8-31-1991

Mobile robot path planning based on hierarchical hexagonal decomposition

Dan Zheng
New Jersey Institute of Technology

Copyright Warning & Restrictions

The copyright law of the United States (Title 17, United States Code) governs the making of photocopies or other reproductions of copyrighted material.

Under certain conditions specified in the law, libraries and archives are authorized to furnish a photocopy or other reproduction. One of these specified conditions is that the photocopy or reproduction is not to be "used for any purpose other than private study, scholarship, or research." If a, user makes a request for, or later uses, a photocopy or reproduction for purposes in excess of "fair use" that user may be liable for copyright infringement,

This institution reserves the right to refuse to accept a copying order if, in its judgment, fulfillment of the order would involve violation of copyright law.

Please Note: The author retains the copyright while the New Jersey Institute of Technology reserves the right to distribute this thesis or dissertation

Printing note: If you do not wish to print this page, then select "Pages from: first page # to: last page #" on the print dialog screen

Mobile Robot Path Planning Based on Hierarchical Hexagonal Decomposition

By

Dan Zheng

Thesis submitted to the faculty of the graduate school of the New Jersey Institute of Technology in partial fulfillment of the requirements for the degree of Master of Science in Electrical Engineering

1991

APPROVAL SHEET

Title of Thesis: **Mobile Robot Path Planning Based on Hierarchical Hexagonal Decomposition**

Candidate: Dan Zheng
Master of Science
in Electrical Engineering, 1991

Abstract and Thesis Approval :

Dr. Edwin S. H. Hou, Advisor — Date
Assistant Professor of
Electrical and Computer Engineering

Dr. Yun-Qing Shi — Date
Assistant Professor of
Electrical and Computer Engineering

Dr. Anthony D. Robbi — Date
Associate Professor of
Electrical and Computer Engineering

New Jersey Institute of Technology
Newark, New Jersey

VITA

Name : Dan Zheng

Present address :

Date of Birth :

Place of Birth :

Education :

1. New Jersey Institute of Technology
   Electrical Engineering Department, M.S.E.E.
   September 1989 — August 1991, Newark, NJ

2. Zhejiang University
   Radio-Electronic Engineering Department, M. Engineering
   September 1983 — July 1986, P.R.C.

3. Xi'an Communications University
   Electronic Engineering Department, B. Engineering
   October 1978 — July 1982, P.R.C.

ACKNOWLEDGEMENTS

The author would like to express appreciation to Dr. Edwin S. H. Hou, his major professor, for his inspirations, guidances and support during the entire course of this thesis. This research is partly supported by a grant from New Jersey Department of Higher Education through NJIT Separately Budgeted Research. Appreciation is also extended to the committee members, Dr. Anthony D. Robbi, and Dr. Yun-Qing Shi for their serving on the examination committee and for evaluating this research. Finally, special thanks go to his wife, Honglei Huang, for her help and encouragement.

ABSTRACT

Title of Thesis :

Mobile Robot Path Planning Based on Hierarchical Hexagonal Decomposition

Name :

Dan Zheng

Master of Science in Electrical Engineering

Thesis directed by :

Dr. Edwin S. H. Hou

In this thesis, a new algorithm based on hierarchical cell decomposition and artificial potential field is developed to solve the find-path problem for a mobile robot. The basic cell structure used for decomposition is a hexagon. The artificial potential field is based on an attractive force from the goal position and repelling forces from the obstacles. Computer simulations of the algorithm for various obstacle scenarios are also presented.

CONTENTS

FIGURES

Chapter 1
Introduction

The autonomous mobile robot has long been a dream of human beings. This dream is gradually becoming a reality due to the tremendous progress in robotics research. A major obstacle in achieving the goal of autonomous mobile robot is the path planning or find-path problem. The problem can be stated as, given an ***object*** with an initial configuration (location and orientation), a goal configuration, and a set of ***obstacles*** distributed in space, find a continuous path for the object from the initial configuration to the goal configuration without colliding with any obstacles along the way.

Although the problem of collision-free path planning is a much researched topic, finding an acceptable solution to it is not easy. This is because the problem is inherently computationally intractable. In fact, it has been shown that the generalized path planning problem known as the "Piano mover's problem" is *P-space hard.* This means that it is at least as hard as a number of familiar problems for which the only known solutions have computational complexity that grows exponentially with the task size [1,2].

Many algorithms for the path planning problem have been proposed. Tomás Lozano-Pérez [3] (1983), Whitesides [4] (1985), and more recently, Sharir [5] (1989) provide an excellent overview of the problem. These approaches can be divided into two categories: graph searching techniques and potential field methods. The graph searching techniques are so named because a chart or a graph is used to show free spaces where no collision will occur and forbidden spaces where a collision will

occur. Based on this graph, a path is then selected by piecing together the free spaces or by tracing around the forbidden spaces. The potential field methods use artificial potential fields applied to the obstacles and goal positions and use the resulting field to influence the path of the robot.

The representative algorithm of the graph searching techniques is the so called Configuration Space (C space) method developed by Lozano-Pérez [6],[7]. The configuration of a rigid object is a set of independent parameters that characterize the position and orientation of every point in the object. Usually, a local coordinate frame is associated with a rigid object (planar polygon). Then, the configuration of the polygon can be specified by the x, y position of the origin of the local coordinate frame, known as the *reference point* and a θ value indicating the rotation of the local frame relative to the global frame. The space of all possible configurations of an object is then its *configuration space.* A point in the configuration space, a *configuration point*, represents a particular position of the object's reference point and an orientation of the object.

The configuration space for a planar polygon is three-dimensional while that of a solid polyhedral is six-dimensional (three translational and three rotational dimensions). Due to the presence of the immovable obstacles some regions of the configuration space are not reachable; these regions are called the *configuration obstacles.* Hence, in the configuration space, the moving object is typically shrunk to a configuration point while the immovable obstacles are expanded to fill all space where the presence of the configuration point would imply a collision with obstacles. The find-path problem is to determine a path in the free space for the configuration point. There are several difficulties in implementing this type of algorithm. They

are:

- Configuration space has usually six dimensions, whereas only three-dimensional or perhaps four-dimensional spaces seem to be tractable within acceptable time limits [8–11]. In practice, for a six degree of freedom arm, building and searching the six dimensional C space, although possible, is extremely cumbersome and slow.

- The graph describing the free space tends to have a very large number of nodes even at a minimal reasonable resolution.

As an alternative to graph searching, the hierarchical cell decomposition technique was introduced by Brooks and Lozano-Pérez [12], with subsequent contributions by other authors [13-16]. It consists of searching and decomposing the C space of the mobile robot into cells at successive levels of approximations. A cell is classified as EMPTY if it does not intersect with any obstacles and FULL if it is completely inside an obstacle. If it is neither EMPTY or FULL, it is classified as MIXED. At each level of decomposition, the algorithm searches for a sequence of adjoining EMPTY cells that connect the initial configuration of the mobile robot to the desired goal configuration. If a solution cannot be found, the algorithm will decompose some MIXED cells into smaller cells, and repeat the search process. The search terminates when a solution has been found, or it is guaranteed that no solution exists, or MIXED cells cannot be decomposed further. The hierarchical cell decomposition approach is relatively easy to implement and quite efficient even when the mobile robot can both translate and rotate [17].

Due to the difficulties with the configuration space approach, another class

of methods based on potential fields have been proposed. The idea of using "potential functions" for the specification of robot was pioneered by Khabit [18] for obstacle avoidance, and further advanced by fundamental work of Hogan [19] for force control, Warren [20] for global path planning, and Koditschek [21] for robot navigation. The methodology was developed independently by Arimoto in Japan [22], and by Soviet investigators as well [23]. The idea is to design an artificial potential field that would attract the robot to its goal configuration while repelling it when it is near obstacles. Although not as thorough as the graph searching techniques, the speed of the algorithms and the easy extension to higher dimensions make them an excellent alternative to the graph searching techniques. Unfortunately, it is acknowledged that there are obstacle arrangements that will result in potential fields with many spurious local minima which would trap the robot.

In this thesis, a new method for path planning is developed by combining the artificial potential field approach and the hierarchical cell decomposition techniques. We will introduce a new structure for the cell used in decomposition — ***hexagon***. A new technique of labelling a cell as PASSABLE or IMPASSABLE is also proposed. A cell which is marked IMPASSABLE will not be selected as a path point, otherwise, it can be selected as a path point.

The rest of the thesis is arranged as follows: the problem formulation and assumptions are described in Chapter 2; the path planning method will be presented in Chapter 3; and simulation results will be illustrated in Chapter 4. Chapter 5 concludes this thesis.

Chapter 2
Problem Formulation

The problem formulation and assumptions will be described in this Chapter. We will also present the model of the mobile robot and the free space the robot travels in.

In many manufacturing environments, automatically guided vehicles or mobile robots are used to transport parts or raw materials between workstations. The movement of the mobile robot is constrained to the two-dimensional plane with two degrees of translational freedom and one degree of rotational freedom. Although the mobile robot and the obstacles are three dimensional, we can project them to the ground and consider them as two-dimensional objects. Therefore, we consider our path planning to be a two-dimensional problem. We assume that if the two-dimensional projection of the mobile robot from its original three-dimensional entity will not collide with any obstacles projected in the same way, then the three-dimensional robot will not collide with any obstacles in three dimensions. Furthermore, without losing generality, it is assumed that all objects are represented by polygons (can be convex or concave).

We also assume that there are fixed obstacles and workstations distributed over the free space the mobile robot is working in. The task of the moving robot is to visit each of the workstations in a predefined order, without colliding with any of the obstacles. Figure 2.1 shows the robot, obstacles, and workstations as well as the task of the mobile robot.

