hòa. Do đó quan hệ phức tạp này chỉ có thể biểu diễn được nhờ chuỗi số Fourier và nó có dạng tổng của các thành phần điều hòa, thậm chí với vận tốc góc thành phần không phải là bội số lần của vận tốc góc thành phần cơ bản.

Trên hình vẽ 4 vẽ đường cong mômen tổng M = f(θ) (đường nét liền) và các thành phần điều hòa của nó (đường nét đứt).



Hình 3.4- Đường cong mômen của động cơ bước theo góc quay θ

Chất lượng của động cơ bước được đánh giá bởi tốc độ dốc của đường cong mômen đồng bộ M = f(θ), đặt biệt là ở đầu của vùng làm việc thuộc đường cong này (phần đậm nét của đường cong M = f(θ) trên hình 4). Độ dốc của đường M = f(θ)

trong vùng này càng lớn thì suất mômen dM/d θ càng lớn và khả năng đồng bộ của động cơ bước càng lớn.

Giả sử tại một vị trí nhất định, rôto mạng một mômen tải (mômen cản) M_C . Để giữ được rôto ở vị trí này ta phải cấp dòng điện cho cuộn dây stato tại vị trí đó và do góc α =0 nên M = M_{max} . Điều kiện để giữ được rôto không trượt khỏi vị trí là:

$$M_C < M_{max} \qquad (1.6)$$

Muốn quay rôto đi một góc α rời khỏi vị trí đang giữ, ta phải cấp dòng điện cho cuộn stato ở vị trí mới, đồng thời ngắt dòng điện của cuộn stato ở vị trí cũ. Điều kiện để rôto quay được góc α là:

$$M_C < M_{max} . \cos\alpha \qquad (1.7)$$

Cần chú ý rằng sự chuyển bước của rôto chỉ thực hiện được trong điều kiện nhất định, khi mà sự dịch chuyển của dòng điện từ F đi một góc α không làm cho động cơ rơi vào vùng mất ổn định của đặc tính góc M = f(θ) (điểm 2 ở đường cong trên hình 3.4).

Nếu như góc α quá lớn thì rôto rơi vào vùng mất ổn định, không bám theo được từ trường và động cơ bị mất bước.

Trong trường hợp tổng quát, để động cơ không bị mất bước cần thực hiện điều kiện sau:

$$M_C < M_{max} . \cos(2\pi/K), \qquad (1.8)$$

Trong đó: K là số bước quay (từ công thức (1.1)).

Như vậy, bước quay α = 360^0/K càng lớn thì mômen tải M_C cho phép trên trục động cơ càng lớn.

3.1.4.Cấu tạo và phân loại động cơ bước:

Để tăng số bước của động cơ (tăng độ phân giải về góc), theo công thức (1.1), cần phải tăng số lượng cuộn dây pha m và tăng số cặp cực p.

Việc tăng số lượng bối dây m trên stato gặp nhiều khó khăn do hạn chế về kích thước của stato và những trở ngại khi đặt các bôbin dây quấn vào các rãnh nửa hở của stato, đồng thời khi số pha m lớn hơn thì mạch điều khiển cũng sẽ phức tạp hơn. Do đó người ta thường làm các động cơ bước với số lượng pha m đủ nhỏ, là 2 pha, 4 pha và 5 pha; trong đó thông dụng nhất là 2pha và 4 pha.

Việc tăng số bước của động cơ cuối cùng được giải quyết bằng tăng số lượng cặp cực rôto. Rôto động cơ bước tạo thành nhiều cặp cực được chế tạo từ vật liệu kỹ thuật đặc biệt có từ hóa cao và chịu được mômen tải lớn, vì chính rôto là bộ phận chịu tải trọng cơ khí thông qua trục của nó.

Người ta thường chế tạo các động cơ bước có các góc bước trong khoảng tầ 0,45 ÷ 15^0 tùy theo mục đích sử dụng. Trong đó thông dụng nhất trên thị trường hiện nay là loại động cơ bước có góc bước $1,8^0$ (ứng với 200 bước trong 1 vòng quay 360^0)

Xét về cấu tạo, động cơ bước có 3 loại chính:

❖ Động cơ bước có rôto được kích thích (có dây quấn kích thích hoặc kích thích bằng nam châm vĩnh cửu).

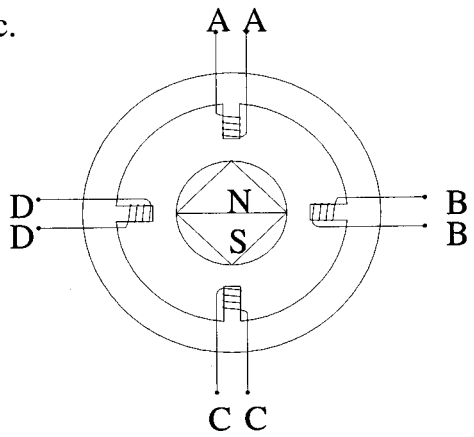
❖ Động cơ bước có rôto không kích thích (động cơ kiểu cảm ứng và động cơ kiểu phản kháng).

❖ Động cơ bước hỗn hợp, kết hợp cả hai loại trên.

3.1.4.1.Động cơ bước nam châm vĩnh cửu:

Thường được cấu tạo với stato có dạng hình móng được từ hóa với cực N và S xen kẽ nhau; rôto thường có răng, được từ hóa vĩnh cửu vuông góc với trục (ngang trục). Loại động cơ này có góc bước trong khoảng $6^0 \div 45^0$, tốc độ chậm nhưng có mômen khá lớn.

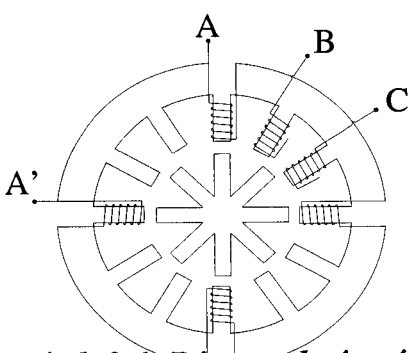
Hình 1.5 là sơ đồ cấu tạo của động cơ bước nam châm vĩnh cửu với m = 4 và 2p = 2; Hình 1.2 (mục 1.2 chương này) vẽ sơ đồ nguyên lý của động cơ loại này với m pha và m cặp cực.



Hình 3.5- Động cơ bước nam châm vĩnh cửu.

3.1.4.2.Động cơ bước có từ trở thay đổi:

Còn gọi là động cơ phản kháng. Cả stato và rôto đều có răng. Rôto được làm bằng vật liệu dẫn từ (sắt non) có từ trở thay đổi theo góc quay. Mỗi răng của stato và rôto gọi là một cực. Mỗi pha trên stato được quấn thành 2 cuộn nối tiếp nhau ở vị trí xuyên tâm đối trên stato, thậm chí thành 4 cuộn đôi một trực giao (hình 3.6).



Hình 3.6- Động cơ bước có từ trở thay đổi.

Gọi N_R là số răng của rôto, N_P là số pha của stato, góc bước của động cơ được tính theo công thức:

$$\alpha = \frac{360^0}{N_R.N_P} \qquad (1.9)$$

Động cơ vẽ trên hình 6 có góc bước là 15^0, vì có số pha là 3 và số răng rôto là 3.

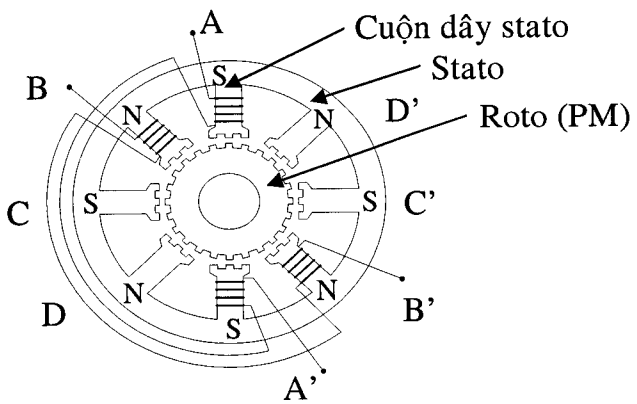
Góc bước của động cơ loại này thường từ $1,8 \div 30^0$.

Chiều quay của động cơ không phụ thuộc vào chiều của dòng điện mà chỉ phụ thuộc vào thứ tự cấp dòng điện trong các cuộn dây. Do đó, trong công thức (1.1) thì $n_1 = 1$, đối với loại động cơ này.

Nhìn chung loại động cơ này có số bước lớn, tần số làm việc khá cao và chuyển động êm nhưng mômen đồng bộ nhỏ.

3.1.4.3.<u>Động cơ bước kiểu hỗn hợp (Hybrid)</u> :

Về cấu tạo, nó kết hợp cả hai loại động cơ trên. Về tính chất, nó phát huy được các ưu điểm của cả động cơ nam châm vĩnh cửu và động cơ phản kháng: có mômen hãm (khi ngắt điện), có mômen giữ và mômen quay lớn, hoạt động với tốc độ cao và có số bước lớn (góc bước trong khoảng từ $0,45^0 \div 5^0$).

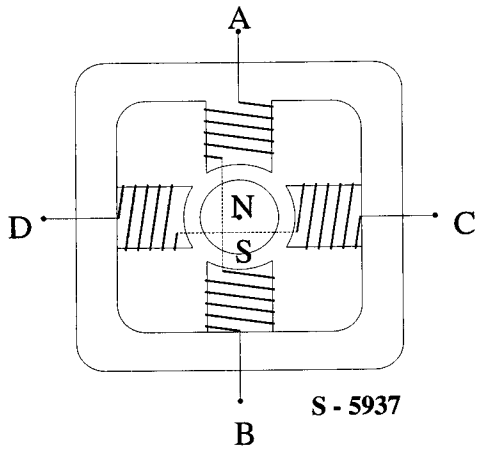


Hình 3.7- Động cơ bước kiểu hỗn hợp

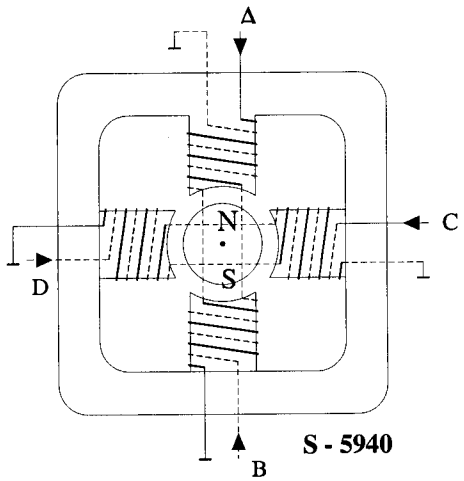
Hiện nay trênt hị trường cómặt chủ yếu là động cơ loại này với cấu tạo 2 pha hoặc 4 pha, góc bước $1,8^0$.

Trên phương diện dòng điện điều khiển, động cơ nam châm vĩnh cửu có thể phân loại làm 2 loại: động cơ đơn cực (điều khiển bằng dòng điện đơn cực) và động cơ lưỡng cực (điều khiển bằng dòng điện lưỡng cực). Về lượng pha trên stato, động cơ được phân thành loại 2 pha, 4 pha và nhiều pha.

Dưới đây vẽ biểu kiến cấu tạo động cơ nam châm vĩnh cửu loại đơn cực và lưỡng cực.



Hình 3.8- Động cơ bước nam châm vĩnh cửu 2 pha kiểu lưỡng cực.
Hai cuộn dây pha đối xứng, vuông góc với nhau và được quấn thành hai phần ở vị trí xuyên tâm đối trên stato.



Hình 3.9- Động cơ bước nam châm vĩnh cửu 4 pha kiểu đơn cực.
Bốn cuộn dây pha đôi một trực giao và đôi một quấn kép lồng vào nhau, mỗi cuộn quấn 2 phần ở vị trí xuyên tâm đến trên stato.

II.CƠ SỞ LÝ THUYẾT ĐIỀU KHIỂN ĐỘNG CƠ BƯỚC:

3.2.1.Chế độ điều khiển động cơ bước:

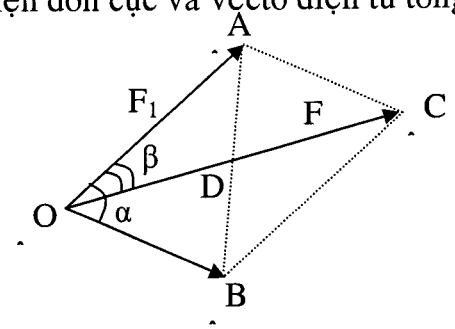
Trong mục II chương I (nguyên lý hoạt động của động cơ bước) đã đề cập đến hai chế độ điều khiển động cơ bước: chế độ một bước và nửa bước.

Trở lại hình 2 vẽ nguyên lý hoạt động của động cơ bước. Xét về bản chất các lực điện từ, có thể điều khiển rôto cố định vào một vị trí bất kỳ trong khoảng không

gian của một bước, chẳng hạn giữa cuộn 1 và cuộn dây 2. Sau đây đưa ra cơ sở lý thuyết của việc điều khiển này.

3.2.1.1.<u>Biểu thức toán học tổng quát cho 3 chế độ điều khiển</u>:

Hình 8 vẽ mối quan hệ giữa các vectơ lực điện từ F_1, F_2 của hai cuộn dây 1 và 2 khi được cấp dòng điện đơn cực và vectơ điện từ tổng F.



Hình 3.10- Giản đồ nguyên lý các lực điện từ khi điều khiển ở chế độ vi bước.

Trên hình 3.10:

F_1: lực điện từ tác động lên rôto khi cuộn dây 1 được kích thích;

F_2: lực điện từ tác động lên rôto khi cuộn dây 2 được kích thích;

F: lực điện từ tổng;

α: góc bước;

β: góc cần điều khiển ;

Nếu ta điều khiển sao cho F_1 và F_2 có trị số không bằng nhau thì lực điện từ tổng F sẽ có hướng thay đổi trong khoảng góc bước α (Từ cạnh OA đến cạnh OB của tam giác OAB – hình 8) và do đó vị trí của rôto thay đổi được và có thể cố định vào vị trí bất kỳ trong khoảng góc bước α.

Gọi β là góc vi bước tạo bởi vectơ F_1 và F , áp dụng tính chất của hình bình hành (OACB) và các hệ thức lượng trong tam giác cho các tam giác OAB, OAC, và OAD (hình 8), ta có các biểu thức sau:

$$AB^2 = OA^2 + OB^2 - 2\,OA.OB.\cos\alpha \qquad (2.1)$$

$$AC^2 = OA^2 + OB^2 - 2\,OA.OB.\cos\alpha \qquad (2.2)$$

$$(AB/2)^2 = OA^2 + (OC/2)^2 - 2\,OA(OC/2).\cos\beta \qquad (2.3)$$

hay: $AB^2 = 4\,OA^2 + OC^2 - 4\,OA.OC.\cos\alpha\beta$

Từ (2.1), (2.2) và (2.3) suy ra:

$$\cos\beta = \frac{OA + OB.\cos\alpha}{\sqrt{OA^2 + OB^2 + 2OAOB.\cos\alpha}} \qquad (2.4)$$

$$\cos\beta = \frac{F_1 + F_2.\cos\alpha}{\sqrt{F_1^{\,2} + F_2^{\,2} + 2F_1.F_2.\cos\alpha}} \qquad (2.5)$$

Từ công thức tổng quát (2.5) suy ra các trường hợp sau đây:

a)Điều khiển cả bước:

❖ Đầu tiên cho $F_2 = 0$ và $F_1 = F$, $\cos\beta = F/F = 1$ nên $\beta = 0$, rôto ở vị trí trục cuộn dây 1.

❖ Sau đó cho $F_1=0$ và $F_2 = F$, $\cos\beta = \cos\alpha$ hay $\beta = \alpha$, rôto ở vị trí trục cuộn dây 2.

b)Điều khiển nửa bước:

❖ Đầu tiên cho $F_2=0$ và $F_1 = F$, rôto ở vị trí trục cuộn dây 1.

❖ Tiếp theo cho $F_1= F_2=F$,

$$\cos\beta = \frac{1+\cos\alpha}{\sqrt{2(1+\cos\alpha)}} = \frac{2\cos^2(\alpha/2)}{\sqrt{4\cos^2(\alpha/2)}} \qquad (2.6)$$

$\cos\beta=\cos(\alpha/2)$, $\beta= \alpha/2$; rôto sẽ ở vị trí trục của cuộn dây thứ 2. Trong trường hợp này rôto sẽ chuyển động từng bước $\theta = \alpha/2$. ($\beta=0, \alpha/2, \alpha$)

c)Điều khiển vi bước:

Nếu ta điều khiển sao cho lực F_1 giảm dần theo từng bước từ F đến 0 và lực F_2 tăng dần từng bước từ 0 đến F thì rôto sẽ quay từng bước từ vị trí OA đến OB.

Trên thực tế, để rôto có thê quay được các bước đều, chẳng hạn theo vi bước θ (ví dụ $\theta = \alpha/10$), thì phải giải phương trình (11) để tìm F_1 và F_2 ứng với các góc quay $\beta = \theta, 2\theta, 3\theta, ..., n\theta$. Phương trình (11) là một phương trình bậc 2 với hai ẩn số, về nguyên tắc là không giải được. Nhưng với điều kiện biên nhất định phương trình (11) sẽ giải được. Sau đây đề xuất hai phương pháp điều khiển ứng với việc chọn hai điều kiện biên khác nhau.

3.2.1.2.Phương pháp điều khiển động cơ bước ở chế độ vi bước:

Ta biết lực điện từ của các cuộn dây stato (F_1 và F_2 v.v...) tỷ lệ thuận với cường độ từ trường và do đó tỷ lệ thuận với dòng điện cấp cho các cuộn dây pha (I_1, I_2, v.v...). Vì vậy để điều khiển được hướng và độ lớn lực điện từ tổng F ta cần điều khiển đồng bộ hai dòng điện I_1 và I_2.

Để cho tiện, trong phương trình (11) ta sẽ ký hiệu $x = F_1$ hoặc $x = I_1$ và $y = I_2$ chung cho cả hai trường hợp

$$\cos\beta = \frac{x + y\cos\alpha}{\sqrt{x^2 + y^2 + 2xy\cos\alpha}} \qquad (2.7)$$

tính lực điện từ và tính dòng điện điều khiển. Lúc đó ta có phương trình:

a)Phương pháp 1:

Ta phân tích góc bước α làm hai miền: miền thứ nhất $0 \le \beta \le \alpha/2$; miền thứ hai $\alpha/2 \le \beta \le \alpha$. Gọi giá trị dòng điện lớn nhất cho phép đối với các cuộn dây là I_M (giá trị này được cung cấp theo Catalog cho mỗi loại động cơ).

Trong miền thứ nhất ta giữ nguyên $I_1 = I_M$ và điều khiển I_2 tăng dần theo từng bước từ 0 đến I_M, vectơ lực F sẽ quay góc β theo từng bước θ từ giá trị 0 đến α/2.

Trong miền thứ hai ta giữ nguyên $I_2 = I_M$ và giảm dần I_1 từ I_M xuống 0, vectơ lực F sẽ quay tiếp tục từ α/2 đến α.

Để có thể điều khiển vectơ F theo đúng được các bước, ta phải tính được các giá trị I_1 và I_2 tương ứng với các bước.

Đối với miền thứ nhất: thay $x = I_1 = I_M$ vào (2.7), ta có phương trình bậc hai để tính y:

$$(\cos^2\beta - \cos^2\alpha).y^2 - 2I_M.\cos\alpha.\sin^2\beta.y - I_M^2.\sin^2\beta = 0 \quad (2.8)$$

Bỏ qua các bước biến đổi lượng giác dài dòng, với biểu thức:

$\Delta' = I_M^2.\sin^2\beta.\cos^2\beta.\sin^2\alpha$, ta có nghiệm của phương trình (13) là:

$$y = I_2 = I_M.\frac{\sin\beta}{\sin(\alpha-\beta)} \quad ; I_1 = I_M ; 0 \le \beta \le \alpha/2 \quad (2.9)$$

Đối với miền thứ hai, thay $y = I_2 = I_M$ vào (12) ta có phương trình để tính x:

$$\sin^2\beta.x^2 + 2I_M\cos\alpha.\sin^2\beta.x - (\cos^2\beta - \cos^2\alpha).I_M^2 = 0$$

Giải phương trình ta được:

$$X = I_1 = I_M.\frac{\sin(\alpha-\beta)}{\sin\beta} \quad ; I_2 = I_M ; \alpha/2 \le \beta \le \alpha \quad (2.10)$$

Thay các giá trị β = 0, θ, 2θ, 3θ, ..., α vào (14) và (15) ta nhận được các giá trị dòng điện cần điều khiển để động cơ bước làm việc ở chế độ vi bước.

c)<u>Phương pháp 2</u>:

Để điều khiển được rôto với góc β từ 0 đến α theo vi bước θ, ta đồng thời giảm I_1 và tăng I_2 tương ứng với các bước cần điều khiển.

Đặt y/x = z, từ (12) ta có phương trình:

$$(\cos^2\beta - \cos^2\alpha).z^2 - 2.\cos\alpha.\sin^2\beta.z - \sin^2\beta = 0 \quad (2.11)$$

Giải phương trình này ta được:

$$z_1 = \frac{F_2}{F_1} = \frac{I_2}{I_1} = \frac{\sin\beta}{\sin(\alpha-\beta)} \quad (2.12)$$

$$z_2 = \frac{F_2}{F_1} = \frac{I_2}{I_1} = \frac{\sin\beta}{\sin(\alpha+\beta)} \quad (2.13)$$

z_2 là nghiệm ngoại lai ứng với trường hợp góc β ở trong miền (α – π, 0) và (α, π- α). Có thể sử dụng biểu thức (2.13) để điều khiển theo phương pháp cấp dòng lưỡng cực (I_1 và I_2 luôn luôn trái dấu).

Từ biểu thức (2.12) ta chọn như sau:

$$I_1 = I.\sin(\alpha-\beta) = (I_M / \sin\alpha).\sin(\alpha-\beta) \quad (2.14)$$

hay $$F_1 = F.\sin(\alpha-\beta) = (F_M / \sin\alpha).\sin(\alpha-\beta) \quad (2.15)$$

$$I_2 = I.\sin(\beta) = (I_M / \sin\alpha).\sin(\beta) \quad (2.16)$$

hay $$F_2 = F.\sin(\beta) = (F_M / \sin\alpha).\sin(\beta) \quad (2.17)$$

Và như vậy luôn đảm bảo $I_1, I_2 \leq I_M$.

Từ biểu thức (2.14) và (2.16) ta tính được dòng điện tổng

$$I_1 + I_2 = I_M . \cos(\alpha/2 - \beta) \approx I_M \qquad (2.18)$$

và lực điện từ tổng:

$$F = \sqrt{F_1^2 + F_2^2 + 2F_1.F_2.\cos\alpha}$$

Sau khi thay F_1 và F_2 từ biểu thức (2.15) và (2.17) ta được:

$$F = F_M \text{ (F_M là lực điện từ max do I_M sinh ra)} \qquad (2.19)$$

Từ biểu thức (2.18) và (2.19) có thể thấy rằng trong quá trình điều khiển vi bước, dòng điện tổng thay đổi rất nhỏ, đồng thời lực điện từ tổng F không đổi. Do đó, điều khiển vi bước theo phương pháp thứ hai này sẽ tạo điều kiện thuận lợi cho người thiết kế bộ nguồn dòng điều khiển động cơ bước, cũng như cho việc tính toán mômen tải tối ưu cho hệ cơ khí sử dụng động cơ bước làm cơ cấu chấp hành.

Các biểu thức toán học ở dạng tường minh (2.9), (2.10), (2.14) và (2.16) cho hai phương pháp điều khiển vi bước đã nêu sẽ là cơ sở để chúng ta thiết lập một hệ điều khiển động cơ bước có độ chính xác cao với sự trợ giúp của máy tính và các công cụ vi xử lý mạnh.

3.2.2.Các đặc trưng của tín hiệu điều khiển động cơ bước:

Đối với động cơ bước, tín hiệu điều khiển là các xung rời rác kế tiếp nhau. Việc điều khiển động cơ bước phụ thuộc vào các tham số sau của xung điều khiển:

- ❖Dòng điện I, kể cả cực tính (liên hệ mật thiết là mức điện áp U).
- ❖Độ rộng xung (liên quan đến khả năng dịch bước).
- ❖Tần số xung (liên quan đến tốc độ quay).
- ❖Cách thức cấp xung, bao gồm thứ tự và số lượng cuộn dây pha được cấp (liên quan đến chiều quay và mômen tải).

Tùy thuộc vào việc cấp xung điện, động cơ bước có 4 trạng thái sau đây:

3.2.2.1.Trạng thái không hoạt động: Khi không có cuộn dây nào được cấp điện:

✓ Đối với động cơ phản kháng: rôto sẽ quay trơn.

✓ Đối với động cơ nam châm vĩnh cửu và động cơ kiểu hỗn hợp: có mômen hãm, rôto có xu hướng dừng ở các vị trí mà đường khép từ thông giữa các cực của rôto và stato là nhỏ nhất.

3.2.2.2.Trạng thái giữ:

Khi một số cuộn dây pha được cấp điện một chiều. Rôto mang tải sẽ được giữ chặt ở vị trí góc bước nhất định do lực điện từ tổng F sinh ra mômen giữ.

3.2.2.3.Trạng thái dịch chuyển bước:

Rôto sẽ dịch chuyển từ vị trí bước đang được giữ sang vị trí bước tiếp theo khi các cuộn dây pha được cấp dòng phù hợp.

3.2.2.4.Trạng thái quay quá giới hạn:

Trong chế độ không tải, nếu xung điều khiển có tần số quá cao, động cơ sẽ quay vượt tốc. Ở trạng thái này động cơ không thể đảo chiều, không thể dừng đúng vị trí, nhưng vẫn có thể tăng và giảm tốc từ từ. Muốn dừng và đảo chiều động cơ phải giảm xuống dưới tốc độ tới hạn để hoạt động trong chế độ bước.

Như vậy, động cơ bước chỉ được coi là làm việc khi ở hai trạng thái

Các nghiên cứu tiếp theo là cho hai trường hợp này.

3.2.3.Điều khiển tốc độ quay của động cơ bước:

Động cơ bước có thể quay với bất kỳ tốc độ nào trong giải từ 0 vòng/phút đến giá trị cực đại cho phép.

Do tính chất đặc biệt, động cơ bước có thể dừng đột ngột ở bất kỳ vị trí nào trong độ phân giải của góc bước khi đang quay với bất kỳ tốc độ nào trong dãy cho phép. Vì vậy, động cơ bước ít khi được dùng cho các thiết bị cần quay với vận tốc đều (trường hợp này ta sử dụng các loại động cơ bước khác đơn giản hơn) mà nó được sử dụng chủ yếu để điều khiển thích nghi, nghĩa là tốc độ quay biến đổi liên tục, thậm chí động cơ phải dừng và đứng yên ở vị trí bám sát.

Với lẽ đó, vận tốc quay của động cơ bước thường luôn được hiểu là vận tốc trung bình.

Giả sử góc bước của động cơ là θ^0 thì để đạt được 1 vòng quay ta phải cho động cơ quay $360^0/\theta^0$ bước quay.

Vận tốc trung bình V của động cơ bước trong thời gian t giây là:

$$V = \frac{n}{t}.\frac{\theta}{360} = f.\frac{\theta}{360} \text{ (vòng/giây)} \qquad (2.20)$$

$$\text{Hay} \quad V = f.\frac{\theta}{60} \text{ (vòng/phút)} \qquad (2.21)$$

Việc điều khiển vận tốc động cơ bước được thực hiện bằng cách thay đổi tần số dịch bước f. Lưu ý rằng tần số dịch bước f trong trường hợp tổng quát không đồng nhất với tần số các xung điều khiển, mà nó là tổ hợp của sự biến đổi các trạng thái của các xung điều khiển đó. Vì vậy việc điều khiển này thường được thực hiện bởi các bộ vi xử lý. Dựa vào đồ thị mômen – vận tốc của động cơ bước có thể thấy rằng với vận tốc dưới 5 vòng/giây (300 vòng/1phút), động cơ còn giữ được mômen cực đại; Trên vận tốc này mômen của động cơ sẽ bị giảm dần theo chiều của vận tốc. Do đó việc lựa chọn tải trọng và vận tốc quay cực đại phải được tính toán trước khi thiết kế hệ truyền động sử dụng động cơ bước .

Một yếu tố rất quan trọng đối với động cơ bước là vận tốc tức thời, vận tốc này phải nhỏ hơn vận tốc cực đại đã được tính toán với một trọng tải cho trước.

Gọi T_{cb} là thời gian giữa hai lần chuyển bước liên tiếp, từ công thức (2.20) ta tính được vận tốc tức thời V_t:

$$V_t = \frac{\theta}{360.T_{max}} \text{ (vòng/giây)} \qquad (2.22)$$

Thời gian T_{cb} không nhất thiết phải cố định nhưng đảm bảo điều kiện:

$$T_{cb} > \frac{\theta}{360.V_{max}}$$

Ví dụ: với $\theta = 1,8^0$, $V_{max} = 15$ vòng/giây (900 vòng/ phút)

thì $T_{cb} > 0,33$ ms, cũng có nghĩa là tần số chuyển bước f < 3kHz.

3.2.4.Điều khiển chiều quay của động cơ bước:

Chiều quay của động cơ một chiều có thể thay đổi bằng cách đảo chiều dòng điện cấp vào.

Đối với động cơ bước, chiều quay nhìn chung không đồng nhất với chiều dòng điện cấp cho các cuộn dây mà nó phụ thuộc vào thứ tự chuyển dịch các bước. Chẳng hạn, rôto đang ở vị trí bước thứ n, nếu ta cấp điện sao cho nó chuyển sang vị trí bước thứ (n + 1) thì động cơ quay phải; Nếu ta cấp điện sao cho rôto chuyển sang vị trí thú (n -1). Bộ tạo xung điều khiển sẽ thực hiện việc này.

Chiều quay của động cơ bước được xác định bằng thứ tự chuyển dịch các trạng thái cấp điện của cuộn dây stato. Đối với động cơ 2 pha, nếu điều khiển cả bước, có 4 trạng thái cấp điện; Nếu điều khiển cả bước sẽ có 8 trạng thái cấp điện.

Đối với động cơ 4 pha nếu cấp xung 1 cực thì cũng có 4 và 8 trạng thái cấp điện vào các cuộn dây cho 2 trường hợp điều khiển cả bước và nửa bước. Bảng 1 nêu các trạng thái cấp điện theo cách đơn giản nhất cho 4 cuộn dây pha.

Trạng thái Cuộn dây	1	2	3	4	5	6	7	8
Cuộn 1	1	1	0	0	0	0	0	1
Cuộn 2	0	1	1	1	0	0	0	0
Cuộn 3	0	0	0	1	1	1	0	0
Cuộn 4	0	0	0	0	0	1	1	1

Bảng 3.1- Trạng thái cấp điện các pha của động cơ 4 pha

Trong bảng: tương ứng với cột các trạng thái, ô nào đánh số 1 là cuộn dây được cấp điện 1 cực, ô nào đánh số 0 là cuộn dây không được cấp điện.

Nếu điều khiển cả bước thì chỉ có 4 trạng thái 1, 3, 5 và 7 hoặc 2, 4, 6 và 3.

Nếu điều khiển nửa bước có cả 8 trạng thái.

Khi đã xác định cách cấp điện như trên, trong lúc hoạt động, động cơ bước chỉ có thể ở 8 trạng thái ổn định đó, ngoài ra không còn trạng thái ổn định nào khác. Mỗi lần dịch chuyển trạng thái cấp điện sang trạng thái liền kề thì động cơ dịch chuyển 1 bước (bước đủ hay bước nửa).

Nếu chiều dịch chuyển từ trái qua phải thì động cơ quay phải, ngược lại nếu chiều dịch chuyển từ phải qua trái thì động cơ quay trái. Ví dụ đang ở trạng thái 8 (cuộn 1 và cuộn 4 cấp điện), nếu dịch trái sang trạng thái 7 (cuộn 1 cắt điện và cuôn 4 giữ nguyên) thì động cơ quay trái; Nếu dịch phải sang trạng thái 1 (cuộn 1 để nguyên, cắt điện cuộn 4) thì động cơ quay phải.

Từ bảng 3.1 ta có thể đưa ra một chú ý hết sức quan trọng: Trong quá trình hoạt động (quay hay giữ) thì ít nhất một cuộn dây pha phải được cấp điện (trạng thái Turn - Off) thì rôto sẽ quay trơn, có nghĩa là nếu tải gây ra mômen quay thì rôto động cơ sẽ bị quay bởi lực bên ngoài. Ngược lại muốn dùng lực ngoài để thay đổi vị trí tải thì phải đưa động cơ về trạng thái Turn hoặc Off. Tầm quan trọng của chú ý này còn nằm ở chỗ: hệ truyền động động cơ bước sẽ không hoạt động đúng được nếu ta điều khiển nó ở hai trạng thái Turn – Off và dịch bước, mà phải điều khiển ở hai chế độ giữ và dịch bước, có nghĩa là bắt buộc phải cấp điện cho cuộn dây pha kể cả khi hệ dừng và lúc hệ chuyển động. Vấn đề cốt lõi của việc điều khiển của động cơ bước là cấp điện lúc động cơ dừng – giữ. Do đó sẽ là sai lầm lớn nếu ta chỉ cấp xung điều khiển lúc động cơ quay còn lúc dừng thì không cấp xung điều khiển.

III.ĐIỀU KHIỂN ĐỘNG CƠ BƯỚC 4 PHA :

3.3.1.Cấu tạo:

Động cơ bước có rất nhiều loại nhưng thông dụng hiện nay là loại stepmotor (động cơ bước) với 4 cuộn dây điều khiển hoạt động. Cụ thể loại này có 5 hoặc 6 dây đầu ra, trong đó 4 dây với 4 màu khác nhau là dây điều khiển các cuộn dây trong động cơ, dây (các dây) còn lại là nối đất (Ground).

Loại động cơ bước khảo sát ở đây là loại dùng nam châm vĩnh cửu (Permanent Magnet). Cấu tạo gồm stato là 4 cuộn dây được đặt cách nhau 90^0 còn rôto gắn trên trục sẽ chứa nam châm vĩnh cửu, nên việc kích hoạt các cuộn dây sẽ làm cho động cơ quay theo 1 góc xác định.

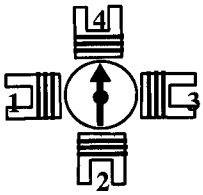
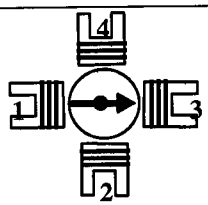
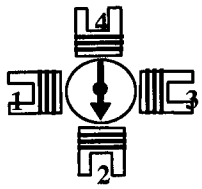
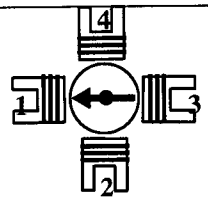
3.3.2.Phương thức hoạt động:

Motor loại này là động cơ bước vì ta có thể điều khiển nó quay theo từng bước xác định nhờ đó có thể định được tốc độ quay và xác định góc quay.

Ở loại motor này để nó hoạt động phải kích hoạt (cho điện thế) đầu vào tuần tự theo chiều quay mà ta muốn. Cụ thể ta có 2 cách cơ bản điều khiển động cơ bước như sau:

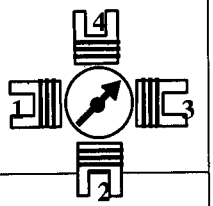
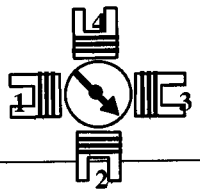
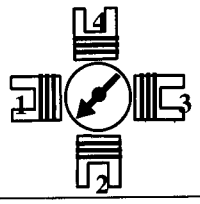
3.3.2.1.Full step (điều khiển toàn bước):

Ta sẽ kích hoạt lần lược các cuộn dây theo chiều quay mong muốn:

Step	Coil4	Coil3	Coil2	Coil1	Minh họa
a.1	On	Off	Off	Off	
a.2	Off	On	Off	Off	
a.3	Off	Off	On	Off	
a.4	Off	Off	Off	On	

Bảng 3.2- Hoạt động của động cơ bước khi điều khiển toàn bước

Cách này có nghĩa là với mỗi lần kích điện thì motor sẽ quay một góc bằng với 1 bước của động cơ, thông thường là $1,8^0$. Thật ra thì vẫn còn cách khác để điều khiển động cơ toàn bước, mà cụ thể là ta sẽ khảo sát sau đây, sẽ làm cho mômen của động cơ tăng lên gần gấp rưỡi. Với cách này ta phải kích điện mỗi lần 2 cuộn dây.

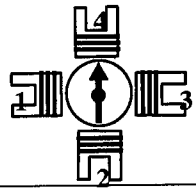
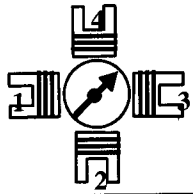
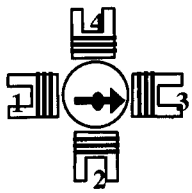
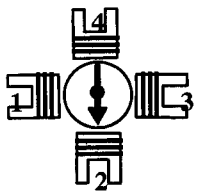
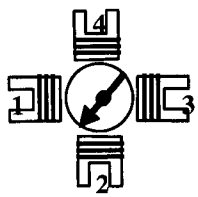
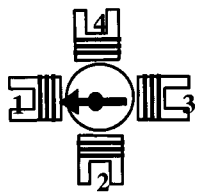
Step	Coil4	Coil3	Coil2	Coil1	Minh họa
b.1	On	On	Off	Off	
b.2	Off	On	On	Off	
b.3	Off	Off	On	On	
b.4	On	Off	Off	On	

Bảng 3.3- Hoạt động của động cơ bước khi kích 2 cuộn dây cùng lúc.

Ta thấy rằng trong trường hợp này motor vẫn quay mỗi lần 1 bước nhưng rõ ràng là mômen xoắn tăng do mỗi bước rôto được kéo quay bởi đồng thời 2 cuộn dây.

3.3.3.2.<u>Half step(điều khiển nửa bước)</u> :

Trong phương pháp điều khiển này thì ở mỗi lần kích hoạt, động cơ sẽ quay 1 góc bằng ½ góc chuẩn (góc quay của fullstep), thường là $0,9^0$. Như vậy, motor sẽ quay "min" hơn so với cách điều khiển fullstep như trên, kết quả là việc xác định vị trí sẽ tốt hơn. Sau đây là 1 chu trình điều khiển stepmotor bằng phương pháp điều khiển nửa bước:

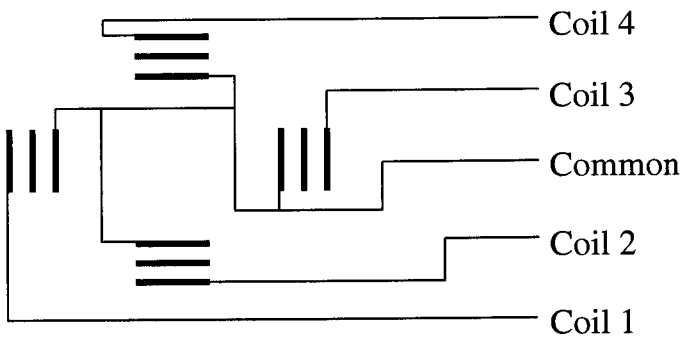
Step	Coil4	Coil3	Coil2	Coil1	Minh họa
a.1	On	Off	Off	OFF	
b.1	On	On	Off	Off	
a.2	Off	On	Off	Off	
b.2	Off	On	On	Off	
a.3	Off	Off	On	Off	
b.3	Off	Off	On	On	
a.4	Off	Off	Off	On	

b.4	On	Off	Offz	On	

Bảng 3.4- Hoạt động của động cơ bước khi điều khiển nửa bước.

3.3.3. Xác định các ngõ vào:

Giờ đây các bạn có thể viết chương trình điều khiển động cơ bước trên bằng máy tính, cổng song song chẳng hạn. Nhưng điều quan trọng là khi lắp ráp mạch thì ta phải xác định các ngõ vào cụ thể của motor để biết đầu là cuộn 1, 2, 3, 4 mà cho tín hiệu điều khiển tương ứng vào. Dưới đây là sơ đồ đơn giản của motor, nhờ vào cách đo điện trở của từng cặp dây, ta sẽ xác định được các dây cụ thể:



Hình 8.1- Sơ đồ đơn giản của động cơ bước.

Phương pháp cụ thể như sau: (Dùng VOM để đo điện trở)

a)Bước 1: Xác định dây GND:

Đối với các motor có 6 dây thì thông thường 2 dây cùng màu sẽ là dây Ground. Phương pháp xác định tổng quát là đo điện trở của từng cặp dây, nếu cặp dây nào có điện trở nhỏ hơn (bằng phân nửa) thì 1 trong 2 dây là dây GND. Ta chỉ cần đo thử mỗi dây với các dây còn lại thì sẽ xác định được dây Ground vì điện trở của nó với dây khác luôn nhỏ hơn.

b)Bước 2: Xác định đầu dây cuộn 1:

Vì motor có 4 cuộn dây đặt cách nhau 90^0 nên vai trò của nó là như nhau, nói cách khác, ta chỉ cần xác định vị trí tương quan giữa chúng để có thể điều khiển chiều motor quay. Vậy nên ta giả sử một dây bất kỳ là dây của cuộn số 1.

c)Bước 3: Xác định đầu dây các cuộn còn lại:

Dùng điện áp hoạt động của động cơ để kích vào một đầu dây, các trường hợp sau sẽ được liệt kê để được xác định các đầu dây còn lại (trong trường hợp này thì dây cuộn 1 luôn được cấp điện).

❖ Động cơ đứng yên: Đây là đầu cuộn dây đối diện (cuộn 3).

- ❖ Động cơ quay sang trái: Đầu dây cuộn 2.
- ❖ Động cơ quay sang phải: Đầu dây cuộn 4.

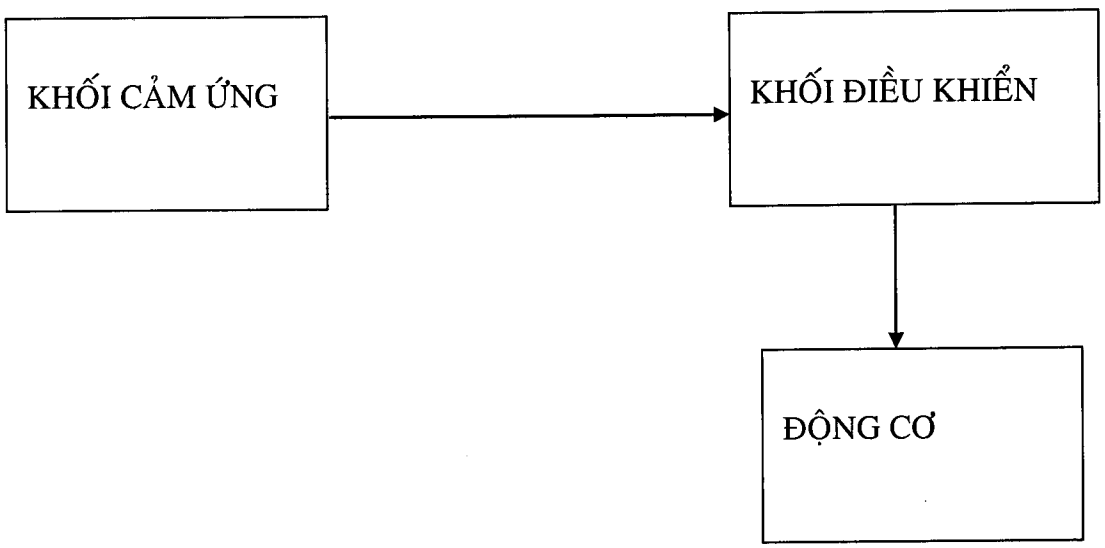
3.3.4. Ứng dụng của động cơ bước:

Ngày nay, mà cụ thể là trong các ngành tự động hóa rất cần điều khiển chính xác cũng như điều khiển tự động qua máy tính nên động cơ bước ngày càng được nghiên cứu và dùng rộng rãi. Động cơ bước dùng trong các máy cần xác định chính độ chính xác như điều chỉnh bước tiến, quay của hệ thống tay quay. Ngòai ra còn được dùng trong máy cần có sự điều khiển tốc độ theo từng thời điểm.

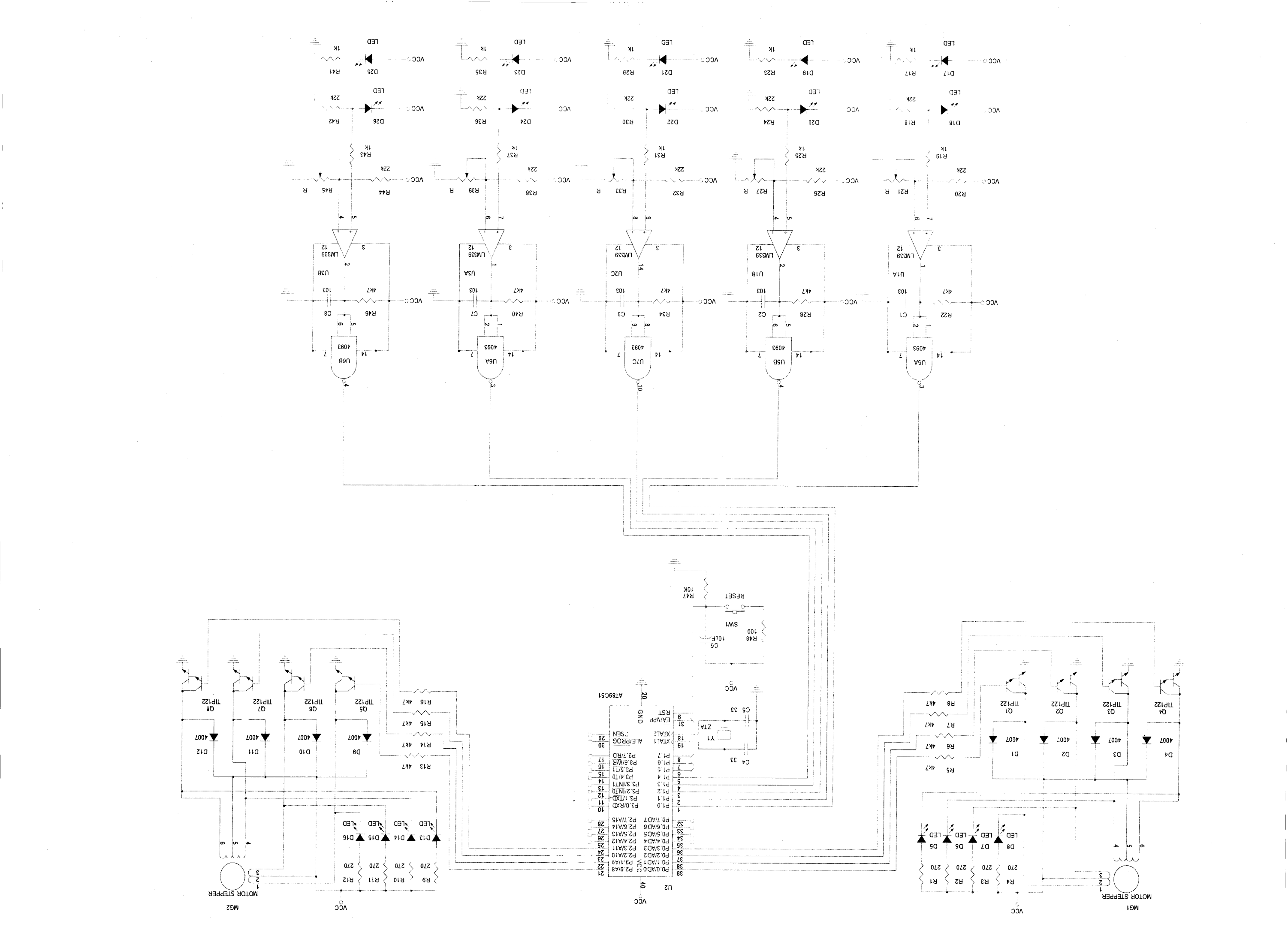
CHƯƠNG 4: *THIẾT KẾ KỸ THUẬT*

I. MÔ HÌNH THIẾT KẾ:

4.1.1. Sơ đồ khối:



4.1.2. Sơ đồ nguyên lý:


AT89C51
U2
RESET
SW1
R47 10K
R48 100
C6 10UF
C5 33
C4 33
X1 ZTA
MG1
MG2
MOTOR STEPPER
LM339
4093
4007
TIP122
LED
VCC

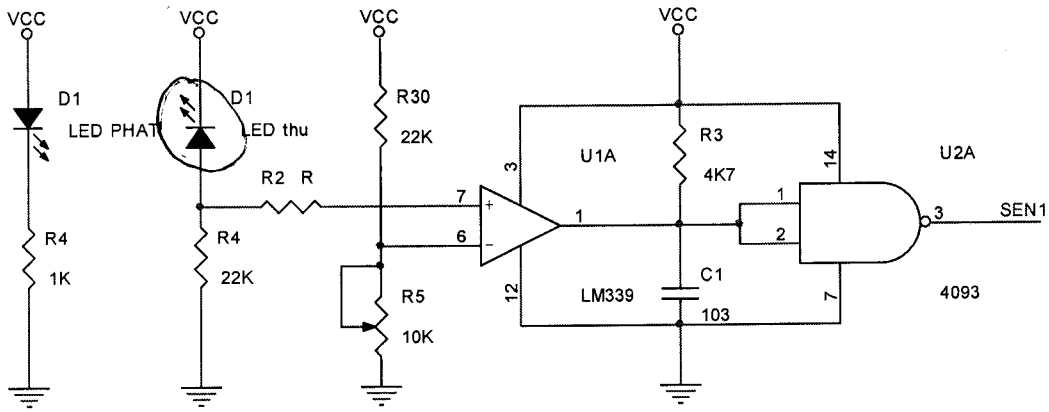
II.PHÂN TÍCH MẠCH ĐIỆN CHI TIẾT CÁC KHỐI:

4.2.1. Thiết kế cụ thể dò đường:

- Nguyên lý hoạt động:

- Tín hiệu đọc về từ led phát sẽ được đưa vào mạch cảm biến để xử lý.Với cách sắp xếp led như phần giới thiệu linh kiện thì khi robot chạy đúng đường thì quang trở thu không bắt được tín hiệu,còn nếu lệch đường thì quang trở sẽ thu được tín hiệu và đưa về mạch cảm biến để xử lý.

- Tính toán:



-Nhiệm vụ:led phát phát liên tục,quang trở thu ánh sáng từ led phát.Khi quang trở thu được ánh sáng thì ngõ ra SEN1 là mức 0 tín hiệu này được đưa về vi điều khiển để kiểm tra và xử lý.

-Tính toán:

$$V7=\frac{R4}{R4+R_{QT}}.V_{CC}$$

$$V6=\frac{R5}{R5+R30}.V_{CC}$$

Khi quang trở không thu ánh sáng thì R_{QT} lớn vì vậy áp vào chân 7 của LM339 sẽ thấp so với áp ở chân 6 của LM339.Nên ngõ ra 1 sẽ là mức thấp được đưa vào cổng đảo của 4093 sẽ cho ra mức cao.

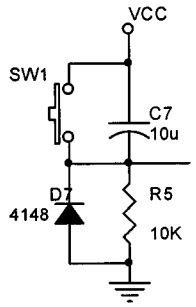
V7<V6

Ngược lại thì có ánh sáng thì điện trở quang trở giảm nên:V7>V6

R5:Là một biến trở tinh chỉnh mục đích để điều chỉnh áp tại chân 6 sao cho khi robot chệch hướng thì quang trở bắt ánh sáng mà ngõ ra 1 của LM339 đáng lẽ phải ở mức cao 5v nhưng lại không đúng.Chọn R5=10k,R4=R30=22k

-4093 là cổng NAND schmitt triggers khi ngõ vào thay đổi một lượng nhỏ là ngõ ra sẽ thay đổi tức thì.

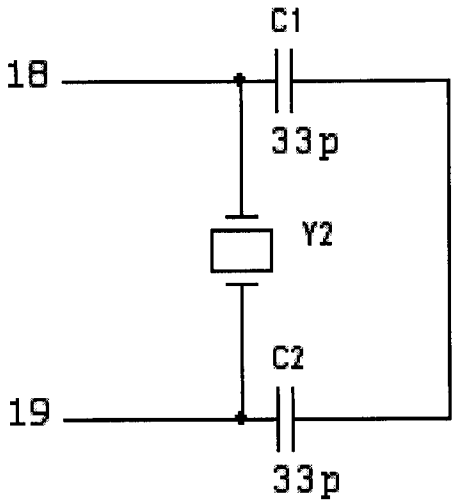
4.2.2 Nguyên lý mạch Reset 89C51:



Hình 4.1- Mạch Reset 89C51.

- Để mạch Reset được thì chân Reset phải ở mức cao ít nhất là 2 chu kì máy.
- Khi vừa mới cấp nguồn, dựa vào tính chất giữa hai bản cực của tụ điện (trước khi tụ điện bắt đầu nạp) thì điện áp giữa hai bản cực là bằng V_{CC} nên chân Reset của 89C51 ở mức cao và mạch được Reset. Khi tụ nạp ($\tau = R.C$) tụ nạp qua diode và qua tụ khi tụ đầy thì điện áp trên tụ là $V_C = V_{CC}$ và lúc này điện áp tại chân Reset sẽ là $V_{CC} - V_C = 0$. Lúc này không còn Reset cho 89C51 nữa.
- Khi nút SW được nhấn thì điện áp tại chân Reset sẽ là V_{CC}, mạch được reset và đồng thời tụ bắt đầu xả và xả rất chậm (do $\tau = R_5.C$) R_5 lúc này là 10K. Do xả rất chậm, phím nhấn được nhấn và thả ra rất nhanh nên điện áp trên tụ giảm nhưng chưa hết (tùy thuộc vào lúc nhấn và giữ). Và khi thả SW ra thì tụ bắt nạp lại qua diode rất nhanh giống ban đầu $V_C = V_{CC}$. Và điện áp tại chân Reset = $V_{CC} - V_C = 0$

4.2.3 Mạch tạo dao động

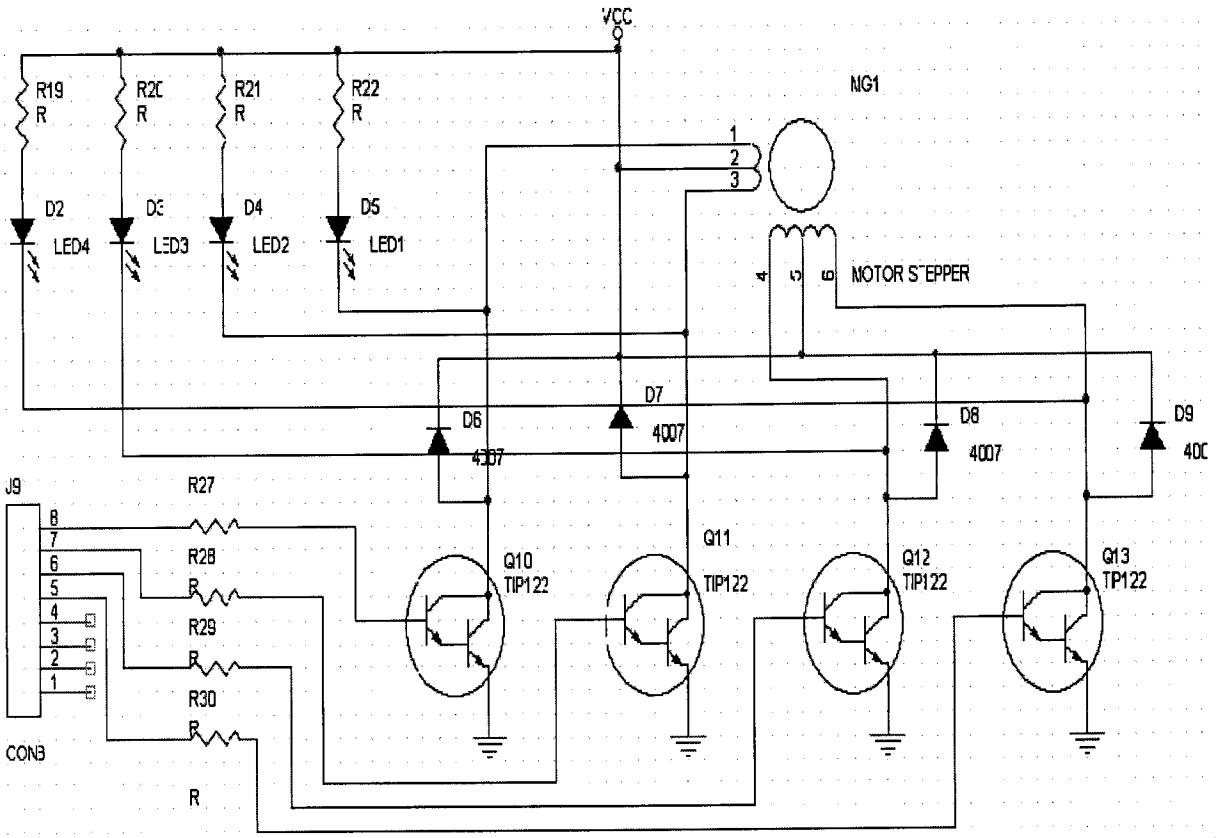


Hình 4.2- Mạch tạo dao động thạch anh.

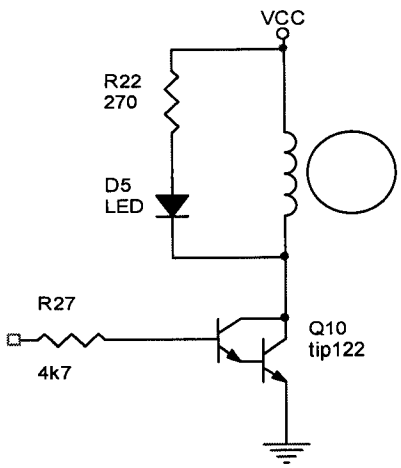
Mạch tạo dao động cho 89C51 có tụ không phân cực C1 và C2 có giá trị là 33p dùng để lọc nhiễu ở tần số cao

Mạch này không dùng chế độ truyền thông nối tiếp nên ta dùng thạch anh 12MHz (chế độ real time)

4.2.4 Mạch điều khiển động cơ:



Hình 4.3- Sơ đồ nguyên lý mạch điều khiển động cơ bước.



Hình 4.4- Mạch điều khiển một cuộn dây của động cơ bước.

Để TST Q10 dẫn thì phải cấp điện áp ở chân B (do 89C51 cấp) . Ta mong muốn TST dẫn bảo hòa do đó:

$$Ic = \frac{Vcc - Vcesat}{Rdc}$$

Ta đo được Rdc của cuộn dây ở trạng thái tỉnh là 5 Ω

$$Ic = \frac{5 - 0.2}{5} = 0.96(A)$$

$$Ib = k\frac{Ic}{hfe}$$ ta chọn k=1 và hfe=1000

$$Ib = \frac{0.96}{1000} = 0.96(mA)$$

Ta chọn TST Tip122 là thỏa

Tip122 có các thông số: Vce=100V
Ic= 5A
Hfe=1000

Lúc này ta tính

$$R27 = \frac{5 - 1.4}{0.96(mA)} = 4.06k$$

Chọn R27 = 4K7

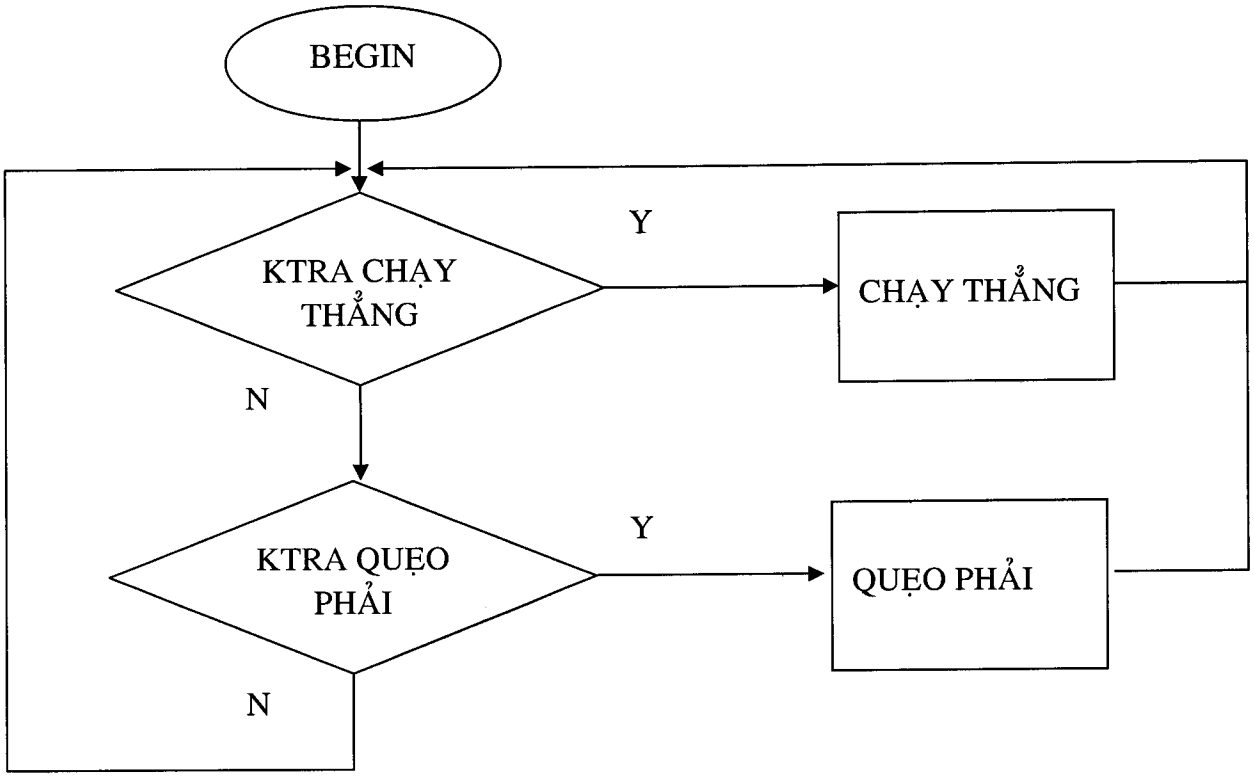
Để LED sáng ta chọn dòng qua LED 10 mA

$$R22 = \frac{5 - 2 - 0.2}{10mA} = 280\ \Omega$$

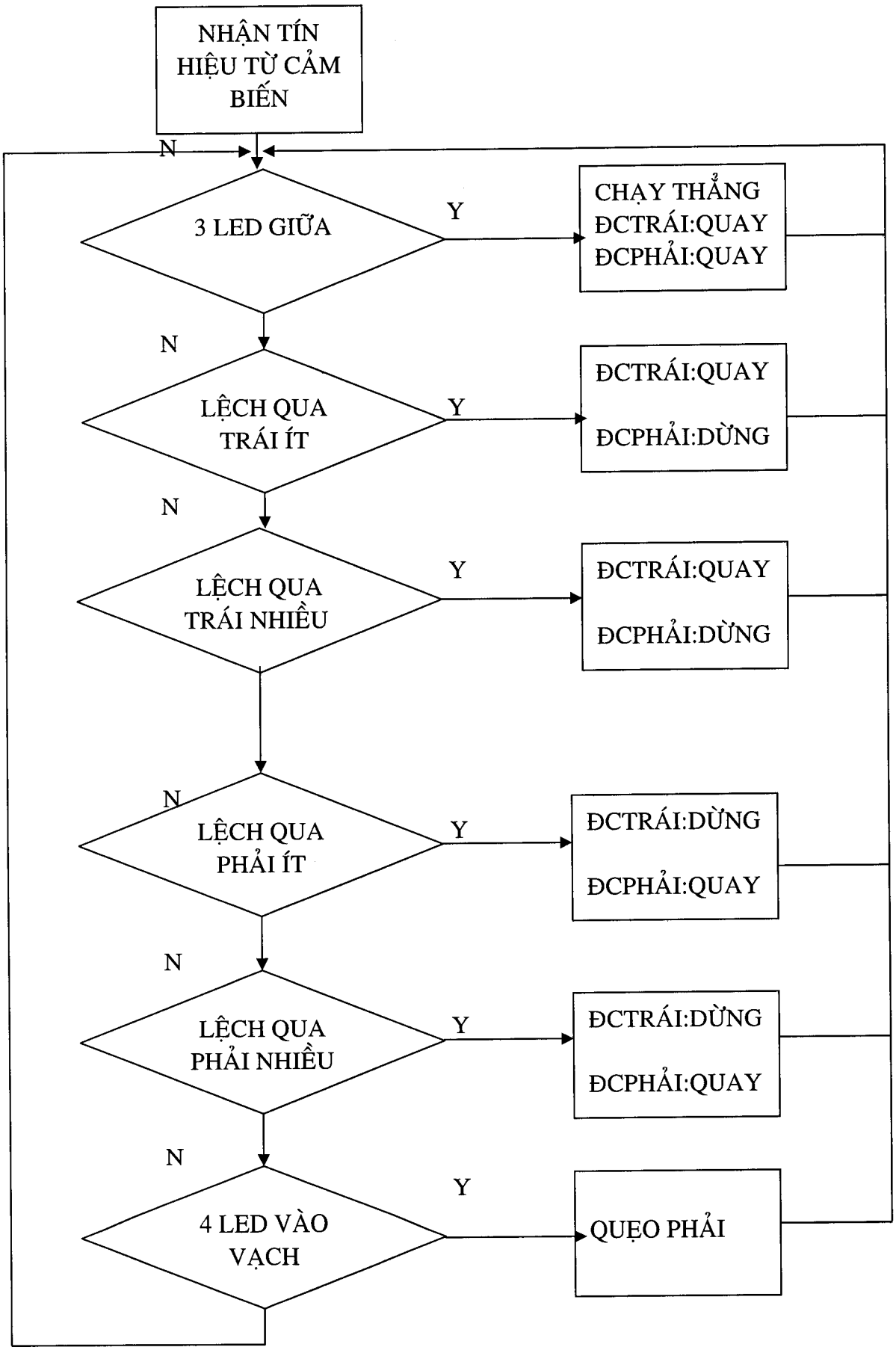
Ta chọn R22 = 270 Ω

III. LƯU ĐỒ VÀ CHƯƠNG TRÌNH:

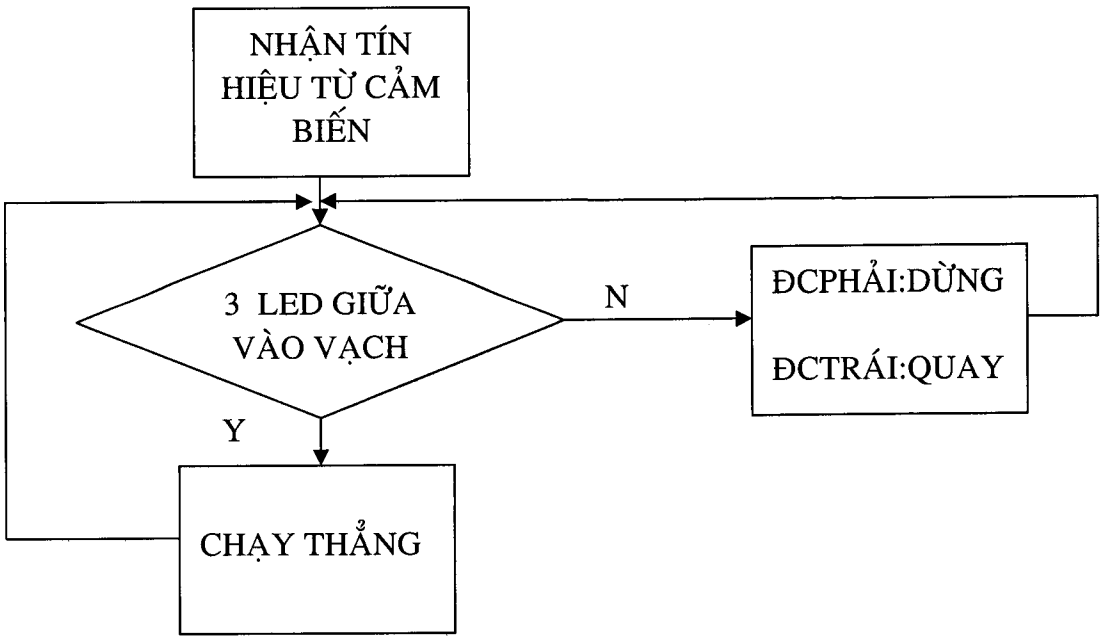
4.3.1 Lưu đồ:



CHẠY THẲNG:


NHẬN TÍN HIỆU TỪ CẢM BIẾN
N
3 LED GIỮA
Y
CHẠY THẲNG
ĐCTRÁI:QUAY
ĐCPHẢI:QUAY
N
LỆCH QUA TRÁI ÍT
Y
ĐCTRÁI:QUAY
ĐCPHẢI:DỪNG
N
LỆCH QUA TRÁI NHIỀU
Y
ĐCTRÁI:QUAY
ĐCPHẢI:DỪNG
N
LỆCH QUA PHẢI ÍT
Y
ĐCTRÁI:DỪNG
ĐCPHẢI:QUAY
N
LỆCH QUA PHẢI NHIỀU
Y
ĐCTRÁI:DỪNG
ĐCPHẢI:QUAY
N
4 LED VÀO VẠCH
Y
QUẸO PHẢI

QUẸO PHẢI:



Động cơ trái: P0 Động cơ phải: P1

4.3.2 <u>Chương trình</u> :

```
TH1    EQU    127
TH2    EQU    126
TH3    EQU    125
TH4    EQU    124
TH5    EQU    123
TH6    EQU    122
TH7    EQU    121
TOCDO  EQU    120
KTCHAYTHANGXONG  EQU    119
KTQUEOPHAIXONG   EQU    118
BIENDICHPHAI   EQU    117
BIENDICHTRAI   EQU    116
MOV    TH1,#14       ;----O GIUA
MOV    TH2,#6        ;----LECH QUA TRAI IT
MOV    TH3,#7        ;----LECH QUA TRAI HOI NHIEU
```

```
MOV    TH4,#3        ;----LECH QUA TRAI NHIEU
MOV    TH5,#12        ;----LECH QUA PHAI IT
MOV    TH6,#8        ;----LECH QUA PHAI HOI NHIEU
MOV    TH7,#24        ;----LECH QUA PHAI NHIEU
MOV    P1,#0FFH
;---------------------------------------
MAIN:
     MOV    TMOD,#11H
     MOV    TOCDO,#8
     MOV    KTCHAYTHANGXONG,#0
     MOV    KTQUEOPHAIXONG,#0
     MOV    BIENDICHPHAI,#08H
     MOV    BIENDICHTRAI,#01H
     MOV    P2,#0
     MOV    P0,#0
BEGIN:
     MOV    A,KTCHAYTHANGXONG
     CJNE   A,#0,BATDAUQUEO
     CALL   QUAY_THUAN
     MOV    KTQUEOPHAIXONG,#0
     SJMP   BEGIN

BATDAUQUEO:
     CALL   QUAYTOILUIPHAI
     MOV    KTCHAYTHANGXONG,#0
     SJMP   BEGIN
;---------------------------------------
QUEOPHAI:
     MOV    TOCDO,#8
     CALL   DELAY
     CALL   QUAYTOILUIPHAI
     CALL   XOA
     MOV    TOCDO,#8
     CALL   DELAY
RET
;---------------------------------------
KTQUEOPHAI:
     MOV    A,P1
     ANL    A,#1FH
```

```
    CJNE   A,#14,QUEOPHAITIEP
    MOV    KTQUEOPHAIXONG,#1
    LJMP   THOATKTQUEOP
QUEOPHAITIEP:
   ; CJNE   A,#15,KTP1
    MOV    A,BIENDICHPHAI
    ANL    A,#0FH
    CJNE   A,#0,GOQP1
    MOV    A,#10H
GOQP1:
    RR     A
    MOV    BIENDICHPHAI,A
    LJMP   THOATKTQUEOP
KTP1:
    ;CJNE   A,#6,KTP2
    MOV    A,BIENDICHPHAI
    ANL    A,#0FH
    CJNE   A,#0,GOQP2
    MOV    A,#10H
GOQP2:
    RR     A
    MOV    BIENDICHPHAI,A
    LJMP   THOATKTQUEOP
KTP2:

THOATKTQUEOP:
RET
;---------------------------------------
QUAYTOILUIPHAI:
QUAYTOILUIPHAISTART:
    MOV    P0,BIENDICHPHAI
    MOV    P2,BIENDICHTRAI
    CALL   DELAY
    CALL   KTQUEOPHAI
    MOV    A,KTQUEOPHAIXONG
    CJNE   A,#0,THOATQUAYTOILUI
    LJMP   QUAYTOILUIPHAISTART
THOATQUAYTOILUI:
RET
```

```
;--------------------------------------
KTCHAYTHANG:
     MOV    A,P1
     ANL    A,#1FH
     ;--QUEOPHAI
     CJNE   A,#15,TTKTTHANG  ;TIEP TUC KIEM TRA THANG
     MOV    KTCHAYTHANGXONG,#1
     LJMP   KT7
     ;--QUEOTRAI
     ;--O GIUA
TTKTTHANG:
    ; MOV    TOCDO,#20
     CJNE   A,TH1,KT1
     MOV    TOCDO,#8
     MOV    A,BIENDICHPHAI
     ANL    A,#0FH
     CJNE   A,#0,GO1
     MOV    A,#10H
GO1:
     RR     A
     MOV    BIENDICHPHAI,A
     MOV    A,BIENDICHTRAI
     ANL    A,#0FH
     CJNE   A,#0,GO2
     MOV    A,#01H
     SJMP   KOQUAYTRAI1
GO2:
     RL     A
KOQUAYTRAI1:
     MOV    BIENDICHTRAI,A
     LJMP   KT7
KT1:
     CJNE   A,TH2,KT2      ;P0 QUAY - P2 DUNG
     MOV    A,BIENDICHPHAI
     ANL    A,#0FH
     CJNE   A,#0,GO3
     MOV    A,#10H
GO3:
     RR     A
```

```
    MOV   BIENDICHPHAI,A
    LJMP  KT7
KT2:
    CJNE  A,TH3,KT3     ;P0 QUAY - P2 DUNG
    MOV   A,BIENDICHPHAI
    ANL   A,#0FH
    CJNE  A,#0,GO4
    MOV   A,#10H
GO4:
    RR    A
    MOV   BIENDICHPHAI,A
    LJMP  KT7
KT3:
    CJNE  A,TH4,KT4     ;P0 QUAY - P2 DUNG
    MOV   A,BIENDICHPHAI
    ANL   A,#0FH
    CJNE  A,#0,GO5
    MOV   A,#10H
GO5:
    RR    A
    MOV   BIENDICHPHAI,A
    LJMP  KT7
KT4:
    CJNE  A,TH5,KT5     ;P2 QUAY - P0 DUNG
    MOV   A,BIENDICHTRAI
    ANL   A,#0FH
    CJNE  A,#0,GO6
    MOV   A,#01H
    SJMP  KOQUAYTRAI2
GO6:
    RL    A
KOQUAYTRAI2:
    MOV   BIENDICHTRAI,A
    LJMP  KT7
KT5:
    CJNE  A,TH6,KT6
    MOV   A,BIENDICHTRAI
    ANL   A,#0FH
    CJNE  A,#0,GO7
```

```
    MOV   A,#01H
    SJMP  KOQUAYTRAI
GO7:
    RL    A
KOQUAYTRAI:
    MOV   BIENDICHTRAI,A
    LJMP  KT7
KT6:
    MOV   A,BIENDICHTRAI
    ANL   A,#0FH
    CJNE  A,#0,GO8
    MOV   A,#01H
    SJMP  KOQUAYTRAI3


GO8:
    RL    A
KOQUAYTRAI3:
    MOV   BIENDICHTRAI,A
KT7:
    RET
;---------------------------------------
QUAY_THUAN:
QUAYTHUANSTART:
    MOV   P0,BIENDICHPHAI
    MOV   P2,BIENDICHTRAI
    CALL  DELAY
    CALL  KTCHAYTHANG
    MOV   A,KTCHAYTHANGXONG
    CJNE  A,#0,THOATQUAYTHUAN
    LJMP  QUAYTHUANSTART
THOATQUAYTHUAN:
RET
;---------------------------------------
XOA:
    CLR   P0.0
    CLR   P2.0
    CLR   P0.1
    CLR   P2.1
```

```
    CLR   P0.2
    CLR   P2.2
    CLR   P0.3
    CLR   P2.3
RET
;-------------------------------------
DELAY:
    PUSH  00
    MOV   R0,TOCDO
AGAIN:
    MOV   TH0,#HIGH(-5000)
    MOV   TL0,#LOW(-5000)
    CLR   TF0
    SETB  TR0
    JNB   TF0,$
    DJNZ  R0,AGAIN
    POP   00
RET
```

KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN.

1.Kết luận:

Với đề tài: ROBOT TỰ HÀNH tôi thấy với nhiệm vụ nhận được tôi đã hoàn thành khá tốt cụ thể như sau:

- Tìm hiểu về cơ khí.
- Tìm hiểu vi điều khiển 89C51.
- Tìm hiểu động cơ bước,tốc độ động cơ...
- Thiết kế mạch điều khiển động cơ theo đường đi dùng cảm biến quang
- Thi công mạch.

Trong quá trình làm mạch tôi đã gặp trở ngại khi quyết định sử dụng 3 cảm biến quang vì khi sử dụng 3 cảm biến quang thì robot dễ bị mất lái,và tôi đã khắc phục bằng cách dùng 5 cảm biến quang để đảm bảo robot bám đường tốt hơn.

- ***Ưu điểm:***
  - Robot chay ổn định.
  - Tự bản thân robot xử lý khi gặp góc quẹo khá tốt.
- ***Khuyết điểm:***
  - Robot chỉ di chuyển tốt trên mặt phẳng đều.
  - Mạch cảm biến còn chưa gọn.

Quá trình làm luận văn tốt nghiệp đã giúp tôi củng cố lại kiến thức của mình và tiếp thu thêm được nhiều kiến thức bổ ích như cách thu phát tín hiệu bằng cảm biến quang, cách sử dụng các công cụ cơ khí,...

Tôi thấy đề tài luận văn tốt nghiệp tôi đã thành công nhưng vẫn còn hạn thiếu sót như mạch in chưa được đẹp, cuốn báo cáo chưa phong phú,mô hình còn đơn giản do thời gian có hạn.

2.Hướng phát triển:

Vì thời gian có hạn nên tôi chưa có điều kiện để phát huy Robot thêm ở phần cơ khí để Robot bám đường tốt hơn. Đồng thời tôi cũng muốn điều khiển Robot bằng tay,hoặc điều khiển từ xa có thêm phần giao tiếp máy tính...

Tài liệu tham khảo

1-Cấu trúc và lập trình họ vi điều khiển 8051-Nguyễn Tăng Cường.
2-Linh kiện bán dẫn- Phạm Hồng Quang.
3-Tài liệu từ đồ án môn học những năm trước.
4-Các trang web tham khảo:
www.alldatasheet.com
www.stepmotor.com
www.dientuvietnam.net

ĐẠI HỌC KỸ THUẬT CÔNG NGHỆ TPHCM
SV NGUYỄN CHÍ THIỆN

Đồ án tốt nghiệp

Sử dụng PLC điều khiển hệ thông truyền động trong robot công nghiệp

CHƯƠNG 1

TỔNG QUAN VỀ ROBOT CÔNG NGHIỆP

1.1 Lịch sử phát triển:

Thuật ngữ Robot xuất hiện vào năm 1920 trong một tác phẩm văn học của nhà văn tiệp khắc có tên là Karel Capek.

Thuật ngữ Inducstrial Robot (IR) xuất hiện đầu tiên ở mỹ do công ty AMF (Americal Machine and Foundry Company) quảng cáo mô phỏng một thiết bị mang dáng dấp và có một số chức năng như tay người được điều khiển tự động thực hiện một số thao tác để sản xuất thiết bị có tên gọi Versatran.

Quá trình phát triển của IR được tóm tắt như sau:

- Từ những năm 1950 ở mỹ xuất hiện viện nghiên cứu đầu tiên.

- Vào đầu những năm 1960 xuất hiện sản phẩm đầu tiên có tên gọi là Versatran của công ty AMF.

- Ở Anh người ta bắt đầu nghiên cứu và chế tạo IR theo bản quyền của mỹ từ những năm 1967.

- Ở các nước tây âu khác như: Đức, ý, Pháp, Thuỵ Điển thì bắt đầu chế tạo IR từ những năm 1970.

- Châu Á có Nhật Bản bắt đầu nghiên cứu ứng dụng IR từ năm 1968.

Đến nay, trên thế giới có khoảng trên 200 công ty sản xuất IR trong số đó có 80 công ty của Nhật, 90 công ty của Tây Âu, 30 công ty của Mỹ và một số công ty của Nga, Tiệp…

1.2 Phân loại rôbốt công nghiệp

1.2.1 Theo chủng loại, mức độ điều khiển, và nhận biết thông tin của tay máy-người máy đã được sản xuất trên thế giới có thể phân loại các IR thành các thế hệ sau:

Thế hệ 1: Thế hệ có kiểu điều khiển theo chu kỳ dạng chương trình cứng không có khả năng nhận biết thông tin.

Thế hệ 2: Thế hệ có điều khiển theo chu kỳ dạng chương trình mềm bước đầu đã có khả năng nhận biết thông tin.

Thế hệ 3: Thế hệ có kiểu điều khiển dạng tinh khôn, có khả năng nhận biết thông tin và bước đầu đã có một số chức năng lý trí của con người.

1.2.2 Phân loại tay máy theo cấu trúc sơ đồ động:

Thông thường cấu trúc chấp hành của tay máy công nghiệp được mô hình hoá trong dạng chuỗi động với các khâu và các khớp như trong nguyên lý máy với các giả thuyết cơ bản sau:

- Chỉ sử dụng các khớp động loại 5 (khớp quay, khớp tịnh tiến, khớp vít).

- Trục quay hướng tịnh tiến của các khớp thì song song hay vuông góc với nhau.

- Chuỗi động chỉ là chuỗi động hở đơn giản:

Ta ví dụ một chuỗi động của một tay máy công nghiệp có 6 bậc tự do, các khớp A, B, F là các khớp tổng quát, có nghĩa là chúng có thể là khớp quay, cũng có thể là khớp tịnh tiến, các khớp D, E, K chỉ là những khớp quay. Các khâu được đánh số bắt đầu từ 0-giá cố định, tiếp đến là các khâu 1, 2, ...n-các khâu động, khâu tổng quát ký hiệu là khâu i, (i= 1, 2, 3, ...n), khâu n cuối cùng mang bàn kẹp của tay máy. Tương tự như bàn tay người để bàn kẹp gồm có 3 loại chuyển động, tương ứng với các chuyển động này là 3 dạng của cấu trục máy như sau:

- Cấu trúc chuyển động toàn bộ (chân người) cấu trúc này thực hiện chuyển động đem toàn bộ tay máy (tay người) đến vị trí làm việc. Cấu trúc này hết sức đa dạng và thông thường nếu không phải là tay máy hoạt động

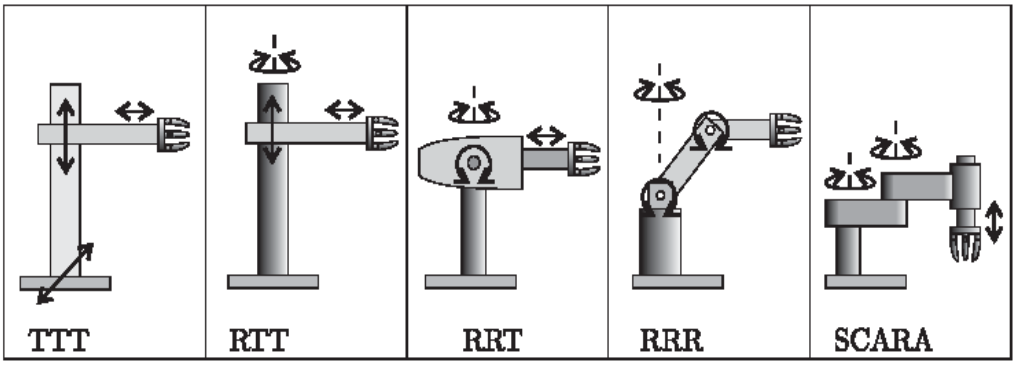
trong hệ thống mà chuyển động này cần có sự kiểm soát. Người ta thường coi tay máy là đứng yên, khâu 0 gọi là giá cố định của tay máy.

- Cấu trúc xác định bàn kẹp bao gồm các khớp A,B và F các khâu 1, 2 và 3, chuyển động của cấu trúc này đem theo bàn kẹp với vị trí làm việc. Do giả thiết về loại khớp động dùng trong chế tạo máy thông thường ta có những phối hợp sau đây của các khớp và từ đó tạo nên những cấu trúc xác định vị trí của bàn kẹp trong các không gian vị trí khác nhau của bản kẹp. Phối hợp TTT nghĩa là 3 khớp đều là khớp tịnh tiến và một khớp quay. Đây là cấu trúc hoạt động trong hệ toạ độ Đề Các so với các toạ độ So vì 3 điểm M nằm trên khâu 3 khớp đều là khớp tịnh tiến và một chuyển động quay (tức là hai toạ độ dài).

Phối hợp TRT, RTT, hay TTR nghĩa là một khớp tịnh tiến hai khớp quay(các cấu trúc 2, 3, và 4). Đây là cấu trúc hoạt động trong hệ toạ độ trụ so với điểm M trên khâu 3 được xác định bởi 2 chuyển động tịnh tiến và một chuyển động quay(tức là hai toạ độ dài một toạ độ gốc).

Phối hợp RTR, RRT, TTR nghĩa là hai khớp tịnh tiến và hai khớp quay(các cấu trúc 5, 6, 7, 8, 9 và 10). Đây là cấu trúc hoạt động trong hệ toạ độ cầu so với hệ So, vì điểm M trên khâu 3 được xác định bởi một chuyển động tịnh tiến và hai chuyển động quay(tức là một toạ độ dài hai toạ độ gốc).

Phối hợp RRR tức là 3 khớp quay(các cấu trúc 11,12) đây là các cấu trúc hoạt trong toạ độ góc so với hệ So, vì điểm M trên khâu 3 được xác định bởi ba chuyển động quay(tức là ba toạ độ góc), cấu trúc này được gọi là cấu trúc phỏng sinh học



Hình 1-1: Một vài cấu trúc của tay máy thường gặp

Tuy nhiên trong thực tế, đối với các tay máy chuyên dùng ta chuyên môn hoá và đặc biệt đảm bảo giá thành và giá đầu tư vào tay máy thấp, người ta không nhất thiết lúc nào cũng phải chế tạo tay máy có đủ số ba khớp động cho cấu trúc xác định vị trí.

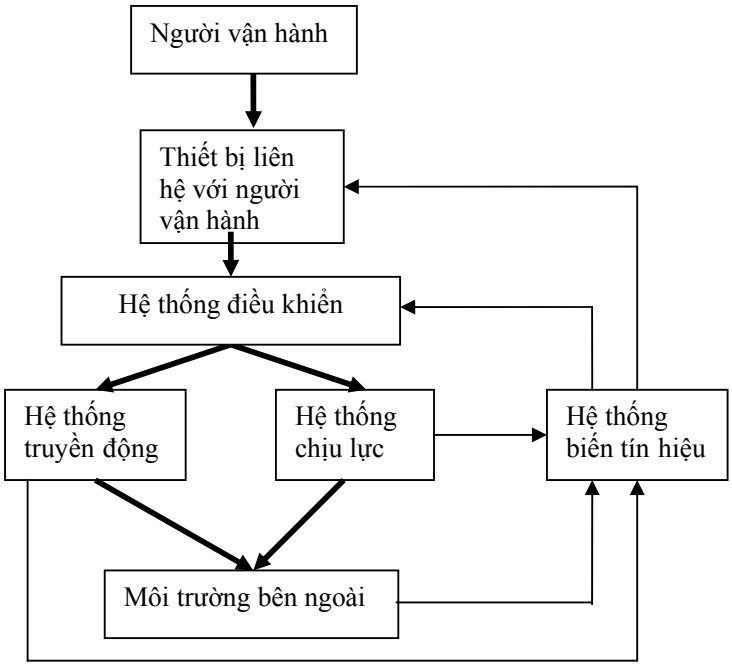
Đối với tay máy công nghiệp đã có hơn 250 loại, trong số đó có hơn 40% là loại tay máy có điều khiển đơn giản thuộc thế hệ thứ nhất.

Sự xuất hiện của IR và sự gia tăng vai trò của chúng trong sản xuất và xã hội loài người làm xuất hiện một nghành khoa học mới là nghành Robot học(Robotic). Trên thế giới nhiều nời đã xất hiện những viện nghiên cứu riêng về Robot.

Ở Việt Nam, từ những năm giữa thập kỷ 80 đã có viện nghiên cứu về Robot.

1.3 Sơ đồ cấu trúc chức năng của Robot:

Vậy Robot là gì? Cho tới hiện nay có rất nhiều định nghĩa về Robot, và hằng năm người ta tổ chức rất nhiều hội nghị khoa học bàn về Robot, nhằm thông tin những thành tựu đã đạt được trong nghiên cứu và chế tạo Robot đồng thời thống kê các thuật ngữ về Robotic, để hiểu được về IR trước hết chúng ta quan sát sở đồ cấu trúc và chức năng của IR như sau:



Hình 1-2. Sơ đồ cấu trúc và chức năng của Robot.

Trong sơ đồ trên, các đường ⟶ chỉ mối quan hệ thông tin thuận, thông tin chỉ huy nhiệm vụ Robot. Các đường ⟶ chỉ mối liên hệ thông tin ngược, thông tin phản hồi về quá trình làm việc của Robot.

Chức năng của bộ phận giao tiếp là liên lạc với người vận hành là thực hiện quá trình "dạy học" cho Robot, nhờ đó Robot biết được nhiệm vụ phải thực hiện.

Chức năng của hệ thống điều khiển là thực hiện việc tái hiện lại các hành động nhiệm vụ đã được "học".

Bộ phận chấp hành giúp cho Robot có đủ "sức" chịu được tải trọng mà Robot phải chịu trong quá trình làm việc, bộ phận này bao gồm:

Phần 1: Bộ phận chịu chuyển động , phần tạo các khả năng chuyển động cho Robot.

Phần 2: Bộ phận chịu lực, phần chịu lực của Robot.

Bộ cảm biến tín hiệu: làm nhiệm vụ nhận biết, đo lường và biến đổi thông tin các loại tín hiệu như : các nội tín trong bản thân Robot, đó là các tín

hiệu về vị trí, vận tốc, gia tốc, trong từng thành phần của bộ phận chấp hành các ngoại tín hiệu, là các tín hiệu từ môi trường bên ngoài có ảnh hưởng tới hoạt động của Robot.

Với cấu trúc và chức năng như trên, Robot phần nào mang đặc tính của con người còn phần máy chính là trạng thái vật lý của cấu trúc.

Với IR các tính chất trên cũng được thể hiện đầy đủ, do đó IR duy trì hình thức mang dáng dấp của tay "người".

Tay máy công nghiệp thường có những bộ phận sau:

Hệ thống điều khiển: thường là loại đơn giản làm việc có chu kỳ vận hành theo nguyên lý của hệ thống điều khiển hở hoặc kín.

Hệ thống chấp hành: bao gồm các nguồn động lực, hệ thống truyền động, hệ thống chịu lực như: các động cơ thuỷ, khí nén, cơ cấu servo điện tử, động cơ bước. Mỗi chuyển động của IR thường có một động cơ riêng và các thanh chịu lực.

- Bàn kẹp: là bộ phận công tác cuối cùng của tay máy, nơi cầm nắm các thiết bị công nghệ háy vật cần di chuyển.

1.4 Ứng dụng Robot trong công nghiệp:

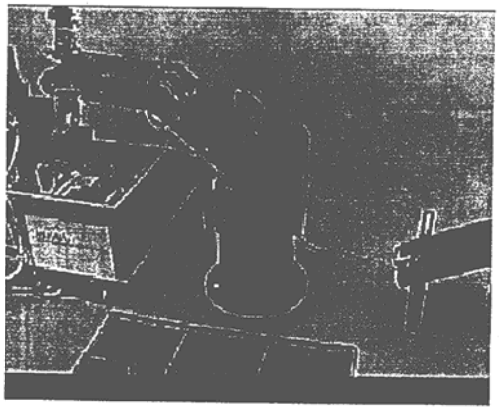
1.4.1 Mục tiêu ứng dụng Robot trong công nghiệp:

Nhằm góp phần nâng cao năng suất dây truyền công nghệ, giảm giá thành, nâng cao chất lượng và khả năng cạnh tranh của sản phẩm, đồng thời cải thiện lao động. Điều đó xuất phát từ những ưu điểm cơ bản của Robot và đã được đúc kết qua nhiều năm được ứng dụng ở nhiều nước.

Những ưu điểm đó là:

- Robot có thể thực hiện một quy trình thao tác hợp lý, bằng hoặc hơn một người thợ lành nghề một cách ổn định trong suốt thời gian làm việc. Vì thế Robot có thể nâng cao chất lượng và khả năng cạnh tranh của sản phẩm. Hơn thế nữa Robot còn có thể nhanh chóng thay đổi công việc, thích nghi nhanh với việc thay đổi mẫu mã, kích cỡ của sảm phẩm theo yêu cầu của thị trường cạnh tranh.

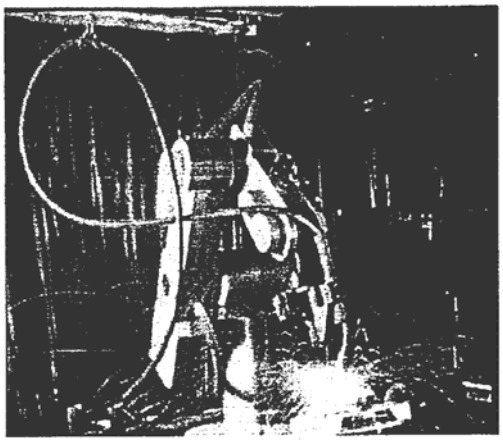
- Có khả năng giảm giá thành sản phẩm do ứng dụng Robot là bởi vì giảm được đáng kể chi phí cho người lao động nhất là ở các nước có mức cao về tiền lương của người lao động, cộng các khoản phụ cấp và bảo hiểm xã hội. Theo số liệu của Nhật Bản thì Robot làm việc thay cho một người thợ thì tiền mua Robót chỉ bằng tiền chi phí cho người thợ trong vòng 3-5 năm, tuỳ theo Robot làm việc ngày mấy ca. Còn ở Mỹ, trung bình trong mỗi giờ làm việc Robot có thể đem lại tiền lời là 13 USD. Ở nước ta trong những năm gần đay có nhiều doanh nghiệp, khoản chi phí về lương bổng cũng chiếm tỷ lệ cao trong giá thành sản phẩm.



Hình 1-2 *Ứng dụng robot phục vụ máy công cụ*

- Việc ứng dụng Robot có thể làm tăng năng suất của dây truyền công nghệ. Sở dĩ như vậy vì nếu tăng nhịp độ khẩn trương của dây truyền sản xuất, nếu không thay thế con người bằng Robot thì thợ không thể theo kịp hoặc rất

chóng mệt mỏi. Theo tài liệu của Fanuc-Nhật Bản thì năng xuất có khi tăng 3 lần.



Hình 1-3: Ứng dụng robot trong công nghệ hàn

- Ứng dụng Robot có thể cải thiện được điều kiện lao động. Đó là ưu điểm nổi bật nhất mà chúng ta cần quan tâm. Trong thực tế sản xuất có rất nhiều nơi người lao động phải lao động suốt buổi trong môi trường bụi bặm, ẩm ướt, nóng nực, hoặc ồn ào quá mức cho phép nhiều lần. Thậm trí ở nhiều nơi người lao động còn phải làm việc dưới môi trường độc hại, nguy hiểm đến sức khoẻ con người, dễ xảy ra tai nạn, dễ bị nhiễm hoá chất độc hại, nhiễm sóng điện từ, phóng xạ...

1.4.2 Các bước ứng dụng Robot:

Việc ưu tiên đầu tư trước hết để nhằm để đồng bộ hoá cả hệ thống thiết bị, rồi tự động hoá và Robot hoá chúng khi cần thiết để quyết định đầu tư cho cả dây truyền công nghệ hoặc chỉ ở một vài công đoạn. Người ta thường xem xét các mặt sau:

- Nghiên cứu quá trình công nghệ được Robot hoá và phân tích toàn bộ hệ thống nếu không thể hiện rõ thì việc đầu tư robot hoá là chưa nên.

- Xác định các đối tượng cần Robot hoá:

Khi xác định cần phải thay thế Robot ở những nguyên công nào thì phải xem xét khả năng liệu Robot có thay thế được không và có hiệu quả hơn không. Thông thường người ta ưu tiên ở những chỗ làm việc quá nặng nhọc,

bụi bặm ồn ào, độc hại, căng thẳng hoặc quá đơn điệu. Xu hướng thay thế hoàn toàn bằng Robot thực tế không hiệu quả bằng việc giữ lại một số công đoạn mà đòi hỏi sự khéo léo của con người.

- Xây dựng mô hình quá trình sản xuất đã được Robot hoá:

Sau khi đã xác định được mô hình tổng thể quá trình công nghệ, cần xác định rõ dòng chuyển dịch nguyên liệu và dòng thành phẩm để đảm bảo sự nhịp nhàng đồng bộ của từng hệ thống. Có thế mới phát huy được hiệu quả đầu tư vốn.

- Chọn lựa mẫu robot thích hợp hoặc chế tạo robot chuyên dùng. Đây là bước quan trọng vì robot có rất nhiều loại với giá tiền khác nhau. Nếu như không chọn đúng thì không những đầu tư quá đắt mà còn không phát huy được hết khả năng, như kiểu dùng người không đúng chỗ. Việc này thường xảy ra khi mua robot nước ngoài, có những chức năng robot được trang bị nhưng không cần dùng cho công việc cụ thể mà nó đảm nhiệm dây truyền sản xuất, vì thế mà đội giá lên rất cao, chỉ có lợi cho nơi cung cấp thiết bị.

Cấu trúc robot hợp lý nhất là cấu trúc theo modun hoá, như thế có thể hạ được giá thành sản xuất, đồng thời đáp ứng được nhu cầu phục vụ công việc đa dạng. Cấu trúc càng đơn giản càng dễ thực hiện với độ chính xác cao và giá thành hạ. Ngoài ra còn có thể tự tạo dựng các robot thích hợp với công việc trên cơ sở mua lắp các modun chuẩn hoá. Đó là hướng triển khai hợplý đối với đại bộ phận xí nghiệp trong nước hiện nay cung như trong tương lai.

1.4.3 Các lĩnh vực ứng dụng robot trong công nghiệp.

- Một trong các lĩnh vực hay ứng dụng robot là kỹ nghệ đúc. Thường trong phân xưởng đúc công việc rất đa dạng, điều kiện làm việc nóng nực, bụi bặm, mặt hàng thay đổi luôn và chất lượng vật đúc phụ thuộc nhiều vào quá trình thao tác.

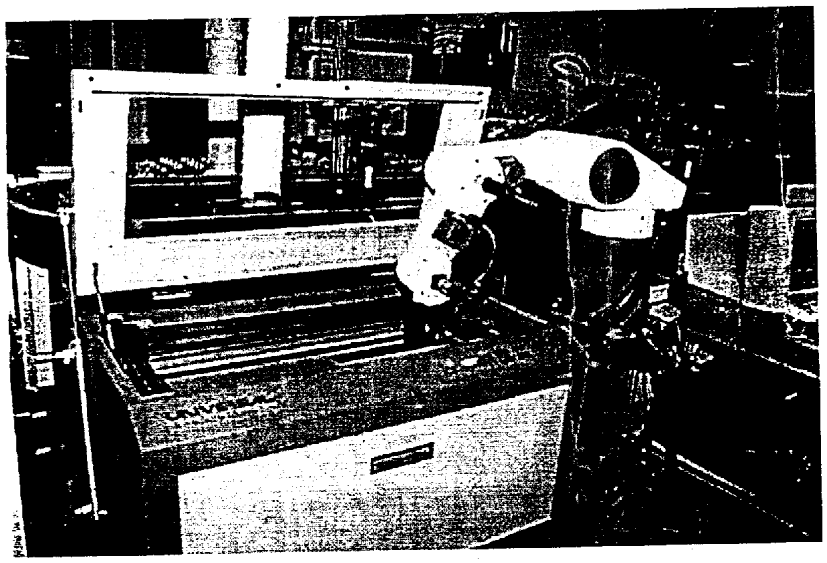
Việc tự động hoá toàn phần hoặc từng phần quá trình đúc bằng các dây truyền tự động thông thường với các máy tự động chuyên dùng đòi hỏi phải có các thiết bị phức tạp, đầu tư khá lớn. Ngày náy ở nhiều nước trên thế giới robot được dùng rộng rãi để tự động hoá công nghệ đúc, nhưng chủ yếu là để

phục vụ các máy đúc áp lực. Robot có thể làm được nhiều việc như rót kim loại nóng chảy vào khuôn, cắt mép thừa, làm sạch vật đúc hoặc làm tăng bền vật đúc bằng cách phun cát... Dùng robot phục vụ các máy đúc áp lực có nhiều ưu điểm: đảm bảo ổn định chế độ làm việc, chuẩn hoá về thời gian thao tác, về nhiệt độ và điều kiện tháo vật đúc ra khỏi khuôn ép... bởi thế chất lượng vật đúc tăng lên.

Trong nghành gia công áp lực điều kiện làm việc cũng khá nặng nề, dễ gây mệt mỏi nhất là ở trong các phần xưởng rèn dập nên đòi hỏi sớm áp dụng robot công nghiệp. Trong phân xương rèn, robót có thể thực hiện những công việc: đưa phôi thừa vào lò nung, lấy phôi đã nung ra khỏi lò, mang nó đến máy rèn, chuyển lại phôi sau khi rèn và xếp lại vật đã rèn vào giá hoặc thùng... Sử dụng các loại robot đơn giản nhất cũng có thể đưa năng xuất lao động tăng lên 1,5-2 lần và hoàn toàn giảm nhẹ lao động của công nhân. So với các phương tiện cơ giới và tự động khác phục vụ các máy rèn dập thì dùng robot có ưu điểm là nhanh hơn, chính xác hơn và cơ động hơn.

- Các quá trình hàn và nhiệt luyện thường bao gồm nhiều công việc nặng nhọc, độc hại và ở nhiệt độ cao. Do vậy ở đây cũng nhanh chóng ứng dụng robot công nghiệp.

Khi sử dụng robot trong việc hàn, đặc biệt là hàn hồ quang với mối hàn chạy theo đường cong không gian cần phải đảm bảo sao cho điều chỉnh được phương và khoảng cách của điện cực so với mặt phẳng của mối hán. Nhiệm vụ đó cần được xem xét khi tổng hợp chuyển động của bàn kẹp và xây dựng hệ thống điều khiển có liên hệ phản hồi. Kinh nghiệm cho thấy rằng có thể thực hiện tốt công việc nếu thống số chuyển động của đầu điện cực và chế độ hàn được điều khiển bằng một chương trình thống nhất, đồng thời nếu được trang bị các bộ phận cảm biến, kiểm tra và điều chỉnh. Ngoài ra robot hàn còn phát huy tác dụng lớn khi hàn trong những môi trường đặc biệt.



ện

- Robot được dùng khá rộng rãi trong gia công và lắp ráp. Thường thường người ta sử dụng robot chủ yếu vào các việc tháo lắp phôi và sản phẩm cho các máy gia công bánh răng, máy khoan, máy tiện bán tự động...

Trong nghành chế tạo máy và dụng cụ đo chi phí về lắp giáp thường chiếm đến 40% giá thành sản phẩm. Trong khi đó mức độ cơ khí hoá lắp ráp không quá 10-15% đối với sản phẩm hàng loạt và 40% đối với sản xuất hàng loạt lớn. Bởi vậy, việc tạo ra và sử dụng robot lắp ráp có ý nghĩa rất quan trọng.

Phân tích quá trình lắp ráp chúng ta thấy rằng con người khi gá đặt các chi tiết để lắp chúng với nhau thì có thể làm nhanh hơn các thiết bị tự động. Nhưng khi thực hiện các động tác khác trong quá trình ghép chặt chúng thì chậm hơn. Bởi vậy yếu tố thời gian và độ chính xác định vị là vấn đề quan trong cần quan tâm nhất khi thiết kế các loại robot lắp ráp. Ngoài yêu cầu hiện nay đối với các loại robot lắp ráp và nâng cao tính linh hoạt để đáp ứng nhiều loại công việc, hạ giá thành và dễ thích hợp với việc sản xuất loạt nhỏ.

Ngày nay đã xuất hiện nhiều loại day truyền tự động gồm các máy vạn năng với robot công nghiệp. Các day truyền đó đạt mức độ tự động cao, tự động hoàn toàn, không có con người trực tiếp tham gia, rất linh hoạt và không

đòi hỏi đầu tư lớn. ở đây các nhà máy và robot trong dây truyền được điều khiển bằng cùng một hệ thống chương trình.

Trong một dây truyền tự động có các máy điều khiển theo chương trình robot có thể đứng một chỗ điều chỉnh trên đường ray hoặc theo di động.

Kỹ thuật robot có ưu điểm quan trọng nhất là tạo nên khả năng linh hoạt hoá sản xuất. Việc sử dụng máy tính điện tử, robot và máy điều khiển theo chương trình đã cho phép tìm được những phương thưc mới mẻ để tạo nên các dây truyền tự động cho sản xuất hàng loạt với nhiều mẫu, loại sản phẩm. Dây truyền tự động "cứng" gồm nhiều thiết bị tự động chuyên dùng đòi hỏi vốn đầu tư lớn, nhiều thời gian để thiết kế và chế tạo trong lúc quy trình công việc luôn luôn cải tiến, nhu cầu đối với chất lượng và quy cách của sản phẩm luôn luôn thay đổi. Bởi vậy nhu cầu " mềm" hóa hay là linh hoạt hoá dây truyền sản xuất ngày càng tăng. Kỹ thuật công nghiệp và máy tính đã đóng vai trò quan trọng trong việc tạo ra các dây truyền tự động linh hoạt.

Xuất phát từ nhu cầu và khả năng linh hoạt hoá sản xuất, trong những năm gần đây không chỉ các nhà khoa học mà cả các nhà sản xuất đã tập trung sự chú ý vào việc hình thành và áp dụng các hệ sản xuất tự động linh hoạt, gọi tắt là hệ sản xuất linh hoạt. Hệ sản xuất linh hoạt ngày nay thường bao gồm các thiết bị gia công được điều khiển bằng chương trình số, các phương tiện vận chuyển và kho chứa trong phân xưởng đã được tự động hoá và nhóm robot công nghiệp ở vị trí trực tiếp với các thiết bị gia công hoặc thực hiện các nguyên công phụ. Việc điều khiển và kiểm tra điều khiển toàn hệ sản xuất linh hoạt là rất thích hợp với quy mô sản xuất nhỏ và vừa, thích hợp với yêu cầu luôn luôn thay đổi chất lượng sảm phẩm và quy trình công nghệ. Bởi vậy ngày nay hệ sản xuất linh hoạt thu hút sự chú ý không những ở các nước phát triển mà ngay cả ở các nước đang phát triển. Trong một số tài liệu nước ngoài hệ FMS (flexible Manufacturing System) nay được diễn giải như hệ sản xuất của tương lai (future Manufacturing System), sự trùng hợp các từ viết tắt này không phải ngẫu nhiên.

Tỷ lệ phân bố các loại công việc được dùng robot:

1. Đúc áp lực	18,3%
2. Hàn điểm	14,7%
3. Hàn hồ quang	12,3%
4. Cấp thoát phôi	9,6%
5. Lắp ráp	9,5%
6. Nghiên cứu, đào tạo	5,7%
7. Phun phủ bề mặt	5,7%
8. Nâng chuyển sắp xếp	3,9%
9. Các việc khác	30,3%

Sự phân bố tỷ lệ các loại robot với các loại phương pháp điều khiển khác nhau:

a. Tay máy điều khiển bằng tay: 4%

b. Robot được điều khiển theo chu kỳ cứng (máy tự động): 5%

c. Robot được điều khiển dùng chương trình dạy học: 29%

d. Robot điều khiển theo chương trình số: 59%

e. Robot được điều khiển có sử lý tinh khôn: 3%

1.4.4 Nội dung nghiên cứu phát triển Robot công nghiệp:

1.4.4.1 Nhận xét về quá trình phát triển robot công nghiệp.

Ra đời từ những năm năm mười, robot công nghiệp đã có những bước phát triển quan trọng. Từ những năm 1960 do sự phát hiện máy vi tính robot công nghiệp đã tiếp thu được thành tựu mới đó và ngày càng hấp dẫn. Cao trào phát triển vào những năm 70 và đánh dấu bằng hội nghị quốc tế lần thứ 6 về " thiết kế chế tạo và ứng dụng robot công nghiệp" Chicago năm 1972, sau

đó lại lắng dần xuống, nhất là sau khủng hoảng dầu mỏ 1975, như để rút kinh nghiệp áp dụng vào chỗ nào là phát huy hiệu quả hơn. Đến những năm 80 thì xuất hiện nhu cầu hình thành các hệ thống sản xuất linh hoạt FMS (Flexible Manufacturing System) mà robot như là bộ phận cấu thành FMS. Nhu cầu đó kích thích sự phát triển của robot công nghiệp. Trong năm 90 robot công nghiệp cũng có bước phát triển mới theo hướng đồng bộ hệ thống trên cơ sở vận dụng những thành tựu của công nghệ thông tin ứng dụng.

Bản thân phần kỹ thuật robot công nghiệp cũng thể hiện các xu thế phát triển sau đây:

1/ Trong giai đoạn đầu phát triển, người ta rất quan tâm đến việc tạo ra những cơ cấu tay máy nhiều bậc tự do, được trang bị nhiều loại cảm biến(sensor) để có thể thực hiện được những công việc phức tạp, như là để chứng tỏ khả năng thay thế con người trong nhiều loại hình công việc.

2/ Khi đã tìm được các địa chỉ ứng dụng trong công nghiệp, thì việc đơn giản hoá kết cấu để tăng độ chính xác định vị và giảm giá thành đầu tư lại là những yêu cầu thực tế đối với thị trường hành hoá cạnh tranh. Ngày càng có nhiều cải tiến trong kết cấu các bộ phận chấp hành, tăng độ tin cậy của các thiết bị điều khiển, tăng mức thuận tiện và dễ dàng khi lập trình...

3/ Để mở rộng phạm vi ứng dụng cho robot công nghiệp nhằm thay thế lao động với nhiều loại hình công việc, ngày càng rõ nét về xu thế tăng cường khả năng nhận biết và xử lý tín hiệu từ môi trường làm việc. Các thành tựu khoa học và tiến bộ kỹ thuật laser,kỹ thuật tia hồng ngoại, kỹ thuật xử lý ảnh... đã ngày càng hiện thực xu thế phát triển robot công nghiệp hướng vào việc thích nghi được với môi trường làm việc.

4/ Cùng với các xu thế trên robot công nghiệp luôn luôn được định hướng tăng cường năng lực xử lý công việc để trở thành các robot tinh khôn nhờ áp dụng các kết quả nghiên cứu về hệ điều khiển nơron và trí khôn nhân tạo...

1.4.4.2 Cơ-tin-điện tử và robot công nghiệp.

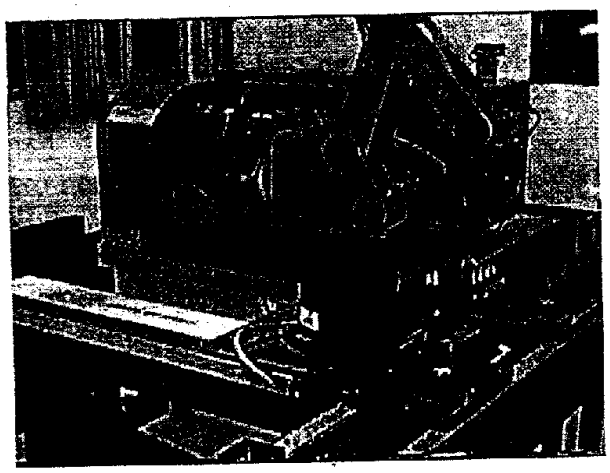
Cơ-tin-điện tử và robot công nghiệp là hai lĩnh vực khoa học kỹ thuật cao rất gắn bó với nhau.ở một số nước chúng kết hợp với nhau như là một ngành học.Trong robot công nghiệp có hầu hết các vấn đề của cơ điện tử. Đồng thời phát triển của cơ điện tử cũng đều phản ánh trong kỹ thuật robot. Vì vậy để nghiên cứu về robot cần xem xét các vấn đề về cơ-tin-điện tử.

Thuật ngữ cơ-tin-điện tử(mechartonics)thể hiện sự kết hợp giữa cơ học máy với công nghệ thông tin vi điện tử học(microelectronics).ý tưởng chủ yếu ban đầu của cơ-tin-điện tử là cài lên các hệ máy,các thiết bị điện tử rồi dần dần bản thân bên trong máy cũng thay đổi đi và chức năng của máy cũng được mở rộng thêm nhiều. Còn về thiết bị điện tử chính xác hơn và vi điện tử thì các tiến bộ mới cũng không ngừng được áp dụng. Từ các mạch tích phân IC(Intergrated circuit), các độ vi sử lý (Microposessor), các bộ điều khiển lập trình được PLC, các máy tính PC...

Phần nối ghép thành hệ thống giữa các thiết bị điều khiển vi điển tử với các thiết bị chấp hành trong máy(có thể là các thiết bị cơ, thiết bị thuỷ khí, thiết bị điện hoặc điển từ) chủ yếu là các bộ cảm biến (sensor), các bộ biến đổi (converter) và các thiết bị của công nghệ thông tin.

1.4.4.3 Robot và hệ sản xuất linh hoạt.

Nhu cầu của thị trường cạnh tranh luôn luôn đòi hỏi các nhả sản xuất phải thay đổi mẫu mã, kích cỡ và thường xuyên cải tiến nâng cao chất lượng sản phẩm.



Hình 1-7

Hình 1- 5: Ứng dụng robot trong dây chuyền sản xuất tự động

Như vậy sự cạnh trang hành hoá đặt ra một vấn đề thời sự là phải có hệ thống thiết bị sản xuất thay đổi linh hoạt được để có thể đáp ứng được với sự biến động thường xuyên của thị trường. Nhờ sự phát triển trong mấy chục năm gần đây của kỹ thuật số và công nghệ thông tin chúng ta mới có khả năng " mềm" hoá hệ thống thiệt bị sản xuất. Trên cơ sở đó đã ra đời hệ thống sản xuất linh hoạt FMS là phương thức sản xuất linh hoạt hiện đại. Nó có ưu điểm là các thiết bị chủ yếu của hệ thống chỉ đầu tư một lần, còn đáp ứng lại sự thay đổi của sản phẩm bằng phần mềm máy tính điều khiển là chính. Hệ thông FMS rất hiện đại nhưng lại thích hợp với quy mô sản xuất vừa và nhỏ. Ngày nay các nước phát triển các hệ thống FMS có xu hướng thay thế dần các thiết bị tự động " cứng" sản xuất hàng loạt lớn sản phẩm. Các hệ thống thiết bị tự động cứng này rất đắt tiền mà khi thay đổi về yêu cầu sản phẩm thì phản đổi mới gần như hoàn toàn. Như vậy, chúng nhanh chóng trở nên lạc hậu vì không thích nghi được với thị trường đầy biến động.

Ý tưởng chủ đạo trong việc tổ chức hệ thống sản xuất hiện đại linh hoạt là " linh hoạt hoá" và "modul hoá". Một hệ thống sản xuất linh hoạt có thể gồm nhiều modul linh hoạt. Một trong những hệ thống như vậy là hệ thống CIM (Computer Intergrated Manufacturing)- Hệ thống tích hợp sản xuất dùng máy tính.

Để tạo ra các modul sản xuất linh hoạt đó cần có một robot như một bộ phận cấu thành. ở đây, robot làm những công việc chuyển tiếp giữa các máy công tác (ví dụ cấp, thoát phôi và dụng cụ cho các máy công tác).

Bản thân cơ cấu tay máy của robot cũng là một cơ cấu linh hoạt. Đó là cơ cấu không gian hở (không khép kín), có bậc tự do dư thừa nên độ cơ động

rất cao. Mỗi khâu của cơ cấu có động lực riêng và chúng được điều khiển băng chương trình thay đổi được. Có loại robot lại có thể tự thay đổi thao tác của mình một cách linh hoạt khi nhận biết được các tín hiệu từ sự hoạt động của bản thânn (nội tín hiệu). Những cơ cấu như vậy là cơ cấu điều khiển linh hoạt.

1.4.4.4 Robot song song:

Sơ đồ động cơ cấu tay máy thông thường là một chuỗi nối tiếp các khâu động, còn trong robot song song (RBSS) ở một khâu nào đó có thể nối động với các khâu khác, tức là nối song song với nhau và cùng hoạt động song song với nhau. Sự khác nhau về sơ đồ động đó cũng gây nhiều đặc điểm khác biệt về động học và động lực học. Ví dụ với robot thông thường thì giải bài toán động học thuận sẽ dễ dàng hơn nhiều so với bài toán động học ngược, còn với robot song song thì hoàn toàn ngược lại.

Vấn đề RBSS trở nên hấp dẫn nhiều nhà nghiên cứu từ giữa thập kỷ 90 khi nó được ứng dụng dưới dạng thiết bị có tên là Hexapod để tạo ra máy công cụ 5 trục CNC có trục ảo. Hexapod là một modul RBSS được kết cấu trên nguyên lý cơ câú Stewart. Cơ cấu này gồm có 6 chân có độ dài thay đổi được, nối với giá và tấm động theo ý muốn. Stewart đã đề xuất sử dụng cơ cấu này để mô phỏng hoạt động của thiết bị bay.

Như đã biết, máy cắt gọt CNC 3 trục không đáp ứng được nhu cầu gia công chính xác các bề mặt phức tạp. Vì thế xuất hiện nhu cầu tạo ra các máy CNC5 trục, tức là ngoài các trục X,Y,Z bổ xung thêm hai trục quay có thể thực hiện được trên bàn máy trên vật gia công hoặc trên giá đỡ trục dụng cụ cắt. Các máy CNC 5 trục này rất đắt tiền, gần gấp đôi máy CNC 3 trục. Nếu sử dụng cơ cấu Hexapod để tạo ra các trục hoặc tạo ra các trục bổ xung thì giá thành máy CNC trục ảo này có thể hạ thấp rất nhiều lần.

Ngoài các ứng dụng trong nghành chế tạo máy, công cụ RBSS còn được áp dụng hiệu quả trong dụng cụ y học, trong hệ thống mô phỏng, trong thiết bị thiên văn và trong kỹ thuật phòng không ...

1.4.4.5 Các xu thế ứng dụng robot trong tương lai:

Robot ngày càng thay thế nhiều lao động

Ở đây chỉ đề cập đến robot công nghiệp. Trong tương lai, kỹ thuật robot sẽ tận dụng hơn nữa các thành tựu khoa học liên nghành, phát triển cả về phần cứng, phần mềm và ngày càng chiếm lĩnh nhiều lĩnh vực trong công nghiệp.

Số lượng lao động được thay thế ngày càng nhiều vì: càng ngày giá thành robot càng giảm, mặt khác chi phí tiền lương và các khoản phụ khác cấp cho người lao động ngày càng cao.

Robot ngày càng trở nên chuyên dụng:

Khi robot công nghiệp ra đời, người ta thương cố gắng làm sao để biểu thị hết khả năng của nó. Vì thế xuất hiện rất nhiều loại robot vạn năng có thể làm được nhiều việc trên dây truyền. Tuy nhiên thực tế sản xuất chứng tỏ rằng, các robot chuyển môn hoá đơn giản hơn, chính xác hơn, học việc nhanh hơn và quan trong là rẻ tiền hơn robot vạn năng. Các robot chuyên dụng hiện đại đều được cấu tạo thành từ các modul vạn năng. Xu thế modul hoá ngày càng phát triển nhằm chuyển môn hoá việc chế tạo các modul và từ các modul đó sẽ cấu thành nhiều kiển robot khác nhau thích hợp cho từng loại công việc.

