Adapting Probabilistic Roadmaps to Handle Uncertain Maps

Patrycja E. Missiuro and Nicholas Roy
MIT Computer Science and Artificial Intelligence Lab (CSAIL)
Stata Center, 32 Vassar Street, 32-331, Cambridge, MA 02139
Email: {patrycja|nickroy}@csail.mit.edu

***Abstract*— Randomized motion planning techniques are responsible for many of the recent successes in robot control. However, most motion planning algorithms assume perfect and complete knowledge of the environment. These algorithms can fail arbitrarily badly if there are errors in the model of the environment. In contrast, real world robot systems have succeeded by using explicit representations of model uncertainty in localization and mapping to compensate for sensor error. In this paper, we propose an extension of the Probabilistic Roadmap algorithm that allows us to compute motion plans that are robust to uncertain environment models. We show that the adapted PRM generates less collision-prone trajectories with fewer samples than the standard method.**

I. Introduction

Randomized motion planners, such as the Probabilistic Roadmap (PRM) [1] and the Rapidly-exploring Randomized Tree (RRT) [2] have been very successful in solving planning problems for robots with many degrees of freedom, problems that were previously considered intractable [3], [4], [5]. However, these algorithms depend on having a complete and accurate model of the world.

In carefully engineered settings, such as a robot manipulator on the factory floor, the "perfect map" assumption is a reasonable one, as the robot's environment can be precisely measured and tightly controlled. In contrast, major successes in planning have been driven by appropriate models of uncertainty. Statistical inference techniques such as Markov localization and the Kalman filter [6], [7], [8] have enabled mobile robots to navigate safely in populated, dynamic, uncertain environments without getting lost. For robots with few degrees of freedom, non-randomized motion planning algorithms have become more robust to sensor and model errors by using robust control techniques that incorporate both the cost of control errors and position uncertainty [9]. In order to allow the same robust operation for high-dimensional robots such as humanoid robots interacting with the real world, we need to incorporate knowledge of model uncertainty into the motion planning.

This paper extends the Probabilistic Roadmap (PRM) algorithm in two ways. Firstly, we show how to modify randomized sampling of poses in order to minimize the number of samples required to express good plans. Our results show that good sampling strategies that respect map uncertainty can be used to substantially reduce the number of samples required to express good (less collision-prone) trajectories in uncertain worlds. Since the trajectory search process is quadratic (Dijkstra search algorithm [10], A*) in the number of nodes, minimizing the set of samples is a major factor in keeping the planning problem tractable. Secondly, we show how to evaluate PRM actions efficiently in the context of uncertainty and generate motion plans with minimal expected cost.

Notation used:

C-space - configuration space; the space of all robot poses. *C-space* $\equiv$ *C-free* $\cup$ *C-obst*
C-free - the space of all collision-free robot poses.
C-obst - all poses resulting in collision with obstacles.

II. The Probabilistic Roadmap Method

Given a map, robot dimensions, and the start and goal positions, the PRM method [1], [11] aims to produce a valid path from start to goal. The PRM solves the motion planning problem in two stages [1], [12][1]:

1) The *preprocessing phase*, during which points are sampled from *C-space*, and only collision-free samples are retained. As a result, an approximation of *C-free* [13] is built from a set of discrete collision-free poses. The retained samples constitute the nodes of a graph. The graph edges are found using a local planner that determines collision-free connectivity of each node to its k nearest neighbors. If the local planner finds the path between two nodes to be collision free, then an edge between the nodes is added to the graph; the edge is labelled with an associated cost, such as distance. The local planner may be relatively simple, such as straight-line distance, or a more sophisticated kinodynamic planner [14].
2) The *query phase*, when a graph search algorithm is used to find a path from the start to the goal location. In the conventional setting, the lowest-cost path generally implies the shortest collision-free path.

The power of the PRM resides in the preprocessing stage, which exploits the fact that even if *C-free* cannot be tractably computed, it is relatively efficient to determine if an arbitrary pose lies in *C-free*. The PRM learns a discrete node set approximation of *C-free* by sampling poses from *C-space* and rejecting samples that lie in *C-obst* (i.e., that collide with

[1]Our work extends a classic variant of the PRM which tests for collisions during sampling and local planning as opposed to the Lazy PRM method [11].

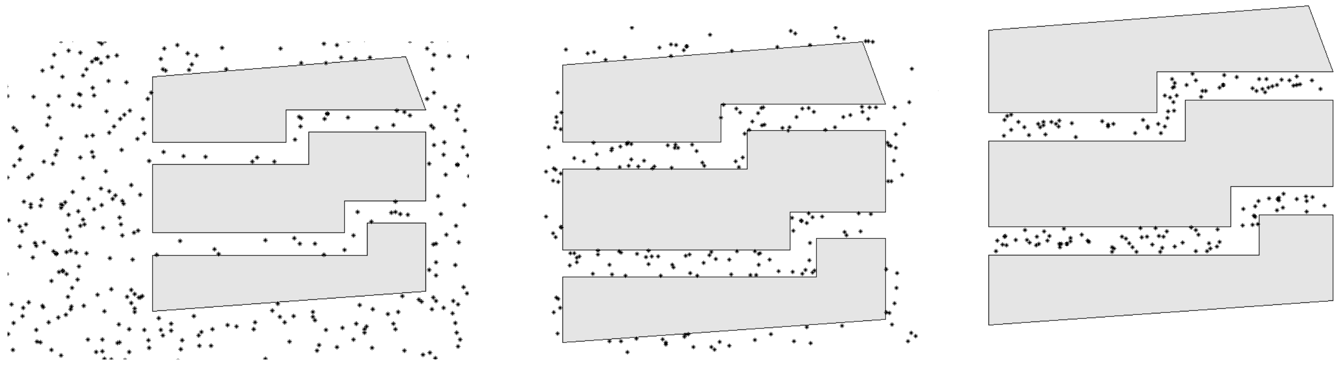


Fig. 1. A comparison of conventional sampling, ~~column-wise~~: Uniform, Gaussian, Bridge

obstacles). If the sampling strategy is good, then a relatively small number of nodes suffices to approximate *C-free*. The issue with PRM is determining good sampling strategies that capture the topology of ~~the~~ *C-free* with as few samples as possible, reducing the time to search for the path during the *query stage* of PRM. Although the PRM solution is complete in the limit of sampling an infinite number of nodes, the rate of convergence depends on the sampling strategy and difficulty of the environment [15]. An example of a challenging *C-free* configuration is a narrow passage where a naïve sampling strategy (such as uniform [16]) may require a large number of samples before it approximates a trajectory through such a corridor. Three commonly used sampling strategies are:

Uniform sampling [16] samples poses with uniform probability from the *C-space* (left in figure 1). The disadvantage of this method is that no information about the map is used and unnecessarily many points are sampled in the empty regions of space.

