2010 Third International Conference on Intelligent Networks and Intelligent Systems

The Path Planning for Mobile Robot Based on Voronoi Diagram

Huiying Dong[1], Wenguang Li[1]
Shenyang Ligong University
School of Information Science and Engineering
Shenyang, China
huiyingdong@163.com

Jiayu Zhu[2], Shuo Duan[1]
Shenyang Normal University
Shenyang, China

***Abstract*—In order to get the shortest path of the mobile robot , Voronoi diagram has been introduced to the robot path planning in the paper. The paper uses the voronoi diagram method to model the work space, and obtained the global shortest path by the improved Dijkstra algorithm, besides, when dealing with the unknown static obstacles, the paper adopt one method of constructing the local Voronoi map of the environment, which can contribute to reduce computation time and satisfied the needs of robot's real-time path planning. Simulation results show that the method is simple and easy to implement.**

Keywords- Mobile Robot; Voronoi Diagram; Dijkstra Algorithm;

I. INTRODUCTION

The path planning is the basic element of the mobile robot, the so-called mobile robot path planning is searching for a road connect the initial position to target position, which is the optimal or sub optimal collision-free path. According to the environmental information that the robot has grasped, the path planning can be divided into two categories: path planning with fully known global environmental information and environmental information is completely unknown or partially unknown[1].The common method that has been used, such as grid method and artificial potential field method and so on, but all these algorithms exist some limitations of the algorithms themselves. The shortcomings of the grid method are when the space increases, it will lead to dramatic increase in storage space that required, and decision was made slowly. The structure of artificial potential field is simple, convenient for real-time control, but may generate minimal path and oscillation in front of obstacles.

Based on Voronoi diagram method for path planning, firstly, there needs known static obstacles which construct the Voronoi diagram, then obtained the shortest path by optimal search algorithm, but due to obstruction is not completely known, during the process of moving of the robot, when the robot detected a new obstacle, need to construct the local environment as a Voronoi diagram. Then the robot generate a new shortest path on the new Voronoi diagram. The advantages of the Voronoi diagram is that it can generate the origin robot path points, and reduce the path search time.

II. ENVIRONMENT MODELING BASED ON VORONOI DIAGRAM

The construct methods of Voronoi diagram that has been commonly used are: half-plane intersection method, incremental construction method and the divide and conquer method etc[2] [3].Half-plane intersection method is based on the definition of Voronoi diagram, obtain the result by calculating the intersection of half-plane directly, its time complexity is $O(n^3)$, its significance only can display in theory, but not used in the actual process. Incremental construction method, firstly, build a simple Voronoi diagram on 2 or 3 generators, then we do not stop adding one generator each time and modify the Voronoi diagram, until all of the generator has been added into the Voronoi diagram. The time complexity in worst-case is $O(n^2)$, in average case is $O(n)$, it is an effective method that is used in practical, besides, it is the mostly used method; The divide and conquer method divide generating set into several parts firstly, construct the local Voronoi diagram for each part, and then merge all the sub-graph, which constitute the entire Voronoi diagram. Its time complexity is $O(n^3)$.

Here the paper reshape the obstacles by pretreatment, namely, if two obstacles are very near, besides the distance between obstacles is less than the minimum safe distance of the robot, then the two obstacles will be merged into one barrier. Paint the circumcircle of obstacles, then expand the circumscribed circle of the obstacles, take the center of a circle as a generator of the Voronoi diagram. The specific steps of incremental construction method to construct the Voronoi diagram is as follows:

Step1:Take three points p_i, p_j, p_k from a known generators $p_1,p_2\ldots p_n$, and connect them into a triangle $p_i p_j p_k$.

Step2:Computing center v and radius d of circumcircle of the triangle $p_i p_j p_k$.

Step3:Calculating the distance $d(p_b, v)$($b=1,2,\ldots n, b\neq i,j,k$) and arrange it from small to large, set the corresponding point as $p_1,p_2\ldots p_{n-2}$,$l=1$.

