Robotic Wheeled Vehicle Ripple Tentacles Motion Planning Method

Hongxiao Yu, Jianwei Gong, Karl Iagnemma

Abstract—This paper describes a nonholonomic robotic wheeled vehicle ripple tentacle motion planning method, aiming to improve the vehicle's trajectory smoothness and avoid frequent weight parameters adjustment in different environments. In the regular tentacle motion planning algorithm, the planning result is selected among the drivable tentacles using a weighted sum cost function. Though the method is simple and easy to understand, it is difficult to adjust the weighted coefficients in different environments. To solve this problem, a geometrical ripple tentacles technique is used to choose a tentacle as a sub-optimal path. Compared with the regular tentacles algorithm, the proposed ripple tentacle algorithm can get a better performance in vehicle's trajectory smoothness with an acceptable runtime expense. And another two traits can also distinguish this method: (a) it can avoid weight parameter adjustment in different environments and varied vehicle's states, and (b) it can be used in both unknown environment and partly known environment with goal point and global reference path. In the totally unknown environment, it acts as a pure obstacle avoidance algorithm, and when there is a global path, it can follow the reference path and avoid hazards simultaneously.

I. INTRODUCTION

There many factors that can affect the mobile robot motion planning performance and the most important are: (1) the limited sensing capabilities, such as adaptabilities to varied distance measuring, uncertain and time-varying dynamic environments, and (2) the complicated control model of robotic vehicles, for example, the temporal and logical constraints of vehicle behavior, and varied control parameters. And most motion planning methods have to adjust their parameters according to these environments and vehicle states differences. But it is difficult to get the optimal or the best parameters. As a result, the mobile robot's trajectory becomes unsmooth, and in worse cases, the maneuvers of the robot become unstable.

Early motion planning research mainly focused on industrial robot systems, and later extended to the robotic vehicles[1]. There are many methods and algorithms for mobile robot motion planning, and Laumond gave a good historical perspective[2].

This paper discusses several recently developed and widely used algorithms. Rapid-Exploring Random Trees algorithm (RRTs) was proposed for nonholonomic motion planning[3]. And Bi-RRTs, an improved version of RRTs algorithm, had a higher planning efficiency[6]. CMU proposed Anytime Dynamic A*(AD*) in the lattice space for path planning, and the pure pursuit algorithm was used to track the reference path. However, AD*'s computational speed is limited especially in high dimensional space[4][5]. Another impressive planning algorithm is the tentacles planning method[7].

In our application with the methods and algorithms listed above and some other existed algorithms, we found several aspects that needed to be improved.

First, the adaptability of the algorithms remained as a challenge. It was very hard to adjust the parameters or coefficients to adapt some algorithms to different applications and plants. For instance, when we tried to apply tentacle planning algorithm in our unmanned ground vehicle for continuous obstacle avoidance, it took us a lot of time to adjust the cost function coefficients.

Second, the specific trajectory obtained from the planner was not always smooth. Sometimes there were some discontinuous changes or a sharp change between the present and the previous result, especially when the vehicle's states such as velocity changed. As a result, the maneuver of the vehicle became unstable.

Third, there are few path planning methods which can satisfy both path following and obstacle avoidance. For example, the VPH+ algorithm was mainly focused on obstacle avoidance [8]. For a practical application, you must combine several algorithms together to build a complete motion planning system.

In this work, we are not going to overcome all these problems of all the algorithms. According to our own needs, we proposed a ripple tentacle selection techniques to improve trajectory smoothness and avoid frequent weight parameters or coefficients adjustment in the application of regular tentacle motion planning algorithm[7]. Though the method is more complicated and a little more time-consuming than the regular methods, the robotic vehicle's trajectory smoothness is highly improved, and moreover, we do not have to adjust the weight coefficients often. Also, because the vehicle’s maneuver is always stable, the time consumed on controlling in all will be relatively reduced.