Robot ngày càng đảm nhận nhiều loại công việc lắp ráp.

Công đoạn lắp ráp thường chiếm tỷ lệ cao so với tổng thời gian sản xuất trên toàn bộ dây truyền. Công việc khi lắp ráp là phải đòi hỏi rất cẩn thận, không được nhầm lẫn, thao tác nhẹ nhàng, tinh tế và chính xác nên cần thợ có tay nghề cao và phải làm việc căng thẳng suất cả ngày.

Khả năng thay thế người lao động ở những khâu lắp ráp ngày càng hiện thực là do đã áp dụng được nhiều thành tựu mới về khoa học trong việc thiết kế, chế tạo robot. Ví dụ đã tạo ra những cấu hình đơn giản và chính xác trên cơ sở sử dụng các vật liệu mới vừa bền, vừa nhẹ. Trong đó nên kể đến các loại robot như Adept Oen, SCARA,... Đồng thời do thừa hưởng sự phát triển kỹ thuật nhận và biến đổi tín hiệu (sensor), đặc kỹ thuật nhận và sử lý tín hiệu ảnh (vision) cũng như kỹ thuật tin học với các ngôn ngữ bậc cao, robot công nghiệp đã có mặt trên nhiều công đoạn lắp ráp phức tạp. Robot di động ngày càng trở nên phổ biến.

Trong các nhà máy hiện đại, tên gọi phương tiện dẫn đường tự động AVG (automatic Guided Vehicles) đã trở thành quen thuộc. Loại đơn giản là những chiếc xe vận chuyển nội bộ trong phân xưởng được điều khiển theo chương trình với một quỹ đạo định sẵn. Càng ngày các thiết bị loại này cũng được hiện đại hoá nhờ áp dụng kỹ thuật thông tin vô tuyến hoặc dùng tia hồng ngoại... Vì vậy AGV đã có thể hoạt động linh hoạt trong phân xưởng. Đó chính là robot linh động và còn gọi là robocar. Một hướng phát triển linh và quan trọng của robocar là không di chuyển bằng các bánh xe mà bằng chân, thích hợp với mọi địa hình.

Robot đi được bằng chân có thể tự leo thang, là một đối tượng đang rất được chú ý trong nghiên cứu không những định hướng trong công nghiệp hạt nhân hoặc trong kỹ thuật quốc phòng mà ngay cả trong công nghiệp dân dụng thông thường. ở đây việc tạo ra các cơ cấu chấp hành cơ khí bền vững, nhẹ nhàng, chính xác và linh hoạt như chân người lại là đối tượng nghiên cứu chủ yếu.

Robot ngày càng trở nên tinh khôn hơn.

Trí khôn nhân tạo là một vấn đề rất quan tâm nghiên cứu với các mục đích khác nhau. Kỹ thuật robot cũng từng bước áp dụng các kết quả nghiên cứu về trí khôn nhân tạo và đưa vào ứng dụng công nghiệp. Trước hết là sử dụng các hệ chuyên gia, các hệ thị giác nhân tạo, mạng nơron và các phương pháp nhận dạng tiếng nói... Cùng với các thành tựu mới trong nghiên cứu về trí khôn nhân tạo, robot ngày càng có khả năng đảm nhận được nhiều nguyên công dây truyền sản xuất đòi hỏi sự tinh khôn nhất định.

Vấn đề thiết bị cảm biến được nhiều ngành kỹ thuật quan tâm và cũng đạt được nhiều thành tựu mới trong thời kì phát triển sôi động của lĩnh vực vi sử lý. Đó cũng là điều kiện thuận lợi trong việc áp dụng chúng trong kỹ thuật robot nhằm tăng cường khả năng thông minh của thiết bị.

Những loại hình được quan tâm nhiều trong công nghiệp là các robot thông minh có các modul cảm biến để nhận biết được khoảng cách để tránh vật cản khi thao tác, cảm biến nhận biết được màu sắc khi phân loại, cảm biến

được lực khi lắp ráp... Khi được lắp thêm các modul cảm biến này robot được gọi với nhiều tên mới. Vídụ: robot "nhìn được" (vision robot), robot lắp ráp (assembli), robot cảnh báo (alarm robot),...

Để thông minh hoá robot bên cạnh việc cài đặt bổ xung các modul cảm biến "nội tín hiệu" và các modul cảm biến "ngoại tín hiệu" thì đồng thời có thể thông minh hoá robot bằng các chương trình phần mềm có khả năng tự thích nghi và tự xử lý các tình huống...

Như vậy bằng cách bổ xung các modul cảm biến và các phần mền phù hợp có thể nâng cấp cải tiến nhiều loại robot. Tuy nhiên bản thân các robot này phải có các cơ cấu chấp hành linh hoạt chính xác. Ngày nay có nhiều loại robot thông minh không những có thể làm việc trong các phân xưởng công nghiệp mà còn thao tác được ở bên ngoài, trên các địa hình phức tạp như các loại robot vũ trụ (space robot), robot tự hành (walking robot), robot cần cẩu(robot crale), tạo dựng từ các modul robot song song...

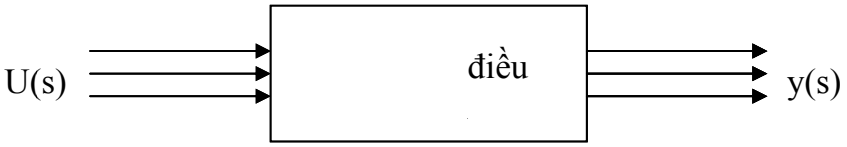
CHƯƠNG 2

LÝ THUYẾT ĐIỀU KHIỂN TỰ ĐỘNG RÔBỐT CÔNG NGHIỆP

2.1 *Khái niệm chung về hệ thống điều khiển tự động rôbốt công nghiệp*

2.1.1 Khái niệm về hệ thống điều khiển tự động

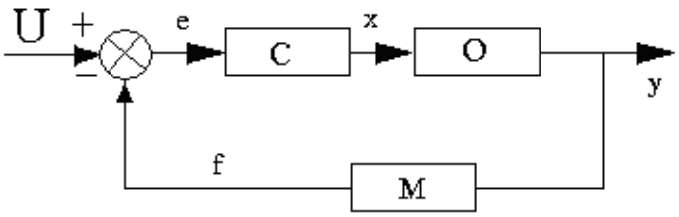
Mỗi hệ thống điều khiển đều có tác động vào và đáp ứng ra hay còn gọi là tín hiệu ra. Trong mỗi hệ thống đều có thể có nhiều tín hiệu vào và nhiều tín hiệu ra.



Như vậy hệ thống điều khiển tự động là tập hợp các thành phần vật lý có mối liên quan và tác động qua lại lẫn nhau để chi huy, hiệu chỉnh bản thân hoặc điều khiển hệ thống khác.

Một hệ thống điều khiển tự động thường có các thành phần như sau:

- *Thiết bị điều khiển (M).*
- *Đối tượng điều khiển (O).*
- *Thiết bị đo lường (M).*

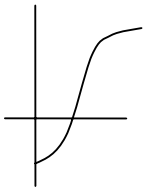
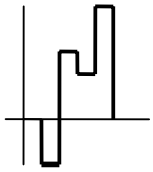
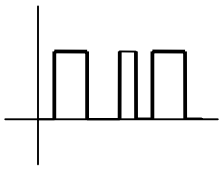
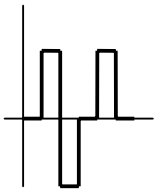


Hình 2.2: Sơ đồ khối các phần tử điều khiển tự động

Trong đó:

- *U: Tín hiệu đầu vào.*
- *y: Tín hiệu đầu ra.*
- *e: Tín hiệu so sánh.*
- *x: Tín hiệu tác động của đối tượng điều khiển.*
- *f: Tín hiệu phản hồi.*

2.1.2 Các loại tín hiệu điều khiển

Tương tự	Rời rạc		
	Tín hiệu số	Tín hiệu nhị phân	Tín hiệu bộ ba
			

2.2 Phân loại hệ thống điều khiển tự động

2.2.1 Hệ thống điều khiển hở:

Là hệ thống điều khiển trong đó các tín hiệu vào và các tín hiệu ra có tính chất độc lập với nhau. Quãng đường hay góc dịch chuyển tỷ lệ với số xung điều khiển. Độ chênh lệch giữa giá trị thực và giá trị cần không được điều chỉnh, xử lý *(Hình 2.4)*.

Trong đó:

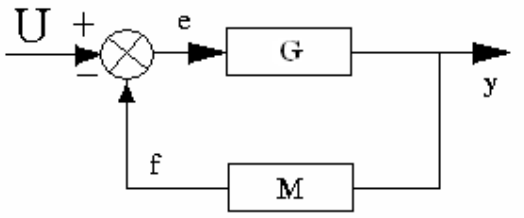
- *U: Tín hiệu đầu vào.*
- *G: Hàm truyền.*
- *y: Tín hiệu đầu ra.*

Đặc điểm của hệ thống hở:

- Độ chính xác phụ thuộc khả năng điều chỉnh và độ tin cậy của thiết bị.
- Bị ảnh hưởng của tác động bên ngoài.
- Tín hiệu ra đáp ứng chậm khi tín hiệu vào thay đổi.

2.2.2 Hệ thống điều khiển kín:

Là hệ thống trong đó các tín hiệu vào và các tín hiệu ra phụ thuộc vào nhau thông qua bộ phản hồi. Tín hiệu phản hồi vị trí có tác dụng tăng độ chính xác điều khiển.



Hình 2.5: Sơ đồ hệ thống điều khiển kín.

Trong đó:

- G: Hàm truyền.
- e: Tín hiệu so sánh.
- f: Tín hiệu phản hồi.
- M: Thiết bị đo lường.

Hệ thống điều khiển kín có các đặc điểm sau:

- Đạt độ chính xác cao.
- Tốc độ đáp ứng nhanh.
- Giảm được tính phi tuyến và nhiễu.
- Tăng được bề rộng dải tần mà tại dải tần này hệ có đáp ứng tốt nhất.

Tuy nhiên với hệ thống này tín hiệu ra có khuynh hướng dao động do quán tính của so sánh tín hiệu.

Trong hệ thống điều khiển kín còn chia ra 2 loại điều khiển: Điều khiển điểm-điểm và điều khiển theo đường.

- Điều khiển điểm-điểm, phần công tác dịch chuyển từ điểm này đến điểm kia theo đường thẳng với tốc độ cao (không làm việc). Nó chỉ làm việc tại điểm dừng. Hệ thống điều khiển kiểu này dùng trên các rôbốt hàn điểm, vận chuyển, tán đinh, bắn đinh, ...
- Điều khiển contour đảm bảo cho phần công tác dịch chuyển theo quỹ đạo bất kỳ, với tốc độ có thể điều khiển được. Có thể gặp hệ thống điều khiển này trên rôbốt hàn hồ quang, phun sơn... .

Ngoài hình thức phân loại trên, người ta còn phân thành:

- *Hệ thống điều khiển tuyến tính.*
- *Hệ thống điều khiển phi tuyến.*
- *Hệ thống điều khiển liên tục.*
- *Hệ thống điều khiển số hay còn gọi là hệ thống điều khiển*

xung-số: Tín hiệu điều khiển là các tín hiệu rời rạc.

- *Hệ thống ngẫu nhiên*.
- *Hệ thống điều khiển tối ưu*: Hệ mà trong đó các thiết bị điều khiển có chức năng tổng hợp để đạt được độ chính xác cao nhất hoặc đạt được thời gian truyền tín hiệu ngắn nhất.

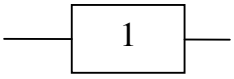
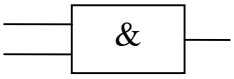
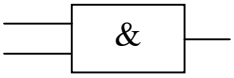
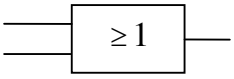
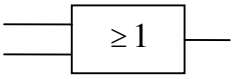
- *Hệ thống điều khiển thích nghi*: Hệ thống tự điều chỉnh, có khả năng thích ứng một cách tự động những biến đổi, tác động của bên ngoài. Hệ có khả năng thay đổi các tham số và cấu trúc một cách tự động để thích nghi với những điều kiện thay đổi bên ngoài.

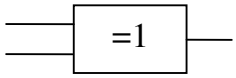
2.3 Phần tử logic trong hệ thống điều khiển tự động

2.3.1 Khái niệm về logic hai trạng thái :

Trong kỹ thuật, đặc biệt là trong điều khiển, ta thường có khái niệm về hai trạng thái: đóng hoặc cắt. Ví dụ: đóng mạch điện (để máy vào làm việc) và tắt máy (để máy nghỉ). Trong toán học để lượng hóa hai trạng thái đối lập của sự việc hay hiện tượng người ta dùng hai giá trị: 0 và 1, giá trị 0 là hàm ý đặc trưng cho đối lập của sự vật hoặc hiện tượng thì giá trị 1 hàm ý đặc trưng cho trạng thái đối lập của sự vật, hiện tượng. Ta gọi đó là giá trị 0 và 1 logíc.

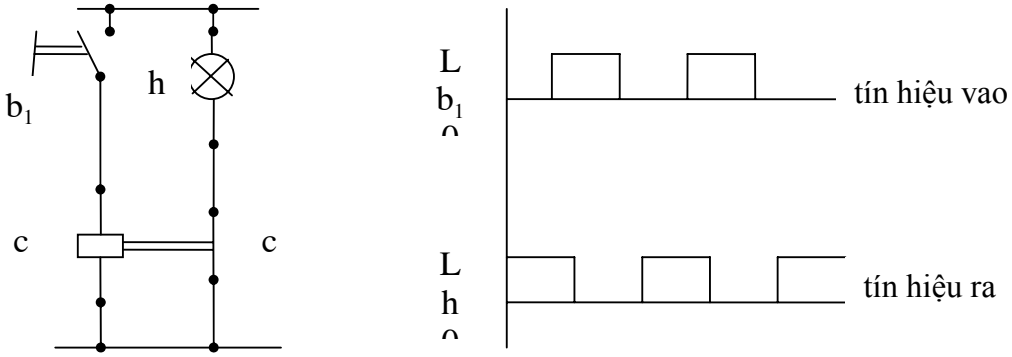
1.Phần tử mạch logic:

Số thứ tự	Ký hiệu	Tên gọi
1		NOT
2		AND
3		NAND
4		OR
5		NOR

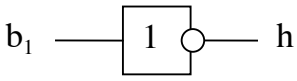
6		XOR

Hình 2- 6: Các phần tử logic cơ bản

a/ Phần tử logic NOT (phủ định)



Ký hiệu



Bảng chân lý

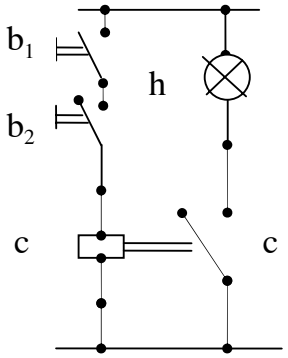
L	h
h	L
o	o

Hình 2-7: phần tử NOT

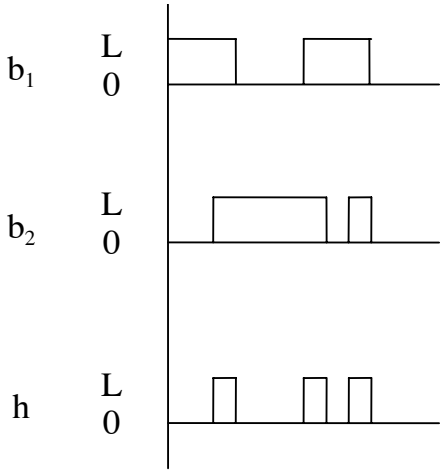
Phần tử logic NOT (minh họa ở phần sau). Khi nhấn b_1 rơle c có điện, bóng đèn mất điện và ngược lại, nhả nút b_1, bóng đèn h sáng.

b/ Phần tử logic AND

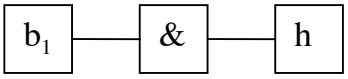
Phần tử logic AND(minh họa ở dưới) . Khi nhấn nut b_1 và đồng thời nhấn b_2 thì rơle c có điện, bóng đèn h sáng. Khi một trong hai hoặc cả hai công tăc b_1, b_2 cùng mở thì rơle c không có điện nên đèn h không sáng.



Sơ đồ tín hiệu



Ký hiệu



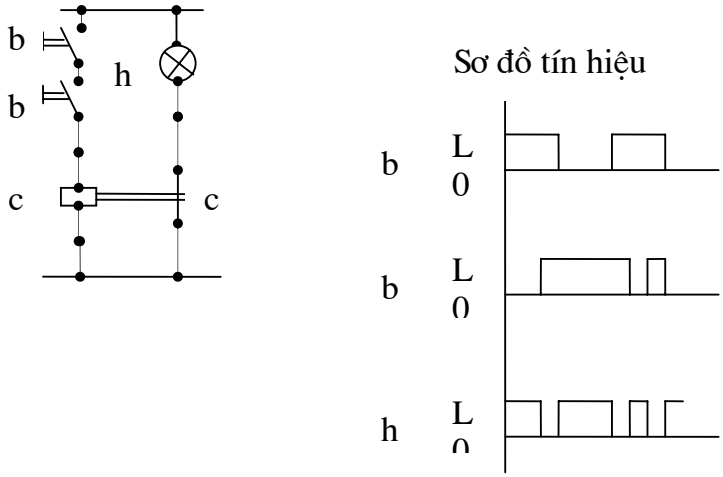
Bảng chân trị

b_1	b_2	h
0	0	0
0	L	0
L	0	0
L	L	L

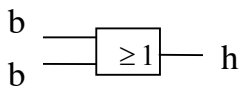
Hình 2-8: Phần tử AND

c/ Phần tử logíc NAND(và - không):

Ta có thể minh họa phần tử này như sau: Khi nhấn nút b_1 và đồng thời nhấn nút b_2, rơle c có điện, đèn h tắt. khi một trong hai nút hoặc cả hai nút mở rơle c không co điện, đèn h sáng.



Ký hiệu



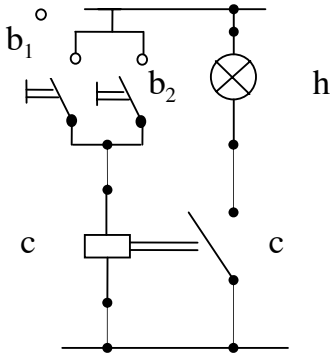
Bảng chân trị

b_1	b_2	h
0	0	0
0	L	L
L	0	L
L	L	0

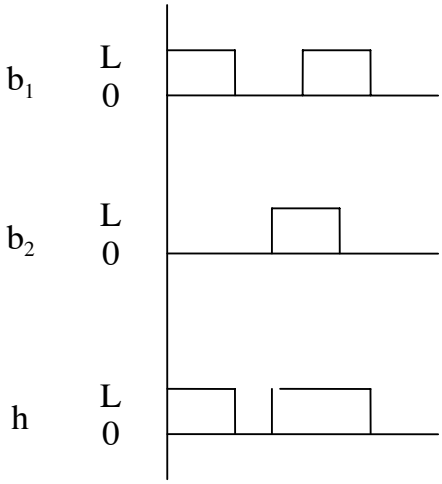
Hình 2-9: phần tử logic NAND

d/ Phần tử logic OR(hoặc):

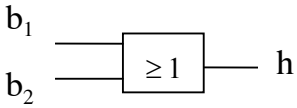
Đèn h sáng lên khi nhấn nút b_1 hoặc b_2. Khi cả hai nút nhấn đều mở, rơle c không có điện, đèn h không sáng.



Sơ đồ tín hiệu



Ký hiệu



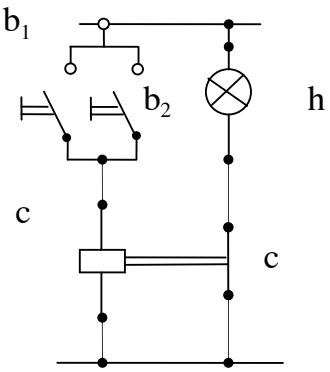
Bảng chân lý

b_1	b_2	h
0	0	0
0	L	L
L	0	L
L	L	L

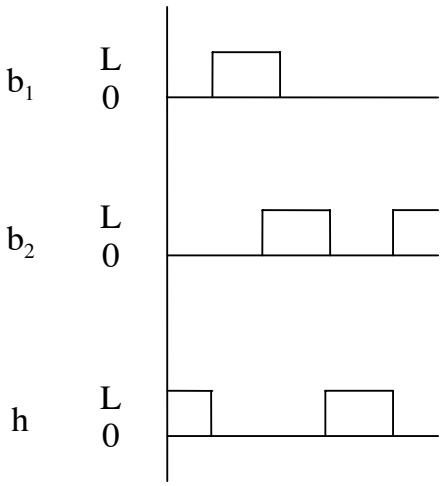
Hình 2-10: phần tử logic OR

e/ Phần tử logic NOR(hoặc-không)

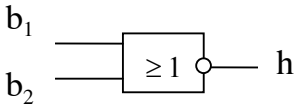
Khi một trong hai nút nhấn b_1,b_2 được thực hiện thì rơle c đều có điện và đèn h bi tắt. Khi không có nút nhấn nào được thực hiện thì rơle c không có điện và đèn h sáng.



Sơ đồ tín hiệu



Ký hiệu



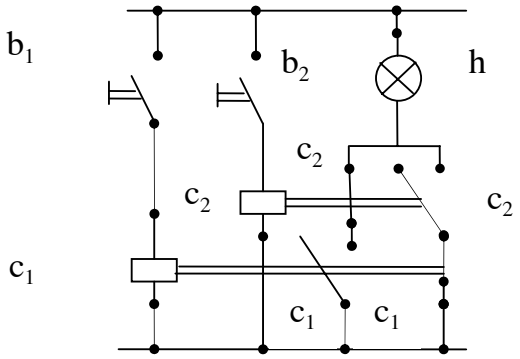
Bảng chân lý

b_1	b_2	h
0	0	L
0	L	0
L	0	0
L	L	L

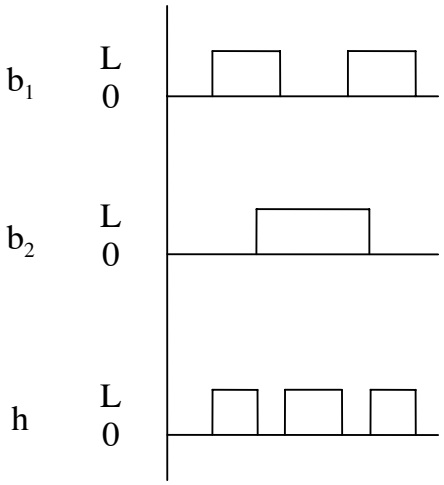
Hình 2-11: Phần tử logic NOR

f/ Phần tử logic XOR (EXC-OR):

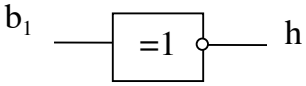
Đèn h sáng khi ta ấn nút b_1 hoặc b_2. Khi cả hai nút đều được nhấn đồng thời thì đèn h sẽ tắt.



sơ đồ tín hiệu



Ký hiệu



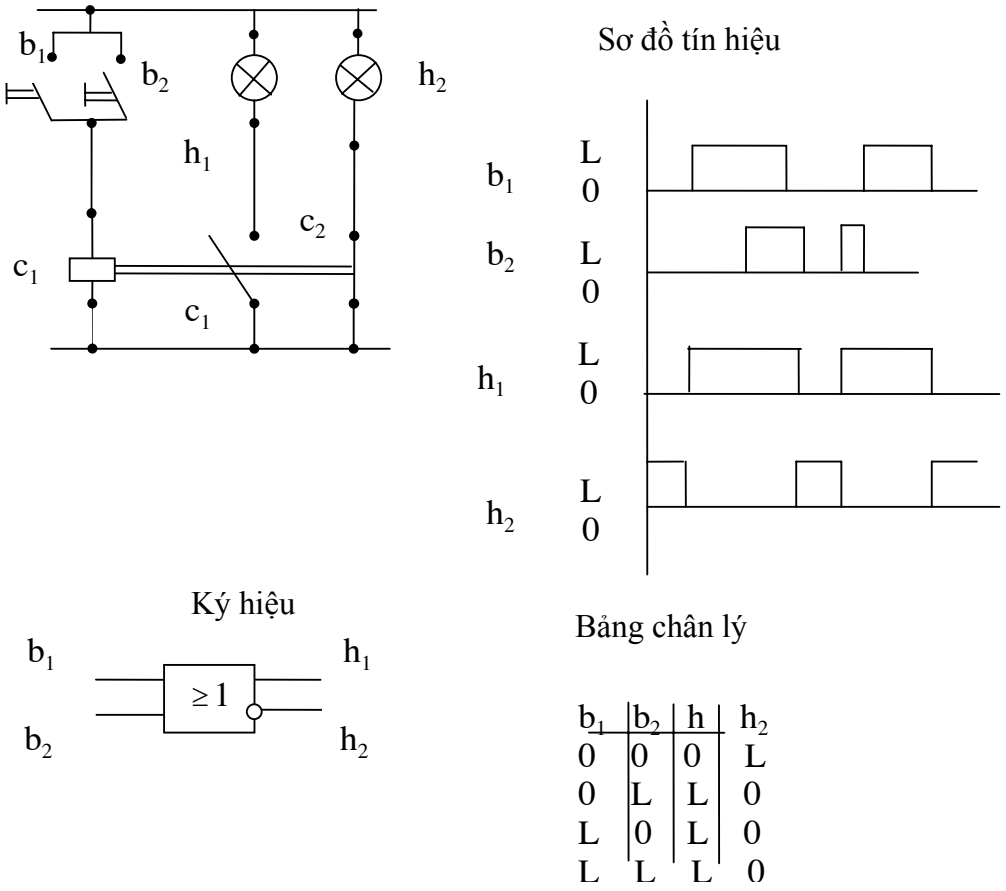
Bảng chân lý

b_1	b_2	h
0	0	0
0	L	L
L	0	L
L	L	0

Hình 2-12: phần tử logic XOR

g/ phần tử logic OR/NOR:

Phần tử này với hai tín hiệu vào b_1 và b_2 và hai tín hiệu ra h_1, h_2 .



b_1	b_2	h	h_2
0	0	0	L
0	L	L	0
L	0	L	0
L	L	L	0

Hình 2-13: phần tử logic OR/NOT

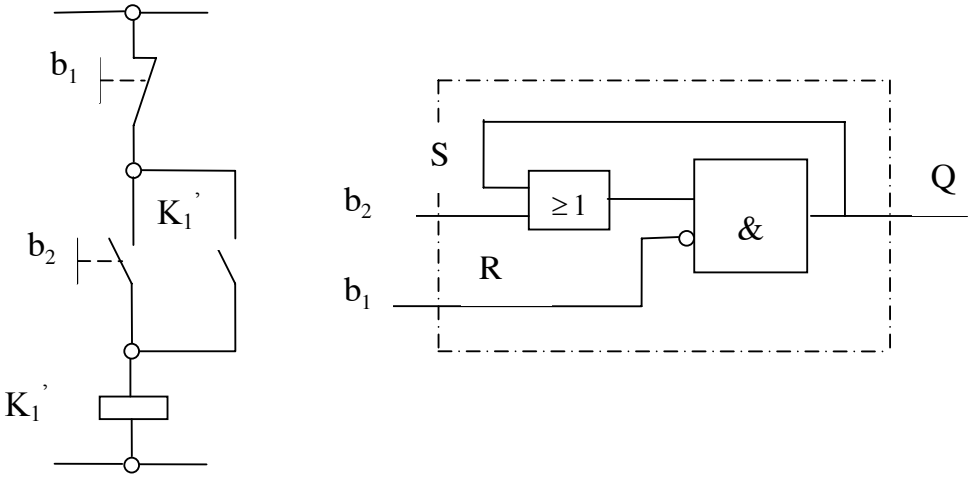
h/ Phần tử nhớ

Các phần tử trình bày ở trên có đặc điểm là tín hiệu ra tức thời phụ thuộc vào tín hiệu vào, điều đó có nghĩa là tín hiệu vào mất, thì tín hiệu ra cũng mất. Trong thực tế tín hiệu thường là dạng xung(nút ấn...). Khi tín hiệu tác động vào là dạng xung, tín hiệu ra thường là tín hiệu duy trì. Như vậy cần phải có phần tử duy trì tín hiệu, trong kỹ thuật điện người ta gọi là tự duy trì. ở hình dưới khi ấn b_2 mạch đóng, dòng điện đi qua rơle K_1, tiếp điểm K_1 được đóng lại. Dòng điện trong mạch vẫn được duy trì, mặc dù nút ấn b_2 nhả ra rồi. Dòng điện trong mạch duy trì cho đến khi nào ta ấn vào nút b_1. Thời gian tự duy trì

của dòng điện trong mạch là khả năng nhớ của mạch điện. Trong kỹ thuật điều khiển gọi là phần tử nhớ Flipfop.

Phần tử Flipfop có hai cổng vào, cổng thứ nhất ký hiệu là S (SET) và cổng thứ hai ký hiệu là R(RESET), như vậy phần tử Flipfop cũng được gọi là phần tử RS-Flipfop.

1) Phần tử RS-Flipfop có RESET trội hơn:

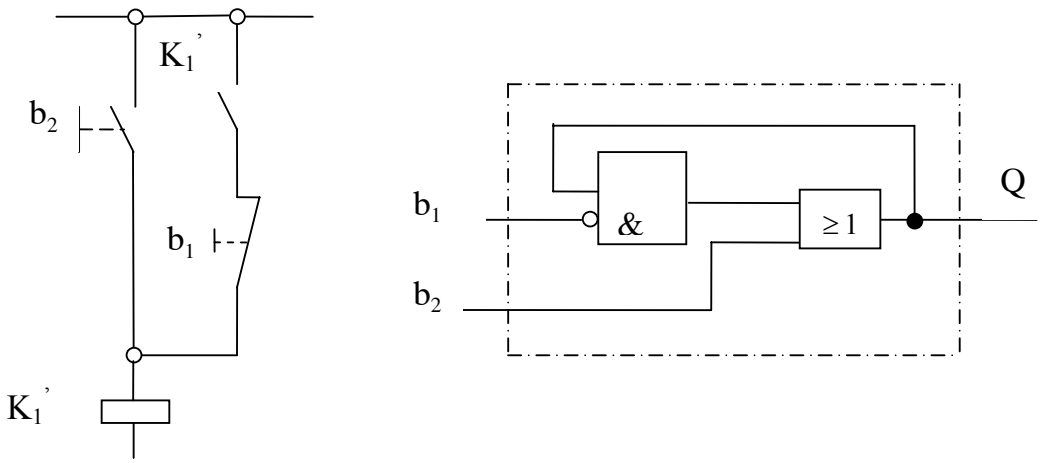


Hình 2-14: Phần tử Flipfop có RESET trội hơn

Nếu cổng SET(b_2) có giá trị L, thì tín hiệu ra Q có giá trị L và được nhớ (mặc dù ngay đó tín hiệu ở cổng SET mất đi) cho đến khi cổng RESET(b_1) có giá trị L thì phần tử Flipfop sẽ quay trở lại vị trí ban đầu. Khi cổng SET va cổng RESET cũng có giá trị L thì cổng ra Q có giá trị "0".

2.3.2 phần tử RS-Flipfop có SET trội hơn:

Nếu cổng SET (b_2) có giá trị L, thì tín hiệu ra Q có giá trị L và được nhớ (mặc dù ngay sau đó tín hiệu ở cổng SET mất đi) cho đến khi cổng RESET (b_1) có giá trị L thì phần tử Flipfop sẽ quay trở lại vị trí ban đầu. Khi cổng SET và cổng RESET cùng có giá trị L thì cổng ra Q có giá trị "1"



Hình 2-15: phần tử Flipfop có SET trội hơn

- *Lý thuyết đại số Boole:*

Một hàm y= f(x_1,x_2,...,x_n)với các biến x_1,x_2,...,x_n chỉ nhận các giá trị 0 hoặc 1 và hàm y cũng chỉ nhận các giá trị 0 hoặc 1 thì được gọi là hàm logic.

Trong kỹ thuật điều khiển, giá trị của các tín hiệu ra được viết dưới dạng biến số đại số Boole.

a) các phép biến đổi hàm một biến:

- Phép toán liên kết AND

- Phép toán liên kết OR

- Phép toán liên kết NOT

b) Luật cơ bản của đại số Boole:

- Luật hoán vị:

A ^ B = B ^ A

A v B = B ^ A

- Luật kết hợp:

(A ^ B) ^ C = A ^ (B ^ C) = B ^ (A ^ C)

(A v B) v C = A v (B v C) = B v (A v C)

- Luật phân phối:

(A ^ B) v C = A ^ (B v C)

(A v B) ^ (A v C) = A v (B ^ C)

- Luật hấp thụ:

A v (A ^B)= A

A ^ (A v B) = A

- Luật bù:

A v ($\overline{A}$ ^ B) = A v B

A ^ ($\overline{A}$ v B) = A ^ B

- Luật De Morgan:

$\overline{A \wedge B} = \overline{A} \vee \overline{B}$

$\overline{A \vee B} = \overline{A} \wedge \overline{B}$

2.4 Hệ thống điều khiển dùng các bộ PLC

2.4.1 Giới thiệu chung về các bộ PLC

PLC là cụm từ viết tắt tiêng Anh:

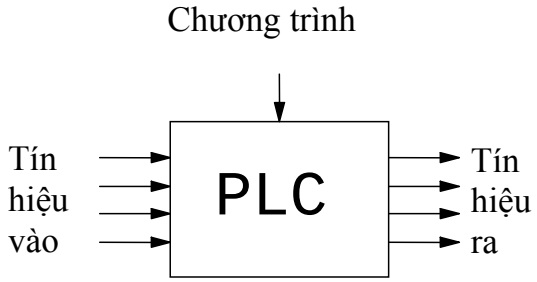
Programmable Logic Controller tức là bộ điều khiển logic có khả năng lập trình được. Bộ điều khiển này thực hiện các chức năng lôgic tương tự 1 panel trễ hay 1 hệ thống điều khiển lôgic ở trạng thái cứng.

PLC được phát triển dựa trên cơ sở vi xử lý sử dụng bộ nhớ lập trình được để lưu trữ và thực hiện các chức năng như phép tính logic, đếm, và các thuật toán để điều khiển máy và các quá trình.

PLC được thiết kế cho phép người sử dụng không cần kiến thức chuyên sâu về máy tính và ngôn ngữ máy tính củng có thể vận hành.

PLC đã được các nhà thiết kế lập trình sẵn sao cho chương trình điều khiển có thể nhập bằng cách sử dụng ngôn ngữ đơn giản. ở đây thuật ngữ

logic được sử dụng vì hầu hết việc lập trình chủ yếu liên quan đến các hoạt động logic thực thi và chuyển mạch. Các thiết bị nhập có thể là công tắc tơ, cảm biến, bàn phím vv... Các thiết bị xuất trong hệ thống được điều khiển như động cơ, các van vv...được nối kết với PLC. Khi ta nhập chương trình vào bộ nhớ của PLC, thiết bị điều khiển sẽ giám sát các tín hiệu vào và các tín hiệu ra theo chương trình này và thực hiện các bước điều khiển đã được lập trình.



Hình 2-16: Thiết bị điều khiển logic lập trình

Ưu điểm nổi bật của PLC là tính linh hoạt.Chúng ta có thể sử dụng cùng một thiết bị điều khiển cơ bản cho nhiều hệ thống điều khiển. Để sửa đổi hệ thống điều khiển người vận hành chỉ cần thay đổi chương trình mà không cần mắc nối lại các thiết bị của hệ thống. Chính vì ưu điểm này mà PLC được sử dụng rộng rãi trong công nghiệp.

PLC đầu tiên xuất hiện vào năm 1969. Ngày nay chúng được sử dụng phổ biến, từ các thiết bị nhỏ, độc lập sử dụng khoảng 20 đầu vào / đầu ra digital, đến các hệ thống nối ghép theo module có thể sử dụng rất nhiều đầu vào / đầu ra, xử lý các tính hiệu digital hoặc analog. Ngoài ra PLC còn có thể điều khiển tỷ lệ, tích phân, đạo hàm.

Trong PLC chương trình được thực hiện theo chu trình lặp. Mỗi vòng lặp được gọi là 1 vòng quét (Scan). Mỗi vòng quét có 4 giai đoạn, bắt đầu bằng giai đoạn:

- Đọc dữ liệu từ cổng vào (Input) tới bộ đệm.

- Thực hiện chương trình.
- Truyền thông nội bộ và kiểm tra lỗi.
- Đưa nội dung của bộ đệm tới các cổng ra (Output).

Các bộ PLC thường gặp:

- Họ Simatic S5, Simatic S7 của hãng Siemens của Cộng hoà liên bang Đức.
- Các họ Series 90 TM của hãng Fanme, Nhật Bản.
- Các họ CQM1, CPM1, CPM1A và SRM của hãng OMRON, Nhật Bản...

Các bộ PLC được ứng dụng rộng rãi trong các lĩnh vực sản xuất công nghiệp, giao thông và đời sống ... .

- Công nghệ cơ khí:
  - Gia công bao bì.
  - Dây chuyền sản xuất xi măng.
  - Công nghệ đúc áp lực... .
- Công nghệ thực phẩm:
  - Các thiết bị sản xuất nước ngọt.
  - Các thiết bị thức ăn gia súc.
- Công nghiệp nhẹ:
  - Ngành nhuộm.
  - Dệt, thêu ren.

Ở Việt Nam bộ điều khiển SIMATIC S5 của hãng SIEMENS đưa vào điều khiển các cơ cấu điều hoà công suất của các tổ máy thuỷ điện Hoà Bình. Công ty dầu khí Việt-Xô Petro dùng các bộ PLC để điều khiển các thiết bị sản xuất khí đốt và dầu nhờn.

Công ty thuốc lá Thăng Long, Công ty dệt 8-3 sử dụng bộ PLC vào dây chuyền tự động .v.v.. .

Vì vậy ứng dụng PLC vào điều khiển máy móc là điều tất yếu cần tìm hiểu nghiên cứu để đạt được hiệu quả tối ưu.

2.4.2 Các bộ phận cơ bản của hệ thống PLC.

Một hệ thống PLC thường có 5 bộ phận cơ bản, bộ xử lý trung tâm(CPU), bộ nhớ, bộ nguồn, thiết bị lập trình và các giao diện nhập/xuất.

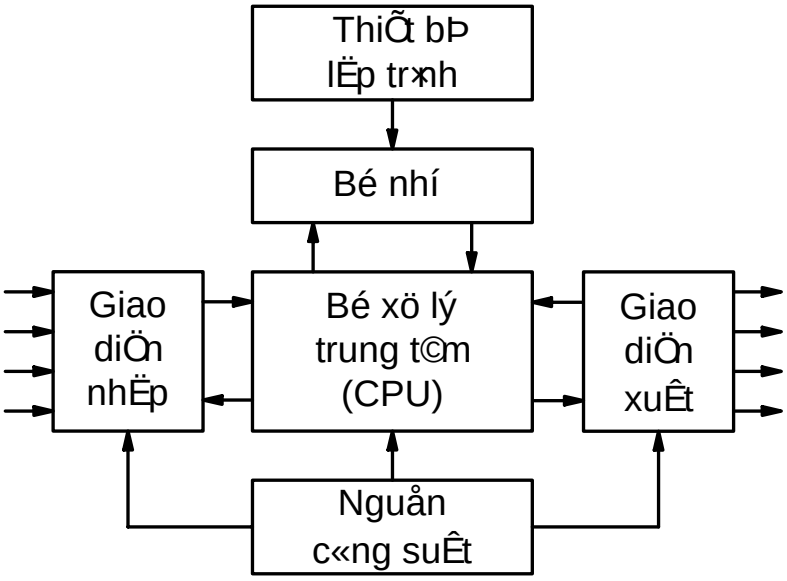
- *Bộ xử lý trung tâm:* Là thiết bị chứa bộ vi xử lý, biên dịch các tín hiệu nhập và thực hiện các hoạt động điều khiển theo chương trình đã được lưu trong bộ nhớ của CPU, đồng thời phát các tín hiệu điều khiển các thiết bị xuất. Đây là nơi xử lý mọi hoạt động của PLC, bao gồm cả viêc thực hiện chương trình.

- *Bộ nhớ:* Là nơi lưu giữ các chương trình được sử dụng cho các hoạt động điều khiển và các trạng thái nhớ trung gian trong quá trình thực hiện, bộ nhớ chịu sự kiểm soát của bộ vi xử lý.

- *Bộ nguồn:* Có nhiệm vụ chuyển đổi điện áp từ xoay chiều(AC) thành điện áp 1 chiều(DC - 5V, 24V) cần thiết cho bộ vi xử lý và các mạch điện trong thiết bị nhập và xuất.

- *Thiết bị lập trình:* Được sử dụng để lập các chương trình cần thiết. Các chương trình này được chuyển đến và lưu trên bộ nhớ của PLC.

- *Các phần tử nhập và xuất:* Là nơi mà bộ xử lý nhận các tín hiệu từ các thiết bị ngoại vi và cấp tín hiệu điều khiển đến các thiết bị bên ngoài. Các tín hiệu vào có thể là các công tắc cơ, các cảm biến nhiệt độ, cảm biến áp suất, cảm biến lưu lượng vv... Còn tín hiệu điều khiển các thiết bị ra có thể là động cơ, các van vv...



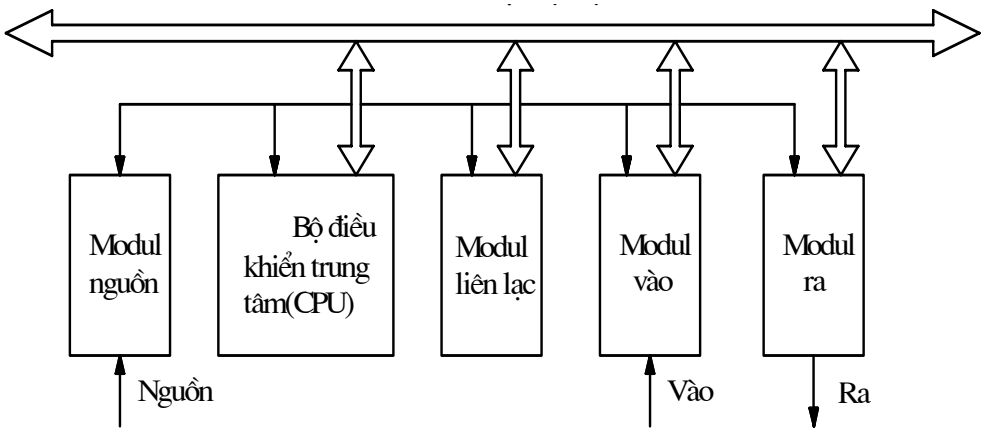
HÖthèng PLC

Hình 2-17: Hệ thống PLC

2.4.3 Cấu trúc của các bộ PLC

Nói chung các bộ PLC của các hãng Tây Đức và của Nhật đều có các cấu trúc cơ bản được trình bày tóm tắt như sau:

Cấu trúc bộ PLC còn được trình bày dưới dạng sơ đồ khối sau:



Hình 2-18: Cấu trúc bộ điều khiển PLC

Theo quan điểm này bộ PLC có các bộ phận chính sau:

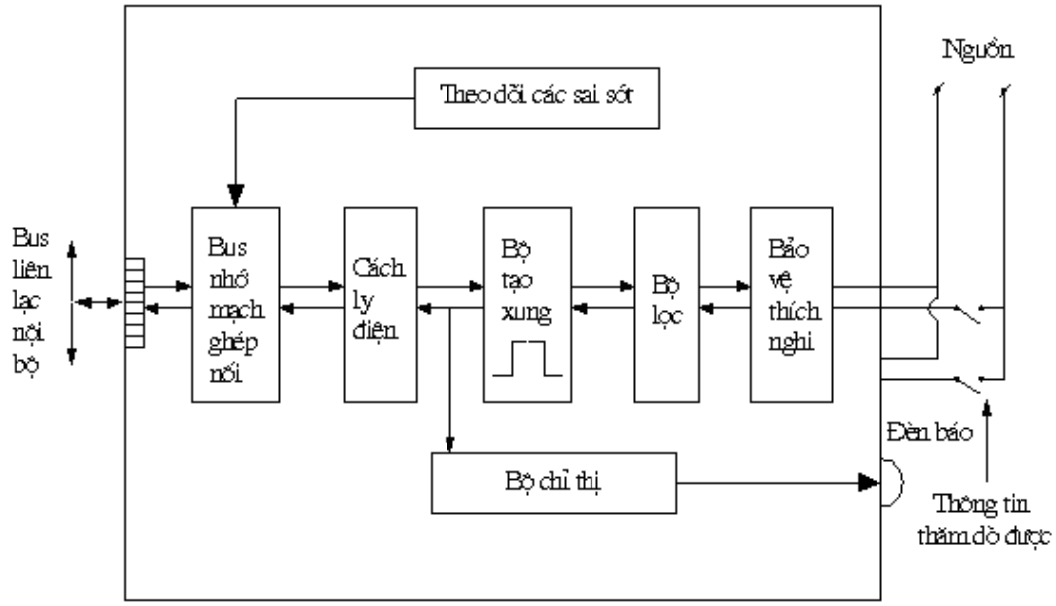
- Modun vào (Input).
- Modun ra (Output).
- Modun liên lạc.
- Bộ điều khiển trung tâm.
- Modun nguồn.

Giữa chúng có hệ thống liên lạc nội bộ.

Ta trình bày sơ lược từng bộ phận:

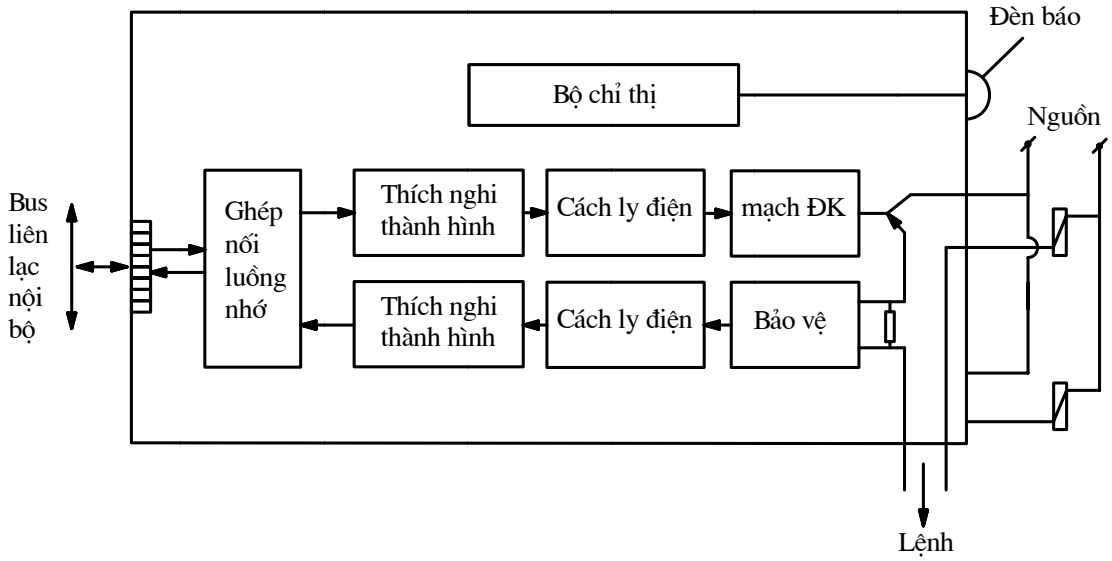
a) Modun vào.

Modun này nhận tín hiệu từ cảm biến, cho phép bộ vi xử lý đọc trạng thái logíc từ các cảm biến (Hình 2-19). Các tín hiệu nhiễu được lọc bỏ, chỉ cho các tín hiệu thực của cảm biến đi qua sau đó hình thành tín hiệu chuẩn dạng xung chữ nhật nhờ bộ tạo xung chuẩn. Mỗi xung cho 1 bù (1 đơn vị nhớ) tập hợp 16 bit = 1 từ (1 word) và tập hợp các từ thành 1 câu (Tổ hợp các số nhị phân 0 & 1). Như vậy các tín hiệu chuẩn sau khi qua bộ cách ly điện và mạch ghép phối Bus nhớ sẽ được chuyển về bộ nhớ chính (Não bộ của bộ PLC hay gọi là bộ vi xử lý trung tâm).



b, Modul ra:

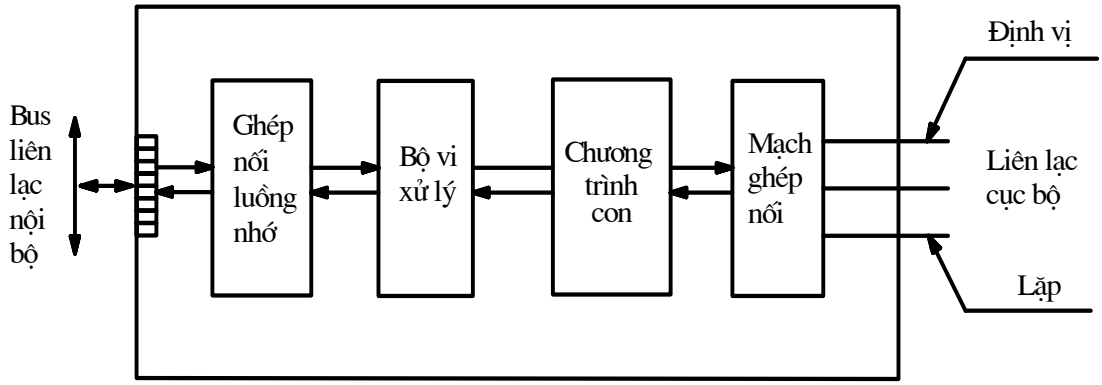
Tín hiệu từ bộ nhớ, nhờ các mạch ghép nối sẽ chuyển tới đầu ra (Output) (Hình 2-20) tiếp theo tín hiệu được chuyển tới mạch thích nghi, tín hiệu chuẩn sẽ qua bộ cách ly điện và tới mạch điều khiển . Từ đây lệnh điều khiển được phát ra.



Hình 2-20: Cấu trúc đầu ra (Output) kiểu số

c, Modun liên lạc.

Đây là bộ phận liên hệ, trao đổi thông tin giữa các bộ phận trong bộ PLC, giữa bộ phận vi xử lý và bộ ngoài vi xử lý. Trên hình 2-21 trình bày cấu trúc của modun liên lạc.



Hình 2-21: Cấu trúc modun liên lạc

d, Bộ vi xử lý (Microprocessor).

Đây là bộ phận chính hay gọi là bộ vi xử lý của PLC, tại đây nó xử lý toàn bộ mọi dữ liệu gửi tới, giải các bài toán, tạo các lệnh điều khiển, bộ nhớ chính cũng nằm ở đây.

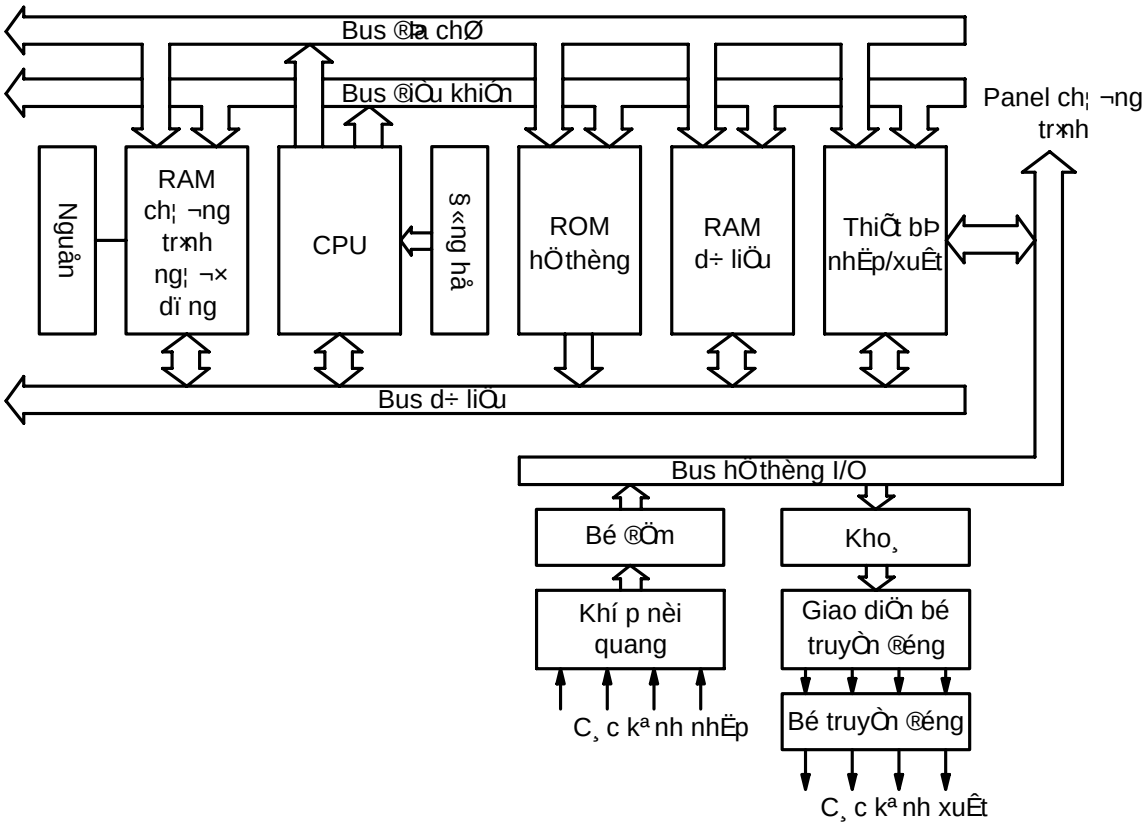
e, Mođun nguồn

Modun nguồn nhận nguồn điện công nghiệp 110V hoặc 220V. Trong PLC tại modun nguồn có bộ phận toả nhiệt, chống đoản mạch bảo vệ an toàn.

2.4.4 Cấu trúc bên trong của PLC.

Cấu trúc cơ bản bên trong của PLC bao gồm bộ xử lý trung tâm(CPU) chứa bộ vi xử lý hệ thống, bộ nhớ, và mạch nhập/xuất. CPU điều khiển và xử lý mọi hoạt động bên trong của PLC. Bộ xử lý trung tâm được trang bị đồng hồ có tần số khoảng 1 đến 8MHz, tần số này quyết định tốc độ vận hành của PLC, cung cấp chuẩn bị thời gian và đồng bộ hoá tất cả các thành phần của hệ

thống. Thông tin trong PLC được truyền dưới dạng tín hiệu digital gọi là các *bus*. Chúng có thể là các vệt dẫn trên bảng mạch in hoặc củng có thể là các dây điện trong cáp bẹ. CPU sử dụng các *bus dử liệu* để gửi dữ liệu giữa các bộ phận, *bus địa chỉ* để gửi địa chỉ các vị trí truy cập dữ liệu được lưu dữ và *bus điều khiển* dẫn các tín hiệu điều khiển nội bộ. *Bus hệ thống* được sử dụng để truyền thông giữa các cổng và thiết bị nhập/xuất.



Hình 2-22: Cấu trúc bên trong của bộ PLC

CPU

Cấu hình của CPU tuỳ thuộc vào bộ vi xử lý, CPU gồm có: *Bộ thuật toán và logic*(ALU) chịu trách nhiệm xử lý dữ liệu, thực hiện các phép toán số học và các phép toán logic.

Bộ nhớ hay còn gọi các thanh ghi bên trong bộ xử lý, được sử dụng để lưu trữ thông tin liên quan đến việc chạy chương trình.

Bộ điều khiển được sử dụng để điều khiển chuẩn thời gian của các phép toán.

Bus

Bus là các đường dẫn dùng để truyền thông bên trong PLC. Thông tin được truyền theo dạng nhị phân, theo nhóm bit, mỗi bit là một số nhị phân 0 hoặc 1 tương ứng với trạng thái on/off. Thuật ngữ từ được sử dụng cho nhóm bít tạo thành thông tin nào đó. Vì vậy mỗi từ 8 bit này có thể là số nhị phân(00100110), cả 8 bít này được truyền đồng thời theo dây song song của chúng.

Hệ thống PLC gồm có bốn bus sau:

- *Bus dữ liệu(Data Bus)* tải dữ liệu được sử dụng trong quá trình xử lý của CPU. Nó là đường truyền qua lại giữa bộ nhớ và bộ xử lý. Bộ xử lý 8 bít có một bus dữ liệu nội có thể thao tác các số 8 bít, có thể thực hiện các phép toán giữa các số 8 bít và phân phối kết quả theo giá trị 8 bit.

- *Bus địa chỉ(Address Bus)* được sử dụng để tải địa chỉ các vị trí trong bộ nhớ. Như vậy mỗi từ có thể được định vị trong bộ nhớ, mỗi vị trí nhớ được gán một địa chỉ duy nhất. Mỗi vị trí được gán một địa chỉ sao cho dữ liệu được lưu trữ ở vị trí nhất định, để CPU có thể đọc hoặc gi ở đó. Bus địa chỉ mang thông tin cho biết địa chỉ sẽ được truy cập. Nếu bus địa có 8 đường truyền thì số lượng địa chỉ sẽ là $2^8 = 256$ địa chỉ. Còn nếu bus có 16 đường truyền thì số lượng địa chỉ là $2^{16} = 65536$ địa chỉ.

- *Bus điều khiển(Control Bus)* dùng để truyền các tín hiệu của bộ điều khiển, tín hiệu được CPU sử dụng để điều khiển các thiết bị nhớ nhận dữ liệu từ thiết bị nhập/xuất và tải các tín hiệu chuẩn thời gian được dùng để đồng bộ hoá các hoạt động.

- *Bus hệ thống(System Bus)* được dùng để truyền thông giữa các cổng nhập/xuất và các thiết bị nhập/xuất.

Bộ nhớ

Trong hệ thống PLC có rất nhiều bộ nhớ như : ROM, RAM, EFROM ...

- ROM (*Bộ nhớ chỉ đọc*) cung cấp dung lượng nhớ cho hệ điều hành và dữ liệu cố định được CPU sử dụng. ROM không bị mất dữ liệu khi mất điện.

- RAM (*Bộ nhớ truy cập ngẫu nhiên*) dành cho chương trình của người dùng và đồng thời là nơi lưu trữ thông tin theo trạng thái của các thiết bị nhập, xuất, các giá trị của đồng hồ định giờ, các bộ đếm và các thiết bị nội vi khác. RAM dữ liêu đôi khi còn được coi là *bảng dữ liệu hay bảng ghi.* Một phần của bộ nhớ này dành cho các địa chỉ của ngỏ vào và ngỏ ra cùng với trạng thái của ngỏ vào và ngỏ ra đó. Một phần dành cho dữ liệu được cài đặt trước, và một phần khác dành để lưu trữ các giá trị của bộ đếm, đồng hồ định giờ vv...Đây là bộ nhớ sơ cấp, trong đó các chỉ lệnh chương trình và dữ liệu được lưu trữ sao cho bộ xử lý trung tâm(CPU) có thể truy cập trực tiếp vào chúng thông qua bus dữ liệu cao tốc của bộ xử lý đó. CPU có thể đọc và ghi dữ liệu từ RAM. Khi mất điện các nội dung trên RAM sẽ bị mất.

- EPROM (*Bộ nhớ chỉ đọc có thể xoá và lập trình được*) đây là bộ nhớ ROM có thể được lập trình và chương trình được lập này được thường trú trong ROM.

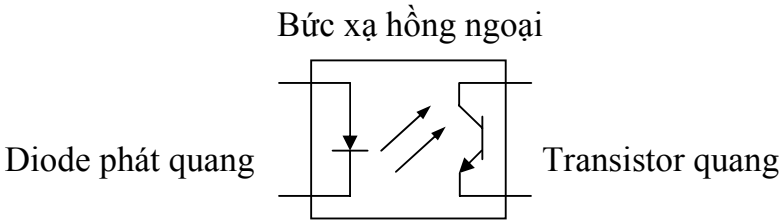
Các PLC đều có một lượng RAM để lưu trữ chương trình do người dùng cài đặt và dữ liệu chượng trình. Tuy nhiên để tránh mất chương trình khi bị mất điện, PLC sữ dụng ắc quy để duy trì nội dung RAM trong một thời gian. Sau khi được cài đặt vào RAM, chương trình có thể được tải vào bộ nhớ EPROM, thường là module có khoá đối với PLC , do đó chương trình trở thành vĩnh cửu. Ngoài ra PLC còn có các bộ đệm tạm thời, lưu trữ các kênh nhập/xuất.

Dung lượng lưu trữ của bộ nhớ được xác định bằng số lượng từ nhị phân có thể lưu trữ được. Nếu dung lượng bộ nhớ là 256 từ, thì bộ nhớ có thể lưu trữ được 256 × 8 = 2048bit nếu sử dụng từ 8bít, và 256 × 16 = 4096bít nếu sử dụng từ 16bít.

Các loại PLC khác nhau có dung lượng khác nhau, có thể từ 1K ÷ 64K.

Bộ cách ly quang điện

Thiết bị nhập/xuất là giao diện giữ hệ thống và thế giới bên ngoài, cho phép thực hiện các nối kết thông qua các kênh nhập/xuất đến thiết bị nhập và thiết bị xuất. Củng từ các thiết bị này chương trình được đưa vào hệ thống từ bảng chương trình. Mỗi điểm nhập/xuất có một địa chỉ duy nhất mà CPU sử dụng.



Hình 2-23: Thiết bị cách điện quang học.

Khi xung digital đi qua diode phát quang, sẽ tạo ra xung hồng ngoại. Xung này được transistor qung học tiếp nhận và làm tăng điện áp trong mạch. Khe hở giữa transistor và diode phát quang sẽ tạo ra sự cách điện nhưng vẫn cho phép xung digital đi vào mạch để làm tăng xung digital trong mạch khác. Tín hiệu nhập có thể sử dụng trong PLC là các tín hiệu on-off 5V, 24V, 110V, 220V.

Thiết bị nhập/xuất.

Thiết bị nhập xuất của PLC được thiết kế sao cho dãi tín hiệu vào có thể biến đổi được thành tín hiệu digital 5V, và tín hiệu ra là khả dụng để tác dụng lên các thiết bị khả dụng. Khả năng này cho phép xử lý dãi tín hiệu vào và tín hiệu ra để các PLC dễ dạng sử dụng. Nói chung các tín hiệu vào từ modul nhập được chọn lựa bằng các công tắc DIP(Dual Inline Package) và được bố trí phía sau các modul.

Chúng chỉ có hai trạng thái on-off và được sử dụng để cài đặt các tham số cho modul, đồng thời củng được dùng để xác lập địa chỉ của các modul.

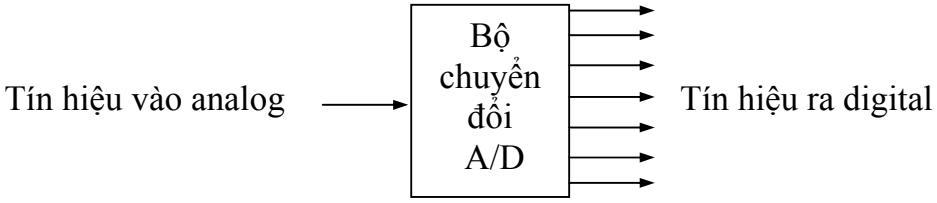
Tín hiệu nhập từ các sensor và tín hiệu xuất đến các thiết bị điều khiển có thể là :

- Tín hiệu rời rạc: Thực chất đây chỉ là các tín hiệu on-off. Nó được nhập từ các thiết bị như công tắc cơ, công tắc gián tiếp, các bộ cảm biến quang điện vv...

- Tín hiệu analog: Là tín hiệu có kích cỡ liên quan đến đại lượng đang được cảm biến. Các tín hiệu này được nhập từ các cảm biến nhiệt độ, cảm biến áp suất, cảm biến khoảng cách dịch chuyển vv...

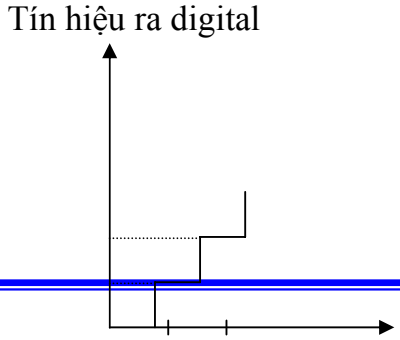
- Tín hiệu digital: Đó là các chuỗi xung cấp vào cho PLC.

Các tín hiệu digital được nhập vào PLC nếu kênh nhập của PLC có khả năng chuyển tín hiệu đó thành tín hiệu digital, qua bộ chuyển đổi A/D(Analog-Digital Change).



Hình 2-24: Bộ chuyển đổi tín hiệu A/D.

Một tín hiệu vào analog tạo thành các tín hiệu on-off trên 8 dây riêng rẽ. 8 tín hiệu này tạo thành từ dưới dạng digital tương ứng với mức tín hiệu analog. Như vậy bộ chuyển đổi 8 bit có thể có $2^8 = 256$ giá trị khác nhau, từ 00000000 ÷ 11111111, nghĩa là từ 0 ÷ 255.



00000010

00000001

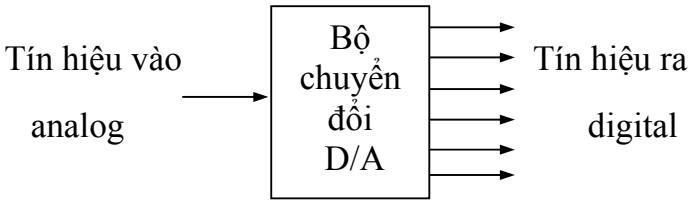
00000000 Tín hiệu vào analog

1 2

Hình 2-25: Các bậc tín hiệu.

Các thiết bị xuất có thể là contactor, các rơle, transistor, triac vv...

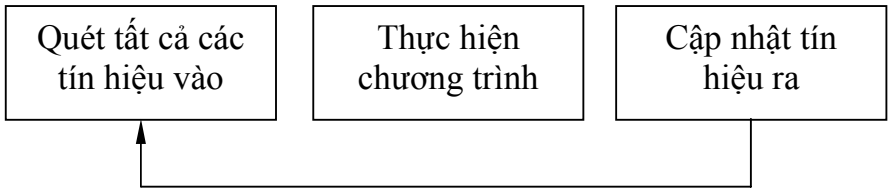
Khi các đầu ra cần các tín hiệu analog ta có thể sử dụng bộ chuyển đổi D/A(Digital-Analog Change). Khi đó tín hiệu vào bộ D/A là các chuỗi bít trên một đướng song song, qua bộ D/A và cho tín hiệu ra analog.



Hình 2-26: Bộ chuyển đổi tín hiệu D/A.

Xử lý các tín hiệu vào-ra(I/O).

PLC chạy liên tục thông qua chương trình và cập nhật kết quả từ các tín hiệu vào. Mỗi vòng làm việc như vậy được gọi là một vòng quét của PLC.



Hình 2-27. Hoạt động của PLC.

a. Cập nhật liên tục.

Trong phương pháp này CPU quét các kênh nhập khi chúng xuất hiện theo các kênh của chương trình. Mỗi điểm nhập sẽ được kiểm tra riêng rẽ và tác động của chúng lên chương trình sẽ được xác định. Sự yêu cầu tầng điểm nhập theo mỗi lệnh chương trình sẽ rất tốn thời gian. Nhiều kênh nhập có thể được quét trước khi chương trình có chỉ thị để thực thi hoạt động cụ thể và tín hiệu ra xuất hiện. Các tín hiệu ra duy trì trạng thái của chúng, bị khoá cho đến khi có sự cập nhật tiếp theo. Chuỗi làm việc như sau.

- Tìm nạp và giải mã lệnh chương trình thứ nhất.

- Quét các ngõ vào tương ứng.

- Tìm nạp và giải mã lệnh chương trình thứ hai.

- Quét các ngõ vào tương ứng, với các chương trình còn lại.

- Cập nhật các ngõ ra.

- Lặp lại toàn bộ chuỗi trên.

b. Sao chép khối tín hiệu nhập/xuất.

Do phương pháp cập nhật liên tục cần có thời gian kiểm tra lần lượt từng điểm nhập, vì vậy khi số điểm nhập lớn thời gian kiểm tra sẽ dài và chương trình chạy sẽ chậm. Để thực hiện chương trình trình nhanh hơn, một vùng đặc biệt của RAM được sử dụng làm bộ nhớ đệm giữa logic điều khiển và bộ nhập/xuất. Mỗi bộ nhập xuất đều có địa chỉ trong vùng nhớ này. Khi một chu kỳ được thực hiện, CPU quét tất cả các tín hiệu nhập và sao chép trạng thái của chúng vào các địa chỉ của bộ nhập/xuất trong RAM.

Khi chương trình được thực hiện, CPU đọc dữ liệu được lưu trữ trong RAM, theo yêu cầu và thực hiện các phép toán logic. Tín hiệu xuất được lưu trữ trong vung RAM dành riêng cho nhập/xuất. Sau mỗi chu kỳ tín hiệu xuất trong RAM đều được chuyển đến các kênh xuất tương ứng. Các tín hiệu xuất vẫn duy trì trạng thái của chúng cho đến khi chu kỳ nhập kế tiếp được khởi động. Chuỗi làm việc như sau.

- Quét tất cả các đơn vị nhập và lưu trữ vào RAM.

- Tìm kiếm, giải mã và thực thi tất cả các chỉ thị của chương trình theo thứ tự và sao chép các chỉ thị xuất vào RAM.

- Cập nhật tất cả các chỉ thị xuất.

- Lặp lại chuỗi trên.

Như vây thời gian cần thiết để hoàn tất một chu trình quét các ngõ vào và cập nhật các ngõ ra theo các lệnh chương trình sẽ ngắn đi và chương trình được thực hiện một cách nhanh chóng.

Tuy vậy các ngõ vào vẫn không được theo dõi một cách liên tục, các mẫu trạng thái của chúng đựoc lấy một cách định kỳ. Thời gian của chu kỳ thường khoảng từ 10 ÷ 50ms, nghĩa là các ngõ vào và các ngõ ra được cập nhật sau 10 ÷ 50ms. Và như vậy sự đáp ứng của hệ thông có thể bị trể. Điều này củng có nghĩa là nếu chu kỳ nhập rất xẩy ra ở thời điểm không thích hợp thì chu kỳ có thể bị bỏ sót. Do vậy một chu kỳ nhập bất kỳ phải được thực hiện trong khoảng thời gian lớn hơn thời gian của một vong quét của PLC.

Để tránh trường hợp bị bỏ sót chu kỳ người ta sử dụng các modul đặc biệt có tác dụng khắc phục các thiếu sót trên.

Giao tiếp qua cổng Serial Port (Port COM):

IBM PC cung cấp 2 cổng nối tiếp: COM1 và COM2. Các cổng này giao tiếp theo tiêu chuẩn RS232. Chúng có thể được nối với một Modem để dùng cho mạng điện thoại, hay nối trực tiếp với một máy tính khác. Dữ liệu được truyền qua cổng này theo cách nối tiếp, nghĩa là dữ liệu được truyền đi nối tiếp nhau trên một đường dây. Do các dữ liệu được truyền đi từng bít một nên tốc độ truyền chậm, các tốc độ truyền là 300, 600, 1200, 2400, 4800, 9600 bps, chiều dài dữ liệu có thể là 5,6,7 hoặc 8 bít kết hợp với các bít Start, Stop, Parity tạo thành một khung (frame). Ngoài ra cổng này còn có các điều khiển thu (Receive), phát (Trans), kiểm tra. Cách giao tiếp này cho phép khoảng cách truyền dữ liệu xa, tuy nhiên tốc độ truyền rất chậm, tốc độ tối đa là 20 kps.

2.4.5 Ngôn ngữ lập trình cho PLC

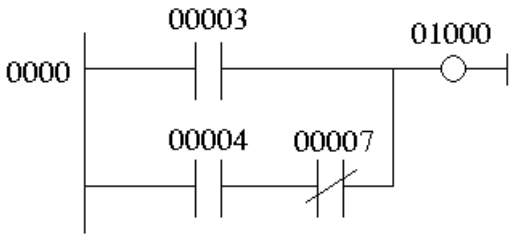
Ngôn ngữ lập trình cho của PLC có thể được dùng trong 3 dạng. Đây là cách nói của hầu hết các chuyên gia. PLC được lập trình theo biểu đồ lôgíc hoặc danh sách lệnh hoặc biểu đồ hình thang. Nhưng chính xác hơn được gọi là ngôn ngữ dẫn Boolean gồm 4 thuật ngữ chính là: LOAD (YES), AND, OR, NOT. Ngoài ra để bổ xung cho nội dung đầu vào và đầu ra người ta đưa vào TIM, CNT, FUN11, và HR.

Ta trình bày sơ lược 3 cách trên (dùng ngôn ngữ SYSWIN của hãng OMRON để minh hoạ).

a. Cách liệt kê (Stament lish) – (Danh sách lệnh).

Địa chỉ	Ngôn ngữ lôgíc		Nội dung
0000	LD		00003
0001	LD		00004
0002	AND	NOT	00007
0003	OR	LD	0004
0004	OUT		01000

b. Sơ đồ hình thang(Ladder Diagram).



Việc kết hợp giữa 2 ngôn ngữ này được dùng rộng rãi trong công nghiệp vì vừa dễ theo dõi và miêu tả một cách sinh động từng bước của ngôn ngữ lập trình cho PLC. (STL kết hợp LAD)

Ví dụ:

Với sơ đồ hình thang trên ta có sơ đồ liệt kê như sau:

Địa chỉ	Ngôn ngữ lôgíc		Nội dung
0000	LD		0000
0001	OR		0004
0002	LD		0001
0003	OR	NOT	0002
0004	AND	LD	
0005	OUT		01001

c. Biểu đồ lôgíc (Đươc đề cập trong chương sau).

2.4.6 Bộ PLC OMRON SYSMAC CPM2A

a. Sơ lược về bộ PLC omron sysmac cpm1a

Bộ PLC OMRON SYSMAC CPM2A do hãng OMRON của Nhật sản xuất.

Trên bộ PLC này được đánh số các đầu vào, ra và COM, điện xoay chiều 220V và điện 1 chiều 24V.

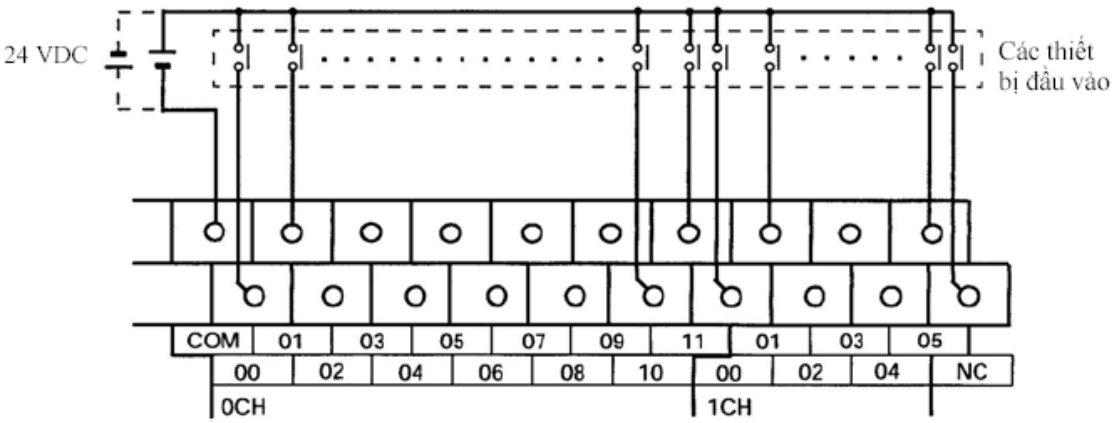
- Có 18 đầu vào được ghi các địa chỉ sau:

INPUT 0000 0001 0002 0003 0004 0005

0006 0007 0008 0009 0010 0011

Và

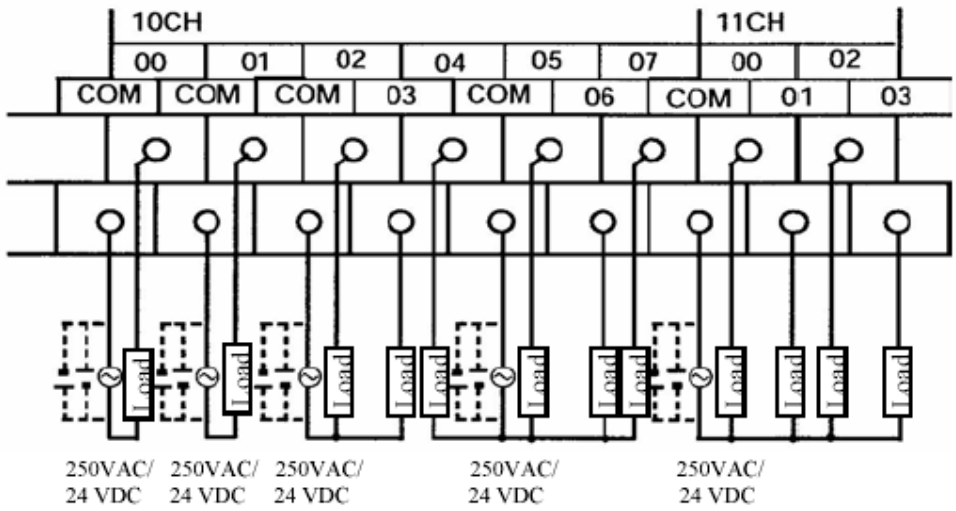
0100 0101 0102 0103 0104 0105



Hình2-29: Cấu hình đầu vào PLC OMRON SYSMAC CPM2A

- Có 12 đầu ra với các địa chỉ sau:

OUTPUT	1000	1001	1002	1003	1004	1005
	1006	1007	1100	1101	1102	1103



Hình 2-30: Cấu hình đầu ra PLC OMRON SYSMAC CPM2A

b. Thuật ngữ đầu vào và đầu ra PLC OMRON SYSMAC CPM2A PLC OMRON SYSMAC CPM2A được thiết kế nhằm tiếp nhận tín hiệu dữ

liệu đầu vào, sau khi sử lý theo chương trình sẽ phát tín hiệu qua các tín hiệu đầu ra.

Một số tín hiệu đầu vào được truy nhập vào PLC thông qua các cực được đấu bằng các dây dẫn. Điểm chính xác nơi đấu đầu vào được gọi là điểm nhập (Input print). Điểm nhập này được đặt ở vị trí cố định trên bộ nhớ của PLC phản ánh tình trạng lôgíc của tín hiệu đang được nhập từ CPU của PLC dựa trên chương trình được viết vào bộ nhớ mà người lập trình viên truy nhập vào.

Để kết thúc & khởi động chương trình, PLC điều khiển bít đầu ra (Output bit) thông qua dây dẫn điện cuối cùng tín hiệu chạm vào 1 thiết bị đầu ra (ví dụ đèn LET, điện solenoid, tín hiệu van...) và chuyển sang "ON" hoặc "OFF".

Toàn bộ hệ thống điều khiển gồm có PLC và các thiết bị đầu vào và đầu ra nối tiếp với những bộ phận năng lượng cần thiết được đặt nơi không có bụi, hơi nước, rung và không nhiễm từ.

c. Cấu trúc vùng dữ liệu.

Trên PLC OMRON SYSMAC CPM2A có nhiều vùng dữ liệu: từ IR và SR ngoài ra còn có vùng đếm và vùng thời gian (Vùng TC) do Operand giả định. Vùng dữ liệu trừ bộ phận trễ "Timers" và các bộ phận đếm "Counters" được viết bằng 1 từ (1 word = 16 bit) đánh số từ 00 ÷ 15. Nhưng từ HR 0000 ÷ HR 0009 v..v...gọi là những biến trung gian để giữ lại trong bộ nhớ hỗ trợ cho các dữ liệu đầu ra.

Khi một số bít và từ ở những vùng dữ liệu nhất định không dùng nhằm một mục đích có sẵn: chúng có thể được sử dụng trong lập trình để điều khiển các bít khác. Ví dụ cổng vào Start (khởi động hệ thống) hoặc Stop (ngắt hệ thống).

Những từ và bít sẵn sàng để sử dụng được gọi là từ làm việc hoặc bít làm việc. Hầu hết các bít chưa sử dụng có thể được dùng như bít làm việc.

Những chức năng vùng nhớ CPM2A:

1. Những bít này được bố trí đầu vào Input hoặc đầu ra Output.

2. Có thể tự ý dùng những bít làm việc trong chương trình.

3. Những bít này có chức năng điều khiển hay tạo cỡ.

4. Những bít này được duy trì trang thái tạm thời ON/OF khi mất năng lượng.

5. Được dùng để lưu các thông số điều khiển hoạt động của PLC.

2.4.7 Bộ PLC LG K7M-DR30S

a. Sơ lược về bộ PLC LG K7M-DR30S

Bộ PLC LG K7M-DR30S do hãng LG của KOREA sản xuất. Loại PLC này sử dụng đầu vào Input là tín hiệu điện 1 chiều 1÷24V, đầu ra Rơ le. Loại PLC này về nguyên tắc vận hành và cấu tạo chung thì tương tự như bộ PLC OMRON SYSMAC CPM2A.

Hình 2-31: Hình dạng PLC LG K7M-DR30S

Trên bộ PLC này được đánh số các đầu vào, ra và COM, điện xoay chiều 220V và điện 1 chiều 24V.

- Có 18 đầu vào được ghi các địa chỉ sau:

INPUT P00 P01 P02 P03 P04 P05

P06 P07 P08 P09 P0A P0B

P0C P0D P0E P0F P10 P11

- Có 12 đầu ra với các địa chỉ sau:

OUTPUT P40 P41 P42 P43 P44 P45

P46 P47 P48 P49 P4A P4B

b. Thuật ngữ đầu vào và đầu ra PLC LG K7M-DR30S

Tương tự như PLC PLC OMRON SYSMAC CPM2A.

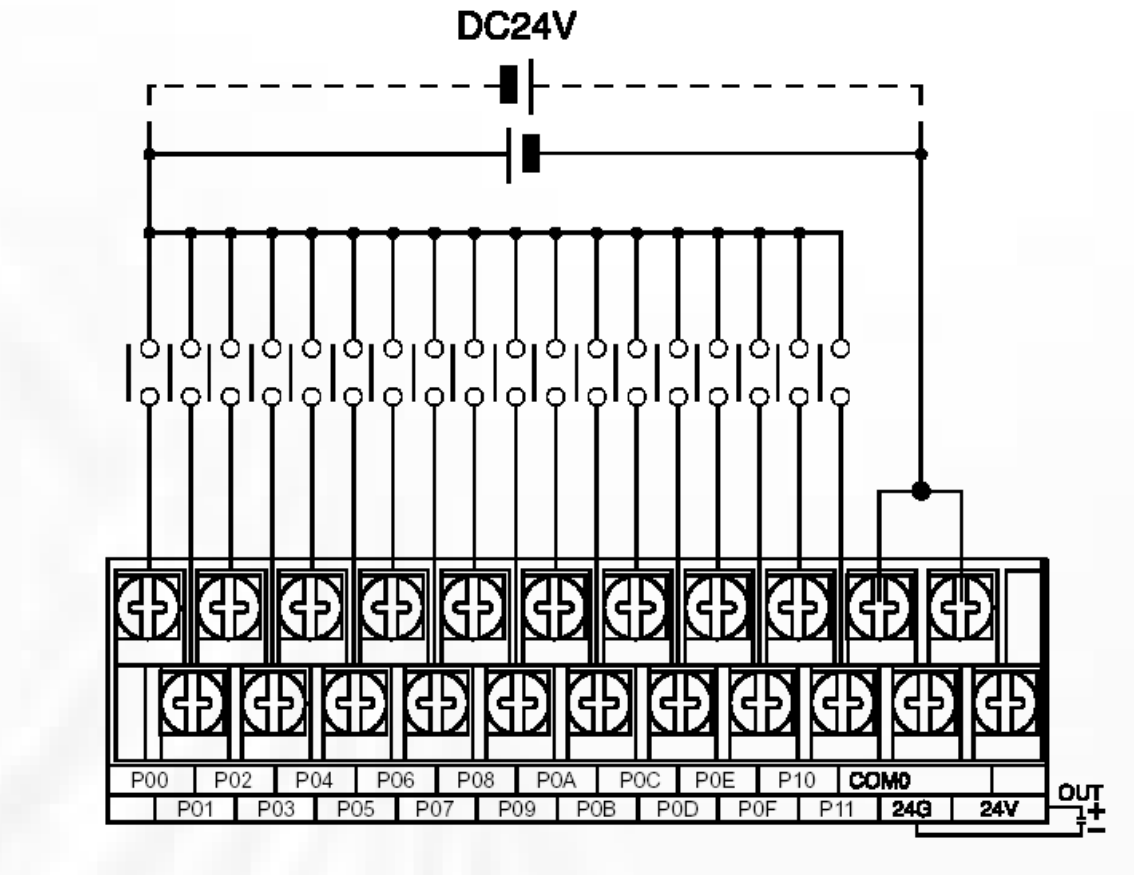
c. Cấu trúc vùng dữ liệu PLC LG K7M-DR30S

PLC LG K7M-DR30S là một hệ thống điều khiển tự động theo chương trình hoạt động theo chu kỳ với chương trình được lưu giữ trong bộ nhớ.

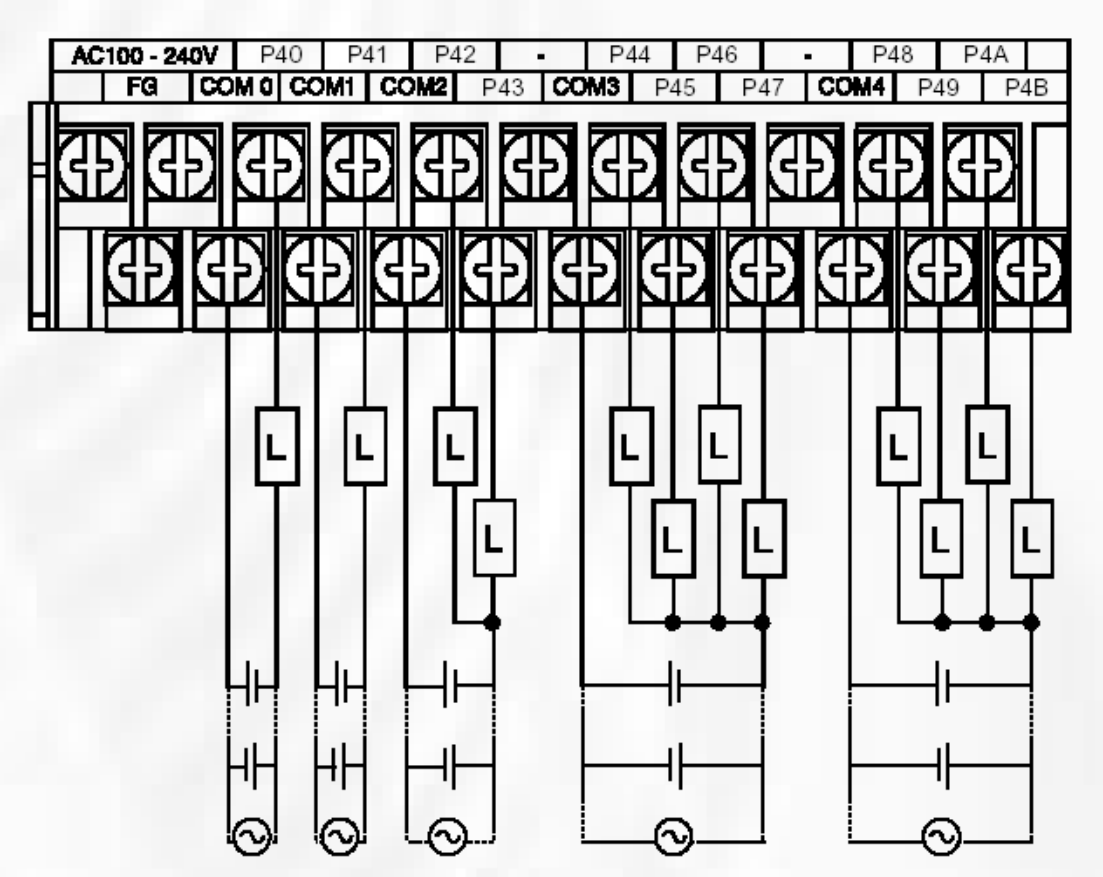
- Phương pháp điều khiển vào ra: phương pháp nạp mới, phương pháp lôgíc.
- Ngôn ngữ lập trình: Hình thang, lôgíc.
- Tốc độ vi xử lý: 0.5µs/ step.
- Dung lượng chương trình: 7kstep.
- Tiếp điểm I/O: 30.
- Tiếp điểm Input: P0000 ÷ P0011.
- Tiếp điểm Output: P0040 ÷ P004B.
- Tiếp điểm phụ: M0000÷M191F.

- Tiếp điểm liên tục: K0000÷K031F.
- Tiếp điểm liên kết: L0000÷L063F.
- Tiếp điểm đặc biệt: F0000÷F063F.
- Timer: 100ms: T000÷T191.
- Counter: C000÷C255.
- Địa chỉ điều khiển bước: S00.00÷S99.99.
- Thanh ghi dữ liệu: D0000÷D4999

Cấu tạo cơ bản của PLC LG K7M-DR30S được giới thiệu như hình sau:



Hình 2-32: Cấu trúc đầu vào PLC LG K7M-DR30S



Hình 2-33: Cấu trúc đầu ra PLC LG K7M-DR30S

CHƯƠNG 3

KẾT CẤU CỦA HỆ THỐNG ĐIỀU KHIỂN TRONG RÔBỐT CẤP PHÔI TỰ ĐỘNG

3.1 Động cơ điện và điều khiển động cơ.

3.1.1 Ứng dụng truyền động điện

Truyền động điện được sử dụng khá nhiều trong kỹ thuật robot. Vì có những ưu điểm như là điều khiển đơn giản không phải dùng các bộ biến đổi phụ thêm, không gây bẩn cho môi trường, các loại động cơ hiện đại có thể lắp trực tiếp trên các khớp quay...

Tuy nhiên so với truyền động thuỷ khí thì truyền động điện có tỷ lệ thấp giữa công suất truyền trên một một đơn vị khối lượng và thông thường đòi hỏi kèm theo hộp giảm tốc cồng kềnh vì trong tay máy tốc độ quay rất chậm...

Trong kỹ thuật robot về nguyên tắc có thể dùng động cơ điện các loại khác nhau, nhưng trong thực tế chỉ có hai loại được dùng nhiều hơn cả. Đó là động cơ điện một chiều và động cơ bước.

Ngày nay do những thành công mới trong nghiên cứu điều khiển động cơ xoay chiều, nên cũng có xu hướng chuyển sang chuyển sử dụng động cơ xoay chiều để tránh phải trang bị thêm bộ nguồn điện một chiều. Ngoài ra loại động cơ một chiều không chổi góp (DC brushless motor) cũng bắt đầu được ứng dụng nhiều.

3.1.2 Động cơ điện một chiều

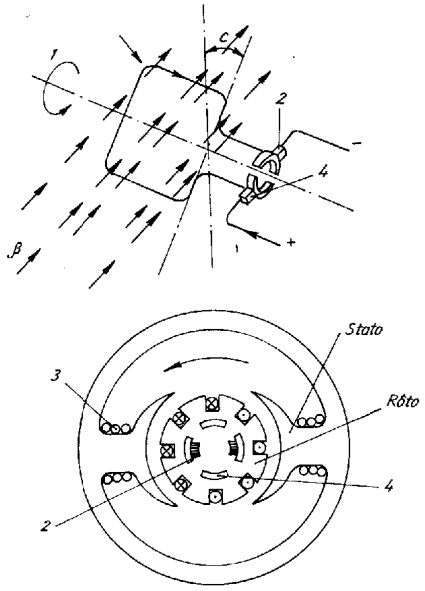
Tổng quan về động cơ điện một chiều

Động cơ điện một chiều gồm có hai phần (hình 4-2)

a. Stato cố định với các cuộn dây có dòng điện cảm hoặc dùng nam châm vĩnh cửu. Phần này gọi là phần cảm. Phần cảm tạo nên từ thông trong khe hở không khí.

b. Roto với các thanh dẫn. Khi có dòng điền một chiều chạy qua và với dòng từ thông xác định, roto sẽ quay. Phần này còn gọi là phần ứng.

Do cách khác nhau khi bố trí dây cuốn phần cảm so với phần ứng ta có những loại động cơ điện một chiều khác nhau:



u

- Động cơ kích từ song song.
- Động cơ kích từ nối tiếp.
- Động cơ kích từ hỗn hợp.

Các đại lượng chủ yếu xác định sự làm việc của động cơ một chiều là:

U - Điện áp cung cấp của phần ứng.

I - Cương độ dòng điện trong phần ứng.

r - Điện trở trong phần ứng.

Φ - Từ thông trong khe hở.

E - Sức phản điện động phần ứng.

Các quan hệ cơ bản khi làm việc là:

$E = U - rI = kn\Phi$ (1)

k phụ thuộc vào đặc tính của dây cuốn và số thanh dẫn tác dụng của phần ứng.

Từ (1) ta có các nhận xét sau:

1) Khởi động E băng 0 khi mở máy, chỉ có điện trở phần ứng r rất nhỏ hạn chế dòng điện. Vì thế cần phản cần phải có biến trở mở máy để duy trì I ở giá trị thích hợp.

2) Số vòng quay: $n = \frac{U - Ir}{k\phi}$ Vậy điều chỉnh tốc độ có thể tiến hành bằng cách tác động vào điện áp U hoặc tác động vào từ thông ϕ.

3) Momen động C xác định từ phương trình cân bằng công suất:

$$EI = 2\pi nC$$

Điều chỉnh tốc độ động cơ điện một chiều:

Về phương diện điều chỉnh tốc độ thì động cơ điên một chiều có nhiều ưu việt hơn hẳn các động cơ khác. Khả năng điều chỉnh tốc độ dễ dàng trong dải rộng và có cấu trúc mạch lực và mạch điều khiển đơn giản.

Như đã nói trên, có hai phương pháp cơ bản để điều chỉnh tốc độ động cơ điện một chiều:

- Tác động lên từ thông ϕ thông qua việc điều chỉnh điện áp dòng kích từ.

- Điều chỉnh điện áp phần ứng.

Khi điều chỉnh tốc độ 0 đến tốc độ định mức bằng cách dữ từ thông không đổi và tác động vào điện áp phần ứng U thì momen sẽ không đổi, còn công suất tăng theo tốc độ.

Khi điều chỉnh tốc độ từ 0 đến tốc độ định mức bằng cách tác động lên từ thông và giữ điện áp phần ứng không đổi thì công suất không đổi, còn momen giảm theo tốc độ.

Khi từ thông tiến về không thì tốc độ tiến tới vô cùng. Vì vậy khi không tải , động cơ kích từ nối tiếp có có tốc độ quá lớn, các loại động cơ kích từ song song hoặc hỗn hợp để quá tốc độ nếu cắt mạch kích từ của nó.

Đảo chiều quay:

Chiều quay của phần ứng phụ thuộc vào chiều dòng điện trong dây cuốn phần ứng và chiều của từ trường. Để đổi chiều quay của động cơ điện một chiều cần đổi hoặc chiều của từ thông hoặc dòng điện phần ứng.

3.1.3 Động cơ bước

3.1.3.1 Đặc điểm:

Ngày nay cùng với sự phát triển của công nghệ chế tạo vi mạch và các thiết bị điều khiển, động cơ bướcđược sử dụng rộng rãi trong nhiều lĩnh vực. Trong hệ thống điều khiển dùng động cơ bước có ưu điểm hơn hẳn động cơ một chiều và xoay chiều bởi hệ không cần các mạch vòng dòng điện và mạch vòng tốc độ mà chỉ cần mạch vòng điều khiển vị trí.

Động cơ bước là một loại động cơ điện chuyển các tín hiệu vào điều khiển(số) thành các chuyển động cơ học(gián đoạn theo bước). Tuỳ thuộc vào yêu cầu truyền động mà ta chọn động cơ bước có chế độ kích thích xung và tần số xung kích thích cho phù hợp.

*Ưu nhược điểm:

Việc sử dụng chúng trong hệ thống điều khiển có nhiều thuận lợi:

- Không cần mạch phản hồi cho cả điều khiển vị trí và vận tốc.

- Thích hợp với các thiết bị điều khiển số.

Ưu điểm lớn nhất của động cơ bước trong điều khiển vị trí là không cần phản hồi (khi điều khiển chính xác số bước quay của động cơ, đếm số bước có thể xác định vị trí chính xác mà không cần đến phản hồi vị trí) và điều khiển số trực tiếp (ghép nối trực tiếp với máy tính). Với khả năng điều khiển số trực tiếp, động cơ bước trở thành rất thông dụng trong các thiết bị hiện đại

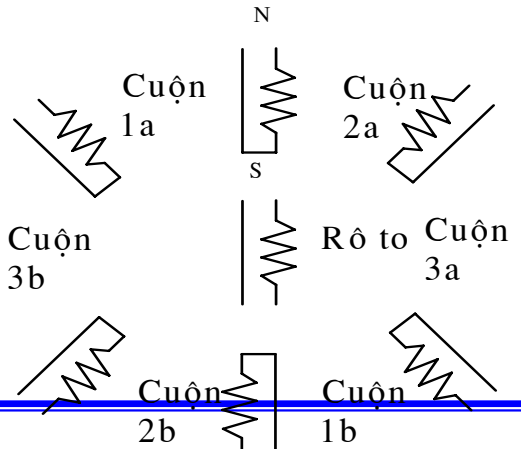
như robot công nghiệp, máy công cụ điều khiển số, các thiết bị ngoại vi của máy tính như trong máy in kim, bộ điều khiển bộ đĩa máy vi tính, máy vẽ...

Tuy vậy, phạm vi ứng dụng động cơ bước ở vùng công suất nhỏ và trung bình. Việc nghiên cứu nâng cao công suất của động cơ bước đang là vấn đề rất được quan tâm hiện nay. Ngoài ra nói chung hiệu suất của nó thấp hơn so với nhiều loại động cơ khác.

3.1.3.2 Cấu tạo.

Phần ứng của động cơ bước là cuộn dây được cuốn xung quanh một lõi thép có dòng điện một chiều chạy qua. Một hệ thống vành trượt cấp điện cho cuộn dây roto quay, như vậy ta có thể coi roto như một nam châm điện. ở mỗi thời điểm thì chỉ có một đôi dây cuốn được nạp điện (ví dụ 1a, 1b).Chúng được quấn dây sao chotừ trường của chúng thẳng hàng nhưng ngược chiều nhau. Ro to se quay cho đến khi từ trường của roto thẳng hàng với cặp từ của stato được cấp điện. Hình vẽ (2-1) cặp từ 1a, 1b được cấp điện đây là trạng thái ổn định động cỡ sẽ giữ nguyên ở vị trí này cặp cuộn day 1 được ngắt điện đồng thời cấp điện cho cuộn dây 2 (2a, 2b) và roto quay cho đến khi từ trường của nó thẳng hàng với từ trường cuộn dây 2. Mỗi bước đơn của cuộn dây 3 pha sẽ quay được một góc 60^0. Bằng cách sử dụng nhiều số đôi cực và ststo được quấn phức tạp hơn ta sẽ tăng được số bước. Thông thường động cơ này đi được 24 bước trong một vòng quay (15^0).

Động cơ bước là động cơ quay gián đoạn từng góc độ, góc quay được xác định bởi các xung đặt vào stato



Hình 3-2 : Cấu tạo động cơ bước

3.1.3.3 Phân loại động cơ bước.

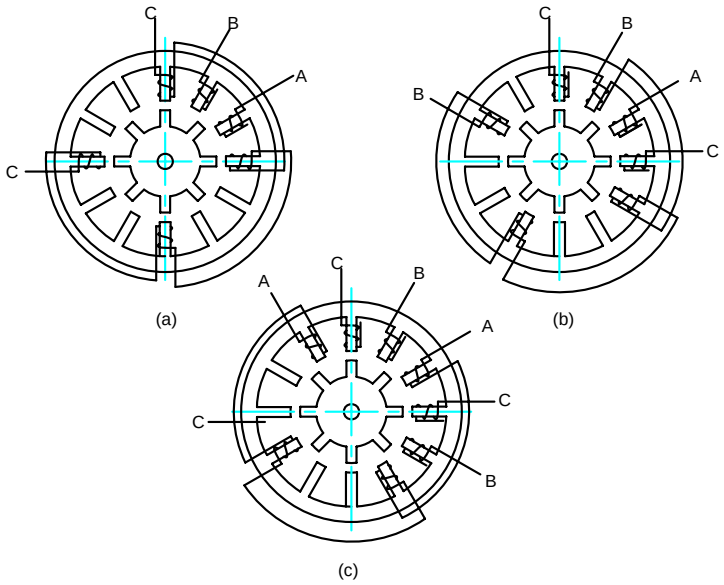
Căn cứ vào cấu tạo, động cơ bước được phân thành hai loại thông dụng là động cơ bước có từ trở biến đổi (Variable reluctance – VR) và động cơ bước nam châm vĩnh cửu (Permanent Magnet – PM), cả hai loại động cơ bước này đều hoạt động bằng cách kích thích xung điện áp chữ nhật vào cuộn dây stato, từ đó sẽ biến đổi thành chuyển dịch theo góc, ngoài ra còn loại động bước kiểu lai, nó là sự kết hợp của hai loại động cơ bước trên.

a, Động cơ bước loại VR:

Là động cơ có các cuộn dây cuốn trên các răng của của stato còn rôto làm bằng sắt non là loại sắt từ không có khả năng giữ từ trường khi ngừng kích thích. Rôto quay khi các răng của rôto bị hút bởi từ trường của các răng stato. Rô to sắt non có quán tính nhỏ hơn các loại khác. Điều này cho phép đáp ứng nhanh hơn. Tuy nhiên do rôto không có mômen dư khi ngừng kích thích , động cơ quay tự do, góc bước của động cơ VR thường là $7,5^0$ và 15^0.

Xét nguyên lý của động cơ VR:

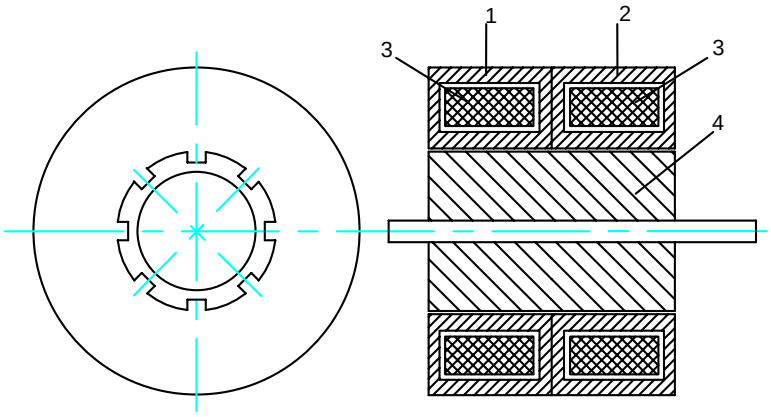
Ta kích thích cuộn dây một pha nào đó bằng nguồn một chiều khi đó lực từ trường do cuộn dây của stato sinh ra sẽ đẩy rôto tới vị trí mà tại đó tiết diện răng rôto và tiết diện răng stato đối diện vì tại vị trí này từ trở nhỏ nhất nên động cơ sẽ ở trạng thái cân bằng. Trên hình vẽ ta xét động cơ VR 3 pha có 12 răng trên stato và 8 răng trên rôto. Giả xử khi pha C được kích thích thì Rôto ở vị trí như hình a, nếu ngừng kích thích pha C và kích thích pha B thì răng rôto có vị trí gần pha B sẽ được đẩy vào vị trí đối diện thẳng hàng với răng của stato khi đó rôto sẽ chuyển động được một bước. Cứ tiếp tục kích thích xung như vậy thì rôto sẽ chuyển động theo vòng trò, đây là lý do tại sao động cơ bước loại VR hoạt động được. Tuy nhiên nếu ta kích thích pha C thay vì pha A thỉ rôto sẽ quay theo chiều kim đồng hồ trở về vị trí pha C ban đầu.



Hình 3-3: Sơ đồ cấu tạo động cơ bước VR

b, Động cơ bước nam châm vĩnh cửu.

Stato gồm một số răng trên đó mang dòng điện kích thích. Rôto là nam châm vĩnh cửu có số cực bằng số răng của stato. Để minh hoạ ta xét động cơ bước gồm hai cụm điển hình, mỗi cụm có một cuộn dây giả xử là cuộn A và cuộn B. Khi được kích thích các cuộn tạo ra các cực như nam châm điện. Các cực được tạo ra trên hai cuộn có vị trí lệch so với nhau một nửa. Nếu cuộn pha A được kích thích với điện áp dương sẽ tạo ra các cực từ như hình vẽ. Cực từ của nam châm vĩnh cửu của rôto sẽ tương tác với cực từ được sinh ra trên stato theo nguyên tắc cùng dấu thì đẩy nhau trái dấu thì hút nhau, đây chính là nguyên nhân làm cho rôto chuyển động. Trên hình a là pha A được kích thích với điện áp dương, khi pha A ngắt điện cho pha B được kích thích với điện áp dương thì rôto sẽ dịch thêm một đoạn cùng chiều đó. Để chuyển động thì pha B không được kích thích và pha A sẽ được kích thích với điện áp âm.



Hình 3-4: Sơ đồ cấu tạo động cơ bước nam châm vĩnh cửu

1,2 : Hai nửa stato có dạng cực móng được từ hoá với cực N và S xen kẽ nhau.

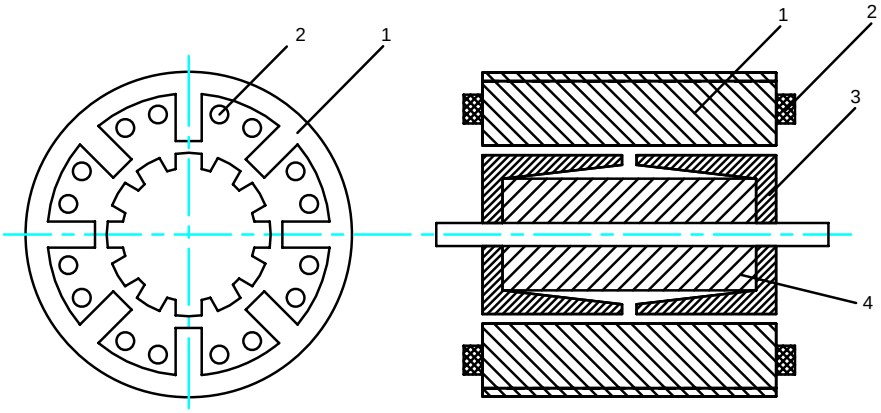
3 : Hai cuộn stato (một cuộn điều khiển đơn cực và một cuộn điều khiển lưỡng cực) được đặt ở trong hai nửa stato.

4 : Rôto nam châm vĩnh cửu có các cực từ xen kẽ.

Động cơ PM có đặc điểm là khi động cơ bị cắt nguồn cấp cho các pha thì vẫn còn có từ trường dư ở các pha stato tác dụng với từ trường nam châm vĩnh cửu trên rôto vì vậy động cơ vẫn được hãm ở vị trí ổn định. Mômen này có thể coi như mômen hãm của động cơ bước, trong khi đó động cơ bước kiểu VR không có khả năng hãm này khi tất cả các cuộn dây không được kích thích do vật liệu làm rôto không có từ dư.

c, Động cơ bước kiểu lai (Hybrid)

Là sự kết hợp của hai loại động cơ kiểu VR và PM, cấu trúc stato gồm một số cực lớn, trên mỗi cực lớn lại được chia thành các răng nhỏ. Rôto là nam châm hình trụ có nhiều răng với kích cỡ giống kích cỡ răng stato. Gốc bước của nó phụ thuộc vào số răng trên stato và rôto.



Hình 3-5: Sơ đồ cấu tạo động cơ bước kiểu lai.

1: Hai pha điều khiển lưỡng cực.

2: Stato dạng răng.

3: Cuộn dây pha điều khiển lưỡng cực.

4: Hai vành răng mgoài của rôto

Phần stato được cấu tạo hoàn toàn giống stato của loại động cơ bước có từ trở thay đổi. Trên các cực của stato được đặt các cuộn dây pha, mỗi cuộn dây pha đượcquấn thành 2 hoặc 4 cuộn dây dặt xen kẽ nhau hình thành lên các cực N và S, đồng thời đối diện với mỗi cực bối dây là các răng của rôto và cũng được đặt xen kẽ giữa hai vành răng của stato.

**** Các thông số của động cơ bước:***

Góc quay:

Động cơ bước quay một góc xác định đối với mỗi xung kích. Góc bước θ càng nhỏ thì độ phân giải càng cao. Số bước s là một thông số quan trọng.

$$s = \frac{360^0}{\theta} \quad (1)$$

Tốc độ quay và tần số xung:

Tốc độ quay của động cơ bước phụ thuộc vào số bước trong một giây. Đối với hầu hết các động cơ bước, số xung cấp cho động cơ bằng số bước nên tốc độ có thể tính theo tần số xung f. Tốc độ quay của động cơ bước có bước tính theo công thức sau:

$$n = \frac{60f}{S} \quad (2)$$

Trong đó : n là tốc độ quay (vòng / giây), f là tần só bước (Hz), s là số bước.

Ngoài ra còn các thông số quan trọng khác như độ chính xác vị trí và tỷ số momen và quán tính roto. Độ chính xác vị trí của động cơ bước phụ thuộc vào đặc tính của động cơ, vào độ chính xác chế tạo... Tỷ số momen và quán tính roto có ảnh hưởng quyết định đến khả năng dừng ngay khi chuỗi xung điều khiển đã ngắt.

3.1.4 Động cơ biến tần.

Việc điều khiển động cơ thông qua việc phát các xung điều khiển từ bộ điều khiển . Động cơ được điều khiển tốc độ bằng cách thay đổi tần số thông qua bộ biến tần (Inverter). Để thay đổi tốc độ của động cơ ta thay đổi tần số thông qua các phím chọn trên bề mặt của Inverter. ứng với một tần số của Inverter ta có một tốc độ của động cơ. Để đảm bảo cho động cơ làm việc an toàn trên động cơ người ta gắn thêm một phanh điện từ. Phanh này được nối qua một rơle và được điều khiển bởi bộ điều khiển. Khi có tín hiệu điều khiển bộ điều khiển ngắt điện áp của rơle và khi đó dòng 220V vào phanh bị cắt (phanh mở) và động cơ làm việc. Khi hiệu giữa giá trị khoảng cách thực và giá trị khoảng cách cần đạt được một giá trị Δ nào đó (do bộ đo thực hiện) bộ điều khiển cấp cho rơle một điện áp và khi đó phanh hoạt động, dừng động cơ đúng với yêu cầu.

3.2 Cảm biến trong rôbốt công nghiệp

3.2.1 Giới thiệu chung:

Trong tất cả các hệ thống tự động, thiết bị tiếp nhận thông tin về diễn biến của môi trường và về diễn biến của các đại lượng vật lý bên trong hệ thống được gọi là cảm biến. Khái niệm cảm biến trong tiếng việt chưa thật chính xác với tiếng Anh (sensor), hay tiếng Pháp (capteur), vì nghĩa cảm biến trong tiếng Việt có phần hẹp hơn. Cảm biến đôi khi chỉ là các trang bị đơn giản dạng như công tắc mini, các công tắc hành trình, các thanh lưỡng kiên (bimetal)...

Đối với người sử dụng, việc nắm được nguyên lý cấu tạo và các đặc tính cơ bản của cảm biến là điều kiện tiên quyết để đảm bảo sự vận hành tốt của hệ thống tự động. Trong các hệ thống vật lý, các đại lượng điều khiển rất đa dạng, do vậy các loại cảm biến cũng rất phông phú.

Trong robot công nghiệp, các thiết bị cảm biến trang bị cho robot để thực hiện việc nhận biết và biến đổi thông tin về hoạt động của bản thân robot và của môi trường, đối tượng mà robot phục vụ. Theo phạm vi ứng dụng các loại cảm biến dùng trong kỹ thuật robot có thể phân ra hai loại:

- Cảm biến nội tín hiệu (internal sensor) đảm bảo thông tin về vị trí, về vận tốc, về lực tác động trong các bộ phận quan trọng của robot. Các thông tin này là những tín hiệu phản hồi phục vụ cho việc điều chỉnh tự động các hoạt động robot.

- Cảm biến ngoại tín hiệu (external sensor) cung cấp thông tin về đối tác và môi trường làm việc phục vụ cho việc nhận dạng các vật xung quanh, thực hiện di chuyển hoặc thao tác trong không gian làm việc. Để làm được việc đó, cần có các loại cảm biến tín hiệu xa, cảm biến tín hiệu gần, cảm biến " Xúc giác" và cảm biến " thị giác" ... Để thực hiện nhiệm vụ các loại cảm biến nội tín hiệu và ngoại tín hiệu nói trên, có thể dùng nhiều kiểu cảm biến thông dụng hoặc chuyên dụng. Các kiểu cảm biến thông dụng không chỉ dùng cho kỹ thuật robot mà còn dùng nhiều trong các thiết bị kỹ thuật khác. Có nhiều tài liệu kỹ thuật về các kiểu cảm biến này.

Tuỳ theo các dạng tín hiệu cần nhận biết mà phân thành các kiểu cảm biến khác nhau:

Cảm biến lực, vận tốc, gia tốc, vị trí, áp suất, lưu lượng, nhiệt độ...

Tuỳ theo cách thức nhận tín hiệu lại phân ra các kiểu khác nhau. Ví dụ: cũng là cảm biến vị trí nhưng có kiểu cảm ứng, kiểu điện dung, kiểu điện trở, kiểu điện quang...

3.2.2 Các loại cảm biến.

3.2.2.1. Cảm biến vị trí

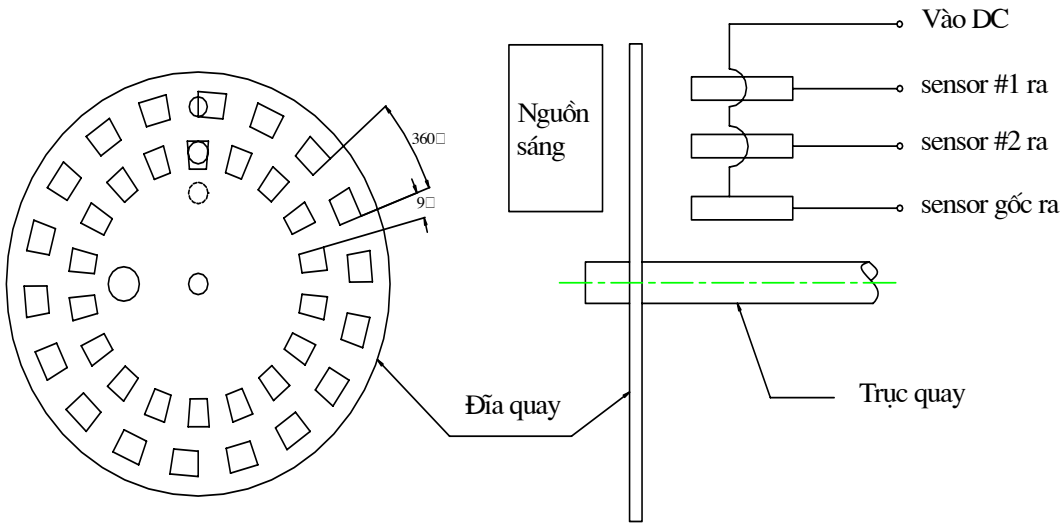
Cảm biến vị trí đặc điểm, như tên gọi của chúng, được dùng để giám sát vị trí tức thời của các cơ cấu. Tuỳ theo dạng chuyển động cần quan tâm mà vị trí có thể được tính theo đơn vị dài hay đơn vị góc. Nhờ các chuyển đổi cơ khí cần thiết có thể dùng sensor đo góc để đo chiều dài và ngược lai. Các sensor đo chiều dài có thể là biến trở, biến thế vi sai, encoder thẳng. Để đo góc quay có các loại sensor đo góc, như biến trở quay, encoder góc,resolver.

Encoder là loại thước đo vị trí theo kiểu số, trong đó toạ độ được mã hoá theo hệ nhị phân. Tuỳ theo đơn vị đo, chúng ta có encoder thẳng (linear encoder) hay encoder góc (rotary encoder). Hai loại này giống nhau về nguyên lý làm việc, chỉ khác nhau ở chỗ các vạch được khắc theo đường thẳng hay theo vòng tròn. Theo phương pháp mã hoá, có 2 loại encoder là tuyệt đối (absolute) hay gia số (incremental).

Thước đo vị trí kiểu gia số có 1 hoặc 2 đĩa quang, được khắc các vùng trong và đục xen kẽ nhau. Nếu dùng một đĩa thì nó được gắn với trục quay. Nếu dùng 2 đĩa thì một đĩa gắn với truc quay, còn đĩa kia cố định. Một phía của đĩa đặt nguồn sáng phía đối diện đặt trục 3 “con mắt điện” để thu tín hiệu của từng vòng tròn. Tại một vị trí nhất định của đĩa, vùng nào cho tia sáng đi qua sẽ được mã hoá là 1, vùng nào ngăn tia sáng sẽ được mã hoá là 0. Số vùng sáng tối trên đĩa quyết định độ phân giải của encoder.

Tại thời điểm bắt đầu làm việc, hệ thống phải được quy không bằng cách quay lỗ sát vòng tròn thứ 2 tới vị trí đối diện nguồn sáng để con mắt thứ 3 nhìn thấy tia sáng. Khi hệ thống bắt đầu làm việc, một bộ xử lý sẽ đếm số lần

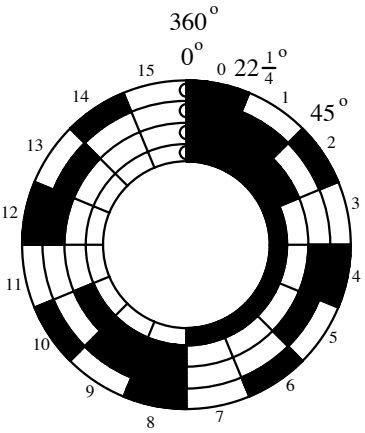
con mắt ngoài cùng nhìn thấy tia sáng, từ đó tính ra góc mà đĩa phải quay. Chiều quay của đĩa được nhận biết nhờ sự phối hợp tín hiệu của 2 vòng: nếu đĩa quay theo chiều kim đồng hồ thì mắt ngoài cùng nhìn thấy tia sáng trước mắt thứ hai và ngược lại. Căn cứ vào chiều quay mà gia số sẽ được cộng hoặc trừ vào tổng số.



Hình 3-6 : Sơ đồ nguyên lý của thước đo vị trí theo gia số.

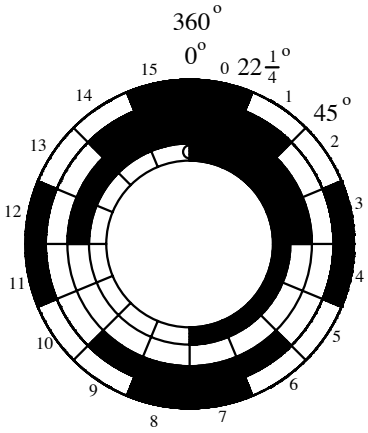
Thước đo vị trí tuyệt đối có một đĩa quang, trên đó có nhiều vòng tròn đồng tâm. Mỗi vòng chứa các vùng trong và đục xen kẽ nhau. Số vòng tròn quyết định độ phân giải của encoder. Nếu số vòng tròn là công suất thì số phần mà một vòng tròn có thể được chia ra bằng 2n, góc nhỏ nhất mà encoder phân biệt được là 360°/2n. Ví dụ nếu số vòng là n = 4 thì số phần chia của vòng tròn là 2^4 = 16, encoder sẽ phân biệt được góc quay 360°/16 = 22,5°. Nếu n = 8 thì góc đó là 360°/4096 = 0.088°.

