Proceeding of the 2015 IEEE
International Conference on Information and Automation
Lijiang, China, August 2015

Trajectory Planning for RoboCup MSL Mobile Robots Based on Bézier Curve and Voronoi Diagram

Yang Xianglin, Zeng Zhiwen, Xiao Junhao and Zheng Zhiqiang

College of Mechatronic Engineering and Automation

National University of Defense Technology

Changsha P. R. China

Yang_xianglin@163.com

Abstract: **In RoboCup middle size league (MSL), the basic purpose for the soccer robots is to realize obstacle avoidance and shooting without stop by utilizing trajectory planning to provide a smooth trajectory. Consider the highly competitive and highly dynamic match environment, in fact, the trajectory planning is very challenge. The implementation for mobile robots generating trajectory is complex in a highly competitive and dynamic environment. In this paper, we propose a new method based on the Dijkstra shortest path algorithm, Bézier curve and Voronoi diagram to create geometric paths. Since the acceleration and velocity limitation should be taken into account, we then transform the geometric paths into a smooth trajectory with specified acceleration and velocity. Finally, simulation results are provided to illustrate the efficiency of the method.**

Index Terms **- RoboCup MSL, Trajectory planning, Bézier curve, Voronoi diagram**

I. INTRODUCTION

Trajectory planning is a fundamental motion plan and control problem for mobile robots in dynamic environments [1]. Generating a collision-free and smooth path is a key issue in many mobile robots, such as space robots [2], underwater robots [3] and soccer robots [4]. The algorithm presented in this paper is developed for our soccer robots participated in the RoboCup middle size league (MSL). The method can also be applied to other setups since this application is quite considered the following features [5]. Firstly, the robots always move in high speed and the maximum speed can reach to 4m/s in the competition; secondly, sudden start and stop is quite common for the robots for avoiding obstacles; lastly, the opposite robots can obstruct our robots from reaching the goal target intelligently. Therefore the environment is highly dynamic which requires the path generation and trajectory tracking algorithm to be computationally efficient, real-time and easy to be implemented in on-board processors.

Many efforts have been conducted to solve the obstacle avoidance problem. Khatib presented a unique real-time obstacle avoidance approach for manipulators and mobile robots based on the artificial potential field concept (APF) [6]. The main idea is to generate attraction and repulsion forces, within the working environment of the robot, to guide it to the goal. Then the complex environment is likely to be indicated by brief mathematical expressions. The method presents efficiently to avoid static obstacles or low-speed motion obstacles. However, avoiding high-speed motion obstacles applied the way tends to be invalid, and the generated path does not seem to be smooth.

The Vector Field Histogram method (VFH) is widely adopted in the trajectory planning [7]. A two-dimensional Cartesian histogram grid is used as a world model by VFH. This world model is updated continuously with range data sampled by on-board sensors and VFH subsequently employs a two-stage data reduction process in order to generate the desired control commands for the vehicle. However, the VFH method cannot deal with situations where the environment is complex and highly dynamic.

Bruijnen proposed an algorithm called subtargets which is computationally efficient and generates a sub-optimal smooth path with bounds on the allowed velocity, acceleration and jerk [8]. The paths generated using the subtargets are smooth and collision-free. Most importantly, the method is able to adapt rapidly in a changing environment. However, there still exist some shortcuts in the subtargets algorithm. Firstly, the velocity and acceleration of the robot can only be limited within their bounds rather than be set to desired profiles. Secondly, the robot may oscillate around obstacles when surrounded by them or a wrong subtargets point may be calculated which is very dangerous for the robot. Cheng proposes an algorithm combining the subtargets method with the Cubic B-spline curve to deal with the above problems [9]. In that case, the oscillation and wrong subtargets point problem can be easily solved. In addition, the motion laws can be taken into account to transfer geometric paths to trajectories, so the velocity and acceleration of the robot can be set arbitrarily within the physical limitations. It outperforms potential field algorithms with respect to global convergence to the target. However, convergence is not guaranteed for arbitrary situations.

