2014 UKACC International Conference on Control
9th - 11th July 2014, Loughborough, U.K

Improving the Velocity Obstacle Approach for Obstacle Avoidance in Indoor Environments

Ahmad Alsaab and Robert Bicker

***Abstract*—The velocity obstacle approach (VO) is considered an easy and simple method to avoid moving obstacles, where the collision cone principle is used to detect the collision situation between two circular-shaped objects. The VO approach has two challenges when applied in indoor environments. The first challenge is to extract collision cones of non-circular objects from sensor data, where applying fitting circle methods generally produce large and inaccurate collision cones specially for line-shaped obstacle such as walls. The second challenge is that the mobile robot cannot sometimes move to its goal because all its velocities to the goal are located within collision cones. The contribution of this paper is that a method was demonstrated to extract the collision cones of circular and non-circular objects using a laser sensor, where the obstacle size and the time to collision are considered to weight velocities of the robot. Experiments carried out in an indoor environment showed that the mobile robot successfully avoided dynamic obstacles which have different abilities to avoid other obstacles.**

***Index Terms*—Velocity obstacle, non-circular objects, collision time**

I. Introduction

THE autonomous ability of mobile robots provides large application areas including routine tasks such as office cleaning and delivery, access to dangerous areas that are unreachable by humans such as cleanup of hazardous waste sites in nuclear power stations and planetary explorations. The key challenge of a mobile robot is to avoid obstacles which may prevent it from reaching its goal. Many algorithms have been suggested to solve this problem such as the potential field algorithm (PFA) and the vector field histogram method (VFH). However, these methods do not consider the kinematic constraints of the mobile robot such as limited velocities and accelerations [1], [2]. The Dynamic Window Approach (DWA) considers these constraints, where a suitable pair of rotation and linear speeds is chosen to avoid obstacles through a circular path [3]. All the previous mentioned algorithms are suitable for avoiding static obstacles. The linear velocity obstacle algorithm (VO) was proposed to avoid moving obstacles, where it defines the collision situation between two circular objects moving with constant velocities [4]. The non-linear V-Obstacle method was introduced as an extension of the VO algorithm to avoid obstacles moving on arbitrary trajectories [5].

The first challenge of the VO algorithm is that in the real workspace, not all obstacles have circular shapes. The second challenge is that the mobile robot cannot sometimes move to its goal because all its velocities to the goal are located within collision cones as shown in Fig. 1, but there is at least one safe path to the goal. Many studies have been implemented the VO algorithm using simulation programs,

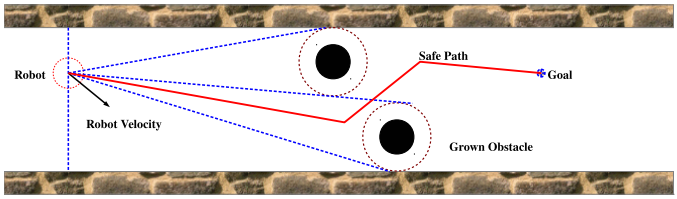


Fig. 1.The robot cannot reach its target according to the VO method

furthermore they have considered that the obstacles have circle shapes and their velocities are known. For instance, [6], [7] proposed the *probabilistic collision cone* principle, where the grown obstacle is extended by uncertainty in the exact radii of the obstacle and robot, the collision probability in the exact collision cone is given a value of 1, while inside the uncertain area the collision probability is given a value between 1 and 0. However, they have not considered the collision time. In [8], [9], the grown obstacle radius was modified depending on the collision distance and collision time. This algorithm can be applied to avoid circular obstacles. For non-circular obstacles, the robot has to fit the sensor data into circles using a circle fitting method [10], [11], [12]. The *optimal time horizon* principle has been proposed in [13], where the smallest time horizon is used to define the collision course velocities. In the simulation test, they used circular obstacles and considered the obstacle's speed known. However, in the real world, the robot has to extract shapes of obstacles from sensor data, grow them using the robot radius and then extract the collision cones.