The proposed ripple tentacles algorithm is qualified for both tentacles selection and global reference path tracking. It can produce a new attractive ripple tentacle to track reference path. Compared with the regular tentacle algorithm, the ripple...
tentacles algorithm takes obstacle avoidance as its primary mission, and has an improvement that the tentacle selection influence factors are independent of each other. As a result, the uncertainty in terms of the weight coefficients of the tentacle selection will be reduced. Also, our test showed that it could meet the requirements for both motion planning and obstacle-avoidance or path following.

II. TESTBED INFORMATION AND THE STRUCTURE OF THE ALGORITHM DESCRIPTION

A. Environment information

For the convenience of the algorithm’s description and simulation, we create a 2D occupancy grid space with 100x125 cells, and the size of each cell is 20cm x 20cm. So the range of our vehicle’s known surrounding environment is a 20m x 25m rectangular area, which is nearly identical to the perception range of sensors mounted on our real test robotic vehicle. The distance between the vehicle’s head and the front end of the grid space is 15m, the rear end of the grid space, 10m. The center of the vehicle is 10m away from each sides of the grid space. The motion planner receives the road features and obstacle information from laser scanners and other sensors mounted on the vehicle. Each cell stores a single binary value which expresses the state of obstacle. If the cell value is 1, the cell is filled with an obstacle, and 0, with no obstacle.

B. Vehicle’s sensor system

The environmental perception system consists of three UTM-30LX laser scanners mounted in front, left and right side of the vehicle. And the vehicle state estimation (position, velocity and heading) is provided by a differential GPS/INS unit. The map update frequency is 100Hz. The grid map is generated using the fusion result of the three laser scanners. And the vehicle kinematics constraints are taken into account in the hazard forecast estimation.

C. Vehicle kinematic model

A bicycle vehicle model[9] is used to estimate the vehicle's states online.

\[
\begin{bmatrix}
\dot{x} \\
\dot{y} \\
\dot{\theta}
\end{bmatrix} = 
\begin{bmatrix}
\cos\theta & \sin\theta & 0 \\
\sin\theta & \cos\theta & 0 \\
\tan\delta & 0 & 1
\end{bmatrix} \begin{bmatrix}
v \\
\delta \\
\delta
\end{bmatrix} + 
\begin{bmatrix}
0 \\
0 \\
1
\end{bmatrix}
\]

(1)

where \(v\) and \(\delta\) are the vehicle’s longitudinal velocity and the angular velocity of the steering wheel, respectively. \((x, y)\) is the position coordinates of the vehicle, \(\theta\) is heading angle, and \(\delta\) is the steering angle.

D. Structure of the proposed algorithm description

The proposed method is eligible for both totally unknown and partly known environments. In Section III, ripple tentacle selection algorithm is used to avoid obstacle in unknown environment. In Section IV, the ripple tentacle selection algorithm is improved as an ellipse hypotrochoid ripple tentacle to track the reference global path. And Section V are experimental results to verify the method.

III. RIPPLE TENTACLE SELECTION METHOD

In unknown environment, it is assumed that there is no goal and no reference path for the robot vehicle. First, tentacles are generated using the same method in the regular tentacle algorithm[7]. Then, a new technique is proposed to select the best drivable tentacle during the motion planning for the robotic vehicle.

A. Generate Tentacles

Tentacles generation method is here the same as the regular tentacle method[7]. A specific number (here, 51) of tentacles with one degree resolution are generated. The number of the tentacles can be adjusted according to the sensing and computational ability. In order to improve the calculation speed, the tentacle templates were generated off-line[7]. The support and classification area of each tentacle picture is generated respectively before the planning, which can be “AND” the obstacle map picture to find the drivable tentacle. And the size of the map and the number and the size of the grids are also pre-generated and stored in a chart.

B. Ripple Tentacle Selection Method

A new technique is proposed to select the best drivable tentacle during local planning. The method can avoid adjusting weight coefficients in different driving unknown environments.

