CAR-TR-565 DACA76-89-C-0019
CS-TR-2717 June 1991

NAVIGATION WITH UNCERTAINTY:
I. Reaching a Goal in a High Collision Risk Region

Philippe Burlina
Daniel DeMenthon
Larry S. Davis

Computer Vision Laboratory
Center for Automation Research
University of Maryland
College Park, MD 20742-3411

Abstract

We present a probabilistic method for noisy sensor based robotic navigation in dynamic environments. The method generates an optimal trajectory by considering as optimality criteria the probability of not colliding with the obstacles and the probability of accessing an operational position with respect to a moving target object.

Estimates of the obstacle's kinematic parameters and measures of confidence in these estimates are used to produce the probability of collision associated with any robot displacement. The probability of collision is derived in two steps: a stochastic model is defined in the kinematic state space of the obstacles, and collision events are given simple geometric characterizations in this state space.

The support of the Defense Advanced Research Projects Agency (ARPA Order No. 6989) and the U.S. Army Engineer Topographic Laboratories under Contract DACA76-89-C-0019 is gratefully acknowledged, as is the help of Sandy German in preparing this paper.

1 Introduction

1.1 Trajectory Control: Dynamic vs. Static Environments

We propose a probabilistic approach to sensor based navigation in dynamic and uncertain environments. The general problem of trajectory control that we address here can be stated as follows:

> An autonomous robot must reach a moving goal point while avoiding surrounding moving obstacles. The motions of the goal and the obstacles are not known *a priori*, but *observations* by sensors carried by the robot provide partial and noisy information about the positions of the goal and the obstacles. The problem is to provide kinematic parameters for a trajectory that will safely carry the robot to the goal.

The path planning problem in known static environments has been extensively addressed in the past. Methods for addressing this problem have primarily relied on purely geometric tools using the idea of configuration space (see ([22, 25]) and on field-optimization methods ([18]). Variants of these methods have been proposed to include certain cases of path planning in known dynamic environments ([14]). Other approaches made use of probabilistic techniques for learning planned paths by updating transition probabilities in a state graph ([12]).

Much less has been done in the case of unknown static environments. In this case an important tool is the *occupancy grid*, used for sensory data fusion and navigation ([11, 19, 20]). The system keeps a map of the space and each space-cell's probability of occupancy is calculated based on models of the sensors' uncertainties. Bayes' rule is used for updating this probability when new sensor readings are collected for a particular neighborhood in the space. Another interesting and related approach is given in [5]: a polar histogram representing a *polar obstacle density* in each direction is compiled from range readings. A similar approach to obstacle avoidance for an autonomous helicopter is proposed in [7]. Very little study has been done yet in the case of unknown dynamic environments. *Collision zones* are used in [17]: simple cases are investigated where moving obstacles are replaced by stationary collision zones as input to classic collision avoidance algorithms. Other approaches to this problem include ([3, 16]).

Our solution to path planning is different from what has been proposed up to now. It accounts for the fact that the environment is dynamic and unknown. The robot relies solely on noisy sensory input. Our treatment of the problem remains general. The method is independent of the nature of the sensors or the particular type of motion parameter recovery scheme used by the robotic system. Instead, we are primarily concerned with the use of measures of uncertainty on the motion parameters in the planning of a collision free trajectory.

The general idea is as follows. Past and present noisy sensor observations of the environment are used to predict future positions of the goal and obstacles. These predictions are used for planning a trajectory toward the goal while avoiding collision with the obstacles. Measures of confidence in the predictions are translated into measures of the likelihood of collision along the trajectories. These measures of risk are used in deriving a safe trajectory toward the goal.

1.2 Uncertainty in Measurements, Modeling, Predictions

The robot has no *a priori* knowledge of the environment. It must rely on monitoring the surrounding obstacles and tracking their motions. Due to modeling and observation errors, the motions of the goal and of the obstacles cannot be known exactly. Errors and uncertainty are common to all engineering problems and are usually taken care of beforehand: a preliminary analysis of the combination and cascading of the errors is carried out so as to quantify the global effect of these errors on the operation of the system. Operational tolerances are then determined and the errors are reduced to fit within these acceptable tolerances. Such a process is not always possible when there is insufficient knowledge of the uncertainty, when this uncertainty varies according to the situation, when it is impossible to guarantee bounds on the errors beforehand, and when acting upon a noncooperative and largely unpredictable target. Such a situation occurs, for example, when designing the guidance system for a plane whose task is to track and follow a hostile plane. Probability theory is a powerful tool in the context of this problem. We propose to use probability theory for describing the motion uncertainty, and to use probabilistic measures for determining the optimal trajectory toward a goal.

Different criteria can be considered when assessing the admissibility of a trajectory—for example, minimum time, minimum energy, or safety. Among these criteria we focus on those that measure the safety of a trajectory and its likelihood of leading to the expected goal. To characterize this admissibility we are interested in obtaining a *collision risk* measure and an *expected success* measure for a given path. Such measures could be taken to be, respectively, the probability of collision and the probability of reaching the goal, or the expected value of the minimum distance to the goal along the trajectory.

1.3 Two Different Approaches to Probabilistic Navigation

Two different and symmetric approaches to probabilistic navigation are being investigated. The first method is presented in this paper. This method is based on producing for every destination of interest at a given time an associated probability of collision which is later used to derive an optimal trajectory. In a second approach, presented in [6], the system produces an overall region of space where the probability of encountering any obstacle is kept, at each instant, less than a predefined tolerable value. The second method takes as input an upper bound on the probability threshold and outputs a space-time domain in which the collision risk is tolerable. These approaches can be motivated by the existence of different navigation tasks or situations.

In some situations the collision risk measure and the expected success measure are not unrelated. Indeed, they are very much dependent in cases where getting closer to the goal necessarily implies higher chances of collision. In such cases a robot that does not take risks will never reach the goal. This situation occurs, for instance, when the goal is located on one of the moving objects.

In contrast, in certain other situations the goal itself is not a source of risk. The navigation strategy to be used depends strongly on the type of navigation situation. In the first case the method deriving probabilities of collision is more appropriate; for the second case the method deriving overall regions with tolerable collision risk levels may be sufficient.

To summarize, the first method addresses the task of reaching a goal in a higher collision risk region, while the second method specializes to the task of avoiding regions of higher collision risk. Practically, those tasks are complementary, and the two methods work together in a general navigation system.

1.4 Overview of the Methods and Summary of this Paper

A Kalman filter is used to obtain the mean and covariance of the distribution of the present and future kinematic state parameters of the obstacles and goal. The probability of collision is derived as follows. We first consider the situation of a unique mobile obstacle. We want to find the probability of collision associated with an elementary displacement of the robot during a time interval related to the obstacle motion. We define a probability distribution on the obstacle kinematic state space. We determine in the obstacle frame of reference the relative displacements that result in a collision during the time interval. From this we characterize the domain in the obstacle kinematic state space that corresponds to a collision event. From this domain and the probability function of the obstacle's kinematic parameters we derive the probability of collision associated with a given robot displacement. This probability is incorporated in a cost function to determine the optimal displacement.

The paper is organized as follows. In Section 2 the method using the probability of collision is presented. We define the assumptions for the problem in Section 2.1. In Section 2.2, a geometric characterization of the collision is given in the frame of reference of the obstacle. The motion model, probabilistic model for the motion state vector, and Kalman filtering tools are briefly introduced in Section 2.3. The calculation of the probability of collision follows in Section 3. Several cases are considered: in Section 3.1 an error in the obstacle position is assumed. This error is generalized to speed and acceleration in Section 3.2. The general case of translation and rotation error is considered in Section 3.3. Section 4 shows how to use the probability of collision to derive the optimal displacement. Finally, an extension of the method to a longer planning horizon is studied. Section 5 concludes this paper and proposes ideas for further study. In Section 6 we present in more detail the probabilistic assumptions and the use of the Kalman filtering method to derive the distribution parameters of the obstacle's motion state.

2 Derivation and Use of the Probability of Collision

We are interested in simultaneously addressing the two following navigational tasks:

- obstacle avoidance (a passive navigational task)
- navigation toward the goal (an active navigational task)

We study a situation in which the robot is assigned the task of reaching a goal point located on the periphery of the obstacle which we call the *target object*. This particular situation captures the types of issues that are involved in navigation in dynamic environments. The robot cannot just avoid the moving object as in classical obstacle avoidance problems. It must go to the vicinity of the object to reach the goal point while avoiding certain parts of the object (here: its protrusions).[1]

Navigation in this context is understood as finding the successive positions that carry the robot toward the goal with an acceptable collision risk level. This section is dedicated to the calculation of the probability of collision and its use as a safety measure to produce an optimal trajectory. Let us first introduce the preliminary assumptions.

