Workspace-based Connectivity Oracle
An Adaptive Sampling Strategy for PRM Planning

Hanna Kurniawati and David Hsu

National University of Singapore, Singapore 117543, Singapore
{hannakur,dyhsu}@comp.nus.edu.sg

Summary. This paper presents *Workspace-based Connectivity Oracle* (WCO), a dynamic sampling strategy for probabilistic roadmap planning. WCO uses both domain knowledge—specifically, workspace geometry—and sampling history to construct dynamic sampling distributions. It is composed of many component samplers, each based on a geometric feature of a robot. A component sampler updates its distribution, using information from the workspace geometry and the current state of the roadmap being constructed. These component samplers are combined through the adaptive hybrid sampling approach, based on their sampling histories. In the tests on rigid and articulated robots in 2-D and 3-D workspaces, WCO showed strong performance, compared with sampling strategies that use dynamic sampling or workspace information alone.

1 Introduction

Probabilistic roadmap (PRM) planning [6] is currently the most successful approach for motion planning of robots with many degrees of freedom (DOFs). PRM planners sample a robot's configuration space $\mathcal{C}$ according to a suitable probability distribution and capture the connectivity of $\mathcal{C}$ in a graph, called a roadmap, which is an extremely simplified representation of $\mathcal{C}$.

Despite their successes, PRM planners behave poorly when $\mathcal{C}$ contains narrow passages. A narrow passage is a small region whose removal changes the connectivity of $\mathcal{C}$. The probability of sampling in narrow passages is low, because of their small volumes. In such spaces, it is difficult for PRM planners to build roadmaps with good connectivity. Although many PRM planners have been proposed (e.g., [4, 9, 12, 16, 17, 18, 21, 22]), narrow passages remain a bottleneck for PRM planning.

With few exceptions, most PRM planners use *static* sampling distributions based on *a priori* assumed geometric properties of the configuration space or the workspace. Interestingly, the first PRM planner [16], which consists of two sampling stages, uses *dynamic* sampling: the second stage exploits information gathered in the first stage to update the sampling distribution and resample $\mathcal{C}$. Recently, with the use of machine learning techniques in PRM planning [5, 13, 19], dynamic sampling has again gained popularity. Dynamic

sampling incrementally infers partial knowledge of key geometric properties of $\mathcal{C}$ during the roadmap construction and uses this knowledge to adapt the sampling distribution. It reveals and exploits the probabilistic foundations of PRM and is a promising way of speeding up PRM planners [11].

To infer geometric properties of $\mathcal{C}$, existing PRM planners with dynamic sampling use sampling history. This is, however, inadequate. For example, to learn the usefulness of sampling a particular region of $\mathcal{C}$, we need many samples in that region. This is difficult to achieve in narrow passages. One way of addressing this issue is to use domain knowledge such as the geometry of robots and obstacles in the workspace $\mathcal{W}$. Compared with $\mathcal{C}$, $\mathcal{W}$ has low dimensionality and an explicit geometric representation, which make it easy to find narrow passages. Narrow passages in $\mathcal{W}$ often suggest the presence and the location of narrow passages in $\mathcal{C}$. Furthermore, workspace geometry provides information complementary to that from sampling history.

In this paper, we present a new PRM planner that combines information from both workspace geometry and sampling history to construct a dynamic sampling distribution. Core to our new planner is a new sampling strategy called *Workspace-based Connectivity Oracle* (WCO). WCO is an ensemble sampler composed of many component samplers. They are all based on a key observation: a collision-free path between a start configuration s and a goal configuration g in a robot's configuration space $\mathcal{C}$ implies a collision-free path in $\mathcal{W}$ for *every* point in the robot between the corresponding start and goal positions of the point. So, if we find a collision-free path in $\mathcal{W}$ for every point in the robot and all these paths correspond to the same path γ in $\mathcal{C}$, then γ is indeed a collision-free path in $\mathcal{C}$ for the robot to move from s to g. Finding a path for every point is, of course, impractical. Nevertheless, we can use the paths of a set of geometric features in $\mathcal{W}$—points, line segments, triangles, etc.— to predict regions of $\mathcal{C}$ that are likely to be useful for connecting disconnected components of a roadmap. Each WCO component sampler is based on a single geometric feature. They are then combined, based on their sampling histories, through the adaptive hybrid sampling (AHS) approach [13], which is a restricted form of reinforcement learning.

