Planning with Uncertainty in Position: an Optimal and Efficient Planner

Juan Pablo Gonzalez and Anthony (Tony) Stentz
The Robotics Institute
Carnegie Mellon University
Pittsburgh, USA
{jgonzale, axs}@ri.cmu.edu

Abstract— **We introduce a resolution-optimal path planner that considers uncertainty while optimizing any monotonic objective function such as mobility cost, risk, or energy expended. The resulting path minimizes the expected cost of the objective function, while ensuring that the uncertainty in the position of the robot does not compromise the safety of the robot or the reachability of the goal. Although the problem domain is stochastic in nature, our algorithm takes advantage of deterministic path-planning techniques to achieve significant performance improvements.**

Index Terms - path planning, uncertainty, optimal planner, error propagation, mobile robot.

I. INTRODUCTION

Current approaches to path planning frequently ignore the inherent uncertainty in the position of mobile robots. If the position uncertainty is small compared to the size of the robot these approaches provide adequate solutions. However, when the uncertainty is large, it can have a significant impact on the robot's ability to follow a planned path and should not be ignored.

The widespread use of GPS (Global Positioning System) alleviates the position estimation problem for outdoor mobile robots. However, there are many scenarios where GPS coverage is limited or unavailable: under tree canopies, in canyons, indoors, underground, and in many urban environments.

In this paper we propose a resolution-optimal path planner that models position uncertainty arising from drift in onboard dead-reckoning, while optimizing any monotonic objective function such as mobility cost, risk, or energy expended. The environment is assumed to be known, and is represented as a grid, in which the cost of each cell corresponds to the cost of traveling from the center of the cell to its nearest edge. Non-traversable areas are assigned infinite cost and considered obstacles. The resulting path minimizes the expected value of the objective function along the path, while ensuring that the uncertainty in the position of the robot does not compromise its safety or the reachability of the goal.

We assume a robot with a reasonably good dead-reckoning system (uncertainty rate less than 10% of distance traveled), which may or may not have GPS on board. We assume that the robot is not be able to detect the obstacles in the environment that are represented in the map. Instead, the robot relies on the planner to keep it safe from these obstacles.

The planner uses a single-parameter Gaussian distribution to model position uncertainty in order to minimize the number of dimensions required. We show that this model is a conservative estimate of the true error propagation model and, depending on the type of error that is dominant in the system, can provide an accurate approximation of the true model.

By using this simplified uncertainty model we are able to plan in a 3-D approximation of the belief space of the robot using efficient deterministic path planning techniques [1]. Thanks to these techniques, and because of the characteristics of the resulting search space, our planner has a computational complexity close to that of a deterministic 2-D planner. The planning time is less than one second for worlds of up to 150x150 cells, and less than ten seconds for worlds of up to 250x250 cells, for uncertainty rates between 1% and 10% of distance traveled[1].

The planner also utilizes GPS-based localization when it is available; in regions with GPS coverage, the uncertainty stops propagating and is reduced to a fixed value.

This paper is organized as follows: Section II contains a review of the relevant literature in the area of path planning with uncertainty. Section III explains the motion model and the uncertainty propagation problem. Section IV explains our approach to the problem. Section V shows some results of the planner applied to synthetic and real geographic data and analyzes the performance of the algorithm. Section VI contains our conclusions and explores future directions for this research.

II. RELATED WORK

There is an abundance of work addressing path planning with uncertainty in the research literature. In the field of classical path planning, Latombe [2][3] has an extensive review on the state of the art as of 1991. Since then, important contributions by Lazanas and Latombe [4], Bouilly [5][6], Haït et al. [7], Fraichard [8], and others have not only expanded the theoretical approaches to planning with uncertainty, but also addressed some of its practical

[1] Processing times for a Pentium M, 1.4GHz.

limitations. There is, however, little work aimed at creating paths that are optimal with respect to more general objective functions. Although the planner proposed by Bouilly [5] calculates an optimal path with respect to uncertainty or path length, the approach is not applicable to finding optimal paths with respect to other important criteria such as mobility cost, risk, or energy expended.

In the field of partially observable Markov decision processes (POMDPs), the problem of planning with uncertainty has been frequently addressed. However, most algorithms become computationally intractable when dealing with worlds with a large number of states. To the best of our knowledge, only Roy and Thrun [9] have solved the problem of finding optimal paths for large, continuous-cost worlds in the presence of uncertainty. The planner they propose includes some of the elements of the planner proposed here but is based on an approximate solution to a POMDP. This approach requires pre-processing of all the states in the search space, which later allows for very fast planning. However, the total planning time (including the pre-processing stage) can take from several minutes to a few hours [10].

The alternative we are presenting is a semi-deterministic approach based on A*, which incorporates some of the advantages of both approaches (classic path planning and POMDPs), and which has much lower average time complexity than the work of Roy and Thrun.

III. Motion Model and Uncertainty Propagation

The first-order motion model for a point-sized robot moving in two dimensions is:

$$\begin{aligned} \dot{x}(t) &= v(t)\cos\theta(t) \\ \dot{y}(t) &= v(t)\sin\theta(t) \\ \dot{\theta}(t) &= \omega(t) \end{aligned} \tag{1}$$

where the state of the robot is represented by x(t), y(t) and θ(t) (x-position, y-position and heading respectively), and the inputs to the model are represented by v(t) and ω(t) (longitudinal speed and rate of change for the heading respectively). Equation (1) can also be expressed as:

$$\dot{\mathbf{q}}(t) = f(\mathbf{q}(t), \mathbf{u}(t)) \tag{2}$$

where $\mathbf{q}(t) = (x(t), y(t), \theta(t))$ and $\mathbf{u}(t) = (v(t), \omega(t))$.