The mobile robot considered here can have both translation and rotation

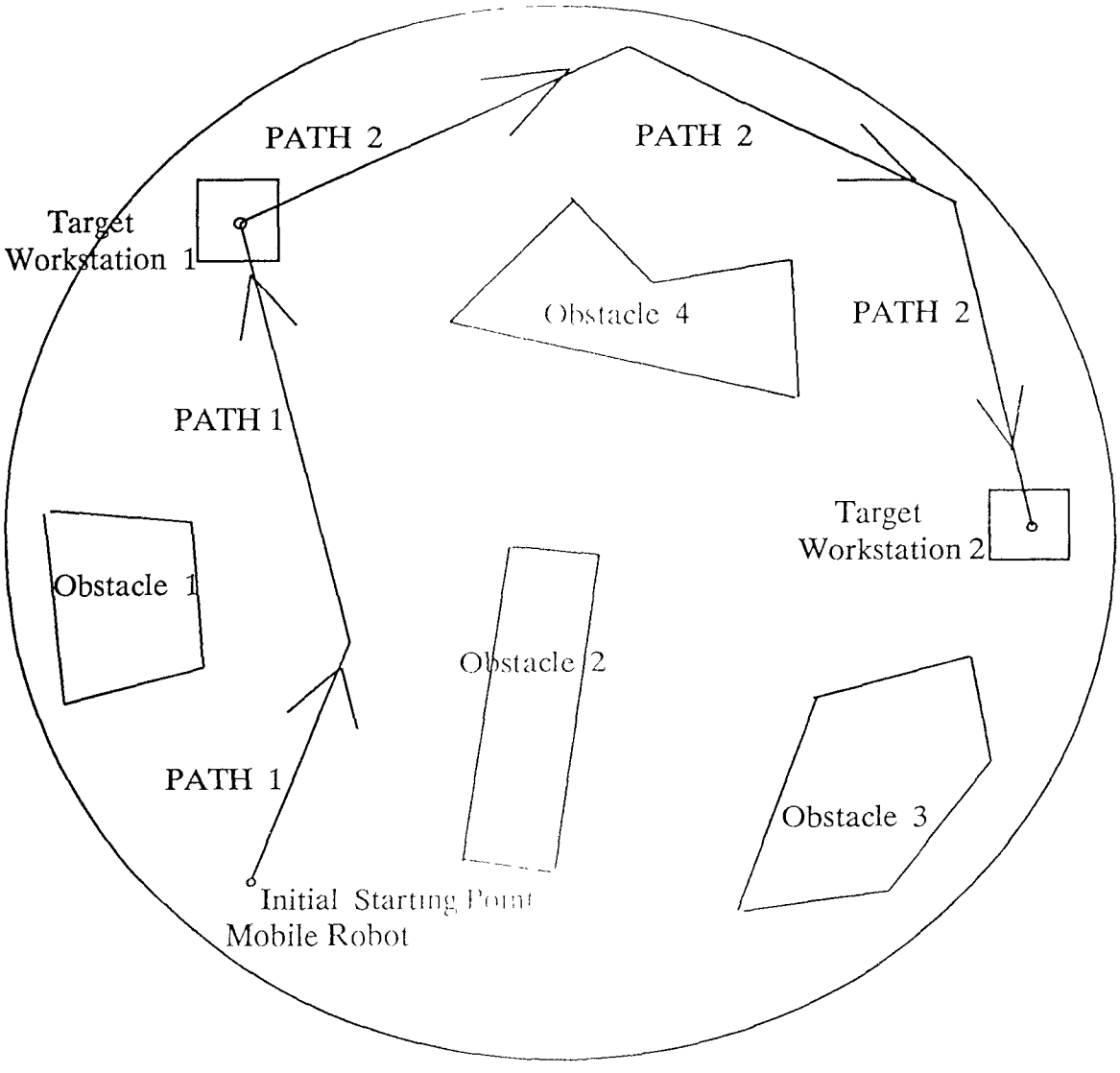


Figure 2.1 Representation of the Problem.

movements. The model of the robot is illustrated in Figure 2.2.

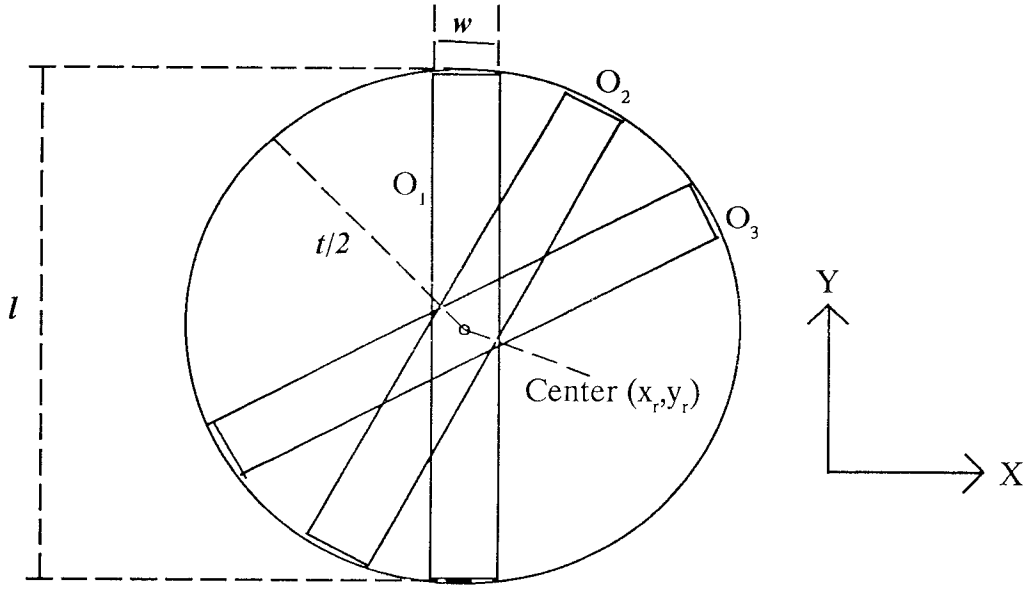


Figure 2.2 Robot Model.

In Figure 2.2, the mobile robot is represented by a rectangle with length l, width w ($l > w$) and center at (X, Y). It can translate along Y axis and rotate about its center. We assume that the rectangle is bounded by a circumscribe circle and the diameter of this circle will be defined as the minimal width, t, for a path. The path planning problem can then be considered as finding a collision-free path with width t for a point (center of the mobile robot) among the obstacles. This model will suffice since if a point can pass through a path with width t, then the mobile robot also can pass through it without any collisions with the obstacles.

A hierarchical decomposition method using hexagonal cells will be used to find the collision-free path. An artificial potential field is used as a criterion function for searching the hexagons created as the result of the decomposition. The potential field approach works as follows. Each obstacle exerts a repulsive force on the moving object while the goal exerts an attractive force. With appropriate choice of the attractive force and repulsive forces, the moving object will then be drawn toward the goal position while simultaneously being forced away from obstacles. It can be shown that if the potential within the obstacles is set to infinity, collisions will never occur. A hierarchical neighborhood searching algorithm will be used to find a collision-free path.

Chapter 3
Path Planning Method

In this chapter, we will describe in detail our method of planning a collision-free path which is based on hierarchical cell decomposition and artificial potential field. We will also discuss the attractive force and repulsive forces used to define the artificial potential field and also the procedure for decomposing a hexagon.

3.1 Potential Field and Region of Influence

There are various ways to define the artificial potential forces. One of the approaches is to use the Newtonian inverse-square law to define the artificial force acting on a moving object by an obstacle. The potential acting on the moving object is then the maximum potential among all the obstacles plus a component decreasing linearly toward the goal configuration [17]. We will adopt this approach with the following modifications:

1) Each obstacle (workstations may also be obstacles) is a source of a repelling force. The magnitude of the repelling force is proportional to $1/r^2$, where r is the distance between the center of the mobile robot and the center of the obstacle.

2) The target workstation is the source of an attractive force, whose magnitude is proportional to $(1 - k\alpha)$, where α is the angle between the line connecting the center of the workstation to the mobile robot and the direction the mobile robot will move to, and k is a scaling factor. The maximum attractive force is thus the straight line path connecting the center of the workstation and the mobile robot. With α, we have taken the rotation in addition to the translation movement of

the mobile robot into our consideration

3) The potential within any obstacle will be defined as infinity.

4) For simplicity sake, we assume that all sources of forces (either attractive or repulsive) are point sources.

5) Therefore, the potential at any point will be the sum of the attractive force due to the target workstation and the repulsive force from each obstacle. It can be written as:

$$Potential(P) = \sum_{i=1}^{n} k_1 \frac{1}{r_i^2} - C(1 - k_2\alpha) \qquad (1)$$

where P is the point under consideration, r_i is the distance between the center of the ith obstacle and the point P, k_1, k_2, and C are constants, and α in radian is determined as shown in Figure 3.1.

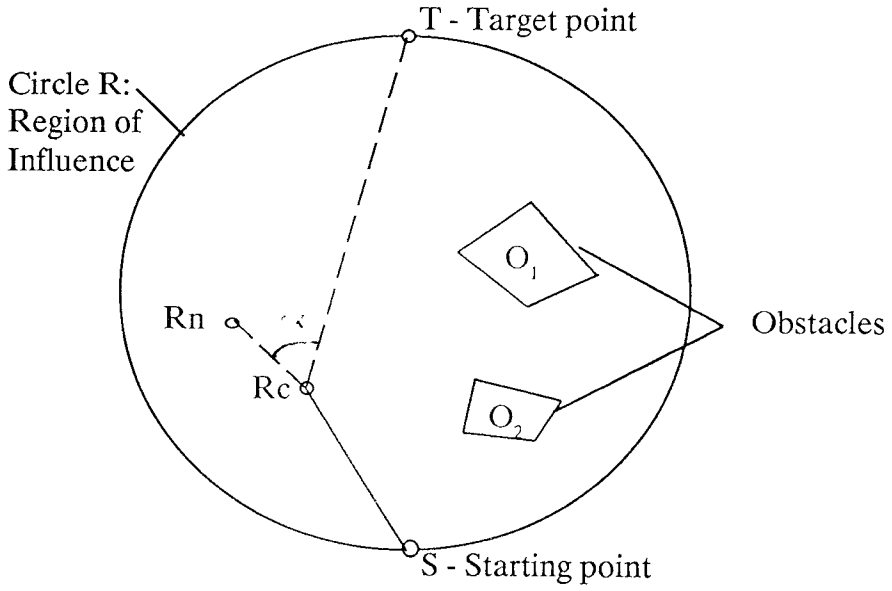


Figure 3.1 Region of Influence and Determination of α.

In the figure, S is the starting point of the mobile robot, T is the center of the immediate target workstation, circle R is the region of influence, R_C is the current position of the robot, and R_N is the possible candidate of the next path point the robot will go to.

