Robotica (1997) volume 15, pp 493–510. Printed in the United Kingdom © 1997 Cambridge University Press

Obstacle avoidance motion planning for mobile robots in a dynamic environment with moving obstacles

Chia-Pin Wu†, Tsu-Tian Lee*, and Chau-Ren Tsai*

(Received in Final Form: October 18, 1996)

SUMMARY

A new real-time obstacle avoidance method for mobile robots has been developed. This method, namely the vector-distance function method, permits the detection of obstacles (both moving and stationary) and generates a path that can avoid collisions. The proposed approach expresses the distance information in a vector form. Then the notion of weighting is introduced to describe relationship between sensors of mobile robots and the target to be reached. Furthermore, R-mode, L-mode and T-mode are introduced to generate a safe path for the mobile robot in a dynamic environments filled with both stationary and moving obstacles. The algorithm can deal with a complicated obstacle environment, such as multiple concave and convex obstacles. Simulation results are included to demonstrate the applicability and effectiveness of the developed algorithm.

KEYWORDS: Mobile robots; Dynamic environment; Moving obstacles; Vector-distance function.

1. INTRODUCTION

The motivation for studying motion planning with multiple moving objects stems from factory automation. An automated factory could utilize many robots, some moving and interacting in common workspace. Some of these robots could be mobile, moving as independent vehicles carrying materials throughout the factory. The aim of this research is to solve the basic problem of motion planning of mobile robots in a dynamic environment with moving and stationary obstacles.

Basically, the obstacle avoidance algorithms can be divided into two categories, namely, model-based and sensor-based. The model-base algorithm generates a collision-free safe path via a built-in obstacle model, while the sensor-based algorithm generates a collision-free safe path via sensors to sense environment information.

Generally speaking, the model-based approach considers the problem in a global way. It assumed that all obstacles in the working space can be exactly and completely described. As a result, the optimal collision-free safe path can be found. Typical approaches for model-base are:
(1) polyhedral obstacle model;[1–5] (2) free space decomposition;[6–8] (3) orthogonal space decomposition;[9,10] (4) grid model;[11] and (5) potential field.[12] However, most of the model-based systems use polyhedral to describe obstacles, then the collision-free safe path is found by using artificial intelligence search algorithms.

For sensor-based systems, the sensed information is recorded on a grid map.[13] Obstacles are represented by cells, on the map. The value in the cell encodes whether there exists or not an obstacle. A sensor-based system usually make decision locally; it cannot ensure a solution to the problem. Recently, Huang and Lee[14] developed a sensor-based system for obstacle avoidance. The algorithm consists of both a heuristic mode and a track mode. In addition, it is shown that the algorithm can always find an safe path for mobile robot except if the problem itself has no solution. Basically, in reference [14]; they assumed all obstacles to be stationary. Hence the speed and acceleration of the mobile robot are not taken into account while developing the algorithm.

In this paper, we shall derive a sensor-based algorithm for collision-free motion planning with both stationary obstacles and moving obstacles. We assume that the mobile robot is moving with constant acceleration, and the friction coefficient is also constant. We shall introduce the notion of vector-distance function to decide locally in which direction to move. In addition, according the mathematical relations between the effective range of sensors, and the speed of mobile robot which we shall derive in this paper, a mobile robot shall be able to decide the maximum possible speed at any particular time and move without colliding with moving and stationary obstacles. The algorithm shall consist of a rotate mode, lock mode, and track mode. The motion direction is determined by a simple equation expressed in a vector-distance function, regardless of which specific mode the mobile robot currently adopts. Since the computation speed of the proposed algorithm is very fast, it can be implemented in real-time.

This paper is organized as follows: In section 2, the notion of vector-distance function is introduced; some

* Department of Electrical Engineering, National Taiwan Institute of Technology, 43 Keelung Road, Section 4, Taipei, Taiwan 106.
† Department of Electrical Engineering, Oriental Institute of Technology, 58 Sze-chuan Road, Section 2, Pan-Chiao, Taipei, Taiwan 220.

https://doi.org/10.1017/S0263574797000593 Published online by Cambridge University Press

mathematical relations that shall be frequently used for motion planning are derived. A general approach for motion planning with moving and possibly time-varying obstacles is presented in section 3. Simulation results are included in section 4. Finally, section 5 provides conclusions on the proposed research.

2. THE VECTOR-DISTANCE FUNCTION AND ITS APPLICATIONS

2.1 The vector-distance function

Consider two forces $\vec{f}_1$ and $\vec{f}_2$ acting on the center C of a object as shown in Figure 2.1. When $\vec{f}_1 + \vec{f}_2 = 0$, the center C will not move. If two external forces $\vec{f}_a$ and $\vec{f}_b$ are applied as shown on Figure 2.2, the center C will move toward a direction on which external force is smaller. If $\|\vec{f}_a\| = \|\vec{f}_b\|$, the center C stays at the same location. Let w_1 and w_2 be two weighting coefficients along the $\vec{f}_1$ and $\vec{f}_2$ directions, respectively. Then, when forces are at equilibrium,

$$w_1(\|\vec{f}_1\| - \|\vec{f}_a\|) = w_2(\|\vec{f}_2\| - \|\vec{f}_b\|) \tag{2.1}$$

or

$$\|f_a\| = \frac{w_1\|\vec{f}_1\| + w_2\|\vec{f}_b\| - w_2\|\vec{f}_2\|}{w_1} \tag{2.2}$$

Hence, center C will move along the $\vec{f}_1$ direction if

$$0 \le \|\vec{f}_a\| \le \frac{w_1\|\vec{f}_1\| + w_2\|\vec{f}_b\| - w_2\|\vec{f}_2\|}{w} \tag{2.3}$$

This notion can be extended to motion planning. Let d_1 and d_2 to be the distance between p_1 and C', and p_2 and C', respectively, while no obstacles are present (see Figure 2.3). Now suppose objects A and B are moving toward point C' (or equivalently, two external forces $\vec{f}_a$ and $\vec{f}_b$ are applied). Let $d_a(d_b)$ be the distance that the object A (object B) move toward C', as shown in Figure 2.4. At equilibrium, we have from (2.3)

$$d_a = \frac{w_1 d_1 + w_2 d_b - w_2 d_2}{w_1} \tag{2.4}$$

when $0 \le d_a < (w_1 d_1 + w_2 d_b - w_2 d_2)/w_1$, C' will move toward p_1. As a result, the effect of d_b on C' will be decreased. Let $w_2 = 1$, $d_1 = d_2$ and assume d_b is very small that can be neglected. Then equation (2.4) becomes

$$d_a \approx \left(1 - \frac{1}{w_1}\right) d_1 \tag{2.5}$$

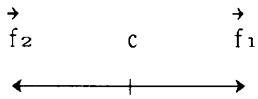


Fig. 2.1. The object C is in equilibrium when $\vec{f}_1 + \vec{f}_2 = 0$.

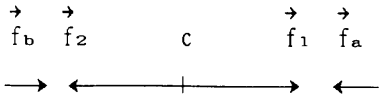


Fig. 2.2. External forces are applied on C.

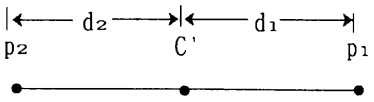


Fig. 2.3. No obstacles is present.

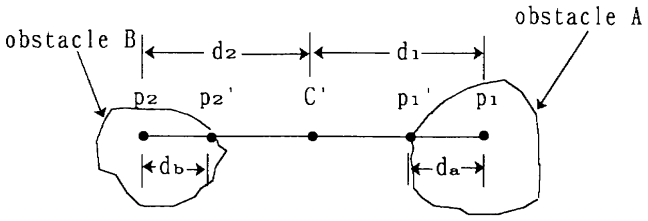


Fig. 2.4. Objects are moving toward C'.

Let $\vec{a}$ and $\vec{b}$ be the vector from C' to p_1 and C' to p_2, respectively. If no external force is acting on $\overrightarrow{c'p_1}$, then $\vec{a} = \overrightarrow{c'p_1}$. if external forces exist so that p_1 is moved to p_1', then $\vec{a} = \overrightarrow{c'p_1'}$. Similarly, $\vec{b} = \overrightarrow{c'p_2}$ when no external force exists, and $\vec{b} = \overrightarrow{c_2'p_2'}$ when external forces exist.

Define a vector-distance function $\overrightarrow{F_1}$ on C' as

$$\overrightarrow{F_1} = w_1\vec{a} + w_2\vec{b} \tag{2.6}$$

where w_1 and w_2 are weighting coefficients.

Similarly, if there are a total number of n external forces acting on C', then the vector-distance function $\vec{F}$ on C' is

$$\vec{F} = \sum_{i=1}^{n} \vec{F}_i = \sum_{i=1}^{n} (w_{i1}\vec{a}_i + w_{i2}\vec{b}_i) \tag{2.7}$$

Since w_{i1} and w_{i2} are relative relation, without loss of generality, we can assume $w_{i2} = 1$. Hence (2.7) becomes

$$\vec{F} = \sum_{i=1}^{n} (w_{i1}\vec{a}_i + \vec{b}_i) \tag{2.8}$$

where weighting coefficients w_{i1} are chosen to reflect the importance of its corresponding vector $\vec{a}_i$.

2.2 Application of vector-distance function to motion planning

If a mobile robot is in an unknown environment, its spatial reasoning system must be able to obtain information about the physical world in real-time. This implies that some kind of sensory data processing system, capable of both collecting and interpreting information, be mounted on the mobile robot. These sensing systems are the input to a spatial representation system that will be used by the robot for reasoning about the outside world. A number of different sensors can be used including touch sensors, range finders and vision systems. Tactile sensors are of limited use to a spatial reasoning system because of their short range. Computer vision has been studied extensively and is now sufficiently developed to be of some practical use. The most popular class of sensor in current mobile robot research is the range-finder. The two types of these sensors are time-of-flight ultrasonic acoustic range-finders and scanning laser range-finder. The targetability of ultrasonic sensors is considerably lower than laser sensors. However, the

https://doi.org/10.1017/S0263574797000593 Published online by Cambridge University Press

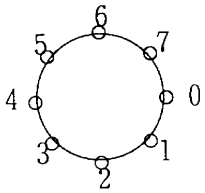


Fig. 2.5. Sensor's locations.

low cast and high reliability make ultrasonic sensors a popular choice.