2 Background

2.1 PRM Basics

The standard multi-query PRM approach consists of two phases, roadmap construction and roadmap query. In the roadmap construction phase, the planner samples $\mathcal{C}$ according to a suitable probability distribution and approximates the connectivity of $\mathcal{C}$ with a roadmap graph $\mathcal{R}$. The nodes of $\mathcal{R}$ represent sampled collision-free configurations, called *milestones*. An edge exists between two milestones if they can be connected with a collision-free straight-line path. In the query phase, the planner is given a start and a goal configuration. It tries to connect the two query configurations to two corre-

sponding milestones in $\mathcal{R}$ and then searches for a path in $\mathcal{R}$ between these two milestones, using standard graph search algorithms. See [6] for details.

The performance of PRM planners depends critically on the quality of roadmaps constructed. A good roadmap has two important properties: *coverage* and *connectivity*. Denote by $\mathcal{F}_c$ the collision-free subset of $\mathcal{C}$. Good coverage means that for any configuration $q \in \mathcal{F}_c$, there is a collision-free straight-line path between q and a milestone in $\mathcal{R}$ with high probability. Good connectivity means that any two milestones in the same connected component of $\mathcal{F}_c$ are also connected by a path in $\mathcal{R}$. Good coverage is relatively easy to achieve through uniform sampling [15]. Good connectivity is more difficult to achieve, especially when $\mathcal{F}_c$ contains narrow passages.

2.2 PRM Planners that Generate Dynamic Sampling Distributions

In addition to the early PRM planner in [16], the planner in [19] also uses a two-stage sampling strategy, but employs more sophisticated techniques to generate the distribution for resampling $\mathcal{C}$ in the second stage. Instead of breaking sampling into two stages, some recent planners use on-line machine learning to update sampling distributions incrementally. In adaptive hybrid sampling [13], the sample distribution is constructed as a linearly weighted combination of component distributions. To adapt the distribution, the weights are updated after each sampling operation during roadmap construction to favor component distributions having the most promising results. In entropy-guided planning [5], an approximate model of $\mathcal{C}$ is built and used to sample $\mathcal{C}$ so that the expected value of a utility function is maximized.

2.3 PRM Planners that Use Workspace Information

Several PRM planners use workspace geometry as a guide for sampling $\mathcal{C}$. Let $\mathcal{F}_\mathcal{W}$ denote the subset of $\mathcal{W}$ that is not occupied by obstacles. Some planners bias sampling by focusing a fixed set of workspace paths, e.g., paths on the medial axis of $\mathcal{F}_\mathcal{W}$ [9, 10, 22]. Other planners bias sampling by identifying important regions in $\mathcal{W}$. For instance, the watershed algorithm focuses on small regions connecting large open regions [21]. Workspace importance sampling (WIS) focuses on regions with small local feature size [17]. These planners all use static sampling distributions.

In summary, workspace geometry has been used in earlier work to construct static sampling distributions for PRM planners. Sampling history has been used to update sampling distributions dynamically. Our new sampling strategy combines the information from both workspace geometry and sampling history to construct a dynamic sampling distribution.

3 Overview

3.1 The WCO Planner

Our planner adopts the standard multi-query PRM approach described in Section 2.1 and uses WCO for sampling $\mathcal{C}$. Since there is no confusion, we use WCO to refer to both the sampling strategy and the planner.