Step 4: If $d(p_l, v) > d$, turn to step 6. Otherwise, turn to step 5.

Step5: Take the other three points p_i, p_j, p_k or p_i, p_l, p_k or p_i, p_j, p_l to form a triangle, if there are multiple points $p_1,p_2\ldots p_m$ within the circle $C(v)$, we can also connect p_1,p_2,p_3 to form a triangles, then turn to step 2.

978-0-7695-4249-2/10 $26.00 © 2010 IEEE
DOI 10.1109/ICINIS.2010.105

Step 6: Determine which edge right side or which two edges right side was p_l on, modify the boundary which Voronoi polygon the p_l belongs to.

Step 7: $l = l + 1$,turn to step 6 until $l > n - 3$.

Step 8: Connect the center of a circle of adjacent triangles in order to construct Voronoi diagram.

Voronoi structure by the method is shown in Figure 1.

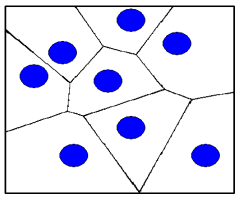

Figure 1 Incremental Construction method generation of Voronoi diagram

III. DIJKSTRA ALGORITHM FOR SEARCHING SHORTEST PATH

Because the starting point and end point are not considered when construct Voronoi diagram, so we need to add the starting point and end point into the diagram before searching path, in this article, the criteria of adding the start and end points to the diagram is: treat the Voronoi vertex and the mid-point of Voronoi edge as temporary points, connect the start point and the end point to the nearest temperary point separately.

In addition to adding the starting point and end point, we also need to calculated the distance between the path points. The paper treat the Voronoi vertex, the starting point ,end point and the mid-points of Voronoi edge that connected the starting point and end point as the path points.

The searched robot path is stored by adjacent matrix, the elements value of the matrix is expressed by the weight of the path points. The corresponding weight value is infinite if there exist no edge between the two weight values. The specific steps of calculating the shortest path between starting point u_0 and end point v_0 are as follows:

Step 1: Suppose $d(u_0)=0$, while all other vertices on v run, Suppose $d(v)=\infty$,besides, make u_0 a label $t \leftarrow u_0$

Step 2: Deal with the vertex v with no label one by one, Calculate the value of d(v), and make its value the minimum.

$$d(v) = \min\{d(v), d(t) + W(t,v)\} \qquad (1\text{-}1)$$

The W(t,v) stands for the weight value between point t and point v, if d(v) has modified the origin value, then it states that t and v on the obtained optimum path were adjacent. Select the minimum d(s), d(s)= min{d(v)}. Make s a label, which demonstrate the length of the optimal path that between vertex u_0 and point s.

Step 3: Determine whether the end v_0 has label. Stop counting if there exists label, that is to say, we have been got the shortest path from u_0 to v_0. Otherwise, $t \leftarrow s$, turn to step 2.

The time complexity of one algorithm's is relative to the running time of the algorithm. Running time of an algorithm is the length of time that spent from start to finish on the computer. The time consumed of a program algorithm does not only depend entirely on the design, but also interrelated to data structures and the computer related hardware and software environment. In other words, running time is different even using the same method when using a different programming language or a different computer to achieve. So here, without regard to computer-related hardware and software factors, only consider the algorithm to run their own workload.

The workload of an running algorithm depends on the scale of problem, the problem scale is a whole number of measure of the size of the of issues, usually denote by n. Therefore, when processing the issue of n scale by the algorithm, the running time is usually expressed by the T(n). T(n) is called the time complexity of the algorithm. Time complexity that we usually encounter: such as, constant order O(1), log order O(log n), linear order O(n), linear log order O(n log n), square order $O(n^2)$ etc.

For the environment map that containing n path points, and there needs cycle n-1 times when seeking the shortest path between aim point to the other path points. During the cycle, need to modify the shortest path when found a shorter path, because the change was cycle in cycle, that is containing two-story cycle, so the time complexity of Dijkstra algorithm for the $O(n^2)$.