Đĩa mã hoá nhị phân



Vùng	Binary	Xám
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110
5	0101	0111
6	0110	0101
7	0111	0100
8	1000	1100
9	1001	1101
10	1010	1111
11	1011	1110
12	1100	1010
13	1101	1011
14	1110	1001
15	1111	1000

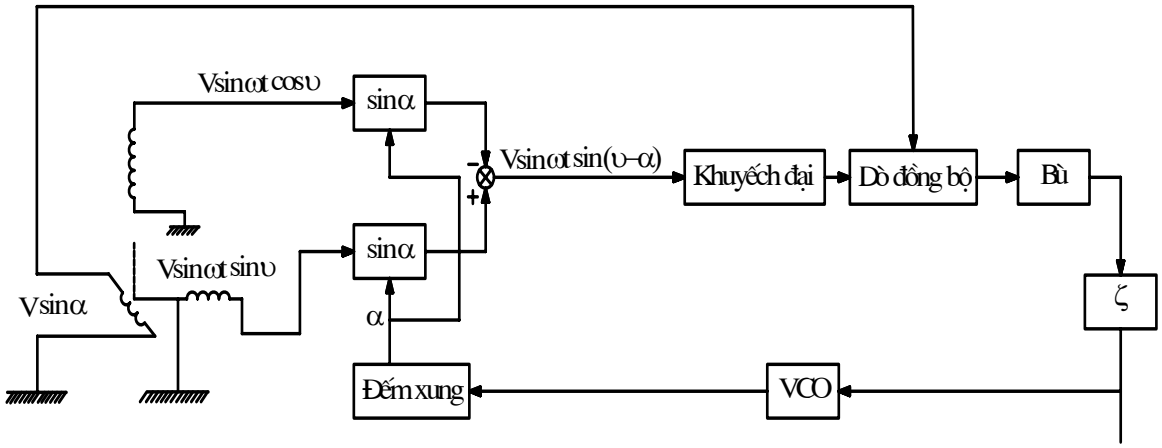
Đĩa mã xám



Hình 3-7 : Sơ đồ nguyên lý của thước đo vị trí tuyệt đối

Resolver không phát tín hiệu như encoder mà phát ra tín hiệu tương tự đại diện cho vị trí của đối tượng đo. Nhìn vẻ ngoài nó giống động cơ điện nhưng nguyên lý làm việc của nó giống biến thế nhiều hơn. Cuộn dây rotor được cấp điện áp DC thông qua các vành dẫn điện.

Điện áp cung cấp cho rotor dạng hình sin, dạng Vsinωt, tần số trong khoảng 0,4÷10 kHz. Stator của resolver có 2 cuộn dây, đặt lệch nhau 90°, còn trên cuộn dây kia có điện áp Vsinωt.sinθ. Rõ ràng giá trị điện áp ra phụ thuộc góc θ giữa rotor và stator. Tín hiệu phản hồi α của góc quay được cung cấp cho 2 cuộn dây qua hàm sinα và cosα, sau khi nhân với tín hiệu đầu vào và cộng đại số được tín hiệu ra là Vsinωt.sin(θ-α). Tín hiệu này được khuếch đại và gửi tới khối đồng bộ, đảm bảo giá trị của nó lphải tỉ lệ với sin(θ-α). Nếu có sai lệch, tín hiệu này được bù bởi thiết bị bù. Sau đó, tín hiệu được tích phân. Mạch phản hồi có bộ tạo dao đông, chuyển đổi điện áp thành tần số, và khối đếm xung. Giá trị α đại diện cho góc quay θ.



Hình 3-8 : Sơ đồ nguyên lý làm việc và xử lý tín hiệu của resolver.

3.2.2.2 Cảm biến vị trí kiểu biến áp:

Cảm biến kiểu này hoạt động theo nguyên lý của một biến áp sai động (differential transfomer): lõi từ chuyển động tương đối với hai cuộn dây cố định và làm việc thay đổi dòng cảm ứng giữa chúng.

Roto có một cuộn dây, còn stato có một hoặc một vài cuộn dây cảm ứng đồng bộ.

Giả thiết cấp vào cuộn dây của roto một điện áp:

$u_r = U \sin \omega t$

thì trong các cuộn dây trên stato sẽ xuất hiện các điện áp cảm ứng:

$u_{s1} = KU_r \cos r \sin (\omega t + \varphi)$

$u_{s2} = KU_r \cos r \sin (\omega t + \varphi)$

Trong đó:

K: hệ số biến áp

φ: độ lệch pha

Từ các công thức trên ta có thể xác định được góc r.

Tuy nhiên sự biến động biên độ điện áp làm giảm độ chính xác tính toán. Bởi thế người ta sử dụng phương pháp giải điều biến đồng bộ trên cơ sở đối chiếu với tín hiệu chuẩn. Trong trường hợp biến áp biến áp xoay nhiều thì độ

chính xác có thể tăng lên. Người ta thường sử dụng kết hợp bộ biến áp xoay chiều với bộ biến áp điện tử số để nâng cao độ chính xác.

Cảm biến vị trí kiểu biến áp làm việc tin cậy, chính xác và phù hợp với dải rộng tần số quay. Vì không dùng đến vòng tiếp điểm như kiểu chiết áp nên chúng có thể làm việc với vận tốc cao (khoảng 9000 vòng/phut). Tuy nhiên lại đòi hỏi độ chính xác cao về chế tạo và lặp ráp, nên giá thành cũng cao hơn nhiều.

Tương tự, đối với trường hợp tịnh tiến có cảm biến vị trí kiểu chiết áp sai động thay đổi tuyến tính LVDT (Linear Variable Differential Transformer). Bao bọc xung quanh lõi từ là một quộn dây sơ cấp và hai cuộn dây thứ cấp giống hệt nhau. Khi dịch chuyển lõi từ độ từ cảm tương hỗ giữa cuộn dây sơ cấp và các cuộn dây thứ cấp sẽ biến đổi tuyến tính theo độ dịch chuyển đó.

3.2.2.3 Vận tốc kế:

Phạm vi sử dụng vận tốc kế (tachometers) và gia tốc kế (accelerometers) ngày nay trở nên thu hẹp, mà thông dụng hơn là xác định theo số gia của các thông tin cảm biến vị trí bằng kỹ thuật số hay kỹ thuật vi mạch. Cách làm đó đạt được độ chính xác cao hơn và ngày càng rẻ hơn.

Thường dùng các thiết bị sau để thực hiện chức năng của vận tốc kế:

- Phát tốc (tachogenerator) dùng một chiều.

Khi Stator được kích từ thì tạo nên từ thông, khi đó trong các cuộn cảm trên roto có suất điện động từ cảm với giá trị trung bình tỷ lệ với góc quay của roto.

- Phát tốc không đồng bộ dòng xoay chiều.

Khi dòng điện xoay chiều chạy qua cuộn dây sơ cấp của Stato thì điện áp trên các đầu dây cuộn thứ cấp của Stato sẽ cùng tần số, biên độ và tỷ lệ thuận với vận tốc góc của roto.

- Phát tốc đồng bộ.

Nam châm vĩnh cửu trên Stato gây dòng điện cảm trong cuộn dây của Stato. Tần số và giá trị điện áp của nó tỷ lệ thuận với vận tốc của roto.

- Phát tốc xung.

Đĩa quay có khe rãnh đặt trước nguồn sáng tạo ra các xung có tần số tỷ lệ với vận tốc góc quay.

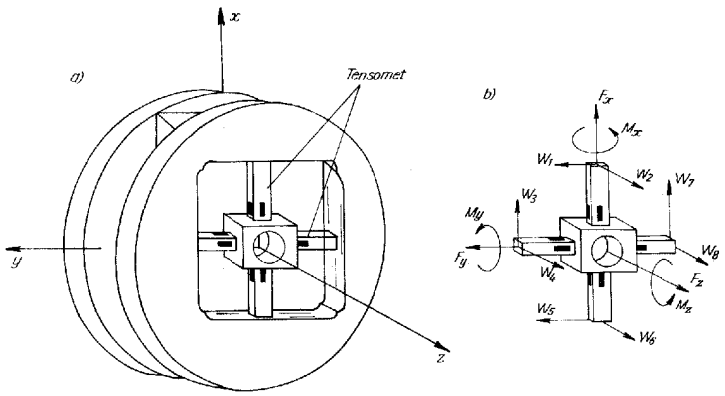
3.2.2.4 Cảm biến lực và cảm biến xúc giác:

3.2.2.4.1 Cảm biến lực:

Cảm biến lực chủ yếu dùng để nhận biết phản lực xuất hiện trong khi lặp ráp các chi tiết máy. Đôi khi cũng dùng để nhận tín hiệu lực trong các khớp động. Thương thường các cảm biến lực được lập ở các khớp quay, khớp cổ tay hoặc trực tiếp trên bàn kẹp. Có nhiều phương pháp nhận biết lực dùng trong cơ cấu cảm biến. Ví dụ trực tiếp suy ra từ sự biến thiên dòng điện ở động cơ một chiều lắp ở các khớp quay. Nhưng phổ biến là dùng phương pháp quen biết do biến dạng dùng các tensomet. Chúng còn được gọi là "tem biến dạng", vì trông giống như những con tem nhỏ chứa bên trong một dây điện trở và được dán trên các thanh biến dạng. Dưới đây giới thiệu một kiểu cảm biến lực thường được lắp ở khớp quay cổ tay, chỗ nối với bàn kẹp và gọi tắt là cảm biến lực chữ thập.

Cảm biến lực chữ thập

Trên hình 3.4 mô tả một kiểu cảm biến lực khớp quay bàn kẹp, với bộ phận chủ yếu có hình chữ thập. Trên mỗi nhánh của hình chữ thập được dán hai cặp tensomet. Qua những cầu đo, sẽ nhận biết được lực tác động lên mỗi nhánh chữ thập.



Giả sử rằng ảnh hưởng qua lại giữa các tensomet là rất nhỏ, có thể bỏ qua. Như vậy từ các số đo của 8 cặp tensomet, bằng cách cộng hoặc trừ giữa chúng tuỳ theo các trường hợp cụ thể, có thể xác định được 3 thành phần lực F (F_x,F_y,F_z) và 3 thành phần momen M (M_x,M_y,M_z). Gọi chung F_M là một vecto mở rộng bao gồm 6 thành phần nói trên và W có thành phần là 8 thành phần do được (W_1,W_2,...,W_8).

$$F_M = R_F W$$

Với $R_F = \begin{bmatrix} r_{11}...r_{18} \\ .... \\ r_{61}...r_{68} \end{bmatrix}$

Các phần tử $r_{ij} \neq 0$ của ma trận R_F là các hệ số tương ứng dễ biến đổi các số đo theo đúng thứ nguyên của chúng: W(von), F(N), M(Nm).

Theo bố trí các tensoment như trên hình 3.4 thì một số phần tử r_{ij} sẽ bằng không và ma trận trên sẽ như sau:

$$R_F = \begin{bmatrix} 0 & 0 & r_{31} & 0 & 0 & 0 & r_{17} & 0 \\ r_{21} & 0 & 0 & 0 & r_{25} & 0 & 0 & 0 \\ 0 & r_{32} & 0 & r_{34} & 0 & r_{36} & 0 & r_{38} \\ 0 & 0 & 0 & r_{44} & 0 & 0 & 0 & r_{48} \\ 0 & r_{52} & 0 & 0 & 0 & r_{56} & 0 & 0 \\ r_{61} & 0 & r_{63} & 0 & r_{65} & 0 & r_{67} & 0 \end{bmatrix}$$

Từ ( ) và ( ) có các thành phần của F_M, ví dụ:

$F_x = r_{13}W_3 + r_{17}W_7$ (*)

Biểu thức (*) hoàn toàn phù hợp với hình 3.4

Để xác định các hệ số r_{ij} trong (*) người ta thường chuẩn mức hoá (calibration) kết hợp tính toán và thực nghiệm. Bản chất việc chuẩn hoá mức ở đây là thiết lập một ma trận R_F^* thoả mãn điều kiện sau:

$W = R_F^* F_M$ (**)

R_F^* là ma trận 8 x 6.

Nhân 2 vế (**) với $(R_F^*)^T$, ta có:

$(R_F^*)^T W = [(R_F^*)^T R_F^*]^{-1} F_M$

Từ đó:

$F_M = [(R_F^*)^T R_F^*]^{-1}(R_F^*)^T W$

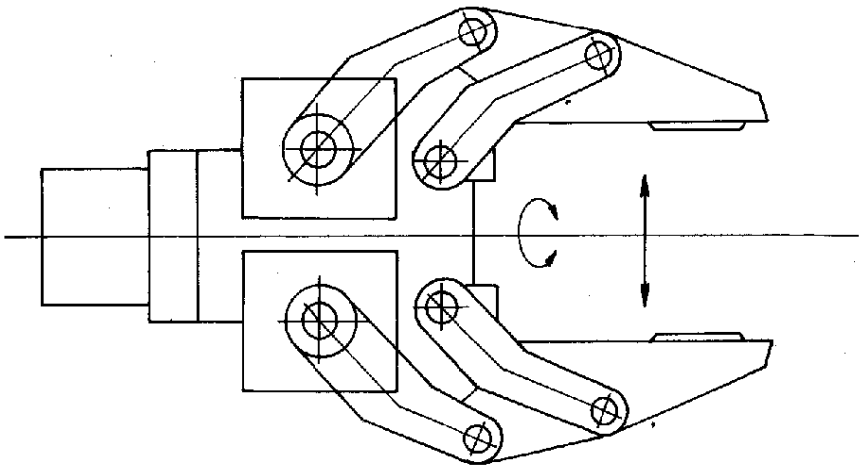
So sánh (11.5) và (11.11), ta có:

$R_F \approx [(R_F^*)^T R_F^*]^{-1}(R_F^*)^T$.

Ma trận R^*_F được thiết lập trên cơ sở các công thức trên và các kết quả thực nghiệm khi cho trước các giá trị của F_M.

***3.2.2.4.2 Cảm biến xúc giác**:*

Trong kỹ thuật người máy thường dùng các cảm biến xúc giác để nhận thông tin về sự tiếp xúc của bàn kẹp với đối tượng hoặc lực cần kẹp...



y kẹp

Có thể phân cảm biến xúc giác ra hai nhóm cơ bản: Cảm biến có ngưỡng tín hiệu rời rạc và cảm biến có tín hiệu tương tự. Cảm biến xúc giác có ngưỡng tín hiệu rời rạc. Đây thực chất là những công tắc tinh vi được lắp đặt ở phía bên trong của các ngón bàn kẹp để cho biết thông tin đã có vật kẹp giữa các ngón chưa, cần kẹp vào vị trí nào của vật...

Các công tắc tinh vi có thể lắp đặt trên bề mặt tiếp xúc phía trong của bàn kẹp. Ngoài ra còn có thể lắp đặt ở phía ngoài bàn kẹp hoặc ở các đầu dò của cơ cấu tay máy để nhận biết thông tin về các chướng ngại vật trên đường di chuyển.

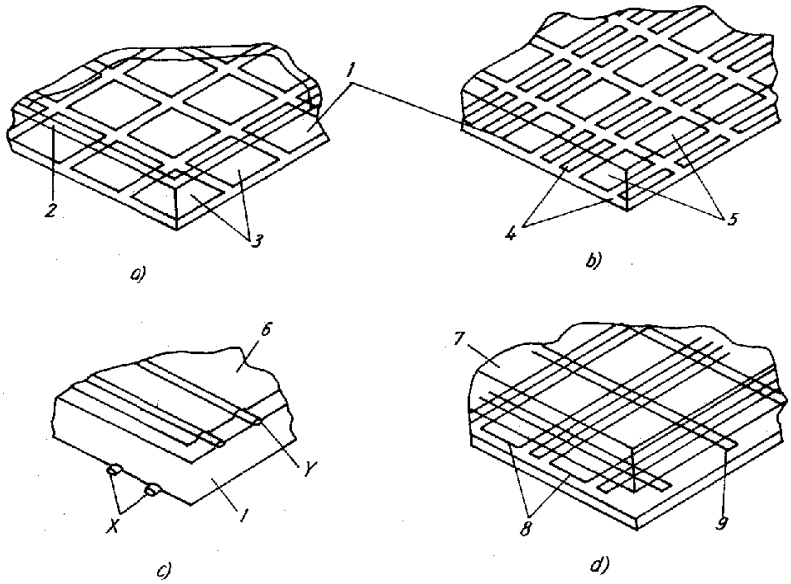
Cảm biến xúc giác dùng tín hiệu tương tự loại đơn giản nhất. Nó gồm một thanh tỳ vào vật nhờ lực lò xo. Lực ngang từ phía lực tác dụng vào thanh làm chuyển động đi một góc quay(ví dụ bằng cơ cấu thanh răng bánh răng). Góc quay này tỷ lệ với lực ngang và được liên tục ghi đo bằng cơ cấu chiết áp. ứng với độ cứng vững đã biết của lò xo, có thể xác định được lực theo độ dịch chuyển của góc ghi đó.

Trong những năm gần đây một vấn đề rất được chú ý là tạo ra các bề mặt xúc giác có khả năng thu nhận cùng một lúc một lượng lớn thông tin. Trên hình 3-10 là một ví dụ minh hoạ. Phía bên trong các ngón của bàn kẹp được

gắn các mặt xúc giác (1). Các mặt phía bên ngoài (2) của bàn kẹp có thể gắn cả các cảm biến tín hiệu rời rạc.

Mặc dù mặt xúc giác thường là tổ hợp nhiều loại cảm biến riêng lẻ, nhưng một hướng có chuyển vọng là cùng dùng các tấm vật liệu bán dẫn, ví dụ trên nền grafit có điện trở thay đổi theo áp lực. Sự thay đổi điện trở dễ dàng biến thành dao động tín hiệu điện. Biên độ của nó tỷ lệ với lực tác động lên bề mặt vật kẹp tại điểm tiếp xúc. Để tạo các mặt xúc giác mà đôi khi còn gọi là " lớp da nhân tạo" có thể có các phương pháp chủ yếu.

Phương pháp ở hình 3-11a tạo ra các cửa sổ giữa khối vật liệu dẫn (1) nằm giữa thân vỏ (2) và hệ thống điện cực (3). Mỗi điện cực là một ô vuông cửa sổ tiếp nhận tín hiệu tại một điểm tiếp xúc với vật. Tuỳ thuộc vào áp lực lên vật liệu bán dẫn, dòng điện từ thân vỏ đến điện cực sẽ biến đổi. Theo phương pháp ở hình 13b có các cặp điện cực (4) trên các tấm mạch điện (5) thì vật liệu dẫn nằm phía trên và cách điện với tấm này trừ các chỗ tiếp xúc với điện cực. áp lực từ đối tượng làm thay đổi điện trở và nhờ các mạch điện tử biến đổi thành các tín hiệu khác nhau.



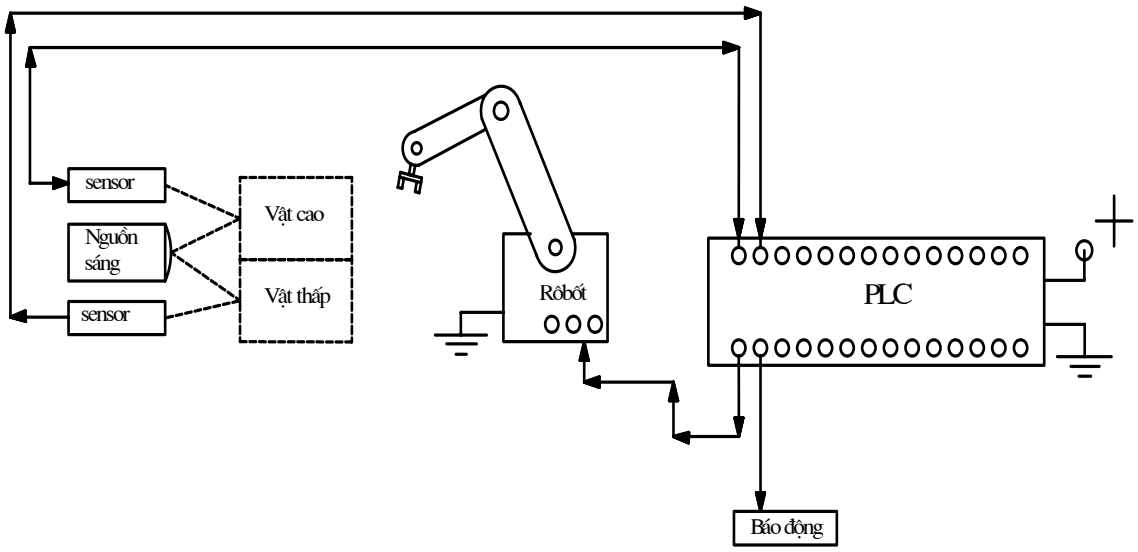
iác

Trong các phương pháp hình 4.5c, vật dẫn điện (1) đặt giữa hai bộ điện cực ngang (X), dọc (Y) vuông góc với nhau. Điểm chéo nhau giữa hai thanh điện cực X,Y có vật dẫn điện chen giữa thành một điểm nhạy cảm. Bộ điện cực có lớp vỏ cao su (6) chuyển dịch tạo lực từ phía ngoài tác động lên mỗi điểm nhạy cảm làm thay đổi điện trở của vật dẫn và do vậy làm thay đổi dòng điện.

3.2.2.5 Thiết bị quan sát (Visual System)

Thiết bị quan sát là một loại sensor đặc biệt, có khả năng nhận biết và xử lý hình ảnh của đối tượng. Thiết bị quan sát được ứng dụng rộng rãi trong công nghiệp nói chung, song ứng dụng trong rôbốt là ứng dụng đặc trưng nhất. Mặt khác, xử lý hình ảnh là một trong những lĩnh vực phát triển mạnh nhất của công nghệ thông tin hiện đại, nên thiết bị quan sát trên rôbốt gắn liền máy tính.

Với một hệ thống quan sát đơn giản rôbốt, để nhận biết 2 vật: một vật cao và một vật thấp. Hệ thống có một nguồn sáng và 2 sensor thu sáng. Nếu có vật cao trước nguồn sáng thì cả 2 sensor đều nhận được ánh sáng phản xạ. Nếu chỉ có vật thấp thì chỉ riêng sensor thấp nhận được. Nếu không có vật nào thì không sensor nào nhận được tín hiệu. “Hình ảnh” của đối tượng chỉ gồm có 2 điểm ảnh (trong kỹ thuật xử lý ảnh, điểm ảnh được gọi là pixel).



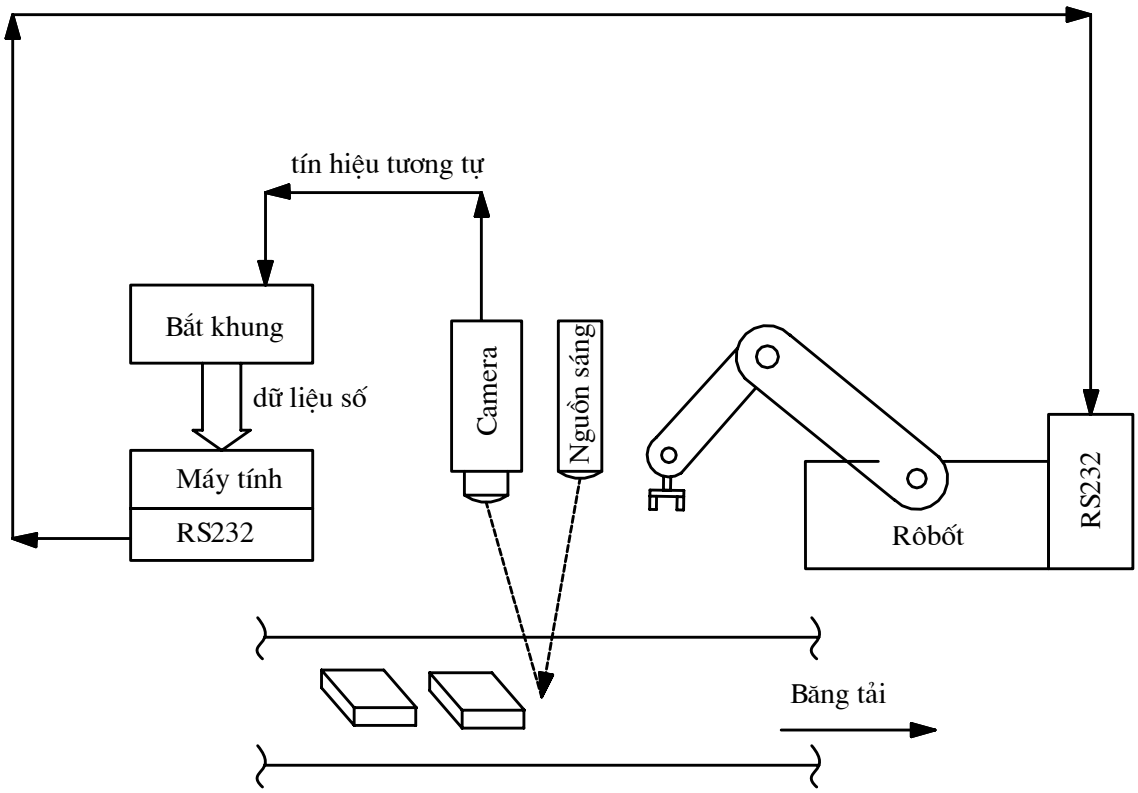
Hình 3-12 : Hệ thống quan sát 2 pixel

Tín hiệu về đối tương, tuy chỉ có 2 pixel nhưng cũng cần bộ phân tích, ví dụ một PLC. Nó sẽ được lập trình để:

- Báo hiệu cho rôbốt nếu vật cao trước mặt. Vì vậy rôbốt chỉ phản ứng với vật cao.

- Nổi hiệu lệnh cho người nếu chỉ có vật thấp trước mặt. Như vậy, người chỉ phải phản ứng nếu gặp vật thấp.

Gọi hệ thống trên là "thiết bị quan sát" có thể hơi lạm dụng từ này và hệ thống như vậy có lẽ không tồn tại trên thực tế, nhưng nó cho một hình dung ban đầu về thiết bị quan sát.



Hình 3-13: Hệ thống nhận dạng chi tiết

Thiết bị quan sát thật sự (Hình 6.130 cũng có các bộ phận cơ bản như hệ thống ví dụ ở trên. Đó là hệ thống đơn giản để nhận dạng chi tiết. Nó có khả năng phân biệt các chi tiết trong trường quan sát của mình.

Hệ thống nhận dạng nói trên gồm có:

- Nguồn sáng: Tia sáng do nó phát ra sẽ bị phản xạ bởi vật và được thu bởi...

- Camera, biến đổi quang năng thành điện năng, cung cấp cho ...

- Bộ thu ảnh, gồm mạch điện tử và phần mềm để phân tích tín hiệu thành các pixel và biểu diễn chúng dưới dạng mã nhị phân. Sơ đồ bố trí các điểm ảnh gọi là bitmap. Sơ đồ này sẽ được chuyển tới...

- Máy tính để lưu trữ và xử lý trực tiếp. Máy tính sẽ so sánh sơ đồ điểm ảnh của vật với sơ đồ điểm ảnh chuẩn (gọi là template) trong thư viện để xem vật với sơ đồ điểm ảnh chuẩn (Gọi là template) trong thư viện để xem vật thuộc loại nào. Máy tính sẽ chỉ cho rôbốt biết chi tiết nó đang nhìn thấy là chi tiết nào, thông qua...

- Giao diện đầu ra. Nó chuyển tín hiệu từ hệ thống nhận dạng cho bộ điều khiển rôbốt. Ví dụ một mã "H" (Nếu chi tiết là hộp), mã "C" (Nếu chi tiết là cờ lê) sẽ được truyền theo giao diện chuẩn RS232.

- Thiết bị nhận dạng càng chính xác nếu số ảnh trên một đơn vị diện tích ảnh (Nghĩa là độ phân giải) càng lớn. Đơn vị chuẩn của độ phân giải là dpi. Màn hình máy tính có độ phân giải cỡ 100 dpi, còn máy in lazer thường có độ phân giải cao hơn (Cỡ 300 dpi trở lên). Độ phân giải của ảnh càng lớn thì tốc độ xử lý và dung lượng bộ nhớ của máy tính càng phải cao. Khả năng nhận dạng chính xác của thiết bị quan sát cần cho những trường hợp sau:

- Phân biệt các chi tiết khá giống nhau.
- Phân biệt sản phẩm tốt và phế phẩm.
- Sử dụng màu sắc để nhận dạng đối tượng.
- Đo kích thước chi tiết.
- Nhận biết vật cản để tránh va chạm.
- Nhận biết khoảng cách và hướng của chi tiết.
- Nhận biết tốc độ và hướng chuyển động của đối tượng.
- Nhận biết đối tượng 3 chiều.

Các thiết bị nhận dạng mục tiêu của máy bay ném bom, nhận dạng đường cho các ô tô tự lái... là những ví dụ về các thiết bị quan sát hiện đại.

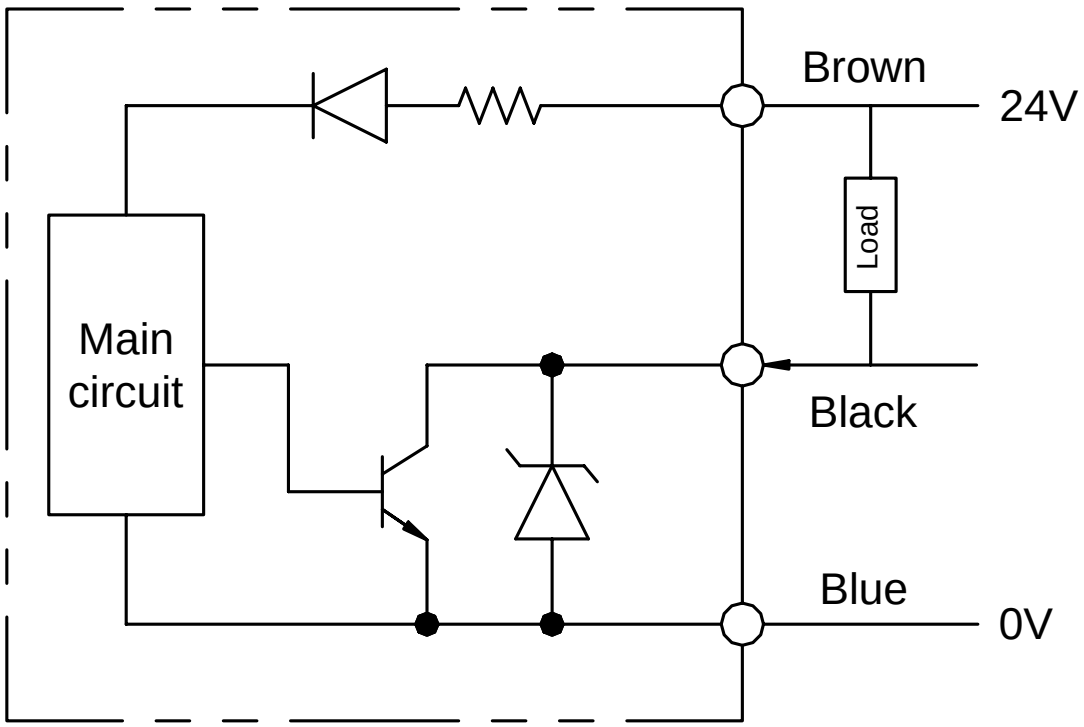
3.2.2.6 Các cảm biến thường gặp trong rôbốt gắp phôi tự động.

Các cảm biến được sử dụng trong robot là các cảm biến điện từ. Với nguyên lý hoạt động như sau: Khi cảm biến đối diện với các vật có từ tính sẽ gây ra hiện tượng thông mạch và trên đường tín hiệu ra có một điện áp ở mức

24V, và đưa về bộ điều khiển dưới dạng xung điện này. Sau khi nhận được tín hiệu từ các cảm biến bộ điều khiển sẽ có tín hiệu điều khiển tương ứng với các hoạt động của robot.

Mỗi cảm biến được sử dụng có 3 dây. Hai dây cung cấp nguồn (Brown và Blue) điện áp 24V, còn dây còn lại (Black) ở mức 0V là dây tín hiệu. Khi cảm biến đối diện với các chất có từ tính ở một khoảng cách nhất định, mạch điện áp đóng với điện áp 24V.

Sau đây là sơ đồ cấu trúc của cảm biến điện từ.



Hình 3-14: Sơ đồ cấu trúc cảm biến điện từ.

3.3 Hệ thống vận hành khí nén trong rôbốt công nghiệp

3.3.1 Giới thiệu chung về hệ thống truyền động tự động trong rôbốt công nghiệp

Hệ truyền động tự động thuỷ khí công nghiệp làm việc theo chu trình là 1 hệ truyền động gồm nhiều hệ truyền động cơ sở làm việc một cách lôgíc sao cho trình tự cơ cấu chấp hành thực hiện xong chúng trở lại vị trí ban đầu và lặp đi lặp lại.

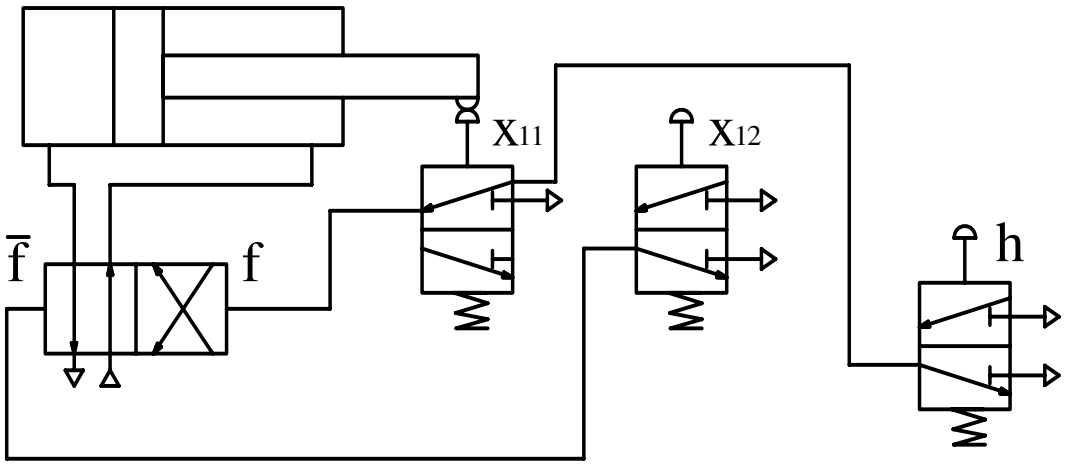
Hệ thống điều khiển các hệ truyền động thuỷ khí phải đảm bảo việc đóng mở các van phân phối tương ứng với các điều kiện làm việc cho trước. Các phương pháp cho điều kiện làm việc của máy tự động và phương pháp hiện thực chúng rất đa dạng. Khi thiết kế các máy tự động với các khâu cứng, điều kiện làm việc thường được cho dưới dạng các chu trình làm việc. Đó là một dạng đồ thị qui ước biểu diễn sự phụ thuộc vào thời gian dịch chuyển của các cơ cấu chấp hành.

Chu trình làm việc là 1 trình tự xác định dịch chuyển của cơ cấu chấp hành mà sau khi thực hiện xong chúng trở về vị trí ban đầu. Hoạt động của máy sẽ thể hiện trong việc thực hiện tuần tự các chu trình làm việc nối tiếp nhau. Với các máy có hệ thống truyền động khí nén, các điều kiện làm việc cũng có thể được mô tả bằng các chu trình hoặc các biểu đồ trình tự làm việc, nhưng thời gian của mỗi một chu trình làm việc không xác định bởi vận tốc các cơ cấu chấp hành mà phụ thuộc vào hàng loạt yếu tố phụ mà ta có thể điều chỉnh được.

Các hệ truyền động tự động khí nén làm việc theo chu trình được chia theo kiểu điều khiển thành 3 nhóm:

- Điều khiển theo vị trí:

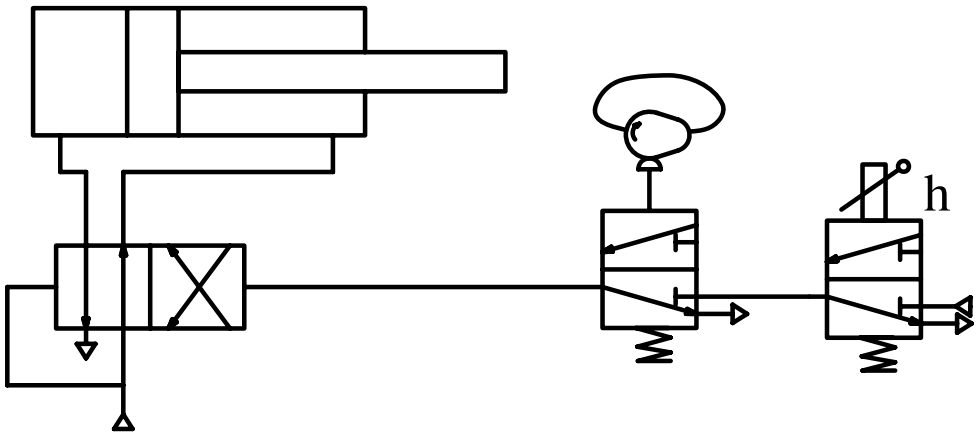
Vị trí tận cùng của các cơ cấu chấp hành được kiểm tra bằng các cảm biến vị trí.



Hình 3-15: Sơ đồ điều khiển xilanh khí nén theo vị trí

- Điều khiển theo thời gian:

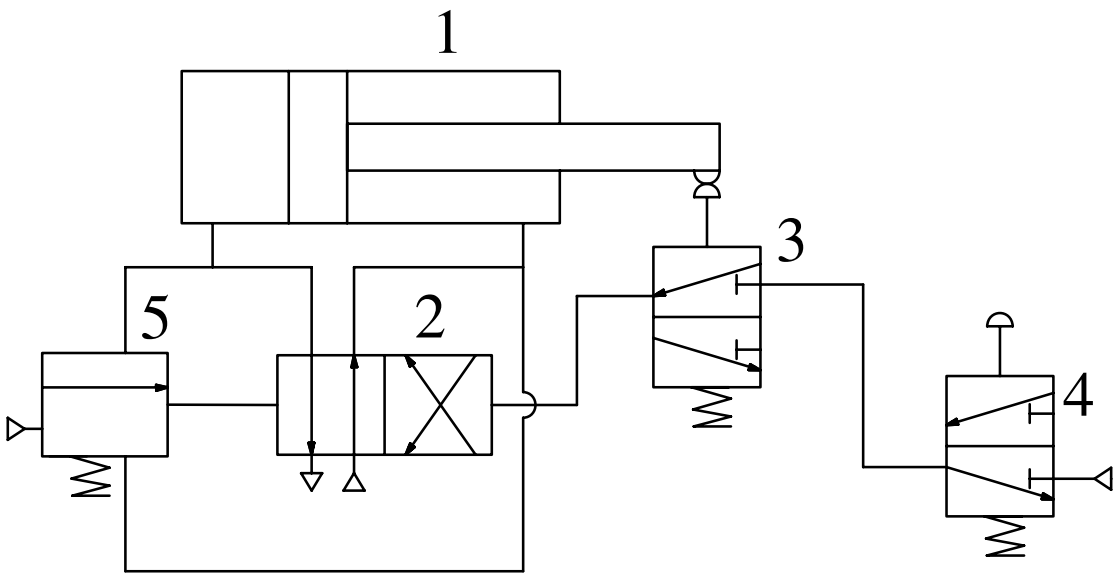
Thời gian thực hiện 1 chu trình được xác định bằng cơ cấu cam hoặc bằng rơle thời gian các loại.



Hình 3-16: Sơ đồ điều khiển theo thời gian bằng cam

- Điều khiển theo áp suất:

Được coi là biến thể của hệ điều khiển theo vị trí. Chúng được sử dụng trong các trường hợp khi cần piston dịch chuyển những khoảng khac nhau phu thuộc vào kích thước của chi tiết được gia công, hoặc do khó khăn trong việc lắp đặt các công tắc hành trình với cần piston vươn dài.



Hình 3-17: Sơ đồ điều khiển xilanh khí nén theo áp suất.

Trong đó:

1) Xilanh chấp hành khí nén

2) Van phân phối 4/2

3) Công tắc hành trình 3/2

4) Công tắc khởi động

Hai cách điều khiển theo thời gian và theo áp suất có nhược điểm là khi tải thay đổi đột ngột hoặc khi các thông số khí thay đổi, chuyển động của cơ cấu chấp hành có thể sảy ra trước. Do vậy các hệ điều khiển theo vị trí, trong đó chuyển động của từng cơ cấu chấp hành chỉ có thể bắt đầu theo 1 trình tự

vị trí xác định của tất cả các cơ cấu chấp hành còn lại là phổ biến trong hệ truyền động tự động khí nén.

3.3.2 Các phần tử của hệ thống điều khiển khí nén trong rôbốt cấp phôi tự động

3.3.2.1 Hệ thống cung cấp và xử lý khí nén

- *Máy nén khí*

a/ Khái niệm: Là những thiết bị chuyển đổi công suất hiệu dụng từ mô tơ điện hoặc động cơ đốt trong thành năng lượng khí nén (áp năng) và nhiệt năng.

b/ Phân loại:

Phân loại theo áp suất:

- Máy nén khí áp suất thấp: P < 15
- Máy nén khí áp suất cao: P >15
- Máy nén khí áp suất rất cao P >30

Phân loại theo nguyên lý hoạt động:

- Máy nén khí theo nguyên lý thay đổi thể tích: Máy nén khí theo kiểu pittong, máy nén khí theo kiểu cánh gạt, máy nén khi theo kiểu root, máy nén khí theo kiểu trục vít.

- Máy nén khí tuabin: Máy nén khí ly tâm và máy nén khí chiều trục.

- *Bình trích chứa khí nén*

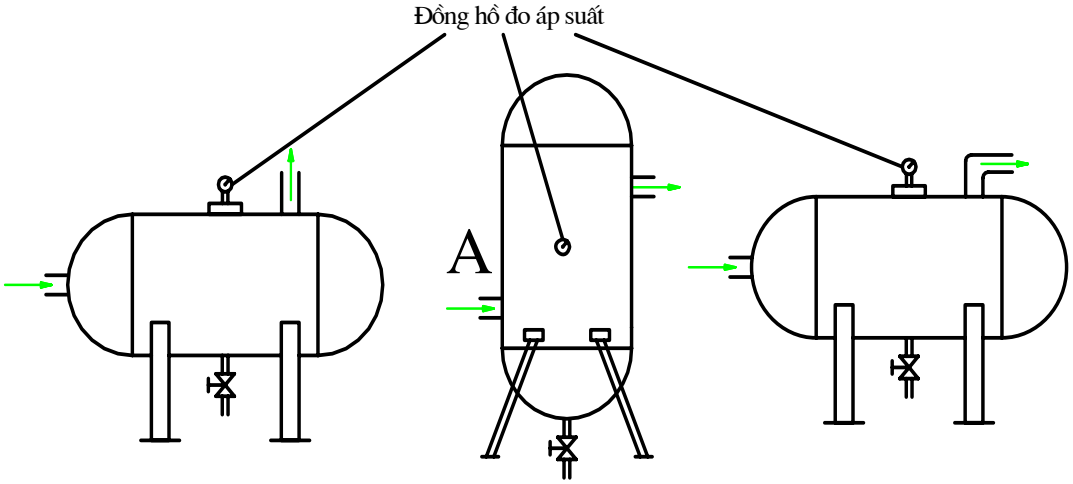
Khí nén sau khi ra khỏi máy nén khí được xử lý và cần phải có một bộ phận lưu chữ để sử dụng cho toàn hệ thống. Bộ phận đó là bình trích chứa khí nén, nó có các tác dụng sau:

- Lưu trữ khí nén nhờ đó hạn chế việc máy nén khí phải làm việc liên tục.

- Làm giảm sự xung động để làm dịu các xung dòng chảy của không khí từ máy nén.

- Chuyển đổi nhiệt, khí nén trong bình chứa sẽ được làm mát, tạo ra sự ngưng tụ nước và nước sẽ được tách ra khỏi khí nén trước khi khí nén đi vào hệ thống phân phối.

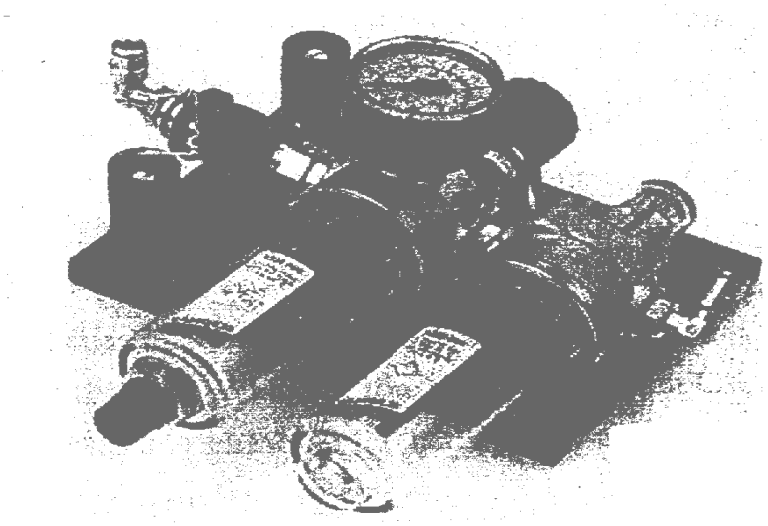
- Kích thước bình trích chứa phụ thuộc vào công suất của máy nén khí và công suất tiêu thụ của các thiết bị sử dụng trong hệ thống khí nén. Ví dụ, sử dụng liên tục hay gián đoạn.



Hình 3-18: Bình trích chứa khí nén

- *Bộ lọc*

Yêu cầu khí nén: Khí nén được tạo ra từ những máy nén khí có chứa đựng nhiều chất bẩn, độ bẩn có thể ở những mức độ khác nhau. Chất bẩn bao gồm: bụi, độ ẩm của không khí được hút vào, những phần tử nhỏ, chất cặn bã của dầu bôi trơn và truyền động cơ khí. Hơn nữa, trong quá trình nén, nhiệt độ khí nén tăng lên, có thể gây nên quá trình ôxy hoá các phần tử kể trên.



]

Như vậy, khí nén bao gồm các chất bẩn được dẫn đi trong ống dẫn khí sẽ gây ăn mòn, gỉ xét trong ống và trong các phần tử của hệ thống điều khiển. Do đó, khí nén sử dụng trong kỹ thuật cần thiết phải được xử lý. Mức độ xử lý khí nén tuỳ thuộc phương pháp xử lý, từ đó xác định chất lượng của khí nén tương ứng cho từng trường hợp ứng dụng cụ thể.

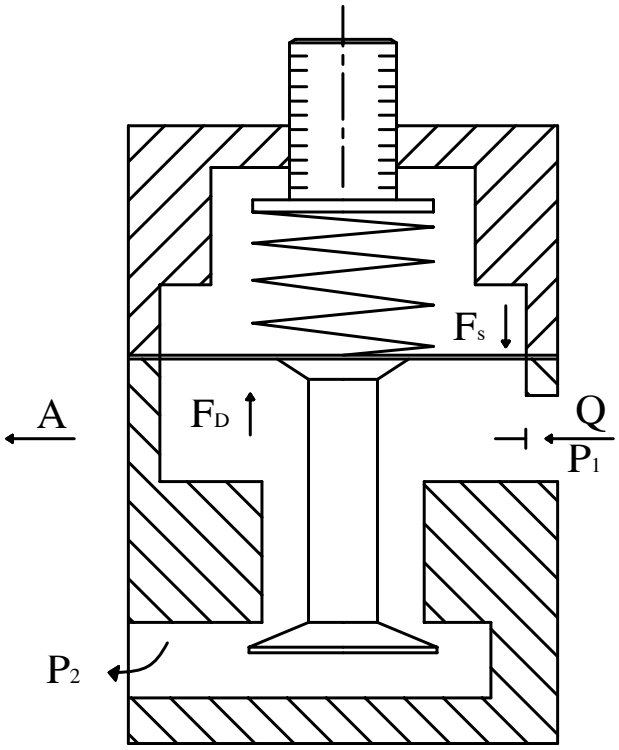
Nhiệm vụ

Bộ lọc trong đường ống được thiết kế để thực hiện hai chức năng: lọc các chất bụi bẩn và tách nước ngưng tụ trong không khí khi đi ngang qua nó.

Bộ lọc có chén lọc bằng nhựa trong suốt Polycarbonate. Chén lọc có thể bị xuống cấp sau một thời gian sử dụng và dưới đó dưới tác dụng của áp suất nó có thể bị vỡ, vì vậy cần phải có bộ phận bảo vệ bằng kim loại. Nếu chén lọc cũ, độ trong suốt không còn thì phải thay chén mới, cũng có thể thay bằng chén kim loại. Khi cần rửa sạch chén lọc, có thể dùng bằng nước xà phòn, nước rửa chén hoặc dầu lửa, không được dùng các dung dịch giải và các dung dịch dung môi khác.

- *Van giảm áp*

Vì áp suất hệ thống phân phối luôn luôn lớn hơn áp suất yêu cầu của các thiết bị hoặc hệ thống vận hành bằng khí nén, nên việc giảm áp là rất cần thiết. Mặc dù thuật ngữ " van giảm áp" đúng với chức năng của van, nhưng thuật ngữ thường dùng hơn là " bộ điều tiết áp suất". Vì lực tác dụng ở đầu ra của xilanh hoặc dụng cụ khí nén phụ thuộc vào áp suất hệ thống nên nói van giảm áp là một bộ điều tiết lực cũng đúng.



Hình 3-20: Kết cấu van giảm áp

Trong sơ đồ:

Fs: Lực của lò xo

P_2: áp suất thứ cấp

P_1: áp suất sơ cấp

A: diện tích màng.

F_d: Lực tác động lên màng do áp suất P_2 tạo nên.

Q: Lưu lượng khí qua van.

ΔP : Là độ chênh lệch áp suất giữa P_1 và P_2.

Có hai loại van giảm áp:

- Van giảm áp không có chức năng giảm áp hệ thống thứ cấp.
- Van giảm áp với chức năng giảm áp hệ thống.

Để hiểu nguyên lý hoạt động ta xem xét loại van giảm áp không có chức năng giảm áp hệ thống thứ cấp. Van giảm áp này có hai chức năng quan trong được tóm tắt như sau:

+ Duy trỳ áp suất thứ cấp(áp suất điều chỉnh) gần như không đổi, không phụ thuộc vào sự dao động về lưu lượng yêu cầu ở phía thứ cấp (phía ra).

+ Trong điều kiện áp suất sơ cấp P_1 không dao động và dòng khí yêu cầu phía thứ cấp ổn định, van duy trì ở một vị trí ổn định và các lực Fs, F_d bằng nhau, nhưng ngược chiều. Trong điều kiện này van sẽ xác lập một vị trí ổn định (được điều tiết bởi màng mà van gắn vào) tạo ra một sự giảm áp ΔP chính xác với yêu cầu.

Giải thích động học:

Khi yêu cầu về lưu lượng Q trong hệ thống tăng lên, áp suất P_2 sẽ giảm và sự cân bằng bị phá vỡ ($F_s > F_d$), lò xo sẽ đẩy màng đi xuống và lỗ thông khí của van mở rộng hơn, độ chênh lệch áp suất ΔP sẽ giảm và áp suất lại tăng lên, sự cân bằng được xác lập trở lại($F_s = F_d$).

Khi áp suất P_1 dao động còn áp suất P_2 cũng có khuynh hướng dao động theo nhưng màng sẽ điều chỉnh đỉnh van liên tục, nhờ đó P_2 được duy trì.

Khi áp suất P_2 tăng, $F_d > F_s$ màng đi lên, lỗ thông khí giảm, ΔP tăng. Vì vậy áp suất lại giảm.

Nếu phía thứ cấp không có yêu cầu dòng chảy khí nén thì áp suất ở buồng thứ cấp (P_2)sẽ tăng lên cho đến khi bằng với áp suất sơ cấp P_1 màng được đẩy xuống phía dưới cho đến khi lực của lò xo $F_s = P_1 x A$. Lúc đó van đóng hoàn toàn.

- *Mạch đường ống:*

Mạch đường ống dẫn khí nén là thiết bị truyền dẫn khí nén từ máy nén khí đến bình trích chứa rồi đến các hệ thống trong bộ điều khiển và cơ cấu chấp hành.

Mạch đường ống dẫn khí nén có thể chia thành hai loại:

- Mạch đường ống được lắp ráp cố định(mạng đường ống trong nhà máy).

- Mạng đường ống được lắp ráp di động (mạng đường ống trong dây truyền hoặc trong máy móc thiết bị).

Thông số cơ bản cho mạng đường ống lắp ráp cố định là ngoài lưu lượng khí nén còn vận tốc dòng chảy, tổn thất áp suất trong ống dẫn khí nén, áp suất yêu cầu, chiều dài ống dẫn và các phụ tùng nối ống.

- Lưu lượng: Phụ thuộc vào vận tốc dòng chảy. Vận tốc dòng chảy trong ống càng lớn thì tổn thất áp suất càng cao.

- Vận tốc dòng chảy: Được chọn theo tài liệu và thường nằm trong khoảng 6-10 m/s. Vận tốc dòng chảy khi qua các đoạn nối ống sẽ tăng lên nhất thời khi dây truyền máy móc đạng vận hành.

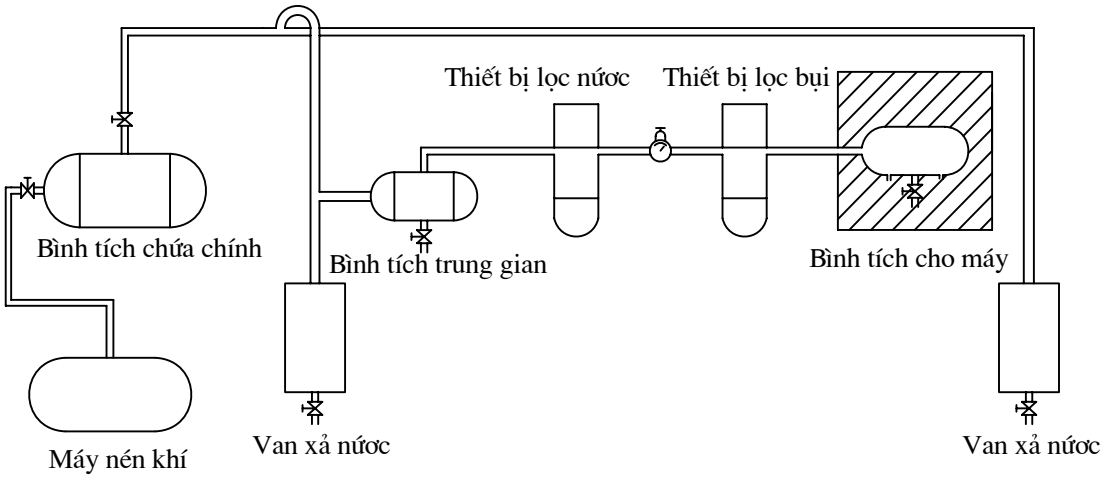
- Tổn thất áp suất: Thông thường yêu cầu tổn thất áp suất là 0,1 bar trong ống dẫn chính. Tuy nhiên trong thực tế, sai số tính đến bằng 5% áp suất yêu cầu. Ví dụ: áp suất yêu cầu của hệ thống là 7 bar thì tổn thất áp suất tính được là 0,35 bar là chấp nhận được. Nếu trong ống dẫn chính có lắp các phụ tùng nối ống, các van thì tổn thất áp suất của hệ thống sẽ tăng lên.

Mạng đường ống lắp ráp di động:

Mạng đường ống lắp ráp di động đa dạng hơn mạng đường ống lắp ráp cố định. Ngoài những đường ống bằng kim loại có thành ống mỏng người ta còn sử dụng các loại ống dẫn khác bằng nhựa, vật liệu tổng hợp các ống bằng cao su, các ống mềm bằng vật liệu tổng hợp. Đường kính ống dẫn phải tương đương với đường kính các mối nối của các phần tử điều khiển.

Ngoài mối nối bằng ren, mạng đường ống lắp ráp di động còn sử dụng các mối nối cắm với các đầu kẹp tuỳ theo áp suất yêu cầu của khí nén cho từng loại máy mà chọn những loại ống dẫn có những tiêu chuẩn kỹ thuật khác nhau.

- ***Sơ đồ lắp ráp hệ thống cung cấp khí nén:***



Hình 3-21 : Hệ thống cung cấp khí nén

a. Hệ thống điều khiển bằng khí nén

Hệ thống điều khiển bằng khí nén được mô tả dưới đây gồm các cụm và phần tử chính có chức năng sau:

- Cơ cấu tạo năng lượng: Máy nén khí, bình tích chứa, bộ lọc...

- Phần tử nhận tín hiệu: Các loại nút bâm...

- Phần tử xử lý: van áp suất, van điều khiển từ xa...

- Phần tử điều khiển: Van đảo chiều.

- Cơ cấu chấp hành: xilanh, động cơ khí nén

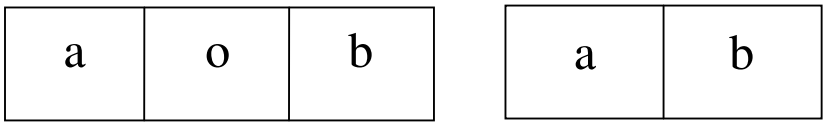
Năng lượng điều khiển có thể bằng khí nén hoặc bằng điện.

Các phần tử chính của hệ thống điều khiển bằng khí nén:

- *Van đảo chiều*

Van đảo chiều có nhiệm vụ điều khiển dòng năng lượng bằng cách đóng mở hay thay đổi vị trí các cửa van để thay đổi các hướng của dòng khí nén.

1) Ký hiệu của van đảo chiều: Vị trí của van được ký hiệu bằng các ô vuông liền nhau với các chữ cái o, a, b, c, ... hay các chữ số 0, 1, 2, 3...



Hình 3 - 22: Ký hiệu van đảo chiều

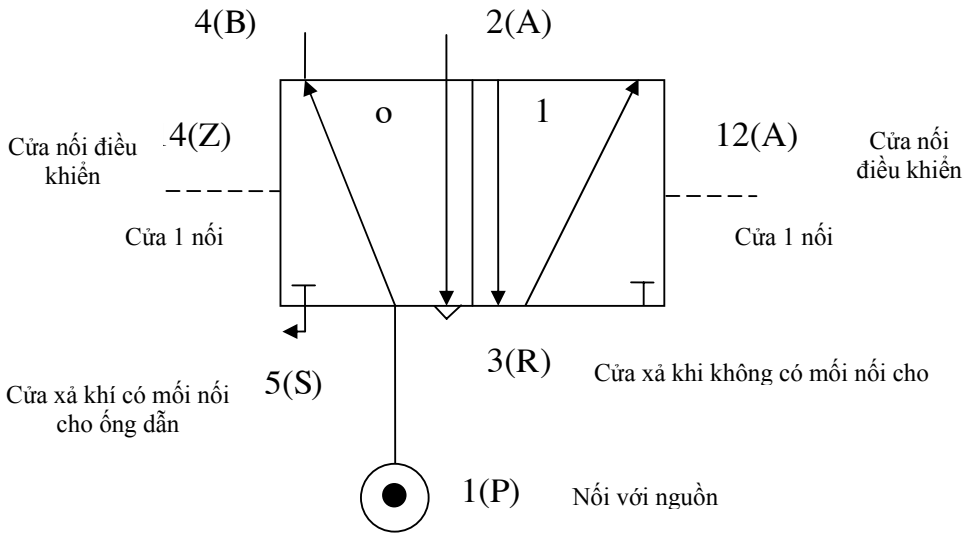
Vị trí không là vị trí mà khi van chưa có tác động của tín hiệu bên ngoài vào. Đối với van 3 vị trí , thì vị trí ở giữa, ký hiệu '0' là vị trí ' không'. Đối với van 2 vị trí thì vị trí 'không' có thể là a hoặc b. Thông thường vị trí bên phải b là vị trí ' không'.

Cửa nối van được ký hiệu như sau:	ISO 5599	ISO 1219
Cửa nối nguồn (từ bộ lọc khí)	1	P
Cửa nối làm việc	2, 4, 6, ...	A, B, C, ...
Cửa xả khí	3, 5, 7, ...	R, S, T, ...
Cửa nối tín hiệu điều khiển	12, 14, ...	X, Y, ...

a b

Hì nh 3-23: Ký hiệu cửa xả

Bên trong ô vuông của mỗi vị trí là các đương mũi tên biểu diễn hướng chuyển động của dòng khí nén qua van. Khi dòng bị chặn thì được biểu diễn bằng dấu gạch ngang.

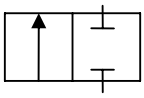
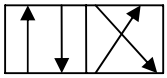
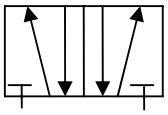


Hình 3-24: Ký hiệu và tên gọi của các cửa van đảo chiều

Hình trên là ký hiệu của van đảo chiều 5/2 (van 5 vị trí 2 cửa).

Cách gọi tên và ký hiệu của một số van đảo chiều:

Tên thiết bị	Ký hiệu

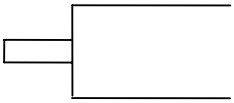
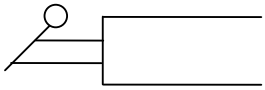
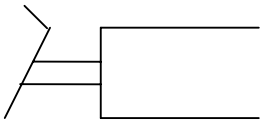
Van đảo chiều 2/2	
Van đảo chiều 4/2	
Van đảo chiều 5/2	

3.3.2.2 Tín hiệu tác độnglên van đảo chiều

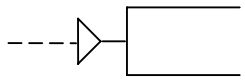
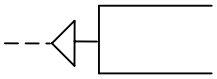
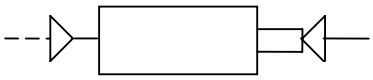
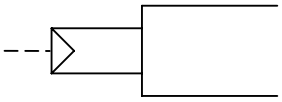
Tín hiệu tác động vào van đảo chiều có 4 loại là: Tác động bằng tay, tác động băng cơ học, tác động bằng khí nén và tác động bằng nam châm điện.

Tín hiệu tác động từ 2 phía (đối với van đảo chiều không có vị trí 'không'. Hay chỉ từ một phía (đối với van đảo chiều có vị trí ' không').

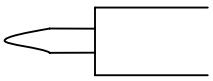
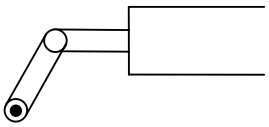
Tác động bằng tay:

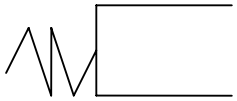
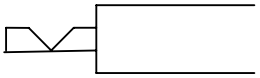
Tên thiết bị	Ký hiệu
Ký hiệu nút nhấn tổng quát	
Nút bấm	
Tay gạt	
Bàn đạp	

Tác động bằng khí tác động nén:

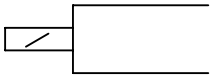
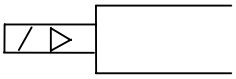
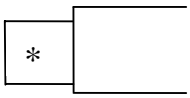
Tên thiết bị	Ký hiệu
Trực tiếp bằng dòng khí nén vào	
Trực tiếp bằng dòng khí nén ra	
Trực tiếp bằng dòng khí nén vào với đường kính 2 đầu nòng van khác nhau.	
Gián tiếp bằng dòng khí nén ra qua van phụ trợ	

Tác động bằng cơ học:

Tên thiết bị	Ký hiệu
Đầu dò	
Cữ chặn bằng con lăn tác động một chiều.	

Lò xo	
Nút nhấn có rãnh định vị	

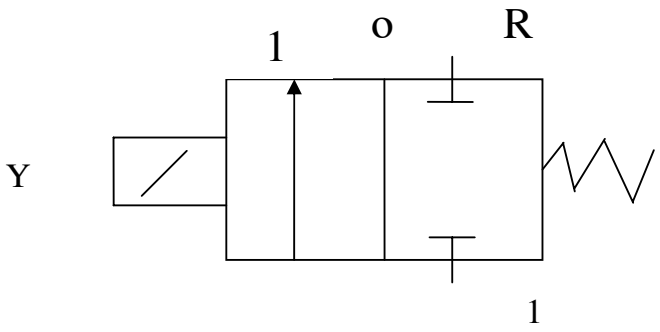
Tác động bằng nam châm điện:

Tên thiết bị	Ký hiệu
Trực tiếp	
Bằng nam châm điện va van phụ trợ	
Tác động theo cách hướng dẫn cụ thể	

3.3.2.3 Van đảo chiều có vị trí 'không':

Van đảo chiều có vị trí ' không' là loại van tác động bằng cơ học và ký hiệu lò xo nằm ngay vị trí bên cạnh ô vuông phía bên phải của ký hiệu van. Tác dụng lên phía đối diện nòng van là tín hiệu bằng cơ, khí nén hay bằng điện. Khi chưa có tín hiệu tác động, vị trí của các cửa nối được biểu diễn trong ô vuông phía bên phải đối với van đảo chiều hai vị trí. Còn đối với van đảo chiều 3 vị trí thì vị trí ' không' nằm ở giữa.

Ví dụ : van đảo chiều 2/2 tác động bằng nam châm điện:



Hình 3-25: Van đảo chiều có vị trí 'không'

Van có hai cửa P và R, hai vị trí 0 và 1. Tại vị trí 0, cửa P và R bị chặn. Khi cuộn Y có điện, từ vị trí 0 van chuyển sang vị trí 1, cửa P nối với R. Khi cuộn hút Y không có điện, do tác động của lò xo phía đối diện, van sẽ quay trở về vị trí ban đầu.

3.3.2.4 Van đảo chiều không có vị trí 'không':

Khi không có tín hiệu tác động lên đầu dòng nữa, thì vị trí 0 của van vẫn được giữ nguyên để đợi tín hiệu tác động từ phía nòng van đối diện. Vị trí tác động ký hiệu là a, b, c, ...

Tín hiệu tác động có thể là:

- Tác động bằng tay hay bàn đạp.

- Tác động bằng dòng khí nén điều khiển đi vào hay đi ra từ hai phía nòng van.

- Tác động trực tiếp bằng điện từ hay gián tiếp bằng dòng khí nén đi qua van phụ trợ.

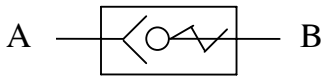
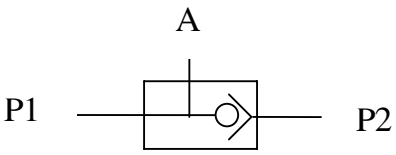
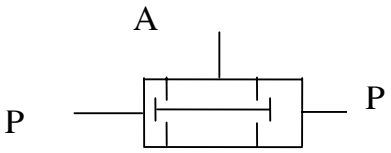
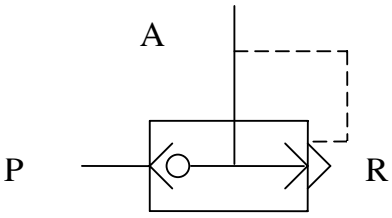
- *Van chắn*

Van chắn là loại van chỉ cho dòng khí nén đi qua một chiều, chiều còn lại bị chặn. Van chắn gồm có các loại sau:

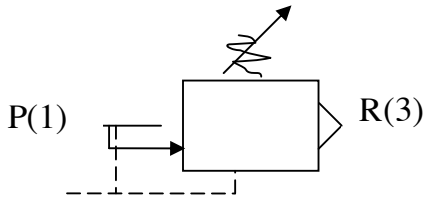
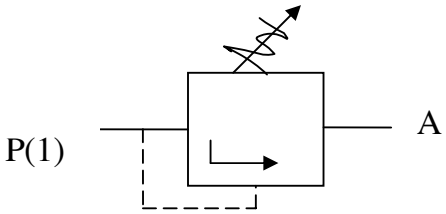
- Van một chiều

- Van logic(OR, AND).

- Van xả khí nhanh.

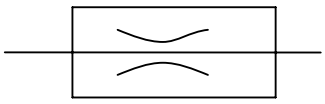
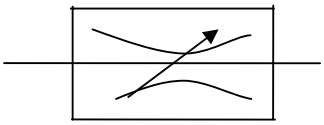
Tên thiết bị	Ký hiệu
Van một chiều có tác dụng chỉ cho dòng khí nén đi qua một chiều, chiều ngược lại bị chặn.	
Van logic OR: Khi có dòng khí nén vào từ P1 thì cửa P2 bị chặn và cửa P1 nối với A. Ngược lại khi dòng khí nén vào P2 thì cửa P1 bị chặn, cửa P2 nối với cửa A.	
Van logic AND: Khi có dòng khí nén vào P1 thì P1 bị chặn, và ngược lại khi có dòng khí nén vào P2 thì P2 bị chặn. Chỉ khi nào cả P1 và P2 có dòng khí nén vào thì mới có khí nén qua cửa A	
Van xả khí nhanh: Khi dòng khí nén vào cửa P, chắn cửa R, cửa P nối với cửa A. Khi dòng khí nén vào từ A, cửa P bị chặn, cửa A nối với cửa R, khí được xả nhanh ra ngoài.	

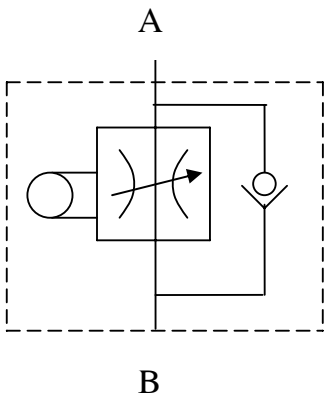
- *Van áp suất*

Tên thiết bị	Ký hiệu
Van an toàn: Bình thường khí áp suất nhỏ hơn hay băng áp suất cho phép, cửa R bị chặn. Khi áp suất lớn hơn cho phép cửa R mở ra khí nén từ P theo R ra ngoài.	
Van tràn: Khi áp suất băng hoặc lớn hơn cho phép thì cửa P nối với cửa A.	

- *Van tiết lưu*

Van tiết lưu có nhiệm vụ thay đổi lưu lượng dòng khí nén, có nghĩa là thay đổi vận tốc của cơ cấu chấp hành.

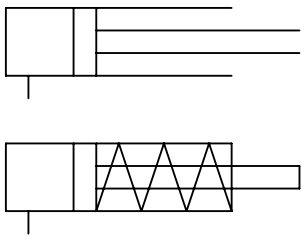
Tên thiết bị	Ký hiệu
Van tiết lưu có tiết diện không đổi: Khe hở của van có tiết diện không đổi, do đó lưu lượng dòng khí không thay đổi.	
Van tiết lưu có tiết diện thay đổi: Lưu lượng dòng khí qua van có thể thay đổi nhờ một vít chỉnh khe hở tiết lưu.	
Van tiết lưu một chiều điều chỉnh bằng tay:Nguyên lý làm	

việc giống như van tiết lưu điều chỉnh bằng tay tuy nhiên dòng khí chỉ bị tiết lưu một chiều từ A qua B.	
Van tiết lưu một chiều điều chỉnh bằng cữ chặn: Dòng khí có thể bị tiết lưu một chiều từ A qua B, tuỳ vào vị trí của cữ chặn mà tiết diện khe hở của van thay đổi làm cho lưu lượng dòng chảy thay đổi.	

3.3.3 Cơ cấu chấp hành

Cơ cấu chấp hành có nhiệm vụ biến đổi năng lượng khí nén thành năng lượng cơ học. Cơ cấu chấp hành có thể thực hiện chuyển động thẳng (xilanh) hoặc chuyển động quay (động cơ khí nén).

- *Xilanh*

Tên thiết bị	Ký hiệu
Xilanh tác dụng đơn: áp lực khí nén chỉ vào một phía, phí còn lại do ngoại lực tác dụng hoặc lò xo.	

Xilanh tác dụng hai chiều: áp suất khí nén được đưa vào cả hại chiều.	

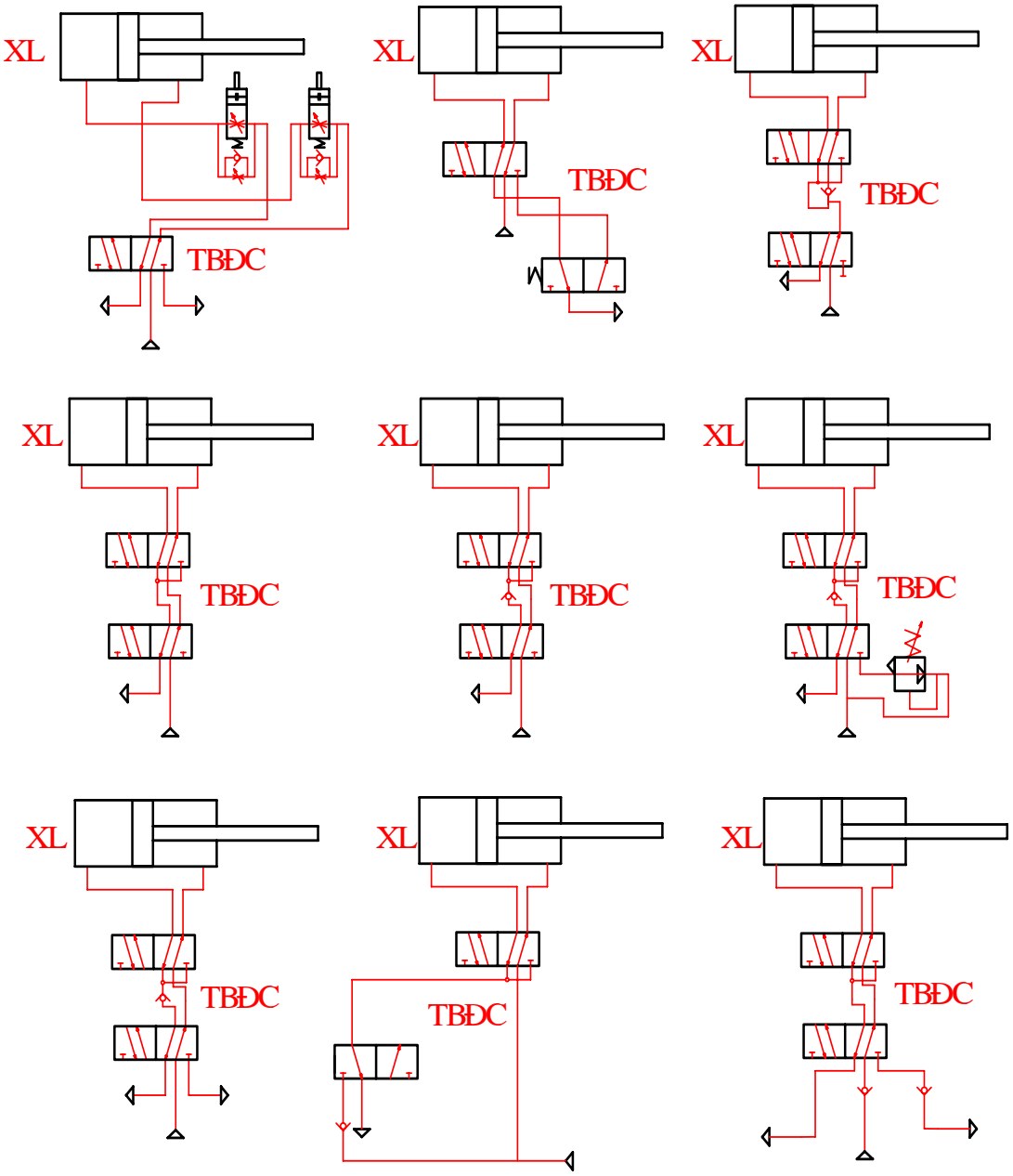
- *Động cơ khí nén*(được trình bày trong phần động cơ).

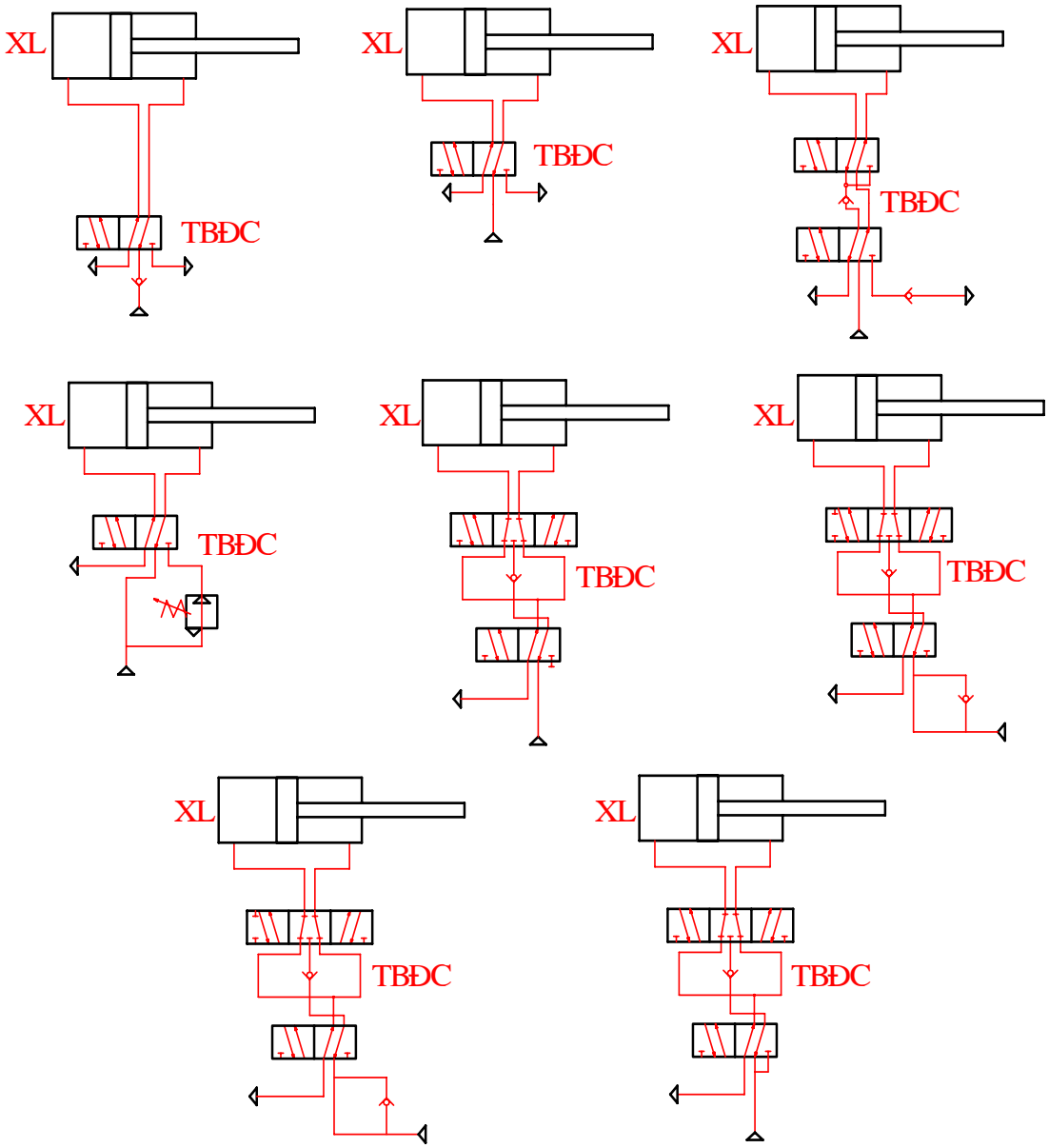
3.4 Các mạch điều khiển khí né trong robot công nghiệp

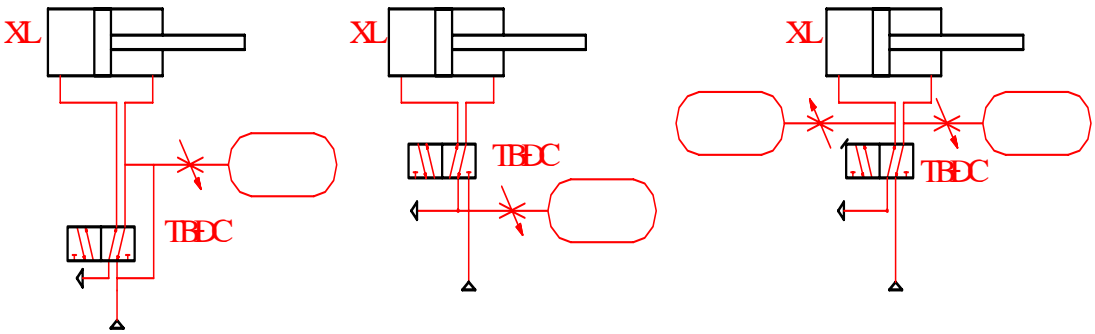
Đối với người máy công nghiệp làm việc trong môi trường nguy hiểm như dễ cháy, dễ nổ ..., việc dùng khí nén trong hệ truyền dẫn của chúng là rất thích hợp. Tuy nhiên dùng hệ truyền dẫn khí nén cũgn không ít nhược điểm. Trước hết là do đặc điểm nén được của chất khí, cho nên chuyển động do chúng được thực hiện thường kèm theo dao động không chính xác lúc dừng, nhất là hoặc các vị trí trung gian. Ngoài ra, còn cần có các biện pháp phun dầu bôi trơn, lọc bụi, lọc ẩm và giảm ồn.

Chính vì những lý do đó, hiện nay các hệ truyền động tự động khí nén thường dùng chủ yếu trong các người máy công nghiệp có điều khiển theo chu kỳ, thực hiện các thao tác đơn giản. Các người máy này thường được thiết kế với sức nâng tải 10 kg và được sử dụng trong các nguyên công đơn giản như cấp phôi liệu cho máy rèn dập, máy cắt kim loại.

Trong giới hạn của đồ án này chúng tôi giới thiệu một vài hệ truyền động khí nén thường gặp trong robot công nghiệp:


XL
TBĐC
XL
TBĐC
XL
TBĐC
XL
TBĐC
XL
TBĐC
XL
TBĐC
XL
TBĐC
XL
TBĐC
XL
TBĐC


XL
TBĐC
XL
TBĐC
XL
TBĐC
XL
TBĐC
XL
TBĐC
XL
TBĐC
XL
TBĐC
XL
TBĐC



Hình 3-26: Các mô hình truyền động khí nén thường gặp trong rôbốt công nghiệp

Trong đó:

- XL : Xilanh khí nén.
- TBĐC : Thiết bị đảo chiều dòng khí nén.

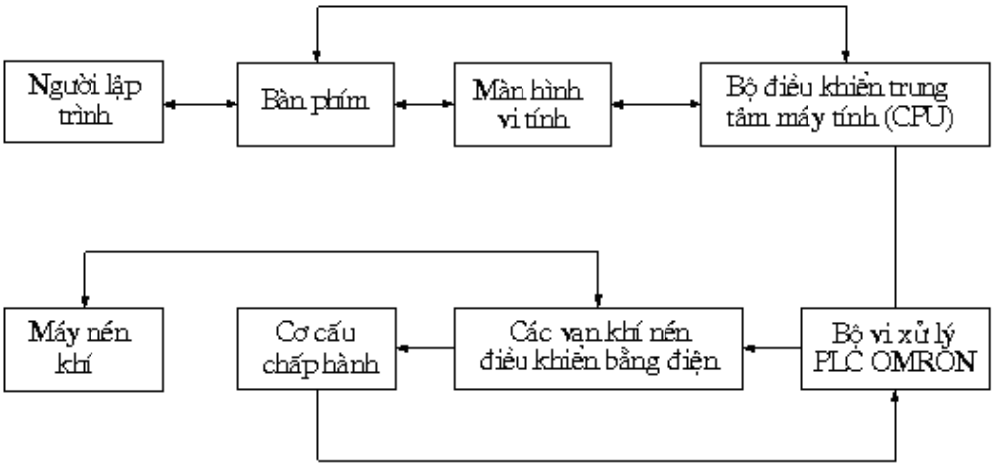
CHƯƠNG 4

PHƯƠNG PHÁP LẬP TRÌNH BẰNG PLC ĐỂ ĐIỀU KHIỂN HỆ THỐNG CHẤP HÀNH TRONG RÔBỐT CÔNG NGHIỆP

4.1 Hệ thống điều khiển thuỷ lực trong rôbốt cấp phôi tự động

4.1.1 Sơ qua về hệ thống lắp đặt

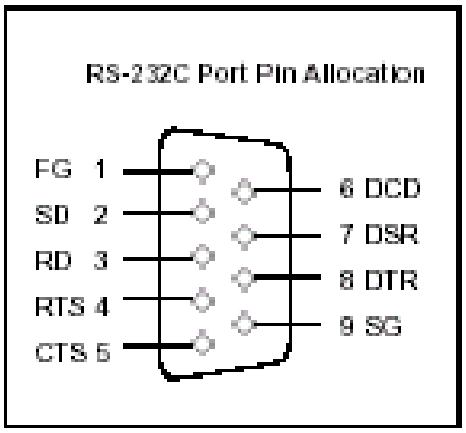
Sơ đồ lắp đặt được trình bày trong sơ đồ khối ở hình sau:



Hình 4-1: Sơ đồ khối hệ thống truyền động-tự động khí nén điều khiển bằng PLC

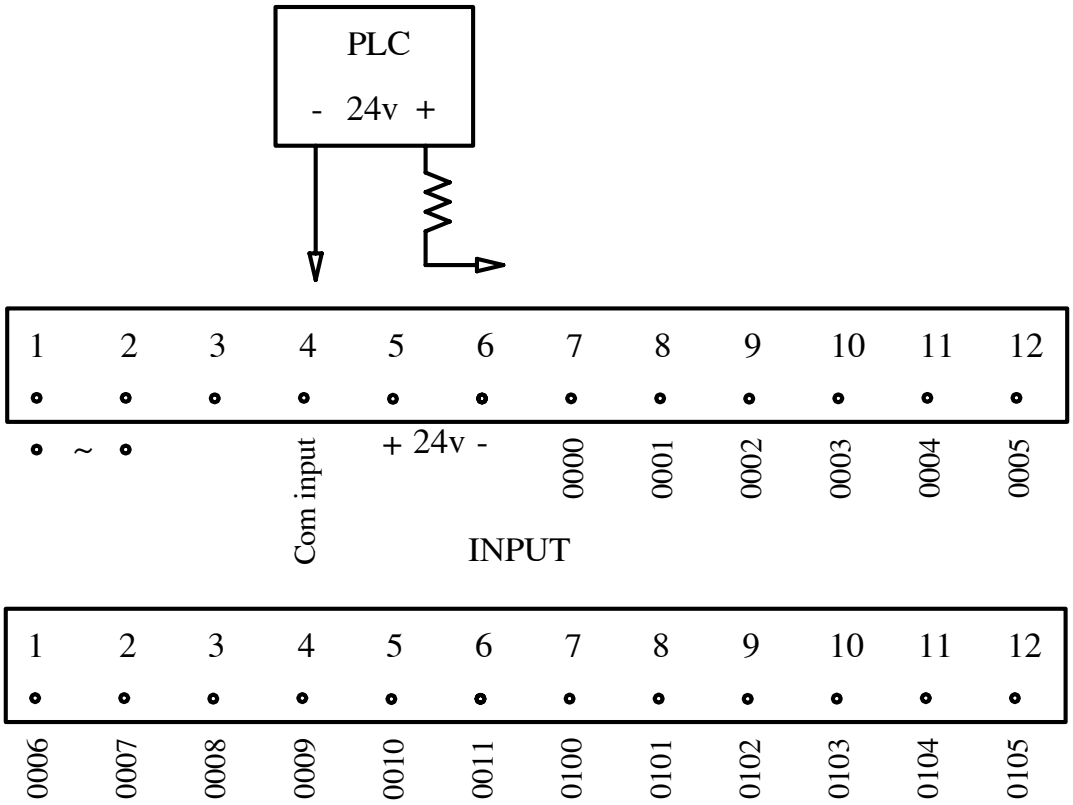
Người lập trình viên sau khi đưa ra các sơ đồ trạng thái gắn các biến cần thiết theo ý đồ và dùng thuật toán lôgíc để lập chương trình cho những bài toán cụ thể. Những bài toán này được biểu diễn dưới dạng sơ đồ hình thang và được ghi vào ổ đĩa của máy tính theo những kí hiệu nhất định để dễ tìm.

Sau khi đã có chương trình chạy ta thực hiện ghép CPU máy tính với bộ vi xử lý. Việc thực hiện ghép này thông qua cổng giao tiếp RS232.

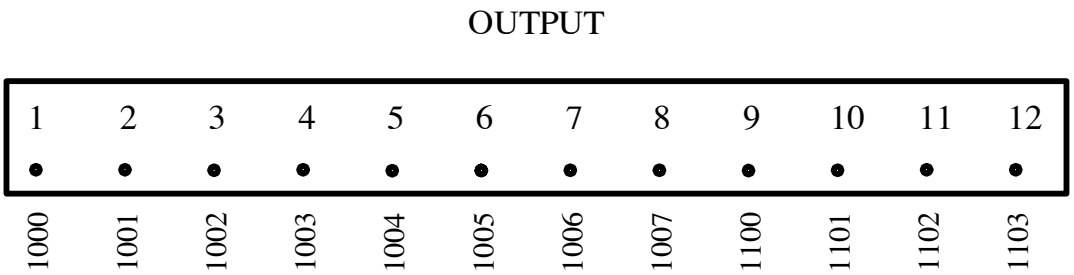


Hình 4.2 Sơ đồ chân cổng giao tiếp máy tính v RS_232C à PLC:

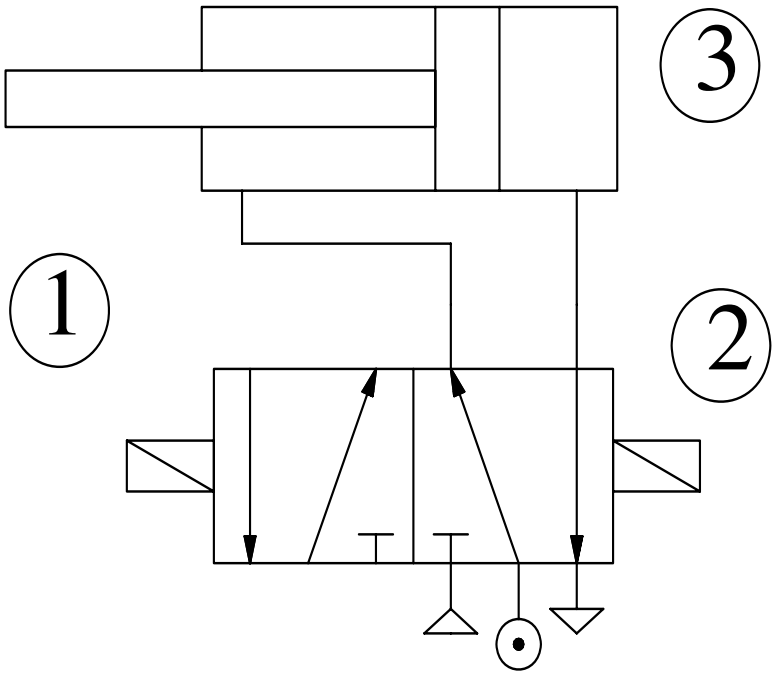
Ta cũng có bảng điều khiển sơ bộ đấu từ PLC với các đầu vào/ra như sau:



Hình 4-3 Sơ đồ ghép nối đầu vào trên bộ PLC OMRON



Hình 4.4: Sơ đồ ghép nối đầu ra trên bộ PLC OMRON



Hình 4-5 Sơ đồ nguyên lý ghép nối giữa van và cơ cấu chấp hành tương ứng

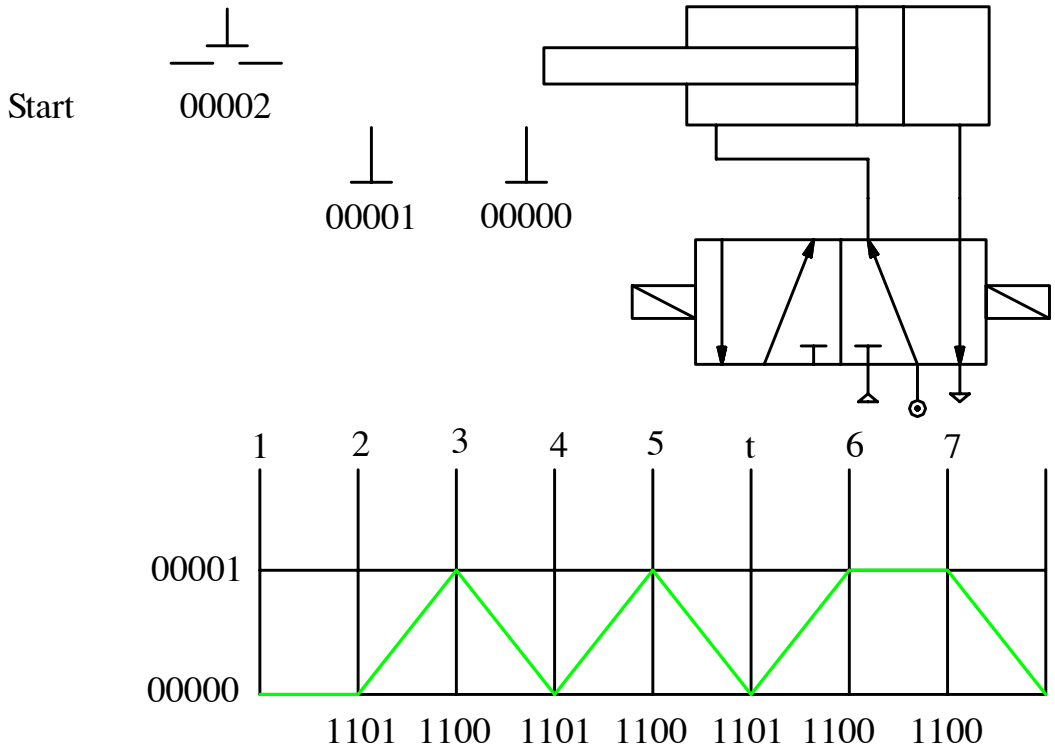
Trên các van có lắp các bộ điều khiển điện (1) & (2) khi có tín hiệu điện tới và phản hồi sẽ đảo chiều dòng khí vào xilanh (3).

Ta có thể thiết kế một hệ thống khí nén trong rôbốt có 5 xilanh . Các xilanh này có thể làm việc riêng lẻ hoặc phối hợp với nhau theo chu trình kín.

Máy nén khí được dẫn khí vào xilanh và các van bằng các đường ống cao su mềm là dung môi vận chuyển thiết bị.

4.2 Dùng bộ PLC OMRON CPM2A điều khiển hệ truyền động tự động một cơ cấu chấp hành

Trước khi dùng PLC điều khiển 1 đối tượng ta phải lập biểu đồ trạng thái hoạt động của cơ cấu chấp hành được chọn.



Hình 4-6 Biểu đồ trạng thái hoạt động của hệ truyền động tự động khí nén có 1 cơ cấu chấp hành

a) Trường hợp điều khiển theo vị trí (không có trễ $\tau = 0$)

Trong trường hợp này do yêu cầu của công nghệ ta có biểu đồ trạng thái như hình 4.5

Ta dùng PLC OMRON SYSMAC CPM2A để điều khiển tự động 1 đối tượng:

Đặt các cấu hình vào ra và lập trình

- PLC đầu vào:

00002 = Start (Công tắc bật)

00000 = Công tắc hành trình dưới

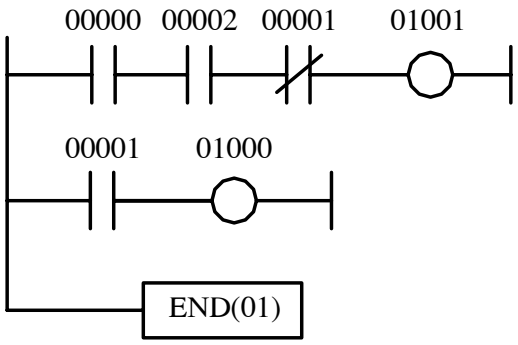
00001 = Công tắc hành trình trên

- PLC đầu ra:

01000 = Tín hiệu đầu ra van điều khiển bằng điện từ (phải)

01001 = Tín hiệu đầu vào van điều khiển bằng điện từ (trái)

Theo biểu đồ trạng thái ra they: Khi nhấn công tắc Start (0002) để vận hành, tín hiệu đầu vào 0000 xuất hiện (công tắc 0000 bị đè) để chạy, khi đó sẽ kích tín hiệu đầu ra 1101 làm việc đảo chiều dòng khí nén làm pittông chạy sang trái đè vào công tắc 0001 (đầu vào) lúc đó tín hiệu được báo về bộ trung tâm PLC kích tín hiệu đầu ra 1100 làm việc và lại đảo chiều về vị trí đầu. Chu trình cứ diễn ra như vậy khép kín. Ta có biểu đồ hình thang và phương trình cứ diễn ra như vậy khép kín. Ta có biểu đồ hình thang và phương pháp liệt kê sau:

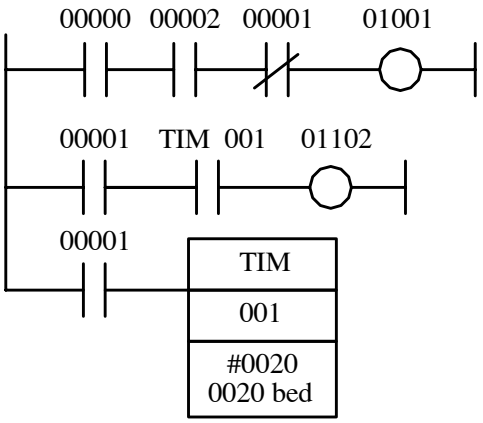


00000	LD	00000
00001	AND	00002
00002	AND NOT	00001
00003	OUT	01001
00004	LD	00001
00005	OUT	01000
00006	END.	

Hình 4-7: Sơ đồ hình thang và phương pháp liệt kê cho một đối tượng tự động điều khiển theo vị trí.

b) Trường hợp có trễ ($\tau \neq 0$)

Hoàn toàn lý luận tương tự trên biểu đồ trạng thái tín hiệu đầu ra 01100 bị trễ thời gian τ



00000	LD	00000
00001	AND	00002
00002	AND NOT	00001
00003	OUT	01001
00004	LD	00001
00005	AND	TIM 001
00006	OUT	01000
00006	LD	00001
00007	TIM	001 #0020
00008	END.	

Hình 4-8 Biểu đồ hình thang và phương pháp liệt kê cho 1 đối tượng tự động điều khiển theo thời gian

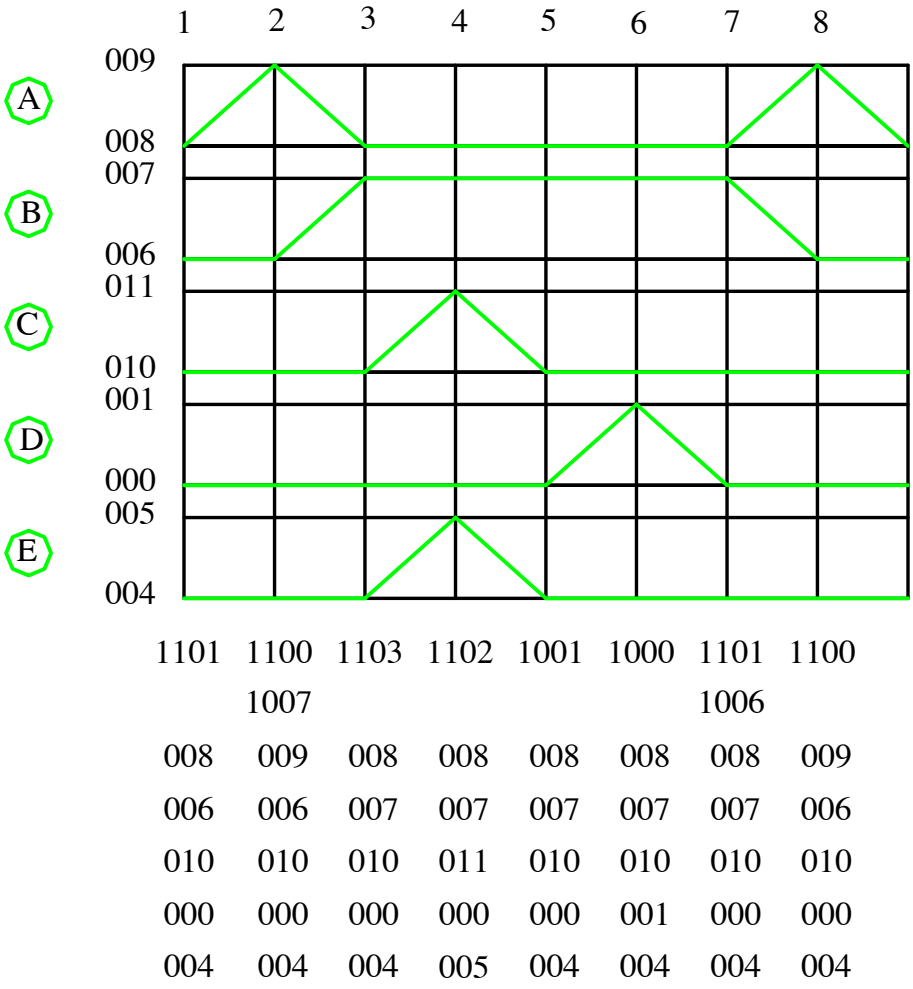
Dùng bộ PLC OMRON SYSMAC CPM2A điều khiển hệ truyền động tự động nhiều cơ cấu chấp hành

Tuỳ thuộc vào bộ PLV có bao nhiêu đầu vào và ra mà ta quyết định được số đối tượng nhiều nhất có thể điều khiển được. Rõ ràng nếu bộ PLC càng nhiều đầu vào và ra thì sẽ điều khiển được nhiều đối tượng.

Trong trường hợp này: Bộ PLC OMRON SYSMAC CPM2A có 12 đầu ra dùng 1 đầu ra tương ứng với đầu vào 00002 làm công tắc Start và một đầu ra dự phòng. Vậy còn 10 đầu ra tương ứng với 10 tín hiệu đóng mở của van điện từ. Vậy ta có khả năng điều khiển tự động được nhiều đối tượng: ở đây ta điều khiển 5 cơ cấu chấp hành ứng với 5 xilanh khí.

Ví dụ dùng bộ PLC OMRON SYSMAC CPM2A điều khiển hệ truyền động tự động 5 xilanh chấp hành có $\tau = 0$ và $\tau \neq 0$.

a) Điều khiển 5 xilanh khí nén tự động ứng với $\tau = 0$.



Hình 4-9 Biểu đồ trạng thái của hệ truyền động tự động khí nén có 5 xilanh chấp hành được điều khiển bằng bộ PLC OMRON SYSMAC CPM2A

Do yêu cầu công nghệ ta có biểu đồ trạng thái của 5 xilanh khí làm việc theo chu trình trên hình 4.8.

Ta phải đặt cấu hình vào ra dựa trên biểu đồ trạng thái:

- PLC đầu vào:

002 = Start (Công tắc bật hệ thống)

$\begin{Bmatrix}008\\009\end{Bmatrix}$ = Công tắc hành trình dưới và trên cho xilanh A

$\begin{Bmatrix}006\\007\end{Bmatrix}$ = Công tắc hành trình dưới và trên cho xilanh B

$\begin{Bmatrix}010\\011\end{Bmatrix}$ = Công tắc hành trình dưới và trên cho xilanh C

$\begin{Bmatrix}000\\001\end{Bmatrix}$ = Công tắc hành trình dưới và trên cho xilanh D

$\begin{Bmatrix}004\\005\end{Bmatrix}$ = Công tắc hành trình dưới và trên cho xilanh E

Vậy ta sử dụng 11 đầu vào để tham gia vào quá trình lập trình

- PLC đầu ra:

1101,1100 ứng với hành trình trái, phải của xilanh A

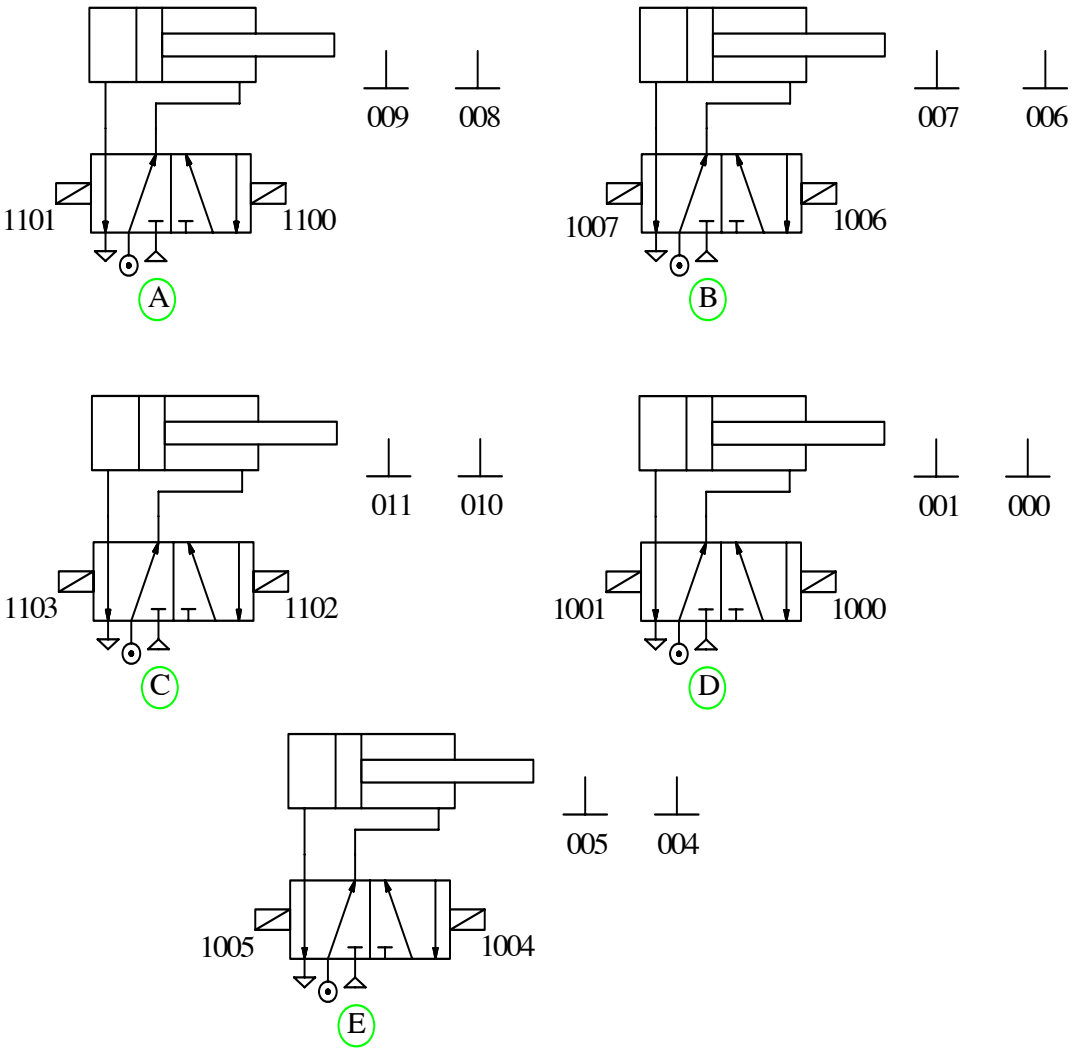
1007,1006 ứng với hành trình trái, phải của xilanh B

1103,1102 ứng với hành trình trái, phải của xilanh C

1001,1000 ứng với hành trình trái, phải của xilanh D

1005,1004 ứng với hành trình trái, phải của xilanh E

Sơ đồ nguyên lý làm việc 5 xilanh dựa trên biểu đồ trạng thái được mô tả trên *hình 4.9*



Hình 4-10 Sơ đồ nguyên lý của 5 xilanh khí sau khi gắn địa chỉ đầu vào và đầu ra.

Lý luận một cách tương tự như trường hợp 1 xilanh ta sẽ viết được biểu đồ hình thang và biểu đồ liệt kê của 5 xilanh làm việc tự động theo chu trình kín.

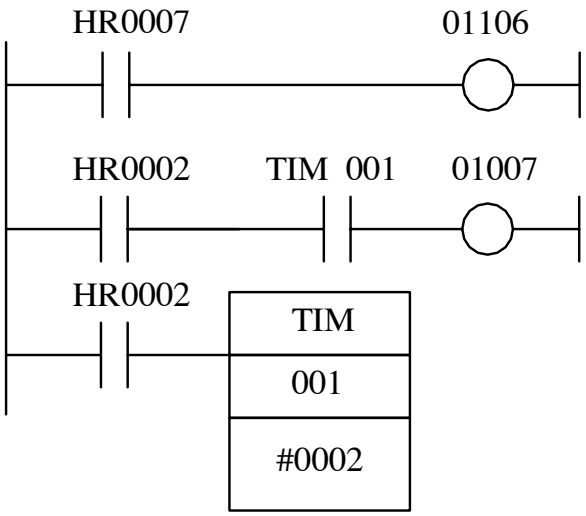
b) Trường hợp $\tau \neq 0$

Trễ trong PLC: Bộ phận trễ trong PLC được nối với 1 máy phát điện. Nhịp đập điện tử bên trong do 1 mạch vi xử lý của PLC điều khiển. Một PLC có thể xử lý ở các nhịp đập dài, ngắn khác nhau. Trên một số sản phẩm của

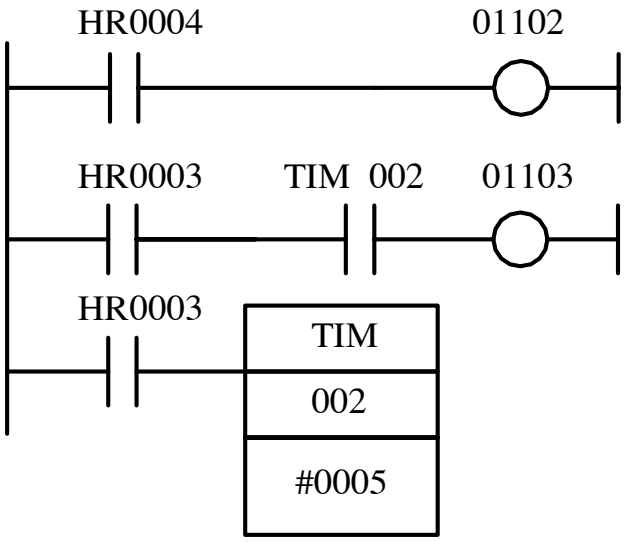
PLC người lập trình có thể quyết định và chọn cơ sở thời gian. Tuy nhiên thời gian OMRON được lập trình với cơ sở thời gian tỷ lệ nhịp đập 0,1 giây. Cơ sở thời gian này do qui trình công nghệ đặt ra.

Ví dụ:

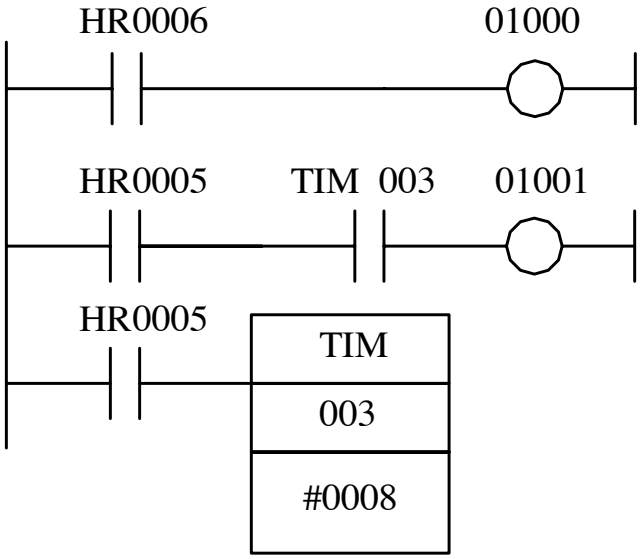
- Cho xilanh (B) trễ 0,2 giây.



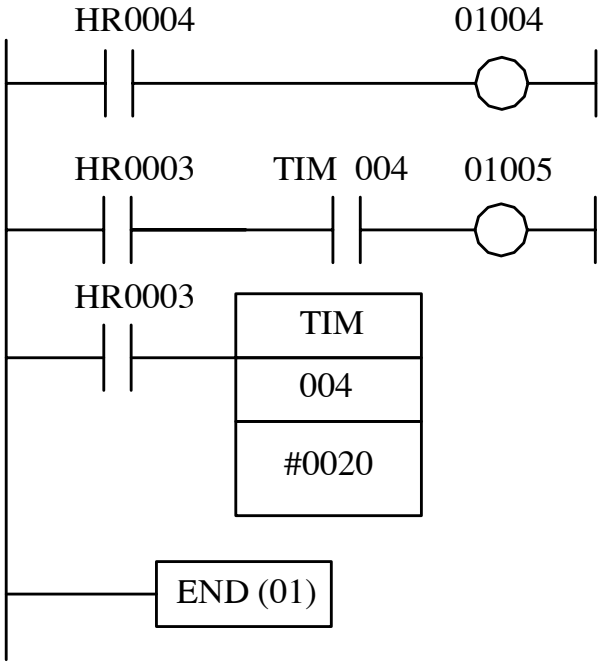
- Cho xilanh (C) trễ 0,5 giây.



- Cho xilanh (D) trễ 0,8 giây.



- Cho xilanh (E) trễ 2 giây.



4.3 Phương pháp điều khiển động cơ sử dụng PLC LG K7M-DR30S

4.3.1 Chương trình điều khiển vị trí động cơ 1 chiều:

Chế độ chạy tay chỉ thực hiện khi chuyển mạch ở vị trí off

Chế độ chạy tự động chỉ thực hiện khi chuyển mạch ở vị trí on

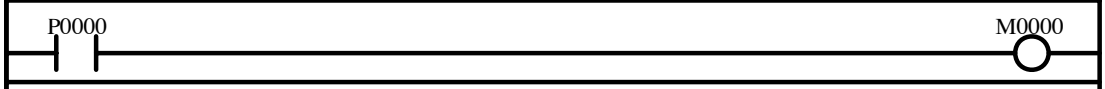
Thay đổi thời gian dừng ở các vị trí bằng việc thay đổi các Timer

Timer T100 thời gian dừng ở vị trí đầu tiên_vị trí 1 ứng với sensor 1

Timer T102 thời gian dừng ở vị trí giữa_vị trí 2 ứng với sensor 2

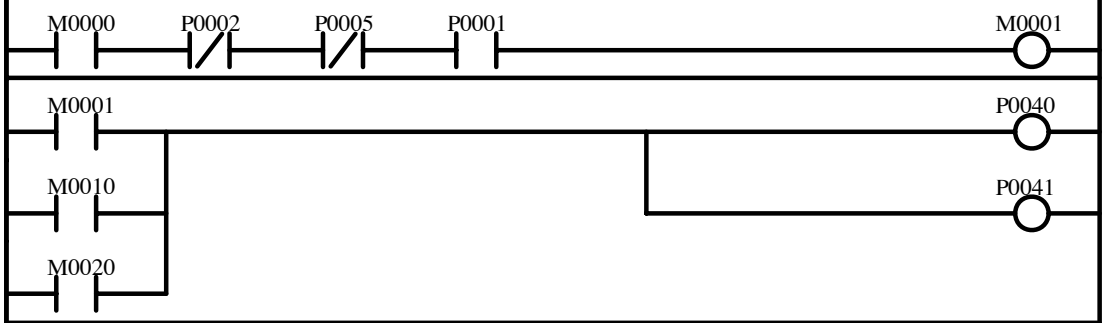
Timer T101 thời gian dừng ở vị trí cuối_vị trí 3 ứng với sensor 3

Chọn chế độ chạy tự động hoặc chạy bằng tay

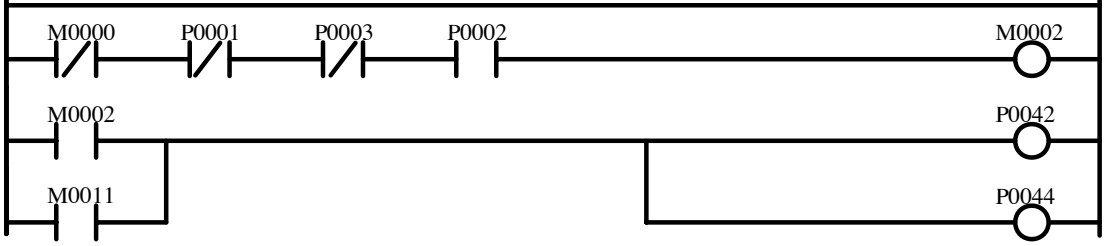


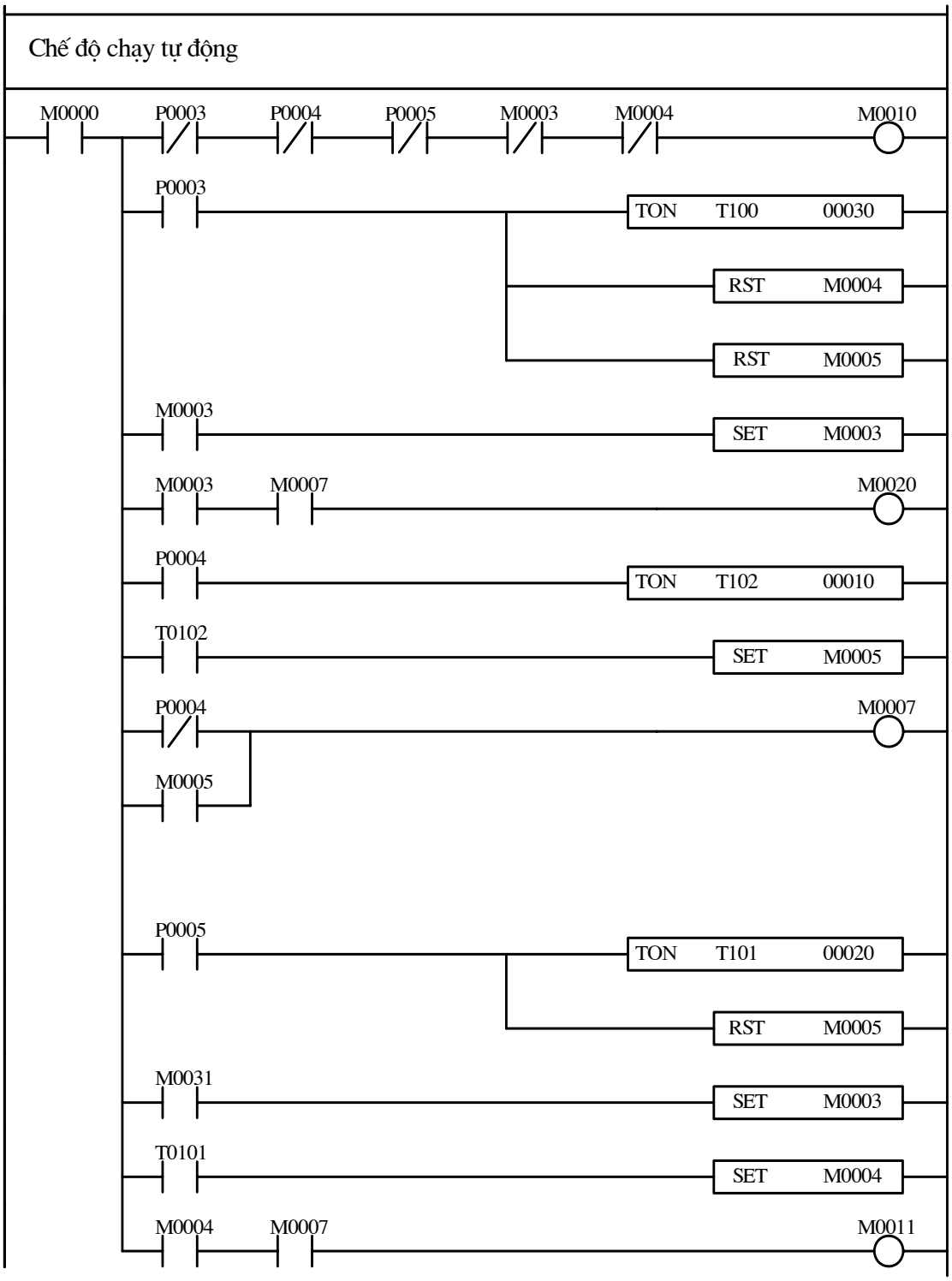
Chế độ chạy bằng tay

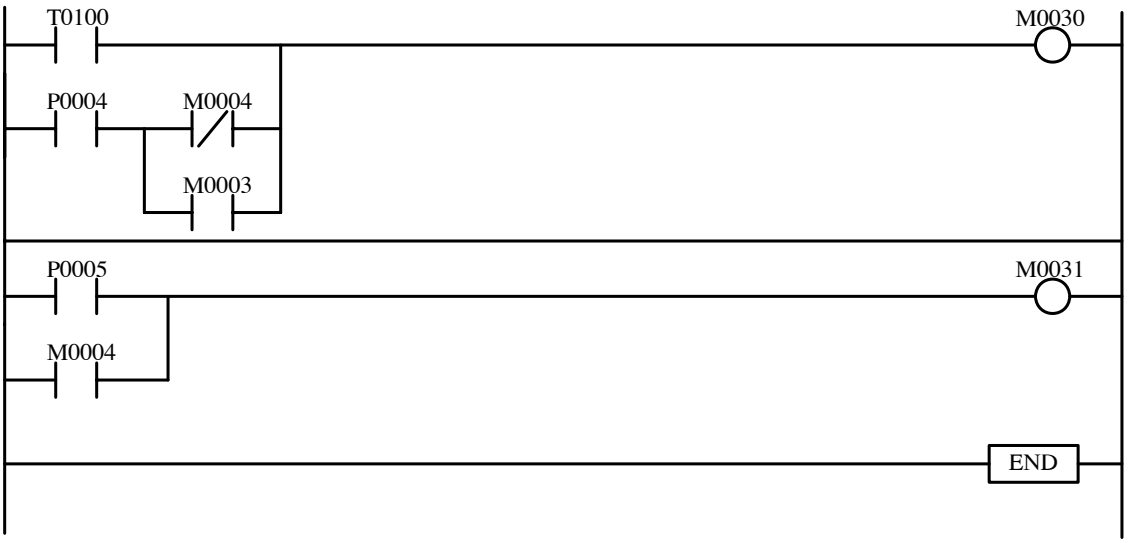
Chạy tiến



Chạy lùi




Chế độ chạy tự động
M0000
P0003
P0004
P0005
M0003
M0004
M0010
P0003
TON T100 00030
RST M0004
RST M0005
M0003
SET M0003
M0003
M0007
M0020
P0004
TON T102 00010
T0102
SET M0005
P0004
M0007
M0005
P0005
TON T101 00020
RST M0005
M0031
SET M0003
T0101
SET M0004
M0004
M0007
M0011

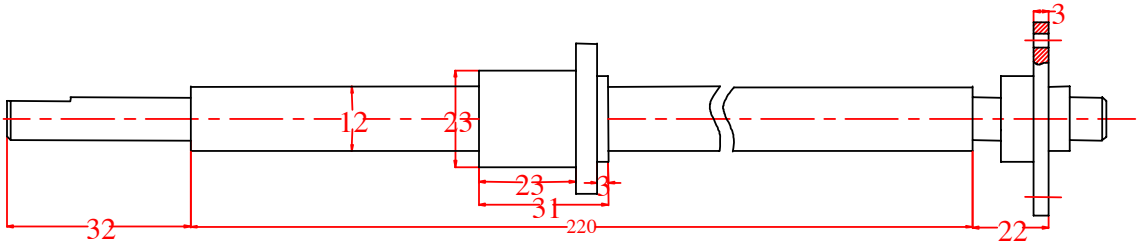


4.3.2 Kết cấu cơ khí trong mô hình mô phỏng về điều khiển hành trình và vị trí bằng vít me-đai ốc.

Nguyên lý chung của hệ thống là dùng vítme - đai ốc để biến chuyển động quay thành chuyển động thẳng, trong đó vítme quay do được nối trục với động cơ một chiều, còn đai ốc được chống xoay nên có thể chuyển động tịnh tiến dọc theo trục của vítme.