Before describing WCO, let us first consider the relationship between $\mathcal{C}$ and $\mathcal{W}$. For a point f in the robot, let $P_f(q)$ be the position in $\mathcal{W}$ of f when the robot is placed at configuration $q \in \mathcal{C}$. We call the mapping $P_f: \mathcal{C} \rightarrow \mathcal{W}$ a projection, as $\mathcal{C}$ has higher dimensionality than $\mathcal{W}$. Similarly, we define the lift mapping $L_f: \mathcal{W} \rightarrow 2^{\mathcal{C}}$. For any $x \in \mathcal{W}$, $L_f(x)$ is the subset of $\mathcal{C}$ such that each configuration in $L_f(x)$ places f at x. For convenience, we extend the definitions of P_f and L_f to subsets of $\mathcal{C}$ and $\mathcal{W}$, respectively, by taking set union. Using this notation, we can state the observation in Section 1 formally:

Proposition 1. *If two configurations $q, q' \in \mathcal{C}$ are connected by a path in $\mathcal{F}_c$, then for any point f in a robot, $P_f(q)$ and $P_f(q')$, the projections of q and q' in $\mathcal{W}$, are connected by a path in $\mathcal{F}_{\mathcal{W}}$.*

During the roadmap construction, WCO maintains a partially constructed roadmap $\mathcal{R}$. Distinct connected components of $\mathcal{R}$ may in fact lie in the same connected component of $\mathcal{F}_c$, due to inadequate sampling of certain critical regions. To sample such regions, WCO examines the workspace paths of a set of *feature points* in the robot and constructs a sampler for each feature point f. To connect two components $\mathcal{R}_1$ and $\mathcal{R}_2$ of $\mathcal{R}$, we use P_f to project the milestones of $\mathcal{R}$ into $\mathcal{W}$ and search for "channels" in $\mathcal{W}$ that connect the projected milestones of $\mathcal{R}_1$ and $\mathcal{R}_2$. These channels suggest the regions of $\mathcal{C}$ that may connect $\mathcal{R}_1$ and $\mathcal{R}_2$. So, we use L_f to lift the channels into $\mathcal{C}$ and adapt the distribution to sample more densely in the regions covered by the lifted channels. To be sensitive to the changes in $\mathcal{R}$, WCO adapts its sampling distribution incrementally whenever a new milestone is added to $\mathcal{R}$.

Although workspace-based PRM planners often consider only a single feature point, this is inadequate. By Proposition 1, a collision-free path in $\mathcal{C}$ implies a collision-free path in $\mathcal{W}$ for every point in the robot. So, we use a set of pre-selected feature points and construct an independent sampler s_i for each feature point f_i, $i = 1, 2, \ldots$. We make two simplifying assumptions. First, a finite number of feature points are sufficient to indicate the important regions of $\mathcal{C}$ for sampling. Second, we can treat the feature points independently. These two assumptions reduce the computational cost and are shown to be effective in identifying important regions of $\mathcal{C}$ (see Section 6). Despite the independence assumption, the kinematic constraints of a robot are not entirely ignored. Implicitly, WCO assigns higher sampling density to regions obeying such constraints. To provide roadmap coverage, we add a uniform sampler s_0 to the WCO samplers $s_1, s_2, \ldots$ and form the set $S = \{s_0, s_1, s_2, \ldots\}$. The component samplers in S are combined through the AHS approach to form an ensemble sampler: each component sampler has an associated weight proportional to the probability of it being used, and the weights are adjusted to reflect the success of the sampler according to the sampling history.

3.2 When is Workspace Connectivity Information Useful?

To represent $\mathcal{F}_{\mathcal{W}}$, the collision-free subset of $\mathcal{W}$, WCO computes a decomposition $\mathcal{T}$ of $\mathcal{F}_{\mathcal{W}}$ into non-overlapping cells. It represents the connectivity of

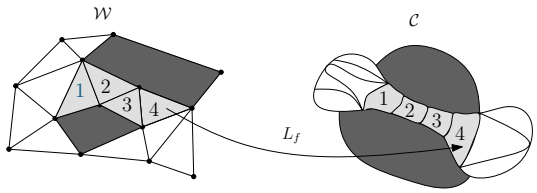


Fig. 1. A partition of $\mathcal{W}$ induces a partition of $\mathcal{C}$. Obstacles are shaded in dark color. A workspace channel and its lifted version are shaded in light color. In general, L_f is a one to many mapping. It maps a region of $\mathcal{W}$ to several regions of $\mathcal{C}$.

