6-1991

Hidden Markov Model for Visual Guidance of Robot Motion in Dynamic Environment

Qiuming Zhu
University of Nebraska at Omaha, qzhu@unomaha.edu

Hidden Markov Model for Dynamic Obstacle Avoidance of Mobile Robot Navigation

Qiuming Zhu

Abstract—**Models and control strategies for dynamic obstacle avoidance in visual guidance of mobile robot are presented. Characteristics that distinguish the visual computation and motion-control requirements in dynamic environments from that in static environments are discussed. Objectives of the vision and motion planning are formulated as: 1) finding a collision-free trajectory that takes account of any possible motions of obstacles in the local environment; 2) such a trajectory should be consistent with a global goal or plan of the motion; and 3) the robot should move at as high a speed as possible, subject to its kinematic constraints. A stochastic motion-control algorithm based on a hidden Markov model (HMM) is developed. Obstacle motion prediction applies a probabilistic evaluation scheme. Motion planning of the robot implements a trajectory-guided parallel-search strategy in accordance with the obstacle motion prediction models. The approach simplifies the control process of robot motion.**

I. Introduction

Real-time visual guidance and control of mobile robots or autonomous vehicles is an area of growing interest to many computer scientists and engineers [1]–[6]. It is a challenging problem because of the complexity of the unknown environment encountered by mobile robots. The environment complexity includes the variations of physical appearances of obstacles, their kinematic behavior, and the planned or unplanned perturbations of their motion. This is particularly true for the collision avoidance where the obstacles (which may be other robots or human operators) in the environment are also moving at relatively high speed [5], [6]. Much research has been done on the visual guidance of mobile robots in static environments where the positions of obstacles do not change [7], [8], [10]. One large and fruitful area is "motion planning" where, given a description of the environment and initial and final positions, a motion-planning algorithm computes a collision-free trajectory. Cognitive capability enables the mobile robot to form and modify a model of the world around it and relate this world model to the task objectives. However, the visual guidance and control of robot motion in an unknown environment where the obstacles are also moving is less systematically studied.

We call the environment where the obstacles are also moving in the dynamic environment. Visual guidance of a mobile robot in such an environment is dealt with in a sequence of operations consisting of obstacle motion detection and robot trajectory planning [6], [17], [18]. In the obstacle motion detection, the vision system acquires necessary information from the scene and determines whether any object, static or in motion, might interfere with the planned trajectory of the mobile robot. If the obstacle in the environment is also in motion, the vision system will have to determine the velocity and acceleration of the obstacle, predict its moving trajectory, and check for possible interferences. If an interference is detected, the trajectory planning process of the robot must be activated to identify a collision-free path for the robot. The reliability and effectiveness of the path planning depends largely upon the correctness of the detection and prediction of the motion states of the obstacle. Obstacle motion may be predicted by a deterministic estimation. In this approach, anticipated motion states of the obstacle are computed by using a linear combination (weighted average) of the previous motion states [6]. Because of the diversity of obstacle motion models, the estimation is not easily justified.

A hidden Markov model (HMM) is used in our research to describe and predict the motions of obstacles in a dynamic environment [12]. Obstacle motions are modeled as a stochastic process in HMM. Probabilistic evaluations are used to represent the obstacle motions and the potential variations of the motion states. A trajectory-guided path-planning approach has been developed in our research [6], [12]. In this approach, the determination of motion trajectory is exercised as a parallel pursuit of several alternatives. The algorithm searches for a collision-free trajectory by inspecting a finite set of candidate trajectories. These candidate trajectories are formed according to the current motion state of the robot and the given goal of the robot motion. A motion trajectory is selected in terms of the probabilistic evaluation. The result of the planning is represented in motion-control parameters directly. The method does not require the construction of a complete environment map and an exhaustive search. Computation therefore is simplified.

Section II of this paper presents, in general, a computational model for dynamic obstacle avoidance in visual guidance of mobile robots. Section III addresses the modeling of motion characteristics of the obstacles in the dynamic environments. The HMM for object motion prediction is illustrated in Section IV. Section V presents the trajectory-guided path-planning algorithm. Computer simulation and performance evaluation are presented in Section VI. Section VII contains concluding remarks.

II. Visual Guidance of Robot Motion

A. Global and Local Path Planning

Motion planning for mobile robots and autonomous vehicles in dynamic environments is conducted in two processes: 1) global path planning, which is aimed at the accomplishment of the mission to be carried out by the robot, and 2) local path planning, which focuses on the finding of a collision-free path within the sight of the robot. Global path planning is performed in terms of the task requirement and a static view of the environment. It is called task-level planning. The result of this planning can be just an outline of the route or a set of subgoal points to be followed by the robot. It does not deal with the details of the motion trajectory and does not foresee any variations of the environment. Dynamic obstacle detection and

Manuscript received April 21, 1990; revised January 29, 1991. A portion of this work was presented at the IEEE International Conference on Systems Engineering, Pittsburgh, PA, August 1990.

The author is with the Computer Vision Laboratory, Department of Mathematics and Computer Science, University of Nebraska at Omaha, NE 68182.

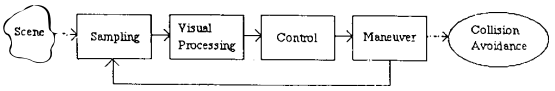

Fig. 1. Diagram of the visual guidance for dynamic obstacle avoidance.

collision avoidance are dealt with in the local path-planning process. Such a planning process emphasizes the safety of the motion. A collision-free trajectory within the sight of the robot is identified in this process. This work addresses on the local path planning.

While local path planning focuses on collision avoidance, it should also consider the global goal of the motion. The objectives of local path planning can then be stated as:

1) finding a collision-free trajectory that takes account of any possible motions of the obstacles in the local environment;
2) such a trajectory should be consistent with the global goal of the motion; and
3) the robot should move at as fast a speed as possible, subject to its kinematic constraints.

B. Motion State Description

The local path planning of mobile robots in dynamic environments is modeled as a discrete-time process. A control cycle starts at the moment that a sample (an image) of a local scene is acquired by a visual sensing device. The image is processed immediately by a visual information processing unit. Any potential obstacles in the environment are detected. Predictions of the obstacle motion are made. A collision-free path is searched for according to the visual information gathered. Maneuver commands are then generated by the motion-control unit. As the robot proceeds, this "sampling—visual processing—collision detection—path planning —maneuver" cycle repeats. Fig. 1 shows a diagram of this process. A series of maneuvers leads to the accomplishment of the task objectives.