In our development, the ultrasonic sensor will be used and the shape of the movile robot is assumed to be circular. There are a total number of 8 ultrasonic sensors, each oriented at a fourty-five degrees interval, mounted on the mobile robot as shown in Figure 2.5. Their corresponding vector directions and weighting coefficients are shown in Figure 2.6. The vector-distance function is

$$\vec{F} = \sum_{i=0}^{7} \vec{f}_i w_i \tag{2.9}$$

where

$$\vec{f}_i = \|\vec{f}_i\| \, e^{-j\pi i/4} \quad (i = 0, 1, \ldots, 7) \tag{2.10}$$

Let the safe distance of the mobile robot be d_{sa}, and the obstacle distance that the ith sensor detects be d_{is}. We select $\|\bar{f}_i\|$ by the following

$$\|\bar{f}_i\| = \begin{cases} d_{sa} & \text{if } d_{is} \geq d_{sa} \\ d_{is} & \text{if } d_{is} < d_{sa} \end{cases} \tag{2.11}$$

The weighting coefficients w_i are selected to represent the distance between the ith sensor and the target position. Those sensors who are more close to the target than the others shall have larger w_i's. Assume the "o"th sensor is the closest sensor to the target T. Then w_0 will be the largest. It should be noted that if weight coefficients to the left of w_0 and to the right of w_0 are symmetry with respect to the motion direction (axis w_0 and w_4) and their corresponding weights are equal (i.e. $w_1 = w_7$, $w_2 = w_6$, $w_3 = w_5$), then the mobile robot will not move when it faces a obstacle that is too large to find its flanks. In this cases, the mobile robot must turn left or right and track along the obstacle's boundary to avoid them. To accomplish a left turn mode. We simply select weighting coefficient to the left of motion axis ($\overrightarrow{w_4 w_0}$) larger than those to the right of the motion axis. Likewise, a right turn can be implemented *vice versa*. Let the target T be along the $\vec{f}_0$ direction, then Figure 2.7a and Figure 2.7b represent the right turn mode and left turn mode, respectively, where weighting coefficients satisfy

$$\begin{aligned} &w_0' > w_i' \quad (i = 1, 2, \ldots, 7), \\ &w_1' > w_7', \quad w_3' \geq w_5', \quad w_2' \geq w_6', \\ &w_i' > w_2' \geq w_3' \geq w_4' \end{aligned} \tag{2.12}$$

It should be noted that the weighting coefficients of the left turn mode and right turn mode will be varying as the target position T is changed. For example, when a right turn mode is employed, the relation between new weighting coefficient w_i' and old weighting coefficient w_i is shown in Figure 2.8a when the target T is on the $\vec{f}_0$ axis (Figure 2.8a). Likewise, the relation becomes Figure 2.8b and Figure 2.8c when the target T is on the $\vec{f}_i$ axis and $\vec{f}_b$ axes, respectively.

For the left-turn mode, as given by Figure 2.7b, the vector-distance function $\vec{F}$ is given by

$$\vec{F} = \vec{f}_0 w_0' + \vec{f}_1 w_7' + \vec{f}_2 w_6' + \vec{f}_3 w_5' + \vec{f}_4 w_4' + \vec{f}_5 w_3' + \vec{f}_6 w_2' + \vec{f}_7 w_1' \tag{2.13}$$

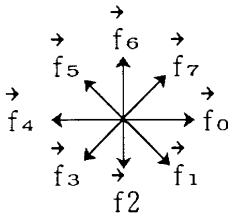


Fig. 2.6.1. Vectors corresponding to sensor locations.

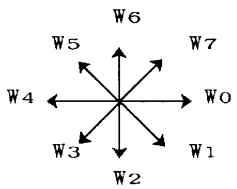


Fig. 2.6.2. Weighting coefficients corresponding to sensor locations.

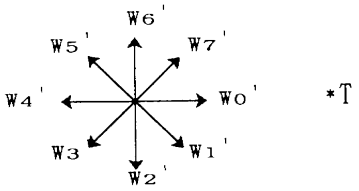


Fig. 2.7a. Right turn mode.

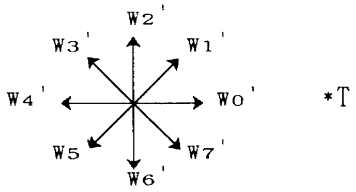


Fig. 2.7b. Left turn mode.

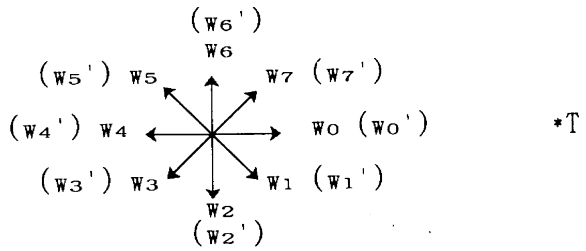


Fig. 2.8a. Relations between W' (new weights) and W (old weights) when the target is along the $\vec{f}_0$ direction.

https://doi.org/10.1017/S0263574797000593 Published online by Cambridge University Press

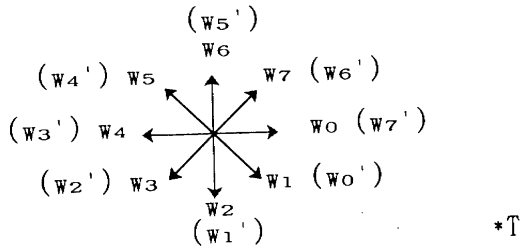


Fig. 2.8b. When the target is along the $\vec{f}_1$ axis.

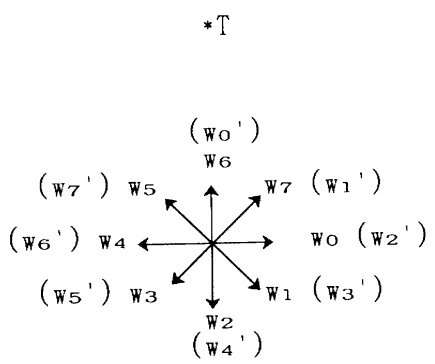


Fig. 2.8c. When the target is along the $\vec{f}_2$ axis.

Likewise for the right-turn mode, as given by Figure 2.7a, the vector-distance function can be expressed as

$$\vec{F} = \vec{f}_0 w_0' + \vec{f}_1 w_1' + \vec{f}_2 w_2' + \vec{f}_3 w_3' + \vec{f}_4 w_4' + \vec{f}_5 w_5' + \vec{f}_6 w_6' + \vec{f}_7 w_7' \tag{2.14}$$

Substituting (2.10) into (2.9), we have

$$\vec{F} = \|\vec{F}\| e^{j\theta} \tag{2.15}$$

where

$$\|\vec{F}\| = \sqrt{R^2 + I^2}, \quad \theta = \tan^{-1}\left(\frac{I}{R}\right) \tag{2.16}$$

$$R = \|\vec{f}_0\| w_0 - \|\vec{f}_4\| w_4 + \cos\left(\frac{\pi}{4}\right) \times \{\|\vec{f}_1\| w_1 - \|\vec{f}_3\| w_3 - \|\vec{f}_5\| w_5 + \|\vec{f}_7\| w_7\} \tag{2.17}$$

$$I = \|\vec{f}_6\| w_6 - \|\vec{f}_2\| w_2 + \sin\left(\frac{\pi}{4}\right) \times \{\|\vec{f}_5\| w_5 + \|\vec{f}_7\| w_7 - \|\vec{f}_1\| w_1 - \|\vec{f}_3\| w_3\} \tag{2.18}$$

The resultant direction of $\vec{F}$ indicates the moving direction in order to avoid colliding with obstacles.

To simplify the problem, we let $w_2' = w_3' = w_4' = w_5' = w_6' = 1$, while w_0', w_1' and w_7' can be determined from equation (2.5). For instance, if w_0' allows obstacles to exist within 20% of its safe distance, then from equation (2.5),

$$\frac{d_a}{d_{sa}} = 0.2 = 1 - \frac{1}{w_0'} \tag{2.19}$$

Hence, $w_0' = 1.25$. Likewise, w_1' and w_7' can be determined.

2.3 The moving speed of mobile robot

It is often desired to move the mobile robot so as to reach the target T as soon as possible. Consequently, the mobile robot must move almost at its maximum speed, while at the same time, not colliding with obstacles. As a result, we must be able to determine the relative distance and speed between the mobile robot and obstacles so that the mobile robot will be able to move as quickly as possible.

Consider a mobile robot moving along the X-axis as shown in Figure 2.9. Assume the friction coefficient f_s between the mobile robot and the ground is constant. Let W be weight of the mobile robot, N be the force vector along Y-axis. Let the mobile robot be moving with constant acceleration or deceleration. Then the distance for the mobile robot to decelerate from speed v_0 to zero speed is

$$x = -\frac{v_0^2}{2a} \tag{2.20}$$

where the minus sign denotes the acceleration is along the negative X-axis. Along the x-axis, we have

$$-f_s = ma = \frac{wa}{g}$$

or

$$a = -\frac{gf_s}{w} \tag{2.21}$$

Similarly, along the Y-axis, we have

$$N = W$$

and

$$f_s = Nu_s = Wu_s \tag{2.22}$$

where u_s is the friction coefficient between the object and the contact plane. Hence

$$a = -gu_s \tag{2.23}$$

and (2.20) becomes

$$x = \frac{v_0^2}{2gu_s} \tag{2.24}$$

Thus the minimum distance x that is required for the mobile robot to stop can be obtained by using the maximum friction coefficient in (2.24). Furthermore, in

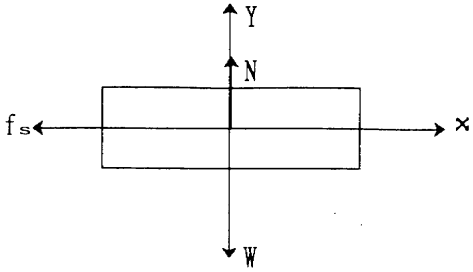


Fig. 2.9. The force and the friction on the mobile robot when it is moving along the x-axis.

https://doi.org/10.1017/S0263574797000593 Published online by Cambridge University Press

order for the mobile robot to have enough distance and time to respond to the environment while moving at the maximum speed Vmax, it is required that the vehicle sensor range S must satisfy

$$s \geq \frac{v_{\max}^2}{2gu_s} \tag{2.25}$$

3. THE PROPOSED OBSTACLE AVOIDANCE ALGORITHM

Several algorithms have been proposed for motion planning in an unknown environment. In particular, Lumelsky and Stepanov[15] developed Bug 1 and Bug 2 algorithms to generate a path toward the target position. Miller[16] proposed to turn right and move along the boundary to across the obstacle in front of moving direction. Zhu[17] used a direction priority list for motion direction. A major drawback of preassigned direction priority is that when the mobile robot faces a large (long in width) obstacle, in general, the mobile robot will move from left to right, then from right to left, back and forth. As a result, the robot cannon find a way out. Recently, Huang and Lee[14] developed the Track-mode (T-mode), which is similar to the Bug 2 of reference [14], for the mobile robot to track along the obstacle's boundary. In the algorithm, they need to calculate 9 decision rules for tracking the obstacle in counter-clockwise direction. For clockwise direction tracking, another 9 decision rules need to be calculated.