A mobile robot avoids dynamic obstacles by collecting information about its surrounding environment using sensors, then clusters the sensor data, and then utilizes a tracking algorithm to estimate the velocities and locations of obstacles. 2D scanning laser sensors are common in mobile robot applications, provide accurate information about the robot workspace, require a relatively small computation time and are unaffected by changes in illumination. There are two types of laser data cluster methods; the distance-based clustering methods [14], [15], [16] and kalman filter-based methods [17]. The KF-based methods detect the segments and their directions precisely, but are more complex than the distance-based methods and require relatively large computation time.

In this work, a reactive control architecture was adopted, where a 2D laser sensor was utilized to recognize the robot's workspace and the laser data was clustered using the distance-based clustering method proposed in [16]. The extended particle filter algorithm (EXPF) proposed in [18] was used to estimate the obstacle velocity. A method was demonstrated to extract the collision cones of circular and non-circular objects, where the collision time and the obstacle size were considered.

978-1-4799-5011-9/14/$31.00 ©2014 IEEE

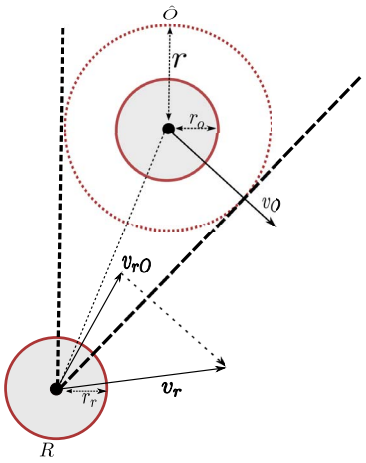


Fig. 2. Velocity obstacle

II. Principle of the Velocity Obstacle Method

Suppose a circular robot (R) and circular obstacle (O) with radii (r_r, r_o), centers(c_r, c_o) and linear velocities (v_r, v_o) respectively. The robot is treated as a point by growing the obstacles using the radius of the robot as shown in Fig. 2.

Let

$$\hat{O} = \{c_o + r : r = r_r + r_o\} \tag{1}$$

$$v_{ro} = v_r - v_o \tag{2}$$

$$\lambda_{ro}(v_{ro}) = \{c_r + \mu v_{ro}\} \tag{3}$$

where $\hat{O}$ represents the grown obstacle, v_{ro} is the relative speed, λ_{ro} is a vector starting from the robot center with the slope of the relative velocity.

The collision cone boundaries are determined by two tangents of the grown obstacle which pass from the robot center c_r. In the VO method, the relative velocity v_{ro} causes collision if the condition is met $\hat{O} \cap \lambda_{ro}(v_{ro}) \neq \phi$. It can define the relative velocity which causes collision as:

$$CC_{ro} = \{v_{ro} : \hat{O} \cap \lambda_{ro}(v_{ro}) \neq \phi\} \tag{4}$$

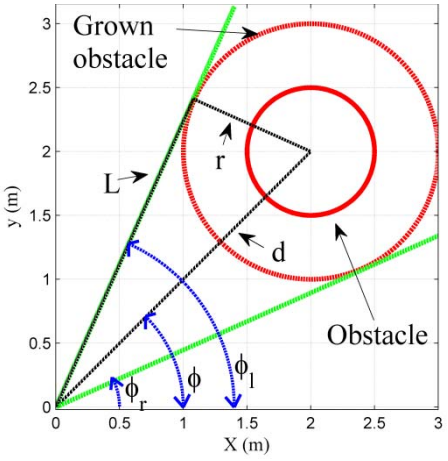
The absolute collision velocities of the mobile robot are defined as following:

$$VO_{ro} = \{v_r : v_r - v_o \in CC_{ro}\} \tag{5}$$

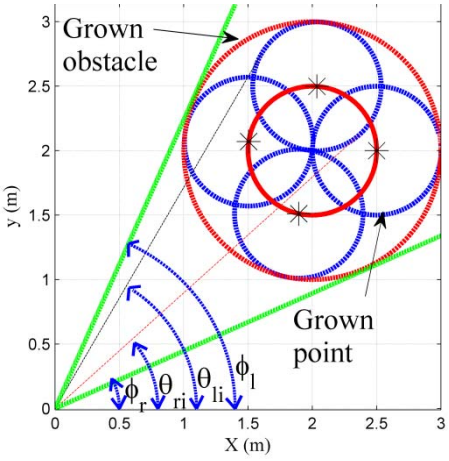
which can be written in another form:

$$VO_{ro} = v_r + CC_{ro} \tag{6}$$

To avoid a collision between the robot and an obstacle, the robot has to select a velocity $v_r \notin VO_{ro}$. To avoid multiple obstacles, the robot has to select a velocity which meets this condition for each obstacle.



(a)



(b)

Fig. 3. Grown obstacle circle

III. Collision Cones of Non-Circular Obstacles

In Fig 3.a, the tangent angles which define the collision cone are given as follows:

$$d = \sqrt{(x_o - x_r)^2 + (y_o - y_r)^2} \tag{7}$$

$$L = \sqrt{d^2 - r^2} \tag{8}$$

$$\phi = \arctan(\frac{y_o - y_r}{x_o - x_r}) \tag{9}$$

$$\phi_l = \arctan(\frac{r}{L}) + \phi \tag{10}$$

$$\phi_r = -\arctan(\frac{r}{L}) + \phi \tag{11}$$

where d is the distance between the robot center and the obstacle center, L is the tangent length, ϕ is the angle between the obstacle and the robot in the global frame, (ϕ_l, ϕ_r) are the left and right tangent angles, and (x_r, y_r) and (x_o, y_o) are the robot and obstacle coordinates.

For a circular obstacle, the collision cone can be found

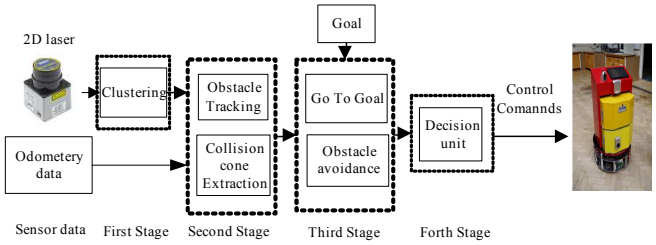


Fig. 4. Behaviour based architecture

by extending the obstacle radius $r = r_o + r_r$, and then the tangents are calculated. However, if each point in the obstacle circumference is grown by r_r (Fig 3.b) the outer boundaries of the grown points form the grown obstacle circle. The left and right tangent angles θ_{ri}, θ_{li} of a grown point can be calculated using equations (7-11). The collision cone can be defined by the right tangent which has the minimum angle and the left tangent which has the maximum angle. Therefore in this study, a method has been demonstrated to find the actual collision cone of an obstacle from laser data using the following steps:

- Each laser point is grown by the robot radius.
- Determining the left tangent angles $\theta_{l1},,,\theta_{ln}$and right tangent angles $\theta_{r1},,,\theta_{rn}$of the grown laser points, where n refers to total number of laser points.
- The maximum left tangent represents the left boundary of the collision cone of the grown obstacle, and the minimum right tangent represents the right boundary of the collision cone.

IV. Behaviour based architecture

A reactive control architecture was adopted in this study to produce the control command for the mobile robot as shown in Fig. 4, in which the sensor data comes from the laser sensor and odometry data. In the first stage, the laser data is clustered into separate groups (obstacles) using the distance-based clustering method proposed in [16]. In the second stage, the obstacles velocities are estimated by the EXPF algorithm proposed in [18], and the collision cones are extracted from the laser data by the proposed method in this paper. The third stage has two behaviours; *Go To Goal* and *Obstacle Avoidance.* In the fourth stage, the outputs of the behaviours are fused to produce the control commands.