The motion state of a robot at a time instance t is represented by

$$(p_r(t), v_r(t), a_r(t))) \tag{1}$$

where $p_r(t)$ denotes the position of the robot, $v_r(t)$ is the velocity, and $a_r(t)$ the acceleration of the robot at time t. We use $s_r(t)$ to denote the path actually travelled by the robot in the time interval $[t, t + \Delta t]$, where Δt is the control cycle. The motion of the robot can be described as

$$s_r(t) = p_r(t + \Delta t) - p_r(t) = \int_t^{t+\Delta t} v_r(\tau)\, d\tau \tag{2}$$

where

$$v_r(\tau) = v_r(t) + \int_t^{\tau} a_r(t)\, dt. \tag{3}$$

We use $T_c(t)$ to denote the *vision and control processing time* of the robot. $T_c(t)$ includes the time spent by the robot to process the scene image, detect the moving obstacles, and identify a collision-free path. This $T_c(t)$ mainly determines, or constrains, the motion speed of the robot. Basically, we should have

$$T_c(t) < \Delta t(t). \tag{4}$$

The notation $\Delta t(t)$ indicates that the control cycle is also a function of t. Let $T_d(t)$ denote the time delay incurred due to the kinematics of the robot; we then have

$$\Delta t(t) = T_c(t) + T_d(t). \tag{5}$$

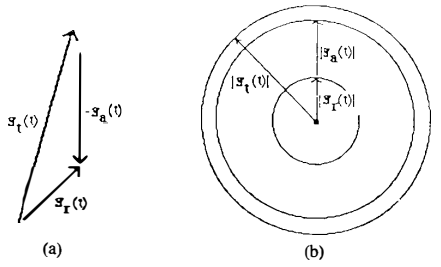

Fig. 2. Relation of s_t, s_r, and s_a in (a) vectorial, and (b) regional diagrams.

Replacing the Δt of (2), by (5), the (2) becomes

$$s_r(t) \leq \int_t^{t+T_c(t)+T_d(t)} v_r(\tau)\, d\tau. \tag{6}$$

C. Safety Motion Principle

The central problem in local path planning is to explore the values of $v_r(\tau)$ in (6). The values are affected by the presence of obstacles in the environment. We classify the moving obstacles in two categories: their motion parameters are known or unknown. One principle for a safe motion of the robot in the dynamic environments is that a robot should always keep a certain distance away from any obstacle whose motion parameters are unknown.

We define a safety range s_a within which the robot has sufficient time to go through a number of sampling and control cycles. By these cycles, the motion states of the obstacles can be detected and proper actions can be taken by the robot to avoid any possible collision. Note that, for a static obstacle, this s_a could be a very small number. For moving obstacles, it must be carefully computed. Let $v_{o_i}(t)$ denote the velocity of obstacle O_i, and $s_{a_i}(t)$ the safety range of the robot with respect to obstacle O_i. $s_{a_i}(t)$ is a function of $v_{o_i}(t)$ and $v_r(t)$, such that

$$s_{a_i}(t) = \alpha \int_t^{t+\Delta t} v_{o_i}(\tau)\, d\tau - \beta \int_t^{t+\Delta t} v_r(\tau)\, d\tau + \gamma_0 \tag{7a}$$

where α and β are two coefficients; $\alpha, \beta \geq 0$. The γ_0 is a constant in which the kinematics of the mobile robot is embedded. Simply assuming $v_{o_i}(t)$ and $v_r(t)$ to be constants during the time interval $\Delta t(t)$, the above equation becomes

$$s_{a_i}(t) = \alpha s_{o_i}(t)\Delta t(t) - \beta s_r(t)\Delta t(t) + \gamma_0. \tag{7b}$$

$s_a(t)$ can be computed as the maximum of $s_{a_i}(t)$'s:

$$s_a(t) = \max\{s_{a_i}(t)\}. \tag{8}$$

A *trajectory-planning range* $s_t(t)$ denotes a space within which local planning is made. From the safety motion principle, we see that

$$s_t(t) \geq s_r(t) + s_a(t). \tag{9}$$

We call (9) the *motion-planning constraint equation*. Fig. 2 illustrates the relations of s_t, s_r, and s_a in vectorial and regional diagrams.

The *field of view* $s_v(t)$ is the scope observed by the vision system. It represents how far the robot vision system looks ahead and how widely it looks around. We should have

$$s_v(t) > s_t(t). \tag{10}$$

The inequality means that obstacles must be observed before reaching the range of local motion planning. Let's use $s_o(t)$ to denote the *observing range* in which the motion states of obstacles are de-

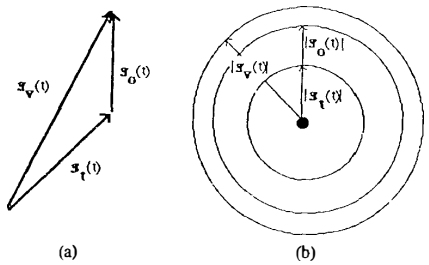

Fig. 3. Relation of s_v, s_t, and s_o in (a) vectorial and (b) regional diagrams.

tected; then we have

$$s_v(t) = s_t(t) + s_o(t). \tag{11}$$

We call (11) the *visual guidance constraint equation*. Fig. 3 illustrates the relations of s_v, s_t, and s_0 in vectorial and regional diagrams.

