Proceeding of the IEEE
International Conference on Information and Automation
Hailar, China, July 2014

# Real-time obstacle avoidance using subtargets and Cubic B-spline for mobile robots

Cheng Shuai, Xiao Junhao and Lu Huimin
College of Mechatronics and Automation
National University of Defense Technology
Chengshuai0528@hotmail.com

***Abstract*—A new method is proposed which combines the subtargets algorithm and B-spline curve to generate collision-free and smooth paths for mobile robots to deal with the obstacle avoidance problem in highly dynamic environments. Firstly the via-points are generated using the subtargets algorithm iteratively. Then the Cubic B-spline curve method is used to create geometric paths by interpolating via-points. Lastly the motion laws are taken into account to transfer geometric paths to trajectories which will be tracked by the robot. Simulation results prove that the method is more efficient in generating collision-free and smooth paths than the original subtargets algorithm. Furthermore, the velocity and acceleration can be set to any desired profiles when tracking trajectories.**

***Index Terms*—Obstacle Avoidance, Subtargets, Cubic *B-spline, mobile robots*.**

## I. INTRODUCTION

Obstacle avoidance is a fundamental motion planning and control problem for mobile robots in dynamic environments [1]. Generating collision-free and smooth path is a fundamental issue in many applications, such as Unmanned Aerial Vehicles [2], underwater robots [3] and soccer robots [4]. In this work, the method is developed for our soccer robots participated in the RoboCup middle size league. However, the method can be applied to other setups since this application is quite complicated considering the following features. 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. Typical obstacle avoidance methods work by generating a direction for the robot to head in, in Cartesian space or configuration space [5, 6]. The most famous one is the Vector Field Histogram method (VFH) [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 can't deal with situations where the environment is complex and highly dynamic. Khatib presented a unique real-time obstacle avoidance approach for manipulators and mobile robots based on the artificial potential field concept (APF) [8]. The main idea is to generate attraction and repulsion forces, within the working environment of the robot, to guide it to the goal. Simmons presented a method called Curvature Velocity Method (CVM) [9] for local obstacle avoidance for indoor mobile robots that formulates the problem as constrained optimization in velocity space. The Lane-Curvature Method (LCM) [10] combines CVM with the Lane Method, which divides the environment into lanes, and then chooses the best lane to follow to optimize travel along a desired heading. The beam curvature method (BCM) [11] combines a so-called beam method (BM), to improve the performance of CVM. BM calculates the best one-step heading which is used by CVM to calculate the optimal linear and angular velocities. Shi presented a local obstacle avoidance approach that combines the BCM with a prediction model of collision. Not only does the method inherit the quickness of BCM and the safety of LCM, but also the proposed prediction based BCM (PBCM) can be used to avoid moving obstacles in dynamic environments [12].

All these methods 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. To solve the problems mentioned above, 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 [13]. 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.

In this work, the subtargets algorithm is combined with the Cubic B-spline curve to deal with the above problems. The subtargets algorithm is used to generate via-points and the order of via-points has no influence in path generation, which means that the positions of the robot and the target can be exchanged when the robot is surrounded by obstacles. 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. The simulation experiment

978-1-4799-4100-1/14/$31.00 ©2014 IEEE

results show that our method is sufficient to deal with the problems mentioned above.

This paper is organized as follows: Section II explains the subtargets algorithm and the method to generate via-points that will be interpolated. In Section III, a collision-free and smooth path is generated using Cubic B-spline curve. Section IV introduces the trajectory tracking method. The simulation experiments are performed in Section V. Section VI concludes this paper.

## II. VIA-POINTS GENERATION USING SUBTARGETS

In our method, the generation of geometric path is based on the interpolation of via-points which are a set of two-dimensional data points on the world coordinate system. The coordinate calculation of via-points is of vital importance because it determinates whether the path is collision-free or not. In addition, the calculation algorithm must be computationally efficient for the purpose of real-time applications. Considering all these requirements, the subtargets algorithm is selected to generate via-points.

### A. Introduction of the subtargets algorithm

The subtargets algorithm aims at determining a subtarget that the robot can go to straightly. Generally four steps are necessary to determine a subtarget: *First obstructor, Grouping, Location of subtargets* and *Iteration procedure*. Fig. 1 shows an example of the determination procedure of subtargets.