A. The Go To Goal behaviour

This behaviour utilizes the current robot position coming from the odometry sensors and the goal position to define the speed and direction to the target. In the global reference frame, the angle and distance of the target are given as follows:

$$\theta_t = \arctan(\frac{y_t - y_r}{x_t - x_r}) \tag{12}$$

$$d_t = \sqrt{(x_t - x_r)^2 + (y_t - y_r)^2} \tag{13}$$

where (x_t, y_t), θ_t and d_t represent the target position, the angle and distance between the robot and the target respectively.

The desired speed to the target is given as:

$$V_t = \begin{cases} v_{max} & if\ d_t > d_{th} \\ v_{max}\frac{d_t}{d_{th}} & otherwise \end{cases} \tag{14}$$

where v_{max} and d_{th} denote the robot's maximum speed and threshold distance respectively.

B. The Obstacle Avoidance behaviour

The collision time and obstacle size are considered by assigning the robot velocities with different weights. All the free collision velocities are given a numerical value 1, while the weights of the collision velocities are given as:

$$W_o(i) = \begin{cases} 0 & if\ t_c(i) < t_{ct} \\ \exp(-a/t_c(i))* & (1 - b * \delta * t_c(i))\ otherwise \end{cases} \tag{15}$$

$$\delta = \min(\phi_l - \theta(i), \theta(i) - \phi_r) \tag{16}$$

where $t_c(i)$ refers to the collision time, (a, b) are constants, t_{ct} is the critical collision time, and $\theta(i)$ is the robot direction.

The robot velocities within a collision cone have different collision distances depending on the obstacle shape. The collision distance of a velocity $(v(i), \theta(i))$ is given as:

$$d_c(i) = P_l(I) \tag{17}$$

$$I = floor(\theta(i)/\sigma) \tag{18}$$

where P_l, I and σ refer to the array of return laser points, the point index and the angle resolution of the laser sensor respectively. $floor$ is a function which returns the integer value.

C. Decision unit

The main task of this block is to fuse the outputs of the behaviours, and then generate the control commands which minimize the collision risk and maximize the speed to the goal. The robot velocities are weighted according to the output of the *Go to Goal* behaviour and the weights W_o coming from the obstacle avoidance behaviour.

$$W(i) = W_o(i) * (a1 + cos(\theta_t - \theta(i)))(a2 - \parallel V_t - v(i) \parallel) \tag{19}$$

where $a1, a2$ are constants.

The maximum weight velocity is chosen to produce control commands given as:

- the rotation speed command:

$$w_c = k_p * \theta_c \tag{20}$$

- the linear speed command:

$$V_c = \begin{cases} v_c & if\ v_c < V_{cmax} \\ 0 & if\ 0 > V_{cmax} \\ V_{cmax} & otherwise \end{cases} \tag{21}$$

$$V_{cmax} = v_{max} - k_a * \frac{w_c}{w_{max}} \tag{22}$$

where (v_c, θ_c) represent the maximum weighted velocity, k_p, k_a are constants, w_{max} represents the maximum rotation speed. w_c, V_c are the control commands.

V. Result and Decision

A. Producing the collision cone

The collision cone produced by the method demonstrated in this paper is compared with that produced by two fitting circle algorithms proposed in [11][12]. Fig. 5 shows a virtual non-circular obstacle, where the laser angular resolution was chosen to be $5°$. The returned laser points are given in Table I.

The circle $C1$ was generated using the method proposed in [11], while the circle $C2$ was produced by the method in [12]. Obviously the circle $C1$ is relatively huge which produces a huge collision cone. Therefore the comparison was made between the collision cone produced by C_2 with that of the algorithm developed in this study.

The calculated radius of $C2$ was $r_o = 0.36\,m$ which was grown by the robot radius $r_r = 0.5\,m$ to determine the collision cone. The tangents angles of the grown obstacle were calculated by applying the equations (7-11). As a result, The collision cone angle produced by $C2$ was $\phi_l - \phi_r = 70°$. It can be seen in Fig. 6, the collision cone does not touch any point of the grown laser point.