Vị trí và hành trình của đai ốc phụ thuộc vào quá trình quay, dừng hoặc quay đảo chiều của vítme.

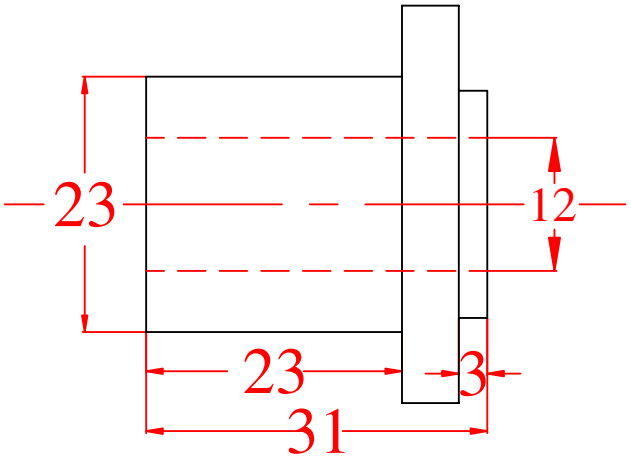
a. Vítme-đai ốc.



H4-11 Truc vít me-đai ốc

Trục vít me có kích thước như hình vẽ, được chế tạo bằng thép hợp kim có bước ren 3mm. Một đầu trục được đỡ bằng ổ lăn đã chế tạo sẵn, đầu kia được đỡ bằng ổ bi đỡ.

Đai ốc được chế tạo bằng thép hợp kim, có kích thước như hình vẽ.

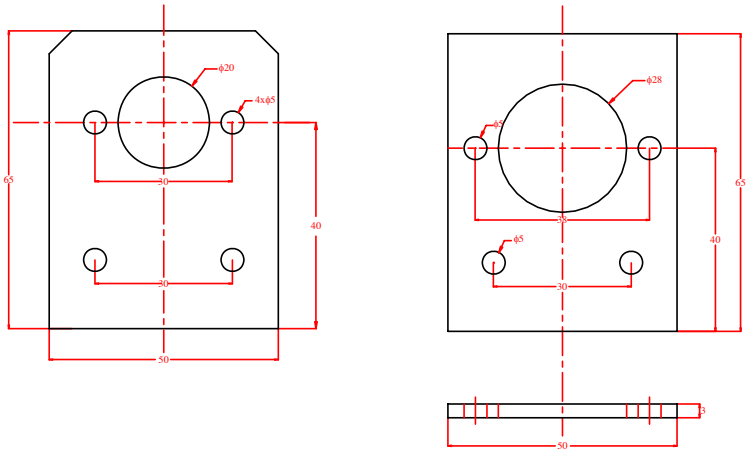


H4-12 Đai ốc

b, Chế tạo các gối đỡ, đế và các chi tiết khác trong mô hình.

+ Hai gối đỡ trục vít me:

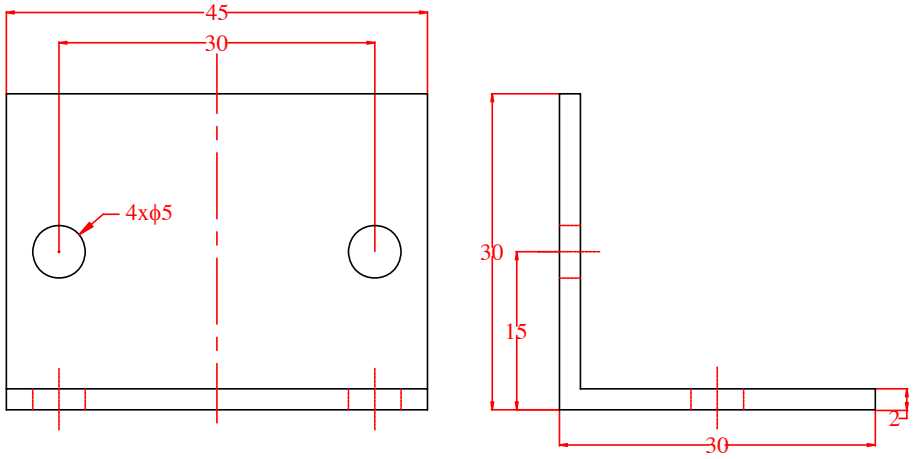
Chế tạo bằng thép tấm C45, dầy 2mm, các lỗ khoan băng mũi khoan ruột gà trên máy khoan cần đạt cấp chính xác 12.



H4-13 Gối đỡ trục vít me

+ Tấm nối chữ L

Được cắt từ thanh cữ L có sẵn bán trên thị trường, 2 đoạn mỗi đoạn dài 40mm, dùng để nối cứng giữa các giá đỡ ổ trục và tấm đế.

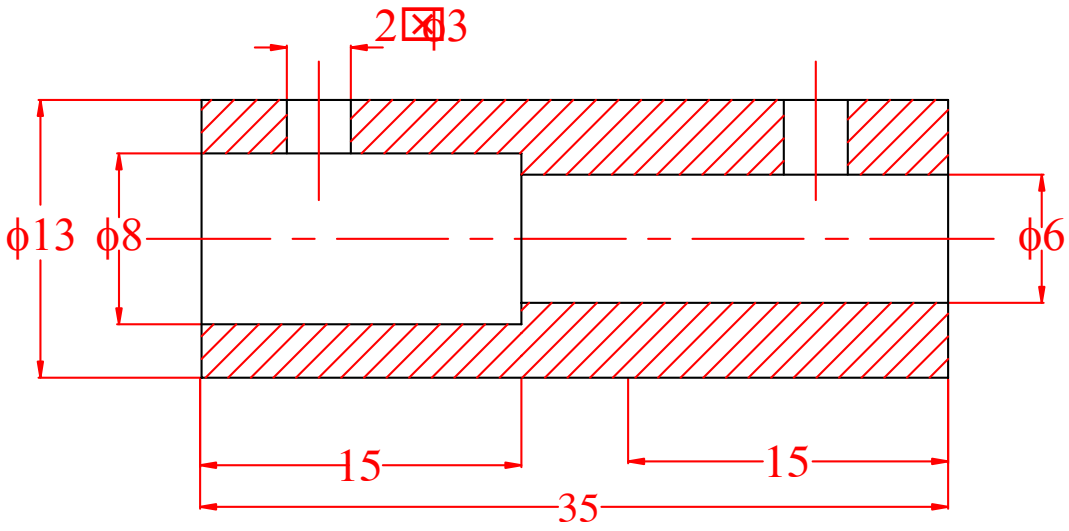


H5-14 Tấm nối chữ L

+ Nối trục

Dùng để nối trục giữa trục động cơ và trục của vít me, nối trục được chế tạo bằng đồng. Do hai đầu trục có đường kính khác nhau (trục động cơ: ϕ6, trục của vít me: ϕ8), nên nối trục được gia công hai lỗ trong một lần gá trên máy tiện. Khoan xuyên tâm lỗ ϕ6 sau đó khoan lỗ ϕ8 dài 15mm.

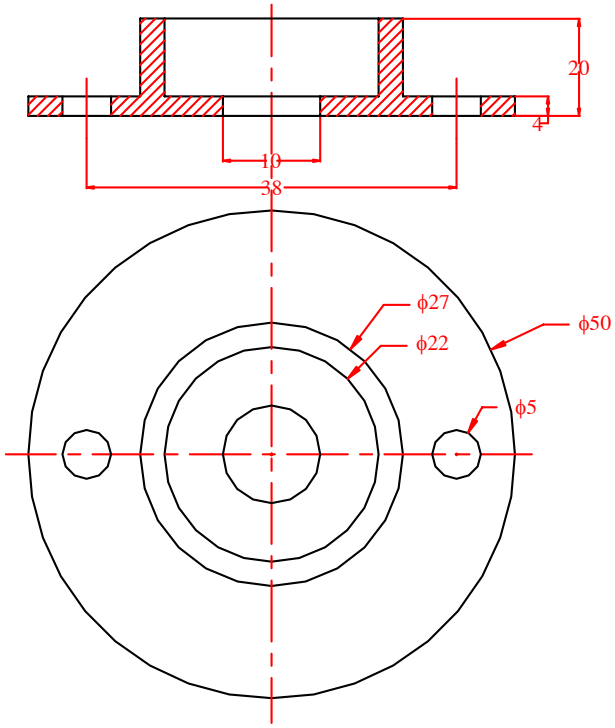
Để chống xoay tương đối giữa hai trục, trên nối trục có khoan và tarô ren hai lỗ ϕ3 vuông góc với tâm trục, khi nối trục ta chỉ cần xiết hai con vít ở hai đầu để đạt được yêu cầu của khớp nối.



H5-15 Nối trục

+ *Gối nắp ổ bi.*

Được dùng để nắp ổ bi(Vòng trong ϕ6, vòng ngoài ϕ22) , chi tiết này được chế tao bằng thép C45, mặt ngoài phẳng có khoan hai lỗ ϕ5 để nắp trên giá đỡ.

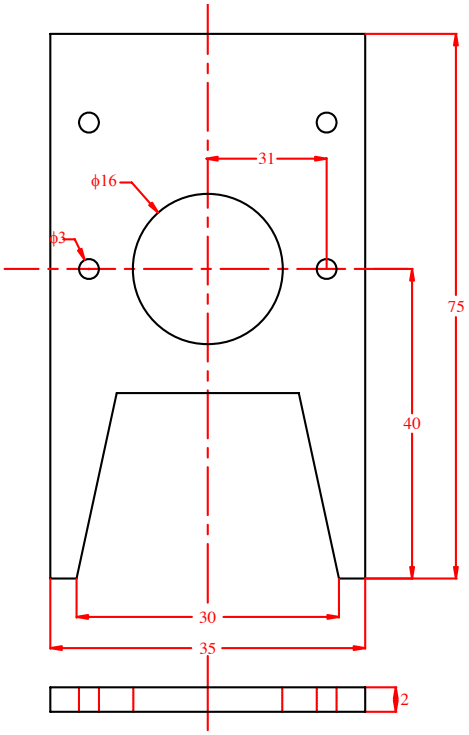


H 5-16 Gối nắp ổ bi

+ Chi tiết chống xoay.

Được dùng để chống xoay đai ốc, trong quá trình làm việc do vítme quay đồng tốc với trục động cơ, muốn biến chuyển động quay của vítme thành chuyển động tịnh tiến của đai ốc ta phải chống xoay cho đai ốc đó.

Chi tiết này được chế tạo bằng đồng, các lỗ trên chi tiết được khoan bằng mũi khoan ruột gà trên máy khoan cần.



H5-17 Chi tiết chống xoay

Nguyên lý hoạt động của mô hình mô phỏng hệ điều khiển tự động trên vítme-đai ốc.

Để điều khiển vị trí và hành trình của đai ốc trên trục vítme ta phải điều khiển được tốc độ, chiều quay và sự đóng mở của động cơ. Khi động cơ quay,

trục vítme cũng quay theo, do được chống xoay nên đai ốc chỉ có thể chuyển động tịnh tiến dọc theo trục vítme. Trên mô hình ta có đặt ba cảm biến để nhận biết vị trí của đai ốc, phụ thuộc vào vị trí của cảm biến và yêu cầu của người lập trình mà động cơ sẽ quay, dừng hay đảo chiều khi đai ốc đi đến vị trí của cảm biến.

KẾT LUẬN

Qua việc nghiên cứu, tổng hợp các vấn đề về robot, các tính chất, cấu tạo và chương trình điều khiển của hệ thống điều khiển tự động, bản đồ án đã đạt được những kết quả sau:

- Nghiên cứu một cách tổng quan về robot: sự phát triển, phân loại,sơ đồ cấu trúc chức năng của robot và những ứng dụng robot trong công nghiệp…

- Xây dựng và phân tích lý thuyết điều khiển robot, giới thiệu một số phương pháp thường sử đụng để điều khiển trong robot công nghiệp.

- Nghiên cứu các phần tử truyền động, cơ cấu chấp hành trong robot cấp phôi tự động, tìm hiểu tính chất các phần tử khí nén, các loại động cơ điện và ứng dụng của chúng, từ đó dưa ra phương án thích hợp trong từng trường hợp cụ thể.

- Giới thiệu các bộ PLC về tính chất và chức năng của chúng cụ thể là hai loại PLC của hai hãng nổi tiếng: OMRON và LG

- Xây đựng thành công một mô hình mô phỏng về điều khiển tự động, trong đó có thể tự lập trình trên bộ PLC PLC *LG K7M-DR30S* để điều khiển hành trình và vị trí cho vítme - đai ốc có sử dụng các cảm biến vị trí.

Với kết quả thu được trong khuôn một đồ án tôt nghiệp, ta có thể lắp đặt các phần tử khí nén, các động cơ và điều khển đươc chúng một cách tự động và có khả năng thay đổi chương trình thông qua việc lập trình trên bộ PLC *LG K7M-DR30S*.

Hoàn thiện một mô hình mô phỏng về điều khiển tự động một động cơ một chiều dùng PLC LG K7M-DR30S , các thiết bị và chương trình đã chạy đúng theo yêu cầu thiết kế một cách ổn định và chính xác.

Nếu có điều kiện, nhóm tác giả sẽ tiếp tục nghiên cứu, phát triển và giải quyết các vấn đề còn thiếu trong bản đồ án này.

TÀI LIỆU THAM KHẢO

[1] Máy điều khiển theo chương trình số & robot công nghiệp.

Tạ Duy Liêm

[2] Robot công nghiệp.

Nguyễn Thiện Phúc

[3] Điều khiển bằng khí nén trong tự động hoá & và kỹ thuật.

Nguyễn Thành Trí

[4] Kỹ thuật ROBOT

Đào Văn Hiệp

[5] Tay máy-Người máy công nghiệp

Nguyễn Thiện Phúc

ĐẠI HỌC THÁI NGUYÊN
KHOA CÔNG NGHỆ THÔNG TIN

NGUYỄN THỊ THU THUỶ

MỘT SỐ PHƯƠNG PHÁP CHÍNH XÁC LẬP LỘ TRÌNH CHUYỂN ĐỘNG CHO ROBOT

Chuyên ngành: Khoa học máy tính

Mã số: 60.48.01

LUẬN VĂN THẠC SĨ CÔNG NGHỆ THÔNG TIN

NGƯỜI HƯỚNG DẪN KHOA HỌC:

PGS – TS ĐẶNG QUANG Á

THÁI NGUYÊN 2008

MỤC LỤC

MỤC LỤC 2
DANH MỤC CÁC HÌNH VẼ, ĐỒ THỊ 4
MỞ ĐẦU 5
CHƯƠNG I: GIỚI THIỆU BÀI TOÁN LẬP TRÌNH CHO ROBOT 7
1.1. Robot nhân tạo 7
1.2. Bài toán lập lộ trình 9
1.3.Ví dụ và những ứng dụng về lộ trình Robot 12
1.4. Những thành phần cơ bản của việc lập lộ trình 16
1.5. Giải thuật, ng- ời lập lộ trình và lộ trình 17
1.6. Kết luận 23
Ch- ơng II- CẤU HÌNH KHÔNG GIAN TRẠNG THÁI 24
2.1.Các Khái niệm cấu hình không gian 24
2.1.1. Ch- ớng ngại *(Obstacle)*..... 24
2.1.2. Không gian trống (Free Space- C_{free})..... 25
2.2. Mô hình cấu hình 26
2.2.1. Mô hình hình học 26
2.2.2. Mô hình nửa Đại số 32
2.3. Các phép biến đổi của robot 35
2.4. Không gian cấu hình ch- ớng ngại vật 37
2.5- Định nghĩa chính xác về vấn đề lập lộ trình chuyển động 38
2.6. Một số mô hình C_{obs} 39
2.7. Kết luận 47
Ch- ơng III- MỘT SỐ PHƯƠNG PHÁP CHÍNH XÁC LẬP LỘ TRÌNH CHUYỂN ĐỘNG 48
3.1.Giới thiệu chung 48
3.2. Biểu diễn không gian chướng ngại vật 50
3.3. Một số giải thuật lập lộ trình chính xác cho robot 53
3.3.1 . Roadmap Visibility Graph – Đồ thị tầm nhìn 53
3.3.2. Vertical Cell Decomposition (phân ly Ô dọc) 59
KẾT LUẬN 68
TÀI LIỆU THAM KHẢO 69
PHỤ LỤC 1 - Chương trình thử nghiệm Visibility Graph 70
PHỤ LỤC 2- Chương trình thử nghiệm Vertical Cell Decomposition 73

DANH MỤC CÁC HÌNH VẼ VÀ ĐỒ THỊ

Hình 1.1 *Các thành phần cấu thành Robot* 9

Hình 1.2 : *Khối Rubik (a), Bài toán xếp hình (b)*....12

Hình1.3: *Tìm giải thuật kéo hai thanh thép tách ra* 12

Hình 1.4: *Sử dụng Robot di động để di chuyển một Piano* 13

Hình 1.5: *Thử nghiệm một số Robot tránh vật cản*....14

Hình 1.6:*Robot tự xây dựng bản đồ môi trường và xácđịnh vị trí của chính nó* 14

Hình 1.7: *Máy Turing*....17

Hình 1.8: *Gianh giới giữa máy và môi trường*....18

Hình 1.9: *Robot với những công tắc đóng vai trò như một máy Turing* 19

Hình 1. 10 :*Mối quan hệ giữ lộ trình và người lập lộ trình*....20

Hình 1.11: *Một cách tiếp cận cải tiến trong kỹ thuật rôbôt* 22

Hình 1.12- *Mô hình có thứ bậc*....22

Hình 2.1: *Cấu hình không gian*....25

Hình 2. 2*: Một Robot điểm di chuyển trong không gian 2D, C-space là* R^2 26

Hình 2.3: *Một robot điểm di chuyển trong không gian 3D, C-space là* R^3....26

Hình 2.4 - *Cách xác định một đa giác lồi bằng phép giao của những nửa - mặt phẳng)* 27

Hình 2.5- *Dấu hiệu của f(x, y) phân chia* R^2 28

Hình 2.6: *Mô tả một đa diện* 31

Hình 2.7 *: f được sử dụng để phân chia* R^2 *vào trong hai vùng*....*32*

Hình 2.8 : *Biểu diễn mô hình đa giác với những lỗ trống*....34

Hình 2.9: *Hai cách giải thích cho phép tịnh tiến*....35

Hình 2.10 - *Khái niệm về C-space và nhiệm vụ là tìm một đường từ qI đến qG trong C_{free} với $C = C_{free} \blacksquare C_{obs}$*.....38

Hình 2.11 *: Một chướng ngại không gian C- một chiều*.....40

Hình 2.12: *Một robot A tam giác và một chướng ngại hình chữ nhật*.....41

Hình 2.13: *Xây dựng C_{obs} trong phép tịnh tiến* 42

Hình 2.14 : *Lấy và sắp xếp các véc tơ pháp tuyến* 42

Hình 2.15:*Hai kiểu va chạm phát sinh cạnh cho C_{obs}*.....43

Hình 2.16 : *Trạng thái va chạm khi n và v vuông góc* 43

Hình 2.17: *Ba kiểu tiếp xúc khác nhau sinh ra các loại C_{obs} khác nhau*.....44

Hình 2.18: *Xây dựng C_{obs} cho phép quay* 45

Hình 3.1: *Một mô hình không gian được chỉ rõ bởi bốn đa giác giác có hướng* 51

Hình 3.2 : *Xây dựng Roadmap Visibility Graph* 54

Hình 3.3: *qI và qG đ- ợc nối tới tất cả đỉnh có thể thấy của roadmap* 54

Hình 3.4 : *Đường đi ngắn nhất trong Roadmap $_s$*.....55

Hình 3.5 *:Sơ đồ thuật toán Visibility Graph*.....57

Hình 3.6: *Một số đường dẫn giải pháp của phương pháp Visibility Graph*.....58

Hình 3.7 : *Bốn tr·ường hợp tổng quát của tia phân ly* 60

Hình 3.8 : *Sử dụng phương pháp phân ly ô dọc để xây dựng một roadmap* 60

Hình 3.9 *: Roadmap bắt nguồn từ sự phân ly ô dọc* 61

Hình 3.10: Ví dụ về đ- ờng dẫn giải pháp.....62

Hình 3.11 : *Ví dụ có 14 sự kiện*.....63

Hình 3.12: *Tình trạng của L đ- ợc chỉ ra sau mỗi 14 sự kiện xuất hiện* 64

Hình 3.13: *Sơ đồ thuật toán ph- ơng pháp Cell Decomposition* 66

Hình 3.13: *Một số đường đi giải pháp của pp Cell Decompsition* 67

MỞ ĐẦU

Tìm đường là một khoa học (hay nghệ thuật) hướng dẫn lộ trình cho robot di chuyển qua môi trường với mong muốn đến đ-ợc đích mà không bị lạc hay va vào những đối tượng khác.

Thông thường, một lộ trình đ-ợc lập trước để dẫn dắt robot đến đích của nó. Với ph-ơng pháp này, môi tr-ờng robot đi qua phải đ-ợc biết hoàn toàn và không thay đổi, robot có thể đi theo một cách hoàn hảo. Hạn chế của việc vạch lộ trình tr-ớc đòi hỏi việc nghiên cứu tìm hiểu việc vạch lộ trình nội tại, phụ thuộc vào các tri thức thu đ-ợc từ môi trường hiện tại đề xử lý các chướng ngại ch-a biết khi robot băng qua môi trường.

Trên thế giới hiện nay robot là một lĩnh vực đ-ợc hết sức quan tâm. Bài toán lập lộ trình cho robot là bài toán cơ bản để thiết kế chế tạo Robot, do vậy việc tìm hiểu bài toán và nghiên cứu các ph-ơng pháp vạch lộ trình là hết sức quan trọng cần thiết cho sự phát triển lĩnh vực thiết kế và chế tạo Robot. Đã có một số nghiên cứu để giải quyết bài toán nh- ứng dụng giải thuật di truyền lập ch-ơng trình tiến hoá, xây dựng một số các thuật toán cho bài toán, nh-ng đây vẫn là một vấn đề mở đang rất đ-ợc quan tâm. Đặc biệt trong n-ớc, đây là một lĩnh vực còn t-ơng đối mới mẻ, hầu nh- ch-a có các tài liệu đề một cách đầy đủ về lĩnh vực này.

Nhận thức đ-ợc vấn đề đó và với sự gợi ý định h-ớng của PGS .TS Đặng Quang Á em đã chọn nghiên cứu đề tài "**Một số phương pháp chính xác lập lộ trình chuyển động cho Robot"**. Nội dung cơ bản của luận văn tốt nghiệp gồm có ba chương:

Chương 1- Trình bày tổng quan bài toán lập lộ trình cho Robot đó là các khái niệm cơ bản về Robot, và bài toán về Robot, thuật toán và một số ví dụ ứng dụng bài toán lập lộ trình cho Robot.

Chương 2- Trình bày các khái niệm về cấu hình không gian trạng thái, cách biểu diễn không gian trong bài toán lập lộ trình cho robot. Đây là

các khái niệm cơ sở để biểu diễn đ- ợc bài toán cho các giải thuật lập lộ trình chuyển động cho robot.

Chương 3- Đi sâu nghiên cứu một số ph- ơng pháp chính xác lập lộ trình chuyển động cho Robot. Cụ thể đó là hai ph- ơng pháp ROADMAP VISIBILITY GRAPH và CELL DECOMPSITION. Đây là những cách tiếp cận tổ hợp tới việc lập lộ trình chuyển động để tìm thấy những đ- ờng đi xuyên qua không gian cấu hình liên tục mà không dùng đến những thuật toán xấp xỉ.

Qua luận văn này em xin chân thành cảm ơn: PGS .TS Đặng Quang Á - Viện Công nghệ thông tin đã tận tình giúp đỡ, động viên, định h- ớng, h- ớng dẫn em nghiên cứu và hoàn thành luận văn này. Em xin cảm ơn các thầy cô giáo trong viện Công nghệ thông tin, các thầy cô giáo khoa Công nghệ thông tin ĐH Thái nguyên, đã giảng dạy và giúp đỡ em trong hai năm học qua, cảm ơn sự giúp đỡ nhiệt tình của các bạn đồng nghiệp .

THÁI NGUYÊN 11/2008

Người viết luận văn

Nguyễn Thị Thu Thuỷ

CHƯƠNG I

GIỚI THIỆU BÀI TOÁN LẬP LỘ TRÌNH CHO ROBOT

1.1. ROBOT NHÂN TẠO.

Cùng với sự phát triển của khoa học, công nghệ robot ngày càng được ứng dụng rộng rãi trong các lĩnh vực của đời sống xã hội. Chúng có thể là những thiết bị điều khiển tự động trong các dây truyền công nghiệp, hoặc có thể là những robot làm việc trong những môi trường phức tạp mà con người đôi khi không thể tiếp cận được như: môi trường nhiệt độ cao, áp suất lớn hay ở ngoài khoảng không vũ trụ. Không chỉ có vậy robot còn được ứng dụng rất nhiều trong đời sống ví dụ như: Robot lau dọn sàn nhà, robot hướng dẫn di chuyển, robot phục vụ trong các toà nhà cao tầng, robot phẫu thuật,...

Robot được ứng dụng rộng rãi và có nhiều tính năng ưu việt như vậy song không phải ai cũng có thể hiểu về nguyên lý của những tác vụ mà robot có thể thực hiện. Sau đây sẽ là những trình bày sơ lược về nguyên tắc cấu tạo và nguyên lý làm việc của một mobile robot.

1.1.1. Tổng quan

- Về cấu tạo: Robot phải dược trang bị bộ cảm nhận để cảm nhận các thông tin về môi trường như: sensor, encoder. Các bộ phận thực hiện hành động: bánh xe để chuyển động, cánh tay robot…
- Các tri thức mà robot cần được trang bị là: Cấu trúc của môi trường làm việc, các hoàn cảnh mà robot có thể gặp và các hành động mà robot cần thực hiện trong các hoàn cảnh đó, ... Các tri thức này cần phải được thể hiện một cách thích hợp sao cho thuận tiện cho việc lưu trữ, tìm kiếm và suy diễn.

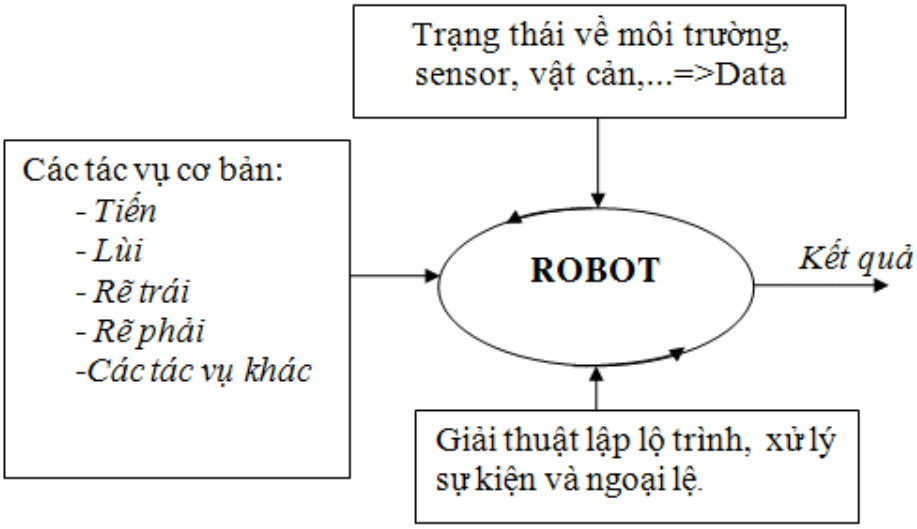
- Các khả năng của robot: Robot cần có khả năng phân biệt được các đối tượng mà nó gặp, thực hiện các thao tác, di chuyển an toàn trong môi trường sao cho đường đi là tối ưu và không va trạm với các vật cản.

1.1.2. Giải pháp thiết kế

Để thiết kế robot ta phải hoàn thiện các công việc sau:

- Xem robot như một đối tượng lập trình bao gồm:
  - Dữ liệu: Các trạng thái của môi trường làm việc, giá trị của sensor, encoder...
  - Tác vụ: Là tập các hành động cơ bản mà robot có thể thực hiện như: tiến, lùi, rẽ trái, rẽ phải, ...
- Mô hình hoá môi trường làm việc.
- Mô hình hoá đối tượng robot sẽ gặp, xử lý các tác vụ trong môi trường làm việc, cùng với việc xử lý dữ liệu và các trạng thái trong môi trường.
- Nhúng các giải thuật tìm đường và giải thuật xử lý sự kiện cho robot để có một đường đi tốt từ vị trí ban đầu tới đích và xử lý các tình huống ngoại lệ như va chạm.
- Phân chia và module hoá các khối trên robot.
- Xây dựng các thành phần robot bao gồm: Lập trình, mạch phần cứng, cơ cấu cơ khí. Cả ba quá trình này phải triển khai đồng bộ với nhau và chúng có tác động rất lớn tới nhau, sự hoàn thiện phần này là tiền đề để xây dựng phần kia.
- Cơ chế hiển thị và Debug lỗi qua các giao tiếp Led/LCD hay với PC.

  Các thành phần cấu thành nên robot có thể được mô hình hoá bởi sơ đồ sau:



Hình 1.1. *Các thành phần cấu thành Robot*

Tất cả các thành phần trên góp phần cấu thành một Robot hoàn chỉnh. Ta có thể ví các cơ cấu cơ khí giống nh- phần thể xác, các mạch điện tử giống nh- các mạch máu, các noron thần kinh và các giác quan bên ngoài. Ch- ơng trình giống nh- bộ não giúp điều khiển cơ thể thông qua hệ thống mạch.

1.2. BÀI TOÁN LẬP LỘ TRÌNH.

Nền tảng quan trọng trong kỹ thuật rôbôt là xây dựng những giải thuật để mô phỏng những nhiệm vụ bậc cao của con ng- ời vào trong những ngôn ngữ bậc thấp của máy để có thể điều khiển robot di chuyển. Những thuật ngữ lập lộ trình và quỹ đạo chuyển động th- ờng đ- ợc sử dụng trong vấn đề này. Việc lập lộ trình chuyển động robot thông th- ờng không quan tâm nhiều đến lĩnh vực động lực học, trọng tâm cơ bản của vấn đề này là tìm đ- ờng và di chuyển đến đích tránh sự va trạm với môi tr- ờng xung quanh. Việc lập lộ trình quỹ đạo thực chất là lấy giải pháp từ một giải thuật lập lộ trình chuyển động robot và xác định làm sao để di chuyển theo giải pháp đó nh- ng ngoài ra còn phải chú trọng tới những hạn chế cơ khí của robot.

Việc lập lộ trình là một vấn đề có nhiều ý nghĩa đối với những lĩnh vực khác nhau:

Trong lý thuyết điều khiển, vấn đề này đ- ợc đề cập tới nh- việc thiết kế những hệ thống vật lý mô tả bởi những ph- ơng trình vi phân. Những hệ thống đó có thể bao gồm những hệ thống cơ khí nh- ô tô hoặc máy bay, những hệ thống điện nh- - lọc tiếng ồn, hoặc cả những hệ thống xuất hiện trong nhiều lĩnh vực đa dạng khác nh- hóa học, kinh tế học, và xã hội học. Tr- ớc đây, lý thuyết điều khiển là điều khiển mờ phản hồi, cho phép một sự hồi đáp có khả năng thích ứng trong thời gian thực hiện, tập trung về sự ổn định, mà bảo đảm rằng vấn đề động lực học không gây cho hệ thống trở nên lộn xộn mất điều khiển. Một tiêu chuẩn quan trọng cho sự tối - - u hóa để tối giản tiêu thụ tài nguyên, nh- năng l- ợng hoặc thời gian. Trong các tài liệu lý thuyết điều khiển gần đây, việc lập lộ trình chuyển động đôi khi quy dẫn đến xây dựng đầu vào cho một hệ thống động lực phi tuyến để điều khiển robot từ một vị trí ban đầu đến một đích xác định .

Trong trí tuệ nhân tạo, những thuật ngữ việc lập lộ trình và việc lập lộ trình AI đảm nhiệm một nhiệm vụ riêng. Thay vào việc di chuyển một pianô qua một không gian liên tục, thì vấn đề lập lộ trình chuyển động cho robot trong trí tuệ nhân tạo với những nhiệm vụ giải quyết một bài toán, nh- bài toán Rubik hoặc một bài toán sliding-tile puzzle (xếp hình). Lộ trình trong trí tuệ nhân tạo đ- ợc định nghĩa là một tập hữu hạn của những hành động có thể đ- ợc áp dụng cho một tập hợp riêng biệt những trạng thái và xây dựng một giải pháp thích hợp cho dãy những hành động đó.

Trong lịch sử, việc lập lộ trình đã xem xét trên những góc độ khác nhau giải quyết những vấn đề khác nhau trong từng lĩnh vực; tuy nhiên, trong những năm gần đây thì sự phân biệt này có vẻ mờ nhạt dần. Trong phạm vi rộng những vấn đề đề cập trong thuật ngữ lập lộ trình đã đ- ợc áp dụng trong tất cả các lĩnh vực trí tuệ nhân tạo, lý thuyết điều khiển, và kỹ thuật rôbôt. Vài nguyên lý cơ bản chung của những vấn đề lập lộ trình sẽ đ- ợc xem xét, nh- ng tr- ớc hết chúng ta coi việc lập lộ trình nh- một nhánh của giải thuật. Từ đây, chúng ta nghiên cứu những giải thuật lập lộ trình. Trọng tâm là thuật toán và những vấn đề cài đặt một số ph- ơng pháp lập lộ trình. Ngoài ra, có nhiều khái niệm không hẳn là thuật toán nh- ng có tác

dụng bổ trợ rất nhiều trong việc xây dựng những mô hình, giải quyết, và phân tích vấn đề lập lộ trình. Đó là các vấn đề để trả lời cho những câu hỏi sau:

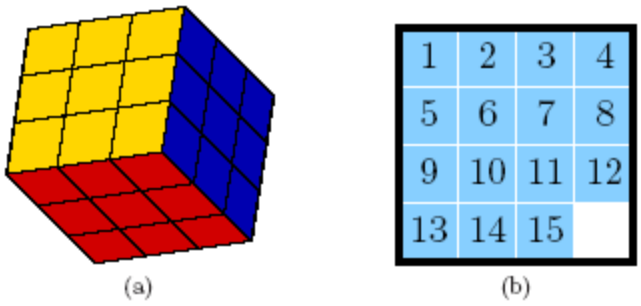
- Thế nào là một lộ trình?
- Một lộ trình đ- ợc mô tả nh- thế nào?
- Nó đ- ợc cài đặt nh- thế nào trong máy tính?
- Nh- thế nào đ- ợc cho là hoàn tất?
- Chất l- ợng của của một lộ trình đ- ợc đánh giá nh- thế nào?
- Ai hoặc cái gì sẽ sử dụng nó?

Ở đây**, khái niệm user** của lộ trình cũng sẽ th- ờng xuyên đ- ợc nhắc đến nh- khái niệm robot hoặc nhà chế tạo. Trong trí tuệ nhân tạo và những lĩnh vực liên quan, ng- ời ta sử dụng thuật ngữ này phù hợp với từ sinh ra một tác nhân thông minh hoặc tác nhân phần mềm. Trong lý thuyết điều khiển th- ờng nhắc tới các nhà chế tạo nh- một ng- ời giám sát, kiểm tra. Trong ngữ cảnh lập lộ trình này đôi khi đ- ợc nhắc tới nh- một chính sách hoặc luật điều khiển. Trong một ngữ cảnh lý thuyết trò chơi, nó có thể có ý nghĩa để h- ớng tới những nhà sản xuất chế tạo nh- những bộ chơi. Những ngôn ngữ nh- robot, đại diện, và ng- ời giám sát có thể thay thế lẫn nhau.

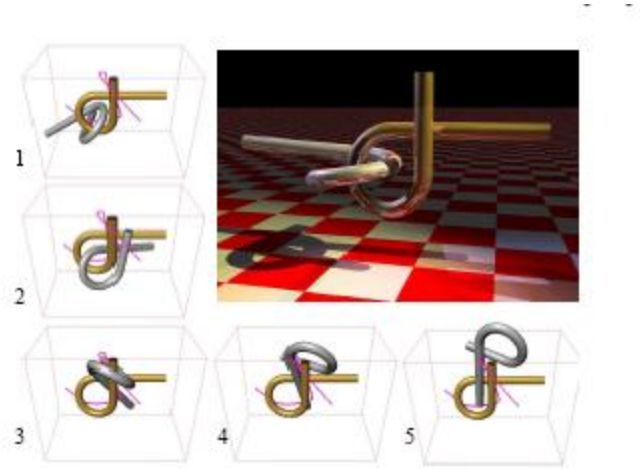
Tại sao phải nghiên cứu những giải thuật lập lộ trình?

Có ít nhất hai lý do cần phải giải quyết cho vấn đề này. Tr- ớc hết, đó là những trò chơi để máy có thể giải quyết những vấn đề gây khó khăn lớn cho con ng- ời. Điều này đòi hỏi những thách thức cần phải thiết kế những giải thuật hiệu quả, và phát triển và bổ sung các mô hình lập lộ trình. Thứ hai, việc lập lộ trình với những giải thuật đã đạt đ- ợc những thành công lớn trong cả lý thuyết và thực tế ở các lĩnh vực khác nhau nh- kỹ thuật rôbôt, thiết kế sản xuất kiểu mẫu thuốc, và những ứng dụng trong không gian vũ trụ. Sự phát triển nhanh trong vài năm gần đây cho thấy những ứng dụng ngày càng hấp dẫn hơn. Điều đó đang thúc đẩy mạnh việc học tập và nghiên cứu những giải thuật lập lộ trình và góp phần cho sự phát triển và sử dụng chúng.

1.3. VÍ DỤ VÀ NHỮNG ỨNG DỤNG VỀ LỘ TRÌNH ROBOT



Hình 1.2 : *Khối Rubik (a), Bài toán xếp hình (b), và những bài toán xếp hình liên quan là những ví dụ khác nhau của vấn đề lập một chiến l-ợc lộ trình đầu tiên.*



Hình 1.3. *Tìm một giải thuật với mục đích sẽ kéo hai thanh thép tách ra .*

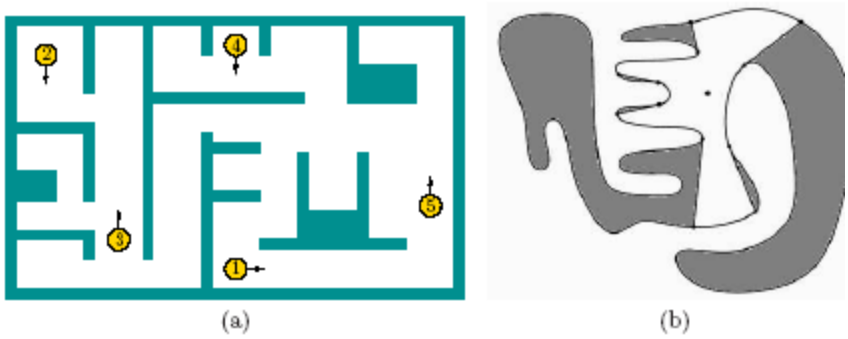
Ví dụ này đ-ợc gọi Alpha 1.0 Puzzle Bài toán này đ-ợc Boris Yamrom đề xuất và nh- một chuẩn đánh giá chính nghiên cứu bởi Nancy Amato tại tr-ờng đại học Texas A&M. Giải pháp cho bài toán này đ-ợc James Kuffner đề xuất.

Việc lập lộ trình chuyển động trong bài toán xếp hình trong *Hình 1.2* có thể dễ dàng thực hiện bởi vì tính đều đặn và đối xứng của những thành phần tham gia vào di chuyển. Bài toán ở *Hình 1.3* lại đ-a ra một vấn đề không có những thuộc tính trên và đồng thời yêu cầu lập lộ trình trong một không gian liên tục. Đây chính là những vấn đề cần đ-ợc giải quyết trong kỹ thuật lập lộ trình chuyển động. Mặc dù vấn đề này có vẻ chỉ đơn thuần là những trò giải trí, những vấn đề t-ơng tự xuất hiện

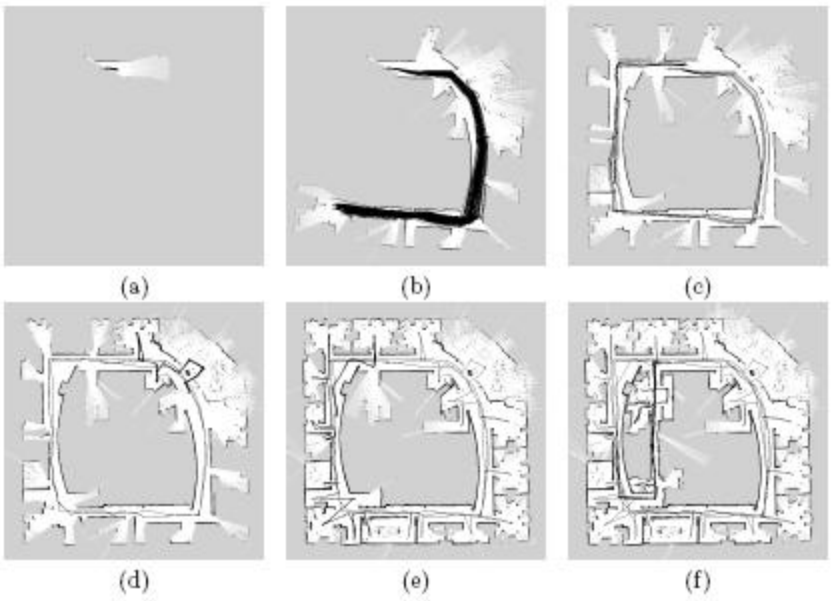
trong những ứng dụng quan trọng. Tri thức chuyển động không chỉ hoàn toàn là trò giải trí, vấn đề này đ- ợc đ- a vào bài toán khác nh- chuyển một đàn pianô qua một căn phòng bằng cách sử dụng ba robot di động với cánh tay thao tác của chúng. Cần phải tránh sự va chạm giữa những robot với những đồ đạc khác. Vấn đề sẽ phức tạp khi những robot, pianô, và không gian xung quanh những mẫu dây chuyền động học khép kín, mà không gian ch- a đ- ợc nhận biết rõ ràng.

Hình 1.4- *Sử dụng những robot di động để di chuyển một Pianô*

Tìm đ- ờng cho những robot di động: Một nhiệm vụ phổ biến cho những robot di động là đòi hỏi chúng tìm đ- ờng đi trong môi tr- ờng trong nhà (*Hình 1.4a*). Một robot có thể đ- ợc yêu cầu để thực hiện những nhiệm vụ nh- xây dựng một bản đồ của môi tr- ờng, xác định vị trí chính xác của nó bên trong một bản đồ, đích cần đến. Đa số các robot hành động bất chấp tình trạng không chắc chắn. Tại một cực trị, nếu xuất hiện mà có nhiều cảm biến thì có lợi bởi vì nó có thể cho phép - ớc l- ợng chính xác môi tr- ờng và robot, xây dựng một bản đồ của môi tr- ờng của nó là tiền đề của nhiều hệ thống hiện nay, đây là một lựa chọn đ- ợc - a chuộng cho việc phát triển những robot đáng tin cậy hoàn thành những nhiệm vụ đặc biệt và chi phí t- ơng đối thấp.



Hình 1.5: *(a) Thử nghiệm thành công một số robot di động định h- ớng trong một môi tr- ờng trong nhà khi phải tránh những sự va chạm với những bức t- ờng và tránh lẫn nhau. (b) Sử dụng một đèn lồng để tìm kiếm những ng- ời mất tích trong một hang.*



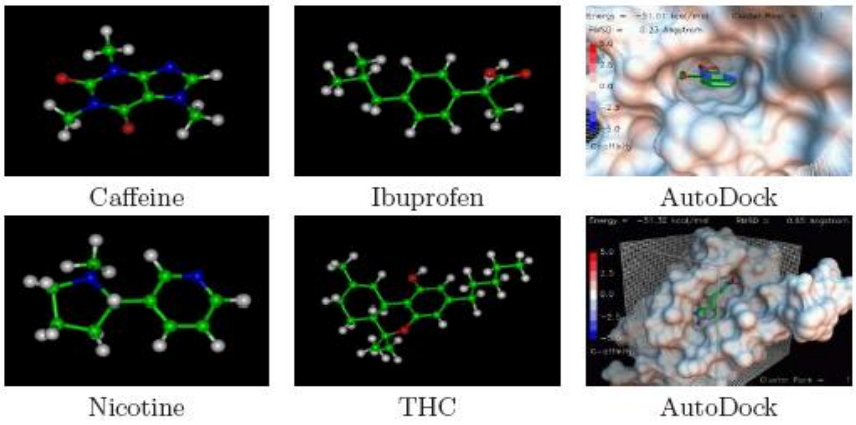
Hình 1.6 : *Một robot di động đáng tin cậy xây dựng tốt một bản đồ về môi tr- ờng của nó (Phòng thí nghiệm nghiên cứu Intel) trong khi đồng thời cục bộ hóa vị trí của chính nó. Điều này đ- ợc hoàn thành bởi sử dụng sensors laze quét thực hiện Bayesian hiệu quả để tính toán trên thông tin về khoảng cách.*

Trò chơi ẩn dấu và tìm kiếm: Một nhiệm vụ quan trọng cho một robot di động là chơi trò chơi ẩn dấu và tìm kiếm. Hãy t- ởng t- ợng vào trong một hang hoàn toàn tối. Bạn đ- ợc đ- a cho một chiếc đèn lồng và yêu cầu để tìm kiếm những ng- ời ở đó

và họ cũng có thể di động, *(Hình 1.4 b)*. Một vài câu hỏi có thể thực hiện đ- ợc nhiệm vụ:

- Liệu có tồn tại một chiến l- ợc bảo đảm rằng ta sẽ tìm thấy mọi ng- ời không? Nếu không, sẽ phải thực hiện nh- thế nào để tiếp tục tìm kiếm và hoàn thành nhiệm vụ này?
- Đâu là nơi ta cần phải di chuyển đến tiếp theo?
- Làm thế nào ta có thể tránh đ- ợc việc thăm dò một chỗ nhiều lần?

Kịch bản này xuất hiện trong nhiều ứng dụng kỹ thuật robot. Ngoài kỹ thuật rôbôt, công cụ phần mềm có thể phát triển giúp những ng- ời trong hệ thống tìm kiếm hoặc làm việc trong những môi tr- ờng phức tạp, có những ứng dụng phải tôn trọng nghiêm ngặt những quy luật nh- tìm kiếm và cứu nguy, nh- làm sạch chất độc trong những tòa nhà có kiến trúc phức tạp và kiên cố. Hiện nay, các giải thuật lập lộ trình cho robot đang đi vào những lĩnh vực v- ợt khỏi kỹ thuật rôbôt đơn thuần nh- máy tính phỏng sinh học, những robot khám bệnh tự động. Những mô hình hình học ứng dụng tới từng phân tử và đ- ợc giải quyết bằng những giải thuật lập lộ trình chuyển động.



Robot ngày càng thay thế nhiều lao động và trở nên chuyên dụng hơn, chúng ngày càng đảm nhận đ- ợc nhiều loại công việc lắp ráp. Đặc biệt, robot di động ngày càng trở nên phổ biến và tinh khôn hơn. Trong viễn cảnh lập lộ trình những giải thuật đã đ- ợc áp dụng tới rất nhiều vấn đề nữa. T- ơng lai sự phát triển và ứng dụng của những giải thuật lập lộ trình là rất to lớn.

1.4. NHỮNG THÀNH PHẦN CƠ BẢN CỦA BÀI TOÁN LẬP LỘ TRÌNH

Mặc dầu đề tài lập lộ trình trải ra một lớp rộng của những mô hình và những vấn đề khác nhau, nh-ng có một số thành phần cơ bản xuyên suốt các chủ đề bao trùm nh- bộ phận của việc lập lộ trình. Chúng ta sẽ nghiên cứu một cách khái quát để hiểu đ-ợc các giải thuật lập lộ trình.

1.4.1. Trạng thái:

Trạng thái của vấn đề lập lộ trình là một không gian gồm tất cả các tình trạng có thể xuất hiện của robot và môi tr-ờng xung quanh. Nh- vị trí và sự định h--ớng của một robot, hay nh- vị trí của những mảnh ghép trong một bài toán đố, hoặc là vị trí và vận tốc của một máy bay trực thăng. Trạng thái có thể rời rạc (hữu hạn, vô hạn đếm đ-ợc) hoặc liên tục (vô hạn không đếm đ-ợc) những tình trạng - đ-ợc cho phép của không gian.

Một điều luôn chú ý là không gian trạng thái đ-ợc biểu diễn không t-ờng minh trong một giải thuật lập lộ trình. Trong đa số các ứng dụng, số chiều của không gian trạng thái (số những trạng thái hoặc tổ hợp phức tạp các trạng thái) là quá lớn để có thể trình bày đ-ợc rõ ràng. Tuy vậy, định nghĩa không gian trạng thái là một thành phần quan trọng trong việc trình bày minh bạch một vấn đề lập lộ trình và trong phân tích, thiết kế những giải thuật mà giải quyết nó.

1.4.2.Thời gian

Là tổng thời gian thực hiện dãy những quyết định của vấn đề lập lộ trình. Trong những giải thuật ng-ời ta chú ý đến thời gian tổng thể đ-a robot từ trạng thái ban đầu đến trạng thái đích. Trong các giải thuật lập lộ trình chuyển động ng-ời ta tránh chỉ rõ thời gian cụ thể trên một đ-ờng đi cụ thể mà -ớc l-ợng thời gian trong tr-ờng hợp xấu nhất.

1.4.3. Hành động (Actions)

Một lộ trình phát sinh những hành động thao tác trên những trạng thái. Thuật ngữ hành động và thao tác là nh- nhau trong trí tuệ nhân tạo; Trong lý thuyết và kỹ thuật điều khiển rôbôt, thuật ngữ này liên quan đến việc nhập đầu vào và điều

khiển. Ở một số cách trình bày vấn đề lộ trình, hành động phải đ- ợc chỉ rõ làm sao thay đổi đ- ợc trạng thái khi nó thực thi. Điều này có thể này đ- ợc biểu thị nh- một hàm đánh giá trạng thái của tr- ờng hợp thời gian riêng biệt hoặc nh- một ph- ơng trình vi phân bình th- ờng cho thời gian liên tục. Có một vài vấn đề, những hành động tự nhiên có thể gây hậu quả rắc rối không nằm trong tầm kiểm soát điều khiển của nhà sản xuất. Điều này làm xảy ra tình trạng không chắc chắn nh- ng có thể - ớc đoán để đ- a tới cho vấn đề lập lộ trình.

1.4.4. Trạng thái ban đầu và kết thúc:

Một lộ trình bắt đầu từ một vài trạng thái ban đầu nào đó và cố gắng đi đến một trạng thái đích xác định hoặc bất kỳ trạng thái nào trong tập hợp của những trạng thái đích. Những hành động sẽ đ- ợc lựa chọn để đạt đ- ợc điều đó.

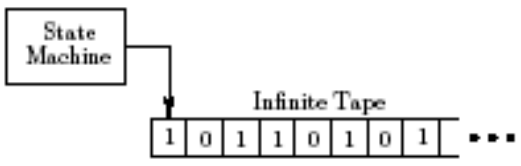
1.4.5. Tiêu chuẩn

Đây là việc mã hóa kết quả mong muốn của một lộ trình bằng không gian trạng thái và các hành động để có thể thực thi đ- ợc. Có hai mối quan tâm khác nhau của bài toán lập lộ trình, dựa trên hai tiêu chuẩn đó là:

1.Tính khả thi : Một lộ trình tìm kiếm sẽ đ- a robot đến trạng thái đích, bất chấp hiệu quả của nó.

2. Sự tối - u : Tìm kiếm mà một lộ trình khả thi mà tối - u hóa, ngoài việc đ- a robot đến trạng thái đích.

1.5. GIẢI THUẬT, NG□ỜI LẬP LỘ TRÌNH VÀ LỘ TRÌNH:



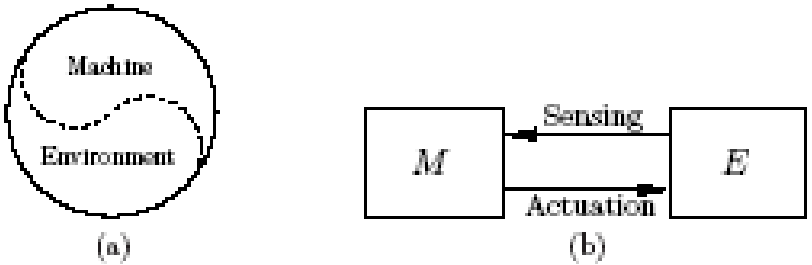
Hình 1.7 **:** *Máy Turing*

1.5.1 Giải thuật

Thế nào là một giải thuật lập lộ trình? Đây là một câu hỏi khó, và không có một định nghĩa toán học chính xác. Thay vào đó, nó sẽ đ- ợc giải thích qua những ý t- ởng chung cùng với nhiều ví dụ về những giải thuật lập lộ trình.

Một câu hỏi cơ bản hơn, thế nào là một giải thuật? Đó là mô hình máy Turing cổ điển, mô hình đã đ- ợc sử dụng để định nghĩa một giải thuật trong lý thuyết khoa học máy tính.

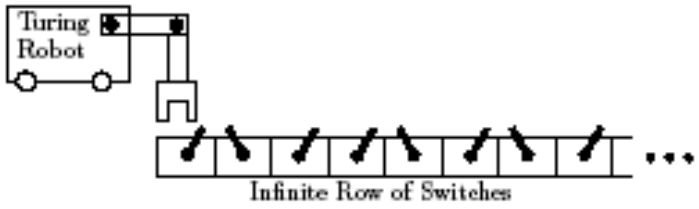
Máy Turing là một máy trạng thái hữu hạn với một đầu đặc biệt mà có thể đọc và viết dọc theo một băng vô hạn *(Hình 1.7)*.



Hình 1.8 : (a) Biểu diễn ranh giới giữa máy và môi tr- ờng bằng một đ- ờng cong đ- ợc vẽ tuỳ ý phụ thuộc vào ngữ cảnh. (b) Biểu diễn ranh giới giữa máy(M), giao diện với môi tr- ờng (E), qua cảm nhận và hành động.

Tuy nhiên, trong ngữ cảnh mở rộng không phức tạp có thể sinh ra những đầu ra mong muốn khác, nh- một lộ trình. Trong các tr- ờng hợp những lộ trình có t- ơng tác với thế giới vật lý có thể mô hình máy Turing không còn phù hợp. Trong *Hình 1.8* cho thấy, ranh giới giữa máy và môi tr- ờng là một đ- ờng bất kỳ thay đổi theo từng bài toán. Khi đó, những sensors cung cấp thông tin về môi tr- ờng; nó trở thành đầu vào cho máy trong suốt thời gian sự thực hiện. Máy sẽ thực hiện hành động theo các chỉ dẫn của một ch- ơng trình, và cung cấp kích thích tới môi tr- ờng. Kích thích có thể thay đổi môi tr- ờng bên trong theo một cách nào đó sau những khoảng thời gian đều đặn bởi những sensors. Bởi vậy, máy và môi tr- ờng của nó gắn bó chặt chẽ với nhau trong suốt thời gian thực hiện. Đây là điều cơ bản đ- ợc sử dụng kỹ thuật rôbôt và nhiều lĩnh vực khác trong đó việc lập lộ trình. Việc sử dụng máy Turing nh- một nền tảng cho những giải thuật thông th- ờng ngầm định rằng đầu tiên phải mô hình hoá đ- ợc thế giới vật lý và sau đó phải viết đ- ợc trên băng tr- ớc

khi giải thuật có thể ra những quyết định. Nếu những sự thay đổi môi tr- ờng th- ờng xuyên xuất hiện trong thời gian thực hiện của giải thuật, thì không dễ dàng dự đoán điều gì sẽ xảy ra. Ví dụ, một robot chuyển động trong một môi tr- ờng lộn xộn mà trong đó có những ng- ời đang đi đi lại lại. Hoặc, một robot ném một vật thể lên trên một bảng khi đó có thể không dự đoán chính xác vị trí dừng lại của vật. Nó có thể sử dụng kết quả của những phép đo với những sensors, nh- ng đây một nhiệm vụ khó để xác định vì rõ ràng là có quá nhiều thông tin cần đ- ợc mô hình để viết trên băng. Trong tr- ờng hợp này mô hình giải thuật trực tuyến sẽ thích hợp hơn. Tuy vậy, máy Turing vẫn là khái niệm giải thuật đủ rộng cho toàn bộ chủ đề mà chúng ta đang quan tâm.



Hình 1.9- *Robot với những công tắc đóng vai trò nh- một máy Turing*

Những quá trình mà xuất hiện trong một thế giới vật lý phức tạp hơn sự t- ơng tác giữa một máy trạng thái và băng ký hiệu. Nó có thể đóng vai trò băng bởi hình dung một robot t- ơng tác với một dãy dài những công tắc đ- ợc miêu tả bên trong *Hình 1.9*. Những sự chuyển mạch phục vụ giống nh- mục đích của băng, và robot mang một máy tính mà có thể đóng vai máy trạng thái hữu hạn.

Trong thực tế, sự t- ơng tác phức tạp cho phép giữa một robot và môi tr- ờng của nó có thể làm cho nhiều mô hình phải tính toán tăng lên. Nh- vậy, thuật ngữ giải thuật sẽ đ- ợc sử dụng có phần không chính xác nh- trong lý thuyết. Ở đây cả những ng- ời lập lộ trình lẫn những lộ trình đ- ợc xem xét nh- những giải thuật.

1.5.2. Ng- ời lập lộ trình

Một ng- ời lập lộ trình đ- ợc hiểu đơn giản là ng- ời lập một lộ trình, có thể là máy hoặc con ng- ời. Nếu ng- ời lập lộ trình là một máy, thì nói chung sẽ đ- ợc xem xét nh- một giải thuật lập lộ trình. Trong nhiều tr- ờng hợp có thể coi nó là một giải thuật trong máy Turing chính xác.

Trong một số tr- ờng hợp, những con ng- ời trở thành những ng- ời lập lộ trình bởi việc phát triển một lộ trình làm việc trong tất cả các tình trạng. Mô hình lập lộ trình đã cho đ- a tới cho con ng- ời, và con ng- ời " tính toán " một lộ trình. Đối với tr- ờng hợp ng- ời lập lộ trình đúng là một con ng- ời thì con ng- ời vẫn làm tròn vai trò của giải thuật.

1.5.3 Lộ trình

1.5.3.1- Khái niệm lộ trình

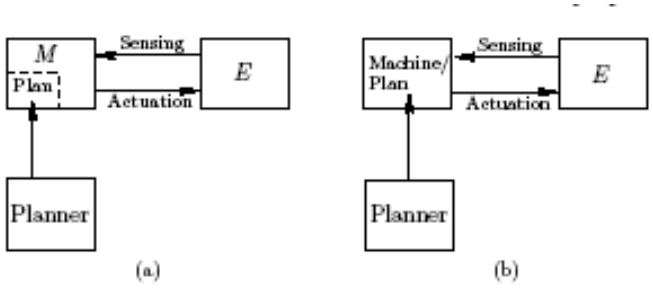
Hiểu một cách đơn giản: Lộ trình là một bản kế hoạch mà nhìn vào đó ng- ời ta có thể xác định đ- ợc cần đi nh- thế nào mà không va phải những ch- ớng ngại vật và đến đ- ợc đích xác định.

Khái niệm cơ sở của lộ trình là không gian. Không gian nói chung chứa đựng các loại thực thể đó là: Ch- ớng ngại vật (Obstacles) , khoảng trống tự do (Free Space) và Robot

- **Ch- ớng ngại vật**: Là thành phần " thường xuyên" chiếm chỗ trong không gian, hay nói cách khác là nơi mà robot không thể đi vào đó. Ví dụ nh- bức t- ờng của một toà nhà.

- **Khoảng trống tự do**: Là nơi còn trống trong không gian mà robot có thể đi vào đó. Để quyết định xem robot có thể đi đ- ợc vào đó hay không chúng ta cần tìm hiểu khái niệm Configuation Space (Cấu hình không gian)

- **Robot:** Những vật thể mà đ- ợc mô hình hoá hình học và có thể kiểm soát theo một lộ trình đã lập.



Hình 1.10 : *Mối quan hệ giữ lộ trình và ng- ời lập lộ trình*

(a) Một lộ trình sản sinh có thể là thực hiện bởi máy. Ng- ời lập lộ trình có thể là một máy hoặc cũng có thể là một con ng- ời. (b) Một lựa chọn khác, ng- ời lập lộ trình có thể thiết kế toàn bộ máy.

Mỗi lần một lộ trình đ- ợc xác định rõ, theo ba cách sử dụng nó :

1. Thực thi : Thực thi lộ trình bằng cách mô phỏng hoặc trên thiết bị cơ khí thực (robot) trong thế giới vật lý thực.

2. Cải tiến : Cải tiến nó để có đ- ợc một lộ trình tốt hơn.

3. Mô hình có thứ bậc : Gói lộ trình nh- một hành động của một ở mức lộ trình cao hơn.

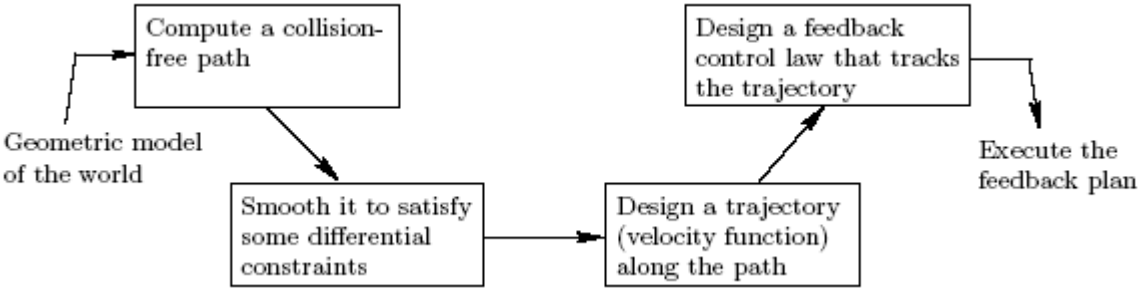
Và chúng sẽ đ- ợc giải thích kỹ hơn nh- sau:

Sự thực thi: Một lộ trình thông th- ờng đ- ợc thực thi bởi một máy. Một con ng- ời có thể thực thi nó theo cách khác. Tuy nhiên, tr- ờng hợp này chúng ta tập trung nghiên cứu sự thực hiện trên máy. Có hai cách chung để thực hiện trên máy.

Thứ nhất, trong *Hình 1.10a*, ng- ời lập lộ trình sinh một lộ trình, mà đ- ợc mã hóa theo một cách nào đó và nhập vào máy. Trong tr- ờng hợp này sau khi đã đ- ợc nhập ch- ơng trình vào thì máy sẽ tự trị tức là tuần tự thực hiện những b- ớc của ch- ơng trình và không còn sự t- ơng tác với ng- ời lập trình nữa. Tất nhiên, mô hình này có thể đ- ợc mở rộng cho phép cải tiến qua thời gian để nhận những lộ trình tốt hơn; Cách tiếp cận này đã có trong những lộ trình thực tế, tuy nhiên, chúng ch- a đ- ợc - a thích bởi những lộ trình cần phải đã đ- ợc thiết kế để tính đến thông tin mới trong thời gian thực thi.

Kiểu thực hiện thứ hai của lộ trình đ- ợc miêu tả trong *Hình 1.10 b*. Trong tr- - ờng hợp này, lộ trình sản sinh bởi ng- ời lập lộ trình mã hóa trọn vẹn trong máy. Đây là một lộ trình đặc biệt chủ định cho máy và đ- ợc thiết kế để giải quyết những nhiệm vụ đặc biệt cho tr- ớc h- ớng tới ng- ời lập lộ trình. Giải thuật này chỉ h- ớng tới để thiết kế cho một số máy để giải quyết đầy đủ một số nhiệm vụ cụ thể. Khi đó chỉ cần một số ít ng- ời và máy cũng có thể giải quyết đ- ợc nhiệm vụ đ- ợc giao.

Nếu lộ trình đ- ợc mã hóa nh- một máy trạng thái hữu hạn, thì đôi khi nó có thể đ- ợc xem xét.

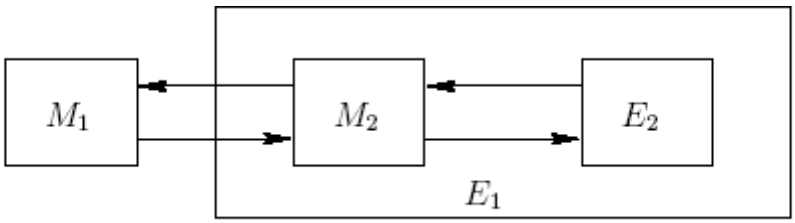


Hình 1.11-*Một cách tiếp cận cải tiến trong kỹ thuật rôbôt*

Sự cải tiến Nếu một lộ trình đ- ợc sử dụng cho sự cải tiến, thì ng- ời lập lộ trình chấp nhận nó nh- đầu vào và xác định một lộ trình mới mà lộ trình này có tính đến nhiều khía cạnh của vấn đề hơn, hoặc nó có thể đơn giản và hiệu quả hơn.

Sự cải tiến có thể đ- ợc ứng dụng nhiều lần, để sản sinh một dãy nối tiếp những lộ trình cải tiến, cho đến khi lộ trình cuối cùng có sự thực thi tốt nhất. *Hình 1.11* cho thấy một cách tiếp cận cải tiến đ- ợc sử dụng trong kỹ thuật rôbôt.

Mô hình thứ bậc là mô hình mà ở đó một lộ trình đ- ợc hợp nhất nh- một hoạt động của một lộ trình lớn hơn.



Hình 1.12- *Mô hình có thứ bậc, môi tr- ờng bản thân một máy có thể chứa đựng một máy khác*

Lộ trình nguyên bản có thể hình dung nh- một ch- ơng trình con trong lộ trình lớn hơn. Để điều này thành công, điều quan trọng là lộ trình nguyên bản bảo đảm sự kết thúc, để lộ trình lớn hơn có thể thực thi chúng nhiều lần khi cần.

Mô hình có thứ bậc có thể đ- ợc biểu diễn với bất kỳ số lộ trình nào, kết quả l- u trong một nút cây của lộ trình mỗi đỉnh của cây là một lộ trình. Trong việc lập lộ trình có thứ bậc, dòng t- ơng tác giữa máy và môi tr- ờng đ- ợc vẽ nhiều h- ớng. Ví dụ, môi tr- ờng E1 của máy M1, có thể chứa máy khác nh- M2, t- ơng tác đó với môi tr- ờng E2 của nó, nh- trong *Hình 1.12*.

1.5.3.2. Lập lộ trình chuyển động

Đây là nguồn chính cảm hứng chính cho những vấn đề và những tất cả các giải thuật của kỹ thuật rôbôt. Những ph- ơng pháp này đủ tổng quát để sử dụng trong những ứng dụng khác trong lĩnh vực khác nhau, nh- máy tính sinh học, thiết kế d- ới sự trợ giúp của máy tính, và đồ hoạ máy tính. Một tiêu đề thay thế chính xác hơn cho việc lập lộ trình chuyển động là " *Việc lập lộ trình trong không gian liên tục* "

1.6. KẾT LUẬN

Ch- ơng này đã giới thiệu tổng quan bài toán lập lộ trình cho robot, ứng dụng của chúng và các khái niệm cơ bản liên quan đến bài toán này nh- giải thuật, ng- ời lập lộ trình và lộ trình. Nội dung của ch- ơng giúp chúng ta có cái nhìn khái quát về bài toán lập lộ trình cho robot và cách tiếp cận với bài toán. Việc mô hình hoá bài toán sẽ đ- ợc giải quyết ở ch- ơng sau.

CH□ƠNG II

CẤU HÌNH KHÔNG GIAN TRẠNG THÁI

Kiến thức nền tảng quan trọng là làm thế nào mô hình hoá đ-ợc robot và không gian trạng thái xung quanh robot. Việc mô hình hoá để giải quyết những vấn đề phức tạp của robot di chuyển và không gian trạng thái xung quanh. Có hai loại mô hình chủ yếu đ-ợc nghiên cứu là mô hình hình học và mô hình nửa đại số. Mẫu nửa đại số là mô hình đáng quan tâm bởi nó là một phần của các giải thuật lập lộ trình chính xác.

Tuy nhiên, để đạt đ-ợc mục đích của việc lập lộ trình thì một điều quan trọng là phải định nghĩa đ-ợc không gian trạng thái. Không gian trạng thái cho việc lập lộ trình chuyển động là một tập hợp những biến đổi có thể ứng dụng đ-ợc cho robot.

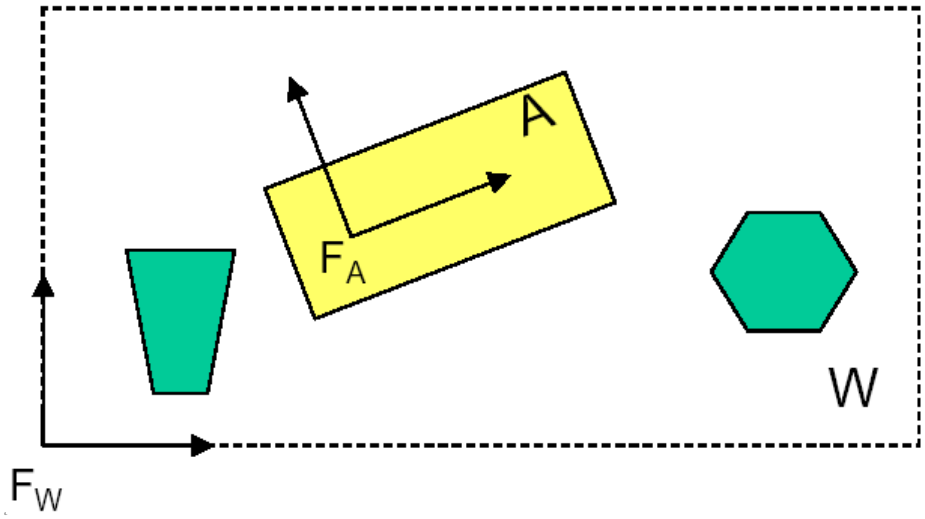
Không gian trạng thái này sẽ nhắc đến nh- *cấu hình không gian*. Một cấu hình không gian phải biểu diễn đ-ợc rõ ràng và dễ hiểu. Nhiều mô hình cấu hình không gian của vấn đề lập lộ trình chuyển động xuất hiện d-ới dạng hình học. Mức trừu t-ợng hóa này rất quan trọng. Các khái niệm của không gian cấu hình liên quan trực tiếp đến toán học, đặc biệt là hình học tôpô.

2.1.CÁC KHÁI NIỆM CẤU HÌNH KHÔNG GIAN:

Cấu hình không gian là không gian của tất cả những trạng thái có thể của bài toán.

2.1.1. Ch- ớng ngại *(Obstacle):*

Là những phần của không gian " th- ờng xuyên " bị choán chỗ, ví dụ nh- trong những bức t- ờng của một tòa nhà.



Ký hiệu:

A: Một thực thể đơn –(Robot)

W: Không gian Euclidean ở đó *A* di chuyển

Hình 2.1- *Cấu hình không gian*

- Cấu hình ch- ớng ngại vật: Là cấu hình của từng ch- ớng ngại vật

- Miền ch- ớng ngại vật: Là hợp của tất cả các cấu hình ch- ớng ngại vật

C-OBSTACLE REGION

From
Robot Motion Planning
J.C. Latombe

$B_1, B_2, \ldots, B_m$ ——— obstacles

CB_i ——— C-obstacle

$\bigcup_{i=1}^{m} CB_i$ ——— C-obstacle region

2.1.2. Không gian trống (Free Space- C_{free}): Là phần bù của toàn bộ không gian với miền ch- ớng ngại vật.

FREE SPACE

$$C_{free} = C \setminus \bigcup_{i=1}^{m} CB_i = \{q \in C : A(q) \cap \bigcup_{i=1}^{m} CB_i = \phi\} \qquad (2.1)$$

Free configuration q iff $q \in C_{free}$

2.2. MÔ HÌNH CẤU HÌNH

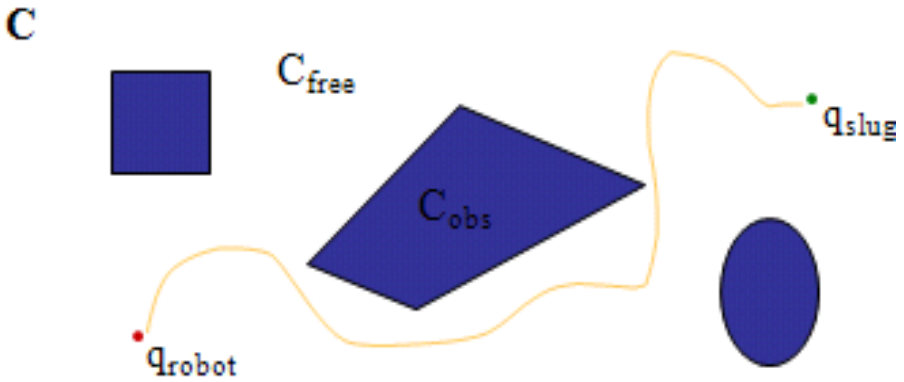
Để có thể thực hiện đ- ợc các giải thuật lập lộ trình ta cần phải biểu diễn đ- ợc không gian cấu hình vào máy. Có nhiều ph- ơng pháp để mô hình hoá không gian ở đây chúng ta quan tâm chủ yếu đến hai loại chính đó là: mô hình hình học và mô hình nửa đại số.

2.2.1. MÔ HÌNH HÌNH HỌC

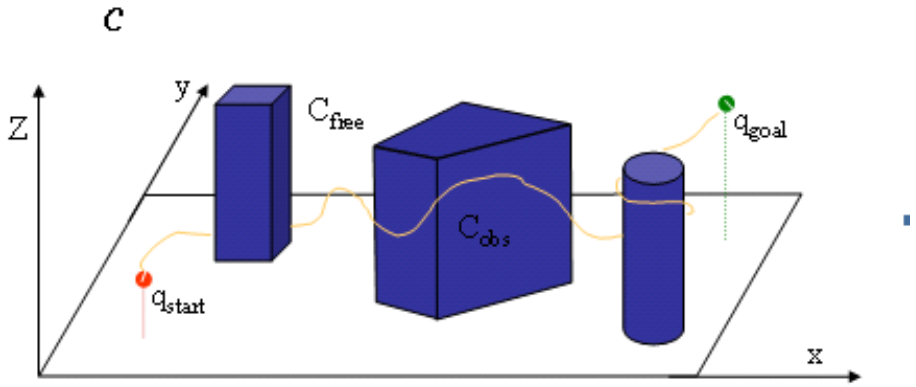
Mô hình hình học và những cách tiếp cận đa dạng trong kỹ thuật robot là một vấn đề rộng lớn, sự lựa chọn mô hình nào thông th- ờng phụ thuộc vào ứng dụng. Nh- ng nói chung trong hầu hết các tr- ờng hợp, có hai lựa chọn chính để biểu diễn W :

1) Không gian 2 chiều, trong $W = R^2$.

2) Không gian 3 chiều, trong $W = R^3$.



Hình 2. 2- *Một Robot điểm di chuyển trong không gian 2D, C-space là* R^2



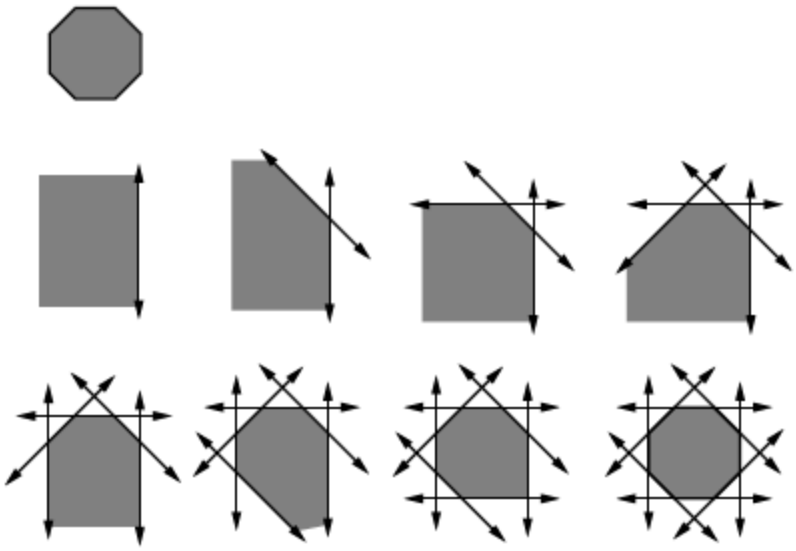
Hình 2. 3- *Một Robot điểm di chuyển trong không gian 3D, C-space là* R^3

Tuy nhiên, trong thực tế có nhiều không gian phức tạp hơn, nh- bề mặt của một hình cầu khi đó cần những không gian có số chiều lớn hơn. Nh- ng lĩnh vực tổng quát đó không đề cập tới trong luận văn này vì những ứng dụng hiện thời của chúng còn có hạn.

2.2.1.1. Mô hình đa giác :

Trong không gian hai chiều 2D, $W = R^2$. Vùng ch- óng ngại vật O là một tập các đa giác lồi. Biểu diễn một m-đa giác trong O đ- ợc mô tả bởi hai đặc tr- ng đó là đỉnh và cạnh.

Mỗi đỉnh t- ơng ứng tới một "góc" của đa giác, và mỗi cạnh t- ơng ứng với một đoạn nối giữa một cặp của đỉnh. Đa giác có thể đ- ợc chỉ rõ bởi đ- ờng nối liên tiếp các cặp đỉnh của m điểm bên trong R^2 theo thứ tự ng- ợc chiều kim đồng hồ: $(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)$.



Hình 2.4 *- Cách xác định một đa giác lồi bằng phép giao của những nửa - mặt phẳng)*

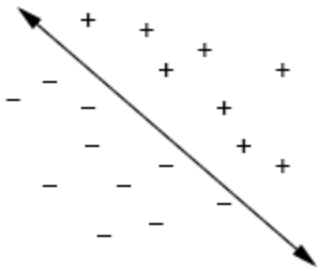
Một hình đa giác trong O có thể đ- ợc biểu thị nh- phép giao của m nửa mặt phẳng. Mỗi nửa mặt phẳng t- ơng ứng tới tập hợp của tất cả các điểm mà nằm ở một phía của đ- ờng thẳng trùng với cạnh của một đa giác. *Hình 2.4* cho thấy một ví dụ của một hình bát giác đ- ợc biểu diễn nh- phép giao của tám nửa mặt phẳng. Một cạnh của đa giác đ- ợc chỉ rõ bởi hai điểm, nh- (x_1, y_1) và (x_2, y_2). Xem xét

ph- ơng trình của một đ- ờng thẳng đi qua (x_1,y_1) và (x_2,y_2). Một ph- ơng trình có thể đ- ợc xác định d- ới dạng: $ax + bx + c = 0$. Trong đó a, b, c ∈R là những hằng số đ- ợc xác định từ x_1, y_1, x_2, và y_2.

Cho ánh xạ $f : R^2 \rightarrow R$ xác định bởi hàm $f(x, y) = ax + bx + c$.

Không mất tính tổng quát ta có thể giả thiết $f(x,y) < 0$ là những điểm nằm bên trái của đ- ờng thẳng, $f(x,y) > 0$ là những điểm nằm bên phải đ- ờng thẳng. Cho $f_i(x, y)$ biểu thị hàm f dẫn xuất từ đ- ờng thẳng mà t- ơng ứng với cạnh đi qua hai điểm từ (x_i, y_i) tới (x_{i+1}, y_{i+1}) với $1 \leq i < m$. Cho $f_m(x, y)$ biểu thị ph- ơng trình đ- ờng thẳng t- ơng ứng với cạnh từ (x_m, y_m) tới (x_1, y_1). Một nửa mặt phẳng H_i với $1 \leq i \leq m$ đ- - ợc xác định một tập con của W:

$$H_i = \{(x, y) \in W \mid f_i(x, y) \leq 0\}. \qquad (2.2)$$



Hình 2.5- *Dấu hiệu của f(x, y) phân chia R^2 vào ba vùng : Hai nửa - mặt phẳng thoả mãn f(x, y) < 0 và f(x, y) > 0 và đ- ờng thẳng f(x, y) = 0.*

Một tập lồi, m – cạnh, vùng O ch- ớng ngại vật đa giác đ- ợc biểu thị nh- sau:

$$O = H_1 \cap H_2 \cap \cdots \cap H_m. \qquad (2.3)$$

Trong đa số các ứng dụng các tập con không lồi có thể vẫn đ- ợc chấp nhận. Khi đó vùng ch- ớng ngại O đ- ợc biểu thị:

$$O = O_1 \cup O_2 \cup \cdots \cup O_n, \qquad (2.4)$$

Trong đó mỗi O_i là một đa giác lồi, với O_i và O_j ($i \neq j$) không cần tách rời nhau.

Với cách này chúng ta có thể biểu diễn đ- ợc rõ ràng các không gian rất phức tạp. Mặc dầu những vùng này có thể chứa đựng nhiều thành phần nh- những lỗ trống. Nói chung, trong những không gian phức tạp hơn thì cần phải biểu diễn thông qua sự kết hợp hữu hạn các phép hợp, giao, và hiệu của tập hợp mẫu; Tuy nhiên, để đơn giản hoá việc biểu diễn các mẫu ng- ời ta cố gắng sử dụng cách chỉ biểu diễn theo hai phép hợp và giao. Một tập hợp hiệu th- ờng tránh đ- ợc sử dụng để biểu diễn mẫu. Để làm đ- ợc nh- vậy ng- ời ta thay những điểm $f_i(x, y) < 0$ trong mẫu H_i bởi những điểm $- f_i(x,y) \geq 0$ và định nghĩa lại một mẫu H_i '.

Một mẫu phức tạp đ- ợc kết hợp bởi những mẫu đơn giản có thể loại bỏ đ- ợc phép hiệu bằng cách áp dụng những phép biến đổi theo các luật của đại số Boolean.

Chú ý rằng sự biểu diễn của một đa giác không lồi không phải là duy nhất. Có nhiều cách để phân tách O thành các đa giác. Do vậy cần phải cẩn thận lựa chọn cách phân tách để tối - u hóa việc tính toán trong những giải thuật sử dụng mô hình. Trong đa số các tr- ờng hợp, những thành phần có thể đ- ợc cho phép giao nhau. Lý t- ởng nhất là việc lựa chọn cách biểu thị O sao cho tối thiểu nhất các mẫu .

Ở đây một logic vị từ đã đ- ợc định nghĩa nh- sau: W : W {TRUE, FALE}. Hàm trả lại giá trị TRUE khi một điểm trong W nằm bên trong O, và ng- ợc lại là False . Cho một đ- ờng thẳng $f(x, y) = 0$ để $e(x, y)$ biểu thị một vị từ lôgíc trả lại giá trị TRUE nếu $f(x, y) = 0$, và ng- ợc lại là FALSE. Một vị từ t- ơng ứng tới một vùng đa giác lồi đ- ợc biểu diễn bởi các phép hội nh- sau:

$$\alpha(x, y) = e_1(x, y) \wedge e_2(x, y) \wedge \cdots \wedge e_m(x, y). \qquad (2.5)$$

Vị từ (x, y) trả về giá trị TRUE nếu điểm (x, y) nằm trong vùng đa giác lồi, ng- ợc lại là FALSE. Một vùng ch- ớng ngại mà gồm có n đa giác lồi đ- ợc biểu diễn bởi tuyển nh- sau:

$$\phi(x, y) = \alpha_1(x, y) \vee \alpha_2(x, y) \vee \cdots \vee \alpha_n(x, y). \qquad (2.6)$$

Mặc dầu tồn tại những ph- ơng pháp hiệu quả hơn, ■ có thể kiểm tra một điểm (x, y) nằm trong O với thời gian O(n), trong đó n là số mẫu mà xuất hiện trong biểu diễn của O (Mỗi mẫu đ- ợc - ớc l- ợng trong hằng số thời gian). Bất kỳ mệnh đề lôgíc phức tạp đến đâu đều có thể đ- ợc tách nhỏ thành những chuẩn tuyển (Đây th- ường được gọi " tổng của những tích " trong khoa học máy tính). Như vậy chúng ta có thể nói bất kỳ một không gian O luôn luôn đ- ợc biểu diễn bằng hợp của hữu hạn các phép giao những mẫu.

2.2.1.2- Mô hình đa diện:

Trong không gian ba chiều $W = R^3$, những khái niệm có thể đ- ợc khái quát hóa rất tốt từ tr- ờng hợp không gian 2D bởi việc thay thế đa giác bằng khối đa diện và thay thế nửa mặt phẳng bởi nửa không gian mẫu.

Một ranh giới biểu diễn có thể đ- ợc định nghĩa d- ới dạng ba đặc tr- ng : đỉnh, cạnh, và mặt. Một vài cấu trúc dữ liệu đ- ợc đ- a ra để biểu diễn đa diện, ví dụ, cấu trúc dữ liệu chứa ba kiểu bản ghi : đỉnh, mặt và nửa cạnh (một nửa cạnh là cạnh có h- ớng).

Giả sử O là một đa diện lồi, nh- trong *Hình 2.5*. Một biểu diễn ba chiều có thể đ- ợc xây dựng từ những đỉnh. Mỗi mặt của O có ít nhất ba đỉnh dọc theo ranh giới của nó. Giả thiết rằng những đỉnh này không cộng tuyến, một ph- ơng trình của mặt phẳng đi qua chúng có dạng:

$$ax + by + cz + d = 0 \qquad (2.7)$$

trong đó a, b, c, d ■ R là những hằng số.

Một lần nữa, f có thể xây dựng bằng ánh xạ $f : R^3$ ■ R và

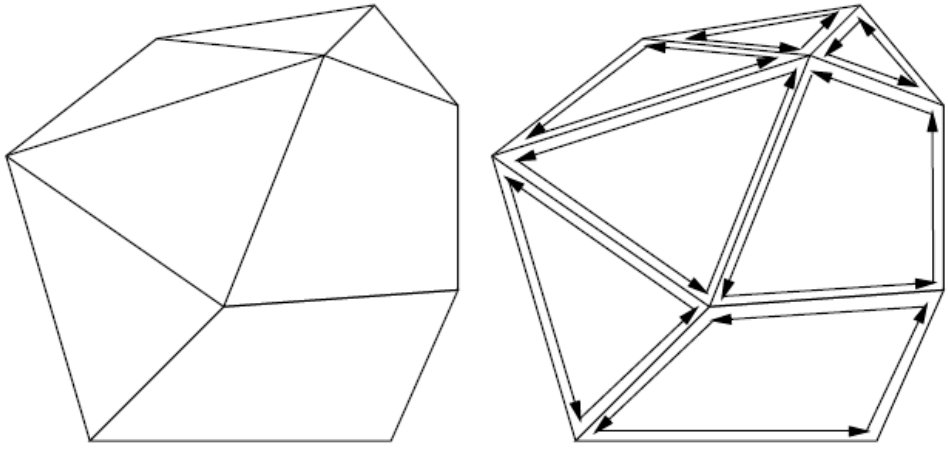
$$f(x, y, z) = ax + by + cz + d. \qquad (2.8)$$

với m mặt. Cho mỗi mặt của O, một nửa - không gian H_i đ- ợc định nghĩa nh- một tập con của W:

$$H_i = \{(x, y, z) \in W \mid f_i(x, y, z) \leq 0\}. \qquad (2.9)$$

Điều quan trọng là chọn f_i để nó giữ những giá trị âm ở trong đa diện. Trong mô hình đa giác, để thích hợp với định nghĩa f_i là việc xuất phát đi vòng quanh biên theo thứ tự ng- ợc chiều kim đồng hồ. Trong tr- ờng hợp một đa diện, ranh giới của mỗi mặt là các cạnh cũng đ- ợc lấy ng- ợc chiều kim đồng hồ. *(Hình 2.6b)*

Ph- ơng trình cho mỗi mặt đ- ợc xác định nh- sau: Chọn ba đỉnh liên tiếp p_1, p_2, p_3 (không đ- ợc cộng tuyến) theo thứ tự ng- ợc chiều kim đồng hồ. Cho v_{12} biểu thị vectơ từ p_1 tới p_2, v_{23} biểu thị vectơ từ p_2 đến p_3. Tích $v = v_{12} \times v_{23}$ luôn luôn là một vectơ nằm trong mặt phẳng gọi là vectơ hồi. Véc tơ [a b c] song song với mặt phẳng. Nếu những thành phần của nó đ- ợc chọn là a = v[1], b=v[2], c = v[3], thì f(x, y, z) = 0 cho mọi điểm trong nửa - không gian chứa đa diện.



Hình 2.6: *(a) Mô tả một đa diện d- ới dạng mặt, cạnh, và đỉnh. (b) Những cạnh của mỗi mặt có thể đ- ợc l- u trữ trong chu trình theo thứ tự ng- ợc chiều kim đồng hồ.*

Trong tr- ờng hợp của một đa giác mẫu, một đa diện lồi có thể đ- ợc định nghĩa nh- giao của một số hữu hạn những nửa - không gian, cho mỗi mặt. Một đa diện không lồi có thể đ- ợc định nghĩa nh- hợp của một số hữu hạn các đa diện lồi. Vị từ (x, y, z) có thể đ- ợc định nghĩa t- ơng tự là TRUE nếu (x, y, z) $\in O$, và FALSE trong tr- ờng hợp ng- ợc lại.

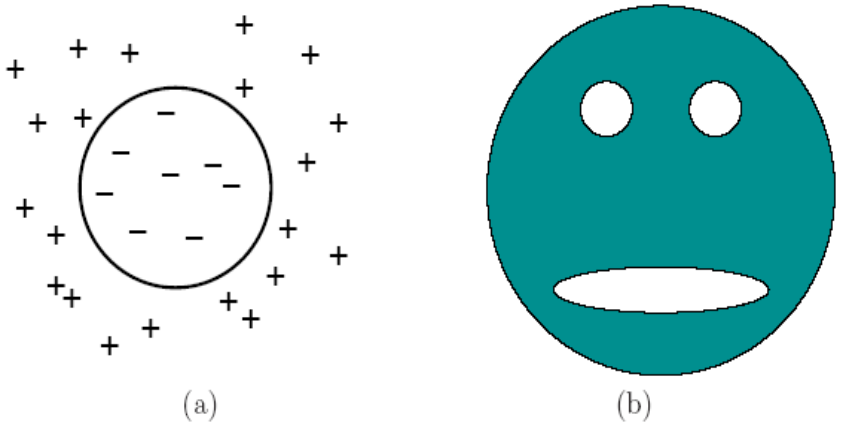
2.2.2. MÔ HÌNH NỬA ĐẠI SỐ

Trong những mô hình đa giác và đa diện, f là một hàm tuyến tính.

Trong tr- ờng hợp của một mô hình nửa đại số của không gian 2D, f là đa thức với những hệ số bất kỳ của hai biến thực x và y. Trong không gian 3 chiều, f là một đa thức với ba biến thực x, y, z. Lớp những mô hình nửa đại số bao gồm cả hai mô hình đa diện và đa giác, mà sử dụng tr- ớc hết cho đa diện. Một tập hợp điểm xác định bởi một mẫu đa thức đơn đ- ợc gọi một tập hợp đại số; Một tập hợp điểm mà có thể thu đ- ợc bởi một số hữu hạn của những phép hợp và phép giao những tập hợp đại số đ- ợc gọi một tập nửa đại số.

Xem xét tr- ờng hợp của không gian 2D. Một biểu diễn 3 chiều có thể đ- ợc định nghĩa sử dụng những mẫu đại số có mẫu dạng:

$$H = \{(x, y) \in \mathcal{W} \mid f(x, y) \leq 0\}. \qquad (2.10)$$



Hình 2.7 *: (a) Hàm f đ- ợc sử dụng để phân chia R^2 vào trong hai vùng.*

(b) Vùng " mặt " được mô hình bằng cách sử dụng bốn mẫu đại số.

Ví dụ 2.1 cho $f = x^2 + y^2 - 4$. Trong tr- ờng hợp này, H đại diện đ- ờng tròn bán kính r =2, tâm đ- ợc đặt đúng ở gốc. Điều này t- ơng ứng tới tập hợp của những điểm (x, y) cho f(x, y) = 0, nh- đ- ợc miêu tả trong *Hình 2.7a*.

Ví dụ 2.2 (khuôn mặt) xem xét việc xây dựng một mô hình của vùng đậm màu trong *Hình 2.7b*. Hãy cho vòng tròn ngoài có bán kính r_1 và tâm đ- ợc đặt tại gốc. Giả thiết " đôi mắt " có bán kính r_2 và r_3 và đ- ợc tâm ở tại (x_2,y_2) và (x_3,y_3), t- ơng ứng cho " miệng " một hình ê-líp với trục chính a và trục phụ b và đ- ợc tâm ở $(0, y_4)$.

Những hàm đ- ợc định nghĩa nh- sau:

$$\begin{aligned} f_1 &= x^2 + y^2 - r_1^2, \\ f_2 &= -\big((x - x_2)^2 + (y - y_2)^2 - r_2^2\big), \\ f_3 &= -\big((x - x_3)^2 + (y - y_3)^2 - r_3^2\big), \\ f_4 &= -\big(x^2/a^2 + (y - y_4)^2/b^2 - 1\big). \end{aligned} \qquad (2.11)$$

Cho f_2, f_3, và f_4, là những ph- ơng trình đ- ờng tròn và hình ê-líp đ- ợc nhân với - 1 để sinh ra những mẫu đại số cho tất cả các điểm bên ngoài đ- ờng tròn hoặc hình ê-líp. Vùng O đậm màu đ- ợc t- ơng ứng nh- sau:

$$\mathcal{O} = H_1 \cap H_2 \cap H_3 \cap H_4. \qquad (2.12)$$

Trong tr- ờng hợp của những mô hình nửa đại số, phép giao của những mẫu không nhất thiết kết quả trong một tập con lồi W. Nói chung, nó có thể cần thiết để hình thành O bởi việc lấy hợp và giao của những mẫu đại số.

Rõ ràng biểu diễn bằng mô hình nửa đại số có thể khái quát hóa dễ dàng tr- ờng hợp không gian 3 chiều.

Dạng đại số nguyên thuỷ của mẫu :

$$H = \{(x, y, z) \in \mathcal{W} \mid f(x, y, z) \leq 0\}, \qquad (2.13)$$

Có thể sử dụng để định nghĩa một biểu diễn của một ch- ớng ngại 3 chiều O và một vị từ lôgíc ■. Những ph- ơng trình *(2.10)* và *(2.13)* đủ để biểu thị bất kỳ mô hình nào cần quan tâm. Có thể định nghĩa mẫu theo nhiều cách khác dựa vào những quan hệ khác nhau, nh- :

f(x, y, z) ■ 0, f(x, y, z) = 0, f(x, y,z) < 0, f(x, y, z) = 0, và f(x, y, z) ■0

Xét mẫu:

$$H = \{(x, y, z) \in \mathcal{W} \mid f(x, y, z) \geq 0\}.$$

có thể biểu diễn theo cách khác nh- - f(x, y, z) ■ 0, và - f có thể đ-ợc xem xét nh- một hàm đa thức mới của x, y, z. Cho một ví dụ qua hệ bằng:

$$H = \{(x, y, z) \in \mathcal{W} \mid f(x, y, z) = 0\}. \qquad (2.15)$$

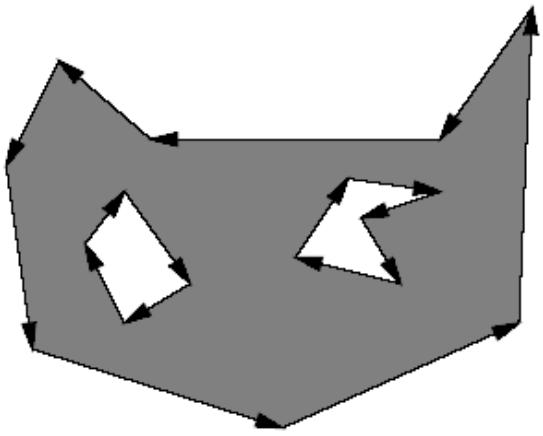
Có thể thay H = H1 ■ H2, với :

$$H_1 = \{(x, y, z) \in \mathcal{W} \mid f(x, y, z) \leq 0\} \qquad (2.16)$$

và

$$H_2 = \{(x, y, z) \in \mathcal{W} \mid -f(x, y, z) \leq 0\}. \qquad (2.17)$$

Quan hệ < tăng thêm sức mạnh có ý nghĩa nào đó khi xây dựng những mô hình không chứa đ-ờng biên ngoài. Chú ý rằng phần đậm màu luôn luôn ở bên trái khi đi theo những mũi tên.



Hình 2.8 : *Một đa giác với những lỗ trống có thể đ-ợc biểu diễn bởi việc sử dụng chu trình khác nhau : Ng-ợc chiều kim đồng hồ cho biên ngoài và thuận chiều kim đồng hồ ở ranh giới giữa phía ngoài lỗ hổng.*

2.3- CÁC PHÉP BIẾN ĐỔI CỦA ROBOT

Xét trong C-không gian 2D một robot có thể quay hoặc tịnh tiến.

1- Phép tịnh tiến: Một robot tĩnh $\mathcal{A} \subset R^2$ đ- ợc tịnh tiến bởi việc sử dụng hai tham số, $x_t, y_t \in R$. $q = (x_t, y_t)$, và h đ- ợc định nghĩa

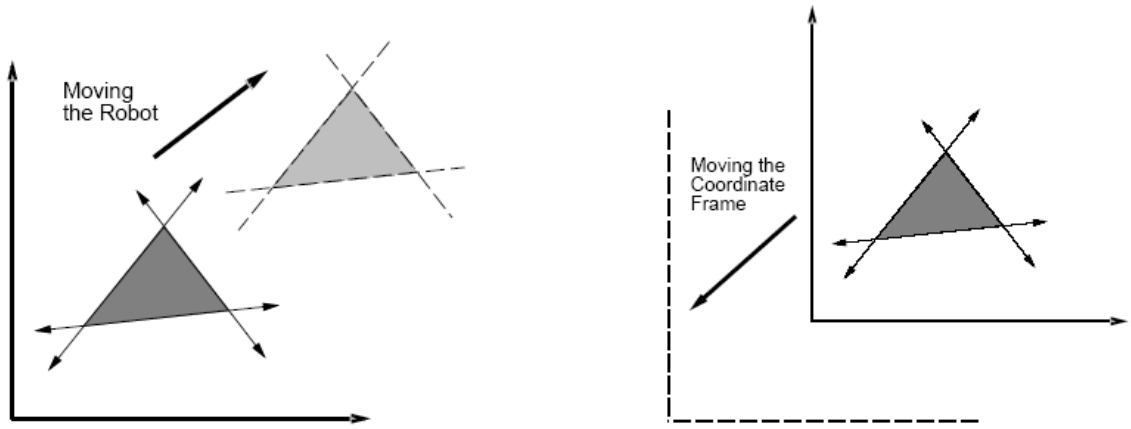
$$h(x, y) = (x + x_t, y + y_t). \qquad (2.18)$$

$\mathcal{A}$ có thể đang tịnh tiến bởi đi một khoảng, khi đó mỗi điểm, (x_i, y_i), lần l- ợt đ- ợc thay thế bằng $(x_i + x_t, y_i + y_t)$.

Trong hình 2.9, có hai cách xem xét sự biến đổi vật thể tĩnh $\mathcal{A}$:

1) Không gian cố định và robot đ- ợc thay đổi.

2) Robot cố định và không gian thay đổi.



(a)) Tịnh tiến Robot *(b) Tịnh tiến khung*

Hình 2.9 - *Hai cách giải thích cho phép tịnh tiến.*

2- Phép quay: Robot, $\mathcal{A}$, có thể đ- ợc quay ng- ợc chiều kim đồng hồ bởi các góc $\theta \in [0, 2\pi)$ bởi ánh xạ mỗi $(x, y) \in \mathcal{A}$ nh- sau:

$$(x, y) \mapsto (x\cos\theta - y\sin\theta,\ x\sin\theta + y\cos\theta). \qquad (2.19)$$

Sử dụng một ma trận quay 2 x 2 :

$$R(\theta) = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}, \qquad (2.20)$$

đ- ợc viết nh- sau:

$$\begin{pmatrix} x\cos\theta - y\sin\theta \\ x\sin\theta + y\cos\theta \end{pmatrix} = R(\theta) \begin{pmatrix} x \\ y \end{pmatrix}. \qquad (2.21)$$

3- Kết hợp phép tịnh tiến và phép quay:

Giả sử sau khi quay với góc [redacted], sau đó tịnh tiến tới x_t, y_t. Điều này có thể sử dụng để đặt robot trong bất kỳ vị trí và sự định h- ớng mong muốn nào. Chú ý rằng những phép tịnh tiến và phép quay theo một chiều. Nếu những thao tác ứng dụng liên tiếp, mỗi (x, y)[redacted] đ- ợc biến đổi :

$$\begin{pmatrix} x\cos\theta - y\sin\theta + x_t \\ x\sin\theta + y\cos\theta + y_t \end{pmatrix}. \qquad (2.22)$$

Phép nhân ma trận sau sinh ra kết quả cho hai thành phần vectơ đầu tiên :

$$\begin{pmatrix} \cos\theta & -\sin\theta & x_t \\ \sin\theta & \cos\theta & y_t \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} x\cos\theta - y\sin\theta + x_t \\ x\sin\theta + y\cos\theta + y_t \\ 1 \end{pmatrix}. \qquad (2.23)$$

Ma trận trung gian 3x3 :

$$T = \begin{pmatrix} \cos\theta & -\sin\theta & x_t \\ \sin\theta & \cos\theta & y_t \\ 0 & 0 & 1 \end{pmatrix}, \qquad (2.24)$$

trình bày một phép quay theo h- ớng tịnh tiến. Ma trận T sẽ đ- ợc quy về nh- một ma trận biến đổi thuần nhất. Đó là điều quan trọng để T đại diện một phép quay theo h- ớng tịnh tiến. Mỗi mẫu có thể đ- ợc biến đổi sử dụng chuyển vị T, kết quả bên

trong một biến đổi không gian 3 chiều của robot. Biến đổi robot đ- ợc biểu thị bởi A(x_t, y_t, ■), và trong tr- ờng hợp này có ba bậc tự do. Ma trận biến đổi thuần nhất là một biểu diễn thuận lợi của những sự biến đổi kết hợp ; Bởi vậy, nó th- ờng xuyên đ- ợc sử dụng trong kỹ thuật rôbôt, máy cơ học, đồ hoạ máy tính, và một số lĩnh vực khác. Nó đ- ợc gọi thuần nhất bởi vì qua R^3 nó là chỉ là một sự biến đổi tuyến tính mà không có bất kỳ tịnh tiến nào. Thủ thuật của việc tăng thêm kích th- ớc để hấp thụ phần tịnh tiến chung trong phép chiếu hình học.

2.4. KHÔNG GIAN CẤU HÌNH CH- ỚNG NGẠI VẬT

Một giải thuật lập lộ trình chuyển động phải tìm thấy một đ- ờng dẫn trong không gian rỗng (Free Space) từ cấu hình ban đầu (qI) đến cấu hình đích (qG). Đầu ch- ơng chúng ta đã có khái niệm sơ khai về cấu hình không gian ch- ớng ngại vật. Bây giờ chúng ta sẽ nghiên cứu chi tiết hơn về vấn đề này.

Vùng ch- ớng ngại vật

Giả thiết không gian $W = R^2$ hoặc $W = R^3$, chứa đựng một vùng ch- ớng ngại $\mathcal{O}$ ■ W. Đồng thời cũng giả thiết $\mathcal{A}$ là một robot cứng, $\mathcal{A}$ ■W, $\mathcal{A}$ và $\mathcal{O}$ đ- ợc trình bày nh- những mô hình nửa đại số (mà bao gồm những mô hình đa diện và đa giác). Cho q■ C biểu thị cấu hình của $\mathcal{A}$, trong đó q= (x_t, y_t,■) với $W = R^2$ và q= (x_t, y_t, z_t, h) với $W = R^3$ (h là đơn vị quaternion).

Vùng ch- ớng ngại, Cobs ■ C, đ- ợc định nghĩa nh- sau:

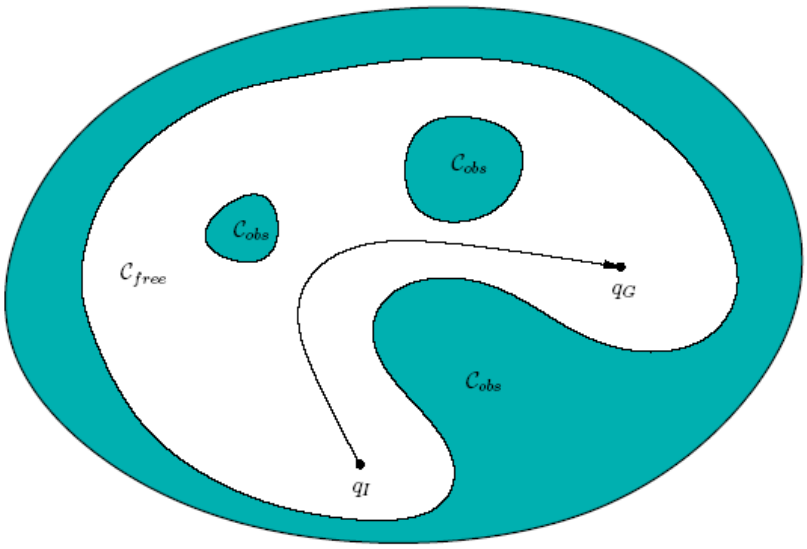
$$\mathcal{C}_{obs} = \{q \in \mathcal{C} \mid \mathcal{A}(q) \cap \mathcal{O} \neq \emptyset\}, \qquad (2.32)$$

C_{obs} là tập hợp của tất cả các cấu hình q, ở đó $\mathcal{A}$(q) (trạng thái của robot tại cấu hình q) giao với vùng ch- ớng ngại $\mathcal{O}$. $\mathcal{O}$ và $\mathcal{A}$(q) là những tập hợp đóng bên trong W, vùng ch- ớng ngại là một tập hợp đóng trong C. Những cấu hình còn lại đ- ợc gọi không gian trống, mà đ- ợc định nghĩa và $C_{free} = C \setminus C_{obs}$. Từ đó C là một không gian tôpô và C_{obs} là đóng, C_{free} phải là một tập hợp mở. Điều đó có nghĩa là robot có thể đến gần những ch- ớng ngại một cách tuỳ ý trong những phần của C_{free} miễn là đ- ờng biên của chúng không giao nhau.

$$\text{int}(\mathcal{O}) \cap \text{int}(\mathcal{A}(q)) = \emptyset \text{ and } \mathcal{O} \cap \mathcal{A}(q) \neq \emptyset, \qquad (2.33)$$

Nếu $\mathcal{A}$ chạm vào $\mathcal{O}$ thì q■ C_{obs}. Điều kiện nhận biết duy nhất là những đ- ờng biên của chúng cắt nhau. Ý t- ởng của robot có thể đến gần những ch- ớng ngại một cách tuỳ ý có thể không có ý nghĩa thực tiễn trong kỹ thuật rôbôt, nh- ng nó làm cho những giải thuật lập lộ trình chuyển động trở nên minh bạch. Khi C_{free} mở, nó không thể đạt đ- ợc sự tối - u nh- tìm kiếm đ- ờng ngắn nhất. Trong tr- ờng hợp này, tập đóng, cl(C_{free}), cần phải thay vào để sử dụng.

2.5- ĐỊNH NGHĨA CHÍNH XÁC VỀ VẤN ĐỀ LẬP LỘ TRÌNH CHUYỂN ĐỘNG:



Hình 2.10 *- Khái niệm về C-space và nhiệm vụ là tìm một đ- ờng từ qI đến qG trong* C_{free} *với* $C = C_{free}$ ■ C_{obs}.

Cuối cùng đã đủ công cụ để định nghĩa chính xác vấn đề lập lộ trình. Vấn đề đ- ợc minh họa trong *Hình 2.11*. Những thành phần chính của vấn đề nh- sau :

1. Một không gian W là một trong hai tr- ờng hợp $W = R^2$ hoặc $W = R^3$.

2. Một vùng ch- ớng ngại là một mô hình nửa đại số $\mathcal{O}$■ W trên không gian.

3. Một robot cũng là một mô hình nửa đại số đ- ợc định nghĩa trong W. Nó có thể là một robot đơn $\mathcal{A}$ hoặc là một tập hợp của m những mối liên kết $\mathcal{A}_1, \mathcal{A}_2, .. \mathcal{A}_m$.

4. Không gian C cấu hình xác định bởi việc chỉ rõ tập hợp của tất cả những sự biến đổi có thể đ- ợc áp dụng cho robot đ- ợc dẫn xuất từ C_{obs} và C_{free}.

5. Trong một cấu hình, qI■ C_{free} là trạng thái ban đầu.

6. Trong một cấu hình, qG■ C_{free} đ- ợc chỉ định là trạng thái đích. Một cặp cấu hình ban đầu và cấu hình đích th- ờng đ- ợc gọi một cặp truy vấn (hoặc truy vấn) và ký hiệu là (qI, qG).

7. Một giải thuật phải tính toán thiết lập đ- ợc một đ- ờng dẫn liên tục đầy đủ từ qI đến qG:

■: [0, 1]■ C_{free}, nh- vậy ■(0) = qI và ■(1) = qG, hoặc phải chỉ ra rằng một đ- ờng dẫn nh- vậy không tồn tại.

2.6. MỘT SỐ MÔ HÌNH C_{OBS}

Quan trọng là làm thế nào để xây dựng một cách trình bày của C_{obs}. Trong một số giải thuật, đặc biệt nh- ph- ơng pháp tổ hợp của Ch- ơng 3, đây là đại diện quan trọng đầu tiên để tìm lời giải cho vấn đề. Trong những giải thuật khác, đặc biệt là lấy mẫu - đặt cơ sở lập cho những giải thuật lập lộ trình, nó giúp cho chúng ta hiểu tại sao những giải thuật lại đ- ợc xây dựng nh- vậy để tránh sự phức tạp của chúng.

2.6.1. MÔ HÌNH C_{OBS} CHO TR□ỜNG HỢP TỊNH TIẾN

Tr- ờng hợp đơn giản nhất để mô tả đặc điểm C_{obs} khi $C = R^n$ với n = 1, 2, 3 và robot chỉ là một thể rắn và chỉ hạn chế bởi phép tịnh tiến. D- ới những điều kiện này, C_{obs} có thể đ- ợc biểu thị nh- một kiểu khúc cuộn.

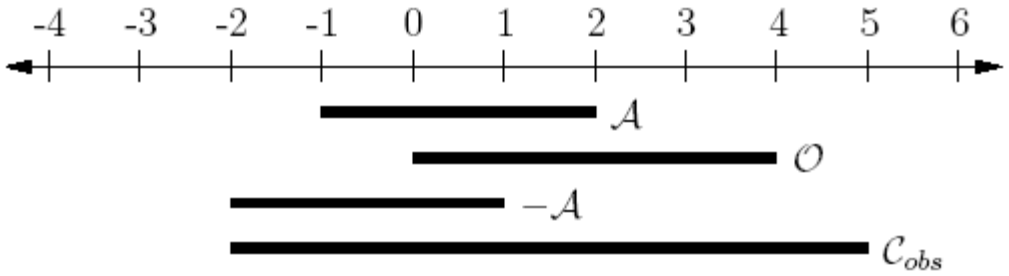
Cho hai tập hợp bất kỳ X, Y■ R^n, *hiệu Minkowski* của chúng đ- ợc định nghĩa nh- sau:

$$X \ominus Y = \{x - y \in \mathbb{R}^n \mid x \in X \text{ and } y \in Y\}, \qquad (2.35)$$

Trong đó x - y là vectơ hiệu trên R^n. Hiệu Minkowski giữa x và y cũng có thể đ- ợc xem xét nh- tổng Minkowski của x và - y.

Tổng Minkowski ■ thu đ- ợc bởi phép cộng đơn giản thêm những phần tử của X và Y trong công thức (4.37), ng- ợc với việc trừ chúng. Tập hợp - Y đ- ợc thu đ- ợc bởi việc thay thế mỗi y■ Y bởi - y.

D- ới dạng hiệu Minkowski, $\mathcal{C}_{obs} = \mathcal{O} \ominus \mathcal{A}(0)$. Để hiểu rõ điều này chúng ta xem xét ví dụ một chiều.



Hình 2.11 : Một ch- ớng ngại không gian C- một chiều

Ví dụ (Ch- ớng ngại C - Không gian một chiều) Trong *Hình 2.11*, cả hai robot $\mathcal{A}$ = [- 1, 2] và vùng $\mathcal{O}$ ch- ớng ngại = [0, 4] là những khoảng trong một không gian một chiều, W = R. Phủ định, - $\mathcal{A}$, của robot đ- ợc là khoảng [- 2, 1]. Cuối cùng, thực hiện tổng Minkowski $\mathcal{O}$ và - $\mathcal{A}$, C_{space} ch- ớng ngại, thu đ- ợc C_{obs} = [- 2, 5].

2.6.1.1- Ch- ớng ngại C - không gian đa giác:

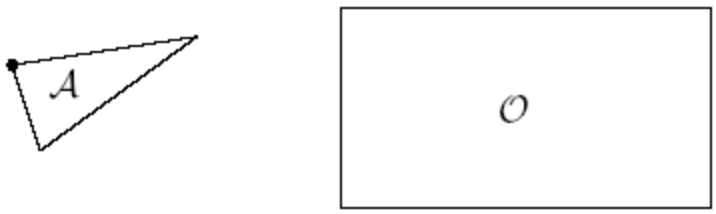
Một giải thuật đơn giản cho việc tính toán C_{obs} tồn tại trong tr- ờng hợp không gian 2D chứa đựng một ch- ớng ngại đa giác lồi $\mathcal{O}$ và một robot đa giác lồi $\mathcal{A}$. Đây th- ờng đ- ợc gọi là *giải thuật ngôi sao*. Trong vấn đề này, C_{obs} cũng là một đa giác lồi.

Ta đã biết những ch- ớng ngại và những robot không lồi có thể đ- ợc mô hình nh- hợp của những thành phần lồi. Nh- vậy những khái niệm sau đây cũng có thể

áp dụng cho những tr- ờng hợp không lồi bởi việc xem xét C_{obs} nh- hợp của những thành phần lồi, từng phần t- ơng ứng tới một thành phần lồi của $\mathcal{A}$ va chạm với một thành phần lồi của $\mathcal{O}$.

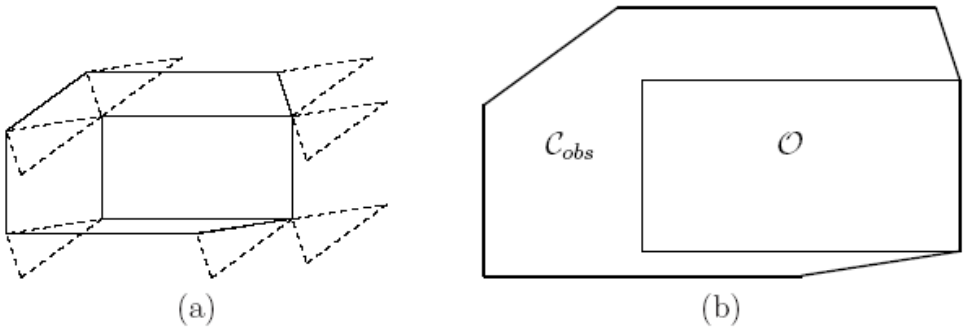
Ph- ơng pháp đ- ợc dựa vào sắp xếp các véctơ pháp tuyến các cạnh của đa giác trên cơ sở những góc của chúng. Khóa xác định mọi cạnh của C_{obs} là một phép tịnh tiến cạnh từ $\mathcal{A}$ hoặc $\mathcal{O}$.

Trong thực tế, mỗi cạnh từ $\mathcal{O}$ và $\mathcal{A}$ đ- ợc sử dụng đúng một lần trong xây dựng cấu trúc của C_{obs}. Vấn đề chỉ là việc xác định rõ thứ tự sắp xếp các cạnh của C_{obs}. Cho n_1, n_2,. . n_n là các vectơ pháp tuyến trong của các cạnh của $\mathcal{A}$ theo thứ tự ng- ợc chiều kim đồng. Gọi $n_1 n_2$...,n_n là các pháp tuyến ngoài của các cạnh $\mathcal{O}$. Sau khi sắp xếp cả hai tập hợp theo ph- ơng song song với các vectơ ta đ- ợc một thứ tự trong S^1. C_{obs} có thể đ- ợc xây dựng thêm những cạnh t- ơng ứng theo thứ tự đã đ- ợc sắp xếp.