All these approaches are efficient in generating collision-free paths and trajectories in static environments or common dynamic environments. But they may not work when the robots and obstacles are moving at high-speed. In addition, most of the methods aim at generating a direction for the robot to head in, which makes it hard to guarantee that the paths are smooth. In this work, based on the Dijkstra shortest path algorithm, Bézier curve and Voronoi diagram is used to create geometric paths. Then we can get the reference acceleration and velocity commands by transforming the geometric path to trajectory. Finally, simulation results are drawn to validate the efficiency of the proposed method.

978-1-4673-9104-7/15/$31.00 ©2015 IEEE

This paper is organized as follows. In Section II, the Bézier curve and Voronoi diagram is introduced. In Section III, generating a collision-free and smooth path based on Bézier curve and Voronoi diagram is given. Section IV introduces the transformation from geometric path to trajectory. The simulation experiments and results are detailed in Section V. Section VI concludes this paper.

II. THE BÉZIER CURVE AND VORONO DIAGRAM

A. *The definition and properties of Bézier curve*

In a given space R^3 $B:[0\ \ 1]\to R^3$, the $n+1$ position vectors are $C_i(i=0,1,2\cdots n)$. Then the Bézier curve can be defined as

$$p(t)=\sum_{i=0}^{n}B_{i,n}(t)p_i,\quad 0\le t\le 1 \tag{1}$$

Where the coefficients p_i are the control points which consist of the characteristic polygon of Bézier curve, and the basis functions $B_{i,n}(t)$ are n-th degree Bernstein polynomials defined by

$$B_{i,n}(t)=\frac{n!}{i!(n-i)!}t^i(1-t)^{n-i} \tag{2}$$

As shown in Fig. 1, Two examples are presented.

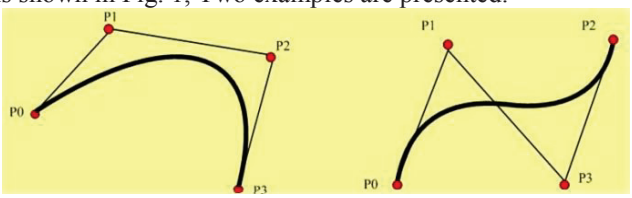

Fig. 1. Examples for third-order Bézier curve.

The properties of Bézier curve are:

- Endpoints: according to the definition of Bézier curve, $p(0)$ and $p(1)$ are separately the origin and terminal of the characteristic polygon;
- Tangent vector : the first derivative of Bézier curve is
  $$\dot{p}(0)=n(p_1-p_0),\qquad \dot{p}(1)=n(p_n-p_{n-1}) \tag{3}$$
  It means that tangent direction of the origin and terminal fit the first and last side's trend;
- Convex hull: in [0, 1], the curve $p(t)$ is completely contained in the convex hull formed by its control points;
- Geometrical invariability: the geometric characteristic of Bézier curve is not related with the variation of coordinate.

B. *The definition and properties of Voronoi diagram*

Voronoi diagram is the segmentation of plane based on a given point set. Given a set of n elements on a plane, the plane is segmented by the follow equation:

$$V(p_i)=\bigcap_{j\ne i}\{p\mid d(p,p_i)<d(p,p_j)\}\quad (j=1,\ 2,\ldots n) \tag{4}$$

Then the segment is called as a Voronoi diagram with generators $p_i(i=1,2,\ldots,n)$, where $d(p,p_i)$ is the set of points that Euclid distance is minimum between p and p_i. A Voronoi diagram is illustrated in Fig. 2.

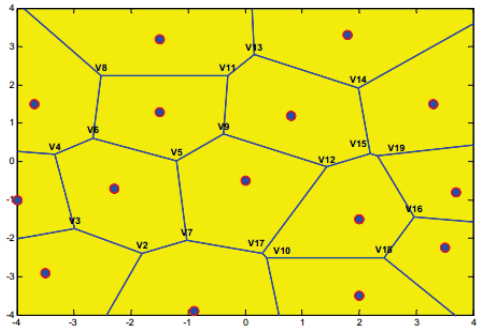

Fig. 2. An example for Voronoi diagram.