Gaussian obstacle-based sampling [17], [18], [19] tries to capture the fact that the optimal length paths in the *C-space* follow around and near the obstacles. It samples points near and only in the presence of obstacles (center in figure 1). A first sample, $\mathbf{s_1}$, is drawn using the uniform sampler. A distance d is sampled from a normal distribution with mean and covariance based on knowledge of obstacle density in *C-space*. A second sample, $\mathbf{s_2}$ is then drawn from a uniform distribution of radius d centered at $\mathbf{s_1}$. If both samples are either inside the *C-free* or in *C-obst*, then both are discarded. Otherwise, the sample inside *C-free* is retained.

Bridge obstacle-based sampling [20] biases the sampling towards narrow passages where retaining connectivity of the *C-free* is difficult (right in figure 1). A first sample $\mathbf{s_1}$ is drawn using the uniform sampler and discarded if it lies in *C-free*. Otherwise, a distance d is drawn from a normal distribution with mean and covariance based on *a priori* knowledge of the topology of *C-space*. A second sample $\mathbf{s_2}$ is drawn from a uniform distribution with radius d centered at $\mathbf{s_1}$. If $\mathbf{s_2}$ lies in *C-free*, both samples are discarded. If, however, $\mathbf{s_2}$ is in *C-obst*, then the midpoint $\mathbf{m}$ between $\mathbf{s_1}$ and $\mathbf{s_2}$ is computed. If pose $\mathbf{m}$ falls within *C-free*, then $\mathbf{m}$ is added to the graph. This sampler cannot generally be used alone since it samples only in corridors.

III. Uncertain Worlds

A. Modelling world uncertainty

The sampling and planning methods described so far assume that the features of the world are known exactly, and that there is no ambiguity in the location of the obstacles. However, this is an unreasonably optimistic assumption, since the mapping techniques used to build models contain sensor and odometry errors. For example, the Kalman filter [7] contains explicit representations of the uncertainty of the world features/vertices. In this paper, we represent the uncertainty in the obstacle ~~position~~ as probability distributions over obstacle features. This choice is motivated by a point-feature-based SLAM algorithm [7] such as the Kalman filter, which represents point feature distributions as multivariate Gaussians.

Figure 2 shows an example map in which the point features (the vertices of the polygons) have explicit uncertainty modelled with normal distributions. In figure 2(a), we visualize the uncertainty of the location of each vertex by drawing one standard deviation uncertainty ellipse, computed from the map covariance Σ, around each vertex. Possible world models are shown in figure 2(b). We do not know *which* world model is the correct one: each of these world models has a different likelihood under the joint probability distributions of the polygon vertices. Also, since the features of the two "$\bigwedge$"-shaped obstacles in figure 2(a) are known with less precision, the trajectories above them are more likely to collide with obstacles realized in figure 2(b). The preferred path would go around these obstacle, not directly across to the goal.

In the following section, we propose a method to sample poses when the world is uncertain. We adapt the sampling strategies to adjust for map uncertainty. In subsequent sections, we describe how to incorporate the obstacle uncertainty into motion costs and trajectory generation.

B. Proposed algorithms for sampling in uncertain worlds

~~In an uncertain world the actual positions of obstacles are unknown. It is unclear whether a sampled pose s collides with some obstacle or is in *C-free*, thus whether we should accept or reject it.~~ We propose that the decision to accept or reject s be a function of the probability that a pose is in *C-free*.

Our method can be logically broken down into two steps:

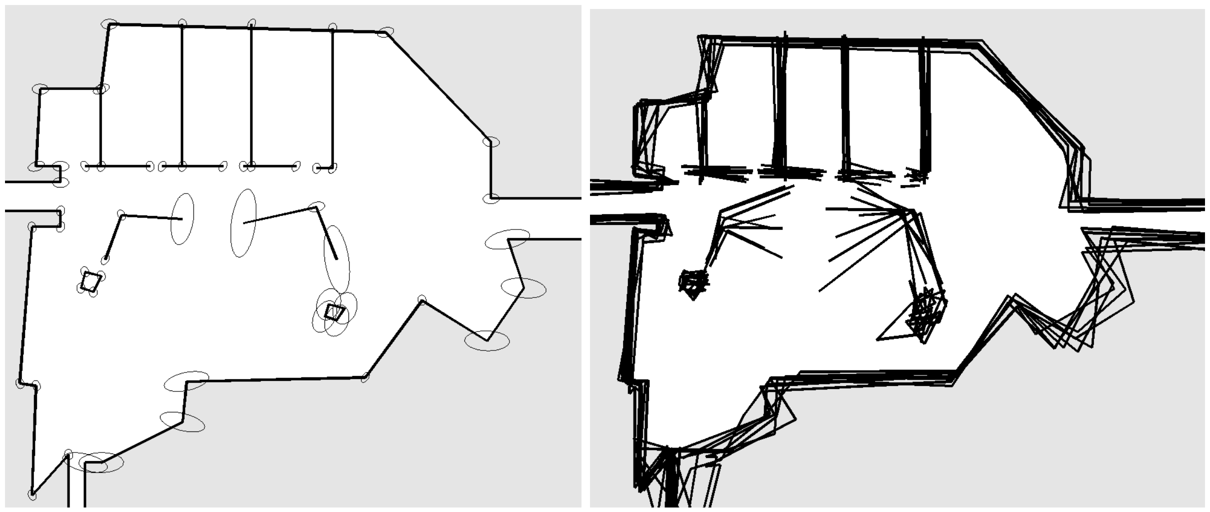


Fig. 2. Map of Robotics3 lab in Stata, MIT used as an example uncertain worldin $\mathcal{R}^2$. Left: the maximum-likelihood map, shown with distributions of the features. Ellipses visualize one standard deviation from the mean. Right: example maps sampled from distributions.