IV. DIJKSTRA ALGORITHM

We found there exists a large number of infinite value when checking the adjacency matrix (in the programming process, the large value of adjacency matrix were written as infinite value), the calculation of the infinite values are invalid when doing calculations by type 1-1. It will waste a lot of time if the environment map contains more path points,.

The basic idea of the improved Dijkstra algorithm: when the program runs to the step 2 that aforementioned, before doing the operation $d(v) = \min\{d(v), d(t) + W(t,v)\}$, Judge if $d(v)$ and $d(t) + W(t,v)$ are infinite, if any, , then continue the operation, otherwise, stop the type operation, by doing so can avoid a large number of invalid calculation and improve the search efficiency.

V. UNKNOWN STATIC OBSTACLES AVOIDANCE DEAL

At any time there would appear of new obstacles in the robot track that have been programmed, because of the complexity of the robot's operating environment. This is particularly important to deal with the unexpected situations, it is related to the safety of the robot and obstacles avoidance, to reach a successful end.

The paper does some assumptions on the robots and their environment ,as following:

1. Assume that the robot can detect real obstacles in front by multiple sensors, and the distance detect by sensors does not affect the normal robot path planning.

2. Assuming that the new obstacles that detected are pre-disposed, simplified to round obstacles, and that the barrier pass through "puffing" process.

3. Assume that the robot can detect center of a circle ofis the obstacle by sensor .

4. Assume that the robot will not detected more than one obstacle at the same time.

When the robot detects obstacle blocking in ahead of the road through the sensors, the robot needs to dispose the detected obstacle, the approach is restructure the environment map . In this article, when the robot reconstruct the map of the environment is not mean that reconstruct the whole environment, but in the local carry that has been newly tested , the most advantage of this construction method is it has reduced the robot computing time, and provide the robot effective protection on real-time processing.

The specific local environmental map structure as follows:

Step 1: Get the center of circle of the new obstacle.

Step 2: Remove the affected path points.

Step 3: Construct the map of local environment.

Step 4: linked the newly formed local Voronoi diagram into the previously formed map of the robot environment , which construct an new environment map.

Through the above approach , the robot can avoid new obstacles by further processing, towards the target, after the local newly environment map has been constructed.

The specific obstacle avoidance algorithm of robot from the start to the end is as follows:

Step 1: construct Voronoi map of the environment.

Step2: adding the starting and ending points into the environment map, and determine the path points of the environmental map.

Step 3: searching shortest path by Dijkstra algorithm.

Step 4: the robot' move towards the end along the searched path.

Step 5: When the robot get to the end, then stop moving, quit the program. When the sensors detect new obstacles in the forward, memorize the barrier's center (generator) and record the current point of the robot.

Step 6: construct the local environment map of the robot , connected current robot position to the nearest path point. Then turn to step 3.

VI. SIMULATION RESULTS AND ANALYSIS OF PATH PLANNING

We could see the different shortest path of robot with different start and end position from Fig 2. All the obstacles in the map are known before, B is the robot's starting point, while the E is target position of the robot, the robot using Dijkstra algorithm to select a shortest path road.

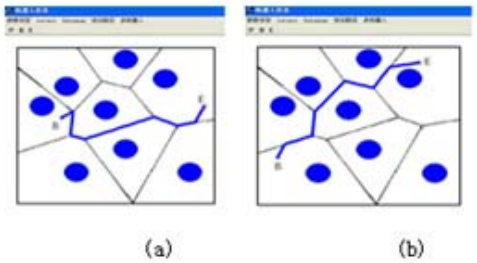

Figure 2 Dijkistra algorithm for path search

Since this paper adopt the improved Dijkstra algorithm for the path searching, when searching in the environment map shown in Fig 2, comparison results that doing calculation of infinite value by classical Dijkstra algorithm and the improved Dijkstra algorithm are shown in Table 1.