*1) First obstructor*

As is shown in Fig. 1, $(a_i, b_i)$ is the coordinate of obstructor $i$ with respect to the robot in the direction of the target. Group $B$ is the set of obstructor $i$ which satisfies:

$$0 < a_i < \left|\vec{t} - \vec{o}_r\right| \bigcap \left|b_i\right| < r_i + r_r \qquad (1)$$

where all objects are modeled as circles. $r_i$ and $r_r$ are the radius of obstructor $i$ and the robot. The first obstructor is obstructor $i$ with the minimum $a_i$.

*2) Grouping*

Group $B$ is the set of obstructors that the robot would hit if it would move straightly to the target. Group $G$ is the set of obstructors that must be avoided containing the first obstructor. In group $G$ each obstructor holds the gap smaller than $2r_r$ with at least one other obstructor.

*3) Location of subtargets*

To compute the location of subtargets two steps are needed. The first step is the determination of the side (left or right) to pass group $G$. The side with the smallest absolute value of the perpendicular distance $b_i$ is selected. Then the angle $\alpha_i$ to pass each object in the group on the selected side is calculated and the obstructor with the largest angle determines the position of the subtarget ( $\vec{s}$ in Fig. 1).

*4) Iteration procedure*

The re-computation of subtarget will ensure the path between the robot and the subtarget is collision-free.

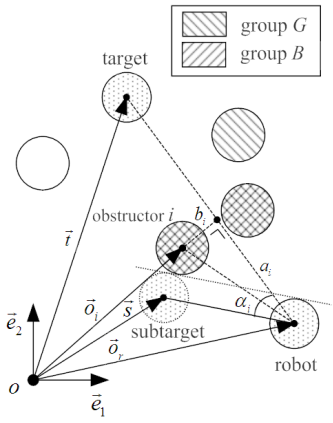


Fig. 1 Determination of subtarget

### B. Via-points generation

In subsection II-*A* the subtargets algorithm is briefly introduced and in this section we use it to generate via-points.

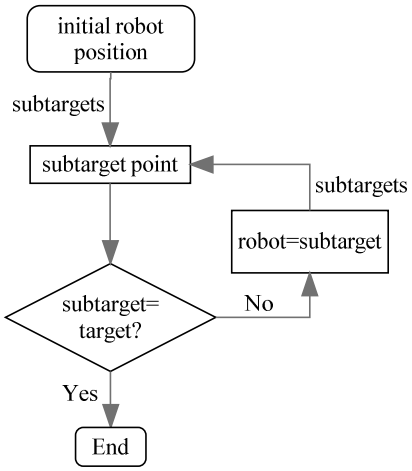


Fig. 2 Via-points generation algorithm

Fig. 2 is the basic procedure to generate via-points using the subtargets algorithm. At the beginning the subtarget is computed according to the initial position of the robot. Then the robot's position is set to be that of the subtarget acquired in the former step. New subtarget computation periods are carried out until the subtarget reaches the target. All subtarget points are recorded in the process and sorted as the via-points.

Obviously the positions of all the objects including the robot and the target on the ground determine the via-points. Furthermore, the via-points will remain unchanged even if the position of the robot and the target is exchanged. This feature is quite useful because the robot may be trapped by or may oscillate around obstructors in some certain cases and the exchange mentioned above can easily solve the problem. The related simulation results will be presented in Section V.

## III. PATH GENERATION USING CUBIC B-SPLINE CURVE

In Section II the method of generating via-points is explained. In this section the via-points will be interpolated to obtain collision-free and smooth geometric paths using Cubic B-spline curve.

### A. Cubic B-spline curve

In the Cartesian space the definition of a trajectory implies the determination of a geometric path to be tracked with a prescribed motion law. A curve in the Cartesian space has a parametric representation as follows:

$$p = p(u), \qquad u \in [u_{\min}, u_{\max}] \tag{2}$$

where $p$ is a continuous vectorial function and describes the curve when the independent variable $u$ ranges over some interval of the domain space [14].