In order to avoid these problems, in this paper, we proposed a obstacle avoidance algorithm which consists of three modes: (1) the rotate mode (R-mode), (2) the lock mode (L-mode), and (3) the track mode (T-mode). Incorporating the vector distance function with these three modes, a mobile robot is capable of moving toward the target position, and finally reach the target position even in an unknown environment with moving obstacles.

3.1 The rotate mode (R-mode)

The rotate mode is to rotate the $\vec{f}_0$ direction (and hence the largest weighting coefficients w_0') toward the target, and then calculate the vector distance function. The resultant direction will be the appropriate direction for the mobile robot to move forward. The mobile robot will use the R-mode and move along this direction until one of the following occurs: (1) the target position is reached, then it stops; or (2) the obstacle is within the safe distance (d_{sa}), therefore we switch to use the lock mode. It should be noted that the mobile robot shall decide whether to use the left-turn mode or to use the right-turn mode by itself. The decision is made by substituting both left-turn mode weighting coefficients and the right-turn mode weighting coefficients to the vector-distance function (2.9), and comparing their corresponding values; which one yields larger value will be chosen to use.

Note that the rotate-mode will face the same problem as that of preassigned direction priority[17] when the obstacle is too large to find its flanks. In this case, the mobile robot will move back and forth and can not find a path out. To avoid this problem, when the obstacle is within the safe distance d_{sa}, we switch from the R-mode to the lock-mode (L-mode).

3.2 The lock-mode (L-mode)

In this mode, the mobile robot fixed its weighting coefficients regardless of the direction toward the target. The weighting coefficients will not vary, following the changes of the motion direction. The weighting coefficients will remain fix until the sensor corresponding to the largest weighting coefficient w_0' does not detect any obstacles within its safe distance. In this situation, the lock-mode is relaxed. Figure 3.1 shows how the lock-mode is used to avoid a obstacle. Notice that when the sensor w_0' detects a obstacle within its safe distance d_{sa}, Figure 3.1a, it begins to use the L-mode such that all the weighting coefficients remain fix and the mobile robot will move along w_1' direction (Figure 3.1b). When there is no obstacle within the safe distance of w_0', as shown in Figure 3.1c, then the lock-mode is relaxed and the R-mode is adopted to rotate w_0' toward the target T. Figure 3.1d shows how the R-mode is employed to rotate w_0' toward the target T and the mobile robot moves directly toward the target T. The resultant path is shown in Figure 3.1e.

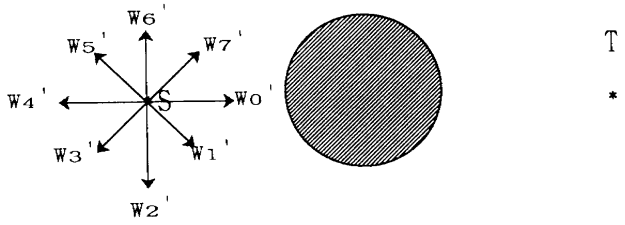


Fig. 3.1a. Motion sequence of the lock mode.

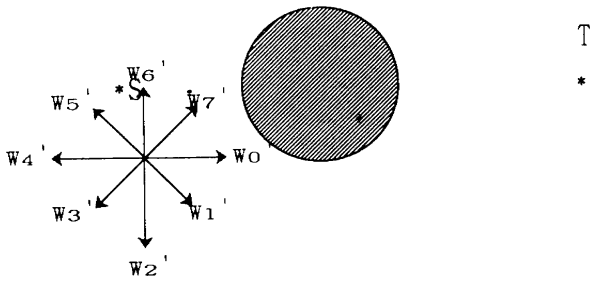


Fig. 3.1b. Motion sequence of the lock mode.

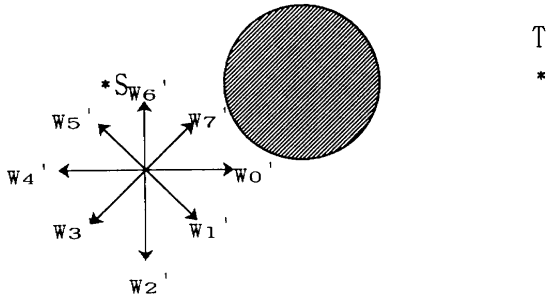


Fig. 3.1c. Motion sequence of the lock mode.

https://doi.org/10.1017/S0263574797000593 Published online by Cambridge University Press

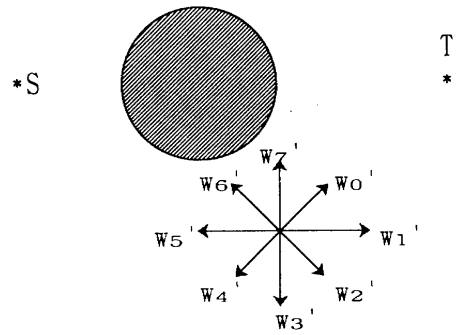


Fig. 3.1d. Motion sequence of the lock mode.

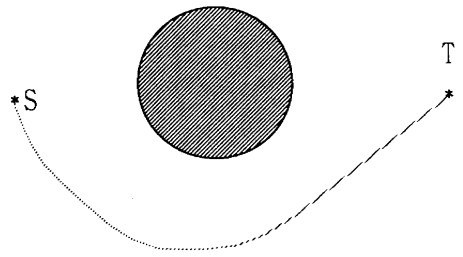


Fig. 3.1e. The complete trajectory of the lock mode.

Note that the L-mode is good for simple obstacles, such as convex and/or slightly concave obstacles. For a complex environment, or in front of a highly concave obstacle, the L-mode and the track mode (T-mode) should be used. The following shows when the T-mode should be employed:

Let the present position be at Q_j, and the distance to the target T be $\overline{Q_jT}$. After n steps, the position becomes Q_{j+n}, and the distance to the target T is $\overline{Q_{j+n}T}$. If

$$\overline{Q_jT} - \overline{Q_{j+n}T} < c_1 \quad (\text{where } c_1 > 0) \tag{3.1}$$

then we switched to use the track mode.

3.3 The track mode (T-mode)

The proposed track mode (T-mode) is similar to the Bug 2 proposed by Lumelsky[15] and also similar to the track mode presented by Huang and Lee.[14] There the track mode is employed between the hit point H and the leave point L as follows:

The hit point H on an obstacle boundary is the point that the mobile robot meets the obstacle while manoeuvering toward the target. After the hit point, the mobile robot should move along the boundary of the obstacle with a safe distance. The point that the mobile robot leaves the obstacles boundary and starts moving in free space toward the target is called the leave point L. The leave point L exists if the leave point L, the hit point H, and the target point T all lie on a straight line, where L is somewhere between H and T. That is, the leave point exists if

$$\angle HLT = 180° \tag{3.2}$$

For stationary obstacles, the hit point H and leave point L must be found in pairs unless the solution does not exist. However, for moving obstacles, the satisfaction of equation (3.2) does not guarantee that the mobile robot will approach the target T after passing a series of hit-leave point pairs. The T-mode presented in this paper can deal with dynamic environments with both stationary and moving obstacles. Furthermore, it does not require the assumption that the speed of the mobile robots should be greater than that of any moving obstacles. Specifically, the proposed T-mode allows the mobile robot to track along the obstacle's boundary with a prescribed safe distance from obstacles. The robot's forward direction is determined by the vector-distance function generated from sensory feedback.

During the track-mode, let the distance between present position Q_j and the target T be $\overline{Q_jT}$. After m steps, let the position Q_{j+m} and the distance to the target T be $\overline{Q_{j+m}T}$. If

$$\overline{Q_jT} - \overline{Q_{j+m}T} < c_2, \quad (\text{where } c_2 > 0) \tag{3.3}$$

Then the T-mode is not effective to track the boundary of obstacle and to find a way out. Hence the T-mode is stopped.

The major concept of the proposed track-mode is described as follows:

Step (1) If the sensor w_1 detects a obstacle (either stationary or moving) within its safe distance, then rotate the weighting coefficients clockwise (for right turn mode) or counter-clockwise (for left turn mode) one step, until there is no obstacle within the safe distance of w_1. Go to step 2.

Step (2) If the sensors w_0 and w_7 do not detect any obstacles within its safe distance, then rotate the weighting coefficients counter-clockwise (for the right-turn mode) or clockwise (for the left-turn mode) one step.

Step (3) When (3.1) or (3.3) is satisfied, or all sensors do not detect any obstacles within their safe distance, then stop the T-mode.

In summary, the proposed algorithm of obstacle avoidance consists of a rotate mode, a lock mode and a track mode. The robot can avoid obstacles efficiently in a simple workspace by the rotate mode, or possibly by the rotate mode and the lock mode. In a complex environment with possible moving obstacles, however, all of the rotate mode, the lock mode and the track mode should be used.

Let

d_{sa}: safe distance from obstacles
d_0: the distance from the sensor w_0' to the target T
d_{is}: the distance from the sensor w_i ($i = 0, 1, \ldots, 7$) to the target T
j: integer, a index for forward steps
i: integer, a index for forward steps during the lock mode or the track mode
n: a constant, preassigned for deciding whether to switch to the T-mode from the R-mode or not after n steps in the R-mode.
m: a constant, preassigned to determine whether to

https://doi.org/10.1017/S0263574797000593 Published online by Cambridge University Press

stop the T-mode or not after m steps in the T-mode

The proposed algorithm is summarized as follows:

(1) Initialization, $j = 1$, current position is the starting position S.
(2) R-mode: From current position rotate w_0' toward T and use (2.9) to decide the moving direction. Then move forward target T. Each step toward the target, j is increased by 1 until one of the following occurs:
   (a) Target is reached, stop.
   (b) If $d_0 < d_{sa}$, to to step 3
(3) L-mode: Lock the weighting coefficients and the current position is Q_j. Calculate (2.9) to decide which direction to go. Let $i = 1$. Each L-mode movement both i and j are increased by 1, respectively. This process continues until one of the following occurs.
   (a) Target is reached, stop.
   (b) If $d_0 > d_{sa}$, go to step 2.
   (c) If $i = n$, and (3.1) is satisfied. Then define the current point as a hit point H. Go to step 4.
(4) T-mode: current position is Q_j and calculate (2.9) to determine the forward direction and track along the boundary. Let $i = 1$. Each T-mode movement, both i and j are increased by 1, respectively. This process continues until one of the following occurs.
   (a) The target is reached, stop.
   (b) Either (3.2) is satisfied, or $d_{is} > d_{sa}$. Define a leave point L. Go to step 2.
   (c) If $i = m$ and (3.3) is satisfied, or the mobile robot returns to H. The obstacle-free path cannot be found, stop.

4. SIMULATION RESULTS

Let the workspace be represented by a two-dimensional grid map with 200×200 pixels. The height and width of each grid is one unit length (one pixel). Let each step be one unit length $= 10^{-3}$ m and let the friction coefficient be $u_s = 0.6$, and the constant acceleration $a = 5.886\ \text{m/sec}^2$. In addition, the range sensor is stimulated by a straight line search method, i.e. we have assumed that the range sensor has perfect directional property, and it will not be affected by the distribution of obstacles.