III. TRAJECTORY PLANNING BASED ON BÉZIER CURVE AND VORONOI DIAGRAM

With the global information given, Voronoi diagram based search is an efficient global planning algorithm. It is consist of environment modeling and path search. The Voronoi diagram property that each Voronoi polygon region is the set of closest points away from the generating element is significant. As a conclusion, when the robot is moving on the edge of Voronoi diagram, the distance between the robot and obstacles is maximal.

A smooth path can be fit based on the Bézier curve. As tangent vector property makes sure that the robot faces the goal when it reaches the shooting point by choosing proper control points. An example has been illustrated in Fig. 3.

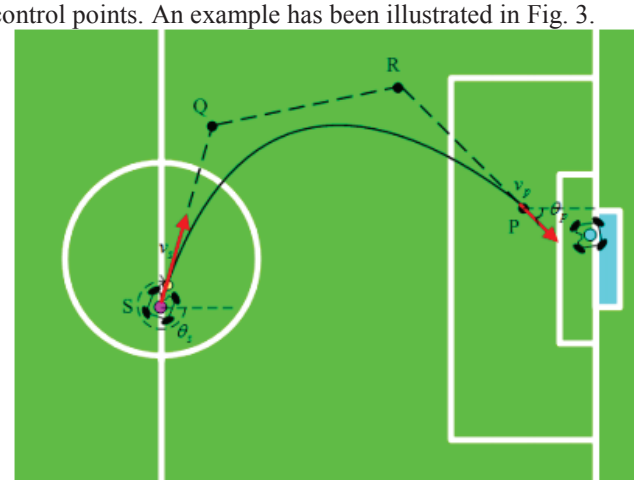

Fig. 3. An example for shooting path planning using Bézier curve.

In this paper, opponent robots are treated as dynamic obstacles, and it is assumed that the efficient intercept rang of the opponent robot is roundness in the time T_d when our robot moving from the initial position to the target point. However, the omni-vision system could not yet obtain the accuracy velocities of opponent robots. And the positions of opponent robots acquired from the omni-vision system is even mixed with noise in a dynamic environment. Assume the accessible domain of the opponent robot is roundness with

radius R_0 in time T_d, R_0 is proportional to T_d. And the generation path is not supposed to cross the roundness in order to avoid to be intercepted by opponent robots.

With the initial position S1 and target position E1 given, the path planning should make sure the robot faces the goal when it reaches the target, so that it can shoot directly. Here it is assumed the robot can obtain the global information of obstacles, and the efficient intercept roundness' radius $R_0 = 2$ m. Then path generation has the following four steps.

Step 1: Detect the positions of opponent robots $R_i(i = 1,2,3,4)$ through the omni-vision system, and take these as part of generation elements. The generation path should also make sure the distance large enough to avoid robots moving out of the field boundary. Combined with the feature points on the field $P_i(i = 1,2,\ldots 12)$, we can achieve the Voronoi diagram illustrated in Fig. 4.

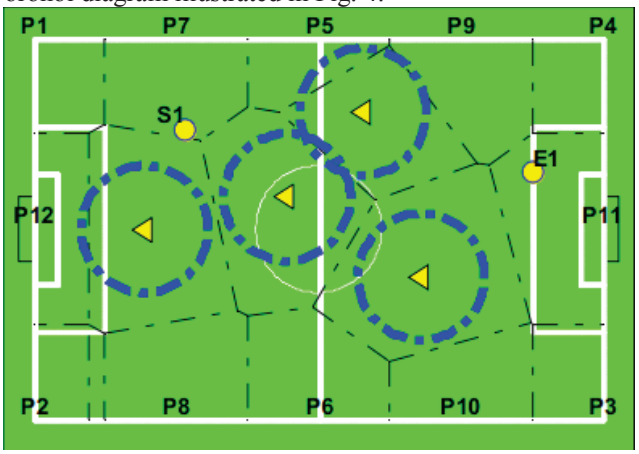

Fig. 4. The generated Voronoi diagrams based on terrain.

Step 2: Combine each vertex in the Voronoi diagrams with robot's initial position and endpoint position as a spares diagram, see Fig. 5. Note that, in the Voronoi diagram, the intercept areas of the opponent robots should not be crossed. Then we can figure out the shortest path in the Voronoi diagram according to the Dijkstra shortest path algorithm.