Splines are piecewise polynomial functions which are used to interpolate sets of data points or to approximate curves. Cubic B-splines is a particularly efficient technique for the computation of splines, which can be viewed as a linear combination of some basic functions, such as:

$$s(u) = \sum_{j=0}^{m} p_j B_j^p(u), \qquad u_{\min} \le u \le u_{\max} \tag{3}$$

where the coefficient $p_j$ , $j$ = 0, . . . , $m$, called control points, which can be calculated by:

$$\begin{cases} p_0 = q_0 \\ p_1 = q_0 + \dfrac{u_4}{3} t_1 \\ p_{n+2} = q_n \\ p_{n+1} = q_n - \dfrac{1-u_{n+3}}{3} t_n \end{cases} \tag{4}$$

where $t$ is the tangent vector, and $q$ is the points to be interpolated or the via-points.

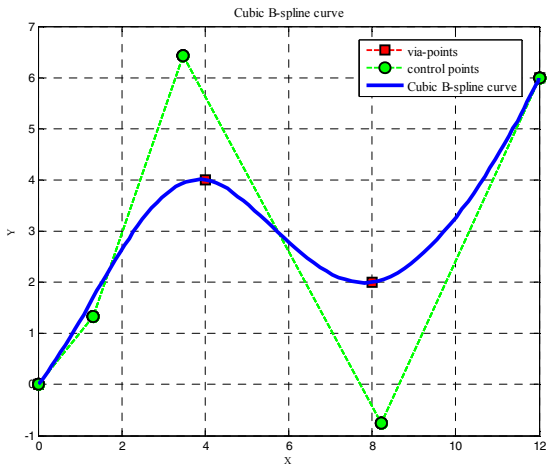


Fig. 3 An example of Cubic B-spline curve

Fig. 3 shows an example of Cubic B-spline curve. In this example, the points to be interpolated are given as:

$$q_1 = [0,0], q_2 = [4,4], q_3 = [8,2], q_4 = [12,6] \tag{5}$$

It can be easily seen that the path generated using Cubic B-spline curve passes all the via-points and is quite smooth.

### B. Geometric path generation

In the competition of RoboCup middle size league, there are ten robots in total including the goalkeepers. So there are 9 obstructors for a moving robot. In this paper all objects are modeled as circles with radius of 0.35m. Supposing the positions of the obstructors are known as is shown in Fig. 4 and Fig. 5, the illustrated geometric path can be obtained using the proposed method.

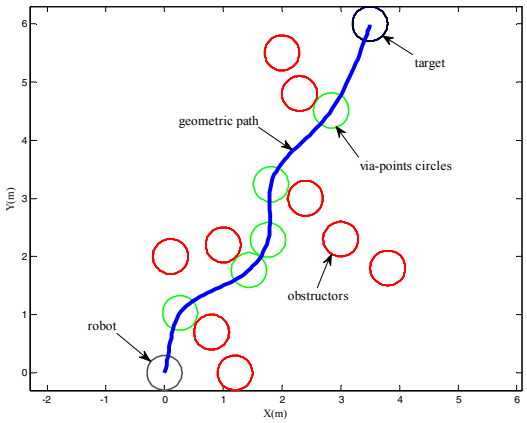


Fig. 4 Geometric path generation when the robot is at (0, 0)

In Fig. 4, the initial position of the robot is (0, 0) and the target is at (3.5, 6). All the obstructors are shown in red circle. It can be seen that 5 subtargets points are generated using the subtargets algorithm and there are 7 via-points including the robot and the target. The geometric path passes all the via-points and is completely collision-free.

In Fig. 5, the initial position of robot is (3, 1) and the target is at (3.5, 6). In this situation 4 via-points are generated and the geometric path is also collision-free and smooth.

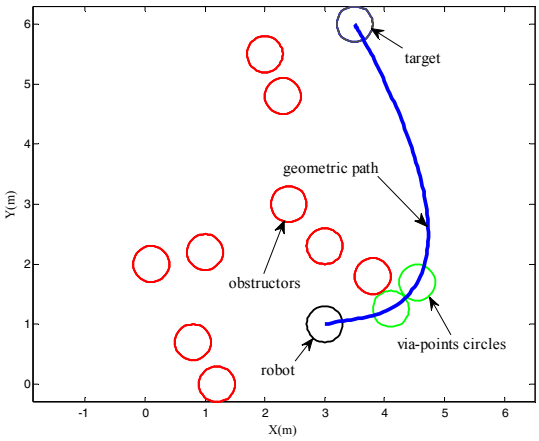


Fig. 5 Geometric path generation when the robot is at (3, 1)

