'89 KACC 1989. 10. 27~28

MODELING AND CONTROL STRATEGIES FOR DYNAMICAL OBSTACLE AVOIDANCE BY MOBILE ROBOTS

Q. Zhu and N. K. Loh

Center for Robotics and Advanced Automation
Oakland University
Rochester, MI 48309, U. S. A

This paper presents a theoretic study and computer simulation of models and approaches for dynamical obstacle avoidance by mobile robots. The movement of obstacles in unknown environment is described by any one or a combination of three models. The control strategy of the mobile robots is formulated based on one of three approaches. A trajectory-guided control strategy for dynamical obstacle avoidance has been developed. The method greatly simplifies the control process of mobile robots, and is computationally attractive.

1. INTRODUCTION

Real-time visual guidance and navigation of mobile robots or unmanned vehicles is a problem with important applications [4, 5, 7, 9, 10]. However, it is also a difficult and challenging problem because of the complexity of the unknown environment which may be encountered by the mobile robots. The environment complexity includes the variations of physical appearances and sizes of the obstacles, their kinematical behaviors, the inertia of their motions, and the planned and unplanned perturbations of their motions. This is particularly true for the collision avoidance where the obstacles (may be other robots or vehicles) in the environment are moving at relatively high speed [2, 6, 8].

Collision avoidance in navigation is a sequence of operations including obstacle detection and trajectory planning. In the obstacle detection process, the vision system determines whether there is any object or obstacle, static or in motion, in the scene which may interfere with the planned trajectory of a mobile robot or autonomous vehicle. If the obstacle in the environment is also in motion, then the sensing systems of the robot will have to determine its velocity and acceleration, predict its trajectory, and then check for possible interferences. If a interference is detected or predicted, the trajectory re-planning processor of the mobile robot must then be activated to identify a collision-free path for safe navigation [1, 2, 3].

Obstacle avoidance in unknown environment can be achieved by utilizing an image space search based control strategy [5, 6, 7]. In this method, a collision-free trajectory can be determined in a local map constructed from the scene. The map depicts the positions and the expected position changes of the obstacles in the environment. A search is then conducted in the collision-free space of the map by extremizing (minimizing or maximizing) certain performance criteria. An unconstrained form of collision-free trajectory, if it exists, can be identified. This method usually involves the processing of a massive number of image elements, and is, therefore, computation intensive and time-consuming.

We have developed a trajectory-guided control strategy for dynamical obstacle avoidance [11, 12]. This method does not require an explicit environment map. It looks for a collision-free trajectory by checking the potential of collisions for each of a finite number of candidate trajectories. These candidate trajectories are formed according to the current state of the mobile robot and ordered according to certain priority measures. The most important feature of the method is that it greatly simplifies the control process of the mobile robot, and is computationally attractive. The complexity of the algorithm is proportional to the number of obstacles observed in the environment.

It is necessary to have a proper description of the motions of the obstacles for the prediction of possible collisions of them with the mobile robot disregarding the control strategies. The movement of obstacles in unknown environment can be described by one or a combination of three models. Control strategies are developed correspondently with the obstacle motion models. The models and the performance of the control strategies developed in this paper have been simulated and useful results have been obtained.

The paper is organized as following. Section 2 describes and defines the dynamical obstacle avoidance problem for the mobile robots, addressing on different types of motion models for the obstacles and the robots. Section 3 presents different approaches and algorithms for dynamical obstacle avoidance. Software development and computer simulation are described briefly in this section. Simulation results and performance evaluation are presented in Section 4. Section 5 contains the conclusion.

2. MODELS OF DYNAMICAL OBSTACLE AVOIDANCE

The increased difficulty of dynamical obstacle avoidance compared with static obstacle avoidance lies in the extra dimension t added to the Cartesian space where the robot moves. In the mobile robot or autonomous land vehicle navigation problem, the obstacle motions are restricted to the (X,Y) two-dimensional spatial space. The computation of dynamical obstacle avoidance is then carried out in the three-dimensional space (X, Y, t).

2.1. Velocity and Acceleration of Obstacle Motion

The first problem in dynamical obstacle avoidance is the detection of the movement of the obstacles in the environment. The motion of a rigid-body object can be fully described by its velocity and acceleration.

