Real-Time Informed Path Sampling for Motion Planning Search

Ross A. Knepper and Matthew T. Mason

Abstract

Mobile robot motions often originate from an uninformed path sampling process such as random or low-dispersion sampling. We demonstrate an alternative approach to path sampling that closes the loop on the expensive collision-testing process. Although all necessary information for collision-testing a path is known to the planner, that information is typically stored in a relatively unavailable form in a costmap or obstacle map. By summarizing the most salient data in a more accessible form, our process delivers a denser sampling of the free path space per unit time than open-loop sampling techniques. We obtain this result by probabilistically modeling—in real time and with minimal information—the locations of obstacles and free space, based on collision-test results. We present CALM, the combined adaptive locality model, along with an algorithm to bias path sampling based on the model's predictions. We provide experimental results in simulation for motion planning on mobile robots, demonstrating up to a 330% increase in paths surviving collision test.

1 Introduction

The motion planning problem is to find a path or trajectory that guides the robot from a given start state to a given goal state while obeying constraints and avoiding obstacles. The solution space is high dimensional, so motion planning algorithms typically decompose the problem by searching for a sequence of shorter, local paths, which solve the original motion planning problem when concatenated.

Each local path comprising this concatenated solution must obey motion constraints and avoid obstacles and hazards in the environment. Many alternate local paths may be considered for each component, so planners select a combination of paths that optimizes some objective function. In order to generate such a set of feasible (i.e. constraint-satisfying) and collision-free paths, the planner must generate a much larger set of candidate paths, each one of which must be verified against motion constraints and collision-tested prior to consideration. Motion planners generate this large collection of paths by sampling—most often at random or deterministically from a low-dispersion sequence.

All the information needed to find collision-free path samples exists within the costmap (sometimes called an obstacle map), but the expensive collision-test process prevents that information from being readily available to the planner. A negative collision-test result (i.e. no collision) is retained for future consideration, but a positive collision-test result is typically thrown away because the path is not viable for execution. Such planners may later waste time sampling and testing similar paths that collide with the same obstacle. The true impact of this effect is illustrated in Figure 1, where 94% of tested paths are in collision. Many of these tests could be avoided with a smarter path sampling strategy.

This policy of discarding information about colliding paths highlights a major inefficiency, which especially impacts real-time planning. Every detected collision provides a known obstacle location.

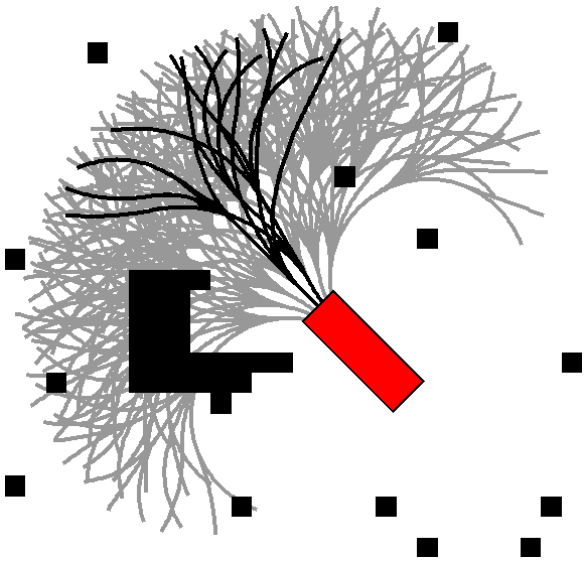


Figure 1: Within a typical set of paths sampled by the planner within each replan cycle, only a small fraction typically survive collision with obstacles (black blocks). Paths emanate from the robot (diagonal rectangle). Those deemed to be in collision are shown in gray, while surviving paths are in black. Less than 6% of paths in this example are collision-free.

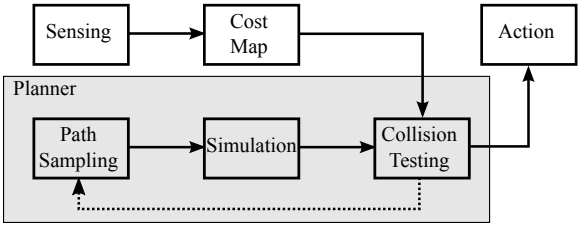


Figure 2: Typical data flow within a robot closes the loop around the sense-plan-act cycle, but the planner itself runs open-loop. We close the planning loop, informing path sampling with the results of collision-testing earlier paths.

This observation may not seem significant at first, as all detected collisions represent known obstacles in the costmap. However, not all such obstacles are equally relevant to a given local planning problem, and so we can benefit by storing relevant costmap obstacles in a form more immediately available to the planner. We argue that the planner may derive increased performance by feeding back the set of collision points, known from prior collision tests, to the path sampling process, as in Figure 2.

Despite the opportunity for such collision-test feedback, many modern sampling-based planners do not incorporate such feedback into the sampling process, be it state-based or action-based sampling. The greatest opportunity is presented by planners that sample a set of paths within a neighborhood. RRT-Blossom (Kalisiak and van de Panne, 2006) and RRT* (Karaman and Frazzoli, 2011) both possess this property, although here we focus on the local planner component of the model-based hierarchical planner (Knepper and Mason, 2008). Each of these algorithms repeatedly samples and collision-tests many outgoing paths from a single point, treating each as if its collision-test outcome is completely independent of the others.

This paper expands upon results presented recently (Knepper and Mason, 2011) and leverages our work exploring local path equivalence (Knepper et al., 2012). Combined, these efforts seek a

deeper understanding of the relationships between path sampling and collision-testing. We offer a real-time, data-driven technique to model the salient portions of the distribution of obstacles in the task space. In utilizing this technique to bias path sampling, we may improve both the planning success rate and the yield of collision-free paths generated by the sampling process, thus making planning both smarter and faster. Such increased speed allows the planner to increase its cycle rate, making the robot more reactive to changes in its environment.

1.1 Path Sampling and Path Parametrization

The general path sampling problem is to supply a sequence of distinct paths $\{p_1, p_2, \dots\} = \mathcal{P} \subset \mathcal{X}$, the continuum space of paths. Often, these paths are not parametrized directly by their geometry but instead are described by their means of generation. For instance, some planners consider only straight-line paths. Given a current robot state $q_0 \in Q$, the configuration space (C-Space), a straight-line path is uniquely specified by connecting q_0 to an arbitrary sampled state $q_f \in Q$. In such planners, it is expected that the robot is able to execute arbitrary paths, and so the boundary value problem is easy to solve because it is under-constrained.

Definition 1. *Given start and end states, the* ***boundary value problem*** *(BVP) is to find any feasible path between the start and goal states (i.e. the local planning problem). A variant of this problem is to find the shortest such path.* □

Some classes of robotic systems possess velocity constraints that limit the direction in which they may move instantaneously. The most well known example of these nonholonomic constraints involves parallel parking a car. In such highly constrained, underactuated systems, the set of feasible paths $\mathcal{F}$ is much smaller than the space of all paths, $\mathcal{X}$. Thus, an arbitrary path sample drawn from $\mathcal{X}$ is unlikely to be in the feasible set $\mathcal{F}$. In such cases, the BVP is difficult to solve.

For constrained systems, we may avoid solving the BVP by instead sampling in U, the space of actions. Suppose we have a "black box" model of the robot's response to a control input, which is a mapping $M : U \to \mathcal{F}$. Sampling in the control space offers several advantages: 1) all sampled paths trivially obey motion constraints; and 2) we may precompute a diverse set of such paths. For a mobile robot, these paths are independent of initial position and heading, depending only on their derivatives (at this stage, we ignore external interference such as wind or wheel slip). Therefore, a relatively small lookup table suffices to describe an expressive set of robot motions. For a manipulator arm, a diverse set of paths can be precomputed as a function of manipulability, somewhat resembling the work of Leven and Hutchinson (2003).

1.2 Outline

In Section 2, we survey the literature in biased and adaptive sampling strategies. We introduce our approach to informed path sampling in Section 3. Then, following some fundamental concepts in Section 4, we delve into a probabilistic model for obstacle location, called locality, in Section 5. Section 6 culminates the combined adaptive locality model (CALM) by incorporating prior knowledge of negative space. Then, in Section 7, we introduce several approaches to the decision problem of selecting paths to test. We report some experimental results in Section 8, and conclude with a discussion of the results in Section 9.

2 Prior Work

The motion planning community has invested considerable effort in the topic of non-uniform and adaptive sampling. There has been a particular focus on exploring this topic as it pertains to probabilistic roadmaps (PRMs), which directly sample states rather than paths. The basic PRM uses rejection sampling to generate a sequence of uniformly-distributed configurations within the free C-Space. Hsu et al. (2006) motivate the need for non-uniform sampling and provide a survey of recent work in non-uniform sampling for PRMs. We update that survey to include recent work and a broader variety of planners.

2.1 Sampling Strategies with Fixed Bias

The primary impetus for biased sampling is the narrow passage problem. That is, some environments' arrangement of obstacles is such that a only a precise motion among the obstacles will avoid collision, and the set of valid C-Space samples that would lead to generating such a path is of small—but non-zero—measure compared to the full C-Space. When a planner must find a path through a narrow passage to solve a problem, unbiased random samplers typically require a great deal of time, memory, and computation to create a connected PRM.

One approach to finding narrow passages observes that they comprise points that are in the neighborhood of an obstacle. By employing a fixed strategy to bias configuration sampling towards obstacle neighborhoods, these approaches sample narrow passages with increased probability. The canonical PRM (Kavraki et al., 1996) accomplishes this task by performing a random walk from nodes that have seen many connection failures due to collision with obstacles. Similarly, in a precursor of the PRM, Horsch et al. (1994) present a roadmap-building planner which tries to unify separate connected components by repeatedly casting rays that bounce off of C-Space obstacles in random directions.

Still other planners utilize a combination of samples in parts of the C-Space that are in obstacles (C_{obs}) and collision-free (C_{free}) to bias sampling near their boundary. Amato and Wu (1996); Amato et al. (1998) propose OBPRM, which samples on the boundary between C_{obs} and C_{free}. Boor et al. (1999) describe the Gaussian sampling approach, in which samples are made in pairs separated by a distance drawn from a Gaussian distribution. A sample is retained when exactly one sample of the pair is collision-free, thus causing most sampled configurations to be close to obstacles. Hsu et al. (2003) introduce the bridge test, which samples three points in a straight line in C-Space. The middle sample is retained only when it is in C_{free} and the two endpoint samples are in C_{obs}. Although this approach performs three times the point collision tests, it rejects points located in expansive spaces, thus saving significantly in the more expensive path collision test. Collins et al. (2003) construct a hierarchical PRM by recursively refining sampling density in C-Space balls that contain both obstacle and free space samples. Another method of sampling near obstacles (Aarno et al., 2004) uses values of a potential function to bias PRM sampling towards regions that neighbor obstacles, including narrow passages. Denny and Amato (2011) propose Toggle-PRM, which simultaneously constructs two PRMs located in C_{free} and C_{obs}. Failed connection attempts in either graph generate "witness" points in the other. Such witness points occupy narrow passages with an elevated probability.

A variety of motion planners capitalize on properties of the Voronoi diagram or Voronoi graph. These structures are defined as the locus of points that are two-way equidistant and d-way equidistant (respectively) to obstacles, where d is the dimension of the space. By biasing sampling toward these structures, a planner takes advantage of two of their properties. First, the Voronoi diagram and graph extend between every pair of neighboring obstacles, and so they can be used to find any

passage, including a narrow one. Second, both structures describe routes through space that reside maximally far from obstacles for optimal safety. Wilmarth et al. (1999) sample on the Voronoi diagram by first sampling uniformly in the space—including within obstacles—then projecting points onto the Voronoi diagram (also called the medial axis). Holleman and Kavraki (2000) and Yang and Brock (2004) offer similar algorithms that operate within the task space. This design decision makes the algorithm more computationally efficient at the cost of being less effective for highly-articulated robots.