In the figures shown below (Fig. 4.1.1–4,4), the unit for time is seconds (sec), for distance is meter (m), for velocity is m/sec.

4.1 Stationary obstacles

Case 1. (Left turn mode)

The environment has several small and convex obstacles as shown in Figure 4.1.1. The starting point is S (20, 20) and the target point is T (180, 180). The parameters are $d_{sa} = 0.02$ m, $v_{\max} = 0.8$ m/sec, $s_{\max} = 0.06$ m, $w_0' = 1.25$, $w_1' = 1.2$, $w_7' = 1.15$, $w_2' = w_3' = w_4' = w_5' = w_6' = 1$. Furthermore, if the vector distance function for the right turn R-mode is equal to that of the left-turn R-mode, then we choose left turn. The velocity and the time at which the result is shown are included in the figure. In addition, the scenario of sensors detect obstacles are also shown on the right side of the figure. Figure 4.1.1 to Figure 4.1.11 shows the snapshot of the result at each particular instant. Figure 4.1.12 shows the complete trajectory, and Figure 4.1.13 shows the velocity of the mobile robot in the time history. The total travel time is $t = 0.792$ sec, the total path is $d = 0.4516$ m, and the average speed is $v_{av} = 0.57$ m/sec.

Case 2. (Right turn mode)

In this case, the starting point, the target point, and the parameters are all the same as that of case 1. The only difference is that when the vector distance function is unable to decide a right turn or a left turn, we arbitrarily choose right turn instead of the left turn. The resultant path and the velocity history are shown in Figure 4.2. The total travel time is $t = 0.6$ sec, the total path is $d = 0.3355$ m, and the average speed is $v_{av} = 0.559$ m/sec.

Case 3. (Left turn mode with small safe distance)

This case is the same as that of case 1, except $d_{sa} = 0.015$ m. The resultant path and its velocity history are shown in Figure 4.3. The total travel time is $t = 0.628$ sec, the total path is $d = 0.3372$ m, and the average speed is $v_{av} = 0.537$ m/sec.

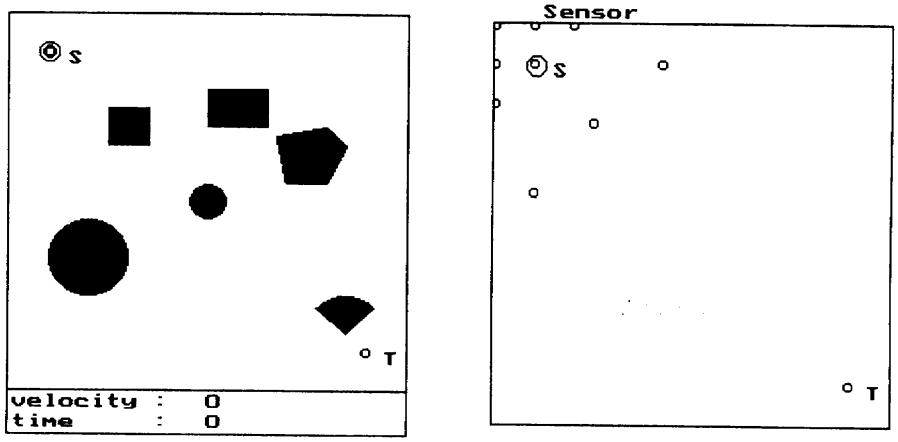


Fig. 4.1.1. The workspace.

https://doi.org/10.1017/S0263574797000593 Published online by Cambridge University Press

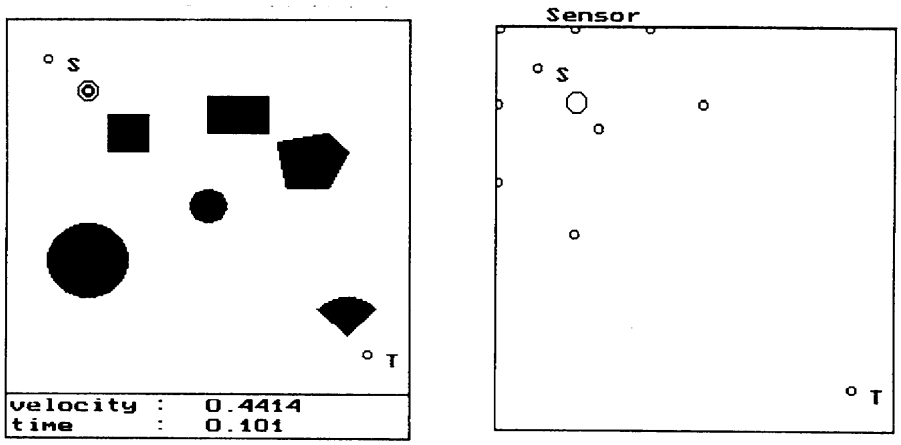


Fig. 4.1.2. The snapshot of robot's motion and detected obstacles.

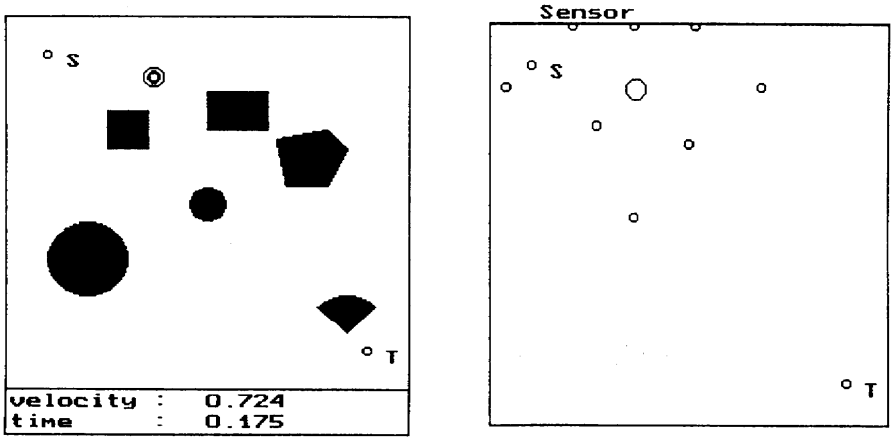


Fig. 4.1.3. The snapshot of robot's motion and detected obstacles.

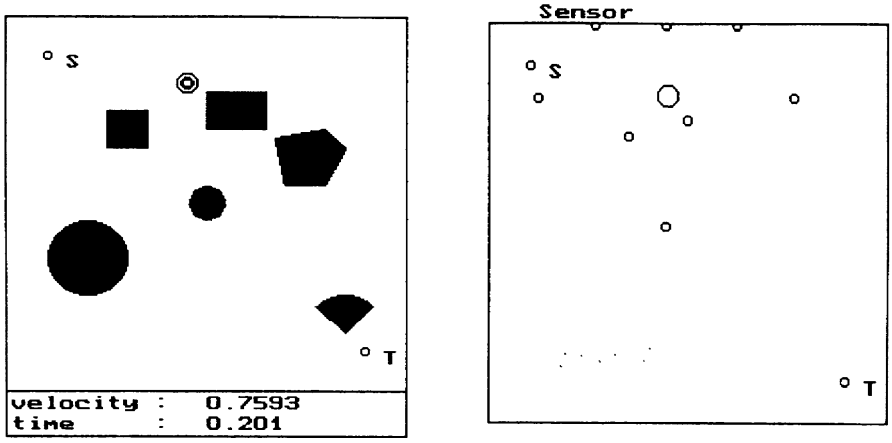


Fig. 4.1.4. The snapshot of robot's motion and detected obstacles.

https://doi.org/10.1017/S0263574797000593 Published online by Cambridge University Press

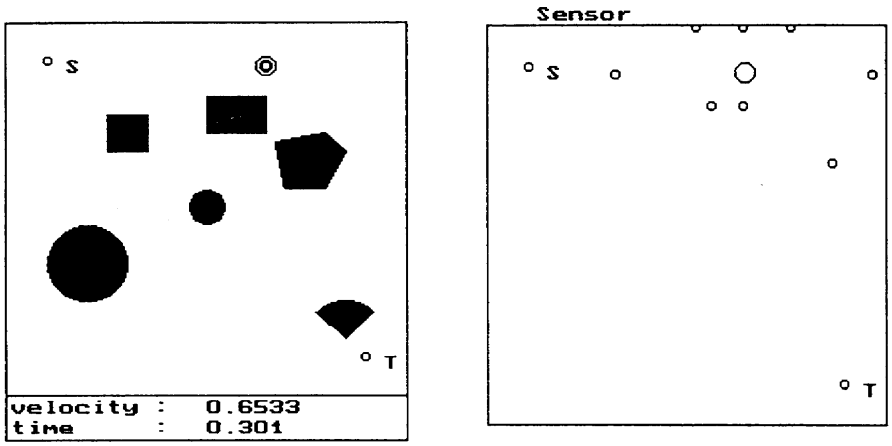


Fig. 4.1.5. The snapshot of robot's motion and detected obstacles.

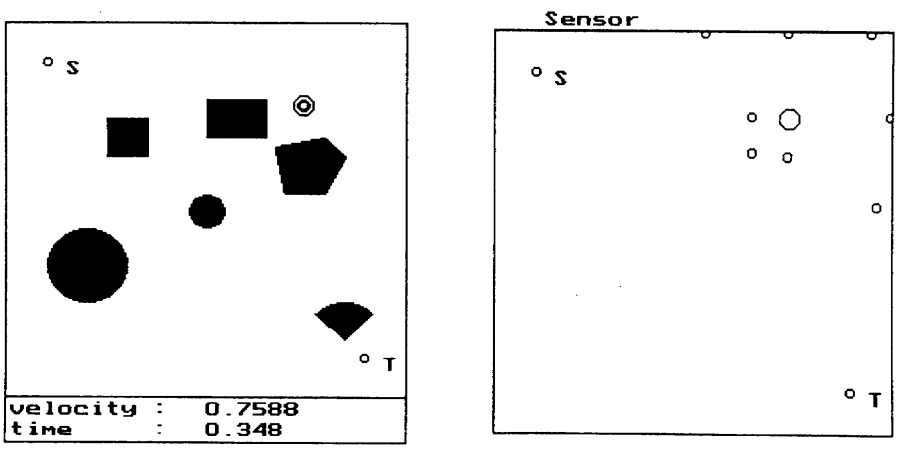


Fig. 4.1.6. The snapshot of robot's motion and detected obstacles.

