ĐẠI HỌC THÁI NGUYÊN
KHOA CÔNG NGHỆ THÔNG TIN

ĐINH THỊ THUÝ QUỲNH

ỨNG DỤNG MẠNG NƠRON TRONG BÀI TOÁN XÁC ĐỊNH LỘ TRÌNH CHO ROBOT

LUẬN VĂN THẠC SĨ CÔNG NGHỆ THÔNG TIN

THÁI NGUYÊN - 2008

ĐẠI HỌC THÁI NGUYÊN
KHOA CÔNG NGHỆ THÔNG TIN

ĐINH THỊ THUÝ QUỲNH

ỨNG DỤNG MẠNG NƠRON TRONG BÀI TOÁN XÁC ĐỊNH LỘ TRÌNH 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

DANH MỤC HÌNH

LỜI NÓI ĐẦU

Nhờ các khả năng: Học, nhớ lại và khái quát hoá từ các mẫu huấn luyện hoặc dữ liệu, mạng nơron nhân tạo trở thành một phát minh mới đầy hứa hẹn của hệ thống xử lý thông tin. Các tính toán nơron cho phép giải quyết tốt những bài toán đặc trưng bởi một số hoặc tất cả các tính chất sau: Sử dụng không gian nhiều chiều, các tương tác phức tạp, chưa biết hoặc không thể theo dõi về mặt toán học giữa các biến. Ngoài ra phương pháp này còn cho phép tìm ra nghiệm của những bài toán đòi hỏi đầu vào là các cảm nhận của con người như: tiếng nói, nhìn và nhận dạng...

Bài toán lập lộ trình cho robot là một bài toán khá phức tạp, do khi tồn tại và hành động trong môi trường robot sẽ phải chịu rất nhiều sự tác động khác nhau. Tuy nhiên, các tính toán nơron lại cho phép giải quyết tốt các bài toán có nhiều tương tác phức tạp. Vì vậy, ứng dụng mạng nơron trong bài toán xác định lộ trình cho robot sẽ hứa hẹn là một giải pháp hiệu quả góp phần nâng cao hiệu năng làm việc của robot nhờ khả năng di chuyển nhanh chóng, chính xác trong các môi trường làm việc của mình.

Trên thế giới, đã có một số nghiên cứu ứng dụng mạng nơron trong bài toán lập lộ trình cho robot. Tuy nhiên, lĩnh vực này còn khá mới mẻ và chưa được ứng dụng rộng rãi ở nước ta. Trong nước cũng chưa có một tài liệu chính thống nào về lĩnh vực này. Với những ứng dụng ngày càng rộng rãi của công nghệ robot, việc nghiên cứu và áp dụng những thành tựu mới của công nghệ thông tin vào thiết kế và cải tiến các kỹ năng trong đó có kỹ năng tránh các vật cản khi di chuyển là một trong những vấn đề nóng đang rất được quan tâm. Chính vì những lý do trên em đã quyết định chọn đề tài: **"Ứng dụng mạng nơron trong bài toán xác định lộ trình cho robot"** Với mục đích tìm

hiểu về mạng nơron nhân tạo và bài toán lập lộ trình cho robot, ứng dụng mạng nơron vào bài toán trên.

Luận văn gồm 3 chương với các nội dung cơ bản sau:

Chương 1: Trình bày tổng quan về cơ sở của mạng nơron nhân tạo, và nêu khái quát những ứng dụng của mạng nơron trong công nghệ robot.

Chương 2: Trình bày: bài toán lập lộ trình và những thành phần của nó, không gian cấu hình, cấu hình chướng ngại vật.

Chương 3: Trình bày: hương pháp lập lộ trình của Yang và Meng, cải tiến mô hình nguyên bản do Yang và Meng đề xuất, cài đặt thử nghiệm hai mô hình đã trình bày, đưa ra những nhận xét về hiệu quả của hai mô hình đó.

Mặc dù đã hết sức nỗ lực, song do thời gian và kinh nghiệm nghiên cứu khoa học còn hạn chế nên không thể tránh khỏi những thiếu sót. Em rất mong nhận được sự góp ý của các thầy cô và bạn bè đồng nghiệp để hiểu biết của mình ngày một hoàn thiện hơn.

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 hướng, hướng 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

Đinh Thị Thuý Quỳnh

CHƯƠNG I

TỔNG QUAN MẠNG NƠRON NHÂN TẠO

1.1. GIỚI THIỆU MẠNG NƠRON

1.1.1 Những kiến trúc tính toán

Khái niệm tính toán có thể được hiểu theo nhiều cách. Trước đây, việc tính toán bị ảnh hưởng bởi quan niệm tính toán theo chương trình (Programed computing). Theo quan điểm này, để giải quyết bài toán thì bước đầu tiên ta cần thiết kế giải thuật sau đó cài đặt giải thuật đó trên cấu trúc hiện hành có ưu thế nhất.

Quan sát các hệ sinh học, đặc biệt là bộ não người ta thấy chúng có những đặc điểm sau:

(1) Bộ não tích hợp và lưu trữ kinh nghiệm: Tức là bộ não có khả năng tự phân loại và liên kết các dữ liệu vào.

(2) Bộ não xem xét kinh nghiệm mới dựa trên những kinh nghiệm đã lưu trữ.

(3) Bộ não có khả năng dự đoán chính xác những tình huống mới dựa trên những kinh nghiệm tự tổ chức trước đây.

(4) Bộ não không yêu cầu thông tin hoàn hảo.

(5) Bộ não thể hiện một kiến trúc chấp nhận lỗi tức là có thể khôi phục sự mất đi của một vài noron bằng cách thích nghi với noron còn lại hoặc bằng cách đào tạo bổ xung.

(6) Cơ chế hoạt động của bộ não đôi khi không rõ ràng trong vận hành. Ví dụ với một số bài toán chúng ta có thể cung cấp nghiệm nhưng không thể giải thích được các bước tìm nghiệm.

(7) Bộ não có khuynh hướng đưa ra những giải pháp trong một trạng thái cân bằng hoặc có khuynh hướng dẫn đến trạng thái đó

Từ đó ta nhận thấy, tính toán dựa trên các hệ sinh học khác với tính toán theo chương trình ở các đặc điểm sau:

- Quá trình tính toán được tiến hành song song và phân tán trên nhiều noron
- Tính toán thực chất là quá trình học chứ không phải theo một sơ đồ định sẵn từ trước.

Dựa trên nhữnh đặc điểm này một phương pháp tính toán mới có nền tảng từ sinh học là mạng noron nhân tạo (Artifical Neural Networks_ ANNs) đã ra đời và có tiềm năng trở thành kiến trúc tính toán chiếm ưu thế.

1.1.2 Lịch sử phát triển của mạng noron.

Mạng noron nhân tạo được xây dựng từ những năm 1940 nhằm mô phỏng một số chức năng của bộ não người. Dựa trên quan điểm cho rằng bộ não người là bộ điều khiển. Mạng noron nhân tạo được thiết kế tương tự như noron sinh học sẽ có khả năng giải quyết hàng loạt các bài toán như tính toán tối ưu, điều khiển, công nghệ robot…

Quá trình nghiên cứu và phát triển noron nhân tạo có thể chia thành 4 giai đoạn như sau:

- Giai đoạn 1: Có thể tính từ nghiên cứu của William (1890) về tâm lý học với sự liên kết các noron thần kinh. Năm 1940 Mc Culloch và Pitts đã cho biết noron có thể mô hình hoá như thiết bị ngưỡng (Giới hạn) để thực hiện các phép tính logic và mô hình mạng noron của Mc Culloch – Pitts cùng với giải thuật huấn luyện mạng của Hebb ra đời năm 1943.

- Giai đoạn 2: vào khoảng gần những năm 1960, một số mô hình noron hoàn thiện hơn đã được đưa ra như: Mô hình Perceptron của Rosenblatt (1958), Adalile của Widrow (1962). Trong đó mô hình Perceptron rất được quan tâm vì nguyên lý đơn giản, nhưng nó cũng có hạn chế vì như Marvin Minsky và Seymour papert của MIT (Massachurehs Insritute of Technology) đã chứng minh nó không dùng được cho các hàm logic phức (1969). Còn Adaline là mô hình tuyến tính, tự chỉnh, được dùng rộng rãi trong điều khiển thích nghi, tách nhiễu và phát triển cho đến nay.

- Giai đoạn 3: Có thể tính vào khoảng đầu thập niên 80. Những đóng góp lớn cho mạng noron trong giai đoạn này phải kể đến Grossberg, Kohonen, Rumelhart và Hopfield. Trong đó đóng góp lớn của Hopfield gồm hai mạng phản hồi: Mạng rời rạc năm 1982 và mạng liên tục năm 1984. Đặc biệt, ông đã dự kiến nhiều khả năng tính toán lớn của mạng mà một nơron không có khả năng đó. Cảm nhận của Hopfield đã được Rumelhart, Hinton và Williams đề xuất thuật toán sai số truyền ngược nổi tiếng để huấn luyện mạng noron nhiều lớp nhằm giải bài toán mà mạng khác không thực hiện được. Nhiều ứng dụng mạnh mẽ của mạng noron ra đời cùng với các mạng theo kiểu máy Boltzmann và mạng Neocognition của Fukushima.

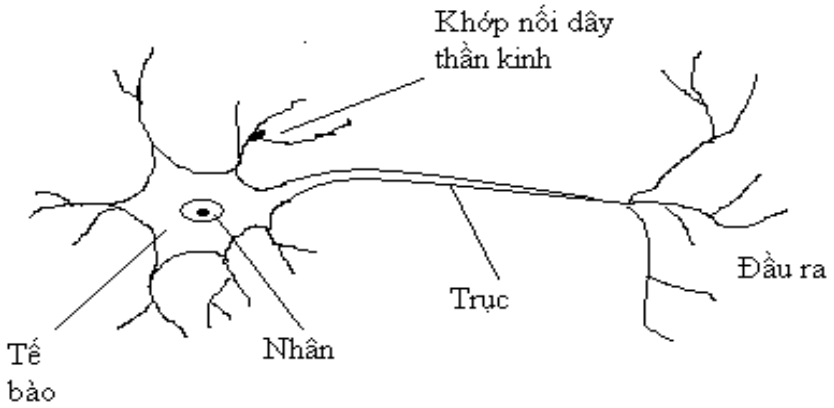
- Giai đoạn 4: Tính từ năm 1987 đến nay, hàng năm thế giới đều mở hội nghị toàn cầu chuyên ngành nơron IJCNN (International Joit Conference on Neural Networks). Rất nhiều công trình được nghiên cứu để ứng dụng mạng nơron vào các lĩnh vực như: Kỹ thuật tính, điều khiển, bài toán tối ưu, y học, sinh học, thống kê, giao thông, hoá học,...Cho đến nay mạng nơron đã tìm và khẳng định được vị trí của mình trong rất nhiều ứng dụng khác nhau.

1.1.3. Nơron sinh học.

Hệ thần kinh gồm hai lớp tế bào: Nơron (tế bào thần kinh) và glia (tế bào glia). Nơron là thành phần cơ bản của hệ thần kinh, chúng có chức năng xử lý thông tin. Glia thực hiện chức năng hỗ trợ. Vì vậy trước khi nghiên cứu về nơron nhân tạo chúng ta sẽ trình bày khái quát về cấu tạo và hoạt động của nơron sinh học.

Nơro sinh học có nhiều loại, chúng khác nhau về kích thước và khả năng thu phát tín hiệu. Tuy nhiên chúng có cấu trúc và nguyên lý hoạt động chung như sau:

Mỗi nơron sinh học gồm có 3 thành phần: Thân nơron với nhân ở bên trong (soma), một đầu dây thần kinh ra (axon) và một hệ thống phân nhánh hình cây (Dendrite) để nhận các thông tin vào. Trong thực tế có rất nhiều dây thần kinh vào và chúng bao phủ một diện tích rất lớn ($0,25mm^2$). Đầu dây thần kinh ra được rẽ nhánh nhằm chuyển giao tín hiệu từ thân nơron tới nơron khác. Các nhánh của đầu dây thần kinh được nối với các khớp thần kinh (synapse). Các khớp thần kinh này được nối với thần kinh vào của các nơron khác. Các nơron có thể sửa đổi tín hiệu tại các khớp. Hình ảnh đơn giản của một nơron thể hiện trong hình 1.1.



Hình 1.1. Mô hình nơron sinh học

Hoạt động của nơron sinh học có thể được mô tả như sau:

Mỗi nơron nhận tín hiệu vào từ các tế bào thần kinh khác. Chúng tích hợp các tín hiệu vào, khi tổng tín hiệu vượt quá một ngưỡng nào đó chúng tạo tín hiệu ra và gửi tín hiệu này tới các nơron khác thông qua dây thần kinh. Các nơron liên kết với nhau thành mạng. Mức độ bền vững của các liên kết này xác định một hệ số gọi là trọng số liên kết.

1.1.4. Nơron nhân tạo.

Mô phỏng nơron sinh học, ta có nơron nhân tạo. Mỗi nơron có rất nhiều dây thần kinh vào, nghĩa là mỗi nơron có thể tiếp nhận đồng thời nhiều dữ liệu. Giả sử nơron i có N tín hiệu đầu vào, mỗi tín hiệu vào S_j được gán một trọng số w_{ij} tương ứng. Ta có thể ước lượng tổng tín hiệu đầu vào đi vào nơron (net_i) theo một số dạng sau:

(i) Dạng truyến tính

$$net_i = \sum_{i=1}^{N} w_{ij} S_j \qquad (1.1)$$

(ii) Dạng toàn phương

$$net_i = \sum_{i=1}^{N} w_{ij} S_j^2 \qquad (1.2)$$

(iii) Dạng mặt cầu

$$net_i = p^2 \sum_{i=1}^{N} w_{ij} (S_j - w_{ij})^2 \qquad (1.3)$$

Trong đó p và w_{ij} lần lượt là bán kính và tâm cầu.

- Hàm kích hoạt.

Hàm biến đổi tín hiệu đầu vào net cho tín hiệu đầu ra out được gọi là hàm kích hoạt. Hàm này có đặc điểm là không âm và bị chặn. Có nhiều dạng

hàm kích hoạt. Người ta thường sử dụng một hàm kích hoạt chung cho toàn mạng.

Một số hàm kích hoạt thường được sử dụng.

+ Hàm McCuloch-Pitts

$$out = f(net) = \begin{cases} 1 \text{ nếu } net \geq \theta \\ 0 \text{ nếu } net < \theta \end{cases} \qquad (1.4)$$

Trong đó θ là ngưỡng.

+ Hàm McCuloch-Pitts trễ

$$out = f(net) = \begin{cases} 1 \text{ nếu } net \geq UTP \\ 0 \text{ nếu } net < LTP \\ f(net) \text{ nếu khác} \end{cases} \qquad (1.5)$$

Trong đó UTP>LTP

UTP là ngưỡng trên (Upper Trip Point)

LTP là ngưỡng dưới (Lower Trip Point)

+ Hàm Sigmoid.

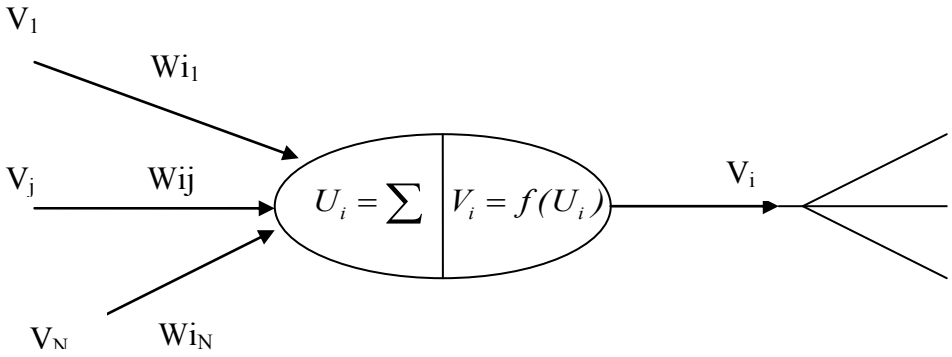
$$out = f(net) = \frac{1}{1 + e^{-\lambda(net+0)}} \qquad (1.6)$$

Trong đó λ >0 là hằng số xác định độ nghiêng của hàm.

- Nút bias:

Là một nút thêm vào nhằm làm tăng khả năng thích nghi của mạng nơron trong quá trình học. Trong các mạng nơron có sử dụng bias, mỗi nơron có thể có một trọng số tương ứng với bias. Trọng số này luôn có giá trị là 1.

Mô hình của một nút xử lý (nút thứ i):



Hình 1.2. Mô hình một nơron nhân tạo.

$$U_i = \sum_{\substack{i=1 \\ j \neq i}}^{N} W_{ij} V_j + \theta \qquad (1.7)$$

$$V_i = f_i(U_i) \qquad (1.8)$$

Trong đó:

U_i là tổng tín hiệu vào tại nơron i.

V_i là tín hiệu ra tại nơron i.

W_{ij} là trọng số liên kết từ nơron i đến nơron j

θ_i là ngưỡng (đầu vào ngoài) kích hoạt nơron i

f_i là hàm kích hoạt của nơron i

1.1.5. Mạng nơron nhân tạo

Mạng nơron nhân tạo (Artificial Neural Network) là một cấu trúc mạng được hình thành nên bởi một số lượng lớn các nơron nhân tạo liên kết với nhau. Mỗi nơron có các đặc tính đầu vào, đầu ra và thực hiện một chức năng tính toán cục bộ.

Với việc giả lập các hệ thống sinh học, các cấu trúc tính toán mạng nơron có thể giải quyết được lớp các bài toán nhất định như: bài toán lập lịch, bài toán tìm kiếm, bài toán nhận dạng mẫu, bài toán xếp loại,... Mạng nơron còn giải quyết được lớp các bài toán sử dụng dữ liệu không đầy đủ, xung đột mờ hoặc xác suất. Những bài toán này được đặc trưng bởi một số hoặc tất cả các tính chất sau: Sử dụng không gian nhiều chiều, các tương tác phức tạp, chưa biết hoặc không thể theo dõi về mặt toán học giữa các biến; không gian nghiệm có thể rỗng, có nghiệm duy nhất hoặc có một số nghiệm bình đẳng như nhau. Ngoài ra, mạng nơron nhân tạo còn thích hợp để tìm nghiệm của những bài toán đòi hỏi đầu vào là những cảm nhận bởi con người như: Tiếng nói, nhìn và nhận dạng,... Tuy nhiên việc ánh xạ từ một bài toán bất kỳ sang một giải pháp mạng nơron lại là một việc không đơn giản.

Mạng nơron là một cấu trúc xử lý song song, thông tin phân tán và có các đặc trưng nổi bật sau:

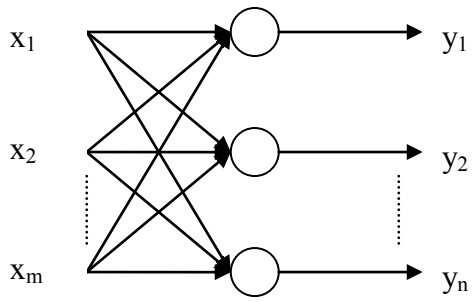
- Là mô hình toán học dựa trên bản chất của nơron sinh học.
- Bao gồm một số lượng lớn các nơron liên kết với nhau.
- Mạng nơron có khả năng học, khái quát hoá tập dữ liệu học thông qua việc gán và hiệu chỉnh các trọng số liên kết.
- Tổ chức theo kiểu tập hợp mang lại cho mạng nơron khả năng tính toán rất lớn, trong đó không có nơron nào mang thông tin riêng biệt.