Garber and Lin (2004) leverage existing physics-based constrained optimization solving software to move towards the Voronoi Diagram, which can then be traversed by a potential field planner. Rickert et al. (2008) espouse a similar planning approach in which overlapping task space bubbles fill the free space. A potential field planner guides the robot through the bubbles. The authors present exploration strategies for cases in which the potential field planner fails.

Interestingly, some planners take the opposite approach to sampling among obstacles by completely ignoring obstacle locations during the preplanning phase. In single-query planning, most of the path segments generated will not ultimately be executed. A greedy approach, called a lazy PRM method (Bohlin and Kavraki, 2000; Nielsen and Kavraki, 2000; Song et al., 2001), holds the potential for increased performance in uncluttered environments. The other side of this tradeoff is that in highly cluttered environments, the planner may suffer decreased performance.

Leven and Hutchinson (2002) introduce a variety of approaches to improve the likelihood that a lazy PRM will yield a collision-free path. The authors also contribute a pair of ideas to the biased sampling discussion. First, they bias sampling density for a kinematic chain toward C-Space regions of low manipulability because the arm is less dexterous in those areas. Second, they increase sampling near joint limits, which are no different than obstacles in their potential to create narrow passages.

In the late 1990s, several techniques based on geometrically deforming the obstacles were proposed. Baginski (1996) deforms links in a kinematic chain until a path is collision-free, then evolves the C-Space trajectory as the link lengths are restored. Ferbach and Barraquand (1997) formulate a series of increasingly relaxed constraints involved in a manipulation problem. By solving the least-constrained instance first and then refining the plan until the fully-constrained version is solved. Hsu et al. (1998) describe a similar notion of dilating the freespace in order to improve the probability of sampling free configurations. They likewise relax the dilation until a path is found in the original configuration space.

Some techniques perform a task space cellular decomposition in order to bias sampling towards certain cells based on their relationship to neighboring obstacles and freespace. Kurniawati and Hsu (2004) propose task space importance sampling, which computes cells in an exact cellular decomposition of the task space formed by a Delaunay triangulation (or tetrahedralization in 3D). The chance of sampling varies inversely with height of triangles where base forms part of task space boundary. This algorithm leads to a higher probability of sampling in narrow passages. van den Berg and Overmars (2005) perform an approximate cellular decomposition on the task space using an octree. They perform watershed labeling of the cells, a technique borrowed from computer vision. The boundary between basins of attraction (pinch-points in the task space) is specially labeled to receive higher probability of sampling within it. Thus, narrow passages are sampled with elevated probability.

Nissoux et al. (1999) construct the visibility PRM, which suppresses samples whose visibility region is not substantially different from some "guard" sample. New samples are retained when they have visibility to zero or two distinct connected components. By suppressing samples that do not aid the connectivity of the PRM, performance improves due to the avoidance of redundant structure.

Finally, Thomas et al. (2007) observe that many of the above biased sampling schemes can be compounded. They show that through a combination of biases, PRM planners can realize performance superior to any individual biased sampling method.

2.2 Adaptive Sampling Strategies

In recent years, the non-uniform sampling field has largely moved towards more sophisticated, adaptive strategies. These strategies typically employ a parametrized model to adjust sampling bias in response to detected collisions.

For instance, Jaillet et al. (2005) restrict sampling to size-varying balls around nodes in an RRT to avoid testing paths that would go through obstacles. The same authors further improve their algorithm to adaptively vary the size of balls (thus the probability of expanding each node) according to past success rates at node expansions (Yershova et al., 2005).

Zucker et al. (2008) apply statistical techniques to learn a feature weighting vector to describe important attributes of common environments. After learning weights, the features bias the sampling distribution of a bidirectional RRT. Features are defined as functions of the task space for computational efficiency.

Missiuro and Roy (2006) incorporate uncertainty in modeling of robot state and obstacle locations. They maintain a probabilistic obstacle map, representing vertices with Gaussian distributions. When sampling to construct a PRM, the probability of retaining a point sampled from a uniform random distribution is proportional to its probability of survival under the obstacle distributions. Routes within the roadmap are selected to minimize cost, which is a function of both length and collision probability.

Blackmore et al. (2011) formulate planning under uncertainty as a linear/Gaussian system and solve it as a Disjunctive Linear Program. Obstacles are polyhedral. They optimize an objective function subject to the constraint that the solution path's collision probability possesses a specified upper bound.

Another recent adaptive approach is to construct a meta-planner with several tools at its disposal; such planners employ multiple sampling strategies (Hsu et al., 2005) or multiple randomized roadmap planners (Morales et al., 2004), based on a prediction of which approach is most effective in a given setting.

2.3 Sampling to Maximize Information Gain

Another aspect of this paper involves careful path sampling to maximize information gain at each step. Many others have addressed this area, again with particular attention to roadmap methods.

Yu and Gupta (2000) perform sensor-based planning in which a PRM is incrementally constructed based on the robot's partial observations of obstacles. Exploratory motions are selected by maximizing information gain.

Burns and Brock (2003) describe an entropy-guided approach to the selection of configuration samples used to unify distinct connected components of the PRM graph. They formulate entropy based on the connectedness of states in the roadmap. The maximum entropy configuration is one that, if sampled, maximally decreases the total entropy in the graph by unifying large connected components. The algorithm strives to achieve zero entropy by assembling a fully connected roadmap.

In later work, Burns and Brock (2005c) augment this approach with the notion of utility. A sample's utility incorporates both the amount of information gained about obstacle locations as well as the sample's contribution for solving the actual planning problem. Thus, utility-guided

sampling selects at each iteration the configuration expected to solve the planning problem most efficiently, given what is not yet known about obstacle locations. We take a similar approach that is balanced between development of an effective path plan and refinement of our obstacle model.

One important feature of our work is the use of information from all collision tests, including positive results, to minimize entropy (that is, uncertainty) in a model approximating obstacle locations. Burns and Brock address this topic as well. They describe an adaptive model of obstacle locations in C-space based on previous collision-test results. Their model utilizes locally weighted regression to sample states (Burns and Brock, 2005a) or paths (Burns and Brock, 2005b) that maximally reduce model uncertainty. We likewise develop a probabilistic model of obstacle location, although ours inhabits the task space. Since we circumvent the C-Space, our algorithm's efficiency is better suited to real-time applications.

Burns and Brock subsequently observe that model refinement is not an end in itself, but merely a means to the end of finding collision-free paths (Burns and Brock, 2005c). We proceed from this observation to consider what level of refinement is appropriate, in the context of constrained paths, based on the maximum width of corridor we are willing to miss discovering.

Kobilarov (2011) samples trajectories adaptively by importance sampling using the cross-entropy method. This method solves a global optimization problem by alternately performing biased random sampling and model parameter updates. Path sample distributions are computed based on a Gaussian mixture model representation, which is then used to bias future samples. The model asymptotically converges toward a delta function describing the optimal path to the goal.

2.4 Our Contributions

We build on the state of the art in several ways. We leverage the geometry of paths and obstacles in order to create a locality model that efficiently estimates the distribution of obstacle- and free-space. Via extrapolation, it requires few samples. Rather than exhaustively searching the space, our objective is to quickly sample and select an appropriate path for execution. Even unbiased optimal sampling in path space is not yet well understood (Knepper and Mason, 2009), and the question of how to correctly bias path sampling is unexplored.

In addition to a locality model, we contribute a path sampling algorithm informed by such a model that both improves the model quality and exploits the model to produce a diverse selection of appropriate paths for execution. The algorithm strikes a principled balance between the goals of exploration and exploitation.

3 Informed Path Sampling Approach

In closing the loop on path sampling, we must feed back knowledge of obstacles reachable by the robot (in the form of collision-test results) into the sample space of paths, be it X or U, so that we can suppress from the sampled path sequence future paths intersecting those regions of the task space. In Sections 3 and 4 we provide algorithmic and probabilistic foundations for our approach. Subsequently, we introduce locality—an approximate probabilistic model of obstacle distributions in the task space. We consider a series of increasingly sophisticated models of locality that retain the trait of real-time computability while helpfully biasing sampling away from obstacles.

Obstacles reside in the task space, W ($\mathbb{R}^2$ or $\mathbb{R}^3$). We describe two set-valued functions,

$$ws\colon Q \to \text{collection of subsets of } X$$
$$cs\colon X \to \text{collection of subsets of } Q.$$

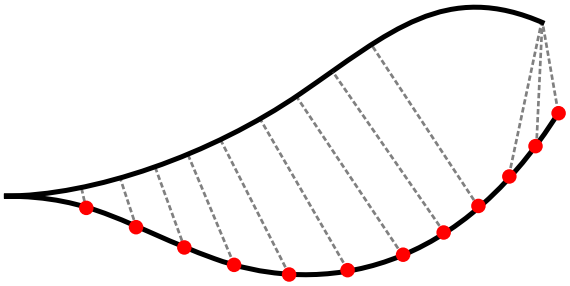


Figure 3: Given a path set of N paths, each discretized into M points, the proximity look-up table (PLUT) stores for each ordered pair of paths a list of shortest distances to each discrete point on the second path. Thus, there are a total of MN^2 unique PLUT entries. Taking any point as a detected collision, the PLUT reveals the closest approach of each other path.

The function $ws(q)$ returns a set of task space points $x \in X$ that the robot occupies while in configuration q (the Minkowski sum of the robot shape with a point). $cs(x)$ returns the set of robot configurations $q \in Q$ in which the robot intersects the task space point x. Supposing that x resides inside an obstacle, these functions enable us to reason about configurations the robot must avoid.

A collision can be described as an ordered pair $c = (p, s)$, with $p \in \mathcal{F}$ and $s \in I = [0, 1]$, a time/distance parameter describing where on the path the collision occurred. A path is a mapping $p : I \to Q$. Thus, c maps directly into a state $q \in Q$, identifying the location of an obstacle. However, this collision state is special because it is known to be reachable by an action $u \in U$. In fact, q is probably reachable by a continuum of other actions, of which we can easily precompute a sampled subset for each possible collision point.

A collision state q also has a correspondence to some known task space point. Often, collision-test routines are able to identify the precise location where a collision occurred. Knowing that task space point $x \in ws(q)$ is part of an obstacle, we may eliminate from our sampled sequence all paths passing through the set $cs(x)$. To determine which paths to eliminate, we must store the list of actions by which they are parametrized.

In the event that the collision tester is unable to supply a task space collision point, a projection of q onto the task space will suffice. For a mobile robot, that point may correspond to the center of the vehicle while in a colliding state. For a kinematic chain, it may be the point along the central axis that comes closest to the obstacle.

In the offline precomputation phase of the planner, possible task space collision points are established in the robot's local reference frame so that their relationship to untested paths may bias future path samples. We construct, a priori, a list of correspondences between possible action samples to accelerate this process. In order to identify the set of paths passing unacceptably close to an obstacle point x, we precompute a proximity look-up table (PLUT), as shown in Figure 3. Suppose our precomputed path set $\mathcal{P}$ contains N paths, each discretized into M points. The PLUT stores, for every ordered pair of paths (p_i, p_j) in $\mathcal{P}$, the shortest distance to the kth discretized point on p_j:

$$\text{PLUT}(p_i, p_j, k) = \min_{x \in p_i} d\left(x, p_j\left(\frac{k}{M}\right)\right), \tag{1}$$

where $d(x_1, x_2)$ gives the Euclidean distance between two points.