$\mathcal{F}_\mathcal{W}$ as an adjacency graph $G_\mathcal{T}$ for the cell decomposition $\mathcal{T}$. Each node of $G_\mathcal{T}$ represents a cell in $\mathcal{T}$, and two nodes are connected by an edge if the corresponding cells are adjacent. By Proposition 1, if two cells $t, t' \in \mathcal{T}$ are disconnected in $\mathcal{F}_\mathcal{W}$, then $L_f(t)$ and $L_f(t')$ are disconnected in $\mathcal{F}_\mathcal{C}$ for any feature point f. Thus, the connectivity information encoded in $G_\mathcal{T}$ can help in capturing the connectivity of $\mathcal{F}_\mathcal{C}$ during the roadmap construction.

Although we often think of $\mathcal{W}$ and $\mathcal{C}$ as two distinct spaces, they are closely related. For a fixed feature point f, $\mathcal{T}$ induces a partition of the collision-free subset of $\mathcal{C}$ into equivalent classes: $\mathcal{F}_\mathcal{C} = \bigcup_{t\in\mathcal{T}}(L_f(t) \cap \mathcal{F}_\mathcal{C})$ and for all $t, t' \in \mathcal{T}$ and $t \neq t'$, $L_f(t) \cap L_f(t') = \emptyset$ (Fig. 1). Two configurations are in the same equivalent class if they project to the same cell in $\mathcal{T}$. WCO exploits this connection extensively to integrate the information from both $\mathcal{W}$ and $\mathcal{C}$.

To connect two milestones m and m' of a roadmap, consider their projections. Suppose that $P_f(m) \in t$ and $P_f(m') \in t'$, where $t, t' \in \mathcal{T}$. A *workspace channel* λ is the set of cells corresponding to the nodes on a path in $G_\mathcal{T}$ between t and t'. The lifted channel $L_f(\lambda)$ suggests a region of $\mathcal{F}_\mathcal{C}$ for sampling in order to connect m and m'. Of course, no particular $L_f(\lambda)$ guarantees that a path between m and m' can be found within it, as the converse of Proposition 1 is not true in general. Nevertheless, a channel helps to improve sampling efficiency by narrowing down the sampling domain to a relevant subset of $\mathcal{F}_\mathcal{C}$.

The usefulness of a workspace channel can be defined formally:

Definition 1. *Let m and m' be two milestones of a roadmap and λ be a workspace channel between the cells containing $P_f(m)$ and $P_f(m')$ for some feature point f. The channel λ has the (n, p)-property if n samples drawn from $L_f(\lambda)$ are sufficient to find a path in $\mathcal{F}_\mathcal{C}$ between m and m' with probability at least p, provided such a path exists.*

A channel $L_f(\lambda)$ has good (n, p)-property if n is small and p is large. It is known that $L_f(\lambda)$ has good (n, p)-property under various conditions, e.g., path clearance [14], ε-complexity [20], and expansiveness [12]. The effectiveness in finding useful workspace channels depends on the cell decomposition and the method of searching for channels. These issues are detailed in the next section.

4 Constructing a WCO Component Sampler

We now describe the construction of a component sampler of WCO for a fixed feature point, specifically, how to extract workspace connectivity (Section 4.1), how to adapt the sampling distribution (Section 4.2), and how to take a sample for rigid and articulated robots (Section 4.3).

4.1 Extracting Workspace Connectivity

WCO computes a cell decomposition $\mathcal{T}$ of $\mathcal{F}_{\mathcal{W}}$ and represents the connectivity of $\mathcal{F}_{\mathcal{W}}$ in the adjacency graph of $\mathcal{T}$. This decomposition is computed only once and used by all WCO component samplers. Many spatial decomposition methods can be used here, e.g., triangulations and quadtrees for 2-D workspaces and their counterparts for 3-D workspaces.