A typical sensor configuration for a mobile robot is to have an odometry sensor and an onboard gyro. We can model the errors in the odometry and the gyro as errors in the inputs where $w_v(t)$ is the error in $v(t)$ (error due to the longitudinal speed control), and $w_\omega(t)$ is the error in $\omega(t)$ (error due to the gyro random walk).

Incorporating these error terms into (1) yields:

$$\begin{aligned} \dot{x}(t) &= (v(t) + w_v(t))\cos\theta(t) \\ \dot{y}(t) &= (v(t) + w_v(t))\sin\theta(t) \\ \dot{\theta}(t) &= \omega(t) + w_\omega(t) \end{aligned} \tag{3}$$

or, in discrete-time:

$$\begin{aligned} x(k+1) &= x(k) + (v(k) + w_v(k))\cos\theta(k)\Delta t \\ y(k+1) &= y(k) + (v(k) + w_v(k))\sin\theta(k)\Delta t \\ \theta(k+1) &= \theta(k) + (\omega(k) + w_\omega(k))\Delta t \end{aligned} \tag{4}$$

Using the extended Kalman filter (EKF) analysis for this system, which assumes that the random errors are zero-mean Gaussian distributions, we can model the error propagation as follows:

$$\mathbf{P}(k+1) = \mathbf{F}(k)\cdot\mathbf{P}(k)\cdot\mathbf{F}(k)^T + \mathbf{L}(k)\cdot\mathbf{Q}(k)\cdot\mathbf{L}(k)^T \tag{5}$$

where

$$\mathbf{P}(k) = E(\hat{\mathbf{q}}(k)\hat{\mathbf{q}}(k)^T) \qquad \mathbf{Q}(k) = \frac{1}{\Delta t}\begin{pmatrix} \sigma_v^{\,2} & 0 \\ 0 & \sigma_\omega^{\,2} \end{pmatrix} \tag{6}$$

$$\mathbf{F}_{ij} = \frac{\partial f(q_i(k), u_j(k))}{\partial q_i}$$

$$\mathbf{F} = \begin{pmatrix} 1 & 0 & -v(k\Delta t)\sin(\theta(k))\cdot\Delta t \\ 0 & 1 & v(k\Delta t)\cos(\theta(k))\cdot\Delta t \\ 0 & 0 & 1 \end{pmatrix} \tag{7}$$

$$\mathbf{L}_{ij} = \frac{\partial f(q_i(k), u_j(k))}{\partial u_j}$$

$$\mathbf{L} = \begin{pmatrix} \cos(\theta(k))\cdot\Delta t & 0 \\ \sin(\theta(k))\cdot\Delta t & 0 \\ 0 & \Delta t \end{pmatrix} \tag{8}$$

Although there are no general closed-form solutions for these equations, Kelly [12] calculated closed-form solutions for some trajectories, and showed that a straight line trajectory maximizes each one of the error terms. We can use the results for this trajectory as an upper bound on the error for any trajectory. For a straight trajectory along the x-axis keeping all inputs constant, the error terms behave as follows. The error due to the longitudinal speed control w_v is reflected in the x direction, and is given by $\sigma_x^2 = \sigma_v^2 \cdot vt$, or $\sigma_x = \sigma_v\sqrt{vt}$. The error due to the gyro random walk w_ω is reflected in the y direction, and is given by $\sigma_y^2 = \sigma_\omega^2 \cdot v^2 t^3$, or $\sigma_y = \sigma_\omega \cdot v \cdot t^{3/2}$.

Additionally, there are errors in the initial position of the robot. Errors in x and y do not increase unless there is uncertainty in the model or in the controls. Errors in the heading angle, however, propagate linearly with t. For initial angle errors of less than 15 degrees, the small angle approximation ($\sin\theta \approx \theta$) can be used to obtain the expression $\sigma_y = \sigma_{\theta_o} . vt$.

The dominant terms in the error propagation model depend on the navigation system and on the planning horizon for the robot. A typical scenario for a mobile robot with good inertial sensors is to have a planning horizon of up to 3km, at a speed of 5 m/s, with longitudinal control error of 10% of the

commanded speed ($\sigma_v = 0.1v = 0.5m/s$) and gyro drift (random walk) of $20°/h/\sqrt{Hz}$ ($\sigma_\omega = 0.005°/s = 8.72 \cdot 10^{-5} rad/s$). In this scenario, the dominant term is the error in the longitudinal control. However, if there is uncertainty in the initial angle, this term becomes the dominant one for errors as small as 0.5 degrees (See Figure 1).

Figure 2 shows the results of combining all the types of error (with the values mentioned above, and an initial heading angle error of 0.5 degrees) for a straight trajectory. Figure 3 shows the same analysis for a random trajectory. See [13] for a more detailed analysis of the different modes of error propagation.

IV. PLANNING WITH UNCERTAINTY

As shown in the previous section, uncertainty in the position of a robot can quickly accrue even when the robot is equipped with good onboard sensors. It is thus important to incorporate position uncertainty into the planning process in order to generate paths that are robust to this uncertainty.