1) Estimation of the collision probability for a sampled pose $\mathbf{s}$.
2) Reject or accept $\mathbf{s}$ based on the collision probability.

We will incorporate the world uncertainty into both phases of the planning process. This does not constitute double-counting; if we knew *a priori* the optimal path, then the optimal sampler would sample only poses on the path. By incorporating knowledge used in cost calculations, we can focus the sampling appropriately. For example, our adapted samplers are biased to sample closer to more certain ~~side~~ of the corridor space. This allows the PRM planner to focus its search in regions where collisions are less likely.

C. Estimation of collision probability

Let us use $\delta(\mathbf{s}, \mathbf{W})$ to denote a function that determines whether or not $\mathbf{s}$ is in the *C-free* of a possible model $\mathbf{W}$:

$$\delta(\mathbf{s}, \mathbf{W}) = \begin{cases} 1 & : \mathbf{s} \in \text{C-free of } \mathbf{W} \\ 0 & : \text{otherwise.} \end{cases} \quad (1)$$

~~Note that the world $\mathbf{W}$ is a single instance of sampled world from the distribution~~ as shown in figure 2(b).

The probability that a sample is in *C-free* can then be computed by

$$p\left(\mathbf{s} \in \text{C-free}\right) = \int_{\mathbf{W}} \delta(s, \mathbf{W}) p(\mathbf{W}) dV_{\mathbf{W}} \quad (2)$$

where $p(\mathbf{W})$ is a multivariate probability density of the world model at the world $\mathbf{W}$, $V_{\mathbf{W}}$ is an integration variable being a volume in a configuration space of all degrees of freedom of all obstacles, and the world $\mathbf{W}$ can be uniquely characterized by the set of vertices of all obstacles.

To simplify the problem we assume that the obstacles are not correlated. Correlation among obstacle configurations requires domain knowledge which is usually not available from sensing and increases computational intractability. Assuming obstacle independence, we can approximate the likelihood that $\mathbf{s}$ is in *C-free* by a product of probabilities that $\mathbf{s}$ does not collide with any of the obstacles:

$$p(\mathbf{s} \in \text{C-free}) \approx \prod_{i \in \text{obstacles}} [1 - p_{coll,i}(\mathbf{s})] \quad (3)$$

where $p_{coll,i}(\mathbf{s})$ is the probability of colliding with obstacle i. Given that, the total collision probability is given by:

$$p_{coll,total}(\mathbf{s}) = 1 - p(\mathbf{s} \in \text{C-free}) = 1 - \prod_{i \in \text{obstacles}} [1 - p_{coll,i}(\mathbf{s})] \quad (4)$$

The remaining task is to compute the probability of collision with one obstacle, $p_{coll,i}(\mathbf{s})$. One simple way to estimate the probability that a pose lies in *C-free* is to use a Monte Carlo technique, sampling random points from the obstacle vertices distribution functions. A possible obstacle results from connecting pairs of neighboring sampled vertices. The probability that the pose $\mathbf{s}$ lies inside some obstacle i is approximated by

$$p_{coll,i}(\mathbf{s}) \approx \frac{N_{collision}}{N_{total}} \quad (5)$$

where $N_{collision}$ is the total number of pose collisions with obstacle i and N_{total} is the total number of trials. The difficulty with the Monte Carlo approximation is that it requires many samples for an accurate estimate, leading to computational intractability. We propose to estimate the collision probability in a more efficient manner, as follows.

IV. Efficient Estimation of Collision Probabilities via Nearest Point Method

On each obstacle in our world model, we can find some point $\mathbf{p}^*$ which is closest to the robot. Our estimate of the collision probability is based on the realization that the likelihood for a robot to collide with a particular obstacle is dominated by likelihood of colliding with $\mathbf{p}^*$. The robot will generally collide with the obstacle if the point $\mathbf{p}^*$ appears in

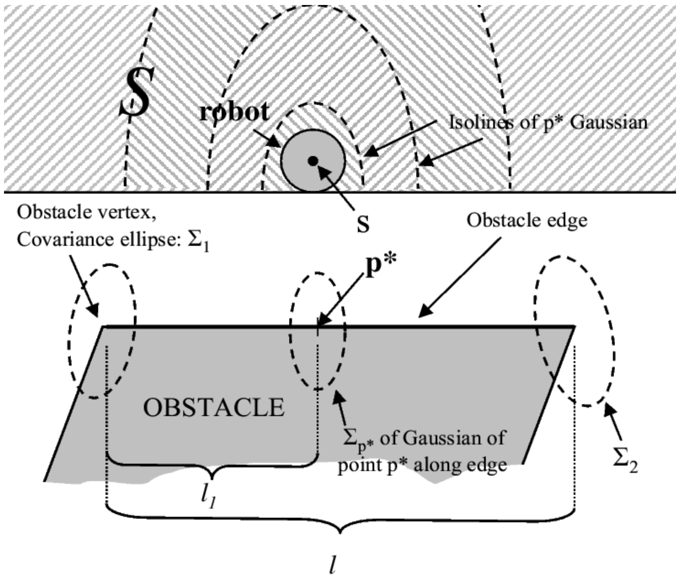


Fig. 3. The distribution of a point $\mathbf{p}^*$ on the obstacle edge which is closest to $\mathbf{s}$, can be computed from normal distributions of neighboring vertices. Integrating $\mathbf{p}^*$ distribution over the region S gives an estimate of the collision likelihood with a particular obstacle.

the half-space bounded by the line tangent to the robot and parallel to the obstacle edge (the area S in figure 3). Due to the uncertainty of the obstacle, the position of $\mathbf{p}^*$ is also uncertain, but the probability of collision can be approximated using only knowledge of the distribution of $\mathbf{p}^*$. Assuming Gaussian distributions on the vertices, if $\mathbf{p}^*$ lies on some "obstacle edge"[2], $\mathbf{p}^*$ is distributed according to a Gaussian with covariance derived from the covariance matrices of the neighboring vertices:

$$\Sigma_{\mathbf{p}^*} = \frac{{l_1}^2}{l^2}\Sigma_2 + \frac{(l-l_1)^2}{l^2}\Sigma_1 \tag{6}$$

where l is the length of the edge, Σ_i are the corresponding covariances of vertices and l_1 is a distance from $\mathbf{p}^*$ to the vertex with covariance Σ_1 (see figure 3) [21]. In an uncertain world, $\mathbf{p}^*$ minimizes the Mahalanobis distance between the robot pose and the obstacle, but we cannot solve for $\mathbf{p}^*$ since we do not have a closed-form solution for the distribution of an entire 'edge' and in practice the statistical distance can be sufficiently approximated by the Euclidean case.