According to Eq. (1), the potential of a point close to an obstacle is large and the attractive force provides a general direction the robot should move to. Therefore, by choosing a path with minimal potential will result in a collision-free path for the robot. Eq. (1) will be used as the criteria function when we perform the neighborhood search.

In order to reduce the amount of computation, we introduce a region of influence for the mobile robot (see Figure 3.1). Only obstacles and workstations falling within this region will be considered when computing the potential. All obstacles and workstations outside this region will not be considered. All the workstations but the immediate target workstation within this region will be considered as obstacles. The region of influence is defined as a circle whose diameter is the distance between the starting position of the moving robot and the center of the immediate target workstation.

3.2 Hierarchical Decomposition of Hexagons

The neighborhood searching algorithm used to find the collision-free path is based on a hierarchical decomposition approach. The basic cell structure for the decomposition is the hexagon. Some advantages of using hexagons are

1) A hexagon may be recursively divided into smaller hexagons and half hexagons.

2) A hexagon has six edges or six neighbors, (see Figure 3.2). When a robot currently stays at the center of a hexagon and is going to go towards the goal, it has at most six choices. This will simplify the procedure of searching.

3) The distances from the center of a hexagon to the center of its neighbors are the same.

Although hexagons cannot be completely divided into smaller full cells, they can be decomposed into a combination of smaller full and half hexagons. Figure 3.3(a) illustrates the decomposition of a full hexagon. We see that a full hexagon is divided into 13 smaller full hexagons and 6 smaller half hexagons. Notice that the half hexagons all appear on the edges of the parent hexagon only. They can be considered as full hexagons except they have been cut in half by the parent hexagon. Figure 3.3(c) illustrates the decomposition of a half hexagon. It is decomposed into 5 smaller full hexagons and 6 smaller half hexagons. The algorithm for performing the above decomposition consists of the following two steps:

1) The coordinates of the center point of the hexagons at the sub-level are computed.

2) Each smaller hexagon at the sub-level is reconstructed using the geometric properties of a hexagon.

From Figure 3.3(b), we can obtain the following properties of a hexagon:

1) Any hexagon may be characterized by the coordinates (H_x, H_y) of its center and its edge length l. Any other geometric feature values of the hexagon can be derived from these parameters. For example, the height (defined as the distance from the center point to one edge) is equal to $l\sqrt{3}/2$, and the coordinate (V_{1x}, V_{1y})

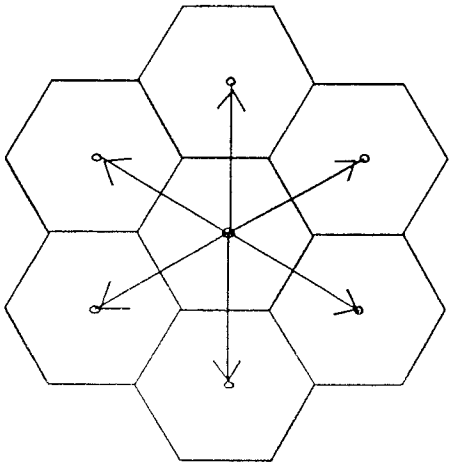
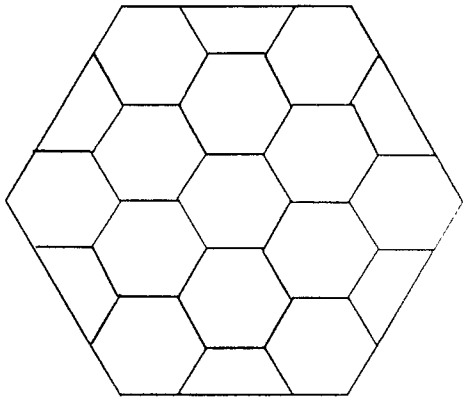
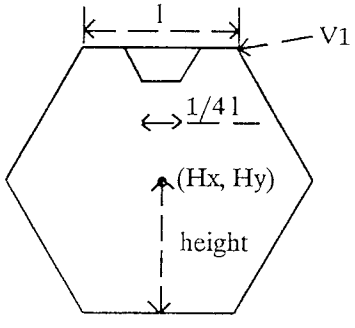


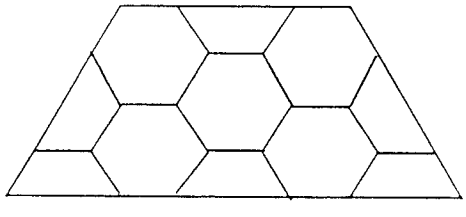
Figure 3.2 The Neighbors of a Hexagon.



(a) Decomposition of a Full Hexagon.



(b) Geometry of a Hexagon.



(c) Decomposition of a Half Hexagon.

Figure 3.3 Decomposition of a Hexagon.

of the vertice V_1 can be determined by

$$V_{1x} = H_x + 0\ 5l, \qquad V_{1y} = H_y + \frac{\sqrt{3}}{2}l;$$

Similarly, we can determine all the coordinates of the vertices in a hexagon.

2) The edge length of a hexagon in the sub-level is 1/4 of the edge length of its parent hexagon.

With these useful features, the decomposition algorithm can be written as follows:

Decomposition Algorithm

Given coordinates of the center point, and edge length of a hexagon

1) Recover the coordinates of the parent hexagon.
2) Take the center point of the parent hexagon as the center of the first sub-level hexagon.
3) Take the midpoint of each consecutive pair of vertices from the parent hexagon as the center of the next 6 sub-level hexagons. These are half hexagons.
4) Take the midpoint of each consecutive pair of centers obtained in 3 as the center of another 6 sub-level hexagons.
5) Take the midpoint of each pair of points consisting of one point obtained in 3 and the center of the parent hexagon as the center of the last 6 sub-level hexagons.
6) If any of the center of the sub-level hexagons is outside the border of the parent hexagon, delete them from the list of sub-level hexagons. This step is used for the decomposition of half hexagons.

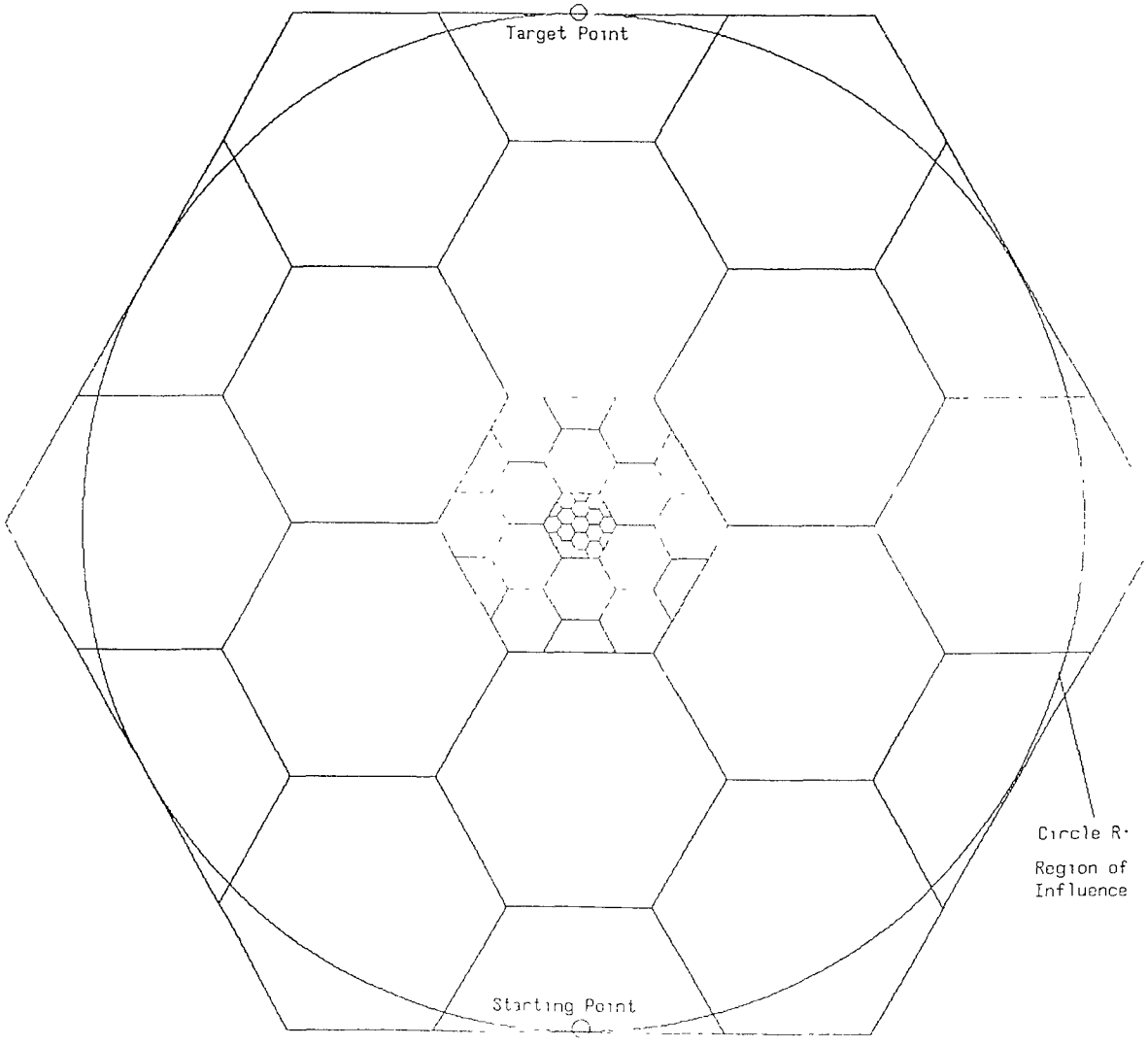


Figure 3.4 Three Level Decomposition of a Hexagon

7) The edge length of the sub-level hexagons is one quarter of the edge length of the parent hexagon.

An example of a 3-level decomposition of a hexagon using the above algorithm is shown in Figure 3.4.