TABLE 1 THE COMPARISONS OF CLASSICAL DIJKSTRA ALGORITHM AND IMPROVED DIJKSTRA ALGORITHM

Algorithm name	Calculation times (Figure 2 (a))	Calculation times (Figure 2 (b))
Classical Dijkstra algorithm	55	66
Improved Dijkstra algorithm	23	26

Table 1 indicates that the improved Dijkstra algorithm has greatly reduced the number of calculation times of infinite value. So it does help to rise the search efficiency.

When the robot environment map exists unknown obstacles, as shown in Fig 3. Then, the path planning of robot to avoid obstacles as shown in Fig 4 and Fig 5.

Compared Fig 2 with Fig 3 , in the robot's shortest path ,there is a unknown obstacle in Figure 3, the obstacle blocks the robot.

As shown in Fig 4 and Fig 5, we can see, (a) stands for when the sensor detects new unknown obstacles, the robot stop moving, and then construct the local new Voronoi map of the environment. And (b) stands for the robot treat the stop point as a new starting point, and then search new path using Dijkstra algorithm.

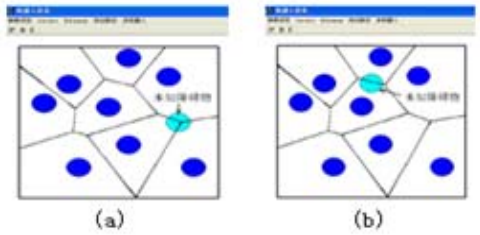

Figure 3 Environment map with unknown obstacles

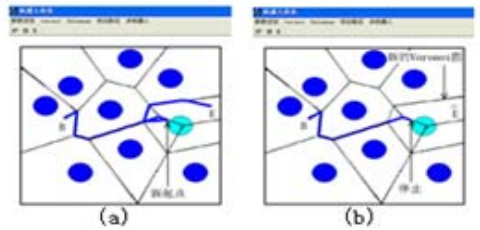

Figure 4 Robot path1 avoid unknown obstacles

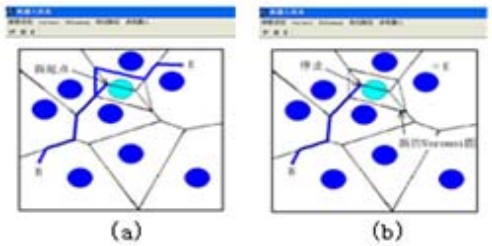

Figure 5 Robot path 2 avoid unknown obstacles

VII. CONCLUSION

This paper introduce Voronoi diagram into the robot path planning, the improved Dijkstra algorithm that has been used is simple and easy to implement, besides it has greatly reduced the number of calculation, when the robot encounters new obstacle during the path planning, what's more the paper introduce Voronoi algorithm to construct local environment map to reduce computation time, satisfied the demand for robot path planning in real time.

REFERENCES

[1] Han-Dong Zhang, Zheng Rui, Yu-wan Cen. Mobile Robot Path Planning Current status and prospects [J]. System Simulation, 2005,17 (2) :pp.439-443.

[2] Lina Chen, Yu-Jie Ma. Planar point set based on the approximate Voronoi diagram construction. Software space .2007,2 (23) :pp.263-264.

[3] Y.Choi , K.Kim,S.Lee.Shape Reconstruction from Unorganized Points Using Voronoi Diagrams.Engineering Science.2003,11(21):pp.446-451.

[4] S.Fortune.A Sweepline Algorithm for voronoi Diagrams.Algorithmic.1987.5(2):pp.153-174.

[5] Zhou's. Computational geometry - Algorithm Design and Analysis. Second Edition. Beijing: Tsinghua University Press .2005.pp.152-153.

[6] Lee J.Calculation of the shortest paths by optimal decomposition.IEEE Tran. On System, Man and Cybemetics.1982(3):pp.410-415.

[7] Bjornsson Y, Enzenberger Holte, M.R. Comparison of different abstration for pathfiding on map. International Joint Cjonference on Artificial Intelligence.2003.pp.1511-1512.