Combining (9) and (11), the vision and control requirement of a local path planning can be specified as

$$s_r(t) \le s_v(t) - s_o(t) - s_a(t). \tag{12}$$

Inserting (12) into (6), we get

$$\int_t^{t+T_c(t)+T_d(t)} v_r(\tau)\, d\tau \le s_v(t) - s_o(t) - s_a(t). \tag{13}$$

We also have

$$\int_t^{t+T_c(t)+T_d(t)} v_r(\tau)\, d\tau \le s_t(t) - s_a(t). \tag{14}$$

D. Environment Complexity and Optimal Motion States

It is essential that the motion of robots is affected by several aspects of the environment complexity. One such complexity is the number of obstacles $n_{\text{obj}}(t)$ within the range $s_v(t)$. We define the *obstacle density*, $d_{\text{obj}}(t)$, which is the number of obstacles $n_{\text{obj}}(t)$ within the unit area of $s_v(t)$:

$$d_{\text{obj}}(t) = n_{\text{obj}}(t) / \| s_v(t) \|. \tag{15}$$

In general, the vision and control processing time $T_c(t)$ is a function of the following components:

$$[s_v(t),\ d_{\text{obj}}(t),\ p_c]$$

where p_c stands for the processing capability of the vision and control system. An optimal control state of robot motion can be derived from the above discussion, which is

$$\int_t^{t+T_c(t)+T_d(t)} v_r(\tau)\, d\tau = s_r(t). \tag{16}$$

An optimal setting of the robot controller system, thus, should be

$$s_r(t) = s_t(t) - s_a(t) = s_v(t) - s_o(t) - s_a(t). \tag{17}$$

III. Obstacle Motion Models

Many researchers have indicated that the problem of dynamic obstacle avoidance is substantially more difficult than the static problem [8], [10]. Unlike static obstacle avoidance, the process in dynamic environments must consider any possible moves of the obstacles in terms of a time axis. In this work, we simply restrict the motion of robots in a two-dimensional space $[X, Y]$. The dynamic obstacle avoidance is then carried out in a three-dimensional space $[X, Y, t]$. The difficulty of dynamic obstacle avoidance lies also on the uncertainty of the obstacle motions. To properly predict the obstacle motions, analysis of their motion models should be made. The following three models are studied in our work: 1) constant velocity model, 2) random motion model, and 3) intentional motion model. Motion states of obstacles can generally be represented by $(p_{o_i}(t)v_{o_i}(t),\ a_{o_i}(t))$. For simplicity, the subscript o_i is omitted in the following descriptions.

A. Constant Velocity Model

This is the simplest case of obstacle motion. We have in this model

$$a(t) = 0 \tag{18}$$

$$v(t) = c \ (c \text{ is a constant vector}) \tag{19}$$

$$p(x, y, t + \Delta t) = p(x, y, t) + c\Delta t. \tag{20}$$

Although obstacles seldom move at a constant velocity, this motion model may serve as a basis for the description of other more complicated motion models. When measuring the motion history of an obstacle, it is considered as moving in a constant velocity within every time interval $[t,\ t + \Delta t]$.

B. Random Motion Model

This model considers that the changes of the motion states of an obstacle are governed by certain probability distribution functions. For example:

$$a(t) = \beta w(t) \tag{21}$$

where $w(t)$ is a random vector. In most cases, the probability distribution functions are unknown. Sometimes they can be assumed as a Gaussian or a uniform distribution. The function can be represented as

$$\text{Prob}(w(t)) = f_p(\mu(w(t)),\ \sigma^2(w(t))) \tag{22}$$

where $\mu(w(t))$ and $\sigma^2(w(t))$ are the mean and variance vectors, respectively, that regulate the distribution of $w(t)$. The f_p is a probability distribution function. It follow that the obstacle velocity can be calculated by

$$v(t) = v(t - \Delta t) + \int_{t-\Delta t}^{t} w(\tau)\, d\tau. \tag{23}$$

The random-motion model can be described alternatively as

$$a(t) = \alpha_1 a(t - \Delta t) + \beta_1 w(t) \tag{24}$$

where α_1 specifies the mechanism of how the obstacle tries to maintain its original acceleration, and β_1 is the intensity of random vector $w(t)$. The velocity is then given by

$$v(t) = v(t - \Delta t) + \int_{t-\Delta t}^{t} \left[\alpha_1 a(\tau - \Delta t) + \beta_1 w(\tau)\right] d\tau. \tag{25}$$

C. Intentional Motion Model

In this model an obstacle moves in a scheduled route, such as a predetermined destination, or a programmed route. The obstacle may also try to avoid collision with others. In this case, we have

$$a(t) = e(t) \tag{26}$$

or

$$a(t) = \alpha_2 a(t - \Delta t) + \beta_2 e(t) \tag{27}$$

where $e(t)$ represents the variations of accelerations resulting from any interval or external forces of the obstacle. α_2 and β_2 are two constants that specify the tendency of acceleration change. The function $e(t)$ depends very much on the particular environmental

$e(t)$

The acquisition of $e(t)$ relies very much on the background knowledge of the obstacles and a thorough observation of the motion histories of the obstacles.

It should be pointed out that the actual motions of obstacles in the environment are often a combination of the above motion models. It is obvious that the motions of obstacles in most cases are best described by a stochastic process.

IV. Hidden Markov Model for Obstacle-Motion Prediction

Prediction of obstacle motion is a fundamental step toward the identification of a collision-free trajectory of the robot motion. In this section we present a computational scheme for the prediction of obstacle motion states in time interval $[t, t + k]$, $k \geq 1$. The states of obstacle motion are denoted as $(p_{o_i}(t + k), v_{o_i}(t + tk), a_{o_i}(t + k))$.

A. Hidden Markov Model

The motion states of obstacles in dynamic environments can be presented in a formal way through Markov models. The simplest form of a Markov model is a Markov chain [9]. An interesting extension of the Markov chain appears in what has become known as the hidden Markov model (HMM) [9]. The name comes from the consideration that the Markov states cannot be observed directly. They can only be inferred from observation of the time series of other relevant events. Thus, the underlying Markov model is "hidden." In HMM, a number of different stochastic processes are combined to produce observations. HMM's are useful for a variety of problems where the time series of observations may go through distinct characteristic changes. Obstacle-motion prediction is one such problem.

In applying the HMM for obstacle-motion prediction, a sequence of probability functions and their parameters are defined. They are:

a) an initial motion state description of an obstacle o_i: $(p_{o_i}(t), v_{o_i}(t), a_{o_i}(t))$, which comes from the vision processing procedures;
b) an initial probability description of the motion dynamics of an obstacle o_i: Prob$(a_{o_i}(t))$;
c) a set of Markov transition parameters $[\mu(a_{o_i}(t + j)), \sigma^2(a_{o_i}(t + j))]$, which represent the mean and variance vectors of a probability transition function;
d) a presumed form of Markov transition function, denoted as $P(a_{o_i}(t + j))$;
e) a computation scheme for deriving Prob$(a_{o_i}(t + j))$ from applying the Markov transition functions; and
f) a computation scheme for deriving Prob$(v_{o_i}(t + j))$ and Prob$(P_{o_i}(t + j))$.