2.1 Modeling the Problem

The only information known a priori about the object to be reached (the *target object*) is the following:

- the shape of the object,

[1] Note that this particular problem setup could also apply to the situation of a plane trying to reach an observation or firing position with respect to a hostile plane while avoiding regions in which it could be shot.

- bounds on its speed and normal/tangential acceleration.

The precise motion of the target object is unknown. It may combine rotations and translations. The proposed approach makes the following geometrical and differential simplifications:

- The world is two dimensional.
- The trajectory of the robot is approximated as a polyline in the plane: between any two neighbor vertices of the polyline, a constant time interval δt has elapsed, in which the robot displacement is at a constant velocity. At any vertex the robot can change its velocity (direction and speed) within limits defined by mechanical and inertial constraints.
- The target object may be represented by a not necessarily convex polygon. The robot is a point.

Based on this framework, the problem consists of determining the optimal robot destination D_{n+1}. As stated previously, the optimality criteria we are interested in are proximity to the goal and probability of collision. A solution to the trajectory control problem is defined to be the solution of the following optimization problem:

> At present time t_n, based on the position D_n of the robot and the estimated target motion state, we wish to find the optimal *reachable* destination for the next time instant D_{n+1}. The optimality criteria are collision safety and proximity to the intermediate goal position.

The destination D_{n+1} is *reachable* if the speed V_n and angle a_n of the robot required to get to D_{n+1} satisfy the predefined dynamic bounds. The intermediate goal is the estimated position of the goal at the next time instant if it is reachable, or the reachable position of the robot that would bring it to the goal point in the smallest number of time steps. We will later show how the proximity of a destination to the intermediate goal can be measured probabilistically and be integrated into a cost function.

We now turn our attention to the derivation of the probability of collision. This is done in three steps. We first give a geometric characterization of the displacements that correspond to a collision. From the Kalman filter we get the parameters (mean and covariance) of the probability distribution of the target's kinematic state. The geometric characterization of collision defines a domain in the target motion state space, in which we integrate the motion state distributions to find the probability of collision.

2.2 Geometric Characterization of Collision Events

The robot in position D_n at time t_n plans to move to position D_{n+1} at time t_{n+1}. We want to obtain a simple geometric characterization of the pairs (D_n, D_{n+1}) that result in a collision. This characterization is done in the target frame of reference. We consider the corresponding robot coordinates D_n^r and D_{n+1}^r in the target frame of reference.

We first compute a differential approximation of the robot trajectory in the target frame of reference. To simplify the analysis of the collision, we have assumed (in the previous subsection) that the robot has a uniform rectilinear motion between time t_n and t_{n+1}. This can done without loss of generality: the speed and acceleration of the robot being bounded, we can choose the time interval δt small enough so that the trajectory of the robot can be differentially approximated as rectilinear and uniform. Since the rotation speed and the normal acceleration of the target are bounded, the changes in the target orientation during a time interval can be made small enough (again by choosing δt small enough) so that the robot's trajectory in the frame of reference of the target during a time interval is arbitrarily close to a line segment. In this context it is valid to approximate the trajectory of the robot in the target reference frame during a time interval by a rectilinear trajectory (D^r_{n+1}, D^r_{n+1}) (Figures 1(a) and 1(b)).

Figure 1: (a)Elementary displacement of the target and robot in the global frame of reference. (b) Corresponding trajectory of the robot in the frame of reference of the target approximated by a rectilinear trajectory during a time interval.

If the relative trajectory of the robot in the target frame is assumed rectilinear, given the present relative position D^r_n and destination D^r_{n+1} of the robot, there is a collision if the trajectory segment (D^r_n, D^r_{n+1}) intersects the target in the target frame (Figure 2 (b)) and no collision otherwise (Figure 2 (a)).

In this case the probability of collision $PC(D_n, D_{n+1})$ is equal to the probability that the corresponding elementary displacement (D^r_n, D^r_{n+1}) intersects the target, i.e.

$$PC(D_n, D_{n+1}) = P\left\{(D^r_n, D^r_{n+1}) \textit{intersects the target}\right\} \cdot \tag{1}$$

Let us now define the *shadow* of the target with respect to a point D as the domain of space that an observer in D cannot see because it is hidden by the target. We denote this subset by $Sh(D)$. Figure 3 shows that the segment (D^r_n, D^r_{n+1}) intersects the obstacle if D^r_{n+1} is in the *shadow* of the obstacle with respect to D^r_n, i.e.

$$\left[D^r_{n+1} \in Sh(D^r_n)\right] \Leftrightarrow \left[\textit{segment } (D^r_n, D^r_{n+1}) \textit{ intersects the target}\right] \cdot\cdot \tag{2}$$

Then the probability of collision based on the observations accumulated so far is given by

$$PC\left(D_n, D_{n+1}\right) = P\left(D^r_{n+1} \in Sh(D^r_n)\right)$$

Figure 2: There is a collision if the trajectory segment (D_n^r, D_{n+1}^r) intersects the target in the target frame (b) and no collision otherwise (a).

2.3 Target Motion State Distribution

We now introduce the probabilistic model describing the uncertainty in the target motion state and the basic tools derived from the Kalman filter that will be needed in the subsequent probability calculation. A more thorough presentation of these tools will be made in Section 4.

The pose of the target in the plane is determined by three parameters: the position coordinates $[x_n, y_n]^T$ in the global reference system and the orientation α_n of the target. The components of the kinematic state vector are the pose parameters and their first and second derivatives, so as to model the target trajectory by trajectory elements with piecewise constant accelerations. The kinematic state evolution of the target is modeled locally as a first order autoregressive time invariant model

$$S_{n+1} = AS_n + V_n \tag{3}$$

where we have

- $S_n = [U_n, \Theta_n]^T$, the target's motion state vector,
- $U_n = \left[X_n, \dot{X}_n, \ddot{X}_n\right]^T$, where $X_n, \dot{X}_n, \ddot{X}_n$ are respectively the position, velocity and acceleration vectors of the target's center of inertia,
- $X_n = [x_n, y_n]^T$, the position of the target's center of inertia,
- $\Theta_n = [\alpha_n, \dot{\alpha}_n, \ddot{\alpha}_n]^T$, the target's orientation angle and its first and second derivatives,
- $V_n = \left[V_n^U, V_n^\Theta\right]^T$, the error in the target motion model; it is assumed to have a zero mean Gaussian distribution and covariance equal to Υ_n, and to be independent,
- A = the model transition matrix.

The motion state S_n is not accessible directly, but is instead derived from observations. The observations are obtained from sensor readings. For instance, observations can consist of the image

Figure 3: Collision occurs when the destination is in the shadow of the starting point.

coordinates of the target vertices. The equation relating the state vector S_n and the observation vector O_n is a nonlinear equation

$$O_n \quad = \quad B(S_n) + W_n \tag{4}$$

where

- $O_n = (m_1, m_2, \ldots, m_N)$, the observation vector at time step t_n, composed of the coordinates of the projections m_i of the N vertices $(M_1, M_2, \ldots, M_N)$,
- $O^n = (O_1, O_2, \ldots, O_n)$, all the observations made up to time t_n,
- $W_n =$ a zero mean Gaussian distributed independent error sequence on the observations with covariance matrix equal to Ω_k, and independent of the model error V_n,
- $B(.) =$ the observation function.