The velocity of a moving obstacle in the two-dimensional plane is given by a vector

$$\mathbf{v}(t) = [v_x(t), v_y(t)]^T = d\mathbf{s}(x,y,t)/dt, \qquad (2.1)$$

where $[.]^T$ denotes the transpose of [.] and

$$\mathbf{s}(x,y,t) = [\, x(t), \; y(t)\,]^T \qquad (2.2)$$

denotes the position vector of the obstacle at time t.

The direction of $\mathbf{v}(t)$ is given by

$$\theta_v(t) = \tan^{-1}(v_y(t) / v_x(t)), \qquad (2.3)$$

and the magnitude of the velocity is given by

$$\|\mathbf{v}(t)\| = \sqrt{(v_x(t)^2 + v_y(t)^2\,)}. \qquad (2.4)$$

The acceleration is given by,

$$\mathbf{a}(t) = [\, a_x(t), a_y(t)]^T = d\mathbf{v}(t)/dt, \qquad (2.5)$$

or

$$\mathbf{a}(t) = d^2\mathbf{s}(x,y,t)/dt^2. \qquad (2.6)$$

The direction of $\mathbf{a}(t)$ is given by

$$\theta_a(t) = \tan^{-1}(\, a_y(t) / a_x(t)), \qquad (2.7)$$

and the magnitude of the acceleration is given by

$$\|\mathbf{a}(t)\| = \sqrt{(a_x(t)^2 + a_y(t)^2)}. \qquad (2.8)$$

In discrete-time case, the above quantities can be calculated by:

$$v_x(t) = (x(t) - x(t-\Delta t))/\Delta t, \qquad (2.9)$$

$$v_y(t) = (y(t) - y(t-\Delta t))/\Delta t, \qquad (2.10)$$

and

$$a_x(t) = (v_x(t) - v_x(t-\Delta t))/\Delta t, \qquad (2.11)$$

$$a_y(t) = (v_y(t) - v_y(t-\Delta t))/\Delta t. \qquad (2.12)$$

where Δt is the sampling interval.

2.2. Detection of Obstacle Motion

One way of determining the velocity and acceleration of moving obstacles is by processing two or more successive scene images [11]. The velocity and acceleration of rigid-body object can be represented by the velocity and acceleration of a characteristic point at the surface of the object in these images.

Let G(t) and G(t-Δt) be two successive image frames at time t and t-Δt, respectively, where Δt is the sampling time interval between the two image frames. Let (x(t), y(t)) and (x(t-Δt), y(t-Δt)) be two characteristic points on obstacle O_i in image G(t) and G(t-Δt), respectively. The velocity of that point can then be calculated by (2.9), and (2.10). The acceleration of the motion can be calculated by utilizing one more image frame G(t-2Δt) and (2.11) and (2.12).

In the following sections, we will assume that the obstacles in the robot motion environment can be distinguished individually. Therefore their velocities and accelerations can be determined from their positions in successive image frames. The case where the obstacles are not distinguishable and the method that does not need to distinguish individual obstacles in the scene for collision avoidance will be treated elsewhere.

2.3. OBSTACLE AND ROBOT MOTION MODEL

2.3.1. Obstacle Motion Models

(a). Constant Velocity Model: A basic and simplest model for obstacle motion can be obtained by assuming constant velocity, that is, the acceleration of the obstacle is zero. We have

$$\mathbf{a}(t) = 0, \qquad (2.13)$$

$$\mathbf{v}(t) = \mathbf{c}, \quad (\mathbf{c} \text{ is a constant vector}), \qquad (2.14)$$

$$\mathbf{s}(x,y,t+\Delta t) = \mathbf{s}(x,y,t) + \mathbf{c}\Delta t. \qquad (2.15)$$

Although obstacles seldom move with constant velocities, this simple motion model is still useful. It constitutes a basic model from which other more complicated motion models may be derived.

Robot motion control requires observation of the velocities of the obstacles in the scene at every sampling interval Δt. An obstacle may be viewed as moving with a constant velocity during Δt. This assumption is particularly valid for small Δt.