In this work, the distance computation is simplified by the assumption that the principal axes of the robot have approximately equal diameter. This assumption allows the treatment of the robot as a disc or ball. In such case, the PLUT gives the Euclidean distance from the given collision point

to the center of the ball. For other rigid body shapes, we may instead need to find the projection of the collision point onto the major axis of the robot. The method even extends to a manipulator arm by projecting onto the axis of the arm. Such cases complicate only the offline computation; the online algorithm runs precisely the same.

At runtime, our approach to path sampling feedback in highly constrained systems keeps the feedback entirely within the action space by which such paths are parametrized. We first collision-test some paths drawn from a low-dispersion sequence (Green and Kelly, 2007). After finding the first collision point, its location biases future action samples. Given a collision $c = (p_j, s)$, we would like to find out if another path p_i would collide with the same obstacle. We simply query the PLUT as

$$\mathrm{PLUT}(p_i, p_j, sM) < r_r, \tag{2}$$

where r_r is the radius of the robot (or an inscribed circle of the robot). When this condition holds, the collision test would fail. With this knowledge, we may eliminate the path without a test, and instead spend the CPU time considering other paths. Thus, we have directly eliminated path p_i based on knowledge of colliding path p_j without computing any task space or configuration space geometry at runtime.

However, we may go beyond short-circuiting the collision test to estimating the probability distribution on obstacle locations using the *principle of locality*, which states that points inside an obstacle tend to occur near other points inside an obstacle. We propose a series of models of locality and two path sampling problems, which we address using these models, which we use to select paths likely to be collision-free. The key to success of this approach is that the final evaluation be less costly than the collision tests it replaces.

4 Probabilistic Foundations

We develop a series of probabilistic models that enable us to rapidly select paths for collision test that maximize one of two properties. First, in order to find valid robot motions, we must sample a selection of collision-free paths for execution. Second, we wish to sample diversely within the free space of paths, including in proximity to obstacles. The precision with which we know the obstacle/free-space boundaries directly relates to the size of narrow corridor we expect to find.

The task space comprises a set of points divided into two categories: obstacle and free. The function

$$\mathrm{obs}\colon W \to \beta, \tag{3}$$

where $\beta = \{\mathrm{true}, \mathrm{false}\}$, reveals the outcome that a particular task space point x is either inside (true) or outside of an obstacle. Building on such outcomes, we then describe an event of interest. A path collision test takes the form

$$\mathrm{collides}\colon \mathcal{P} \to \beta, \tag{4}$$

which returns the disjunction of $\mathrm{obs}(x)$ for all x within the swept volume (or *swath*) of the path. A result of true indicates that this path intersects an obstacle. Note that it is not important here precisely how *collides* is implemented, although it typically possesses the characteristic of being costly to invoke. For details on our implementation of *collides* for experimental purposes, see Section 8.1.

Using the above concepts as a basis, we pose two related problems:

1. **Pure Exploitation**. We are given a set of task space points inside obstacles, $C = \{x_1, \ldots x_m\}$ and a set of untested paths $\mathcal{P}_{unknown} = \{p_1, \ldots p_n\}$. Knowing only a finite subset of the continuum of obstacle-points, find the path that minimizes the probability of collision:

$$p_{next} = \underset{p_i \in \mathcal{P}}{\text{argmin}} \Pr(\text{collides}(p_i, C)). \tag{5}$$

2. **Pure Exploration**. Suppose we have a model of uncertainty $U(\mathcal{P}_{safe}, C)$ over the collision status of a set of untested paths, $\mathcal{P}_{unknown}$, in terms of a set of tested paths and known obstacle-points. Find the path $p_{next} \in \mathcal{P}_{unknown}$ giving the greatest reduction in expected uncertainty:

$$\begin{aligned} U_{exp}(p_i) = &U\left(\mathcal{P}_{safe} \cup \{p_i\}, C\right) \Pr\left(\neg \,\text{collides}(p_i, C)\right) \\ &+ U\left(\mathcal{P}_{safe}, C \cup \{c_i\}\right) \Pr\left(\text{collides}(p_i, C)\right) \end{aligned} \tag{6}$$

$$p_{next} = \underset{p_i \in \mathcal{P}_{unknown}}{\text{argmax}} \; U(\mathcal{P}_{safe}, C) - U_{exp}(p_i). \tag{7}$$

These two problems are effectively to maximize the characteristics encapsulated in the utility functions of Burns and Brock (2005c) (see Section 2.3). Pure Exploration and Pure Exploitation are both inherently forms of search, but they have different goals. Pure Exploration strives to fully understand the search space, whereas Pure Exploitation seeks to utilize all currently available information to greedily concentrate search.

5 Locality

Thus far, we have demonstrated how a single failed collision test may serve to eliminate an entire set of untested paths from consideration because they pass through the same obstacle point. We may extend this approach one step further using the principle of locality.

Definition 2. *The **principle of locality** states that, given a robot state c in collision with an obstacle, there exists a neighborhood ball of obstacle points containing c.* □

Two contributing factors combine to produce the locality effect. First, the non-zero volume of the robot means that even a point obstacle results in a set of robot states $cs(t)$ in collision with that point. The second factor is that real-world obstacles occupy some volume in space. That is, both obstacles and freespace tend to be "thick."

Given a known collision point, we employ the principle of locality to define a function expressing the probability that a new path under test is in collision with the same obstacle. A locality model takes the following general form:

$$\text{loc}(p_i \mid C) = \Pr(\text{collides}(p_i) \mid C) \tag{8}$$

Here, C may be a single point collision outcome or a set of collisions. If omitted, it is assumed to be the set of all known collisions.

This function depends on many factors, including the size and shape of the robot as well as the distribution on size and shape of obstacles in the environment. The most important parameter, however, is the distance between the new path and the known collision site. Thus, we may establish a rapidly computable first-order locality model in which we abstract away the size and shape of obstacles using a single distribution on radius, as in Figure 4.

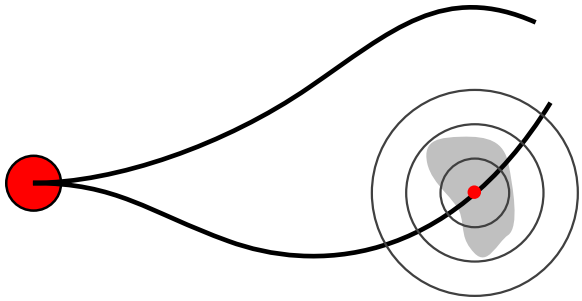


Figure 4: The robot (disc at left) considers two paths. First, the bottom path fails its collision test. The locality model does not know the full extent of the obstacle (gray), but it can approximate the obstacle using a probability distribution (concentric circles) and can estimate the likelihood of the top path colliding.

We discuss several intermediate locality model formulations before coming to the final form. These intermediate steps serve three functions. First, after giving the exact locality model formulation, we then approximate it in a way that is efficient to compute online. Second, we make a probabilistic independence assumption in order to simplify attribution of multiple path collisions to multiple obstacles. Finally, we add an adaptive aspect that incorporates collision-test successes as well as failures. This adaptive aspect compensates for the conservatism introduced by the independence assumption, and it efficiently utilizes all available information.

5.1 General Locality Model

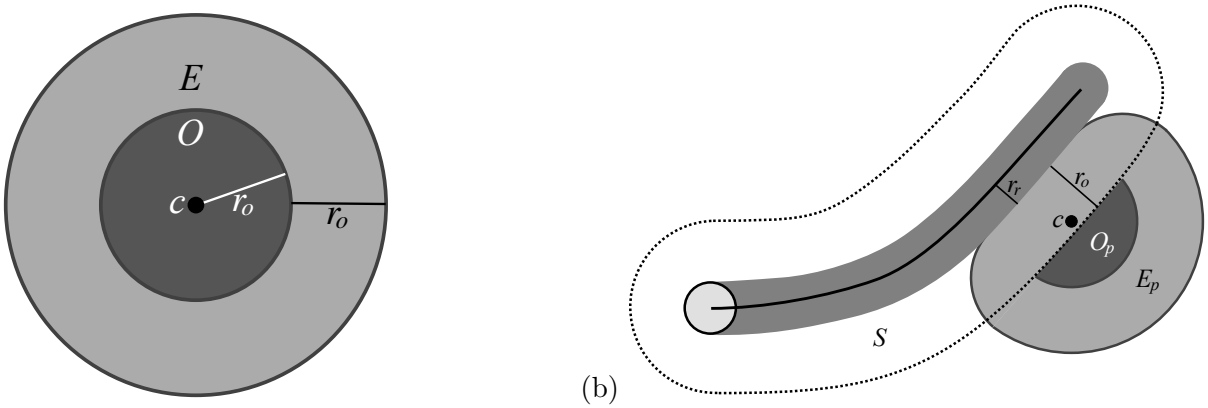
By explicitly modeling locality, we may reason about which paths are more or less likely to be in collision with any known obstacle, even with only partial information about its location. A path sampling algorithm informed by a locality model provides a path sequence ordered by likelihood of collision, given currently known collision sites. We propose here a general model of locality that can be expected to produce collision-free path samples with high probability.

In constructing a general locality model, we abstract away many parameters; we consider both the robot and obstacles to be balls (in $\mathbb{R}^2$ or $\mathbb{R}^3$), and the obstacles are assumed uniform in radius. We relax some of these assumptions later, in Section 5.4. For now, these restrictions permit us to simplify the model by removing bearing from consideration. Thus, the general model's prediction of future collisions is purely a function of *range* from the known collision site to the path. The fixed radii of both the obstacles (r_o) and the robot (r_r) result in the intuitive notion of locality—that its influence is over a limited range only.

The precise formulation of the general locality model, as depicted in Figure 5, is based on maintaining a probability distribution on possible locations of the obstacle centroid, given a known collision site. In this naive model, the location of the centroid is described by a uniform distribution over $B(r_o)$, a ball of radius r_o centered at the colliding position of the robot. A path p_i sweeps out a swath $S(p_i)$ of radius $r_o + r_r$. Any non-empty set $B(r_o) \cap S(p_i)$ represents some probability of collision. This general model then predicts that the probability of collision is

$$\mathrm{loc}_{general}(p_i \mid c) = \frac{|B(r_o) \cap S(p_i)|}{|B(r_o)|}. \tag{9}$$

Note here that although we are treating the robot as a disc, a slightly more complex computation allows the use of other shapes. Since this computation is performed offline, we can actually permit arbitrarily complex robot shapes, so long as we can compute its intersection with a disc-shaped obstacle as in (9).



(a) (b)

Figure 5: In the general locality model, obstacles are treated as discs of radius r_o. **(a)** Given a point c known to be in collision with an obstacle, the disc O of radius r_o represents possible locations of the centroid of the obstacle. The larger disc E of radius $2r_o$ comprises points possibly occupied by some part of the obstacle. **(b)** The probability that a new candidate robot path is collision-free equals the fraction of possible obstacle centroids outside a swath of radius $r_o + r_r$. The region O_p represents the set of possible obstacle centroids consistent with the collision-free hypothesis, while the region E_p depicts corresponding obstacle extents. The probability that the new path is safe is obtained by computing the measure of O_p as a fraction of O. Here, that probability is approximately 40%.

If we regard p_i as a straight line, then in 2D the probability of collision is the ratio of the area of a circular segment to the area of the whole circle, which is (Beyer, 1991):

$$f_{segment}(r) = \begin{cases} \frac{1}{\pi r_e^2}\left(r_e^2 \cos^{-1}\frac{r-r_e}{r_e} - (r-r_e)\sqrt{2r_e r - r^2}\right) & \text{if } 0 \le r \le 2r_e \\ 0 & \text{otherwise,} \end{cases} \tag{10}$$

where r is the range between the path and the collision point. We call r_e the range of effect, which we set equal to r_o here.