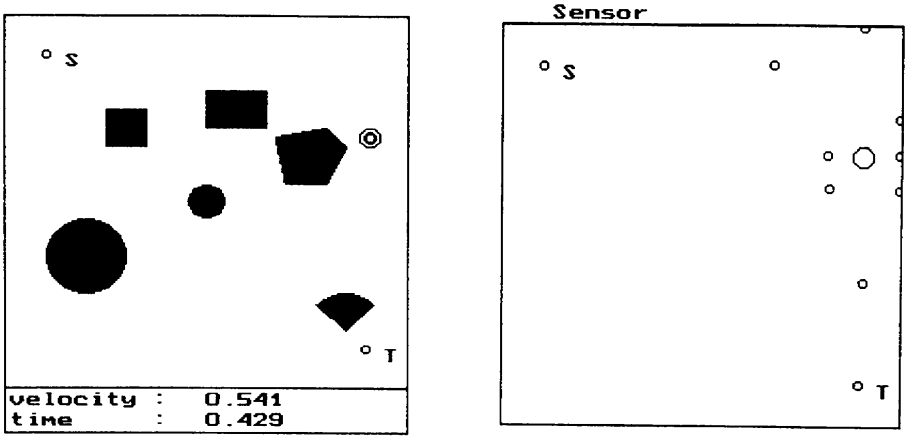


Fig. 4.1.7. The snapshot of robot's motion and detected obstacles.

https://doi.org/10.1017/S0263574797000593 Published online by Cambridge University Press

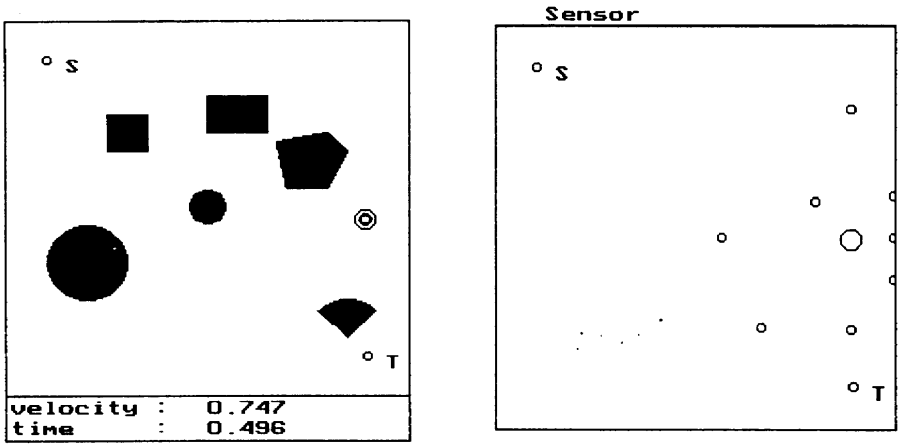


Fig. 4.1.8. The snapshot of robot's motion and detected obstacles.

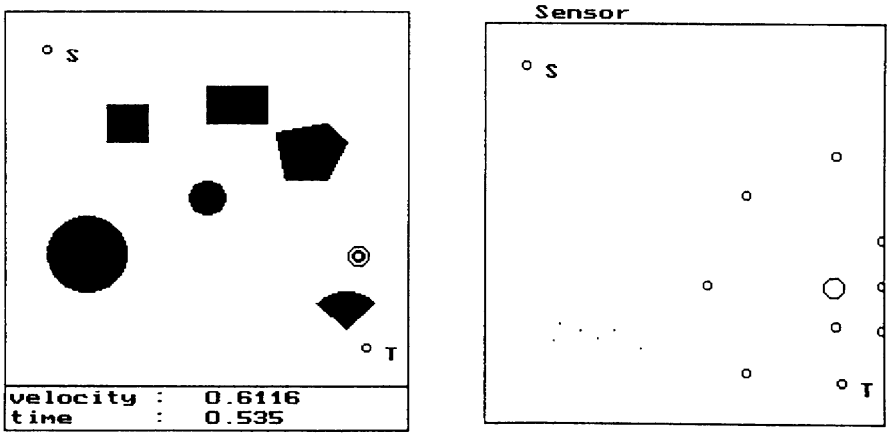


Fig. 4.1.9. The snapshot of robot's motion and detected obstacles.

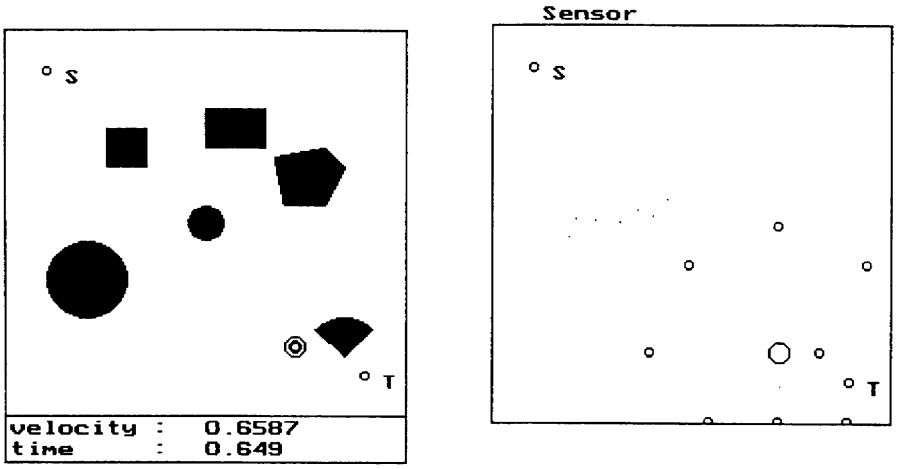


Fig. 4.1.10. The snapshot of robot's motion and detected obstacles.

https://doi.org/10.1017/S0263574797000593 Published online by Cambridge University Press

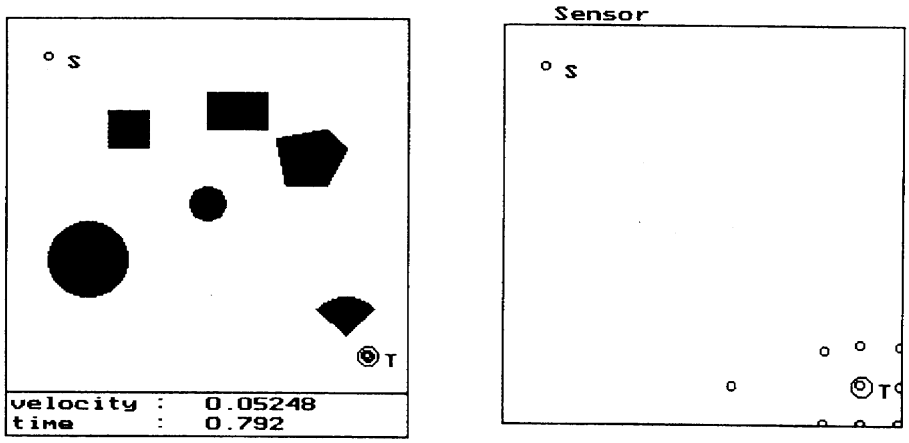


Fig. 4.1.11. The snapshot of robot's motion and detected obstacles.

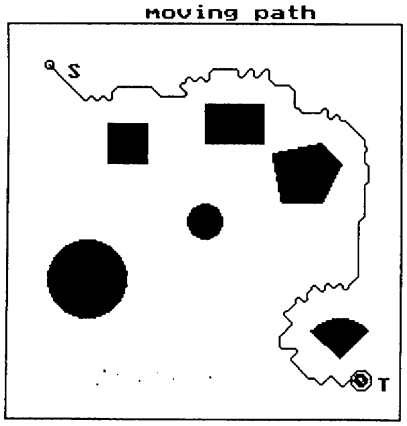


Fig. 4.1.12. The complete path of the mobile robot.

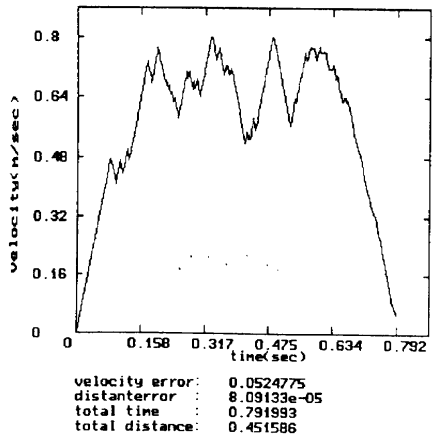


Fig. 4.1.13. The velocity vs time of the mobile robot.

Case 4. (Right turn mode)

This case is the same as that of case 2, except $d_{sa} = 0.015$ m. The resultant path and its velocity history are shown in Figure 4.4. The total travel time is $t = 0.506$ sec, the total path is $d = 0.2666$ m, and the average speed is $v_{av} = 0.527$ m/sec.

For comparison, these results are summarized in Table I.

As expected, reducing the safe distance d_{sa} will reduce the travel path and the travel time. Note that case 4 has the lowest average speed as compared to the other 3 cases. However, its total travel time is also the smallest. This means that increasing the speed of mobile robot does not guarantee that the total travel time can be reduced. Instead, if a shortest path can be found, the traveling time might be the smallest even if the robot is not moving with the highest average speed.

4.2 Dynamic environments

Case 5.

The environment consists of both stationary and moving obstacles. The velocity and motion direction of each moving obstacles are different. In Figure 4.5.1, the velocity of the circular obstacle is 0.5 m/sec, and are 0.4 m/sec and 0.3 m/sec, for the sector type obstacle and the square type obstacle, respectively. We assume a poor sensor is used so that it can only detect obstacles inside the range of 0.06 m. In addition, let $d_{sa} = 0.02$ m, $v_{\max} = 0.8$ m/sec, $s_{\max} = 0.06$ m, $w_0' = 1.25$, $w_1' = 1.2$, $w_7' = 1.15$, and $w_i' = 1$ $(i = 2, 3, 4, 5, 6)$. The starting point is S $(20, 90)$ and the target point is T $(180, 100)$. Figures 4.5.1–4.5.10 shows how the mobile reacts with the dynamic environments. Notice that in Figure 4.5.4, at that particular instant, the mobile robot has reached its maximum velocity, i.e. $v = 0.8$ m/sec. Since then, it almost moves with a maximum velocity during the time interval of $[0.153, 0.257]$, i.e. from Figure 4.5.4 to Figure 4.5.8. The resultant velocity vs. time is shown in Figure 4.5.12. The total travel time is $t = 0.423$ sec and the total path is $d = 0.2276$ m. Observing the velocity profile of the mobile robot as shown in Figure 4.5.12, we may say that the proposed algorithm is very efficient and robust.