(b). Random Motion Model: The random motion model assumes that the acceleration of an obstacle in the scene changes randomly with the changes governed by certain probability distribution functions, such as a Gaussian or uniform distribution. We have

$$\mathbf{a}(t) = \mathbf{w}(t), \qquad (2.16)$$

where $\mathbf{w}(t)$ is a random noise vector which may be uniformly distributed within certain ranges or a Gaussian process. It follows that the random velocity can then be calculated by

$$\mathbf{v}(t) = \mathbf{v}(t-\Delta t) + \mathbf{w}(t)\Delta t. \qquad (2.17)$$

Alternatively, the random motion model can be described as

$$\mathbf{a}(t) = \alpha_1\mathbf{a}(t-\Delta t) + \beta_1\mathbf{w}(t), \qquad (2.18)$$

where α_1 specifies the mechanism of how the object maintains its acceleration with a given mean and β_1 is the intensity of the external force $\mathbf{w}(t)$. The external force $\mathbf{w}(t)$ is modeled by

$$\mathbf{w}(t) = \mathbf{f}(t)/m, \qquad (2.19)$$

where $\mathbf{f}(t)$ is the random force vector exerted on the obstacle and m is the mass of the obstacle. The random velocity is then given by

$$\mathbf{v}(t) = \mathbf{v}(t-\Delta t) + [\alpha_1 \mathbf{a}(t-\Delta t) + \beta_1 \mathbf{w}(t)]\Delta t. \quad (2.20)$$

(c). Intentional Model: In the intentional model, the obstacles move in a planned schedule, such as a pre-determined or programmed destination, fixed route, decoy motion, attempting attacks, etc. The obstacles may maintain their pre-planned motions, disregarding all environmental changes (e.g., collision with others). They may also monitor the changes and modify their motion parameters accordingly. In either case, we have

$$\mathbf{s}(x,y,t+\Delta t) = \mathbf{s}(x,y,t) + \mathbf{v}(t)\Delta t, \quad (2.21)$$

$$\mathbf{v}(t) = \mathbf{v}(t-\Delta t) + \mathbf{a}(t)\Delta t. \quad (2.22)$$

But the variations of $\mathbf{a}(t)$ now depend on the control force exerted on the obstacle, i.e.,

$$\mathbf{a}(t) = \alpha_2 \mathbf{a}(t-\Delta t) + \beta_2 \mathbf{e}(t), \quad (2.23)$$

where $\mathbf{e}(t)$ represents the variations of accelerations of the obstacle resulted from the obstacle's intentional control strategies, and α_2 and β_2 are constants.

The variations of the accelerations of the obstacle motion in this model can be predicted or estimated by observations of its previous motion history and the environmental conditions. However, the prediction or estimation is usually complicated, depending on the actual environmental conditions. For example, an obstacle may move in a hostile manner, such as an enemy's missile attempting an attack. In this case, the obstacle wants to follow and collide with the robot. The obstacles may also move in a friendly manner, that is, every obstacle may want to avoid collisions with other objects. It should be pointed out that the actual motion of an obstacle may be a combination of all three motion models.

2.3.2. Mobile Robot Motion Model

The motion of a mobile robot can be described in a similar way as for a rigid-body object. The control signal is the current acceleration generated by the vision and decision (path planning) systems.

The motion model follows the general rigid-body object motion rule:

$$\mathbf{v}_R(t) = \mathbf{v}_R(t_0) + \int \mathbf{a}_R(\tau)d\tau, \quad (2.24)$$

$$\mathbf{R}(x,y,t) = \mathbf{R}(x,y,t_0) + \int \mathbf{v}_R(\tau)d\tau, \quad (2.25)$$

where $\mathbf{R}(x,y,t)$ denotes the position of the robot at time t, $\mathbf{v}_R(t)$ and $\mathbf{a}_R(t)$ denote velocity and acceleration of the robot, respectively.

During the sampling interval, $\mathbf{R}(x,y,t)$ can be expressed as

$$\begin{aligned}\mathbf{R}(x,y,t) &= \mathbf{R}(x,y,t-\Delta t) + \mathbf{v}_R(t)\Delta t & (2.26a)\\ &= \mathbf{R}(x,y,t-\Delta t) + \mathbf{v}_R(t-\Delta t)\Delta t + \mathbf{a}_R(t)(\Delta t)^2, & (2.26b)\end{aligned}$$

where the robot is assumed to move with constant acceleration within each sampling interval.

3. APPROACHES FOR DYNAMICAL OBSTACLE AVOIDANCE

3.1. Collision Prediction and Path Planning

Obstacle detection is often treated as geometric intersections [1, 2]. The approach is to inspect the images of the scene to determine whether any obstacle and the trajectory of the robot occupy a common region. An empty set implies a collision-free region. Unlike static obstacle avoidance, the dynamical obstacle avoidance process must be performed by taking into account of all possible positions of the obstacles along the time axis.