Starting with the mean and variance vectors

$$\left[\mu\left(a_{o_i}(t + j)\right), \sigma^2\left(a_{o_i}(t + j)\right)\right], \qquad j = 1, 2, \ldots, k \quad (28)$$

we have the Markov transition function

$$P\left(a_{o_i}(t + j)\right) = p\left(\mu\left(a_{o_i}(t + j)\right), \sigma^2\left(a_{o_i}(t + j)\right)\right) \quad (29)$$

where p is a probability density function. The initial mean and variance vector of the transition function are calculated by

$$\mu\left(a_{o_i}(t)\right) = f_\mu\left(a_{o_i}(t - 1), a_{o_i}(t - 2), \ldots, a_{o_i}(t - N)\right) \quad (30)$$

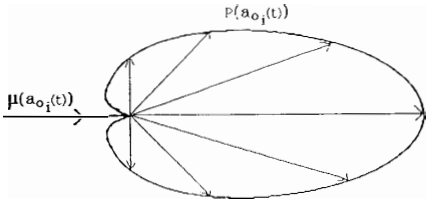

Fig. 4. Gaussian Markov transition function.

and

$$\sigma^2\left(a_{o_i}(t)\right) = f_\sigma\Big(\left(a_{o_i}(t - 1) - a_{o_i}(t - 2)\right)^2, \left(a_{o_i}(t - 2) - a_{o_i}(t - 3)\right)^2, \ldots, \left(a_{o_i}(t - N + 1) - a_{o_i}(t - N)\right)^2\Big). \quad (31)$$

The f_μ and f_σ are two linear functions obtained by applying an autoregressive moving average (ARMA) method [6]. We call the mean and variance vectors of the above expressions the hidden Markov parameters. When no knowledge about the actual motion model is available, the Markov transition function is presumed to satisfy a two-dimensional Gaussian density

$$P\left(a_{o_i}(t)\right) = \frac{1}{\sqrt{2\pi}\,\sigma\left(a_{o_i}(t)\right)} e^{-\frac{1}{2}\frac{\left(a_{o_i}(6) - \mu\left(a_{o_i}(t)\right)\right)^2}{\sigma^2\left(a_{o_i}(t)\right)}}. \quad (32)$$

A diagram of such a probability assumption is shown in Fig. 4.

The Markov transition function $P(a_{o_i}(t))$ is extended to the entire time interval $[t, t + k]$. That is

$$P\left(a_{o_i}(t + j)\right) = P\left(a_{o_i}(t + j - 1)\right) = \cdots = P\left(a_{o_i}(t)\right). \quad (33)$$

We then have the predicted probability of $a_{o_i}(t + k)$ at the time $t + k$

$$\begin{aligned}\text{Prob}&\left(a_{o_i}(t + k)\right) \\ &= \text{Prob}\left[a_{o_i}(t), a_{o_i}(t + 1), \ldots, a_{o_i}(t + k - 1)\right] \\ &= \left[\prod_{j=0}^{k-1} P\left(a_{o_i}(t + j)\right)\right]. \end{aligned} \quad (34)$$

We also have the mean and variance vectors for the hidden Markov transition function $P(v_{o_i}(t))$ and $P(p_{o_i}(t))$

$$\left[\mu\left(v_{o_i}(t + j)\right), \quad \sigma^2\left(v_{o_i}(t + j)\right)\right], \quad j = 0, 1, \ldots, k \quad (35)$$

$$\left[\mu\left(p_{o_i}(t + j)\right), \quad \sigma^2\left(p_{o_i}(t + j)\right)\right], \quad j = 0, 1, \ldots, k. \quad (36)$$

The initial states of above parameters can be computed by

$$\begin{aligned}\mu\left(v_{o_i}(t)\right) &= \mu\left(v_{o_i}(t - 1) + \left(a_{o_i}(t - 1)\right)\right) \\ &= \left(v_{o_i}(t - 1)\right) + \mu\left(a_{o_i}(t - 1)\right) \end{aligned} \quad (37)$$

$$\sigma^2\left(v_{o_i}(t)\right) = \sigma^2\left(v_{o_i}(t - 1) + \left(a_{o_i}(t - 1)\right)\right) = \sigma^2\left(a_{o_i}(t - 1)\right) \quad (38)$$

and

$$\begin{aligned}\mu\left(p_{o_i}(t)\right) &= \mu\left(p_{o_i}(t - 1) + v_{o_i}(t - 1) + \frac{1}{2}a_{o_i}(t - 1)\right) \\ &= p_{o_i}(t - 1) + \mu\left(v_{o_i}(t - 1)\right) + \frac{1}{2}\mu\left(a_{o_i}(t - 1)\right) \end{aligned} \quad (39)$$

$$\sigma^2\big(p_{o_i}(t)\big) = \sigma^2\Big(p_{o_i}(t-1) + v_{o_i}(t-1) + \frac{1}{2}a_{o_i}(t-1)\Big)$$

$$= \sigma^2\big(v_{o_i}(t-1)\big) + \frac{1}{2}\sigma^2\big(a_{o_i}(t-1)\big). \quad (40)$$

Notice that the time interval $\Delta t = 1$ in the above equations. We then have the hidden Markov transition function

$$P\big(v_{o_i}(t)\big) = \frac{1}{\sqrt{2\pi}\,\sigma\big(v_{o_i}(t)\big)} e^{-\frac{1}{2}\frac{(v_{o_i}(t)-\mu(v_{o_i}(t)))^2}{\sigma^2(v_{o_i}(t))}} \quad (41)$$

and

$$P\big(p_{o_i}(t)\big) = \frac{1}{\sqrt{2\pi}\,\sigma\big(p_{o_i}(t)\big)} e^{-\frac{1}{2}\frac{(p_{o_i}(t)-\mu(p_{o_i}(t)))^2}{\sigma^2(p_{o_i}(t))}}. \quad (42)$$