Definition 3. *The **range of effect** of a known collision point describes the radius around that point at which paths are regarded to be at elevated risk of collision with the known obstacle.* □

5.2 Simple Locality Model

We now propose an even simpler locality model, which closely approximates (10) but makes use of the existing PLUT. Instead of total path area, we consider only the point on the prospective path under test that most closely approaches the known collision point. This new locality model employs the raised cosine distribution:

$$f_{rcd}(r, r_e) = \begin{cases} \frac{1}{2r_e}\left[1 + \cos\left(\pi\frac{r}{2r_e}\right)\right] & \text{if } 0 \le r \le 2r_e \\ 0 & \text{otherwise.} \end{cases} \tag{11}$$

For a straight or gently curving path, this approximation is very good (Figure 6). Then, the probability that a new path p_i will collide with the same obstacle represented by the previous collision $c = (p_j, s)$ is simply

$$\text{loc}_{simple}(p_i \mid c) = f_{rcd}(\text{PLUT}(p_i, p_j, sM) - r_r, r_e(c)). \tag{12}$$

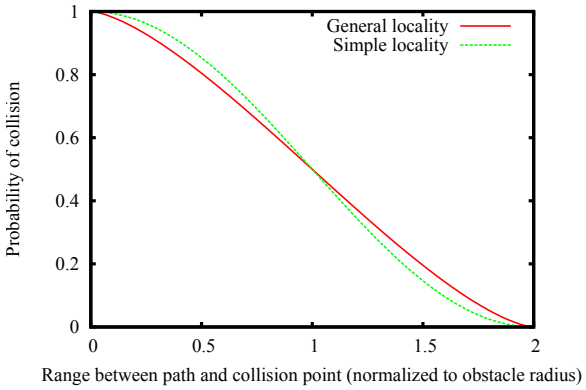


Figure 6: For a straight-line path, the the general locality model closely resembles the simple locality model. The latter is based on the raised cosine distribution applied to the range of closest approach.

Note that with the simple locality model we are no longer maintaining an explicit probability distribution on the location of an obstacle but instead a heuristic estimate of the risk of a single path relative to a single collision site.

5.3 Handling Multiple Collision Sites

Given a known collision site, both (9) and (12) provide a tool for selecting a candidate path to minimize the probability of collision. However, we have not yet addressed the issue of multiple known collision sites. Naturally, the principle of locality applies among obstacle points just as it does between an obstacle and a path,

$$\Pr(\text{obs}(c_1) \mid \text{obs}(c_2) = \mathit{true}) \geq \Pr(\text{obs}(c_1)). \tag{13}$$

In particular, the likelihood that two task space points have the same obstacle outcome correlates strongly with the distance between them, by virtue of describing the same obstacle. The estimate of collision likelihood for an untested path depends on what statistical independence assumptions we make among known collision points.

Figure 7 depicts a situation in which two collision sites appear to be correlated. However, many possible policies for estimating statistical independence among a set of collision points, such as clustering techniques, are complex to compute and implement.

In contrast, we may conservatively assume that all collisions are independent, in which case basic probability theory states that

$$\text{loc}(p) = \text{loc}(p \mid \{c_1, \ldots, c_n\}) = 1 - \prod_{i \in \{1,\ldots,n\}} (1 - \text{loc}(p \mid c_i)). \tag{14}$$

If some collision sites are actually part of the same obstacle, then we are overestimating the likelihood of collision for p. In the absence of any knowledge regarding correlation, however, the most conservative policy is the safest. In the next section, we explore an information theoretic approach to safely adjusting this pessimistic model.

5.4 Adaptive Locality Model

The locality models presented in Sections 5.1–5.2 incorporate only positive collision-test outcomes. Those static models conservatively estimate an obstacle distribution spread over a large but finite

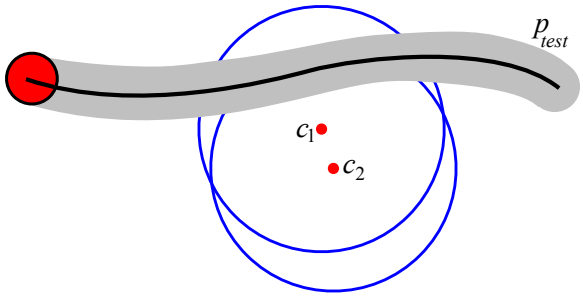


Figure 7: Two collision sites c_1 and c_2 are located in close proximity. Intuition suggests that c_2 should be ignored when computing the risk of collision of path p_{test}. Either the two sites belong to the same obstacle, or else the obstacle at c_2 is "blocked" by the obstacle at c_1.

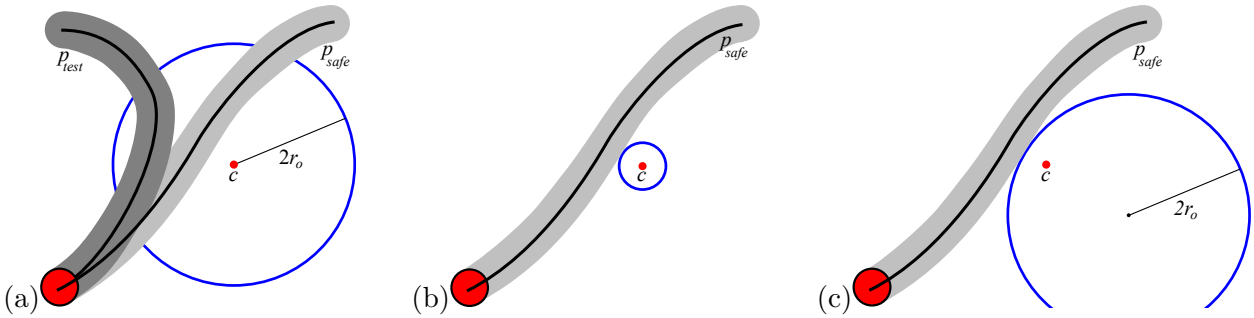


Figure 8: **(a)** Given a collision point c and neighboring collision-free path p_{safe}, the circle represents a distribution on obstacle locations, some of which are invalidated by p_{safe}. The more distant candidate path p_{test} is not at risk of collision with the obstacle represented by c.
(b) and **(c)** Two simple hypotheses on obstacle scale and position explain these two results. The distribution shown in Figure 5(b) is simpler to represent during online path sampling.

range of effect. We now construct an *adaptive* locality model capable of incorporating both positive and negative collision-test outcomes.

If we should happen to discover a safe path p_{safe} passing within collision site c's range of effect, then we may use this new information to refine the obstacle model of c. We adjust the locality function to act over a smaller range in the direction of p_{safe} in order to be consistent with observations. As Figure 8a shows, no path p_{test} that is separated from c by p_{safe} can possibly be at risk of collision with this obstacle. This adaptive model effectively relaxes the earlier, rigid assumptions on obstacle size and independence of collision sites. In modeling geometric relations between safe paths and obstacle points, we depart from prior work addressing locality.

Following an update to the model in the form of a safe path, all future probability estimates involving collision point c incorporate this new information. Although the independence assumption may initially make nearby paths like p_{test} appear riskier than they should (Figure 7), the adaptive model rapidly cancels out this effect after finding a safe path to shrink each collision point's range of effect. This approach, pairing an independence assumption with an adaptive model, is well-suited to real-time path sampling because it scales at worst linearly in the number of collisions detected. With clever organization of the collision points into a KD-tree or similar data structure, logarithmic complexity is achievable.

In addressing the problem of how to adaptively adjust obstacle distributions in reaction to a collision-free path, a variety of approaches present themselves. One possible approach is to shrink the range of effect for the obstacle at c, as in Figure 8b, which supposes that the obstacle is smaller

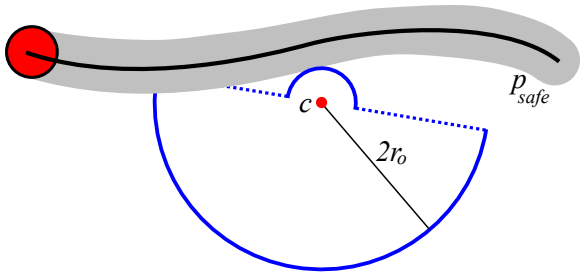


Figure 9: The range of effect on each side of collision site c is maintained separately. The left range began at $2r_o$, but it was reduced after successfully collision-testing path p_{safe}.

than initially thought. Another approach, to shift the entire distribution away from the safe path as in Figure 8c, assumes that the obstacle size was correctly estimated, but its position was off.

We adopt a compromise position. We prefer that the collision site remains the center of a distribution in order to keep range checks efficient via look-up table. However, we also prefer to avoid altering the range of effect of the opposing side, about which we have no new data. We therefore split the range of effect into several regions of influence ("sides") centered around each collision site. In 2D, we have left and right sides of the obstacle, as in Figure 9. In 3D, the division is topologically more arbitrary, although geometrically expedient to split the obstacle into four sides.

In splitting the locality model into several directions, we require a rule to consistently associate each path with a particular side of the collision point in order that similar paths will be associated to each other. The sides are defined relative to the pose of the robot before executing the path. The sides meet at the line $\mathbf{a}$, an axis running through the start pose and the collision point. We assign names to the sides describing their position relative to the robot's frame of reference. Sides are determined by

$$left = \text{sgn}(\mathbf{t} \times \mathbf{p} \cdot \mathbf{u}) \tag{15}$$

$$top = \text{sgn}(\mathbf{t} \times \mathbf{p} \cdot \mathbf{a} \times \mathbf{u}), \tag{16}$$

where $\mathbf{u}$ denotes the robot's up vector, $\mathbf{p}$ the projection of c onto the path, and $\mathbf{t}$ the tangent vector of the path at this point, as in Figure 10. These sides may be precomputed for each path. In 2D, it is particularly convenient to augment the PLUT with a sign indicating on which side of the path each possible collision point lies.

Figure 11 shows a family of paths on the left side of an obstacle. We deem each path equally likely to collide with the obstacle because they each approach equally near to the collision point, c. This assignment of paths to a single side of an obstacle places assumptions on the path's shape. We assume here that curvature is bounded and that paths are reasonably short. Our previous work (Knepper et al., 2012) thoroughly discusses these assumptions.

6 Modeling Negative Space

Thus far, we have focused entirely on constructing a model of the distribution of obstacles in space. In this section, we introduce a complementary model of negative space—that is, space that the robot can travel through. To provide data about negative space, we draw on recent results in the classification of local paths (Knepper et al., 2012). That paper discusses an algorithm for clustering paths that are equivalent in the sense that one can be continuously deformed to another

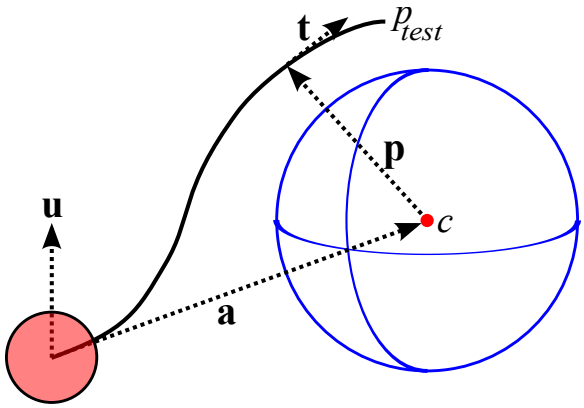


Figure 10: For a rigid body robot in three dimensions, such as an unmanned aerial vehicle or autonomous underwater vehicle, the adaptive locality model's range of effect is split into four sides. The robot's up vector, **u**, and the vector pointing toward the collision point, **a**, are used to define which of four sides the path p_{test} is on. As illustrated, the path is on the top-left side.