## IV. FROM GEOMETRIC PATH TO TRAJECTORY

For a mobile robot, physical limitations such as velocities and accelerations must be taken into account when following a geometric path, which can be explained as the problem of composing the geometric path with the motion law. Equation (3) describes a curve parametrically in the Cartesian space and the trajectory is completely defined when the following motion law is provided:

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

A generic feed rate profile is available for discrete time systems with sampling period $T_s$ in [14] as follows:

$$u_{k+1} = u_k + \frac{v_k T_s}{\left|\frac{dp}{du}\right|_{u_k}} + \frac{T_s^2}{2}\left( \frac{a_k}{\left|\frac{dp}{du}\right|_{u_k}} - v_k^2 \left[ \frac{\left(\frac{dp}{du}^T \bullet \frac{d^2 p}{du^2}\right)}{\left|\frac{dp}{du}\right|^4} \right]_{u_k} \right) \tag{7}$$

where $v_k = v(kT_s)$ and $a_k = a(kT_s)$ are the samples of the desired profiles of velocity and acceleration at time $kT_s$. So at each sample period, a new *u* is calculated and combined with equation (3), the desired target position is obtained which can be described as:

$$\begin{cases} x_{desired} = x_{k+1} \\ y_{desired} = y_{k+1} \end{cases} \tag{8}$$

where $(x_{k+1}, y_{k+1})$ is the desired target position at time $(k+1)T_s$. The coordinate system of the robot is shown in Fig. 6.

Considering our soccer robot is omnidirectional with four wheels, it is easy to control the motion with a kinematic model, which means the robot can move to the target position exactly following the path once given the desired velocity in Cartesian space,. According to equation (8) the desired velocity and acceleration can be easily obtained as follows:

$$\begin{cases} vx_{desired} = vx_{k+1} = \dfrac{x_{k+1} - x_k}{T_s} \\ vy_{desired} = vy_{k+1} = \dfrac{y_{k+1} - y_k}{T_s} \end{cases} \tag{9}$$

$$\begin{cases} ax_{desired} = \dfrac{vx_{k+1} - vx_k}{T_s} \\ ay_{desired} = \dfrac{vy_{k+1} - vy_k}{T_s} \end{cases} \tag{10}$$

However, the calculated velocity and acceleration may exceed the capabilities of the robot, so saturations are necessary to avoid the overflow. Taking $vx_{desired}$ for example,

$$vx_{desired} = \begin{cases} vx_{\max}, & vx_{desired} \geq vx_{\max} \\ vx_{desired}, & others \\ -vx_{\max}, & vx_{desired} \leq -vx_{\max} \end{cases} \tag{11}$$

where $vx_{\max}$ is the maximum allowed velocity that the robot can supply. The saturations of $vy_{desired}$, $ax_{desired}$ and $ay_{desired}$ are the same as $vx_{desired}$.

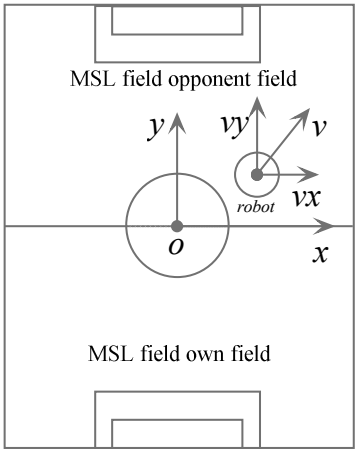


Fig. 6 The coordinate system of the robot

The robot receives the velocity command $(vx, vy)$ and then a PID controller is enough to allow the robot to reach the desired velocity in a short time and with less overshoot. The outputs of the PID controller are the accelerations of four wheels (if the robot is equipped with four wheels), which are driven by Maxon brushless DC motors and controlled by ELMO motor controllers.

## V. SIMULATION RESULTS

### A. Path generation results

As mentioned in Section II-B, the combination of the subtargets algorithm and the Cubic B-spline curve allow the exchange of positions of the robot and the target. The advantage of this exchange is shown in the comparison of Fig. 7 and Fig. 8.

In Fig. 7 the target is at (0, 0) and is surrounded by 7 obstructors. The initial position of the robot is at (1.8, 0). In this situation, the robot would oscillate between the left-limit position and the right-limit position if the subtargets algorithm was just simply used to generate a subtarget point for the robot to reach, which is a shortcut of the subtargets algorithm.