Based on past and present observations O^n, the Kalman filtering method provides the optimal estimates of the present time target motion state $S_{n|n}$ and the optimal prediction of the target motion state one step forward in the future $S_{n+1|n}$, along with their respective covariance matrices $\Sigma_{n|n}$ and $\Sigma_{n+1|n}$. As a result of assuming Gaussian errors in the motion and observation equations, the conditional probability distribution of the state vector conditioned by the observations $f_{S_n|O^n}(.)$ is Gaussian with mean $S_{n|n}$ and covariance $\Sigma_{n|n}$,[2] i.e.

$$f_{S_n|O^n}(S) = N_{(S_{n|n},\Sigma_{n|n})}(S)$$

where $N(E[Y],\Sigma^Y)(.)$ denotes the Gaussian distribution having mean $E[Y]$ and covariance Σ^Y, i.e.

$$N_{(E[Y],\Sigma^Y)}(Y) = \frac{1}{\sqrt{(2\pi)^p \det(\Sigma^Y)}} \exp \frac{-(Y-E[Y])^T(\Sigma^Y)^{-1}(Y-E[Y])}{2}.$$

where p is the dimension of Y. Similarly, the target kinematic parameters S_{n+1} at time t_{n+1} conditioned on the observations made up to the present time are Gaussian with mean equal to the predicted state $S_{n+1|n}$, and with covariance matrix equal to the error covariance $\Sigma_{n+1|n}$, i.e.

$$f_{S_{n+1}|O^n}(S) = N_{(S_{n+1|n},\Sigma_{n+1|n})}(S).$$

We now turn to the derivation of the probability of collision.[3] In the remainder of this paper, the probabilities and probability densities that will be derived are conditional densities. These probabilities and densities are conditioned by all the sensor observations O_n that have been made up to the present time, and should be understood as such even when the notations $P\{.|O^n\}$, $f_{.|O^n}(.)$ and $PC(.|O^n)$ are omitted for simplicity.

3 Calculation of the Probability of Collision

At the present time t_n, we want to calculate the probability of collision for a given displacement (D_n, D_{n+1}). We have the probability distributions for the target motion parameters S_n and S_{n+1} at time t_n and t_{n+1}. If we consider the corresponding positions of the robot *in the reference frame of the target*, then the target position is completely defined in this frame, and the relative positions of the robot D^r_n and D^r_{n+1} are random vectors directly related to the target kinematic states S_n and S_{n+1} in the global frame. The probability of collision can be obtained by integrating over the domain of displacements (D^r_n, D^r_{n+1}) corresponding to a collision:

$$PC(D_n, D_{n+1}) \quad = \quad \int_{\forall (D_1,D_2)|D_2 \in Sh(D_1)} f_{D^r_n,D^r_{n+1}}(D_1,D_2)\, dD_1 dD_2. \tag{5}$$

Unfortunately, we cannot do so directly. As said before, we have access to the marginal distributions of S_{n+1} and S_n. But the kinematic state S_{n+1} is not independent of S_n, so that it is difficult to derive their joint distribution. Equivalently, it is difficult to derive a joint distribution $f_{D^r_n,D^r_{n+1}}(.,.)$ because the random vector D^r_{n+1} is not independent of D^r_n. The solution is to use the autoregressive motion state evolution equation (3) to express D^r_n and D^r_{n+1} as functions of independent random variables whose distributions are known. Their joint distributions are then expressed as the product of the marginal distributions of the independent random variables. We may then integrate to find the probability of collision.

We consider the derivation and calculation for cases having increasing degrees of complexity. We investigate first the case of errors only in the position component X_n of S_n, then the more

[2] See Section 4 for more details

[3] See Section 4 for a complete discussion of the probabilistic assumptions.

general case of errors in U_n (translation errors in position, speed and acceleration), and finally the global case of errors in all components of the kinematic state S_n (translation and rotation).

Let us first introduce some preliminary definitions. Recall the global evolution equation for the target kinematic state S_n

$$S_{n+1} = A\, S_n + V_n. \tag{6}$$

The expanded form showing the translation and rotation contributions is

$$\begin{bmatrix} U_{n+1} \\ \Theta_{n+1} \end{bmatrix} = \begin{bmatrix} A^U & 0_{6\times 3} \\ 0_{3\times 6} & A^\Theta \end{bmatrix} \begin{bmatrix} U_n \\ \Theta_n \end{bmatrix} + \begin{bmatrix} V_n^U \\ V_n^\Theta \end{bmatrix}. \tag{7}$$

The translation and rotations contributions can be written separately:

$$U_{n+1} = A^U U_n + V_n^U \tag{8}$$
$$\Theta_{n+1} = A^\Theta \Theta_n + V_n^\Theta. \tag{9}$$

When considering only translation errors, we are primarily interested in the subsystem (8). Since a uniformly accelerated movement model was assumed (Sections 2.2 and 2.4), equation (8) is equivalent to

$$\begin{bmatrix} X_{n+1} \\ \dot{X}_{n+1} \\ \ddot{X}_{n+1} \end{bmatrix} = \begin{bmatrix} X_n + \dot{X}_n + \frac{\ddot{X}_n}{2} \\ \dot{X}_n + \ddot{X}_n \\ \ddot{X}_n \end{bmatrix} + \begin{bmatrix} V_n^X \\ V_n^{\dot{X}} \\ V_n^{\ddot{X}} \end{bmatrix}. \tag{10}$$

Therefore in equation (8) A^U can be written in a compact form as

$$A^U = \begin{bmatrix} E_1 + E_2 + \frac{1}{2}E_3 \\ E_2 + E_3 \\ E_3 \end{bmatrix} \tag{11}$$

where we define E_1, E_2, E_3 to be respectively

$$E_1 = \begin{bmatrix} I_{2\times 2} & 0_{2\times 2} & 0_{2\times 2} \end{bmatrix}, \tag{12}$$

$$E_2 = \begin{bmatrix} 0_{2\times 2} & I_{2\times 2} & 0_{2\times 2} \end{bmatrix}, \tag{13}$$

$$E_3 = \begin{bmatrix} 0_{2\times 2} & 0_{2\times 2} & I_{2\times 2} \end{bmatrix}. \tag{14}$$

It will be useful to also subdivide the covariance matrix $\Sigma_{n|n}$ of the estimated kinematic state S_n into translation and rotation components

$$\Sigma_{n|n} = \begin{bmatrix} \Sigma_n^U & \Sigma_n^{U\Theta} \\ \Sigma_n^{\Theta U} & \Sigma_n^\Theta \end{bmatrix} \tag{15}$$

and into position, speed and acceleration:

$$\Sigma_n^U = \begin{bmatrix} \Sigma_n^X & \Sigma_n^{X\dot{X}} & \Sigma_n^{X\ddot{X}} \\ \Sigma_n^{\dot{X}X} & \Sigma_n^{\dot{X}} & \Sigma_n^{\dot{X}\ddot{X}} \\ \Sigma_n^{\ddot{X}X} & \Sigma_n^{\ddot{X}\dot{X}} & \Sigma_n^{\ddot{X}} \end{bmatrix}. \tag{16}$$

We finally distinguish between the translation and rotation components of the error vector in the evolution equation (3) V_n and its covariance Υ_n as

$$V_n = \begin{bmatrix} V_n^U \\ V_n^\Theta \end{bmatrix} \quad , \quad V_n^U = \begin{bmatrix} V_n^X \\ V_n^{\dot{X}} \\ V_n^{\ddot{X}} \end{bmatrix}. \tag{17}$$

$$\Upsilon_n = \begin{bmatrix} \Upsilon_n^U & \Upsilon_n^{U\Theta} \\ \Upsilon_n^{\Theta U} & \Upsilon_n^\Theta \end{bmatrix} \tag{18}$$

We now focus the presentation on the special cases we defined previously.

3.1 Probability of Collision for Position Errors

3.1.1 Derivation

Let us turn first to the case of translational errors only, and let us further suppose that all model errors in the translation state component can be considered negligible compared to the position error in X_n. This corresponds to setting the evolution model translation errors V_n^U and rotation errors V_n^Θ in equations (6) and (10) to

$$V_n^\Theta = 0_{3\times 1} \quad , \quad V_n^U = \begin{bmatrix} V_n^X \\ V_n^{\dot{X}} \\ V_n^{\ddot{X}} \end{bmatrix} = \begin{bmatrix} V_n^X \\ 0_{2\times 2} \\ 0_{2\times 2} \end{bmatrix} \tag{19}$$

and restricting the covariance matrix Υ_n^U of the error vector V_n^U to

$$\Upsilon_n = \begin{bmatrix} \Upsilon_n^U & 0_{6\times 2} \\ 0_{2\times 6} & 0_{2\times 2} \end{bmatrix}. \tag{20}$$

$$\Upsilon_n^U = \begin{bmatrix} \Upsilon_n^X & 0_{2\times 4} \\ 0_{4\times 2} & 0_{4\times 4} \end{bmatrix}. \tag{21}$$

As a result of this choice, all components of the estimation covariance $\Sigma_{n|n}$ in equations (15, 16) are zero except Σ_n^X, i.e., X_n is the only random part in S_n.