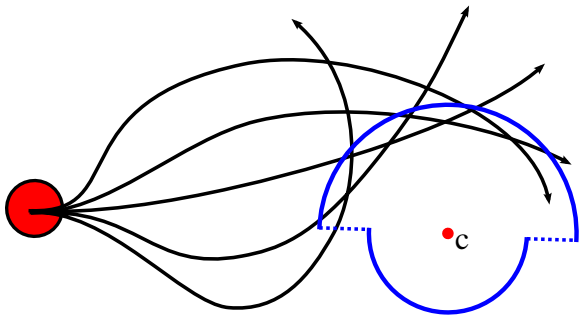


Figure 11: A family of paths, all of which pass to the left of the collision site, c. Despite the variety of shapes, each path intrudes equally into the left range of effect of c, and thus they would each reduce its left range of effect equally.

(akin to homotopy) while respecting kinodynamic and length constraints. The result is a small set of intuitive clusters of paths indicating the corridors that avoid obstacles. This classification comes at an extremely low overhead.

In order to better inform the planner's search for viable paths, we take the results of the previous replan cycle as input, under the assumption that a small time has elapsed between cycles, and consequently the results from the previous cycle remain informative (Knepper and Mason, 2009). Often, the corridors look approximately the same as they did a moment before. For each class of paths, we find an approximate center path as well as an approximate radius—each as measured by the Hausdorff metric, which gives the greatest separation between two paths. Specifically,

$$\mu_{\mathrm{H}}(p_i, p_j) = \max\left(\max_{x_i \in p_i} \min_{x_j \in p_j} d(x_i, x_j), \max_{x_j \in p_j} \min_{x_i \in p_i} d(x_i, x_j)\right), \tag{17}$$

where $d(x_i, x_j)$ gives the Euclidean distance between two points in the task space.

The corridor center and radius parameters can be computed inexpensively because the classification algorithm already computes and utilizes the Hausdorff metric in finding path equivalence. Let p_{center} be the center of a path equivalence class E from the last replan cycle with radius r_E. To find the center, we first find two paths representing approximate edges. Note that there is not necessarily one path in the set that forms a right or left envelope of the corridor. We start with

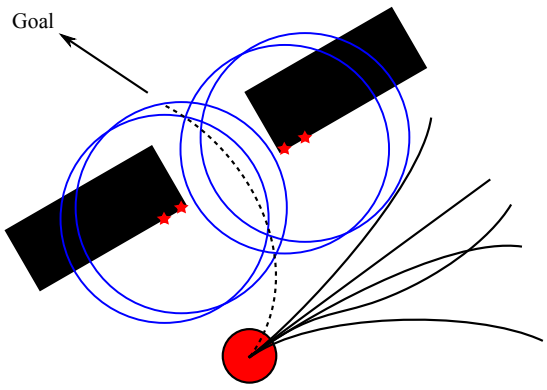


Figure 12: The simple locality model is sensitive to the sequence in which path samples are tested. Given a gap between two obstacles (black rectangles), a few collisions (stars with circles indicating range of effect) can make a desired path (dashed path) look extremely unsafe due to the conservative independence assumption. Instead, the model sometimes drives sampling towards areas of the space that are less useful (solid paths), leading to failure. The combined adaptive locality model (CALM) addresses this problem by introducing an elevated prior probability for path safety in the neighborhood of a group of paths that were safe in the previous replan cycle.

an arbitrary path, from which we find the most distant member of the set. We then find the most distant member from that, and these two paths approximate the boundaries. The path most nearly equidistant becomes p_{center}, and the mean distance to the two edges is called r_E.

The effect of these path classes on locality is to mitigate nearby failures. If the path sampler is unlucky while searching for a narrow corridor, then it may discover many collision points along the walls to either side of the corridor. Due to the conservative independence assumption, the model subsequently predicts an extremely low probability of any path within the corridor being safe, as depicted in Fig. 12.

In contrast, the combined locality model places a positive prior survival probability on paths in the neighborhood of the previously chosen corridor. Consider a known collision point, c under the simple, adaptive locality model described in Sections 5.2–5.4. Suppose we have an untested candidate path, p_i, that is within range of both c and p_{center}. We may combine positive and negative obstacle data as

$$\text{loc}_{combined}(p_i \mid c) = 1 - [1 - \text{loc}_{simple}(p_i \mid c)]\, f_{rcd}(\mu_{\text{H}}(p_i, p_{center}), r_E). \tag{18}$$

This model places a high likelihood on path survival if the path is far from the collision point *or* if it is near the center of the previous corridor. We call this model the combined adaptive locality model (CALM).

7 Path Selection

In this section, we discuss several approaches to path selection based on the tools provided by a locality model.

7.1 Pure Exploitation

Given a locality model, we have the means to address Problem 1, Pure Exploitation:

$$p_{next} = \underset{p_i \in \mathcal{P}}{\text{argmin}}\ \text{loc}(p_i). \tag{19}$$

Such a policy rapidly generates many paths with a high likelihood of successful collision test. This set of paths may not be the most useful for motion planning, though, as they tend to all closely resemble each other in shape. In order to encourage the path sampler to explore more widely, we may restrict the set $\mathcal{P}$ in (19) to a subset of, for example, 10% of the total path set. We construct a "bag of paths" that are drawn from the low dispersion sequence in order. Paths are sampled from the bag with replacement. This approach forces some amount of path diversity, although the pure exploitation approach still produces paths that tend to cluster closely together. As a side effect, the exploitation strategy produces few failed collision tests and so the locality model remains poorly learned.

7.2 Pure Exploration via Path Entropy

Next, we reap maximal advantage from the adaptive locality model in order to solve Problem 2, Pure Exploration. It is important to select paths for collision test that cause the model to rapidly converge to an accurate description of obstacles, while simultaneously minimizing failed collision tests. Given a set of collision sites, the path that best improves the model is that path with maximum entropy according to the current model parameters.

Definition 4. *An untested path's* ***entropy*** *(sometimes called Shannon entropy) refers to the expected amount of information about the path's safety that would be gained from collision-testing it. A path's entropy is*

$$H(\text{collides}(p)) = -\Pr(\text{collides}(p)) \log \Pr(\text{collides}(p)) - \Pr(\neg\,\text{collides}(p)) \log \Pr(\neg\,\text{collides}(p)). \quad \square \tag{20}$$

In order to maximize our understanding of the true distribution of obstacles with the fewest possible samples, we choose to sample the maximum entropy path:

$$p_{test} = \underset{p_i \in \mathcal{P}}{\text{argmax}}\; H(\text{collides}(p_i)). \tag{21}$$

Based on current knowledge, the maximum entropy path has maximal uncertainty with regard to its collision with obstacles; its probability of collision is nearest to 50%. Testing this particular path will therefore increase total knowledge more than any other. The result will be either a path that significantly reduces the range of effect for some known collision point(s) or a new collision point that is far from known collisions. In either case, model accuracy increases with maximal efficiency.

Shannon (1948) introduced entropy as a measure of the uncertainty associated with a random variable. He applied this notion to the prediction of English text transmitted through a noisy communication channel, noting that the information content of a given string is much lower than is suggested by the number of possible symbols, since some symbols are more likely than others.

The principle of maximum entropy was identified by Jaynes (1957), who addressed the problem of estimating a probability distribution given only partial knowledge. In that work, the maximum entropy belief is chosen for the unknown variables. Such a distribution is is maximally noncommittal on those variables and so it is the least biased hypothesis.

Maximum entropy has also been specifically applied to decision theory, as we employ it here. When forced to make a decision on the basis of partial information, Grünwald and Dawid (2004) show that the decision that maximizes entropy also minimizes the worst case expected outcome. In our application, the worst outcome corresponds to a collision-test result that would have been predicted from prior information. This worst-case outcome takes one of two forms: a path passing through a known collision site is certain to collide, whereas retesting a known-safe path is certain

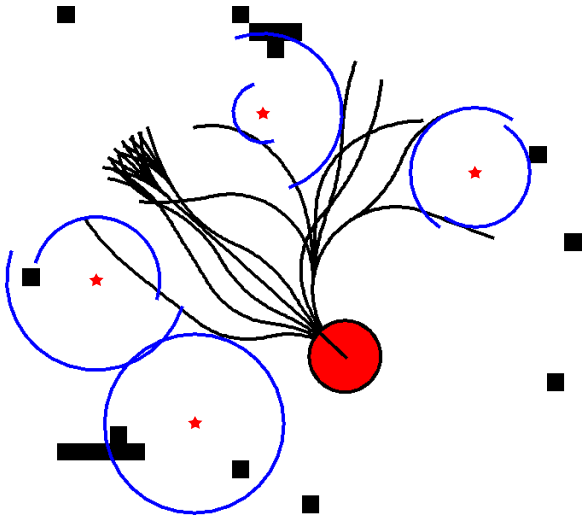


Figure 13: Visualization of locality model, showing: obstacles (black squares), mobile robot (notched disc) collision-free paths (emanating from robot), known collision sites (stars), and their ranges of effect (concentric semicircles). Note that the stars correspond to the nearest edge of each C-space obstacle—the point most relevant from the robot's current pose. Many obstacles are irrelevant to the current local plan and thus are neglected by the model. Some safe paths appear to intrude into the model's obstacle regions due to approximations in the PLUT, which assumes a perfectly kinematic motion model. Highly dynamic systems require a higher-dimensional PLUT indexed by start state.

to give a collision-free result. By instead choosing to test the maximum-entropy path, this Γ-minimax approach (Vidakovic, 2000) is capable of reasoning simultaneously about an entire family of probability distributions, called Γ—in our case, a range of theories about possible obstacle locations.

If the maximum entropy policy is pursued repeatedly, path selection proceeds to discover a sequence of safe paths and collision sites that are progressively nearer to each other, thus establishing precisely the boundaries separating the obstacles from free space. Knowing these boundaries may accelerate the process of sampling and testing paths densely within the free space.

7.3 Hybrid Path Sampling Strategy

We utilize a separate strategy to combine the characteristics of exploration and exploitation in a hybrid approach. Rickert et al. (2008) stress the importance of balancing between these two activities during planning. We do not believe that these two goals are inherently in conflict, as a naive combination of Problems 1 and 2 might suggest. Instead, we are solving a third search problem involving the joint maximization of the utility functions describing exploration and exploitation characteristics. In order to succeed at the ultimate motion planning goal, the path sampler must deliver a large, diverse set of collision-free paths. The most efficient approach thus samples paths that simultaneously explore and exploit to varying degrees.

In the absence of any uncertainty from our locality model (such as before the first collision site has been discovered), we sample from a low-dispersion sequence (Green and Kelly, 2007). Until one of these paths collides with an obstacle, our locality model is completely uninformed, and so we can only explore with maximum diversity.

After the locality model has been informed by one or more collisions, our approach thresholds based on the locality model's estimate of collision. Drawing once again from a bag of paths (Sec-

tion 7.1), the sampler immediately returns the first path found in the low-dispersion sequence that is at least 50% likely to survive. In the event that the bag contains no paths meeting this criterion, the path most likely to survive is then selected for collision test.

This strategy combines exploration and exploitation in a principled manner. The precomputed diverse sequence of paths is allowed to proceed to collision test uninterrupted, so long as we believe the paths are a good investment. As the maximal survival estimates drop below the point of maximum entropy, exploitation of the model shows diminishing returns, indicating that either there are few viable paths or the model is inadequately informed. Thus, the sampler transitions to an exploratory strategy by selecting the path nearest to 50% likelihood of survival. Paths that are just above 50% survival probability fulfill both the exploration and exploitation goals, and so the transition between strategies is seamless.

Figure 13 visualizes a snapshot of the locality model during a replan cycle, while sampling with a hybrid strategy. Indicated in the figure are known collision sites, ranges of effect, and surviving paths. Note that only four failed collision tests were sufficient to achieve a model that adequately summarizes all salient obstacles in the vicinity of the robot.

