Obstacle Prediction–based Dynamic Path Planning for a Mobile Robot

Zhenyu Wu, Lin Feng*
School of Innovation Experiment, Dalian University of Technology, Dalian 116024, P.R. China, fenglin@dlut.edu.cn

Abstract
A path-planning method based on a combination of the static global and the dynamic local path-planning methods is proposed for robot path planning under a complex environment. There are known static obstacles and unknown dynamic obstacles in any complex environment. The local-path planner dynamically generates a local path using obstacle-motion prediction and a rolling window for dynamic path planning to partially adjust the global path. Simulation results show that the mobile robot achieves both overall and local obstacle-avoidance during motion, in association with an optimum path, which verifies the feasibility and effectiveness of the method.

Keywords: Mobile Robot, Path Planning, Obstacle Prediction

1. Introduction

The task of path planning for mobile robots has continued to challenge researchers. It could be described as a process of determining a path in configuration space between the initial and final configurations of the robot such that (1) the robot does not collide with obstacles and (2) the planned motion is consistent with the kinematic constraints of the vehicle [1]. Most previous studies in this realm have been aimed at solving problems related to avoiding static obstacles [2]. To date, this type of problem has been thoroughly studied. Recent interests have shifted to dynamic obstacle-avoidance under complicated circumstances, which includes avoidance of both static and dynamic obstacles. Many approaches have been provided to solve this problem, and some important achievements have been obtained. Ivan Koryakovskiy [3] has presented a strategy of dynamic path planning using genetic computation. Qidan Zhu [4] provides a dynamic collision-avoidance planning algorithm based on velocity-change space. Lu Li [5] outlines a method using particle-swarm optimization to accomplish dynamic path planning. However, dynamic path planning is generally in the initial stages of development; thus, many theories and methods still need improvements in future studies. Nowadays, a combination of many path-planning methods is used to achieve dynamic path planning, and this has become a trend in the path-planning field.

Considering the differences between the information obtained in relation to a given environment, there are two main categories of path planning: Global Path Planning, which encompasses all the acquired knowledge of a robot to reach a goal; and Local Navigation, which is the process of using only the robot’s currently sensed information regarding its immediate world [6]. The local navigation algorithm includes artificial potential-field approach, fuzzy logic algorithm [7,8], and genetic algorithm [9]. The main algorithms of global path planning include the A* algorithm, the visibility-graph algorithm, and the topology-map algorithm [10]. However, a single algorithm has disadvantages when the surrounding environment is complex. Global path planning can easily obtain a static optimum path; however, imperfections caused by a large scanning space, a complicated algorithm, and limited efficiency make real-time collision-avoidance impossible. On the contrary, the local navigation algorithm can satisfy the requirements of collision-avoidance with good efficiency, but achieving the optimum path is difficult. In this article, we present a new solution by combining the advantages of both types of algorithms. This solution predicts the trajectory of dynamic obstacles to execute local navigation, which is used to adjust the global optimum path and thereby achieve local real-time path planning when encountering both static and dynamic obstacles in a complicated environment.
2. Model of Dynamic Prediction

Complicated environments generally contain static obstacles, whose position information is known; and dynamic obstacles, whose position information is uncertain. In the study of path planning, numerous effective algorithms for avoiding static obstacles have been furnished, while avoiding dynamic obstacles reasonably becomes a bottleneck. Precise prediction of future collision cannot be achieved if the recent state of motion is not determined. A method usually followed involves expanding the current dynamic obstacles into static ones. Usually, the expansion radius is the sum of the dynamic obstacle’s real size and the maximum displacement during a single planning cycle. Then, robots can attempt static path planning. Although collision-avoidance is realizable with this approach, a larger planning area, a suboptimal resultant path and the death area in the path can still be considered imperfections. Information related to the real-time motion of a dynamic obstacle in each single planning cycle is difficult to know; nevertheless, the complete trajectory of a dynamic obstacle is regular. Hence, the motion state of a dynamic obstacle in the next planning cycle is predictable if the information on current motion is taken into account.

A comprehensive consideration of practicability and efficiency is required during construction of a prediction model. A very complicated model will cost too much time, which may weaken the performance of real-time control. A dynamic obstacle moves continuously; hence, it can be treated as undergoing uniform linear motion in two adjacent path-planning cycles \( T \) when \( T \) is small, because the sampling rate of the robotic vision system is high. In a complex environment, the trajectory of a dynamic obstacle shifts with time and is given by the following expression:

\[
S(s, y) = f(t)
\]

(1)