Mạng nơron nhân tạo có một số mô hình thông dụng sau:

a. Mạng truyền thẳng:

- Mạng truyền thẳng một lớp: Là mô hình liên kết cơ bản và đơn giản nhất. Các nơron tổ chức lại với nhau tạo thành một lớp, tín hiệu được truyền theo một hướng nhất định nào đó. Các đầu vào được nối với các nơron theo trọng

số khác nhau, sau quá trình xử lý cho ra một chuỗi các tín hiệu ra. Nếu mạng là mô hình LTU thì nó được gọi là mạng perception, còn mạng nơron theo mô hình LGU thì được gọi là Adaline.



Hình 1.3. Mô hình mạng truyền thẳng một lớp

Với mỗi giá trị đầu vào x =[x_1, x_2, ..., x_m]^T qua quá trình xử lí của mạng sẽ thu được một bộ đầu ra tương ứng y =[y_1, y_2,..., y_n]^T với phương pháp xác định như sau:

$$y_i = f_i(\sum_{j=1}^{m} w_{ij}x_j - \theta_i) \qquad i = \overline{1,n} \qquad (1.9)$$

Trong đó:

m: Số tín hiệu vào.

n: Số tín hiệu ra.

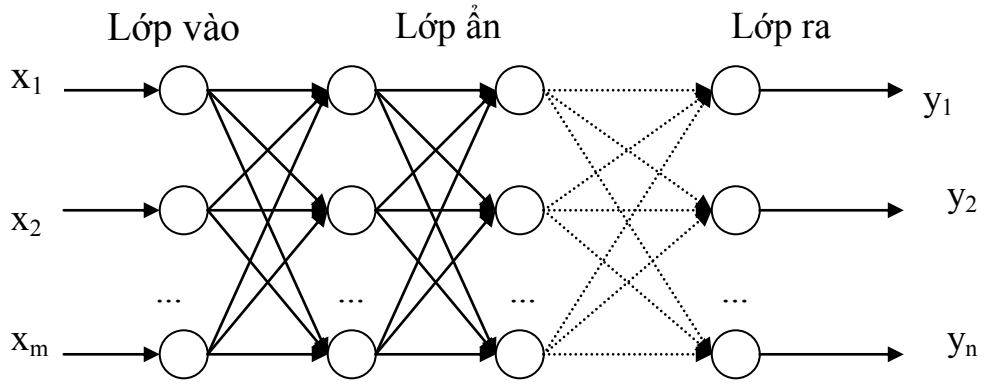
$W_i^T = [w_{i1}, w_{i2}, ..., w_{in}]^T$ là véc tơ trọng số của nơron thứ i.

f_i: là hàm kích hoạt của nơron thứ i.

θ: Là ngưỡng của nơron thứ i.

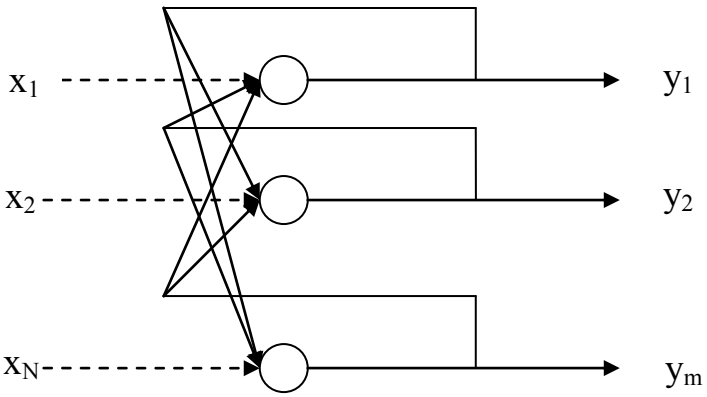
- Mạng truyền thẳng nhiều lớp: Với cấu trúc đơn giản như trên, khi giải quyết các bài toán phức tạp mạng truyền thẳng một lớp sẽ gặp rất nhiều khó

khăn. Để khắc phục nhược điểm này, người ta đưa ra mạng truyền thẳng nhiều lớp. Đây là mạng truyền thẳng gồm nhiều lớp kết hợp với nhau. Lớp nhận tín hiệu gọi là lớp đầu vào (input layer), lớp đưa các tín hiệu ra gọi là lớp đầu ra (output layer), các lớp ở giữa lớp vào và lớp ra gọi là lớp ẩn (hidden layers). Cấu trúc của mạng nơron truyền thẳng nhiều lớp được mô tả trong hình 1.4.



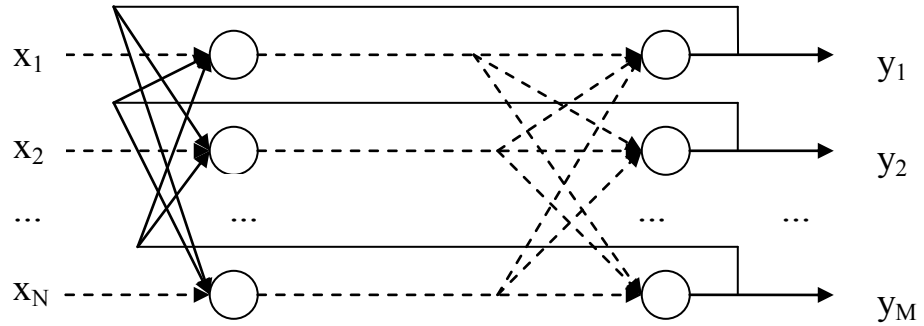
Hình 1.4. Mạng nơron truyền thẳng nhiều lớp.

- Mạng hồi quy.
  - Mạng hồi quy một lớp có nối ngược.



Hình 1.5. Mạng hồi quy một lớp có nối ngược.

  - Mạng hồi quy nhiều lớp có nối ngược.



Hình 1.6. Mạng hồi quy nhiều lớp có nối ngược.

1.1.6. Tiếp cận nơron cho tính toán.

1.1.6.1. Đào tạo và lập trình.

Ngày nay máy tính được ứng dụng rộng rãi trong tất cả các lĩnh vực của đời sống xã hội. Giải quyết một bài toán bằng máy tính cũng có rất nhiều phương pháp khác nhau. Thông thường, thì phương pháp lập trình chiếm ưu thế. Tuy nhiên lập trình đòi hỏi một cú pháp hình thức và một loạt các ngôn ngữ, cũng như kỹ năng của con người. Một giải pháp điển hình để giải quyết vấn đề trong hệ sinh học là ***đào tạo.*** Ví dụ, trẻ con không được "lập trình" nhưng chúng học theo ví dụ và thích nghi. Dĩ nhiên, để tiếp cận đào tạo khả thi, máy tính phải có thể đào tạo được và phải có dữ liệu đào tạo. Một trong những giải pháp để giải quyết vấn đề này là sử dụng mạng nơron. Mạng nơron có những đặc điểm nổi bật sau:

- Các hệ nơron hoạt động như các hệ thông tin có thể đào tạo được, thích nghi và thậm chí tự tổ chức.
- Các mạng nơron phát triển một chức năng dựa trên dữ liệu đào tạo mẫu.

- Các mạng nơron có thể cung cấp những kiến trúc tính toán thông qua đào tạo hơn là thiết kế.

1.1.6.2. Luật học

Các luật học đóng vai trò quan trọng trong việc xác định một mạng nơron nhân tạo. Một cách đơn giản về khái niệm học của mạng nơron là cập nhật các trọng số trên cơ sở các mẫu. Theo nghĩa rộng thì học có thể được chia làm hai loại: Học tham số và học cấu trúc.

a. Học tham số: Các thủ tục học này nhằm tìm kiếm ma trận trọng số sao cho mạng có khả năng đưa ra dự báo sát với thực tế. Dạng chung của luật học tham số có thể được mô tả như sau:

$$\Delta W_{ij} = \eta r x_j, i = \overline{1,N}; j = \overline{1,M}$$

Tron đó:

ΔW_{ij} là sự thay đổi trọng số liên kết từ nơron j đến nơron i.

x_j là tín hiệu vào nơron j.

η là tốc độ học nằm trong khoảng (0,1).

r là hằng số học.

Vấn đề đặt ra ở đây là tín hiệu học r được sinh ra như thế nào để hiệu chỉnh trọng số của mạng.

Có thể chia học tham số ra thành ba lớp nhỏ hơn: Học có chỉ đạo, học tăng cường và học không có chỉ đạo. Việc xác định r phụ thuộc vào từng kiểu học.

- *Học có tín hiệu chỉ đạo:* Là quá trình mạng học dựa vào sai số giữa đầu ra thực và đầu ra mong muốn để làm cơ sở cho việc hiệu chỉnh trọng số. Sai số này chính là hằng số học r. Luật học điển hình của nhóm này là luật

Delta của Widrow(1962) nêu ra dùng để xấp xỉ trọng số của Adaline dựa trên nguyên tắc giảm gradient.

Trong nhóm luật học này cũng cần phải kể đến luật học perceptron của Rosenblatt(1958). Về cơ bản luật học này thay đổi các giá trị trọng số trong thời gian học, còn luật perceptron thì thêm hoặc bỏ trọng số tuỳ theo giá trị sai số là dương hay âm.

Một loạt các luật học khác cũng dựa trên tư tưởng này. Luật Oja là cải tiến và nâng cấp của luật Delta. Luật truyền ngược là mở rộng của luật Delta cho mạng nhiều lớp. Đối với mạng truyền thẳng thường sử dụng luật truyền ngược để chỉnh trọng số với tín hiệu chỉ đạo từ bên ngoài và người ta gọi mạng này là mạng lan truyền ngược.

- *Học không có tín hiệu chỉ đạo:* Luật học này sử dụng đầu ra của mạng làm cơ sở để hiệu chỉnh các trọng số liên kết. Hay trong luật này chính là tín hiệu ra của mạng. Điển hình là luật Hebb(1949) thường dùng cho các mạng tự liên kết. Luật LVQ (learning Vector Quantization) dùng cho mạng tự tổ chức một lớp mạng ánh xạ đặc trưng của Kohonen.

Luật học Hebb là luật sinh học xuất phát từ tiên đề của Hebb cho rằng: Giữa hai nơron có quan hệ và có thay đổi thế năng mạng thì giữa chúng có sự thay đổi trong số liên kết. Nói cách khác trọng số được điều chỉnh theo mối tương quan trước và sau, nghĩa là:

$$\Delta W_{ij} = \eta y_i x_j, i = \overline{1,N}, j = \overline{1,M} \qquad (1.11)$$

Trong đó:

ΔW_{ij} là sự thay đổi trọng số liên kết từ nơron i đến nơron j.

x_j là tín hiệu vào nơron j

y_i là tín hiệu ra của nơron i.

η là tố độ học, nằm trong khoảng(0,1).

Luật Hebb giải thích việc chỉnh trọng số trong phạm vi cục bộ của mạng mà không cần tín hiệu chỉ dạo từ bên ngoài. Hopfield cũng cải tiến luật Hebb cho các mạng tự liên kết thành 16 dạng khác nhau theo kiểu luật Hebb, luật đối Hebb, luật Hopfield,...

Như vậy ứng với mỗi nhóm mạng thường áp dụng một luật học nhất định. Nếu tồn tại hàng chục loại mạng khác nhau thì số luật học có thể tăng lên rất nhiều lần.

Đối với mạng phản hồi thường sử dụng luật Hebb và các luật cải tiến của nó để chỉnh trọng số mà không cần tín hiệu chỉ đạo từ bên ngoài.

- *Học tăng cường:* Trong một số trường hợp, thông tin phản hồi chỉ là tín hiệu bao gồm hai trạng thái cho biết tín hiệu đầu ra của mạng là đúng hay sai. Quá trình học dựa trên các thông tin hướng dẫn như vậy được gọi là học có củng cố (học tăng cường) và tín hiệu mang thông tin phản hồi được gọi là tín hiệu củng cố cho quá trình học. Ta có thể thấy rằng quá trình học này là một dạng của quá trình học có tín hiệu chỉ đạo bởi vì mạng nhận được một số thông tin phản hồi từ bên ngoài.

b. <u>Học cấu trúc:</u> Tìm kiếm các tham số của cấu trúc mạng để tìm ra một cấu trúc mạng hoạt động tốt nhất. Trong thực tế việc học cấu trúc là tìm ra số lớp ẩn và tìm ra số nơron trong mỗi lớp đó. Giải thuật di truyền thường được sử dụng trong các cấu trúc nhưng thường chạy rất lâu, thậm chí ngay cả đối với mạng có kích thước trung bình. Ngoài ra kỹ thuật gọt tỉa mạng hay mạng tăng dần cũng được áp dụng trong việc học cấu trúc của mạng có kích thước tương đối nhỏ.

1.2. PHẠM VI ỨNG DỤNG CỦA MẠNG NƠRON.

1.2.1. Những bài toán thích hợp.

Mạng nơron được coi như một hộp đen để biến đổi véc tơ đầu vào m biến thành vectơ đầu ra n biến. Tín hiệu ra có thể là các tham số thực (tốt nhất nằm trong khoảng [0,1], hoặc [-1,1], số nhị phân 0,1, hay số lưỡng cực -1, +1). Số biến của vectơ ra không hạn chế song sẽ ảnh hưởng tới thời gian tính và tải nguyên liệu của máy tính. Nói chung, các lớp bài toán áp dụng cho nơron có thể phân chia làm 4 loại:

- Phân lớp (clasification).
- Mô hình hoá (modening).
- Biến đổi, thực hiện ánh xạ từ không gian đa biến này vào không gian đa biến khác tương ứng (transformation add mapping).
- Liên kết và kỹ thuật dịch chuyển cửa sổ (asosiation and moving window).

1.2.1.1. Phân loại.

Một trong các công việc đơn giản và thường được sử dụng nhiều trong quản lý các đối tượng đa biến là phân loại (phân lớp một đối tuợng vào các nhóm, nhóm con hay chủng loại). Ví dụ: bài toán phân lớp ảnh, nhận dạng mẫu,...

Khi phải phân loại một quyết định phức tạp, chúng ta phải bắt đầu với việc nghiên cứu, thống kê các mối liên quan giữa nhiều đối tượng. Có thể nói việc xây dựng một cây phân lớp và các quyết định phải được thực hiện trước khi thủ tục học được tiến hành. Nếu kết quả cuối cùng không thoả mãn, chúng ta cần phải xem xét lại cách biểu diễn các đối tượng hoặc cây phân lớp hoặc thay đổi cả hai.

1.2.1.2. Mô hình hoá.

Các hệ thống phân loại đưa ra các câu trả lời rời rạc như có, không hoặc một số nguyên định danh các đối tượng đầu vào thuộc lớp nào. Mô hình hoá yêu cầu hệ thống phải sản sinh ra các câu trả lời mang tính liên tục. Trong quá trình mô hình hoá cần một số lượng nhỏ các số liệu để xây dựng mô hình. Mô hình này có thể đưa ra các dự báo cho tất cả các đối tượng đầu vào. Việc tìm ra đường cong phù hợp với các số liệu thực nghiệm là một trong những ứng dụng thuộc dạng này. Trong bất kỳ loại mô hình nào cũng phải tuân theo một giả định là: Các thay đổi nhỏ của tín hiệu vào chỉ gây ra những biến đổi nhỏ của tín hiệu ra.

Trong các vấn đề đa biến mạng nơron có nhiều ưu thế hơn so với các mô hình hoá cổ điển sử dụng các hàm giải tích. Bởi vì trong phương pháp mô hình hoá cổ điển, đối với mỗi đầu ra ta phải xác định một hàm cụ thể cùng một bộ các tham số. Trong khi đó đối với mạng nơron thì không phải quan tâm tới những hàm đó. Tuy nhiên, trong các phương pháp mô hình hoá cổ điển, các hệ số có thể có một số ý nghĩa nào đó đối với vấn đề cần giải quyết, trái lại các trọng số của mạng không mang một ý nghĩa nào cả.

Trong nhiều ứng dụng khá đặc biệt, khi sai số thực hiện khá lớn chúng ta có thể mô hình hoá bằng cách cân xứng hoá giữa tín hiệu vào và tín hiệu ra. Trong các trường hợp này, sử dụng mạng như một bảng tra là đủ, mặc dù các bảng này sẽ cho lời giải gống nhau trong một khoảng nào đó của tín hiệu vào.

Đối với việc chọn chiến lược học, chúng ta cần quan tâm tới sự phân bố của các đối tượng dùng để học. Nếu số lượng đối tượng dùng cho việc học là ít và được phân bố đều trong toàn không gian, khi đó số liệu có thể được dùng ngay cho việc mô hình hoá. Trái lại, nếu các đối tượng là nhiều, sẵn có nhưng phân bố ngẫu nhiên trong không gian biến, đầu tiên ta phải giảm thiểu chúng

sao cho vẫn bao trùm toàn không gian, sau đó mới dùng làm số liệu cho việc mô hình hoá.

1.2.1.4. Liên kết.

Liên kết là tìm ra đối tượng đích có mối quan hệ với một đối tượng vào, thậm chí cả khi đối tượng vào bị hỏng hoặc hoàn toàn không biết. Theo một nghĩa nào đó, liên kết có thể được coi là phân loại. Thủ tục học cho vấn đề này là học có tín hiệu chỉ đạo.

Lĩnh vực nghiên cứu các quá trình phụ thuộc thời gian là một trong những lĩnh vực chính trong nghiên cứu quá trình điều khiển. Ở đây, người sử dụng dự báo được hành vi của hệ thống đa biến dựa trên một chỗi số liệu được ghi nhận theo thời gian. Trong mô hình hoá phụ thuộc thời gian, các biến của các tín hiệu vào bao gồm các giá trị hiện tại và quá khứ của các biến quá trình, trong đó tín hiệu ra dự đoán giá trị trong tương lai của những biến quá trình đó. Về nguyên tắc các hiểu biết này có thể có độ dài tuỳ ý, nhưng trong quá trình kiểm soát, hiểu biết tương lai chỉ bao gồm một bước thời gian. Việc học dịch chuyển tới bước tiếp theo tạo ra các cửa sổ bao gồm số bước thời gian của vectơ ra. Để tạo ra mô hình hoàn chỉnh của một quá trình, tất cả các biến quá trình phải được huấn luyện tại đầu ra của mạng, nhưng không phải tất cả các biến trong quá trình đều ảnh hưởng như nhau đối với kết quả cuối cùng, chỉ có một số biến là đáng quan tâm. Do đó chúng ta chỉ phải chọn các biến đó cho quá trình học.

Kỹ thuật dịch chuyển cửa sổ có thể được sử dụng để giải quyết các vấn đề chuỗi các sự kiện và đối tượng như trong các lĩnh vực về môi trường theo thời gian, kiểm soát hỏng hóc.