Tentacle ripple algorithm is inspired by ripples produced when a stone is thrown into the water. From the center point where the stone is thrown into there will be many concentric circular ripples spreading around in all directions. When an obstacle appears on the ripple, the ripple will disappear and stop spreading. But the other parts of ripples are still spreading in their original direction.

The vehicle's rear axis center is regarded as the circle center to generate the concentric circles like ripples. At each time step, the ripple radius will be increased with a value \(\Delta r\), here we specify \(\Delta r = 0.5m\). Every tentacle in the drivable area will be expanded with the ripple circles in turn, and let the initial radius of the ripple be the crash distance \(l_i[7]\). The ripple along the \(i\)th tentacle will keep on spreading until it clashes an obstacle or obstacles, and mark the current ripple radius as \(r_{stop}^{(i)}\) (stop radius), if there is no obstacle, then let the biggest radius be \(r_{stop}\). All the stop radius of the drivable tentacles can be stored in a set \(R_{stop}\).

The tentacle with the largest stop radius in the corresponding area is regarded as the best tentacle. If different tentacles have the largest stop radius, the tentacle with the smallest change from its former one will be chosen. In this
way we can have a smoother path and make the vehicle move stably.

Also, in order to avoid a sharp change between the current selection path and the previous one, we use a Kalman filter [14] to predict the next tentacle. This can make the vehicle move smoothly.

Then we take the width and length of the vehicle, and the minimum turning radius into account as the constraints in trajectory following control. Fig.1 illustrates the tentacle selection process.

In order to make the vehicle's trajectory smoother, the desired tentacle selection range would be limited by the current steering angle. We specify the current steering angle \( \delta \) as the center of the inner planning searching steering angle range \( \Delta \delta \), with a certain angle at both left and right side respectively, \( [\delta - \Delta \delta / 2, \delta + \Delta \delta / 2] \); here \( \Delta \delta \) is set to be 6 degrees. The steering angle range can be considered as the biggest angle range for the steering wheel to turn in one control iteration.

If a drivable tentacle cannot be found in \( [\delta - \Delta \delta / 2, \delta + \Delta \delta / 2] \), the searching criteria will be expanded to all the steering angle range. In this process, different tentacle corresponds to different vehicle velocity sets [7].

Fig.2 shows the result of an example result for the sub-optimal tentacle selection. In this scenario, the NO.2 tentacle is the current steering angle, and NO. 1 is selected as an expected tentacle. We can see that the ripple selection technique is well suited for unknown environment motion planning and control.

IV. ELLIPSE HYPOTROCHOID RIPPLE TENTACLE AND GLOBAL REFERENCE PATH TRACK CONTROL

In mobile robot planning, there is a goal point and global reference path in most cases, and this is a partly known environment for the robot. In this case, the ripple tentacle is improved through use of an ellipse hypotrochoid ripple tentacle method to track the global reference path. In this section, a global reference path is generated using Bi-RRTs. Then, ellipse hypotrochoid ripple tentacle method is used to track the reference path.

A. Generate Tentacles

The global planner is based on the bidirectional RRTs method[10][11], by growing two trees from the initial and goal configuration simultaneously. The kinematic bicycle model described in (1) is used in Bi-RRTs algorithm.

Fig.3 is an example result of Bi-RRT global planning. In the practical applications, we should adopt a kind of anytime RRTs [12]. But in this paper, the main objective is to evaluate the newly developed ripple tentacles selection method in a small map, so a global path generated in advance by Bi-RRTs is used as a path during the whole process. In Fig.3, the rectangle indicates the sensing area of the robotic vehicle.
ellipse growth should be determined. There might be a lot of intersections between the reference path and the selected drivable tentacle, and we can draw the lines from these intersections to the center of the vehicle’s X axis as the major axis of a series ellipses (Fig. 4).