The description of the velocity and the position of the obstacle will be expressed in probability density functions

$$\text{Prob}\big(v_{o_i}(t+k)\big)$$

$$= \text{Prob}\big[v_{o_i}(t), v_{o_i}(t+1), \ldots, v_{o_i}(t+k-1)\big]$$

$$= \left[\prod_{j=0}^{k-1} P\big(v_{o_i}(t+j)\big)\right] \quad (43)$$

and

$$\text{Prob}\big(p_{o_i}(t+k)\big)$$

$$= \text{Prob}\big[p_{o_i}(t), p_{o_i}(t+1), \ldots, p_{o_i}(t+k-1)\big]$$

$$= \left[\prod_{j=0}^{k-1} P\big(p_{o_i}(t+j)\big)\right] \quad (44)$$

where the $P(a_{o_i}(t+j))$, $P(v_{o_i}(t+j))$, and $P(p_{o_i}(t+j))$ are the Markov transition functions. Equation (44) represents the probability value of position $p(x, y)$ being occupied by obstacle o_i at time $t+k$. In practice, we need to find the probability that position $p(x, y)$ is occupied by any obstacle in the scene. This is calculated by

$$\text{Prob}\big(p(t+k)\big) = \text{Max}\big\{\text{Prob}\big(p_{o_i}(t+k)\,|\,\forall o_i\big\}. \quad (45)$$

V. Trajectory-Guided Motion Planning

An image-space-based search approach is commonly used for robot motion planning in a static environment [3]–[5]. The approach can be extended to dynamic environments. To apply the approach, a collision-free trajectory is determined in a local map constructed from the scene images. The map depicts the positions and expected position changes of obstacles. These positions form "forbidden regions." A graph search is conducted to identify a collision-free path in a collision-free space of the map. The processing involves the search of a massive number of map elements, and therefore is computationally intensive. Moreover, the results need to be further converted from the geometric representation of the trajectory to the robot motion-control parameters, which are usually the driving forces corresponding to motion acceleration values. The map-based search approach therefore is less attractive than a direct control approach.

We have developed a trajectory-guided strategy [6], [12] for robot motion planning. The method inspects a finite set of candidate trajectories based on the probabilistic evaluations of the candidates using the HMM. The planning process attempts to proceed in several directions simultaneously, i.e., perform a parallel search.

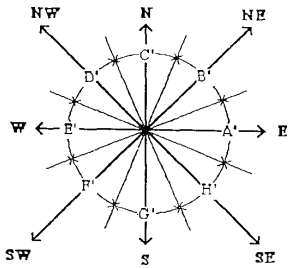

Fig. 5. Tessellation of local path planning space.

A. Representation of Candidate Trajectory

Recall the description of the motion state of a mobile robot:

$$\big(p_r(t), v_r(t), a_r(t)\big) = \big(\big[p_{r_x}(t), p_{r_y}(t)\big], \big[v_{r_x}(t), v_{r_y}(t)\big], \big[a_{r_x}(t), a_{r_y}(t)\big]\big). \quad (46)$$

It is more convenient to describe the maneuvers of robot motion in terms of 1) moving direction changes and 2) moving speed changes. Polar coordinates are then used here:

$$a_r(t) = \big[\theta_{a_r}(t), A_{a_r}(t)\big] \quad (47)$$

where

$$\theta_{a_r}(t) = \tan^{-1}\big(a_{r_y}(t)/a_{r_x}(t)\big) \quad (48)$$

and

$$A_{a_r}(t) = \sqrt{a_{rx}(t)^2 + a_{ry}(t)^2}. \quad (49)$$

Considering a grid space that is tessellated in eight geographical directions, a candidate trajectory of the robot can always have a $\theta_{a_r}(t)$ points to one of these directions:

$$\{\text{E}, \text{NE}, \text{N}, \text{NW}, \text{W}, \text{SW}, \text{S}, \text{SE}\}.$$

It is the quantization of $\theta_{a_r}(t)$. The $A_{a_r}(t)$ can also be quantized into

$$\{\ldots, -2A_r, -1A_r, 0, 1A_r, 2A_r, \ldots\}$$

where A_r is the unit speed change. A candidate-trajectory set is a Cartesian production of the above quantization. A collision-free trajectory is evaluated within the planning range $s_t(t)$. A time interval $[t, t+k]$ is related to $s_t(t)$ by

$$\int_t^{t+k}\left[v_r(t) + \int_t^{t+k}\big(a_i(\tau)\big)\,d\tau\right]dt = S_t(t). \quad (50)$$

B. Formation of Candidate Trajectory Set

Let $p_g(x, y)$ be a subgoal identified in the global planning. x and y are the two-dimensional coordinates of the point. $p_r(t)$ denotes the current position of the robot. A vector

$$\theta(t) = p_g(x, y) - p_r(t) \quad (51)$$

gives a quantitative expression of the relative position of the robot and the goal. Dividing the tessellated motion space into eight sections, as denoted by the letters A' to H' in Fig. 5, $\theta(t)$ is always allocated in one of these sections. A priority list of $\theta_{a_r}(t)$ can then be formed according to its relative position with $\theta(t)$. This priority list is shown in Table I. Numbers on the first row denote the priority order. The smaller number represent a higher priority. A simplification of the above case can be made by transforming the point $p_g(x, y)$ to one fixed direction, say, north. A vertical line passing $p_g(x, y)$ will then allocate $\theta(t)$ to only three possible sectors:

TABLE I
DIRECTION PRIORITY LISTS OF $\theta_{a_r}(t)$

$\theta(t)$	1	2	3	4	5	6	7	8
A'	E	NE	SE	N	S	NW	SW	W
B'	NE	N	E	NW	SE	W	S	SW
C'	N	NW	NE	W	E	SW	SE	S
D'	NW	W	N	SW	NE	S	E	SE
E'	W	SW	NW	S	N	SE	NE	E
F'	SW	S	W	SE	NW	E	N	NE
G'	S	SE	SW	E	W	NE	NW	N
H'	SE	E	S	NE	SW	N	W	NW

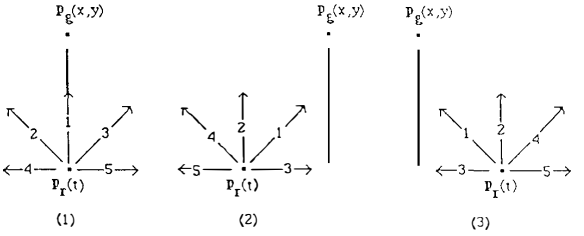

Fig. 6. Motion direction priority with respect to a global direction.