Hình 2.12- *Một robot $\mathcal{A}$ tam giác và một ch- ớng ngại hình chữ nhật.*

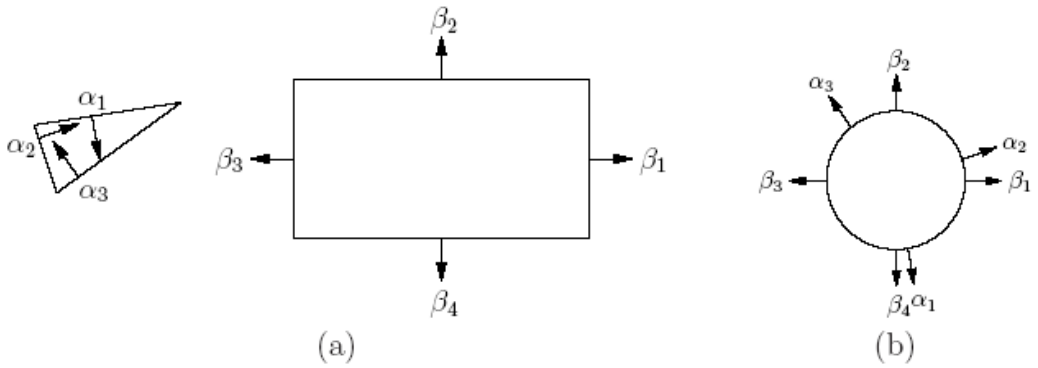
Ví dụ 2.3 (Một robot tam giác và ch- ớng ngại hình chữ nhật) Để hiểu đ- ợc ph- ơng pháp chúng ta quay lại xét tr- ờng hợp một robot tam giác và một ch- ớng ngại hình chữ nhật, trong *Hình 2.12*. Chấm đen trên $\mathcal{A}$ biểu thị gốc khung vật thể của nó. Chúng ta cho robot tr- ợt xung quanh ch- ớng ngại và luôn giữ chúng ở trạng thái tiếp xúc nhau nh- trong *Hình 2.12(a)*.



Hình 2.13: *Xây dựng C_{obs} trong phép tịnh tiến*

(a) Tr- ợt robot xung quanh ch- ớng ngại trong khi giữ chúng tiếp xúc với nhau. (b) Những cạnh theo dấu vết ngoài của $\mathcal{A}$ và C_{obs}.

Điều này phù hợp với việc xem xét tất cả các cạnh của mọi cấu hình bên trong C_{obs} (đ- ờng biên của C_{obs}). Dấu vết ngoài của của $\mathcal{A}$ là những cạnh của C_{obs}, ta nhìn thấy trong *Hình 2.13 b*. có bảy cạnh, và mỗi cạnh t- ơng ứng với một cạnh của $\mathcal{A}$ hoặc một cạnh của $\mathcal{O}$. Những ph- ơng của véc tơ pháp tuyến đ- ợc định nghĩa nh- trong *Hình 2.14a*. Khi sắp xếp nh- trong *Hình 2.14b*, những cạnh của C_{obs} đ- ợc thêm vào cấu trúc.



Hình 2.14 : *(a) Lấy véc tơ pháp tuyến trong của các cạnh của $\mathcal{A}$ và véc tơ pháp tuyến ngoài của $\mathcal{O}$. (b) Sắp xếp các vec tơ pháp tuyến của các cạnh xung quanh S^1 ta đ- ợc thứ tự của những cạnh bên trong C_{obs}.*

Chi phí thời gian của giải thuật là $\mathcal{O}(n + m)$, trong đó n là số cạnh của $\mathcal{A}$, và m là số cạnh của $\mathcal{O}$. Chú ý rằng những góc có thể đ- ợc sắp xếp trong thời gian tuyến tính bởi vì chúng đã đ- ợc sắp xếp theo thứ tự ng- ợc chiều kim đồng hồ quanh $\mathcal{A}$ và

$\mathcal{O}$; khi đó chúng ta chỉ cần chúng chỉ cần dùng thuật toán hòa trộn. Nếu hai cạnh là cộng tuyến, thì chúng có thể đ- ợc đặt end-to-end thành một cạnh đơn của C_{obs}.

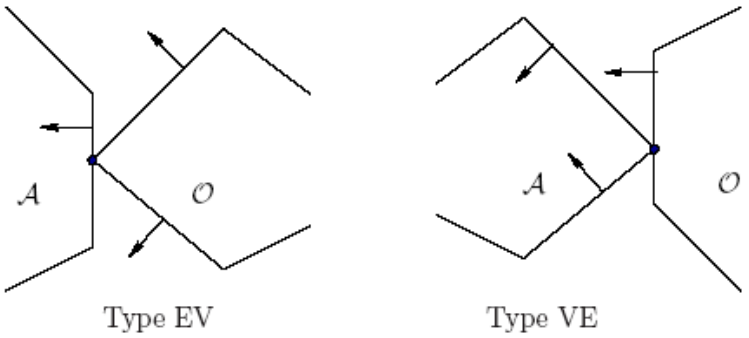
Tính toán đ- ờng biên của C_{obs} : Ph- ơng pháp nhanh chóng xác định cạnh thêm vào cho C_{obs}. Chúng có thể có cấu trúc đặc d- ới dạng nửa -mặt phẳng.

Có hai khác nhau phát sinh cạnh cho C_{obs}, nh- trong *Hình 2.15*.

- *Kiểu tiếp xúc EV* là tr- ờng hợp một cạnh của $\mathcal{A}$ tiếp xúc với một đỉnh của $\mathcal{O}$.

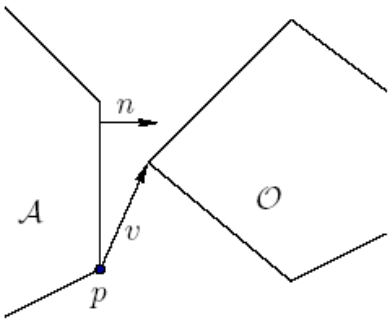
- *Kiểu tiếp xúc VE* là tr- ờng hợp một đỉnh bất kỳ của $\mathcal{A}$ tiếp xúc với một cạnh của $\mathcal{O}$.

Những mối quan hệ giữa các vectơ pháp tuyến với các cạnh cũng đ- ợc thấy trong *Hình 2.15*.



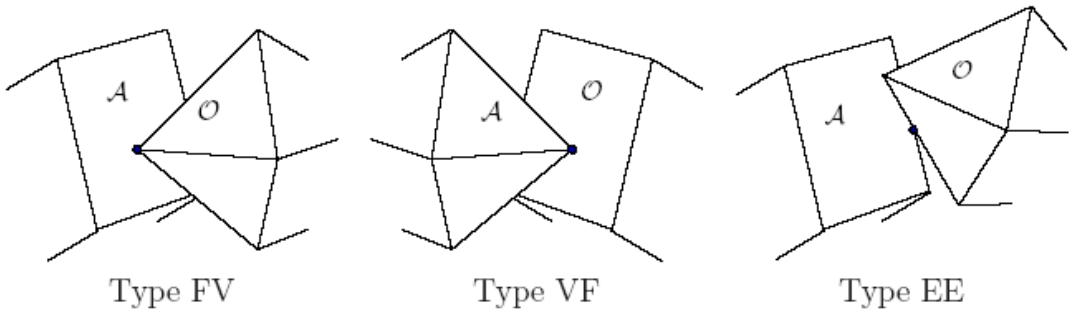
Hình 2.15 :

Hai kiểu va chạm khác nhau, mỗi kiểu phát sinh một loại cạnh khác nhau của Cobs.



Hình 2.16: *Trạng thái va trạm khi n và $\mathcal{A}$ vuông góc.*

2.6.1.2- Một ch- ớng ngại vật C - không gian đa diện: Hầu hết ý t- ởng của không gian đa giác có thể khái quát hóa rất tốt cho tr- ờng hợp một robot đa diện có khả năng chỉ tịnh tiến trong không gian không gian 3 chiều mà chứa đựng những ch- ớng ngại đa diện. Nếu $\mathcal{A}$ và $\mathcal{O}$ là khối đa diện lồi, những kết quả C_{obs} là một đa diện lồi.



Hình 2.17: *Ba kiểu tiếp xúc khác nhau, sinh một các loại C_{obs} khác nhau.*

Có ba loại những tiếp xúc khác nhau trong C:

1. Kiểu FV: Một mặt của $\mathcal{A}$ và một đỉnh của $\mathcal{O}$

2. Kiểu VF: Một đỉnh của $\mathcal{A}$ và một mặt của $\mathcal{O}$

3. Kiểu EE: Một cạnh của $\mathcal{A}$ và một cạnh của $\mathcal{O}$.

Trong *Hình 2.17*. Mỗi nửa - không gian định nghĩa một mặt của đa diện,C_{obs}. Chi phí thời gian để xây dựng C_{obs} là $\mathcal{O}(n + m + k)$, trong n là số mặt của $\mathcal{A}$, m là số mặt của $\mathcal{O}$, và k số mặt của Cobs, thông th- ờng là nm.

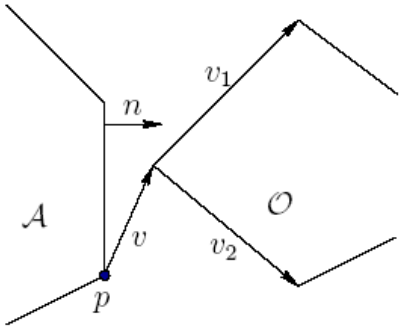
2.6.2. MÔ HÌNH C_{OBS} CHO TR□ỜNG HỢP TỔNG QUÁT

Trong thực tế, những tr- ờng hợp trong đó C_{obs} là đa giác hoặc đa diện thì khá hạn chế. Đa số các vấn đề sinh ra C-space obstacles vô cùng phức tạp. Nh- ng một điểm thuận lợi là C_{obs} có thể sử dụng những mô hình nửa đại số cho bất kỳ robot và ch- ớng ngại nào.

Chúng ta đi xem xét tr- ờng hợp của một robot đa giác lồi và một ch- ớng ngại đa giác lồi trong một không gian 2D. Giả thiết bất kỳ sự biến đổi nào trong SE(2) đều có thể áp dụng cho $\mathcal{A}$; nh- vậy, $C = R^2 \times S^1$ và q = (xt, yt,). Nhiệm vụ sẽ định

nghĩa một tập hợp của những mẫu đại số có thể định nghĩa đ- ợc những mẫu C_{obs}. Một lần nữa, chúng ta thấy việc phân biệt giữa những tiếp xúc kiểu VE và EV là rất quan trọng. Bây giờ chúng ta sẽ xem xét làm thế nào để xây dựng những mẫu đại số cho tiếp xúc kiểu EV ; Kiểu VE có thể đ- ợc xây dựng một cách t- ơng tự.

Hạn chế trong tr- ờng hợp chỉ sử dụng phép tịnh tiến, chúng ta thì có thể xác định tất cả các tiếp xúc kiểu EV bằng việc sắp xếp vectơ pháp tuyến cạnh. Với phép quay, sắp xếp vectơ pháp tuyến của cạnh phụ thuộc vào góc . Điều này có nghĩa là khả năng ứng dụng của một tiếp xúc kiểu EV phụ thuộc vào , sự định h- ớng robot. Quay lại sự ràng buộc vectơ pháp tuyến trong h- ớng vào giữa vectơ pháp tuyến ngoài của cạnh $\mathcal{O}$ chứa đựng đỉnh va chạm, nh- trong *Hình 2.18*. Sự ràng buộc này có thể đ- ợc biểu thị d- ới dạng những tích vô h- ớng của những vectơ v_1 và v_2.



Hình 2.18: *Xây dựng C_{obs} cho phép quay.*

Sự phát biểu l- u tâm tới h- ớng của những vectơ pháp tuyến có thể t- ơng đ- ơng với việc trình bày rõ ràng góc giữa n và v_1, n và v_2, phải nhỏ hơn p/2. Sử dụng những tích vô h- ớng, n $v_1 = 0$ và n $v_2 = 0$. Nh- trong tr- ờng hợp tịnh tiến, điều kiện n v = 0 là điều kiện của sự va chạm. Ta thấy n phụ thuộc vào . Cho bất kỳ q C, nếu n() $v_1 = 0$, n() $v_2 = 0$, và n() v(q) > q, thì q C_{free}. Để cho H_f biểu thị tập hợp của những cấu hình thỏa mãn những điều kiện này có nghĩa là tập các điểm trong C_{free}. Hơn nữa, mọi kiểu khác với hai kiểu tiếp xúc EV và VE với có thể có nhiều điểm hơn trong C_{free}. H_f C_{free}, phần bù của nó $C \setminus H_f$, là một tập chứa C_{obs} (C_{obs} $C \setminus H_f$). Để cho $H_A = C \setminus H_f$. Sử dụng những mẫu:

$$H_1 = \{q \in \mathcal{C} \mid n(\theta) \cdot v_1 \leq 0\}, \tag{2.36}$$

$$H_2 = \{q \in \mathcal{C} \mid n(\theta) \cdot v_2 \leq 0\}, \tag{2.37}$$

$$H_3 = \{q \in \mathcal{C} \mid n(\theta) \cdot v(q) \leq 0\}, \tag{2.38}$$

Để cho $H_A = H_1 \cup H_2 \cup H_3$..

Điều đó cho biết những C_{obs} ■ H_A, trừ phi H_A có chứa điểm trong C_{free}. Tình trạng t- ơng tự để xây dựng một mô hình của một đa giác lồi từ những nửa - mặt phẳng. Trong sự thiết đặt hiện thời, bất kỳ cấu hình nào bên ngoài của H_A phải thuộc trong C_{free}. Nếu H_A giao với tất cả những tập hợp cho mỗi tiếp xúc kiểu EV và kiểu VE, thì kết quả là C_{obs}. Mỗi tiếp xúc có cơ hội để loại bỏ một phần của C_{free} từ sự xem xét. Dần dần, đạt tới mức độ từng phần của C_{free} đ- ợc loại bỏ để những cấu hình duy nhất còn lại phải thuộc trong C_{obs}. Với bất kỳ kiểu va chạm EV nào (H_1■ H_2) \ H_3■C_{free}. Phát biểu t- ơng tự cho những va chạm kiểu VE. Một vị từ lôgíc, có thể xây dựng để xác định q■ C_{obs} với thời gian tuyến tính theo số những mẫu của C_{obs}.

Một vấn đề quan trọng còn lại. Biểu thức n(■) không phải là một đa thức bởi vì cos(■) và sin(■) vẫn là những đối t- ợng trong ma trận quay của $S\mathcal{O}(2)$. Nếu một đa thức có thể là thay thế cho những biểu thức này, thì mọi thứ có thể trở nên cố định vì biểu thức của vectơ pháp tuyến (không phải là một đơn vị chuẩn tắc) và tích vô h- ớng là cả hai hàm tuyến tính, do đó thay đổi đa thức vào trong đa thức. Tuy nhiên, một cách tiếp cận đơn giản hơn là sử dụng những số phức để đại diện phép quay. Khi $a + bi$ đ- ợc sử dụng để đại diện phép quay, mỗi ma trận quay trong $S\mathcal{O}(2)$ đ- ợc đại diện nh- công thức

$$R(a,b) = \begin{pmatrix} a & -b \\ b & a \end{pmatrix}. \tag{2.39}$$

và ma trận 3x3 biến đổi thuần nhất trở thành:

$$T(a,b,x_t,y_t) = \begin{pmatrix} a & -b & x_t \\ b & a & y_t \\ 0 & 0 & 1 \end{pmatrix}. \qquad (2.40)$$

Sử dụng ma trận này để thay đổi một điểm [x y 1] trong những tọa độ điểm ($ax-by+x_t$, $bx+ay+y_t$). Nh- vậy, bất kỳ điểm biến đổi trên $\mathcal{A}$ là một hàm tuyến tính của a, b, x_t, và y_t.

2.7. KẾT LUẬN.

Ch- ơng này trình bày các vấn đề biểu diễn các không gian cấu hình và các phép biến đổi của robot trong không gian đặc biệt là không gian C_{obs}.Đây chính là những kiến thức làm nền tảng cho các ph- ơng pháp lập lộ trình chính xác.

CH- ƠNG III

MỘT SỐ PH- ƠNG PHÁP CHÍNH XÁC LẬP LỘ TRÌNH CHUYỂN ĐỘNG

3.1.GIỚI THIỆU CHUNG

Trong vấn đề lập trình cho robot có rất nhiều những giải thuật lập lộ trình, mỗi giải thuật có những tiềm năng và ứng dụng nhất định. Các ph- ơng pháp có thể bao gồm các ph- ơng pháp lấy mẫu cơ sở, ph- ơng pháp lập lộ trình chính xác.

Cách tiếp cận tổ hợp tới việc lập lộ trình chuyển động để tìm thấy những đ- ờng đi xuyên qua không gian cấu hình liên tục mà không dùng đến những thuật toán xấp xỉ. Nhờ có tính chất này nên cách tiếp cận này còn gọi là giải thuật lập lộ trình chính xác.

Sự biểu diễn quan trọng: Khi nghiên cứu những giải thuật này điều quan trọng là việc xem xét cẩn thận định nghĩa đầu vào :

- Mô hình nào đ- ợc sử dụng để mô hình hoá robot và ch- ớng ngại vật?
- Tập hợp những biến đổi nào đ- ợc áp dụng cho Robot?
- Số chiều của không gian?
- Robot và không gian có thoả mãn tính chất lồi không?
- Chúng có là phân đoạn tuyến tính?

Chỉ định rõ các đầu vào có thể xác định tập các thể hiện của bài toán mà trên đó các thuật toán sẽ tác nghiệp. Nếu những thể hiện có những tính chất thuận lợi nhất định (số chiều của không gian thấp, những mô hình thể hiện là không gian lồi…) thì một giải thuật tổ hợp có thể cung cấp một giải pháp rất tốt và thực tế. Nếu tập hợp của những thể hiện quá rộng, thì một yêu cầu phải thoả mãn cả tính chất trọn vẹn lẫn những giải pháp thực hành có thể vô lý, những công thức chung của vấn đề lập lộ trình chuyển động không thể đạt đ- ợc. Tuy vậy, vẫn tồn tại một số điểm chung để

hoàn thành những giải thuật lập lộ trình chuyển động.

Tại sao cần phải nghiên cứu ph- ơng pháp lập lộ trình tổ hợp?

Có hai lý do cần nghiên cứu những ph- ơng pháp tổ hợp để tiếp cận tới việc lập lộ trình chuyển động, đó là :

Thứ nhất, trong nhiều ứng dụng, việc lập lộ trình chuyển động có thể chỉ quan tâm đến một lớp đặc biệt, ví dụ với không gian chỉ là không gian hai chiều 2D, và robot chỉ có khả năng tịnh tiến. Khi tập hợp nhiều lớp đặc biệt thì những giải thuật thực thi có thể đ- ợc phát triển. Những giải thuật này là đầy đủ, không phụ thuộc vào sự xấp xỉ, và tỏ ra thực hiện tốt hơn hơn những ph- ơng pháp lập lộ trình lấy mẫu cơ sở.

Thứ hai, những ph- ơng pháp tổ hợp vừa đáng chú ý lại vừa thoả mãn cho một lớp rộng những giải thuật lập lộ trình chuyển động.

Tuy nhiên, cũng cần phải chú ý với đa số các ph- ơng pháp có thể thực thi, nh- ng ng- ợc lại còn một số giải thuật còn quá phức tạp và khó ứng dụng trong thực tế. Dù một giải thuật trong lý thuyết cho các chi phí về thời gian thực hiện rất nhỏ, nh- ng trong thực tế khi thực hiện thì nó khó có thể đạt đ- ợc mức chi phí đó. Đôi khi ng- ời ta phải chấp nhận tr- ờng hợp những giải thuật có thời gian chạy xấu hơn nhiều so với giải thuật lý thuyết nh- ng lại dễ hiểu và dễ thực hiện hơn. Đây cũng vẫn là một vấn đề mở để những ng- ời lập trình cần cố gắng xây dựng những giải thuật ngày càng hiệu quả hơn cho dù hiện nay kết quả chủ yếu vẫn là trên lý thuyết. Vì vậy nó luôn thúc đẩy mọi ng- ời tìm kiếm những giải thuật đơn giản và hiệu quả hơn trong thực tế.

Roadmaps

Hầu nh- tất cả cách tiếp cận của việc lập lộ trình chính xác là xây dựng một đ- ờng đi theo một cách nào đó để giải quyết những truy vấn. Khái niệm này đã đ- ợc giới thiệu. Nh- ng trong ch- ơng này yêu cầu Roadmaps phải đ- ợc định nghĩa chính xác hơn bởi vì các giải thuật ở đây cần xây dựng trọn vẹn. Một số giải thuật lập lộ trình tổ hợp đ- ợc tiếp cận theo ý t- ởng tr- ớc hết xây dựng một sự phân ly C_{free} và từ đó sẽ xây dựng lên đ- ờng đi. Một số ph- ơng pháp khác lại trực tiếp xây dựng một

đ- ờng đi mà không xem xét đến sự phân ly ô.

Định nghĩa Roadmap: Cho G là một đồ thị tôpô ánh xạ vào trong C_{free}. Lát cắt S■ C_{free} là tập hợp của tất cả các điểm trong tầm với bởi G, (S= $\bigcup_{e}$■ $e([0,1])$ trong đó e([0,1])■ C_{free} là ảnh của đ- ờng đi e). Đồ thị G đ- ợc gọi một Roadmap nếu nó thỏa mãn hai điều kiện quan trọng sau:

1. Tính dễ tiếp cận : Từ bất kỳ q ■ C_{free}, phải tính toán đ- ợc một đ- ờng đi hiệu quả và đơn giản ■: [0, 1]■■C_{free} sao cho ■(0) = q và ■(1) = s, trong đó s là một điểm bất kỳ trong S. Thông th- ờng, s là điểm gần nhất với q, với giả thiết C là một không gian metric.

2. Duy trì kết nối : Thứ nhất, luôn luôn có thể nối đ- ợc qI và qG tới một vài s_1 và s_2, theo thứ tự trong S. Thứ hai yêu cầu rằng nếu tồn tại một đ- ờng đi ■:[0,1]■■ C_{free} sao cho ■(0) = qI và ■(1) = qG, thì ở đó cũng tồn tại một đ- ờng đi ■' : [0,1] ■■ S, mà ■'(0) = s_1 và ■'(1) = s_2. Nh- vậy, những giải pháp không lỗi bởi vì G luôn giữ liên kết với C_{free}. Điều này bảo đảm rằng phát triển những giải thuật hoàn chỉnh.

Khi thỏa mãn những thuộc tính này, một đ- ờng đi sẽ cung cấp một sự biểu diễn rời rạc cho vấn đề lập lộ trình chuyển động. Một truy vấn, (qI, qG), đ- ợc giải quyết bởi việc nối mỗi truy vấn điểm cho Roadmap và sau đó biểu diễn một sự tìm kiếm đồ thị rời rạc trên G. Duy trì tính chất đầy đủ, là điều kiện đầu tiên bảo đảm rằng bất kỳ truy vấn nào đều có thể đ- ợc nối tới G, và điều kiện thứ hai bảo đảm rằng sự tìm kiếm luôn luôn tiếp nối nếu tồn tại một giải pháp.

3.2. BIỂU DIỄN KHÔNG GIAN CH- ỚNG NGẠI VẬT

Tr- ớc khi nghiên cứu mẫu chung nhất của vấn đề lập lộ trình tổ hợp, chúng ta nên tiếp cận những tr- ờng hợp đơn giản và trực quan với những giải thuật minh bạch cho tr- ờng hợp $C = R^2$ và C_{obs} là đa giác. Hầu hết những điều đó không thể trực tiếp đ- ợc mở rộng tới những không gian có số chiều cao hơn nh- ng những nguyên lý chung vẫn t- ơng tự; Bởi vậy, rất cần thiết để tiếp cận việc lập lộ trình chuyển động chính xác trong không gian hai chiều. Có một vài ứng dụng của những giải thuật này có thể trực tiếp áp dụng. Một ví dụ cho việc lập lộ trình nhỏ cho một

Robot di động có thể đ- ợc mô hình nh- một điểm di động trong một toà nhà mà bản đồ sàn nhà đ- ợc mô hình bằng một đa giác 2D

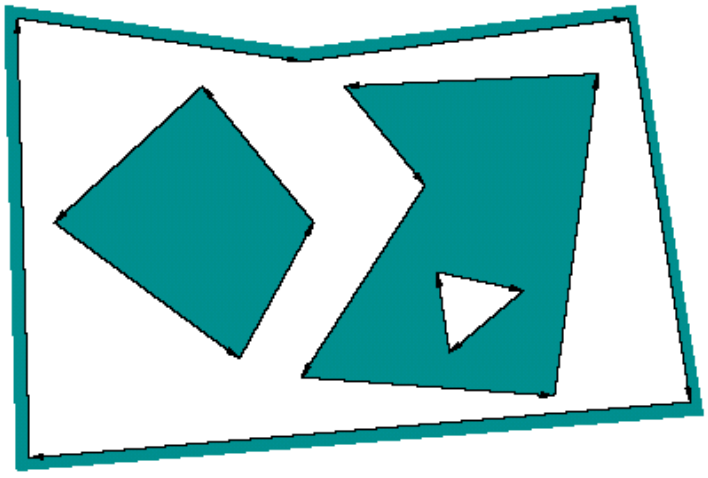
3.2.1 Biểu diễn

Giả thiết :

- W = R^2;
- Những ch- ớng ngại O là đa giác;
- Robot A là một robot điểm chỉ có khả năng tịnh tiến.

Giả sử C_{obs} cũng là đa giác, tr- ờng hợp đặc biệt A là một điểm trong W, những ánh xạ trực tiếp từ O tới C_{obs} không có bất kỳ sự biến dạng nào.

Nh- vậy, những vấn đề ở đây đ- ợc xem xét nh- việc lập lộ trình cho một robot điểm. Nếu A không là robot điểm thì hiệu Minkowski của O và A đ- ợc tính toán. Với tr- ờng hợp cả hai A và O là lồi, giải thuật mô hình C_{obs} rõ ràng trong tr- - ờng hợp tịnh tiến có thể đ- ợc áp dụng tính toán mỗi thành phần của C_{obs}.



Hình 3.1 : *Một mô hình không gian đ- ợc chỉ rõ bởi bốn đa giác có h- ớng đơn giản.*

Trong tr- ờng hợp tổng quát, cả hai A và O có thể là không lồi và thậm chí chúng có thể chứa đựng những lỗ, nh- những C_{obs} trong *hình 3.1*. Trong tr- ờng hợp này, A và O đ- ợc phân ly vào trong những thành phần lồi, và hiệu Minkowski đ- ợc sử dụng để tính toán cho mỗi cặp của những thành phần. Mỗi lần hiệu Minkowski đ- ợc tính toán, chúng cần hợp nhất để thu đ- ợc một cách biểu diễn d- ới dạng những

đa giác đơn giản, nh- những đa giác trong *Hình 3.1*.

Thực thi các giải thuật trong phần này sẽ giúp chúng ta sử dụng một cấu trúc dữ liệu cho phép truy cập thuận tiện vào các thông tin trong mô hình. Để biểu diễn đ- ợc chúng ta cần phải trả lời đ- ợc những câu hỏi:

- Biểu diễn đ- ờng biên ngoài nh- thế nào?

- Những lỗ ở trong những ch- ớng ngại đ- ợc biểu diễn ra sao?

- Làm thế nào chúng ta biết những lỗ nào bên trong những ch- ớng ngại vật?

Những truy vấn này có thể đ- ợc trả lời hiệu quả bởi việc sử dụng cấu trúc dữ liệu danh sách liên thông hai đầu. Chúng ta sẽ cần biểu diễn những mô hình và mọi thông tin khác của những giải thuật lập lộ trình để duy trì trong thời gian thực hiện.

Có ba bản ghi khác nhau :

Đỉnh : Mỗi đỉnh v chứa đựng một con trỏ tới một điểm (x, y)■C = R^2 và một con trỏ tới nửa –cạnh nào đó mà v là gốc của nó.

Mặt : Mỗi mặt có một con trỏ tới một chu trình nửa –cạnh trên biên giới bao quanh mặt; giá trị con trỏ là NIL nếu mặt là biên giới ở ngoài cùng. Mặt cũng chứa đựng một danh sách những con trỏ cho mỗi thành phần có quan hệ (nh- các lỗ) chứa đựng ở trong mặt đó.

Nửa – cạnh: Mỗi Nửa –cạnh đ- ợc định h- ớng để phần ch- ớng ngại luôn luôn ở bên trái nó. Các nửa cạnh luôn luôn đ- ợc xếp trong những chu trình để hình thành ranh giới của một mặt. Những chu trình nh- vậy luôn định h- ớng để phần ch- ớng ngại vật luôn luôn ở bên trái nó. Nó giữ mối liên hệ với nửa cạnh tiếp theo và nửa cạnh tr- ớc nó trong chu trình.

Ví dụ trong *Hình 3.1*, có bốn chu trình của những nửa cạnh mà mỗi chu trình là giới hạn của một mặt khác nhau. Các nửa cạnh luôn theo một h- ớng nhất định nên những chu trình gianh giới ngoài của những ch- ớng ngại vật luôn luôn chạy ng- ợc chiều kim đồng hồ (bên trái), và những chu trình ranh giới của lỗ trống luôn

theo chiều thuận chiều kim đồng hồ.

3.3. MỘT SỐ GIẢI THUẬT LẬP LỘ TRÌNH CHÍNH XÁC CHO ROBOT

Có rất nhiều giải thuật khác lập lộ trình chính xác. Ở đây chúng ta tập chung vào hai giải thuật lập lộ trình chính xác tiêu biểu đó là Visibility Graph và Cell decomposition.

3.3.1 . ROADMAP VISIBILITY GRAPH – ĐỒ THỊ TẦM NHÌN

Với mục đích tìm thấy những đ-ờng đi ngắn nhất cho robot đã dẫn tới ph-ơng pháp Roadmap **Visibility graph**. Ý t-ởng của giải thuật này có thể đ-ợc coi là ví dụ đầu tiên cho một giải thuật lập lộ trình chuyển động.

Roadmap Visibility graph cho phép l-ớt qua những góc của C_{obs}. Trong thực tế, đây là một vấn đề đ-a ra t-ơng đối khó bởi vì C_{free} là một tập hợp mở. Bởi vì với bất kỳ đ-ờng đi nào ■ : [0,1]■ C_{free}, luôn luôn có thể để tìm thấy một đ-ờng ngắn hơn. Do lý do này, chúng ta phải xem xét vấn đề xác định những đ-ờng đi ngắn nhất trong cl(C_{free}) – tập đóng của C_{free}. Điều này có nghĩa rằng robot đ-ợc cho phép "chạm" hoặc " lướt qua " những chướng ngại, nh-ng nó không đ-ợc phép xuyên qua chúng. Trên thực tế ng-ời ta tính toán điều chỉnh một l-ợng nhỏ cho đ-ờng đi giải pháp của lộ trình, để chúng có thể đến rất gần C_{obs} nh-ng không va vào chúng. Điều này làm tăng thêm ở mức độ không đáng kể độ dài đ-ờng dẫn. L-ợng bổ sung có thể đ-ợc làm nhỏ tùy ý cho đ-ờng đi có thể đến gần C_{obs} tuỳ ý.

3.3.1.1. Xây dựng Roadmap Visibility Graph

Giả thiết rằng tất cả đỉnh của một đa giác lồi không có ba đỉnh liên tiếp nào cộng tuyến là đỉnh phản xạ. Đồ thị G với:

■ Các đỉnh là các đỉnh phản xạ.

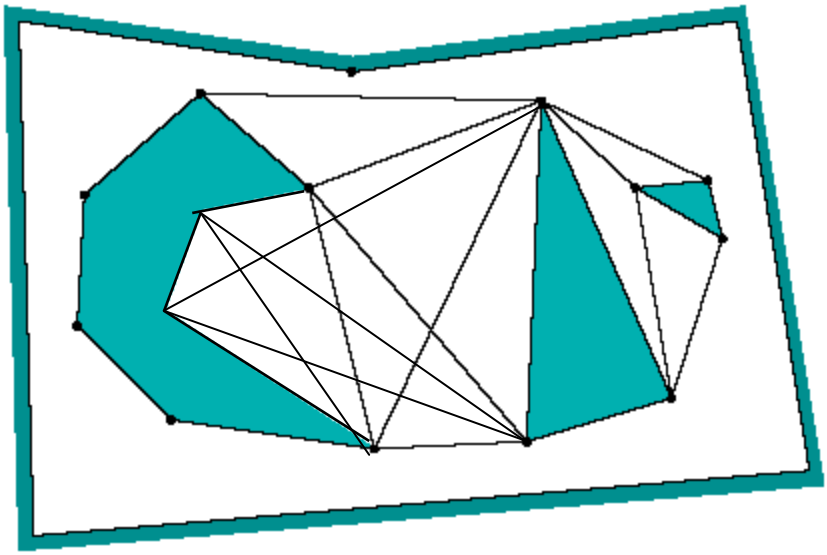
■ Những cạnh của G đ-ợc hình thành từ hai nguồn khác nhau :

- *Đỉnh phản xạ liên tiếp* : Nếu hai đỉnh phản xạ là điểm cuối của một cạnh của C_{obs}, thì một cạnh giữa chúng đ-ợc xây dựng bên trong G.
- *Tiếp tuyến cạnh* : Nếu một đ-ờng tiếp tuyến có thể đ-ợc vẽ xuyên qua một

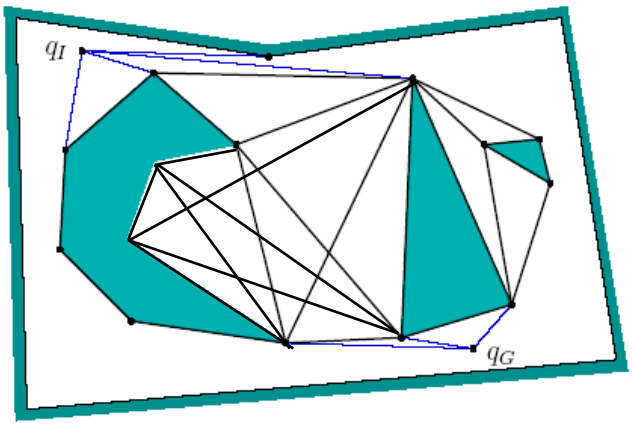
cặp của đỉnh phản xạ, thì một cạnh t- ơng ứng đ- ợc xây dựng bên trong G.

Một đ- ờng tiếp tuyến, là một đ- ờng liên quan tới hai đỉnh phản xạ của C_{obs} và các đỉnh này phải nhìn thấy lẫn nhau.

Một ví dụ của roadmap kết quả đ- ợc cho thấy trong *Hình 3.2*.



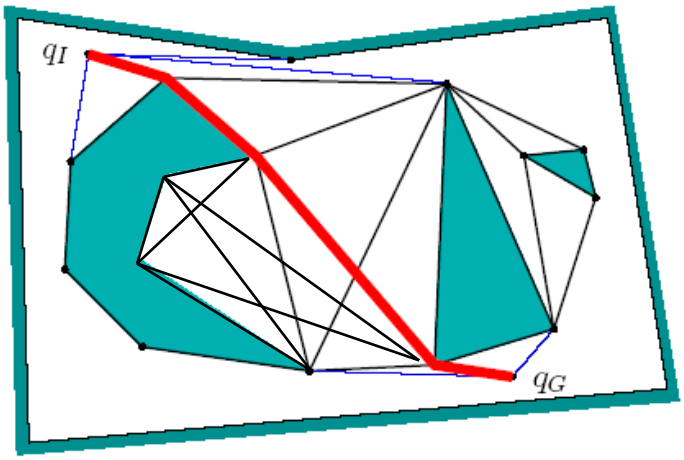
Hình 3.2 : *Xây dựng Roadmap Visibility Graph bao gồm các cạnh giữa đỉnh phản xạ liên tiếp trên C_{obs} và cả những cạnh tiếp tuyến.*



Hình 3. 3 : *qI và qG đ- ợc nối tới tất cả đỉnh có thể thấy của roadmap để tìm kiếm đ- ờng dẫn*

3.3.1.2. Giải pháp cho truy vấn:

qI và qG đ- ợc nối tới tất cả đỉnh roadmap mà có thể thấy trong *Hình 3.3*, chính điều đó làm cho roadmap mở rộng có thể tìm kiếm đ- ợc một giải pháp. Nếu giải thuật của Dijkstra đ- ợc sử dụng, và nếu mỗi cạnh đ- ợc đ- a vào một giá trị t- - ơng ứng thì giá của đ- ờng đi chính là độ dài của đ- ờng đi đó, đ- ờng đi kết quả giải pháp là đ- ờng đi ngắn nhất giữa qI và qG. Đ- ờng đi ngắn nhất cho ví dụ trong *Hình 3.3* đ- ợc cho thấy trong *Hình 3.4*.



Hình 3.4 : *Đ- ờng đi ngắn nhất trong Roadmap là đ- ờng đi ngắn nhất giữa qI và qG.*

Đánh giá giải thuật: Nếu tiếp tuyến kiểm tra đ- ợc thực hiện đơn giản, thì kết quả giải thuật yêu cầu thời gian $O(n^3)$, trong đó n là số đỉnh của C_{obs}. Có $O(n^2)$ những cặp của đỉnh phản xạ cần kiểm tra, và mỗi sự kiểm tra yêu cầu O(n) thời gian kiểm tra nhất định để đảm bảo rằng không có bờ nào khác ngăn chúng nhìn thấy nhau. Nguyên lý **planesweep** có thể đ- ợc làm thích nghi để thu đ- ợc một giải thuật tốt hơn với thời gian cần thiết chỉ là $O(n^2 \lg n)$.

Ý t- ởng thực hiện nguyên lý **planesweep** : Dùng một tia quay quét từ mỗi đỉnh phản xạ, v. Một tia đ- ợc khởi động tại ■ = 0, và những sự kiện xuất hiện khi tia chạm đỉnh. Một tập hợp của tiếp tuyến xuyên qua v có thể đ- ợc tính toán bên trong theo cách này trong thời gian O(nlgn). Từ đó có O(n) đỉnh phản xạ, tổng thời gian thực hiện là $O(n^2 \lg n)$. Ta thấy một giải thuật có thể tính toán đ- ờng dẫn ngắn

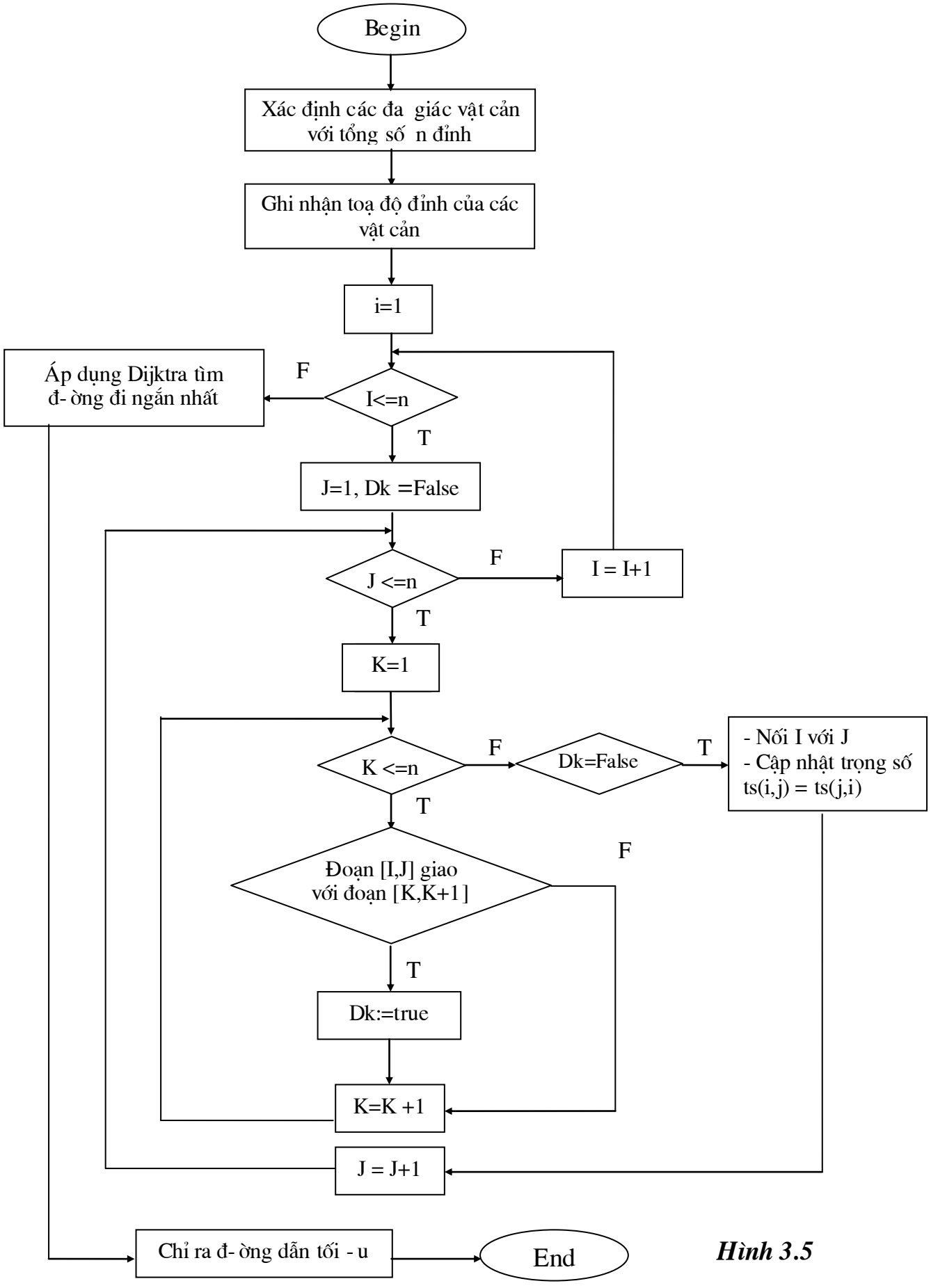
nhất trong thời gian O(nlgn + m), trong đó m là tổng số cạnh trong roadmap. Nếu vùng ch- ớng ngại đ- ợc mô tả bởi một đa giác đơn giản, thì sự nhóm thời gian có thể đ- ợc giảm tới O(n).

3.3.1.3. Ch- ơng trình thử nghiệm và đánh giá

a) Ch- ơng trình thử nghiệm

Ch- ơng trình đ- ợc viết bằng ngôn ngữ Visual Basic (xem phụ lục 1) dựa theo sơ đồ thuật toán nh- (hình 3.5).

SƠ ĐỒ THUẬT TOÁN VISIBILITY GRAPH

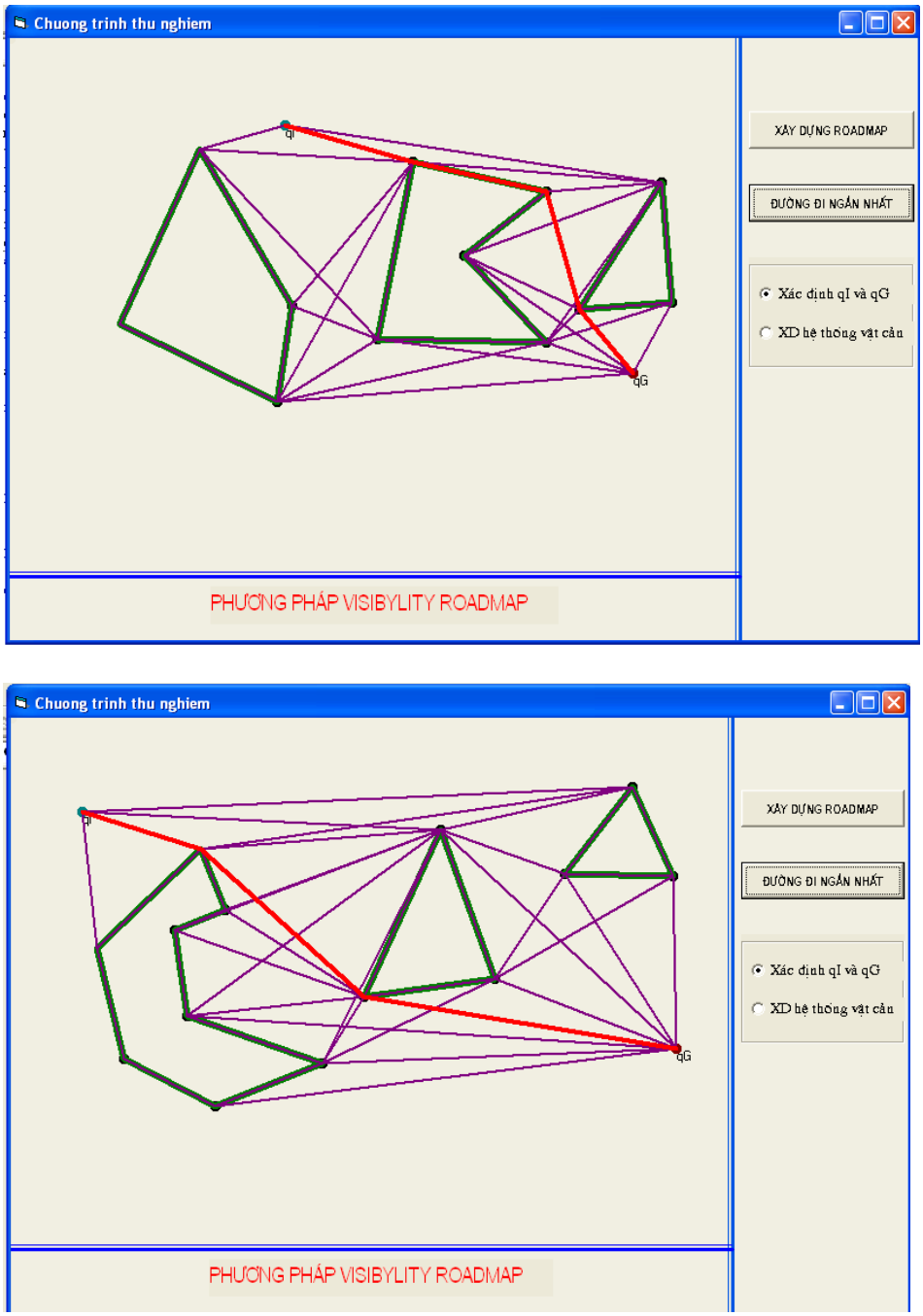


Hình 3.5

b-Kết quả đạt đ- ợc:

Ch- ơng trình cho phép ng- ời sử dụng có thể xây dựng không gian trạng thái bất kỳ. Trên cơ sở đó xây dựng đồ thị tầm nhìn và tìm đ- ợc đ- ờng đi tối - u cho robot từ qI đến qG bất kỳ.

Ch- ơng trình trên đã đ- ợc thử nghiệm với một số không gian cấu hình nh- sau:



Hình 3.6- *Một số đ- ờng đi giải pháp của ch- ơng trình thực nghiệm giải thuật Visibility Graph*

3.3.2. VERTICAL CELL DECOMPOSITION (PHÂN LY Ô DỌC)

Những ph- ơng pháp tổ hợp phải xây dựng một cấu trúc dữ liệu hữu hạn để mã hoá chính xác vấn đề lập lộ trình. Những giải thuật phân ly ô h- ớng tới việc chia cắt C_{free} thành một tập hợp hữu hạn những vùng gọi là những ô.

Sự phân ly ô cần phải thỏa mãn ba thuộc tính :

1.Việc xây dựng một đ- ờng đi từ một điểm bên trong một ô phải dễ dàng. Ví dụ, nếu mỗi ô lồi, thì bất kỳ cặp điểm nào trong một ô đều có thể nối đ- ợc bởi một đoạn thẳng.

2. Sự cung cấp thông tin cho những ô liền kề phải dễ dàng để xây dựng roadmap.

3. Cho tr- ớc một qI và qG, sự phân ly ô cần phải xác định đ- ợc những ô nào chứa chúng.

Nếu một sự phân ly ô thỏa mãn những thuộc tính này, thì vấn đề lập lộ trình chuyển động đ- ợc biến đổi thành vấn đề tìm kiếm đồ thị. Tuy nhiên, trong sự thiết đặt hiện thời, toàn bộ đồ thị, G, thông th- ờng đ- ợc biết tr- ớc. Điều này không giả thiết riêng cho những vấn đề lập lộ trình.

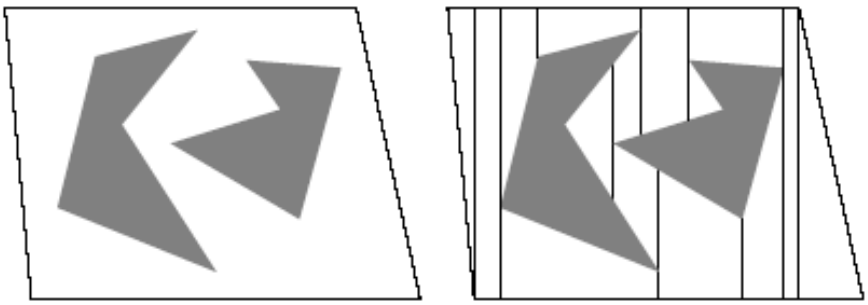
3.3.2.1. Định nghĩa sự phân ly dọc:

Phần này chúng ta giới thiệu một giải thuật mà xây dựng một sự phân ly ô dọc. C_{free} đ- ợc phân chia vào trong một tập hợp hữu hạn của những 2-cell và 1-cell. Mỗi 2 - cell là một hình thang có những cạnh dọc hoặc là một hình tam giác (là một hình thang suy biến). Với lý do này, ph- ơng pháp này còn đ- ợc gọi sự phân ly hình thang.

Sự phân ly đ- ợc định nghĩa nh- sau: Cho P biểu thị tập hợp của đỉnh định nghĩa C_{obs}. Tại mỗi p■ P, dùng những tia thẳng h- ớng h- ớng lên hoặc xuống xuyên qua C_{free}, cho đến khi va phải C_{obs} thì dừng lại.

Hình 3.7 : *Bốn tr- ờng hợp tổng quát của tia phân ly: 1) Có thể theo hai h- ớng xuống xuôi hoặc h- ớng lên, 2) Chỉ h- ớng lên, 3) Chỉ xuôi xuống, và 4) Không thể mở rộng.*

Khi phân ly sẽ có bốn tr- ờng hợp, nh- trong *Hình 3.7*, phụ thuộc vào là nó có thể để mở rộng theo hai ph- ơng h- ớng hay không. Nếu C_{free} đ- ợc phân chia theo những tia này, thì kết quả là một sự phân ly dọc. Ví dụ với C_{obs} trong *Hình 3.7 a* sử dụng sự phân ly dọc C_{free} ta đ- ợc hình *Hình 3.7 b*.



Hình 3.8 : *Sử dụng ph- ơng pháp phân ly ô dọc để xây dựng một roadmap, đ- ợc tìm kiếm để sinh ra một giải pháp cho một truy vấn.*

Chú ý rằng chỉ những hình thang và những hình tam giác thu đ- ợc gọi là những 2 - cell trong C_{free}. Mỗi 1-cell là một đoạn dọc đáp ứng nh- một cạnh giữa hai 2 - cell. Khi phân ly chúng phải bảo đảm rằng topology của C_{free} đ- ợc biểu diễn chính xác.

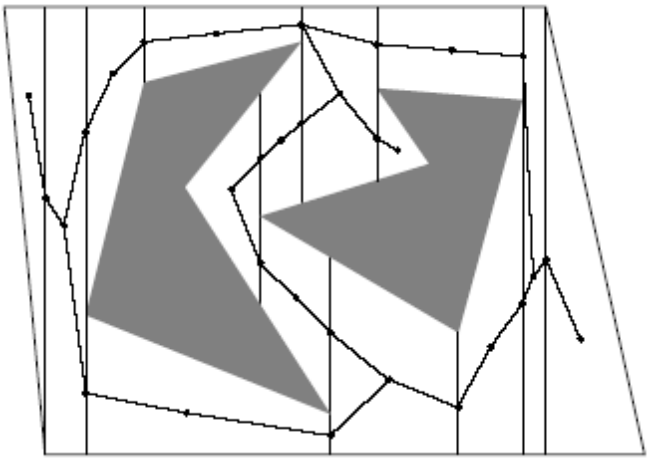
Ta đã biết rằng C_{free} đã đ- ợc định nghĩa là một tập hợp mở. Mỗi 2- cell thật sự cũng đ- ợc định nghĩa là một tập hợp mở trong R^2; nh- vậy, nó là phần trong của một hình thang hoặc hình tam giác và 1- cell là những điểm trên các đoạn thẳng.

3.3.2.2 Định nghĩa roadmap trong ph- ơng pháp

Để điều khiển những truy vấn lập lộ trình chuyển động, một roadmap đ- ợc

xây dựng từ sự phân ly dọc.

Cho mỗi ô C_i, gọi q_i là một đỉnh sao cho $q_i \in C_i$ khi đó q_i đ- ợc gọi là điểm mẫu. Những điểm mẫu có thể đ- ợc lựa chọn theo nhiều cách, ví dụ nh- những trọng tâm trong các ô, nh- ng sự lựa chọn đặc biệt không phải là quá quan trọng, có thể tồn tại nhiều cách lựa chọn điểm mẫu khác.



Hình 3.9 *: Roadmap bắt nguồn từ sự phân ly ô dọc.*

Cho G(V,E) là một đồ thị tôpô định nghĩa nh- sau: Với mỗi ô, C_i, định nghĩa một đỉnh $q_i \in V$. Với mỗi 2- cell, định nghĩa một cạnh từ điểm mẫu đã lựa chọn của nó đến điểm mẫu đã lựa chọn của mỗi 1- cell nằm dọc theo ranh giới của nó. Mỗi cạnh là một đoạn thẳng nối giữa các điểm lựa chọn của các ô. Đồ thị kết quả là một roadmap *(Hình 3.9)*. Điều kiện dễ tiếp cận đ- ợc thỏa mãn bởi vì mỗi điểm mẫu có thể đạt đ- ợc bởi một đ- ờng đi nhờ vào tính lồi của mỗi ô. Điều kiện kết nối cũng đ- ợc thỏa mãn vì G nhận đ- ợc từ sự phân ly ô, mà khi phân ly vẫn giữ quan hệ thuộc C_{free}.

Mỗi lần roadmap xây dựng xong, các thông tin về ô không cần đ- ợc l- u giữ nữa. Đối với việc trả lời cho những truy vấn lập lộ trình chính là roadmap đã xây dựng.

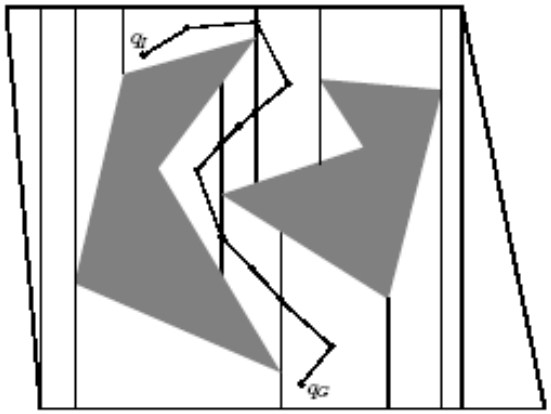
3.3.2.3. Tìm lời giải cho một truy vấn

Một lần roadmap thu đ- ợc, nó có thể trả lời rõ ràng cho một truy vấn của vấn đề lập lộ trình chuyển động từ qI đến qG. Cho C_0 và C_k biểu thị những ô chứa đựng

qI và qG t- ơng ứng. Trong đồ thị G, tìm kiếm một đ- ờng đi có thể nối từ điểm mẫu của C_0 tới điểm mẫu của C_k. Nếu không có đ- ờng đi nh- vậy nào tồn tại, thì giải thuật tuyên bố không tồn tại giải pháp. Nếu tồn tại một đ- ờng đi thì cho C_1, C_2, . . ., C_{k-1} lần l- ợt đi qua tất cả những điểm mẫu của các 1 - cell và 2- cell từ C_0 đến C_k.

Một đ- ờng đi giải pháp có thể đ- ợc hình thành một cách đơn giản bằng cách "*Nối những điểm*", q_0, q_1, q_2, . . ., q_{k-1}, q_k biểu thị những điểm mẫu dọc theo đ- ờng đi bên trong G. Đ- ờng đi giải pháp, ■: [0, 1]■ C_{free}, đ- ợc hình thành bởi việc đặt ■(0) = qI, ■(1) = qG, và việc đến thăm mỗi điểm trong dãy các điểm từ q_0 đến q_k đi theo một đ- ờng đi ngắn nhất. Ví dụ, Giải pháp trong *Hình 3.10*.

Trong việc lựa chọn những điểm mẫu, điều đó quan trọng để bảo đảm rằng mỗi đoạn đ- ờng đi từ điểm mẫu của một ô đến điểm mẫu của những ô láng giềng của nó không có va chạm xảy ra.



Hình 3.10 : *Ví dụ một đ- ờng đi giải pháp*

3.3.2.4. Đánh giá giải thuật: Hiệu quả tính toán sự phân ly sẽ đ- ợc xem xét. Thực chất trong vấn đề này các b- ớc đều đơn giản và thực hiện bởi ph- ơng pháp brute - force (bắt ép thô bạo) . Nếu C_{obs} có n đỉnh, thì cách tiếp cận này cần ít nhất thời gian là $O(n^2)$ vì phải kiểm tra sự giao nhau giữa mỗi tia dọc và mỗi đoạn. Nếu tổ chức cẩn thận các b- ớc tính toán thì kết quả chạy thời gian chỉ còn là O(nlgn).

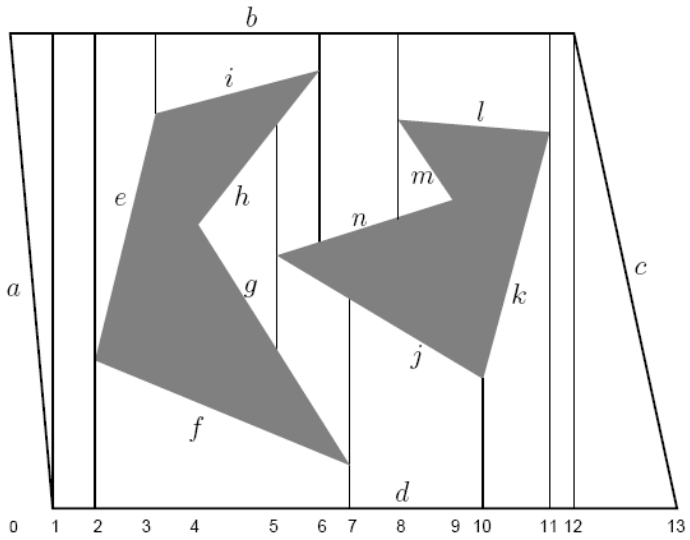
3.3.2.5. Nguyên lý quét mặt phẳng:

Cơ sở của giải thuật là nguyên lý mặt- quét (hoặc đ- ờng - quét) từ mẫu hình

học trên máy tính, đây cũng là cơ sở của nhiều giải thuật lập lộ trình tổ hợp và nhiều giải thuật chung khác. Nhiều vấn đề hình học tính toán bởi máy tính có thể đ- ợc xem xét nh- sự phát triển của những cấu trúc dữ liệu và giải thuật mà khái quát hóa phân loại vấn đề cho nhiều chiều. Nói cách khác, những giải thuật cẩn thận " sắp xếp " các thông tin hình học.

Từ "Quét" được sử dụng khi trình bày những giải thuật này vì có thể hình dung đó là một đ- ờng (hoặc mặt phẳng) quét ngang qua không gian, chỉ dừng khi đạt đến trạng thái thay đổi nào đó khi tìm thấy thông tin.

Trên đây mới là sự miêu tả trực quan, còn việc quét ch- a đ- ợc biểu diễn rõ ràng bằng giải thuật. Để xây dựng sự phân ly dọc, hình dạng một đ- ờng dọc là đ- ờng quét từ x = - ■ tới x =■, giả sử (x, y) là một điểm trong $C = R^2$. Dữ liệu đầu vào là tập hợp P của đỉnh C_{obs}. Bởi vậy chúng ta chỉ quan tâm đến những vấn đề xuất hiện ở những điểm này. Sắp xếp những điểm trong P theo thứ tự toạ độ x tăng dần. Giả thiết thông th- ờng không có hai điểm nào có cùng tọa độ x. Những điểm trong P bây giờ sẽ đ- ợc thăm theo thứ tự đã sắp xếp. Mỗi lần thăm một điểm sẽ đ- ợc coi là một sự kiện. Tr- ớc, sau, và giữa mỗi sự kiện, một danh sách L chứa một vài cạnh C_{obs} sẽ đ- ợc xác nhận. Danh sách này phải đ- ợc duy trì trong suốt thời gian theo thứ tự những cạnh đến khi nào va phải đ- ờng quét dọc. Danh sách đ- ợc sắp xếp theo thứ tự tăng dần.



Hình 3.11 : *Ví dụ có 14 sự kiện.*

Event	Sorted Edges in L	Event	Sorted Edges in L
0	$\{a, b\}$	7	$\{d, j, n, b\}$
1	$\{d, b\}$	8	$\{d, j, n, m, l, b\}$
2	$\{d, f, e, b\}$	9	$\{d, j, l, b\}$
3	$\{d, f, i, b\}$	10	$\{d, k, l, b\}$
4	$\{d, f, g, h, i, b\}$	11	$\{d, b\}$
5	$\{d, f, g, j, n, h, i, b\}$	12	$\{d, c\}$
6	$\{d, f, g, j, n, b\}$	13	$\{\}$

Hình 3.12: *Tình trạng của L đ- ợc chỉ ra sau mỗi 14 sự kiện xuất hiện.*

Những b- ớc thực hiện giải thuật *Hình 3.11 và 3.12* thể hiện sự trình diễn giải thuật. Đầu tiên, L trống rỗng, sau đó với mỗi sự kiện sẽ đ- a vào danh sách những cạnh biểu diễn của C_{free} có liên quan đến sự kiện. Mỗi thành phần liên quan của C_{free} sinh ra một phần tử đơn trong cấu trúc dữ liệu. Sau vài b- ớc lặp ta xây dựng đ- ợc L chính xác. Mỗi sự kiện sẽ xuất hiện là một trong số bốn tr- ờng hợp trong *Hình 3.7*.

Việc duy trì L trong một cây tìm nhị phân cân đối, có thể đ- ợc xác định trong thời gian O(lg n), tốt hơn rất nhiều so với thời gian O(n) của tr- ờng hợp bắt buộc phải kiểm tra mọi đoạn. Việc phụ thuộc vào bốn tr- ờng hợp xuất hiện từ *Hình 3.7*, dẫn tới nh- ng cập nhật khác nhau của L đ- ợc thực hiện.

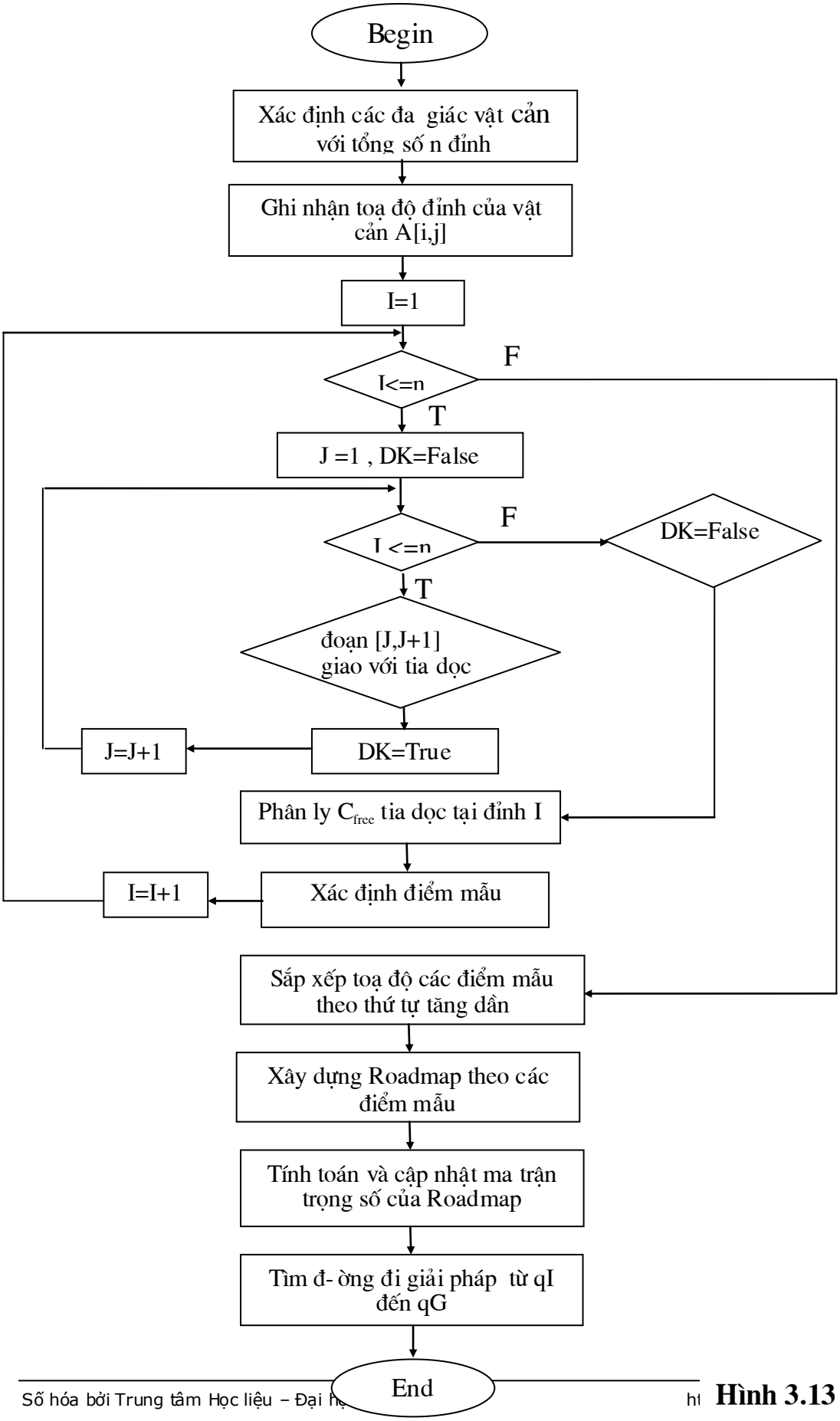
Nếu tr- ờng hợp xuất hiện đầu tiên, thì hai cạnh khác nhau đ- ợc chèn vào, và thể hiện đó trên p là hai nửa đoạn. Hai tr- ờng hợp tiếp theo trong *Hình 3.7* thì đơn giản hơn; chỉ một thể hiện đơn đ- ợc thực hiện.Tr- ờng hợp cuối cùng, không xuất hiện việc phân ly.

Một lần thực hiện thao tác phân ly bề mặt của C_{free} thì L đ- ợc cập nhật. Khi tia quét qua p, luôn luôn có hai cạnh bị ảnh h- ởng. Ví dụ, trong tr- ờng hợp đầu tiên và cuối cùng của *Hình 3.7*, hai cạnh phải đ- ợc chèn vào trong L (ảnh đối xứng của những tr- ờng hợp này là nguyên nhân hai cạnh sẽ đ- ợc xóa đi từ L). Nếu hai tr- ờng hợp giữa xuất hiện, thì một cạnh đ- ợc thay thế bởi đối t- ợng khác trong L. Những thao tác thêm vào và xóa đi này có thể đ- ợc thực hiện trong thời gian O(lgn). Từ đó khi có sự kiện n, thời gian cho xây dựng giải thuật là O(nlgn).

3.3.2.6. Ch- ơng trình thử nghiệm

Ch- ơng trình đ- ợc viết bằng ngôn ngữ Visual Basic (xem phụ lục 2) dựa theo sơ đồ thuật toán nh- (hình 3.13).

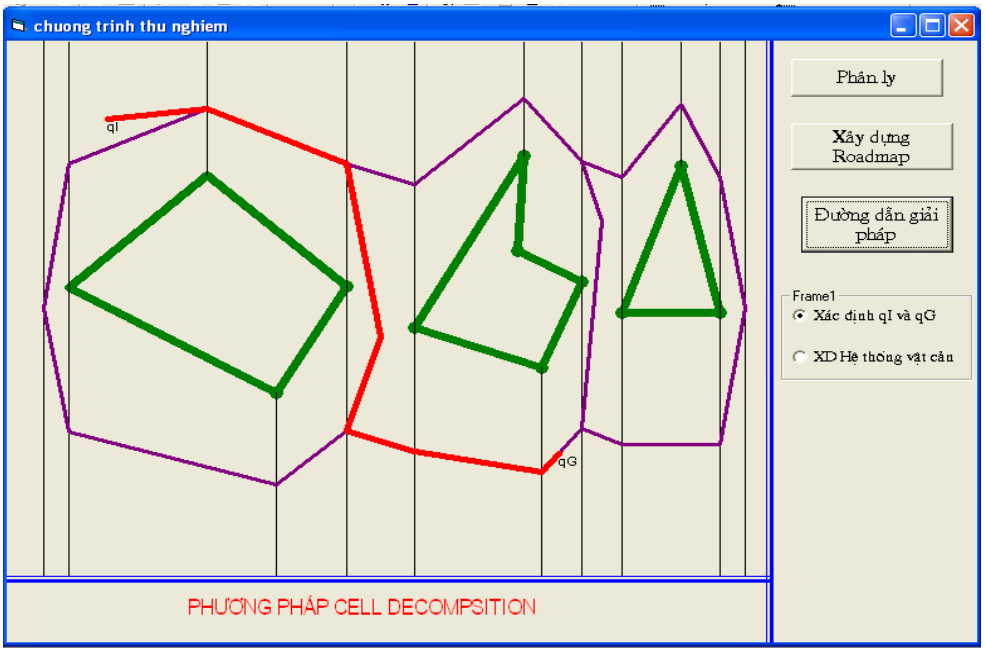
SƠ ĐỒ THUẬT TOÁN CELL DECOMPOSITION

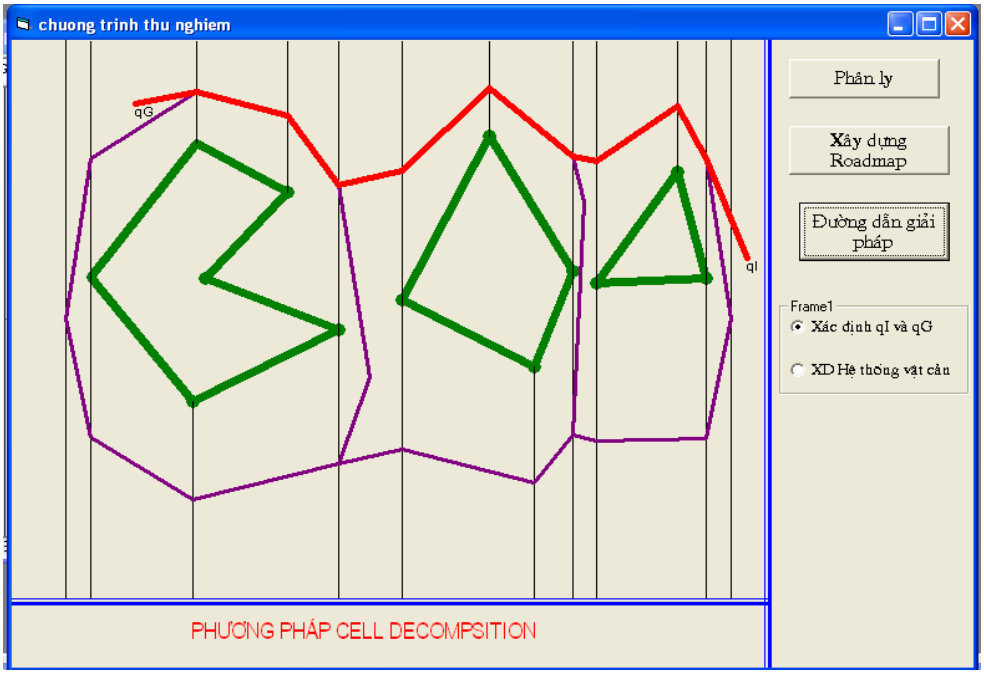


Hình 3.13

Kết quả đạt đ- ợc:

Ch- ơng trình cũng đáp ứng đ- ợc cho không gian trạng thái bất kỳ với đích và nguồn bất kỳ. Ví dụ:





Hình 3.14- *Một số đ- ờng đi giải pháp của ch- ơng trình thực nghiệm giải thuật Cell Decompsition*

KẾT LUẬN

Trong luận văn **" Một số ph- ơng pháp chính xác lập lộ trình lập lộ trình chuyển động cho Robot "** em đã thực hiện đ- ợc một số công việc sau:

1. Đã trình bày đ- ợc tổng quan của bài toán lập lộ trình chuyển động cho robot và các ứng dụng của bài toán trong thực tế.

2. Khái quát đ- ợc cơ sở toán học để biểu diễn không gian cấu hình của bài toán cho các giải thuật lập lộ trình cho Robot .

3. Đi sâu nghiên cứu hai ph- ơng pháp chính xác lập lộ trình chuyển động cho Robot đó là Rodmap Visibility Graph và Vertical Cell Decomposition. Cài đặt thực nghiệm thành công hai ph- ơng pháp trên với không gian trạng thái bất kỳ và tìm đ- ợc đ- ờng đi tối - u cho Robot.

4. H- ớng mở rộng của đề tài này là tiếp tục mở rộng nghiên cứu áp dụng cho các ph- ơng pháp lập lộ trình chính xác trên và một số các ph- ơng pháp khác nh- Vonoroi Diagram hay Cylindrical Algebraic Decomposition trong các không gian trạng thái có số chiều lớn hơn. Mở rộng bài toán sang các phương pháp lấy mẫu cơ sở….

TÀI LIỆU THAM KHẢO

1. A. Abrams and R. Ghrist(2002), *Finding topology in a factory: Configuration spaces*. The American Mathematics Monthly.

2. B. Aronov and M. Sharir(December-1997). *On translational motion planning of a convex polyhedron in 3-space*. SIAM Journal on Computing.

3. G. Dudek, M. Jenkin (2000), *Computational Principles of Mobile Robots*, MIT Press.

4. J.C. Latombe (1991), *Robot Motion Planning*, Kluwer Academic Publishers.

5. J. Angeles. Spatial Kinematic Chains (1982). *Analysis, Synthesis, and Optimisation*. Springer-Verlag, Berlin.

6. J. F. Canny (1988). *The Complexity of Robot Motion Planning*. MIT Press, Cambridge, MA.

7. M. A. Armstrong (1983). *Basic Topology*. *Springer-Verlag*, New York.

8. P. K. Agarwal, B. Aronov, and M. Sharir (1999). *Motion planning for a convex polygon in a polygonal environment*. Discrete and Computational Geometry.

9. Steven M. LaValle (2006), *Planning algorithms*, Cambridge University Press, 842 pages.

PHỤ LỤC 1
CH- ƠNG TRÌNH PH- ƠNG PHÁP VISIBILITY GRAPH

```
Option Explicit
Dim dx, dy, a, b, c, i, j, n, k, k1, k2, s, t, u, v, di, ddx, ddy, dcx, dcy As Integer
Dim maxint, minp As Single
Dim ts(1 To 50, 1 To 50) As Single
Dim truoc(1 To 50), d(1 To 50) As Single
Dim final(1 To 50) As Boolean
Dim Ar(1 To 50), Br(1 To 50), dc(1 To 10) As Integer
Dim x1, x2, y1, y2, xp1, xp2, yp1, yp2, xc, yc As Integer
Dim dk, dk1 As Boolean
----------
Private Sub Command1_Click()
For i = 1 To n
  For j = 1 To n
    ts(i, j) = maxint
  Next j
Next i
For s = 2 To t
For i = 1 To n
  For j = 1 To n
  dk = False
   'Loai truong hop hai diem cung da giac
   If (j <> i + 1) And (i <= dc(s)) And (j <= dc(s)) Then
       dk = True
   End If
   If (j <> i + 1) And (i >= dc(s)) And (j >= dc(s)) Then
     dk = True
   End If    'Các truong hop khac
  For k = 1 To n - 3
    If (i <> j) Then
       x1 = Ar(j) - Ar(i)
      y1 = Br(j) - Br(i)
      x2 = Ar(k + 1) - Ar(k)
      y2 = Br(k + 1) - Br(k)
    If ((y1 * Ar(k) - x1 * Br(k) - Ar(i) * y1 + Br(i) * x1) * (y1 * Ar(k + 1) - x1 * Br(k + 1) - Ar(i) *
y1 + Br(i) * x1) < 0) And ((y2 * Ar(i) - x2 * Br(i) - Ar(k) * y2 + x2 * Br(k)) * (y2 * Ar(j) - x2 *
Br(j) - Ar(k) * y2 + x2 * Br(k)) < 0) Then
       dk = True
    End If    'Loại tr- ờng hợp điểm cuối mỗi đa giác
     k1 = dc(s - 1) + 1
     k2 = dc(s) + 1
     x2 = Ar(k2) - Ar(k1)
     y2 = Br(k2) - Br(k1)
    If ((y1 * Ar(k1) - x1 * Br(k1) - Ar(i) * y1 + Br(i) * x1) * (y1 * Ar(k2) - x1 * Br(k2) - Ar(i) *
y1 + Br(i) * x1) < 0) And ((y2 * Ar(i) - x2 * Br(i) - Ar(k1) * y2 + x2 * Br(k1)) * (y2 * Ar(j) - x2 *
Br(j) - Ar(k1) * y2 + x2 * Br(k1)) < 0) Then
      dk = True
    End If
```

```
    End If
      Next k
    If dk = False And (Ar(j) <> 0) And (Ar(i) <> 0) Then
      DrawWidth = 2
      Line (Ar(i), Br(i))-(Ar(j), Br(j)), &H800080
      ts(i, j) = Sqr((Ar(j) - Ar(i)) * (Ar(j) - Ar(i)) + (Br(j) - Br(i)) * (Br(j) - Br(i)))
      ts(j, i) = Sqr((Ar(j) - Ar(i)) * (Ar(j) - Ar(i)) + (Br(j) - Br(i)) * (Br(j) - Br(i)))
    End If
  Next j
Next i
Next s
End Sub
---------------------------------------------------------------------------------
Private Sub Command3_Click() 'duong di ngan nhat
s = n - 1
t = n ' Khoi tao
For v = 1 To n
    d(v) = ts(s, v)
    truoc(v) = s
    final(v) = False
Next v
truoc(s) = 0
d(s) = 0
final(s) = True
Do While Not (final(t))
    minp = maxint
    For v = 1 To n
        If (Not (final(v))) And (minp > d(v)) Then
            u = v
            minp = d(v)
        End If
    Next v
    final(u) = True
    If Not (final(t)) Then
        For v = 1 To n
            If (Not (final(v))) And (d(u) + ts(u, v) < d(v)) Then
                d(v) = d(u) + ts(u, v)
                truoc(v) = u
            End If
        Next v
  End If
Loop
DrawWidth = 4
i = truoc(t)
Line (Ar(t), Br(t))-(Ar(i), Br(i)), &HFF&
Do While i <> s
  Line (Ar(i), Br(i))-(Ar(truoc(i)), Br(truoc(i))), &HFF&
  i = truoc(i)
Loop
End Sub
Private Sub Form_Load()
maxint = 1000000
```

```
n = 1
t = 1
di = 1
dc(1) = 0
End Sub
-----------------------------------------------------------------------------------------
Private Sub Form_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As
Single)
If O2.Value = True Then
  Select Case Button
    Case vbLeftButton
       If c <> 2 Then
          Circle (X, Y), 30
          a = X
          b = Y
          c = 2
          dx = a
          dy = b
         Ar(n) = a
         Br(n) = b
       Else
          n = n + 1
          Circle (X, Y), 30
          DrawWidth = 6
          Line (a, b)-(X, Y), QBColor(2)
          a = X
          b = Y
          Ar(n) = a
          Br(n) = b
       End If
    Case vbRightButton
       c = 1
        DrawWidth = 6
       Line (dx, dy)-(a, b), QBColor(2)
       n = n + 1
       Ar(n) = dx
       Br(n) = dy
       t = t + 1
       dc(t) = n
       n = n + 1
    End Select
 Else
    If O1.Value = True Then
       Select Case Button
          Case vbLeftButton
             Circle (ddx, ddy), 30, &H8000000F
             Circle (X, Y), 30, QBColor(3)
             ddx = X
             ddy = Y
             Print "ql"
             Ar(n) = ddx
             Br(n) = ddy
```

```
            n = n + 1
          Case vbRightButton
            Circle (dcx, dcy), 30, &H8000000F
            Circle (X, Y), 30, QBColor(4)
            dcx = X
            dcy = Y
            Print "qG"
            Ar(n) = dcx
            Br(n) = dcy
          End Select
      Else
          MsgBox " Hay chon nut..."
    End If
  End If
  End Sub
```

PHỤ LỤC 2
CH- ƠNG TRÌNH PH- ƠNG PHÁP CELL DECOMPSITION

```
Option Explicit
Dim dx, dy, a, b, c, i, j, n, k, k1, k2, s, t, u, v, td1, td2, ql, qG As Integer
Dim maxint, minp, tgx, tgy, tg1, tg2, ddx, ddy, dcx, dcy, tdy1max, tdy2min, tdxmax, tdxmin
As Single
Dim tsxmax(1 To 5), tsymax(1 To 5), tsxmin(1 To 5), tsymin(1 To 5), dn1(1 To 5), dn2(1
To 5) As Single
Dim ts(1 To 60, 1 To 60) As Single
Dim truoc(1 To 60), d(1 To 60), dinh(1 To 60) As Single
Dim final(1 To 50) As Boolean
Dim Ar(1 To 60), Br(1 To 60), dc(1 To 10) As Single
Dim x1, x2, y1, y2, xp1, xp2, yp1, yp2, xc, yc As Integer
Dim tdx1(1 To 60), tdy1(1 To 60), tdx2(1 To 60), tdy2(1 To 60) As Single
Dim dk, dk1 As Boolean
-------------------------------------------------------------------------------------------
-
Private Sub Command1_Click() ' phan ly
DrawWidth = 2
n = n - 1
For s = 2 To t
  For i = 1 To n
    dk = False
    For j = 1 To n - 1
      If (j <> dc(s)) Then
        If ((Ar(j) - Ar(i)) * (Ar(j + 1) - Ar(i)) < 0) And ((Br(i) > Br(j)) Or (Br(i) > Br(j + 1))) Then
          dk = True
          dk1 = True
        End If
      End If
    Next j
 If dk1 = False Then 'loai truong hop canh noi giua hai da giac
     k1 = dc(s - 1) + 1
```

```
        k2 = dc(s) + 1
        If ((Ar(k2) - Ar(i)) * (Ar(k1) - Ar(i)) < 0) And ((Br(i) > Br(k2)) Or (Br(i) > Br(k1))) Then
           dk = False
        End If
      End If
    If (i <> 1) And (i <> n) And (dk = False) Then 'Phan ly va cap nhat ma tran trung diem 1
        DrawWidth = 1
        Line (Ar(i), Br(i))-(Ar(i), 0)
        tdx1(td1) = Ar(i)
        tdy1(td1) = Br(i) / 2
        td1 = td1 + 1
    End If
Next i
Next s
    Line (tdx1(1) - 300, 0)-(tdx1(1) - 300, 7000) 'phan ly ngoai vung da giac
    tdxmin = tdx1(1) - 300
    tdx1(td1) = tdxmin
    tdy1(td1) = 7000 / 2
    td1 = td1 + 1
'**************************************
For i = 1 To n
dk = False
dk1 = False
    For j = 1 To n - 1
    If (j <> dc(s)) Then
       If ((Ar(j) - Ar(i)) * (Ar(j + 1) - Ar(i)) < 0) And ((Br(i) < Br(j)) Or (Br(i) < Br(j + 1))) Then
         dk = True
         dk1 = True
       End If
     End If
      Next j
  If dk1 = False Then      'loai truong hop canh noi giua hai da giac
       k1 = dc(s - 1) + 1
       k2 = dc(s) + 1
       If ((Ar(k2) - Ar(i)) * (Ar(k1) - Ar(i)) < 0) And ((Br(i) > Br(k2)) Or (Br(i) > Br(k1))) Then
          dk = False
       End If
   End If
If (i <> 1) And (i <> n) And (dk = False) Then  'Phan ly va cap nhat ma tran trung diem 2
    DrawWidth = 1
    Line (Ar(i), Br(i))-(Ar(i), 7000)
    tdx2(td2) = Ar(i)
    tdy2(td2) = Br(i) + (7000 - Br(i)) / 2
    td2 = td2 + 1
End If
Next  i
td1 = td1 - 1'Sap xep mang td1 va tim tung do lon nhat
 tdy1max = tdy1(2)
For i = 3 To td1
   If tdy1(i) > tdy1max Then
      tdy1max = tdy1(i)
   End If
```

```
Next i
  For i = 1 To td1 - 1
    For j = i + 1 To td1
      If tdx1(i) > tdx1(j) Then
          tgx = tdx1(i)
          tdx1(i) = tdx1(j)
          tdx1(j) = tgx

          tgy = tdy1(i)
          tdy1(i) = tdy1(j)
          tdy1(j) = tgy
      End If
    Next j
  Next i
    td2 = td2 – 1 'Sap xep mang td2 va tim tung do nho nhat
  tdy2min = tdy2(1)
  For i = 1 To td2 - 1
   If tdy2(i) < tdy2min Then
       tdy2min = tdy2(i)
    End If
  Next i
   For i = 1 To td2 - 1
    For j = i + 1 To td2
      If tdx2(i) > tdx2(j) Then
          tgx = tdx2(i)
          tdx2(i) = tdx2(j)
          tdx1(j) = tgx
          tgy = tdy2(i)
          tdy2(i) = tdy2(j)
          tdy2(j) = tgy
      End If
    Next j
  Next i
Line (tdx2(td2) + 300, 0)-(tdx2(td2) + 300, 7000) 'phan ly ngoai
    td2 = td2 + 1
    tdxmax = tdx2(td2 - 1) + 300
    tdx2(td2) = tdx2(td2 - 1) + 300
    tdy2(td2) = 7000 / 2
For s = 2 To t ' Tim toa do giua cac vat can
    tsxmax(s) = Ar(s)
    tsxmin(s) = Ar(dc(s))
  For i = dc(s - 1) + 1 To dc(s)
     If Ar(i) > tsxmax(s) Then
           tsxmax(s) = Ar(i)
           tsymax(s) = Br(i)
       End If
       If Ar(i) < tsxmin(s) Then
           tsxmin(s) = Ar(i)
           tsymin(s) = Br(i)
      End If
    Next i
  Next s
```

```
End Sub
-----------------------------------------------------------------------------------------
Private Sub Command3_Click() ' xay dung do thi
For i = 1 To td1 + td2' Khoi tao mang trong so
  For j = 1 To td1 + td2
    ts(i, j) = maxint
  Next j
Next i
DrawWidth = 3
If ddx < tdx1(1) Then
  If ddy < tdy1max Then
    If Sqr((tdx1(1) - ddx) * (tdx1(1) - ddx) + (tdy1(1) - ddy) * (tdy1(1) - ddy)) < Sqr((tdx1(2) -
ddx) * (tdx1(2) - ddx) + (tdy1(2) - ddy) * (tdy1(2) - ddy)) Then
      Line (ddx, ddy)-(tdx1(1), tdy1(1)), QBColor(5)
      ql = 1
    Else
      Line (ddx, ddy)-(tdx1(2), tdy1(2)), QBColor(5)
      ql = 2
    End If
   Else
   If Sqr((tdx2(1) - ddx) * (tdx2(1) - ddx) + (tdy2(1) - ddy) * (tdy2(1) - ddy)) < Sqr((tdx2(2) -
ddx) * (tdx2(2) - ddx) + (tdy2(2) - ddy) * (tdy2(2) - ddy)) Then
     Line (ddx, ddy)-(tdx2(1), tdy2(1)), QBColor(5)
      ql = 1
    Else
      Line (ddx, ddy)-(tdx2(2), tdy2(2)), QBColor(5)
      ql = 2
    End If
   End If
 Else
  If ddx >= tdx2(td2) Then
    If ddy < tdy1max Then
     If Sqr((tdx1(td1) - ddx) * (tdx1(td1) - ddx) + (tdy1(td1) - ddy) * (tdy1(td1) - ddy)) <
Sqr((tdx1(td1 - 1) - ddx) * (tdx1(td1 - 1) - ddx) + (tdy1(td1 - 1) - ddy) * (tdy1(td1 - 1) - ddy))
Then
        Line (ddx, ddy)-(tdx1(td1), tdy1(td1)), QBColor(5)
        ql = td1
     Else
        Line (ddx, ddy)-(tdx1(td1 - 1), tdy1(td1 - 1)), QBColor(5)
        ql = td1 - 1
     End If
    Else
    If Sqr((tdx2(td2) - ddx) * (tdx2(td2) - ddx) + (tdy2(td2) - ddy) * (tdy2(td2) - ddy)) <
Sqr((tdx2(td2 - 1) - ddx) * (tdx2(td2 - 1) - ddx) + (tdy2(td2 - 1) - ddy) * (tdy2(td2 - 1) - ddy))
Then
        Line (ddx, ddy)-(tdx2(td2), tdy2(td2)), QBColor(5)
        ql = td2
     Else
        Line (ddx, ddy)-(tdx2(td2 - 1), tdy2(td2 - 1)), QBColor(5)
        ql = td2 - 1
     End If
    End If
```

```
   Else    i = 1
  If ddy < tdy1max Then
      Do While tdx1(i) < ddx
           i = i + 1
      Loop
      If dcx > tdx1(i) Then
         Line (ddx, ddy)-(tdx1(i), tdy1(i)), QBColor(5)
         ql = i
      Else
         Line (ddx, ddy)-(tdx1(i - 1), tdy1(i - 1)), QBColor(5)
         ql = i - 1
      End If
  Else
     Do While tdx2(i) < ddx
           i = i + 1
     Loop
     If dcx > tdx2(i) Then
          Line (ddx, ddy)-(tdx2(i), tdy2(i)), QBColor(5)
          ql = i
      Else
        Line (ddx, ddy)-(tdx2(i - 1), tdy2(i - 1)), QBColor(5)
        ql = i - 1
      End If
 End If
 End If
End If
 '*****************************************
k1 = 1
For  i = 1 To td1 – 1 'noi cac diem mau
   Line (tdx1(i), tdy1(i))-(tdx1(i + 1), tdy1(i + 1)), QBColor(5)
    ts(k1, k1 + 1) = Sqr((tdx1(i + 1) - tdx1(i)) * (tdx1(i + 1) - tdx1(i)) + (tdy1(i + 1) - tdy1(i)) *
(tdy1(i + 1) - tdy1(i)))
    ts(k1 + 1, k1) = Sqr((tdx1(i + 1) - tdx1(i)) * (tdx1(i + 1) - tdx1(i)) + (tdy1(i + 1) - tdy1(i)) *
(tdy1(i + 1) - tdy1(i)))
    k1 = k1 + 1
Next  i
Line (tdx1(td1), tdy1(td1))-(tdx2(td2), tdy2(td2)), QBColor(5)
    ts(k1, k1 + 1) = Sqr((tdx2(td2) - tdx1(td1)) * (tdx2(td2) - tdx1(td1)) + (tdy2(td2) -
tdy1(td1)) * (tdy2(td2) - tdy1(td1)))
    ts(k1 + 1, k1) = Sqr((tdx2(td2) - tdx1(td1)) * (tdx2(td2) - tdx1(td1)) + (tdy2(td2) -
tdy1(td1)) * (tdy2(td2) - tdy1(td1)))
    k1 = k1 + 1
 Line (tdx1(1), tdy1(1))-(tdx2(1), tdy2(1)), QBColor(5)
For i = 1 To td2 - 1
   Line (tdx2(i), tdy2(i))-(tdx2(i + 1), tdy2(i + 1)), QBColor(5)
      ts(k1, k1 + 1) = Sqr((tdx2(i + 1) - tdx2(i)) * (tdx2(i + 1) - tdx2(i)) + (tdy2(i + 1) - tdy2(i)) *
(tdy2(i + 1) - tdy2(i)))
      ts(k1 + 1, k1) = Sqr((tdx2(i + 1) - tdx2(i)) * (tdx2(i + 1) - tdx2(i)) + (tdy2(i + 1) - tdy2(i)) *
(tdy2(i + 1) - tdy2(i)))
      k1 = k1 + 1
Next i
k = 1'Ghi diem ngat
```

```
i = 1
 For s = 2 To t - 1
  Do While tdx1(i) <> tsxmax(s) And (i < k1)
    i = i + 1
  Loop
  dn1(k) = i
  i = 1
  Do While (tdx2(i) <> tsxmax(s)) And (i < k1)
    i = i + 1
  Loop
  dn2(k) = i
 k = k + 1
Next s
'*************************************************
 'Noi diem dich  voi do thi
 If dcx > tdx2(td2) Then 'diem dich nam sau
   If dcy < tdy1max Then 'diem cuoi nam tren
    If Sqr((tdx1(td1) - dcx) * (tdx1(td1) - dcx) + (tdy1(td1) - dcy) * (tdy1(td1) - dcy)) <
Sqr((tdx1(td1 - 1) - dcx) * (tdx1(td1 - 1) - dcx) + (tdy1(td1 - 1) - dcy) * (tdy1(td1 - 1) - dcy))
Then
      Line (dcx, dcy)-(tdx1(td1), tdy1(td1)), QBColor(5)
      qG = td1
   Else
      Line (dcx, dcy)-(tdx1(td1 - 1), tdy1(td1 - 1)), QBColor(5)
      qG = td1 - 1
   End If
   Else
   If Sqr((tdx2(td2) - dcx) * (tdx2(td2) - dcx) + (tdy2(td2) - dcy) * (tdy2(td2) - dcy)) <
Sqr((tdx2(td2 - 1) - dcx) * (tdx2(td2 - 1) - dcx) + (tdy2(td2 - 1) - dcy) * (tdy2(td2 - 1) - dcy))
Then
      Line (dcx, dcy)-(tdx2(td2), tdy2(td2)), QBColor(5)
      qG = td2
   Else
      Line (dcx, dcy)-(tdx2(td2 - 1), tdy2(td2 - 1)), QBColor(5)
      qG = td2 - 1
   End If
   End If
  Else
 If dcx <= tdx1(1) Then ''diem dich nam truoc
   If dcy < tdy1max Then
    If Sqr((tdx1(1) - dcx) * (tdx1(1) - dcx) + (tdy1(1) - dcy) * (tdy1(1) - dcy)) < Sqr((tdx1(2) -
dcx) * (tdx1(2) - dcx) + (tdy1(2) - dcy) * (tdy1(2) - dcy)) Then
     Line (dcx, dcy)-(tdx1(1), tdy1(1)), QBColor(5)
     qG = 1
    Else
      Line (dcx, dcy)-(tdx1(2), tdy1(2)), QBColor(5)
      qG = 2
    End If
   Else
   If Sqr((tdx2(1) - dcx) * (tdx2(1) - dcx) + (tdy2(1) - dcy) * (tdy2(1) - dcy)) < Sqr((tdx2(2) -
dcx) * (tdx2(2) - dcx) + (tdy2(2) - dcy) * (tdy2(2) - dcy)) Then
     Line (dcx, dcy)-(tdx2(1), tdy2(1)), QBColor(5)
```

```
    qG = 1
   Else
     Line (dcx, dcy)-(tdx2(2), tdy2(2)), QBColor(5)
     qG = 2
   End If
  End If

 Else 'diem dich nam trong vung phan ly
  j = 1
  If dcy < tdy1max Then
      Do While (tdx1(j) < dcx) And (j < td1)
           j = j + 1
      Loop
      If ddx > tdx1(j) Then
         Line (dcx, dcy)-(tdx1(j), tdy1(j)), QBColor(5)
         qG = j
      Else
         Line (dcx, dcy)-(tdx1(j - 1), tdy1(j - 1)), QBColor(5)
         qG = j - 1
      End If
  Else
     Do While (tdx2(j) < dcx) And (j <= td2)
          j = j + 1
     Loop
     If ddx < tdx2(j) Then
         Line (dcx, dcy)-(tdx2(j - 1), tdy2(j - 1)), QBColor(5)
         qG = j - 1
     Else
         Line (dcx, dcy)-(tdx2(j), tdy2(j)), QBColor(5)
         qG = j
     End If
  End If
 End If
End If
'************************************************
 'Duong di giua cac vat can
  For s = 2 To t - 1
    Line (tsxmax(s), tsymax(s) / 2)-(tsxmax(s) + (tsxmin(s + 1) - tsxmax(s)) / 2, tsymax(s) +
(tsymin(s + 1) + (7000 - tsymin(s + 1)) / 2 - tsxmax(s)) / 2), QBColor(5)
    tg1 = tsxmax(s) + (tsxmin(s + 1) - tsxmax(s)) / 2 - tsxmax(s)
    tg2 = tsymax(s) + (tsymin(s + 1) + (7000 - tsymin(s + 1)) / 2 - tsxmax(s)) / 2 - tsymax(s)
/ 2
     k1 = k1 + 1
    Line (tsxmax(s) + (tsxmin(s + 1) - tsxmax(s)) / 2, tsymax(s) + (tsymin(s + 1) + (7000 -
tsymin(s + 1)) / 2 - tsxmax(s)) / 2)-(tsxmax(s), tsymax(s) + (7000 - tsymax(s)) / 2),
QBColor(5)
    tg1 = tsxmax(s) - tsxmax(s) + (tsxmin(s + 1) - tsxmax(s)) / 2
    tg2 = tsymax(s) + (7000 - tsymax(s)) / 2 - tsymax(s) + (tsymin(s + 1) + (7000 - tsymin(s
+ 1)) / 2 - tsxmax(s)) / 2
     k1 = k1 + 1
Next s
End Sub
```

```
---------------------------------------------------------------------------------
Private Sub Command4_Click()  ' Tim duong di ngan nhat
s = ql
n = td1 + td2 - 1
t = qG
' Khoi tao
For v = 1 To n
          d(v) = ts(s, v)
          truoc(v) = s
           final(v) = False
Next v
truoc(s) = 0
d(s) = 0
final(s) = True
'buoc lap
Do While Not (final(t))
            minp = maxint
           For v = 1 To n
                    If (Not (final(v))) And (minp > d(v)) Then
                              u = v
                              minp = d(v)
                     End If
             Next v
   final(u) = True
   If Not (final(t)) Then
      For v = 1 To n
         If (Not (final(v))) And (d(u) + ts(u, v) < d(v)) Then
            d(v) = d(u) + ts(u, v)
            truoc(v) = u
         End If
      Next v
  End If
Loop
DrawWidth = 5
i = t
' noi duong dan giai phap
If (ddy < tdy1max) And (dcy < tdy1max) Then ' hai diem thuoc nua tren
   Line (ddx, ddy)-(tdx1(ql), tdy1(ql)), &HFF&
   Do While i <> s
     Line (tdx1(i), tdy1(i))-(tdx1(truoc(i)), tdy1(truoc(i))), &HFF&
     i = truoc(i)
  Loop
  Line (dcx, dcy)-(tdx1(qG), tdy1(qG)), &HFF&
Else
 If (ddy > tdy1max) And (dcy > tdy1max) Then 'Hai diem thuoc nua duoi
   Line (ddx, ddy)-(tdx2(ql), tdy2(ql)), &HFF&
 Do While i <> s
    Line (tdx2(i), tdy2(i))-(tdx2(truoc(i)), tdy2(truoc(i))), &HFF&
    i = truoc(i)
  Loop
  Line (dcx, dcy)-(tdx2(qG), tdy2(qG)), &HFF&
 Else
```

```
  If (ddy < tdy1max) Then ' ql nua tren qG nua duoi
    Line (ddx, ddy)-(tdx1(ql), tdy1(ql)), &HFF&
    If tdx1(dn1(1)) < tdx1(ql) Then
       k1 = dn1(1)
       k2 = ql - 1
     Else
       k1 = ql
       k2 = dn1(1) - 1
     End If
    For i = k1 To k2
        Line (tdx1(i), tdy1(i))-(tdx1(i + 1), tdy1(i + 1)), &HFF&
    Next i
    s = 2
    Line (tsxmax(s), tsymax(s) / 2)-(tsxmax(s) + (tsxmin(s + 1) - tsxmax(s)) / 2, tsymax(s) +
(tsymin(s + 1) + (7000 - tsymin(s + 1)) / 2 - tsxmax(s)) / 2), &HFF&
    Line (tsxmax(s) + (tsxmin(s + 1) - tsxmax(s)) / 2, tsymax(s) + (tsymin(s + 1) + (7000 -
tsymin(s + 1)) / 2 - tsxmax(s)) / 2)-(tsxmax(s), tsymax(s) + (7000 - tsymax(s)) / 2), &HFF&
    If tdx2(dn2(1)) < tdx2(qG) Then
       k1 = dn2(1)
       k2 = qG - 1
     Else
       k1 = qG
       k2 = dn2(1) - 1
     End If
    For i = k1 To k2
        Line (tdx2(i), tdy2(i))-(tdx2(i + 1), tdy2(i + 1)), &HFF&
    Next i
    Line (dcx, dcy)-(tdx2(qG), tdy2(qG)), &HFF&

  Else ' ql nua duoi qG nua tren
          Line (ddx, ddy)-(tdx2(ql), tdy2(ql)), &HFF&
         If tdx2(ql) < tdx2(dn2(1)) Then
          k1 = ql
          k2 = dn2(1) - 1
    Else
         k2 = ql - 1
         k1 = dn2(1)
    End If
    For i = k1 To k2
         Line (tdx2(i), tdy2(i))-(tdx2(i + 1), tdy2(i + 1)), &HFF&
    Next  i
    s = 2
    Line (tsxmax(s), tsymax(s) / 2)-(tsxmax(s) + (tsxmin(s + 1) - tsxmax(s)) / 2, tsymax(s) +
(tsymin(s + 1) + (7000 - tsymin(s + 1)) / 2 - tsxmax(s)) / 2), &HFF&
    Line (tsxmax(s) + (tsxmin(s + 1) - tsxmax(s)) / 2, tsymax(s) + (tsymin(s + 1) + (7000 -
tsymin(s + 1)) / 2 - tsxmax(s)) / 2)-(tsxmax(s), tsymax(s) + (7000 - tsymax(s)) / 2), &HFF&
    If tdx1(qG) < tdx1(dn1(1)) Then
       k1 = qG
       k2 = dn1(1) - 1
    Else
       k1 = dn1(1)
       k2 = qG - 1
```

```
    End If
    For i = k1 To k2
        Line (tdx1(i), tdy1(i))-(tdx1(i + 1), tdy1(i + 1)), &HFF&
    Next i
    Line (dcx, dcy)-(tdx1(qG), tdy1(qG)), &HFF&
  End If
 End If
End If
End Sub
---------------------------------------------------------------------------
Private Sub Form_Load()
n = 1
t = 1
dc(1) = 0
dk1 = False
td1 = 1
td2 = 1
ddx = 0
ddy = 0
dcx = 0
dcy = 0
maxint = 100000
End Sub
---------------------------------------------------------------------------------------------------
Private Sub Form_MouseDown(Button As Integer, Shift As Integer, x As Single, y As
Single)
If O2.Value = True Then
 Select Case Button
  Case vbLeftButton
     If c <> 2 Then
        Circle (x, y), 30, QBColor(2)
        a = x
        b = y
        c = 2
        dx = a
        dy = b
       Ar(n) = a
       Br(n) = b
     Else
       n = n + 1
        Circle (x, y), 30, QBColor(2)
        DrawWidth = 7
        Line (a, b)-(x, y), QBColor(2)
        a = x
        b = y
        Ar(n) = a
        Br(n) = b
     End If
  Case vbRightButton
     c = 1
      DrawWidth = 7
     Line (dx, dy)-(a, b), QBColor(2)
```

```
      n = n + 1
      Ar(n) = dx
      Br(n) = dy
      t = t + 1
      dc(t) = n
      n = n + 1
    End Select
Else
  If O1.Value = True Then
     Select Case Button
        Case vbLeftButton
           Circle (ddx, ddy), 30, &H8000000F
           Circle (x, y), 30, QBColor(3)
           ddx = x
           ddy = y
           Print "ql"
        Case vbRightButton
           Circle (dcx, dcy), 30, &H8000000F
           Circle (x, y), 30, QBColor(4)
           dcx = x
           dcy = y
           Print "qG"
     End Select
  Else
     MsgBox "Hay chon mot nut...!"
  End If
End If
End Sub
```

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA ĐIỆN – ĐIỆN TỬ
BỘ MÔN VIỄN THÔNG

LUẬN VĂN TỐT NGHIỆP

ROBOT TỰ HÀNH TRÁNH VẬT CẢN SỬ DỤNG THIẾT BỊ KINECT

GVHD: PGS. TS. Hoàng Đình Chiến
SVTH : Nguyễn Hồng Đức 40700566
Nguyễn Văn Đức 40700577

- Tp. Hồ Chí Minh, Tháng 1-2012 -

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐH BÁCH KHOA
Thành phố Hồ Chí Minh
ଓ ★ ଌ

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc Lập – Tự Do – Hạnh Phúc
ଓ ★ ଌ

Số:_____/BKĐT
Khoa: Điện – Điện tử
Bộ Môn: Viễn Thông

NHIỆM VỤ LUẬN VĂN TỐT NGHIỆP

Họ và tên: **NGUYỄN HỒNG ĐỨC** MSSV: **40700566**
Họ và tên: **NGUYỄN VĂN ĐỨC** MSSV: **40700577**
Ngành: **VIỄN THÔNG** LỚP: **DD07DV4**

1. **Đầu đề luận văn: "Robot tự hành tránh vật cản sử dụng thiết bị Kinect"**
2. **Nhiệm vụ** (Yêu cầu về nội dung và số liệu ban đầu):
   .....
   .....
   .....
   .....
3. **Ngày giao nhiệm vụ luận văn:**
4. **Ngày hoàn thành nhiệm vụ:**
5. **Họ và tên người hướng dẫn:** **Phần hướng dẫn**
   .....

Nội dung và yêu cầu LVTN đã được thông qua Bộ Môn.
Ngày . .....tháng .....năm 2012

CHỦ NHIỆM BỘ MÔN
(*Ký và ghi rõ họ tên*)

NGƯỜI HƯỚNG DẪN CHÍNH
(*Ký và ghi rõ họ tên*)

PHẦN DÀNH CHO KHOA, BỘ MÔN:
Người duyệt (chấm sơ bộ):
Đơn vị:
Ngày bảo vệ:
Điểm tổng kết:
Nơi lưu trữ luận văn:

TRƯỜNG ĐH BÁCH KHOA
KHOA ĐIỆN – ĐIỆN TỬ

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc Lập – Tự Do – Hạnh Phúc
--o0o--
Ngày tháng năm 2012

PHIẾU CHẤM BẢO VỆ LVTN
(*Dành cho người hướng dẫn*)

Họ và tên: **NGUYỄN HỒNG ĐỨC** MSSV: **40700566**
Họ và tên: **NGUYỄN VĂN ĐỨC** MSSV: **40700577**
Ngành: **VIỄN THÔNG** LỚP: **DD07DV4**

1. Đề tài: **"Robot tự hành tránh vật cản sử dụng thiết bị Kinect"**
2. Họ tên người hướng dẫn: **PGS. TS. HOÀNG ĐÌNH CHIẾN**
3. Tổng quát về bản thuyết minh:

Số trang	.....	Số chương	.....
Số bảng số liệu	.....	Số hình vẽ	.....
Số tài liệu tham khảo	.....	Phần mềm tính toán	.....

4. Tổng quát về các bản vẽ:
- Số bản vẽ: bản A1 bản A2 khổ khác
- Số bản vẽ tay số bản vẽ trên máy tính

5. Những ưu điểm chính của LVTN:

6. Những thiếu sót chính của LVTN:

7. Đề nghị: Được bảo vệ ☐, Bổ sung thêm để bảo vệ ☐,
Không được bảo vệ ☐.
8. 3 câu hỏi sinh viên trả lời trước Hội Đồng:
a)

b)

c)

9. Đánh giá chung (*bằng chữ: giỏi, khá, TB*): Điểm …….

Ký tên (*ghi rõ họ tên*)

TRƯỜNG ĐH BÁCH KHOA
KHOA ĐIỆN – ĐIỆN TỬ

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc Lập – Tự Do – Hạnh Phúc
--o0o--
Ngày tháng năm 2012

PHIẾU CHẤM BẢO VỆ LVTN
(*Dành cho người phản biện*)

Họ và tên: **NGUYỄN HỒNG ĐỨC** MSSV: **40700566**
Họ và tên: **NGUYỄN VĂN ĐỨC** MSSV: **40700577**
Ngành: **VIỄN THÔNG** LỚP: **DD07DV4**

10. Đề tài: **"Robot tự hành tránh vật cản sử dụng thiết bị Kinect"**

11. Họ tên người phản biện:

12. Tổng quát về bản thuyết minh:

Số trang	.....	Số chương	.....
Số bảng số liệu	.....	Số hình vẽ	.....
Số tài liệu tham khảo	.....	Phần mềm tính toán	.....

13. Tổng quát về các bản vẽ:
- Số bản vẽ: bản A1 bản A2 khổ khác
- Số bản vẽ tay số bản vẽ trên máy tính

14. Những ưu điểm chính của LVTN:

15. Những thiếu sót chính của LVTN:

16. Đề nghị: Được bảo vệ ☐, Bổ sung thêm để bảo vệ ☐,
Không được bảo vệ ☐.

17. 3 câu hỏi sinh viên trả lời trước Hội Đồng:
a)

b)

c)

18. Đánh giá chung (*bằng chữ: giỏi, khá, TB*): Điểm …….

Ký tên (*ghi rõ họ tên*)

LỜI CẢM ƠN

Lời đầu tiên, chúng em xin gửi đến Thầy, PGS. TS. Hoàng Đình Chiến lời cảm ơn chân thành và sâu sắc nhất. Nhờ có sự hướng dẫn và giúp đỡ tận tình của Thầy trong suốt thời gian qua, chúng em đã có thể thực hiện và hoàn thành Đồ Án Môn Học 2, Thực Tập Tốt Nghiệp và Luận Văn Tốt Nghiệp. Những lời nhận xét, góp ý và hướng dẫn tận tình của Thầy đã giúp chúng em có một định hướng đúng đắn trong suốt quá trình thực hiện Đề tài, giúp chúng em nhìn ra được những ưu khuyết điểm của Đề tài và từng bước hoàn thiện hơn.

Đồng thời, chúng em xin trân trọng cảm ơn các Thầy Cô của Trường Đại Học Bách Khoa nói chung và của khoa Điện - Điện Tử nói riêng đã dạy dỗ chúng em suốt quãng thời gian ngồi trên ghế giảng đường Đại học. Những lời giảng của Thầy Cô trên bục giảng đã trang bị cho chúng em những kiến thức và giúp chúng em tích lũy thêm những kinh nghiệm.

Chúng em cũng xin gửi lời cảm ơn tới hai cựu sinh viên Bách Khoa: anh Nguyễn Quốc Thịnh và anh Nguyễn Thanh Tâm đã tận tình hướng dẫn chúng em định hướng đúng trong những ngày bắt đầu nhận đề tài luận văn.

Bên cạnh đó, chúng tôi xin cảm ơn sự hỗ trợ và giúp đỡ của bạn bè trong thời gian học tập tại Trường Đại Học Bách Khoa và trong quá trình hoàn trình hoàn thành Luận Văn Tốt Nghiệp này.

Cuối cùng, chúng con cũng chân thành cảm ơn sự động viên và sự hỗ trợ của gia đình và cha mẹ trong suốt thời gian học tập. Đặc biệt, chúng con xin gửi lời cảm ơn trân trọng nhất đến cha mẹ, người đã sinh ra và nuôi dưỡng chúng con nên người. Sự quan tâm, lo lắng và hy sinh lớn lao của cha mẹ luôn là động lực cho chúng con cố gắng phấn đấu trên con đường học tập của mình. Một lần nữa, chúng con xin gửi đến cha mẹ sự biết ơn sâu sắc nhất.

Hồ Chí Minh, ngày 8 tháng 1 năm 2012

NGUYỄN HỒNG ĐỨC

NGUYỄN VĂN ĐỨC

TÓM TẮT LUẬN VĂN

Theo dự đoán trong tương lai, robot sẽ là tâm điểm của một cuộc cách mạng lớn sau Internet. Con người sẽ có nhu cầu sở hữu một robot cá nhân như nhu cầu một máy tính PC bây giờ. Với xu hướng này, cùng các ứng dụng truyền thống khác của robot trong công nghiệp, y tế, giáo dục đào tạo, giải trí và đặc biệt là trong an ninh quốc phòng thì thị trường robot sẽ vô cùng to lớn. Đề tài luận văn hướng tới việc ứng dụng công nghệ xử lý ảnh mới cho robot tự hành, tạo tiền đề cho việc xây dựng một robot dịch vụ hoàn chỉnh, có khả năng phục vụ cho đời sống con người.

Trong khuôn khổ của luận văn, nhóm sẽ tập trung xây dựng một mô hình mobile robot hoàn chỉnh có khả năng tìm đường đến đích và tránh chướng ngại vật trên quãng đường di chuyển. Một điểm mới được nhấn mạnh là khối thị giác máy tính cho mobile robot,với sự hỗ trợ của thiết bị chơi game Kinect có khả năng khôi phục môi trường phía trước robot dưới dạng 3D, đáp ứng được sự chính xác cần thiết khi phối hợp với các giải thuật điều khiển truyền thống cho robot.