However if the positions of the robot and the target are exchanged, the oscillation can be easily avoided. In our method, this exchange is allowed because it won't influence the generation of the geometric paths which is explained in Section II.

Fig. 8 shows the path generation results after the position exchange of the robot and the target. It can be seen that the oscillation problem is solved and the path is successfully generated.

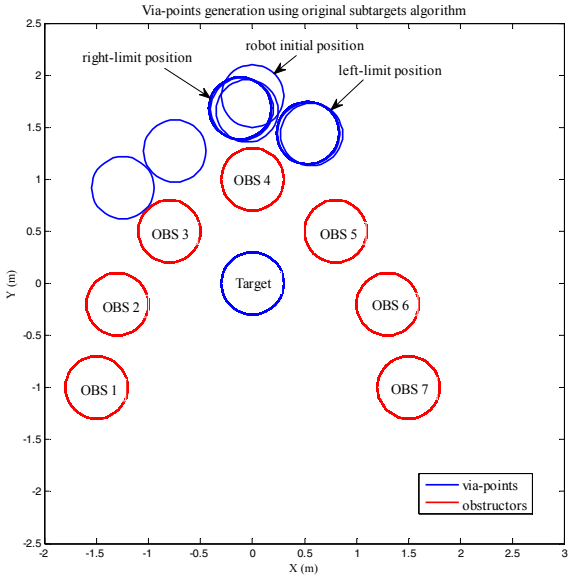


Fig. 7 Path generation using subtargets

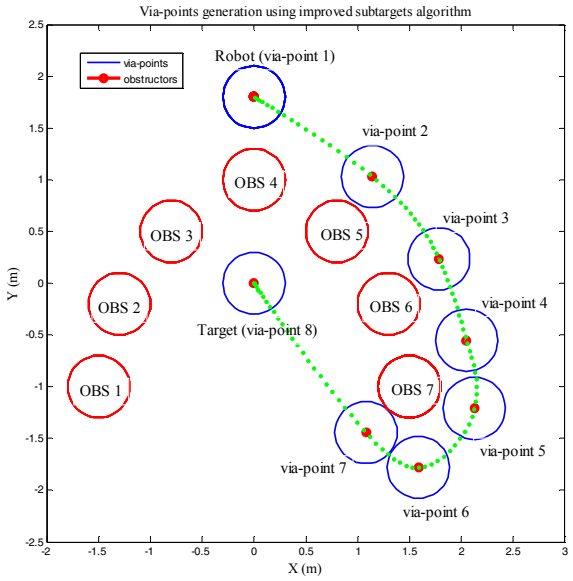


Fig. 8 Path generation using subtargets after the position exchange of the robot and the target

## B. *Trajectory tracking results*

### 1) *Simulation conditions*

The simulations are implemented in MATLAB Simulink and the step time is set to 10ms for real time purpose. The robots and obstructors have a radius of 0.35m. The maximum velocity and acceleration of the robot are 5m/s and 8m/s$^2$. Fig. 4 shows the positions of obstructors, target and the path that the robot will track.

Considering the dynamic features of both obstructors and targets, the subtarget points and geometric path are re-generated every 500ms which can ensure the obstacles avoidance in real time.

The velocity profile is given as *sin(20*t)+3* and the acceleration profile is the derivative of it where $t$ is the simulation time. The simulation is stopped once the robot reaches the target.

To ensure the clarity of the simulation, all objects are standing still. But our method will show the same efficiency even if the environment is highly dynamic because the re-computing period is short enough and the computation cost is quite low.

### 2) *Simulation results*

The velocity profile and the robot's velocity are shown in Fig. 9. It indicates that the robot can track the desired velocity quite accurately. Furthermore, at time 0 the velocity increases smoothly rather than sudden change.

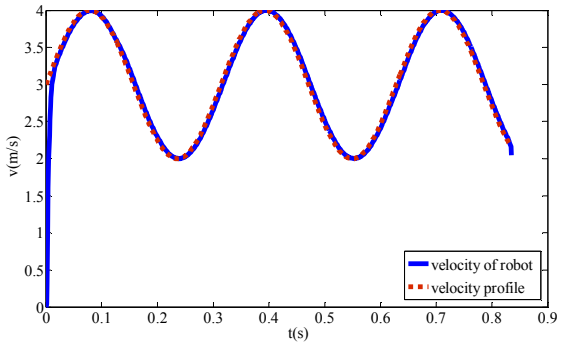


Fig. 9 Velocity tracking results

Fig. 10 shows the acceleration tracking results. The robot's acceleration exactly follows the profile with a minimal delay and is controlled under the physical limitation (8m/s$^2$).