8 Experimental Analysis

We conducted a set of experiments in simulation in order to obtain a sufficient quantity of trials to recognize statistically meaningful trends. Here we report results for a class of car-like mobile robot.

8.1 Setup

Experimental trials comprise sets of one-hundred simulated planning problems; in each one, the robot attempts to navigate through a cluttered, fully-known 2D environment with various levels and types of task space obstacle coverage. Each planning problem involves a randomized query comprising start and goal poses separated by a fixed straight-line distance. The robot is a disc of 0.41 m diameter and minimum turning radius of 0.48 m.

We present results for three environment categories. The first two categories comprise sets of point obstacles distributed uniformly at random. The obstacles are placed at a specified density in a 20 m x 20 m room, where queries are of length 14 m. Since they are randomly generated, it is impossible to know whether all problems posed to the planner are solvable. As the obstacle density increases, the likelihood of a given problem being unsolvable increases significantly. Thus, we provide results at two different densities, equating to easy (1%) and hard (1.5%) task space obstacle coverage.[1]

We also provide results in a structured office environment. The office plan of Willow Garage (2008), sampled at a 10 cm resolution, totals 55.7 m x 44.4 m. All 100 problems comprise randomized queries of straight-line length 39 m within this office map.

For each problem, the robot moves continuously at runtime while replanning at a fixed rate. A low-fidelity global planner helps guide the robot to the goal using an implementation of D*-Lite running in an 8-connected grid. Meanwhile, local paths of length 1.8 m are sampled until the replan cycle time runs out.

Local path collision test is performed by stepping through many poses along the path from start to end at a fine increment of 10 ms. At each pose, the collision tester checks for geometric overlap of the robot with obstacles in the full costmap. Any overlap causes an immediate return value of true; otherwise, false is returned and the path is marked as a survivor after reaching the end of the path at 6.0 sec.

[1] Compared to our earlier work, these problems appear harder because we eliminate a potentially confounding variable by disallowing path fragments shorter than the full 1.8 m path length.

Paths are selected for execution from amongst the survivors using a path classification technique described in our earlier work (Knepper et al., 2012). This technique first finds the path that minimizes the combination of local and global time to goal. It then performs an optimization, returning an equivalent surviving path that passes farther from the nearest obstacle, thus improving safety while retaining goal-directedness.

In our study, the independent variable—replan cycle time—varied between 0.0125 seconds and 0.2 seconds. This interval reflects the amount of time during which the planner may sample and collision-test paths before it must select one and send it to the robot for execution. Thus, we effectively vary the number of paths that can be sampled before making a decision. Without overhead for computing path samples, the reported cycle times correspond to a range between roughly 22 and 700 path samples. At a variety of points along this range, we recorded two dependent variables: fraction of paths surviving collision test, and overall planner success rate.

In experimentation, we consider several path selection strategies. See Section 7 for details.

- Low dispersion—sequence generated by the Green-Kelly algorithm (Green and Kelly, 2007).
- Pure exploitation—sample as far as possible from obstacles.
- Pure exploration—find boundaries; selects maximum entropy path.
- Hybrid sampling approach—selects next untested path in the low dispersion sequence predicted to have at least a 50% chance of survival, or else the safest available path.

Additionally, we compare three different locality models:

- Combined adaptive locality model (CALM), as described in Section 6.
- k-nearest neighbor voting—estimates probabilities of collision for untested paths according to the unweighted average outcome of neighboring, tested paths.
- Locally-weighted regression—incorporates proximity-based weighting into the nearest neighbors estimate.

The default range of effect used by CALM is the radius of the robot plus 10 cm, the minimal size of an obstacle. The last two competing locality models correspond to models utilized in prior work. The k-nearest neighbor voting method is an adaptation of the locality model used by Burns and Brock (2005b). In this model, the likelihood of a path's successful collision test is estimated by the average outcome of the k nearest neighboring paths. Path proximity for surviving paths is measured by the Hausdorff metric, whereas for failed paths, we utilize the PLUT to find the distance to the actual collision site. Note that both of these distances result from the Euclidean distance of a projection of a point on one path onto the opposite path in the task space, and so they are comparable.

The locally-weighted regression method is also patterned after the model from Burns and Brock (2004). It is a generative model that estimates the outcome of a particular collision test based on the proximity of k neighboring tested points/paths, weighted by the proximity of each to the path under consideration. It was necessary to adapt the technique slightly in order to employ it for path sampling as opposed to point sampling in the configuration space. To account for the fact that we are sampling in a non-Euclidean path space, we eliminated from this computation the notion of a "mean path coordinate," $\bar{p}$—instead folding it into the computation of the expected outcome. The equations were adapted from Burns and Brock (2004) as follows. First, we have a Gaussian distance weighting function based on the weighted Hausdorff distance between paths,

$$w(p, p_i) \quad = \quad e^{-a\,\mu_{\mathrm{H}}(p,p_i)^2}. \tag{22}$$

Next, we compute the weighted mean outcome, $\bar{y} \in [-1, 1]$, of our prospective path's k neighbors, p_i, using their outcomes, $y_i \in \{-1, 1\}$. Although we cannot write a mean path's coordinates, we can compute Δp, a prospective path p's distance from the mean.

$$\Delta p = \frac{\sum_i w(p, p_i)(\mu_{\mathrm{H}}(p, p_i))}{\sum_i w(p, p_i)} \tag{23}$$

$$\bar{y} = \frac{\sum_i w(p, p_i) y_i}{\sum_i w(p, p_i)} \tag{24}$$

Then we compute covariance as

$$\sigma_{py} = \frac{\sum_i w(p, p_i) \Delta p^2}{\sum_i w(p, p_i)} \tag{25}$$

$$\sigma_p^2 = \frac{\sum_i w(p, p_i) \left(y_i - \bar{y}\right)^2}{\sum_i w(p, p_i)}. \tag{26}$$

Finally, we can estimate the outcome for our candidate path, p,

$$\hat{y}(p) = \bar{y} + \frac{\sigma_{py}}{\sigma_p^2} \Delta p. \tag{27}$$

In their later work, Burns and Brock opted for the simpler k-nearest neighbors approach over this full statistical method for computational efficiency. We compare both methods here with the knowledge that we are considering fewer paths at one time. Our planning architecture amortizes planning time over execution time, and it can therefore spend additional time selecting each path.

8.2 Results

We present results on each of the strategies in Figures 14–17. In these plots, we separate out several effects for study.

Figure 14 plots results on the effect of sampling bias induced by collision-test feedback. Here we compare the biased path selection strategies from Section 8.1 against the unbiased low-dispersion path sampling approach. All biased sampling methods use CALM to compute probabilities. For sparse and dense randomized obstacles, we present both the yield of collision-free paths as well as the overall planning success rate.

Figure 15 presents the effects of several alternative models of locality. CALM is compared against k-nearest neighbors and locally-weighted regression (Section 8.1). In order to maximize fairness of the comparison among locality models, the hybrid selection method (Section 7.3) is used to sample paths in each case. A relatively small $k = 5$ was necessary particularly for rapid replan cycle times where few paths can be sampled. Note that for the first k paths sampled, k-nearest neighbors reduces to a tie that is resolved using the low-dispersion sequence. Again, the yield of safe paths and planner success rate are reported.

We also present results for planning in a structured office environment map in Fig. 16. In these plots, we present a few of the best-performing methods for path sampling and locality modeling. As before, results are presented for the yield of collision-free paths as well as overall planning success rate.

Finally, Fig. 17 presents the efficiency with which each combination of path selection algorithm and locality model samples paths. Low-dispersion path sampling has no overhead and so it is 100% efficient. Other methods introduce overhead both in computing the locality model and using it to select the next path sample.

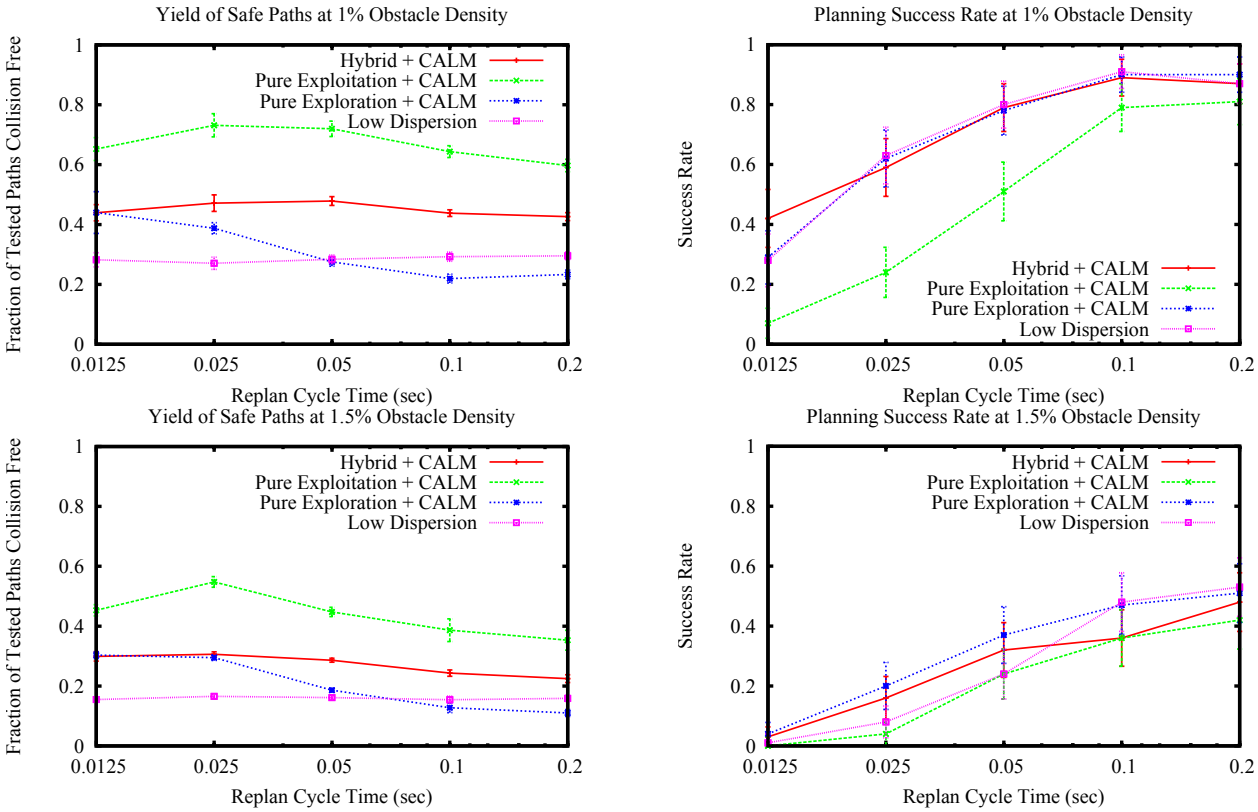


Figure 14: Comparison of biased versus unbiased sampling. Here, we demonstrate the advantages of biased path sampling. We compare low dispersion path sampling against locality-based sampling approaches using pure exploration, pure exploitation, and hybrid sampling with the combined adaptive locality model. Error bars indicate 95% confidence.

8.3 Analysis

Figure 14 reveals a statistically significant increase in the number of safe paths produced per unit time using CALM for both pure exploitation (up to 3.3x in these results) and the Hybrid path sampler (above 1.8x), compared to a fixed low dispersion sequence. However, path survival rate does not convey the entire planning story.