Nhóm sinh viên thực hiện

NGUYỄN HỒNG ĐỨC

NGUYỄN VĂN ĐỨC

Đề mục Trang

Lời cảm ơn i
Tóm tắt luận văn ii
Mục lục iii
Danh mục từ viết tắt v
Danh mục hình viii
Danh mục bảng xi

Mục lục

Chương 1: Giới thiệu 1
1.1 Xu hướng phát triển của robot hiện đại 2
1.2 Những vấn đề của robot di động 2
1.3 Mục tiêu luận văn và phương pháp thực hiện 3
1.4 Sơ lược về nội dung luận văn 4
Chương 2: Tìm hiểu về Kinect 5
2.1 Giới thiệu chung 6
2.2 Những thành phần chính của Kinect 7
2.3 Tính toán độ sâu 8
2.4 Một số đặc tính khác 12
Chương 3: Thư viện xử lý ảnh 15
3.1 Thư viện hỗ trợ Kinect 16
3.2 So sánh Kinect SDK beta và OpenNI 17
3.3 Point Cloud Library 20
Chương 4: Phát hiện vật cản 22
4.1 Các phương pháp phát hiện vật cản không sử dụng camera 23
4.1.1 Dùng công tắc hành trình 23
4.1.2 Dùng cảm biến siêu âm [13] 23
4.2 Các phương pháp phát hiện vật cản sử dụng camera 26

4.2.1 Xử lý ảnh với một camera (Monocular vision) 26
4.2.2 Xử lý ảnh với hai camera (Stereo vision) 29
4.3 Phát hiện vật cản sử dụng Kinect 31
Chương 5: Module điều khiển động cơ 39
5.1 PIC 18F4550 40
5.1.1 Giới thiệu chung 40
5.1.2 Những module chính sử dụng trong luận văn 45
5.2 Mạch công suất (mạch cầu H) 55
Chương 6: Động cơ và giải thuật PID vị trí 56
6.1 Động cơ Servo DC 57
6.1.1 Động cơ DC 57
6.1.2 Encoder 59
6.2 Giải thuật PID vị trí [19] 63
Chương 7: Tính toán tọa độ Robot và Kinect 68
7.1 Các phép chuyển đổi hệ trục tọa độ cơ bản 69
7.2 Tính toán tọa độ robot 70
7.3 Tính toán tọa độ Kinect 73
Chương 8: Chương trình điều khiển 76
8.1 Nội dung chương trình điều khiển 77
8.2 Giải thuật chương trình do máy tính xử lý 77
8.3 Giải thuật chương trình do vi điều khiển xử lý 88
KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 90
TÀI LIỆU THAM KHẢO 92
PHỤ LỤC 1: Kết hợp thư viện OpenNI và Code Laboratories Kinect (CL) để sử dụng chức năng điều khiển động cơ Kinect. 93
PHỤ LỤC 2: Cách đấu dây dùng pin 12V thay adapter và tạo đế gắn lên robot cho Kinect 96
PHỤ LỤC 3: Kích thước robot 99

Danh mục từ viết tắt

A

ADC Analog Digital Converter
AGV Autonomous Guided Vehicles
API Application Programming Interface
AUV Autonomous Underwater Vehicles
AUX AUXiliary

C

CCP Capture/Compare/PWM
CL Code Laboratories
CMOS Complementary Metal – Oxide – Semiconductor
CNC Computerized Numerical Control
CPU Central Processing Unit

E

ECCP Enhanced Capture/Compare/PWM
EEPROM Electrically Erasable Programmable Read – Only Memory
EUSART Enhanced Universal Synchronous Asynchronous Receiver Transmitter

F

FLANN Fast Library for Approximate Nearest Neighbors

G

GPIO General Purpose Input Ouput

H

HDD Hard Disk Drive

I

ICD In – Circuit Debugger
ICSP In – Circuit Serial Programming
IR Infrared

J

JNA Java Native Access
JNI Java Native Interface

M

MFC Microsoft Foundation Class Library
MSSP Master Synchronous Serial Port
MUX Multiplexer

N

NI Natural Interaction
NUI Natural User Interface

Q

QVGA Quarter Video Graphics Array

R

RAM Random Access Memory
RANSAC RANdom SAmple Consensus
RGB Red, Green, Blue

S

SDK Software Development Kit

SSP Synchronous Serial Port
SXGA Super eXtended Graphics Array

T

TOF Time Of Flight
TTL Transistor – Transistor Logic

U

UAV Unmanned Arial Vehicles
USART Universal Synchronous Asynchronous Receiver Transmitter
USB Universal Serial Bus

V

VGA Video Graphics Array
VTK Visualization Toolkit

Danh mục hình

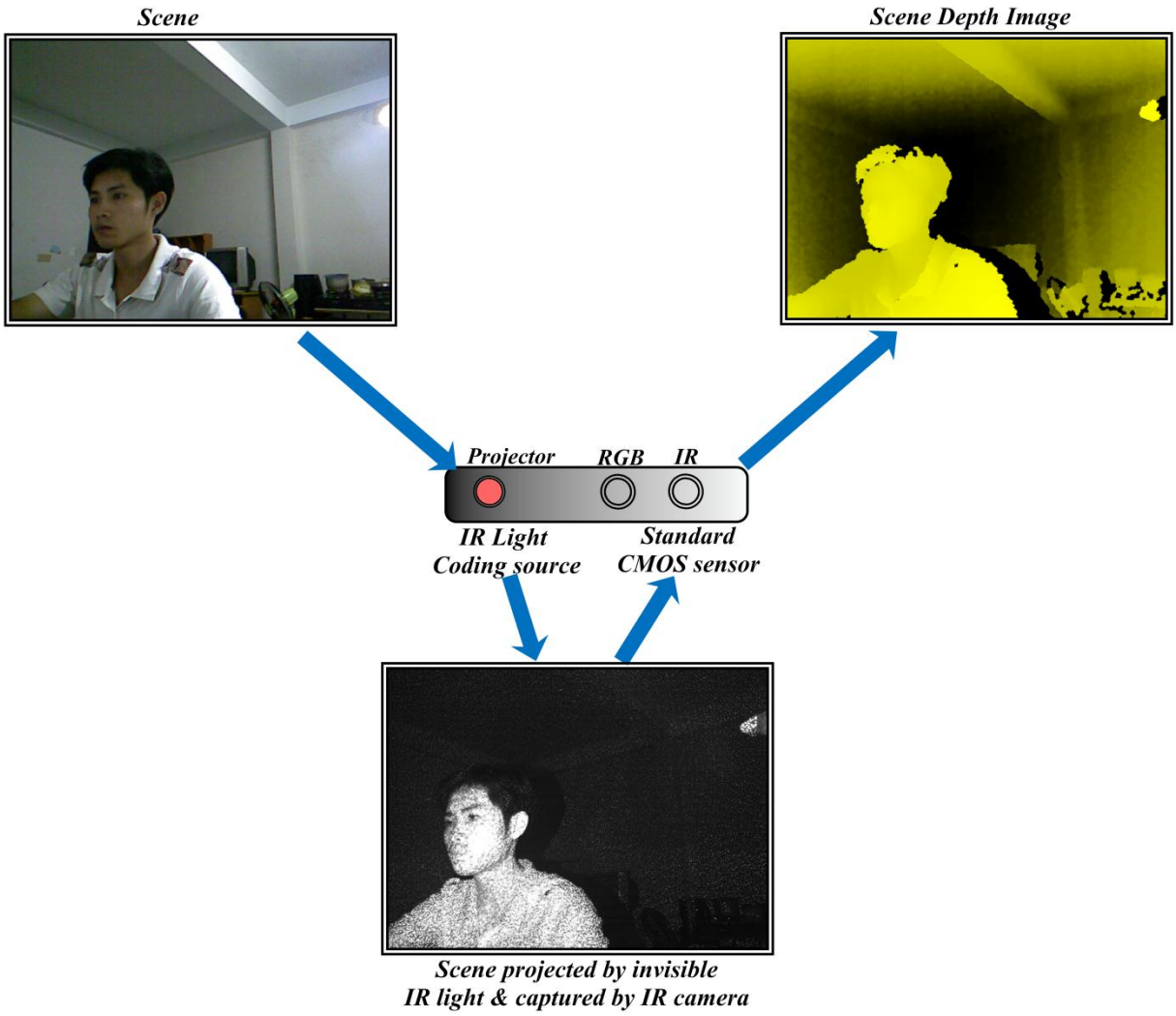
Hình 2.1: Thiết bị Kinect 6
Hình 2.2: Những thành phần chính của Kinect 7
Hình 2.3: Động cơ điều khiển góc ngẩng Kinect 8
Hình 2.4: Bên trong Kinect: RGB, IR camera và IR projector 8
Hình 2.5: Quá trình thu về bản đồ độ sâu ảnh 9
Hình 2.6: Mẫu hình được chiếu bởi projector và chụp lại bằng IR camera 10
Hình 2.7: Tính toán khoảng cách tới một điểm chiếu từ Projector 11
Hình 2.8: Kinect adapter 14
Hình 3.1: Thư viện OpenNI phối hợp giữa phần cứng và ứng dụng đầu cuối 19
Hình 3.2: Point cloud library logo 20
Hình 4.1: Mô hình robot dùng công tắc hành trình 23
Hình 4.2: Cảm biến siêu âm 24
Hình 4.3: Hiện tượng Forecasting 25
Hình 4.4: Hiện tượng Crosstalk 25
Hình 4.5: Thị trường của robot với optical flow 26
Hình 4.6: Ảnh gốc và ảnh sau khi tách biên 27
Hình 4.7: Hạn chế của phương pháp dò biên 28
Hình 4.8: Phương pháp dò nền 28
Hình 4.9: Phương pháp Stereo Vision 29
Hình 4.10: Sơ đồ xử lý: phát hiện và tách vật cản 31
Hình 4.11: Ảnh RGB và bản đồ độ sâu 32
Hình 4.12: RGB point cloud 33
Hình 4.13: Voxel grid 34
Hình 4.14: Pass through không có và có voxel grid 35
Hình 4.15: Tìm mô hình đường thẳng bằng thuật toán RANSAC 36
Hình 4.16: Point cloud sau khi thực hiện xong bước lọc và phân đoạn 37

Hình 4.17: Object cluster.....38
Hình 5.1: PICLAB-V2 và board mạch cầu H 40
Hình 5.2: PIC 18F4550 40
Hình 5.3: Sơ đồ chân PIC18F4550 41
Hình 5.4: Sơ đồ khối của PIC 18F4550 44
Hình 5.5: Sơ đồ khối bộ dao động của PIC 18F4550 45
Hình 5.6: Cấu tạo của chân GPIO.....48
Hình 5.7: Sơ đồ khối logic của hệ thống ngắt trong PIC18F4550.....49
Hình 5.8: Sơ đồ khối của bộ Timer 1.....50
Hình 5.9: Sơ đồ khối của bộ Timer 2.....51
Hình 5.10: Sơ đồ khối bộ PWM.....52
Hình 5.11: Giản đồ thời gian của sóng điều xung tại chân CCPx.....52
Hình 5.12: Sơ đồ khối bộ truyền của module EUSART.....54
Hình 5.13: Sơ đồ khối bộ nhận của module EUSART 55
Hình 6.1: Cặp động cơ Servo DC.....57
Hình 6.2: Điều chỉnh độ rộng xung PWM 58
Hình 6.3: Dạng sóng áp và dòng trên động cơ.....59
Hình 6.4: Optical Encoder.....60
Hình 6.5: Hai kênh A và B lệch pha trong encoder 61
Hình 6.6: Encoder đi kèm động cơ Servo DC.....62
Hình 6.7: PID vòng kín 63
Hình 6.8: Đáp ứng của các hệ thống điều khiển 65
Hình 6.9: Quá trình tính toán PID.....66
Hình 7.1: Phép tịnh tiến.....69
Hình 7.2: Phép quay 70
Hình 7.3: Mô hình robot.....71
Hình 7.4: Tọa độ robot (quan sát từ trên xuống).....72
Hình 7.5: Hệ trục tọa độ Kinect 73
Hình 7.6: Đồng nhất hệ trục tọa độ Kinect và Robot.....74

Hình 7.7: Tọa độ Kinect trước (trái) và sau (phải) khi chuyển trục 75
Hình 8.1: Nội dung chương trình điều khiển 77
Hình 8.2: Xử lý đa tiến trình 78
Hình 8.3: Giao diện chương trình điều khiển 78
Hình 8.4: Sơ đồ giải thuật điều khiển robot do máy tính xử lý 80
Hình 8.5: Tính góc quay về đích 81
Hình 8.6: Vật cản bên trái robot 83
Hình 8.7: Vật cản bên phải robot 84
Hình 8.8: Vật cản nằm ở giữa đường di chuyển của robot 85
Hình 8.9: Đi một đoạn an toàn về phía phải vật cản 86
Hình 8.10: Cờ báo có vật cản và đường trống 87
Hình 8.11: Không gian cho robot lách qua 88
Hình 8.12: Sơ đồ giải thuật trên vi điều khiển 89

Danh mục bảng

Bảng 2.1: Góc mở và tiêu cự RGB và IR camera 13
Bảng 2.2: Công suất tiêu thụ trên Kinect 14
Bảng 5.1: Bảng mô tả các chức năng từng chân của PIC18F4550 42
Bảng 6.1: Ảnh hưởng của các thành phần Kp, Ki, Kd đối với hệ kín.....64

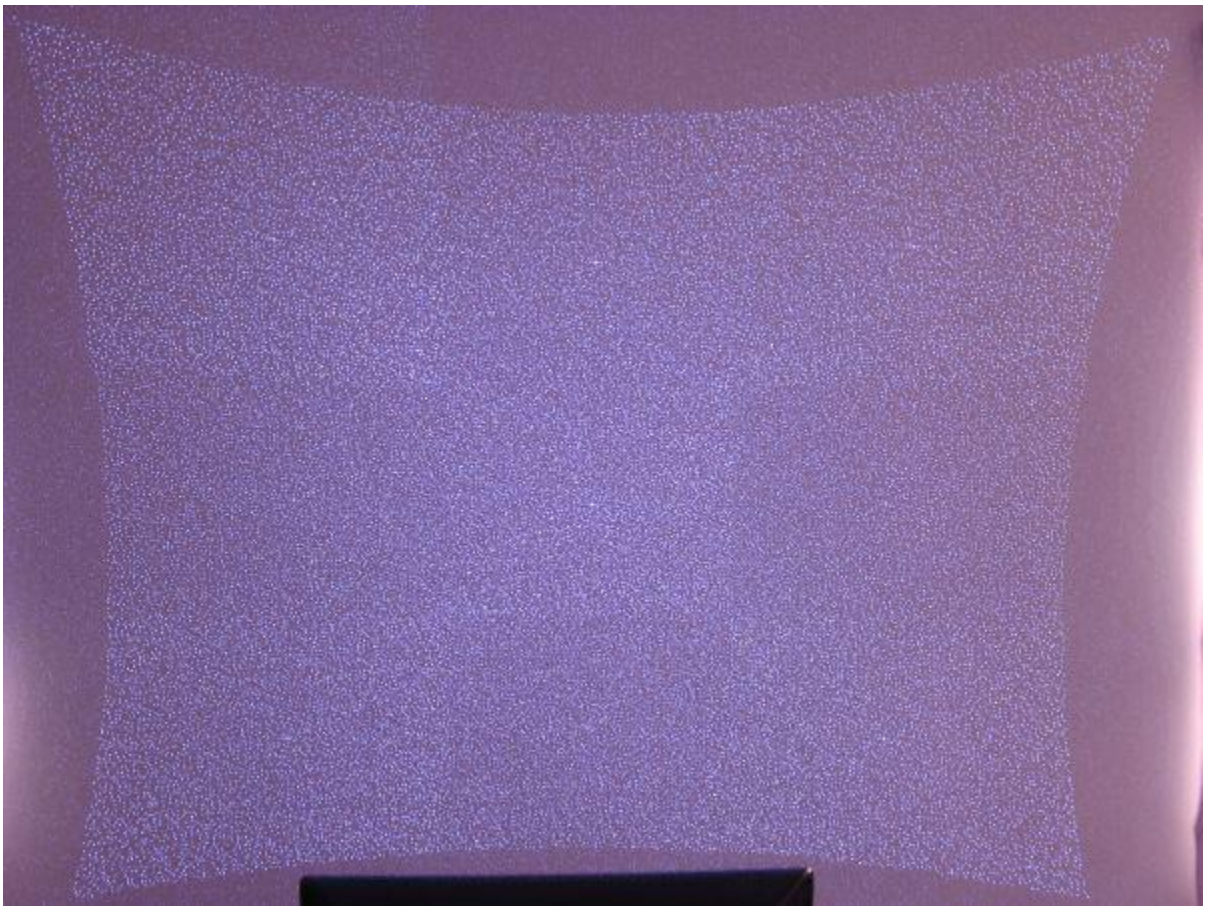


Hình 2.5: Quá trình thu về bản đồ độ sâu ảnh

Khác với kỹ thuật Stereo Camera với việc dùng cặp camera giống nhau để xây dựng nên bản đồ độ sâu, hay kỹ thuật Time-Of-Flight (TOF) định nghĩa khoảng cách bằng ước lượng thời gian di chuyển của tia sáng đi và về trong không gian; kỹ thuật Light Coding dùng một nguồn sáng hồng ngoại chiếu liên tục kết hợp với một camera hồng ngoại để tính toán khoảng cách [3]. Công việc tính toán này được thực hiện bên trong Kinect bằng chip **PS1080 SoC** của PrimeSense. Công nghệ mới này được cho là đáp ứng chính xác hơn, giá cả rẻ hơn cho việc sử dụng ở môi trường trong nhà.

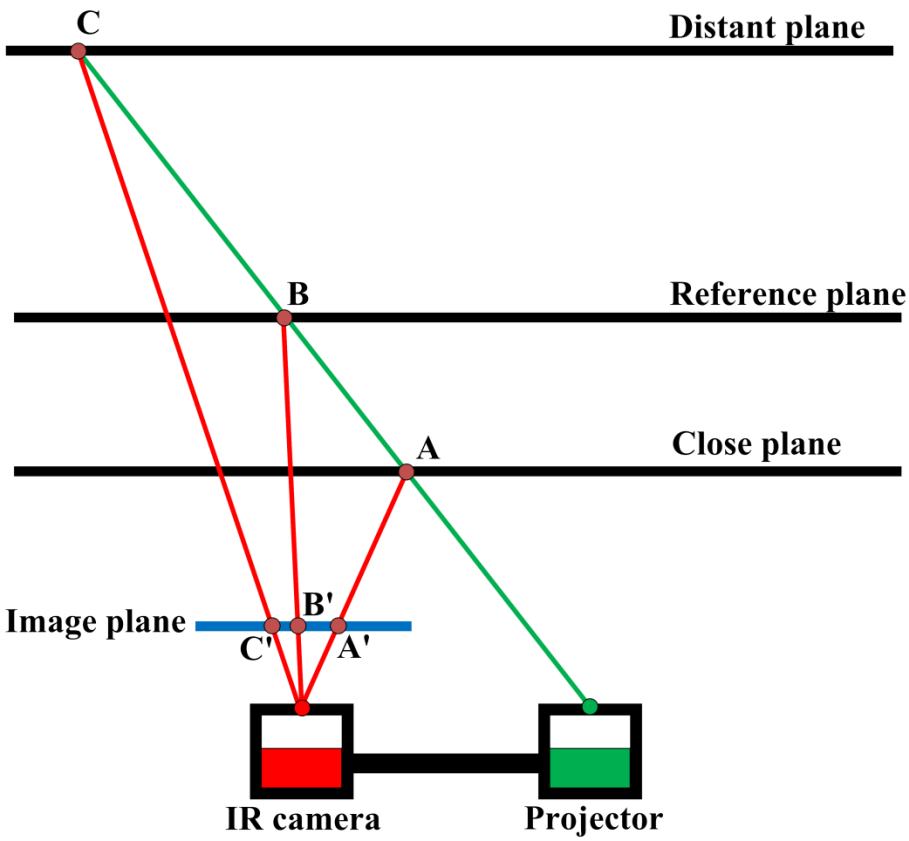
Projector sẽ chiếu một chùm sáng hồng ngoại, tạo nên những đốm sáng ở không gian phía trước Kinect, tập hợp đốm sáng được phát ra này là cố định. Những đốm sáng này được tạo ra nhờ một nguồn sáng truyền qua lưới nhiễu xạ (diffraction

gratings). Tập hợp các đốm sáng này được IR camera chụp lại, thông qua giải thuật đặc biệt được tích hợp trong **PS1080 SoC** [4] cho ra bản đồ độ sâu. Bản chất của giải thuật này là các phép toán hình học dựa trên quan hệ giữa hai cảm biến IR camera và Projector mà ta sẽ đề cập sau. Hình 2.6 cho ta thấy rõ mẫu hình tập hợp các đốm sáng từ Projector và được chụp lại bởi IR camera.



Hình 2.6: Mẫu hình được chiếu bởi projector và chụp lại bằng IR camera

Để hiểu cách thức Kinect ước lượng khoảng cách tới vật thể trong môi trường như thế nào, ta quan sát hình 2.7 trong trường hợp phân tích với một điểm đơn giản.



Hình 2.7: Tính toán khoảng cách tới một điểm chiếu từ Projector [5]

Ta giả sử Projector phát đi một tia sáng dọc đường màu xanh lá, nó sẽ được chụp lại dưới dạng một đốm sáng bởi IR camera khi chạm vào bề mặt vật thể trong không gian. Ta xét ba mặt phẳng ở ba khoảng cách khác nhau: mặt phẳng gần Kinect (close plane), mặt phẳng ở xa Kinect (distant plane) và mặt phẳng tham chiếu (reference plane) ở giữa hai mặt phẳng trên. Trong đó, mặt phẳng tham chiếu ngầm được biết trước bên trong Kinect với đầy đủ thông tin về khoảng cách. Ngoài ra, ta cũng đề cập thêm mặt phẳng ảnh (image plane) của IR camera, là mặt phẳng hình chiếu của các điểm trong không gian thu về bởi IR camera. Ta xét trong ba trường hợp khi tia sáng màu xanh lá chạm vào ba điểm trên ba mặt phẳng lần lượt là A, B, C; ba điểm này được chiếu lên mặt phẳng ảnh tương ứng là A', B', C'. Quan sát vị trí A', B' và C', ta có nhận xét: điểm A càng gần Kinect (hay close plane càng gần Kinect) thì A' càng xa B' về phía bên phải; ngược lại, điểm C càng xa Kinect (hay distant plane càng xa Kinect) thì C' càng xa B' về phía bên trái. Từ đó: khi ta biết trước hướng, điểm xuất

phẩm thương mại của Microsoft nên các thông số kỹ thuật chi tiết không được công bố. Các thông số được trình bày dưới đây là kết quả đo đạc thực nghiệm:

- ***Tiêu cự, góc mở IR camera và RGB camera:***

Hai camera RGB và IR được đặt cách nhau 2.5 cm nên có chút khác nhau ở khung hình thu về từ hai camera. Để đảm bảo khung hình RGB có thể chứa được khung hình IR, người ta thiết kế góc mở của RGB camera lớn hơn. Điều này cũng dẫn đến tiêu cự của RGB camera nhỏ hơn. Các thông số trong bảng 2.1 được đo đạc bằng thực nghiệm:

Feature		**RGB camera**	**IR camera**
Field of View (degrees)	*Horizontal*	~62^o	~58^o
	Vertical	~48^o	~44^o
	Diagonal	~72^o	~69^o
Focal length (pixels)		525	580

Bảng 2.1: Góc mở và tiêu cự của RGB và IR camera [3]

- ***Nguồn cung cấp và công suất tiêu thụ:***

Vì Kinect cần nhiều điện năng để hoạt động nên cổng USB của Xbox-360 không thể đáp ứng mà phải qua một cổng chia để chia thành 2 kết nối riêng là USB và kết nối nguồn, giúp cho thiết bị kết nối với Xbox-360 bằng cổng USB trong khi nguồn điện cần cho Kinect là 12VDC được lấy từ adapter. Phiên bản Xbox-360 mới sẽ không cần adapter vì nó có các AUX port đặc biệt để cung cấp cho cổng kết nối. Với kết nối USB ta hoàn toàn có thể cho Kinect giao tiếp với máy tính. Cách thay adapter bằng nguồn pin 12V dùng trên mobile robot được nói thêm ở phần phụ lục 2.

Hình 2.8: Kinect adapter

Công suất tiêu thụ đo bằng thực nghiệm:

Power consumption (idle)	~3.3W
Power consumption (active)	~4.7W

Bảng 2.2: Công suất tiêu thụ trên Kinect [3]

- ***Môi trường hoạt động:***

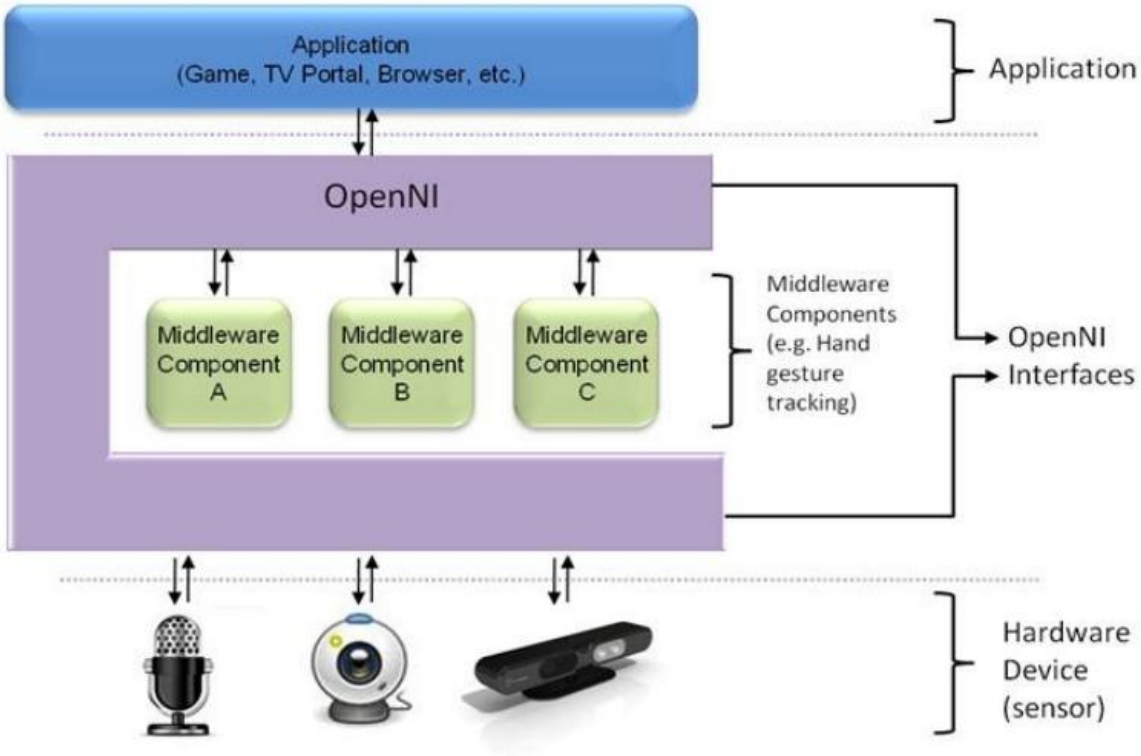
Kinect là thiết bị được thiết kế cho việc sử dụng ở môi trường trong nhà (indoor). Ở môi trường ngoài trời, kết quả thử nghiệm cho bản đồ độ sâu không chính xác vào thời điểm ánh sáng mạnh, nhưng cho kết quả chấp nhận được khi ánh sáng yếu (vào thời điểm buổi chiều tối).

- Không hỗ trợ nhận biết cử chỉ.
- Chỉ hỗ trợ trên Win7 (x86 và x64) với đòi hỏi khá cao:
  - Máy tính dual-core, tốc độ 2.66 GHz hoặc nhanh hơn.
  - Windows 7 với hỗ trợ của DirectX 9.0 trở lên.
  - Ram tối thiểu 2 GB.
  - Lập trình trên Visual Studio 2010.
- Không hỗ trợ việc thu ảnh trực tiếp từ IR camera.
- Vùng Kinect không nhìn thấy trong khoảng 0÷0.8 mét trước Kinect.

- ***OpenNI:***
  - ✓ *Ưu điểm:*
    - Cho phép xây dựng các ứng dụng thương mại hóa.
    - Hỗ trợ phát hiện cử chỉ.
    - Skeleton tracking: hỗ trợ bám từng phần cơ thể người và hand tracking thông qua liên kết với các middleware. Hơn nữa, tiêu thụ công suất của CPU ít hơn so với khi sử dụng Kinect SDK.
    - Hỗ trợ truy xuất các sensor của Kinect đồng thời.
    - Hỗ trợ cho Windows, Linux và Mac OSX.
    - Cho phép truy xuất hình ảnh thu về từ IR camera.
    - Vùng Kinect không nhìn thấy trong khoảng chấp nhận được là 0.5 mét trước Kinect.
  - ✓ *Khuyết điểm:*
    - Không hỗ trợ phần xử lý âm thanh cho dãy microphone.
    - Skeleton tracking: bám đặc tính cơ thể còn nhiều lỗi và chưa được ổn định như trên Kinect SDK.
    - Không hỗ trợ động cơ điều khiển góc ngẩng (xử lý vấn đề này bằng chút thủ thuật kết hợp với Code Laboratories Kinect, xem thêm phần phụ lục 1).
    - Việc cài đặt có phần rối rắm hơn Kinect SDK.

Kết luận: Sinh viên cũng đã có thời gian làm việc trên cả hai thư viện, nên có những kết luận sau:

Kinect SDK beta mạnh hơn OpenNI ở đặc tính bám cơ thể người ổn định hơn, do đó sẽ đáp ứng chính xác với các cử chỉ cơ thể người; tuy nhiên để phát hiện cử chỉ ta phải tự viết giải thuật trong khi trên OpenNI đã hỗ trợ sẵn. Ngoài ra, Kinect SDK hơn OpenNI ở phần hỗ trợ cho xử lý âm thanh, cho phép xây dựng các ứng dụng điều khiển bằng giọng nói dễ dàng hơn.



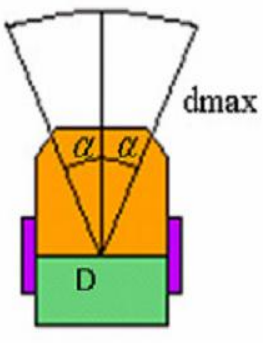
Hình 3.1: Thư viện OpenNI phối hợp giữa phần cứng và ứng dụng đầu cuối

OpenNI cho phép thu bản đồ độ sâu trong giới hạn từ 0.5 mét trở về phía trước Kinect trong khi Kinect SDK thì giới hạn này là 0.8 mét về phía trước, chất lượng bản đồ độ sâu thu về trên hai thư viện là như nhau. Kinect SDK là thư viện mới ra đời đang trong giai đoạn beta và cũng chưa có sự hỗ trợ của các thư viện xử lý ảnh khác như Point Cloud. Trong khi đó OpenNI ra đời trước và gần như được tích hợp với thư viện xử lý ảnh Point Cloud. Do đó, với ứng dụng Kinect trong đề tài robot tự hành tránh vật

Có thể nói PCL là sự kết hợp của nhiều module nhỏ. Những module này thực chất cũng là các thư viện thực hiện các chức năng riêng lẻ trước khi được PCL đóng gói. Các thư viện cơ bản này là:

- ***Eigen:*** một thư viện mở hỗ trợ cho các phép toán tuyến tính, được dùng trong hầu hết các tính toán toán học của PCL.
- ***FLANN:*** (Fast Library for Approximate Nearest Neighbors) hỗ trợ cho việc tìm kiếm nhanh các điểm lân cận trong không gian 3D.
- ***Boost:*** giúp cho việc chia sẻ con trỏ trên tất cả các module và thuật toán trong PCL để tránh việc sao chép trùng lặp dữ liệu đã được lấy về trong hệ thống.
- ***VTK:*** (Visualization Toolkit) hỗ trợ cho nhiều platform trong việc thu về dữ liệu 3D, hỗ trợ việc hiển thị, ước lượng thể tích vật thể.
- ***CMinPack:*** một thư viện mở giúp cho việc giải quyết các phép toán tuyến tính và không tuyến tính.

lúc thu sóng về, sau đó kết hợp với vận tốc sóng siêu âm (khoảng 343 m/s) để biết được quãng đường mà sóng đã đi.

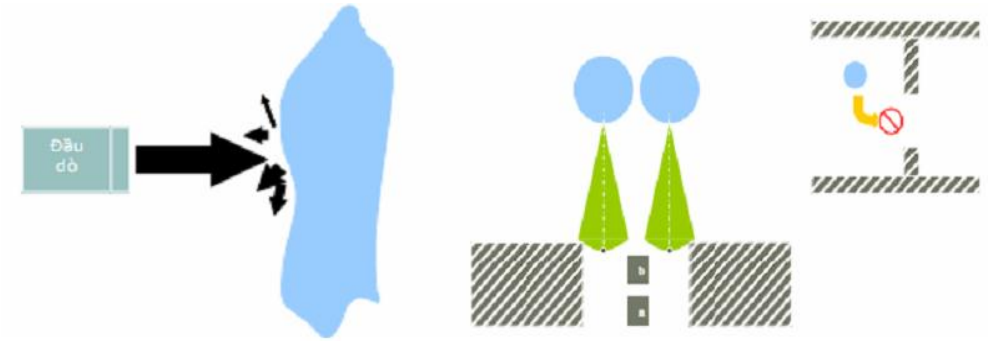


Hình 4.2: Cảm biến siêu âm

Các cảm biến được đặt lệch một góc α, khoảng cách lớn nhất (tính từ D) mà các cảm biến có thể nhận diện được là dmax; dmax và α phải đảm bảo sao cho cảm biến có vùng kiểm tra đủ rộng để khi tiến thẳng robot có thể nhận diện được vật cản.

- ***Ưu điểm:*** xử lý nhanh, kết quả tương đối chính xác.
- ***Khuyết điểm:***
  - Cảm biến siêu âm chỉ nhận biết được vật cản khi mặt phẳng quét của cảm biến cắt ngang vật cản, do đó nó sẽ không phát hiện ra những vật cản nhỏ, thấp và nằm sát mặt đất.
  - Do sử dụng sóng siêu âm và sự phản xạ của nó để tính khoảng cách và phát hiện vật cản nên giải thuật điều khiển khá phức tạp và phát sinh một số trường hợp sai số khó khắc phục: sai số lặp, hiện tượng Forecasting, hiện tượng đọc chéo (Crosstalk).
    - ***Sai số lặp:*** sai số lặp là sai số luôn xảy ra với tất cả các thiết bị đo lường, trong đó có cả cảm biến siêu âm.
    - ***Hiện tượng forecasting:*** hiện tượng Forecasting là hiện tượng phản xạ góc sai lệch của cảm biến. Theo nguyên lý TOF, để có khoảng cách đúng, cảm biến siêu âm phải hướng vuông góc với bề mặt chướng ngại vật cần đo. Tuy nhiên, các chướng ngại vật không bao giờ là phẳng, mịn nên tia phản

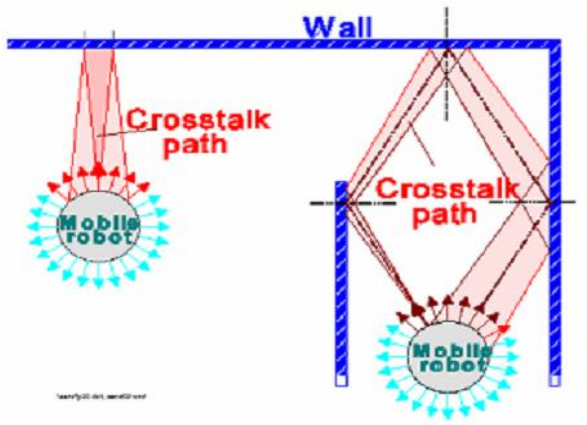
xạ có thể không tương ứng với góc tới. Các chùm tia phản xạ này có năng lượng phản xạ thấp hơn. Tuy vậy, ở một khoảng cách nào đó, cảm biến siêu âm vẫn có thể ghi nhận được những tín hiệu phản xạ này. Kết quả là thông số đọc về từ cảm biến siêu âm bị lệch do góc mở của cảm biến siêu âm lớn.



Hình 4.3: Hiện tượng Forecasting

Ngoài ra, vì góc mở rộng nên không chỉ sai về nhận dạng vị trí chướng ngại vật mà khoảng cách ghi nhận cũng bị sai lệch. Tuy nhiên sai số này không đáng kể như sai số do hiện tượng đọc chéo gây ra.

- ***Hiện tượng crosstalk:*** hiện tượng đọc chéo (Crosstalk) là hiện tượng mà cảm biến siêu âm này ghi nhận tín hiệu phản xạ hoặc trực tiếp từ cảm biến siêu âm khác; hoặc sau quá trình sóng siêu âm truyền đi và phản xạ qua các bề mặt nó quay lại cảm biến theo một cách không mong muốn.

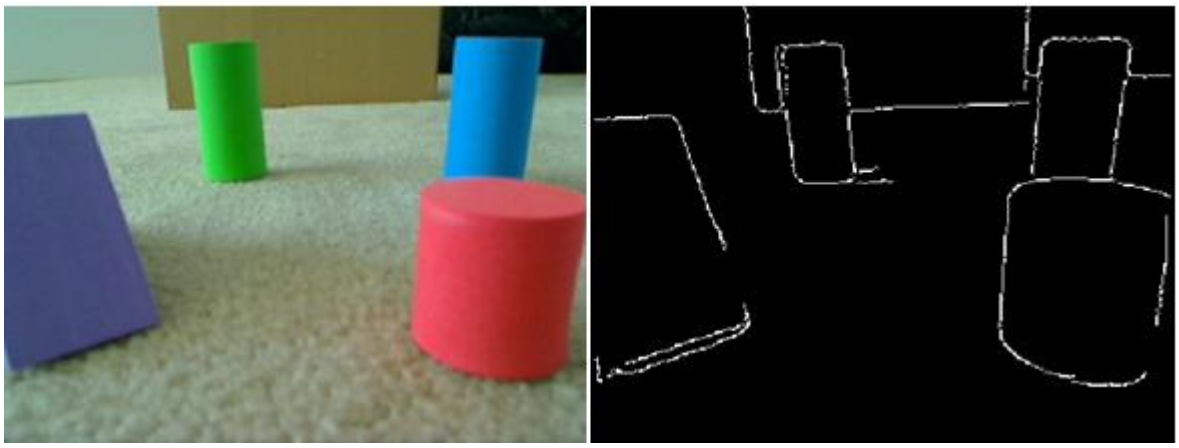


Hình 4.4: Hiện tượng Crosstalk

- Chỉ tối ưu đối với những vật có góc cạnh, có nhiều điểm đặc biệt. Giả sử nếu gặp một bức tường trắng, robot không thể phân biệt được phần tử nào là phần tử tương ứng khi xét từ frame này sang frame khác.
- Dễ bị nhiễu: nếu nền có hoa văn hay đường viền sẽ gây nhiễu do camera sẽ bám theo các phần tử đặc biệt trên nền. Ngoài ra, những vật trong thị trường của robot phải đứng yên, nếu có vật chuyển động ở xa nhưng vẫn trong thị trường của robot sẽ làm tăng optical flow khiến robot nhầm lẫn đang có vật cản ở trước mặt.
- Tốc độ xử lý chậm do yêu cầu tính toán nặng.

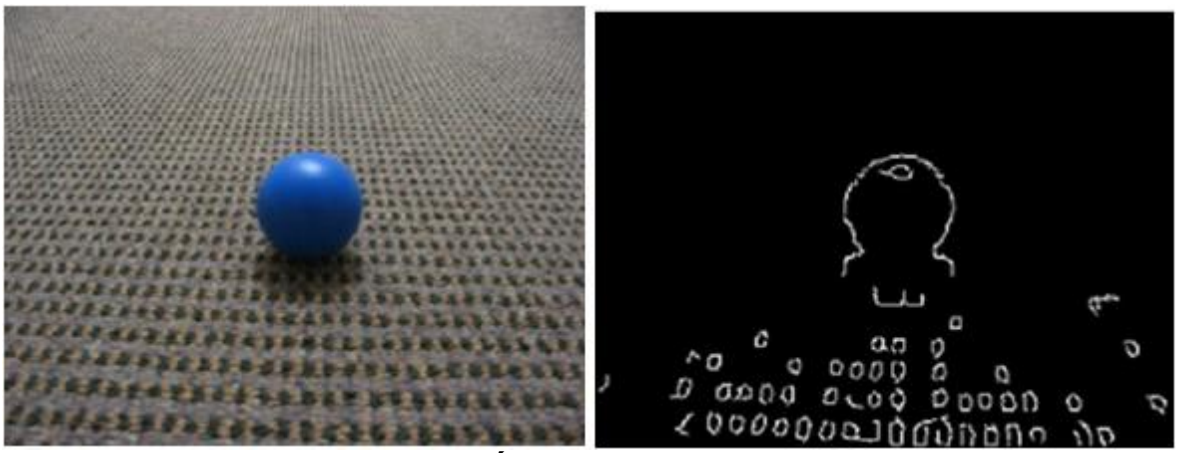
- ***Edge Detection [14]:***

+ *Phương pháp:* phương pháp này dùng những kỹ thuật tách biên (như Canny) cho ta ảnh chỉ hiển thị đường biên của vật thể như hình 4.6. Từ đó giúp ta phân biệt được nền và các vật cản. Vật cản sẽ là những vật có viền bao quanh, còn nền là vùng không gian còn lại.



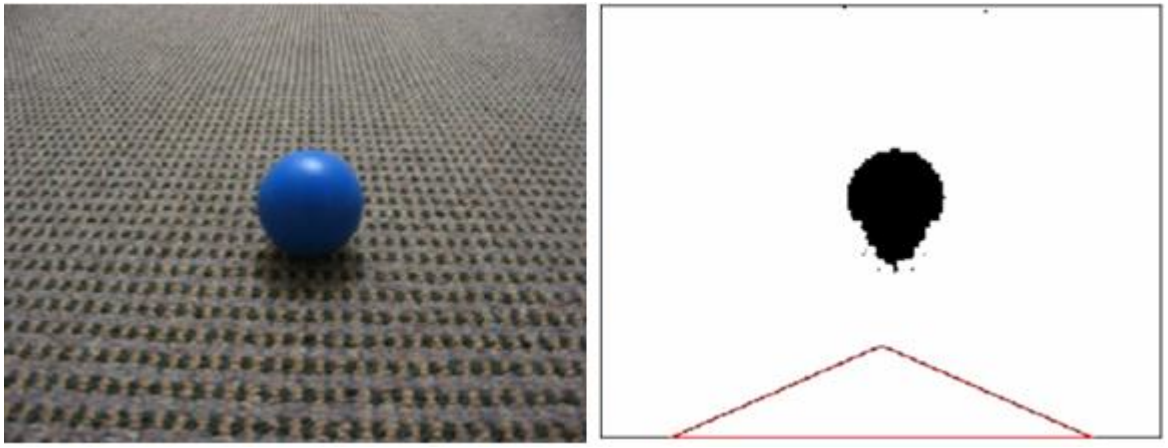
Hình 4.6: Ảnh gốc và ảnh sau khi tách biên

+ *Ưu điểm:* xử lý nhanh, dò tìm ra vật cản tốt, chính xác.
+ *Khuyết điểm:* chỉ hoạt động tốt trong điều kiện nền đơn sắc và không có hoa văn hay họa tiết.



Hình 4.7: Hạn chế của phương pháp dò biên

- ***Floor Finder Technique [14]:***
  - *Phương pháp:* phương pháp này dựa trên màu sắc của các điểm ảnh, những điểm ảnh không trùng màu với màu nền thì được xem là vật cản. Ta giả định vùng không gian nhỏ trước mặt robot là không có vật cản. Bằng cách lấy giá trị màu sắc các điểm ảnh trong vùng này, ta được một tập mẫu màu sắc của nền. So sánh giá trị màu của từng điểm ảnh còn lại trong hình với tập mẫu này ta sẽ xác định được điểm ảnh nào thuộc về nền, điểm ảnh nào thuộc về vật cản.



Hình 4.8: Phương pháp dò nền

  - *Ưu điểm:* đơn giản, hiệu quả, phát hiện vật cản chính xác, không phụ thuộc vào hình dạng và kích thước của vật cản. Đồng thời, tốc độ xử lý nhanh do yêu cầu tính toán không nhiều.

Nhận xét:

Với sự phát triển của công nghệ hiện nay thì tốc độ xử lý không còn là vấn đề khó khăn nữa; với mục đích tránh vật cản một cách tốt nhất thì phương pháp cuối cùng, "Stereo Vision" nổi trội hơn cả.

Sinh viên đã có thời gian thực hiện và hoàn thành việc phát hiện vật cản bằng phương pháp stereo vision trên hai camera. Kết quả phụ thuộc ở khâu hiệu chỉnh ban đầu rất nhiều. Vì thế, sinh viên tìm đến thiết bị có khả năng hỗ trợ việc lấy bản đồ độ sâu trực tiếp, mà không bận tâm nhiều đến việc hiệu chỉnh cho camera, đó là thiết bị chơi game Kinect.

Kinect cùng mục đích với phương pháp stereo vision là thu về bản đồ độ sâu nhưng cách thực hiện lại có chút khác biệt, dù sử dụng giải thuật tính toán tương tự nhau. Điều này đã được trình bày ở mục 2.3. Công nghệ mà Kinect sử dụng được xem như sự giao thoa giữa stereo vision và range finder (phương pháp đo đạc khoảng cách bằng sóng như laser, hồng ngoại hay sóng siêu âm). Kết quả thu về trên Kinect chính xác hơn, ổn định hơn, tiêu tốn tài nguyên máy tính ít hơn nhiều so với việc sử dụng hai camera trong phương pháp stereo vision. Do đây là một công nghệ mới nên sinh viên gọi là phương pháp xử lý ảnh trên Kinect và được trình bày chi tiết trong mục 4.3.

- ***Depth map:***

Hình 4.11: Ảnh RGB và bản đồ độ sâu

Depth map hay bản đồ độ sâu chứa thông tin vị trí vật trong không gian phía trước Kinect. Bản đồ độ sâu được OpenNI hỗ trợ ở các độ phân giải và tốc độ thu như sau:

- SXGA_15Hz: độ phân giải 1280×1024, tốc độ 15 fps.
- VGA_30Hz: độ phân giải 640x480, tốc độ 30 fps.
- QVGA_30Hz: độ phân giải 320x240, tốc độ 30 fps.

Việc lựa chọn độ phân giải càng nhỏ thì tốc độ xử lý khi qua thư viện Point Cloud càng được tăng lên.

- ***Point cloud:***

Hình 4.12: RGB point cloud

Point cloud **P** là một tập hợp các phần tử $\mathbf{p_i}$, mỗi phần tử này sẽ chứa tập các giá trị biểu diễn không gian nD (thường thì n = 3) [15]:

$$P = \{p_1, p_2, ..., p_i, ..., p_n\}, p_i = \{x_i, y_i, z_i\}$$

Bên cạnh thông tin dữ liệu là XYZ, mỗi điểm $\mathbf{p_i}$ còn có thể chứa thêm các thông tin khác như: RGB colors, intensity values, …

Mỗi giá trị $\mathbf{x_i}$, $\mathbf{y_i}$, $\mathbf{z_i}$ được lưu trữ bằng kiểu **float32**. Nếu ta chọn độ phân giải VGA thì ta có: **n =** 640x480 = 307200 phần tử **p,** đây là một con số khá lớn. Vì vậy:

- Xử lý chậm (dữ liệu lớn dẫn đến việc tính toán sẽ lâu hơn).
- Dữ liệu kiểu float 32 bit nên cần không gian lưu trữ lớn (cần RAM và HDD lớn).

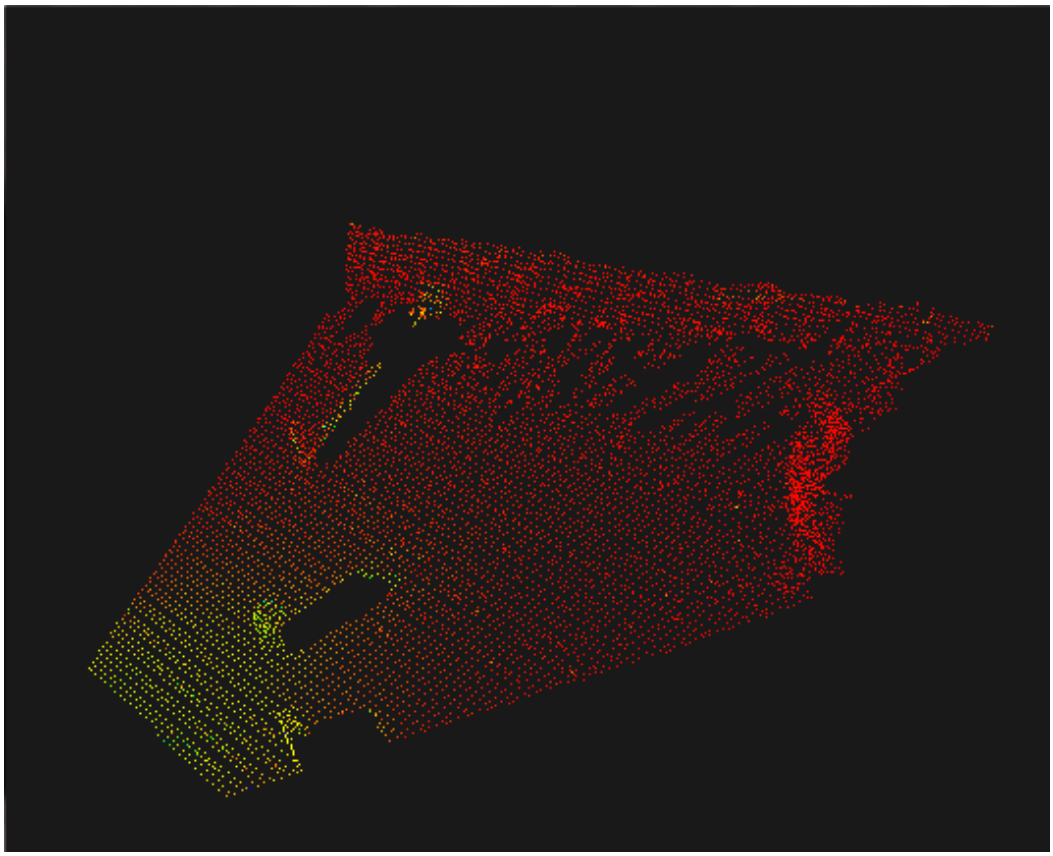
Máy tính sinh viên sử dụng có cấu hình như sau:

- ***CPU:*** Core i5-2430, 2.4 GHz
- ***RAM:*** 4 GB
- ***VGA:*** GeForce GT540 1GB

Tốc độ thu về point cloud đo được ở bước này vào khoảng 13÷15 fps.

Chính vì hạn chế này mà ta cần phải thực hiện các bước lọc tiền xử lý để đáp ứng được yêu cầu xử lý thời gian thực. Công việc này được PCL định nghĩa là "*downsampling*" và "*removing points*". Hai bước *downsampling* và *removing points* được sử dụng là ***Voxel Grid*** và ***Pass Through.***

- ***Voxel grid:***

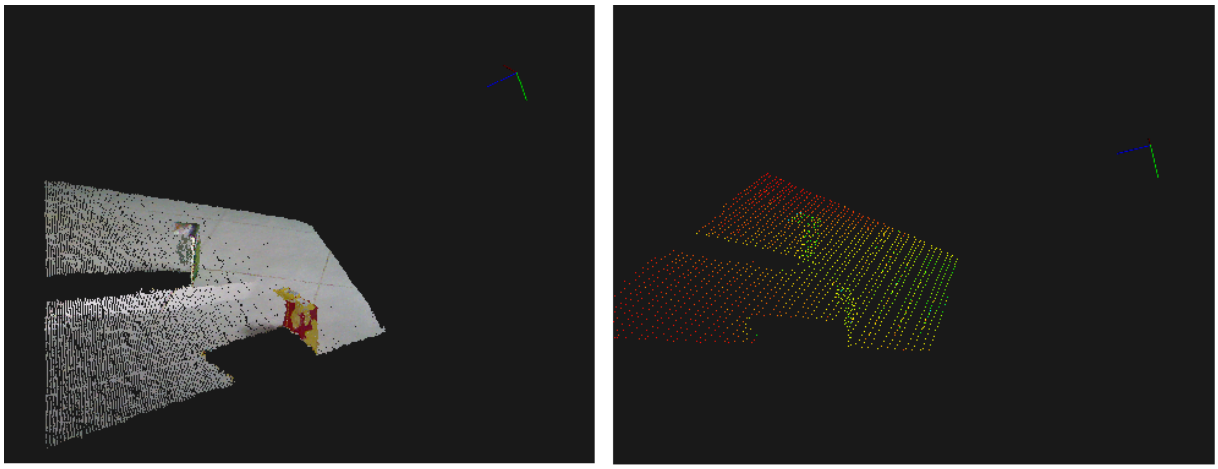


Hình 4.13: Voxel grid

Voxel grid làm giảm mật độ số điểm xuống; tập hợp các điểm quá gần nhau sẽ chỉ cần một điểm đại diện. Ta chọn giá trị mật độ phù hợp mà vẫn đảm bảo quan sát rõ hình dạng vật thể. Mật độ ta chọn ở đây là 3 centimet theo ba chiều X, Y và Z.

```
Hàm hỗ trợ: setLeafSize(0.03f, 0.03f, 0.03f);
```

- ***Pass through:***



Hình 4.14: Pass through không có (trái) và có voxel grid (phải)

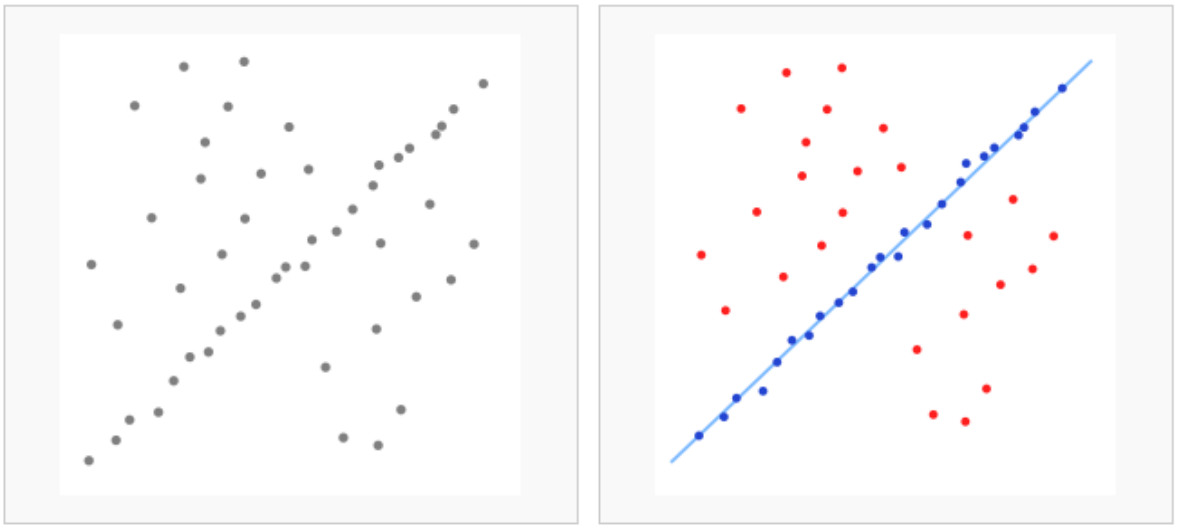
Pass through sẽ giới hạn không gian của point cloud theo các chiều X, Y và Z. Ở đây ta chỉ giới hạn theo chiều Z. Như đã phân tích trong mục 2.3, giá trị khoảng cách theo chiều Z mà Kinect có thể nhìn thấy khoảng 0.5÷5.0 mét, đây là tầm nhìn không cần thiết cho mobile robot, việc giới hạn lại sẽ giúp cho PCL xử lý nhanh hơn. Vì thế, ta chỉ cần cho Kinect nhìn trong khoảng 0.5÷1.4 mét, là tầm quan sát đủ cho ứng dụng robot tránh vật cản. Tốc độ thu hình sau khi kết hợp pass through và voxel grid (Hình 4.14 (phải)) đạt giá trị 30 fps, tức đạt tốc độ tối đa.

```
Hàm hỗ trợ:  setFilterFieldName ("z");
             setFilterLimits(0.5, 1.4);
```

- ***Plannar segmentation:***

Plannar segmentation sẽ tách các point cloud có cấu trúc phẳng, sau đó tách ra point cloud có tổng số điểm lớn nhất, bằng giải thuật RANSAC (RANdom SAmple Consensus). Đây là giải thuật ước lượng một mô hình toán học từ một tập hợp các điểm có chứa nhiễu (outliers). Giải thuật này lần đầu được công bố vào năm 1981 bởi Fischler và Bolles [16].

Hình 4.15 cho ta thấy ứng dụng thuật toán RANSAC tìm mô hình đường thẳng (có dạng $ax + by + c = 0$) trong một tập hợp các điểm có chứa nhiễu.



A data set with many outliers for which a line has to be fitted.

Fitted line with RANSAC, outliers have no influence on the result.

Hình 4.15: Tìm mô hình đường thẳng bằng thuật toán RANSAC

Giải thuật tìm đường thẳng bằng thuật toán RANSAC được mô tả như sau:

- ***Đầu vào:***
  data - tập hợp các điểm
  k - số lần lặp
  t - ngưỡng (threshold) sai số để xác định điểm nào đó có khớp mô hình không
- ***Đầu ra:***
  best_model - mô hình tốt nhất
  best_consensus_set - tập hợp các điểm khớp với best_model
  best_model = null
  best_consensus_set = null
  best_num_points = 0
- ***Lặp k lần***:
  consensus_set = tập hợp 2 điểm ngẫu nhiên thuộc data
  model = mô hình đường thẳng suy ra từ 2 điểm trên

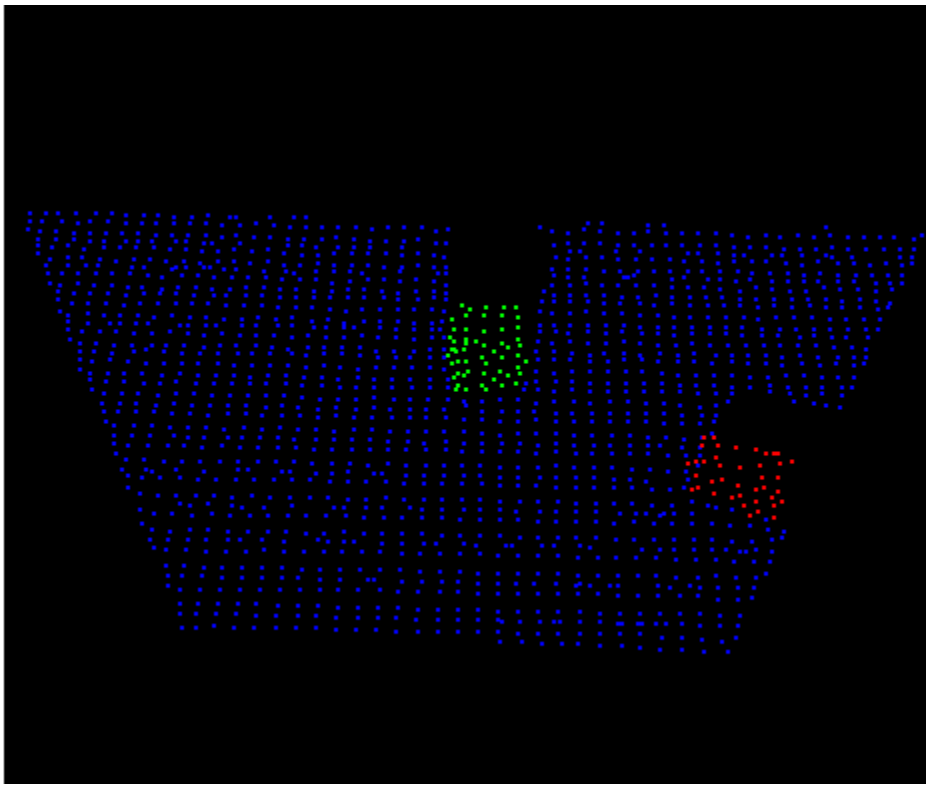
  Với mỗi điểm thuộc data *nhưng không thuộc* consensus_set, *ta xét:*
  distance = khoảng cách từ điểm đến đường thẳng;
  if distance < t (điểm thuộc mô hình nếu sai số nhỏ hơn mức ngưỡng đặt trước)
  thêm điểm đó vào consensus_set

  num_points = số lượng phần tử trong consensus_set
  if num_points > best_num_points
  best_model = model
  best_consensus_set = consensus_set
  best_num_points = num_points
- ***Trả về giá trị:*** best_model và best_consensus_set

Với mô hình mặt phẳng α có dạng *ax + by + cz + d =0*, thuật toán RANSAC cũng làm được điều tương tự như đối với mô hình đường thẳng, thay vì chọn hai điểm bất kỳ để tìm mô hình thì mặt phẳng cần ba điểm. Quan sát hình 4.16, point cloud có màu xanh dương là mặt phẳng nền nhà, được tìm ra nhờ thuật toán RANSAC.

```
Hàm hỗ trợ:  setModelType(pcl::SACMODEL_PLANE);
             setMethodType(pcl::SAC_RANSAC);
             setDistanceThreshold(0.02);//threshold = 2cm
```

- ***Euclidean cluster extraction:***

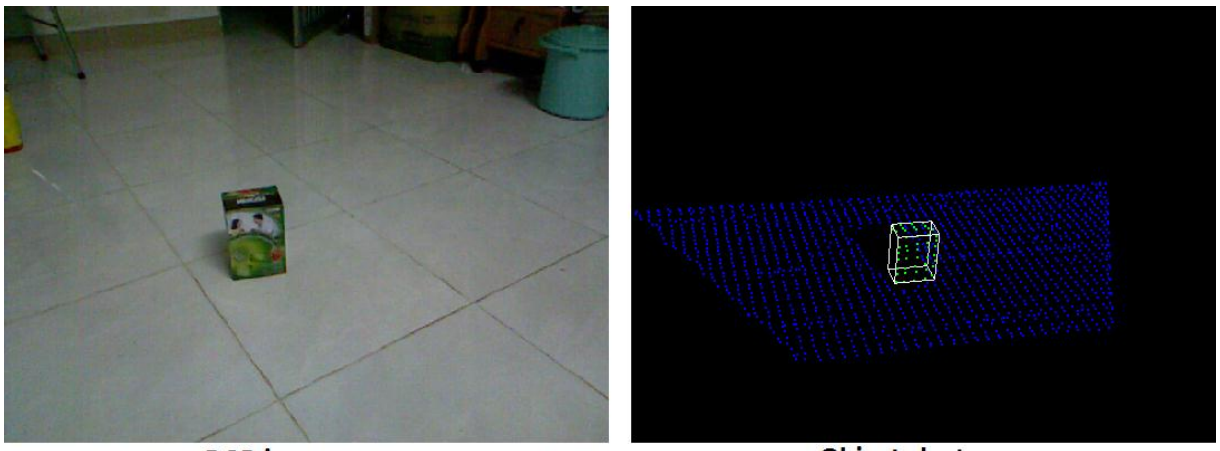


Hình 4.16: Point cloud sau khi thực hiện xong bước lọc (filtering) và phân đoạn (segmentation)

Euclidean cluster extraction làm công việc tách các point cloud có mặt trên nền nhà, tập hợp các điểm gần nhau sẽ được nhóm lại thành một point cloud hay cluster, mỗi cluster đại diện một vật thể. Hình 4.16 cho ta hai cluster: cluster màu đỏ (cluster gần Kinect) và cluster màu xanh lá (cluster xa Kinect) trên mặt phẳng nền nhà (point cloud màu xanh dương).

```
Hàm hỗ trợ:  setInputCloud(cloud_filtered);
             setIndices(inliers);
             filter(*cloud_cluster);
```

- ***Object clusters:***



Hình 4.17: Object cluster

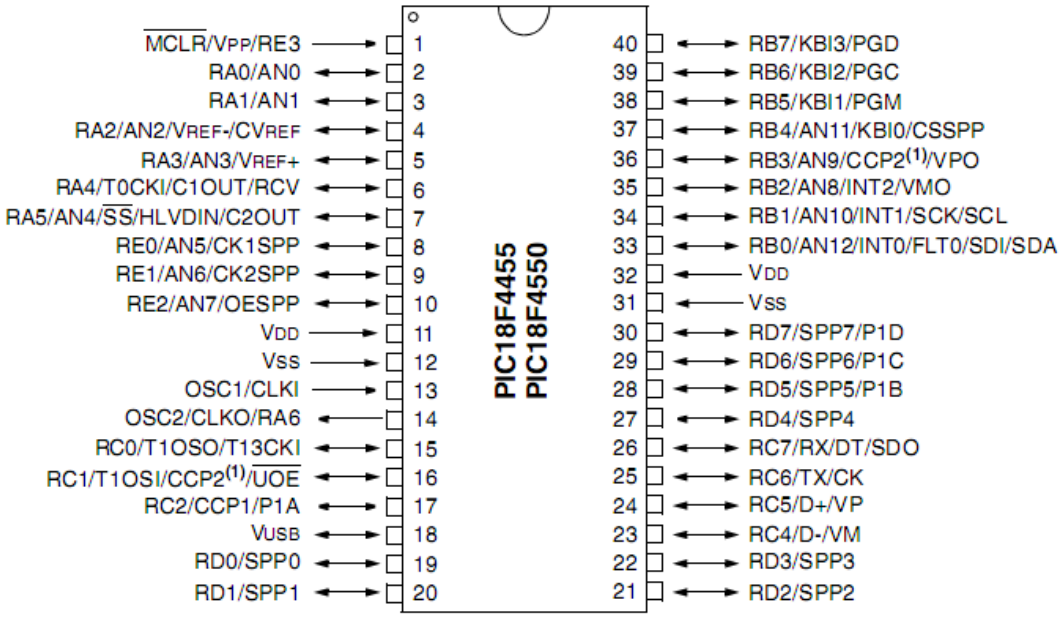
Object clusters là các vật cản mà ta có được sau bước Euclidean cluster extraction. Bước này làm nhiệm vụ phân tích đặc tính vật cản về kích thước cũng như vị trí vật trong không gian phía trước Kinect. Đây là những thông tin cần thiết cho kế hoạch tránh vật cản của mobile robot. Công việc này được thực hiện trên từng cluster, và đã được hỗ trợ hàm cần thiết bởi PCL.

```
Hàm hỗ trợ:  getMinMax3D(&cloud_cluster, min_point, max_point);
```

PIC 18F4550 dùng bộ nhớ Flash, có thể ghi nhiều lần, dung lượng lớn đáp ứng được hầu hết các ứng dụng trong thực tế:

- Điện áp hoạt động rộng từ 2V đến 5.5V.
- Bộ nhớ chương trình Flash 32K ô nhớ cho phép ghi 100,000 lần. Bộ nhớ dữ liệu RAM có 2048 Bytes gồm các thanh ghi chức năng đặc biệt và các thanh ghi đa mục đích. Ngoài ra, PIC18F4550 còn được tích hợp 256 Bytes EEPROM cho phép ghi đến 1,000,000 lần.
- Có 5 Port I/O với 34 chân (Port A, Port B, Port C, Port D, Port E).
- Có 13 kênh đọc ADC 10 bit.
- Có 1 kênh CCP (Capture/Compare/PWM) và 1 kênh ECCP (Enhanced Capture/Compare/PWM).
- Giao tiếp SSP (Synchronous Serial Port) và MSSP (Master Synchronous Serial Port).
- Module Enhanced USART hỗ trợ RS-485, RS-232.
- Có 1 timer 8 bit, 3 timer 16 bit.
- Có 3 ngắt ngoài.

Sau đây là sơ đồ chân của PIC18F4550 trong hộp DIP-40:



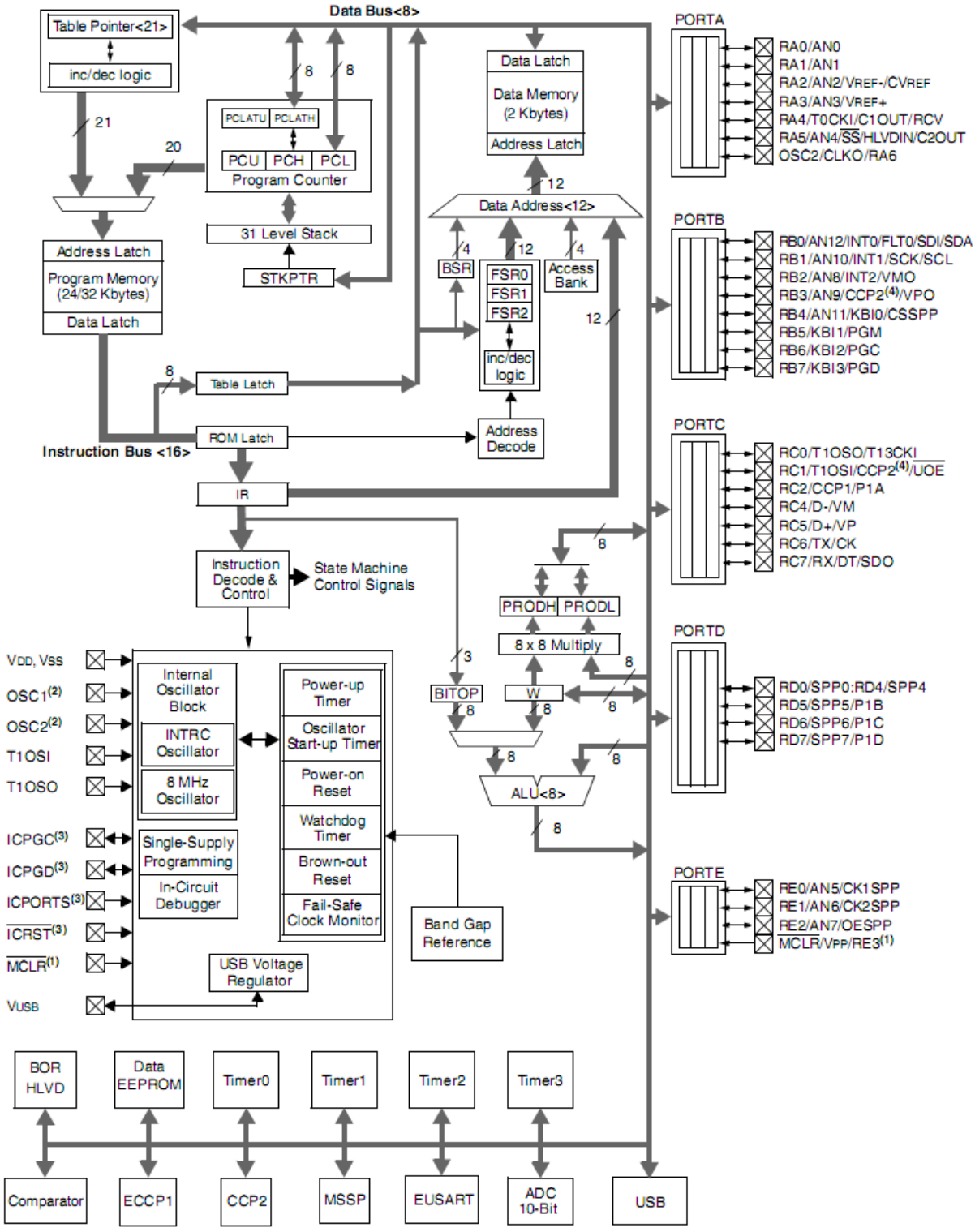
Hình 5.3: Sơ đồ chân PIC18F4550

Sau đây là bảng hệ thống chức năng các chân và sơ đồ khối của PIC18F4550:

Chân	Hướng	Mô tả chức năng và các đặc tính
AN0-AN12	I	13 kênh Input có tác dụng là cổng biến đổi ADC
Avdd		Nguồn dương cho module ADC
Avss		GND cho module ADC
CLKI	I	Lối vào của xung Clock ngoài, luôn kết hợp với chân OSC1
CLKO	O	Lối ra của bộ dao động tinh thể, nối với tinh thể hoặc bộ cộng hưởng trong chế độ dao động thạch anh. Luôn kết hợp với chân chức năng OSC2
CN0 - CN7	I	Khai báo thay đổi lối vào
CN17 - CN18		
COFS	I/O	Cổng giao tiếp chuyển đổi dữ liệu đồng bộ khung
CSCK	I/O	Cổng giao tiếp chuyển đổi dữ liệu Clock vào ra nối tiếp
CSDI	I	Lối vào dữ liệu nối tiếp
CSDO	O	Lối ra dữ liệu nối tiếp
C1RX	I	Cổng nhận bus CAN1
C1TX	O	Cổng phát bus CAN1
EMUD	I/O	Cổng vào ra dữ liệu kênh truyền thông sơ cấp của ICD
EMUC	I/O	Vào ra xung nhịp kênh sơ cấp
EMUD1	I/O	Vào ra dữ liệu kênh thứ cấp
EMUC1	I/O	Vào ra dữ liệu kênh thứ cấp
EMUD2	I/O	Vào ra dữ liệu kênh thứ cấp
EMUC2	I/O	Vào ra dữ liệu kênh thứ cấp
EMUD3	I/O	Vào ra dữ liệu kênh thứ cấp
EMUC3	I/O	Vào ra dữ liệu kênh thứ cấp
IC1 - IC8	I	Các cổng vào của module Capture
INT0 - INT2	I	Các ngắt ngoài
LVDIN	I	Cổng vào phát hiện sụt thế
/MCLR	I	Chân Reset, mức tích cực thấp

OSC1	I	Lối vào bộ giao động tinh thể. Bộ đệm Trigger Schmitt được sử dụng khi cấu hình trong chế độ RC
OSC2	O	Lối ra bộ dao động tinh thể
PGD	I/O	Vào ra dữ liệu của ICSP
PGC	I	Lối vào Clock của ICSP
RA0 - RA6	I/O	Port A
RB0 - RB7	I/O	Port B
RC0 - RC7	I/O	Port C
RD0 - RD7	I/O	Port D
RE0 - RE3	I/O	Port E
SCK1	I/O	Vào ra Clock đồng bộ của khối SPI1
SDI1	I	Lối vào dữ liệu của khối SPI1
SDO1	O	Lối ra dữ liệu của SPI1
SS1	I	Slaver đồng bộ
SCL	I/O	Vào ra Clock nối tiếp của I2C
SDA	I/O	Vào ra Data nối tiếp đồng bộ của I2C
SOSCO	O	Lối ra bộ dao động tinh thể công suất thấp 32Khz
SOSCI	I	Lối vào bộ dao động 32Khz
T1CK	I	Lối vào xung Clock ngoài của Timer1
T2CK	I	Lối vào xung Clock ngoài của Timer2
U1RX	I	Cổng nhận khối UART1
U1TX	O	Cổng phát khối UART1
U1ARX	I	Cổng nhận mở rộng khối UART1
U1ATX	O	Cổng phát mở rộng khối UART1
VDD		Chân nguồn dương của PIC
VSS		Chân GND
Vref+	I	Lối vào Vref+ (cao) chuẩn của ADC
Vref-	I	Lối vào Vref- (thấp) chuẩn của ADC

Bảng 5.1: Bảng mô tả các chức năng từng chân của PIC18F4550



Hình 5.4: Sơ đồ khối của PIC 18F4550

Hoạt động của bộ dao động trong PIC 18F4550 được điều khiển thông qua hai thanh ghi Configuration và hai thanh ghi control. Các thanh ghi CONFIG1L và CONFIG1H lựa chọn chế độ dao động và các options cho prescaler/postcaler của USB. Vì là các bits Configuration nên chúng được set khi thiết bị được lập trình và giữ nguyên cấu hình này cho đến khi thiết bị được lập trình mới lại. Thanh ghi OSCCON lựa chọn chế độ Active Clock, nó được dùng chủ yếu trong việc điều khiển clock switching trong chế độ quản lý năng lượng. Thanh ghi OSCTUNE dùng cho việc loại bỏ nguồn tần số INTRC, cũng như lựa chọn nguồn clock tần số thấp.

PIC 18F4550 có thể thực hiện 12 chế độ dao động khác nhau. Người lập trình có thể điều chỉnh các bit Configuration FOSC3:FOSC0 để lựa chọn chế độ dao động thích hợp:

- ***XT:*** Crystal/Resonator
- ***XTPLL:*** Crystal/Resonator with PLL enabled
- ***HS:*** High-Speed Crystal/Resonator
- ***HSPLL:*** High-Speed Crystal/Resonator with PLL enabled
- ***EC:*** External Clock with FOSC/4 output
- ***ECIO:*** External Clock with I/O on RA6
- ***ECPLL:*** External Clock with PLL enabled and FOSC/4 output on RA6
- ***ECPIO:*** External Clock with PLL enabled, I/O on RA6
- ***INTHS:*** Internal Oscillator used as microcontroller clock source, HS Oscillator used as USB clock source
- ***INTXT:*** Internal Oscillator used as microcontroller clock source, XT Oscillator used as USB clock source
- ***INTIO:*** Internal Oscillator used as microcontroller clock source, EC Oscillator used as USB clock source, digital I/O on RA6
- ***INTCKO:*** Internal Oscillator used as microcontroller clock source, EC Oscillator used as USB clock source, FOSC/4 output on RA6

Clock hệ thống của PIC18F4550 có thể được chọn từ hai nguồn dao động nội (Internal Oscillator) hoặc dao động ngoại (Primary Oscillator và Secondary Oscillator)

nhờ bộ chọn kênh MUX. Bộ MUX được điều khiển bởi các bit FOSC<3:0> (bit 3, bit 2, bit 1, bit 0 của thanh ghi CONFIG1 16-bit định vị tại địa chỉ 2007H và 2008H trong bộ nhớ chương trình) và các bit SCS<1:0> (bit 1, bit 0 của thanh ghi OSCCON).Các bits SCS dùng để lựa chọn chế độ Primary Clock, T1OSC hay Internal Oscillator. Các bit FOSC<3:0> được sử dụng để cấu hình bộ dao động Primary Clock.

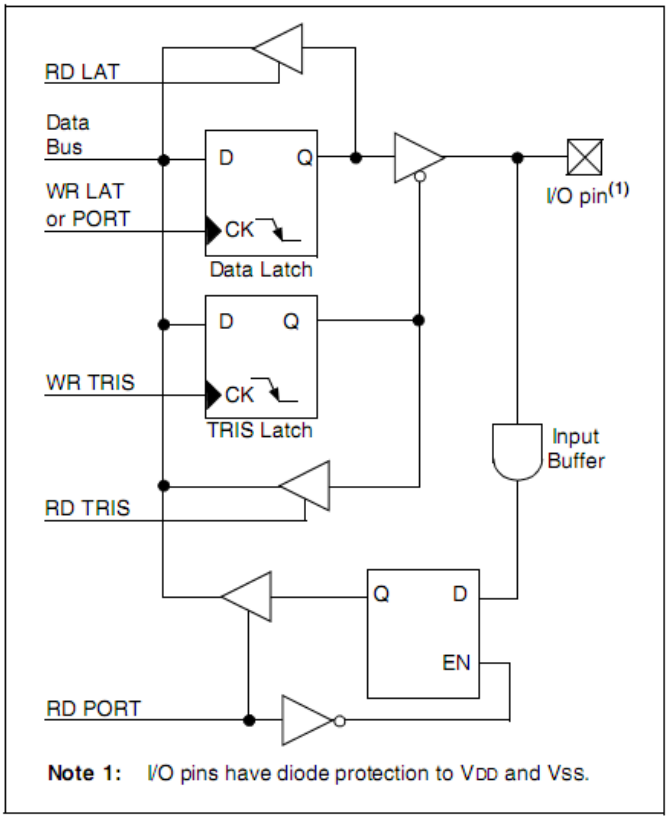
Bộ dao động nội gồm 2 bộ dao động HFINTOSC 8MHz và LFINTOSC 31kHz. Clock 8MHz của bộ HFINTOSC được chia thành các tần số 8MHz, 4MHz, 2MHz, 1MHz, 500kHz, 250kHz, 125kHz nhờ bộ chia tần số postscaler. Tần số nguồn clock (INTOSC direct, INTRC direct hoặc INTOSC postscaler) được lựa chọn bằng việc điều chỉnh các bits IRCF trong thanh ghi OSCCON.

Bộ dao động ngoại (được tích hợp bên trong PIC) cần được kết nối với các bộ lọc tại các chân OSC1, OSC2. Trong đề tài này, ta sử dụng thạch anh 12MHz để tạo thành xung clock 12MHz cung cấp cho clock hệ thống.

- **Các port I/O:**

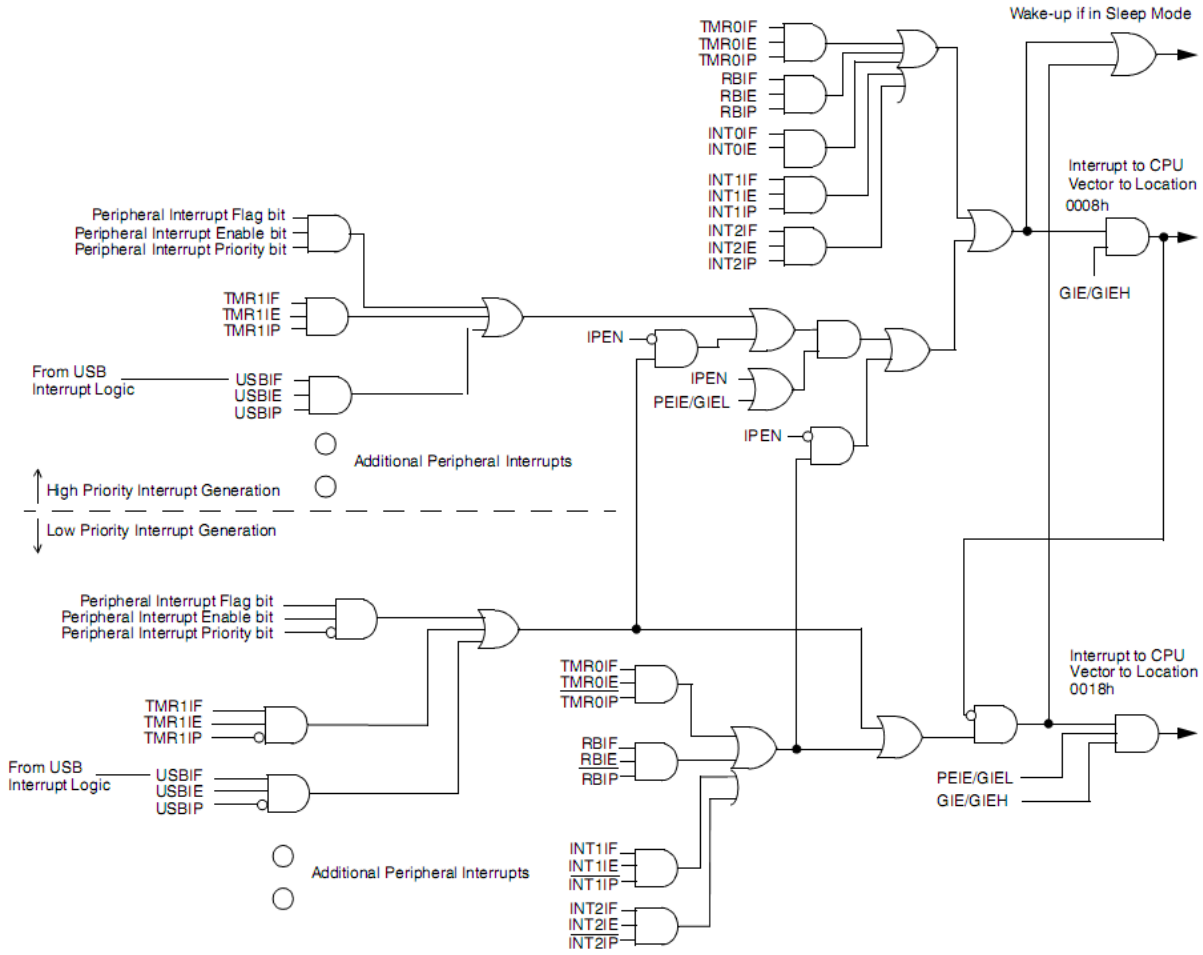
PIC18F4550 tất cả 34 chân I/O mục đích thông thường (GPIO: General Purpose Input Ouput) có thể được sử dụng. Tùy theo những thiết bị ngoại vi được chọn mà một vài chân có thể không được sử dụng ở chức năng GPIO. Thông thường, khi một thiết bị ngoại vi được chọn, những chân liên quan của thiết bị ngoại vi có thể không được sử dụng ở chức năng GPIO. 34 chân GPIO được chia cho 5 Port: Port A gồm 7 chân, Port B gồm 8 chân, Port C gồm 8 chân, Port D gồm 8 chân và Port E gồm 3 chân.

Mỗi port được điều khiển bởi 2 thanh ghi 8-bit, thanh ghi Port và thanh ghi Tris. Thanh ghi Tris được sử dụng để điều khiển port là nhập hay xuất. Mỗi bit của Tris sẽ điều khiển mỗi chân của port đó, nếu giá trị của bit là 1 thì chân liên quan là nhập, ngược lại nếu giá trị của bit là 0 thì chân liên quan là xuất. Thanh ghi Port được sử dụng để chứa giá trị của port liên quan. Mỗi bit của thanh ghi Port sẽ chứa giá trị của chân liên quan.



Hình 5.6: Cấu tạo của chân GPIO

- **Ngắt ngoài trên các chân RB:**



Hình 5.7: Sơ đồ khối logic của hệ thống ngắt trong PIC18F4550

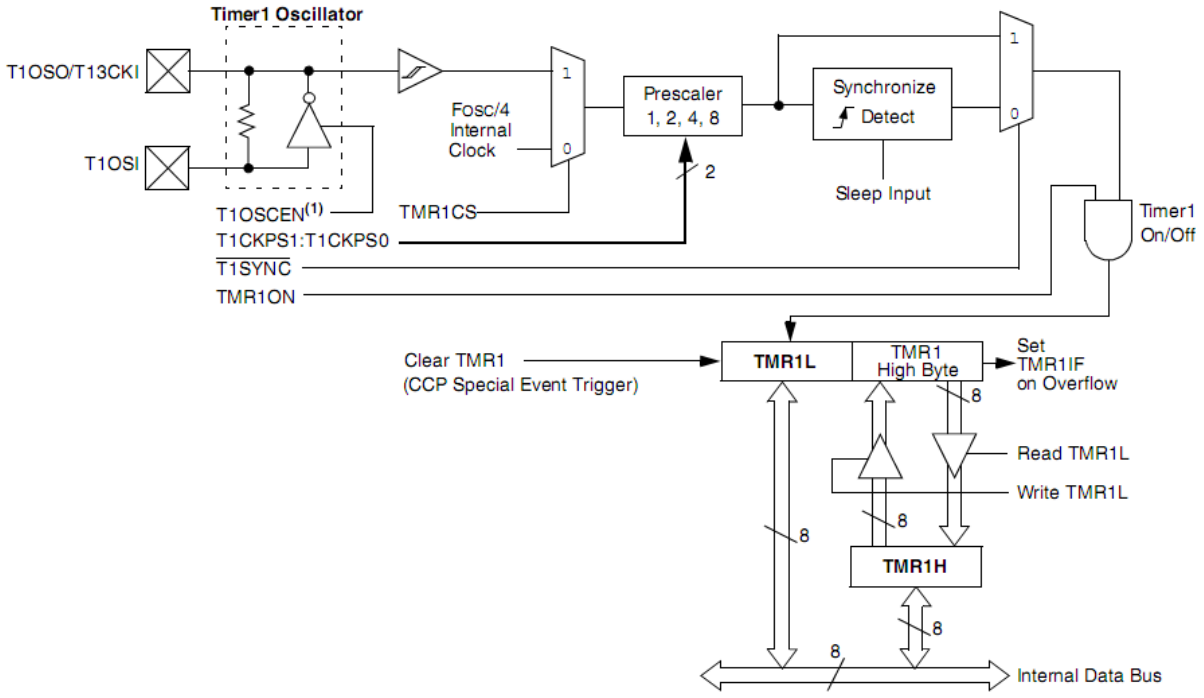
Các ngắt ngoài INT0 ứng với chân RB0, INT1 ứng với chân RB1, INT2 ứng với chân RB2.

Các ngắt ngoài trên các chân RB được kích khởi theo sườn. Sườn lên nếu như bit INTEDG = 1 (bit 6 của thanh ghi OPTION_REG), sườn xuống nếu INTEDG = 0. Khi một sườn thích hợp xuất hiện trên chân RB, cờ INTF được bật lên 1. Ngắt này có thể được cho phép nếu bit INTE=1, không cho phép nếu INTE=0. Cờ INTF cần được xóa bằng phần mềm trong trình phục vụ ngắt trước khi cho phép ngắt trở lại.

Trong đề tài này, ta dùng ngắt ngoài INT1 (trên chân RB1) và INT2 (trên chân RB2) để đọc encoder về phục vụ cho việc tính toán PID.

- **Cấu tạo và hoạt động của bộ Timer 1 và Timer 2:**

Bộ Timer 1 là bộ định thời 16-bit có cấu tạo như hình sau:

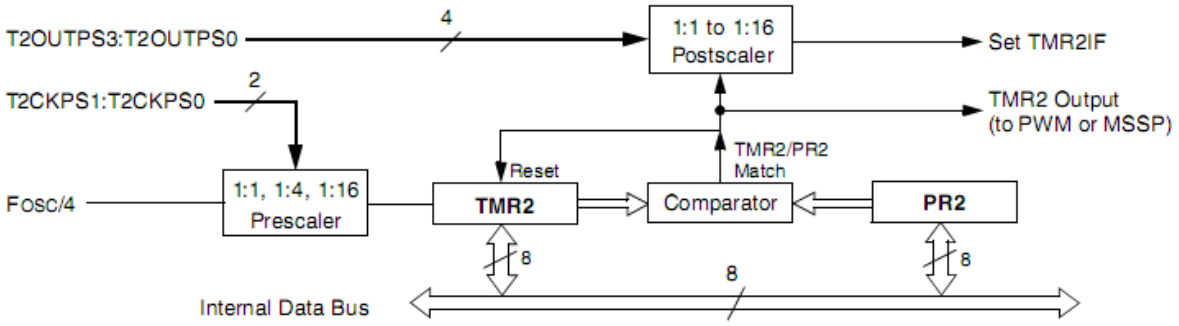


Hình 5.8: Sơ đồ khối của bộ Timer 1

Bộ Timer 1 là bộ đếm lên 16-bit được truy xuất gián tiếp thông qua cặp thanh ghi TMR1H, TMR1L. Khi được đọc hoặc ghi các thanh ghi này sẽ cập nhật trực tiếp giá trị cho bộ Timer. Khi được sử dụng với nguồn clock nội, bộ Timer 1 sẽ có vai trò là bộ định thời. Khi được sử dụng với nguồn clock ngoại, nó sẽ có vai trò là định thời hoặc bộ đếm. Sử dụng bit TMR1CS để chọn nguồn clock.

Các bit T1CKPS<1:0> định giá trị cho bộ chia tần số Prescaler. Khi bộ TMR1 tràn (từ FFFFh đến 0000h) cờ ngắt TMR1IF sẽ được thiết lập lên 1. Nếu lúc này cờ TMR1IE =1, cờ PEIE=1 và GIE=1 thì ngắt Timer1 sẽ xảy ra. Cờ TMR1IF cần được xóa trong trình phục vụ ngắt Timer 1. Trong đề tài này, ta dùng Timer 1 cho việc lấy mẫu của bộ PID số.

Sau đây là sơ đồ khối bộ Timer 2:



Hình 5.9: Sơ đồ khối của bộ Timer 2

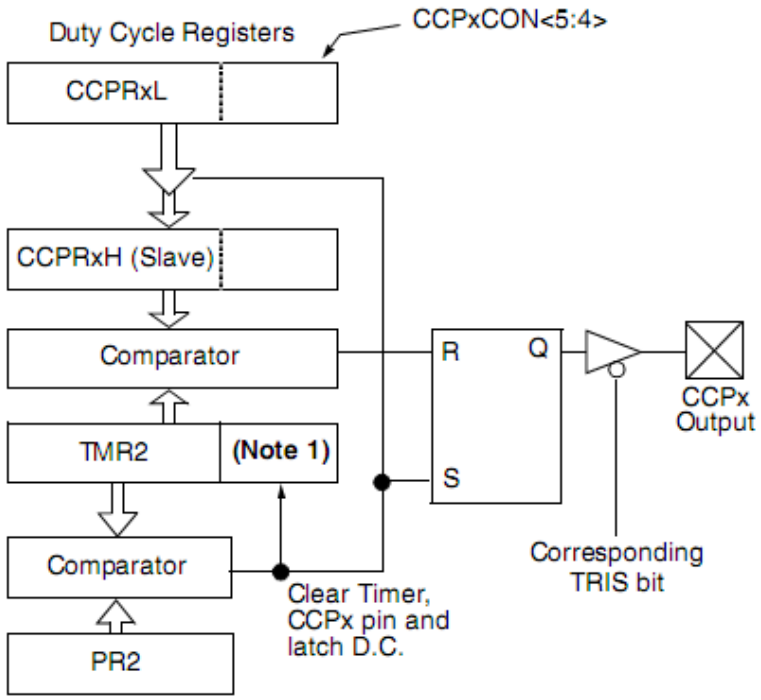
TMR2 tăng từ 00H với mỗi clock (FOSC/4), hai bit counter/prescaler trên ngõ vào clock cho ngõ vào trực tiếp với lựa chọn divide-by-4 hoặc divide-by-16 prescale. Chúng được lựa chọn bằng bit prescaler control, T2CKPS1:T2CKPS0 (T2CON<1:0>). Giá trị của TMR2 được so sánh với giá trị của thanh ghi chu kì, PR2, trong mỗi chu kì clock. Khi hai giá trị này tương ứng nhau, bộ so sánh tạo ra một tín hiệu điều khiển ở ngõ ra của bộ Timer, tín hiệu này cũng reset lại giá trị của TMR2 về 00H ở chu kì kế tiếp và lái ngõ ra bộ counter/postscaler. Các thanh ghi TMR2 và PR2 đều có thể đọc và ghi. Thanh ghi TMR2 bị xóa khi có một thiết bị nào đó reset, trong khi đó thanh ghi PR2 khởi tạo ở FFH. Ngõ ra không chia tỉ lệ của TMR2 chủ yếu được dùng cho module CCP, cơ sở thời gian cho các phép toán trong chế độ PWM.

Trong đề tài này, ta dùng Timer 2 cho việc điều xung PWM.

- **Cấu tạo và hoạt động của khối điều xung PWM:**

PIC18F4550 có hai bộ điều xung, hai bộ này sẽ tạo ra các tín hiệu điều xung trên các chân CCP1 và CCP2. Độ rộng, chu kỳ và độ phân giải của hai bộ điều xung được xác định bởi các thanh ghi PR2, T2CON, CCPR1L, CCPR2L, CCP1CON, CCP2CON. Ở chế độ điều xung (PWM), chân CCPx có thể tạo ra đầu ra PWM với độ phân giải lên đến 10 bit.

Vì chân CCPx được nhập chung với đường dữ liệu ở port B hoặc port C nên để các chân CCPx (CCP1 và CCP2) hoạt động ở chế độ PWM, cần xóa bit TRIS tương ứng của các chân đó. Sơ đồ khối của các bộ điều xung được mô tả trong hình dưới đây:

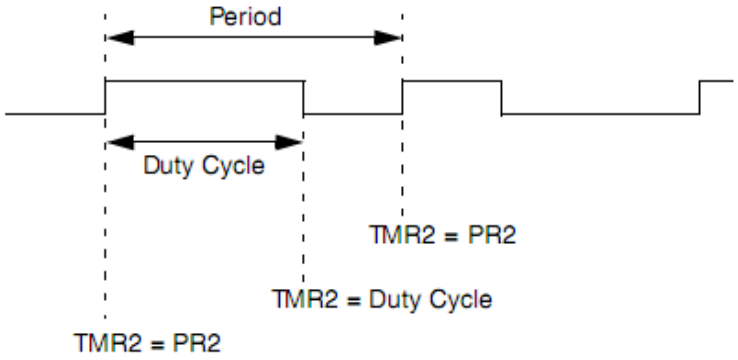


Note 1: The 8-bit TMR2 value is concatenated with the 2-bit internal Q clock, or 2 bits of the prescaler, to create the 10-bit time base.

Hình 5.10: Sơ đồ khối bộ PWM

Chú thích (Note 1) trong hình trên biểu thị rằng thanh ghi 8-bit TMR2 được kết hợp với 2-bit prescaler của bộ dao động nội để tạo ra bộ định thời 10-bit. Các thanh ghi CCPRxH là các thanh ghi chỉ đọc, kết hợp với 2 bit 5 và 4 của các thanh ghi CCPxCON có vai trò định độ rộng của xung; các thanh ghi này được ghi gián tiếp thông qua các thanh ghi CCPRxL. Thanh ghi 8-bit PR2 định chu kỳ cho xung ra.

Sóng điều xung tại các chân CCPx có giản đồ thời gian như sau:



Hình 5.11: Giản đồ thời gian của sóng điều xung tại chân CCPx

Thanh ghi TMR2 kết hợp với 2 bit prescaler sẽ đếm lên nhờ xung clock của hệ thống. Khi giá trị của TMR2 nhỏ hơn giá trị của CCPRxL:CCPxCON<5:4>, chân CCPx ở mức cao. Khi giá trị của TMR2 bằng với giá trị này, bộ so sánh sẽ đảo chân CCPx xuống mức 0. Khi giá trị của TMR2 bằng với PR2, TMR2 sẽ được xóa về 0 đồng thời kết thúc chu kỳ xung, chân CCPx lại được thiết lập ở mức cao.

Chu kỳ của xung được tính theo công thức sau:

Chu kỳ PWM = [(PR2)+1] × 4 × T_{OSC} × (giá trị Pre-scale của TMR2)

Trong đó:

T_{OSC} là chu kỳ của clock hệ thống. $T_{OSC} = 1/F_{OSC.}$

Trong đề tài, ta cài đặt: *setup_timer_2(T2_DIV_BY_4, 255, 1),* do đó

$T_{PWM} = (PR2+1) \times 4 \times T_{OSC} \times \text{Pre-scale} = 256 \times 4 \times (12 \times 10^6)^{-1} \times 4.$

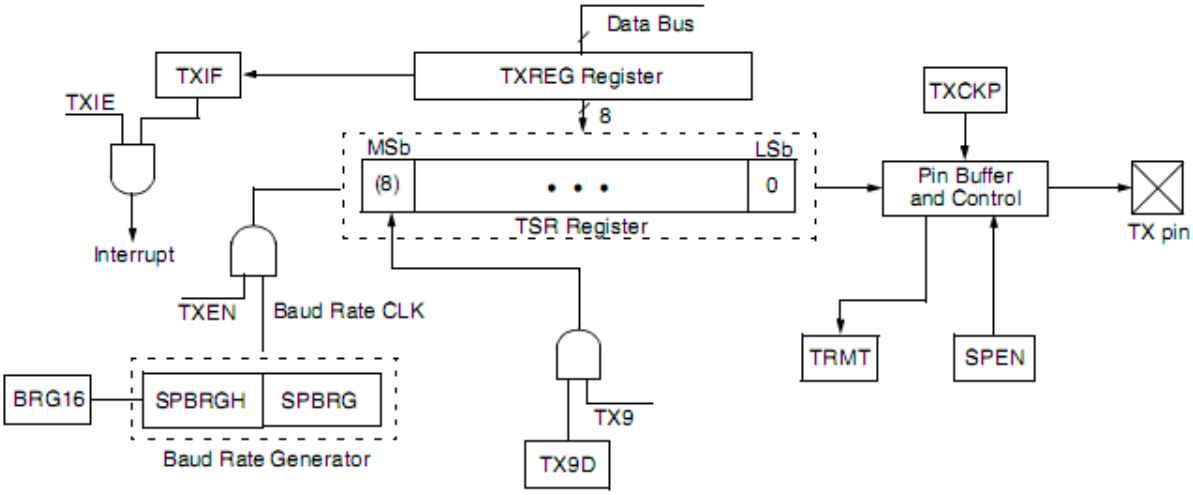
$f_{PWM} = 1/ T_{PWM} \approx 2{,}93$ (kHz).

- **Hoạt động của khối giao tiếp EUSART:**

Khối giao tiếp nối tiếp EUSART (Enhanced Universal Synchronous Asynchronous Receiver Transmitter) cho phép cấu hình hoạt động ở chế độ giao tiếp nối tiếp đồng bộ và không đồng bộ. Trong đề tài này, chế độ giao tiếp không đồng bộ được sử dụng. Do đó ở đây ta sẽ tập trung mô tả hoạt động của module EUSART ở chế độ không đồng bộ.

Hoạt động truyền:

Sơ đồ khối bộ truyền được thể hiện trong hình dưới:



Hình 5.12: Sơ đồ khối bộ truyền của module EUSART

Bộ phận chính của khối truyền là thanh ghi truyền TSR. Thanh ghi này không thể truy cập bằng phần mềm mà phải truy cập gián tiếp qua thanh ghi đệm truyền TXREG. Quá trình truyền được khởi tạo bằng cách ghi dữ liệu truyền vào thanh ghi TXREG. Nếu đây là dữ liệu truyền lần đầu tiên hoặc dữ liệu truyền trước đã truyền hoàn tất thì dữ liệu trong TXREG ngay lập tức sẽ được truyền vào thanh ghi TSR. Nếu thanh ghi TSR vẫn còn chứa dữ liệu của ký tự truyền trước thì dữ liệu mới trong TXREG sẽ được giữ cho đến khi bit Stop của ký tự đang truyền hoàn tất. Sau đó, dữ liệu chờ trong TXREG sẽ được truyền vào TSR.

Hoạt động nhận:

Sơ đồ khối bộ nhận được thể hiện trong hình dưới:

(phần kích từ – thường đặt trên stato) tạo ra từ trường đi trong mạch từ, xuyên qua các vòng dây quấn của phần ứng (thường đặt trên roto). Khi có dòng điện chạy trong mạch phần ứng, các thanh dẫn phần ứng sẽ chịu tác động bởi các lực điện từ theo phương tiếp tuyến với mặt trụ roto, làm cho roto quay.

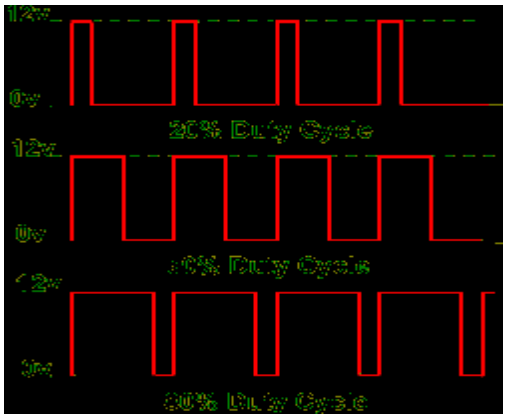
Tùy theo cách mắc cuộn dây roto và stato mà người ta có các loại động cơ sau:

Động cơ kích từ độc lập: Cuộn dây kích từ (cuộn dây stato) và cuộn dây phần ứng (roto) mắc riêng rẽ nhau, có thể cấp nguồn riêng biệt.

Động cơ kích từ nối tiếp: Cuộn dây kích từ mắc nối tiếp với cuộn dây phần ứng.

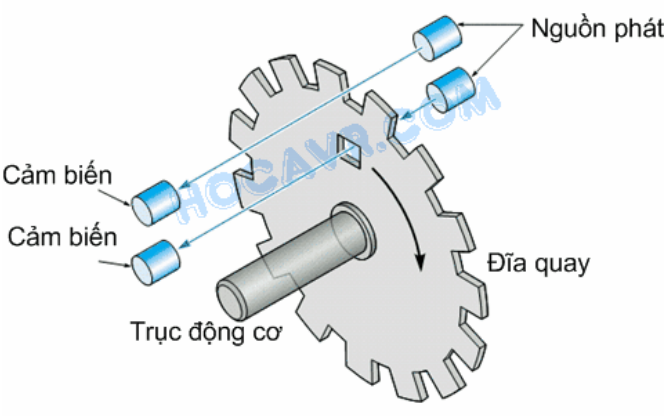
Đối với loại động cơ kích từ độc lập, người ta có thể thay thế cuộn dây kích từ bởi nam châm vỉnh cữu, khi đó ta có loại động cơ điện một chiều dùng nam châm vĩnh cửu.

Đối với loại động cơ kích từ độc lập dùng nam châm vĩnh cửu, để thay đổi tốc độ, ta thay đổi điện áp cung cấp cho roto. Việc cấp áp một chiều thay đổi thường khó khăn, do vậy người ta dùng phương pháp điều xung (PWM):



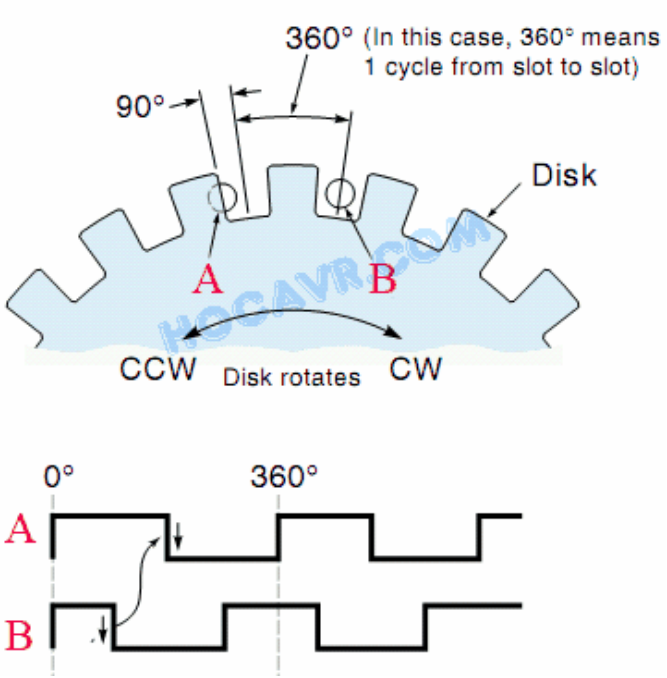
Hình 6.2: Điều chỉnh độ rộng xung PWM

Mạch điều khiển động cơ bằng phương pháp PWM hoạt động dựa theo nguyên tắc cấp nguồn cho động cơ bằng chuỗi xung đóng mở với tốc độ nhanh. Nguồn DC được chuyển đổi thành tín hiệu xung vuông (chỉ gồm hai mức 0 volt và xấp xỉ điện áp hoạt động). Tín hiệu xung vuông này được cấp cho động cơ. Nếu tần số chuyển mạch đủ lớn động cơ sẽ chạy với một tốc độ đều đặn phụ thuộc vào momen của trục quay.



Hình 6.4: Optical Encoder

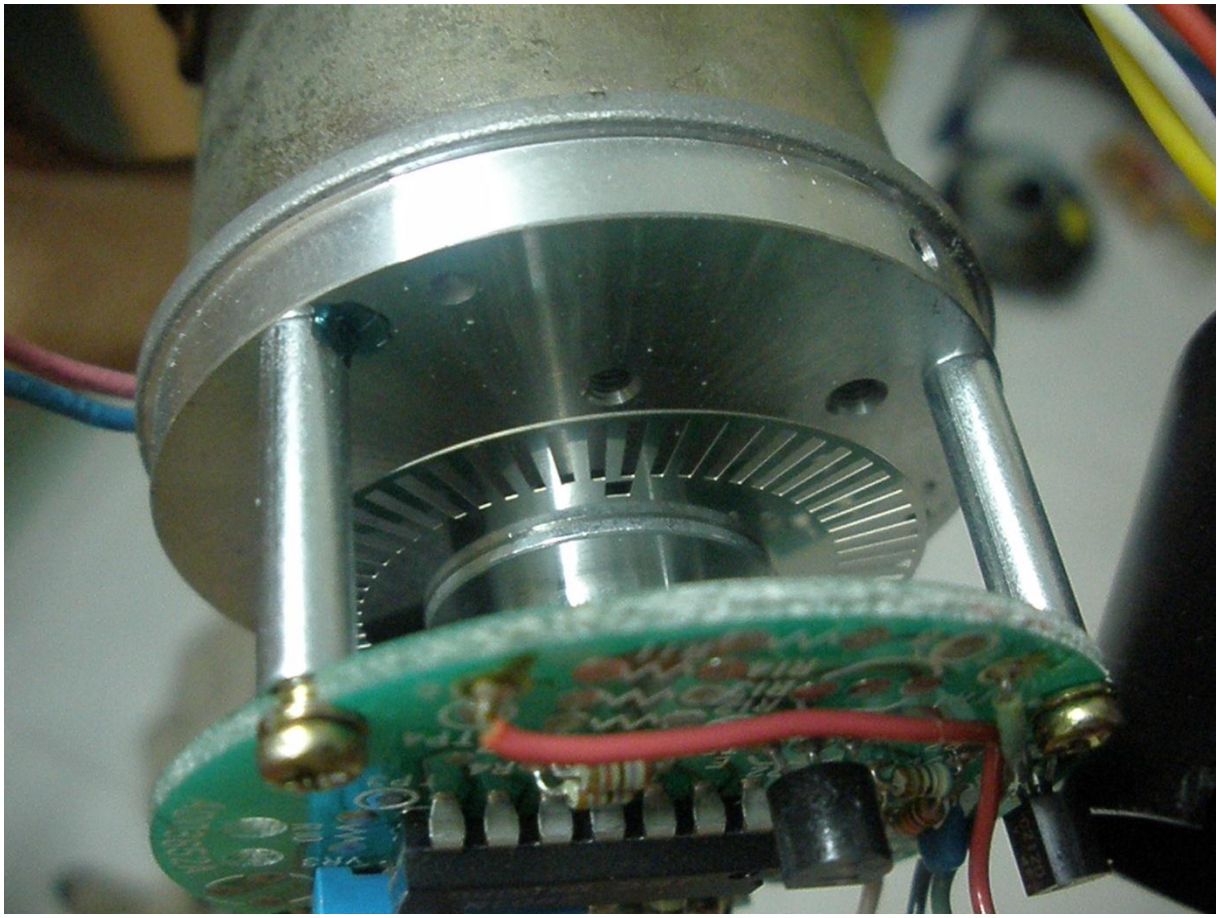
Encoder thường có ba kênh (ba ngõ ra) bao gồm kênh A, kênh B và kênh Z (Index). Trên hình 6.4, ta thấy một lỗ nhỏ trên đĩa quay và một cặp phát – thu dành riêng cho lỗ nhỏ này. Đó là kênh Z của encoder. Cứ mỗi lần động cơ quay được một vòng, hồng ngoại từ nguồn phát sẽ xuyên qua lỗ nhỏ đến cảm biến quang và cảm biến lại thu được một tín hiệu. Như thế cứ mỗi vòng quay của động cơ, lại xuất hiện một xung ở kênh Z. Bên ngoài đĩa quay được chia thành các rãnh nhỏ và có một cặp thu – phát khác dành cho các rãnh này. Đây chính là kênh A của encoder, hoạt động của kênh A cũng tương tự kênh Z, điểm khác nhau là trong một vòng quay của động cơ, sẽ có N xung xuất hiện trên kênh A. N là số rãnh trên đĩa và còn được gọi là độ phân giải (resolution) của encoder. Mỗi loại encoder có một độ phân giải khác nhau, có khi trên mỗi đĩa chỉ có vài rãnh nhưng cũng có trường hợp có đến hàng nghìn rãnh được chia. Để điều khiển động cơ, ta phải biết được độ phân giải của encoder đang dùng. Độ phân giải ảnh hưởng đến độ chính xác điều khiển và cả phương pháp điều khiển. Ngoài ra, trên encoder còn có một cặp thu phát khác được đặt trên cùng đường tròn với kênh A nhưng lệch một chút (lệch M+0.5 rãnh), đây là kênh B của encoder. Tín hiệu xung từ kênh B có cùng tần số với kênh A nhưng lệch pha 90^o. Bằng cách phối hợp kênh A và kênh B ta có thể biết chiều quay của động cơ.



Hình 6.5: Hai kênh A và B lệch pha trong encoder

Khi cảm biến A bắt đầu bị che thì cảm biến B vẫn nhận được hồng ngoại xuyên qua và ngược lại. Hình trên là dạng xung ngõ ra trên 2 kênh. Xét trường hợp động cơ quay cùng chiều kim đồng hồ, tín hiệu di chuyển từ trái sang phải. Lúc tín hiệu kênh A chuyển từ mức cao xuống thấp (cạnh xuống) thì kênh B đang ở mức thấp. Ngược lại, nếu động cơ quay ngược chiều kim đồng hồ, tín hiệu di chuyển từ phải qua trái. Lúc này, tại cạnh xuống của kênh A thì kênh B đang ở mức cao. Như vậy, bằng cách phối hợp hai kênh A và B, ta không những xác định được góc quay (thông qua số xung) mà còn biết được chiều quay của động cơ (thông qua mức của kênh B ở cạnh xuống của kênh A).

Encoder được sử dụng trong đề tài là loại 60 xung trên một vòng quay với đủ ba kênh A, B và Z, tuy nhiên chỉ cần sử dụng 2 kênh A, B là đủ cho ứng dụng điều khiển vị trí.



Hình 6.6: Encoder đi kèm động cơ Servo DC

Cách đọc encoder bằng PIC:

Ta sử dụng ngắt ngoài của PIC 18F4550 để đọc encoder về, đây là phương pháp đơn giản nhưng chính xác. Ý tưởng của phương pháp này rất đơn giản, ta nối kênh A của encoder của động cơ bên phải với ngắt ngoài (chân RB1) và kênh B với một chân nào đó bất kỳ (ở đây ta dùng chân RD1). Cứ mỗi lần ngắt ngoài xảy ra, tức có 1 xung xuất hiện ở kênh A thì trình phục vụ ngắt ngoài tự động được gọi. Trong trình phục vụ ngắt này ta kiểm tra mức của kênh B, tùy theo mức của kênh B ta sẽ tăng biến đếm xung lên 1 hoặc giảm đi 1. Làm tương tự với encoder của động cơ bên trái, nối kênh A của encoder với ngắt ngoài (chân RB2) và kênh B với một chân nào đó bất kỳ (chân RD2).

- Thành phần vi phân (K_D) làm tăng độ ổn định của hệ thống, giảm độ vọt lố và cải thiện tốc độ đáp ứng của hệ.

Ảnh hưởng của các thành phần K_P, K_I, K_D đối với hệ kín được tóm tắt trong bảng sau:

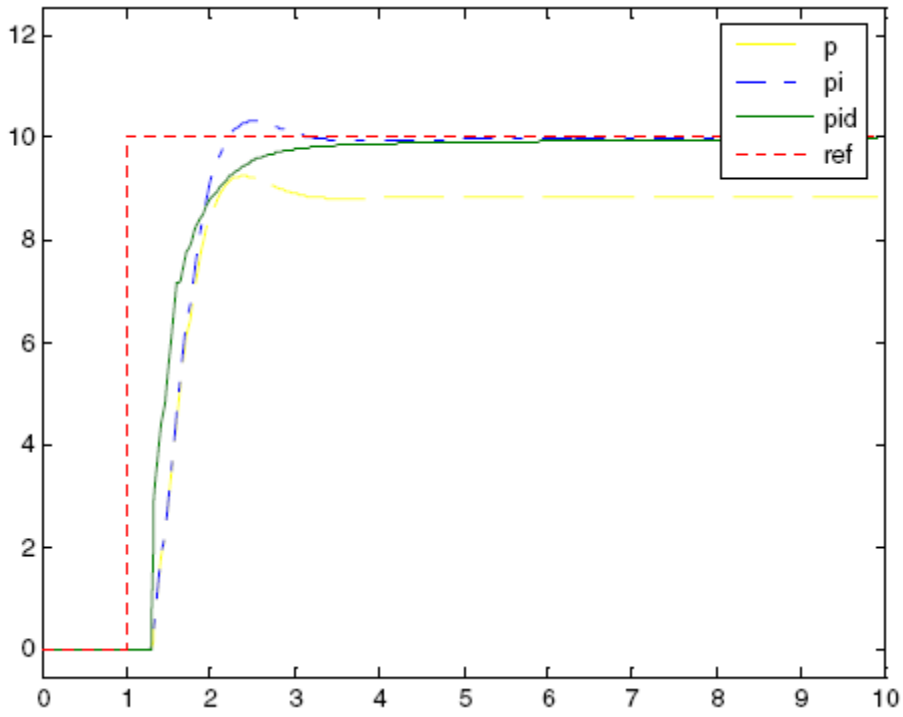
Đáp ứng của hệ thống	Thời gian tăng	Vọt lố	Thời gian ổn định	Sai lệch so với trạng thái bền
K_P	Giảm	Tăng	Ít thay đổi	Giảm
K_I	Giảm	Tăng	Tăng	Triệt tiêu
K_D	Ít thay đổi	Tăng	Tăng	Ít thay đổi

Bảng 6.1: Ảnh hưởng của các thành phần Kp, Ki, Kd đối với hệ kín

Hệ PID có thể sử dụng ở chế độ P, PI, PD tùy vào đặc tính của hệ thống.

- *Hiệu chỉnh tỉ lệ P:* K_P càng lớn thì sai số xác lập càng nhỏ, vọt lố cao, tuy nhiên ta không thể tăng K_P đến vô cùng khi e_{xl} không bằng không, $K_P \leq K_{gh}$.
- *Hiệu chỉnh tỉ lệ PD:* Hiệu chỉnh PD làm nhanh đáp ứng của hệ thống, giảm thời gian quá độ nhưng lại nhạy với nhiễu tần số cao.
- *Hiệu chỉnh tỉ lệ PI:* Hiệu chỉnh tỉ lệ PI làm chậm đáp ứng quá độ, tăng độ vọt lố, giảm sai số xác lập.
- *Hiệu chỉnh tỉ lệ PID:* Cải thiện đáp ứng quá độ (giảm vọt lố, giảm thời gian quá độ), giảm sai số xác lập.

Kết quả của bộ điều khiển thường được đánh giá thông qua việc đo đạc đáp ứng của bộ điều khiển trong một hệ thống nhất định như hình sau:



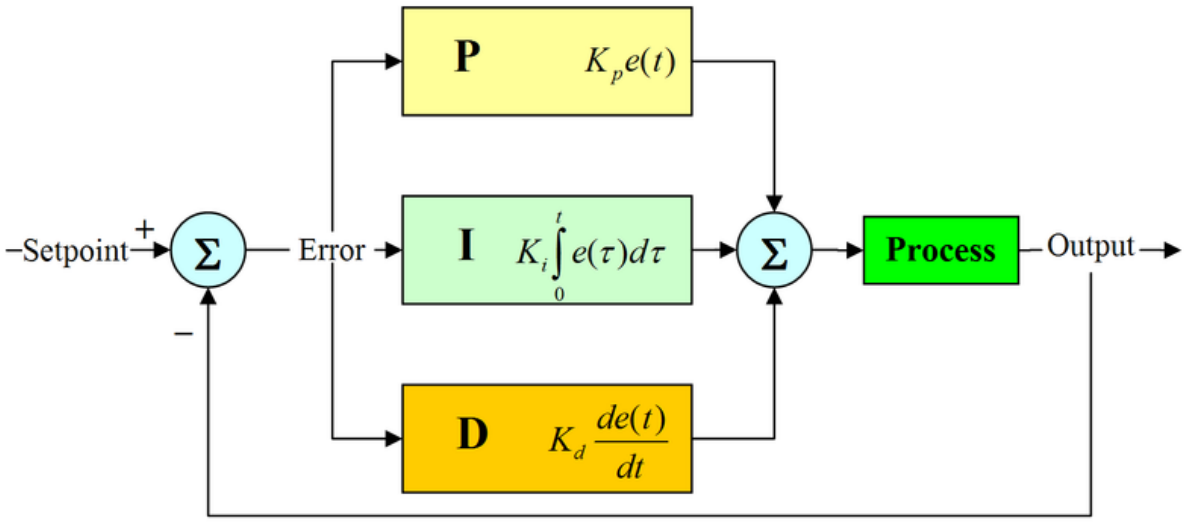
Hình 6.8 Đáp ứng của các hệ thống điều khiển

Hình 6.8 mô tả đáp ứng của hệ thống điều khiển theo sự thay đổi và đáp ứng của ba phương pháp điều khiển P, PI, PID. Trong đó đáp ứng của bộ PID là tốt nhất với độ vọt lố ít và thời gian ổn định nhanh.

Bộ PID có thể làm việc trên dữ liệu liên tục hoặc dạng rời rạc. Hiện nay, PID rời rạc được ứng dụng rộng rãi trong các hệ điều khiển do hầu hết các hệ thống hiện nay đều là hệ thống số. Công thức bộ PID có thể được mô tả như sau:

$$u(n) = K_p e(n) + K_i \sum_{k=0}^{n} e(k) + K_d (y(n) - y(n-1))$$

Quá trình tính toán của bộ PID:



Hình 6.9: Quá trình tính toán PID

Bộ điều khiển PID có hàm truyền dạng liên tục như sau:

$$G(s) = K_p + \frac{K_I}{s} + K_D \times s$$

Biến đổi Z của nó như sau:

$$G(z) = K_P + \frac{K_I \times T}{2}\left(\frac{z+1}{z-1}\right) + \frac{K_D}{T}\left(\frac{z-1}{z}\right)$$

Viết lại G(z) như sau:

$$G(z) = \frac{(K_P + K_I \times \frac{T}{2} + \frac{K_D}{T}) + (-K_P + K_I \times \frac{T}{2} - 2 \times \frac{K_D}{T}) \times z^{-1} + \frac{K_D}{T} \times z^{-2}}{1 - z^{-1}}$$

Đặt:

$$a_0 = K_P + K_I \times \frac{T}{2} + \frac{K_D}{T}$$

$$a_1 = -K_P + K_I \times \frac{T}{2} - 2 \times \frac{K_D}{T}$$

$$a_2 = \frac{K_D}{T}$$

Suy ra:

$$G(z) = \frac{a_0 + a_1 \times z^{-1} + a_2 \times z^{-2}}{1 - z^{-1}}$$

Từ đó, ta tính được tín hiệu điều khiển **u(k)** khi tín hiệu vào **e(k)** như sau:

$$u(k) = G(z) \times e(k) = \frac{a_0 + a_1 \times z^{-1} + a_2 \times z^{-2}}{1 - z^{-1}} \times e(k)$$

Suy ra:

$$u(k) = u(k-1) + a_0 \times e(k) + a_1 \times e(k-1) + a_2 \times e(k-2)$$

Trong đó: **T** là thời gian lấy mẫu

e(k) là sai số hiện tại: **e(k) = giá trị đặt – giá ngõ ra**

e(k-1) và **e(k-2)** là các sai số tích lũy

u(k) là ngõ ra bộ điều khiển

K_P, K_I, K_D là các thông số P, I, D của bộ điều khiển

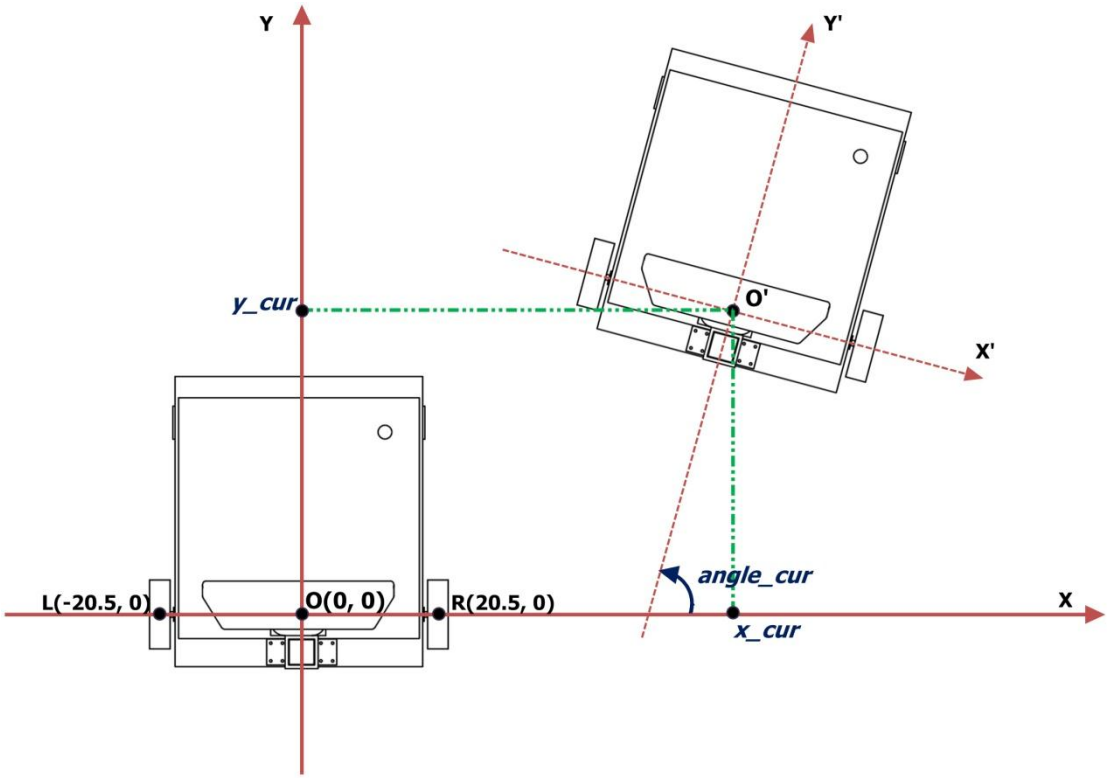
Cách điều chỉnh các hệ số K_P, K_I, K_D:

Đầu tiên, ta chỉnh cho $K_P = 1$ và K_D, $K_I = 0$ nghĩa là chỉ điều khiển P. Sau đó, tăng K_P lên dần dần và quan sát động cơ. Khi thấy động cơ dao động (nghĩa là nó lúc nhanh lúc chậm), ta lấy K_P tại đó nhân với 0.6 để tính toán: $K_{P\_t} = 0.6 \times K_{P\_dao\_động}$. Sau đó, ta tăng K_D lên dần dần (giữ nguyên K_P), nếu K_D quá lớn sẽ làm cho động cơ bị dao động mạnh. Giảm K_D lại cho đến khi động cơ hết dao động. Điều chỉnh K_I là khó nhất, bắt đầu từ giá trị rất nhỏ (ví dụ lấy $K_I = 1/K_D$ chẳng hạn). Hệ số K_I không cần lớn vì động cơ tự nó đã chứa thành phần K_I, thể hiện ở moment quán tín hay sức ì của động cơ. Do đó, thường với đối tượng điều khiển là nhiệt độ hay động cơ (các đối tượng có quán tính) thì chỉ cần bộ điều khiển PD là đủ.

truyền động, ta đặt gốc tọa độ robot tại vị trí chính giữa hai tâm các bánh này (gọi là tâm robot).

Hình 7.3: Mô hình robot

Quan sát hình chiếu bằng của mô hình robot ở hình 7.4, tọa độ gốc của robot được chọn là tâm O(0, 0). Vị trí tương ứng của tâm hai bánh xe trái và phải là R(20.5, 0), L(-20.5, 0) (đơn vị là centimet). Trong quá trình di chuyển, tọa độ của robot sẽ được cập nhật so với vị trí lúc khởi động khi robot đến một vị trí mới. Các giá trị cập nhật bao gồm: tọa độ x hiện tại (**x_cur**), tọa độ y hiện tại (**y_cur**) và góc hiện tại (trục O'Y' so với trục OX, **angle_cur**). Như vậy, tại thời điểm bắt đầu khởi động: **x_cur = 0, y_cur = 0** và **angle_cur = 90^o.**



Hình 7.4: Tọa độ robot (quan sát từ trên xuống)

Khi di chuyển robot sẽ có hai động tác chính: đi thẳng (tiến hoặc lùi) một đoạn **L** và xoay một góc **anpha** quanh tâm robot. Bài toán chỉ đơn thuần áp dụng các phép tịnh tiến và xoay cơ bản trong không gian 2D. Phương trình tính toán tọa độ mới của robot cho từng động tác:

- ***Đi thẳng một đoạn L:***

$$x\_new = x\_cur \pm L.\cos(angle\_cur)$$
$$y\_new = y\_cur \pm L.\sin(angle\_cur)$$
$$angle\_new = angle\_cur$$

Dấu "+" cho trường hợp tiến và "-" cho trường hợp lùi.