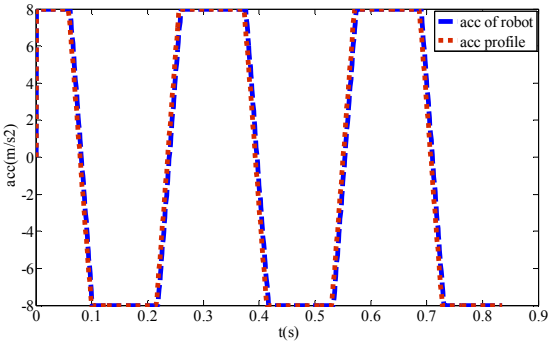


Fig. 10 Acceleration tracking results

As mentioned above, the robot must move along the geometric path in Fig. 4. The position tracking errors determine whether the robot is able to move to target without

any collision. Fig. 11 shows the ($x$, $y$) errors which are lower than 0.06m and it can enable the robot to avoid obstacles in the path when moving towards the target.

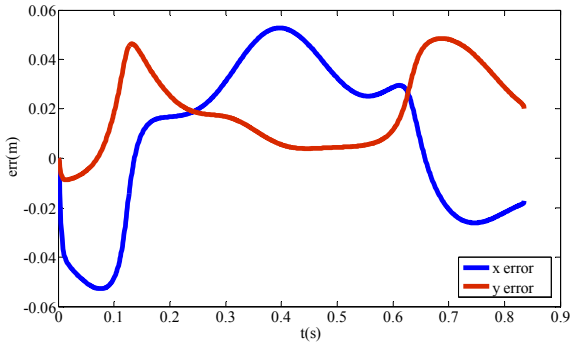


Fig. 11 Position errors

## VI. CONCLUSIONS

This paper proposes a new method of obstacles avoidance for mobile robot in real time, which combines the advantages of the subtargets algorithm and Cubic B-spline curve. The subtargets algorithm is used to generate via-points which can deal with complex situations in highly dynamic environments. A collision-free and smooth geometric path is generated using Cubic B-spline curve which makes it possible to control the velocity and acceleration of the robot to arbitrary desired profiles.

The simulation results prove that the method is more efficient in obstacles avoidance and the motion features of the robot can be controlled precisely.

## REFERENCES


[1] Masehian E, Katebi Y. Robot motion planning in dynamic environments with moving obstacles and target[J]. International Journal of Mechanical Systems Science and Engineering, 2007, 1(1): 20-25.

[2] Rathbun D, Kragelund S, Pongpunwattana A, et al. An evolution based path planning algorithm for autonomous motion of a UAV through uncertain environments[C]. Digital Avionics Systems Conference, 2002. Proceedings. The 21st. IEEE, 2002, 2: 8D2-1-8D2-12 vol. 2.

[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] Omid Reza Esmaeili Motlagh, Tang Sai Hong, Napsiah Ismail, Development of a new minimum avoidance system for a behavior-based mobile robot, Fuzzy Sets and Systems 160 (13) (2009) 1929-1946.

[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] 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.

[9] Simmons R. The curvature-velocity method for local obstacle avoidance[C].Robotics and Automation, 1996. Proceedings. 1996 IEEE International Conference on. IEEE, 1996, 4: 3375-3382.

[10] Ko N Y, Simmons R G. The lane-curvature method for local obstacle avoidance[C].Intelligent Robots and Systems, 1998. Proceedings., 1998 IEEE/RSJ International Conference on. IEEE, 1998, 3: 1615-1621.

[11] Fernández J L, Sanz R, Benayas J A, et al. Improving collision avoidance for mobile robots in partially known environments: the beam curvature method[J]. Robotics and Autonomous Systems, 2004, 46(4): 205-219.

[12] Shi C, Wang Y, Yang J. A local obstacle avoidance method for mobile robots in partially known environment[J]. Robotics and Autonomous Systems, 2010, 58(5): 425-434.

[13] 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.

[14] Biagiotti L, Melchiorri C. Trajectory planning for automatic machines and robots[M]. Berlin: Springer, 2008.