The trajectory of a dynamic obstacle is continuous. Hence, the movement between two adjacent planning cycles can be treated as approximately uniform rectilinear motion when the vision system of mobile robots consists of a tiny planning cycle \( T \), which results from a high sampling rate. An obstacle’s position \( S_1 \) at time \( T_1 \) can be predicted according to its positions \( S_0 \) at time \( T_0 \) and \( S_1 \) at time \( T_1 \), which are given by

\[
S_2 = S_1 + \Delta S
\]

(2)

The predicted coordinates of the dynamic obstacle at time \( T_2 \) are given as

\[
\begin{cases}
  x(2) = 2x(1) - x(0) \\
  y(2) = 2y(1) - y(0)
\end{cases}
\]

(3)

3. A Path-Planning Algorithm

3.1 General Description

Our algorithm contains two general aspects: global static obstacle-avoidance path planning, and local dynamic collision-avoidance navigation. We adopt the A* algorithm \(^{[11]} \) to obtain the static optimum path and use trajectory prediction to achieve real-time collision-avoidance. We assume that there are only static obstacles at the initial time and use the A* algorithm to obtain a global optimum path from the initial position to the target position. Then, the robot measures the distance \( L \) between the center of the dynamic obstacle and itself when moving along the optimum path \( S \). A threshold \( M \), decided by (1) the velocity of the robot and (2) the sizes of the
robot and the obstacle, is set up for distance \( L \). The robot predicts the future position of the dynamic obstacle if \( L \) is smaller than \( M \). Local navigation can be accomplished according to the result of the prediction and the value of \( L \). The final optimum path can be obtained by repeating this process for each single planning cycle until the robot attains the target position.

3.2 Global Optimum Path

The A* algorithm is well known as one of the best-first search searching methods among the numerous global path-planning algorithms. The A* algorithm is actually an enlightening searching method, and its instinctive nature is “enlightening,” which avoids blindness during the path–planning process. Hence, the A* algorithm is one of the popular path-planning algorithms. The D* algorithm \(^{12}\) is an improvement on the A* algorithm. It can continually replan the path using the A* algorithm in a dynamic environment. The evaluation function, \( f(n) \), of the A* algorithm \(^{13}\) is given by the following expression:

\[
f(n) = g(n) + h(n)
\]

(4)

where \( g(n) \) is the real cost for the node \( n \) to move from its initial position to the target position, and \( h(n) \) is the predicted cost for the node \( n \) to move from its initial position to the target position along the optimum path. We plan the global optimum path using the grid method, and the cost between two nodes in a grid map is usually measured by their Euclidean distance. Thus, \( h(n) \) can be represented as

\[
h(n) = \sqrt{(G.x - n.x)^2 + (G.y - n.y)^2}
\]

(5)

The global optimum path derived from the A* algorithm reflects the trend and is treated as a reference for local navigation. In other words, the local navigation is adjusted to the global optimum path. The steps of the A* algorithm are as follows:

Step 1: Build OPEN and CLOSE lists. Initial them with NULL and insert the initial position \( R \) of the robot into the OPEN list.

Step 2: If OPEN = NULL, go to Step 5. If not, pick out the least node \( N \) in \( f(n) \) and go to Step 4 if \( N = G \), or else extend node \( N \) to eight-connected grids and put \( N \) into CLOSE list.

Step 3: Update the OPEN and CLOSE lists according to the values of \( h(n) \) and \( f(n) \) of the eight-connected grids. Go to Step 2.

Step 4: Find the target position \( G \) and return the available path, or else go to Step 5.

Step 5: Fail to accomplish planning and return no available path.

3.3 Collision-avoidance during Local Navigation

A mobile robot continually measures the distance \( L \) between a dynamic obstacle and itself when moving along the global optimum path derived from the A* algorithm and predicts the trajectory of the dynamic obstacle when \( L \) is not bigger than \( M \). In this study, the robot measures the motion angle \( \theta \) according to the positions of the obstacle in two contiguous planning cycles and defines an influential area that extends \( \pm \frac{\pi}{4} \) rad around the motion angle. In this area, the motion of the robot is affected by the dynamic obstacle, and the effect of the dynamic obstacle on the mobile robot is measured by the Effect Factor (EF). The EF decreases in steps along the predicted moving direction \( \Phi \) of the robot to reflect the trend of the impact caused by the dynamic obstacle on the robot. The initial value and the step-length of the EF are determined by the availability and the optimum trajectory of the path. If there is no position change of the obstacle in two contiguous planning cycles, \( \theta \) does not exist; thus, EF only includes the initial value. Here, \( \Phi \) can be given by the expression
The prediction results and the immediate value of $L$ are the parameters needed for the local navigation algorithm if local navigation is required. If so, we use rolling-path planning [14] to build a local rolling window for the robot’s current position. This rolling window intersects the known global optimum path at a subtarget position. The subtarget position here is the final target in the current rolling window. Let the initial value of the subtarget grid be one; to build a dynamic-obstacle map, obtain the value of any other grid from the enlightening function $g(x, y) = dx + dy + 1$ and initialize the value of the dynamic obstacle’s grid, as shown in Figures 1 and 2. Next, plan the path using collision-avoidance between the robot’s current position and the subtarget position in accordance with the values of the grids in the local rolling window.