To predict possible collisions, the trajectories of the mobile robot and the moving obstacles must be identified in terms of their velocities, accelerations and physical appearance. Let the current time of computation be t_0, i.e., starting time, and let $\mathbf{R}(t_0+t)$ denote the position of the robot at time t_0+t. A collision-free path is a function of t and is subject to the control strategy employed by the robot. A sequence of such positions from $\mathbf{R}(t_0)$ to $\mathbf{R}(t_0+t)$ determined by the robot control system will be called the *planned trajectory* of the robot motion. Collision avoidance is achieved by computing the intersections of the planned trajectory of the robot with the possible positions of the obstacles during the time interval $[t_0, t_0+t]$. Let $\mathbf{I}_R(t_0+t)$ denotes the set of points occupied by the robot for the time interval $[t_0, t_0+t]$ and $\mathbf{I}_{O_j}(t_0+t)$ the set of points occupied by the predicted trajectory of the moving obstacle O_j for the time interval $[t_0, t_0+t]$. When the intersection of these two sets is non-empty, i.e., $\mathbf{I}_R(t_0+t) \cap \mathbf{I}_{O_j}(t_0+t) \neq \mathbf{0}$, a collision with the obstacle O_j is possible and the collision-free path planning process must be activated.

We define a p^{th} collision-free path of the robot (only local path planning of the robot motion is discussed here) as a set of connected collision-free spaces along the time axis from t_0 to t_0+t. Given the sampling time interval Δt, $t/\Delta t$ predicted images can be constructed according to the velocities and accelerations of the obstacles detected in the scene. The collision-free elements of the collision-free set can then be computed. Each of the image frames $G(t_0+k\Delta t)$, $k=1,2,\ldots,t/\Delta t$, can be used to determine a collision-free set in the scene. The major task here is to identify a p^{th} connected path that is wide enough in the space represented by the $t/\Delta t$ images from $G(t_0)$ to $G(t_0+t)$.

Two dominant methods are used in the path planning process for dynamical obstacle avoidance.

(a). Collision-Free Space Guided Method

This approach will identify the collision-free spaces on the $t/\Delta t$ predicted image frames, and then look for a collision-free path. If there are multiple choices of such paths, the robot motion control system can find a best one from the alternatives by extremizing (minimizing or maximizing) certain performance criteria. The advantage of this method is that it is a global search method. A best, unconstrained form of collision-free trajectory, if it exists, can be identified and a best path will never be missed. The disadvantage is that the search is time consuming. Moreover, the robot will only use Δt part of the trajectory in its actual motion. A new best path must be computed at every sampling interval.

(b). Trajectory Guided Method

This method looks for a collision-free trajectory by checking the possibility of collisions for each of a finite number of candidate trajectories of the robot. The advantage of the method is that it is relatively simple and can be computed fast. The disadvantage is that the type and form (shape) of the trajectory of the robot motion is constrained and must be predefined. Usually a straight line form for the planned trajectory is assumed. A feasible path may be missed in case no feasible path spanning the time interval t is in straight line form (a curve form of a feasible collision-free path may exist). The result is that no collision-free trajectory can be identified. However, a best path among all possible paths can always be identified.

3.2. Obstacle Motion Predictions

While the motion of obstacles in unknown environment may be described by any one or a combination of above models, the control strategy of the mobile robot can, however, be formulated based on one of the following approaches:

(a). Anticipation Approach: It is assumed that the obstacle will move with constant velocity. The position of the obstacle at time t_0+t is calculated by

$$s(x,y,t_0+t) = s(x,y,t_0) + v(t_0)t. \qquad (3.1)$$

The intersection between the point set of the obstacle and the point set of the robot within the class of candidate trajectories is then computed.

In the collision free space guided method, we will first construct the images from the current time t_0 to time t with respect to the current environmental conditions and the measured velocity $v(t_0)$. We then search from the images collected to find a collision-free path which satisfies the constraints and specified performance criteria.