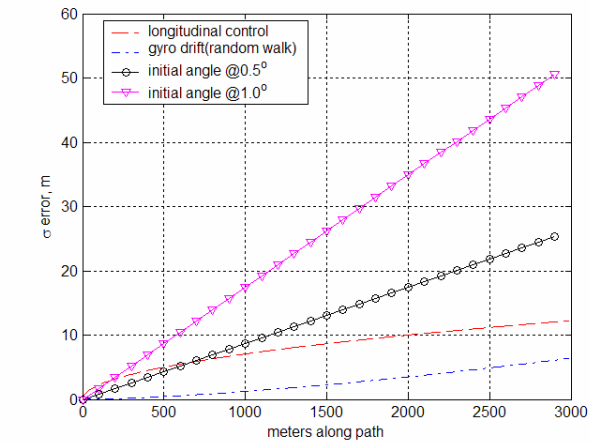

Figure 1. Comparison between different values of initial angle error and longitudinal control error

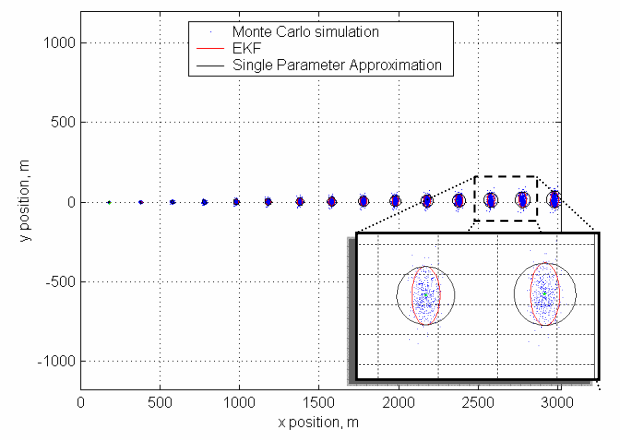

Figure 2. Error propagation for a straight trajectory with all error sources combined and inset showing detail. The blue dots are the Monte Carlo simulation, the red ellipse is the EKF model (2σ), and the black circle is the single parameter approximation (2σ)

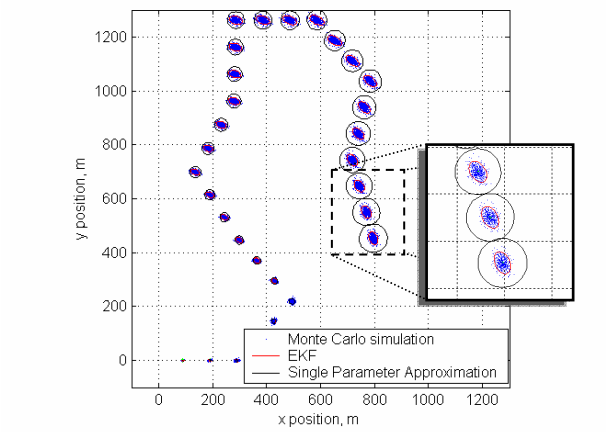

Figure 3. Error propagation for a random trajectory with all error sources combined and inset showing detail. The blue dots are the Monte Carlo simulation, the red ellipse is the EKF model (2σ), and the black circle is the single parameter approximation (2σ).

A. Simplified error propagation model

In order to keep the planning problem tractable and efficient, we use a single-parameter error propagation model. We approximate the probability density function (pdf) of the error with a symmetric Gaussian distribution, centered at the most likely location of the robot at step *k*:

$$\mathbf{q} = (x, y)$$
$$\mathbf{q} : N(\mathbf{q}_c^k, \sigma^k) \tag{9}$$

where $\mathbf{q}_c^k$ is the most likely location of the robot at step *k*, and $\sigma^k = \sigma_x^k = \sigma_y^k$ is the standard deviation of the distribution.

Let us define:

$$\varepsilon^k = 2 \cdot \sigma^k \tag{10}$$

We can then model the boundary of the uncertainty region as a disk centered at $\mathbf{q}_c^k$ with a radius ε^k. As shown in the previous section, this model is a conservative estimate of the true error propagation model and, depending on the type of error that is dominant in the system, can provide an accurate approximation of the true model. Since the dominant term in the error propagation for our planning horizon is linear with distance traveled, we use the following model to propagate the uncertainty:

$$\varepsilon(\mathbf{q}_c^k) = \varepsilon(\mathbf{q}_c^{k-1}) + \alpha_u d(\mathbf{q}_c^{k-1}, \mathbf{q}_c^k) \tag{11}$$

where α_u is the uncertainty accrued per unit of distance traveled, $\mathbf{q}_c^{k-1}$ is the previous position along the path, and $d(\mathbf{q}_c^{k-1}, \mathbf{q}_c^k)$ is the distance between the two adjacent path locations $\mathbf{q}_c^{k-1}$ and $\mathbf{q}_c^k$.

Equivalently, we can define the uncertainty at location $\mathbf{q}_c^k$ as:

$$\varepsilon(\mathbf{q}_c^k) = \varepsilon(\mathbf{q}_c^0) + \alpha_u D(\mathbf{q}_c^k, \mathbf{q}_c^0) \tag{12}$$

where $\mathbf{q}_c^0$ is the most likely initial location of the robot, and $D(\mathbf{q}_c^k, \mathbf{q}_c^0)$ is the total distance traveled along the path from $\mathbf{q}_c^0$ to $\mathbf{q}_c^k$. The uncertainty rate α_u is typically between 0.01 and 0.1 (1% to 10%) of distance traveled.

By modeling the error propagation in this manner, we are assuming that the dominant term is the uncertainty in the initial angle. Even though we are not explicitly modeling θ as a state variable, the effects of uncertainty in this variable are accounted for in the uncertainty propagation model for $\mathbf{q}=(x,y)$.

B. Implementation

In a deterministic planner, the location of the robot on the plane is usually defined as $\mathbf{q}=(x,y)$, where x and y are deterministic variables. Because the position of the robot is now a probability distribution, the new representation for the robot's position is

$$p(\mathbf{q} \mid \mathbf{q}_c{}^k, \varepsilon^k) \tag{13}$$

where $\mathbf{q}$ is a random variable representing the position of the robot, $\mathbf{q}_c{}^k$ is the most likely location of the robot, and ε^k is the uncertainty in the position. As mentioned previously, this probability density function is modeled as a Gaussian distribution centered at $\mathbf{q}_c{}^k$ and with standard deviation $\sigma^k = \varepsilon^k / 2$.