3.3 Neighborhood Searching Algorithm

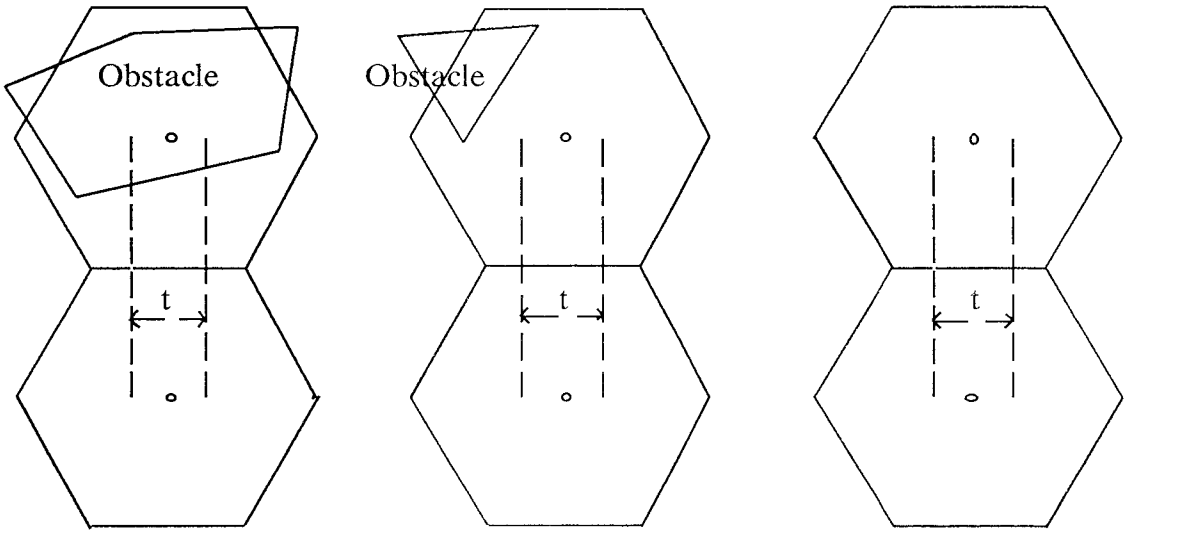
In this section, we will describe in detail our collision-free path planning algorithm. Our method first forms a hexagon containing the starting and goal point. A neighborhood searching algorithm guided by an artificial potential function is used to search for the collision-free path. The hexagons are decomposed hierarchically whenever necessary.

We assume that the work space has already been decomposed to a specific sub-level of hexagons depending on the sizes of the obstacles, and that the mobile robot will travel from the center of a hexagon to the center of another hexagon.

3.3.1 Attribute of a Hexagon

Each hexagon is associated with two attributes, PASSABLE or IMPASSABLE, and EMPTY or MIXED. The second attribute is assigned only to hexagons that are PASSABLE. Such a classification will speed up the neighborhood searching algorithm very effectively. The concept of PASSABLE and IMPASSABLE hexagons is illustrated in Figure 3.5. A hexagon is PASSABLE from its neighbor, if there is a collision-free path of width t from the center of the lower hexagon to the center of the upper hexagon in the figure, otherwise, it is IMPASSABLE. The potentials for the IMPASSABLE hexagons are assigned the value of infinity to ensure that

they will not be selected. For the PASSABLE hexagons, if any parts of an obstacle fall into the inscribed circle of the hexagon (see Figure 3.5(b)), the hexagon is defined as MIXED, otherwise it is EMPTY. According to the above definitions, the hexagon in Figure 3.5(a) is IMPASSABLE, the one in Figure 3.5(b) is PASSABLE but MIXED, and the one in Figure 3.5(c) is PASSABLE and EMPTY.



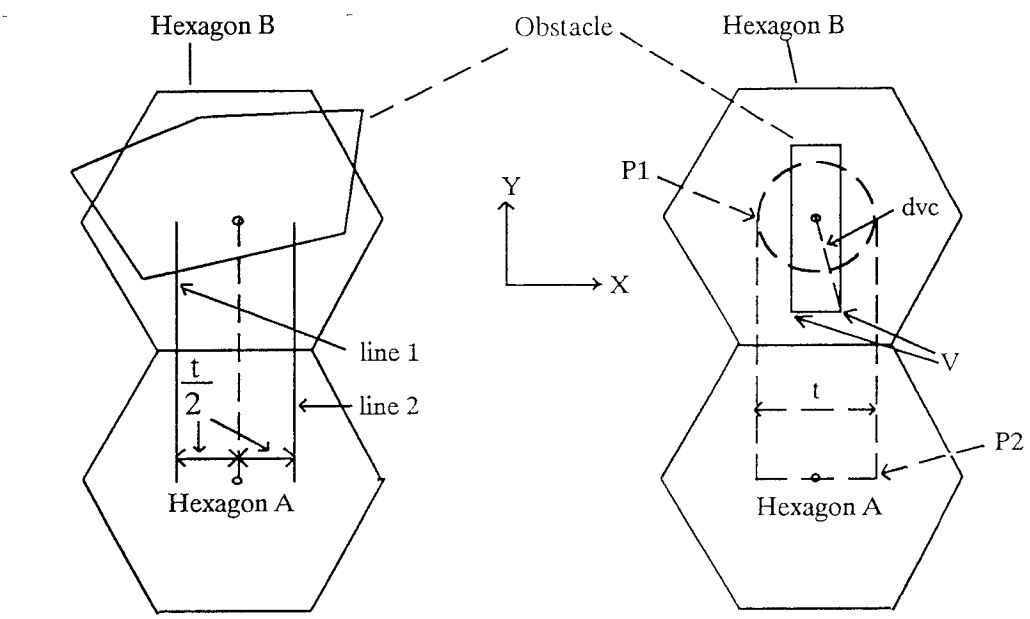
(a) IMPASSABLE. (b) PASSABLE & MIXED. (c) PASSABLE & EMPTY.

Figure 3.5 Attribute of a Hexagon.

To detect if a hexagon is PASSABLE or IMPASSABLE, we use the following method. Suppose the mobile robot is going to move from hexagon A to hexagon B. We first form two equal length and parallel line segments (see Figure 3.6(a)) which are at equidistance $(t/2)$ from the line connecting the centers of hexagons A and B. If at least one of the line segments intersects with an obstacle, hexagon B is labelled as IMPASSABLE; otherwise a second test is carried out to see if any vertices of the

obstacles fall into the region bounded by the dashed lines (see Figure 3.6(b)) If there is such a vertex, the corresponding obstacle with this vertex must be partly or fully inside the above region, therefore hexagon B is also IMPASSABLE; otherwise it is PASSABLE. The second test is implemented by comparing the coordinates of the vertex V under test with the coordinates of the points P_1 and P_2, and by comparing the distance d_{vc} from the vertex to the center of hexagon B with $t/2$. When either of the following conditions is true, hexagon B is IMPASSABLE:

$$d_{vc} \leq t/2 \qquad or \qquad P_{1x} \leq V_x \leq P_{2x} \quad \& \quad P_{2y} \leq V_y \leq P_{1y}$$



(a) Intersection. (b) Coordinate and Distance Comparisons.

Figure 3.6 Determination of the IMPASSABLE Attribute.

3.3.2 Selecting a Neighbor

As we perform the search, the mobile robot will be at the center of a hexagon. From there it will move a step toward the goal, or the target workstation. A hexagon has six neighbors, so the robot has at most six different neighbors to move to. Usually, the number of neighbors is less than six due to the presence of obstacles and the bounds of the region of influence. For each neighboring hexagon, a test is made to see if it is PASSABLE. If it is, then the potential at the center of the candidate hexagon is computed according to Eq. (1). The hexagon with the minimum potential will be selected as the next path point and the robot will move to the center of that hexagon.

3.3.3 Searching with Hierarchical Decomposition

The searching algorithm guided by the potential function will attempt to search for a sequence of adjacent EMPTY cells connecting the initial configuration of the robot to the goal configuration. If no such sequence is found, it decomposes some mixed cells into smaller cells, labels them appropriately, and searches again for a sequence of EMPTY cells. The process ends when (1) a solution has been found, (2) it is guaranteed that no solution can be found, or (3) MIXED cells cannot be decomposed further.

The searching algorithm proceeds by examining the neighboring hexagons at its current position. It computes the potential function of the neighboring hexagons that are PASSABLE. It then selects the hexagon with the smallest potential value and takes that hexagon as the next point on the path. If there are obstacles in the selected hexagon, i.e. MIXED, then the hexagon is decomposed

and the search step size is decreased. If there are no obstacles, i.e. EMPTY, then the search step size will remain unchanged or becomes larger. The latter means that when the EMPTY hexagon lies on the edge of a larger hexagon at the previous level of decomposition, and the sub-level search has been finished, the search step size reverts to the one for the larger hexagon. This search process continues until a solution is found, or a path cannot be found.