(b). Extended Anticipation Approach: This approach utilizes the collection of image frames {G(t-m), G(t-m+1),..., G(t-2), G(t-1), G(t)}, where without loss of generality, the sampling interval is normalized to $\Delta t = 1$. The velocity profile {$\mathbf{v}$(t-m), $\mathbf{v}$(t-m+1),..., $\mathbf{v}$(t-2), $\mathbf{v}$(t-1), $\mathbf{v}$(t)} of each obstacle is then calculated. The anticipated velocity of the obstacle at time t can be determined by

$$v_{O_j}(t) = f(v_{O_j}(t-1), v_{O_j}(t-2), v_{O_j}(t-3),...,v_{O_j}(t-N)), \qquad (3.2)$$

where the predicted velocity is expressed as a function of previous velocities. Certain strategy must be applied to determine which is the anticipated velocity. As an example, consider N = 3. One strategy may be just taking two equal velocities among the three as the anticipated velocity at time t. If all three are different, then take the current one or the average of the three.

(c). Probabilistic Approach: In this approach, each obstacle is assumed to move in a way governed by a specific probability distribution. The variations of the direction and magnitude of the accelerations of each obstacle will be assumed to satisfied a two-dimensional Gaussian distribution function in this study.

It is a complicated task to compute the probability of the points occupied by the obstacle O_j at time t with respect to every possible changes of the accelerations of the obstacle. Let Prob($\mathbf{a}_{O_j}$(t)) be the probability that obstacle O_j takes on the acceleration $\mathbf{a}$(t) at time t. In a simplified version, we can assume that after the acceleration of the obstacle has been changed at t_0, it does not change again from $t_0+\Delta t$ to t_0+t. The probability that the position s(x, y, t) is occupied by obstacle O_j at time t is then calculated by

$$\text{Prob}(s(x,y,t) \text{ is occupied by } O_j) = \text{Prob}(\mathbf{a}_{O_j}(t_0)). \qquad (3.3)$$

In a complete probabilistic model, the acceleration of the object must be allowed to change continuously. For discrete $\mathbf{a}_{O_j}$(t), we have

$$\text{Prob}(s(x,y,t) \text{ is ccupied}) = 1 - \Pi[1 - \text{Prob}(\mathbf{a}_{O_j}(t)]. \qquad (3.4)$$

It is possible to have multiple objects reaching the position s(x,y,t). A simplification of above evaluation is to take the maximum of the possibilities, that is

$$\text{Prob}(s(x,y,t) \text{ is occupied}) = \text{Max}(\text{Prob}(s(x,y,t) \text{ is ccupied by } O_j, j = 1,2,...,n)). \qquad (3.5)$$

Some approximation methods for the computation of Prob(s(x,y,t) is ccupied) may be applied. For example, consider a ring layer structure or a regional assignment scheme. Then only the probability that the obstacle will arrive at a ring or a region at time t will need to be computed.

3.3. Algorithms and Computation Procedures

The algorithms and computation procedure of the above methods are described below.

(a). Collision-Free Space Guided Procedure

Step 1: Make a binary map for every image frame from t_0 to t_0+t in time intervals of Δt according to the detected and computed velocities and accelerations (as well as the probabilities) of the obstacles in the scene, such that black pixel denote the space occupied by the obstacles, and white pixels form the collision-free space.

Step 2: Starting at t_0, record all collision-free elements around the current position. To begin the search of a feasible path, a direction priority list may be constructed according to the current robot position and the chosen goal or subgoal of the robot motion.

Step 3: Advance one time interval from $t_0+(k-1)\Delta t$ to $t_0+k\Delta t$. Calculate if an M^{th} width connected path exists (M is the required dimension of free space for robot motion, is proportional to the size of the robot, and is also a function of the robot dynamics).

Step 4: If such a path does not exist, record the k reached so far and return to the position $t_0+(k-1)\Delta t$ to try for another path direction.

Step 5: When t_0+t is reached, find the center line of this collision-free path, and determine the allowable accelerations for the robot for following the computed collision-free path.

Step 6: If no completely collision-free path is found, find the path with maximum k.

(b). Trajectory Guided Procedure

Step 1: Form a set of candidate trajectories, $S_i = \{S_i, i=1,2,\ldots,m\}$, which specifies the direction, velocity and acceleration to be utilized by the robot from time t_0 to t_0+t. The candidate trajectory may be the one inherited from time $t_0-\Delta t$, or derived from the current position and parameters, such as the desired direction or a global or subgoal position of the robot. Priorities may be assigned to these candidates.