Having a probability distribution over possible robot positions rather than a fixed location means that we are planning over a belief space of the robot [1]. The belief space for this simplified model can be represented by the augmented state variable

$$\mathbf{r} = (\mathbf{q}, \varepsilon) \tag{14}$$

hence defining a 3-D configuration space where x and y are the first two dimensions, and ε is the third dimension.

In order to use a deterministic planner to plan over this space we need to define the transition cost between adjacent cells. In our 3-D configuration space, we are interested in calculating the cost of moving between the configuration $\mathbf{r}^k$ (at path step k) and an adjacent configuration $\mathbf{r}^{k+1}$ (at path step k+1). This is equivalent to calculating the expected cost of going from a most likely workspace location $\mathbf{q}_c{}^k$, with uncertainty ε^k to an adjacent most likely workspace location $\mathbf{q}_c{}^{k+1}$ with uncertainty ε^{k+1} (See [13] for details):

$$C(\mathbf{r}^k, \mathbf{r}^{k+1}) = E\left[C\left((\mathbf{q}_c{}^k, \varepsilon^k), (\mathbf{q}_c{}^{k+1}, \varepsilon^{k+1}) \right) \right] \tag{15}$$

Equivalently,

$$E\left[C\left((\mathbf{q}_c{}^k, \varepsilon^k), (\mathbf{q}_c{}^{k+1}, \varepsilon^{k+1}) \right) \right] = \tag{16}$$

$$\sum_{\forall i} \sum_{\forall j} C_o\left(\mathbf{q}_i{}^k, \mathbf{q}_j{}^{k+1} \right) \cdot p\left(\mathbf{q}_i{}^k, \mathbf{q}_j{}^{k+1} \mid \mathbf{q}_c{}^k, \varepsilon_k, \mathbf{q}_c{}^{k+1}, \varepsilon^{k+1} \right)$$

where $\mathbf{q}_i{}^k$ is each of the i possible states at path step k, $\mathbf{q}_j{}^{k+1}$ is each of the j possible states at path step k+1, and $C_o(\mathbf{q}_i{}^k, \mathbf{q}_j{}^{k+1})$ is the deterministic cost of traveling from $\mathbf{q}_i{}^k$ to $\mathbf{q}_j{}^{k+1}$ (see Figure 4)[2].

Since we are assuming a low uncertainty rate ($\alpha_u < 0.1$), we can make additional simplifications that transform (16) into:

$$E\left[C\left((\mathbf{q}_c{}^k, \varepsilon^k), (\mathbf{q}_c{}^{k+1}, \varepsilon^{k+1}) \right) \right] = \tag{17}$$

$$a \cdot C_E(\mathbf{q}_c^k, \varepsilon_k) + b C_E(\mathbf{q}_c^{k+1}, \varepsilon^k)$$

where a and b are constants determined by the relative position of $\mathbf{q}_c{}^k$ and $\mathbf{q}_c{}^{k+1}$, and

$$C_E(\mathbf{q}_c^k, \varepsilon^k) = \sum_{\forall i} C_o\left(\mathbf{q}_i^k \right) \cdot p\left(\mathbf{q}_i^k \mid \mathbf{q}_c^k, \varepsilon^k \right) \tag{18}$$

is the expected cost of traversing cell $\mathbf{q}_c^k$ if the uncertainty at this location is ε^k. Therefore,

$$C(\mathbf{r}^k, \mathbf{r}^{k+1}) = a \cdot C_E(\mathbf{q}_c^k, \varepsilon_k) + b C_E(\mathbf{q}_c^{k+1}, \varepsilon^k). \tag{19}$$

The planner used for planning with uncertainty is a modified version of A* in 3-D in which the successors of each state are calculated only in a 2-D plane, and state dominance is used to prune unnecessary states.

The planner works as follows: we have a start location $\mathbf{q}^0$ with uncertainty ε^0, an end location $\mathbf{q}^f$ with uncertainty ε^f, and a 2-D cost map C. We form the augmented state variable $\mathbf{r}^0 = (\mathbf{q}^0, \varepsilon^0)$ and place it in the OPEN list. States in the OPEN list are sorted by its expected total cost to the goal:

$$C_{ET}(\mathbf{q}^k, \varepsilon^k) = C_E\left((\mathbf{q}^0, \varepsilon^0), (\mathbf{q}^k, \varepsilon^k) \right) + h_E(\mathbf{q}_k, \mathbf{q}_f) \tag{20}$$

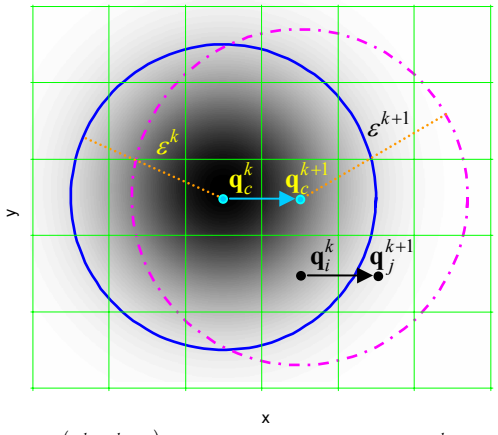

Figure 4. $p\left(\mathbf{q}_i^k \mid \mathbf{q}_c^k, \varepsilon_k\right)$ and the state transitions from $\mathbf{r}^k = (\mathbf{q}_c{}^k, \varepsilon^k)$ to $\mathbf{r}^{k+1} = (\mathbf{q}_c{}^{k+1}, \varepsilon^{k+1})$

[2] As mentioned previously, we assume a discretized grid of states corresponding to a known map.

where $C_E\left((\mathbf{q}^0,\varepsilon^0),(\mathbf{q}^k,\varepsilon^k)\right)$ is the expected cost of the best path from the start location $\mathbf{r}^0=(\mathbf{q}^0,\varepsilon^0)$ to the current state $\mathbf{r}^k=(\mathbf{q}^k,\varepsilon^k)$, and $h_E(\mathbf{q}_k,\mathbf{q}_f)$ is the heuristic of the expected cost from the current state to the goal . The heuristic used is a function of the Euclidean distance between $\mathbf{q}_k$ and $\mathbf{q}_f$.