In the target frame the vector D_n^r, i.e. the relative position of the robot with respect to the target, is given by the change of reference frame

$$D_n^r = R_{-\alpha_n}(D_n - X_n) \tag{22}$$

where R_α denotes the rotation matrix of angle α. D_n and α_n being fixed, the random vector D_n^r depends only on X_n. Equation (22) indicates that D_n^r is an affine function of X_n which we may rewrite as

$$D_n^r = L_1(X_n).$$

As said before, D_n^r and D_{n+1}^r being dependent, to calculate the probability of collision we must express them as functions of independent random vectors. Here we do so by extracting from D_{n+1}^r the participation of D_n^r. From equations (22) and (10) D_{n+1}^r, the relative position of the robot at time t_{n+1} can be expressed as an affine function of the relative position D^r_n and of the motion model error V^X_n:

$$
\begin{aligned}
D^r_{n+1} &= R_{-\alpha_{n+1}}(D_{n+1} - X_{n+1}) \\
&= R_{-\alpha_{n+1}}(D_{n+1} - X_n - \dot{X}_n - \frac{\ddot{X}_n}{2} - V^X_n) \\
&= R_{-\alpha_{n+1}}(D_{n+1} + R_{\alpha_n} D^r_n - D_n - \dot{X}_n - \frac{\ddot{X}_n}{2} - V^X_n)
\end{aligned}
\quad (23)
$$

The uncertainty at the present time in the position of the robot in the target reference frame and the uncertainty in the motion model evolution are carried over to the relative position of the robot at the next time instant.

From equation (24) we rewrite D^r_{n+1} as the sum of the affine functions of the two independent random vectors D^r_n and V^X_n,

$$D^r_{n+1} = L_2(D^r_n) + L_3(V^X_n) \quad (24)$$

where

$$L_2(D^r_n) = R_{(\alpha_n - \alpha_{n+1})} D^r_n + R_{-\alpha_{n+1}}(D_{n+1} - D_n - \dot{X}_n - \frac{\ddot{X}_n}{2}) \quad (25)$$

$$L_3(V_n) = -R_{-\alpha_{n+1}} V^X_n. \quad (26)$$

D^r_n and V^X_n are indeed independent since D^r_n depends only on X_n and as a result of the stochastic model embedded in the Kalman filter formulation, X_n and V^X_n are independent random vectors. V^X_n has a Gaussian distribution, i.e.

$$f_{V^X_n}(V^X) = N_{(0,\Upsilon^X_n)}(V^X)$$

with Υ^X_n given by the probabilistic model. As discussed in Section 2.4, X_n is Gaussian. Therefore D^r_n, being an affine function of X_n, is also Gaussian:

$$f_{D^r_n}(D^r) = N_{(E[D^r_n],\Sigma^{D^r}_n)}(D^r)$$

with mean $E[D^r_n]$ and variance $\Sigma^{D^r}_n$ equal to

$$E[D^r_n] = L_1(X_{n|n}) \quad (27)$$

$$\Sigma^{D^r}_n = R_{-\alpha_n} \Sigma^X_n R_{\alpha_n} \quad (28)$$

where $X_{n|n}$ and Σ^X_n are the outputs of the Kalman Filter.

We now express the probability of collision. We have a collision event for all pairs (D^r_n, D^r_{n+1}) for which D^r_{n+1} belongs to the shadow of the target with respect to D^r_n. By equation (24) it is equivalent to consider all pairs $\left(D^r_n, V^X_n\right)$ such that

$$V^X_n \in \mathcal{D}(D^r_n)$$

where the domain $\mathcal{D}(D^r_n)$ is equal to

$$\mathcal{D}(D^r_n) = \left\{V^X_n : L_2(D^r_n) + L_3(V^X_n) \in Sh(D^r_n)\right\}. \quad (29)$$

The domain of integration of V_n^X is a function of D_n^r. Thus we proceed as follows:

$$\begin{aligned} PC(D_n, D_{n+1}) &= P\{D_{n+1}^r \in Sh(D_n^r)\} \\ &= \int_{\forall D^r, V^X} P\{V^X \in \mathcal{D}(D^r) | D_n^r = D^r\} f_{D_n^r}(D^r) dD^r \quad . \end{aligned} \tag{30}$$

Since the random vector V_n^X is independent of D_n^r, the probability of V_n^X conditioned on D_n^r is simply equal to the unconditional probability of V_n^X, i.e. we have for a given $D_n^r = D^r$

$$\begin{aligned} P\{V_n^X \in \mathcal{D}(D^r) | D_n^r = D^r\} &= P\{V_n^X \in \mathcal{D}(D^r)\} \\ &= \int_{\mathcal{D}(D^r)} f_{V_n^X}(V^X) dV^X \quad . \end{aligned} \tag{31}$$

Finally using equations (30) and (31) we have

$$PC(D_n, D_{n+1}) = \int_{\forall D^r} f_{D_n^r}(D^r) \left(\int_{\mathcal{D}(D^r)} f_{V_n^X}(V^X) dV^X \right) dD^r \quad . \tag{32}$$

3.1.2 Method of Computation

The first step consists in the determination of the shadow with respect to a given D_n^r. If the obstacle is a polygon, the line segments of the obstacle that control the shape of the shadow include the line segments that are visible from D_n^r. In Figure 4 (a) it is a single line segment, AD. In Figure 4 (b) they are the segments BC, DA, and the line from D_n^r to D. If the point D_n^r were located somewhere else in space, other line segments would define the shadow. We can divide the space around the obstacle into regions over which the same line segments control the shadow. These regions comprise all the points from which the same edges are visible. These are regions for which the obstacle presents the same *aspect*. These regions and their corresponding lists of visible edges can be precomputed for a given obstacle; they are shown in Figure 5. The shadow of the target with respect to any D_n^r is then determined by testing what aspect region of the space the point D_n^r belongs to.

Figure 4: Aspects of the Shadow

We now propose a method for the numerical computation of the probability $PC(D_n, D_{n+1})$ in equation (32). From the Kalman filter we get the mean $X_{n|n}$ and the covariance Σ_n^X of X_n

Figure 5: Regions of space presenting the same shadow aspect

from which we calculate the mean and covariance of D_n^r as in equations (27, 28). Let us suppose without loss of generality that the covariance matrix of D_n^r is already diagonal. Note that this can be accomplished directly at the Kalman filtering stage.[4] We create a sampling grid for the random vector D_n^r on a rectangle $\mathcal{R}_D$ centered at the mean $E[D_n^r]$ with dimensions equal to a few standard deviations (the square roots of the diagonal elements), so that the probability of D_n^r being outside this rectangle is negligible. A rectangle is used because it is a convenient boundary for the numerical integration of the probability distribution $f_{D_n^r}(.)$. The possible positions of D_n^r can be quantized by dividing the rectangle $\mathcal{R}_D$ into cells of area Δa, and considering only positions of D_n^r at centers of cells. The probability for D_n to belong to a cell i centered at $[x_i, y_i]$ is approximated by

$$f_{D_n^r}([x_i, y_i]^T)\Delta a$$

which is equal to

$$\frac{\Delta a}{\sqrt{(2\pi)^n \det(\Sigma_n^{D^r})}} \exp \frac{-([x_i, y_i]^T - E[D_n^r])^T (\Sigma_n^{D^r})^{-1} ([x_i, y_i]^T - E[D_n^r])}{2}.$$

In the same fashion we create a rectangular sampling grid $\mathcal{R}_V$ for the random vector V_n^X. In this grid, the probability of V_n^X belonging to a cell i centered at $[x_i, y_i]$ of area Δb is approximated by $f_{V_n^X}([x_i, y_i]^T)\Delta b$.

Consider a given relative robot position D_n^r. According to equation (32), if we fix a given position D_n^r the conditional probability of collision is obtained by summing over all the vectors V_n^X

[4] See the method proposed in [24].

in $\mathcal{R}_V$ that yield a collision from D_n^r. The set of such vectors $\mathcal{D}(D_n^r)$ has been shown to be equal to

$$\mathcal{D}(D_n^r) = \left\{ V_n^X \in \mathcal{R}_V : L_2(D_n^r) + L_3(V_n^X) \in Sh(D_n^r) \right\} \cdot \tag{33}$$

The probability of collision for a given D_n^r is thus approximated as

$$\sum_{[x_i,y_i]^T in \mathcal{D}(D_n^r)} f_{V_n^X}([x_i, y_i]^T) \Delta b \cdot \tag{34}$$

This realizes the inner integral in equation (32). To get the overall probability of collision associated with a given displacement (D_n, D_{n+1}) we sum over all D_n^r in $\mathcal{R}_D$ weighted by the corresponding probability of D_n^r, i.e.

$$\sum_{D^r \in \mathcal{R}_D} \Delta a \left(f_{D_n^r}(D^r) \sum_{V^X \in \mathcal{D}(D^r)} f_{V_n^X}(V^X) \Delta b \right) \cdot \tag{35}$$

This double summation over the sampled positions and sampled error vectors resulting in collisions is illustrated in Figure 6.