Step 2: For each of the candidate trajectories in order of the priority, starting from the current position $\mathbf{R}(t_0)$ of the robot, explore the positions $\mathbf{R}(t_0+k\Delta t, S_i)$, $k = 1, 2, \ldots, t/\Delta t$, until $\mathbf{R}(t_0+t, S_i)$.

Step 3: Let $\mathbf{I}_R(t_0+k\Delta t, S_i)$, be the point set occupied by the robot as it moves to position $\mathbf{R}(t_0+k\Delta t, S_i)$ along trajectory S_i. For each such set, check if there is any intersection with the positions of the moving obstacles O_j, $i=1,\ldots,n$, that is, compute the intersection of $\mathbf{I}_R(t_0+k\Delta t, S_i) \cap \mathbf{I}_{O_j}(t_0+k\Delta t)$, $j = 1, 2, \ldots, n$.

Step 4: If $\mathbf{I}_R(t_0+k\Delta t, S_i) \cap \mathbf{I}_{O_j}(t_0+k\Delta t) \neq \mathbf{0}$, a collision is detected for this candidate trajectory. Record k, which is the index of collision free space reached for this candidate trajectory. Go to Step 2 to try for another new trajectory.

Step 5: If $\mathbf{I}_R(t_0+k\Delta t, S_i) \cap \mathbf{I}_{O_j}(t_0+k\Delta t) = \mathbf{0}$ for all j and k, then a collision-free trajectory for the robot motion has been identified. Retrieve the corresponding directions and accelerations of this trajectory. The robot control system can then issue the control commands to move forward with the desired accelerations in the desired directions.

4. COMPUTER SIMULATION AND RESULTS

The simulation starts with a binary image where objects in the scene are identified. The current velocity is obtained by taking two sequential images. The disparity of the obstacle locations in these two images are calculated. The simulation is originally designed to have the robot operate on an open territory, but a road or any other environmental constraints can be simulated easily. A global goal (in direction and position given by a center line) is pursued by the motion of the robot. It guides the local path planning and finding. The underlying robot motion policy is:
(1) Travel as fast as possible (under the limitation of the mechanically achievable speed);
(2) Change the velocity smoothly and as small as possible (under the limitation of motion dynamics).

The following parameters are arbitrary set and can be changed any time during the simulations:
(1) The number of objects in the environment,
(2) The field of view of the mobile robot,
(3) The simulation speed.

The approaches and algorithms simulated are evaluated according to the following:
(1) Speed of the process;
(2) Complexity of the algorithms;
(3) Collision rate;
(4) Deviation of the trajectory;
(5) Reliability of the algorithm.

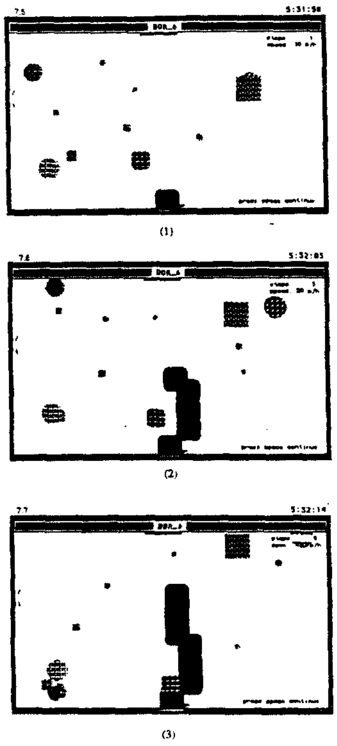
Fig. 4.1 video-type display of the simulation.

Fig. 4.1 shows a video-type display of the simulation, where a black square represents the mobile robot. Strip of black blocks represents the path travelled by the robot. The simulation results are shown in Figs. 4.2 - 4.3. We used the *field of view* (s_v) to measure how far away the robot system looks ahead. Fig. 4.2 shows the collision rate with respect to the field of view and different models of objects in the scene. Fig. 4.3 shows the average allowable maximum speed (AAMS) of the robot with respect to the field of view and the models of objects.

It is observed from the simulation results that the increase of the field of view does not proportionally improves the collision rate. The reason is because: (1) we can predict the movement of a nearbying obstacle more reliably with its previous movement history, (2) collision avoidance in dynamical environment is basically a local operation, the moving state of a far away object has less meaning to the current trajectory of the robot. On the other hand, the increase of the field of view increases the vision computation time, thus slows down the speed of the robot. An optimal field of view exists under certain speed of the robot and the number of obstacles in the scene. It is obvious that the increase of the number of obstacles increases the vision computation time and slows down the robot too.