\[
\begin{align*}
  x &= (a - b) \cos t + h \cos \left( \frac{a - b}{b} t \right) \\
  y &= (a - b) \sin t - h \sin \left( \frac{a - b}{b} t \right)
\end{align*}
\]

where \(a\) and \(b\) are respectively a fixed circle radius and rolling circle radius, and \(h\) is the distance from the center of the interior circle and \(t\) is the angle formed by the horizontal and the center of the rolling circle. When \(a = 2b\), the hypotrochoid is an ellipse. Fig. 5 shows the ellipse hypotrochoid which equations can be described as:

\[
\begin{align*}
  x &= (b + h) \cos t \\
  y &= (b - h) \sin t
\end{align*}
\]

We want to get ripple ellipses in the same shape. The ellipse parameters become \((b + h), t = t(b + h), (b - h), t = t(b - h), \quad (t > 1)\). Keep the ellipse shape and increase ellipse size. But the best obstacle avoidance tentacle may be far away from the vehicle. If the ellipse is near the vehicle, the ellipse size could be expanded fast, and vice versa.

This method could improve the speed of finding the best ripple ellipse tentacle. When the tentacle’s longest spread distance is far from the vehicle, the ellipse can be described by the following equation:

\[
\begin{align*}
  \frac{x^2}{(\log_2 \varphi(b + h), i)^2} + \frac{y^2}{(\log_2 \varphi(b - h), i)^2} &= 1 \\
  i = 2 \ldots \varphi > 2 \quad h \neq b
\end{align*}
\]

Here, the initial state of ellipse is defined as \(a = l_c\), where \(l_c\) is the crash distance, \(b = l_c / 2\). For convenience, we define the parameter \(h = \beta b\) \((0 < \beta < 1)\). As a result, the shape and size of ellipse can be determined by one parameter \(b\) only. Choose the stop major axis of the ellipse as the method of ripple tentacle to find the suitable tentacle which can track reference path and avoid obstacle. Fig. 4 also shows the result of ellipse hypotrochoid ripple tentacle. The determined tentacle sign is \(T_{num1}\). The above is one of the bases of tentacle vote selection.

Another vote selection is a classical tracking method works also like the pure pursuit control law \([15]\). First, we scale the look-ahead distance with the longitudinal velocity \(V_s\) of the vehicle, then scale the look-ahead distance. \(L\) is the wheelbase of our vehicle. The initial look-ahead distance value is the crash distance \(l_c\) \([7]\), and \(l_c = kV_s\). So the control law can be given as:

\[
\delta = \tan^{-1} \left( \frac{2L \sin(\alpha)}{kV_s} \right)
\]

If we get the \(\delta\) value, the tentacle number should be confirmed \(T_{num2}\). This is another vote rule that determines the sub-optimal path of vehicle. The tentacle performance index is formed as weight sum of the metrics considering \(T_{num}\) and \(T_{num2}\). The final tentacle \(\varphi\) can be chosen as:

\[
\varphi = \left[ p_0 T_{num1} + p_1 T_{num2} \right] 0 < p_0, p_1 < 1
\]

In our application \(p_0 = 0.4\) and \(p_1 = 0.6\). The tentacle final number is \(\varphi\).

V. EXPERIMENTS AND SIMULATION

Two experiments were conducted to analyze the tentacle selection and path following performance of the proposed algorithm. All the experiments were run on the Inter P8400 3.25G with 2G RAM under Microsoft Visual Studio 2008 (C++) environment, and based on BIT unmanned vehicles platform (Fig. 6).

\[
\text{Fig.6 The BIT unmanned vehicle platform}
\]

A. Experiment of choosing sub-optimal tentacle with different weight coefficients

In this first experiment, the vehicle was driving in an unknown environment with some simple obstacles placed in an S-bend. The width of the path is 4m; the radius of the S-bend is 9m. And Fig.7 and Fig.8 show the results of the regular tentacle method and the proposed ripple tentacle.
method driving in unknown environments respectively. In Fig. 7, NO.1 was the tentacle selected by the regular tentacle method [7] with the two different weight coefficients $w_{\text{clearance}}, w_{\text{flatness}}$ in same obstacle avoidance scenario. In this case, we could see that the result of different weight coefficients in the same frame were totally different. As a result, it is hard for us to find the optimal path in practical applications. Furthermore, two different factors will affect each other. So if there is an unreasonable result, it will also be hard to locate the cause of the problem.