Note that in Figure 4.5.2, the result of vector distance function is to decide to turn left, which is apparently a correct decision. If the right turn mode is preassigned, such as those algorithms of Miller[16] and Zhu[17] that have preassigned priority, then the total travel path and the travel time would become considerable large. Furthermore, the algorithm of Huang and Lee[14] requires to

https://doi.org/10.1017/S0263574797000593 Published online by Cambridge University Press

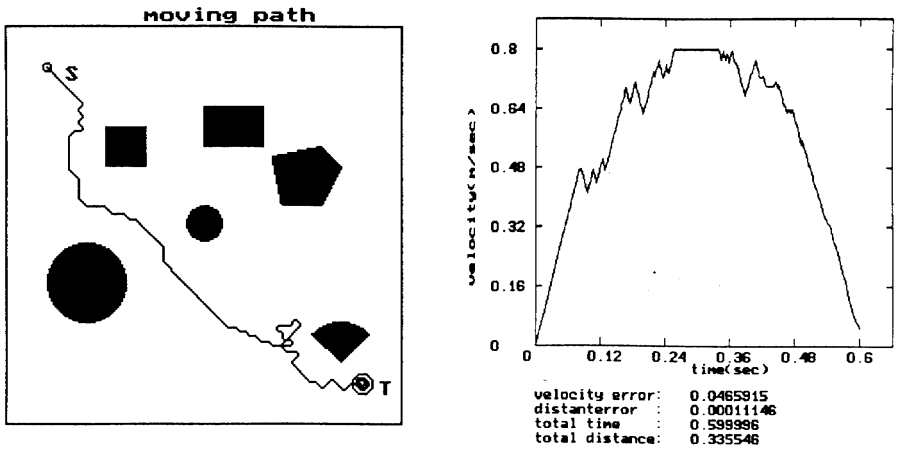


Fig. 4.2. Stationary obstacles (case 2).

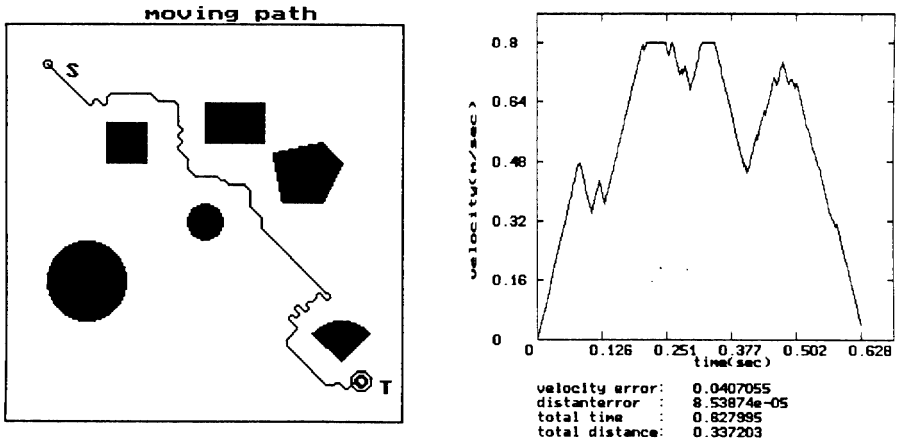


Fig. 4.3. Stationary obstacles (case 3).

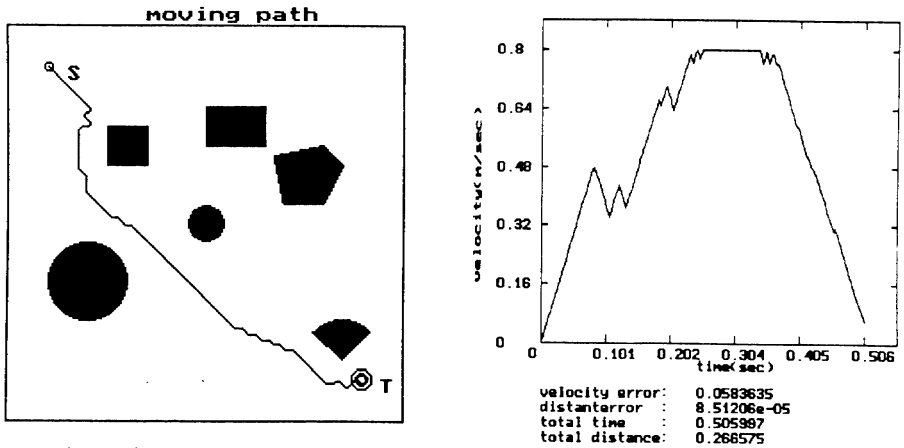


Fig. 4.4. Stationary obstacles (case 4).

Table I. Comparison of simulation results

Figure	d (m)	t (sec)	v_{av} (m/sec)
4.1	0.4516	0.792	0.57
4.2	0.3355	0.6	0.559
4.3	0.3372	0.628	0.537
4.4	0.2666	0.506	0.527

calculate 9 decision rules for tracking the obstacle in counter-clockwise direction, and likewise another 9 decision rules for clockwise direction. In the proposed approach, only one equation, i.e. equation (2.9), needs to be calculated to determine the direction for the next move. Hence the computational effort can be greatly reduced.

https://doi.org/10.1017/S0263574797000593 Published online by Cambridge University Press

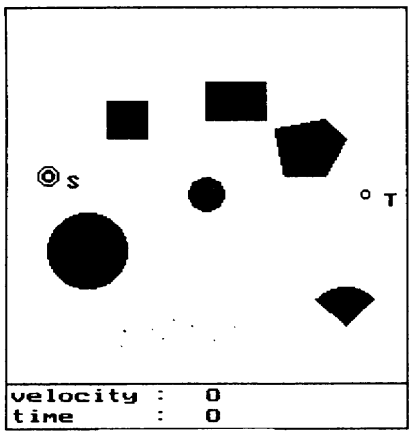


Fig. 4.5.1. The work space.

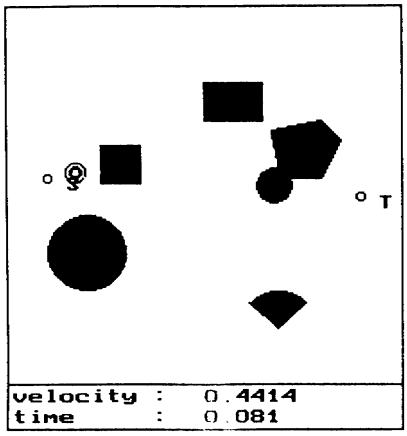


Fig. 4.5.2. The snapshot of how robot reacts with moving obstacles.

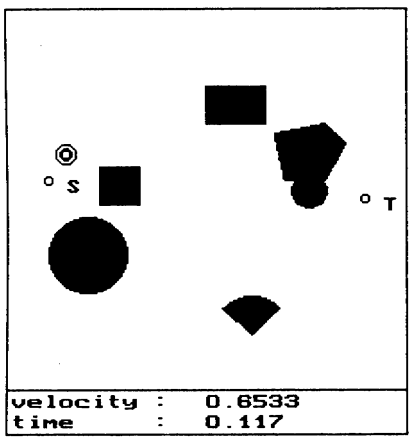


Fig. 4.5.3. The snapshot of how robot reacts with moving obstacles.

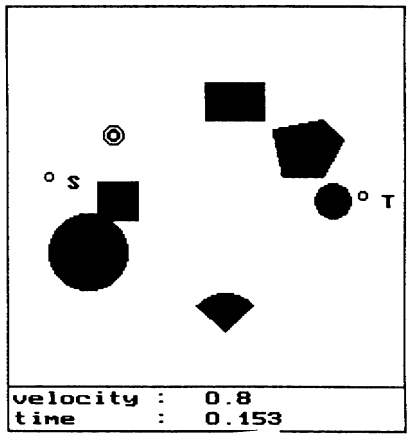


Fig. 4.5.4. The snapshot of how robot reacts with moving obstacles.

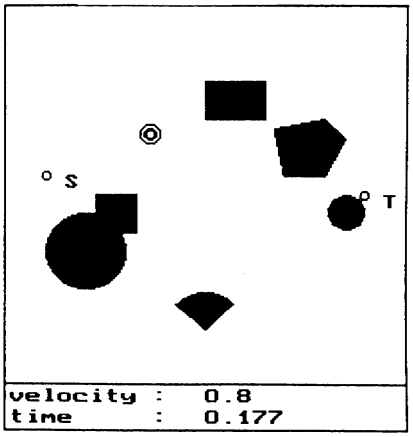


Fig. 4.5.5. The snapshot of how robot reacts with moving obstacles.

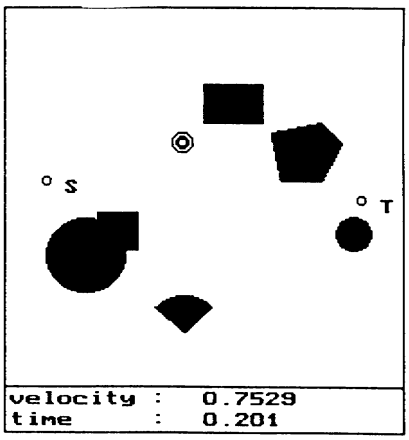


Fig. 4.5.6. The snapshot of how robot reacts with moving obstacles.

Case 6.

The working environment is a room-like environment. The starting position is S (120, 170) and the target position is T (120, 20). A concave obstacle is rotating with angular velocity of 12.22 rad/sec. When it hits the horizontal obstacles, it will bounce back with same angular velocity. The rectangular obstacle in the middle is extending and shrinking in a horizontal direction with a velocity of 0.7 m/sec. Note that, at some particular instants, it may extend to touch the wall on the other side so that the mobile robot is in a box-in situation. Figure 4.6 shows the simulation results. The resultant moving path is shown in Figure 4.6.19, and its velocity profile is given in Figure 4.6.20. The total travel path is

https://doi.org/10.1017/S0263574797000593 Published online by Cambridge University Press

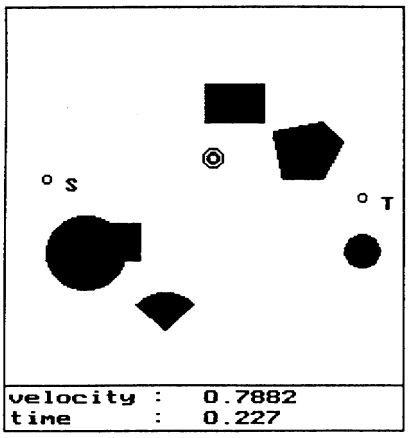


Fig. 4.5.7. The snapshot of how robot reacts with moving obstacles.

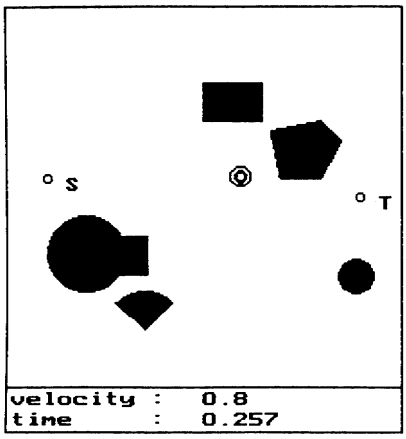


Fig. 4.5.8. The snapshot of how robot reacts with moving obstacles.