In order to estimate the collision probability, we integrate a Gaussian function $\mathcal{N}(\mathbf{p}^*, \Sigma_{\mathbf{p}^*})$ over the half-space:

$$p_{coll,i}(\mathbf{s}) \approx \int_S \mathcal{N}(\mathbf{p}^*, \Sigma_{\mathbf{p}^*})dV. \tag{7}$$

We ran experiments to validate the Nearest Point method using Monte Carlo method as a benchmark. The experiments ~~showed that~~ Nearest Point approximation accurately estimates the collision probability. Due to lack of space, we omit ~~the detailed results~~.

[2]The notion of an *'obstacle edge'* is not well-defined since our world model consists of polygons defined by distributions over vertices. We define an *'obstacle edge'* as the line connecting the means of a pair of vertices.

A. Rejection function based on collision probability

Equation 4 provides a way to estimate the cumulative collision probability for a given pose ~~efficiently; we~~ reject or accept sampled poses based on this probability, $p_{coll,total}(\mathbf{s})$: the higher the collision likelihood of a sampled pose, the less likely our sampling algorithm will accept such pose. In practice, we ~~shaped~~ our rejection function to generally reject samples that ~~lay~~ inside the "nominal obstacle~~.~~"[3] based on the notion that samples where robot is 50% or more likely to collide with obstacles are costly and unlikely to be used by the planning stage. This indicates that our sampling strategy is unsatisfactory from a mathematical perspective, but our experimental results ~~will~~ indicate that our sample measure outperforms conventional techniques.

B. Pose rejection algorithm

We summarize the resulting accept/reject decision in the following algorithm:

Algorithm 1: Accept or reject pose $\mathbf{s}$
Input: pose $\mathbf{s}$, all obstacles
Output: boolean s_{reject} describing whether to reject pose

```
(1)   foreach obstacle $i$
(2)       Compute $p_{coll,i}$, collision probability for $\mathbf{s}$ and
          obstacle $i$, using Monte Carlo or Nearest Point
          method or any other
(3)   Compute the resulting collision probability for all the
      obstacles:
      $p_{coll,total}(\mathbf{s}) = 1 - \prod_i (1 - p_{coll,i}(\mathbf{s}))$.
(4)   $p_{reject}(\mathbf{s}) \approx p_{coll,total}(\mathbf{s})$.
(5)   $s_{reject}$ = randomly sample from Bernoulli distribution
      where $p = p_{reject}(\mathbf{s})$.
(6)   return $s_{reject}$
```

Adapted Sampling Strategies

We modify each sampling strategy to accept or reject samples stochastically, depending on the likelihood of pose $\mathbf{s}$ lying in *C-free*.

Adapted Uniform Sampling: poses are drawn uniformly at random from *C-space* and each sample is retained or rejected based on collision likelihood as summarized in Algorithm 1 (see figure 4 left).

Adapted Gaussian sampling: Gaussian sampling generates samples that are close to obstacles by generating pairs of samples that lie on opposite sides of an obstacle edge, one inside the obstacle, the other outside. ~~In an uncertain world,~~ we retain samples based on probability that the *pair* of samples lies on opposite sides of an edge. In contrast to the uniform sampler, we first generate a sample $\mathbf{s}_1$ inside the "nominal obstacle," where the probability of rejection is high, in our setup $p_{reject}(\mathbf{s}) = 1$. Let us call this sample an *"anchor"*. Next, we sample a distance d, and generate a second sample

[3]We define "nominal obstacle" as the obstacle with the vertices at the maximum likelihood ~~positions~~ which are the means of Gaussian distributions.

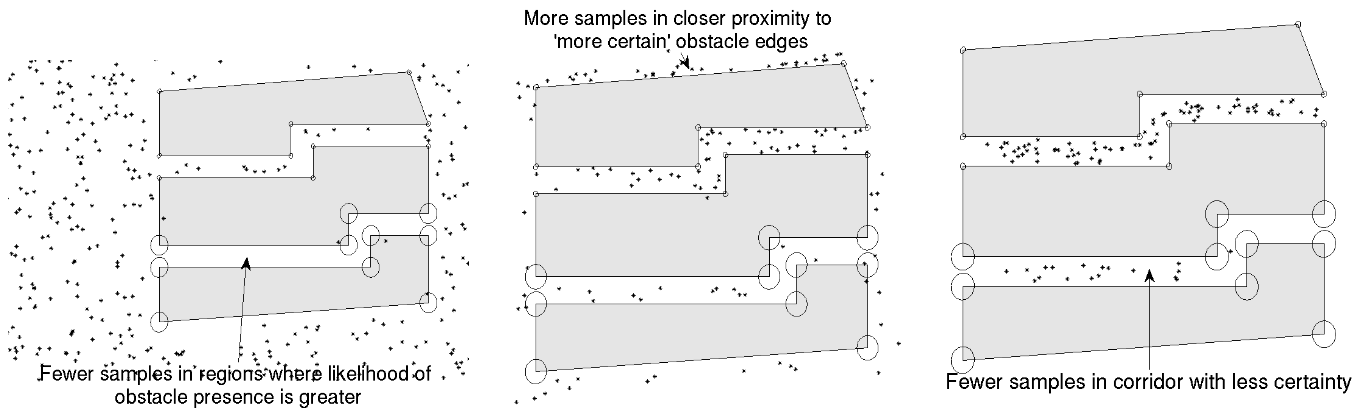

More samples in closer proximity to 'more certain' obstacle edges
Fewer samples in regions where likelihood of obstacle presence is greater
Fewer samples in corridor with less certainty

Fig. 4. A comparison of the sampling strategies adapted for uncertainty sampling, ~~column-wise~~: Uniform, Gaussian, Bridge

s_2 from a uniform distribution of radius d[4]. This sample is then retained according to Algorithm 1.