1.2.2. Các lĩnh vực ứng dụng của mạng nơron

Kể từ khi ra đời và phát triển mạng nơron đã được ứng dụng trong rất nhiều lĩnh vực. Do vậy, liệt kê được tất cả các ứng dụng của mạng nơron là không thực tế. Tuy nhiên, ta có thể đưa ra một số ứng dụng điển hình của mạng nơron như sau:

- Xử lý ảnh, nhìn máy: Gồm trùng khớp ảnh, tiền xử lý ảnh, phân đoạn và phân tích ảnh, nén ảnh,...
- Xử lý tín hiệu: Phân tích tín hiệu địa chấn và hình thái học.
- Nhận dạng mẫu: Gồm việc tách các nét đặc biệt của mẫu, phân loại và phân tích tín hiệu của rada, nhận dạng và hiểu tiếng nói, nhận dạng vân tay, ký tự, chữ viết,...
- Y học: Phân tích và hiểu tín hiệu điện tâm đồ, chuẩn đoán bệnh, xử lý ảnh y học.
- Quân sự: Các hệ phát hiện thuỷ lôi, phân loại luồng rada, nhận dạng người nói.
- Các hệ tài chính: Gồm phân tích thị trường chứng khoán, định giá bất động sản, cấp phát thẻ tín dụng và thương mại an toàn.
- Trí tuệ nhân tạo: Gồm các hệ chuyên gia,...
- Dự đoán: Dự đoán các trạng thái của hệ thống,...
- Quy hoạch, kiểm tra và tìm kiếm: Gồm cài đặt song song các bài toán thoả mãn ràng buộc, tìm nghiệm bài toán người du lịch, điều khiển và robot

1.2.3. Ưu nhược điểm của mạng nơron.

Ưu điểm:

- Xử lý song song
- Thiết kế hệ thống thích nghi
- Không đòi hỏi các đặc trưng mở rộng của bài toán (chủ yếu dựa trên tập học).
- Có thể chấp nhận lỗi do tính song song.

Nhược điểm:

- Không có các quy tắc hoặc hướng dẫn thiết kế rõ ràng đối với một ứng dụng nhất định.
- Không có cách tổng quát để đánh giá hoạt động bên trong mạng
- Việc học đối với mạng có thể khó (hoặc không thể) thực hiện.
- Khó có thể đoán trước được hiệu quả của mạng trong tương lai (khả năng tổng quát hoá).

1.3. MẠNG HOPFIELD

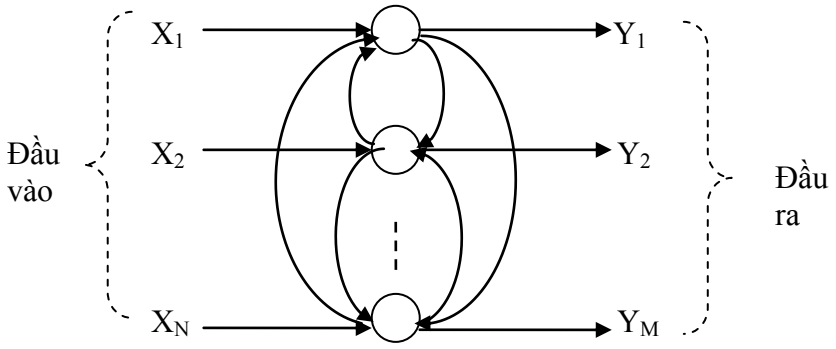
Trong mạng hồi quy tín hiệu ra của một nơron có thể được truyền ngược lại làm tín hiệu vào cho các noron ở các lớp trước, hoặc các nơron trong cùng một lớp. Phần này sẽ trình bày mô hình mạng tiêu biểu thuộc lớp mạng hồi quy, đó là mạng Hopfield.

Mạng Hopfield được bắt đầu nghiên cứu từ năm 1982. Đây là mạng một lớp với thông tin và quá trình xử lý có nối ngược. Công trình của Hopfield có rất nhiều ứng dụng, đặc biệt trong bộ nhớ liên kết và trong các bài toán tối ưu điển hình như bài toán lập lộ trình di chuyển cho robot.

Giả sử mạng được xây dựng dưới dạng mạng một lớp, mỗi nơron được truyền ngược lại làm tín hiệu vào cho các nơron khác nhưng bản thân các

nơron không tự liên kết với chính nó. Khi đó mô hình mạng Hopfield được biểu diễn như hình 1.7.

Tín hiệu ra của nơron j nào đó được truyền ngược lại làm tín hiệu vào cho các nơron khác trong mạng một cách đầy đủ thông qua trọng số tương ứng.



Hình 1.7. Mô hình mạng Hopfiled.

Ký hiệu w_{ij} là liên kết giữa hai nơron i và j ($w_{ij} = w_{ji}$), V_i là đầu ra của nơron i. Ta coi véc tơ (V_1, V_2, ..., V_n) là trạng thái của mạng. Tại mỗi thời điểm t mỗi nơron i tổng hợp các tín hiệu V_j từ các nơron khác và tín hiệu từ bên ngoài (bias).

$$U_i = \sum_i W_{ij} V_j(t) + I_i \qquad (1.13)$$

Tuỳ theo từng hàm kích hoạt f_i mà nơron i cho đầu ra là

$$V_i(t+1) = f_i(V_i(t)).$$

Mạng đạt trạng thái cân bằng nếu: $V_i(t) = V_i(t+1)$, $\forall i$.

Ta định nghĩa hàm năng lượng của mạng là:

$$E = E(V_1,...,V_n) = -\frac{1}{2}\sum_{i=1}^{n}\sum_{\substack{j=1 \\ i\neq j}}^{n} W_{ij} V_i V_j - \sum_{i=1}^{n} I_i V_i \qquad (1.14)$$

Tuỳ theo phương thức hoạt động của mạng mà người ta phân mạng Hopfield thành mạng Hopfield rời rạc và mạng Hopfield liên tục.

1.3.1. Mạng Hopfield rời rạc.

Mạng Hopfield rời rạc là mạng được tính rời rạc (đầu ra rời rạc) và làm việc ở chế độ không đồng bộ.

Trường hợp mạng nhận các giá trị nhị phân {0, 1}:

- Hàm kích hoạt được xác định như sau: $f_i \equiv f$

$$f(net) = \begin{cases} 1 \ khi \ \ net \geq 0 \\ 0 \ khi \ \ net < 0 \end{cases} \qquad (1.15)$$

Việc cho hàm kích hoạt (1.15) tương đương với quy tắc chuyển trạng thái của mạng .

$V_i(t+1) = V_i(t) + \Delta V_i$

Trong đó ΔV_i được cho bởi công thức

$$V_i = \begin{cases} 1 \ \ nếu \sum_j W_{ij} V_j(t) + I_i \geq 0 \ \ và \ \ V_i(t) = 0 \\ -1 \ \ nếu \ \sum_j W_{ij} V_j(t) + I_i \leq 0 \ \ và \ \ V_i(t) = 1 \\ 0 \quad trong \ các \ trường \ hợp \ khác \end{cases} \qquad \textbf{(1.16)}$$

1.3.2. Mạng Hopfield liên tục.

Mạng Hopfield liên tục là mạng mà trạng thái của nó được mô tả bởi phương trình động học:

$$\frac{dU_i}{dt} = \sum_j W_{ij} V_j + I_i \qquad (1.17)$$
$$V_i = f_i(U_i)$$

Trong đó f_i là hàm kích hoạt.

ở đây ta cũng giả thiết $W_{ij} = W_{ji}$ và $W_{ii}=0$. Dễ thấy rằng nếu hàm năng lượng được cho bởi 1.14 thì.

$$\frac{dU_i}{dt} = -\frac{\partial E}{\partial V_i}$$

1.4. MẠNG NƠRON TRONG KỸ THUẬT ROBOT

Những nghiên cứu của mạng nơron được xuất phát từ ý tưởng là tìm hiểu những nguyên lý hoạt động của bộ não con người, từ đó ứng dụng để thiết kế các hệ thống có thể thực hiện những nhiệm vụ phức tạp.

Thực tế, mạng nơron có liên quan đến các lĩnh vực như: học, thích nghi, khái quát hoá và tối ưu hóa. Trong các lĩnh vực này thì: sự đoán nhận, học, hoạt động và quyết định là những vấn đề liên quan mật thiết với kỹ thuật dò đường. Việc ứng dụng mạng nơron vào kỹ thuật tìm đường cho phép cải thiện những khả năng học và thích nghi đáp ứng được những thay đổi trong môi trường có thông tin không chính xác, không nhất quán và không đầy đủ. Kỹ thuật nơron có khả năng xử lý hiệu quả những dữ liệu không chính xác, kích thước lớn, đây sẽ là công việc khó khăn nếu sử dụng phương pháp truyền thống.

Mạng nơron là một hệ thống cho phép xử lý những thông tin song song và phân tán trên từng nơron, những nơron này được kết nối với nhau theo một mô hình nhất định. Việc học trong mạng nơron có thể được giám sát hoặc không được giám sát. Học giám sát là quá trình học sử dụng những thông tin mẫu đã được phân loại, trong khi học không giám sát chỉ sử dụng những thông tin tối thiểu không được phân loại. Những giải thuật học không giám sát có độ phức tạp tính toán thấp hơn cho kết quả chính xác hơn những giải thuật học giám sát.

Ngoài ra mạng nơron còn được ứng dụng trong bài toán phân loại và nhận dạng. Giải pháp giải quyết bài toán phân loại trong lộ trình di chuyển của người máy là succesfully phương pháp này có nền tảng là mạng nơron cạnh tranh (Bekey, G.A. & Goldberg, K. 1993). Không chỉ có vậy mạng nơron này còn được ứng dụng trong việc xác định các quỹ đạo di chuyển của người máy.

Để giúp robot tránh những chướng ngại vật mạng nơron với phương pháp huấn luyện là trượt dốc và lan truyền đã được sử dụng. Để dẫn đường cho người máy di chuyển trong môi trường hoạt động mạng nơron giám sát đã được sử dụng. Trong môi trường hoạt động của mình người máy học bởi mạng nơron, tại mỗi bước robot dự đoán các bước kế tiếp và từ đó phát sinh những tín hiệu điều khiển robot di chuyển.

Có thể nói việc ứng dụng mạng nơron để lập lộ trình di chuyển cho robot sẽ giúp cho robot di chuyển linh hoạt hơn và đây cũng là một công việc quan trọng trong kỹ thuật robot. Từ những ứng dụng của mạng nơron trong kỹ thuật robot, ta nhận thấy việc ứng dụng công nghệ này là vô cùng quan trọng, nó sẽ là giải pháp khả thi có tính đột phá để nâng cao khả năng hoạt động của robot trong môi trường hoạt động, từ đó ứng dụng vào thực tế cuộc sống.

1.5. NHẬN XÉT

Mạng truyền thẳng và mạng hồi quy là hai mô hình tiêu biểu của mạng nơron nhân tạo, Mỗi loại mạng sẽ có những ưu nhược điểm riêng. Nắm vững những ưu nhược điểm của chúng sẽ gúp ta lựa chọn mô hình mạng thích hợp cho từng ứng dụng sẽ thiết kế. Những ưu nhược điểm của từng mô hình mạng sẽ được thể hiện qua những nhận xét sau:

- Mạng truyền thẳng một lớp dễ phân tích nhưng không mô tả được mọi hàm. Mạng nhiều lớp khắc phục được nhược điểm trên

nhưng lại rất khó phân tích và gặp khó khăn trong quá trình xây dựng mạng. Mặt khác mạng truyền thẳng nhiều lớp có thể gây sai số tích luỹ qua các lớp.

- Mạng phản hồi một lớp (tiêu biểu là mạng Hopfield) có cấu trúc đơn giản vì thế dễ phân tích, không chứa sai số tích luỹ.
- Mạng nơron truyền thẳng chỉ đơn thuần tính toán các tín hiệu ra dựa trên các tín hiệu vào và trọng số liên kết giữa các nơron đã xác định sẵn ở trong mạng. Do đó chúng không có trạng thái bên trong nào khác ngoài trọng số W. Đối với mạng hồi quy, trạng thái bên trong của mạng được lưu trữ tại các ngưỡng của nơron. Nói chung các mạng hồi quy không ổn định, mạng cần phải tính toán rất lâu, thậm chí có thể lặp vô hạn trước khi đưa ra kết quả mong muốn. Quá trình học của mạng hồi quy cũng phức tạp hơn mạng truyền thẳng rất nhiều. Tuy vậy các mạng hồi quy có thể cho phép mô phỏng các hệ thống tương đối phức tạp trong thực tế.

CHƯƠNG 2

GIỚI THIỆU BÀI TOÁN LẬP LỘ TRÌNH CHO ROBOT

2.1. GIỚI THIỆU 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, môi trường 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.

2.1.1. Tổng quan

Các nhân tố cấu thành robot:

- Về cấu tạo: Robot phải dược trang bị bộ cảm nhận để cảm nhận các thông tin về môi trường như: sensor, encoder, camera,... Các bộ phận thực hiện hành động: bánh xe để chuyển động, cánh tay…
- 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.

2.1.2. Giải pháp thiết kế

Để thiết kế robot ta phải hoàn thiện các công đoạn 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

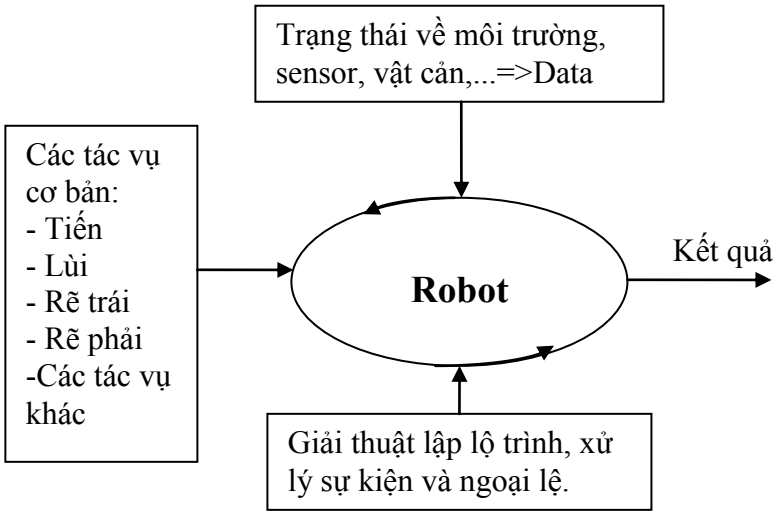
* Nhúng các giải thuật tìm đường và giải thuật xử lý sự kiện cho robot để có một đường đi tốt từ vị trí ban đầu tới đích và xử lý các tình huống ngoại lệ như va chạm.

* 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 2.1. 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 như thể xác. Các mạch điện tử giống như các mạch máu, các nơron thần kinh, các giác quan bên ngoài. Chương trình giống như bộ não giúp điều khiển cơ thể thông qua hệ thống mạch.

2.2. BÀI TOÁN LẬP LỘ TRÌNH.

2.2.1. Mở đầu.

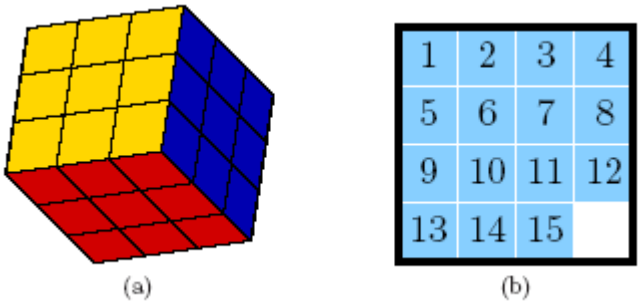
Để robot có thể hoạt động trong môi trường và thực hiện tốt các chức năng của nó thì ngoài các cơ cấu cơ khí, các mạch điện tử ra thì các chương trình điều khiển là không thể thiếu. Như đã trình bày ở trên, chương trình có thể ví như bộ não để điều khiển mọi hoạt động của robot. Như vậy để robot có thể hoạt động hiệu quả thì chương trình phải được thiết kế tốt, phù hợp với các đặc tính điện tử, cơ khí. Nền tảng của các chương trình này chính là các giải thuật nhằm mô phỏng những hoạt động bậc cao của con người vào trong những mô tả mức thấp để sao cho có thể hướng dẫn robot hoạt động. Một trong những giải thuật như vậy là giải thuật lập lộ trình chuyển động cho robot. Giải thuật này sẽ hướng dẫn robot di chuyển từ vị trí ban đầu tới vị trí

đích sao cho tránh được những va trạm trên đường đi. Ta có thể hình dung việc lập lộ trình tương tự như bài toán di chuyển một chiếc piano. Giả sử ta cần thiết kế một giải thuật giúp máy tính thiết kế chính xác đường đi để di chuyển chiếc piano từ vị trí này đến vị trí khác với dữ liệu đầu vào là cấu trúc toà nhà và vị trí của piano. Việc lập lộ trình cho robot thông thuường không quan tâm đến động lực học mà chỉ quan tâm tới việc tìm đường và di chuyển đến đích tránh va trạm với môi trường xung quanh.

Khái niệm lập lộ trình xuất hiện trong khá nhiều lĩnh vực tiêu biểu như: Lý thuyết điều khiển và trí tuệ nhân tạo.

- Trong lý thuyết điều khiển: Vấn đề này được đề cập tới như việc thiết kế những hệ thống vật lý mô tả bởi những phương trình vi phân. Những hệ thống đó có thể bao gồm những hệ thống cơ khí như ô tô hoặc máy bay, những hệ thống điện như lọc tiếng ồn, hoặc cả những hệ thống xuất hiện trong nhiều lĩnh vực đa dạng khác như hóa học, kinh tế học và xã hội học. Trước 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 ưu hóa để tối giản tiêu thụ tài nguyên, như năng lượng hoặc thời gian. Trong các tài liệu về lý thuyết điều khiển gần đây, việc lập lộ trình chuyển động đôi khi được quy dẫn đến việc xây dựng đầu vào tới một hệ thống động lực phi tuyến để điều khiển robot từ vị trí ban đầu tới một vị trí đích xác định. Trong lĩnh vực này luôn mong muốn có một thuật toán lý tưởng sao cho vẫn xử lý tốt bài toán khi những dữ liệu đầu vào là không chắc chắn hoặc xuất hiện từ những mẫu không chính xác.

- Trong trí tuệ nhân tạo: Thuật ngữ lập lộ trình AI lại thể hiện những đặc điểm riêng biệt. Thay vì việc di chuyển trong một không gian liên tục bài toán sẽ được quy dẫn về vấn đề tìm kiếm lộ trình trong một không gian trạng thái, tương tự như bài toán khối lập phương Rubik hoặc bài toán dịch chuyển số. Mặc dù những vấn đề này hoàn toàn có thể được mô hình hoá trong không gian liên tục song việc giải quyết bài toán trong không gian trạng thái cho phép xây dựng các thuật toán lựa chọn một dãy hoạt động thích hợp để điều khiển hoạt động của robot.



Hình 2.2. Khối Rubik (a), bài toán dịch chuyển số (b).