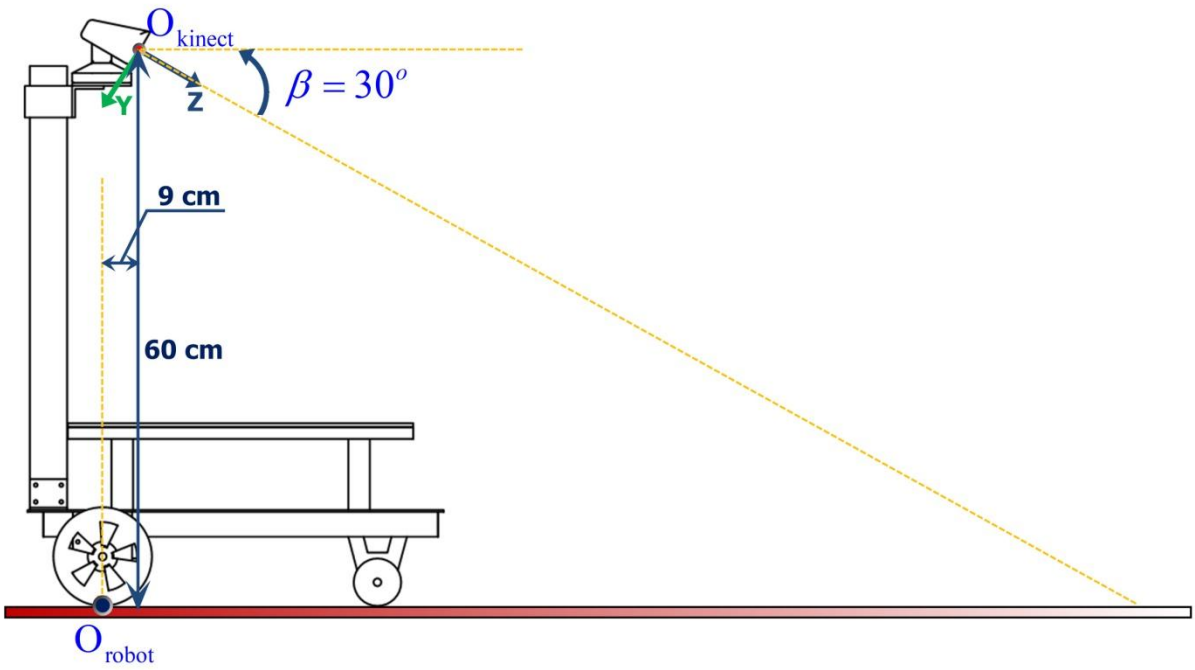
- ***Xoay một góc anpha:***

$$x\_new = x\_cur$$
$$y\_new = y\_cur$$
$$angle\_new = angle\_cur + anpha$$

cản của robot. Chính vì lý do đó mà ta bố trí Kinect được đặt trên cao và hướng xuống với một góc nghiêng nhất định, đồng thời được đặt lùi phía sau robot để nhìn được một cách tổng quan nhất môi trường phía trước robot.

Chính điều này đặt ra bài toán chuyển dời hệ trục tọa độ của Kinect về hệ trục tọa độ của robot để đồng bộ hóa. Có chút khác biệt giữa hai hệ trục: đối với hệ trục tọa độ Kinect, phương Z đại diện cho khoảng cách tới vật thể, phương Y đại diện cho chiều cao vật thể lần lượt tương ứng với phương Y và Z của hệ trục tọa độ robot. Phương X còn lại không có gì thay đổi.

Mục đích của ta là đưa mặt phẳng OXZ của Kinect về trùng với mặt phẳng OXY của robot. Quan sát hình 7.6, ta thấy ngoài việc nghiêng một góc 30^o, Kinect còn lệch theo chiều Z một đoạn 9 cm và chiều Y một đoạn 60 cm.



Hình 7.6: Đồng nhất hệ trục tọa độ Kinect và robot

Ta lần lượt thực hiện các công việc sau đối với hệ trục Kinect:

- Xoay mặt phẳng OYZ quanh trục X ngược chiều kim đồng hồ một góc 30^o.

$$\begin{bmatrix} X_1 \\ Y_1 \\ Z_1 \end{bmatrix} = \begin{bmatrix} X & Y & Z \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\beta & \sin\beta \\ 0 & -\sin\beta & \cos\beta \end{bmatrix} = \begin{bmatrix} X \\ Y\cos\beta - Z\sin\beta \\ Y\sin\beta + Z\cos\beta \end{bmatrix}$$

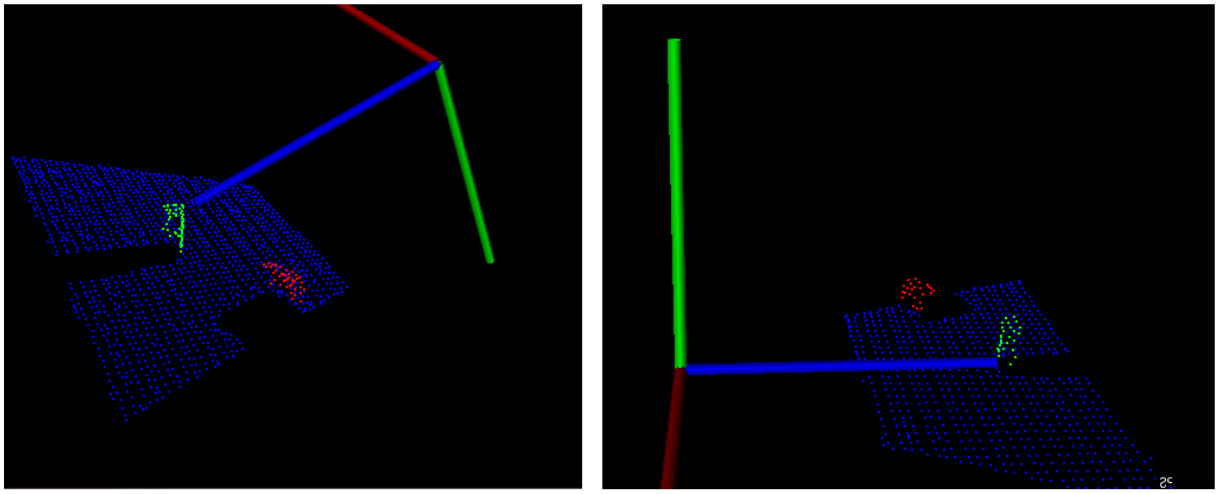
- Tịnh tiến trục Y xuống một đoạn 60 cm và tịnh tiến trục Z về sau một đoạn 9 cm.

$$\begin{bmatrix} X_2 \\ Y_2 \\ Z_2 \end{bmatrix} = \begin{bmatrix} X_1 \\ Y_1 \\ Z_1 \end{bmatrix} + \begin{bmatrix} 0 \\ -60 \\ 9 \end{bmatrix} = \begin{bmatrix} X \\ Y\cos\beta - Z\sin\beta - 60 \\ Y\sin\beta + Z\cos\beta + 9 \end{bmatrix}$$

- Đảo chiều Y.

$$\begin{bmatrix} X_3 \\ Y_3 \\ Z_3 \end{bmatrix} = \begin{bmatrix} X_2 \\ -Y_2 \\ Z_2 \end{bmatrix} = \begin{bmatrix} X \\ -Y\cos\beta + Z\sin\beta + 60 \\ Y\sin\beta + Z\cos\beta + 9 \end{bmatrix}$$

Hình 7.7 cho ta thấy hệ trục Kinect sau khi thực hiện các phép chuyển đổi. Hệ trục có các chiều X, Y, Z tương ứng với các màu đỏ, xanh lá, xanh dương.



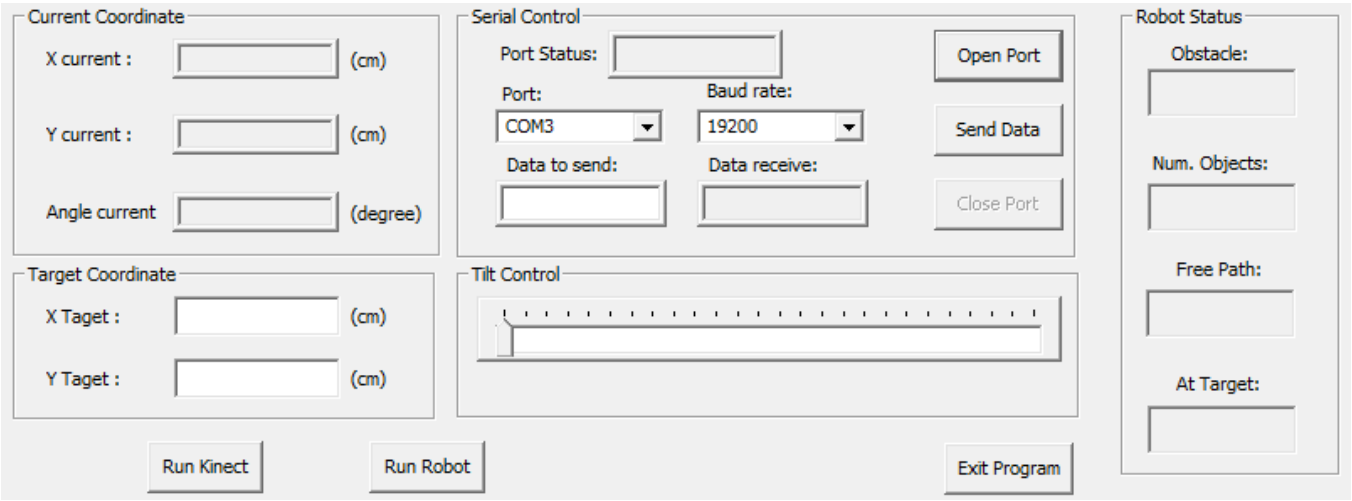
Hình 7.7: Tọa độ Kinect trước (trái) và sau (phải) khi chuyển trục

Hình 8.2: Xử lý đa tiến trình

Luồng Kinect (số 1) làm nhiệm vụ phân tích môi trường, đưa ra các thông tin về kích thước, vị trí vật cản nếu xuất hiện trước robot, đã được trình bày ở mục 4.3.

Luồng giao tiếp với vi điều khiển (số 3) theo chuẩn RS232 làm nhiệm vụ truyền lệnh xuống và nhận thông tin vị trí lên từ vi điều khiển.

Luồng xử lý kế hoạch di chuyển của robot sử dụng thông tin từ luồng số 1 và luồng số 3, điều khiển robot di chuyển hợp lý về đích.



Hình 8.3: Giao diện chương trình điều khiển

Hình 8.3 là giao diện điều khiển viết bằng MFC trong bộ Visual Studio 2010 của Microsoft. Vị trí đích cần di chuyển đến sẽ được nhập vào khối **Target Coordinate**; tọa độ hiện tại của robot sẽ tự động hiện thị trong khối **Current Coordinate**; tùy chọn

các thông số cho giao tiếp RS232, trạng thái cổng giao tiếp được đặt trong khối **Serial Control;** trong khối **Robot Status** hiển thị các thông tin về *cờ phát hiện vật cản*, *cờ phát hiện đường trống*, số lượng vật cản và trạng thái robot đã ở vị trí đích hay chưa. **Tilt Control** điều khiển góc ngẩng Kinect.

- ***Định dạng điều khiển qua giao tiếp RS232***

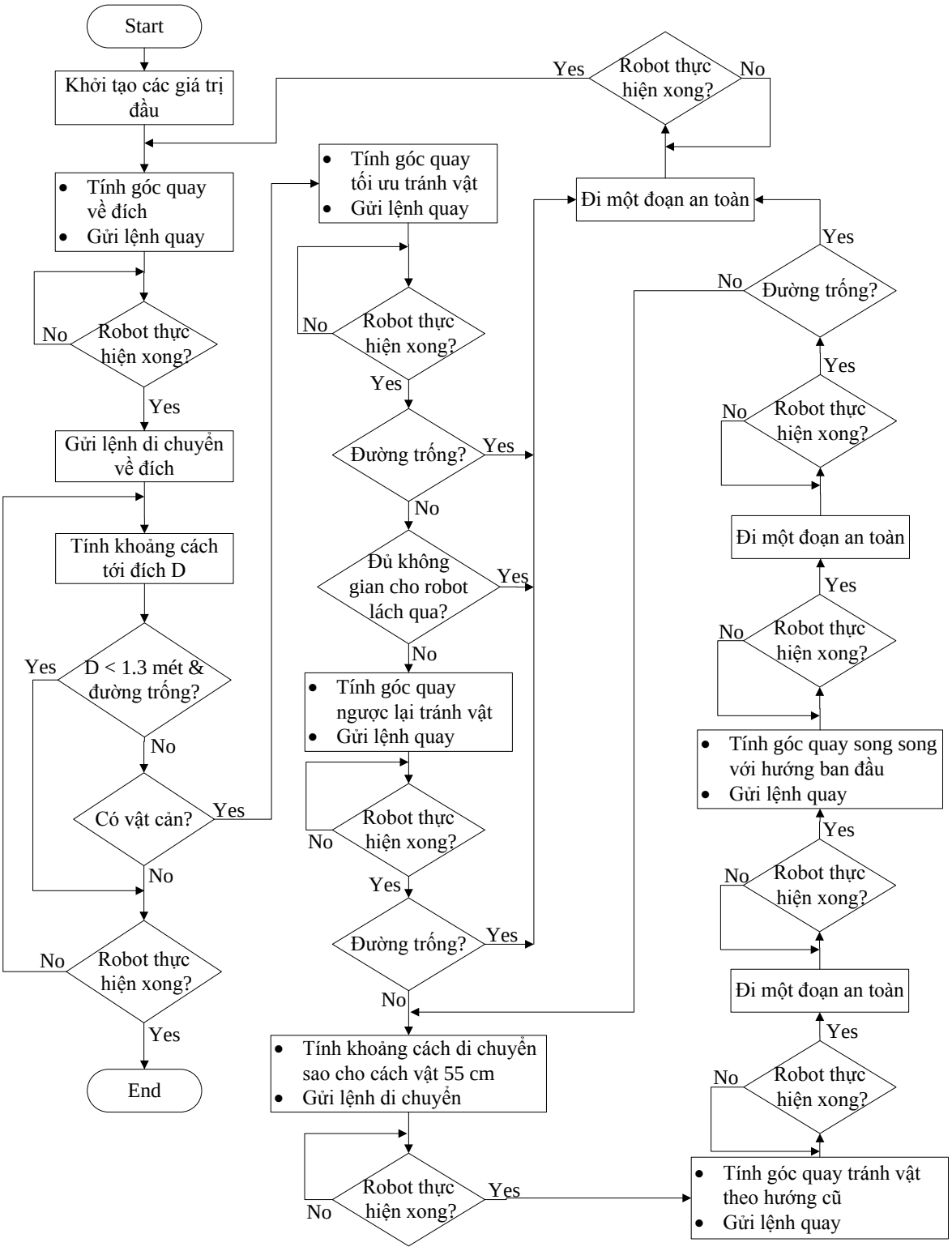
Lệnh điều khiển truyền xuống qua giao tiếp nối tiếp được định dạng theo mẫu sau: ****[kí tự điều khiển][giá trị điều khiển]#,*** trong đó:

- ***kí tự điều khiển:*** R(quay phải), L(quay trái), F(đi thẳng), T(đi thẳng về đích).
- ***giá trị điều khiển:*** là giá trị góc quay hay quãng đường di chuyển, định dạng kiểu dữ liệu double, lấy ba chữ số thập phân, đơn vị là centimet.
- ***'*', '#':*** là các ký tự đặc biệt cho biết ký tự bắt đầu và kết thúc chuỗi.

Lệnh nhận lên từ vi điều khiển có dạng: ***X[giá trị 1]Y[giá trị 2]A[giá trị 3]N,*** trong đó:

- ***X, Y, A:*** lần lượt là ký tự bắt đầu cho các giá trị đại diện tọa độ X, Y, góc hiện tại của robot. ***N*** là ký tự kết thúc chuỗi.
- ***giá trị 1÷3:*** tương ứng là giá trị tọa độ, góc hiện tại của robot.

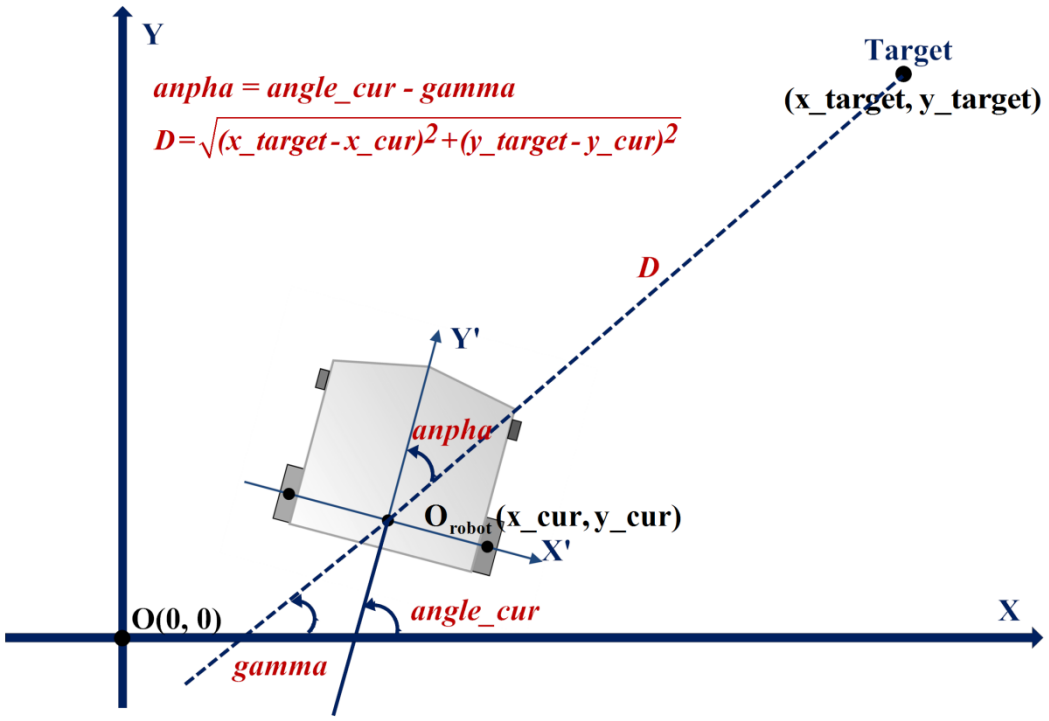
- ***Giải thuật điều khiển robot ứng xử với vật cản trên đường đi đến đích:***



Hình 8.4: Sơ đồ giải thuật điều khiển robot do máy tính xử lý

Một số hàm chú ý:

- ***Tính góc quay về đích:***



Hình 8.5: Tính góc quay về đích

Hình 8.5 mô tả một trường hợp đích nằm bên phải robot, lúc này robot sẽ quay một góc **anpha** về bên phải để hướng về đích. Ta có cách tính cho các trường hợp như sau (ta giả sử y_cur luôn dương, robot luôn di chuyển về phía trước):

$$\text{gamma} = \arctan\left(\frac{\text{y\_target} - \text{y\_cur}}{\text{x\_target} - \text{x\_cur}}\right)$$

- ✓ ***gamma < 0 (đích nằm bên trái robot):***
  - ***0 ≤ angle_cur < PI + gamma:***

    + anpha = PI – angle_cur + gamma

    + Quay về bên TRÁI robot
  - ***PI + gamma ≤ angle_cur < 2*PI + gamma:***

    + anpha = angle_cur – (PI + gamma)

+ Quay về bên PHẢI robot

- ***2*PI + gamma ≤ angle_cur ≤ 2*PI:***

  + anpha = 3*PI - angle_cur + gamma

  + Quay về bên TRÁI robot

✓ ***gamma ≥ 0 (đích nằm bên phải robot):***

- ***0 ≤ angle_cur < gamma:***

  + anpha = gamma - angle_cur

  + Quay về bên TRÁI robot

- ***gamma ≤ angle_cur < PI + gamma:***

  + anpha = angle_cur - gamma

  + Quay về bên PHẢI robot

- ***PI + gamma ≤ angle_cur ≤ 2* PI:***

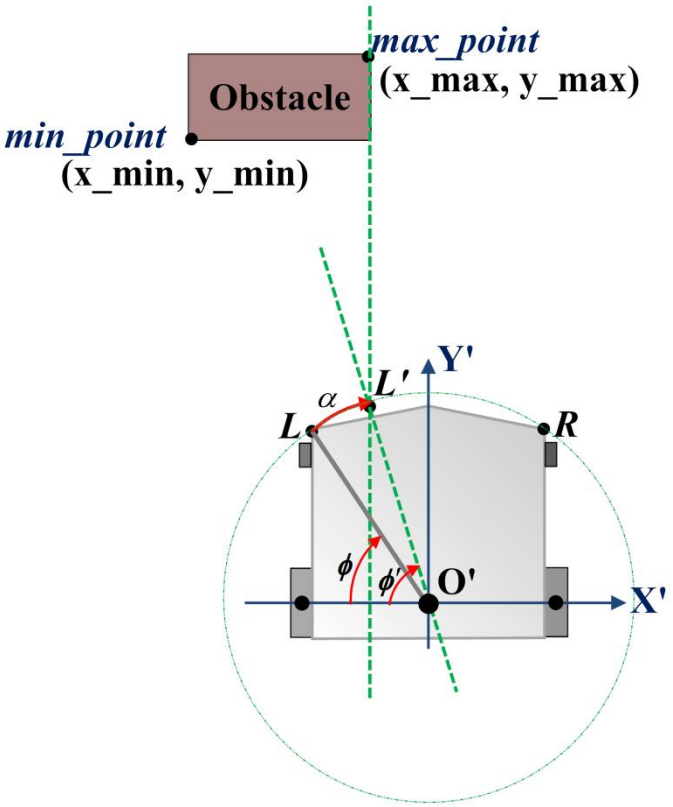
  + anpha = 2*PI - angle_cur + gamma

  + Quay về bên TRÁI robot

▪ ***Tính góc quay tối ưu tránh vật:***

Mục đích là điều khiển robot theo hướng có góc quay nhỏ hơn khi tránh vật cản. Các trường hợp có thể xảy ra là:

- Vật cản nằm bên trái robot

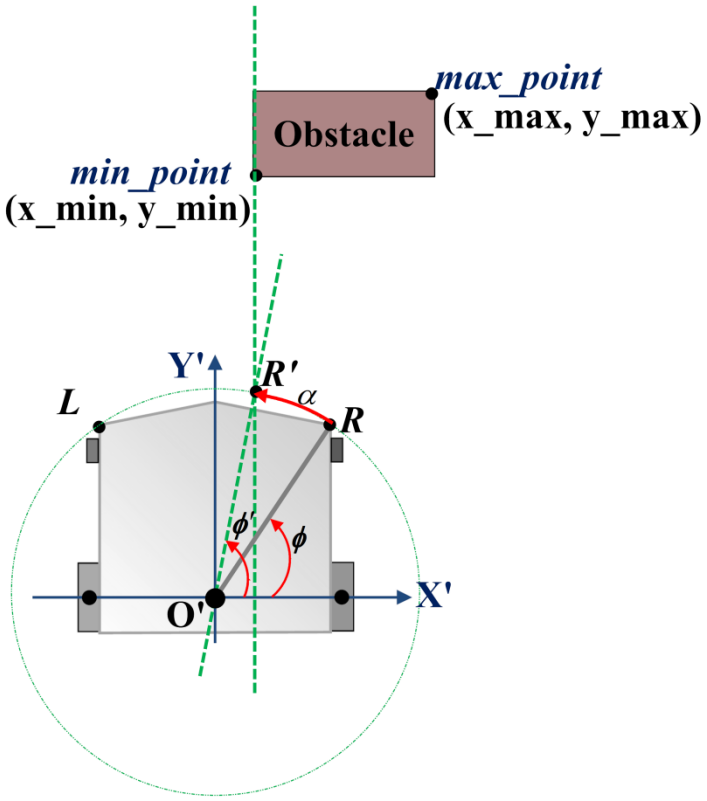


Hình 8.6: Vật cản bên trái robot

Quan sát hình 8.6, góc quay cần thiết để thoát khỏi vật cản là **anpha**(α) khi robot ở vị trí cách vật cản một đoạn. Góc **anpha** hoàn toán tính được khi ta biết được tọa độ **max_point,** góc ϕ và độ lớn **O'L**.

$\alpha = \phi' - \phi = \cos^{-1}(\frac{-x\_max}{O'L}) - \phi$, hướng quay về phía bên phải robot.

- Vật cản nằm bên phải robot

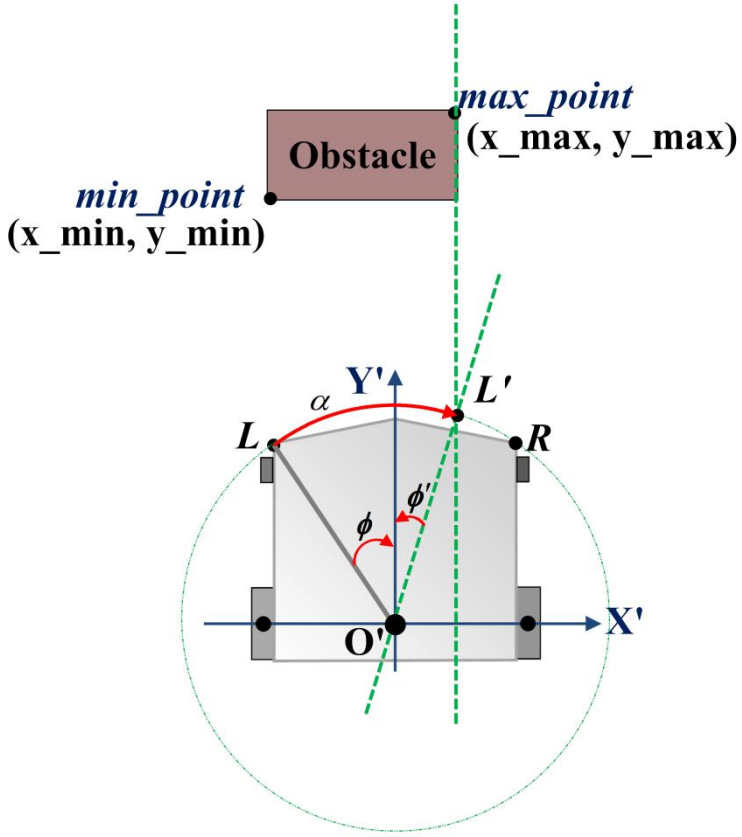


Hình 8.7: Vật cản bên phải robot

Tương tự trường hợp vật cản nằm bên trái, góc **anpha** được tính như sau:

$\alpha = \phi' - \phi = \cos^{-1}(x\_\min / O'L) - \phi$, hướng quay về phía bên trái robot.

- Vật cản nằm ở giữa đường robot



Hình 8.8: Vật cản nằm ở giữa đường di chuyển của robot

Trong trường hợp này, robot sẽ tìm góc quay né vật sao cho **anpha** nhỏ nhất; lúc đó, nếu độ lớn vật cản nhiều hơn về phía bên trái thì robot quay về bên phải (như trên hình 8.8) và ngược lại. Công thức tính trong từng trường hợp: (trong đó góc ϕ và O'L = O'L' biết trước)

Quay trái:

$$\alpha = \phi + \phi' = \phi + \sin^{-1}(\frac{-x\_\min}{O'L})$$

Quay phải:

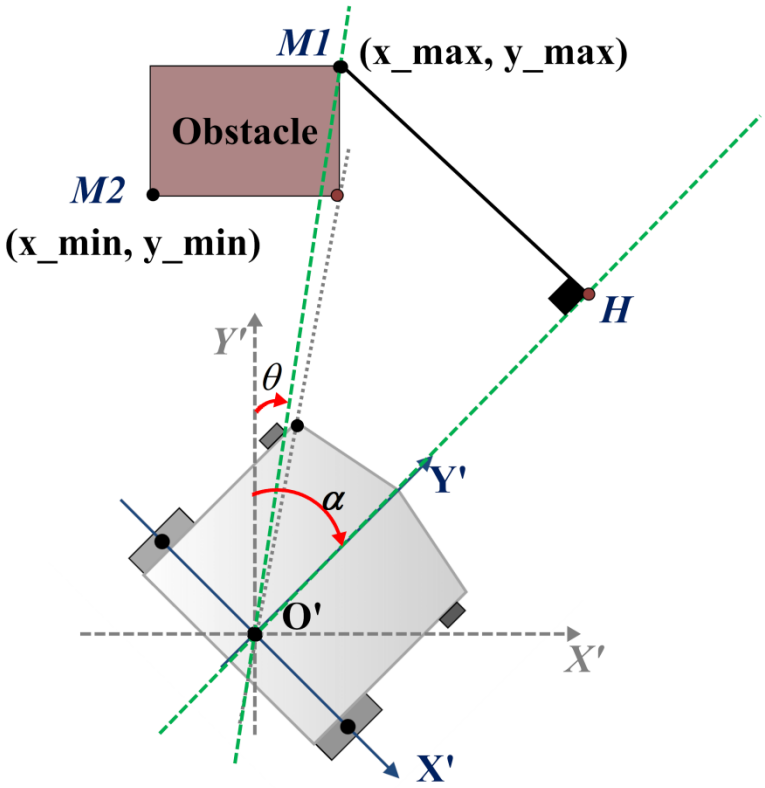
$$\alpha = \phi + \phi' = \phi + \sin^{-1}(\frac{x\_\max}{O'L})$$

- ***Tính góc quay ngược lại tránh vật:***

Như các trường hợp quay một góc xác định né vật cản, nhưng thay vì quay theo hướng có góc quay nhỏ hơn thì robot sẽ quay theo hướng ngược lại. Công thức tính hoàn toàn tương tự.

- ***Đi một đoạn an toàn:***

Sau khi quay một góc tránh vật, robot sẽ di chuyển tiếp một đoạn để thoát hoàn toàn khỏi vật.



Hình 8.9: Đi một đoạn an toàn về phía phải vật cản

Quãng đường di chuyển an toàn để thoát khỏi vật cản bằng đoạn **O'H** cộng thêm một giá trị cố định vừa đủ **deltaMove**(ở đây chọn **deltaMove = 15 cm** dựa trên thực nghiệm).

Đi bên phải vật cản:

$$OH' = OM_1 \times \cos(\alpha - \theta)$$

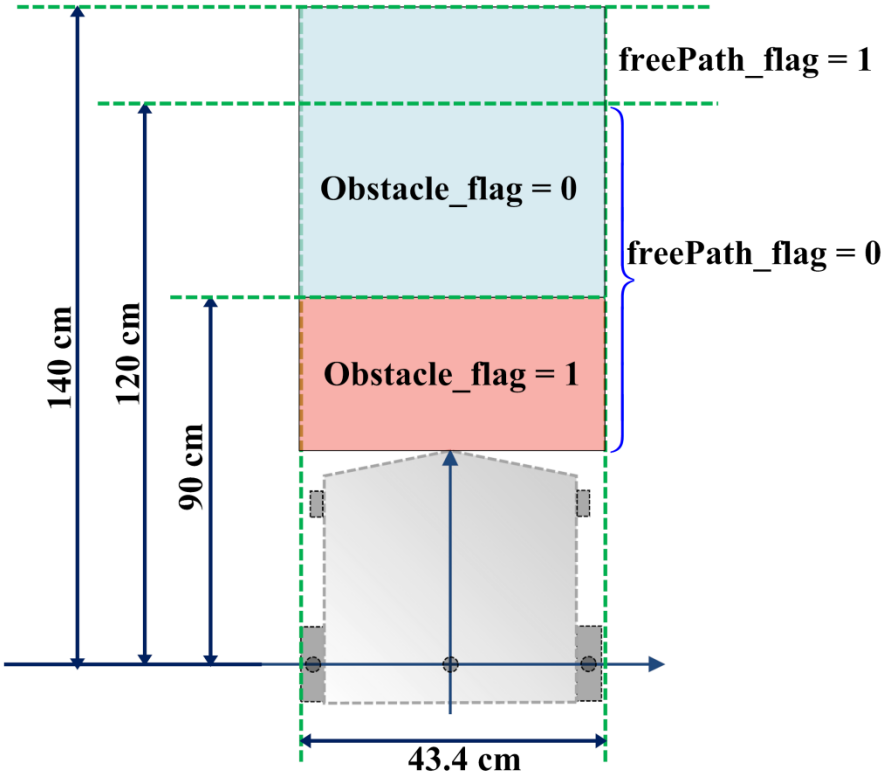
$$= \sqrt{(x\_max - x\_cur)^2 + (y\_max - y\_cur)^2} \times \cos(\alpha - \tan^{-1}(\frac{|x\_max|}{y\_max}))$$

Đi bên trái vật cản:

$$OH' = OM_1 \times \cos(\alpha - \theta)$$

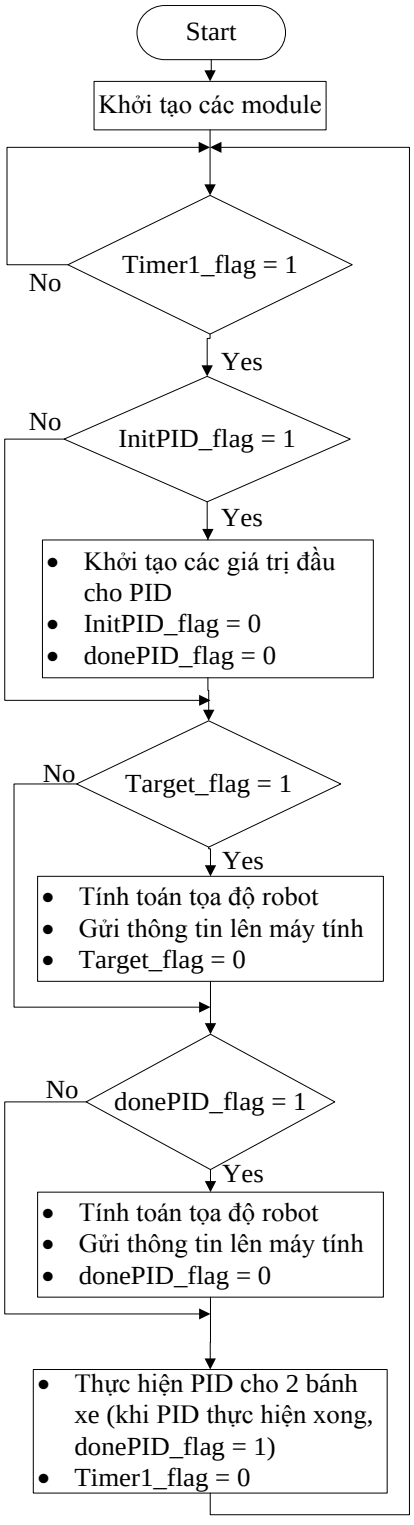
$$= \sqrt{(x\_min - x\_cur)^2 + (y\_max - y\_cur)^2} \times \cos(\alpha - \tan^{-1}(\frac{|x\_min|}{y\_max}))$$

- ***Cờ báo có vật cản (Obstacle_flag) và đường trống (freePath_flag):***



Hình 8.10: Cờ báo có vật cản và đường trống

Hình 8.10 biểu diễn không gian xuất hiện vật cản phía trước robot, lúc đó các cờ tương ứng sẽ bật lên trong các vùng xác định. Tầm nhìn xa của robot giới hạn ở khoảng cách 140 cm.



Hình 8.12: Sơ đồ giải thuật trên vi điều khiển

- ***Khởi tạo các module:***

Khởi tạo các module sử dụng trên PIC: các port I/O, timer, ngắt ngoài, PWM và giao tiếp RS232.

- ***Timer1_flag:***

Cờ báo bằng 1 khi timer1 tràn sau 5 ms, phục vụ cho việc lấy mẫu, tính toán PID.

- ***InitPID_flag:***

Cờ báo khởi tạo PID, bằng 1 khi nhận được lệnh từ máy tính.

- ***Khởi tạo các giá trị đầu cho PID:***

Khởi tạo các giá trị ban đầu cho tính toán PID: các thông số k_p, k_i, k_d; giá trị đặt.

- ***Target_flag:***

Cờ báo lên 1 khi robot đang di chuyển về đích và gặp vật cản.

- ***Tính toán tọa độ robot:***

Tính toán tọa độ robot, đã được trình bày tại mục 7.2.

- ***Gửi thông tin lên máy tính:***

Thông tin về tọa độ và góc hiện tại của robot: (x_cur, y_cur, angle_cur).

- ***Thực hiện PID trên hai bánh xe:***

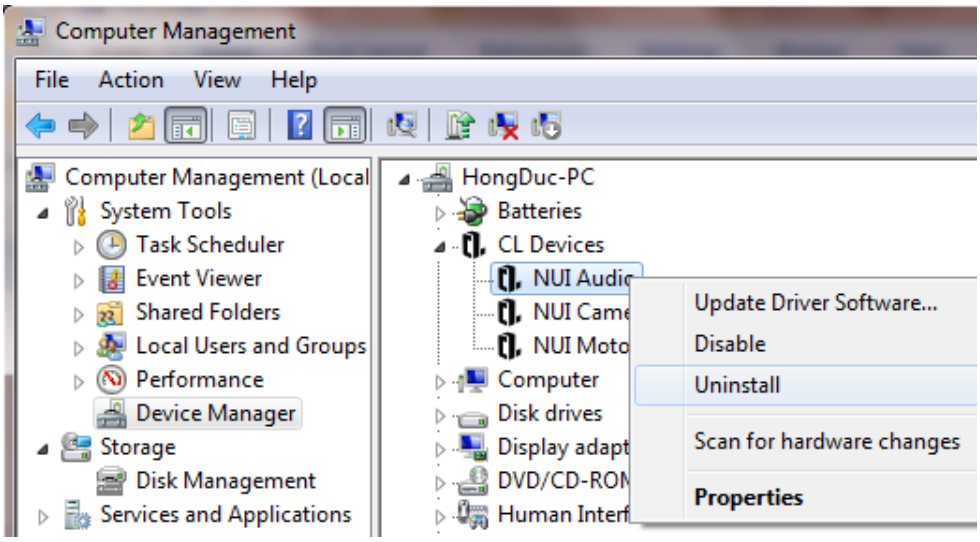
Dựa trên giải thuật PID vị trí đã được trình bày ở mục 6.2. PID được thực hiện xong khi sai số nằm trong khoảng cho phép (xấp xỉ bằng không), lúc đó donePID_flag bằng 1.

Một con robot rất đáng quan tâm là iRobot với các lựa chọn khác nhau, giá dao động từ 130$ ÷300$, chi tiết xem tại: http://store.irobot.com

- Phần thị giác robot chỉ quan sát được không gian phía trước robot nên robot có thể sẽ đụng vật cản bên hông hay phía sau khi di chuyển lùi. Vấn đề này có thể được giải quyết bằng biệc trang bị thêm các cảm biến phát hiện vật cản xung quanh robot; hoặc thiết kế thêm hệ thống xoay cho phần thị giác, thị giác sẽ xoay một góc xác định trước khi quyết định di chuyển.
- Tích hợp thêm các cảm biến định vị cần thiết để tăng sự chính xác.
- Chương trình được viết trên môi trường Windows với sự hỗ trợ từ thư viện Point Cloud còn nhiều hạn chế, nhất là ở tốc độ xử lý. Sẽ tối ưu nhất nếu viết trên môi trường Linux và nếu phát triển thành sản phẩm thương mại sẽ hạ được giá thành phần mềm xuống, tăng tính cạnh tranh.

▪ ***Hướng phát triển:***

- Mục đích của đề tài tạo ra nền tảng cho việc xây dựng một mô hình robot dịch vụ hoàn chỉnh: một robot có khả năng làm các công việc thay cho con người như: bưng bê đồ ăn, lau nhà, hướng dẫn khách hàng, …
- Bên cạnh đó, với sức mạnh của thiết bị Kinect có thể giúp ta xây dựng được bản đồ (mapping) dạng 3D. Và việc tích hợp lên robot sẽ giúp ta xây dựng được bản đồ ở những khu vực mà con người không thể vào được.

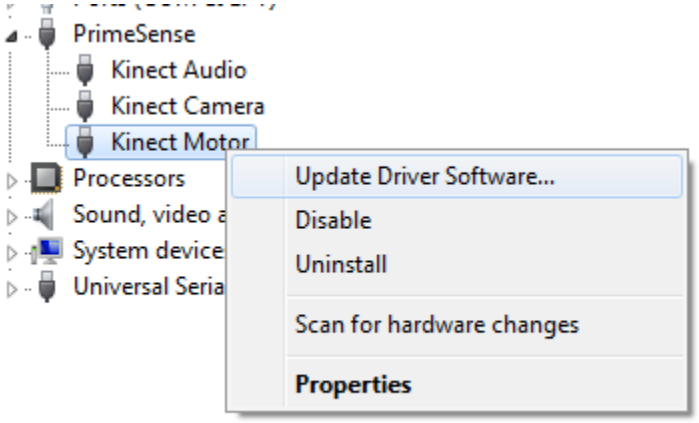


- ***Bước 3:***

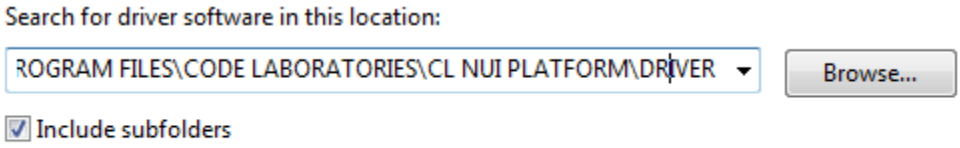
  Cài đặt OpenNI và các chương trình kèm theo.

- ***Bước 4:***

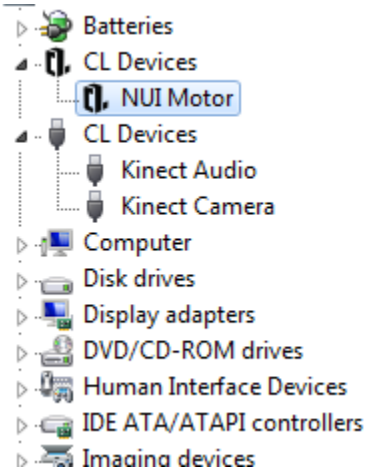
  Vào lại cửa sổ như ở bước 1, lúc này Kinect được nhận bởi PrimeSense. Ta nhấp phải chuột trên Kinect Motor và chọn Update Driver Software

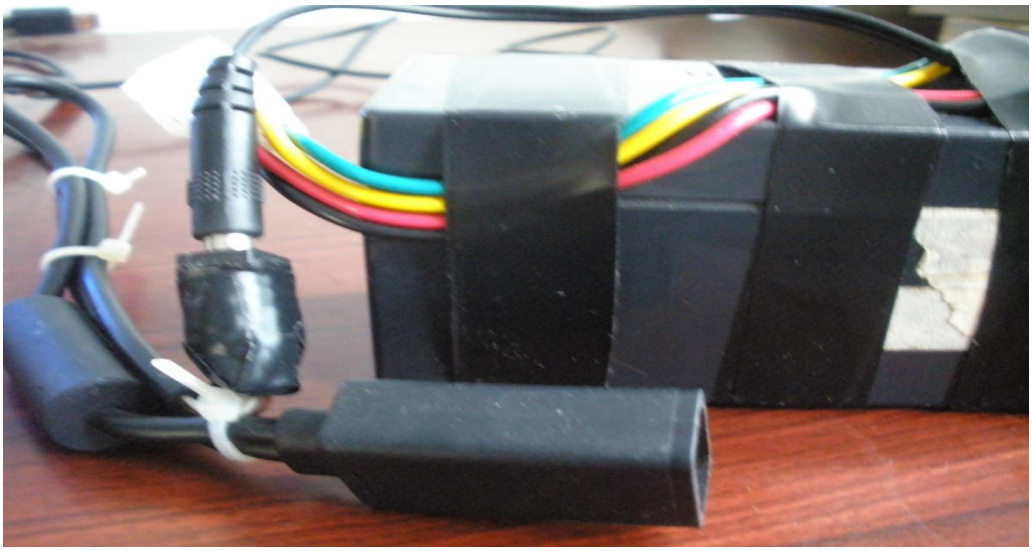


Và duyệt tới thư mục driver của CL:



Lúc này ta có thể sử dụng chức năng điều khiển động cơ Kinect trên thư viện OpenNI kết hợp với CL.





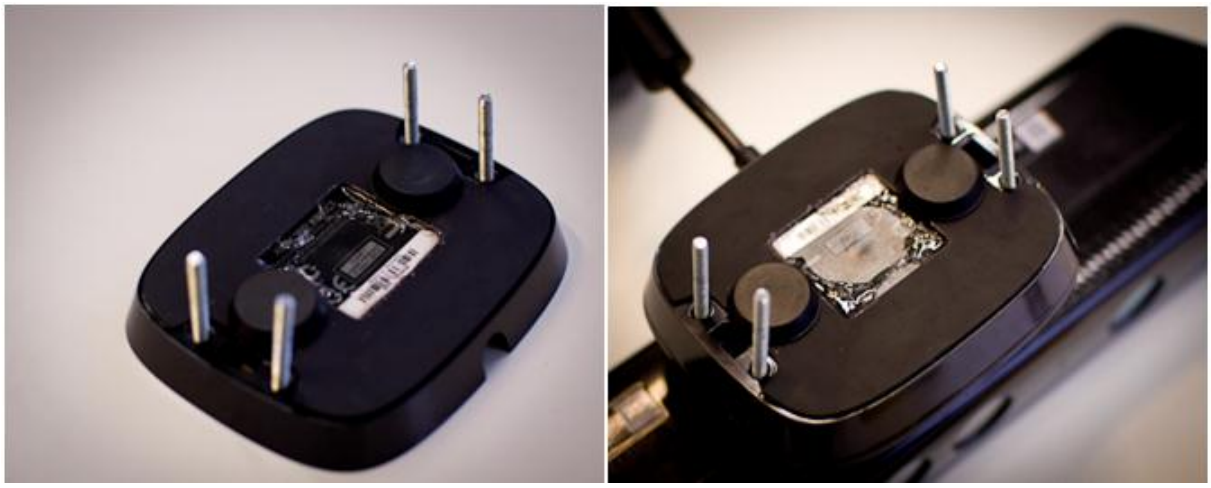
- **Tạo đế cho Kinect để gắn lên robot**

  Đây là bước quan trọng nhằm giữ sự an toàn cho Kinect và tránh bị rung khi di chuyển (sẽ ảnh hưởng tới kết quả xử lý ảnh trên máy tính).

  Đầu tiên tháo phần đế gắn trên Kinect:

Sau khi tháo:

Khoan bốn lỗ và gắn bốn ốc từ trong đế ra:



Và kết quả cuối cùng:

Đồ án tốt nghiệp

Thiết kế bộ Điều khiển trượt cho Robot 2 bậc tự do

Thiết kế bộ Điều khiển trượt cho tay máy Robot 2 bậc tự do và mô phỏng trên Matlab – Simulink

CHƯƠNG I: TỔNG QUAN VỀ ROBOT CÔNG NGHIỆP

I.1.Robot công nghiệp:

I.1.1. Sự ra đời của Robot công nghiệp :

Thuật ngữ "Robot" lần đầu tiên xuất hiện năm 1922 trong tác phẩm "Rosum's Universal Robot " của Karal Capek. Theo tiếng Séc thì Robot là người làm tạp dịch. Trong tác phẩm này nhân vật Rosum và con trai ông đã tạo ra những chiếc máy gần giống như con người để hầu hạ con người.

Hơn 20 năm sau, ước mơ viễn tưởng của Karel Capek đã bắt đầu hiện thực. Ngay sau chiến tranh thế giới lần thứ 2, ở Mỹ đã xuất hiện những tay máy chép hình điều khiển từ xa, trong các phòng thí nghiệm phóng xạ. Năm 1959, Devol và Engelber đã chế tạo Robot công nghiệp đầu tiên tại công ty Unimation.

Năm 1967 Nhật Bản mới nhập chiếc Robot công nghiệp đầu tiên từ công ty AMF của Mỹ. Đến năm 1990 có hơn 40 công ty của Nhật, trong đó có những công ty khổng lồ như Hitachi, Mitsubishi và Honda đã đưa ra thị trường nhiều loại Robot nổi tiếng.

Từ những năm 70, việc nghiên cứu nâng cao tính năng của robot đã chú ý nhiều đến sự lắp đặt thêm các cảm biến ngoại tín hiệu để nhận biết môi trường làm việc. Tại trường đại học tổng hợp Stanford, người ta đã tạo ra loại Robot lắp ráp tự động điều khiển bằng vi tính trên cơ sở xử lý thông tin từ các cảm biến lực và thị giác. Vào thời gian này công ty IBM đã chế tạo Robot có các cảm biến xúc giác và cảm biến lực điều khiển bằng máy vi tính để lắp ráp các máy in gồm 20 cụm chi tiết .

Những năm 90 do áp dụng rộng rãi các tiến bộ khoa học về vi xử lý và công nghệ thông tin, số lượng Robot công nghiệp đã tăng nhanh, giá thành giảm đi rõ rệt, tính năng đã có nhiều bước tiến vượt bậc. Nhờ vậy Robot công nghiệp đã có vị trí quan trọng trong các dây truyền sản xuất

hiện đại. Ngày nay, chuyên ngành khoa học nghiên cứu về Robot "Robotics" đã trở thành một lĩnh vực rộng trong khoa học, bao gồm các vấn đề cấu trúc cơ cấu động học, động lực học, lập trình quỹ đạo, cảm biến tín hiệu, điều khiển chuyển động v.v…

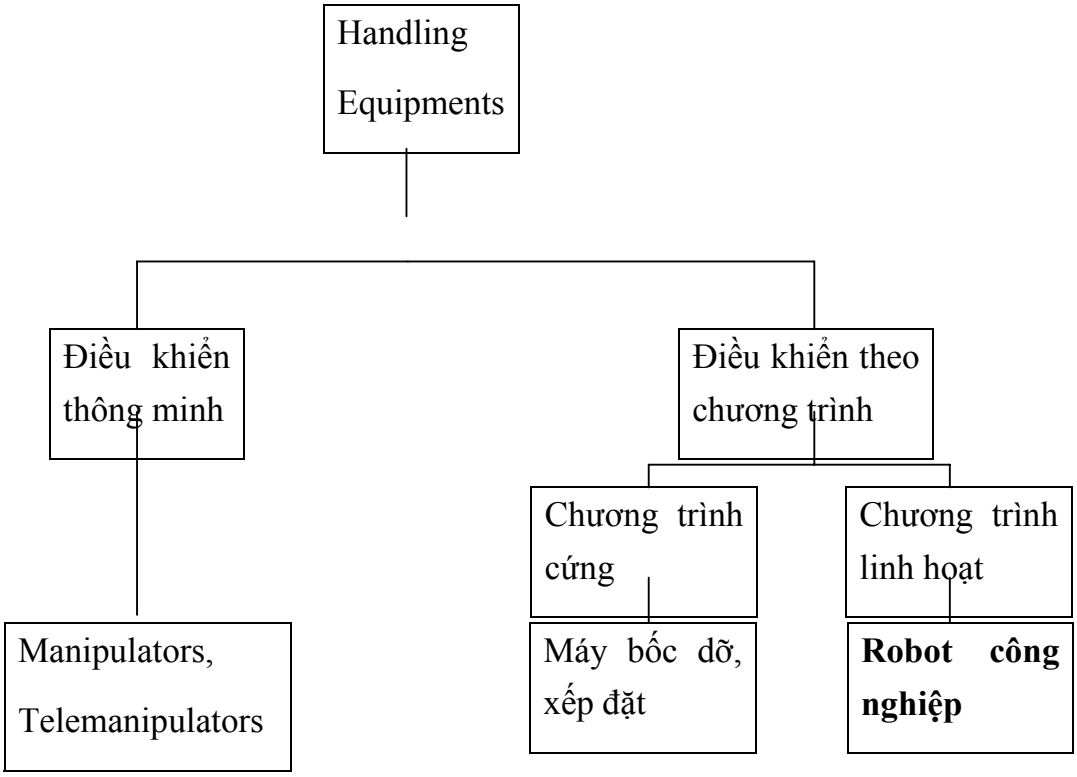
I.1.2.Phân loại tay máy Robot công nghiệp:

Ngày nay, khi nói đến Robot thường ta hay hình dung ra một cơ chế máy móc tương tự con người, có khả năng sử dụng công cụ lao động để thực hiện các công việc thay cho con người, thậm chí có thể tính toán hay có khả năng hành động theo ý chí.

Trong thực tiễn kỹ thuật, khái niệm Robot hiện đại được hiểu khá rộng, mà theo đó Robot là "*tất cả các hệ thống kỹ thuật có khả năng cảm nhận và xử lý thông tin cảm nhận được, để sau đó đưa ra hành xử thích hợp*". Theo cách hiểu này, các hệ thống xe tự hành, hay thậm chí một thiết bị xây dựng có trang bị cảm biến thích hợp như Camera, cũng được gọi là Robot. Các khái niệm như Hexapod, Parallel Robot, Tripod, Gait Biped, Manipulator Robocar hay Mobile Robot nhằm chỉ vào các hệ thống Robot không còn gắn liền với các hình dung ban đầu của con người.

Trong nội dung đồ án chỉ nhằm vào đối tượng Robot công nghiệp (RBCN), thực chất là một thiết bị tay máy (Handling Equipment). Công nghệ tay máy (Handling Technology) là công nghệ của dạng *thiết bị kỹ thuật có khả năng thực hiện các chuyển động theo nhiều trục trong không gian*, tương tự như ở con người.

Về cơ bản có thể phân thiết bị tay máy (hình 1.1) thành 2 loại chính : Điều khiển (ĐK) theo chương trình hay ĐK thông minh :



Hình 1.1 : ***Phân loại thiết bị tay máy***

+ Loại ĐK theo chương trình gồm 2 họ:

- *Chương trình cứng* : Các thiết bị bốc dỡ, xếp đặt có chương trình hoạt động cố định. Ta hay gặp họ này trong các hệ thống kho hiện đại. Chúng có rất ít trục chuyển động và chỉ thu thập thông tin về quãng đường qua các tiếp điểm hành trình. Ta không thể ĐK chúng theo một quỹ đạo mong muốn.

- *Chương trình linh hoạt* : Là họ Robot mà người sử dụng có khả năng thay đổi chương trình ĐK chúng tuỳ theo đối tượng công tác. Ta hay gặp chúng trong các công đoạn như hàn, sơn hay lắp ráp của công nghiệp Ôtô. Trong hình 1.1 ta gọi là **Robot công nghiệp**.

+ Loại ĐK thông minh có 2 kiểu chính :

- *Manipulator*: Là loại tay máy được ĐK trực tiếp bởi con người, có khả năng lặp lại các chuyển động của tay người. Bản chất là dạng thiết

bị hỗ trợ cho sự khéo léo, cho trí tuệ, cho hệ thống giác quan (Complex Sensorics) và kinh nghiệm của người sử dụng. Hay được sử dụng trong các nhiệm vụ cần chuyển động phức hợp có tính chính xác cao, hay môi trường nguy hiểm cho sức khoẻ, môi trường khó tiếp cận v.v...

- *Telemanipulator*: Là loại Manipulator được điều khiển từ xa và người ĐK phải sử dụng hệ thống Camera để quan sát môi trường sử dụng.

Theo tiêu chuẩn châu Âu EN775 và VDI 2860 của Đức có thể hiểu "*Robot công nghiệp là một Automat sử dụng vạn năng để tạo chuyển động nhiều trục, có khả năng lập trình linh hoạt các chuỗi chuyển động và quãng đường (góc) để tạo nên chuyển động theo quỹ đạo. Chúng có thể được trang bị thêm các ngón (Grippe), dụng cụ hay các công cụ gia công và có thể thực hiện các nhiệm vụ của đôi tay (Handling) hay các nhiệm vụ gia công khác"*

Như vậy, RBCN khác các loại tay máy còn lại ở 2 điểm chính là *"sử dụng vạn năng"* và *"khả năng lập trình linh hoạt"*.

I.2. Ứng dụng của Robot công nghiệp :

I.2.1.Mục tiêu ứng dụng Robot công nghiệp :

Mục tiêu ứng dụng Robot công nghiệp nhằm nâng cao năng suất dây truyền công nghệ, giảm giá thành, nâng cao chất lượng và khả năng cạnh tranh của sản phẩm, đồng thời cải thiện điều kiện lao động. Điều đó xuất phát từ những ưu điểm cơ bản của Robot đó là :

- Robot có thể thực hiện một quy trình thao tác hợp lý bằng hoặc hơn người thợ lành nghề một cách ổn định trong suốt thời gian dài làm việc. Do đó Robot giúp nâng cao chất lượng và khả năng cạnh tranh của sản phẩm.

- Khả năng giảm giá thành sản phẩm do ứng dụng Robot là vì giảm được đáng kể chi phí cho người lao động.

- Robot giúp tăng năng suất dây chuyền công nghệ.

- Robot giúp cải thiện điều kiện lao động. Đó là ưu điểm nổi bật nhất mà chúng ta cần quan tâm. Trong thực tế sản xuất có rất nhiều nơi người lao động phải làm việc trong môi trường ô nhiễm, ẩm ướt, nóng nực. Thậm

chí rất độc hại đến sức khoẻ và tính mạng như môi trường hoá chất, điện từ, phóng xạ …

I.2.2.Các lĩnh vực ứng dụng Robot công nghiệp :

Robot công nghiệp được ứng dụng rất rộng rãi trong sản xuất, xin được nêu ra một số lĩnh vực chủ yếu :

- Kỹ nghệ đúc
- Gia công áp lực
- Các quá trình hàn và nhiệt luyện
- Công nghệ gia công lắp ráp
- Phun sơn, vận chuyển hàng hoá (Robocar)…

I.2.3. Các xu thế ứng dụng Robot trong tương lai :

- Robot ngày càng thay thế nhiều lao động
- Robot ngày càng trở lên chuyên dụng
- Robot ngày càng đảm nhận được nhiều loại công việc lắp ráp
- Robot di động ngày càng trở lên phổ biến
- Robot ngày càng trở lên tinh khôn

I.2.4. Tình hình tiếp cận và ứng dụng Robot công nghiệp ở Việt Nam :

Trong giai đoạn trước năm 1990, hầu như trong nước hoàn toàn chưa du nhập về kỹ thuật Robot, thậm chí chưa nhận được nhiều thông tin kỹ thuật về lĩnh vực này. Tuy vậy, với mục tiêu chủ yếu là tiếp cận lĩnh vực mới mẻ này trong nước đã có triển khai các đề tài nghiên cứu khoa học cấp nhà nước: Đề tài 58.01.03 và 52B.03.01.

Giai đoạn tiếp theo từ năm 1990 các ngành công nghiệp trong nước bắt đầu đổi mới. Nhiều cơ sở đã nhập ngoại nhiều loại Robot công nghiệp phục vụ các công việc như: tháo lắp dụng cụ, lắp ráp linh kiện điện tử, hàn vỏ Ôtô xe máy, phun phủ các bề mặt …

Một sự kiện đáng chú ý là tháng 4 năm 1998, nhà máy Rorze/Robotech đã bước vào hoạt động ở khu công nghiệp Nomura Hải Phòng. Đây là nhà máy đầu tiên ở Việt Nam chế tạo và lắp ráp Robot.

Những năm gần đây, Trung tâm nghiên cứu kỹ thuật Tự động hóa, Trường đại học Bách Khoa Hà Nội, đã nghiên cứu thiết kế một kiểu Robot mới là Robot RP. Robot RP thuộc loại Robot phỏng sinh (bắt chước cơ cấu tay người). Hiện nay đã chế tạo 2 mẫu: Robot RPS-406 dùng để phun men và Robot RPS-4102 dùng trong công nghệ bề mặt.

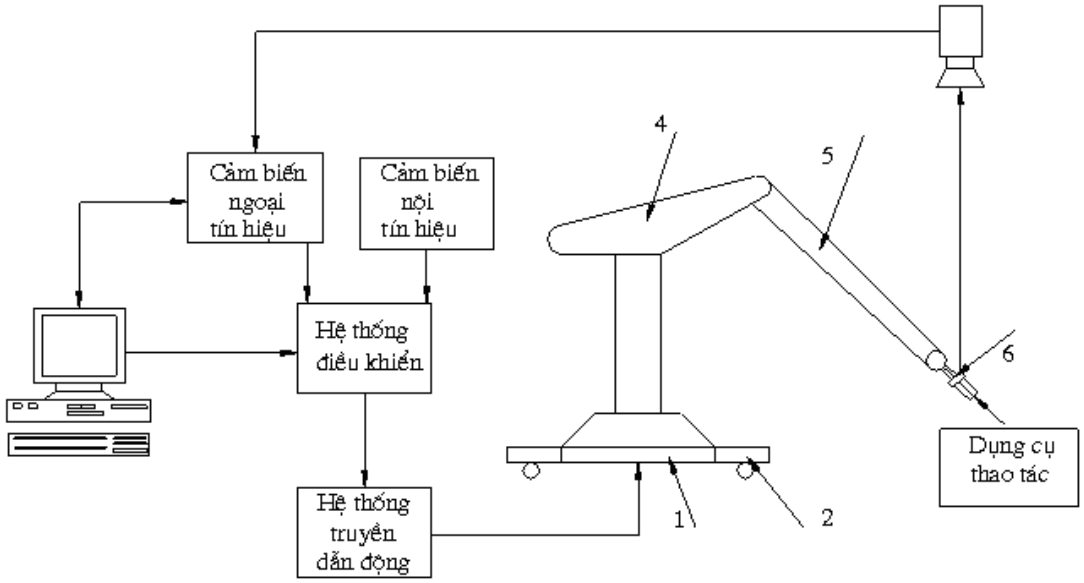
Ngoài ra Trung tâm còn chế tạo các loại Robot khác như: Robot SCA mini dùng để dạy học, Robocar công nghiệp phục vụ phân xưởng, Robocar chữ thập đỏ cho người tàn tật … Bên cạnh đó còn xây dựng các thuật toán mới để điều khiển Robot, xây dựng "thư viện" các mô hình của Robot trên máy tính …

I.3.Cấu trúc của Robot công nghiệp:

I.3.1.Các bộ phận cấu thành Robot công nghiệp :

Trên hình 1.2 giới thiệu các bộ phận chủ yếu của Robot công nghiệp:

Tay máy gồm các bộ phận: Đế 1 đặt cố định hoặc gắn liền với xe di động 2, thân 3, cánh tay trên 4, cánh tay dưới 5, bàn kẹp 6.



Hình 1.2: ***Các bộ phận cấu thành Robot công nghiệp***

Hệ thống truyền dẫn động có thể là cơ khí, thuỷ khí hoặc điện khí: là bộ phận chủ yếu tạo nên sự chuyển dịch các khớp động.

Hệ thống điều khiển đảm bảo sự hoạt động của Robot theo các thông tin đặt trước hoặc nhận biết trong quá trình làm việc.

Hệ thống cảm biến tín hiệu thực hiện việc nhận biết và biến đổi thông tin về hoạt động của bản thân Robot (cảm biến nội tín hiệu) và của môi trường, đối tượng mà Robot phục vụ (cảm biến ngoại tín hiệu).

I.3.2.Bậc tự do và các toạ độ suy rộng :

I.3.2.1.Bậc tự do :

Robot công nghiệp là loại thiết bị tự động nhiều công dụng. Cơ cấu tay máy của chúng phải được cấu tạo sao cho bàn kẹp giữ vật kẹp theo một hướng nhất định nào đó và di chuyển dễ dàng trong vùng làm việc. Muốn vậy cơ cấu tay máy phải đạt được một số *bậc tự do* chuyển động.

Thông thường các khâu của cơ cấu tay máy được nối ghép với nhau bằng các khớp quay hoặc khớp tịnh tiến. Gọi chung chúng là *khớp động*. Các khớp quay hoặc khớp tịnh tiến đều thuộc khớp động học loại 5.

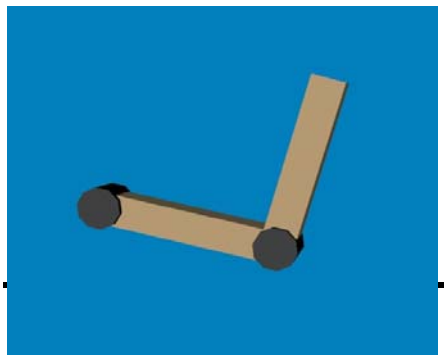
Công thức tính số bậc tự do :

$$W = 6n - \sum_{1}^{5} i p_i \qquad (1.1)$$

với n : số khâu động

P_i : số khớp loại i

Ví dụ: Tay máy có 2 khớp quay như hình vẽ 1.3 :



Số khâu động n = 2

Khớp quay là khớp loại 5 .

Do đó W = 6.2 – (5.1 + 5.1) = 2 bậc tự do

Hình 1.3: ***Tay máy 2 khớp quay***

I.3.2.2. Toạ độ suy rộng :

Các cấu hình khác nhau của cơ cấu tay máy trong từng thời điểm xác định bằng các độ dịch chuyển góc hoặc độ dịch chuyển dài của các khớp quay hoặc khớp tịnh tiến.

Các độ dịch chuyển tức thời đó, so với giá trị ban đầu nào đó lấy làm mốc tính toán, được gọi là các toạ độ suy rộng (generalized joint coordinates). Ở đây ta gọi chúng là các *biến khớp* (toạ độ suy rộng) của cơ cấu tay máy và biểu thị bằng :

$$q_i = \delta_i \theta_i + (1 - \delta_i) S_i \quad (1.2)$$

với $\delta_i = \begin{cases} 1, \text{đối với khớp quay} \\ 0, \text{đối với khớp tịnh tiến} \end{cases}$

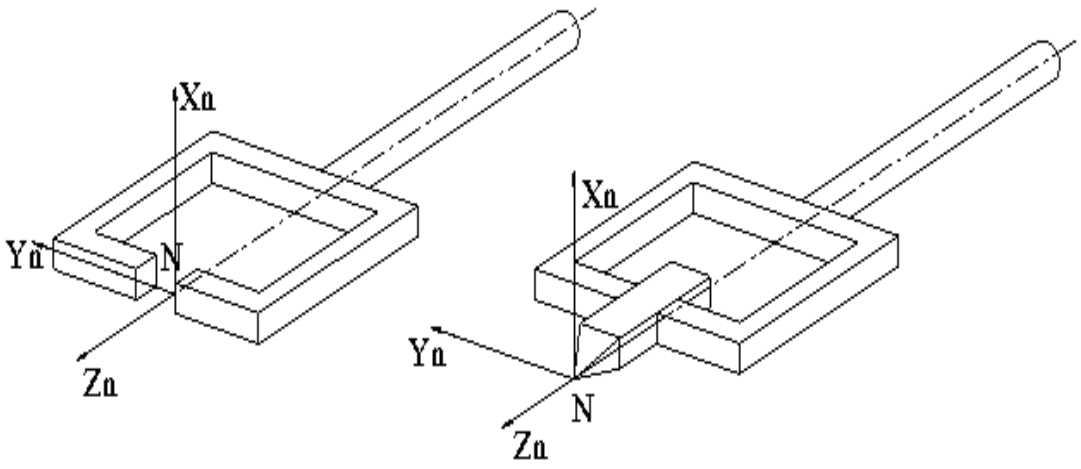
θ_i - Độ dịch chuyển góc của các khớp quay

S_i - Độ dịch chuyển tịnh tiến của các khớp tịnh tiến

I.3.3.Nhiệm vụ lập trình điều khiển Robot:

I.3.3.1. Định vị và định hướng tại "điểm tác động cuối" :

Khâu cuối cùng của tay máy thường là bàn kẹp (gripper) hoặc là khâu gắn liền với dụng cụ thao tác (tool). Điểm mút của khâu cuối cùng là điểm đáng quan tâm nhất vì đó là điểm tác động của Robot lên đối tác và được gọi là "điểm tác động cuối" (end-effector). Trên hình 1.4 điểm E là "điểm tác động cuối".

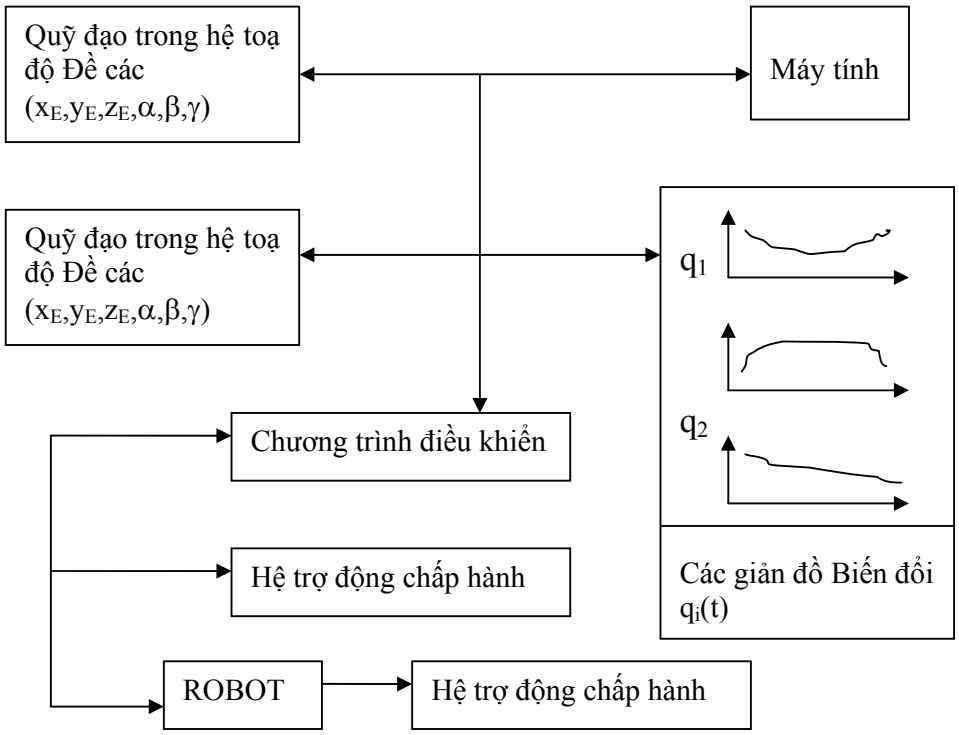


Hình 1.4: ***Định vị và định hướng tại " điểm tác động cuối"***

Chính tại "điểm tác động cuối" E này cần quan tâm không những vị trí nó chiếm trong không gian làm việc mà cả hướng tác động của khâu cuối đó. Vị trí của điểm E được xác định bằng 3 toạ độ x_E, y_E, z_E trong hệ trục toạ độ cố định. Còn hướng tác động của khâu cuối có thể xác định bằng 3 trục x_n,y_n, z_n gắn liền với khâu cuối tại điểm E, hoặc bằng 3 thông số góc α,β,γ nào đó.

I.3.3.2. Lập trình điều khiển Robot công nghiệp :

Trên hình 1.5 mô tả 1 sơ đồ lập trình điều khiển Robot công nghiệp. Khi robot nhận nhiệm vụ thực hiện một quy trình công nghệ nào đó, ví dụ "điểm tác động cuối" E phải bám theo một hành trình cho trước. Quỹ đạo hành trình này thường cho biết trong hệ toạ độ Đề các x_0, y_0, z_0 cố định. Ở mỗi vị trí mà điểm E đi qua xác định bằng 3 toạ độ cố định x_E, y_E, z_E và 3 thông số góc định hướng α,β,γ. Từ các thông số trong hệ toạ độ Đề các đó tính toán các giá trị biến khớp q_i tương ứng với mỗi thời điểm t. Đó là nội dung của bài toán Động học ngược sẽ trình bày trong chương II.



Hình 1.5: ***Sơ đồ lập trình điều khiển***

I.4. Các phép biến đổi toán học cho Robot :

I.4.1.Biến đổi toạ độ dùng Ma trận:

I.4.1.1. Vector điểm và toạ độ thuần nhất :

Vector điểm (point vector) dùng để mô tả vị trí của điểm trong không gian 3 chiều.

Trong không gian 3 chiều, một điểm M có thể được biểu diễn bằng nhiều vector trong các hệ toạ độ (coordinate frame) khác nhau:

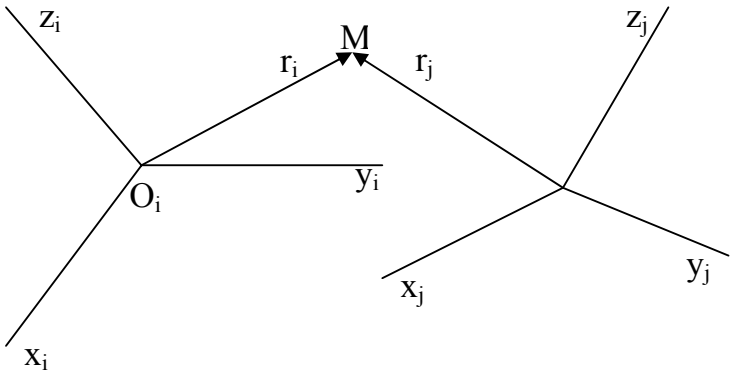
Trong hệ toạ độ $o_ix_iy_iz_i$ điểm M xác định bằng vector r_i :

$$r_i = (r_{xi}, r_{yi}, r_{zi})^T \qquad (1.3)$$

và cùng điểm M đó trong hệ toạ độ $o_jx_jy_jz_j$ được mô tả bởi vector r_j :

$$r_j = (r_{xj}, r_{yj}, r_{zj})^T \qquad (1.4)$$

Ký hiệu ( )^T là biểu thị phép chuyển vị (Transportation) vector hàng thành vector cột.



Hình 1.6: ***Biểu diễn 1 điểm trong không gian***

Vector $r = (r_x, r_y, r_z)^T$ trong không gian 3 chiều, nếu được bổ sung thêm một thành phần thứ 4 và thể hiện bằng 1 vector mở rộng :

$$\tilde{r} = (\omega r_x, \omega r_y \omega r_z, \omega) \qquad (1.5)$$

thì đó là cách biểu diễn vector điểm trong không gian toạ độ thuần nhất (homogeneous coordinate).

Để đơn giản có thể bỏ qua ký hiệu (˜) đối với vector mở rộng (1.5)

Các toạ độ thực của vector mở rộng này vẫn là:

$$r_x = \frac{\omega r_x}{\omega} \qquad r_y = \frac{\omega r_y}{\omega} \qquad r_z = \frac{\omega r_z}{\omega} \qquad (1.6)$$

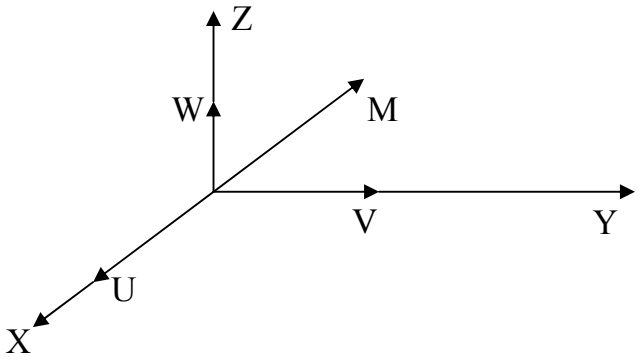
Không phải duy nhất có một cách biểu diễn vector trong không gian tọa độ thuần nhất, mà nó phụ thuộc vào giá trị của ω. Nếu lấy ω = 1 thì các tọa độ biểu diễn bằng toạ độ có thực. Trong trường hợp này vector mở rộng được viết là:

$$r = (r_x, r_y, r_z)^T \qquad (1.7)$$

Nếu lấy $\omega \neq 1$ thì các toạ độ biểu diễn gấp ω lần toạ độ thực, nên có thể gọi ω là *hệ số tỷ lệ*. Khi cần biểu diễn sự thay đổi toạ độ kèm theo thì có sự biến dạng tỷ lệ thì dùng $\omega \neq 1$.

I.4.1.2.Quay hệ toạ độ dùng Ma trận 3x3:

Trước hết thiết lập quan hệ giữa 2 hệ toạ độ XYZ và UVW chuyển động quay tương đối với nhau khi gốc O của 2 hệ vẫn trùng nhau (hình 1.7)



Hình 1.7: ***Các hệ toạ độ***

Gọi (i_x, j_y, k_z) và (i_u, j_v, k_w) là các vector đơn vị chỉ phương các trục OXYZ và OUVW tương ứng.

Một điểm M nào đó được biểu diễn trong hệ toạ độ OXYZ bằng vector:

$$r_{xyz}=(r_x,r_y,r_z)^T \quad (1.8)$$

còn trong hệ toạ độ OUVW bằng vector:

$$r_{uvw} = (r_u,r_v,r_w)^T \quad (1.9)$$

Như vậy :

$$r = r_{uvw}= r_u i_u + r_v j_v + r_w k_w$$

$$r = r_{xyz}= r_x i_x + r_y j_y + r_z k_z \quad (1.10)$$

Từ đó ta có

$$\left.\begin{aligned} r_x &= i_x.r = i_x i_u r_u + i_x j_v r_v + i_x k_w r_w \\ r_y &= j_y.r = j_y i_u r_u + j_y j_v r_v + j_y k_w r_w \\ r_z &= k_z.r = k_z i_u r_u + k_z j_v r_v + k_z k_w r_w \end{aligned}\right\} \quad (1.11)$$

Hay viết dưới dạng ma trận:

$$\begin{pmatrix} r_x \\ r_y \\ r_z \end{pmatrix} = \begin{pmatrix} i_x i_u & i_x j_v & i_x k_w \\ j_y i_u & j_y j_v & j_y k_w \\ k_z i_u & k_z j_v & k_z k_w \end{pmatrix} . \begin{pmatrix} r_u \\ r_v \\ r_w \end{pmatrix} \qquad (1.12)$$

Gọi R là Ma trận quay (rotation) 3x3 với các phần tử là tích vô hướng 2 vector chỉ phương các trục tương ứng của 2 hệ toạ độ OXYZ và OUVW.

Vậy (1.12) được viết lại là:

$$\left.\begin{aligned} r_{xyz} &= R.r_{uvw} \\ r_{uvw} &= R^{-1}.r_{xyz} \end{aligned}\right\} \qquad (1.13)$$

I.4.1.3.Biến đổi Ma trận dùng toạ độ thuần nhất:

Bây giờ thiết lập quan hệ giữa 2 hệ toạ độ: hệ toạ độ $o_jx_jy_jz_j$ sang hệ toạ độ mới $o_ix_iy_iz_i$. Chúng không những quay tương đối với nhau mà tịnh tiến cả gốc toạ độ: gốc o_j xác định trong hệ $x_iy_iz_i$ bằng vector p:

$$p=(a,-b,-c,1)^T \qquad (1.14)$$

Giả sử vị trí của điểm M trong hệ toạ độ $x_jy_jz_j$ được xác định bằng vector r_j:

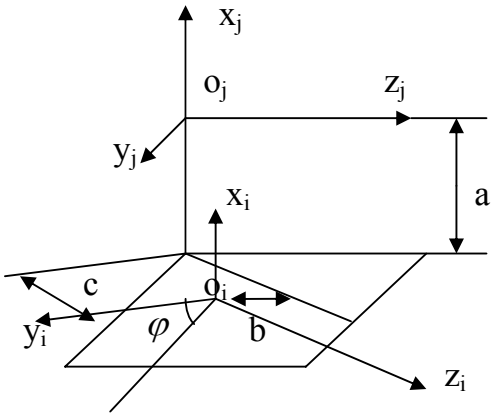
$$r_j = (x_jy_jz_j,1)^T \qquad (1.15)$$

và trong hệ toạ độ $x_iy_iz_i$ điểm M được xác định bằng vector r_i:

$$r_i = (x_iy_iz_i,1)^T \qquad (1.16)$$

Từ hình (1.8) có thể dễ dàng thiết lập mối quan hệ giữa các toạ độ:

$$\left.\begin{aligned} x_i &= x_j + at_j \\ y_i &= y_j\cos\varphi - z_j\sin\varphi - bt_j \\ z_i &= y_j\sin\varphi + z_j\cos\varphi - ct_j \\ t_i &= t_j = 1 \end{aligned}\right\} \qquad (1.17)$$



Hình 1.8: ***Các hệ toạ độ***

Sắp xếp các hệ số ứng với x_j,y_j,z_j và t_j thành một ma trận:

$$T_{ij} = \begin{bmatrix} 1 & 0 & 0 & a \\ 0 & \cos\varphi & -\sin\varphi & -b \\ 0 & \sin\varphi & \cos\varphi & -c \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (1.18)$$

và viết phương trình biến đổi toạ độ như sau:

$$r_i = T_{ij}\, r_j \qquad (1.19)$$

Ma trận T_{ij} biểu thị bằng ma trận 4x4 như phương trình (1.18) và gọi là ma trận thuần nhất. Nó dùng để biến đổi vector mở rộng từ hệ toạ độ thuần nhất này sang hệ toạ độ thuần nhất kia.

I.4.1.4. Ý nghĩa hình học của Ma trận thuần nhất:

Từ (3.19) nhận thấy ma trận thuần nhất 4x4 là một ma trận gồm 4 khối :

$$T_{ij} = \begin{pmatrix} 1 & 0 & 0 & a \\ 0 & \cos\varphi & -\sin\varphi & -b \\ 0 & \sin\varphi & \cos\varphi & -c \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad (1.20)$$

Hoặc viết rút gọn là:

$$T_{ij} = \begin{pmatrix} R_{ij} & p \\ 0 & 1 \end{pmatrix} \qquad (1.21)$$

Trong đó:

R_{ij} - ma trận quay 3x3

p – ma trận 3x1 biểu thị 3 toạ độ của điểm gốc hệ toạ độ 0_j trong hệ toạ độ o_i, x_i, y_i, z_i

1x3 – ma trận không

1x1 – ma trận đơn vị

Như vậy ma trận thuần nhất 4x4 là ma trận 3x3 mở rộng, thêm ma trận 3x1 biểu thị sự chuyển dịch gốc toạ độ và phần tử a_{44} biểu thị hệ số tỷ lệ.

Dễ dàng nhận thấy ma trận R_{ij} chính là ma trận quay 3x3, nếu suy từ ma trận quay trong (1.12) sang trường hợp hình 1.8 ta có:

$$R_{ij} = \left[a_{ij} \right] = \begin{pmatrix} \cos(x_i, x_j) & \cos(x_i, y_j) & \cos(x_i, z_j) \\ \cos(y_i, x_j) & \cos(y_i, y_j) & \cos(y_i, z_j) \\ \cos(z_i, x_j) & \cos(z_i, y_j) & \cos(z_i, z_j) \end{pmatrix} \qquad (1.22)$$

và các góc cosin chỉ phương này đều liên hệ đến góc φ (hình 1.8).

Nếu chú ý về quan hệ giữa 2 cặp trục,ví dụ, $\cos(x_i,y_j) = \cos(y_i, x_j)$ …

ở đây dễ dàng nhận được biểu thức:

$$R_{ij} = R_{ij}^{-1} = R_{ij}^{T} \qquad (1.23)$$

Mô tả tổng quát hơn nếu một điểm M nào đó được xác định trong hệ toạ độ thuần nhất UVW bằng vectơ mở rộng r_{uvw} , thì trong hệ toạ độ thuần nhất XYZ điểm đó xác định bằng vector mở rộng r_{xyz}:

$$R_{xyz} = T.r_{uvw} \qquad (1.24)$$

Trong đó T là ma trận thuần nhất 4x4, có thể viết khai triển ở dạng sau:

$$T=\begin{pmatrix} n_x & s_x & a_x & p_x \\ n_y & s_y & a_y & p_y \\ n_z & s_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad (1.25)$$

hoặc $T=\begin{pmatrix} R & p \\ 0 & 1 \end{pmatrix}$ (1.26)

Ta tìm hiểu ý nghĩa hình học của ma trận T. Như đã trình bày khi phân tích các khối của ma trận 4x4, ma trận 3x1 tương ứng với toạ độ điểm gốc của hệ toạ độ UVW biểu diễn trong hệ XYZ.

Nếu 2 gốc toạ độ trùng nhau thì các thành phần của ma trận 3x1 này đều là 0. Khi đó xét trường hợp:

$r_{uvw} = (1,0,0,1)^T$

tức là $r_{xyz} = i_u$

thì dễ dàng nhận thấy cột thứ nhất hoặc vectơ **n** của ma trận (1.25) chính là các toạ độ của vectơ chỉ phương trục OU biểu diễn trong hệ toạ độ XYZ.

Tương tự khi xét các trường hợp

$r_{uvw} = (0,1,0,1)^T$

và $r_{uvw} = (0,0,1,1)^T$

cũng đi đến nhận xét cột thứ 2 (hoặc vectơ **s**) ứng với các toạ độ của vectơ chỉ phương trục OV và cột thứ 3 (hoặc vectơ **a**) ứng với các toạ độ vector chỉ phương trục OW.

Như vậy*, ma trận thuần nhất T 4x4 hoàn toàn xác định vị trí và định hướng của hệ toạ độ UVW so với hệ toạ độ XYZ. Đó là ý nghĩa hình học của ma trận thuần nhất 4x4.*

I.4.2.Các phép biến đổi cơ bản:

I.4.2.1.Phép biến đổi tịnh tiến:

Từ (1.18) hoặc (1.25), biểu thị ma trận thuần nhất khi chỉ có biến đổi tịnh tiến mà không có quay ($\varphi = 0$), ta có:

$$T = \begin{bmatrix} 1 & 0 & 0 & p_x \\ 0 & 1 & 0 & p_y \\ 0 & 0 & 1 & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = T_p(p_x, p_y, p_z) \qquad (1.27)$$

Đó là ma trận biến đổi tịnh tiến (Tranlation)

Gọi **u** là vector biểu diễn một điểm trong không gian cần dịch chuyển tịnh tiến:

$u = (x, y, z)^T$

và **p** là vector chỉ hướng và độ dài cần dịch chuyển

$p = \quad (p_x, p_y, p_z)^T$

thì **v** là vector biểu diễn điểm toạ độ trong không gian đã được tịnh tiến tới:

$\mathrm{v} = T_p(\ \mathrm{v} = T_p(p_x, p_y, p_z)^T \mathrm{u} \qquad (1.28)$

I.4.2.2. Phép quay quanh các trục toạ độ :

Từ ma trận quay 3x3 trong biểu thức (1.12) ta xây dựng ma trận $R(x,\alpha)$ cho trường hợp hệ toạ độ UVW quay quanh trục OX một góc α nào đó. Trong trường hợp này $i_x = i_u$:

$$R(x,\alpha) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha & 0 \\ 0 & \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (1.29)$$

Tương ứng cho trường hợp quay quanh trục OY một góc φ :

$$R(y,\varphi)=\begin{bmatrix}\cos\varphi & 0 & \sin\varphi & 0\\ 0 & 1 & 0 & 0\\ -\sin\varphi & 0 & \cos\varphi & 0\\ 0 & 0 & 0 & 1\end{bmatrix} \qquad (1.30)$$

và trường hợp quay quanh trục OZ một góc θ :

$$R(z,\theta)=\begin{bmatrix}\cos\theta & -\sin\theta & 0 & 0\\ \sin\theta & \cos\theta & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix} \qquad (1.31)$$

Cột thứ 4 của các ma trận 4x4 trên có 3 phần tử đều bằng 0 vì ở đây không có sự tịnh tiến. Các ma trận này được gọi là các ma trận quay (rotation) cơ bản. Các ma trận quay khác có thể xây dựng từ các ma trận cơ bản này.

CHƯƠNG II: HỆ PHƯƠNG TRÌNH ĐỘNG HỌC VÀ ĐỘNG LỰC HỌC CỦA ROBOT CÔNG NGHIỆP:

II.1. Hệ phương trình động học Robot :

II.1.1. Đặt vấn đề :

Cơ cấu chấp hành của Robot thường là một cơ cấu hở gồm một chuỗi các khâu (link) nối với nhau bằng các khớp (joints). Các khớp động này là khớp quay (R) hoặc khớp tịnh tiến (T). Để Robot có thể thao tác linh hoạt cơ cấu chấp hành của nó phải có cấu tạo sao cho điểm mút của khâu cuối cùng đảm bảo dễ dàng di chuyển theo một quỹ đạo nào đó, đồng thời

khâu này có một hướng nhất định theo yêu cầu. Khâu cuối cùng này thường là bàn kẹp (griper), điểm mút của nó chính là "điểm tác động cuối" E (end-effector).

Để xét vị trí và hướng của E trong không gian ta gắn vào nó một hệ toạ độ động thứ n và gắn với mỗi khâu động một hệ toạ độ khác, còn gắn liền với giá đỡ một hệ toạ độ cố định. Đánh số ký hiệu các hệ này từ 0 đến n bắt đầu từ giá cố định. Khi khảo sát chuyển động của Robot cần biết "định vị và định hướng" tại điểm tác động cuối trong mọi thời điểm. Các lời giải của bài toán này được xác định từ những phương trình Động học của Robot. Các phương trình này là mô hình Động học của Robot. Chúng được xây dựng trên cơ sở thiết lập các mối quan hệ giữa các hệ toạ độ động nói trên so với hệ toạ độ cố định.

II.1.2. Xác định trạng thái của Robot tai điểm tác động cuối :

Trạng thái của Robot tại "điểm tác động cuối" hoàn toàn xác định bằng sự định vị và định hướng tại điểm tác động cuối đó.

Như đã đề cập ở phần I.4.1.4 biểu thị sự định vị và định hướng đó bằng ma trận trạng thái cuối T_E :

$$T_E = \begin{bmatrix} n_x & s_x & a_x & p_x \\ n_y & n_y & a_y & p_y \\ n_z & s_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (2.1)$$

Trong đó các phần tử của ma trận 3x1 là toạ độ p_x , p_y, p_z của "điểm tác động cuối" E. Mỗi cột của ma trận quay 3x3 là một vectơ đơn vị chỉ phương một trục của hệ toạ độ động NSA (chính là UVW) biểu diễn trong toạ độ cố định XYZ.

Hệ toạ độ gắn liền với bàn kẹp của Robot có các vectơ đơn vị chỉ phương các trục như sau :

a - vector có hướng tiếp cận (approach) với đối tác .

s - vector có hướng đường trượt (sliding) đóng mở bàn kẹp .

n - vector pháp tuyến (normal).

II.1.3. Mô hình động học :

II.1.3.1. Ma trận quan hệ :

Chọn hệ toạ độ cố định gắn liền với giá đỡ và các hệ toạ độ gắn với từng khâu động. Ký hiệu các hệ toạ độ này từ 0 đến n, kể từ giá cố định trở đi.

Một điểm bất kì nào đó trong không gian được xác định trong hệ toạ độ thứ i bằng bán kính r_i và trong hệ toạ độ cố định x_0, y_0, z_0 được xác định bằng bán kính vector r_0 :

$$r_0 = A_1A_2\ldots A_ir_i \qquad (2.2)$$

hoặc $$r_0 = T_ir_i \qquad (2.3)$$

với $$T_i = A_1A_2\ldots A_i \text{ , } i= 1, 2, \ldots n \qquad (2.4)$$

Trong đó ma trận A_1 mô tả vị trí hướng của khâu đầu tiên; ma trận A_2 mô tả vị trí và hướng của khâu thứ 2 so với khâu đầu; ma trận A_i mô tả vị trí và hướng của khâu thứ i so với khâu thứ i-1.

Như vậy, tích của các ma trận A_i là ma trận T_i mô tả vị trí và hướng của khâu thứ i so với giá trị cố định. Thường kí hiệu ma trận T với 2 chỉ số: trên và dưới. Chỉ số dưới chỉ khâu đang xét còn chỉ số trên để chỉ toạ độ được dùng để đối chiếu. Ví dụ, biểu thức (2.4) có thể viết lại là :

$$T_i = {}^0T_i = A_1\,{}^1T_i \qquad (2.5)$$

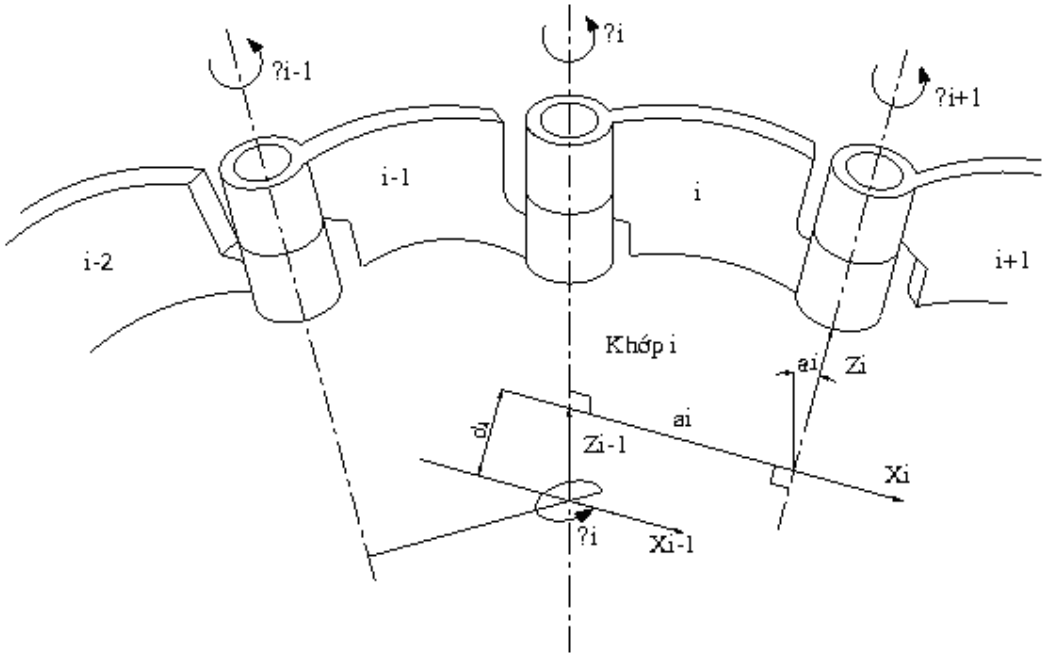
với $$ {}^1T_i = A_2A_3\ldots A_i \qquad (2.6)$$

là ma trận mô tả vị trí và hướng của khâu thứ i so với khâu thứ nhất. Trong kí hiệu thường bỏ qua chỉ số trên nếu chỉ số đó bằng 0.

Denavit & Hartenberg đã đề xuất dùng ma trận thuần nhất 4x4 mô tả quan hệ giữa 2 khâu liên tiếp trong cơ cấu không gian .

II.1.3.2. Bộ thông số DH :

Dưới đây trình bày cách xây dựng các hệ toạ động đối với 2 khâu động liên tiếp i và i+1. Hình dưới đây là trường hợp 2 khớp động liên tiếp là 2 khớp quay.



Hình 2.1: ***Các hệ toạ độ đối với 2 khâu động liên tiếp***

Trước hết xác định bộ thông số cơ bản giữa 2 trục quay của khớp động i+1 và i :

a_i là độ dài đường vuông góc chung giữa 2 trục khớp động i+1 và i .

α_i là góc chéo giữa 2 trục khớp động i+1 và i .

d_i là khoảng cách đo dọc trục khớp động i từ đường vuông góc chung

giữa trục khớp động i+1 và trục khớp động i tới đường vuông góc chung giữa khớp động i và trục khớp động i -1.

θ_i là góc giữa 2 đường vuông góc chung nói trên.

Bộ thông số này được gọi là bộ thông số Denavit – Hartenberg (DH).

Biến khớp (joint variable):

Nếu khớp động i là khớp quay thì biến khớp là θ_i

Nếu khớp động i là khớp tịnh tiến thì biến khớp là d_i

Để kí hiệu thêm biến khớp dùng thêm dấu * và trong trường hợp khớp tịnh tiến thì a_i được xem là bằng 0.

II.1.3.3. Thiết lập hệ toạ độ :

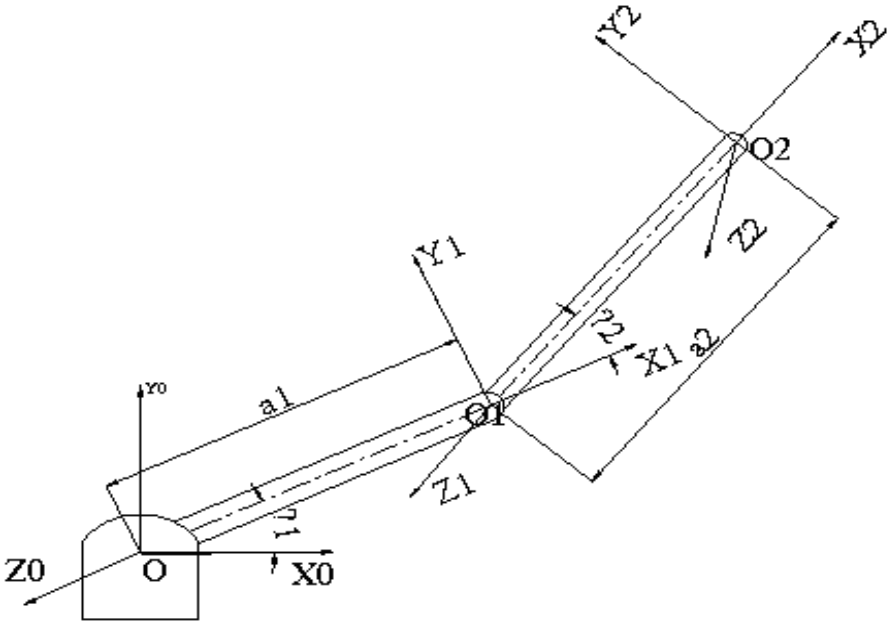
Gốc của hệ toạ độ gắn liền với khâu thứ i (gọi là hệ toạ độ thứ i) đặt tại giao điểm giữa đường vuông góc chung (a_i) và trục khớp động i+1.

Trường hợp 2 trục giao nhau thì gốc hệ toạ độ lấy trùng với giao điểm đó. Nếu 2 trục song song với nhau thì chọn gốc toạ độ là điểm bất kì trên trục khớp động i+1.

Trục z_i của hệ toạ độ thứ i nằm dọc theo trục khớp động i+1.

Trục x_i của hệ toạ độ thứ i nằm dọc theo đường vuông góc chung hướng từ khớp động i đến khớp động i+1. Trường hợp 2 trục giao nhau, hướng trục x_i trùng với hướng vector tích $z_i \times z_{i-1}$, tức là vuông góc với mặt phẳng chứa z_i, z_{i-1}.

Ví dụ : Xét tay máy có 2 khâu phẳng như hình 2.2.



Hình 2.2: ***Tay máy 2 khâu phẳng (vị trí bất kỳ)***

Gắn các hệ toạ độ với các khâu như hình vẽ :

- Trục z_0 , z_1 và z_2 vuông góc với mặt tờ giấy.

- Hệ toạ độ cố định là $o_0x_0y_0z_0$ chiều x_0 hướng từ o_0 đến o_1.

- Hệ toạ độ $o_1x_1y_1z_1$ có gốc o_1 đặt tại tâm trục khớp động 2.

- Hệ toạ độ $o_2x_2y_2z_2$ có gốc o_2 đặt tại tâm trục khớp động cuối khâu 2.

Bảng thông số DH của tay máy này như sau :

Khâu	θ_i	α_i	a_i	d_i
1	θ_1^*	0	a_1	0
2	θ_2^*	0	a_2	0

II.1.3.4. Mô hình biến đổi :

Trên cơ sở đã xây dựng các hệ toạ độ với 2 khâu động liên tiếp như trên đã trình bày. Có thể thiết lập mối quan hệ giữa 2 hệ toạ độ liên tiếp theo 4 phép biến đổi :

+ Quay quanh trục z_{1-1} góc θ_i .

+ Tịnh tiến dọc trục z_{i-1} một đoạn d_i .

+ Tịnh tiến dọc trục x_{i-1} (đã trùng với x_i) một đoạn a_i .

+ Quay quanh trục x_i một góc α_i .

Bốn phép biến đổi này được biểu thị bằng tích các ma trận thuần nhất sau

$$A_i = R(z,\theta_i).T_p(0,0,d_i).T_p(a_i,0,0).R(x,\alpha_i) \qquad (2.7)$$

Các ma trận ở vế phải phương trình (2.7) tính theo các công thức (1.27),(1.29),(1.31). Sau khi thực hiện phép nhân các ma trận nói trên, ta có:

$$A_i = \begin{bmatrix} C_{\theta i} & -S_{\theta i}C_{\alpha i} & S_{\theta i}S_{\alpha i} & a_i C_{\theta i} \\ S_{\theta i} & C_{\theta i}C_{\alpha i} & -C_{\theta i}S_{\alpha i} & a_i S_{\theta i} \\ 0 & S_{\alpha i} & C_{\alpha i} & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (2.8)$$

Trong khớp tịnh tiến : a = 0 .

II.1.3.5. Phương trình động học :

Với Robot có n khâu, ma trận mô tả vị trí và hướng điểm cuối E của tay máy được miêu tả :

$$T_n = A_1 A_2 \ldots A_n \qquad (2.9)$$

Mặt khác, hệ toạ độ tại "điểm tác động cuối" này được mô tả bằng ma trận T_E. Vì vậy hiển nhiên là:

$$T_E = T_n \qquad (2.10)$$

Tức là ta có :

$$T_n = \begin{bmatrix} n_x & s_x & a_x & p_x \\ n_y & n_y & a_y & p_y \\ n_z & s_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (2.11)$$

Phương trình (2.11) là phương trình động học cơ bản của Robot.

II.2. Tổng hợp chuyển động Robot :

II.2.1. Nhiệm vụ :

Nhiệm vụ tổng hợp chuyển động bao gồm việc xác định các bộ lời giải $q_i(t)$, (i = 1,..., n), với q_i là toạ độ suy rộng hoặc là biến khớp.

Biết quy luật chuyển động của bàn kẹp, cần xác định quy luật thay đổi các biến khớp tương ứng. Đó là nội dung chính của việc tổng hợp quỹ đạo chuyển động Robot.

Có thể xem quỹ đạo chuyển động là tập hợp liên tiếp các vị trí khác nhau của bàn kẹp. Tại mỗi vị trí trên quỹ đạo cần xác định bộ thông số các biến khớp q_i. Đó là nội dung của bài toán động học ngược (inverse kinematics problem) của Robot.

II.2.2. Bài toán động học ngược :

Bài toán động học ngược được đặc biệt quan tâm vì lời giải của nó là cơ sở chủ yếu để xây dựng chương trình điều khiển chuyển động của Robot bám theo quỹ đạo cho trước.

Xuất phát từ phương trình động học cơ bản (2.11) ta có :

$$T_n = A_1A_2\ldots A_n = \begin{bmatrix} n_x & s_x & a_x & p_x \\ n_y & n_y & a_y & p_y \\ n_z & s_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (2.12)$$

Các ma trận A_i là hàm của các biến khớp q_i. Vector định vị bàn kẹp $\mathbf{p} = (p_x,p_y,p_z)^T$ cũng là hàm của q_i. Các vector **n, s, a** là các vector đơn vị chỉ phương các trục của hệ toạ độ gắn liền với bàn kẹp biểu diễn trong hệ toạ độ cố định XYZ. Các vector này vuông góc với nhau từng đôi một nên trong 9 thành phần của chúng chỉ tồn tại độc lập chỉ có 3 thành phần. Hai ma trận ở vế phải và vế trái của phương trình (2.12) đều là các ma trận thuần nhất 4x4. So sánh các phần tử tương ứng của 2 ma trận trên ta có 6 phương trình độc lập với các ẩn q_i (i = 1, 2,...,n).

II.2.3. Các phương pháp giải bài toán động học ngược :

Trường hợp tổng quát ta xét hệ phương trình động học của Robot có n bậc tự do.

Vế trái của phương trình (2.12) theo các kí hiệu như (2.4)-(2.6) có thể viết lại như sau:

$$T_n = T_i \cdot {}^{i}T_n \tag{2.13}$$

Nhân 2 vế của (2.13) với T_i^{-1} ta có:

$$A_i^{-1}...A_2^{-1}A_1^{-1}\,T_n = {}^{i}T_n \tag{2.14}$$

Kết hợp (2.12) ta có:

$${}^{i}T_n = A_i^{-1}...A_2^{-1}A_1^{-1}\begin{bmatrix} n_x & s_x & a_x & p_x \\ n_y & s_y & a_y & p_y \\ n_z & s_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.15}$$

với i=1,...,n-1

Ứng với mỗi giá trị của i, khi so sánh các phần tử tương ứng của 2 ma trận ở biểu thức (2.15) ta có 6 phương trình tồn tại độc lập để xác định biến khớp q_i.

II.3. Động lực học Robot:

II.3.1.Nhiệm vụ và phương pháp phân tích Động lực học Robot:

Nghiên cứu Động lực học Robot là giai đoạn cần thiết trong việc phân tích cũng như tổng hợp quá trình điều khiển chuyển động. Trong nghiên cứu Động lực học Robot thường giải quyết 2 nhiệm vụ sau đây :

+ Nhiệm vụ thứ nhất là xác định momen và lực động xuất hiện trong quá trình chuyển động. Khi đó quy luật biến đổi của biến khớp $q_i(t)$ xem như đã biết.

+ Nhiệm vụ thứ hai là xác định các sai số động. Lúc này phải khảo sát các phương trình chuyển động của cơ cấu tay máy đồng thời xem xét các đặc tính động lực của động cơ truyền động.

Có nhiều phương pháp nghiên cứu Động lực học Robot nhưng thường dùng hơn cả là phương pháp Lagrange bậc 2 vì khi kết hợp với mô hình Động lực học kiểu DH (Denavit-Hartenberg) ta sẽ được các phương trình Động lực học ở dạng vector ma trận, rất gọn nhẹ và thuận tiện cho việc nghiên cứu giải tích và tính toán trên máy tính.

Các phương trình Động lực học Robot được thiết lập dựa trên cơ sở phương trình Lagrange bậc 2:

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} = F_{Mi} \qquad ,i=1,...,n \qquad (2.16)$$

Trong đó :

L- hàm Lagrange $L = K - P$ (2.17)

K, P- động năng và thế năng của cơ hệ.

F_{Mi} - động lực, hình thành trong khớp động thứ i khi thực hiện chuyển động.

q_i - biến khớp (toạ độ suy rộng)

$\dot{q}_i$ - đạo hàm bậc nhất của biến khớp theo thời gian.

Đồng thời khi mô tả vị trí giữa 2 hệ toạ độ thứ i và i-1 dùng ma trận thuần nhất A_i hoặc viết đầy đủ hơn là ${}^{i-1}A_i$. Dùng ma trận này có thể mô tả vị trí trạng thái trong hệ toạ độ thứ i-1 của một điểm bất kì thuộc hệ toạ độ thứ i.

Các biến khớp q_i là bộ các thông số dịch chuyển của các khớp động của Robot. Vị trí trạng thái của điểm tác động cuối của Robot hoàn toàn được xác định bởi bộ biến khớp q_i này.

II.3.2.Vận tốc và gia tốc:

Để xây dựng mô hình Động lực học Robot dùng phương trình Lagrange bậc 2, cần biết vận tốc của điểm bất kì trên tay máy.

Điểm M nào đó trong hệ toạ độ i, xác định bằng véc tơ mở rộng ${}^{i}r_i$:

$${}^{i}r_i = (x_i, y_i, z_i, 1)^T, \qquad (2.18)$$

Kí hiệu ${}^{i}r_i$ có nghĩa là điểm M cho biết trong hệ toạ độ i và được biểu thị cũng trong hệ toạ độ i. Còn khi dùng kí hiệu ${}^{0}r_i$ thì có nghĩa là điểm M cho biết trong hệ toạ độ i, nhưng được biểu thị trong hệ toạ độ x_0, y_0, z_0, tức là trong hệ toạ độ cơ bản.

Như trước đây, dùng ma trận ${}^{i-1}A_i$ để mô tả vị trí tương đối giữa hệ toạ độ thứ i đối với hệ toạ độ i-1 và ma trận ${}^{0}A_i$ để mô tả quan hệ giữa hệ toạ độ thứ i và hệ toạ độ cơ bản.

Vậy quan hệ giữa ${}^{0}r_i$ và ${}^{i-1}r_i$ có thể biểu thị như sau :

$$ {}^{0}r_i = {}^{0}A_i \ {}^{i}r_i \tag{2.19} $$

với $$ {}^{0}A_i = {}^{0}A_1 \ {}^{1}A_2 \dots {}^{i-1}A_i \tag{2.20} $$

Ma trận ${}^{i-1}A_i$ đã có từ biểu thức (2.8):

$$ {}^{i-1}A_i = \begin{bmatrix} C_{\theta i} & -C_{\alpha i}S_{\theta i} & \mathrm{S}_{\alpha i}\mathrm{S}_{\theta i} & \mathrm{a}_i C_{\theta i} \\ S_{\theta i} & C_{\alpha i}C_{\theta i} & -\mathrm{S}_{\alpha i}C_{\theta i} & \mathrm{a}_i S_{\theta i} \\ 0 & \mathrm{S}_{\alpha i} & C_{\alpha i} & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.21} $$

Biểu thức (2.21) là viết cho trường hợp khớp quay i, còn nếu khớp động là khớp tịnh tiến thì a_i =0 và từ (2.21) ta có :

$$ {}^{i-1}A_i = \begin{bmatrix} C_{\theta i} & -C_{\alpha i}S_{\theta i} & \mathrm{S}_{\alpha i}\mathrm{S}_{\theta i} & 0 \\ S_{\theta i} & C_{\alpha i}C_{\theta i} & -\mathrm{S}_{\alpha i}C_{\theta i} & 0 \\ 0 & \mathrm{S}_{\alpha i} & C_{\alpha i} & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.22} $$

Đối với khớp quay thì θ_i là biến khớp và đối với khớp tịnh tiến thì d_i là biến khớp.

Các phần tử khác không của ma trận ${}^{0}A_i$ đều là hàm của θ_j, d_j, α_j và a_j (j = 1 ,2 ,…, i). Trong đó α_i, α_j lại là thông số xác định bằng cấu trúc cụ thể của tay máy. Do vậy các phần tử này là hàm của biến khớp q_i nói chung ($q_i \equiv \theta_j$ đối với khớp quay và $q_i \equiv d_i$ đối với khớp tịnh tiến).

Vi phân biểu thức (2.19) với lưu ý rằng các vectơ ${}^{i}r_i$ là không đổi với hệ toạ độ thứ i vì giả thiết rằng các khâu của tay máy là vật rắn tuyệt đối, ta có:

$$ {}^{0}V_i \equiv \mathrm{V_i} = \frac{d}{dt}({}^{0}r_i) = \frac{d}{dt}({}^{0}A_i \, {}^{i}r_i) = $$

$$= {}^0\dot{A}_1{}^1A_2\ldots{}^{i-1}A_i{}^ir_i + {}^0A_1{}^1\dot{A}_2\ldots{}^{i-1}A_i{}^ir_i + \ldots + {}^0A_1\ldots\overset{i-1}{\dot{A}_i}{}^ir_i + {}^0A_i{}^i\dot{r}_i$$

$${}^0V_i = \left[\sum_{j=1}^{i}\frac{\partial^0 A_i}{\partial q_j}\right]{}^ir_i \qquad (2.23)$$

Đạo hàm của ma trận ${}^{i-1}A_i$ đối với biến khớp q_i có thể dễ dàng xác định theo công thức sau :

$$\frac{\mathrm{d}^{i-1}A_i}{dq_i} = D_i^{i-1}A_i \qquad (2.24)$$

Trong đó đối với khớp quay :

$$D_i = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \qquad (2.25)$$

và đối với khớp tịnh tiến :

$$D_i = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \qquad (2.26)$$

Trong trường hợp i = 1, 2,…,n ta có :

$$\frac{\partial^0 A_i}{\partial q_j} = \frac{\partial}{\partial q_j}(A_1A_2\ldots A_{j-1}A_j\ldots A_{i-1}A_i) \qquad (2.27)$$

Trong các ma trận bên vế phải chỉ có A_j phụ thuộc vào q_j, do đó theo (2.24) ta có :

$$\frac{\partial^0 A_i}{\partial q_j} = A_1A_2\ldots A_{j-1}\frac{dA_j}{dq_j}\ldots A_{i-1}A_i \qquad (2.28)$$

với D_j tính theo (2.25) hoặc (2.26) :

$$\frac{\partial^0 A_i}{\partial q_j} = A_1A_2\ldots A_{j-1}D_j\ldots A_{i-1}A_i \qquad (2.29a)$$

Phương trình (2.29a) mô tả sự thay đổi vị trí các điểm của khâu thứ i gây nên bởi sự dịch chuyển của khớp động thứ j .

Kí hiệu vế trái của (2.29a) là U_{ij} và đơn giản hoá cách viết (2.29a) như sau :

$$U_{ij} = \begin{cases} {}^{0}A_{j-1} D_j^{j-1} A_i, j \le i \\ 0, j > i \end{cases} \qquad (2.29b)$$

vậy công thức (2.23) có thể viết lại là:

$$V_i = \left[\sum_{j=1}^{i} U_{ij} \dot{q}_j \right] {}^{i}r_i \qquad (2.30)$$

Tiếp theo, từ biểu thức (2.23) xác định gia tốc:

$$a = \frac{dV_i}{dt} = \left[\sum_{s=1}^{i} \frac{\partial^0 A_i}{\partial q_s} \ddot{q}_s + \sum_{s=1}^{i} \sum_{k=1}^{i} \frac{\partial^2 A_i}{\partial q_s \partial q_k} \dot{q}_s \dot{q}_k \right] {}^{i}r_i \qquad (2.31)$$

II.3.3. Động năng tay máy:

Kí hiệu K_i là động năng của khâu i (i =1,2, …, n) và dK_i là động năng của một chất điểm khối lượng dm thuộc khâu i:

$$dK_i = \frac{1}{2}(\dot{x}_1^2 + \dot{y}_1^2 + \dot{z}_1^2) dm = \frac{1}{2} Tr(V_i V_i^T) dm \qquad (2.32)$$

Trong đó Tr là vết của ma trận :

$$TrA = \sum_{i=1}^{n} a_{ii}$$

Ta có :

$$dK_i = \frac{1}{2} Tr \left[\sum_{p=1}^{i} U_{ip} \dot{q}^{i}{}_{p} r_i \left[\sum_{r=1}^{i} U_{ir} \dot{q}_r {}^{1} r_i \right]^T \right] dm$$

$$= \frac{1}{2} Tr \left[\sum_{p=1}^{i} \sum_{r=1}^{i} U_{ip}^{i} r_i^1 r_i^T U_{ir}^T \dot{q}_p \dot{q}_r \right] dm$$

$$= \frac{1}{2}\mathrm{Tr}\left[\sum_{p=1}^{i}\sum_{r=1}^{i} U_{ip}\,({}^{i}r_i\,dm\,{}^{i}r^{T}{}_{i})\,U_{ir}^{T}\,\dot{q}_p\,\dot{q}_r\right] \qquad (2.33)$$

Như đã nói ở trên ma trận U_{ij} biểu thị sự thay đổi vị trí của các điểm thuộc khâu i gây nên bởi sự dịch chuyển của khớp động j. Ma trận này đều như nhau tại mọi thời điểm thuộc khâu i và không phụ thuộc vào sự phân bố khối lượng trên khâu i, tức là không phụ thuộc vào dm. Cũng vậy, đạo hàm của biến khớp q_i theo thời gian không phụ thuộc vào dm. Do vậy ta có:

$$K_i \int dK_i = \frac{1}{2}\mathrm{Tr}\left[\sum_{p=1}^{i}\sum_{r=1}^{i} U_{ip}\left(\int {}^{i}r_i\,{}^{i}r_i^{T}\,dm\right)U_{ir}^{T}\,\dot{q}_p\,\dot{q}_r\right] \qquad (2.34)$$

Phần trong ngoặc đơn là ma trận quán tính J_i của khâu i:

$$J_i = \int {}^{i}r_i^{i}r_i^{T}\,dm = \begin{bmatrix} \int x_i^2\,dm & \int x_i y_i\,dm & \int x_i z_i\,dm & \int x_i\,dm \\ \int x_i y_i\,dm & \int y_i^2\,dm & \int z_i y_i\,dm & \int y_i\,dm \\ \int x_i z_i\,dm & \int z_i y_i\,dm & \int z_i^2\,dm & \int z_i\,dm \\ \int x_i\,dm & \int y_i\,dm & \int z_i\,dm & \int dm \end{bmatrix}. \qquad (2.35)$$

Nếu dùng Tenso quán tính I_{ij}:

$$I_{ij} = \int\left[\delta_{ij}\left[\sum_k x_k^2 - x_i x_j\right]\right]dm, \qquad (2.36)$$

Với các chỉ số i, j, k lấy lần lượt bằng các giá trị x_i , y_i, z_i, đó là các trục của hệ toạ độ i, và δ_{ij} là kí hiệu Cronecke, thì ma trận J_i có thể biểu thị ở dạng sau:

$$J_i = \begin{bmatrix} \frac{-I_{xx}+I_{yy}+I_{zz}}{2} & I_{xy} & I_{xz} & m_i\bar{x}_i \\ I_{xy} & \frac{I_{xx}-I_{yy}+I_{zz}}{2} & I_{yz} & m_i\bar{y}_i \\ I_{xz} & I_{yz} & \frac{-I_{xx}+I_{yy}+I_{zz}}{2} & m_i\bar{z}_i \\ m_i\bar{x}_i & m_i\bar{y}_i & m_i\bar{z}_i & m_i \end{bmatrix} \qquad (2.37)$$

Ở đây ${}^{1}\bar{r}_i = \left(\bar{x}_i, \bar{y}_i, \bar{z}_i, 1\right)^T$ - bán kính vector biểu diễn trọng tâm của khâu thứ i trong hệ toạ độ i. Công thức (2.37) viết thành :

$$J_i = \begin{bmatrix} \frac{-k_{i11}^2+k_{i22}^2+k_{i33}^2}{2} & k_{i12}^2 & k_{i13}^2 & \bar{x}_i \\ k_{i12}^2 & \frac{k_{i11}^2-k_{i22}^2+k_{i33}^2}{2} & k_{i23}^2 & \bar{y}_i \\ k_{i13}^2 & k_{i23}^2 & \frac{k_{i11}^2+k_{i22}^2-k_{i33}^2}{2} & \bar{z}_i \\ \bar{x}_i & \bar{y}_i & \bar{z}_i & 1 \end{bmatrix} \qquad (2.38)$$

Ở đây $K_{ijk} = \frac{I_{jk}}{m_i}$ và j = 1,2,3 ; k = 1, 2, 3

${}^{i}\bar{r}_i = \left(\bar{x}_i, \bar{y}_i, \bar{z}_i, 1\right)^T$ - bán kính véc tơ biểu diễn trọng tâm của khâu thứ i trong hệ toạ độ i .

Vậy, động năng của toàn cơ cấu tay máy bằng tổng đại số động năng của các khâu động :

$$K = \sum_{i=1}^{n} K_i = \frac{1}{2}\sum_{i=1}^{n} Tr\left[\sum_{p=1}^{i}\sum_{r=1}^{i} U_{ip} J_i U_{ir}^T \dot{q}_p \dot{q}_r\right] = \frac{1}{2}\sum_{i=1}^{n}\sum_{p=1}^{i}\sum_{r=1}^{i}\left[Tr\left(U_{ip} J_i U_{ir}^T\right)\dot{q}_p \dot{q}_r\right] \qquad (2.39)$$

Lưu ý rằng, các ma trận J_i (i=1,2,3,…n) chỉ phụ thuộc vào sự phân bố khối lượng của khâu i trong hệ toạ độ i mà không phụ thuộc vào vị trí và vận tốc. Vì thế cần tính ma trận J_i chỉ 1 lần.