"*B*," "*C*," and "*D*." Priorities of $\theta_{a_i}(t)$ in these cases are depicted in Fig. 6. The number on the directional line indicates priority.

C. Evaluation of Candidate Trajectories

The candidate trajectory is selected according to a probabilistic evaluation of collisions. Using $a_i(t) = [\theta_{a_i}(t),\ A_{a_i}(t)]$ to denote the ith trajectory in the candidate set, the corresponding velocities and positions of the robot in $[t,\ t+k]$ can be obtained by

$$v_r(t+k \mid a_i(t)) = v_r(t) + \int_t^{t+k} a_i(\tau)\, d\tau \tag{52}$$

and

$$p_r(t+k \mid a_i(t)) = p_r(t) + \int_t^{t+k} v_r(\tau \mid a_i(t))\, d\tau \tag{53}$$

where

$$\begin{aligned} a_i(t) &= \left[a_{i_x}(t),\, a_{i_y}(t)\right]^T \\ &= \left[A_{a_i}(t)\cos(\theta_{a_i}(t)),\ A_{a_i}(t)\sin(\theta_{a_i}(t))\right]. \end{aligned} \tag{54}$$

The probability that a position $p_r(t+j \mid a_i(t))$, $j = 1, 2, \ldots, k$, is occupied by an obstacle, $\text{Prob}(p_r(t+j \mid a_i(t)))$, is computed by

$$\text{Prob}(p_r(t+j \mid a_i(t)) = \max(\text{Prob}(p_{o_i}(t+j)) \mid \forall o_i)). \tag{55}$$

A threshold value P_ϵ is used to control the termination of an evaluation process. When $\text{Prob}(p_r(t+j) \mid a_i(t)) > P_\epsilon$, the evaluation of $a_i(t)$ is abandoned.

An overall evaluation of a candidate trajectory consists of two parts: 1) a deterministic value and 2) a probabilistic value. It is expressed as

$$\begin{aligned} \text{Val}(a_i(t+j)) = \big[& D(p_r(t+j \mid a_i(t))), \\ & (1 - \text{Prob}(p_r(t+j \mid a_i(t))))\big]. \end{aligned} \tag{56}$$

Here, $D(p_r(t+j \mid a_i))$ is the Euclidean distance or

$$\begin{aligned} D(p_r(t+j \mid a_i(t))) = \big| & p_r(t+j \mid a_i(t)) \\ & - p_r(t+j-1 \mid a_i(t))\big|. \end{aligned} \tag{57}$$

A step-by-step description of the procedure is listed below.

D. Trajectory-Guided Path Planning (An HMM Process)

1) Form a candidate trajectory set, $C_a(t) = \{a_i(t),\ i = 1, 2, \ldots, m\}$, $a_i(t) = [\theta_{a_i}(t),\ A_{a_i}(t)]$.
2) Order $\{a_i(t),\ i = 1, 2, \ldots, m\}$ according to the relative position of $\theta(t)$ and $\theta_{a_i}(t)$.
3) Evaluate $\{a_i(t),\ i = 1, 2, \ldots, m\}$ simultaneously while processors are available,
   FOR each candidate trajectory $a_i(t)$,
   3.1) Initialize an evaluation value $\text{VAL}(a_i(t)) = 0$.
   3.2) FOR each anticipated motion step j, $j = 1, 2, \ldots, k$, of $a_i(t)$, where
   $$\int_t^{t+j} \left[v_r(t) + \int_t^{t+j} (a_i(\tau))\, d\tau \right] dt \le S_t(t)$$
   3.2.1) Calculate
   $$\text{Prob}(p_r(t+j \mid a_i(t)) = \text{Max}(\text{Prob}(p_{o_i}(t+j) \mid \forall o_i)$$
   3.2.2) IF $\text{Prob}(p_r(t+j \mid a_i(t)) > P_\epsilon$, terminate the evaluation of $a_i(t)$;
   Otherwise calculate
   $$\begin{aligned} \text{Val}(a_i(t)) = \text{Val}(a_i(t)) + \big[& D(p_r(t+j \mid a_i(t))), \\ & (1 - \text{Prob}(p_r(t+j \mid a_i(t))))\big] \end{aligned}$$
   3.3) When $j = k$, issue motion command in $a_i(t)$, process terminates.
4) If no motion command has been issued, select an $a_i(t)$ that has the largest $\text{Val}(a_i(t))$ and issue the motion command.

VI. COMPUTER SIMULATION

The computer simulation starts with a binary image where the number of obstacles and the motion models can be set *a priori*. Initial positions and the motion parameters of obstacles are generated randomly. The system is simulated with 1) a different number of obstacles, $d_{\text{obj}}(t)$, 2) a different combination of obstacle motion models, and 3) different values of the field of view s_v. A global goal is specified by a center line of the motion space. Performances of the algorithms are evaluated according to the following criteria: 1) the collision-warning rate, 2) the average vision and control processing time $T_c(t)$, and 3) the deviation of the local path from the global route. A collision-warning signal is generated when the distance between the robot and an obstacle is less than $s_a(t)$. The summation of warning signals is divided by the total number of motion steps to get the collision-warning rate. The average deviation indicates how well the robot has kept the designated global route when making maneuvers to avoid obstacles.

The performance of the HMM for obstacle-motion prediction was compared with a deterministic prediction approach (DPA) [6]. The results of the simulation expose many important properties. Fig. 7 shows a sequence of graphical display of the simulation, where obstacles are presented in different shapes and patterns. Simulation results are illustrated as follows. For the convenience of comparison, all measurements are scaled to a range of 1 to 10. Fig. 8(a) shows the collision-warning rate with respect to the changes of $s_v(t)$ for the HMM and the DPA. Fig. 8(b) shows the collision warning rate with respect to $d_{\text{obj}}(t)$ of the two approaches. Fig. 9(a) shows the average $T_c(t)$ with respect to the changes of $s_v(t)$ for the HMM and the DPA. Fig. 9(b) shows the average $T_c(t)$ with respect to the obstacle density $d_{\text{obj}}(t)$ of the two approaches. Fig. 10(a) shows the average deviation of the motion trajectory with respect to the changes of $s_v(t)$ for the HMM and the DPA. Fig. 10(b) shows the average deviation with respect to $d_{\text{obj}}(t)$ of the two approaches.