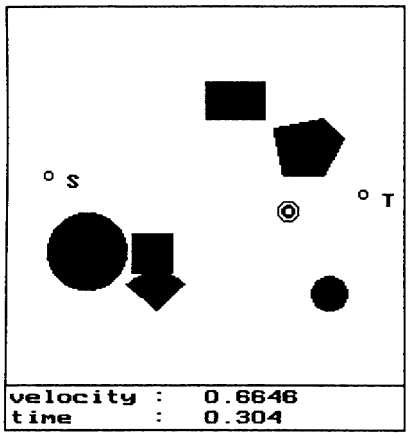


Fig. 4.5.9. The snapshot of how robot reacts with moving obstacles.

$d = 0.516$ m and the total travel time is $t = 0.917$ sec. It should be noted that in Figure 4.6.1, the vector distance function yields a correct decision to move. If instead a wrong decision is made so that the mobile robot is moving in other directions, then either a longer path is travelled to reach the target or the robot cannot find a proper way out to reach the target. In any case, again,

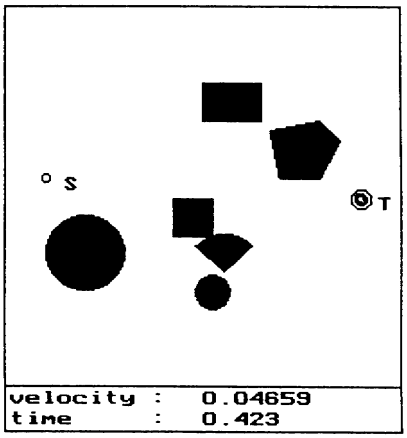


Fig. 4.5.10. The snapshot of how robot reacts with moving obstacles.

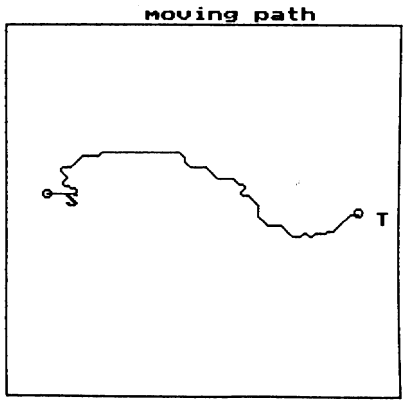


Fig. 4.5.11. The resultant path of the mobile robot.

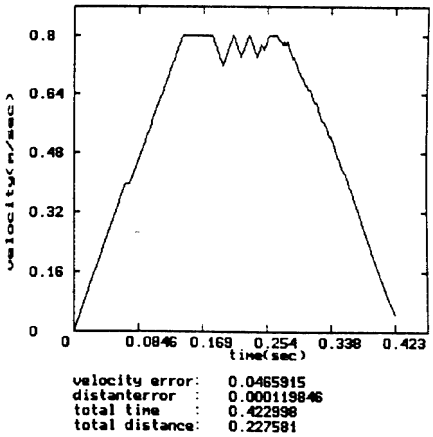


Fig. 4.5.12. The velocity vs time of the mobile robot.

this shows the effectiveness of the vector-distance function in determining motion directions.

5. CONCLUSIONS

The problem of dynamic obstacle avoidance of a mobile robot navigating in an unknown environment has been investigated. As a result, we have developed a computationally feasible aglorithm for obstacle avoidance in the presence of stationary and/or moving obstacles. In our approach, we do not require to predict

https://doi.org/10.1017/S0263574797000593 Published online by Cambridge University Press

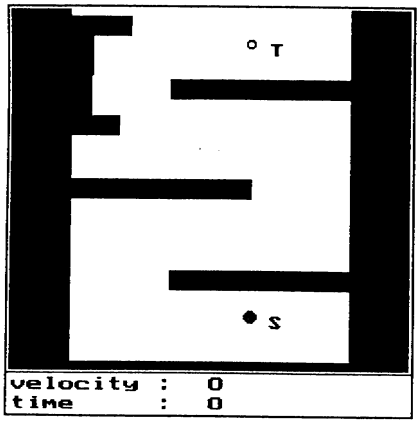


Fig. 4.6.1. The work space.

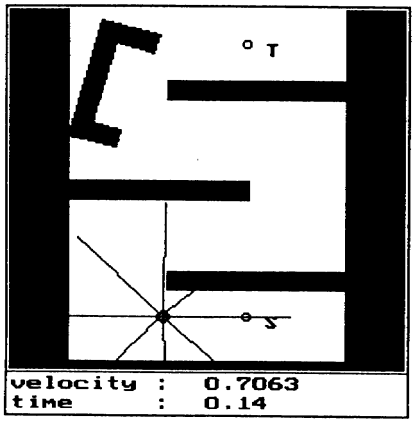


Fig. 4.6.2. The snapshot of how robot reacts with moving obstacles.

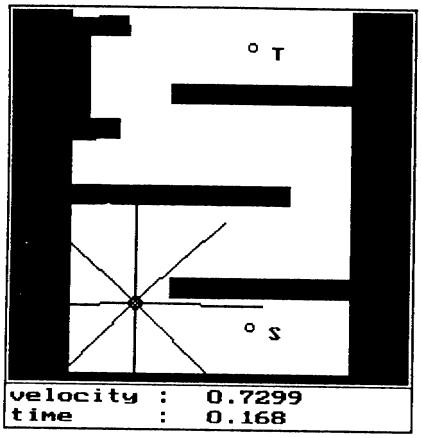


Fig. 4.6.3. The snapshot of how robot reacts with moving obstacles.

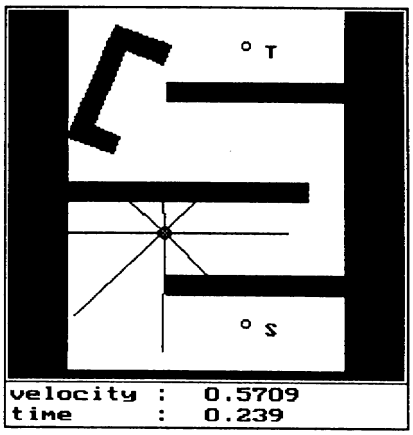


Fig. 4.6.4. The snapshot of how robot reacts with moving obstacles.

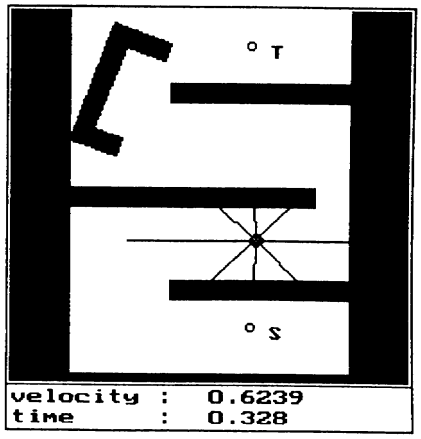


Fig. 4.6.5. The snapshot of how robot reacts with moving obstacles.

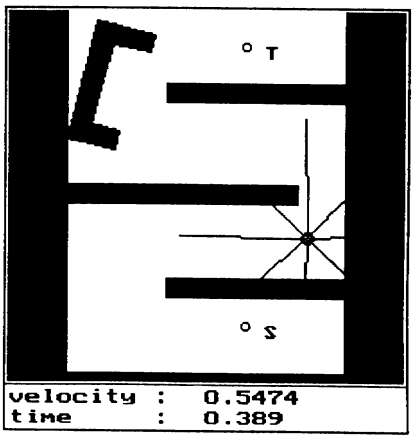


Fig. 4.6.6. The snapshot of how robot reacts with moving obstacles.

the motions of obstacles. Instead, according to the sensory feedback which indicates the location of obstacles, and based on the derived mathematical relations describing the effective range of sensors and the speed of the mobile robot, our method will be able to generate the maximum possible speed at any particular instant, to determine the motion direction, and to move in this direction. The motion direction is determined by a very simple equation expressed by a vector-distance function. The proposes algorithm consists of a rotate mode, lock mode and track mode. In general, for small convex obstacles, the rotate mode is sufficient. For dynamic environments, all the modes are required.

https://doi.org/10.1017/S0263574797000593 Published online by Cambridge University Press

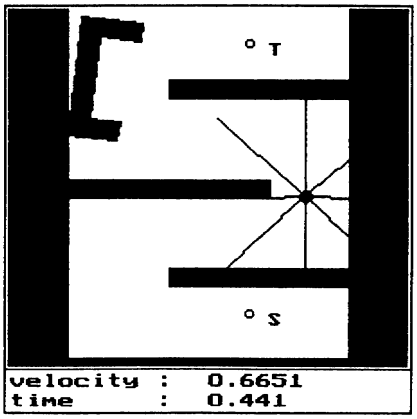


Fig. 4.6.7. The snapshot of how robot reacts with moving obstacles.

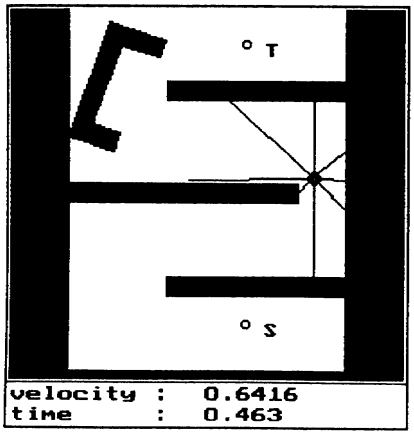


Fig. 4.6.8. The snapshot of how robot reacts with moving obstacles.

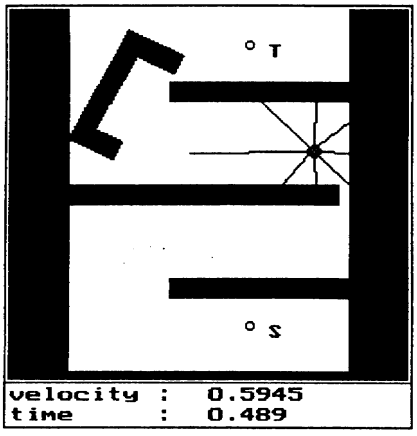


Fig. 4.6.9. The snapshot of how robot reacts with moving obstacles.

Simulation results show that the algorithm can always find a safe path, if it exists, even in the presence of moving and large concave obstacles. Since the computation is simple, it can be implemented in real-time.

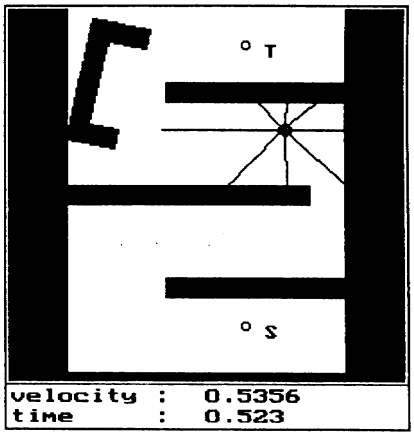


Fig. 4.6.10. The snapshot of how robot reacts with moving obstacles.