II.3.4 .Thế năng tay máy:

Thế năng P_i của khâu i:

$$P_i = - m_i.g.{}^{0}r_i = -m_i.g.\left({}^{0}A_i\, {}^{1}r_i\right) \qquad (2.40)$$

i=1,2,…,n

Trong đó

${}^{i}r_i$, ${}^{0}r_i$ - bán kính vec tơ biểu diễn trọng tâm của khâu i trong hệ toạ độ cơ bản .

g - vector gia tốc trọng trường, g = (0, 0 , -g, 0)

(gia tốc trọng trường g = 9,8062 m/s^2)

Thế năng của toàn cơ cấu n khâu động :

$$P = \sum_{i=1}^{n} P_i = -\sum_{i=1}^{n} m_i g \left({}^{0}A_i \, {}^{i}\bar{r}_i \right) . \qquad (2.41)$$

II.3.5.Mô hình động lực học tay máy:

Để xây dựng mô hình động lực học tay máy dùng phương trình Lagrange bậc 2 (phương trình 2.16):

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}}\right) - \frac{\partial L}{\partial q_i} = F_{Mi} \text{ , I = 1,2, …,n}$$

Phương trình trên cho biểu thức tính động lực F_{Mi}. Đó là lực hoặc mô men tạo nên bởi nguồn động lực ở khớp động i để thực hiện chuyển động khâu i.

Thay (2.39), (2.41) vào (2.16), cuối cùng ta có :

$$F_{Mi} = \sum_{j=1}^{n}\sum_{k=1}^{j} T_r\left(U_{jk} J_j U_{ji}^{T}\right)\ddot{q}_k + \sum_{j=1}^{n}\sum_{k=1}^{j}\sum_{m=1}^{j} Tr\left(U_{jkm} J_j U_{ji}^{T}\right)\dot{q}_k \dot{q}_m - \sum_{j=1}^{n} m_j g U_{ji} \, {}^{j}\bar{r}_j$$

i=1,2,…,n (2.42)

Biểu thức (2.42) có thể viết gọn lại như sau :

$$F_{Mi} = \sum_{j=1}^{n} D_{ik}\ddot{q}_k + \sum_{k=1}^{n}\sum_{m=1}^{n} h_{ikm}\dot{q}_k\dot{q}_m + c_i \qquad (2.43)$$

Hoặc là dưới dạng ma trận :

$$F_{Mi} = D(q)\ddot{q} + h(q,\dot{q}) + c(q) \qquad (2.44)$$

Trong đó :

$F_M(t)$ – vector (nx1) lực động, tạo nên ở n khớp động :

$$F_M(t) = [F_{M1}(t), F_{M2}(t), ..., F_{Mn}(t)]^T \quad (2.45)$$

q(t) – vector (nx1) biến khớp :

$$q(t) = [q_1(t), q_2(t), ..., q_n(t)]^T \quad (2.46)$$

$\dot{q}(t)$ - vec tơ (nx1) tốc độ thay đổi biến khớp :

$$\dot{q}(t) = [\dot{q}_1(t), \dot{q}_2(t), ..., \dot{q}_n(t)]^T \quad (2.47)$$

$\ddot{q}(t)$ - vectơ (nx1) gia tốc biến khớp :

$$\ddot{q}(t) = [\ddot{q}_1(t), \ddot{q}_2(t), ..., \ddot{q}_n(t)]^T \quad (2.48)$$

D(q) – Ma trận (nxn) , có các phần tử D_{ik} sau đây :

$$D_{ik} = \sum_{j=\max(i,k)}^{n} Tr\left(U_{jk} J_j U_{ji}^T\right) \qquad i,k = 1,2, \ldots. n \quad (2.49)$$

H(q, $\dot{q}$) – vectơ (nx1) lực ly tâm và Coriolit

$$h(q,\dot{q}) = (h_1, h_2, ..., h_n)^T$$

$$h_i = \sum_{k=1}^{n} \sum_{m=1}^{n} h_{ikm} \dot{q}_k \dot{q}_m \qquad i=1,2,\ldots,n \quad (2.50)$$

$$h_{ikm} = \sum_{j=\max(i,k,m)}^{n} Tr\left(U_{ikm} J_j U_{ji}^T\right) \qquad i,k,m = 1,2,\ldots,n \quad (2.51)$$

C(q) – vec tơ (nx1) lực trọng trường.

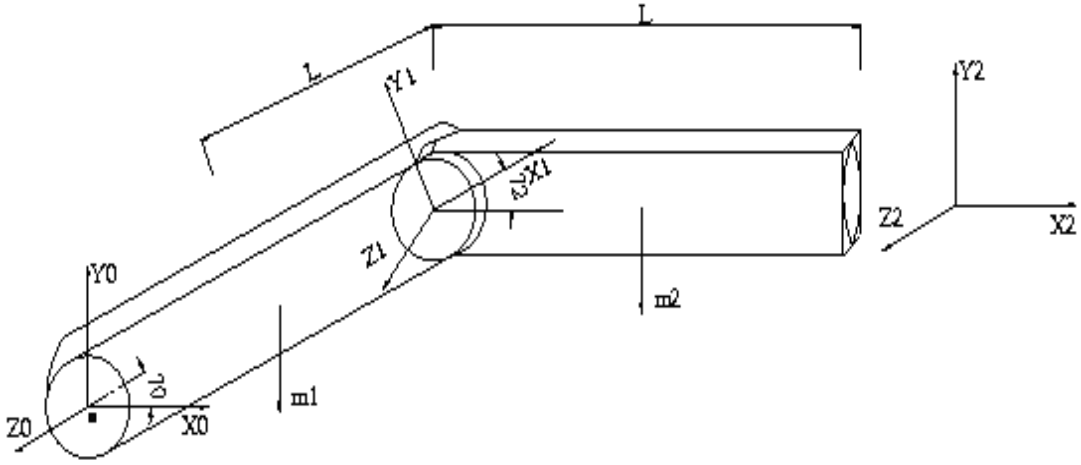
$$c(q) = (c_1, c_2, ..., c_n)^T \quad c_i = \sum_{j=1}^{n} \left(-m_i g U_{ji}\, {}^j\bar{r}_j \right) \quad (2.52)$$

II.3.6. Động lực học của cơ cấu tay máy 2 khâu:

Trong phần này dẫn ra ví dụ minh họa xây dựng mô hình động lực học của cơ cấu tay máy 2 khâu toàn khớp (hình 2.3) .

Như đã chỉ trên hình 2.3 các trục Z_i đều trùng phương với các trục khớp quay động. Khối lượng của các khâu tương ứng là m_1, m_2; Bộ thông số DH của tay máy ghi trong bảng sau :

Khâu	θ_i	α_i	A_i	d_i	Khớp
1	θ_i^*	0	1	0	R
2	θ_i^*	0	1	0	R



Hình 2.3: ***Cơ cấu tay máy 2 khâu***

Các ma trận ${}^{i-1}A_i$ (i =1,2) sẽ là :

$${}^0A_1 = \begin{bmatrix} C_1 & S_1 & 0 & lC_1 \\ S_1 & C_1 & 0 & lS_1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad {}^1A_2 = \begin{bmatrix} C_2 & -S_2 & 0 & lC_2 \\ S_2 & C_2 & 0 & lS_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$${}^0A_2 = {}^0A_2\,{}^1A = \begin{bmatrix} C_{12} & -S_{12} & 0 & l(C_{12}+C_1) \\ S_{12} & C_{12} & 0 & l(S_{12}+S_1) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

vẫn như trước đây dùng các kí hiệu sau :

$C_i = \cos\theta_i$; $S_i = \sin\theta_i$; $C_{ij} = \cos(\theta_i+\theta_j)$; $S_{ij} = \sin(\theta_i+\theta_j)$.

Theo (2.30), ta có :

$$U_{11}=\frac{\partial^0 A_1}{\partial\theta_1}=D_1{}^0A_1=\begin{bmatrix}0&-1&0&0\\1&0&0&0\\0&0&0&0\\0&0&0&0\end{bmatrix}\begin{bmatrix}C_1&-S_1&0&lC_1\\S_1&C_1&0&lS_1\\0&0&1&0\\0&0&0&1\end{bmatrix}=\begin{bmatrix}-S_1&-C_1&0&-lS_1\\C_1&-S_1&0&+lC_1\\0&0&0&0\\0&0&0&0\end{bmatrix}$$

Tương tự, đối với U_{12}, và U_{22} :

$$U_{21}=\frac{\partial^0 A_2}{\partial\theta_1}=D_2{}^0A_2=\begin{bmatrix}0&-1&0&0\\1&0&0&0\\0&0&0&0\\0&0&0&0\end{bmatrix}\begin{bmatrix}C_{12}&-S_{12}&0&l(C_{12}+C_1)\\S_{12}&C_{12}&0&l(S_{12}+S_1)\\0&0&1&0\\0&0&0&0\end{bmatrix}=\begin{bmatrix}-S_{12}&-C_{12}&0&-l(S_{12}+S_1)\\C_{12}&-S_{12}&0&l(C_{12}+C_1)\\0&0&0&0\\0&0&0&0\end{bmatrix}$$

$$U_{22}=\frac{\partial^0 A_2}{\partial\theta_2}={}^0A_1D_2{}^1A_2=\begin{bmatrix}C_2&-S_1&0&lC_1\\S_1&C_1&0&lS_1\\0&0&1&0\\0&0&0&1\end{bmatrix}\begin{bmatrix}0&-1&0&0\\1&0&0&0\\0&0&0&0\\0&0&0&0\end{bmatrix}\begin{bmatrix}C_2&-S_2&0&lC_2\\S_2&C_2&0&lS_2\\0&0&1&0\\0&0&0&0\end{bmatrix}=\begin{bmatrix}-S_{12}&-C_{12}&0&-lS_{12}\\C_{12}&-S_{12}&0&lC_{12}\\0&0&0&0\\0&0&0&0\end{bmatrix}$$

Theo (2.37) và giả thiết cả các thành phần mômen ly tâm quán tính đều bằng 0 , ta có :

$$J_1=\begin{bmatrix}1/3m_1l^2&0&0&-1/2m_1l^2\\0&0&0&0\\0&0&0&0\\-1/2m_1l^2&0&0&m_1\end{bmatrix},\quad J_2=\begin{bmatrix}1/3m_2l^2&0&0&-1/2m_2l^2\\0&0&0&0\\0&0&0&0\\-1/2m_2l^2&0&0&m_2\end{bmatrix}.$$

Trên cơ sở (2.49), ta có :

$$D_{11}=T_2\left(U_{11}J_1U_{11}^T\right)+T_2\left(U_{21}J_2U_{21}^T\right)=T_2\left\{\begin{bmatrix}-S_1&-C_1&0&-lS_1\\C_1&-S_1&0&lC_1\\0&0&0&0\\0&0&0&0\end{bmatrix}\begin{bmatrix}1/3m_1l^2&0&0&-1/2m_1l^2\\0&0&0&0\\0&0&0&0\\-1/2m_1l^2&0&0&m_1\end{bmatrix}U_{11}^T\right\}+$$

$$+T_2\left\{\begin{bmatrix}-S_{12}&-C_{12}&0&-l(S_{12}+C_1)\\C_{12}&-S_{12}&0&l(C_{12}+C_1)\\0&0&0&0\\0&0&0&0\end{bmatrix}\begin{bmatrix}1/3m_2l^2&0&0&-1/2m_2l^2\\0&0&0&0\\0&0&0&0\\-1/2m_2l^2&0&0&m_2\end{bmatrix}U_{21}^T\right\}=1/3m_1l^2+4/3m_2l^2+m_2C_2l^2$$

$$D_{12} = D_{21} = Tr\left(U_{22} J_2 U_{21}^T\right) = T_2\left\{\begin{bmatrix} -S_{12} & -C_{12} & 0 & -lS_{12} \\ C_{12} & -S_{12} & 0 & lC_{12} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} 1/3m_2l^2 & 0 & 0 & -1/2m_2l^2 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -1/2m_2l^2 & 0 & 0 & m_2 \end{bmatrix} U_{21}^T\right\}$$

$$= m_2l^2(-1/6 + 1/2 + 1/2C_2) = 1/3m_2l^2 + 1/2m_2l^2 \ .$$

$$D_{22} = Tr\left(U_{22} J_2 U_{22}^T\right) = T_2\left\{\begin{bmatrix} -S_{12} & -C_{12} & 0 & -lS_{12} \\ C_{12} & -S_{12} & 0 & lC_{12} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} 1/3m_2l^2 & 0 & 0 & -1/2m_2l^2 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -1/2m_2l^2 & 0 & 0 & m_2 \end{bmatrix} U_{22}^T\right\}$$

$$= 1/3m_2l^2S_{12}^2 + 1/3m_2l^2C_{12}^2 = 1/3m_2l^2 \ .$$

Tính các số hạng trong biểu thức (2.50) đối với i = 1, ta có :

$$h_1 = \sum_{k=1}^{2}\sum_{m=1}^{2} h_{1km}\,\dot{\theta}_k\,\dot{\theta}_m = h_{111}\,\dot{\theta}_1\dot{\theta}_2 + h_{121}\,\dot{\theta}_1\dot{\theta}_2 + h_{122}\,\dot{\theta}_2^2 + h_{122}\,\dot{\theta}_2^2$$

Và theo (2.51) tính các hệ số h_{ikm}, rồi thay vào phương trình trên, ta được:

$$h_1 = -1/2m_2S_2l^2\,\dot{\theta}_2^2 \ - m_2S_2l^2\ \dot{\theta}_1\dot{\theta}_2 \ .$$

Tương tự đối với i = 2

$$h_2 = \sum_{k=1}^{2}\sum_{m=1}^{2} h_{2km}\,\dot{\theta}_k\,\dot{\theta}_m = h_{211}\,\dot{\theta}_1^2 + h_{212}\,\dot{\theta}_1\dot{\theta}_2 + h_{221}\,\dot{\theta}_2\dot{\theta}_1 + h_{222}\,\dot{\theta}_2^2 = 1/2m_2S_2l^2\,\dot{\theta}_1^2$$

Như vậy :

$$H(\theta,\dot{\theta}) = \begin{bmatrix} 1/2m_2S_2l^2\,\dot{\theta}_2^{\,2} - m_2S_2l^2\,\dot{\theta}_1\dot{\theta}_2 \\ 1/2m_2S_2l^2\,\dot{\theta}_1^{\,2} \end{bmatrix}.$$

Trên cơ sở (2.52), ta có :

$$c_1 = -\left(m_1gU_{11}^1\,\bar{r}_1 + m_2gU_{21}^2\,\bar{r}_2\right) =$$

$$= -m(0,-g,0,0)\begin{bmatrix} -S_1 & -C_1 & 0 & -lS_1 \\ C_1 & -S_1 & 0 & -lC_1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} -1/2 \\ 0 \\ 0 \\ 1 \end{bmatrix} - m_2(0,-g,0,0)\begin{bmatrix} -S_{12} & -C_{12} & 0 & -l(S_{12}+S_1) \\ C_{12} & -S_{12} & 0 & l(C_{12}+C_1) \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} -1/2 \\ 0 \\ 0 \\ 1 \end{bmatrix} =$$

$$= 1/2m_1glC_1 + 1/2m_2glC_{12} + 1/2m_2glC_1 \ ;$$

$$c_2 = -m_2(0,-g,0,0)\begin{bmatrix} -S_{12} & -C_{12} & 0 & -lS_{12} \\ C_{12} & -S_{12} & 0 & -lC_{12} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} -1/2 \\ 0 \\ 0 \\ 1 \end{bmatrix} =$$

$$= -m_2(1/2glC_{12} - glC_{12})$$

Vậy vector trọng trường sẽ là:

$$c(\theta) = \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} = \begin{bmatrix} 1/2m_1glC_1 + 1/2m_2glC_{12} + m_2glC_1 \\ 1/2m_2glC_{12} \end{bmatrix} .$$

Cuối cùng ta có phương trinh động lực học của cơ cấu tay máy 2 khâu ở dạng sau :

$$F_m(t) = D(\theta)(\ddot{\theta})(t) + h(\theta,\dot{\theta}) + c(\theta)$$

$$\begin{bmatrix} F_{M1} \\ F_{M2} \end{bmatrix} = \begin{bmatrix} 1/3m_1l^2 + 4/3m_2l + m_2C_2l & 1/3m_2l^2 + 1/2m_2l^2C_2 \\ 1/3m_2l^2 + 1/2m_2l^2C_2 & 1/3m_2l^2 \end{bmatrix} +$$

$$\begin{bmatrix} 1/2m_2S_2l^2\,\dot{\theta}_2^{\,2} - m_2S_2l^2\,\dot{\theta}_1\dot{\theta}_2 \\ 1/2m_2S_2l^2\,\dot{\theta}_1^{\,2} \end{bmatrix} + \begin{bmatrix} -1/2m_2glC_1 - 1/2m_2glC_{12} + m_2glC_1 \\ 1/2m_2gl^2C_{12} \end{bmatrix} .$$

CHƯƠNG III : THIẾT KẾ BỘ ĐIỀU KHIỂN TRƯỢT CHO TAY MÁY ROBOT 2 BẬC TỰ DO

III.1.Hệ phi tuyến :

III.1.1.Hệ phi tuyến là gì ?

Để định nghĩa được rõ ràng một đối tượng hay hệ thống như thế nào được gọi là phi tuyến trước tiên ta nên định nghĩa lại hệ tuyến tính.

Xét một hệ thống MIMO, viết tắt của nhiều vào / nhiều ra (Multi Inputs – Multi Outputs) với r tín hiệu vào $u_1(t)$, $u_2(t)$, … , $u_r(t)$ và s tín hiệu ra $y_1(t)$, $y_2(t)$, … , $y_s(t)$. Nếu viết chung r tín hiệu đầu vào thành vectơ

$$\underline{u}(t) = \begin{pmatrix} u_1(t) \\ \vdots \\ u_r(t) \end{pmatrix}$$

và s tín hiệu đầu ra thành $\underline{y}(t) = \begin{pmatrix} y_1(t) \\ \vdots \\ y_s(t) \end{pmatrix}$ thì mô hình hệ thống được quan tâm ở đây là mô hình toán học mô tả quan hệ giữa vectơ tín hiệu vào $\underline{u}(t)$ và tín hiệu ra $\underline{y}(t)$, tức là mô tả ánh xạ T : $\underline{u}(t) \mapsto \underline{y}(t)$.

Ánh xạ này (Thường còn gọi là toán tử - operator) viết lại như sau :

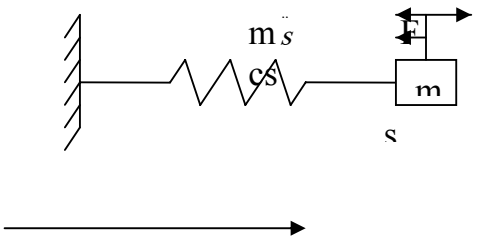
$$\underline{y}(t) = T(\underline{u}(t)).$$

nếu ánh xạ T thoả mãn :

$$T(a_1\underline{u}_1(t)) + T(a_2\underline{u}_2(t)) = a_1 T(\underline{u}_1(t)) + a_2 T(\underline{u}_2(t)) \ , \qquad (3.1)$$

trong đó a_1 và $a_2 \in R$, thì hệ đó được nói là tuyến tính. Tính chất (3.1) của hệ tuyến tính, trong điều khiển, còn được gọi là nguyên lý xếp chồng.

Ví dụ : Xét 1 hệ gồm 1 lò xo c và 1 vật khối lượng m làm 1 ví dụ. Vật sẽ chuyển động trên trục nằm ngang dưới tác động của lực F (hình 3.1).



Hình 3.1: ***Ví dụ về một đối tượng tuyến tính***

Nếu F được xem như là tín hiệu vào và quãng đường **s** mà vật đi được là tín hiệu ra (đáp ứng của hệ) thì theo các tiên đề cơ học của Newton, tác động vào vật và ngược hướng với F có hai lực cân bằng: Lực cản của lò xo F_1 =cs

trong trường hợp |s| tương đối nhỏ và $F_2 = m\ddot{s}$ của chuyển động. Với nguyên lý cân bằng lực ta có ánh xạ T : $F(t) \mapsto s(t)$ mô tả quan hệ vào / ra của hệ :

$$m\ddot{s} + cs = F. \qquad (3.2a)$$

Giả sử rằng dưới tác động của lực F_1 hệ có đáp ứng s_1 và của F_2 thì từ :

$$m\ddot{s}_1 + cs_1 = F_1$$

$$m\ddot{s}_2 + cs_2 = F_2$$

có ngay được

$$m(a_1\ddot{s}_1 + a_2\ddot{s}_2) + c(a_1s_1 + a_2s_2) = a_1F_1 + a_2F_2 ,$$

trong đó a_1 , a_2 là những số thực tuỳ ý . Nói cách khác dưới tác động của lực $a_1F_1 + a_2F_2$

vật sẽ đi được một quãng đường là $a_1s_1 + a_2s_2$. Bởi vậy T thoả mãn (3.1) và do đó trong trường hợp |s| tương đối nhỏ và lực cản của lò xo được xác định gần đúng bằng công thức F_1 = cs thì hệ thống là xo + vật là một hệ tuyến tính .

Ngược lại, nếu lực cản lò xo lại được tính theo $F_1 = cs + \varepsilon s^3$, với c và ε là 2 hằng số, mà trong thực tế người ta vẫn sử dụng, thì quan hệ vào / ra của hệ sẽ là :

$$m\ddot{s} + cs + \varepsilon s^3 = F \qquad (3.2b)$$

và khi đó (3.2b) không còn thoả mãn nguyên lý xếp chồng (3.1)

$$m(a_1\ddot{s}_1 + a_2\ddot{s}_2) + c(a_1 s_1 + a_2 s_2) + \varepsilon(a_1 s_1^3 + a_2 s_2^3) \neq a_1 F_1 + a_2 F_2$$

$$m(a_1\ddot{s}_1 + a_2\ddot{s}_2) + c(a_1 s_1 + a_2 s_2) + \varepsilon(a_1 s_1^3 + a_2 s_2^3) \neq a_1 F_1 + a_2 F_2$$

Nói cách khác, dưới tác động của lực $a_1F_1 + a_2F$ thì quãng đường của vật đi được không phải là $a_1s_1 + a_2s_2$. Vậy ở trường hợp này hệ có tính phi tuyến.

III.1.2.Mô hình trạng thái và quỹ đạo trạng thái của Hệ phi tuyến:

III.1.2.1.Mô hình trạng thái:

Mô hình động của đối tượng, hệ thống phi tuyến được xây dựng từ quan hệ vào – ra qua việc thêm các biến $x_1(t)$, $x_2(t)$, … , $x_n(t)$ gọi là biến trạng thái, sao cho quan hệ giữa vector tín hiệu ra y(t) với n biến này và tín hiệu vào u(t) chỉ còn lại thuần tuý là một quan hệ đại số. Những biến trạng thái này, về mặt ý nghĩa vật lý, là những đại lượng mà sự thay đổi của nó sẽ quyết định tính chất động học của đối tượng.

Ví dụ 1 : Từ mô hình (3.1) của đối tượng lò xo + vật, nếu thêm biến trạng thái $x_1 = s$, $x_2 = \dot{s}$ sẽ có:

$$s = (1 \quad 0)\begin{pmatrix} X_1 \\ X_2 \end{pmatrix} = (1 \quad 0).\underline{x} + 0.F$$

và đấy là một phương trình đại số. Ngoài ra còn có phần các phương trình vi phân bao gồm :

$$\dot{X}_1 = X_2$$

Suy ra từ định nghĩa về x_1 , x_2 và :

$$\dot{x}_2 = -\frac{c}{m}x_1 + \frac{1}{m}F$$

thu được bằng cách thay trực tiếp x_1, x_2 vào phương trình (3.1).

Viết chung hai phương trình vi phân trên lại với nhau sẽ được :

$$\underline{\dot{x}} = \begin{pmatrix} \dot{x}_1 \\ \dot{x}_2 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -\frac{c}{m} & 0 \end{pmatrix} \underline{x} + \begin{pmatrix} 0 \\ \frac{1}{m} \end{pmatrix} F$$

Nói chung, một hệ phi tuyến SISO có quan hệ vào – ra giữa tín hiệu vào u(t) và ra y(t) dạng:

$$y^{(n)} = f(y^{(n-1)}, ..., \dot{y}, y, u)$$

Trong đó ký hiệu $y^{(k)}$ chỉ đạo hàm bậc k của y(t), tức là

$$y^{(k)} = \frac{d^k y}{dt^k},$$

thì với các biến trạng thái được định nghĩa như sau :

$$x_1 = y, x_2 = \dot{y}, x_3 = \ddot{y}, ..., x_n = y^{(n-1)}$$

hệ sẽ có mô hình hai phần: phần các phương trình vi phân bậc nhất

$$\begin{aligned} \dot{x}_1 &= x_2 \\ &\vdots \\ \dot{x}_{n-1} &= x_n \\ \dot{x}_n &= f(x_n, ..., x_2, x_1, u) \end{aligned}$$

và phương trình đại số : $Y=x_1$

Tổng quát lên thì một hệ phi tuyến, sau khi định nghĩa các biến trạng thái $x_1(t)$, $x_2(t)$, …, $x_n(t)$, sẽ mô tả bởi :

- Mô hình trạng thái tường minh autonom

$$\begin{aligned} \underline{\dot{x}} &= \underline{f}(\underline{x}, \underline{u}) \\ \underline{y} &= \underline{g}(\underline{x}, \underline{u}) \end{aligned} \quad \text{trong đó } \underline{x}(t) = \begin{pmatrix} x_1(t) \\ \vdots \\ x_n(t) \end{pmatrix}$$

- Mô hình trạng thái tường minh không autonom

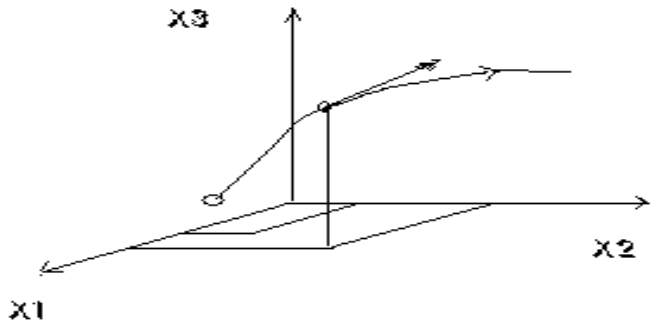
$$\begin{aligned} \dot{\underline{x}} &= \underline{f}(\underline{x}, \underline{u}, t) \\ \underline{y} &= \underline{g}(\underline{x}, \underline{u}, t) \end{aligned},$$

- hoặc mô hình trạng thái không tường minh

$$\begin{aligned} \underline{f}(\underline{x}, \dot{\underline{x}}, \underline{u}, t) &= \underline{0} \\ \underline{g}(\underline{x}, \underline{u}, \underline{y}, t) &= \underline{0} \end{aligned}$$

III.1.2.2.Quỹ đạo trạng thái :

Từ phương trình trạng thái mô tả hệ thống với một tín hiệu đầu vào $\underline{u}(t)$ xác định cho trước và với một điểm trạng thái ban đầu $\underline{x}_0 = \underline{x}(0)$ cũng cho trước ta sẽ có được nghiệm $\underline{x}(t)$ mô tả sự thay đổi trạng thái hệ thống theo thời gian dưới tác động của kích thích $\underline{u}(t)$ đã cho. Biểu diễn $\underline{x}(t)$ trong không gian n chiều R^n (còn gọi là không gian trạng thái) như một đồ thị phụ thuộc tham số t có mũi tên chỉ chiều tăng của t ta được một quỹ đạo trạng thái (hình 3.2a). Tập tất cả các quỹ đạo trạng thái ứng với một tín hiệu đầu vào $\underline{u}(t)$ cố định nhưng với những điểm trạng thái ban đầu $\underline{x}_0$ khác nhau được gọi là *họ các quỹ đạo trạng thái* (hình 3.2b) .



Hình 3.2a. ***Quỹ đạo trạmg thái của hệ có 3 biến trạng thái***

x_1

x_2

Hình 3.2b. ***Họ các quỹ đạo trạng thái của hệ có 2 biến trạng thái***

III.1.3. Điểm cân bằng và điểm dừng của hệ thống:

III.1.3.1. Điểm cân bằng:

Định nghĩa 1: Một điểm trạng thái $\underline{X}_e$ được gọi là điểm cân bằng (*equilibrium point*) nếu như khi đang ở điểm trạng thái $\underline{X}_e$ và không có một tác động nào từ bên ngoài thì hệ sẽ nằm nguyên tại đó.

Căn cứ theo định nghĩa như vậy thì điểm cân bằng $\underline{X}_e$ của hệ thống phải là nghiệm của phương trình:

$$\frac{d\underline{x}}{dt} = \underline{f}(\underline{x}, \underline{u}, t)\Big|_{\underline{u}=\underline{0}} = \underline{0}$$

Như vậy điểm cân bằng là điểm mà hệ thống sẽ nằm yên tại đó, tức là trạng thái của nó sẽ không bị thay đổi ($\frac{d\underline{x}}{dt} = \underline{0}$) khi không có sự tác đng từ bên ngoài ($\underline{u} = \underline{0}$).

III.1.3.2.Điểm dừng của hệ :

Định nghĩa 2: Một điểm trạng thái $\underline{x}_d$ được gọi là điểm dừng của hệ thống nếu như hệ đang ở điểm trạng thái $\underline{x}_d$ và với tác động $\underline{u}(t) = u_d$ cố định, không đổi cho trước, thì hệ sẽ nằm nguyên tại đó.

Rõ ràng là điểm dừng theo định nghĩa vừa nêu sẽ là nghiệm của :

$$\frac{d\underline{x}}{dt} = \underline{f}(\underline{x}, \underline{u}_d, t)\Big|_{\underline{u}=\underline{u}_d} = \underline{f}(\underline{x}, \underline{u}_d, t) = \underline{0}$$

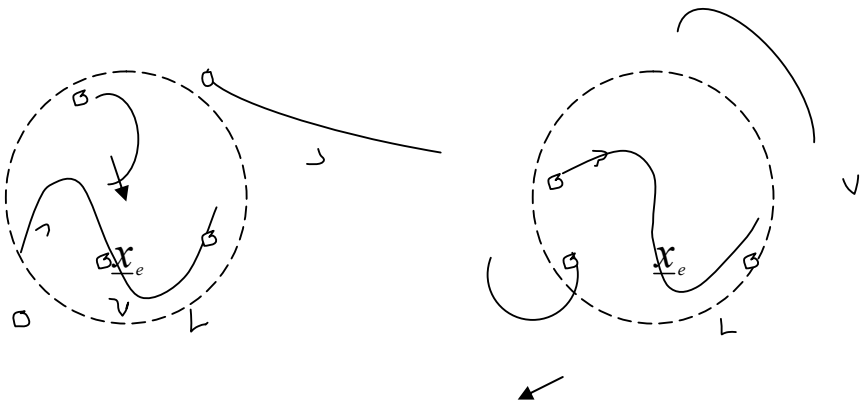
trong đó $\underline{u}_d$ là đã cho trước.

III.1.3.3 Tính ổn định tại một điểm cân bằng:

Định nghĩa 3 : Một hệ thống được gọi là ổn định (tiệm cận) tại điểm cân bằng $\underline{x}_e$ nếu như có một tác động tức thời (chẳng hạn như nhiễu tức thời) đánh bật hệ ra khỏi $\underline{x}_e$ và đưa tới điểm $\underline{x}_0$ thuộc một lân cận nào đó của $\underline{x}_e$ thì sau đó hệ có khả năng tự quay về được điểm cân bằng $\underline{x}_e$ ban đầu.

Theo định nghĩa trên thì ta có thể nhận biết được hệ có ổn định hay không tại một điểm cân bằng thông qua dạng họ các đường quỹ đạo trạng thái của nó. Nếu hệ ổn định tại một điểm cân bằng $\underline{x}_e$ nào đó thì mọi đường quỹ đạo trạng thái $\underline{x}(t)$ xuất phát từ một điểm $\underline{x}_0$ thuộc lân cận của $\underline{x}_e$ đều phải kết thúc tại $\underline{x}_e$

a) Miền ổn định O b)



Hình 3.3. ***a) Điểm cân bằng ổn định***

b) Điểm cân bằng không ổn định

Chú ý rằng tính ổn định của hệ phi tuyến chỉ có ý nghĩa khi đi cùng với điểm cân bằng $\underline{x}_e$. Có thể hệ sẽ ổn định tại điểm cân bằng này, song lại không ổn định ở điểm cân bằng khác. Điều này cũng khác so với khái niệm ổn định ở hệ tuyến tính. Vì hệ tuyến tính thường chỉ có một điểm cân bằng là gốc toạ độ ($\underline{x}_e = \underline{0}$) nên khi hệ ổn định tại $\underline{0}$, người ta cũng nói thêm luôn một cách ngắn gọn là hệ ổn định .

Ngoài ra, do khái niệm ổn định ở hệ phi tuyến bị gắn với lân cận điểm cân bằng $\underline{x}_e$ nên cũng có thệ mặc dù hệ ổn định tại điểm cân bằng $\underline{x}_e$ song với một lân cận quá nhỏ thì sẽ không có định nghĩa sử dụng. Nói cách khác, về mặt ứng dụng, nó được xem như không ổn định. Bởi vậy, đối với hệ phi tuyến, việc xác định xem hệ có ổn định tại điểm cân bằng $\underline{x}_e$ hay không là chưa đủ mà còn phải chỉ ra miền ổn định của nó tại $\underline{x}_e$, tức là phải chỉ ra được lân cận O của $\underline{x}_e$ sao cho hệ có khả năng tự quay về được $\underline{x}_e$ từ bất kì một điểm $\underline{x}_0$ nào đó thuộc O (hình 3.3). Miền ổn định O càng lớn thì tính ổn định của hệ tại $\underline{x}_e$ càng tốt .

Nhiệm vụ đầu tiên của bộ điều khiển là phải giữ cho hệ thống ổn định. Nếu như ban đầu đối tượng không ổn định, tức là khi có nhiễu từ bên ngoài tác động đưa nó ra khỏi điểm làm việc và nó không có khả năng tự quay về thì bộ điều khiển phải tạo ra tín hiệu điều khiển dẫn đối tượng quay trở về điểm làm việc ban đầu.

III.1.4 Tiêu chuẩn ổn đinh Lyapunov :

Một trong những điều kiện, hay tiêu chuẩn chất lượng đầu tiên mà bộ điều khiển cần phải mang đến được cho hệ thống là tính ổn định. Tại sao lại như vậy ? Từ khái niệm về tính ổn định của hệ thống tại một điểm cân bằng đã được nêu trong định nghĩa 3 ta thấy rõ nếu một hệ quá nhạy cảm với tác động nhiễu đến nỗi chỉ một tác động tức thời không mong muốn rất nhỏ đã làm cho hệ bị bật ra khỏi điểm cân bằng (hoặc điểm làm việc) mà sau đó hệ không có khả năng tự tìm về điểm cân bằng ban đầu thì chất lượng của hệ không thể gọi là tốt được.

Bởi vậy, kiểm tra tính ổn định của hệ (tại một điểm cân bằng) cũng như miền ổn định O tương ứng phải là công việc đầu tiên ta phải tiến hành khi phân tích hệ thống. Tiêu chuẩn Lyapunov là một công cụ hữu ích giúp ta thực hiên được điều đó.

Định nghĩa 4 : Một hệ thống có mô hình không kích thích :

$$\frac{d\underline{x}}{dt} = \underline{f}(\underline{x}, \underline{u}, t)\Big|_{\underline{u}=\underline{0}} = \underline{\tilde{f}}(\underline{x}, t)$$

(3.3)

với một điểm cân bằng là gốc toạ độ $\underline{0}$, được gọi là :

a) Ổn định Lyapunov tại điểm cân bằng $\underline{0}$ nếu với $\varepsilon > 0$ bất kì bao giờ cũng tồn tại δ phụ thuộc ε sao cho nghiệm $\underline{x}(t)$ của (3.3) với $\underline{x}(0) = \underline{x_0}$ thoả mãn : $\|\underline{x}_0\| < \delta \Rightarrow \|\underline{x}(t)\| < \varepsilon, \forall t \geq 0.$

b) Ổn định tiệm cận Lyapunov tại điểm cân bằng $\underline{0}$ nếu với $\varepsilon > 0$ bất kì bao giờ cũng tồn tại δ phụ thuộc ε sao cho nghiệm $\underline{x}(t)$ của (3.3) với

$\underline{x}(0) = \underline{x_0}$ thoả mãn :

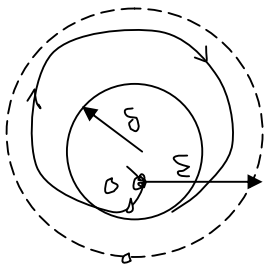
$\lim_{t\to\infty} \underline{x}(t) = \underline{0}$.

III.1.4.1.Tiêu chuẩn Lyapunov:

Để làm quen và tiếp cận tiêu chuẩn Lyapunov ta hãy bắt đầu từ hệ bậc 2 có mô hình trạng thái autonom khi không bị kích thích :

$$\frac{d\underline{x}}{dt} = \underline{f}(x_1, x_2) \qquad \text{với } \underline{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \qquad (3.4)$$

Hệ trên được giả thiết là cân bằng tại gốc toạ độ $\underline{0}$.



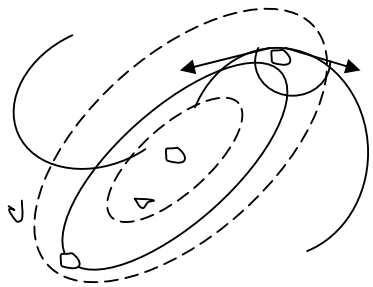
Hình 3.4 : ***Minh họa khái niệm ổn định Lyapunov:***

Hình 3.4 minh hoạ cho định nghĩa 4 về tính ổn định Lyapunov tại $\underline{0}$ đã gợi ý cho ta một hướng khá đơn giản để xét tính ổn định cho hệ (3.4) tại $\underline{0}$. Chẳng hạn bằng cách nào đó ta đã có được họ các đường cong khép kín v bao quanh gốc toạ độ $\underline{0}$. Vậy thì để kiểm tra hệ có ổn định tại $\underline{0}$ hay không ta chỉ cần kiểm tra xem quỹ dạo pha $\underline{x}(t)$, tức là nghiệm của (3.4) đi

từ điểm trạng thái đầu $\underline{x}_0$ cho trước nhưng tuỳ ý nằm trong miền bao bởi một trong các đường cong khép kín đó, có cắt các đường cong v này theo hướng từ ngoài vào trong hay không (hình 3.5).

- Nếu $\underline{x}(t)$ không cắt bất cứ một đường cong họ v nào theo chiều từ trong ra ngoài thì hệ sẽ ổn định tại $\underline{0}$.

- Nếu $\underline{x}(t)$ cắt mọi đường cong họ v theo chiều từ ngoài vào trong thì hệ sẽ ổn định tiệm cận tại $\underline{0}$.



Hình 3.5: ***Một gợi ý về việc kiểm tra tính ổn định của hệ tại O***

Rõ ràng là cần và đủ để quỹ đạo pha $\underline{x}(t)$ của hệ không cắt bất cứ một đường cong khép kín thuộc họ v theo chiều từ trong ra ngoài là tại điểm cắt đó, tiếp tuyến của $\underline{x}(t)$ phải tạo với vector Δ_v, được định nghĩa là vector vuông góc với đường cong đó theo hướng từ trong ra ngoài, một góc φ không nhỏ hơn 90^0. Nói cách khác, hệ sẽ ổn định tại $\underline{0}$ nếu như có được điều kiện:

$$0 \geq |\Delta_v| . \left| \frac{d\underline{x}}{dt} \right| . \cos\varphi = \Delta_v^T \frac{d\underline{x}}{dt} \qquad (3.5)$$

tại mọi giao điểm của $\underline{x}(t)$ với các đường cong thuộc họ v.

Vấn đề còn lại là làm thế nào có được các đường cong v sao cho việc kiểm tra điều kiện (1.48) được thuận tiện. Câu trả lời là sử dụng hàm xác định dương V($\underline{x}$)được định nghĩa như sau :

Định nghĩa 5 : Một hàm thực nhiều biến, có thể không dừng $V(\underline{0},t)$, được gọi là hàm xác định dương nếu :

a) $V(\underline{0},t)=0$

b) Tồn tại hai hàm một biến, dừng $\gamma_1(a)$ và $\gamma_1(b)$ liên tục, đơn điệu tăng với $\gamma_1(0)=\gamma_2(0)=0$ sao cho :

$$0<\gamma_1(\|\underline{x}\|)\le V(\underline{x},t)\le\gamma_2(\|\underline{x}\|) \quad \text{với mọi } \underline{x}\neq\underline{0} \qquad (3.6)$$

Hàm $V(\underline{x},t)$ sẽ xác định dương trong toàn bộ không gian trạng thái nếu còn có :

$$\lim_{a\to\infty}\gamma_1(a)=\infty \Rightarrow \lim_{\|\underline{x}\|\to\infty} V(\underline{x},t)=\infty \quad .$$

Định lý 1 : Hệ phi tuyến (có thể không autonom) cân bằng tại gốc toạ độ và khi không bị kích thích thì được mô tả bởi hình :

$$\frac{d\underline{x}}{dt}=\underline{f}(\underline{x},t) \qquad (3.7)$$

sẽ ổn định Lyapunov tại $\underline{0}$ với miền ổn định O nếu :

a) Trong O tồn tại một hàm xác định dương $V(\underline{0},t)$.

b) Đạo hàm của nó tính theo mô hình (1.51) có giá trị không dương trong O, tức là :

$$\frac{dV}{dt}=\frac{\partial V}{\partial t}+\frac{\partial V}{\partial \underline{x}}f(\underline{x},t)\le 0 \text{ với mọi } \underline{x}\in O. \qquad (3.8)$$

Định lý 2: Hệ phi tuyến (có thể không autonom) cân bằng tại gốc toạ độ và khi không bị kích thích thì được mô tả bởi mô hình.

$$\frac{d\underline{x}}{dt}=\underline{f}(\underline{x},t) \qquad (3.9)$$

sẽ ổn định tiệm cận Lyapunov tại $\underline{0}$ với miền ổn định O nếu :

a)Trong O tồn tại một hàm xác định dương $V(\underline{x},t)$.

b) Đạo hàm của nó tính theo môt hình (1.51) có giá trị âm trong O với $\underline{x} \neq \underline{0}$, tức là :

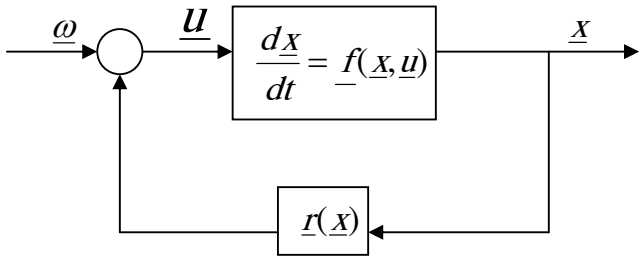
$$\frac{dV}{dt} = \frac{\partial V}{\partial t} + \frac{\partial V}{\partial \underline{x}} \underline{f}(\underline{x}, t) < 0 \text{ với mọi } \underline{x} \in O \text{ và } \underline{x} \neq \underline{0} \quad . \qquad (3.10)$$

III.1.4.2.Tiêu chuẩn Lyapunov phục vụ thiết kế bộ điều khiển:

Ngoài việc kiểm tra tính ổn định, tiêu chuẩn Lyapunov còn được sử dụng để thiết kế bộ điều khiển ổn định đối tượng phi tuyến. Chẳng hạn đối tượng có mô hình :

$$\frac{d\underline{x}}{dt} = \underline{f}(\underline{x}, \underline{u})$$

và được điều khiển bằng bộ điều khiển phản hồi trạng thái $\underline{r}(\underline{x})$



Hình 3.6: ***Ứng dụng tiêu chuẩn Lyapunov để thiết kế bộ điều khiển***

Vậy hệ kín khi không bị kích thích ($\underline{\omega} = \underline{0}$) sẽ có mô hình :

$$\frac{d\underline{x}}{dt} = \underline{f}(\underline{x}, \underline{r}(\underline{x}))$$

Gọi $V(\underline{x})$ là hàm xác định dương thích hợp, khi đó để hệ kín ổn định tiệm cận với miền ổn định là O thì bộ điều khiển cần tìm $\underline{r}(\underline{x})$ phải thoả mãn :

$$L_f V = \frac{\partial V}{\partial \underline{x}} \underline{f}(\underline{x}, \underline{r}(\underline{x})) < 0 \text{ với mọi } \underline{x} \neq \underline{0} \text{ , } \underline{x} \in O \qquad (3.11a)$$

$$\text{Và } \frac{\partial V}{\partial \underline{x}} \underline{f}(\underline{x}, \underline{r}(\underline{x})) = 0 \text{chỉ khi } \underline{x} = \underline{0} \qquad (3.11b)$$

III.2.Bậc tương đối của hệ phi tuyến:

Bậc tương đối của hệ SISO:

Để dễ tiếp cận tới khái niệm bậc tương đối ta xét trường hợp đặc biệt với đối tượng tuyến tính, mô tả bằng hàm truyền đạt hợp thức chặt (strickly proper):

$$G(s) = \frac{b_0 + b_1 + \ldots + b_m s^m}{a_0 + a_1 s + \ldots + a_n s^n} \qquad (3.12)$$

Khi đó bậc tương đối được hiểu là hiệu r = (n-m) ≥1

Giả sử rằng đối tượng trên, bên cạnh hàm truyền đạt (3.12) còn có mô hình tương đương trong không gian trạng thái :

$$\begin{cases} \dfrac{d\underline{x}}{dt} = A\underline{x} + b\underline{u} \\ y = \underline{c}^T \underline{x} \end{cases} \qquad \underline{x} \in R^n, A \in R^{nxn}, \underline{b} \in R^{nx1}, \underline{c} \in R^{1xn} \qquad (3.13)$$

Vậy thì do

$$G(s) = c^T(sI-A)^{-1}\underline{b}$$

Ta có :

$$\lim_{s\to\infty} s^r G(s) = \frac{b_m}{a_n} \Leftrightarrow \lim_{s\to\infty} s^r [\underline{c}^T(sI - A)^{-1}\underline{b}] = \frac{b_m}{a_n}$$

$$\Leftrightarrow \lim_{s\to\infty} \sum_{k=0}^{\infty} \frac{\underline{c}^T A^k \underline{b}}{s^{k+1-r}} = \frac{b_m}{a_n}$$

Hơn nữa

$$\lim_{s\to\infty} \frac{1}{s^{k+1-r}} = 0 \text{ khi } k > r-1$$

nên chuỗi trên trở thành tổng của hữu hạn r phần tử đầu tiên

$$\lim_{s\to\infty} \sum_{k=0}^{r-1} \frac{\underline{c}^T A^k \underline{b}}{s^{k+1-r}} = \frac{b_m}{a_n}$$

Từ đây, để vế trái bằng giá trị hữu hạn thì cần và đủ là :

$$\underline{c}^T A^k \underline{b} = \begin{cases} = 0 & \text{khi } 0 \le k \le r-2 \\ \neq 0 & \text{khi } k = r-1 \end{cases} \qquad (3.14)$$

Nói cách khác, bậc tương đối r = n-m còn có thể được xác định trực tiếp từ mô hình trạng thái (3.13) của hệ theo công thức (3.14).

Chuyển sang hệ phi tuyến và với sự gợi ý của công thức tính (3.14), khái niệm bậc tương đối của hệ ALI có 1 tín hiệu vào, một tín hiệu ra, được định nghĩa như sau :

Định nghĩa 6: Cho hệ SISO với cấu trúc ALI :

$$\begin{cases} \dfrac{d\underline{x}}{dt} = \underline{f}(\underline{x}) + \underline{h}(x)u \\ y = g(\underline{x}) \end{cases} \qquad (3.15)$$

Bậc tương đối tại điểm trạng thái $\underline{x}$ của hệ là số tự nhiên r mà trong lân cận $\underline{x}$ thoả mãn :

$$L_h L_f^h g(\underline{x}) = \begin{cases} = 0 & \text{khi } 0 \le k \le r-2 \\ \neq 0 & \text{khi } k = r-1 \end{cases} \qquad (3.16)$$

Có thể thấy được ngay rằng với f(x)= Ax , H(x)= b , g(x)=c^Tx , hai công thức (3.14) và (3.16) sẽ đồng nhất, vì :

L_f^h g(x)= $c^T A^T x \Rightarrow L_h L_f^k$ g(x)=$c^T A^k b$

Ví dụ : Xét hệ Val der Pol có mô hình trạng thái như sau :

khi đó thì do :

$$L_h g(x) = \frac{\partial g}{\partial \underline{x}} h(\underline{x}) = [1 \quad 0] \begin{bmatrix} 0 \\ 1 \end{bmatrix} = 0$$

$$L_h L_f g(x) = \frac{\partial (L_f g)}{\partial \underline{x}} \underline{h}(\underline{x}) \frac{\partial}{\partial \underline{x}} \left(\frac{\partial g}{\partial \underline{x}} \underline{f} \right) h(\underline{x})$$

$$= \frac{\partial}{\partial \underline{x}}\left[\begin{pmatrix}1 & 0\end{pmatrix}\begin{pmatrix}x_2 \\ ax_2(1-bx_1^2)-x_1\end{pmatrix}\right]\begin{pmatrix}0 \\ 1\end{pmatrix} = \begin{pmatrix}1 & 0\end{pmatrix}\begin{pmatrix}0 \\ 1\end{pmatrix} = 1 \neq 0$$

Bậc tương đối của hệ bằng 2 (tại mọi $\underline{x}$).

Tuy nhiên, cũng cần phải để ý rằng hệ phi tuyến (3.15) có thể có bậc tương đối khác nhau ở những điểm trạng thái khác nhau. Ngoài ra, khác với hệ tuyến tính, không phải ở bất cứ điểm trạng thái x nào trong không gian trạng thái, hệ phi tuyến phẳng có bậc tương đối. Chẳng hạn, hệ sẽ không có bậc tương đối tại điểm trạng thái x_0 mà trong lân cận của nó có :

$$L_h g(\underline{x}) \neq 0, L_h L_f g(\underline{x}) \neq 0, \ldots, L_h L_f^h g(\underline{x}) \neq 0, \ldots$$

III.3.Tính động học không:

Rất nhiều khái niệm sử dụng trong hệ phi tuyến được chuyển thể từ hệ tuyến tính, chẳng hạn khái niệm bậc tương đối, hệ thụ động, … cũng như vậy là tính động học không (zero dynamic). Do đó, để dễ tiếp cận tới khái niệm này, ta nên bắt đầu từ hệ tuyến tính.

Xét hệ phi tuyến SISO có mô hình trạng thái :

$$\begin{cases} \dfrac{d\underline{x}}{dt} = \underline{f}(\underline{x}) + \underline{h}(x)u \\ y = g(\underline{x}) \end{cases} \qquad (3.17)$$

Tính động học không (zero dynamic) của hệ (3.17) được định nghĩa như sau :

Định nghĩa 7 : Nếu hệ (3.17) có ít nhất một điểm trạng thái đầu $\underline{x}_0 \neq \underline{0}$ và ứng với nó là tín hiệu điều khiển $u_0(t)$ sao cho tín hiệu đầu ra y(t) đồng nhất bằng không thì hệ được gọi là có tính động học không (zero dynamic).

Ta có thể thấy được là để hệ có tính động học không thì cần thiết phải có $g(x_0) = 0$. Giả sử rằng hệ (3.17) có bậc tương đối là r, tức là :

$$L_h L_f^k g(x) = \begin{cases} = 0 & \text{nếu } 0 \le k \le r\text{-}2 \\ \neq 0 & \text{nếu } k = r - 1 \end{cases} \qquad (3.18)$$

Khi đó, với phép đổi trục toạ độ vi phôi :

$$\underline{z} \begin{bmatrix} z_1 \\ \vdots \\ z_{r-1} \\ z_r \\ z_{r+1} \\ \vdots \\ z_n \end{bmatrix} = \underline{m}(\underline{x}) = \begin{bmatrix} g(\underline{x}) \\ \vdots \\ L_f^{r-2} g(\underline{x}) \\ L_f^{r-1} g(\underline{x}) \\ m_{r+1}(\underline{x}) \\ \vdots \\ m_n(\underline{x}) \end{bmatrix} \quad \text{với } L_h m_k(\underline{x}) = 0 \text{ , } k=r+1 \text{ , } \dots \text{ , } n$$

hệ (1.18) đã cho sẽ được đưa về dạng chuẩn

$$\frac{d\underline{z}}{dt} = \frac{d}{dt} \begin{bmatrix} z_1 \\ \vdots \\ z_{r-1} \\ z_r \\ z_{r+1} \\ \vdots \\ z_n \end{bmatrix} = \begin{bmatrix} z_2 \\ \vdots \\ z_r \\ a(\underline{z}) + b(\underline{z})u \\ c_1(\underline{z}) \\ \vdots \\ c_{n-r}(\underline{z}) \end{bmatrix} \text{ , } y = z_1 \qquad (3.19)$$

Trong đó

$A(\underline{z}) = L_f^r g(\underline{m}^{-1}(\underline{z}))$, $b(\underline{z}) = L_h L_f^{r-1} g(\underline{m}^{-1}(\underline{z}))$, $c_i(\underline{z}) = L_h m_{r+1}(\underline{m}^{-1}(\underline{z}))$

sử dụng kí hiệu :

$$\underline{z} = \begin{bmatrix} \underline{\xi} \\ \underline{\eta} \end{bmatrix} \text{ với } \underline{\xi} = \begin{bmatrix} z_{r+1} \\ \vdots \\ z_n \end{bmatrix} \text{ , } \underline{\eta} = \begin{bmatrix} z_{r+1} \\ \vdots \\ z_n \end{bmatrix} \text{ và } \underline{c}(\underline{z}) = \begin{bmatrix} c_1(\underline{z}) \\ \vdots \\ c_{n-r}(\underline{z}) \end{bmatrix}$$

thì mô hình (3.19) được viết thành

$$\frac{d\underline{z}}{dt} = \frac{d}{dt}\begin{bmatrix} z_1 \\ \vdots \\ z_{r-1} \\ z_r \\ \underline{\eta} \end{bmatrix} = \begin{bmatrix} z_2 \\ \vdots \\ z_r \\ a(\underline{\xi},\underline{\eta}) + b(\underline{\xi},\underline{\eta})u \\ \underline{c}(\underline{\xi},\underline{n}) \end{bmatrix} \quad , y = z_1 \qquad (3.20)$$

Giả sử rằng hệ (3.17) có tính động học không ứng với trạng thái đầu $x_0 \neq 0$ và tín hiệu điều khiển $u_0(t)$ thích hợp. Vậy thì từ $y(t) = z_1(t) = 0$ ta suy ra được :

$$z_1(t) = \ldots = z_r(t) = 0$$

và do đó là $\xi = 0$. Điều này dẫn đến :

$$a(\underline{0},\underline{\eta}) + b(\underline{0},\underline{\eta})u_0 = 0 \Leftrightarrow u_0(t) = -\frac{a(\underline{0},\underline{\eta})}{b(\underline{0},\underline{\eta})} \qquad (3.21a)$$

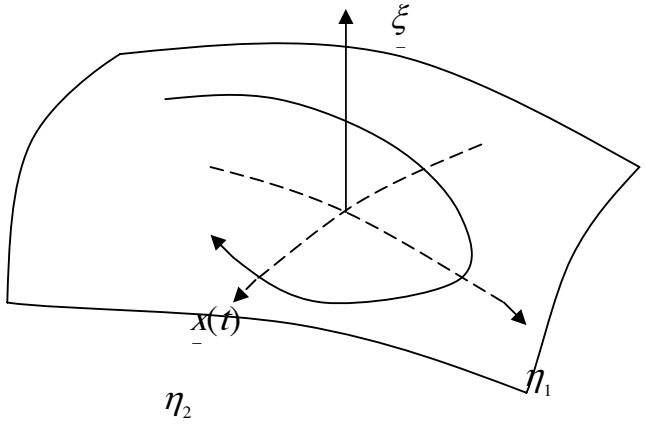
$$\frac{d\underline{\eta}}{dt} = \underline{c}(0,\underline{\eta}) \qquad (3.21b)$$

Đó cũng là hai phương trình phân tích tính động học không của hệ (3.17) thông qua mô hình tương đương (3.20) của nó. Điều kiện để có phương trình (3.21b) là hệ (3.17) phải có bậc tương đối r nhỏ hơn n ($r < n$).

Từ $\xi=0$ cũng như phép biến đổi trục toạ độ (3.18) và 2 phương trình (3.21) ta thấy, ở chế độ động học không, quỹ đạo trạng thái x(t) phải thoả mãn : $g(\underline{x}) = L_f g(\underline{x}) = \ldots = L_f^{r-1} g(\underline{x}) = 0$.

Nói cách khác x(t) của động học không sẽ chỉ nằm trong đa tạp (hình 3.7)

$$K = \{ \underline{x} \in R^n | g(\underline{x}) = L_f g(\underline{x}) = \ldots = L_f^{r-1} g(\underline{x}) = 0 \} \qquad (3.22)$$



Hình 3.7: ***Quỹ đạo trạng thái của Hệ phi tuyến, khi đang ở chế độ Động học không, luôn nằm trong đa tạp K.***

Tuy rằng nằm trong đa tạp K, song việc quỹ đạo x(t) ở chế độ động học không (ứng với tín hiệu điều khiển $u_0(t)$ thích hợp) có tiến về gốc toạ độ 0 hay không thì chưa được đảm bảo và điều này không được quyết định bởi hệ phi tuyến (3.17) có ổn định hay không. Nó chỉ có thể tiến về $\underline{0}$ n như hệ (3.21b) là ổn định tiệm cận Lyapunov, tức là phải tồn tại 1 hàm xác định dương Q(η) sao cho :

$$\frac{\partial Q}{\partial \underline{\eta}} \underline{c}(0, \underline{\eta}) < 0 \quad \text{khi} \quad \underline{\eta} \neq 0$$

III.4.Thiết kế bộ Điều khiển trượt cho tay máy:

III.4.1.Điều khiển trượt:

Hệ phi tuyến có mô hình

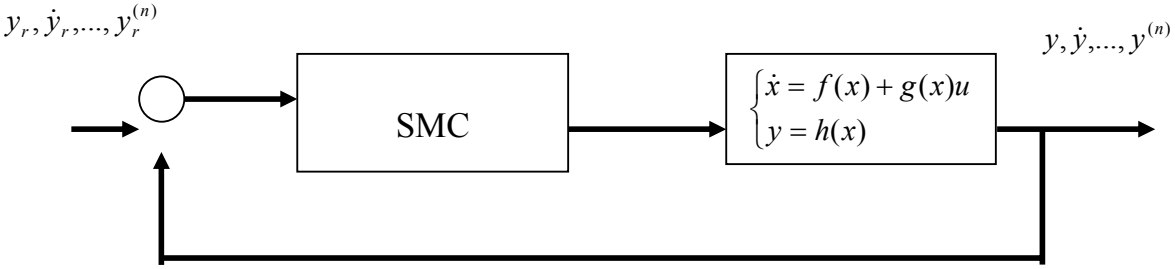
$$\begin{cases} \dot{x} = f(x,u) = f(x) + g(x)u \\ y = h(x) \end{cases} \qquad (3.23)$$

Trong đó y là tín hiệu đầu ra, u là tín hiệu đầu vào, $x = [x_1, x_2, .., x_n]^T$ là vector trạng thái của hệ, $f(x) = [f_1(x), f_2(x), ..., f_n(x)]^T$, $g(x) = [g_1(x), g_2(x), ..., g_n(x)]^T$

Hệ phi tuyến có bậc tương đối là p nếu:

$$\begin{cases} \dfrac{d^p h(x)}{dt^p} = L_f^p h(x) + L_g L_f^{p-1} h(x) u \\ L_g L_f^{p-1} h(x) \neq 0; L_g L_f^i h(x) = 0, i = 1,2,3,..., p-2 \end{cases} \quad (3.24)$$

Sơ đồ điều khiển:



III.4.1.1.Trường hợp bậc tương đối của hệ bằng bậc của hệ p=n:

Để có thể thiết kế được bộ điều khiển thì hệ (3.23) phải tồn tại mặt trượt. Hệ (3.23) có mặt trượt S khi thoả mãn:

- $S = e + \sum_{i=1}^{n-1} \lambda_i e^{(i)}$ (3.25)
- $A(S) = 1 + \lambda_1 S + ... + \lambda_{n-1} S^{(n-1)}$

là đa thức Hurwitz để có:

$$\lim_{t \to \infty} e(t) = 0 \quad (3.26)$$

- S(0) = 0 (3.27)

Điều kiện để (3.23) trượt về điểm cân bằng là phải thoả mãn điều kiện trượt. Điều kiện trượt được xây dựng trên cơ sở đảm bảo hệ kín ổn định tiệm cận, có nghĩa là cho hệ trong hình trên tồn tại 1 hàm Lyapunov. Giả sử hệ có hàm Lyapunov có dạng sau:

$$V(x,t) = \frac{1}{2} S^2 \quad (3.28)$$

là hàm xác định dương. Đạo hàm của nó có dạng sau:

$$\frac{dV}{dt} = S\dot{S} \qquad (3.29)$$

Hệ (3.23) ổn định tiệm cận khi (3.29) là hàm có dấu xác định âm:

$$S\dot{S} < 0 \Rightarrow \begin{cases} S > 0, \dot{S} < 0 \\ S < 0, \dot{S} > 0 \end{cases} \qquad (3.30)$$

Như vậy $\dot{S}$ phải trái dấu với S, do vậy ta có:

$$\dot{S} = -Kh(S) \qquad (3.31)$$

h(S) cùng dấu với S do vậy để thoả mãn điều kiện trượt ta có thể chọn hàm h(S) có các dạng sau: hàm dấu Sig(S), hàm bão hoà Saturation(S), hàm h(S)=Tan(S)

Theo (3.25) ta có:

$$\dot{S} = \dot{e} + \sum_{i=2}^{n} \lambda_{i-1} e^{(i)} = \dot{e} + \lambda_1 \ddot{e} + ... + \lambda_n e^{(n)} = -Kh(S) \qquad (3.32)$$

Ta có:

$$e^{(n)} = y_r^{(n)} - (L_f^n h(x) + L_g L_f^{n-1} h(x) u) \qquad (3.33)$$

Do vậy: $\dot{e} + \lambda_1 \ddot{e} + ... + \lambda_n y_r^{(n)} - \lambda_n (L_f^n h(x) + L_g L_f^{n-1} h(x) u) = -Kh(S)$ (3.34)

Tín hiệu điều khiển tìm được:

$$u(t) = \frac{Kh(S) + \dot{e} + \lambda_1 \ddot{e} + ... + \lambda_n y_r^{(n)} - \lambda_n L_f^n h(x)}{\lambda_n L_g L_f^{n-1} h(x)} \qquad (3.35)$$

III.4.1.2. Trường hợp bậc tương đối của hệ p<n

Hệ (3.23) phải thoả mãn động học không.

Xây dựng mặt trượt :

- $S = e + \sum_{i=1}^{p-1} \lambda_i e^{(i)}$ (3.36)
- $A(s) = 1 + \lambda_1 S + ... + \lambda_{p-1} S^{(p-1)}$ là đa thức Hurwitz, để có $\lim_{t \to \infty} e(t) = 0$ (3.37)
- $S(0) = 0$, mặt trượt phải đi qua gốc toạ độ và thoả mãn điều kiện trượt.

Hoàn toàn tương tự như trong trường hợp trên, ta xây dựng hàm Lyapunov có dạng sau:

$V = \frac{1}{2}S^2$ xác định dương

$\dot{V} = S\dot{S}$ xác định âm

Ta có:

$$\begin{aligned}
&\dot{S} = -Kh(S)\\
&\dot{S} = \dot{e} + \lambda_1\ddot{e} + ... + \lambda_{p-1}e^{(p)}\\
&e = y_r(t) - y(t)\\
&e^{(p)} = y_r^{(p)}(t) - y^{(p)}(t)\\
&y^{(p)}(t) = L_f^p h(x) + L_g L_f^{(p-1)} h(x)u\\
&\dot{S} = \dot{e} + \lambda_1\ddot{e} + ... + \lambda_{p-1}y_r^{(p)}(t) - \lambda_{p-1}(L_f^p h(x) + L_g L_f^{(p-1)} h(x)u) = -Kh(S)
\end{aligned} \tag{3.38}$$

Tín hiệu điều khiển:

$$u(t) = \frac{Kh(S) + \dot{e} + \lambda_1\ddot{e} + ... + \lambda_{p-1}(y_r^{(p)} - L_f^p h(x))}{\lambda_{p-1}L_g L_f^{(p-1)} h(S)} \tag{3.39}$$

III.4.2. Thiết kế bộ điều khiển trượt cho tay máy n bậc tự do:

Mô hình động lực học của tay máy:

$$\tau = H(q)\ddot{q} + h(q,\dot{q}) \tag{3.40}$$

với H(q) là ma trận quán tính xác định dương, đối xứng.

Chúng ta giả sử rằng các giá trị ước lượng $\hat{H}(q)$ và $\hat{h}(q,\dot{q})$ quan hệ với giá trị thực $H(q)$ và $h(q,\dot{q})$ bởi bất đẳng thức sau:

$$\left\|\hat{H}^{-1}(q)H(q)\right\| \le \beta(q) \tag{3.41}$$

và $$\left\|h(q,\dot{q}) - \hat{h}(q,\dot{q})\right\| \le \Delta h_{\max}(q,\dot{q}) \tag{3.42}$$

với $\beta(q)$ và $\Delta h_{\max}(q,\dot{q})$ là những hàm đã biết.

Viết lại biểu thức động lực học dưới dạng:

$$\ddot{q} = f(q,\dot{q}) + B(q)\tau \tag{3.43}$$

Với $f(q,\dot{q}) = -H^{-1}(q)h(q,\dot{q})$ (3.44)

$B(q) = H^{-1}(q)$ (3.45)

Nhiệm vụ của điều khiển là tìm mô men thích hợp τ sao cho vector vị trí q của tay máy bám theo quỹ đạo mong muốn q_d.

Chúng ta định nghĩa sai lệch trạng thái e và mặt trượt như sau:

$e = q_d - q$ (3.46)

$S = Ce + \dot{e}; C = C^T > 0$ (3.47)

Rõ ràng rằng S=0 thì $q(t) \to q_q(t)$. Quả thực với S=0 ta có thể viết lại như sau:

$S = 0 \Rightarrow Ce + \dot{e} = 0 \Rightarrow \dot{e} = -Ce$

Như vậy hệ thống ổn định tiệm cận nếu có e = 0 và theo đó điều kiện bám $q(t) \to q_q(t)$ sẽ được đảm bảo.

Do vậy vấn đề điều khiển là phải tìm mô men τ thích hợp sao cho vector trạng thái của hệ thống có thể bám được trên mặt trượt. Hay phải tìm τ thỏa mãn điều kiện trượt. Điều kiện trượt có thể xác định theo tiêu chuẩn Lyapunov.

Chúng ta định nghĩa hàm Lyapunov như sau:

$V = \frac{1}{2} S^T S > 0$ (3.48)

Đạo hàm của (3.48) có dạng:

$\dot{V} = S^T \dot{S}$ (3.49)

Như vậy, nếu $\dot{V} < 0$ thì với $V \to 0$ dẫn tới $S \to 0$ và $e \to 0$

Do vậy, điều kiện đủ của điều kiện trượt là:

$S^T \dot{S} < 0$ (3.50)

Khi đó điều kiện trượt đảm bảo cho hệ kín ổn định toàn cục, tiệm cận và điều kiện bám được thực hiện mặc dù mô hình không chính xác, nhiễu,…

Nếu điều kiện trượt có thể thỏa mãn theo đó:

$$S^T \dot{S} \le -\alpha \|S\| \le 0; \alpha > 0; \|S\| = \sqrt{\sum_{i=1}^{n} S_i^2} \qquad (3.51)$$

Tiếp đó, mặt phẳng trượt S=0 sẽ đạt được với thời gian giới hạn nhỏ hơn T_0 ở đó:

$$T_0 = \frac{1}{2\alpha} \|S(q(0))\| \qquad (3.52)$$

Biểu thức trên được chứng minh như sau:

Từ (29) ta có:

$$\frac{S^T \dot{S}}{\|S\|} \le -\alpha \qquad (3.53)$$

Thay $\dot{V} = S^T \dot{S}$ và $\|S\| = (2V)^{\frac{1}{2}}$ vào (3.53) sau đó tích phân hai vế với $t=0 \rightarrow t_{reach}$, $S(q(t_{reach}))=0$ ta có:

$$\int_0^{t_{reach}} \frac{\dot{V}}{(2V)^{\frac{1}{2}}} dt = \frac{1}{\sqrt{2}} \left[V^{\frac{1}{2}} \right]_0^{t_{reach}} = \frac{\|S(q(0))\|}{2} \le -\alpha t_{reach} \rightarrow t_{reach} \le \frac{\|S(q(0))\|}{2\alpha} = T_0 \quad (3.54)$$

Bây giờ chúng ta tìm đầu vào bộ điều khiển τ thỏa mãn điều kiện trượt.

Lấy đạo hàm biểu thức (3.47) ta có:

$$\dot{S} = C\dot{e} + \ddot{q} - \ddot{q}_d \qquad (3.55)$$

Thay biểu thức (3.39) vào ta có:

$$\dot{S} = C\dot{e} + f(q, \dot{q}) + B(q)\tau - \ddot{q}_d \qquad (3.56)$$

Do đó tín hiệu điều khiển có dạng

$$\tau = \hat{B}^{-1} \left[\tau_{eq} - KSgn(s) \right] \qquad (3.57)$$

với:

$$\tau_{eq} = \ddot{q} - C\dot{e} - \hat{f}(q,\dot{q})$$
$$Sgn(s) = [Sgn(s_1), Sgn(s_2),..., Sgn(s_n)]^T \quad (3.58)$$

K>0, K là ma trận khuyếch đại nxn.

Ma trận khuyếch đại K phải chọn đủ lớn để điều kiện trượt được thỏa mãn mặc dù có tham số không rõ, nhiễu,…

Trong trường hợp ước lượng chính xác $\hat{B} = B, \hat{f} = f$ thì điều kiện trượt được viết lại như sau:

$$S^T\dot{S} = -S^T KSgn(s) \le -\alpha\|S\| \quad (3.59)$$

Nếu chọn $K \ge \beta I; \beta > \alpha$ (3.60)

và $S^T\dot{S} = -\beta|S| = -\beta\sum_{i=1}^{m}|S_i| \le -\beta\sqrt{\sum_{i=1}^{m}S_i^2} = -\beta\|S\| \le -\alpha\|S\|$ (3.61)

thì chế độ trượt xảy ra.

Ta nhận thấy rằng, đầu vào điều khiển được gián đoạn qua s(t) như cho ở biểu thức (3.57). Hiện tượng chattering xảy ra. Bởi vì trong thực tế, sự chuyển đổi là không lý tưởng. Trong trường hợp sai số ước lượng là không đủ nhỏ thì việc chọn K là không đơn giản như biểu thức trên.

Trong trường hợp đó $\dot{S}$ cho dưới dạng:

$$\dot{S} = -\ddot{q}_d + C\dot{e} + f(q,\dot{q}) + B(q)\hat{B}^{-1}\tau_{eq} - B(q)\hat{B}^{-1}KSgn(s) \quad (3.62)$$

đặt $f = \hat{f} + (f - \hat{f}); R = B(q)\hat{B}^{-1}$ dẫn tới:

$$\dot{S} = (R - I)\tau_{eq} + (f - \hat{f}) - RKSgn(s) \quad (3.63)$$

Từ đây, điều kiện trượt là:

$$S^T\dot{S} = S^T\left\{(R - I)\tau_{eq} + (f - \hat{f}) - RKSgn(s)\right\} \le -\alpha S^T Sgn(s) \quad (3.64)$$

Do vậy, nếu chọn K để:

$$S^T RKSgn(s) \ge S^T\left\{(R - I)\tau_{eq} + (f - \hat{f})\right\} + \alpha S^T Sgn(s) \quad (3.65)$$

thì điều kiện trượt như ở trên $S\dot{S} < 0$ được thỏa mãn và điều kiện trượt đạt được.

$$\left\|R^{-1}\right\| = \left\|\hat{B}B^{-1}\right\| \le \beta(q) \qquad (3.66)$$

Từ biểu thức (3.41) và (3.42) ta có bất đẳng thức:

$$\left\|R^{-1}\right\| = \left\|\hat{B}B^{-1}\right\| \le \beta(q) \qquad (3.67)$$

$$\left\|R^{-1}(f-\hat{f})\right\| = \left\|\hat{B}(\hat{h}-h)\right\| \le \left\|\hat{B}\right\|\Delta h_{\max} \qquad (3.68)$$

Từ đây, ta có thể chọn ma trận K thoả mãn điều kiện trượt như sau:

$$\|K\| \ge \|(1-\beta)I\|\|\tau_{eq}\| + \left\|\hat{B}\right\|\Delta h_{\max} + \alpha\|\beta I\| \qquad (3.69)$$

III.4.3. Ứng dụng Điều khiển trượt cho tay máy Robot 2 bậc tự do:

III.4.3.1. Phương trình động lực học tay máy hai bậc tự do toàn khớp quay:

Bộ thông số tay máy: $m_1 = m_2 = 1$ kg

$l_1 = l_2 = 1$ m

Phương trình động lực học:

$$\begin{bmatrix} F_1 \\ F_2 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} \\ h_{21} & h_{22} \end{bmatrix}\begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix} + \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix}\begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix} + \begin{bmatrix} g_1 \\ g_2 \end{bmatrix} \qquad (3.70)$$

Trong đó F_1, F_2 là lực được tạo ra ở các khớp động, ma trận H là ma trận xác định dương và đối xứng, ma trận C là ma trận lực ly tâm, G là ma trận lực trọng trường.

Giá trị của các ma trận khi thay giá trị được xác định như sau:

- Ma trận H:

$$\begin{aligned} h_{11} &= 5 + 3\cos\theta_2 \\ h_{12} &= h_{21} = 1 + 3/2\cos\theta_2 \\ h_{22} &= 1 \end{aligned} \qquad (3.71)$$

- Ma trận C:

$$\begin{aligned} c_{11} &= -3\dot{\theta}_2 \sin\theta_2 \\ c_{12} &= -3/2\dot{\theta}_2 \sin\theta_2 \\ c_{21} &= -3\dot{\theta}_1 \sin\theta_2 \\ c_{22} &= 0 \end{aligned} \tag{3.72}$$

- Ma trận G:

$$\begin{aligned} g_1 &= 15\cos\theta_1 - 15\cos(\theta_1 + \theta_2) \\ g_2 &= 15\cos(\theta_1 + \theta_2) \end{aligned} \tag{3.73}$$

III.4.3.2. Mô hình động lực học tay máy hai bậc tự do:

Chúng ta đặt các biến trạng thái là tín hiệu góc quay và vận tốc của các khớp tay máy:

Khớp 1:

$$\begin{cases} x_{11} = \theta_1 \\ x_{12} = \dot{x}_{11} = \dot{\theta}_1 \\ \dot{x}_{12} = \ddot{\theta}_1 \end{cases} \tag{3.74}$$

Khớp 2:

$$\begin{cases} x_{21} = \theta_2 \\ x_{22} = \dot{x}_{21} = \dot{\theta}_2 \\ \dot{x}_{22} = \ddot{\theta}_2 \end{cases} \tag{3.75}$$

Tín hiệu vào u:

$$\begin{cases} u_1 = F_1 \\ u_2 = F_2 \end{cases} \tag{3.76}$$

Từ biểu thức (3.70) ta có:

$$\begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_1 \end{bmatrix} = H^{-1}\begin{bmatrix} F_1 \\ F_2 \end{bmatrix} - H^{-1}\begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix}\begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix} - H^{-1}\begin{bmatrix} g_1 \\ g_2 \end{bmatrix} \tag{3.77}$$

trong đó H^{-1} là ma trận nghịch đảo của ma trận H.

Tính H^{-1} và kết hợp tất cả các phương trình trên và thay vào (3.77) để tính được:

Khớp 1

$$\begin{cases} \dot{x}_{11} = x_{12} \\ \dot{x}_{12} = \dfrac{1}{4 - 9/4\cos^2 x_{21}}(u_1 + \dfrac{3}{2}x_{22}(2x_{12} + x_{22})\sin x_{21} - 15(\cos x_{11} - \cos x_{21})) - \\ -(1 + \dfrac{3}{2}\cos x_{21})(u_2 - \dfrac{3}{2}x_{12}^2 \sin x_{21} - 15\cos(x_{11} + x_{21})) \end{cases} \quad (3.78)$$

Khớp 2:

$$\begin{cases} \dot{x}_{21} = x_{22} \\ \dot{x}_{22} = \dfrac{1}{4 - 9/4\cos^2 x_{21}}(-(1 + \dfrac{3}{2}\cos x_{21})(u_1 + \dfrac{3}{2}x_{22}(2x_{12} + x_{22})\sin x_{21} - 15(\cos x_{11} - \cos x_{21}))) + \\ +(5 + 3\cos x_{21})(u_2 - \dfrac{3}{2}x_{12}^2 \sin x_{21} - 15\cos(x_{11} + x_{21}))) \end{cases} \quad (3.79)$$

III.4.3.3. Thiết kế bộ điều khiển trượt cho tay máy 2 bậc tự do:

Xác định bậc tương đối cho khớp 1 và khớp 2:

Từ phương trình trạng thái của các khớp (3.78), (3.79) và biểu thức (3.24) ta có được ngay là trong trường hợp này là p=n hay bậc tương đối của từng khớp p=2.

Xây dựng mặt trượt cho từng khớp:

$$\begin{aligned} S_1 &= e_1 + \lambda_1 \dot{e}_1 \\ S_2 &= e_2 + \lambda_2 \dot{e}_2 \end{aligned} \quad (3.80)$$

ở đây:

$$\begin{aligned} e_1 &= x_{d1} - x_{r1} \\ e_2 &= x_{d2} - x_{r2} \end{aligned} \quad (3.81)$$

λ_1, λ_2 là những số thực dương.

Điều kiện để xảy ra chế độ trượt cho hệ trên:

$$\dot{V} = S\dot{S} < 0 \quad (3.82)$$

Xây dựng bộ điều khiển:

Từ (3.78) và (3.79) nếu đặt:

$$\begin{cases} \dot{x}_{11} = x_{12} \\ \dot{x}_{12} = f_1(x) + g_1(x,u) \end{cases} \quad (3.83)$$

và:

$$\begin{cases} \dot{x}_{21} = x_{22} \\ \dot{x}_{22} = f_2(x) + g_2(x,u) \end{cases} \quad (3.84)$$

Ta có:

$$\begin{aligned} &\dot{S}_1 = \dot{e}_1 + \lambda_1 \ddot{e}_1 = \dot{e}_1 + \lambda_1 \ddot{x}_{1d} - \lambda_1 \ddot{x}_{11} = \dot{e}_1 + \lambda_1 \ddot{x}_{1d} - \lambda_1 (f_1(x) + g_1(x,u)) \\ &= (\dot{e}_1 + \lambda_1 \ddot{x}_{1d} - \lambda_1 f_1(x)) - \lambda_1 g_1(x,u) = -K_1 h(S_1) \end{aligned} \quad (3.85)$$

$$\Rightarrow g_1(x,u) = \frac{K_1 h(S_1) + (\dot{e}_1 + \lambda_1 \ddot{x}_{1d} - \lambda_1 f_1(x))}{\lambda_1}$$

Tương tự ta cũng có:

$$\Rightarrow g_2(x,u) = \frac{K_2 h(S_2) + (\dot{e}_2 + \lambda_2 \ddot{x}_{2d} - \lambda_2 f_2(x))}{\lambda_2} \quad (3.86)$$

Theo (3.77) ta có:

$$\begin{bmatrix} g_1(x,u) \\ g_2(x,u) \end{bmatrix} = H^{-1} u \quad (3.87)$$

$$\begin{bmatrix} f_1(x) \\ f_2(x) \end{bmatrix} = -H^{-1} \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix} \begin{bmatrix} x_{12} \\ x_{22} \end{bmatrix} - H^{-1} \begin{bmatrix} g_1 \\ g_2 \end{bmatrix} \quad (3.88)$$

Chú ý: $g_1 \neq g_1(x,u)$

Từ (3.58) ta có được:

$$\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = H \begin{bmatrix} \frac{K_1 h(S_1) + \dot{e}_1 + \lambda_1 \ddot{x}_{1d}}{\lambda_1} - f_1(x) \\ \frac{K_2 h(S_2) + \dot{e}_2 + \lambda_2 \ddot{x}_{2d}}{\lambda_2} - f_2(x) \end{bmatrix} \quad (3.89)$$

Thay (3.88) vào (3.89) ta có được bộ điều khiển:

$$\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = H\left[\begin{array}{c} \dfrac{K_1 h(S_1) + \dot{e}_1 + \lambda_1 \ddot{x}_{1d}}{\lambda_1} \\ \dfrac{K_2 h(S_2) + \dot{e}_2 + \lambda_2 \ddot{x}_{2d}}{\lambda_2} \end{array}\right] + \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix}\begin{bmatrix} x_{12} \\ x_{22} \end{bmatrix} + \begin{bmatrix} g_1 \\ g_2 \end{bmatrix} \quad (3.90)$$

III.4.3.4. Tính toán giá trị đặt θ_i cho tay máy hai bậc tự do:

Để tính toán giá trị đặt cho tay máy hai bậc tự do chúng ta cần giải bài toán động học ngược, từ đó tính toán giá trị đặt cho các khớp:

$${}^0A_2 = {}^0A_1\,{}^1A_2 = \begin{bmatrix} C_{12} & -S_{12} & 0 & l(C_{12}+C_1) \\ S_{12} & C_{12} & 0 & l(S_{12}+S_1) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.91)$$

theo cách biến đổi toạ độ ta có được:

$$\begin{bmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{bmatrix} = {}^0A_2 \begin{bmatrix} x_2 \\ y_2 \\ z_2 \\ 1 \end{bmatrix} \quad (3.92)$$

Tuy nhiên, trong bài toán này có x_2, y_2, z_2=0 vì ta chỉ quan tâm tới chuyển động của tâm bàn kẹp do vậy từ (3.91) và (3.92) ta có:

$$\begin{cases} x_0 = l(C_{12} + C_1) \\ y_0 = l(S_{12} + S_1) \end{cases} \quad (3.93)$$

$$\begin{aligned} &\Rightarrow \frac{x_0^2 + y_0^2}{l^2} = 2 + 2\cos(\theta_2) \\ &\Rightarrow \frac{x_0^2 + y_0^2 - 2l^2}{2l^2} = \cos(\theta_2) \\ &\Rightarrow \theta_2 = \arccos\left(\frac{x_0^2 + y_0^2 - 2l^2}{2l^2}\right) \end{aligned} \quad (3.94)$$

Từ (3.93) ta có:

$$\begin{cases} x_0 - lC_1 = lC_{12} \\ y_0 - lS_1 = lS_{12} \end{cases} \Rightarrow \begin{cases} (x_0 - lC_1)^2 = (lC_{12})^2 \\ (y_0 - lS_1)^2 = (lS_{12})^2 \end{cases} \quad (3.95)$$

$$\Rightarrow x_0^2 + y_0^2 - 2l(x_0 C_1 + y_0 S_1) = 0$$
$$\Rightarrow \frac{\sqrt{x_0^2 + y_0^2}}{2l} = \frac{x_0}{\sqrt{x_0^2 + y_0^2}} C_1 + \frac{y_0}{\sqrt{x_0^2 + y_0^2}} S_1 \quad (3.96)$$

Đặt

$$\begin{cases} \cos(\alpha) = \dfrac{x_0}{\sqrt{x_0^2 + y_0^2}} \\ \sin(\alpha) = \dfrac{y_0}{\sqrt{x_0^2 + y_0^2}} \end{cases} \quad (3.97)$$

Chọn $\theta_1 > \alpha$ ta có:

$$\theta_1 = \arccos\left(\frac{\sqrt{x_0^2 + y_0^2}}{2l} \right) + \alpha \quad (3.98)$$

Như vậy, nếu yêu cầu của bài toán là điều khiển tâm bàn kẹp đi theo một quỹ đạo đã được định trước và được xác định bởi:

$$\begin{cases} x = x(t) \\ y = y(t) \\ z = z(t) = 0 \end{cases}$$

thì giá trị đặt cho các khớp phải là:

$$\begin{cases} \theta_1 = \arccos\left(\dfrac{\sqrt{x_0^2 + y_0^2}}{2l} \right) + \alpha \\ \theta_2 = \arccos\left(\dfrac{x_0^2 + y_0^2 - 2l^2}{2l^2} \right) \end{cases} \quad (3.99)$$

CHƯƠNG IV: MÔ PHỎNG QUÁ TRÌNH CHUYỂN ĐỘNG CỦA ROBOT DÙNG BỘ ĐIỀU KHIỂN TRƯỢT TRÊN NỀN MATLAB AND SIMULINK:

IV.1. Tổng quan về Matlab-Simulink:

Matlab là một bộ chương trình phần mềm lớn của lĩnh vực toán số. Tên của bộ chương trình chính là từ viết tắt của từ Matrix Laboratory, thể hiện định hướng chính của chương trình là các phép tính vectơ và ma trận. Phần cốt lõi của chương trình bao gồm một số hàm toán, các chức năng xuất nhập cũng như các khả năng điều khiển chu trình mà nhờ đó ta có thể dựng nên các Scripts.

Thêm vào phần cốt lõi, có thể dùng các bộ công cụ Toolbox với phạm vi chức năng chuyên dụng mà người sử dụng cần. Simulink là một Toolbox có vai trò đặc biệt quan trọng: vai trò của một bộ công cụ mạnh phục vụ mô hình hoá và mô phỏng các hệ thống kĩ thuật - Vật lý, trên cơ sở sơ đồ cấu trúc dạng khối.

Giao diện đồ họa trên màn hình của Simulink cho phép thể hiện hệ thống dưới dạng sơ đồ tín hiệu với các khối chức năng quen thuộc. Simulink cung cấp cho người dùng một thư viện rất phong phú, có sẵn với số lượng lớn các khối chức năng cho các hệ tuyến tính, phi tuyến và gián đoạn. Hơn thế người sử dụng có thể tạo nên các khối riêng cho mình.

Sau khi đã xây dựng mô hình của hệ thống cần nghiên cứu, bằng cách ghép các khối cần thiết, thành sơ đồ cấu trúc của hệ, ta có thể khởi động quá trình mô phỏng. Trong các quá trình mô phỏng ta có thể trích tín hiệu hiện tại vị trí bất kì của sơ đồ cấu trúc và hiển thị đặc tính của tín hiệu đó trên màn hình. Hơn thế nữa, nếu có nhu cầu ta còn có thể cất giữ các đặc tính đó vào môi trường nhớ. Việc nhập hoặc thay đổi tham số của tất cả các khối cũng có thể thực hiện được rất đơn giản bằng cách nhập trực tiếp hay thông qua matlab. Để khảo sát hệ thống, ta có thể sử dụng thêm các Toolbox như Signal Processing (xử lý tín hiệu), Optimization (tối ưu) hay Control System (hệ thống điều khiển).

IV.2. Các thao tác thực hiện mô phỏng:

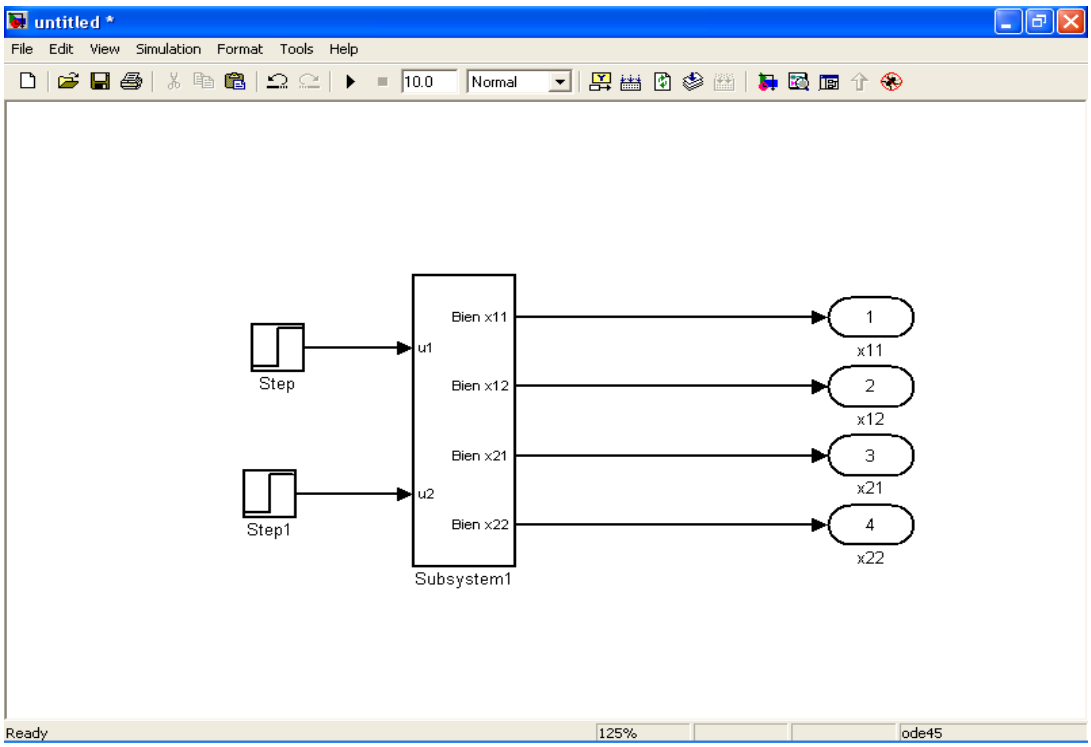
Khớp 1:

$$\begin{cases} \dot{x}_{11} = x_{12} \\ \dot{x}_{12} = \dfrac{1}{4 - 9/4\cos^2 x_{21}}(u_1 + \dfrac{3}{2}x_{22}(2x_{12} + x_{22})\sin x_{21} - 15(\cos x_{11} - \cos x_{21})) - \\ -(1 + \dfrac{3}{2}\cos x_{21})(u_2 - \dfrac{3}{2}x_{12}^2 \sin x_{21} - 15\cos(x_{11} + x_{21})) \end{cases}$$

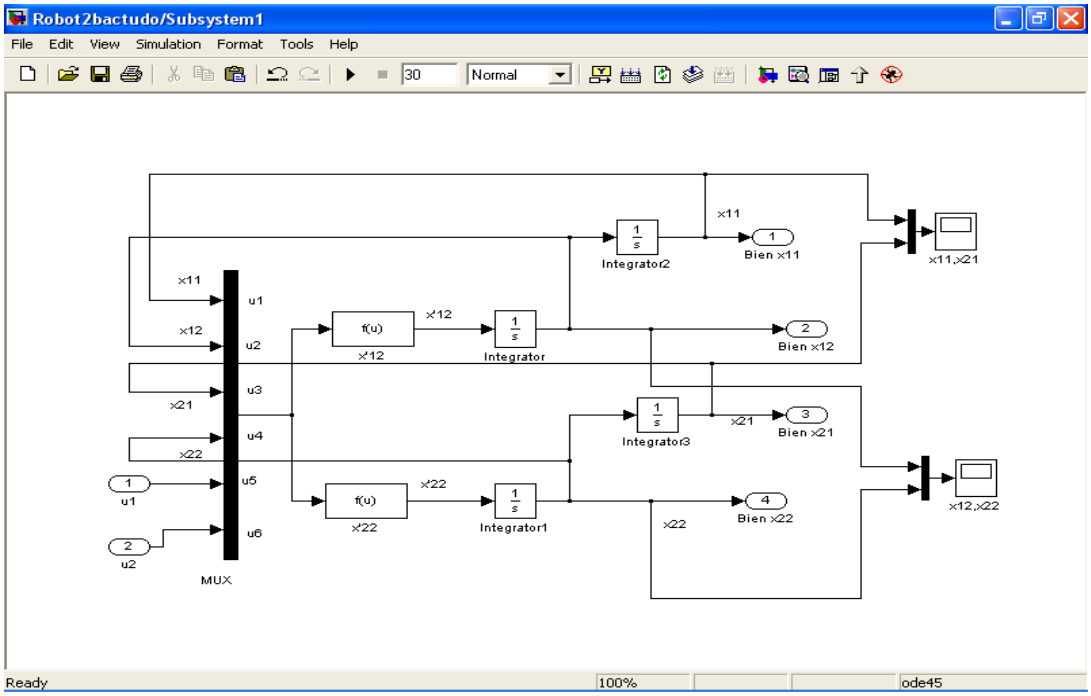
Khớp 2:

$$\begin{cases} \dot{x}_{21} = x_{22} \\ \dot{x}_{22} = \dfrac{1}{4 - 9/4\cos^2 x_{21}}(-(1 + \dfrac{3}{2}\cos x_{21})(u_1 + \dfrac{3}{2}x_{22}(2x_{12} + x_{22})\sin x_{21} - 15(\cos x_{11} - \cos x_{21}))) + \\ +(5 + 3\cos x_{21})(u_2 - \dfrac{3}{2}x_{12}^2 \sin x_{21} - 15\cos(x_{11} + x_{21}))) \end{cases}$$

Ta có mô hình Simulink của Robot 2 bậc tự do:



Khối Subsytem:



Ở đây ta sử dụng các khối:

+ intergrator: khối tích phân với các tham số của khối mặc định cho trước

+ khối mux: chập tín hiệu đơn thành tín hiệu tổng hợp của nhiều tín hiệu

+ khối input, ouput: đầu vào và đầu ra của tín hiệu.

+ khối hàm: biểu diễn 1 hàm toán học khi có tín hiệu đi vào là các biến, tín hiệu ra thu được là hàm cần biểu diễn:

$\dot{x}_{12}$ =(u[5]+1.5*u[4]*sin(u[3])*(2*u[2]+u[4])-15*(cos(u[1])-cos(u[3]))-(1+1.5*cos(u[3]))*(u[6]-1.5*u[2]*u[2]*sin(u[3])-15*cos(u[1]+u[3])))/(4-2.25*cos(u[3])*cos(u[3]))

Và: $\dot{x}_{22}$ =(-(1+1.5*cos(u[3]))*(u[5]+1.5*u[4]*sin(u[3])*(2*u[2]+u[4])-15*(cos(u[1])-cos(u[3])))+(5+3*cos(u[3]))*(u[6]-1.5*u[2]*u[2]*sin(u[3])-15*cos(u[1]+u[3])))/(4-2.25*cos(u[3])*cos(u[3]))

+ Các khối scope (thuộc thư viện con sinks): Hiển thị các tín hiệu của quá trình mô phỏng theo thời gian. Nếu mở cửa sổ Scope sẵn từ trước khi bắt đầu mô phỏng ta có thể theo dõi trực tiếp diễn biến của tín hiệu.

Ta sử dụng nguồn tín hiệu u_1, u_2 là 1(t)

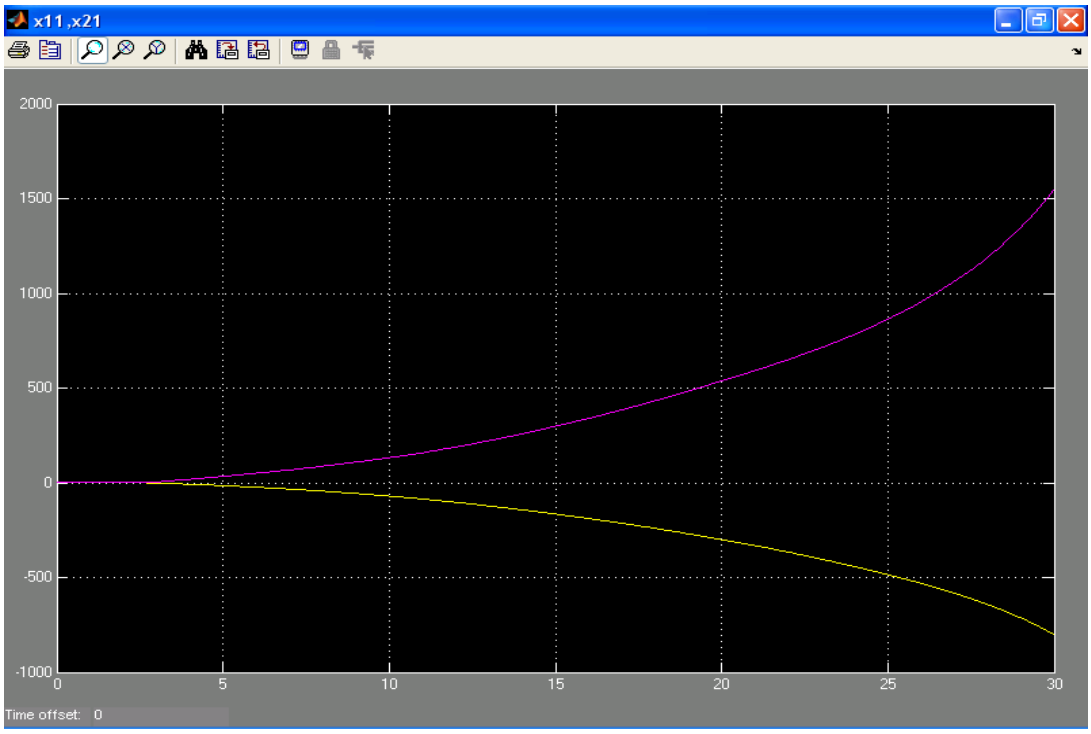
Các hàm x'_{12}, và x'_{22} là :

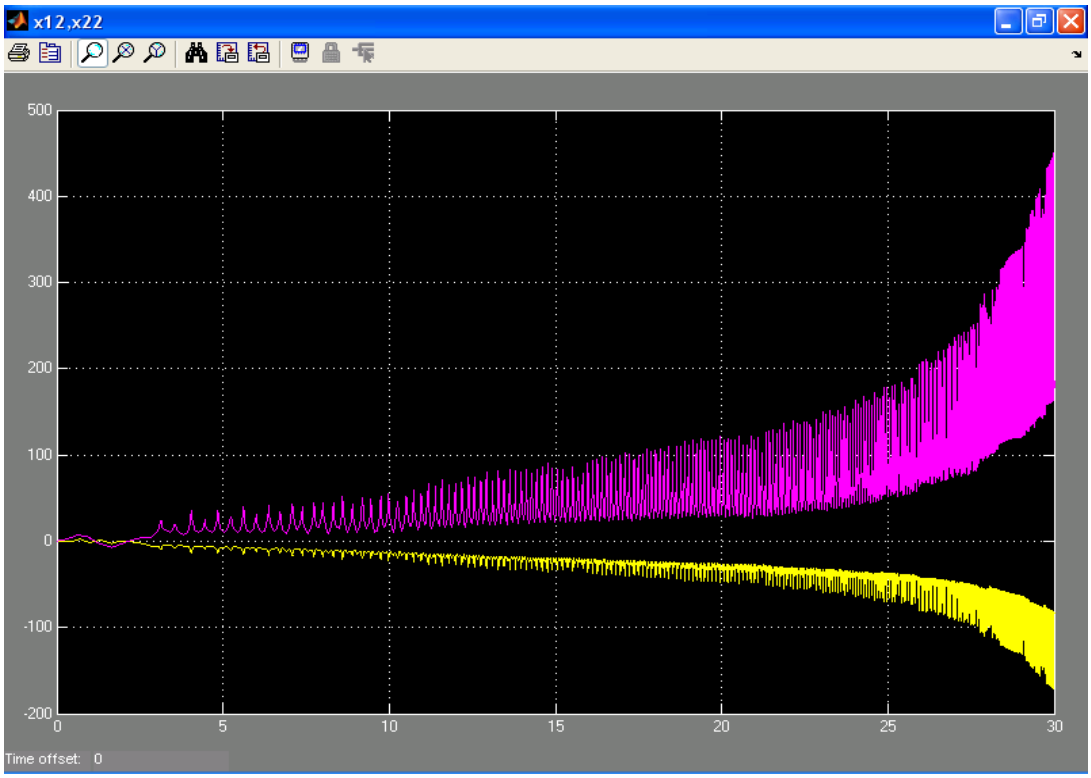
X'_{12} = (u[5]+1.5*u[4]*sin(u[3])*(2*u[2]+u[4])-15*(cos(u[1])-cos(u[3]))-(1+1.5*cos(u[3]))*(u[6]-1.5*u[2]*u[2]*sin(u[3])-15*cos(u[1]+u[3])))/(4-2.25*cos(u[3])*cos(u[3])).

X'_{22} = (-(1+1.5*cos(u[3]))*(u[5]+1.5*u[4]*sin(u[3])*(2*u[2]+u[4])-15*(cos(u[1])-cos(u[3])))+(5+3*cos(u[3]))*(u[6]-1.5*u[2]*u[2]*sin(u[3])-15*cos(u[1]+u[3])))/(4-2.25*cos(u[3])*cos(u[3]))

với u[1], u[2], u[3], u[4], u[5], u[6] tương ứng là các vị trí thứ tự trên khối Mux. u[1] = x_{11}, u[2] = x_{12}, u[3] = x_{21}, u[4] = x_{22}, u[5] = u_1, u[6] = u_2.

Sau khi mô phỏng ta có đồ thị các đường đặc tính của các biến trạng thái x_{11}, x_{12}, x_{21},x_{22} là :





Mô phỏng dạng hàm điều khiển:

Từ các công thức:

$$\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = H\begin{bmatrix} \dfrac{K_1 h(S_1) + \dot{e}_1 + \lambda_1 \ddot{x}_{1d}}{\lambda_1} \\ \dfrac{K_2 h(S_2) + \dot{e}_2 + \lambda_2 \ddot{x}_{2d}}{\lambda_2} \end{bmatrix} + \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix}\begin{bmatrix} x_{12} \\ x_{22} \end{bmatrix} + \begin{bmatrix} g_1 \\ g_2 \end{bmatrix}$$

- Ma trận H:

$$h_{11} = 5 + 3\cos\theta_2$$
$$h_{12} = h_{21} = 1 + 3/2\cos\theta_2$$
$$h_{22} = 1$$

- Ma trận C:

$$c_{11} = -3\dot{\theta}_2 \sin\theta_2$$
$$c_{12} = -3/2\dot{\theta}_2 \sin\theta_2$$
$$c_{21} = -3\dot{\theta}_1 \sin\theta_2$$
$$c_{22} = 0$$

- Ma trận G:

$$g_1 = 15\cos\theta_1 - 15\cos(\theta_1 + \theta_2)$$
$$g_2 = 15\cos(\theta_1 + \theta_2)$$

Ta có:

$$U1=h_{11}\frac{K_1 h(S_1) + \dot{e}_1 + \lambda_1 \ddot{x}_{1d}}{\lambda_1} + h_{12}\frac{K_2 h(S_2) + \dot{e}_2 + \lambda_2 \ddot{x}_{2d}}{\lambda_2}$$

$+c_{11}x_{12}+c_{12}x_{22}+15\cos(x_{11})$

$-15sos(x_{11}+x_{21})$

$$U_2=h_{21}\frac{K_1 h(S_1) + \dot{e}_1 + \lambda_1 \ddot{x}_{1d}}{\lambda_1} + h_{22}\frac{K_2 h(S_2) + \dot{e}_2 + \lambda_2 \ddot{x}_{2d}}{\lambda_2} + c_{21}x_{12}+c_{22}x_{22}$$

$+15\cos(x_{11}+x_{21})$

Chuyển về dạng hàm của sơ đồ Simulink:

U_1=u[5]*(5+cos(u[3]))+(1+1.5*cos(u[3]))*u[6]-3*u[2]*u[3]*sin(u[3])

-1.5*u[3]*u[4]*sin(u[3])+15*cos(u[1])-15*cos(u[1]+u[3])

U_2=(1+1.5*cos(u[3]))*u[5]+u[6]-3*u[1]*u[2]*sin(u[3])+15*cos(u[1]+u[3])

với u[1]=x_{11}, u[2]=x_{12}, u[3]=x_{21}, u[4]=x_{22}.

$$U[5]= \frac{K_1 h(S_1) + \dot{e}_1 + \lambda_1 \ddot{x}_{1d}}{\lambda_1}$$

$$U[6]= \frac{K_2 h(S_2) + \dot{e}_2 + \lambda_2 \ddot{x}_{2d}}{\lambda_2}$$

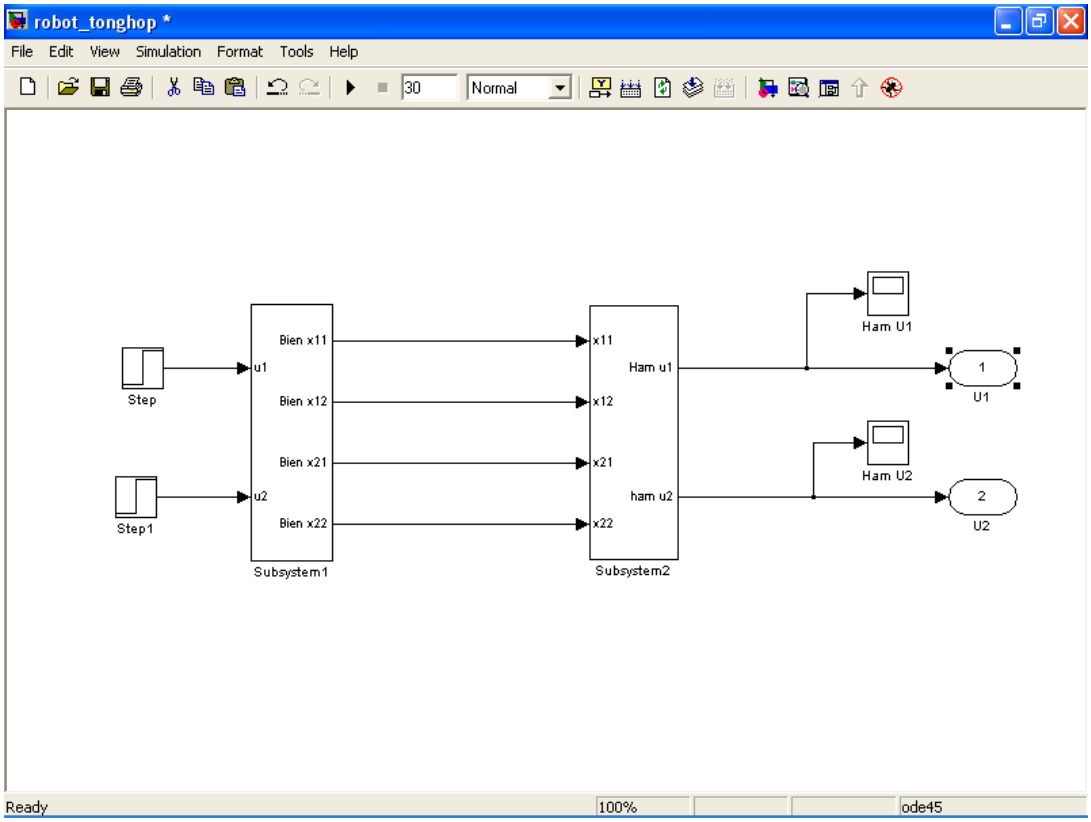
Theo công thức kinh nghiệm ta chọn: $K_1=K_2=500$, $\lambda_1=\lambda_2=0{,}156$.

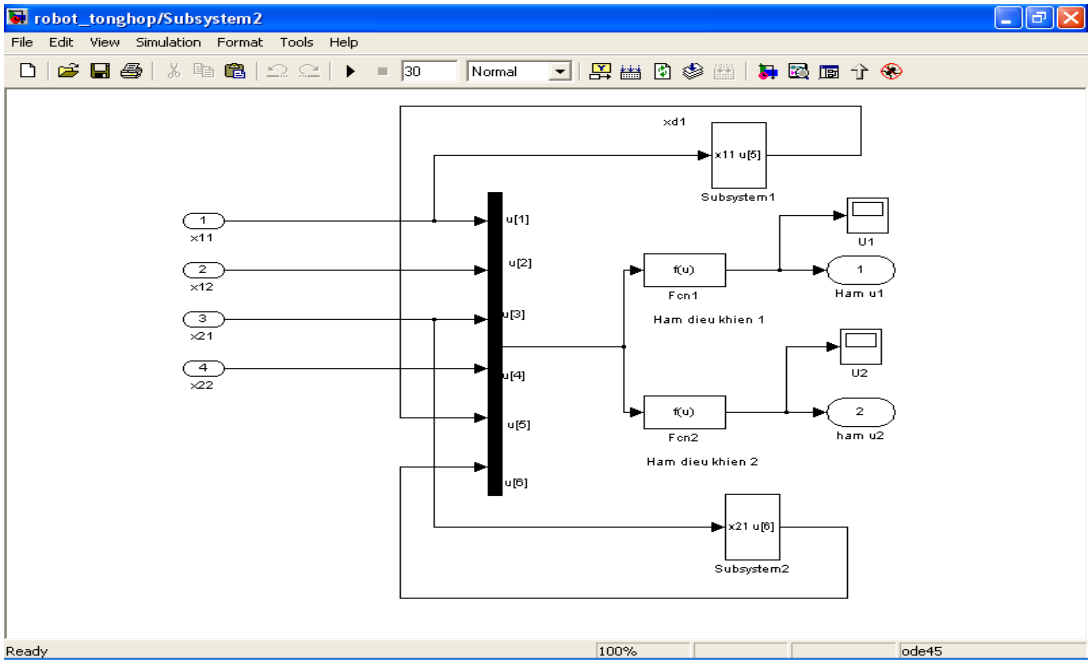
Trường hợp này ta dùng hàm $H(S_1)$,$H(S_2)$ là các hàm giới hạn đầu vào trong khoảng giá trị upper và giá trị lower.

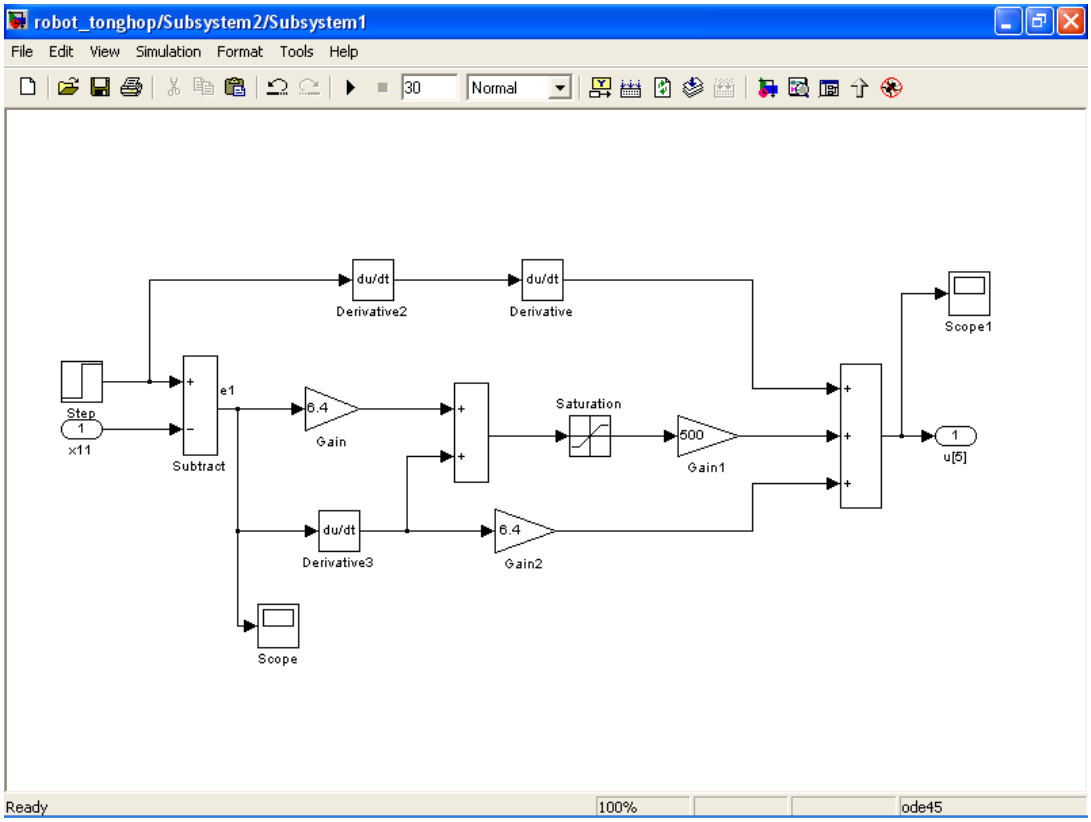
từ đó ta có:

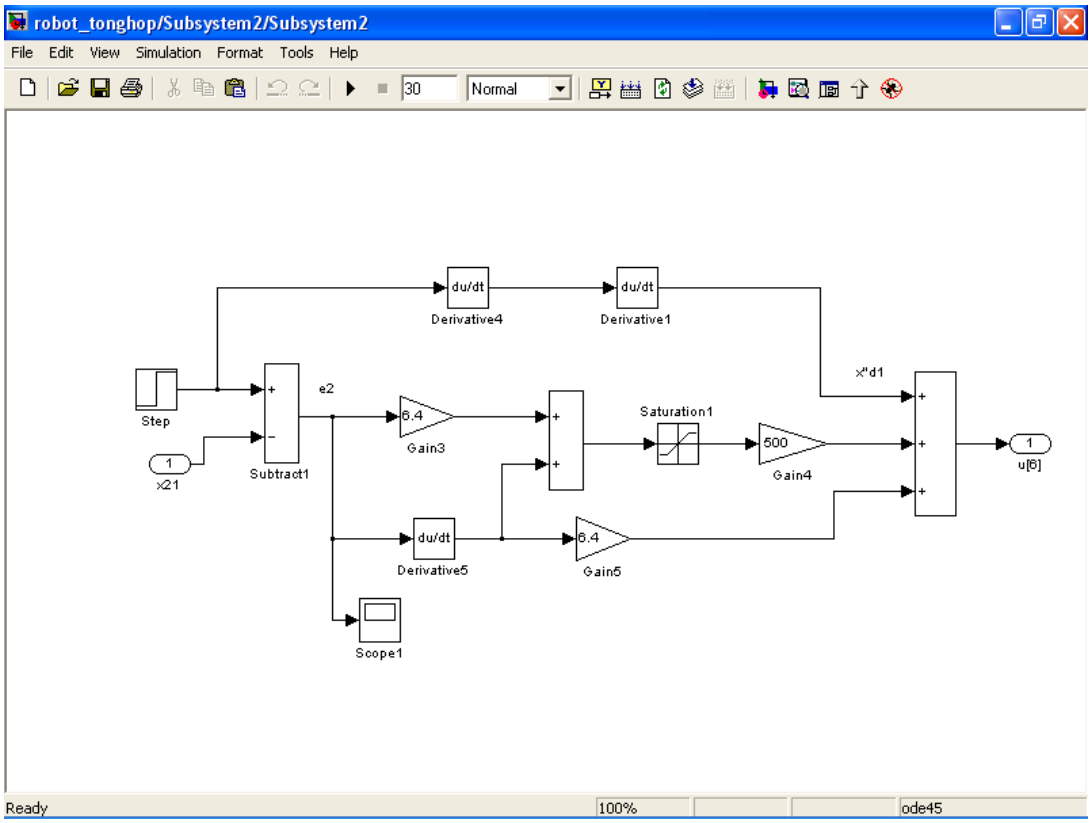
$$u[5]= \frac{K_1}{\lambda_1}H(S_1)+ \frac{1}{\lambda_1}\dot{e}_1 + \ddot{x}_{1d}$$ (Khối subsytem)

$$u[6]=\frac{K_2}{\lambda_2}H(S_2)+\frac{1}{\lambda_2}\dot{e}_2+\ddot{x}_{2d}$$ (Khối subsytem2)






robot_tonghop/Subsystem2/Subsystem1
File Edit View Simulation Format Tools Help
30
Normal
du/dt
Derivative2
du/dt
Derivative
Scope1
Step
1
x11
e1
Subtract
6.4
Gain
Saturation
500
Gain1
1
u[5]
du/dt
Derivative3
6.4
Gain2
Scope
Ready
100%
ode45



Ta chọn khoảng thời gian mô phỏng là từ 0 ->30 s , tức giá trị stop time = 30 ở trong Congiguration Parameters.

Khi chay sơ đồ Simulink ta được các kết quả đường đặc tuyến của 2 hàm điều khiển U_1, U_2, và sai số $\dot{e}_1$, $\dot{e}_2$ là:

Ta có các giá trị đặt x_{d1} và x_{d2} là:

Từ các công thức:

$$\begin{cases} x_0 = l(C_{12} + C_1) \\ y_0 = l(S_{12} + S_1) \end{cases}$$

$$\begin{cases} \cos(\alpha) = \dfrac{x_0}{\sqrt{x_0^2 + y_0^2}} \\ \sin(\alpha) = \dfrac{y_0}{\sqrt{x_0^2 + y_0^2}} \end{cases}$$

$$\begin{cases} \theta_1 = \arccos\left(\dfrac{\sqrt{x_0^2 + y_0^2}}{2l}\right) + \alpha \\ \theta_2 = \arccos\left(\dfrac{x_0^2 + y_0^2 - 2l^2}{2l^2}\right) \end{cases}$$

=>x_{d1}=

acos(sqrt((cos(u[1]+u[3])+cos(u[1]))^2+(sin(u[1]+u[3])+sin(u[1]))^2)/2)+
acos((cos(u[1]+u[3])+cos(u[1]))/sqrt((cos(u[1]+u[3])+cos(u[1]))^2+(sin(u[1])+sin(u[1]
+u[3]))^2))

x_{d2} = acos(((cos(u[1]+u[3])+cos(u[1]))^2+(sin(u[1]+u[3])+sin(u[1]))^2-2)/2)

Kết luận

Các vấn đề đã được giải quyết :

Các vấn đề còn tồn tại :

Tài liệu tham khảo :

1. Nguyễn Thiện Phúc: **Robot công nghiệp**. NXB KHKT 2004.

2. Nguyễn Thiện Phúc: **Người máy công nghiệp và sản xuất tự động linh hoạt**. NXB KHKT 1991.

3. Nguyễn Phùng Quang: **Điều khiển Robot công nghiệp - Những vấn đề cần biết**. Tạp chí Tự động hoá ngày nay - số tháng 4, 5, 6 / 2006.

4. Nguyễn Thiện Phúc: **Robot - Thế giới công nghệ cao của bạn**. NXB KHKT 2005.

5. Đào Văn Hiệp: **Kỹ thuật Robot**. NXB KHKT 2002.

6. Đinh Gia Tường, Tạ Khánh Lâm: **Nguyên lý máy**. NXB GD 2003.

7. Nguyễn Hồng Thái: **Xây dựng thuật toán điều khiển và mô phỏng động Robot nhiều bậc tự do**. Thư viện ĐHBK HN 2002.

8. Lê Huy Tùng: **Điều khiển trượt thích nghi động cho đối tượng phi tuyến có mô hình không tường minh**. Thư viện ĐHBK HN 2003.

9. Nguyễn Doãn Phước, Phan Xuân Minh, Hán Thành Trung: **Lý thuyết điều khiển phi tuyến**. NXB KHKT 2003.

10. Nguyễn Doãn Phước, Phan Xuân Minh: **Hệ phi tuyến**. NXB KHKT 2000.

11. Nguyễn Thương Ngô: **Lý thuyết điều khiển tự động hiện đại**. NXB KHKT 2003.

12. Nguyễn Thương Ngô: **Lý thuyết điều khiển thông thường và hiện đại**. NXB KHKT 2002.

13. Nguyễn Doãn Phước**: Lý thuyết điều khiển tuyến tính**. NXB KHKT 2002.

14. Nguyễn Phùng Quang: **Matlab & Simulink dành cho kỹ sư điều khiển tự động**. NXB KHKT 2004.

15. Nguyễn Hoàng Hải: **Lập trình Matlab**. NXB KHKT 2003.

16. Solomon: **Stability of nonlinear control systems**. 1965.

17. **Applied Asymptotic Methods in Nonlinear Oscillitions**. Thư viện ĐHBK HN 1994.

18. Harry: **Nonlinear Modulation Theory**. 1971.

19. Jakub Mozaryn, Jerzt E.Kurek: **Design of the Sliding Mode Control for the Puma 560 Robot**. Institute of Automatic Control and Robotics Warsaw university of Technology.

20. Martin Ansbjerg Kjaer: **Sliding Mode Control**. Sweden February 6th 2004.

21. C.Abdallah, D.Dawson, P.Dorato, and M.Jamshidi: **Survey of Robust Control for Rigid Robots** .

22. Mark W.Spong: **Motion control of Robot Manipulators**. The coordinated Science Laboratory, University of Illinois at Urbana-Champaign, 1308 W. Main St, Urbana, III. 61801 USA.

23. . Andre` Jaritz and Mark W. Spong: **An Experimental Comparison of Robust control Algorithms on a Direct Drive Manipulator** .IEEE Transaction on Control Systems Technology, Vol. 4, No. 6, November 1996.

24. Austin Blaquie`re: **Nonlinear system analysis**. Academic Press New York and London 1966.