It can be easily seen from Fig. 8 that the HMM approach has a

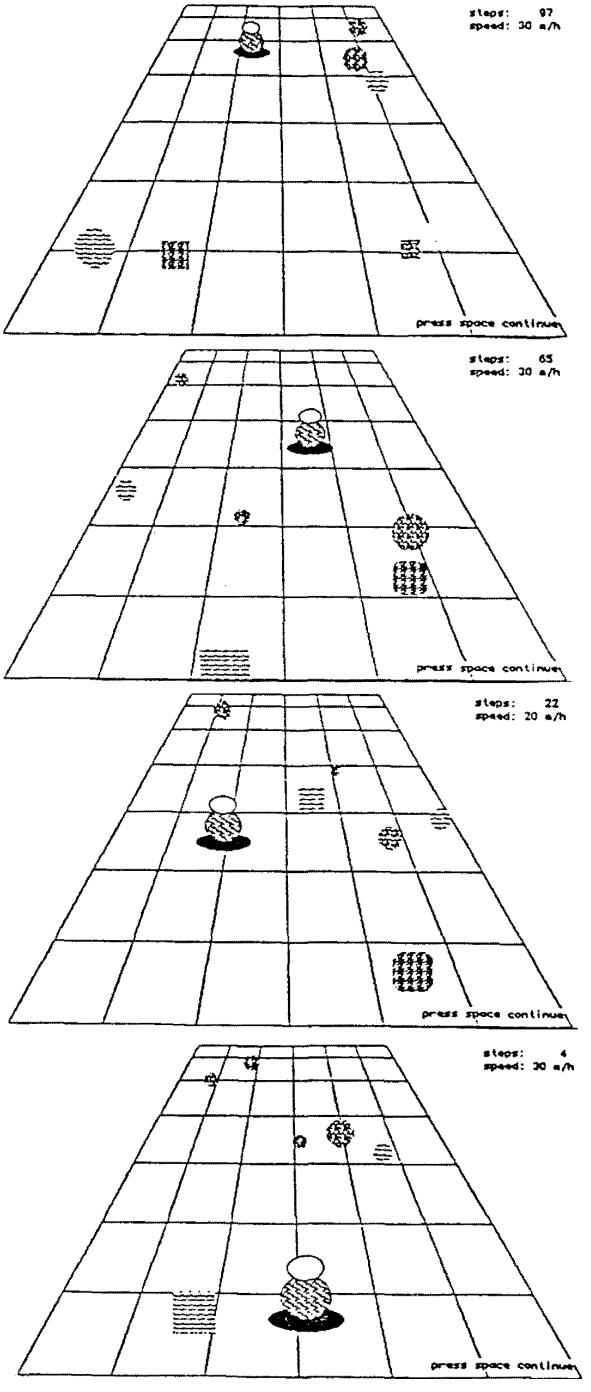

Fig. 7. A sequence of display of the motion simulation.

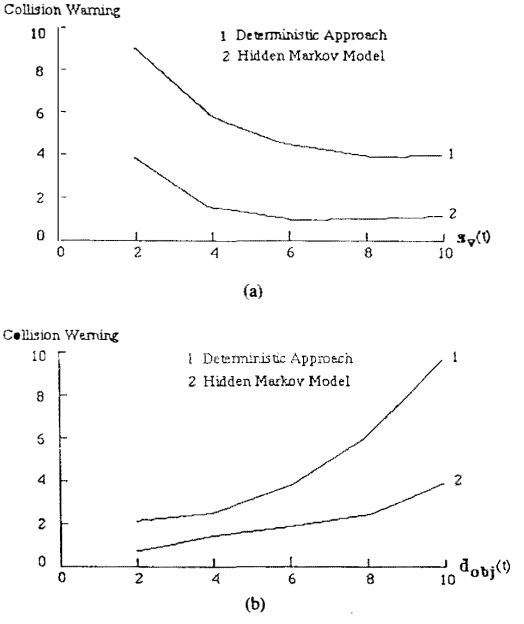

Fig. 8. Collision-warning rate with respect to (a) $s_v(t)$ and (b) $d_{obj}(t)$.

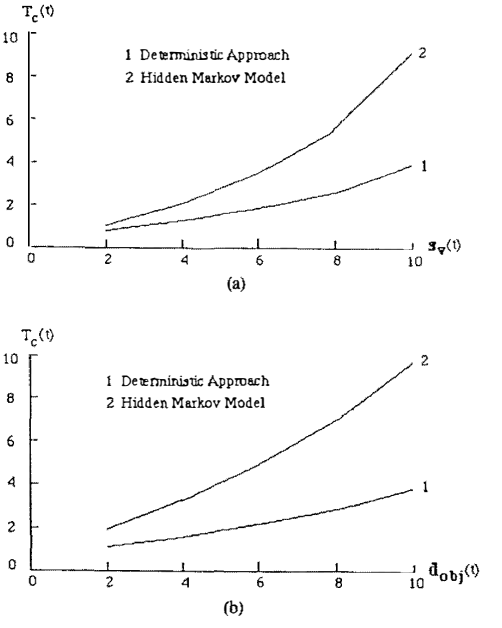

Fig. 9. Average $T_c(t)$ with respect to (a) $s_v(t)$ and (b) $d_{obj}(t)$.

lower collision-warning rate than the DPA. The slope of the curve is also slower in the HMM approach than the DPA when the $d_{obj}(t)$ increases. On the other hand, as a tradeoff for safety, the results also show that the $T_c(t)$ and deviation increases more in HMM than the DPA. This is worthwhile as safety is the main consideration in most robot motion environments. From the simulation results, we also observe that, when the $s_v(t)$ is beyond a certain range, its increase has only a very limited effect on the improvement of the motion parameters. This is because collision avoidance in dynamic environments is basically a local operation. When the $s_v(t)$ is sufficiently large, compared to the $s_r(t)$, the further increase of $s_v(t)$ does not proportionally improve the motion performance. It is also worth indicating that the ratio of s_v versus $s_r(t)$, expressed as