The state $\mathbf{r}^k$ with lowest expected total cost to the goal is popped from the OPEN list. Next, $\mathbf{r}^k$ is expanded. To determine if a successor $\mathbf{r}_j^{k+1}=(\mathbf{q}_j^{k+1},\varepsilon_j^{k+1})$ should be placed in the OPEN list, we use state dominance as follows: a state $\mathbf{r}_j^{k+1}$ is placed in the OPEN list if

1) no states with (x,y) coordinates $\mathbf{q}_j^{k+1}$ have been expanded, or

2) the expected cost from $\mathbf{r}_0$ to $\mathbf{r}_j^{k+1}$ is lower than the cost to any other state with the same (x,y) coordinates $\mathbf{q}_j^{k+1}$, or

3) the uncertainty ε_j^{k+1} is lower than the uncertainty of any other state with the same (x,y) coordinates $\mathbf{q}_j^{k+1}$

In other words, a state is only expanded if it can provide a path with lower uncertainty or lower cost from the start location. Additionally, since ε is a function of $\mathbf{q}$, the successors of $\mathbf{r}^k$ may be calculated in the 2-D workspace defined by $\mathbf{q}^k$, instead of the full 3-D configuration space defined by $\mathbf{r}^k$ (See Figure 5).

As the states are placed in the OPEN list, their uncertainty is updated using (11), and the expected cost of $\mathbf{r}_j^{k+1}$ is updated according to (17) and (20). However, if any states within 2σ in the uncertainty region of $\mathbf{r}_j^{k+1}$ are labeled as obstacles then the expected total path cost of $\mathbf{r}_j^{k+1}$ is set to infinity, thereby preventing any further expansion of that state.

As in A*, the process described above is repeated until a cell $\mathbf{r}^{k+1}$ with the same (x,y) coordinates as the goal position ($\mathbf{q}^{k+1}=\mathbf{q}^f$), and uncertainty less than the target uncertainty ($\varepsilon^{k+1}\le\varepsilon^f$) is popped off the OPEN list. The path connecting the backpointers from $\mathbf{r}^{k+1}$ to $\mathbf{r}^0$ is the optimal path between $\mathbf{q}^0$ and $\mathbf{q}^f$ with a final uncertainty lower than ε^f. If the OPEN list becomes empty and no such state has been found, then there is not a path between $\mathbf{q}^0$ and $\mathbf{q}^f$ such that the final uncertainty is lower than ε^f .

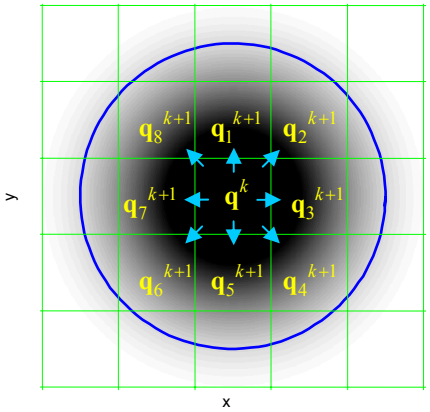

Figure 5. Successors of state $\mathbf{q}^k$

C. *Using localization regions*

The planner can also take advantage of localization regions, such as areas with GPS coverage. If a state propagation yields a distribution that is contained in a GPS region (within 2σ), the uncertainty of this state is set to zero (or some constant value associated with the accuracy of GPS within the region) instead of using (11) to propagate the uncertainty. This approximation preserves the deterministic nature of the belief space.

V. RESULTS

A. *Route planning example*

The following is an example of our planner applied to the problem of finding the best path between two locations on opposite sides of Indiantown Gap, PA. The required task is to find the path with lowest cost from the start to the end location, with a final uncertainty smaller than 500 m, and which avoids all known obstacles in the map (within 2σ). The cost map used is defined as the mobility cost of traveling through each cell in the map, and is proportional to the slope of the terrain. Slopes greater than 30° are considered obstacles. This example shows the results of using our planner to calculate a path with uncertainty rates of 0%, 2% and 4% of distance traveled, and uses the results of a Monte Carlo simulation to illustrate the importance of planning with uncertainty.

In Figure 6 we can see the results when uncertainty is not being considered. Figure 7 shows the resulting path when we use our planner and a propagation model with an uncertainty rate of 2%. Figure 8 shows the resulting path when the uncertainty rate is increased to 4%.

In order to understand the importance of planning with uncertainty, a set of Monte Carlo simulations were run for each of the previous three scenarios with and without considering uncertainty:

1) Planning without considering uncertainty: we planned a path without uncertainty, and simulated executions with uncertainty rates of 0%, 2% and 4%.

2) Planning considering uncertainty: we planned and simulated executions with uncertainty rates of 0%, 2% and 4%.

In both cases, we calculated the probability of collision as the percentage of simulations that resulted in a trajectory through obstacles. Table I shows the results of the simulations.

When there is no uncertainty (for example, if there is good GPS coverage throughout the path), the planner that considers uncertainty produces the same path as the planner without it (as expected).