Building on our earlier work [17], we have chosen to sample the boundary of obstacles in $\mathcal{W}$ and construct a Delaunay triangulation [8] over the sampled points. Under reasonable assumptions, the constructed triangulation is conforming [1], meaning that every triangle in the resulting triangulation lies either entirely in $\mathcal{F}_{\mathcal{W}}$ or its complement. Although helpful, this property is not required for our purposes. Throughout the rest of the paper, triangles refer to triangles in 2-D workspaces and tetrahedra in 3-D workspaces.

4.2 Adapting the Sampling Distribution

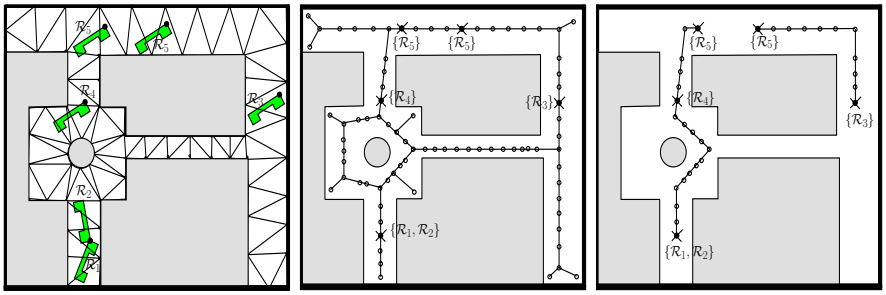
A skeleton of a WCO component sampler is shown in Algorithm 1. Let us now look at how it represents and updates the sampling distribution based on workspace channels. During the roadmap construction, WCO maintains a partially constructed roadmap $\mathcal{R}$. To sample a new milestone, each component sampler maintains a separate sampling distribution $\pi_{\mathcal{T}}$ defined over $\mathcal{T}$. The distribution $\pi_{\mathcal{T}}$ assigns equal probabilities to all triangles of $\mathcal{T}$ inside workspace channels and zero probabilities to all other triangles.

To find workspace channels, we first project milestones of $\mathcal{R}$ to $\mathcal{W}$ (Algorithm 1, line 8). Suppose that a milestone m belongs to a roadmap component $\mathcal{R}_i$ of $\mathcal{R}$. We associate $\mathcal{R}_i$ with the triangle $t \in \mathcal{T}$ that contains $P_f(m)$. Thus, each triangle t contains a set of labels that indicates the roadmap components which t is associated with. A triangle t is a *terminal* if its label set is non-empty, meaning that t contains at least one projected milestone. See Fig. 2*a* for an example.

Next, we find channels that connect terminals with different label sets by considering the adjacency graph $G_{\mathcal{T}}$. We compute a subgraph of $G_{\mathcal{T}}$, called a *channel graph* G', that spans all the terminals and connect them together. The workspace channels are then the triangles corresponding to the vertices of G'. The intuition behind the channel graph is very much like that of a roadmap in the configuration space: it uses simple paths, in this case, the shortest paths to connect every pair of two terminals that are close to each other and have different label sets. See Fig. 2 for an example.

Algorithm 1 A WCO component sampler.

1: Given a feature point f, sample a configuration q, based on the sampling distribution defined over the decomposition $\mathcal{T}$.
2: **if** q is collision-free **then**
3: Insert q into the roadmap $\mathcal{R}$ as a new milestone m.
4: $N_m \leftarrow$ the set of at most M milestones closest to m among all existing milestones of $\mathcal{R}$ within a distance of $D_{\max}$ from m. M and $D_{\max}$ are fixed constants.
5: **for** each $m' \in N_m$ **do**
6: **if** m' and m lie in different connected components of $\mathcal{R}$ **then**
7: Check whether there is a collision-free straight-line path between m and m'. If so, insert an edge between m and m' into $\mathcal{R}$,
8: Project m to $\mathcal{W}$.
9: Update the label sets for all affected triangles in $\mathcal{T}$.
10: Delete paths in G' that connect terminals with the same label set.
11: Let $t \in \mathcal{T}$ be the triangle that contains $P_f(m)$. Perform a breadth-first search from t and stop when reaching the first terminal t' other than t.
12: Add the path between t and t' to G' if t and t' hold different label sets.
13: Update the sampling distribution.