We require the *"anchor"* points to be inside the "nominal obstacle," ~~because we do not want to produce too distant poses.~~ When we originally did not use this heuristic, many of the resulting points were very far from the obstacles. This is a side effect of the fact that, if the first point is already outside the "nominal obstacle," and happens to be labelled as being in *C-obst* (since the process is stochastic), the second sample may end up far from the obstacle. Since $\mathbf{s_2}$ is far from an obstacle, $p_{reject}(\mathbf{s_2})$ will be small, and $\mathbf{s_2}$ will be readily accepted as a sample. By keeping the first sample anchored, we retain the *near-obstacle* feature of the original method. Figure 4, middle, shows Gaussian sampling in an uncertain world.

Adapted bridge sampling: Once again, in order to retain the *near-obstacle* characteristics of the original bridge method, the adapted bridge method secures anchoring of the first two sampled points inside the "nominal obstacle" by generating two samples that each have $p_{\text{reject}} = 1$. ~~Next, the midpoint $\mathbf{m}$ is computed. This midpoint is then retained based on Algorithm 1.~~ As a result, narrow pathways characterized by higher certainty in the obstacle positions are favored, and the search for a path with ~~lower collision chance~~ is focused on those passages (see figure 4, right)).

Once a set of samples has been generated using any hybrid of sampling strategies, we use a *local planner* to determine if pairs of samples can be connected. We have not modified this stage of the PRM, and we use k-nearest-neighbors planner using the Euclidean distance metric in a world defined by "nominal obstacles." We ignore uncertainty in the local planner, because we want a spatially interconnected graph.

V. Planning in Uncertain Worlds

To generate good trajectories, the *query phase* of PRM also needs to incorporate the world uncertainty. Standard planners find paths that minimize some quantity such as distance or travel time. This quantity is represented via a cost function associated with travelling the segments comprising the optimal path. We extend this approach with a minimum-collision-cost *MCC* planner which generates trajectories that minimize the expected cost of collision when travelling edges in the graph. We define the expected cost of traversing a path segment between robot poses $\mathbf{s_1}$ and $\mathbf{s_2}$ as:

$$C(\mathbf{s_1},\mathbf{s_2}) = \; p_{seg\_coll,total}(\mathbf{s_1},\mathbf{s_2}) * C_{collision} + [1 - p_{seg\_coll,total}(\mathbf{s_1},\mathbf{s_2})]||\mathbf{s_1},\mathbf{s_2}|| \quad (8)$$

where $p_{seg\_coll,total}(\mathbf{s_1},\mathbf{s_2})$ is the probability that a robot collides with any obstacle while travelling on the line segment from $\mathbf{s_1}$ to $\mathbf{s_2}$, and $C_{collision}$ is some fixed estimate of how much it would cost when a robot collides with something. $C_{collision}$ can also be chosen to be a function of distance; for example, travelling 1 km less may be worth the risk of colliding with an obstacle.

With the previous assumption that individual obstacle distributions are independent, the total probability of collision can be approximated as:

$$p_{seg\_coll,total}(\mathbf{s_1},\mathbf{s_2}) = 1 - \prod_{i \in \text{obstacles}} [1 - p_{seg\_coll,i}(\mathbf{s_1},\mathbf{s_2})] \quad (9)$$

where $p_{seg\_coll,i}(\mathbf{s_1},\mathbf{s_2})$ is the probability of hitting a particular obstacle i when travelling between $\mathbf{s_1}$ and $\mathbf{s_2}$. We cannot use the approximation technique of equation 7 to calculate the probability of collision, as this method computes the probability of a single pose. Integrating equation 7 along the edge would overestimate the probability of collision due to violated independence assumptions. We therefore use a Monte Carlo technique as in equation 5, but for any collision during simulated motion along the edge. We note that in the example case of 2D world with polygonal obstacles and a holonomic robot, a collision occurs when the line segment from $\mathbf{s_1}$ to $\mathbf{s_2}$ ~~gets within minimum~~ distance to obstacle i, that is, a robot travelling along this path segment collides

[4]We need to keep in mind that, because our map is uncertain, the variance of normal distribution of d must be greater than the variance in map features/vertices. This is because the sampled points need to be placed sufficiently far from the obstacles in order not to have high rejection probability.

with i.[5] We compute the joint probability $p_{seg\_coll,total}(\mathbf{s_1},\mathbf{s_2})$ via equation 9, and incorporate it into the cost of traversing segment $(\mathbf{s_1},\mathbf{s_2})$ using equation 8. Finally, a graph search algorithm (A*) uses the estimated collision cost along each roadmap edge and returns the minimum cost path.

VI. Experiments: Adapted Sampling and Planning

In our preliminary experiments, we measured the effects of adapted sampling strategies on the overall performance of the PRM algorithm. ~~Each experiment was a motion-planning problem in the 2D plane for a simple circular robot from any start to any goal location.~~ The performance ~~was~~ measured by repeatedly simulating a robot trajectory in a large set of worlds (one simple world shown in figure 6) and recording ~~how many collisions occurred.~~

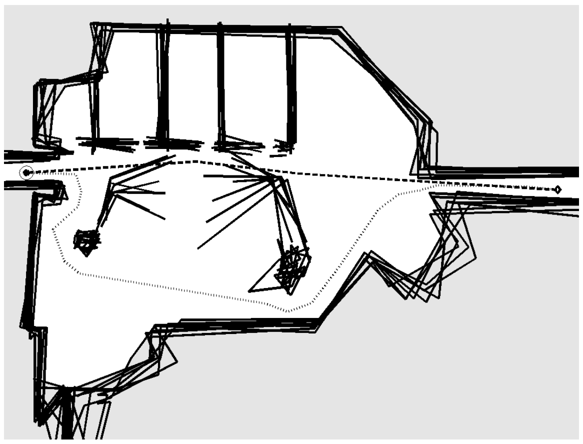


Fig. 6. Map of the MIT Stata Robotic Lab area, trajectories with obstacles perturbed, standard A* planner (dashed), minimum-collision-cost A* planner (dotted)

We examined planner robustness under three sampling methods in conventional and adapted form, for a total of 6 variants. We used the Nearest Point method to determine whether a sampled pose results in a collision for the adapted sampling variants. The local planner used the straight-line distance and maximum of 12 nearest neighbors ($k = 12$). In the query stage, we always used the minimum-collision-cost planner in order to focus on the impact of the sampling method. The resulting path was smoothed by rerunning the *MCC* planner on the fully-connected set of path nodes, including start and goal.