Figure 6: Summation over the sampled relative robot positions and target motion evolution error vectors resulting in a collision.

It is possible to reduce the complexity of this method by direct analytical calculation of the innermost integral in equation (32). If D_n^r is given, the term

$$L_2(D_n^r) + L_3(V_n^X)$$

is simply an isometry of V_n^X. In this case,

$$L_2(D_n^r) + L_3(\mathcal{R}_V)$$

the image of the rectangle $\mathcal{R}_V$ by this isometry is also a rectangle obtained by simple rotation and translation of $\mathcal{R}_V$. Thus an equivalent calculation of the inner integral is the integration of a normal distribution of mean $L_2(D_n^r)$ and variance $R_{-\alpha_{n+1}} \Upsilon_n R_{\alpha_{n+1}}$ over the polygon defined by

$$[L_2(D_n^r) + L_3(\mathcal{R}_V)] \cap Sh(D_n^r). \tag{36}$$

The intersection of the shadow with a rectangle is a polygon whose vertices can be determined using an algorithm of the type described in [21]. Finally, to get the integral of the Gaussian distribution over a polygon we use the analytical expression given in [1] (see Figure 7).

Figure 7: Inner integral of a Gaussian distribution on a polygonal domain.

3.2 Probability of Collision for Translation Error

3.2.1 Derivation

We now consider the more general case of errors in the translation component of the motion U_n, i.e. we suppose that

$$V_n^{\Theta} = 0_3 \tag{37}$$

and

$$\Upsilon_n = \begin{bmatrix} \Upsilon_n^U & 0_{6\times 2} \\ 0_{2\times 6} & 0_{2\times 2} \end{bmatrix}. \tag{38}$$

We first express the relative position of the robot in the target frame as affine functions of the two independent vectors U_n and V_n^X. As previously we have

$$\begin{aligned} D_n^r &= R_{-\alpha_n}(D_n - X_n) \\ &= R_{-\alpha_n}(D_n - E_1 U_n) \end{aligned} \tag{39}$$

Again D_n^r is an affine function of the random vector U_n that we rewrite as

$$D_n^r = H_1(U_n)\cdot \tag{40}$$

From equations (40, 12, 8) we write the relative position D_{n+1}^r of the robot at time t_{n+1} as an affine function of the translation component of the target motion state U_n and of the motion model error V_n^X as

$$\begin{aligned} D_{n+1}^r &= R_{-\alpha_{n+1}}(D_{n+1} - X_{n+1}) \\ &= R_{-\alpha_{n+1}}(D_{n+1} - (E_1 + E_2 + \frac{E_3}{2})U_n - V_n^X) \end{aligned} \tag{41}$$

which we rewrite more simply as

$$D_{n+1}^r = H_2(U_n) + H_3(V_n^X) \tag{42}$$

where $H_2(.)$ and $H_3(.)$ are the affine functions defined as

$$H_2(U_n) = R_{-\alpha_{n+1}}\left(D_{n+1} - (E_1 + E_2 + \frac{E_3}{2})U_n\right) \tag{43}$$

$$H_3(V_n^X) = -R_{-\alpha_{n+1}} V_n^X\cdot \tag{44}$$

U_n and V_n^X both have a Gaussian distribution and are independent. Their mean and variance are respectively $U_{n|n}$ and $\Sigma_{n|n}^U$ (the output of the Kalman filter) for U_n, and 0 and Υ_n^X (given in the stochastic model) for V_n^X.

Collision occurs for all the pairs $\left(U_n, V_n^X\right)$ for which D_{n+1}^r belongs to the shadow of the target with respect to D_n^r. But from equations (42, 40), collision occurs for all pairs (U_n, V_n^X) for which

$$V_n^X \in \mathcal{D}(U_n)$$

where $\mathcal{D}(U_n)$ is now defined as

$$\mathcal{D}(U_n) = \left\{V_n^X : H_2(U_n) + H_3(V_n^X) \in Sh(H_1(U_n))\right\}\cdot \tag{45}$$

We integrate on this domain to get the probability of collision as

$$\begin{aligned} PC(D_n, D_{n+1}) &= \int_{\forall U, V^X} P\{V^X \in \mathcal{D}(U) | U_n = U\} f_{U_n}(U) dU \\ &= \int_{\forall U} f_{U_n}(U) \left(\int_{\mathcal{D}(U_n)} f_{V_n^X}(V^X) dV^X\right) dU \end{aligned} \tag{46}$$

where

$$f_{V_n^U}(V^X) = N_{(0, \Upsilon_n^X)}(V^U)$$

is given, and

$$f_{U_n}(U) = N_{(U_{n|n}, \Sigma_{n|n}^U)}(U)$$

is derived from the Kalman estimation stage.

3.2.2 Method of Computation

The computation is again done along the same lines as the computation done in the case of position uncertainties. We first create a sampling grid $\mathcal{R}_U$ for the random vector U_n. To do so we first diagonalize the covariance matrix Σ_n^U. We create a rectangular sampling grid $\mathcal{R}_U$ for the random vector U_n centered on its mean $U_{n|n}$, with the axes of the grid given by the eigenvectors and the size of the rectangular sampling grid taken proportional to the square roots of the respective eigenvalues (λ times the respective standard deviations along each eigenaxis). A sampling grid for V_n^X is obtained in the same way. The size of the sampling grids is chosen according to the desired approximation error. We sum over the sampling grids $\mathcal{R}_U$ and $\mathcal{R}_V$ and look for pairs $\left(U_n, V_n^X\right)$ satisfying the collision condition

$$V_n^X \in \mathcal{D}(U_n) \cap \mathcal{R}_V$$

and compute their respective probabilities as

$$\sum_{U \in \mathcal{R}_U} f_{U_n}(U)\Delta a \sum_{V^X \in \mathcal{D}(U_n) \cap \mathcal{R}_V} f_{V_n^X}(V^X)\Delta b \tag{47}$$

where Δa and Δb are respectively the volumes of the cells of $\mathcal{R}_U$ and $\mathcal{R}_V$.

3.3 Probability of Collision in the General Case

The derivation of the probability of collision in the general case of errors in both the rotational and translational components of the kinematic state is an extension of the cases derived previously. Now the angle Θ_n is a random vector and we have

$$\Theta_{n+1} = A^\Theta \Theta_n + V_n^\Theta.$$

From equation (22) D_n^r is a nonlinear function of independent random vectors, and we may rewrite it as

$$D_n^R = NL_1(U_n, \Theta_n).$$

Similarly from equation (42) we see that D_{n+1}^r is a nonlinear function of independent random variables and we may rewrite it as

$$\begin{aligned} D_{n+1}^r &= NL_2(U_n, \Theta_n, V_n^\Theta) + NL_3(\Theta_n, V_n^U, V_n^\Theta) \\ &= NL_4(S_n, V_n) \end{aligned} \tag{48}$$

so that we may write as in (46) that

$$PC(D_n, D_{n+1}) = \int_{\forall S} f_{S_n}(U) \left(\int_{\forall V : NL_4(S,V) \in Sh(NL_1(S))} f_{V_n}(V) dV \right) dS.$$

The computation of the probability of collision follows from this equation as presented in the previous subsection.

The most costly operation in this computation is to check if a point belongs to the shadow of another point. The complexity of the overall numerical method is proportional to the sampling size chosen for each dimension of the random vectors, and exponential with the dimension of the error component, that is the number of components of the kinematic state vector that are considered to be corrupted by errors.

The structure of this calculation is a convolution with a complicated domain of integration. The use of rectangular or hypercubic sampling grids makes it suitable to implement on a SIMD parallel machine. This could allow considerable speedup for real time implementation.

3.4 Collision Probability Profiles

We give in this section examples of probability of collision profiles for different situations and error values in the case of position error. The shape of the target used is described in Figure 8.

Figure 8: Target shape placed at estimated position and relative robot position.