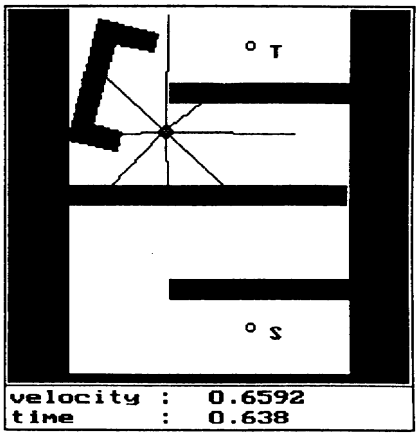


Fig. 4.6.11. The snapshot of how robot reacts with moving obstacles.

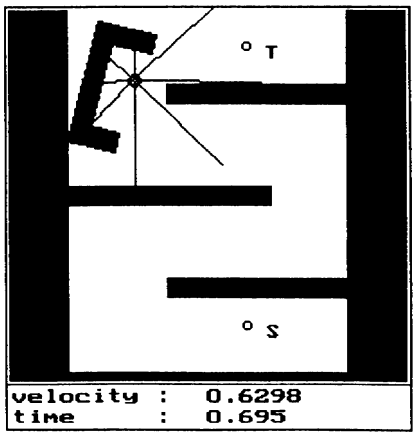


Fig. 4.6.12. The snapshot of how robot reacts with moving obstacles.

ACKNOWLEDGEMENT

The authors would like to thank the National Science Council, Taiwan, Rep. of China, for the support of this research under contract NSC 83-0404-E-011-045.

https://doi.org/10.1017/S0263574797000593 Published online by Cambridge University Press

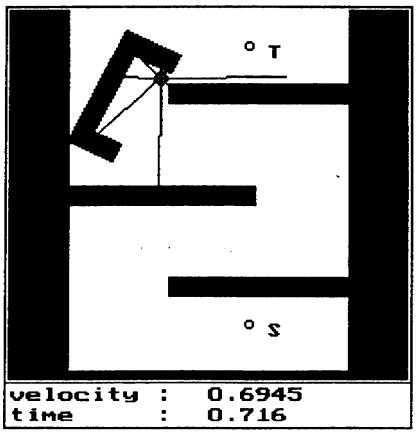


Fig. 4.6.13. The snapshot of how robot reacts with moving obstacles.

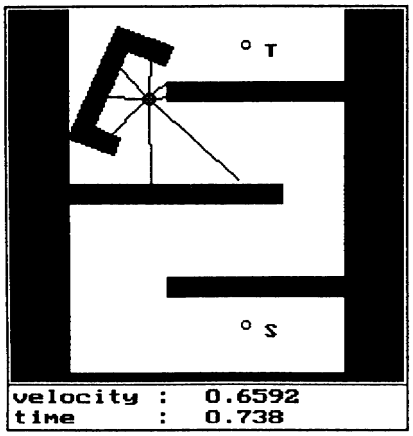


Fig. 4.6.14. The snapshot of how robot reacts with moving obstacles.

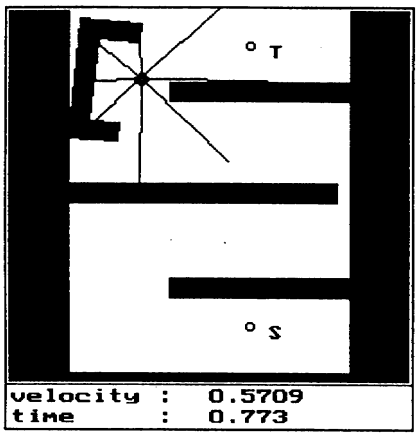


Fig. 4.6.15. The snapshot of how robot reacts with moving obstacles.

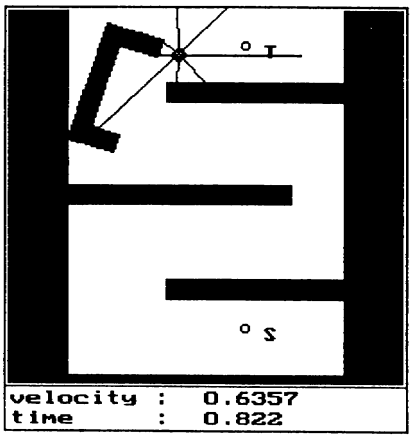


Fig. 4.6.16. The snapshot of how robot reacts with moving obstacles.

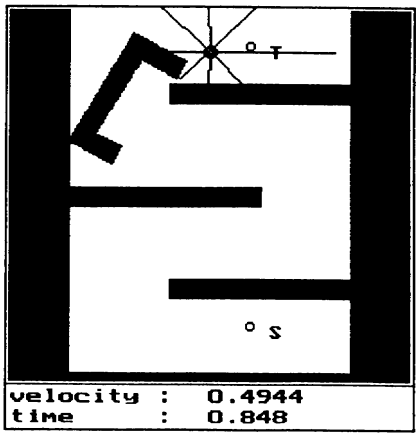


Fig. 4.6.17. The snapshot of how robot reacts with moving obstacles.

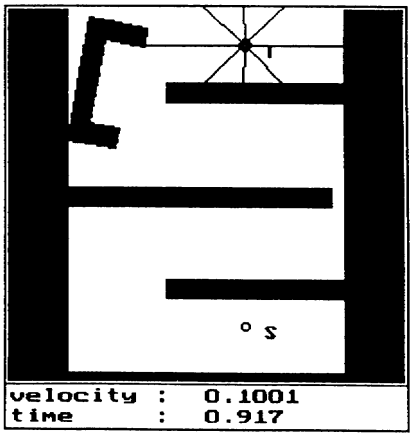


Fig. 4.6.18. The snapshot of how robot reacts with moving obstacles.

https://doi.org/10.1017/S0263574797000593 Published online by Cambridge University Press

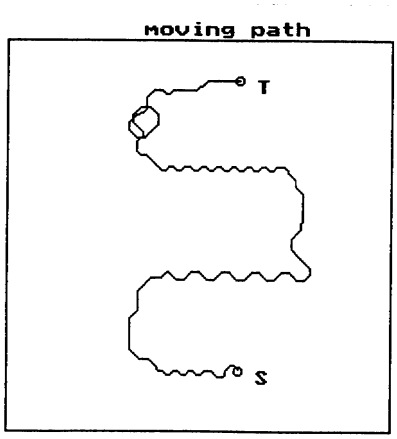


Fig. 4.6.19. The resultant path.

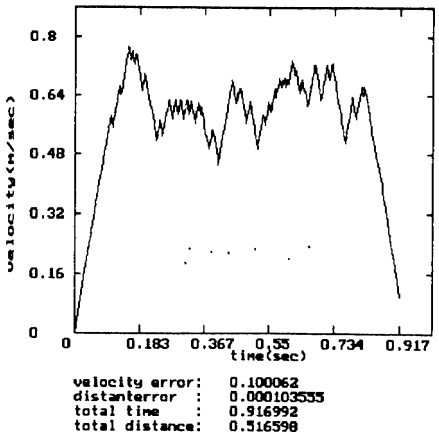


Fig. 4.6.20. The velocity profile.

References

1. D.M. Keirsey, E. Koch, J. Mckisson, A.M. Meystel and J.S.B. Mitchell, "Algorithm of Navigation for a Mobile Robot" *IEEE. Int. Conf. on Robotics and Automation* (1984) pp. 574–583.
2. T. Lozano-Perez and M.A. Welsey, "An Algorithm for Planning Collision-Free Paths among Polyhedral Obstacles" *Communication of the ACM* **22,** No. 10, 560–570 (1979).
3. C.L. Shih, T.T. Lee and W.A. Gruver, "A Unified Approach for Robot Motion Planning with Moving Polyhedral Obstacles" *IEEE Trans on SMC.* **20,** No. 4, 903–915 (1990).
4. Y.H. Liu and S. Arimoto, "Finding the Shortest Path of a Disc Among Polygonal Obstacles Using a Radius-independent Graph" *IEEE Trans. on Robotics and Automation* **11,** No. 5, 682–691 (1995).
5. L.E. Kavraki, Petr Svestka, Jean-Claude Latombe and Mark H. Overmars, "Probabilistic Roadmaps for Path Planning in Higher-Dimensional Configuration Spaces" *IEEE Trans. on Robotics and Automation* **12,** No. 4, 566–580 (1996).
6. R.A. Brooks, "Solving the Find-Path Problem by Good Representation of Free Space" *IEEE Trans. on SMC* **13,** No. 3, 190–197 (1993).
7. S.S. Iyengar, C.C. Jorgensen, S.V.N. Rao and C.R. Weisbin, "Robot Navigation Algorithms using Learned Spatial Graphs" *Robotica* **4,** Part 2, 93–100 (1986).
8. D.T. Kuan, J.C. Zamiska and R.A. Brooks, "Natural Decomposition of Free Space for Path Planning" *IEEE International Conf. on Robotics and Automation* (1985) **Vol 1,** pp. 168–173.
9. S. Kambhampati and L.S. Davis, "Multiresolution Path Planning for Mobile Robots" *IEEE J. Robotics and Automation* **2,** No. 3, 135–145 (1986).
10. E.K. Wong and K.S. Fu, "A Hierarchical-Orthogonal-Space Approach to Collision-Free Path Planning" *IEEE International Conf. on Robotics and Automation* (1985) **Vol. 1,** pp. 506–511.
11. E. Palma-Villalon and P. Dauchez, "World Representation and Path Planning for a Mobile Robot" *Robotica* **6,** Part 1, 35–40 (1988).
12. O. Khatib, "Real-time Obstacle Avoidance for Manipulators and Mobile Robots" *IEEE International Conf. on Robotics and Automation* (1985) **Vol. 1,** pp. 500–505.
13. D.F. Cahn and S.R. Phillips, "ROBNAV: A Range-Based Robot Navigation and Obstacle Avoidance Algorithm" *IEEE Trans. on SMC* **5,** No. 9, 544–551 (1975).
14. H.P. Huang and P.C. Lee, "A Real-time Algorithm for Obstacle Avoidance of Autonomous Mobile Robots" *Robotica* **10,** part 3, 217–227 (1992).
15. V.J. Lumelsky and A.A. Stepanov, "Dynamic Path Planning for a Mobile Robot Automation with Limited Information on the Environment" *IEEE Trans. on Automatic Control* **31,** No. 11, 1058–1063 (1986).
16. D.P. Miller, "Planning and Problem Solving" **In**: *Tutorials on Autonomous Mobile Robots, 1989 Int. Conf. on Robotics and Automation,* Scotlsdale, Arizona (May 14–19 (1989) pp. 1–12.
17. Qiuming Zhu, "Hidden Markov Model for Dynamic Obstacle Avoidance of Mobile Robot Navigation" *IEEE Transactions on Robotics and Automation* **7,** No. 3, 390–397 (1991).

https://doi.org/10.1017/S0263574797000593 Published online by Cambridge University Press