Plots in figure 5 show that adapted sampling improves the performance of the planner resulting in higher quality paths (fewer collisions) while keeping the number of nodes relatively small. Particularly, in the case of sample numbers less than 200, the adapted sampling methods do substantially better, resulting in collision rates that the standard sampling needs more than 1000 points to obtain. This is because the adapted sampling methods bias the sampling into regions of certainty, encouraging pathways through those regions with fewer samples overall.

[5]For computational efficiency, we do not perform the Monte Carlo trial if the obstacle and the path edge are statistically far from each other and just set $p_{seg\_coll,i}(\mathbf{s_1},\mathbf{s_2}) = 0$

The adapted sampling methods add a small amount of computational overhead to the standard sampler~~,~~ because samples are rejected based on their collision probability and more loop iterations are required to collect samples. We found that the time to sample was anywhere from one to three times as long as the standard sampling method. However, the sampling stage of the PRM algorithm is a very small fraction (generally less than 5%) of the overall time to plan and the planner benefits from a reduced sample set to plan with.

A. Modelling Orientation

Figure 7 shows planning for rectangular holonomic robots of varying dimensions, using standard and *MCC* planners. For these rectangular robots, we used basic (a) and adapted uniform sampling (b) to sample robot poses (x, y, θ) for a total of 3 dimensions. Note that in both (a) and (b), as the length of the robot increases, the path needs to accommodate ~~it~~ by moving the robot away from the obstacle before it can turn. In figure 7(a) the differences in paths are due purely to robot geometry since conventional planner is used. In figure 7(b) path changes due to both geometry and uncertainty and the robots' trajectories obtained with *MCC* planner are ~~further~~ from the uncertain obstacle. ~~Simulations by perturbing~~ the obstacles show that the collision rates decrease from an average of 87% for the conventional planner to 35% for the *MCC* planner.

B. Modelling Environmental Dynamics

Figure 8 shows results obtained when planning a path for a circular B21 robot using a conventional, minimum-distance planner (the solid trajectory) and the probabilistic, minimum-collision-cost (*MCC*) planner (dashed) employing segment collision cost in equation 8. The conventional planner returns a path that is $41.2m$ long where some path segments have $p_{seg\_coll,total}$ estimated by *MCC* to be as high as 0.84. The *MCC* planner selects the longer ($51.0m$) path, but with smaller (less than 0.40 for a given segment) collision likelihood.

VII. Applicability to Multidimensional Problems

We have shown ways to incorporate uncertainty metrics into the PRM and demonstrated the concepts on 2D world examples, including robots with rotational dependencies. However, our extension to model uncertainty can be applied to problems with more dimensions, involving robots with multiple degrees of freedom operating in 3D. There, sampling can be decomposed for each dof of the robot. For example, each limb of a humanoid robot can be treated independently and the overall collision probability estimated by combining the probability of collision ~~of~~ individual components. It has been shown experimentally [15] that when ~~problem dimensionality~~ increases, randomized probabilistic sampling and planning methods fare better than deterministic methods with respect to computational complexity. Occupancy grids (aka evidence grids) [22] are deterministic and their complexity increases as

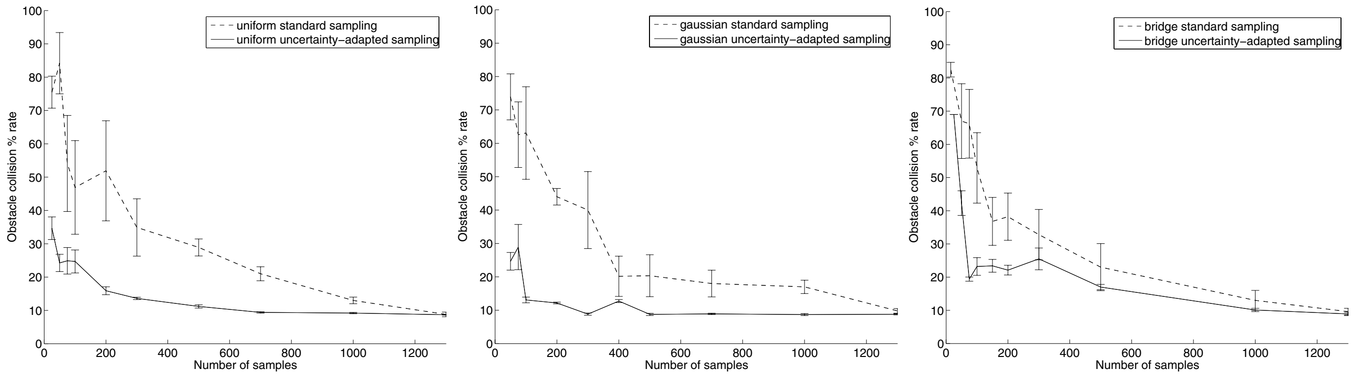


Fig. 5. The PRM performance in terms of collision rates using original and adapted sampling as a function of the sample number. The minimum-collision-cost planner was used in both cases in order to focus on the influence of the sampling strategy. We note that collision rates when sample numbers are small (under 100) are substantially lower when using our adapted sampling techniques.

$O(n^k)$ where k is the number of dimensions, and n is inversely proportional to graph resolution.