Most important is the motion planner's ability to successfully solve planning queries, and Hybrid+CALM shows good performance across the entire range of problems examined here. It significantly outperforms pure exploitation+CALM and also the two nearest-neighbor locality methods (k-nearest neighbors and locally-weighted regression). There is some indication of increased performance relative to low dispersion. All these trends are clearest at the low end of replan cycle time, indicating that Hybrid+CALM is able to make the best use out of limited available information.

The nearest neighbor locality methods require more path samples before achieving a model of sufficient fidelity to produce good results. This is especially so in the dense obstacle environment, demonstrating these models' dependence on a critical mass of samples yielding safe paths; in order for a region in path space to appear safe to the model, a cluster of $k/2$ already-safe paths must be present.

The comparison between the randomized and office environments is also revealing. In overall success rate, the pure exploration approach dominates the randomized obstacle environments, particularly at higher obstacle densities. However, its performance flags somewhat in the office

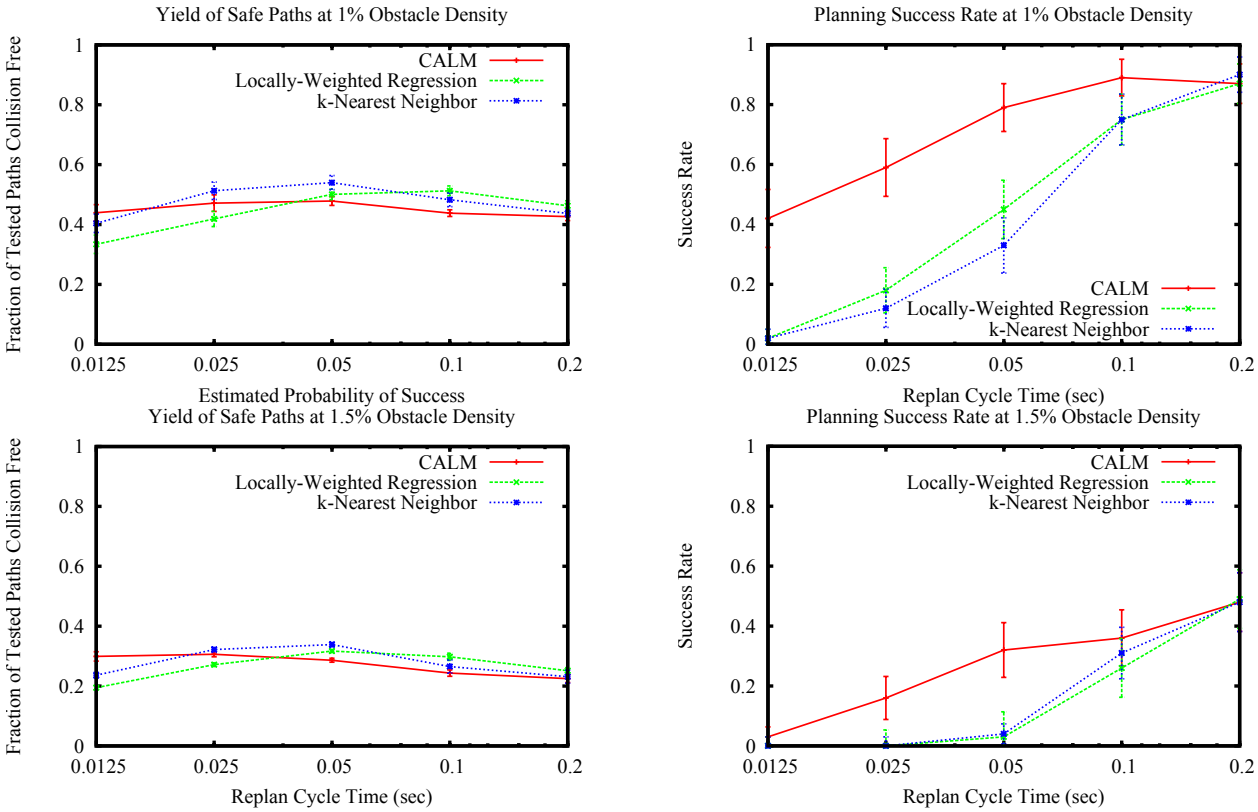


Figure 15: Comparison of obstacle location models. CALM, presented in Sections 5.2–6, is compared with two methods employed for similar purposes in the work of Burns and Brock. The two major differences in CALM are the use of safe paths to "block" nearby obstacles and the incorporation of safe path corridors from the prior replan cycle. For the methods borrowed from Burns and Brock, a value of $k = 5$ was used. Error bars indicate 95% confidence.

environment. This distinction can be explained by the abundance of wide open spaces in the office. The pure exploration approach only samples near the edge of obstacles, and so it produces fewer paths centrally located in rooms. Such paths are the safest and often position the robot best to enter corridors with proper alignment. By contrast, the Hybrid approach balances sampling between open space and boundaries, thus achieving both exploration and exploitation.

Figures 18–19 provide additional data to probe more deeply into these results. In these figures, sets of one-hundred planning problems were simulated in a randomized world of density 1% with a replan cycle time of 0.1 sec. Figure 18 depicts the fraction of paths surviving collision test as a function of the estimated likelihood of path safety at the time each path was tested. A perfect model would appear as a diagonal line of slope one. Deviations from this ideal reveal the extent of predictive power for each locality model. Each of the models shown approximates a positive slope, indicating general correctness.

We observe that all three locality methods possess a monotonically-increasing curve with slope less than one. CALM is on average more optimistic than the others about the collision chances of a given path. This result occurs despite the pessimistic independence assumption made by CALM. We believe that the net optimism occurs due to the simplification of locality into two sides of an obstacle. The model fails to account for the true variety of shapes in the obstacles encountered by the robot. Despite this apparent shortcoming, we see that CALM produces qualitatively correct predictions that lead to a high planning success rate.

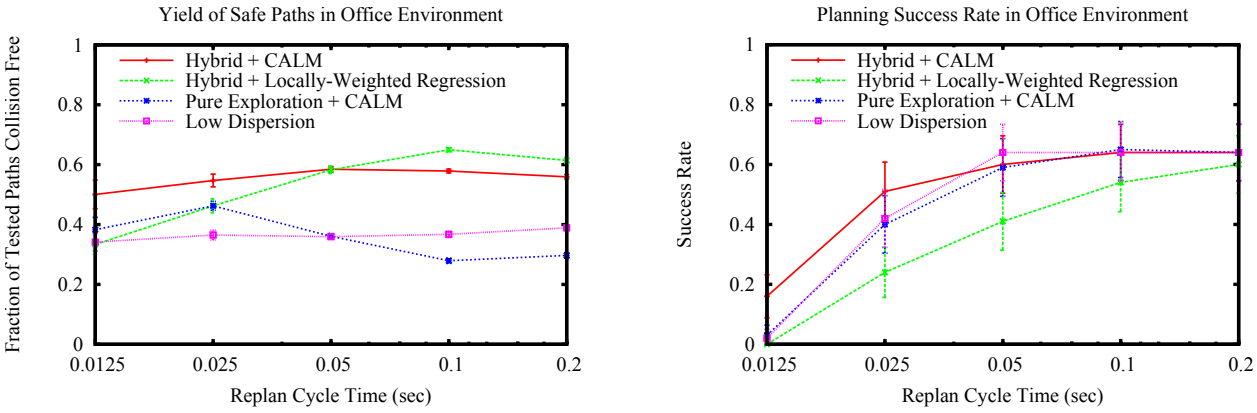


Figure 16: Simulated Willow Garage office environment experiments. Error bars indicate 95% confidence.

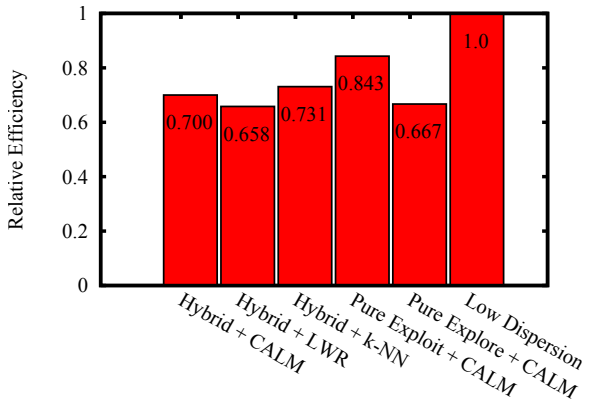


Figure 17: Path sampling efficiency for each combination of sampling method and locality model. Overhead is introduced by both the sampling and locality modeling processes. The low-dispersion sequence is 100% efficient because it is precomputed.

At the end of each replan cycle, only one path ultimately matters—that is, the path ultimately selected for execution. Figure 19 examines the output of each replan cycle as a function of the estimated survival likelihood of that path at the time it was tested. Note that the locality estimates were not used here to select paths; they were only recorded in histogram form. Rather, path selection was performed via the uninformed low-dispersion path sequence, and so variations among the methods are directly due to the differing locality models' estimates.

It is clear that the nearest-neighbor-based planners do not consistently select the safest paths for execution. In fact, a pure shortest-path selection heuristic would show a skew to the left in the plot because the shortest path around an obstacle approaches very close to the obstacle, and thus under uncertainty the path should be assigned a low probability of survival.

By contrast, we employ a more sophisticated path selection approach for all three locality methods that optimizes safety while ensuring progress toward the goal. For the nearest-neighbor models, such a heuristic results in the selection of paths spread out across the risk spectrum. That this spread is likely the result of the robot passing through corridors of varying diameter. Nevertheless, these locality models appear to give little insight into which paths the planner is most likely to actually select for execution. A locality model informed by the path sampler's preferences could be expected to result in a higher performance planner.

In contrast, CALM rarely selects uncertain paths. The sharp spike near 100% survival reflects

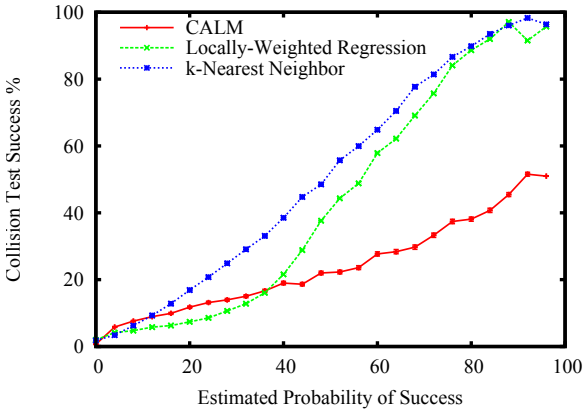


Figure 18: The predictive power of each locality model is shown by the rate of correct estimates. Each time a path is collision-tested, the estimated probability of the path is noted and compared to the actual collision-test outcome. The fraction of overall successes (collision-free) out of total collision tests in each of 25 bins is plotted across the range of collision estimates. 95% confidence error bars are imperceptibly small.

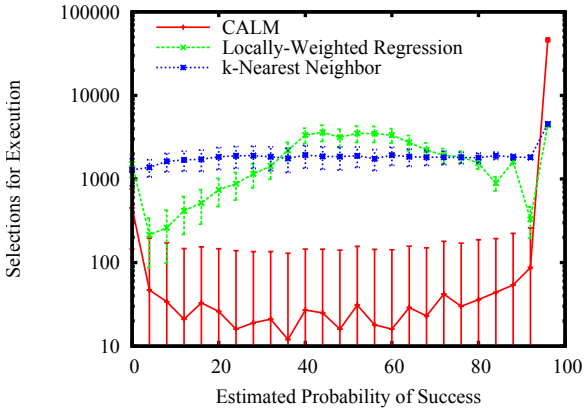


Figure 19: This histogram depicts the rate at which the motion planner selects for execution different types of paths, as categorized by the likelihood of a path's survival. Path survival probability is shown as estimated immediately before the path is selected for collision test. The collision test returned false (safe) for all paths shown, hence they were available to be selected for execution. Error bars indicate 95% confidence.

the adaptive model, which conforms to various sizes of obstacle and corridor. Even in a tight corridor, if a gap has been established then other paths can be expected to exist with high likelihood.