For the proposed algorithm in this study, Table II lists the calculated tangent angles of the grown laser points, it can be seen that the second point has the maximum left angle and minimum right angle which form the collision cone. Hence, it is not always correct to find the collision cone based on the first and last laser points of the laser cluster. The angle $\phi_l - \phi_r$ of the collision cone was $60°$. It can be noticed from Fig.7 that the collision cone is touching the grown obstacle at two points. Clearly the proposed method produced a collision cone more accurate and smaller than that produced by the circle fitting methods.

B. Experiment results

The experiments were implemented using three robots; two Pioneer P3-DX robots and Nubot robot which was made in Newcastle university. The experiments aimed to show that the robot can avoid collision with dynamic obstacles which have different intelligence levels. The obstacles were classified according to the obstacle avoidance method into three classes; non-intelligent obstacles which don't have any obstacle avoidance algorithm, the low-level intelligent obstacles have ability

Table I Collected laser points

laser point	1	2	3	4
Angle(degree)	40	45	50	55
Distance (m)	1.4	1	1.14	1.8

Table II collision cones angles

laser point	1	2	3	4
left angle(degree)	60.1	75	70.1	71.1
Right angle(degree)	19	15	29	38.8

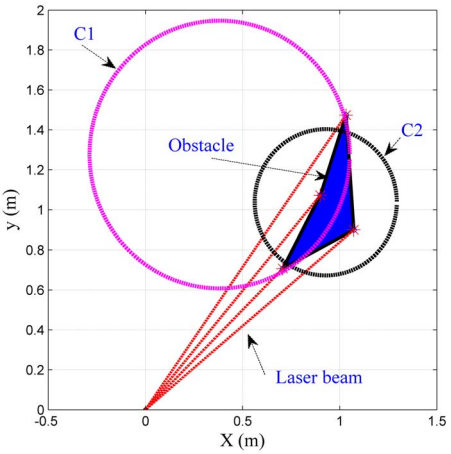


Fig. 5. Fitting laser points into a circle

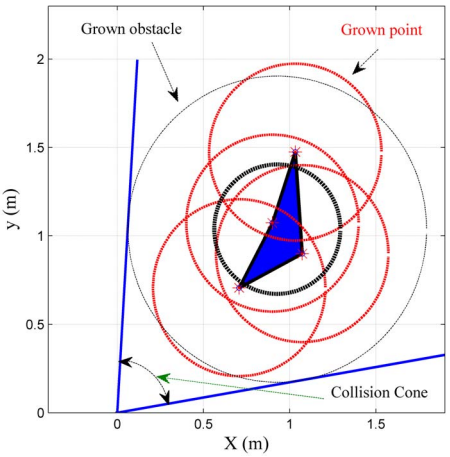


Fig. 6. Collision cone produced by the fitting circle method

to avoid static obstacles, and intelligent obstacles can avoid the static and moving obstacles.

The maximum linear and rotation speeds of the robots were adjusted to $0.5m/s$ and $100degree/s$ respectively. In equation 14, the threshold distance was chosen to be $d_{th} = 1\,m$. The parameters a and b in equation 15 were chosen to be 5 and 0.5 respectively, while the critical collision time is $t_{ct} = 1s$. The weighting function parameters in equation 19 are adjusted to $\alpha 1 = 0.1,\ \alpha 2 = 0.6$. The parameters K_p , K_a in equation 20 and 22 are selected as 1 and 0.5 respectively.