We must point out that estimating collision probabilities in 3D gets difficult since ~~now the closest point~~ belongs to an obstacle plane, and can no longer be approximated by a linear combination of the endpoints. To remedy this problem, a new uncertainty model is needed. Our future work will focus on modelling uncertainty in higher dimensions and methods to incorporate stochastic world information into collision probabilities and path costs.

VIII. Related Work

Most randomized planning algorithms are not explicitly robust to model errors. Some sampling methods such as Medial Axis PRM (MAPRM) [23] have attempted to generate trajectories that are robust to model errors since samples on the medial axis of the plane maximize their distance from obstacles. However, such conservative sampling methods do not incorporate uncertainty into the cost function and cannot bias samples to be closer to more certain obstacles [23].

In RRTs [2], which utilize a local search planner and keep a fully connected tree, adapted sampling could be done on a local level and the decision whether or not to extend a branch could utilize the cost function of the minimum-collision-cost planner. The Probabilistic Roadmap of Trees method (PRT) [14] uses a random tree algorithm as a subroutine in PRM, where the nodes in the PRM roadmap are trees. Since multiple RRTs can be grown in parallel we could achieve substantial speedups as we can simultaneously explore and evaluate multiple regions of a map for trajectory quality. Our modified sampling and planning framework could therefore be incorporated into both the local planner (RRT) and the global planning mechanism (PRM).

Leven and Hutchinson in [24] address a problem of changing environments ~~by~~ a variant on PRM, where sampled nodes are updated according to information about the changing state of the world. Unlike our work, they do not rate feasibility of collisionless travel through different regions of space and their motion plan does not rate uncertainty. Similarly, Berg and Overmars in [25] use the PRM to plan in dynamic environments by first generating a global path assuming a static world, and then using local planners to deal with moving obstacles, therefore avoiding having to recompute the global trajectory. Similarly, their global path does not incorporate uncertainty, and their motion plan selects a global trajectory assuming that the world is certain.

IX. Conclusion

We demonstrated that conventional, purely geometric motion planning algorithms can be extended to allow robust motion planning when the true state of the world is not known exactly. In particular, we proposed incorporating uncertainty into the PRM sampling and planning. We adapted three popular sampling techniques: Uniform, Gaussian, and Bridge to focus samples on regions of higher certainty. In the planning stage, we modelled the cost of potential collisions in travelling through uncertain regions of the configuration space. The experiments showed that using a stochastic rejection function in sampling biases the path into regions of more certainty, resulting in fewer overall collisions than the traditional approach, and with small sample numbers. Our preliminary results show that the uncertainty-adapted PRM leads to substantially more robust paths than the conventional PRM in the face of map uncertainty.

References

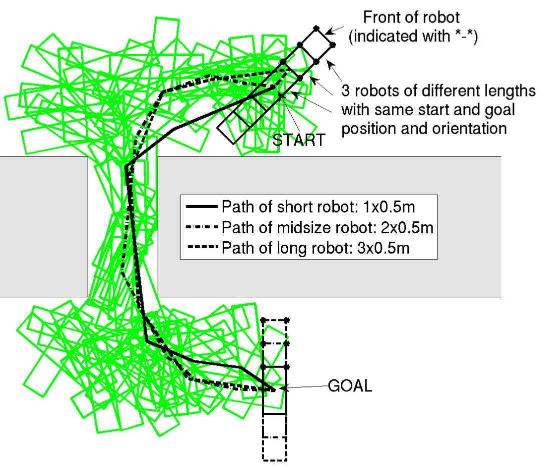
[1] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. Overmars, "Probabilistic roadmaps for path planning in high dimensional configuration spaces," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[2] S. M. LaValle and J. J. Kuffner, "Randomized kinodynamic planning," *International Journal of Robotics Research*, vol. 20, no. 5, pp. 378–400, May 2001.

[3] E. Frazzoli, M. Dahleh, and E. Feron, "Real-time motion planning for agile autonomous vehicles," in *Proc. AIAA Conf. on Guidance, Navigation and Control*, 2000.

[4] M. G. Choi, J. Lee, and S. Y. Shin, "Planning biped locomotion using motion capture data and probabilistic roadmaps," *ACM Trans. Graph.*, vol. 22, no. 2, pp. 182–203, 2003.

[5] J. Kuffner, K. Nishiwaki, S. Kagami, M. Inaba, and H. Inoue, "Motion planning for humanoid robots," in *Proc. 20th Int'l Symp. Robotics Research (ISRR'03)*, Italy, 2003.


Front of robot
(indicated with *-*)
3 robots of different lengths
with same start and goal
position and orientation
START
Path of short robot: 1x0.5m
Path of midsize robot: 2x0.5m
Path of long robot: 3x0.5m
GOAL

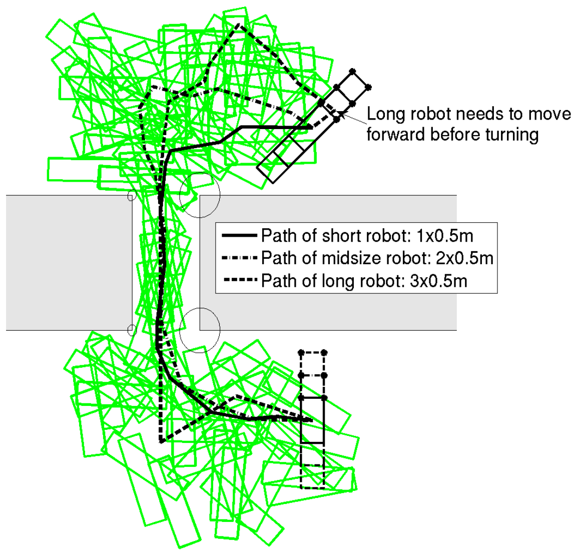

Long robot needs to move
forward before turning
Path of short robot: 1x0.5m
Path of midsize robot: 2x0.5m
Path of long robot: 3x0.5m

Fig. 7. Rectangular robot sampling of 3 dimensions: position and orientation, and planning; top: standard PRM planning without uncertainty, bottom: sampling and planning incorporating uncertainty. Also note how the robot trajectories change with increasing robot length.