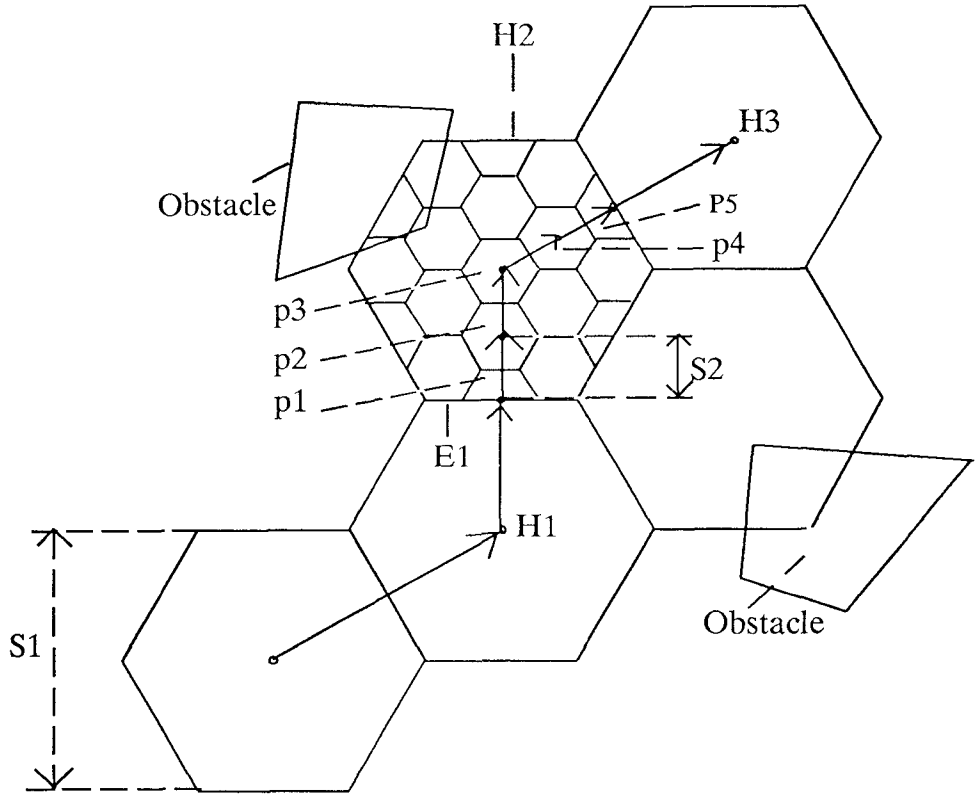


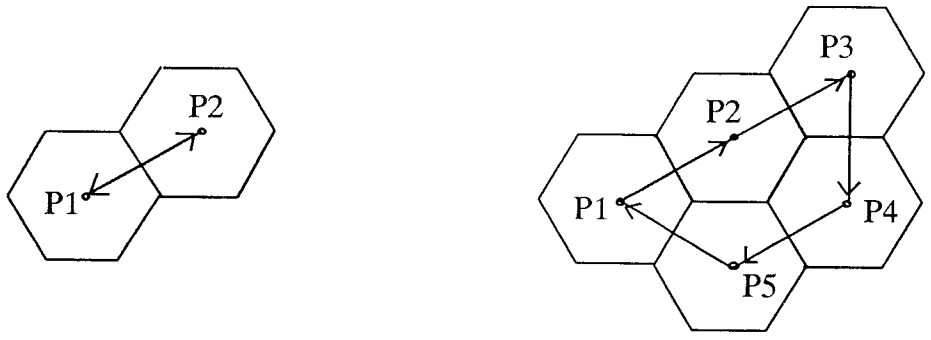
Figure 3.7 The Searching Process.

Figure 3.7 illustrates this searching process. Suppose the current position

of the robot is at the center of H_1, and it selects H_2 as the next path point. As H_2 is a MIXED hexagon, it is decomposed into the next sub-level. Since the robot is supposed to enter H_2 from H_1, it must pass through the edge E_1. Along this edge, there are only three possible sub-hexagons and it will choose one of them according to their potentials. After that, the search step will be changed from S_1 to S_2 which is 1/4 of S_1. Then the robot will resume the search with the step size S_2 until it encounters P_5, where it goes out of hexagon H_2 and enters H_3 which is an empty neighbor hexagon of H_2. Then from P_5, it will directly goes to the center of H_3. Since the robot is now at the center of a larger hexagon, the search step size will be reverted to S_1.

3.3.4 Local Minima

Like any other methods using artificial potential field, our path planner using the algorithms described above might be trapped into local minima. We can eliminate local minima very effectively using the following techniques.



(a) Single Point Cycle.

(b) Multi-Point Cycle.

Figure 3.8 Local Minima.

Consider the situation shown in Figure 3.8(a). It is possible to place obstacles such that the robot will moved to P_2 when at P_1 and move to P_1 when at P_2. This will put the robot into an infinite cycle. To avoid this situation, our algorithm will remember the last point where it came from and its potential is set to infinity in the next round of selection. Therefore, this point cannot be selected again in the next round.

Another class of trap is shown in Figure 3.8(b). In this case, the robot is trapped into an endless loop consisting of several path points. Such a local minima can be eliminated by marking P_1 as an IMPASSABLE hexagon, that is, set the potential of this hexagon to the infinity. This can be done after this point has been visited twice.

3.4 Collision-free Path Planning Algorithm

The complete algorithm for finding the collision-free path can be described as

Find Collision-free Path Algorithm

1) Obtain the obstacle information, such as the number of obstacles, vertices and center of each obstacle; obtain initial configuration of the mobile robot and the goal configuration.
2) Determine the region of influence according to the starting point and the target point.
3) Construct the initial hexagon which is the smallest hexagon bounding the region of influence.
4) Decompose the initial hexagon to a specific sub-level, depending on the average

length of the edges of all the obstacles, set the edge length of the smallest hexagon as the search step size. Then repeat steps 5 to 7 until the robot reaches the target workstation.

5) From the current position, compute the potential function of each neighboring hexagon which is PASSABLE and not a cycle point, and select the hexagon with the smallest potential energy as the next path point. Then determine if this point has been passed or selected twice. If it is, mark the point as a cycle point, and add it to the list of the cycle points to ensure that it will not be selected again.

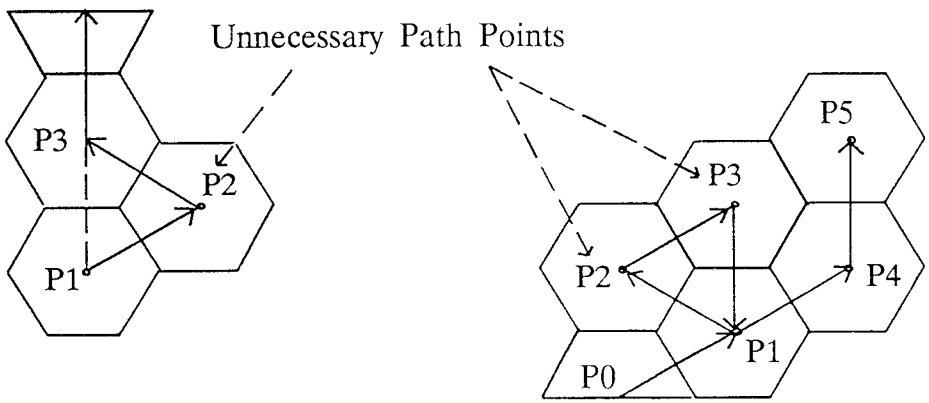
6) If the selected hexagon is MIXED, go to step 7; otherwise the selected hexagon is EMPTY and the robot is still in the sub-level search, go to step 5; otherwise if the selected hexagon is EMPTY and by passing through which, the robot goes out of the sub-level, then from the center of the selected hexagon, directly go to the center of its parent hexagon the selected hexagon belongs to, and reset the search step size to the edge length of the larger hexagon. Go to step 5.

7) Decompose the selected hexagon to the next sub-level, prepare the sub-level searching by choosing one of the hexagons on the edge through which the robot is supposed to enter, and directly go to the center of that hexagon, then decrease the search step size by 4, go to step 5.

3.5 Post Processing

After the initial path has been found, a post processing step is carried to test for redundancy and remove unnecessary path points. Redundancy occurs in two cases: a single unnecessary path point; or a few path points which formed a loop. The former occurs when three path points P_1, P_2, and P_3 are non co-linear, the

distance from P_1 to P_2 equal to that from P_1 to P_3, and the robot may go directly from P_1 to P_3 without touching any obstacles. Under these conditions, P_2 is an unnecessary path point, and may be deleted (see Figure 3.9(a)). The second type of redundancy occurs when a local minima is found. In that case, the path points in the loop are unnecessary because the robot started from a certain point, after traveled a few points, it went back to the original point, and then began a new path. Therefore these cycle points may be removed from the initial path. Figure 3.9(b) illustrates this type of redundancy.



(a) Single Extra Point. (b) Multi Extra Points.

Figure 3.9 Redundancy.

Chapter 4
Simulation Result

We have implemented the algorithms described in Chapter 3 with computer simulation. The program for the algorithms is implemented on a Sun Sparc-I workstation. It allows users to specify the obstacles, the starting and goal point configuration. The result, i.e., the collision-free path, is displayed graphically.

The algorithm has been extensively tested with many different examples. Some representative ones are illustrated in Figure 4.1 – Figure 4.20. In these figures, the decomposition result is shown in different levels of approximation using different sizes of hexagon: the largest hexagon corresponds to the region of influence under the given positions of the starting point and the destination point; the midsized hexagons are the initial approximation of the region of influence based on the sizes of the obstacles. The smallest hexagons correspond to the decomposition of its parent hexagon which is necessary when the robot changed the searching step size. The collision-free paths are shown with line segments connecting the centers of the hexagons passed. The obstacles are shown as shaded polygons. The t value shown in the figures indicates the minimal path width.

In all the cases we have tested, the collision-free path can be found in less than 0.5 second of CPU time. Our method is suitable for any polygon obstacle, either convex or concave.

There are three constants in Eq. (1), k_1, k_2, and C. In the simulation, we found that these constants do not have a critical impact on the results as long as the attractive force dominates. However when the sum of the repelling forces is

much larger than the attractive force, the algorithm might fail because in this case, the directional guide to the goal is so weak that the mobile robot can not sense it. For the examples shown here, k_2 is set to $1/\pi$ to make the value in the parentheses in Eq. (1) positive (since α is less than or equal to π radians), k_1 is set to 100, and C is set to 1.

Figures 4.1 – 4.2 are the cases with $t = 24$ points which mean that the minimal path width is nearly the height(25 points) of a midsized hexagon in the figures. While Figures 4.3 – 4.13, and Figures 4.17 – 4.20 are the cases with t=48 points which is near the double height (50 points) of the smallest hexagons in the figures.

The region of influence for Figure 4.1 is initially decomposed into 361 full and half hexagons. The starting and goal configuration are respectively at the bottom and top of the largest hexagon. Initially, the step size of the mobile robot is at l_1 (the edge length of the midsized hexagons) and the robot proceeds directly toward the goal due to the potential function. After the first two steps, the potential function for a direct path to the goal increases significantly due to the presence of the obstacle O_1. The robot is steered away from O_1 by the potential function and the robot selects the hexagon H_1, which is PASSABLE and MIXED. H_1 is then decomposed into smaller hexagons and the robot proceeds with step size equal to $l_1/4$. As the robot navigates around the obstacle, it reaches the hexagon H_2 which is PASSABLE and EMPTY, and the step size returns to the original value. The algorithm proceeds and finally reaches the goal configuration.

After the initial path has been found, it is tested for redundancy. For

example, there is a single unnecessary path point in Figure 4.2 which is shown as a stand-along point without any line segments connecting to it. For the local minima cases, we have shown them in Figures 4.4, 4.5, 4.7, and 4.11 – 4.13.

Figure 4.5 and 4.6 are for the same scenarios. They show the initial path and the final path, before and after the redundancy is removed, respectively.