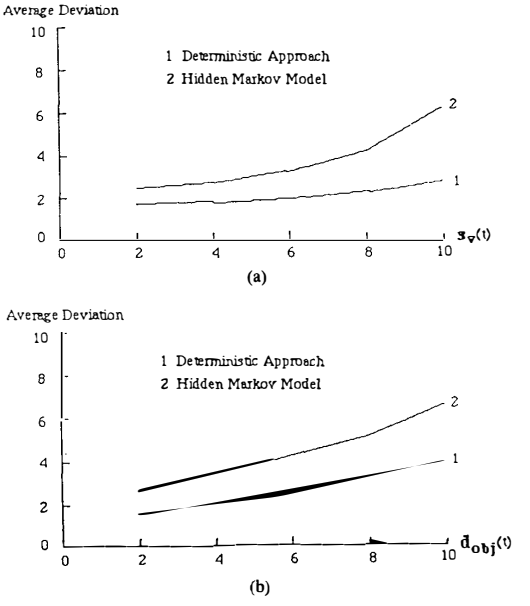

Fig. 10. Average deviation of trajectory with respect to (a) $s_v(t)$ and (b) $d_{obj}(t)$.

$|s_v|/|s_r(t)|$, reflects the controllability of the visual guidance system at a certain level. The larger this ratio is, the easier for the robot to avoid collisions.

VII. Conclusions

The problem of visual guidance and dynamic obstacle avoidance of a mobile robot or autonomous vehicle navigating in an unknown environment has been investigated. For collision avoidance in dynamic environments, where both the robot and the obstacles in the scene are moving, one critical problem is to accurately predict the motions of the obstacles. The prediction has to be made using only the limited amount of information from the observations. The uncertain nature of obstacle motion makes such a prediction difficult. Basic motion models of obstacles have been studied in this paper. The HMM for obstacle-motion prediction is described and simulated. The visual guidance and motion-control algorithms discussed in this paper have the following properties: 1) the stochastic model provides an appropriate description of obstacle motions in dynamic environments; 2) the HMM for obstacle-motion prediction reflects the stochastic nature of obstacle motions and is consistent with the motion characteristics of the obstacles; and 3) the trajectory-guided local-path planning algorithm unifies the visual processing and motion control processes in a systematic representation of candidate trajectories. The algorithm is especially beneficial for being able to pursue several potential paths in parallel.

Acknowledgment

The author wishes to thank C. F. Mott and K. Zebolsky for their assistance in preparing this manuscript. Thanks also to the referees for their valuable comments and suggestions.

References

[1] A. Chattergy, "Some heuristics for the motion of a robot," *Int. J. Robotics Res.*, vol. 4, no. 1, pp. 59-66, Spring 1985.

[2] J. L. Crowley, "Motion for an intelligent mobile robot," in *Proc. 1st Conf. Artificial Intell. Application* (Denver, CO), Dec. 1984, pp. 51-56.

[3] A. Elfes, "Using occupancy grids for mobile robot perception and navigation," *IEEE Computer*, pp. 46-57, June 1989.

[4] L. Gouzenes, "Strategies for solving collision-free trajectories problems for mobile and manipulator robots," *Int. J. Robotics Res.*, vol. 3, no. 4, pp. 51-65, Winter 1984.

[5] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *Int. J. Robotics Res.*, vol. 5, no. 1, pp. 90-98, Spring 1986.

[6] N. K. Loh, Q. Zhu, A. K. Chang, and L. McTamaney, "Dynamic obstacle avoidance of mobile robots," in *Proc. 2nd Ann. Mini-Symp. Unmanned Ground Vehicles* (Sterling Heights, MI), Sept. 1989.

[7] T. Lozano-Perez and M. A. Wesley, "An algorithm for planning collision-free paths among polyhedral obstacles," *Commun. Ass. Comput. Math.*, vol. 22, no. 10, pp. 560-570, Oct. 1979.

[8] J. J. Nitao and A. M. Parodi, "An intelligent pilot for an autonomous vehicle system," in *Proc. IEEE 2nd Conf. Artificial Intell. Applications* (Miami Beach, FL), 1985, pp. 176-183.

[9] L. R. Rabiner and B. H. Juang, "An introduction to hidden Markov models," *IEEE ASSP Mag.*, vol. 3, no. 1, pp. 4-16, Jan. 1986.

[10] S. Tsugawa, T. Hirose, and T. Yatabe, "An intelligent vehicle with obstacle detection and navigation functions," in *Proc. IECON'84*, 1984, pp. 303-308.

[11] Q. Zhu, "Structure pyramids for representing and locating moving obstacles," in *Proc. IEEE CVPR Conf.*, June 1988, pp. 832-837.

[12] —, "A stochastic algorithm for visual guidance of robot motion in a dynamic environment," in *Proc. IEEE Int. Conf. Syst. Eng.* (Pittsburgh, PA, Aug. 9-11, 1990).

Weld Pool Edge Detection for Automated Control of Welding

D. Brzakovic and D. T. Khani

***Abstract*—This work describes a vision system that determines the edges of the weld pool in sequences of gas-tungsten-arc welding images acquired by a coaxial viewing system. The vision system employs a transformation that maps the edge of a weld pool into a vertical line. The weld pool edge is detected in the transform domain by employing a directional filter, which retains only intensity changes of interest, and a one-dimensional edge detector. The edge of the weld pool, in the physical domain, is determined using the inverse transformation. The transformation employs parameters that are updated when processing a sequence of images and are initially determined by analyzing the first image frame in the physical domain.**

***Key Words*—Directional filtering, edge detection, inverse transformation, temporal changes, transformation.**

I. Introduction

For the past two decades there has been a great deal of interest in automating various welding processes. The objective of the automation is to ease the job of the machine operator and in some cases

Manuscript received October 1, 1989; revised January 25, 1991. This work was supported by the Center for Measurement and Control, University of Tennessee, and by the National Science Foundation under Grant CDR-8611139.

D. Brzakovic is with the Department of Electrical and Computer Engineering, University of Tennessee, Knoxville, TN 37996.

D. T. Khani was with the Department of Electrical and Computer Engineering, University of Tennessee, Knoxville, TN 37996. He is now with Martin Marietta Electronic Systems, Orlando, FL 32862.