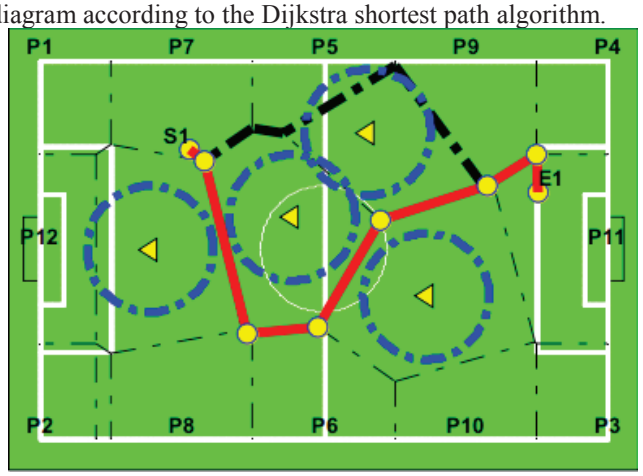

Fig. 5. An Example for finding shortest and efficient path using the Dijkstra algorithm.

As shown above, though the black path is shortest, it turns out to be inaccessible. Since it cross the opponent robots' intercept areas. The red path is then the shortest accessible path figured out through the Dijkstra algorithm.

Step 3: Take the nodes on the shortest path figured out through the Dijkstra algorithm as the control points of Bézier curve, illustrated in Fig. 6. Consider the efficiency of the algorithm, firstly, we will check the distance between adjacent points. If the distance is below threshold value (50 cm), then replace them with their midpoint as the new control point. And according to the property of tangent vector, in order to shoot on moving, it is supposed that the final side of characteristic polygon should point to the goal. Then we can figure out the intersection where the line between the goal and destination crosses the final side of Voronoi diagram edge. And take it as the new control point.

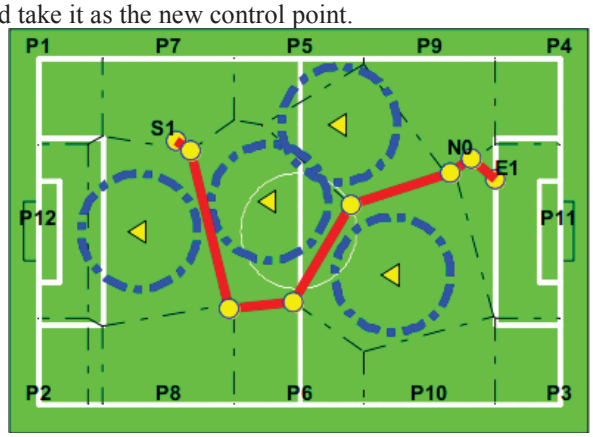

Fig. 6. Adjusting the control point so that the end side point to the goal.

Step 4: Fit the Bézier curve, and check the generation path whether it crosses the intercept areas of opponent robots, illustrated in Fig. 7 and Fig. 8. If the path is not travelable, we should modify the control points along the Voronoi diagram edge and increase new control point M_0. The new control point is decided by the corresponding convex polygon and taken as the midpoint of the corresponding Voronoi diagram edge. Through iteration, we can finally get a collisionless geometric path. And the final Bézier curve is described as

$$Bez(x(s), y(s)), s \in [0,1] \quad (5)$$

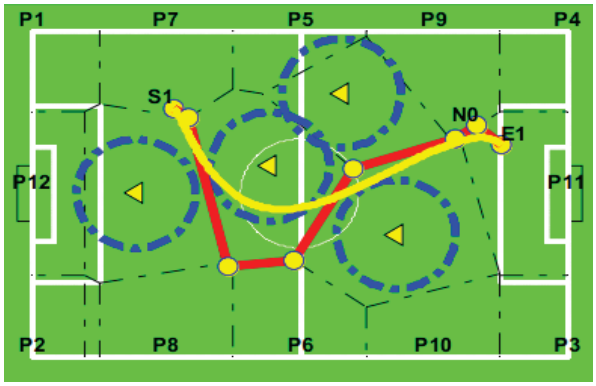

Fig. 7. The generated curve crossing the opponent robots' intercept area.