Figure 4.3 is a case when at the right beginning of the searching, the robot could not proceed with the normal step. Thus it initiated sub-level searching immediately. Figure 4.4 shows a case when there is an obstacle in the hexagon in which the target point is located.

The most difficult case given here is Figure 4.13, where the robot is initially trapped in a local minima but eventually escapes and reaches the goal. At the start configuration, it is misled by the potential function and proceeds to its left side where there does not exist a path to the goal configuration. After exploring several possible paths, it successfully finds an opening and goes out of the local minima, and finally reaches the goal configuration.

Figures 4.14 – 4.16 are used to show the effect of the minimal path width. As shown in the figures, as the width is increased from 24 points to 72 points, the shape of the path generated is slightly changed to avoid collision with obstacles. When the width is larger than the minimum gap between obstacles, a path can not be found, and the program stops.

Figures 4.17 – 4.20 are simulated according to the paper [16], for the purpose of comparison with their planner.

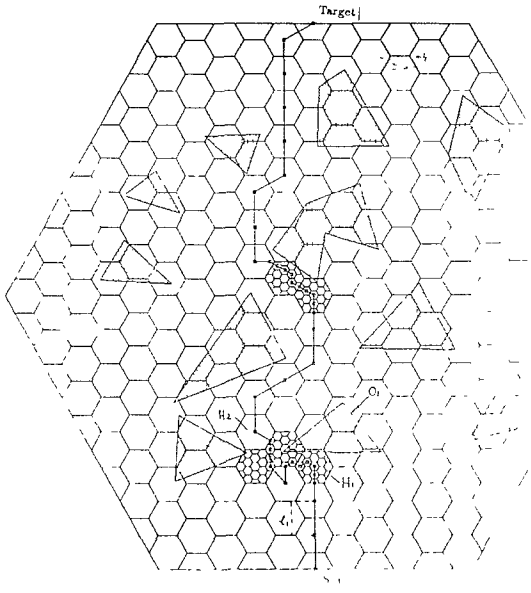


Figure 4.1 Example 1, t=24 points

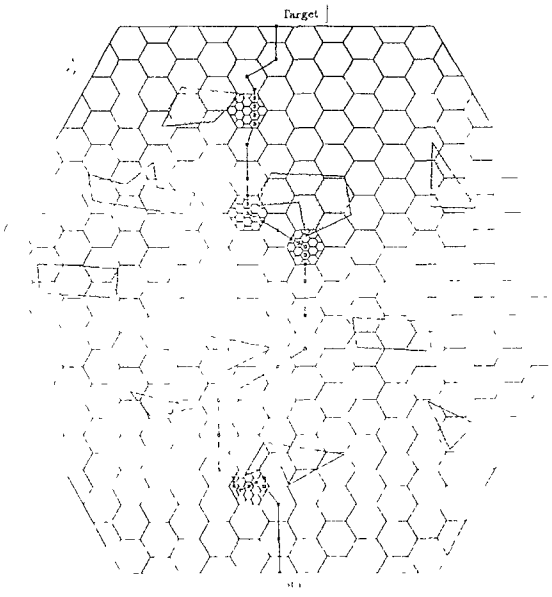


Figure 4.2 Example 2, t=24 points

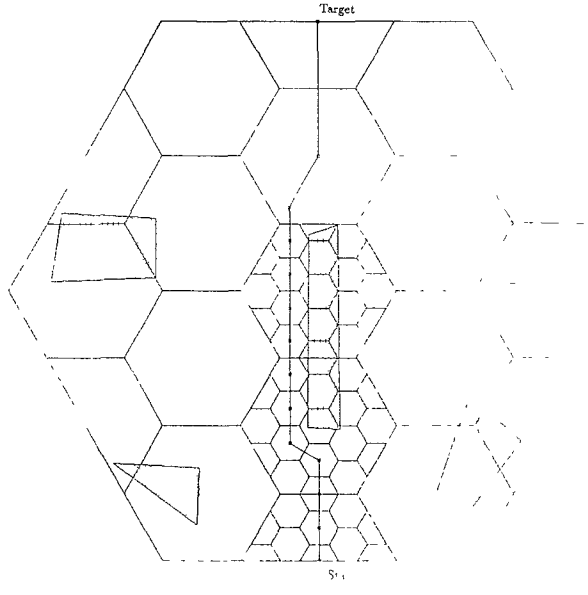


Figure 4.3 Example 3, t=48 points

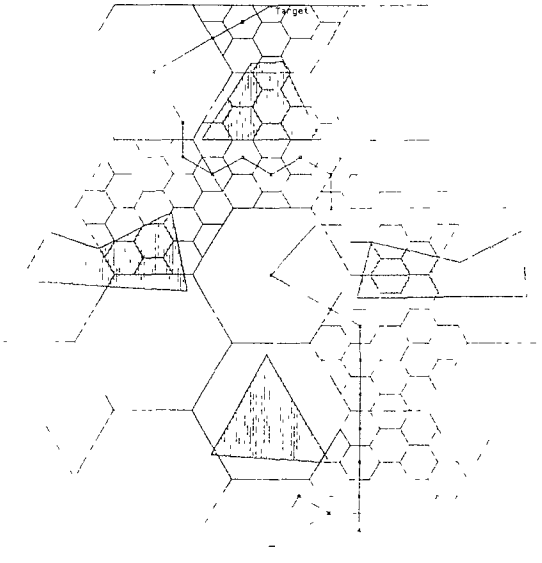


Figure 4 4 Example 4, t=48 points

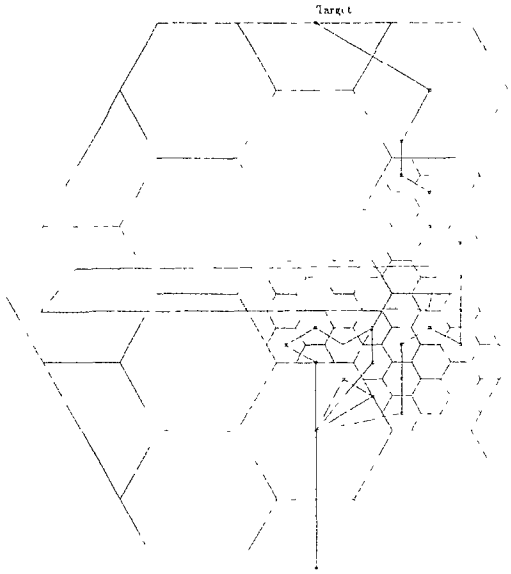


Figure 4.5 Example 5, t=48 points

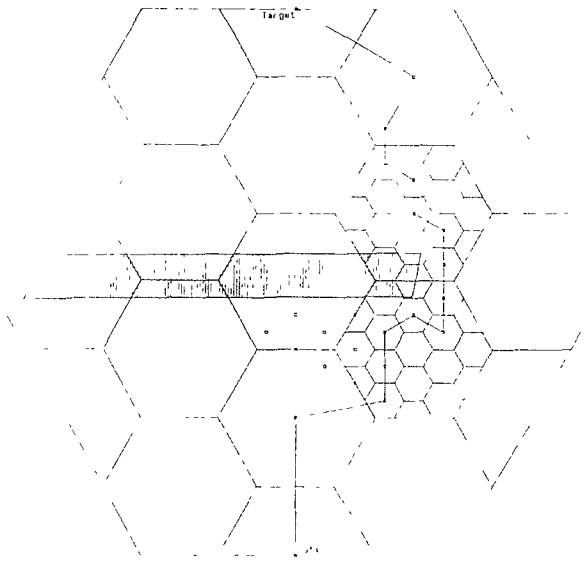


Figure 4.6 Example 6, t=48 points

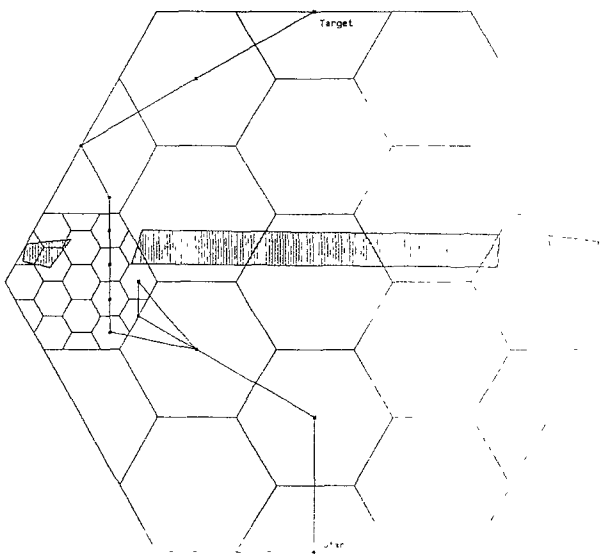


Figure 4.7 Example 7, t=48 points

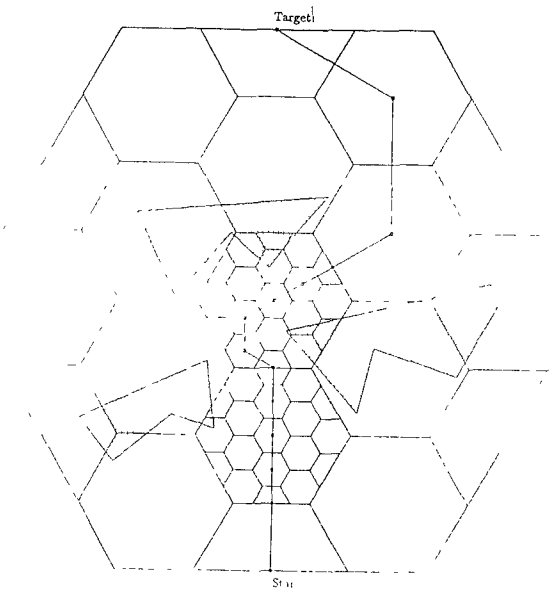


Figure 4.8 Example 8, t=48 points

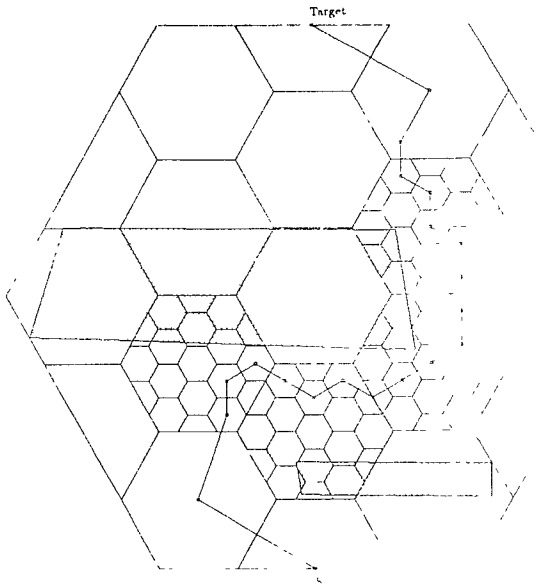


Figure 4.9 Example 9, t=48 points

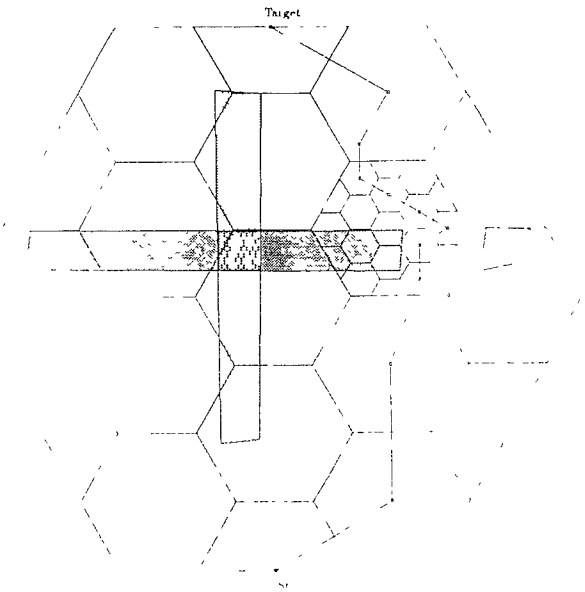


Figure 4.10 Example 10, t=48 points

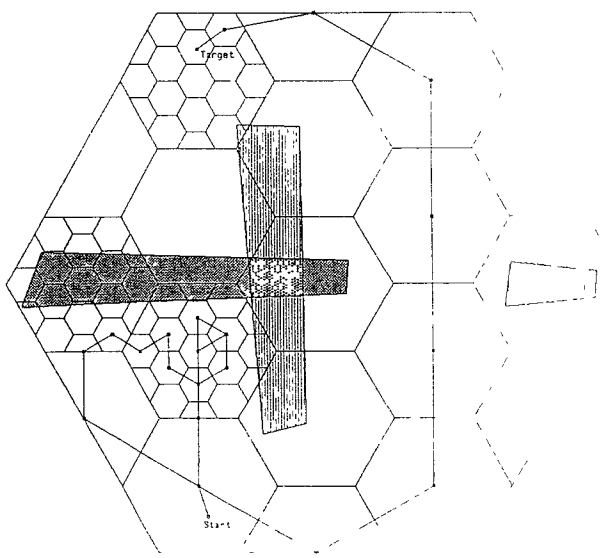


Figure 4.11 Example 11, t=48 points

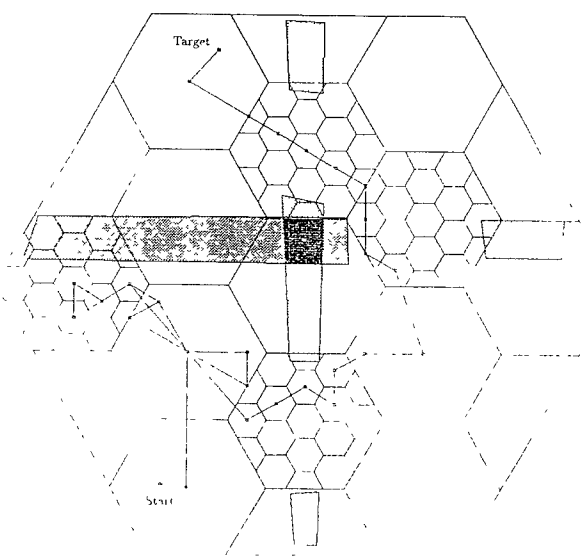


Figure 4.12 Example 12, t=48 points

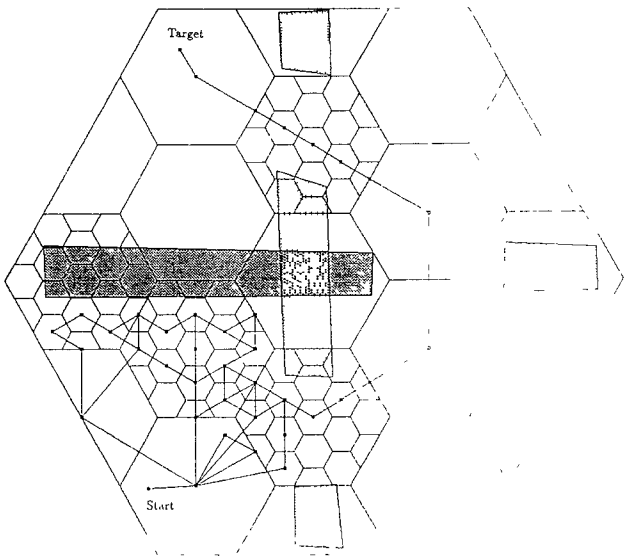


Figure 4.13 Example 13, t=48 points

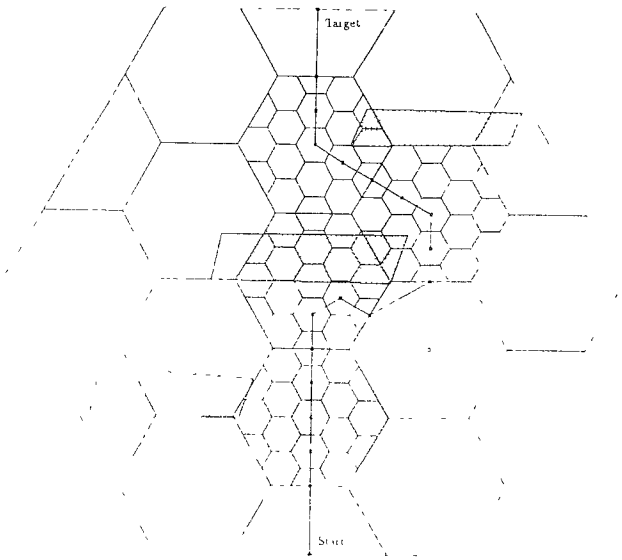


Figure 4.14 Example 14, t=24 points

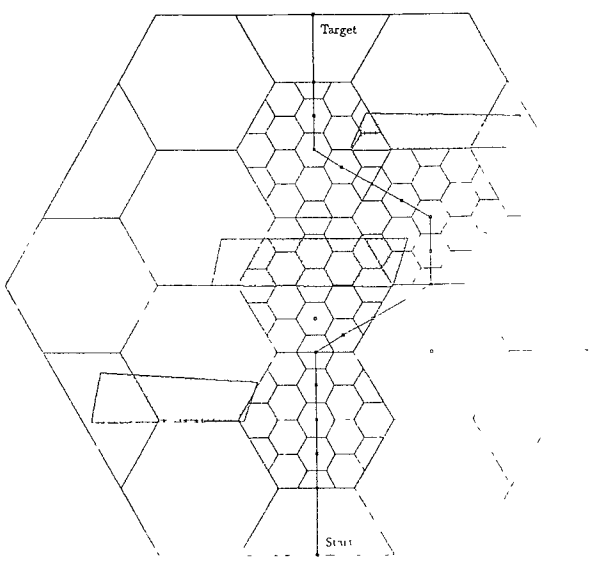


Figure 4.15 Example 15, t=48 points

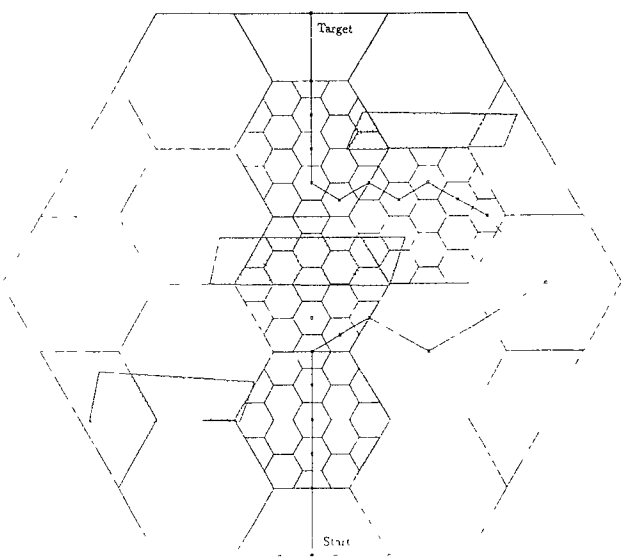


Figure 4.16 Example 16, t=72 points

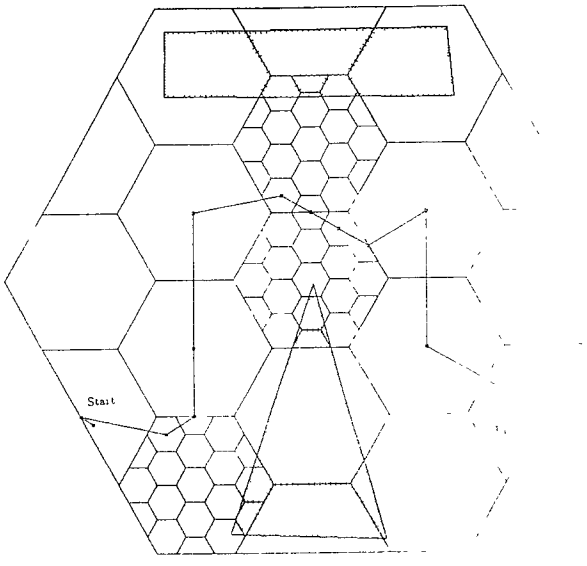


Figure 4.17 Example 17, t=48 points

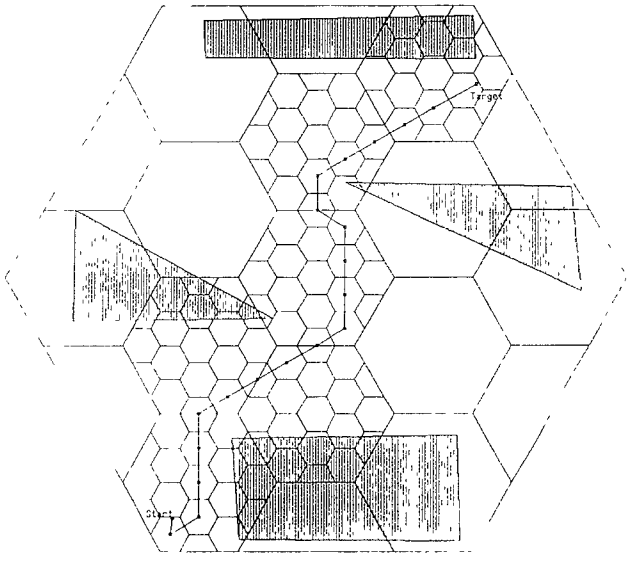


Figure 4.18 Example 18, t=48 points

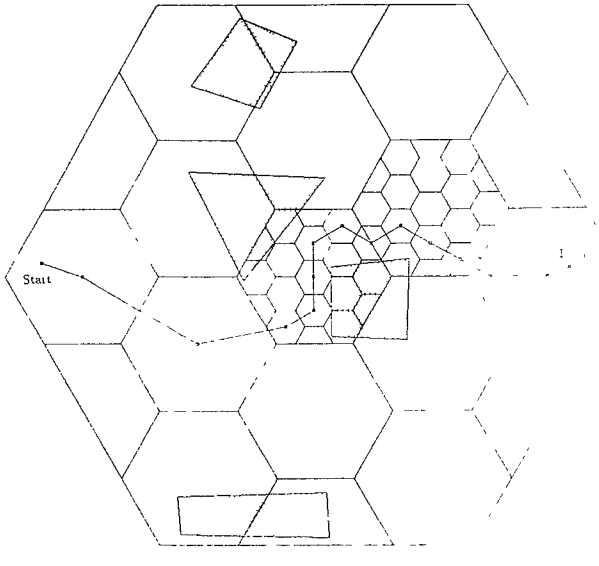


Figure 4.19 Example 19, t=48 points

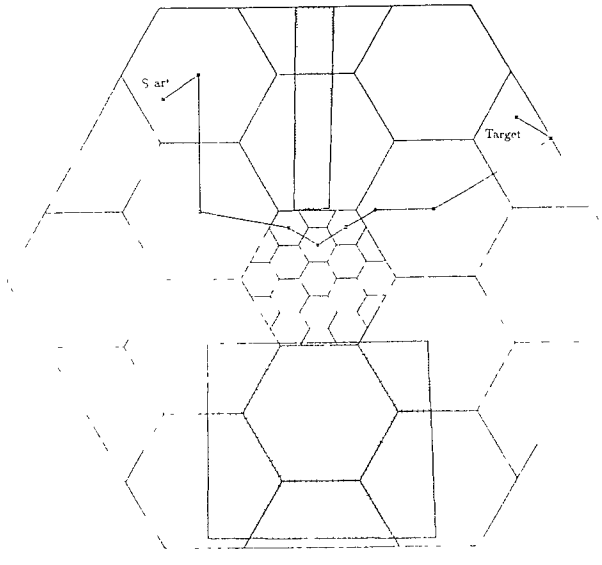


Figure 4.20 Example 20, t=48 points

Chapter 5
Conclusion

A new collision-free path planning method for a mobile robot based on the hierarchical decomposition of hexagons and an artificial potential field is presented in this thesis. The work space of the mobile robot is hierarchically decomposed into cells of full or half hexagons. Two attributes — PASSABLE /IMPASSABLE and EMPTY/MIXED, are defined for the cells to facilitate the searching algorithm. A neighborhood searching algorithm guided by an artificial potential field is used to search the hexagonal cells to find a collision-free path. Two attributes of this approach are its fast response and robustness. The algorithm has been tested extensively and can find a collision-free path even for very complicated scenarios. The main contribution of this research is in combining the hierarchical decomposition technique with an artificial potential field to find collision-free paths.