Thuật ngữ lập lộ trình bao hàm rất nhiều thao tác song trong khuân khổ của đề tài ta chỉ quan tâm tới các thuật toán lập lộ trình. Muốn hiểu sâu sắc về các giải thuật lập lộ trình ta phải trả lời được các câu hỏi:

- Thế nào là một lộ trình?
- Một lộ trình được mô tả như thế nào?
- Nó được cài đặt như thế nào trong máy tính?
- Thế nào được coi là hoàn tất?
- Chất lượng của nó được đánh giá ra sao?
- Đối tượng nào sẽ sử dụng nó?

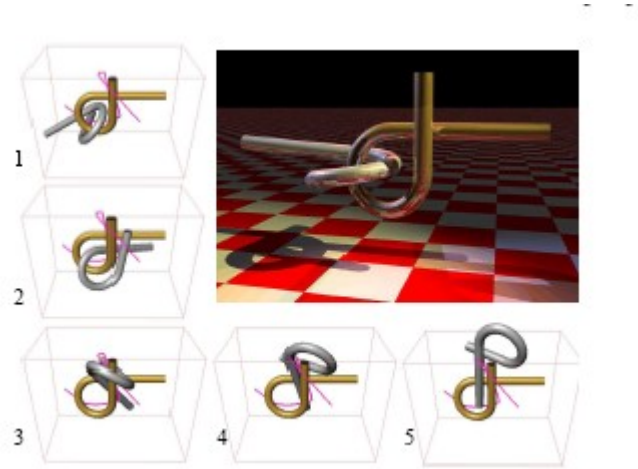
Có thể nói lập lộ trình là một công việc khá phức tạp vì vậy đòi hỏi phải nghiên cứu và đưa ra những giải pháp (giải thuật) hợp lý để giải quyết

bài toán này. Mặt khác việc lập lộ trình dựa trên các giải thuật đã đạt được những thành công lớn trong các lĩnh vực: công nghệ, lý thuyết khoa học, công nghệ robot, thiết kế sản xuất, không gian vũ trụ,...Những lý do trên đã thúc đẩy việc học tập và nghiên cứu những giải thuật lập lộ trình, góp phần phát triển và ứng dụng chúng vào các lĩnh vực trong thực tế.

2.2.2. Các ví dụ thực tế.

Ví dụ 1: Bài toán Alpha 1.0 puzzle do Boris Yamrom đề xuất. Bài toán này đã được Nancy Amato xây dựng như là một chuẩn để đánh giá các nghiên cứu về việc lập lộ trình của trường đại học Texas A&M. Giải pháp cho bài toán này do James Kuffner đề xuất. Bài toán này thể hiện trong hình 2.3.

Việc giải các bài toán trong hình 2.2 có thể dễ dàng được thực hiện bởi tính đều đặn và đối xứng của những thành phần tham gia vào di chuyển. Tuy nhiên, những đặc điểm này không có trong bài toán 2.3. Bên cạnh đó nó còn yêu cầu phải giải quyết vấn đề trong không gian liên tục. Bài toán này sẽ được giải quyết trong một vài phút trên máy tính cá nhân chuẩn sử dụng kỹ thuật thăm dò nhanh tróng trên cấu trúc cây dầy đặc.

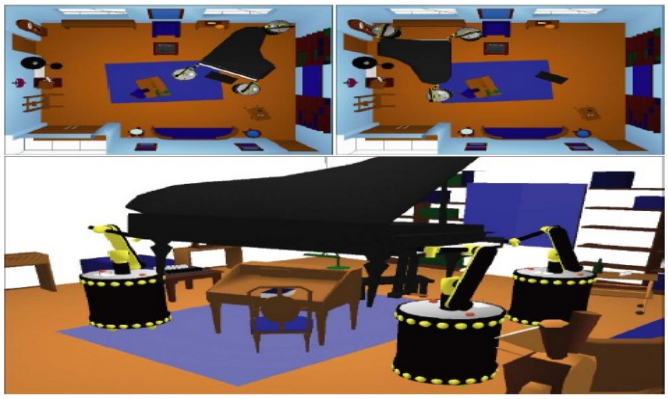


Hình 2.3. Tìm giải thuật để kéo hai thanh thép tách ra

Mặc dù các vấn đề trình bày ở trên chỉ là trò chơi giải trí song trong thực tế có rất nhiều ứng dụng quan trọng có nguyên lý tương tự như những trò

chơi này. Điều này khẳng định lý thuyết chuyển động không đơn thuần là trò chơi giải trí mà nó có rất nhiều ứng dụng trong thực tế.

Ví dụ 2: Di chuyển một piano lớn qua một căn phòng bằng cách sử dụng ba robot di động với cánh tay thao tác bên trên chúng. Hình 2.4 mô tả quá trình di chuyển. Trong quá trình di chuyển yêu cầu phải tránh được những va trạm giữa robot với những đồ vật khác. Vấn đề sẽ trở nên phức tạp hơn khi cấu trúc của căn phòng không được biết trước.



Hình 2.4. Sử dụng robot di động để di chuyển piano

Ví dụ 3: Giả sử ta đang ở trong một hành lang hoàn toàn tối, với một chiếc đèn trên tay. Yêu cầu phải tìm kiếm một người bất kỳ nào đó. Khi thực hiện nhiệm vụ này có một vài câu hỏi được đặt ra:

- Liệu có tồn tại một chiến lược đảm bảo rằng sẽ tìm thấy tất cả mọi người không?

- Nếu không thì thực hiện như thế nào để tiếp tục tìm kiếm và hoàn thành nhiệm vụ này?

- Đâu là nơi ta cần dịch chuyển đến ở bước tiếp theo?

- Làm thế nào để tránh được việc thăm dò một chỗ nhiều lần?

Những vấn đề đề xuất trong ví dụ này cũng xuất hiện trong nhiều ứng dụng của kỹ thuật robot. Để đảm bảo cho robot thực hiện các nhiệm vụ trên

thì chúng phải được trang bị những hệ thống cảm biến và cần phải có một chiến lược để robot định vị những vật khác.

Qua những ví dụ nêu trên ta nhận thấy các giải thuật lập lộ trình không chỉ có ứng dụng trong công nghệ robot mà nó có thể được ứng dụng trong rất nhiều lĩnh vực của thực tế như: Tìm kiếm cứu nguy, làm sạch chất độc, thiết kế an toàn trong các toà nhà, giúp những máy bay tự lái vượt qua những chướng ngại vật, là cơ sở tính toán cho tầu vũ trụ tránh những va trạm trong môi trường phức tạp. Thậm chí thuật toán lập lộ trình còn được ứng dụng trong những kỹ thuật máy tính phỏng sinh học như robot khám bệnh, những mô hình hình học ứng dụng tới từng phân tử cũng có thể được giải quyết bởi giải thuật lập lộ trình.

2.2.3. Bài toán lập lộ trình chuyển động cho robot.

Bài toán lập lộ trình có thể được phát biểu như sau:

Cho đối tượng với vị trí ban đầu và vị trí đích với một tập các chướng ngại vật có các vị trí khác nhau trong không gian làm việc. Yêu cầu tìm ra một đường đi liên tục từ vị trí ban đầu đến vị trí đích sao cho tránh được những va trạm với những vật cản trên đường đi. Quá trình xác định lộ trình thường chia làm hai thao tác chính đó là: xây dựng không gian cấu hình và tìm đường.

Có thể tóm tắt bài toán như sau:

Đầu vào (Input):

Những mô tả hình học của người máy, môi trường và những chướng ngại vật, vị trí ban đầu và vị trí đích.

Đầu ra (output):

Đường đi từ vị trí đầu đến vị trí đích hoặc thông báo không tồn tại đường đi.

2.3. CÁC THÀNH PHẦN CƠ BẢN CỦA VIỆC LẬP LỘ TRÌNH

Mặc dù công việc lập lộ trình là một công việc khá phức tạp và phải trải qua rất nhiều công đoạn, song ta có thể hệ thống hoá các thành phần của bài toán này. Cụ thể việc lập lộ trình gồm những thành phần cơ bản sau:

2.3.1. Trạng thái.

- Trạng thái của bài toán lập lộ trình là một hình trạng của bài toán nó chính là vị trí tương đối của robot với các đối tượng trong môi trường di chuyển

- Không gian trạng thái gồm tất cả các hình trạng có thể xuất hiện.

Không gian trạng thái có thể bao bồm các đối tượng rời rạc (hữu hạn, hoặc vô hạn đếm được) hoặc liên tục (vô hạn không đếm được). Việc xây dựng không gian trạng thái hoàn toàn phụ thuộc vào thuật toán lập lộ trình tương ứng. Trong đa số các ứng dụng, kích thước của không gian trạng thái (số những trạng thái hoặc tổ hợp các trạng thái) là quá lớn để có thể thể hiện rõ ràng từng trạng thái. Tuy nhiên, định nghĩa không gian trạng thái vẫn là một thành phần quan trọng trong việc trình bày, phân tích, thiết kế những giải thuật để giải quyết bài toán lập lộ trình.

2.3.2. Thời gian

Toàn bộ vấn đề lập lộ trình là những dãy quyết định được thực hiện trong suốt thời gian thực hiện. Thời gian lập lộ trình ảnh hưởng đến các quyết định liên tiếp trong quá trình xác lập lộ trình. Sự tối ưu về mặt thời gian trong bài toán này có thể hiểu như việc điều khiển một chiếc ô tô vượt qua một chướng ngại vật càng nhanh càng tốt.

Trong những giải thuật người ta chú ý đến thời gian tổng thể đưa robot từ trạng thái ban đầu đến trạng thái đích. Trong các giải thuật lập lộ trình chuyển động người ta tránh chỉ rõ thời gian cụ thể trên một lộ trình cụ thể mà ước lượng thời gian trong trường hợp xấu nhất.

2.3.3. Hành động (Actions)

Trên một lộ trình có thể phát sinh những hành động (thao tác) trên những trạng thái. Trong lý thuyết và kỹ thuật điều khiển rôbôt, thuật ngữ hành động liên quan đến việc nhập đầu vào và điều khiển. Ở một số cách tiếp cận về vấn đề lộ trình, hành động phải được chỉ rõ để sao cho có thể thay đổi được trạng thái khi nó thực thi. Điều này có thể được biểu thị như một hàm đánh giá trạng thái của trường hợp thời gian riêng biệt hoặc như một phương trình vi phân bình thường cho thời gian liên tục.

Trong thực tế có một vài vấn đề về hành động cần được quan tâm, đó là: Những hành động tự nhiên có thể gây hậu quả rắc rối cho sự điều khiển của người chế tạo. Điều này làm nảy sinh những hành động nhưng ta hoàn toàn có thể ước đoán để ứng dụng vào vấn đề lập lộ trình.

2.3.4. 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 được lựa chọn để đạt được điều đó.

2.3.5. Tiêu chuẩn

Ta sẽ đánh giá các thuật toán lập lộ trình dựa trên hai tiêu chuẩn sau:

- Tính khả thi : Sẽ tìm được một lộ trình để đưa robot đến trạng thái đích, không quan tâm đến hiệu quả của quá trình này.
- Sự tối ưu : Lộ trình tìm được là khả thi và tối ưu theo một tiêu chí nào đó (tối ưu về mặt thời gian, tối ưu về không gian,...)

Trong luận văn này, chúng ta chỉ tập trung vào tính khả thi, việc đạt được sự tối ưu là rất khó khăn, đây có thể là hướng để mở rộng đề tài.

2.3.6. Giải thuật

Giải thuật là một thành phần quan trọng không thể thiếu trong công việc lập lộ trình. Nó là nền tảng để xác định nhanh chóng, chính xác một lộ trình di chuyển của robot.

2.3.7. Người lập lộ trình

Một người lập lộ trình có thể hiểu là đối tượng tạo nên lộ trình, nó có thể là máy móc hoặc con người. Nếu người lập lộ trình là một máy, thì nó sẽ được xem như một giải thuật lập lộ trình.

Trong vài trường hợp, con người trở thành những người 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 được đưa tới cho con người, và con người " tính toán " một lộ trình tương ứng. Trong trường hợp này con người vẫn làm tròn vai trò của giải thuật.

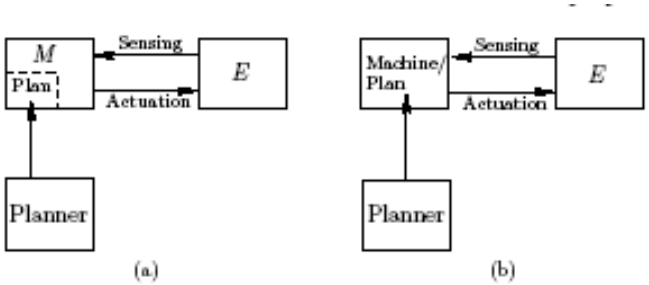
2.3.8. Lộ trình

Hiểu một cách đơn giản: Lộ trình là một bản kế hoạch mà nhìn vào đó người ta có thể xác định được cần đi như thế nào mà không va phải những chướng ngại vật và đến được 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: Chướng ngại vật (Obstacles), khoảng trống tự do (Free Space) và robot

- **Chướng ngại vật**: Là thành phần " thường xuyên" chiếm chỗ trong không gian, hay nói cách khác là nơi mà robot không thể đi vào. Ví dụ như bức tường của một toà nhà.
- **Khoảng trống tự do**: Là nơi còn trống trong không gian mà robot có thể đi vào. Để quyết định xem robot có thể đi được vào đó hay không chúng ta cần tìm hiểu khái niệm Configuation Space (Cấu hình không gian)

- **Robot:** Những vật thể được mô hình hoá hình học và có thể kiểm soát theo một lộ trình đã lập.



Hình 2.5 : (a) Người lập lộ trình thiết kế giải thuật lập lộ trình. (b) Người lập lộ trình thiết kế toàn bộ máy.

Khi sử dụng một lộ trình ta cần quan tâm đến những khái niệm sau:

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 được một lộ trình tốt hơn.

3. Mô hình có thứ bậc : Coi lộ trình như một hành động của một lộ trình ở mức độ cao hơn.

Những khái niệm này có thể được giải thích kỹ hơn như sau:

Sự thực thi: Một lộ trình thông thường được thực thi bởi một máy. Một con người cũng có thể thực thi một lộ trình. Tuy nhiên, trong trường hợp này chúng ta tập trung nghiên cứu quá trình thực hiện lộ trình trên máy. Có hai cách để thực hiện trên máy, đó là:

Cách 1: Trong ***Hình 2.5a***, người lập lộ trình tạo ra một lộ trình, mã hóa nó theo một cách nào đó và nhập vào máy. Trong trường hợp này sau khi đã được nhập chương trình thì máy sẽ tự trị tức là tuần tự thực hiện những bước của chương trình và không còn sự tương tác với người lập trình nữa. Tất nhiên, mô hình này có thể được 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

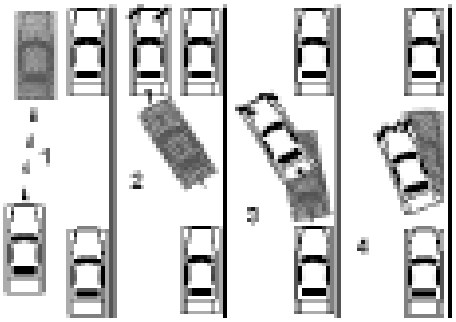
tế, tuy nhiên, chúng chưa được ưa chuộng bởi những lộ trình cần phải thiết kế trước để tính đến thông tin mới trong thời gian thực thi.

Cách 2: Được miêu tả trong ***Hình 2.5 b***. Trong trường hợp này, lộ trình sản sinh bởi người lập lộ trình mã hóa trọn vẹn trong máy. Đây là một lộ trình đặc biệt chủ định cho máy và được thiết kế để giải quyết những nhiệm vụ đặc biệt cho trước hướng tới người lập lộ trình. Giải thuật này chỉ hướng 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 người và máy cũng có thể giải quyết được nhiệm vụ được giao.

Sự Cải tiến: Nếu một lộ trình được sử dụng để cải tiến, thì người lập lộ trìn sẽ coi nó như đầu vào và xác định một lộ trình mới. Lộ trình mới này tính đến nhiều khía cạnh của vấn đề hơn, hoặc nó có thể đơn giản và hiệu quả hơn.

Sự cải tiến có thể được ứng dụng nhiều lần, để sản sinh một dãy các lộ trình cải tiến, khi đó lộ trình cuối cùng có sự thực thi tốt nhất.

(a)



(b)

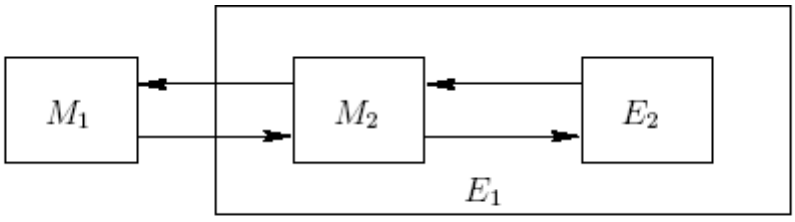
(c) (d)

Hình 2.6. ***Một số lộ trình và sự cải tiến lộ trình***

Ví dụ:

- Lộ trình đầu tiên (a)- di chuyển một robot di động trong nhà, chấp nhận một sự va chạm - tự do trên đường đi qua tòa nhà.
- Cải tiến sang lộ trình thứ hai (b) lộ trình đòi hỏi phải thỏa mãn những sự ràng buộc cơ sở khác nhau của sự điều khiển chuyển động.
- Lộ trình thứ ba (c) xem xét làm sao để di chuyển robot dọc theo đường với những tốc độ khác nhau, đồng thời quan tâm đến động lực học.
- Lộ trình thứ tư (d) hợp nhất thông tin phản hồi để bảo đảm rằng người máy có khả năng càng đóng càng tốt với lộ trình mặc dù hành vi là không thể đoán trước.

Mô hình thứ bậc: Một lộ trình được coi như một hoạt động của một lộ trình lớn hơn.



Hình 2.7: Mô hình có thứ bậc, một máy có thể chứa đựng một máy khác

Lộ trình nguyên bản có thể hình dung như một chương trình con trong lộ trình lớn hơn. Để thực hiện được điều này lộ trình nguyên bản phải bảo đảm tính dừng, để lộ trình lớn hơn có thể thực thi chúng nhiều lần khi cần.

Mô hình có thứ bậc có thể được biểu diễn với bất kỳ số lộ trình nào, kết quả của mỗi lộ trình sẽ được lưu trong một nút của cây lộ trình.