(a) (b) (c)

Fig. 2. (a) Milestones projected to the triangulated workspace. The labels indicate the roadmap components to which the milestones belong. The feature point of the rigid robot is marked by a black dot. (b) The adjacency graph $G_{\mathcal{T}}$. Terminals are marked by crosses. (c) The channel graph G' . Paths that connect terminals with the same label set (e.g., the path between the two terminals labelled $\{\mathcal{R}_5\}$) are not in G', as they connect terminals corresponding to milestones in the same connected components of $\mathcal{R}$ and hence unlikely to help in improving the connectivity of $\mathcal{R}$.

The channel graph is computed incrementally, as new milestones are added (Algorithm 1, lines 10–12). The incremental construction allows WCO to respond to changes in $\mathcal{R}$ and simplifies computation. To see that G' indeed "connects" all the terminals together, note that the channel graph G' clearly contains all the terminals. Furthermore, it is *weakly connected* in the sense that between every pair of terminals t and t', there is a sequence of terminals $t_i, i = 1, 2, \ldots, n$ with $t_1 = t$ and $t_n = t'$ such that every adjacent pair t_i and t_{i+1} either have exactly the same label set or have a path between them in

G'. In the example shown in Fig. 2*c*, the two terminals $\{\mathcal{R}_3\}$ and $\{\mathcal{R}_4\}$ are weakly connected.

The incremental construction of the channel graph is quite efficient. Using the union-find data structure [7], we can project a milestone and update the label sets (Algorithm 1, lines 8–9) in $O(|\mathcal{R}|)$ worst-case time, where $|\mathcal{R}|$ is the number of different labels and is equal to the number of roadmap components. A loose upper bound for updating G' (Algorithm 1, lines 10–12) is $O(|\mathcal{T}|)$, where $|\mathcal{T}|$ is the number of triangles in $\mathcal{T}$. In practice, the upper bound is rarely reached. The entire update takes little time, compared with other parts of the planner (see Section 6.1).

4.3 Sampling a Configuration

To generate a sample from $\mathcal{C}$ (Algorithm 1, line 1), we perform two simple steps. First, we sample a point $x \in \mathcal{F}_{\mathcal{W}}$ by picking a triangle $t \in \mathcal{T}$ according to the distribution $\pi_{\mathcal{T}}$ and then picking a point $x \in t$ uniformly at random. Next, we sample a configuration from $L_f(x)$. The details depend on the specifics of the robot's kinematics and are described below separately for rigid and articulated robots.

The configuration q of a rigid robot consists of a positional component q_x, which specifies the position of the robot's reference point in the workspace, and an orientational component q_θ, which specifies the orientation of the robot. To sample a configuration, we first pick q_θ uniformly at random. We then pick a point $x \in \mathcal{F}_{\mathcal{W}}$, as described above, and compute q_x so that at $q = (q_x, q_\theta)$, the robot's feature point f lies at x and the robot has orientation q_θ.

For an articulated robot, the configuration q specifies its joints parameters $q_1, q_2, \ldots$. Suppose that the feature point f lies in the ℓth link of the robot. To sample a configuration, we again pick a point $x \in \mathcal{F}_{\mathcal{W}}$ and then find the joint parameters $q_1, q_2, \ldots, q_\ell$ that place f at x by solving the robot's inverse kinematics (IK) equations. If IK has no solution, we must pick another x. If IK has more than one solution, we pick one at random. We then sample the other joint parameters $q_{\ell+1}, q_{\ell+2}, \ldots$ uniformly at random. Various improvements can be made to speed-up this process. For instance, we may restrict the sampling domain according to the reachability of each feature point.