Future research may be done to loosen the limitation of the minimal path width t to the width w of the rectangle for the mobile robot model. Since the mobile robot can rotate, the above improvement can be implemented by determining from where and how much the mobile robot should start to rotate when approaching an obstacle to avoid collision with it and then to pass through it.

Bibliography

[1] J. Reif, "Complexity of the mover's problem and generalizations," *Proceedings 20th Symposium of the Foundations of Computer Sciences*, 1979.

[2] J. E. Hopcroft, J. T. Schwartz, and M. Sharir, "On the complexity of motion planning for multiple independent objects: Pspace Hardness of the 'Warehouseman's Problem', " Robotics Report 14, N.Y.U. Courant Institute Department of Computer Science, New York, 1984.

[3] T. Lozano-Pérez, M. Brady et al., *Task planning in robot motion: planning and control*, MIT Press, 1983.

[4] S. H. Whitesides, "Computational geometry and motion planning," in: G. T. Toussaint (ed.) *Computational Geometry*, North-Holland, 1985, pp. 377-427.

[5] M. Sharir, "Algorithmic motion planning," *Computer*, Vol. 22, No. 3, March 1989, pp. 9-20.

[6] T. Lozano-Pérez, "Automatic planning of manipulator transfer movements," *IEEE Trans. on SMC*, Vol. 11, 1981, pp. 681-698.

[7] T. Lozano-Pérez, "Spatial planning: a configuration space approach," *IEEE Trans. on Computers*, Vol. 32, 1983, pp. 108-120.

[8] R. A. Brooks, "Planning collision-free motions for pick-and-place operations," *J. Robotics Research*, Vol. 2, 1983, pp. 19-44.

[9] B. R. Donald, "A search algorithm for motion planning with six degrees of

freedom," *Artificial intelligence*, Vol. 31, 1987, pp. 295-253.

[10] T. Hasegawa, "Collision avoidance using characterized description of the free space," *Proc. Int. Conf. on Advanced Robotics* (Tokyo, 1985), pp. 69-76.

[11] T. Lozano-Pérez, et al, "Task-level planning of pick-and-place robot motions," *Computer*, Vol. 22, No. 3, 1989, pp. 21-29.

[12] R. A. Brooks and T. Lozano-Pérez, "A subdivision algorithm in configuration space for findpath with rotation," *Proc. 8th Int. Joint Conf. Artificial Intelligence* (Karlsruthe, FRG, 1983), pp. 799-806.

[13] B. Faverjon, "Object level programming for industrial robots," *Proc. 1986 IEEE Int. Conf. on Robotics and Automation* (San Francisco, CA, April 7-10), pp. 1406-1412.

[14] S. Kambhampati and L. S. Davis, "Multiresolution path planning for mobile robots," *IEEE J. Robotics Automation*, Vol. RA-2, No. 3, 1986, pp. 135-145.

[15] C. Laugier and F. Germain, "An adaptive collision-free trajectory path planner," *Int. Conf. Advanced Robotics* (Tokyo, Japan, 1985).

[16] D. Zhu and J.C. Latombe, "New heuristic algorithms for efficient hierarchical path planning," *IEEE Trans. on Robotics and Automation*, Vol. 7, No. 1, 1991, pp. 9-20.

[17] T. Hague, M. Brady and S. Cameron, "Using moments to plan path for the Oxford AGV," *Proc. 1990 IEEE Int. Conf. Robotics and Automation* (Cincinnati, OH, May 13-18), pp. 210-215.