Như vậy mô hình chung của lộ trình có thứ bậc là một cây trong đó mỗi đỉnh của cây là một lộ trình. Đỉnh gốc là lộ trình chính. Con của một đỉnh bất kỳ là một lộ trình con. Không có giới hạn tới chiều sâu cây hoặc số các đỉnh con. Trong việc lập lộ trình có thứ bậc, dòng tương tác giữa máy và môi trường được vẽ nhiều hướng. Ví dụ, môi trường E1 của máy M1, có thể chứa máy khác như M2, tương tác đó với môi trường E2 của nó, như trong ***Hình2.7.***

2.3.9. Lập lộ trình chuyển động

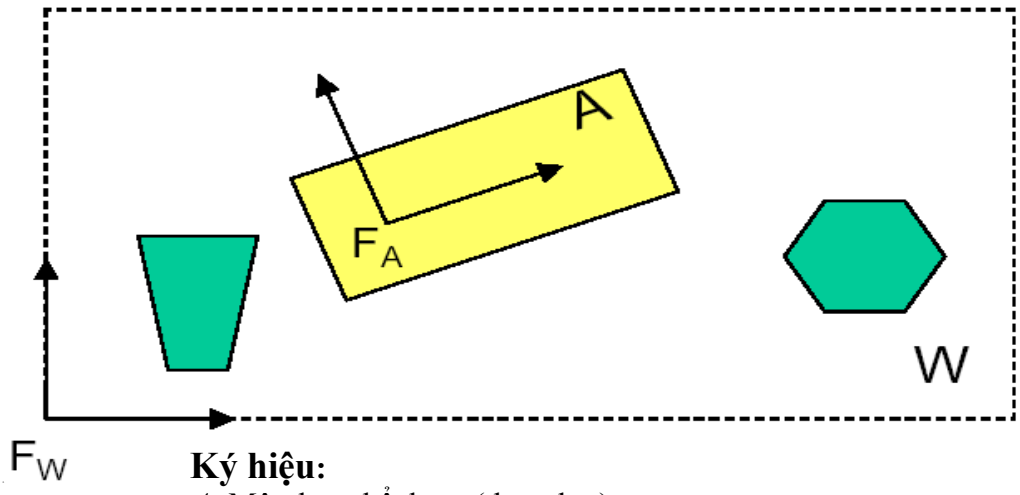
Đây là nguồn cảm hứng chính cho những vấn đề và tất cả các giải thuật của kỹ thuật rôbôt. Những phương pháp này đủ tổng quát để sử dụng trong nhiều ứng dụng của nhiều lĩnh vực khác nhau, như máy tính sinh học, thiết kế với sự trợ giúp của máy tính, đồ hoạ máy tính... Một cách nói khác chính xác hơn cho việc lập lộ trình chuyển động là " Lập lộ trình trong không gian liên tục "

2.4. KHÔNG GIAN CẤU HÌNH

2.4.1. Các khái niệm không gian cấu hình.

Không gian cấu hình (cấu hình không gian) là không gian của tất cả những cấu hình có thể của robot.

2.4.1.1. Chướng ngại (Obstacle): Là những phần của không gian "Thường xuyên" bị choán chỗ, ví dụ, như trong những bức tường của một tòa nhà.



Ký hiệu:
A: Một thực thể đơn –(the robot)
W: Không gian Euclidean ở đó *A* di chuyển;
B1,…Bm: chướng ngại vật phân bổ cố định trong ***W***

Hình 2.8- *Không gian cấu hình*

- Cấu hình chướng ngại vật: Là cấu hình của từng chướng ngại vật
- Miền chướng ngại vật: Là hợp của tất cả các cấu hình chướng ngại vật

From
Robot Motion Planning
J.C. Latombe

C-OBSTACLE REGION

$B_1, B_2, \ldots, B_m$ ——— obstacles

CB_i ——— C-obstacle

$\bigcup_{i=1}^{m} CB_i$ ——— C-obstacle region

2.4.1.2 Không gian trống (Free Space- C_{free}): Là phần bù của toàn bộ không gian với miền chướng ngại vật.

FREE SPACE

$$C_{free} = C \setminus \bigcup_{i=1}^{m} CB_i = \{q \in C : A(q) \cap \bigcup_{i=1}^{m} CB_i = \phi\}$$

Free configuration q iff $q \in C_{free}$

2.4.2. Mô hình cấu hình.

Để có thể thực hiện được các giải thuật lập lộ trình ta cần phải biểu diễn được không gian cấu hình vào máy. Có nhiều phương pháp để mô hình

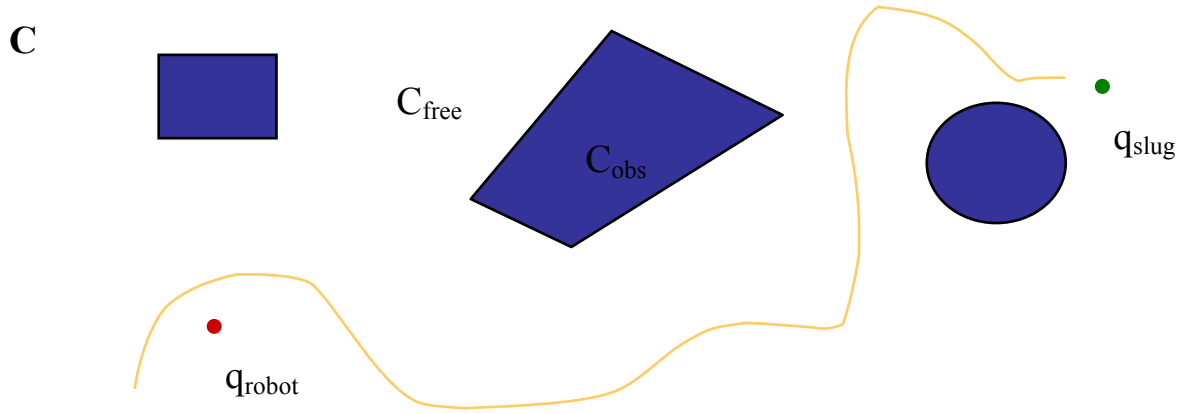
hoá không gian ở đây chúng ta quan tâm chủ yếu đến hai loại chính đó là: mô hình hình học và mô hình nửa đại số.

2.4.2.1. Mô hình hình học.

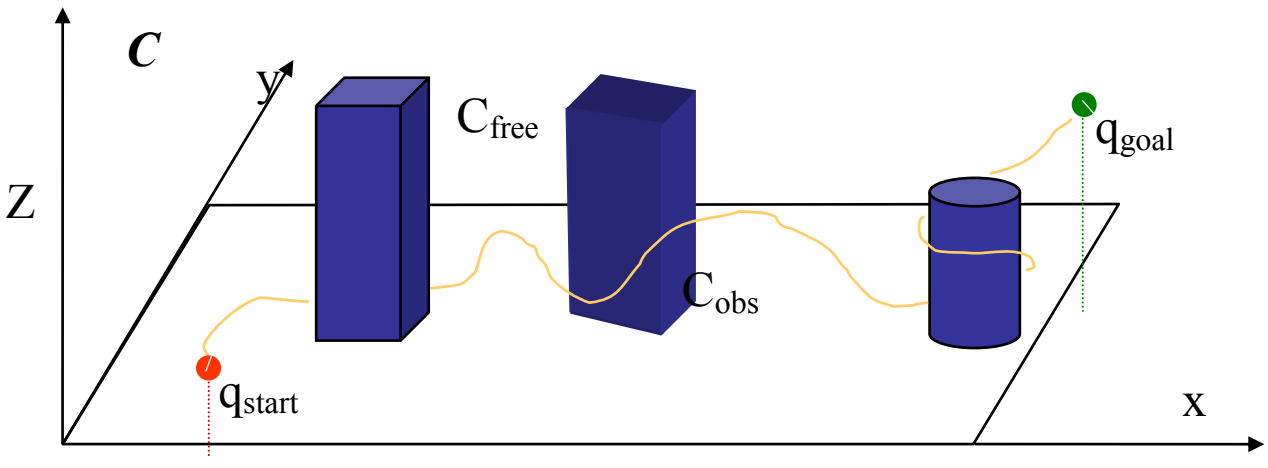
Có rất nhiều phương pháp và kỹ thuật trong mô hình hình học. Tuỳ theo từng ứng dụng mà ta có thể lựa chọn các giải pháp khác nhau. Tuy nhiên có hai giải pháp thường được lựa chọn để biểu diễn W, đó là:

1) Không gian 2 chiều, khi đó $W = R^2$.

2) Không gian 3 chiều, khi đó $W = R^3$.



Hình 2. 9 Một Robot điểm di chuyển trong không gian 2D, C-space là R^2



Hình 2.10: Một Robot điểm di chuyển trong không gian 3D, C-space là R^3

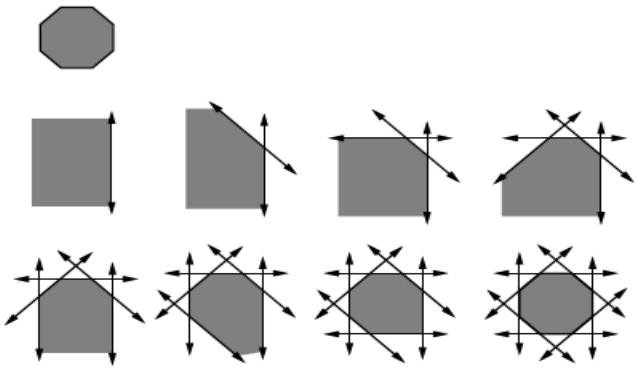
Tuy nhiên, trong thực tế có nhiều không gian phức tạp hơn, như bề mặt của một hình cầu, khi đó cần những không gian có số chiều lớn hơn để biểu

diễn chúng. Song trong khuôn khổ của luận văn ta không bàn tới những lĩnh vực này.

1- Mô hình đa giác :

Trong không gian hai chiều 2D, $W = R^2$. Vùng chướng ngại vật O là một tập các đa giác lồi. Biểu diễn một m-đa giác trong O được mô tả bởi hai đặc trưng đỉnh và cạnh.

Mỗi đỉnh tương ứng tới một "góc" của đa giác, và mỗi cạnh tương ứng với một đoạn nối giữa một cặp của đỉnh. Đa giác có thể được chỉ rõ bởi đường nối liên tiếp các cặp đỉnh của m điểm bên trong R^2 theo thứ tự ngược chiều kim đồng hồ: $(x_1, y_1), (x_2, y_2),... , (x_m, y_m)$.



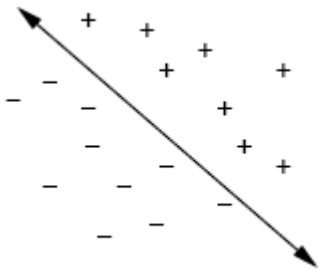
Hình 2.11* : *Một đa giác lồi có thể được xác định bởi phép giao của các nửa - mặt phẳng.

Một hình đa giác trong O có thể được biểu thị như phép giao của m nửa mặt phẳng. Mỗi nửa mặt phẳng tương ứng tới tập hợp của tất cả các điểm mà nằm ở một bên của đường thẳng trùng với cạnh của một đa giác. ***Hình 2.11*** cho thấy một ví dụ của một hình bát giác được biểu diễn như phép giao của tám nửa mặt phẳng. Một cạnh của đa giác được chỉ rõ bởi hai điểm, như (x_1, y_1) và (x_2, y_2). Xem xét phương trình của một đường thẳng đi qua (x_1,y_1) và (x_2,y_2). Một phương trình có thể được xác định có dạng: $ax+ by + c = 0$. Trong đó $a, b, c \in R$ là những hằng số được xác định từ x_1, y_1, x_2, y_2.

Cho ánh xạ $f : R^2 \longrightarrow R$ xác định bởi hàm $f(x, y) = ax + by + c$.

Không mất tính tổng quát ta có thể giả thiết f(x,y) < 0 là những điểm nằm bên trái của đường thẳng, f(x,y)> 0 là những điểm nằm bên phải. Để cho $f_i(x, y)$ biểu thị hàm f dẫn xuất ra từ đường thẳng mà tương ứng với cạnh từ (x_i, y_i) tới (x_{i+1}, y_{i+1}) với $1 \le i < m$. Để cho $f_m(x, y)$ biểu thị phương trình đường thẳng mà tương ứng với cạnh từ (x_m, y_m) tới (x_1, y_1). Để cho một nửa mặt phẳng H_i cho $1 \le i \le m$ được xác định một tập con của W:

$$H_i = \{(x,y) \in W \mid f_i(x,y) \le 0\} \qquad (2.1)$$



Hình 2.12: Dấu hiệu của f(x, y) phân chia R^2 vào ba vùng : f(x, y) < 0 , f(x, y) > 0, f(x, y) = 0.

Một tập lồi, m – cạnh, vùng O chướng ngại đa giác được biểu thị như sau:

$$O = H_1 \cap H_2 \ldots \cap H_m \qquad (2.2)$$

Trong đa số các ứng dụng các tập con không lồi có thể vẫn được chấp nhận. Khi đó vùng chướng ngại O được biểu thị như sau:

$$O = O_1 \cap O_2 \ldots \cap O_m \qquad (2.3)$$

Trong đó mỗi O_i là một đa giác lồi, với O_i và O_j ($i \neq j$) không cần rời nhau. Với cách này, chúng ta có thể biểu diễn được rõ ràng các không gian rất phức tạp. Trong những không gian phức tạp ta cần phải biểu diễn thông qua sự kết hợp hữu hạn giữa phép hợp, giao, và hiệu của tập hợp mẫu. Tuy nhiên, để đơn giản hoá việc biểu diễn các mẫu người ta cố gắng sử dụng cách biểu diễn theo phép hợp và giao của các mẫu. Một tập hợp hiệu thường tránh sử dụng để

biểu diễn mẫu. Để làm được như vậy người 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 được kết hợp bởi những mẫu đơn giản có thể loại bỏ được 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 ưu hóa việc tính toán trong những giải thuật sử dụng mô hình. Trong đa số các trường hợp, những thành phần có thể được cho phép giao nhau. Lý tưởng nhất là việc lựa chọn cách biểu thị O sao cho tối thiểu nhất các mẫu.

Ở đây một logic vị từ đã được định nghĩa như sau: Φ: W $\rightarrow$ {TRUE, FALE}. Hàm trả về giá trị TRUE cho một điểm trong W nằm bên trong O, ngược lại là FALSE. Cho một đường thẳng f(x, y) = 0 để e(x, y) biểu thị một vị từ lôgíc trả về giá trị TRUE nếu f(x, y) = 0, và ngược lại là FALSE . Một vị từ tương ứng tới một vùng đa giác lồi được biểu diễn bởi các phép hội như sau:

$$\alpha(x,y) = e_1(x,y) \wedge e_2(x,y) \wedge ... \wedge e_m(x,y) \qquad (2.4)$$

Vị từ α (x, y) trả về giá trị TRUE nếu điểm (x, y) nằm trong vùng đa giác lồi, ngược lại là FALSE. Một vùng chướng ngại mà gồm có n đa giác lồi được biểu diễn bởi tuyển như sau:

$$\phi(x,y) = \alpha_1(x,y) \vee \alpha_2(x,y) \vee ... \vee \alpha_n(x,y) \qquad (2.5)$$

Mặc dầu tồn tại những phương pháp hiệu quả hơn, song ϕ có thể kiểm tra cho một điểm (x, y) nằm trong trong O với thời gian O(n), trong đó n là số mẫu xuất hiện trong biểu diễn của O (Mỗi mẫu được ước lượng trong hằng số thời gian). Bất kỳ mệnh đề lôgíc phức tạp đến đâu đều có thể được tách nhỏ thành những chuẩn tuyển (Đây thường được gọi " tổng của những tích " trong

khoa học máy tính). Như vậy chúng ta có thể nói bất kỳ một không gian O nào đều luôn được biểu diễn bằng hợp của hữu hạn các phép giao những mẫu.

2- Mô hình đa diện:

Trong không gian ba chiều $W = R^3$, những khái niệm có thể được khái quát hóa từ trường hợp không gian 2D bởi việc thay thế đa giác bằng khối đa diện và việc thay thế nửa mặt phẳng bởi nửa không gian mẫu. Một danh giới biểu diễn có thể được định nghĩa dưới dạng ba đặc trưng : Đỉnh, cạnh, và mặt. Một vài cấu trúc dữ liệu được đưa ra để biểu diễn đa diện, ví dụ, cấu trúc dữ liệu chứa ba kiểu bản ghi : mặt, nửa cạnh, và đỉnh (một nửa cạnh là cạnh có hướng).

Giả sử O là một đa diện lồi, như trong ***Hình 2.13***. Một biểu diễn ba chiều có thể được 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 phương trình của mặt phẳng đi qua chúng có dạng:

$$ax + by + cz + d = 0 \qquad (2.6)$$

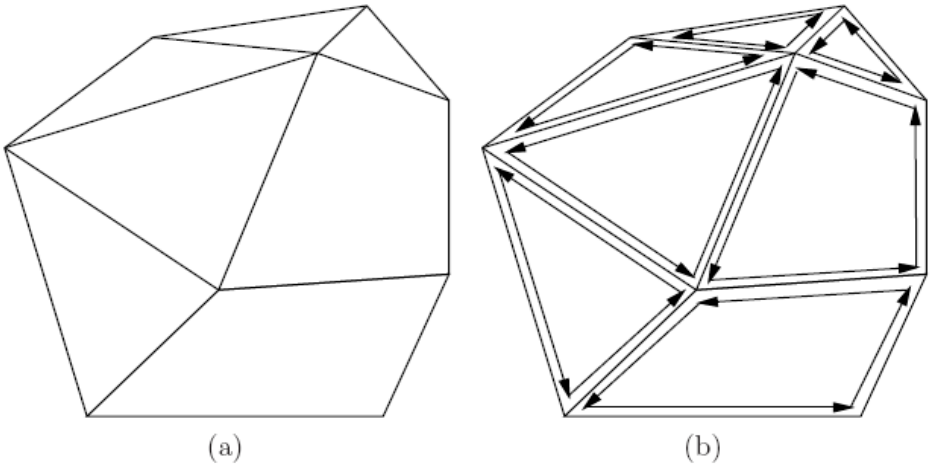
Trong đó a, b, c, d $\in$ R là những hằng số. Một lần nữa, f có thể xây dựng bằng ánh xạ $f : R^3 \rightarrow R$ và

$$f(x, y, z) = ax + by + cz + d. \qquad (2.7)$$

với m mặt. Cho mỗi mặt của O, một nửa - không gian H_i được định nghĩa như một tập con của W:

$$H_i = \{(x, y, z) \in W \mid f_i(x, y, z) \leq 0\} \qquad (2.8)$$

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 việc xuất phát đi quanh biên theo thứ tự ngược chiều kim đồng hồ. Trong trường hợp của một đa diện, ranh giới của mỗi mặt là các cạnh cũng được lấy ngược chiều kim đồng hồ; ***(Hình 2.13b)***.



Hình 2.13: ***(a)Đa diện. (b) biểu diễn các cạnh của mỗi mặt trong đa diện.***

Phương trình cho mỗi mặt được xác định như sau: Chọn ba đỉnh liên tiếp p_1, p_2, p_3 (không cộng tuyến) theo thứ tự ngược 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 $v= v_{12}xv_{23}$ là 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 được 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.

Trong trường hợp đa giác mẫu, một đa diện lồi có thể được định nghĩa như phép giao của một số hữu hạn của các nửa - không gian tương ứng với mỗi mặt. Một đa diện không lồi có thể được định nghĩa như phép hợp của một số hữu hạn các đa diện lồi. Vị từ ϕ(x, y, z) có thể được định nghĩa tương tự là TRUE nếu (x, y, z)$\in$ O, và FALSE trong trường hợp ngược lại.