![Fig. 7 Contrast the results of the results of the regular tentacle method with different weight coefficients in the same frame](image)

Fig.8 (a) showed that the ripple tentacle method selected the 10th tentacle as the best path, and Fig. 8(b) showed that the regular method with the weight coefficients $w_{\text{clearance}}=0.5$ and $w_{\text{flatness}}=0.5$ chose the 9th tentacle as the best one. The 10th tentacle can spread farther than 9th tentacle and the last frame tentacle number is 13th. In this case, the ripple tentacle method could get a satisfying performance and get much smoother tentacle without adjust the weight coefficients in different environments.

![Fig. 8 Contrast different methods of tentacle selection in the same frame](image)

Fig. 9 showed that the chosen number of the tentacle with different weight and ripple method. Since the tentacle number range was [0,50] which corresponds to the steering angle criteria [-25,25], steering angle of the ripple tentacle method did not change intensely, while the change range of the regular tentacle method was bigger. And the regular method oscillated immensely when the value of flatness weight ($w_{\text{flatness}}$) was smaller. What’s more, when the value of flatness became larger, the value of clearness weight would be decreased, as a result, the result path might not be suitable for obstacle avoidance. But the experiments showed that the ripple method could satisfy both obstacle avoidance and sub-optimal path in various environments without readjust any other parameters.

![Fig. 9 Different weight and ripple method choosing tentacle number (image best viewed in color)](image)

Fig. 10 showed the runtime of the two methods. It took the regular tentacle method about 30ms to finish its calculation every time, while the ripple tentacle method, nearly 50ms. Although the ripple tentacle method took more time during planning computation, it could always get a reasonable and smoother path without adjust the weight coefficients.

![Fig 10. Runtime of three selection tentacle method.](image)

### B. Global reference path following experiment

In the second experiment, the ellipse hypotrochoid ripple was used to track the global reference path to verify the path following function which is eligible for both obstacle avoidance and global reference path following.

![Fig. 11 showed that the ripple circles were attacked to be elliptical by the reference path. As different tentacles and reference path have different intersection points, we will find the tentacles with the largest length of semi-major axis of the ellipse, like the ripple method finding the biggest stop radius. The number of tentacles can be selected. The look-ahead distance is very significant to determine the point of intersection. It is of great help in choosing the tentacle.](image)
Fig. 11 Hypotrochoid ripple tracking the reference path

Fig. 12 and 13 show the tracking results with the heading deviation of vehicle, PI/2 and PI/4. The experiment shows the ellipse hypotrochoid ripple have a good result of tracking the reference path with the same vehicle velocity.

![Tracking path](image1)

![Deviation](image2)

Fig. 12 Tracking result and deviation figure with the heading deviation PI/2.

![Tracking path](image3)

![Deviation](image4)

Fig. 13 Tracking result and deviation figure with the heading deviation PI/4.

VI. CONCLUSIONS

This paper mainly describes the improvement of the tentacle motion planning method, using ripple method to selection the path among the drivable tentacles. This can avoid weight coefficients adjustment in different environments; also the tracking path of the robotic vehicle is smoother than the regular method. Although the runtime of the proposed method is a little bit more than the regular one, and the design is more complicated, it is acceptable with the improved path smoothness and the algorithm’s satisfying adaptability. Moreover, the proposed method is both eligible for unknown and partly/all known environment mobile robot motion planning and control.

Reference