The neighbor-based methods in both of these plots use a value of $k = 24$ in order to map estimates uniformly onto the 25 bins used to generate the histograms.

9 Discussion

In real-time planning, performance is highly sensitive to the computational cost associated with the collision-test process. Methods that alleviate some of the computation of this process can be beneficial, provided that such alternatives are less costly than the collision tests they replace. In particular, we avoid performing many of those collision tests that are most likely to fail since they are least likely to yield a viable path for execution on the robot.

Success at real-time planning stems from several factors, including the yield of collision-free paths. The quality of a set of collision-free paths is reflected in the amount of choice presented to the planner. Choice in turn breaks down into two characteristics: quantity and diversity. A perfect path sampler would eliminate only colliding paths from consideration and would thus never impair choice. Given the uncertainty of the process, however, it is inevitable that some paths that would have tested collision-free will be missed. The impact of this effect on overall choice depends on which other path is tested instead. A net increase in choice available to the planner can be achieved if the would-be colliding paths can be replaced with collision-free paths without adversely impacting the overall diversity of collision-free paths discovered by the planner. More study is needed to understand how best to balance these factors.

In this paper, we present a strategy for informed path sampling that guides the search away from obstacles and towards safe or unexplored parts of the task space. Although obstacle information is already available to the planner in costmap form, we obtain a significant increase in performance by representing the most salient subset of those obstacles in a more immediately accessible form.

We utilize a proximity look-up table to accelerate this process by precomputing the relationships among motions that obey constraints. Even so, our statistical model describing nearby obstacles and their relationship is necessarily simple. This model makes use of the principle of locality to maximally reduce uncertainty by searching appropriately far from obstacle locations already discovered in prior collision tests. Using our probabilistic locality model, we trade off between exploration and exploitation in order to discover a variety of safe paths while avoiding searching many colliding paths.

We have demonstrated the machinery to improve the path sampling process by biasing search away from obstacles. We must concede, however, that many in the robotics research community will continue to utilize low-dispersion and even random sequences for path sampling because the machinery we describe comes at a cost in implementation complexity. Still, in the long run robots must become smarter about how they conduct search. As robotic products become more widely commercialized, all computing must fit in the limited computational budget provided by economical embedded processors.

Additionally, robot motion planners will soon need to evaluate paths with respect to many more factors than they do today. Robots that operate in human environments must be situationally aware and responsive. They must generate smooth, intuitive, spontaneous motion that considers many more factors than most state-of-the-art motion planners, including safety, visibility, social convention, and so on. Each of these factors comes at a computational cost. Limits on the total number of paths the planner can evaluate per unit time place a premium on each path tested.

Finally, to form common ground with humans, robots must be able to reason similarly to

humans about the space they inhabit. They must regard each obstacle not as a point cloud but as a single obstacle region, and they must consider corridors of free space not as collections of paths but as a single option for traversal. For these reasons, we believe that a synopsis of spatial structures—both obstacle and free—such as is provided by CALM will be increasingly important to robotic motion planners of the future.

Funding

This work is sponsored by the Defense Advanced Research Projects Agency. This work does not necessarily reflect the position or the policy of the Government. No official endorsement should be inferred.

References

D. Aarno, D. Kragic, and H. I. Christensen. Artificial potential biased probabilistic roadmap method. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 461–466, New Orleans, USA, April 2004.

N. Amato and Y. Wu. A randomized roadmap method for path and manipulation planning. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 113–120, Minneapolis, USA, 1996.

N. M. Amato, O. B. Bayazit, L. K. Dale, C. Jones, and D. Vallejo. OBPRM: An obstacle-based PRM for 3D workspaces. In *Proceedings of the Workshop on the Algorithmic Foundations of Robotics*, pages 155–168, Houston, TX, USA, March 1998.

B. Baginski. Local motion planning for manipulators based on shrinking and growing geometry models. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3303–3308, Minneapolis, USA, 1996.

W. H. Beyer, editor. *CRC Standard Mathematical Tables and Formulae*. CRC Press, Boca Raton, FL, 1991.

L. Blackmore, M. Ono, and B. C. Williams. Chance-constrained optimal path planning with obstacles. *IEEE Transactions on Robotics*, 27(6), 2011.

R. Bohlin and L. E. Kavraki. Path planning using lazy PRM. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 521–528, San Francisco, USA, April 2000.

V. Boor, M. H. Overmars, and A. F. van der Stappen. The Gaussian sampling strategy for probabilistic roadmap planners. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1018–1023, 1999.

B. Burns and O. Brock. Information theoretic construction of probabilistic roadmaps. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, Las Vegas, NV, USA, October 2003.

B. Burns and O. Brock. Model-based motion planning. Technical Report 04-32, Computer Science Department, University of Massachusetts – Amherst, September 2004.

B. Burns and O. Brock. Sampling-based motion planning using predictive models. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Barcelona, Spain, April 2005a.

B. Burns and O. Brock. Single-query entropy-guided path planning. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Barcelona, Spain, April 2005b.

B. Burns and O. Brock. Toward optimal configuration space sampling. In *Proceedings of the Robotics: Science and Systems Conference*, Cambridge, MA, USA, June 2005c.

A.D. Collins, P.K. Agarwal, , and J.L. Harer. HPRM: A hierarchical prm. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 4433–4438, 2003.

J. Denny and N. Amato. Toggle prm: Simultaneous mapping of c-free and c-obstacle: A study in 2d. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, San Francisco, USA, September 2011.

P. Ferbach and J. Barraquand. A method of progressive constraints for manipulation planning. *IEEE Transactions on Robotics and Automation*, 13(4):473–485, 1997.

M. Garber and M. Lin. Constraint-based motion planning using voronoi diagrams. In *Proceedings of the Workshop on the Algorithmic Foundations of Robotics*, 2004.

C. Green and A. Kelly. Toward optimal sampling in the space of paths. In *Proceedings of the International Symposium on Robotics Research*, Hiroshima, Japan, November 2007.

P. D. Grünwald and A. P. Dawid. Game theory, maximum entropy, minimum discrepancy and robust bayesian decision theory. *The Annals of Statistics*, 32(4), 2004.

C. Holleman and L. E. Kavraki. A framework for using the workspace medial axis in PRM planners. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1408–1413, San Francisco, USA, April 2000.

T. Horsch, F. Schrwarz, , and H. Tolle. Motion planning with many degrees of freedom—random reflections at c-space obstacles. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3318–3323, San Diego, USA, 1994.

D. Hsu, L. E. Kavraki, J. C. Latombe, R. Motwani, and S. Sorkin. On finding narrow passages with probabilistic roadmap planners. In *Robotics: The Algorithmic Perspective*, pages 141–154, 1998.

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*, Taipei, Taiwnn, September 2003.

D. Hsu, G. Sánchez-Ante, and Z. Sun. Hybrid PRM sampling with a cost-sensitive adaptive strategy. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3885–3891, 2005.

D. Hsu, J.C. Latombe, and H. Kurniawati. On the probabilistic foundations of probabilistic roadmap planning. *International Journal of Robotics Research*, 25(7):627–643, 2006.

L. Jaillet, A. Yershova, S. M. LaValle, and T. Simeon. Adaptive tuning of the sampling domain for dynamic-domain RRTs. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, 2005.

E. T. Jaynes. Information theory and statistical mechanics. *Physical Review*, 106(4):620–630, May 1957.

M. Kalisiak and M. van de Panne. RRT-blossom: RRT with a local flood-fill behavior. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Orlando, USA, May 2006.

S. Karaman and E. Frazzoli. Sampling-based algorithms for optimal motion planning. *International Journal of Robotics Research*, 30(7):846–894, June 2011.

L. E. Kavraki, P. Švestka, J. C. Latombe, and M. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, 12(4), August 1996.

R. A. Knepper and M. T. Mason. Empirical sampling of path sets for local area motion planning. In *Proceedings of the International Symposium of Experimental Robotics*, Athens, Greece, July 2008.

R. A. Knepper and M. T. Mason. Path diversity is only part of the problem. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Kobe, Japan, May 2009.

R. A. Knepper and M. T. Mason. Realtime informed path sampling for motion planning search. In *Proceedings of the International Symposium on Robotics Research*, Flagstaff, USA, August 2011.

R. A. Knepper, S. S. Srinivasa, and M. T. Mason. Toward a deeper understanding of motion alternatives via an equivalence relation on local paths. *International Journal of Robotics Research*, 31(2):168–187, February 2012.

M. Kobilarov. Cross-entropy randomized motion planning. In *Proceedings of the Robotics: Science and Systems Conference*, Los Angeles, USA, June 2011.

H. Kurniawati and D. Hsu. Workspace importance sampling for probabilistic roadmap planning. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, pages 1618–1623, Sendai, Japan, September 2004.

P. Leven and S. Hutchinson. A framework for real-time path planning in changing environments. *International Journal of Robotics Research*, 21(12):999–1030, 2002.

P. Leven and S. Hutchinson. Using manipulability to bias sampling during the construction of probabilistic roadmaps. *IEEE Transactions on Robotics and Automation*, 19(6):1020–1026, 2003.

P. E. Missiuro and N. Roy. Adapting probabilistic roadmaps to handle uncertain maps. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Orlando, USA, May 2006.

M. Morales, L. Tapia, R. Pearce, S. Rodriguez, and N. M. Amato. A machine learning approach for feature-sensitive motion planning. In *Proceedings of the Workshop on the Algorithmic Foundations of Robotics*, pages 361–376, Utrecht/Zeist, The Netherlands, July 2004.

C. Nielsen and L. E. Kavraki. A two-level fuzzy prm for manipulation planning. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, pages 1716–1722, November 2000.

C. Nissoux, T. Simeon, and J.-P. Laumond. Visibility-based probabilistic roadmaps. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, pages 1316–1321, 1999.

M. Rickert, O. Brock, and A. Knoll. Balancing exploration and exploitation in motion planning. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2812–2817, 2008.

C. E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27: 379–423, 623–656, July, October 1948.

G. Song, S. Miller, and N. M. Amato. Customizing prm roadmaps at query time. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Seoul, Korea, May 2001.

S. Thomas, M. Morales, and X. Tangand N. Amato. Biasing samplers to improve motion planning performance. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1625–1630, 2007.

J. van den Berg and M. H. Overmars. Using workspace information as a guide to non-uniform sampling in probabilistic roadmap planners. *International Journal of Robotics Research*, 24(12), December 2005.

B. Vidakovic. Γ*-minimax: A paradigm for conservative robust Bayesians*, volume 152 of *Lecture Notes in Statistics*, pages 241–259. Springer-Verlag, New York, 2000.

Willow Garage. `http://pr.willowgarage.com/wiki/Maps?action=AttachFile&do=get&target=willow-full-2008-11-26-100mm.png`, 2008. Accessed May 24, 2012.

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 IEEE International Conference on Robotics and Automation*, pages 1024–1031, Detroit, USA, May 1999.

Y. Yang and O. Brock. Adapting the sampling distribution in PRM planners based on an approximated medial axis. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 4405–4410, April 2004.

A. Yershova, L. Jaillet, T. Simeon, and S. M. LaValle. Dynamic-domain RRTs: Efficient exploration by controlling the sampling domain. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Barcelona, Spain, April 2005.

Y. Yu and K Gupta. An information theoretic approach to view point planning for motion planning of eye-in-hand systems. In *International Symposium on Robotics*, pages 306–311, 2000.

M. Zucker, J. Kuffner, and J. A. Bagnell. Adaptive workspace biasing for sampling-based planners. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Pasadena, USA, May 2008.