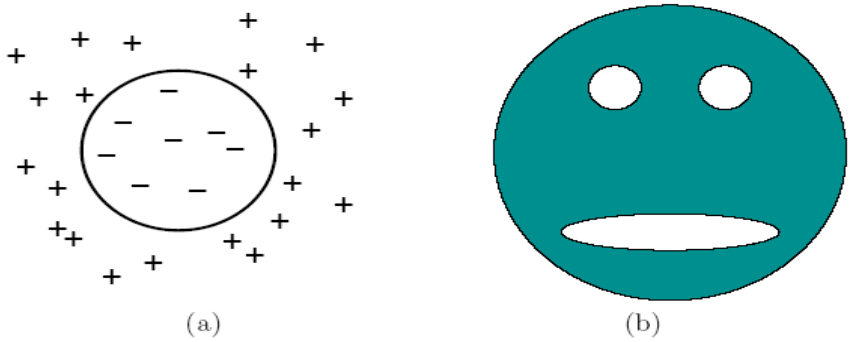
2.4.2.2. Mô hình nửa đại số.

Trong những mô hình đa giác và đa diện, f là một hàm tuyến tính. Trong mô hình nửa đại số đối với không gian 2D, f là đa thức với những hệ số bất kỳ và hai biến thực x và y. Trong không gian 3 chiều, f là một đa thức với ba biến thực x, y, z. Lớp mô hình nửa đại số bao gồm cả hai mô hình đa diện và đa giác, mà sử dụng trước hết cho đa diện. Một tập hợp điểm xác định bởi một mẫu đa thức đơn được gọi là tập hợp đại số; Một tập hợp điểm có thể thu

được bởi một số hữu hạn của những phép hợp và phép giao những tập hợp đại số được gọi một tập nửa đại số.

Xem xét trường hợp của không gian 2D. Một biểu diễn 3 chiều có thể được định nghĩa như sau:

$$H = \{(x, y) \in W \mid f(x, y) \leq 0\} \qquad (2.9)$$



Hình 2.14 : (a) sử dụng f để phân chia R^2 thành hai vùng. (b) sử dụng bốn mẫu đại số để mô hình hoá vùng mặt.

Ví dụ 1 : cho $f = x^2 + y^2 - 4$. Trong trường hợp này, H đại diện đường tròn bán kính r=2, tâm được đặt đúng ở gốc. Điều này tương ứng tới tập hợp của những điểm (x, y) cho f(x, y) = 0, điều này được miêu tả trong **Hình 2.14a.**

Ví dụ 2 : (khuôn mặt) xem xét việc xây dựng một mô hình của vùng đậm màu trong ***Hình 2.14b***. Giả sử vòng tròn ngoài có bán kính r_1 và tâm được đặt tại gốc. Giả thiết " đôi mắt " có bán kính r_2 và r_3 và tâm tương ứng là (x_2, y_2) và (x_3, y_3) cho " miệng " là một hình ê-líp với trục chính a và trục phụ b tâm là (0, y_4).

Khi đó các hàm được định nghĩa như sau:

$$\begin{aligned} f_1 &= x^2 + y^2 - r_1^2 \\ f_2 &= -((x - x_2)^2 + (y - y_2)^2 - r_2^2) \\ f_3 &= -((x - x_3)^2 + (y - y_3)^2 - r_3^2) \\ f_4 &= -(x^2 / a^2 + (y - y_4)^2 / b^2 - 1) \end{aligned} \qquad (2.10)$$

f_2, f_3, và f_4, là những phương trình đường tròn và hình ê-líp được nhân với - 1 để sinh ra những mẫu đại số cho tất cả các điểm bên ngoài đường tròn hoặc hình ê-líp. Vùng O đậm màu được tương ứng như sau:

$$O = H_1 \cap H_2 \cap H_3 \cap H_4 \qquad (2.11)$$

Trong trường hợp của những mô hình nửa đại số, phép giao của các mẫu không nhất thiết có kết quả trong một tập con lồi W. Nhìn 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 trường hợp không gian 3 chiều.

Ta có dạng đại số nguyên thuỷ của mẫu :

$$H = \{(x, y, z) \in W \mid f(x, y, z) \leq 0\} \qquad (2.12)$$

Có thể sử dụng để định nghĩa một biểu diễn của chướng ngại 3 chiều O và một vị từ lôgíc ϕ. Phương trình (3.10) và (3.13) đủ để biểu thị bất kỳ mô hình nào mà ta cần quan tâm. Có thể định nghĩa mẫu theo nhiều cách khác nhau dựa vào những quan hệ khác nhau, như :

$f(x, y, z) \geq 0, f(x, y, z) = 0, f(x, y,z) < 0, f(x, y, z) = 0,$ và $f(x, y, z) \neq 0$

Xét mẫu:

$$H = \{(x, y, z) \in W \mid f(x, y, z) \geq 0\} \qquad (2.13)$$

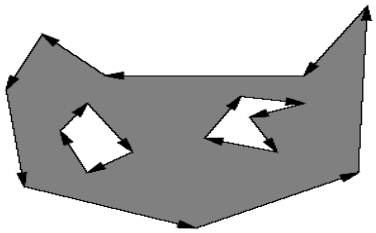
Có thể biểu diễn theo cách khác như - f(x, y, z) $\leq$ 0, và - f có thể được xem xét như một hàm đa thức mới của x, y, z. Một ví dụ qua hệ bằng:

$$H = \{(x, y, z) \in W \mid f(x, y, z) = 0\} \qquad (2.14)$$

Có thể thay H = H1 $\cap$ H2, với H_1: $H = \{(x, y, z) \in W \mid f(x, y, z) \leq 0\}$ (2.15)

và H_2 $H = \{(x, y, z) \subset W \mid f(x, y, z) \leq 0\}$ (2.16)

Quan hệ $<$ tăng thêm sức mạnh sẽ rất có ý nghĩa khi xây dựng những mô hình không chứa đường biên ngoài. Chú ý rằng phần đậm màu luôn luôn ở bên trái khi đi theo những mũi tên.



Hình 2.15 : ***Biểu thị một đa giác với những lỗ. Ngược chiều kim đồng hồ cho biên ngoài và thuận chiều kim đồng hồ cho biên trong***

2.4.3. Không gian cấu hình chướng ngại

Một giải thuật lập lộ trình chuyển động phải tìm thấy một đường dẫn trong không gian rỗng (Free Space) từ cấu hình ban đầu(qI) đến cấu hình đích (qG).

2.4.3.1. Định nghĩa cấu hình chướng ngại.

Đầu chương chúng ta đã có khái niệm sơ khai về cấu hình không gian chướng ngại vật. Bây giờ chúng ta sẽ nghiên cứu chi tiết hơn về vấn đề này.

1-Vùng chướng ngại cho một vật thể

Giả thiết không gian W = R^2 hoặc W = R^3, Chứa đựng một vùng chướng ngại O$\subset$ W. Đồng thời cũng giả thiết ở đây một robot cứng, A $\subset$W, (A và O được trình bày như những mô hình nửa đại số (bao gồm những mô hình đa diện và đa giác). Cho q$\in$ C biểu thị cấu hình của A, trong đó q=(x_t, y_t, θ) với W = R^2 và q = (x_t, y_t, z_t, h) với W = R^3 (h là đơn vị quaternion).

Vùng chướng ngại, Cobs $\subseteq$ C, được định nghĩa như sau:

$$C_{obs} = \{q \in C \mid A(q) \cap O \neq \phi\} \qquad (2.17)$$

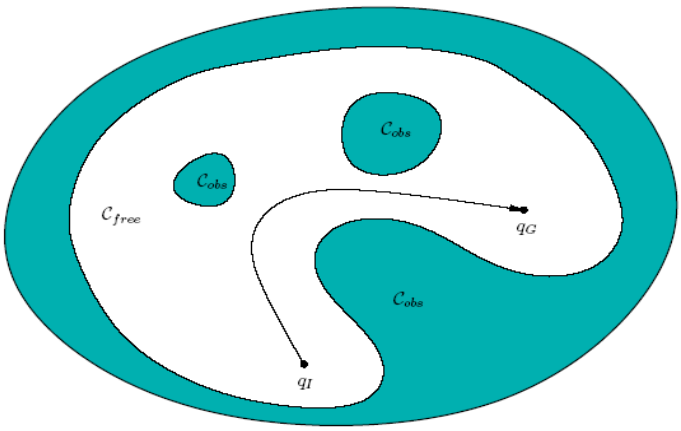
C_{obs} là tập hợp của tất cả các cấu hình q, ở đó A(q) (trạng thái của robot tại cấu hình q) giao với vùng chướng ngại O. O và A(q) là những tập hợp đóng bên trong W, vùng chướng ngại là một tập hợp đóng trong C. Những cấu hình còn lại được gọi không gian trống, mà được định nghĩa và C_{free}= C\C_{obs}.

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 chướng ngại một cách tuỳ ý trong những phần của C_{free} miễn là đường biên của chúng không giao nhau.

$$int(O) \cap int(A(q)) = \phi \quad and \quad O \cap A(q) \neq \phi \qquad (2.18)$$

Nếu A chạm vào O thì $q \in C_{obs}$. Điều kiện nhận biết duy nhất là những đường biên của chúng cắt nhau.Ý tưởng robot có thể đến gần những chướng ngại một cách tuỳ ý có thể không có ý nghĩa thực tiễn trong kỹ thuật rôbôt, nhưng nó làm cho những giải thuật lập lộ trình chuyển động trở nên minh bạch. Khi C_{free} mở, nó không thể đạt được sự tối ưu như tìm kiếm đường ngắn nhất. Trong trường hợp này, tập đóng, $cl(C_{free})$, cần phải thay vào để sử dụng.

2- Chướng ngại vật có nhiều vật thể: Nếu robot phức tạp thì vấn đề trở nên rắc rối hơn, ở đây chúng ta chỉ nghiên cứu giải thuật với các robot điểm.



Hình 2.16 : ***C-space và nhiệm vụ***
tìm đường từ qI đến qG trong C_{free}. $C = C_{free} \cup C_{obs}$.

Chúng ta sẽ dùng ký hiệu Ai cho mối liên kết i, mặc dù vài tham số của q có thể không thích hợp cho mối liên kết chuyển động Ai. Ví dụ, trong một dây chuyền động học, cấu hình của vật thể thứ hai không phụ thuộc vào góc giữa vật thể thứ chín và vật thể thứ mười.

Gọi P là tập hợp của những cặp va chạm, trong đó mỗi sự va chạm là một cặp đôi, $(i,j) \in$ P,trong đó i,j là chỉ số mối liên kết và $i, j \in \{ 1, 2, \ldots, m \}$,

với $i \neq j$. Nếu (i,j) xuất hiện bên trong P, nó có nghĩa là A_i và A_j chưa cho phép để trong một cấu hình, q, nghĩa là $A_i(q) \cap A_j(q) \neq \phi$

Cho m vật thể, P thông thường có kích thước $O(m^2)$; Tuy nhiên, trong thực tế có thể để loại trừ nhiều cặp bởi sự phân tích hình học nào đó của sự kết nối.

Sử dụng P, xem xét những sự va chạm của robot có thể định nghĩa :

$$C_{obs} = \left(\bigcup_{i=1}^{m} \{ q \in C \mid A_i(q) \cap O \neq \phi \} \right) \cup \left(\bigcup_{[i,j] \in P} \{ q \in C \mid A_i(q) \cap A \}(q) \neq \phi \right) \quad (2.19)$$

Như vậy, một cấu hình $q \in C$ trong C_{obs} nếu tồn tại ít nhất một mối liên kết va chạm với O hoặc một cặp trong những mối liên kết P va chạm với O.

2.4.4. Định nghĩa chính xác về vấn đề lập lộ trình chuyển động.

Cuối cùng ta đã đủ công cụ để định nghĩa chính xác vấn đề lập lộ trình. Cụ thể bài toán này có các thành phần chính sau:

1. Một không gian W là một trong hai trường hợp $W = R^2$ hoặc $W=R^3$.

2. Một vùng chướng ngại là một mô hình nửa đại số $O \subset W$ trên không gian.

3. Một robot cũng là một mô hình nửa đại số được định nghĩa trong W. Nó có thể là một robot đơn A hoặc là một tập hợp của m những mối liên kết $A_1, A_2, . , A_m$.

4. Không gian C cấu hình xác định bởi việc chỉ rõ tập hợp của tất cả những sự biến đổi có thể được áp dụng cho robot được dẫn xuất từ C_{obs} và C_{free}.

5. Trong một cấu hình, $qI \in C_{free}$ là trạng thái ban đầu.

6. Trong một cấu hình, $qG \in C_{free}$ được chỉ định là trạng thái đích. Một cặp cấu hình ban đầu và cấu hình đích thường được gọi một cặp truy vấn (hoặc truy vấn) ký hiệu là (qI, qG).

7. Một giải thuật phải tính toán thiết lập được một đường dẫn liên tục đầy đủ từ qI đến qG:

τ : [0, 1]$\rightarrow$ C_{free}, như vậy $\tau(0)$ = qI và $\tau(1)$ = qG, hoặc phải chỉ ra rằng một đường dẫn như vậy không tồn tại.

CHƯƠNG 3

ỨNG DỤNG MẠNG NƠRON NHÂN TẠO TRONG BÀI TOÁN LẬP LỘ TRÌNH CHO ROBOT.

3.1. MẠNG NƠRON NHÂN TẠO VÀ BÀI TOÁN LẬP LỘ TRÌNH.

Những năm trước đây, một số nghiên cứu về khoa học máy học được trình bày và đã được áp dụng để hướng dẫn robot cải thiện các khả năng và thao tác. Một trong những vấn đề quan trọng trong thiết kế và phát triển hệ thống người máy di động thông minh là khả năng tìm đường, điều này bao gồm cả khả năng lập lộ trình trong môi trường hoạt động của nó. Tuy nhiên, môi trường hoạt động của robot có thể không chính xác, rộng lớn, thay đổi hoặc có cấu trúc không rõ ràng. Người máy phải hiểu rõ về cấu trúc môi trường và đi được đến đích mà không có sự va chạm, điều này đòi hỏi người máy phải có khả năng nhận thức, xử lý dữ liệu, đoán nhận, học, suy luận, hành động và ra quyết định. Những khả năng này được xây dựng dựa trên chìa khóa là một loại trí tuệ nhân tạo, tái sản xuất loại trí tuệ này, cho tới nay vẫn là một tham vọng mà con người mong muốn đạt tới để xây dựng và phát triển những hệ máy thông minh, đặc biệt là đối với sự chuyển động của người máy.

Để đạt được sự hợp lý trong các hành động tự động của mình thì người máy đòi hỏi phải có hai khả năng cơ bản là: Cảm nhận và suy luận, dựa trên những thông tin về các trạng thái lân cận thu nhận được bởi hệ thống cảm biến. Những khả năng này được thể hiện qua các giải thuật và căn cứ vào các giải thuật này để điều khiển robot hoạt động.

Trong kỹ thuật robot thì việc xác định lộ trình là một công việc quan trọng. Ta có thể hiểu khái quát bài toán xác định lộ trình như sau: Cho đối tượng với vị trí ban đầu và vị trí đích với một tập các chướng ngại vật có các

vị trí khác nhau trong không gian làm việc. Công việc xác định lộ trình là tìm ra một đường đi liên tục từ vị trí ban đầu đến vị trí đích sao cho tránh được những va trạm với những vật cản trên đường đi. Thông thường quá trình xác định lộ trình có thể chia làm hai thao tac chính, đó là: xây dựng không gian tìm kiếm và tìm đường. Ta có thể tham khảo các cách tiếp cận liên quan tới việc lập lộ trình chuyển động trong (Latombe, J.C1991)

+ Không gian tìm kiếm là xây dựng các cấu hình các mối quan hệ của đối tượng đã cho và các chướng ngại vật.

+ Quá trình tìm đường là xác định một đường đi từ vị trí đầu đến vị trí đích sao cho tránh được sự va chạm với các vật cản.

Nhiều phương pháp sử dụng ý tưởng xây dựng không gian cấu hình đã được đề xuất để giải quyết bài toán tìm đường. Tuy nhiên các phương pháp này cũng tỏ rõ một số nhược điểm như: những tính toán để tạo ra không gian cấu hình từ cấu trúc của người máy và các chướng ngại vật rất phức tạp, số bước tìm kiếm sẽ tăng theo cấp luỹ thừa với số nút tương ứng. Những nhược điểm nêu trên chính là động lực để các nhà khoa học nghiên cứu giải pháp mới đó là sử dụng những giải thuật song song với tốc độ tính toán được cải tiến để giải quyết bài toán trên.

Mạng nơron là một cấu trúc mạng cho phép dữ liệu được xử lý song song và phân tán gần như đồng thời trên nhiều nơron. Do đó, giải pháp sử dụng mạng nơron giải quyết bài toán lập lộ trình là một hướng đi đúng đắn. Chương này, sẽ giới thiệu một số cách tiếp cận sử dụng mạng nơron để lập lộ trình cho robot di chuyển tự do giữa những chướng ngại vật đã biết trong cấu trúc môi trường hoạt động của robot.

3.2. ỨNG DỤNG MẠNG HOPFIELD GIẢI QUYẾT BÀI TOÁN LẬP LỘ TRÌNH CHO ROBOT.

3.2.1. Khái quát một số phương pháp lập lộ trình và khả năng ứng dụng của mạng Hopfield.

Một trong những vấn đề then chốt của hệ thống robot di động là lập lộ trình chuyển động trong thời gian thực. Việc di chuyển đòi hỏi Robot phải có khả năng tránh các chướng ngại vật và tìm đường đi tới đích. Trong lĩnh vực này có một vài cách tiếp cận, xung quanh vấn đề môi trường tĩnh và động.

Một trong những phương pháp tìm đường tự động là phương pháp wallfollowing. Trong phương pháp này, Robot tự động tìm đường dựa vào chuyển động dọc theo những bức tường ở một khoảng cách đặt sẵn, trong khi xem xét những chướng ngại chỉ là những bức tường khác. Tuy nhiên, với những tính toán đơn giản, cách tiếp cận này chỉ có những ứng dụng đối với một số robot không đòi hỏi phải có những chuyển động phức tạp như: Robot dọn dẹp sàn nhà trong một hành lang dài. Một sự cải tiến dựa trên phương pháp này là cách tiếp cận dò tìm mép, nơi những vị trí các đường biên thẳng đứng của chướng ngại được định rõ và robot di chuyển xung quanh tất cả các mép đó. Tuy nhiên cách tiếp cận này phụ thuộc quá lớn vào sự chính xác của bộ sensor(cảm biến). Sự phối hợp có tác động lớn tới khả năng tự động tìm đường là phương pháp Potential Fields, được đề xuất bởi Khatib. Ở đây, những chướng ngại được xem như những tâm điểm đẩy ra và những đích như những tâm điểm hấp dẫn. Robot đi ngang qua con đường của đường dốc ít khả năng nhất. Một nhược điểm của phương pháp này là phải giả thiết rằng mô hình của những chướng ngại vật phải được biết trước.