5. CONCLUSIONS

The problem of visual guidance for dynamical obstacle avoidance by mobile robots has been investigated. Three motion models for the obstacles have been developed: (1)

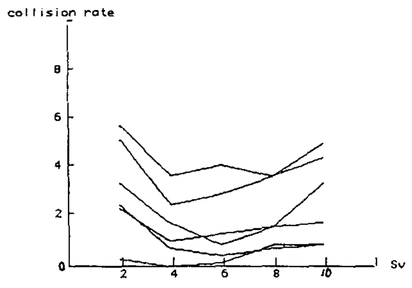

Fig. 4.2 collision rate with respect to field of view and different models of objects.

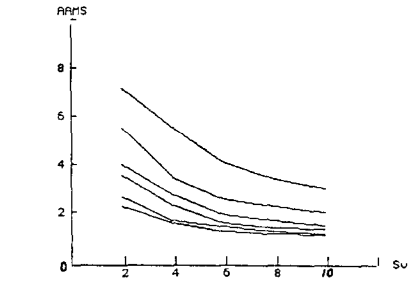

Fig. 4.3 average allowable maximum speed (AAMS) with respect to field of view and models of objects.

Constant Velocity Model, (2) Random Motion Model, (3) Intentional Model. Three approaches for robot motion have also been studied: (1) Anticipation Approach, (2) Extended Anticipation Approach, (3) Probabilistic Approach. Extensive simulations have been carried out. Video-type of graphical displays have been presented. The research provides a complete treatment of the problem, including theoretical and software development. The simulation results provide some useful insight into the problem of visual guidance and control of mobile robot for obstacle avoidance.

REFERENCES

[1] R. A. Brooks, "Solving the Find-Path Problem by Good Representation of Free-Space", *IEEE Trans. Systems, Man, and Cybernetics,* Vol. SMC-13, No. 3, pp. 190-197, March/April 1983.

[2] J. Canny, "Collision detection for Moving Polyhedra", *IEEE Trans. Pattern Analysis and Machine Intelligence,* Vol. PAMI-8, No. 2, pp. 200-209, March 1986.

[3] A. Chattergy, "Some Heuristics for the Navigation of a Robot", *International Journal of Robotics Research,* Vol. 4, No. 1, pp. 59-66, Spring 1985.

[4] J. L. Crowley, "Navigation for an Intelligent Mobile Robot", *First Conference on Artificial Intelligence Application,* IEEE Computer Society, Denver CO. pp. 51-56. December 1984.

[5] L. Gouzenes, "Strategies for Solving Collision-free Trajectories Problems for Mobile and Manipulator Robots", *International Journal of Robotics Research,* Vol. 3, No.4, pp. 51-65, Winter 1984.

[6] O. Khatib, " Real-time Obstacle Avoidance for Manipulators and Mobile Robots", *International Journal of Robotics Research,* Vol. 5, No.1, pp. 90-98, Spring 1986.

[7] T. Lozano-perez and M. A. Wesley, "An Algorithm for Planning Collision-free Paths among Polyhedral Obstacles", *Communication ACM,* Vol. 22, No. 10, pp. 560-570, October 1979.

[8] W. Martin and J. K. Aggarwal, "Survey : Dynamic Scene Analysis", Computer Vision, Graphics, and Image Processing, Vol.7, No. 3, pp. 356-374, June 1978.

[9] J. J. Nitao and A. M. Parodi, "An Intelligent Pilot for an Autonomous Vehicle System", *IEEE Second Conf. on Art. Intel. Appli.,* Miami Beach, FL. pp. 176-183, 1985.

[10] S. Tsugawa, T. Hirose, and T. Yatabe, "An Intelligent Vehicle with Obstacle Detection and Navigation Functions", *Proceedings IECON'84,* pp. 303-308, 1984.

[11] Q Zhu, "Structure pyramids for Representing and Locating Moving Obstacles", *Proceedings of IEEE CVPR Conference,* pp. 832-837, June 1988.

[12] Q. Zhu and N. K. Loh, "Dynamic Obstacle Avoidance By Mobile Robots", Technical Report TR-04-89-CRAA-01, Center for Robotics and Advanced Automation, Oakland University, Rochester, MI, 48309, April 1989.