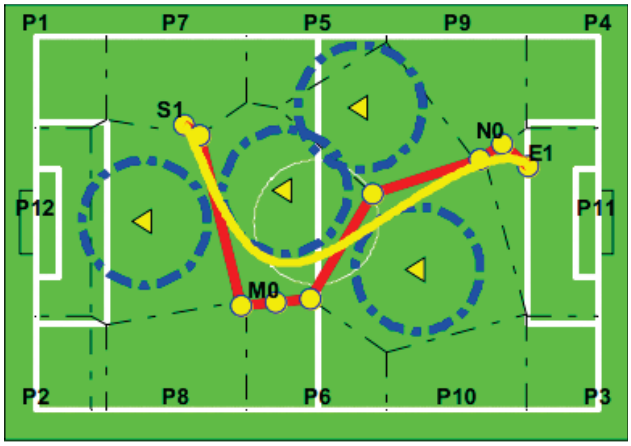

Fig. 8. Adjusting the path by adding control point.

IV. The Transformation from Geometric Path to Trajectory

From section III, we have obtained the geometric path $Bez(x(s), y(s)), s \in [0,1]$. Because acceleration and velocity should be taken into account, it is supposed to transfer the geometric path to trajectory. The method described in the paper [1] can be used to transfer the geometric path to trajectory.

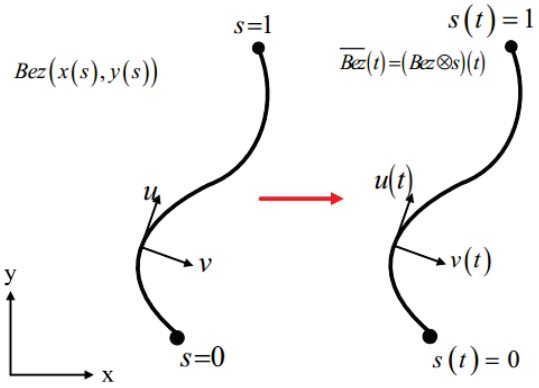

Fig. 9 The transformation from geometric path to trajectory.

As shown in Fig. 9, taking s as the function of time t:

$$s = s(t) \tag{6}$$

Define the corresponding trajectory:

$$\overline{Bez}(t) = (Bez \otimes s)(t) \tag{7}$$

Then its first-order derivative and second-order derivative respectively are:

$$\dot{\overline{Bez}}(t) = v(t) = \frac{dBez}{ds}\dot{s}(t) \tag{8}$$

$$\ddot{\overline{Bez}}(t) = a(t) = \frac{dBez}{ds}\ddot{s}(t) + \frac{d^2Bez}{ds^2}\dot{s}^2(t) \tag{9}$$

Then we can figure out first derivative and second derivative of $s(t)$:

$$\dot{s}(t) = \frac{v(t)}{\left|\frac{dBez}{du}\right|} \tag{10}$$

$$\ddot{s}(t) = \frac{a(t)}{\left|\frac{dBez}{du}\right|} - \frac{v^2(t)\left(\frac{dBez^T}{ds}\bullet\frac{d^2Bez}{ds^2}\right)}{\left|\frac{dBez}{du}\right|^4} \tag{11}$$

So the Taylor expansion of $s(t)$ is

$$s_{k+1} = s_k + T_s\dot{s}_k + \frac{T_s^2}{2}\ddot{s}_k + O\left(\frac{T_s^n}{n!}s_k^{(n)}\right) \tag{12}$$

where T_s is the sample time. Ignoring last item, merely consider the second-order system:

$$s_{k+1} = s_k + T_s\frac{v_k}{\left|\frac{dBez}{du}\right|} + \frac{T_s^2}{2}\left(\frac{a_k}{\left|\frac{dBez}{ds}\right|} - \frac{v_k^2\left(\frac{dBez^T}{ds}\bullet\frac{d^2Bez}{ds^2}\right)}{\left|\frac{dBez}{ds}\right|^4}\right) \tag{13}$$

Where $v_k = v(kT_S)$, $a_k = a(kT_S)$. According to the recursion formula, we can determine the reference point on $(k+1)T_S$.