The setup is as follows (all vectors are given in the global coordinate system). The target center of inertia estimated position is

$$X_n = \begin{bmatrix} 0. \\ 0. \end{bmatrix}$$

The target position estimation error covariance is

$$\Sigma_n^X = \begin{bmatrix} 0.25 & 0.00 \\ 0.00 & 0.25 \end{bmatrix}. \tag{49}$$

The target speed and acceleration are

$$\ddot{X}_n \quad = \quad \dot{X}_n \quad = \quad \begin{bmatrix} 0. \\ 0. \end{bmatrix}$$

The target translation motion error covariance is

$$\Upsilon_n^X = \begin{bmatrix} 0.25 & 0.00 \\ 0.00 & 0.25 \end{bmatrix}. \tag{50}$$

The robot position is

$$X_n = \begin{bmatrix} 0. \\ -1. \end{bmatrix}$$

The current and next target orientation angle are

$$\Theta_n \quad = \quad \Theta_{n+1} \quad = \quad 0_{3\times1}$$

For the conditions stated above we give the resulting probability of collision associated with a set of candidate destinations. This set of candidate destinations is given on a 25-sample grid of destinations centered on the position $D_{n+1} = [-0.5, 0.0]$ and of width equal to 1.00. The plot of the probability of collisions is given in Figures 9 and 10.

Figure 9: Probability of collision for first example, side view along y axis.

Figure 10: Probability of collision for first example, front view along x axis.

A second example uses the same conditions as above but the target has a rotation of $\frac{\pi}{8}$ during the time interval. The corresponding results are given in Figures 11 and 12.

4 Optimal Destination

We have provided a probabilistic measure of collision safety. From this we want to produce an optimal destination for the robot. There are many ways in which to define an optimal destination depending on the problem definition, the type of cost function, and the size and shape of the search space. There are also various methods for solving these optimization problems. It is not our purpose here to explore those different methods. In this section, we propose two different

Figure 11: Probability of collision for second example, front view along vertical axis.

Figure 12: Probability of collision for second example, front view along horizontal axis.

definitions of optimality. In the first definition, the probability of collision is used to define a constraint; the optimal destination is defined as the one that minimizes the expected distance to the intermediate goal under a constrained probability of collision. In the second definition, the optimal destination is the one that minimizes a cost function including the probability of collision along with the probability of getting close enough to the intermediate goal. We first consider the definition of the search space.

4.1 Set of Candidate Destinations

By assumption the only obstacle in the space is the target itself. We suppose we are close enough to the target to be able to reach it in the next few time steps. We are not interested in situations where we are so far from the target that we have a remote chance of reaching it and of colliding with it.[5] The robot trajectory control is based on the following strategy: move toward the goal while respecting dynamic constraints and minimizing the risk of colliding.

[5] "Far" means that the probability of colliding at the next time step is less than a predefined value for all reachable destinations.

More precisely, let us introduce the following definitions: Let us call $G_{n+k|n}$ the prediction at time t_n of the goal point position k steps in the future. Let R_{n+1} be the set of reachable positions at the next time step, and R_{n+k} the set of reachable positions at time t_{n+k}. We define the intermediate goal Z_{n+1} as follows:

- $Z_{n+1} = G_{n+1|n}$ if $G_{n+1|n} \in R_{n+1}$, where $G_{n+1|n}$ is the predicted position of the goal point of the target at the next time step (see Figure 13).
- Otherwise (i.e. if $G_{n+1|n} \notin R_{n+1}$) consider the minimum time step k such that $G_{n+k|n} \in R_{n+k}$. Since we are "close" enough to the target, we suppose that such a time exists. Then the intermediate goal Z_{n+1} is the position of the robot at time t_{n+1} in R_{n+1} if the robot were to reach the goal at time $n + k$. We may have many choices for Z_{n+1}; in this case we take the position corresponding to the simplest trajectory to $G_{n+k|n}$, i.e. possibly a rectilinear trajectory (see Figure 14).

Figure 13: Intermediate goal.

From the intermediate goal Z_{n+1} we get a set of candidate destinations C_{n+1} by taking the intersection of a sampling grid centered on Z_{n+1} and the reachable set R_{n+1}, as shown in Figure 15 (a). According to the situation other types of grids can be used. We can also consider looking for the optimal destination in a one dimensional space. We can restrict C_{n+1} to lie along:

- The segment going from the present position D_n to the intermediate goal position G_{n+1}. This corresponds to varying only the speed while keeping a constant direction (see Figure 15 (b)).
- The arc of a circle centered on D_n and with radius $|Z_{n+1} - D_n|$. This corresponds to looking for the safest steering direction and keeping a constant speed (see Figure 15 (c)).

We now turn to the definition of optimality.

4.2 Minimum Distance with Tolerable Risk Level

The first optimality criterion we consider is the minimum distance to the goal. We choose as next position D^*_{n+1} the position closest to the intermediate goal that has an acceptable risk level, i.e.,

Figure 14: Intermediate goal for unreachable goal.

the optimal destination D_{n+1} is defined as

$$D^*_{n+1} = \mathrm{Argmin}_{\forall D_{n+1} \in \Gamma} \left(d(D_{n+1}, Z_{n+1}) \right), \tag{51}$$

where Γ is the set of destinations with an acceptable risk level:

$$\Gamma = \{ D_{n+1} \in C_{n+1} \mid PC(D_n, D_{n+1}) \leq P_{\text{threshold}} \}. \tag{52}$$

The search for this position is done as follows: We first look at the four candidates one unit away (Figure 16 (a)) from the intermediate goal in C_{n+1} and calculate their collision probabilities. If none of them satisfies the bound then we look at the four candidate positions $\sqrt{2}$ unit away (Figure 16 (b)) then two units away (Figure 16 (c)), $\ldots$, until a position with tolerable risk level is found.

The cost function above considers $d(D_{n+1}, Z_{n+1})$, the conditional expected distance from the destination point to the goal. This expected distance is not meaningful enough in representing the "likelihood" of getting to the expected goal point. To remedy this problem, an alternative optimality definition is presented in the following section.

4.3 Composite Cost Function

The criteria for the optimal robot destination include collision safety and closeness to the intermediate goal. We have shown how to give a probabilistic measure for the safety. We derive here a probabilistic measure for the closeness criterion, and include both measures in a composite cost function.

In general our robot has a certain kind of operation to accomplish on the target. For this operation to be successful, the robot must assume a particular position and orientation relative to the target object. But the position of the target being random, we can measure in a probabilistic sense how close we are to the subgoal position and incorporate it with the probability of collision in a broader cost function.

Figure 15: Sets of candidate destinations.

Figure 16: Search for the closest solution with a tolerable probability of collision.

Reaching an exact position with respect to the target has probability zero. Moreover, in practical situations, it also makes more sense to define a tolerance set of positions for which the robotic operation is expected to be carried out. So we can define a subset of *operational positions* ΔO in the target frame of reference. The destination D_{n+1} is operational if $D^r_{n+1} \in \Delta O$, and we note that

$$PO(D_{n+1}) = P(D^r_{n+1} \in \Delta\ O)\cdot$$

The optimal solution of the problem is the destination D_{n+1} that minimizes a composite cost function

$$D^*_{n+1} = \mathrm{Argmin}_{\forall D_{n+1} \in C_{n+1}} \left(\gamma PC(D_n, D_{n+1}) + (1-\gamma)(1 - PO(D_{n+1}))\right), \tag{53}$$

with given $\gamma \in [0,1]$. Deriving $PO(D_{n+1})$ is simple. Using the results of the previous subsection and equation (24) we have

$$PO(D_{n+1}) = \int_{\forall U} f_{U_n}(U) \left(\int_{\forall V^U\, L_2(U)+L_3(V^U) \in \Delta O} f_{V_n}(V) dV^U \right) dU \cdot \tag{54}$$

This integration is analogous to the one done in equation (32). The variables are the same but the domain of integration is simpler since it is constant. This suggests that the computation of $PO(.)$ can be integrated inside the loop for calculating $PC(.)$ so that we can get $PO(.)$ at no extra computational cost. The weight γ is determined empirically according to the situation and weight safety vs. proximity to the goal.