5 Constructing the Ensemble Sampler

WCO is an ensemble sampler composed of many component samplers, which all have different distributions due to the different feature points used. If WCO uses a single feature point, i.e., a single component sampler, then to perform well, this component sampler must generate a good distribution everywhere in $\mathcal{C}$. In general, such a sampler is difficult to construct. Using multiple feature points simplifies the task. It is sufficient for a component sampler to work well in only part of $\mathcal{C}$, provided that several component samplers can be combined effectively to generate a distribution good in entire $\mathcal{C}$.

Algorithm 2 Workspace-based Connectivity Oracle.

1: Let p_i be the probability of picking a component sampler s_i. Initialize $p_i = 1/K$ for $i = 0, 1, \ldots, K-1$.
2: **for** $t = 1, 2, \ldots$ **do**
3: Pick a component sampler s_i from $S = \{s_0, s_1, \ldots, s_{K-1}\}$ with probability p_i.
4: Sample a new configuration q using the component sampler picked.
5: **if** a new milestone m is added to the roadmap $\mathcal{R}$ **then**
6: Update the distributions for each component sampler $s_i, i = 1, \ldots, K-1$.
7: Update the probabilities $p_i, i = 0, 1, \ldots, K-1$.

5.1 Combining Samplers through AHS

Recall from Section 3.1 that WCO uses a set of component samplers, $S = \{s_0, s_1, \ldots, s_{K-1}\}$, where s_0 is a special, uniform sampler and each $s_i, i = 1, 2, \ldots, K-1$ is based on a feature point of the robot. We combine the component samplers through AHS. Each sampler s_i has an associated weight w_i, which reflects the success of s_i according to its sampling history. The sampler s_i is chosen to run with probability p_i that depends on w_i. To adapt the ensemble distribution, we adjust the weights so that the component samplers with better performance have higher weights. See Algorithm 2 for an outline of the algorithm.

In iteration t of Algorithm 2, we choose s_i with probability

$$p_i = (1-\eta)\frac{w_i(t)}{\sum_{i=0}^{K-1} w_i(t)} + \frac{\eta}{K}, \tag{1}$$

where $w_i(t)$ is the weight of s_i in iteration t and $\eta \in (0, 1]$ is a small fixed constant. We use the chosen s_i to sample a new milestone m and assign to s_i a reward r that depends on the effect of m on the roadmap $\mathcal{R}$:

- The milestone m reduces the number of connected components of $\mathcal{R}$. In this case, m merges two or more connected components and improves its connectivity. We set $r = 1$.
- The milestone m increases the number of connected components of $\mathcal{R}$. In this case, m creates a new connected component and potentially improves the coverage of $\mathcal{R}$. We also set $r = 1$.
- Otherwise, $r = 0$.

We then update the weight of s_i:

$$w_i(t+1) = w_i(t) \exp\left((r/p_i)\eta/K\right). \tag{2}$$

Note that the exponent depends on the received reward r weighted by the probability p_i of choosing s_i. If a sampler is not chosen, then its weight remains the same as before: $w_i(t+1) = w_i(t)$. More details on AHS are available in [13].

Although there are many possible schemes for updating the weights, AHS has an important advantage. It can be shown that under suitable assumptions, the ensemble sampler generated by AHS is competitive against the best component sampler [13]. More precisely, the following competitive ratio holds:

$$R_{\max} - R \leq (\mathrm{e} - 1)\eta R_{\max} + \frac{K \ln K}{\eta}, \tag{3}$$

where R is the expected total reward received by the ensemble sampler and $R_{\max}$ is the total reward received by the best component sampler if it is always chosen to run. This result can be interpreted as saying that the ensemble sampler performs almost as well as the best component sampler, without knowing in advance which component sampler is the best. With some small variations on the scheme for updating the weights, one can also show that the modified ensemble sampler is competitive against any linearly weighted combination of component samplers, an even stronger result theoretically [2]. This is one reason why we choose AHS for combining component samplers.