1) Passing through a narrow Gap: The mobile robot (Pioneer P3-DX) had to pass through a narrow space as shown in Fig. 8. The robot's target and start point were chosen to be $(x_t = 3m,\ y_t = 0m)$ and $(x_r = 0m,\ y_r = 0m)$ respectively, while the narrow space was located at $x = 1.55\,m$. The widths of the robot and narrow space were $0.45\,m$ and $0.6\,m$ respectively. Therefore the robot had $7.5\,cm$ free space at each side whilst passing through the narrow gap. Fig. 9 shows that the robot was able to pass the narrow space without any collision. The robot implemented a straight line to pass the narrow space, and then turned to the goal in a curved path. Fig. 10 shows the robot speed vs the x coordinate of the mobile robot. The robot speed increased form $0\,m/s$ at $x = 0$ to achieve a speed of $0.5\,m/s$ at $x = 0.6\,m$. At $x = 1.42\,m$ the

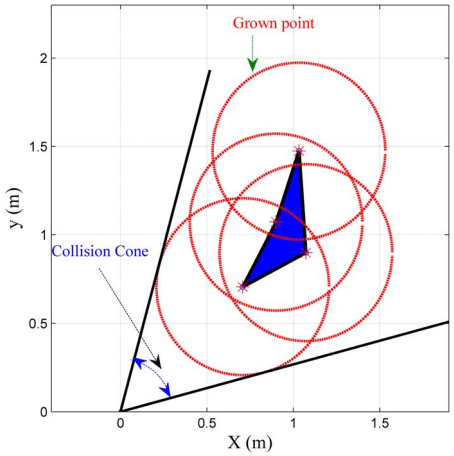


Fig. 7. Collision cone produced by the proposed method

robot started to decrease its speed when passing the narrow space, where the speed became $0.34\,m/s$ at $x = 1.57\,m$, and then increased to stop the robot at the goal.

2) Avoiding a non-intelligent obstacle: The Nubot robot was used as a non-intelligent moving obstacle which was not programmed to avoid obstacles (Fig. 11). The robot (Pioneer) avoided the non-intelligent obstacle moving with a fixed speed $0.5\,m/s$. The start positions of the robot and obstacle were chosen as $(0\,m, 0\,m)$ and $(5\,m, 0\,m)$ respectively. As shown in Fig. 12, the Pioneer robot moved on a straight line, executed an avoidance manoeuvre at $t = 3.6\,s$, and then started to return to its goal at $t = 7.5\,s$.

3) Avoiding a low-level intelligent moving obstacle : Here the Nubot robot, having the ability to avoid static obstacles based on the potential field algorithm, where the Nubot robot used the laser data to produce the repulsive force and the odometry data to produce the attractive force. The Pioneer started to avoid the Nubot robot at $t = 3\,s$ as shown in Fig. 13 and then turned to its goal $(5\,m, 0\,m)$ at $t = 8\,s$. The low-level intelligent Nubot robot implemented a small avoidance manoeuvre thereby avoiding the Pioneer robot at moment $t = 7\,s$. It can be noticed in Fig. 13, that the Pioneer robot started the avoidance manoeuvre $4\ s$ before the Nubot robot.

4) Avoiding an intelligent moving obstacle: In this experiment, the obstacle (Nubot) was programmed with the same navigation algorithm of the Pioneer robot. As shown in Fig. 14, the robot started the avoidance action at $t = 3\,s$, while the Nubot robot started at $t = 4\,s$. The two robots turned to their targets nearly at the same moment $t = 6.3\,s$, forming paths that are nearly symmetric.

5) Avoiding two dynamic obstacles: In this test, the Pioneer robot had to avoid two moving obstacles, the Nubot robot (obstacle B) moving in the same direction and another Pioneer robot (obstacle A) moving in the opposite direction. The speed of the obstacle B was adjusted to $0.33m/s$, while that of obstacle A was adjusted to $0.4m/s$. At $t = 2.4\,s$ the mobile robot turned to avoid the collision with the obstacle A and followed the obstacle B as shown Fig. 15. At $t = 6\,s$ the robot reduced its speed to avoid collision with the obstacle B, and started to pass the obstacle B after the obstacle A passed.