Những nhược điểm nêu trên sẽ phần nào được khắc phục khi sử dụng mạng nơron. Những cách tiếp cận Mạng Nơron đã được sử dụng trong khá nhiều thuật toán lập lộ trình và phần này sẽ trình bày cách tiếp cận dựa trên

mạng nơron hồi quy Hopfield. Cụ thể, ta đã sử dụng một mô hình lỏng lẻo dựa vào học cạnh tranh. Các chướng ngại vật có thể xuất hiện ở những vị trí bất kì trên đường di chuyển của Robot và đương nhiên chúng cũng có hình giáng tuỳ ý. Không giống đa số các cách tiếp cận trước đây, phương pháp này không yêu cầu bức tranh toàn cảnh về lộ trình của robot. Mỗi nơron trong mạng nơron chỉ có những kết nối cục bộ, chúng liên tục cập nhật để thể hiện những hoạt động được phát sinh trong môi trường hoạt động từ đó hướng dẫn Robot về đích. Tất cả điều này xảy ra trong thời gian thực và khi đó quá trình xác định đường đi sẽ diễn ra một các nhanh chóng.

3.2.2. Phương pháp do Yang và Meng đề xuất.

Ý tưởng

Mô hình của chúng ta thực chất dựa vào nơron được mô phỏng theo cách thức hoạt động của nơron sinh học. Phần này sẽ trình bày những mô hình mạng được phát triển bởi Yang và Meng. Mô hình này sẽ là cơ sở để giải quyết bài toán trên.

Một nơron có n đầu vào. Khi đó, kiến trúc mạng sẽ tương ứng tới một không gian cấu hình Robot N- chiều. Môi trường hoạt động Nơron về cơ bản là môi trường động trong không gian cấu hình ζ. Những nơron được sắp đặt trong ngữ cảnh rời rạc của không gian cấu hình ζ. Nơron i được gắn liền với đại lượng x_i là tín hiệu vào tại nơron i và các tín hiệu từ bên ngoài I_i. Phương trình (1) xác định trạng thái của nơron i:

$$\frac{dx_i}{dt} = -Ax_i + (B - x_i)([I_i]^+ + \sum_{j=1}^{k} w_{ij}[x_j]^+) - (D + x_i)[I_i]^- \qquad (3.1)$$

Những tham số *A, B* và *D* cho biết tốc độ thấp, cao và rất cao của hoạt động thần kinh, tương ứng. Biến x_i là một biến liên tục $\in [-D, B]$. Giá trị *I=E* nếu nơron thứ i tương ứng tới một đích, *I = - E* nếu nó là một chướng ngại, trong các trường hợp khác *I = 0*. (I là tín hiệu từ bên ngoài _bias) *E* là một số

dương tương đối lớn. $[I_i]^-$ là kìm hãm được nhập vào được gây ra bởi những chướng ngại, trong khi $([I_i]^+ + \sum_{j=1}^{k} w_{ij}[x_j]^+)$ Là kích thích nhập vào cho đích và kết nối cục bộ giữa những nơron.

- Hàm phi tuyến $[a]^+$ và $[a]^-$ được định nghĩa như sau:

$$[a]^+ = Max\{a, 0\}$$

$$[a]^- = Max\{-a, 0\}.$$

- q_i và q_j là véc tơ xác định vị trí của nơron i và j tương ứng.

- Trọng số w_{ij} tương ứng của đầu vào i với đơn vị j được xác định như sau:

$w_{ij} = f(|d_{ij}|)$, ở đây $d_{ij} = |q_i - q_j|$ Là khoảng cách giữa i và những nơron q trong ζ.

f(a) là hàm kích hoạt. Hàm này đã được Yang và Meng sử dụng.

$$f(a) = \begin{cases} \dfrac{\mu}{a} & 0 < a < r_0 \\ 0 & a \geq r_0 \quad va \quad a \leq 0 \end{cases} \qquad (3.2)$$

μ là một số dương. Hàm chức năng này bảo đảm rằng một nơron có những kết nối cục bộ trong một vùng nhỏ xung quanh nó với bán kính là r_0.

Như vậy hiệu ứng của những chướng ngại là đại lượng cục bộ còn hiệu ứng của đích là đại lượng toàn cục. Sự chuyển động của Robot được quyết định bởi việc chọn một nơron chiếm ưu thế trong không gian của các nơron. Cho a được định vị trong S, vị trí này biểu thị bởi q_p, định vị vị trí q_n tiếp theo bằng cách gọi lệnh định vị.

$$q_n \Leftarrow x_{q_n} = max(x_i, i = 1,2,...,k) \qquad (3.3)$$

Trong đó *k* là số những nơron lân cận. Bước tiếp theo ta lại tiếp tục xem xét những nơron lân cận với nơron tương xứng với vị trí hiện tại của Robot và

Robot sẽ di chuyển tới vị trí xác định bởi một nơron lân cận chiếm ưu thế tiếp theo (bao gồm cả nơron tương xứng với vị trí hiện thời).

Tóm tắt phương pháp

- Ký hiệu:

  qI: Vị trí đầu.

  qG: Vị trí đích.

  q_p: Vị trí hiện hành.

  q_n: Vị tí tiếp theo.

  Kt: dùng để kiểm tra xem có còn xác định được vị trí tiếp theo khác với vị trí hiện hành không.

- Phương pháp

  B1: Khởi tạo

  + Xác định vị trí đầu qI

  + Xác định vị trí đích qG

  + Gán q_p=qI

  B2: Chừng nào q_p<>qG và KT=true thì còn làm các công việc sau:

  + Tính các x_i căn cứ vào phương trình động học (3.1) và hàm f(a) (3.2)

  + Xác định vị trí tiếp theo q_n dựa vào (3.3)

  + q_p= q_n

  B3: Nếu q_p=qG thì

  + Thông báo xác định thành công một lộ trình

  Ngược lại

  + Thông báo không xác định được lộ trình

  B4: Kết thúc

Nhận xét:

Trước khi thảo luận về những kết quả của phương pháp đang nghiên cứu, ta nên xem xét các phương trình (3.1), (3.2) và (3.3), đây sẽ là cơ sở để trình bày các vấn đề tiếp theo.

i. Theo (3.3), một nơron với giá trị x_i càng cao sẽ càng có khả năng thu hút Robot, trong khi một nơron với một giá trị thấp hơn sẽ đẩy lùi robot.

ii. B và - D là cận trên và cận dưới của x_i.

iii. Số hạng thứ hai sử dụng toán tử $[\ ]^+$, Số hạng này hiển thị những ảnh hưởng của những giá trị dương tính (Đây có thể là những đích tiềm tàng hoặc những điểm thu hút I_i và x_j). Số hạng thứ ba sử dụng toán tử $[]^-$ là ảnh hưởng của những giá trị ngược (những chướng ngại tiềm tàng hoặc những phản xạ của I_i).

iv. Chính trong Số hạng thứ hai ảnh hưởng của những nơron lân cận được xem xét. Giá trị này thể hiện những đặc điểm trong khi tính đến ảnh hưởng (của) những nơron lân cận thông qua số lượng $[\ x_i\ ]^+$ đã được chọn. Điều này bảo đảm rằng những nơron nối ngược lân cận sẽ không ảnh hưởng đến hoạt động của nơron i. Điều này có nghĩa là mặc dầu có những tín hiệu về một đích nào đó được truyền lan từ một nơron khác tới lân cận của nó song thông tin về một chướng ngại vật sẽ không thể sinh ra theo phương thức tương tự như trên nếu ta áp dụng phương trình (3.1). Điều này thể hiện sự thu hút robot về phía đích và khước từ chướng ngại vật trên đường chúng di chuyển. Trong thực tế, khi được đặt trong một mê cung phức tạp với một đích ở xa robot thường vấp phải chướng ngại vật hoặc tìm cách đi xuyên qua nó. Những ví dụ dưới đây sẽ thể hiện rõ hơn về những nhận xét trên.

v. Phép toán cộng trong (3.1) được thực hiệu qua tất cả các nơron lân cận kể cả chính nó. Từ (3.2) ta có $w_{ii}=f(0)=0$ do $d_{ii}=0$.

vi. Theo (3.3), vị trí mới của robot sẽ được xác định bằng cách tìm ra nơron có giá trị tín hiệu đầu vào cao nhất, kể cả bản thân nó. Vì vậy robot không thể lùi lại khi gặp một mê cung phức tạp phía trước trên lộ trình của mình. Trong trường hợp xấu nhất robot có thể tự giao động quanh một vị trí nào đó. Từ đó ta có thể thấy rõ sự logic của việc xác định sự ưu tiên của các nơron mà robot đi qua.

3.2.3. Mô hình Yang và Meng cải tiến

Ý tưởng

Dựa vào những nhận xét nêu trên mô hình có thể được sửa đổi như sau:

+ Theo iv ta cần sửa đổi (1) như sau:

$$\frac{dx_i}{dt} = -Ax_i + (B - x_i)([I_i]^+ + \alpha\sum_{j=1}^{k} w_{ij}[x_j]^+) - (D + x_i)([I_i]^- + \beta\sum_{j=1}^{k} w_{ij}[x_j]^-) \quad (3.4)$$

Trong đó: $0 \le \alpha \le 1, \quad 0 \le \beta \le 1$

Chú ý: $\alpha = 1, \quad \beta = 0$ ứng với mô hình nguyên bản của Yang và Meng

+ Để giải quyết những vấn đề đã đề cập trong vi, một kỹ thuật mới được đưa ra. Để đảm bảo cho Robot có khả năng quay lại vị trí thước đó mà nó đã đi qua, ngoài giá trị nhập vào I_i ta cần đưa vào thêm nột giá trị để xác định quá trình quay trở lại của robot (gọi giá trị này là giá trị ngược).Ta có thể lấy ngay ví dụ trong mô phỏng trên như sau: Vị trí đến thăm vào lần cuối cùng ta cho giá trị ngược là -E / 8, vị trí đến thăm trước đó 2 bước ta xác định giá trị ngược là -E / 16 và vị trí đến thăm trước đó 3 bước thì giá trị được gán là -E / 32... Tất cả những vị trí đã duyệt qua ta đều xác định giá trị ngược này và cập nhật cho I_i tương ứng. Điều này cũng đảm bảo robot sẽ không bị kẹt tại một vị trí như đã nêu ở vi. Mà tại mỗi bước robot bắt buộc phải di chuyển tới một vị trí nào đó hiệu quả hơn.

Tóm tắt phương pháp

- Ký hiệu:

qI: Vị trí đầu.

qG: Vị trí đích.

q_p: Vị trí hiện hành.

q_n: Vị tí tiếp theo.

T_i: giá trị ngược của nơron i

Kt: dùng để kiểm tra xem có còn xác định được vị trí tiếp theo không

- Phương pháp

B1: Khởi tạo

+ Xác định vị trí đầu qI

+ Xác định vị trí đích qG

+ Khởi tạo T_i

+ Gán q_p=qI

B2: Chừng nào q_p<>qG và KT=true thì còn làm các công việc sau:

+ Tính các x_i căn cứ vào phương trình động học (3.4) và hàm f(a) (3.2)

+ Xác định vị trí tiếp theo q_n dựa vào (3.3)

+ q_p= q_n

+ Xác định giá trị ngược T_n

+ I_n= T_n

B3: Nếu q_p=qG thì

+ Thông báo xác định thành công một lộ trình

Ngược lại

+ Thông báo không xác định được lộ trình

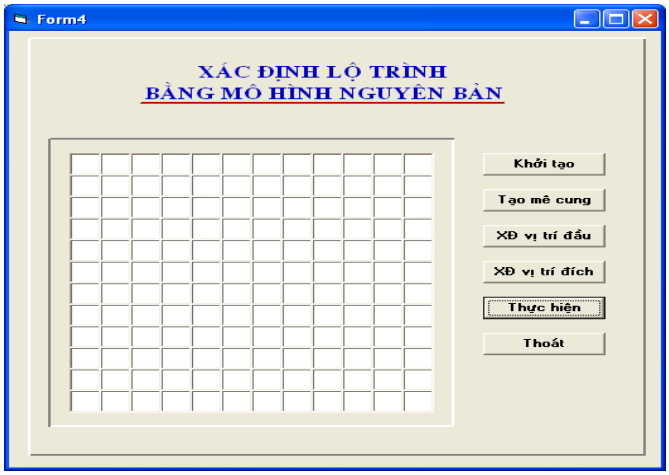
B4: Kết thúc

3.3. CÁC KẾT QUẢ THỬ NGHIỆM

3.3.1. Chương trình đề mô

Với những lý thuyết đã nghiên cứu, tác giả đã cài đặt mô hình thử nghiệm bằng ngôn ngữ Visual Basic.

Môi trường làm việc của robot được thể hiện bằng các mê cung, các mê cung này được tạo thành dựa trên lưới với các ô định trước. Cụ thể giao diện chương trình được trình bày trong hình 3.1 và hình 3.2:



Hình 3.1. Giao diện chương trình mô hình nguyên bản.

Himh 3.2. Giao diện chương trình mô hình cải tiến.

Chức năng và hoạt động của các nút lệnh trong chương trình:

+ Nút khởi tạo: Đưa lưới trở về trạng thái sẵn sàng để tạo một mê cung và xác định một lộ trình mới.

+ Nút tạo mê cung: Sau khi click vào nút này ta có thể click vào các ô trên lưới để tạo mê cung.

+ Nút XĐ vị trí đầu: Cho phép xác định vị trí xuất pháp của robot.

+ Nút XĐ vị trí cuối: Cho phép xác định vị trí đích.

+ Nút Thực hiện: Khi click nút này trên lưới sẽ hiển thị sự di chuyển của robot và lộ trình tương ứng.

Để cài đặt các chương trình thử nghiệm ta cần phải lựa chọn các tham số cho phù hợp. Trong chương trình đề mô các tham số được lựa chọn như sau:

+ Mô hình nguyên bản:

A=10, B=D =1, μ=1, r_0=2 và E = 100.

+ Mô hình cải tiến:

E = 100, A = 10, B = 5, D = 1, r_0 = 1.41, α = 0.9 và β = 0.2.

Chương trình cài đặt bằng ngôn ngữ Visual basic sẽ thể hiện trực quan các kết quả mô phỏng vì nó cung cấp nhiều công cụ để thể hiện giao diện đồ hoạ sinh động thân thiện với người sử dụng.

Robot thể hiện trong chương trình đềmô là các robot điểm còn trong thực tế với những robot phức tạp ta cần phải tính đề tác động của cấu trúc của robot đối với việc di chuyển và lộ trình mà not xác định.

Khi cải đặt chương trình trong thực tế các tham số nhập từ môi trường ngoài sẽ được robot thu nhận thông qua các thiết bị cảm biến trong quá trình di chuyển của nó. Với phương pháp này robot cũng không cần phải hiểu cụ thể về cầu trúc môi trường cũng như các chướng ngại vật trong quá trình di chuyển nó sẽ căn cứ vào thuật toán mà xác định vị trí khả thi để di chuyển tới. Với phương pháp này thì công việc khó khăn là xây dựng cấu hình không gian và cấu hình chướng ngại trình bày trong chương 2 là không còn cần thiết, do đó sẽ đẩy nhanh được tốc độ xác định lộ trình.

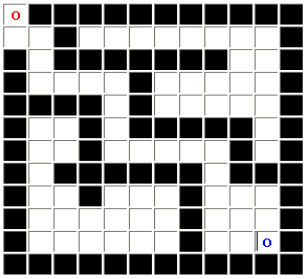
3.3.2. So sánh các kết quả

Trong mục này ta sẽ đưa ra các kết quả so sánh giữa mô hình nguyên bản và mô hình sửa đổi, trong nhiều trường hợp mê cung và đích có độ phức tạp khác nhau.

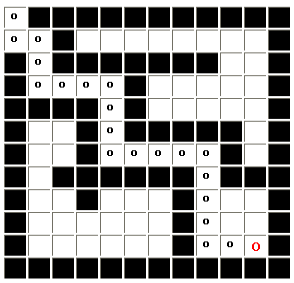
Cho hai robot điểm di chuyển trong hai mê cung có cấu trúc giống hệt nhau. Robot 1 sử dụng phương pháp lập lộ trình do Yang và Meng đề xuất. Robot 2 sử dụng mô hình cải tiến để xác định lộ trình. Các dấu tròn đánh dấu con đường mà robot đã đi qua. Ta sẽ quan sát quá trình di chuyển của robot và đưa ra những nhận xét cho hai mô hình.

Từ những kết quả của chương trình mô phỏng ta có thể thấy rõ tính khả thi cũng như hiệu quả của từng phương pháp. Đồng thời thấy được những ưu nhược điểm của chúng. Từ đó có thể lựa chọn chính xác các phương án trong các trường hợp cụ thể.

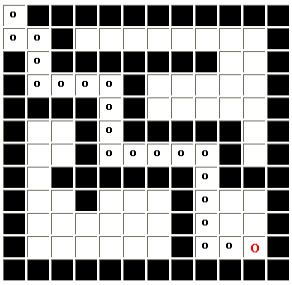
+ Mê cung 1



Hình 3.3a. Mê cung 1

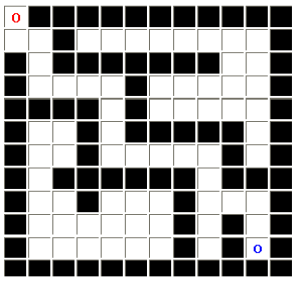


Hình 3.3b. Mô hình nguyên bản

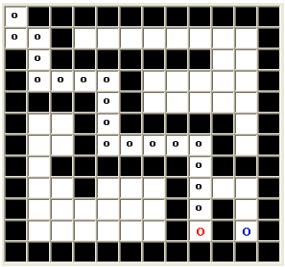


Hình 3.3c. Mô hình sửa đổi

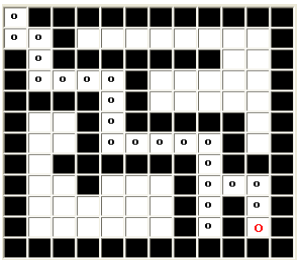
+ Mê cung 2



Hình 3.4a. Mê cung

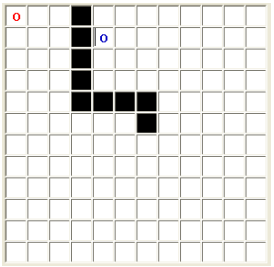


Hình 3.4b. Mô hình nguyên bản

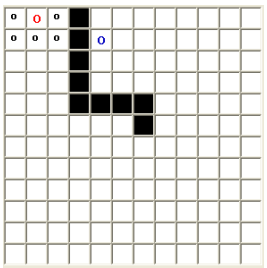


Hình 3.4c. Mô hình sửa đổi

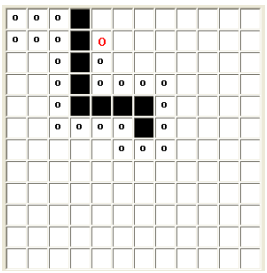
+ Mê cung 3



Hình 3.5a. Mê cung



Hình 3.5b. Mô hình nguyên bản



Hình 3.5b. Mô hình sửa đổi

Nhận xét:

+ Trong mê cung 1 đường đi tới đích là khá dễ dàng. Khi đó cả hai mô hình đều dẫn dắt robot tới đích và hai mô hình đều thực hiện tốt như nhau.

+ Trong mê cung 2 ta đặt một vật cản chia đôi khu vực đích. Với mô hình nguyên bản robot mắc bẫy và không hoàn thành sứ mệnh của mình. Trong khi mô hình sửa đổi có thể hướng dẫn robot tránh bẫy bằng cách loại trừ những noron hiện thời và bắt buộc robot đi qua con đường có thể băng qua vật cản để đến đích.