TABLE I COMPARISON BETWEEN PLANNING WITH AND WITHOUT UNCERTAINTY

	Planner without uncertainty			Planner with uncertainty		
Uncert. rate	Expected cost	Average cost	Prob. of collision	Expected cost	Average cost	Prob. of collision
0%	9337	9353	0%	9337	9353	0%
2%	9337	27705	11%	18679	19052	5%
4%	9337	37726	36%	42677	49727	5%

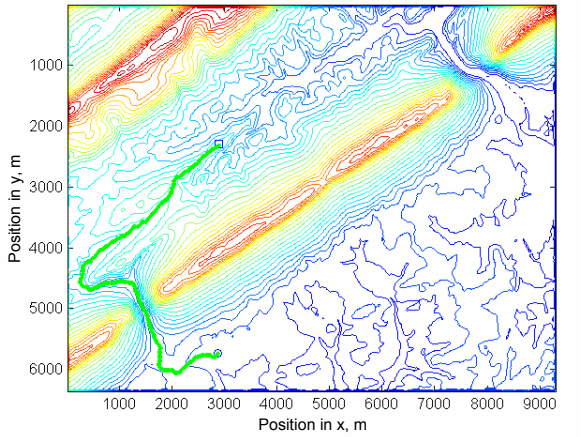

Figure 6. Path calculated assuming no position uncertainty. The start location is the small square in the top left, and the goal is the small circle in the bottom

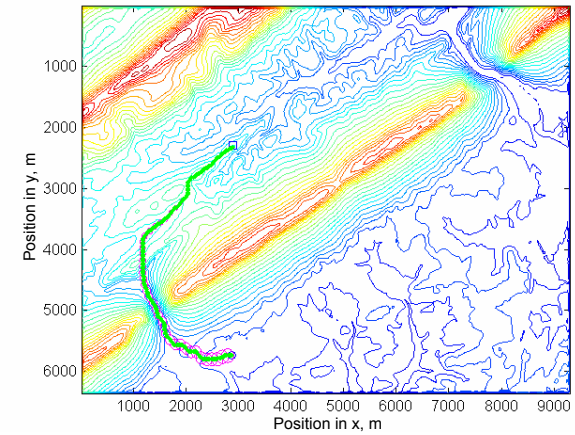

Figure 7. Path calculated with uncertainty rate $\alpha_u = 2.0\%$.

If there is a 2% uncertainty rate in the motion model, and we fail to account for it, the average cost of the resulting path is 27705, and the probability of collision is 11% (The expected cost of this path as reported by the planner with no uncertainty is 9337). If we use our planner with uncertainty, the average cost is 19052 (30% lower). The expected cost calculated by the planner (18679), is also much closer to the average cost of the simulations, and the probability of collision is significantly lower (5%).

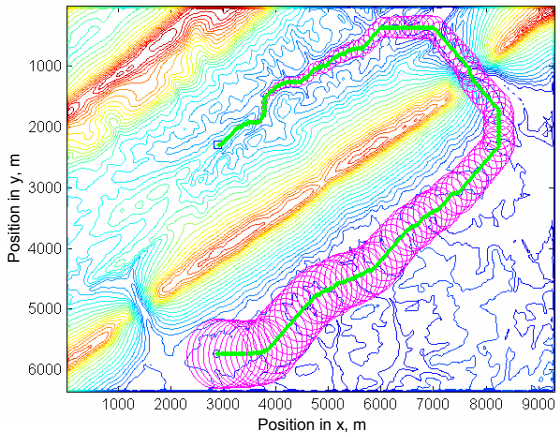

Figure 8. Path calculated with uncertainty rate $\alpha_u = 4.0\%$.

If there is a 4% uncertainty rate in the motion model but we do not account for it, the average cost of the resulting path is 37726, while the expected cost as calculated by the planner is still 9337. The probability of collision is now 36%. When we use the planner that considers uncertainty to solve this case, the average cost is 49727, the expected cost 42677, and the probability of collision 5%. The collision checking criterion for the planner with uncertainty is 2σ, which implies that paths with probabilities of collision of more than 15% would be rejected. Under this criterion, the path returned by the planner without uncertainty considerations is not a feasible path (36% probability of collision).

B. *Route Planning with GPS regions*

As discussed earlier, our planner can use GPS regions to improve the localization of the robot while planning. Figure 9 shows the resulting path for a 4% uncertainty rate when there is a GPS region in the left side of the map (small rectangle). In this case the lowest cost path is obtained by going to the GPS region first (which resets the uncertainty to zero) and then going through the gap in the lower left corner (which was not feasible without the GPS region)

C. *Complexity*

The worst-case space complexity of the current implementation of the planner is $O(n_x \cdot n_y \cdot n_u)$, where n_x, n_y and n_u are the dimensions along the x, y and u directions.

The average time complexity is $O(Q+R)$, where $Q = \alpha_u \cdot (n_x \cdot n_y)^2 \cdot n_u$ is the number of operations required to calculate the expected cost, $R = n_x \cdot n_y \cdot n_u \cdot \log(n_x \cdot n_y \cdot n_u))$ is the number of operations required by A* to calculate the path, and α_u is the uncertainty rate. For $\alpha_u > 0$, the Q term dominates the time complexity of the algorithm. If $\alpha_u = 0$, or the calculation of the expected value is performed beforehand, then the R term becomes the dominant one.

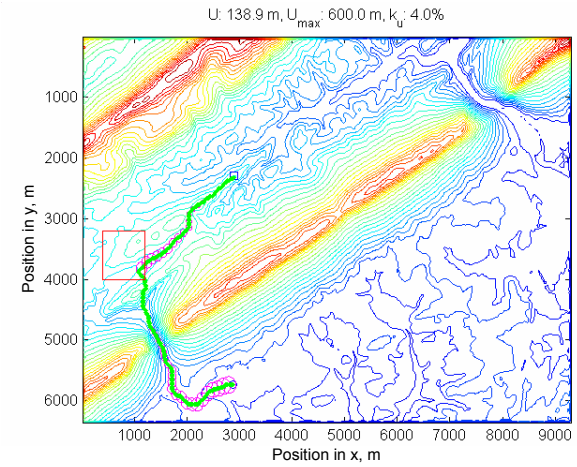

Figure 9. Path calculated with uncertainty rate $\alpha_u = 4.0\%$, and using GPS regions (small square area on the left).