Fig. 8. Pioneer P3-DX and a narrow space

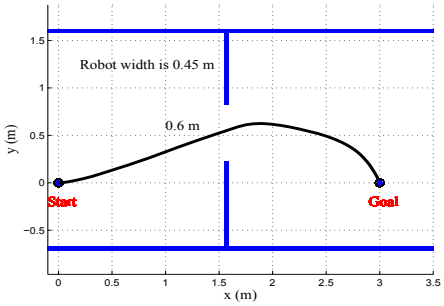


Fig. 9. The robot path during passing the narrow gap

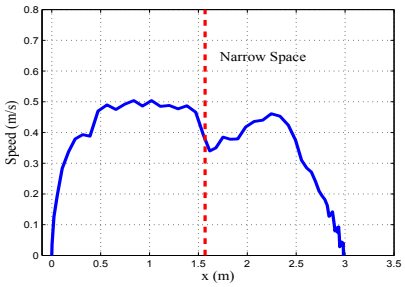


Fig. 10. The robot speed during passing the narrow space

Fig. 11. Pioneer P3-DX and Nubot

VI. Conclusion

In this paper, a method was proposed to find the collision cones of circular and non-circular obstacles from the laser data, in which each laser point is grown by the robot radius, where the maximum left tangent represented the left boundary of the collision cone, while the minimum right tangent represented the right boundary of the collision cone. The collision cone produced by the proposed method is more accurate if it

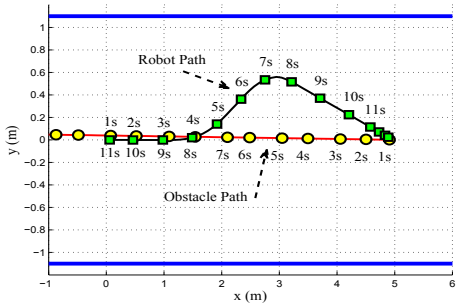


Fig. 12. The robot avoided a non-intelligent obstacle

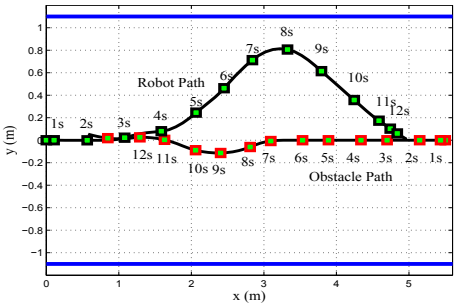


Fig. 13. The robot and low-level intelligent obstacle paths

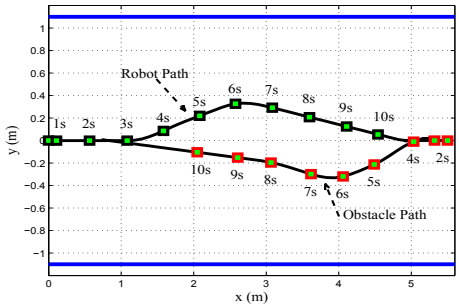


Fig. 14. Intelligent obstacle avoidance

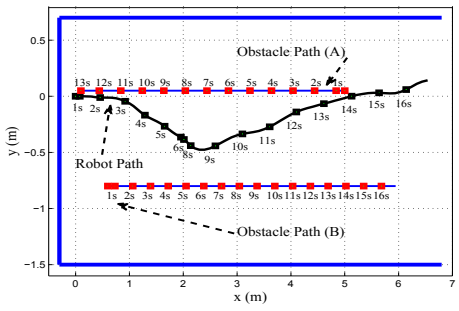


Fig. 15. Two obstacles avoidance

is compared with that produced by the circle fitting methods. The collision time and the obstacle size were also considered to weigh the robot velocities.

The experiment results showed that the mobile robot was able to successfully pass a narrow gap. Again, it avoided obstacles which had different ability to avoid other obstacles. The robot was able to avoid two moving obstacles, where one of them moved in the same direction, while the other moved in the opposite direction.

REFERENCES

[1] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," in *in Robotics and Automation. Proceedings. 1985 IEEE International Conference*, 1985, pp. 500–505.