V. Results

In order to test the designed global trajectory planning efficient, it is assumed that the global information of obstacles can be obtained. Then two stages simulation experiments are designed to test whether the designed trajectory can generate a geometric path to avoid the interception of opponent robots and whether the designed trajectory can obtain any settled acceleration and velocity.

Stage 1: Obtain a smooth obstacle avoidance geometric path. A standard field for RoboCup MSL scaled as a 18m×12m rectangle. The opponent robots position distribution is shown below. The initial position and destination are separately: 1, S1(-425,315), E1(680,180); 2, S1(-425,315), E1(680,-80); 3, S1(-530,-395), E1(680,180); 4, S1(-530,-395), E1(680,-80).

Stage 2: Transfer the geometric path to trajectory. Adopt the lemniscate's acceleration and velocity as the referent acceleration and velocity of Bézier curve, where maximum velocity is $v_{\max} = 300$ cm/s , maximum velocity is $a_{\max} = 324$ cm/s^2. Check whether the referent acceleration and velocity achieve the maximum. And check whether the referent acceleration and velocity follow the settlement to verify the efficiency of the transformation from geometric path to trajectory. In Fig. 10, some typical case for planning available geometric path are presented.

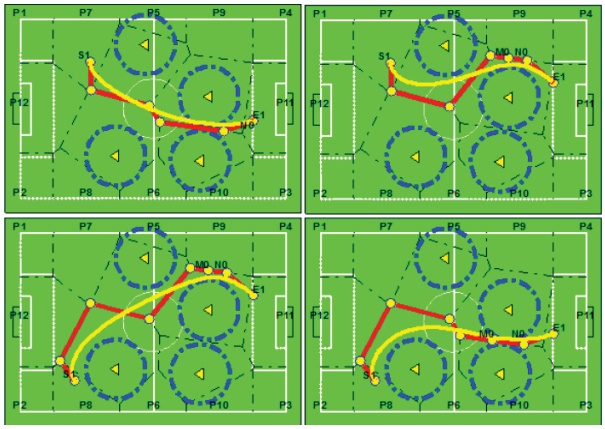

Fig. 10. Examples of some typical case for planning available geometric path.

As shown above, the planned geometric path is determined by the nodes on the shortest path of the Voronoi diagram and characteristic polygon consist of the origin and terminal point of robots. Tangential direction of the Bézier curve on the target point points to the goal. It means that the robot can shoot directly when arrived at the target point. Furthermore, the robot can totally avoid the interception of opponent robots along the path. So the trajectory planning is efficient.

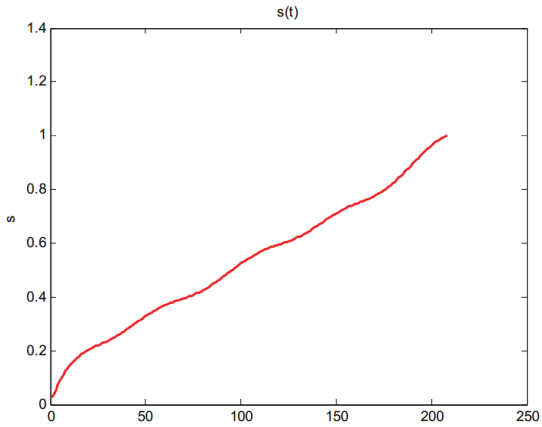

Fig. 11. The curve of adjusted control parameters against time.

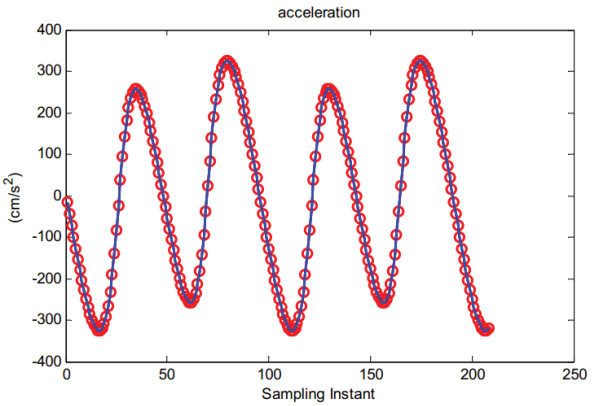

Fig. 12. The acceleration curve for actual trajectory against planned trajectory.