However, because the uncertainty is dependent on x and y, the states expanded by the algorithm when using state dominance are far fewer than $n_x \cdot n_y \cdot n_u$, and the average time and space complexity of the algorithm is consequently much lower as well. The complete search space is a 3-D volume of dimensions $n_x \times n_y \times n_u$. State dominance reduces it to thin 3-D volume in most cases.

Figure 10 shows an example where state dominance greatly reduces the size of the search space. The figure on the center shows a cross-section of the search space at y=20, and the figure on the right shows the same cross-section when state dominance is used. We can see that state domi nance produces a search space with significantly fewer states expanded (the search space on the right is only a few states thick, while the one on the center is a full 3-D volume).

The average number of states expanded by the algorithm is $n_x \cdot n_y \cdot \bar{n}_u$, where $\bar{n}_u$ is the average number of propagations along the uncertainty axis (the thickness of the search space). $\bar{n}_u$ is typically much smaller than the size of the uncertainty dimension (n_u), making $n_x \cdot n_y \cdot \bar{n}_u << n_x \cdot n_y \cdot n_u$. Figure 11 shows $\bar{n}_u$ vs. n_u for a batch of 1800 simulations with α_u varying between 1% and 10%, and $n_x = n_y$ varying from 50 to 250 cells. We can see that even though $\bar{n}_u$ does increase with n_u, it is always a small fraction of n_u. For $n_u = 100$ the average value of $\bar{n}_u$ is 3.4, and the maximum value is 7.9.

D. *Performance*

Our algorithm takes significant advantage of two tools available for deterministic search: heuristics and state dominance. For this reason, the algorithm does not require a lengthy pre-processing stage as required by Roy and Thrun [9]. In practice, we have found the planning time to be under 1 second for worlds up to 150x150, and under 10 seconds for worlds up to 250x250. Figure 12 shows the average planning time for 1800 simulations modeling 10 different worlds with sizes from 50x50 to 250x250. The uncertainty levels used varied from 1 to 100, and the uncertainty rates from 1% to 10%. The processor used was a Pentium M 1.4GHz.

Our algorithm requires significantly fewer state expansions than processing all the states in the search space. Depending on the size of the world and the uncertainty rate the speed-up factor of our algorithm is between 8 and 80, depending on the size and characteristics of the world (see Figure 13).

VI. Conclusions and Future Work

We have introduced an efficient path planner that calculates resolution-optimal paths while considering uncertainty in position. The resulting paths are safer and have lower average costs than paths calculated without considering uncertainty.

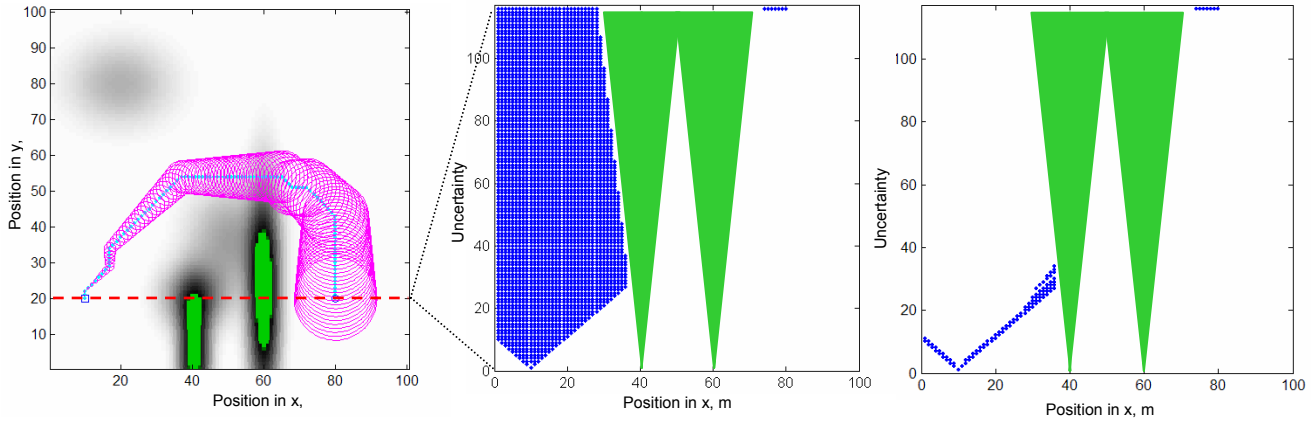

Figure 10. Effect of state dominance on reducing the search space. Left: sample world used and resulting path (darker areas are higher cost, and green areas are non-traversable). Center: cross-section of state expansion in (x,y,ε) for y=20 without state dominance (blue dots are states expanded, solid green are obstacles). Right: same cross-section with state dominance. Notice how the number of states expanded is significantly reduced.