4.4 Extension to Longer Horizon Planning

We can extend the previous method to plan a two-step trajectory or an m-step trajectory. Given that we are interested in a single position vis-a-vis the target, at each time step only one intermediate goal position is defined; optimization of the probability of collision is performed for candidate destinations sampled around the intermediate goal position. We show here how to calculate the overall probability of collision associated with a two-step trajectory. We show how the one-step case extends to the two-step case. We sketch this derivation without going into the details. Similarly to equation (40) we write

$$D^r_{n+2} = R_{-\alpha_{n+2}}[D_{n+2} - (E_1 + 2E_2 + 2E_3)U_n - (E_1 + E_2 + \frac{E_3}{2})V_n - V^X_{n+1}]$$

and given that

$$\Theta_{n+2} = (A^\Theta)^2\Theta n + A^\Theta V^\Theta_n + V^\Theta_{n+1}$$

from the two previous equations we can express D^r_{n+2} as

$$\begin{aligned} D^r_{n+2} &= F_1(V^\Theta_n, V^\Theta_{n+1}, S_n) + F_2(\Theta_n, V_n, V^\Theta_{n+1}) + F_3(\Theta_n, V_n, V_{n+1}) \\ &= F_4(S_n, V_n, V_{n+1}) \end{aligned} . \tag{55}$$

The probability that there will not be any collision on the candidate trajectory (D_n, D_{n+1}, D_{n+2}) is obtained by considering all triples $(D^r_n, D^r_{n+1}, D^r_{n+2})$ such that

$$(D^r_{n+1} \notin Sh(D^r_n)) \wedge (D^r_{n+2} \notin Sh(D^r_{n+1}))$$

which is equivalent to consider all triples of vectors (S_n, V_n, V_{n+1}) such that

$$(NL_4(S_n, V_n) \notin Sh(NL_1(S_n))) \wedge (F_4(S_n, V_n, V_{n+1}) \notin Sh(NL_4(S_n, V_n)))$$

Thus to find the probability of collision we calculate

$$\begin{aligned} &PC(D_n, D_{n+1}, D_{n+2}) \\ &= 1 - \int_{\forall S} f_{S_n}(S) \\ &\left(\int_{\forall V: NL_4(S,V) \notin Sh(NL_1(S))} f_{V_n}(V) \left(\int_{\forall V_1: F_4(S,V,V_1) \notin Sh(NL_4(S,V))} f_{V_{n+1}}(V_1) dV_1 dV\right)\right) dS \end{aligned} . \tag{56}$$

The further ahead we look, the less meaningful the information on the predicted state, because of the accumulation of the model errors. The prediction covariance matrix takes into account the accumulated errors. We can decide that it is worth planning m time steps in the future if the norm $||\Sigma_{X_{n|n}}||$ is less than a predefined value. As we shall see in Section 6 (68),

$$\Sigma_{n+m|n} \quad = \quad A^m \Sigma_{n|n} A^{m\,T} + \sum_{i=1}^{m} A^{i-1} \Upsilon_{n+m-i} A^{i-1^T}. \tag{57}$$

Given Υ the sum term can be calculated off-line for each $n = 2, \ldots,$ and thus only the present covariance needs to be calculated.

Further formal extensions to the method proposed in this chapter can be derived to include cases where the robot is not a point but a concave polygon and cases of many obstacles. The methods extend to 3D. Complexity related problems in the three dimensional case include the determination of the shadow for a particular target shape, basically the computation of a 3D visibility graph.

5 Conclusion

The probabilistic formalism provides a framework in which the likelihood of accomplishing the navigational and operational goals of a robotic system can be assessed. There are many situations in robotic satellite maintenance, autonomous vehicle guidance, and autonomous industrial robot guidance in which risk assessment is essential. While the risk is understood in this report as collision risk, we can extend this notion to any other spatial hazard that the robotic system may encounter.

This probabilistic framework could be incorporated into a low level trajectory controller that could operate with a higher level planner. The high level planner may provide navigational subgoals such as moving toward a position in space, reaching this position, locking in translation or rotation with a target object, going into orbit around a moving object, reaching a desirable relative position, avoiding certain other positions relative to the object, ... along with the acceptable degree of risk.

This paper has presented one approach to probabilistic sensor-based navigation in dynamic environments. The originality of this approach lies in its considering as optimality criteria the probability of colliding with the obstacle and the probability of reaching an operational position. Estimates of the obstacle motion and measures of confidence in those estimates induce a probability of collision associated with each robot displacement, which can be used for optimizing the trajectory of the robot. A second approach using measures of confidence in the motion estimates to produce entire regions with tolerable associated risk levels is presented in [6]. These two approaches are complementary and can be used simultaneously. These approaches can be extended to planning in situations in which a robot works in an hostile environment comprised of several moving obstacles or possibly hostile agents. As a possible strategy, the robot may approach one of the agents by optimizing its trajectory with respect to the probability of intercepting this agent while maintaining a tolerable level of safety with respect to this agent, using the first approach. Meanwhile the robot stays clear of the other agents, using the second approach. Practically the second approach could provide spatial constraints as input to the optimization performed in the first approach. The structure and complexity of the calculations makes the approach presented here suitable for SIMD parallel implementation.

References

[1] M. Abramovitz and I.A. Stegun. *Handbook of Mathematical Functions.* U.S.N.B.S. Applied Math. Series, U.S.G.P.O., 1964.

[2] N. Ayache and O.D. Faugeras. Building, registrating, and fusing noisy visual maps. *International Journal of Robotics Research*, 17(6):45–65, 1988.

[3] A. Basu. A framework for motion planning in the presence of moving obstacles. Technical Report CS-TR-2378, University of Maryland, Center for Automation Research, 1989.

[4] G.J. Bierman. A comparison of discrete linear filtering algorithms. *IEEE Transactions on Aerospace and Electronic Systems*, 9:28–37, 1973.

[5] J. Borenstein and Y. Koren. Real time obstacle avoidance for fast mobile robots in cluttered environments. In *Proc. IEEE Conf. Robotics and Automation*, pages 572–577, 1990.

[6] P. Burlina, D. DeMenthon, and L.S. Davis. Navigation with uncertainty II: Avoiding high risk regions. Technical Report in preparation, University of Maryland, Center for Automation Research, 1991.

[7] V.H.L. Cheng and B. Sridhar. Integration for active and passive sensors for obstacle avoidance. *Control Systems*, 10(4):43–50, 1990.

[8] J.L. Crowley and F. Ramparany. Mathematical tools for representing uncertainty in perception. In *Proc. Spatial Reasoning and Multisensor Fusion Workshop*, 1987.

[9] H.F. Durrant-Whyte. Consistent integration and propagation of disparate sensor observations. *International Journal of Robotics Research*, 6(3):3–24, 1987.

[10] H.F. Durrant-Whyte. Uncertain geometry in robotics. In *Proc. IEEE Int. Conf. Robotics and Automation*, 1987.

[11] A. Elfes. Occupancy grids: A stochastic spatial representation for active robot perception. In *Proc. Sixth Conf. Uncertainty in AI*, 1990.

[12] B. Faverjon and P. Tournassoud. A local based approach for path planning of manipulators with a high number of degrees of freedom. In *Proc. IEEE Int. Conf. Robotics and Automation*, 1987.

[13] J.T. Feddema, C.S. Lee, and O.R. Mitchell. Automatic selection of image features for visual servoing of a robot manipulator. In *Proc. IEEE Conf. Robotics and Automation*, pages 832–837, 1989.

[14] K. Fujimura and H. Samet. Accessibility: A new approach for planning a path among moving obstacles. In *Proc. Conf. Computer Vision and Pattern Recognition*, pages 803–807, 1988.

[15] R.M. Gray and L.D. Davisson. *Random Processes, A Mathematical Approach for Engineers.* Prentice-Hall, Englewood Cliffs, NJ, 1986.

[16] N.C. Griswold and J. Eem. Control for mobile robots in the presence of moving objects. *IEEE Transactions on Robotics and Automation*, 6:263–68, 1990.

[17] N. Kehtarnavaz and N. Griswold. Establishing collision zones for obstacles moving with uncertainty. *Computer Vision Graphics and Image Processing*, 49:95–103, 1990.

[18] O. Khatib. Real time obstacle avoidance for manipulators and mobile robots. *International Journal of Robotics Research*, 1(5):90–99, 1986.

[19] H.P. Moravec. A Bayesian method for certainty grids. Technical Report, Robotics Institute, Carnegie-Mellon University, 1988.

[20] H.P. Moravec. Sensor fusion in certainty grids for mobile robots. *AI Magazine*, 9(2):61–74, 1988.

[21] J. O'Rourke, C.B. Chen, T. Olson, and D. Naddor. A new linear algorithm for intersecting convex polygons. *Computer Graphics and Image Processing*, 19:384–391, 1982.

[22] J.T. Schwartz and C.K. Yap. *Algorithmic and Geometric Aspects of Robotics.* Erlbaum, Hillsdale, NJ, 1987.