[18] O. Khabit, "Real time obstacle avoidance for manipulators and mobile robots," *J. Robotics Research*, Vol. 5(1), 1986, pp. 90-99.

[19] N. Hogan, "Impedance control: an approach to manipulation," *ASME Journal of Dynamics Systems, Measurement, and Control*, Vol. 107, Mar 1985, pp. 1-7.

[20] C. W. Warren, "Global path planning using artificial potential fields," In *Proc. 1989 IEEE Int. Conf. on Robotics and Automation* (Scottsdale, AZ, May 14-19), pp. 316-321.

[21] D. E. Koditschek, "Exact robot navigation by means of potential functions: some topology considerations," Technical report 8611, Center for Systems Science, Yale University, 1986.

[22] F. Miyazaki and S. Arimoto, "Sensory feedback based on the artificial potential for robots," *Proceedings 9th IFAC*, (Budapest, Hungary, 1984).

[23] V. V. Pavlov and A. N. Voronin, "The method of potential functions for coding constraints of the external space in an intelligent mobile robot," *Soviet Automatic Control*, Vol. 6, 1984.

[24] A. Bowyer and J. Woodwark, *A programmer's geometry*, London, 1983.

Appendices

Appendice A
User's Guide to the Programs

There are five programs which can be used by a user, which are:

get_input [parameter] > filename1

get_schedule [parameter] > filename2

prepare filename1 filename2]

plan_path

display

The get_input program is used to obtain the coordinates of the vertices of obstacles using mouse as an input devicce. When you click the left mouse button, you specify the coordinates for a vertex of an obstacle. After you have specified all the vertices, click the middle mouse button, the obstacle will be displayed on the screen. After you have specified the obstacles you want, move the mouse pointer to within the window the program created for you, then click the right mouse button to exit the program. This program should be run before the plan_path program is run.

The parameter following the program command is a digit [0-3], which is used to indicate the level of approximation you want the program to decompose the region of influence.

The get_schedule program is used to get the starting point of the mobile robot and the target point. Use the left mouse button to select those two points. Each time

you click the botton, you will get one point. Then click the right mouse button to exit the program(the mouse pointer should be within the window the program created for you). This program should be run before the plan_path program is run.

The meaning of the parameter is the same as for the get_input program.

The prepare program is used to prepare two binary files of "obstacle" and "schedule" for the use of the plan_path program. It has two parameters which are exactly the ones you created using get_input and get_schedule programs. This program must be run before the plan_path program is run, otherwise you will get the "core dumped" run time error.

The plan_path program, run by itself, doesn't have parameters. It is used to find a collision-free path guided by an artificial potential field. Finally you should run display prorgram to put all the results into one screen.

The get_input, get_schedule, and display programs must be run on SunView environment.

The above programs may be put together into an ASCII command file such as follows:

```
get_input 1 > obs.dat
get_schedule 1 > sche.dat
prepare obs.dat sche.dat
plan_path
display
```