+ Trong mê cung 3 là một trường hợp đơn giản. Tuy nhiên với mô hình nguyên bản thì lại gặp trở ngại và robot không thể đi được đến đích. Truy nhiên phương trình (3.4) lại khắc phục được điều này khi đó robot sẽ xoay sở và tìm cách đi xung quanh chướng ngại vật để đến đích.

+ Từ các kết quả trên ta cũng nhận thấy việc lựa chọn các tham số cũng là công việc khá quan trọng, nó quyết định quá trình xác định các lộ trình. Song làm cách nào để lựa chọn được các tham só tối ưu là một vấn đề khá khó khăn và đây cũng có thể coi là một hướng mở của luận văn.

+ Mô hình nguyên bản có thể không xác định được lộ trình mặc dù thực tế lộ trình vẫn tồn tại. Phương pháp cải tiến sẽ luôn xác định được một lộ trình nếu tồn tại đường đi từ vị trí đầu đến vị trí đích.

3.3.3. Kết luận

Phương pháp nguyên bản của Yang và Meng đã trình bày ở trên được phát triển với mục đích điều khiển các thiết bị máy móc chuyển động tránh các vật cản. Đây là một mô hình lỏng lẻo với nền tảng là học cạnh tranh được ánh xạ vào một mạng nơron. Trong đó, mỗi nơron chỉ có những kết nối cục bộ và phương pháp này không đòi hỏi bức tranh toàn cảnh về môi trường hoạt động của robot khi xác định lộ trình.

Tuy nhiên phương pháp này sẽ kém hiệu quả khi robot hoạt động trong những môi trường phức tạp. Để khắc phục điều này người ta đã cải tiến phương pháp của Yang và meng bằng cách thay đổi một số tham số trong (3.1), khi đó ta có phương trình (3.4). Phương pháp này cho phép robot xác định được lộ trình và tránh được bẫy trên đường di chuyển.

Để mô phỏng hoạt động của hai mô hình nêu trên, người ta đã thể hiện hoạt động của robot trong 3 mê cung với cấu trúc khác nhau. Khi đó rõ ràng phương pháp cải tiến tỏ ra rất hiệu quả. Trong khi, phương pháp nguyên bản chỉ tìm ra đường đi trong mê cung 1 thì phương pháp cải tiến đã giúp robot xoay sở và tìm ra đường tới đích trong cả 3 mê cung. Với phương pháp cải tiến tại mỗi bước robot bắt buộc phải tiến tới một vị trí khác có nhiều khả năng tiến tới đích hơn điều này giúp robot không bị kẹt tại một điểm và tăng khả năng di chuyển đến đích của robot. Tuy nhiên, những phương pháp này vẫn thể hiện một nhược điểm đó là khi robot gặp trở ngại và không thể đến đích chúng bắt buộc phải xác định lại toàn bộ lộ trình và điều này hoàn toàn có thể được khắc phục bằng cách lựa chọn những tham số tối ưu để có một lộ trình tốt nhất. Việc xác định các tham số tối ưu là một công việc không đơn giản vì chưa có một cơ sở khoa học nào để hướng dẫn việc này. Do vậy, đây có thể coi là một hướng mở của luận văn.

KẾT LUẬN

Trong luận văn "**Ứng dụng mạng nơron trong bài toán xác định lộ trình cho robot**" em đã hoàn thành những nhiệm vụ sau:

1. Đã hệ thống cơ sở của mạng nơron nhân tạo, đặc biệt là mạng Hopfield, nêu khái quát những ứng dụng của mạng nơron trong công nghệ robot.
2. Đã trình bày khái quát về một phương pháp thiết kế một mobile robot, bài toán lập lộ trình và các thành phần của nó.
3. Đã trình bày về cấu hình không gian và cấu hình chướng ngại vật, một trong những công việc khá phức tạp khi xác định lộ trình cho robot bằng phương pháp truyền thống
4. Đã nghiên cứu về phương pháp lập lộ trình dựa vào mạng nơron do Yang và Meng đề xuất và phương pháp cải tiến dựa trên mô hình nguyên bản của Yang và Meng.
5. Đã cài đặt thử nghiệm hai mô hình đã nghiên cứu trên máy tính, kết quả đạt được phản ánh chính xác những kết quả nghiên cứu.

Các định hướng nghiên cứu tiếp theo

Để xác định một lộ trình tốt nhất thì việc lựa chọn tham số phù hợp là vô cùng cần thiết, tuy nhiên việc đó khá khó khăn vì chưa có một cơ sở khoa học nào về lĩnh vực này. Vì vậy hướng nghiên cứu tiếp theo của luận văn là tìm ra phương pháp lựa chọn tham số sao cho lộ trình tìm được là tối ưu.

TÀI LIỆU THAM KHẢO

1. Đặng Quang Á, Một cách nhìn về việc sử dụng mạng Hopfield giải các bài toán thỏa mãn rang buộc và tối ưu có ràng buộc, Báo cáo tại Hội thảo quốc gia "Một số vấn đề chọn lọc của công nghệ thông tin", Hải phòng 6/2001.
2. Đặng Quang Á, Ứng dụng của mạng nơ ron trong tính toán, Sách "Hệ mờ, mạng nơ ron và ứng dụng", Chủ biên: Bùi công Cường, Nguyễn Doãn Phước, Nhà XBKH-KT, Hà nội, 2001, 199-211.
3. Nguyễn Đình Thúc, Lập trình tiến hoá, Nhà XB Giáo dục, 2001.
4. Bekey, G. A. & Goldberg, K.Y, Neural Networks in Robotics. luwer Academic Publishers, ISBN 0-7923-9268-X, Boston(1993).
5. Brady M. , Robot Motion: Planning and Control, The MIT Press, ISBN 0-262-02182-X, Cambridge(1982).
6. Janglová D. , Neural Networks in Mobile Robot Motion, International Journal of Advanced Robotic Systems,V. 1 No 1 (2004), 15-22.
7. Subhrajit Bhattacharya, Siddharth Talapatra, Robot Motion Planning Using Neural Networks: A Modified Theory, International Journal of Lateral Computing, Vol.2, No.1, 2005, 9-13.

PHỤ LỤC

Mã lệnh chương trình thử nghiệm.

```
Option Explicit
Dim nutlenh, di, dj, ci, cj As Integer
Dim x(1 To 12, 1 To 12) As Double
Dim E(1 To 12, 1 To 12) As Double
Dim Tt(1 To 12, 1 To 12) As Double
Dim i, j, k, a, b, p, duong  As Integer
Dim m, tong, delta As Double
Dim kt As Boolean
Dim hd(1 To 12, 1 To 12) As Integer
Private Sub Cmbkhoitao_Click()
Dim i, j As Integer
'ma tran kich thich phan ung sua
For i = 1 To 12
For j = 1 To 12
    x(i, j) = -0.5
    E(i, j) = 0
    Tt(i, j) = 0
    hd(i, j) = 0
Next
Next
'khoi tao nut lenh
nutlenh = 0
'Khoi tao luoi
For i = 1 To 12
For j = 1 To 12
  If i <> 11 Then
     T(Val(Str(i) & Str(j))).BackColor = &H80000005
     T(Val(Str(i) & Str(j))).Text = ""
     T(Val(Str(i) & Str(j))).Font.Size = 8
     T(Val(Str(i) & Str(j))).ForeColor = &H0&
     T(Val(Str(i) & Str(j))).Font.Bold = False
  Else
     If j > 9 Then
       T(Val(Str(i) & Str(j))).BackColor = &H80000005
       T(Val(Str(i) & Str(j))).Text = ""
       T(Val(Str(i) & Str(j))).Font.Size = 8
```

```
      T(Val(Str(i) & Str(j))).ForeColor = &H0&
      T(Val(Str(i) & Str(j))).Font.Bold = False
    Else
      T(Val(Str(i) & Str(0) & Str(j))).BackColor = &H80000005
      T(Val(Str(i) & Str(0) & Str(j))).Text = ""
      T(Val(Str(i) & Str(0) & Str(j))).Font.Size = 8
      T(Val(Str(i) & Str(0) & Str(j))).ForeColor = &H0&
      T(Val(Str(i) & Str(0) & Str(j))).Font.Bold = False
    End If
  End If
Next j
Next i
Timer1.Enabled = False
End Sub
Private Sub Cmbmecung_Click()
        nutlenh = 1
End Sub
Private Sub Cmbthoat_Click()
        Unload Me
End Sub
Private Sub Cmbthuchien_Click()
        i = di
        j = dj
        kt = False
        duong = 1
        Timer1.Enabled = True
 End Sub
Private Sub Cmbvtdau_Click()
        nutlenh = 2
End Sub
Private Sub Cmbvtdich_Click()
        nutlenh = 3
End Sub
Private Sub Form_Load()
        nutlenh = 0
        Timer1.Enabled = False
End Sub
Sub T_Click(Index As Integer)
'tao me cung
Dim i, j As Integer
```

```
Dim p, q, xhd As Integer
If nutlenh = 1 Then
   T(Index).BackColor = &H80000006
   For i = 1 To 12
   For j = 1 To 12
     If (i <> 11) Or (j > 9) Then
      If Val(Str(i) & Str(j)) = Index Then
         x(i, j) = -1
         E(i, j) = -100
         'Text1.Text = E(i, j)
         Tt(i, j) = 1
     End If
      Else
       If Val(Str(i) & "0" & Str(j)) = Index Then
         x(i, j) = -1
         E(i, j) = -100
         Tt(i, j) = 1
         'Text1.Text = E(i   , j)
       End If
     End If
   Next j
   Next i
End If
'xac dinh vi tri dau
If nutlenh = 2 Then
   T(Index).ForeColor = &HFF&
   T(Index).Font.Bold = True
   T(Index).Font.Size = 12
   T(Index).Text = " o"
   For i = 1 To 12
   For j = 1 To 12
   If (i <> 11) Or (j > 9) Then
      If (Val(Str(i) & Str(j)) = Index) Then
        di = i
        dj = j
        Tt(i, j) = 1
      End If
   Else
     If (Val(Str(i) & "0" & Str(j)) = Index) Then
      di = i
```

```
        dj = j
        Tt(i, j) = 1
      End If
    End If
    Next j
    Next i
End If
'xac dinh vi tri dich
If nutlenh = 3 Then
    T(Index).ForeColor = &HC00000
    T(Index).Font.Bold = True
    T(Index).Font.Size = 12
    T(Index).Text = " o"
    For i = 1 To 12
    For j = 1 To 12
    If (i <> 11) Or (j > 9) Then
      If (Val(Str(i) & Str(j)) = Index) Then
         E(i, j) = 100
         ci = i
         cj = j
      End If
    Else
      If (Val(Str(i) & "0" & Str(j)) = Index) Then
         E(i, j) = 100
         ci = i
         cj = j
      End If
    End If
    Next j
    Next i
    End If
End Sub
Public Function max(a As Double, b As Double) As Double
Dim m As Double
    If a > b Then
       m = a
    Else
       m = b
    End If
    max = m
```

```
End Function
Public Function tinh_x(a As Integer, b As Integer) As Integer
   Dim tong, delta As Double
   tong = 0
   If (a - 1 > 0) Then
     tong = tong + max(x(a - 1, b), 0)
   End If
   If (b - 1 > 0) Then
      tong = tong + max(x(a, b - 1), 0)
   End If
   If a + 1 <= 12 Then
      tong = tong + max(x(a + 1, b), 0)
   End If
   If b + 1 <= 12 Then
      tong = tong + max(x(a, b + 1), 0)
   End If
   delta = -10 * x(a, b) + (1 - x(a, b)) * (max(E(a, b), 0) + tong) - (-1 + x(a, b))
* max(-E(a, b), 0)
   x(a, b) = x(a, b) + delta
   tinh_x = 1
End Function
Private Sub Timer1_Timer()
If ((i <> ci) Or (j <> cj)) And (kt = False) Then
   If i <> 11 Or j > 9 Then
    T(Val(Str(i) & Str(j))).ForeColor = &H0&
    T(Val(Str(i) & Str(j))).Font.Size = 1
    T(Val(Str(i) & Str(j))).Font.Bold = True
    T(Val(Str(i) & Str(j))).Text = " o"
   Else
    T(Val(Str(i) & Str(0) & Str(j))).ForeColor = &H0&
    T(Val(Str(i) & Str(0) & Str(j))).Font.Size = 1
    T(Val(Str(i) & Str(0) & Str(j))).Font.Bold = True
    T(Val(Str(i) & Str(0) & Str(j))).Text = " o"
   End If
Else
   If i <> 11 Or j > 9 Then
    T(Val(Str(i) & Str(j))).ForeColor = &HFF&
    T(Val(Str(i) & Str(j))).Font.Size = 12
    T(Val(Str(i) & Str(j))).Font.Bold = True
    T(Val(Str(i) & Str(j))).Text = " o"
```

```
    Else
     T(Val(Str(i) & Str(0) & Str(j))).ForeColor = &HFF&
     T(Val(Str(i) & Str(0) & Str(j))).Font.Size = 12
     T(Val(Str(i) & Str(0) & Str(j))).Font.Bold = True
     T(Val(Str(i) & Str(0) & Str(j))).Text = " o"
    End If
End If
If ((i <> ci) Or (j <> cj)) And (kt = False) Then
   If (i - 1 > 0) Then
    If E(i - 1, j) <> -100 Then
     'k = tinh_x(i - 1, j)
     a = i - 1
     b = j
     tong = 0
     If (a - 1 > 0) Then
       tong = tong + max(x(a - 1, b), 0)
    End If
    If (b - 1 > 0) Then
       tong = tong + max(x(a, b - 1), 0)
    End If
    If a + 1 <= 12 Then
       tong = tong + max(x(a + 1, b), 0)
   End If
    If b + 1 <= 12 Then
       tong = tong + max(x(a, b + 1), 0)
    End If
    delta = -10 * x(a, b) + (1 - x(a, b)) * (max(E(a, b), 0) + tong) - (-1 + x(a,b))
* max(E(a, b), 0)
    x(a, b) = x(a, b) + 0.1 * delta
    '******
    End If
   End If
   If (j - 1 > 0) Then
    If E(i, j - 1) <> -100 Then
      'k = tinh_x(i, j - 1)
      a = i
      b = j - 1
      tong = 0
    If (a - 1 > 0) Then
      tong = tong + max(x(a - 1, b), 0)
```

```
  End If
   If (b - 1 > 0) Then
      tong = tong + max(x(a, b - 1), 0)
   End If
   If a + 1 <= 12 Then
      tong = tong + max(x(a + 1, b), 0)
   End If
   If b + 1 <= 12 Then
      tong = tong + max(x(a, b + 1), 0)
  End If
   delta = -10 * x(a, b) + (1 - x(a, b)) * (max(E(a, b), 0) + tong) - (-1 + x(a,b))
* max(E(a, b), 0)
   x(a, b) = x(a, b) + 0.1 * delta
   '******
    End If
  End If
  If (j + 1 <= 12) Then
   If E(i, j + 1) <> -100 Then
     'k = tinh_x(i, j + 1)
     a = i
     b = j + 1
     tong = 0
   If (a - 1 > 0) Then
     tong = tong + max(x(a - 1, b), 0)
   End If
   If (b - 1 > 0) Then
      tong = tong + max(x(a, b - 1), 0)
   End If
   If a + 1 <= 12 Then
      tong = tong + max(x(a + 1, b), 0)
   End If
   If b + 1 <= 12 Then
      tong = tong + max(x(a, b + 1), 0)

   End If
   delta = -10 * x(a, b) + (1 - x(a, b)) * (max(E(a, b), 0) + tong) - (-1 + x(a,b))
* max(-E(a, b), 0)
   x(a, b) = x(a, b) + 0.1 * delta
   '********
    End If
```

```
  End If
  If (i + 1 <= 12) Then
   If E(i + 1, j) <> -100 Then
     'k = tinh_x(i + 1, j)
     a = i + 1
     b = j
     tong = 0
   If (a - 1 > 0) Then
     tong = tong + max(x(a - 1, b), 0)
   End If
   If (b - 1 > 0) Then
      tong = tong + max(x(a, b - 1), 0)
   End If
   If a + 1 <= 12 Then
      tong = tong + max(x(a + 1, b), 0)
   End If
   If b + 1 <= 12 Then
      tong = tong + max(x(a, b + 1), 0)
   End If
   delta = -10 * x(a, b) + (1 - x(a, b)) * (max(E(a, b), 0) + tong) - (-1 + x(a,b))
* max(-E(a, b), 0)
   x(a, b) = x(a, b) + 0.1 * delta
   '*****
    End If
  End If
  'xac dinh gia tri lon nhat
  m = -1000000
  p = 0
  If i - 1 > 0 Then
   If (m < x(i - 1, j)) And E(i - 1, j) <> -100 And Tt(i - 1, j) = 0 Then
      m = x(i - 1, j)
   End If
  End If
  If (j - 1 > 0) Then
   If (m < x(i, j - 1)) And E(i, j - 1) <> -100 And Tt(i, j - 1) = 0 Then
      m = x(i, j - 1)
   End If
  End If
  If (i + 1 <= 12) Then
      If (m < x(i + 1, j)) And E(i + 1, j) <> -100 And Tt(i + 1, j) = 0 Then
```

```
            m = x(i + 1, j)
        End If
    End If
    If (j + 1 <= 12) Then
      If (m < x(i, j + 1)) And E(i, j + 1) <> -100 And Tt(i, j + 1) = 0 Then
        m = x(i, j + 1)
     End If
    End If
    'dinh vi vi tri tiep theo
    If i - 1 > 0 Then
      If (m = x(i - 1, j)) And (p = 0) And (E(i - 1, j) <> -100) And Tt(i - 1, j) = 0
Then
        i = i - 1
        p = 1
      End If
    End If
    If j - 1 > 0 Then
      If (m = x(i, j - 1)) And (p = 0) And (E(i, j - 1) <> -100) And Tt(i, j - 1) = 0
Then
        j = j - 1
        p = 1
      End If
     End If
    If i + 1 <= 12 Then
      If (m = x(i + 1, j)) And (p = 0) And (E(i + 1, j) <> -100) And Tt(i + 1,j)=0
Then
        i = i + 1
        p = 1
      End If
    End If
    If j + 1 <= 12 Then
      If (m = x(i, j + 1)) And (p = 0) And (E(i, j + 1) <> -100) And Tt(i, j +1)=0
Then
        j = j + 1
        p = 1
      End If
    End If
If m <> -1000000 Then
If i <> 11 Or j > 9 Then
    T(Val(Str(i) & Str(j))).ForeColor = &HFF&
```

```
    T(Val(Str(i) & Str(j))).Font.Size = 12
    T(Val(Str(i) & Str(j))).Font.Bold = True
    T(Val(Str(i) & Str(j))).Text = " o"

  Else
    T(Val(Str(i) & Str(0) & Str(j))).ForeColor = &HFF&
    T(Val(Str(i) & Str(0) & Str(j))).Font.Size = 12
    T(Val(Str(i) & Str(0) & Str(j))).Font.Bold = True
    T(Val(Str(i) & Str(0) & Str(j))).Text = " o"
  End If
    Tt(i, j) = Tt(i, j) + 1
   duong = duong + 1
Else
   kt = True
End If
End If
End Sub
```