Local navigation in this study is based on the artificial potential-field approach [15], and the result of this navigation is improved by dynamic prediction to achieve better performance of real-time control in a dynamic environment. Because of the existence of the global optimum path, the drawback of the artificial potential-field approach—which may result in the robot never reaching the target when an obstacle is near the target—is overcome. The steps of local navigation are as shown below:

**Step 1:** Robot moves along known path $S$ and continually detects the distance $L$ from the dynamic obstacle to itself. Go to Step 2 when $L \leq M$.

**Step 2:** Predict the next position of the dynamic obstacle. Go to Step 2 if (a) it will affect Robot’s current optimum path or (b) current distance $L$ is of a bigger value than $L’$ in the former.

**Step 3:** Create a local rolling window for the robot and make this window intersect with the known path $S$ at subtarget. Set up the initial window map and name it Initial; set up the obstacle map and name it Obstacle. Add them together to form the local environment map and name it Map.

**Step 4:** Set the current grid position of robot as the reference and find the subtarget along the minimum grid value of this reference’s eight-connected grids the get the local path $P$. Ensure that the total value is the minimum when more than one minimum grid value exists in a group of eight-connected grids.

**Step 5:** Adjust the path $S$ with the local path $P$. Go to Step 1.

4. Simulations and Analysis

We assume that the robot moves in a two-dimensional limited area and that there are a certain number of static obstacles and one dynamic obstacle. Let the dynamic obstacle move from $O_s$ to $O_g$.
at uniform velocity along the yellow trace, as shown in Figures 3 and 4, where $R$ denotes the starting position of the robot and $G$ denotes the target position. We assume that the length of each single step of the robot and the dynamic obstacle equals one length of a grid. The result of global path planning based on the prediction of a dynamic obstacle is shown in Figure 3. The simulation result of the D* algorithm is shown in Figure 4. The global path derived from the A* algorithm is shown in Figure 5, where R denotes the starting position of the robot and G denotes the target position.

**Figure 3.** Path Based on Obstacle Prediction

**Figure 4.** Result of D* Algorithm

**Figure 5.** Result of A* Algorithm
Our algorithm is based on the global optimum path achieved from the A* algorithm and improves the result through prediction. This improvement is accomplished by setting up an optimum threshold $M$. The local collision-avoidance process is shown in Figure 6, where Figure 6(a), Figure 6(b), Figure 6(c) and Figure 6(d) are pictures of four timings during this process.

**Figure 6. Local Navigation and Collision-avoidance**

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Length of the Path</th>
<th>Times of Global Path Planning</th>
<th>Time Consuming (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>D* Algorithm</td>
<td>78.6</td>
<td>3</td>
<td>56.07</td>
</tr>
<tr>
<td>Algorithm in this Study Medium</td>
<td>78</td>
<td>1</td>
<td>31.015</td>
</tr>
</tbody>
</table>

The goals, namely, global path planning and local collision-avoidance, are achieved in the experiment. The performance of real-time control is improved through the local adjustments to the global optimum path. The building of local rolling windows takes into account the shifts in the distance between the robot and the dynamic obstacle. Hence, the adaptability of local navigation is improved. Compared with the D* Algorithm, the building of local rolling windows works better by rapidly repeating local navigation during the entire path-planning process. Thus, it reduces the number of repetitions of global path planning during the entire path-planning process and improves the efficiency of the same. The algorithm proposed in this study makes use of the known information about dynamic obstacles reasonably to combine the prediction of dynamic obstacles with the building of rolling windows to improve the performance of real-time control during global path planning by local collision-avoidance. At the same time, adjusting the obtained global path through local navigation based on rolling windows yields a better path as shown in Table 1.

**5. Conclusions**

Dynamic path planning not only has to meet the requirements of real-time control of performance but also should examine the adaptability and ensure the optimum result. In this study, we present an algorithm of dynamic path planning based on position prediction of a dynamic obstacle. This
algorithm combines the merits of the optimum path obtained from global path planning in a static environment with the efficiency of local navigation under a dynamic environment by adjustment of the path derived from global path planning using the result obtained from the rolling window during local navigation. The result of the experiment shows that this algorithm can yield an efficient path that avoids the dynamic obstacle.

6. Acknowledgement

This work is supported by the National Natural Science Foundation of P.R. China (Grant No. 61173163), Program for New Century Excellent Talents in University (Grant No. NCET-09-0251).

7. Reference