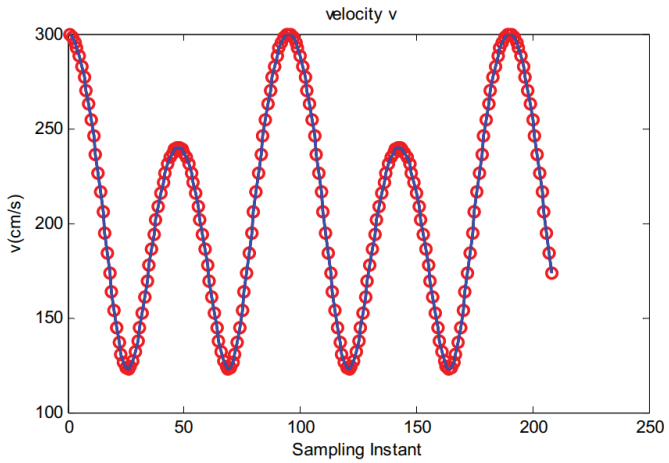

Fig. 13. The velocity curve for actual trajectory against planned trajectory.

Given the geometric path obtained in stage 1, adjust the parameters according the algorithm proposed in section IV. And we will achieve the trajectory $\overline{Bez}(t) = (Bez \otimes s)(t)$. Fig. 11. shows the mapping curve of origin parameters S against time t. Fig. 12. and Fig. 13. show that the planned trajectory have the some curves as the given acceleration and velocity. Therefore, the method can be used to transform any geometric path to trajectory according to the given acceleration and velocity. It allows a big freedom for the control of acceleration and velocity, which is meaningful for the mobile robots in the game. Since it can confuse and disturb the opponent robots. In consequence, our robots will have more chances to win. Furthermore, world model combining the information of trajectory consist of geometric path and velocity, is convenient for the cooperation between multi-robots.

VI. Conclusions

The results of section V proves the design of the trajectory planning to be effective. It can generate a smooth obstacle avoidance curve, and is available to define the trajectory according to settlement of reference acceleration and velocity. Since the tangential direction of trajectory points to goal at the end, it is convenient for the robot shooting directly when reaches the destination.

References

[1] L. Biagiotti and C. Melchiorri. Trajectory Planning for Automatic Machines and Robots. Springer Berlin Heidelberg, 2008.

[2] Mingming Wang, Jianjun Luo, Ulrich Walter. Mingming Wang, Jianjun Luo, Ulrich Walter[J]. Acta Astronautica, Volume 112, July–August 2015, Pages 77-88.

[3] Eichhorn M. A reactive obstacle avoidance system for an autonomous underwater vehicle[C].IFAC world congress. 2005: 3-8.

[4] Zeng Z, Lu H, Zheng Z. High-speed trajectory tracking based on model predictive control for omni-directional mobile robots[C]. Control and Decision Conference (CCDC), 2013 25th Chinese. IEEE, 2013: 3179-3184.

[5] Robin Soetens, Ren´e van de Molengraft and Bernardo Cunha. RoboCup MSL - History, Accomplishments, Current Status and Challenges Ahead. Springer International Publishing Switzerland, 2015.
[6] O.Khatib, Real-time obstacle avoidance for manipulators and mobile robots,in: Proc. IEEE Intl. Conference on Robotics and Automation, St. Louis, MO,March 1985, pp. 500-505.
[6] T.Tsubouchi, M.Rude, Motion planning for mobile robots in a time-varying environment, Journal of Robotics and Mechatronics 8 (1) (1996)15-24.
[7] J.Borenstein, Y.Koren, The vector field histogram-fast obstacle avoidance for mobile robots, IEEE Journal of Robotics and Automation 7 (3) (1991) 278-288.
[8] Bruijnen D, van Helvoort J, Van de Molengraft R. Realtime motion path generation using subtargets in a rapidly changing environment[J]. Robotics and Autonomous Systems, 2007, 55(6): 470-479.
[9] Cheng Shuai, Xiao Junhao and Lu Huimin. Real-time obstacle avoidance using subtargets and Cubic B-spline for mobile robots[C]. Information and Automation (ICIA). 2014, 7: 634 – 639.