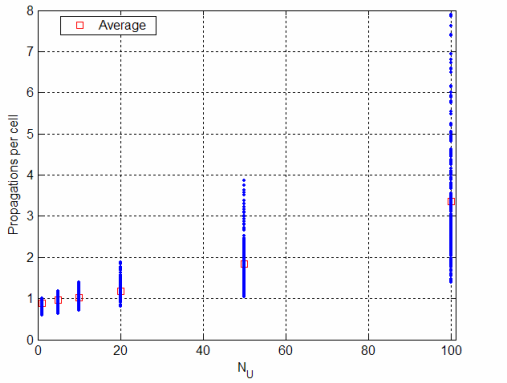

Figure 11. Propagations per cell vs number of uncertainty levels

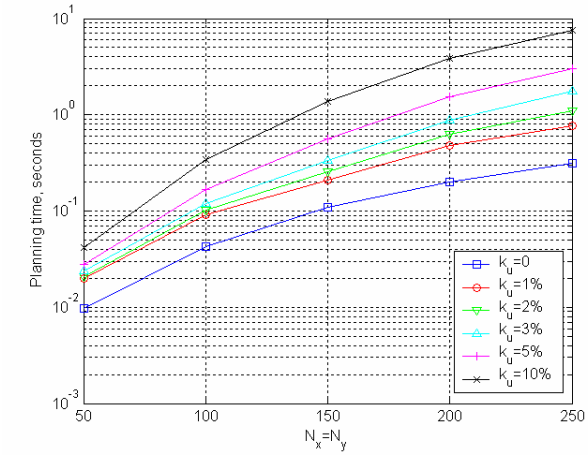

Figure 12. Planning time for different world sizes and uncertainty rates

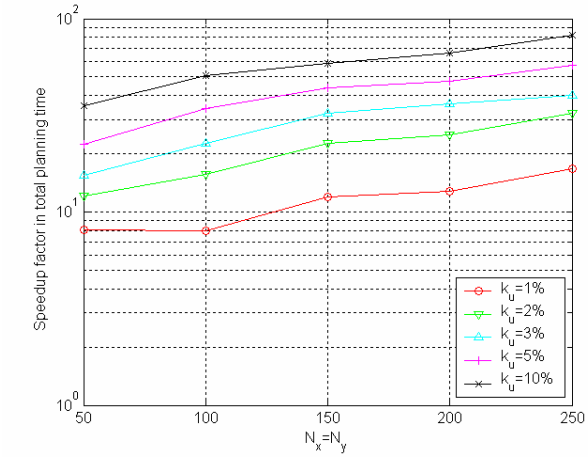

Figure 13. Speed up factor in total planning time compared to preprocessing all states.

The algorithm provides an alternative approach to the work of Roy and Thrun [9]. Our approach models a stochastic problem as a deterministic search over a simplified belief space, enabling the use of a deterministic planner such as A* to solve the planning problem. Because of the characteristics of the search space and the motion model, there are significant performance gains from the use of state dominance. Our approach runs significantly faster since it expands far fewer states.

Future work includes producing a version of the algorithm that allows for more complex representations of the error propagation model, as well as the ability to use obstacles and features in the environment for localization during the planning process.

ACKNOWLEDGMENTS

The authors would like to thank Drew Bagnell and David Ferguson for their input at different stages in the development of this research.

This work was sponsored by the U.S. Army Research Laboratory under contract "Robotics Collaborative Technology Alliance" (contract number DAAD19-01-2-0012). The views and conclusions contained in this document do not represent the official policies or endorsements of the U.S. Government.

REFERENCES

[1] B. Bonet and H. Geffner, "Planning with incomplete information as heuristic search in belief space," in Proceedings of the 6th International Conference on Artificial Intelligence in Planning Systems (AIPS), pp. 52-61, AAAI Press, 2000.

[2] J.C. Latombe, A. Lazanas, and S. Shekhar, "Robot Motion Planning with Uncertainty in Control and Sensing," Artificial Intelligence J., 52(1), 1991, pp. 1-47.

[3] J.C. Latombe, *Robot Motion Planning*. Kluwer Academic Publishers.

[4] A. Lazanas, and J.C. Latombe, "Landmark-based robot navigation". In Proc. 10th National Conf. on Artificial Intelligence (AAAI-92), 816--822. Cambridge, MA: AAAI Press/The MIT Press. http://citeseer.nj.nec.com/lazanas92landmarkbased.html

[5] B. Bouilly. "Planification de Strategies de Deplacement Robuste pour Robot Mobile". PhD thesis, Insitut National Polytechnique, Tolouse, France, 1997

[6] B. Bouilly, T. Siméon, and R. Alami. "A numerical technique for planning motion strategies of a mobile robot in presence of uncertainty". In Proc. of the IEEE Int. Conf. on Robotics and Automation, volume 2, pages 1327--1332, Nagoya (JP), May 1995. http://citeseer.nj.nec.com/bouilly95numerical.html

[7] A. Haït, T. Simeon, and M. Taïx, "Robust motion planning for rough terrain navigation".Published in IEEE Int. Conf. on Intelligent Robots and Systems Kyongju, Korea. http://citeseer.nj.nec.com/319787.html

[8] Th. Fraichard and R. Mermond "Integrating Uncertainty And Landmarks In Path Planning For Car-Like Robots" Proc. IFAC Symp. on Intelligent Autonomous Vehicles March 25-27, 1998. http://citeseer.nj.nec.com/fraichard98integrating.html

[9] N. Roy, and S. Thrun, "Coastal navigation with mobile robots". In Advances in Neural Processing Systems 12, volume 12, pages 1043—1049

[10] N. Roy, Private Conversation. September 1, 2004

[11] K. Goldberg, M. Mason, and A. Requicha. "Geometric uncertainty in motion planning". Summary report and bibliography. IRIS TR. USC Los Angeles

[12] A. Kelly, "Linearized Error Propagation in Odometry", The International Journal of Robotics Research. February 2004, vol. 23, no. 2, pp.179-218(40).

[13] J.P. Gonzalez and A. Stentz, "Planning with Uncertainty in Position: an Optimal Planner", tech. report CMU-RI-TR-04-63, Robotics Institute, Carnegie Mellon University, November, 2004.