[2] J. Borenstein and Y. Koren, "The vector field histogram-fast obstacle avoidance for mobile robots," in *Robotics and Automation, IEEE Transactions on*, vol. 7, 1991, pp. 278–288.

[3] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," in *Robotics and Automation Magazine, IEEE*, vol. 4, 1997, pp. 23–33.

[4] P. Fiorini and Z. Shiller, "Motion planning in dynamic environments using velocity obstacles," in *International Journal of Robotics Research*, vol. 7, 1998, pp. 760–772.

[5] Large, Frederic, Laugier, Christian, and Z. Shiller, "Navigation among moving obstacles using the nlvo: Principles and applications to intelligent vehicles," in *Autonomous Robots*, vol. 19, 2005, pp. 159–171.

[6] B. Kluge, "Recursive agent modeling with probabilistic velocity obstacles for mobile robot navigation among humans," in *in Intelligent Robots and Systems, 2003. (IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, vol. 1, 2003, pp. 376–380.

[7] B. Kluge and E. Prassler, "Reflective navigation: individual behaviors and group behaviors," in *in Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*, vol. 4, 2004, pp. 4172–4177.

[8] Z. Xunyu, P. Xiafu, and Z. Jiehua, "Dynamic collision avoidance of mobile robot based on velocity obstacles," in *in Transportation, Mechanical, and Electrical Engineering (TMEE), 2011 International Conference on*, 2011, pp. 2410–2413.

[9] M. Yamamoto, M. Shimada, and A. Mohr, "Online navigation of mobile robot under the existence of dynamically moving multiple obstacles," in *in Assembly and Task Planning, 2001, Proceedings of the IEEE International Symposium on*, 2001, pp. 13–18.

[10] J. Guivant, E. Nebot, and H. Durrant-Whyte, "Simultaneous localization and map building using natural features in outdoor environments," in *Intelligent Autonomous Systems VI*, 2000, pp. 581–588.

[11] W. Gander, G. H. Golub, and R. Strebel, "Least squares fitting of circles and ellipses," in *BIT*, vol. 34, 1994, pp. 558–875.

[12] J. Vandorpe, H. V. Brussel, and H. Xul, "Exact dynamic map building for a mobile robot using geometrical primitives produced by a 2d range finder," in *in Robotics and Automation, 1996. Proceedings., 1996 IEEE International Conference on*, vol. 1, 1996, pp. 901–908.

[13] Z. Shiller, O. Gal, and T. Fraichard, "The nonlinear velocity obstacle revisited: the optimal time horizon," in *Workshop on Guaranteeing Safe Navigation in Dynamic Environments, the 2010 IEEE Int. Conf. on Robotics and Automation, Anchorage, AK (US)*, 2010.

[14] K. J. Lee, "Reactive navigation for an outdoor autonomous," in *in Mechanical and Mechatroinc Engineering vol. Master: University of Sydney*, 2001.

[15] K. Dietmayer, J. Sparbert, and D. Streller, "Model based object classification and object tracking in traffic scenes from range images," in *In Proceedings of intelligent Vehicles Symposium*, 2001.

[16] S. Santos, F. S. J.E. Faria and, R. Araujo, and U. Nunes, "Tracking of multi-obstacles with laser range data for autonomous vehicles," in *in 3rd National Festival of Robotics Scientific Meeting (ROBOTICA) Lisbon, Portugal*, 2003, pp. 59–65.

[17] K. Rebai, A. Benabderrahmane, O. Azouaoui, and N. Ouadah, "Moving obstacles detection and tracking with laser range finder," in *in Advanced Robotics, 2009. ICAR 2009. International Conference onl*, 2009, pp. 1–6.

[18] M. M. Romera, M. A. S. VÃązquez, and J. C. G. GarcÃ■a, "Tracking multiple and dynamic objects with an extended particle filter and an adapted k-means clustering algorithm," in *Preprints of the 5th IFAC/EURON Symposium on Intelligent Autonomous Vehicles (IAV)*, 2004.