[23] O. Silven. Estimating the pose and motion of a known object for real time motion tracking. Technical Report CS-TR-2461, University of Maryland, Center for Automation Research, 1990.

[24] C.L. Thornton and G.J. Bierman. UDUt covariance factorization for Kalman filtering. *Control and Dynamic Systems*, 16:76–247, 1980.

[25] C.K. Yap. Algorithmic motion planning. In J.T. Schwartz and C.K. Yap, editors, *Advances in Robotics vol. 1: Algorithmic and Geometric Aspects.* Erlbaum, Hillsdale, NJ, 1986.

6 Motion Probabilistic Models, Prediction and Estimation

Our approach makes use of probability distributions for the positions of the target. The Kalman filtering method provides this information. In our model the motion parameters of the target object are Gaussian. The Kalman filter yields both the mean (equal to the optimal estimate) and the covariance matrix of the target motion parameters. The assumptions made about the probabilistic model and the details of this method are presented in this section.

6.1 Motion and Observation Models

The movement of the target or obstacle being tracked is represented by a state vector whose components are the position and orientation parameters and their first and second derivatives. The evolution of the kinematic state is represented as

$$S_{n+1} = AS_n + V_n \tag{58}$$

where we have defined

- $S_n = [U_n, \Theta_n]^T$, the target motion state vector,
- $U_n = \left[X_n, \dot{X}_n, \ddot{X}_n\right]^T$, where $X_n, \dot{X}_n, \ddot{X}_n$ are respectively the position, velocity and acceleration vectors of the target center of inertia,
- $X_n = [x_n, y_n]^T$, the position of the target center of inertia,
- $\Theta_n = [\alpha_n, \dot{\alpha}_n, \ddot{\alpha}_n]^T$, the target orientation angle and its first and second derivatives,
- $V_n = \left[V_n^U, V_n^\Theta\right]^T$, the error in the target motion model, having zero mean Gaussian distribution and covariance equal to Υ_n,
- A = the model transition matrix.

The kinematic state is related to the sensor observation by

$$O_n = B(S_N) + W_n \tag{59}$$

where we have defined

- $O_n = (x_{M_1}, x_{M_2}, \ldots, x_{M_N})$, the observation vector at time step n, composed of the n-vertex coordinates in the camera image,
- $O^n = (O_1, O_2, \ldots, O_n)$, all the observations made up to time n,
- W_n = a zero-mean Gaussian distributed error on the observations with covariance matrix Ω_k,

- $B(.)$ = the observation function.

Uncertainty arises from errors occurring in the robot motion control, from robot calibration errors, from the observation uncertainty, and from the incomplete model of the target motion. These uncertainties are incorporated in the motion model error V_n and observation error W_n, whose covariances are derived according to the problem at hand.

6.2 Estimation and Prediction

Based on this motion evolution and observation model, the Kalman filtering method is a means of iteratively calculating the optimal estimate of the motion state. It produces estimates and predicts the motion state of the target, based on the observations made so far. We denote by $S_{n+k|n}$ the linear minimum variance estimate of the state at time $n+k$, based on the observed vectors O_0, O_1, O_n up to present time n,

$$\begin{aligned} S_{n+k|n} &= \hat{E}[S_{n+k}|O_1, \ldots, O_n] \\ &= \hat{E}[S_{n+k}|O^n]. \end{aligned} \tag{60}$$

The covariance matrix $\Sigma_{n+k|k}$ of the estimated vector gives a measure of reliability of the estimates, and is equivalent to a covariance on the estimation error

$$\Sigma_{p|q} = \hat{E}[S_{p|q} S_{p|q}^T]. \tag{61}$$

For planning purposes, we are interested in obtaining:

- $S_{n|n}$ the filtered estimate of the target motion state at time n based on all observations O^n made up to time n,
- $S_{n+k|n}$, the prediction estimate of the motion state k steps in the future based on all observations O^n made up to time n.

The Kalman filtering loop is given by the following two stages, estimation and filtering (see [15]). The estimation stage is

$$S_{n|n-1} = A\, S_{n-1|n-1} \tag{62}$$

$$\Sigma_{n|n-1} = A\, \Sigma_{n-1|n-1}\, A^T + \Upsilon_{n-1} \tag{63}$$

where

$$\Upsilon_n = E[V_n V_n^T]$$

is the covariance of the model error.

The estimation stage makes use only of the evolution model (equation (58)). From the filtered state $S_{n-1|n-1}$ based on the $n-1$ first observations, equation (62) gives the estimation of the state at time n. Equation (63) gives the estimation covariance from the filtered state covariance and the model noise covariance. The filtering stage is

$$K_n = \Sigma_{n|n-1}\, J\, (\Omega_n + J\, \Sigma_{n|n-1}.J^T)^{-1} \tag{64}$$

$$S_{n|n} = S_{n|n-1} + K_n\, (O_n - B(S_{n|n-1})) \tag{65}$$

$$\Sigma_{n|n} = (I - K_n\, J)\, \Sigma_{n|n-1} \tag{66}$$

with

$$\Omega_n = E[W_n W_n^T]$$

the covariance on the observation error.

J is the Jacobian of the observation function. In the case where the observations are feature point coordinates in the image, its derivation is given in [13]. K_n is the innovation weight (the innovation is the difference between the value of the state estimated from the motion evolution model and the value of the state coming from the observation). The Kalman filter used for the tracking is an extended Kalman filter, since the observation equation is not linear, i.e. we have $O_n = B(S_n) + W_n$ instead of $O_n = B\, S_n + W_n$ for some matrix B. In this case we must use the Jacobian of the vertex junction coordinates with respect to the state vector S_n. The filtering process is based on the observation equation. It has an intuitive interpretation. The filtered value consists of a sum of the estimated value and the innovation, with a weight factor depending on the covariance of the observation error. The greater the observation error, the smaller the weighting factor, the more we rely on the value estimated according to the motion model. On the other hand, when the observation error is small, the observation is given a greater weight. At each step we can perform a prediction m steps in the future:

$$S_{n+m|n} = A^m \, S_{n|n}. \tag{67}$$

and the covariance matrix for this prediction is

$$\begin{aligned} \Sigma_{n+m|n} &= A^m \, \Sigma_{n|n} \, A^{m\,T} + \Upsilon_{n+m-1} + A \, \Upsilon_{n+m-2} \, A^T + .. + A^{m-1} \, \Upsilon_n \, A^{m-1\,T} \\ &= A^m \Sigma_{n|n} A^{m\,T} + \sum_{i=1}^{m} A^{i-1} \Upsilon_{n+m-i} A^{i-1^T}. \end{aligned} \tag{68}$$

This is only a generalization of the estimation step in equation (62). For our application we are more interested in the case where $m = k + 1$, i.e. the one-step prediction. The Kalman estimate is optimal in the following sense:

- It is a Linear Mean Square error estimate, no matter what the distribution of the errors is.
- Moreover, if the errors are Gaussian, as assumed here, the Kalman filter also gives the Mean Square Error estimate.[6] This estimate $S_{p|q}$ is equal to the conditional mean of the state S_p conditioned on the observations O^q.

In the case of an extended Kalman filter, the future state conditioned on the observations is not necessarily a Linear Minimum Variance Estimate but may in some cases be a Minimum Variance Estimate; thus it is not optimal, but it has been shown that it still behaves well [23] and behaves the best among many linear estimation methods ([4]).

6.3 The Gaussian Assumption

For a classic Kalman filter with Gaussian errors, the conditional distribution of the state given the observations made so far is indeed Gaussian with first and second moments respectively given by the estimate of the state and the error covariance in the estimated state. In the more general case of an extended Kalman filter with a nonlinear recovery function, the situation is slightly different: the conditional distribution of the state is not exactly Gaussian, but the distribution of the state

[6] Also called the Minimum Variance Estimate.

is. Indeed this follows simply from the motion state evolution which is linear and corrupted by Gaussian noise. Therefore it is a sensible approximation to consider the conditional distribution to be Gaussian. It has been shown that assuming Gaussian errors in the observation equation is a good approximation and works well indeed [23]. Another distribution might have been assumed at a much higher calculation cost. In any case, concerning the justification of the Gaussian assumption, it is an accepted fact that calculations and combinations of geometric errors are best handled with Gaussian distributions ([2, 8, 9, 10]): since two moments are sufficient to characterize such a distribution and any affine combination of Gaussian errors remains Gaussian, it is suitable for the manipulation of errors.