[6] R. Smith, M. Self, and P. Cheeseman, *Autonomous Robot Vehicles*. Springer-Verlag, 1990, ch. Estimating uncertain spatial relationships in robotics, pp. 167–193.

[7] J. Leonard and H. Durrant-Whyte, “Mobile robot localization by tracking geometric beacons,” *IEEE Transactions on Robotics and Automation*, vol. 7, no. 3, pp. 376–382, June 1991.

[8] D. Fox, W. Burgard, and S. Thrun, “Markov localization for mobile robots in dynamic environments,” *Journal of Artificial Intelligence Research*, vol. 11, pp. 391–427, 1999.

[9] N. Roy and S. Thrun, “Coastal navigation with mobile robots,” in *Advances in Neural Processing Systems 12 (NIPS)*, S. A. Solla, T. K. Leen, and K. R. Müller, Eds. Denver, CO: MIT Press, 1999, pp. 1043–1049.

[10] E. W. Dijkstra, “A note on two problems in connexion with graphs,” in *Numerische Mathematik*. Mathematisch Centrum, Amsterdam, The Netherlands, 1959, vol. 1, pp. 269–271.

[11] R. Bohlin and L. E. Kavraki, “Path planning using lazy prm,” in *Proceedings of the IEEE International Conference on Robotics and Automation*. San Fransisco, CA: IEEE Press, April 2000, pp. 521–528.

[12] S. R. Lindemann and S. M. LaValle, “Current issues in sampling-based motion planning,” in *Proceedings Eighth International Symposium on Robotics Research*, 2004.

[13] T. Lozano-Perez, “Spatial planning: A configuration space approach,” *IEEE Transactions on Computers*, vol. C-32, no. 2, pp. 108–120, February 1983.

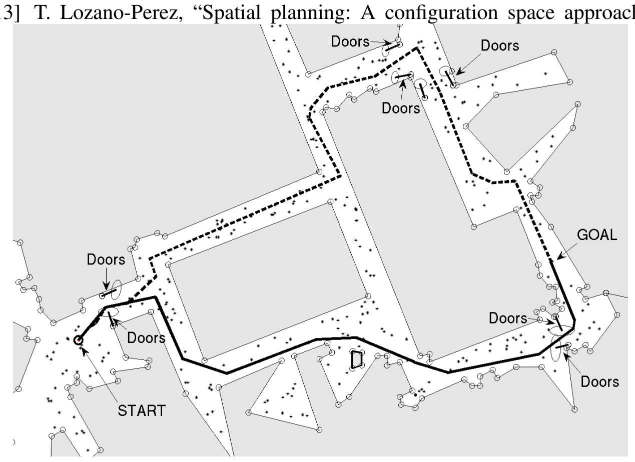


Fig. 8. Map of the 3rd floor MIT Stata Center, the minimum-collision-cost planner (dashed) selects the longer trajectory for the B21 robot to reach the goal based on the uncertain position of the doors along its route, while standard A* planner (solid) selects the shortest path trajectory.

[14] M. Akinc, K. E. Bekris, B. Y. Chen, A. M. Ladd, E. Plaku, and L. E. Kavraki, “Probabilistic roadmaps of trees for parallel computation of multiple query roadmaps,” in *Proceedings of the International Symposium on Robotics Research (ISRR)*, Sienna, Italy, 2003.

[15] D. Hsu, J. Latombe, and H. Kurniawati, “On the probabilistic foundations of probabilistic roadmap planning,” in *Proc. Int. Symp. on Robotics Research*, 2005.

[16] Y. K. Hwang and N. Ahuja, “Gross motion planning - a survey,” *ACM Comput. Surv.*, vol. 24, no. 3, pp. 219–291, 1992.

[17] N. Amato and Y. Wu, “A randomized roadmap method for path and manipulation planning,” 1996.

[18] D. Hsu, L. E. Kavraki, J.-C. Latombe, R. Motwani, and S. Sorkin, “On finding narrow passages with probabilistic roadmap planners,” pp. 141–153, 1998.

[19] V. Boor, M. H. Overmars, and A. F. van der Stappen, “The gaussian sampling strategy for probabilistic roadmap planners,” in *Proceedings of the 1999 IEEE International Conference on Robotics and Automation*, vol. 1, 1999, pp. 1018–1023.

[20] D. Hsu, T. Jiang, J. Reif, and Z. Sun, “The bridge test for sampling narrow passages with probabilistic roadmap planners,” in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2003, pp. 4420–4426.

[21] J.-C. Latombe, *Robot Motion Planning*. Dordrecht, The Netherlands: Kluwer, 1991, vol. SECS 0124.

[22] H. Moravec and A. Elfes, “High resolution maps from angle sonar,” in *Proc. IEEE Int. Conf. on Robotics and Automation*, 1985, pp. 116–121.

[23] S. A. Wilmarth, N. M. Amato, and P. F. Stiller, “Maprm: a probabilistic roadmap planner with sampling on the medial axis of the free space,” in *Proceedings of the 1999 IEEE ICRA*, vol. 2, 1999, pp. 1024 – 1031.

[24] P. Leven and S. Hutchinson, “Toward real-time path planning in changing environments,” in *Proceedings of the Workshop on Algorithmic Foundations of Robotics (WAFR)*, 2000.

[25] J. P. van den Berg and M. H. Overmars, “Roadmap-based motion planning in dynamic environments,” in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*. IEEE/RSJ, 2004, pp. 1598–1605.

[26] S. M. LaValle, *Planning Algorithms*. [Online], 2004, available at http://msl.cs.uiuc.edu/planning/.

[27] S. M. LaValle, M. S. Branicky, and S. R. Lindemann, “On the relationship between classical grid search and probabilistic roadmaps,” *International Journal of Robotics Research*, 2003.

[28] S. M. LaValle and J. J. Kuffner, “Rapidly-exploring random trees: Progress and prospects,” in *Workshop on the Algorithmic Foundations of Robotics*, 2000.

[29] J. P. van den Berg and M. H. Overmars, “Using workspace information as a guide to non-uniform sampling in probabilistic roadmap planners,” in *Proceedings of the International Conference on Robotics and Automation 2004*. IEEE, 2004, pp. 453–460.