The ensemble sampling distribution π is a linearly weighted combination of component distributions: $\pi = \sum_{i=0}^{K-1} p_i.\pi_i$, where p_i is the probability for choosing s_i and π_i is the distribution for s_i. Each WCO component sampler maintains its own workspace channels and only samples in the lifted channels in $\mathcal{C}$. Since the component sampling distributions are combined linearly, the intersections of lifted channels from different component samplers have higher probability of being sampled. These intersections contain the configurations that simultaneously place multiple feature points in their respective workspace channels. Thus, although each component sampler operates independently, the ensemble sampler automatically takes into account a robot's kinematic constraints on the feature points.

5.2 Choosing Feature Points

We must still choose a set of feature points. By Proposition 1, a collision-free path in $\mathcal{C}$ implies a collision-free path in $\mathcal{W}$ for every point in the robot. So, to infer the configuration-space path from the workspace paths of a finite set of feature points, these workspace paths must be *representative*. Ideally, the feature set is small, because we construct a component sampler for each feature point. A large number of component samplers increase both the difficulty of identifying the good ones through AHS and the computational cost.

Since small feature sets are preferred, we choose feature points to be spaced far apart. The reason is that feature points close together generate similar sampling distributions. Below we give specific choices for rigid and articulated robots. These heuristics worked well in our experiments, but more research is needed to develop a principled method for selecting feature points.

For a rigid robot, the feature set is the union of two point sets, CH and MP. CH consists of the vertices on the convex hull of the robot. MP contains a single point in the middle of the robot, e.g., the centroid. For an articulated robot, we take CH and MP with respect to each rigid link of the robot and take their union.

6 Implementation & Experimental Results

We implemented the new planner in C++, using the Qhull [3] library for workspace triangulation. We tested the planner and compared it with other

PRM planners. For each planner, we set the required parameters by performing 10 trial runs and choosing the values that gave good results. We ran the planner 30 times independently on each test environment. Each run was terminated once the query was solved. Note that we did not insert the query configurations into the roadmap as milestones or used any information from the query configurations to bias sampling. The experiments were conducted on a PC with a 3 GHz processor and 1 GB memory. The results reported below are the averages of 30 runs.

6.1 Comparison with other PRM planners

Since WCO uses workspace information and generates a dynamic sampling distribution, we compared it with PRM planners of these two classes. For the former, we compared with workspace importance sampling (WIS) [17]. For the latter, we compared with the original AHS [13], whose component samplers consist of a uniform sampler, several Gaussian samplers, and several bridge tests. These two planners were chosen because they are closely related to our work, and both have shown strong performance in narrow passages sampling. We also ran a PRM planner with uniform sampling to benchmark the difficulty of test environments.

We tested our planner on Tests 1–4. In all tests, WCO uses $\mathrm{CH} \cup \mathrm{MP}$ as the feature set. However, since Test 4 uses a common articulated robot with a fixed base and all rotational joints, the workspace displacement of the robot's links near the base is very limited. To improve computational efficiency, we consider only the feature points in the furthest link, which contains the end-effector and the large plate.

WIS uses a single feature point. In our tests, it used MP for a rigid robot (Tests 1–3) and MP of the furthest link for an articulated robot (Test 4).

Overall, WCO performed much better than the original AHS and WIS (see Table 1). Although WCO incurs the additional costs of processing the workspace geometry and updating the sampling distribution, it uses fewer milestones and places them in strategic locations. It improves the overall performance by reducing the total number of collision checks needed for sampling new milestones and connecting milestones in the roadmap. See Fig. 4 for an illustration of the differences between WCO and the other planners.

For comparison between WCO and the original AHS, it is especially interesting to consider Test 2. The start configuration s and the goal configuration g, when projected to $\mathcal{W}$, are very close; however, to go from s to g, the robot must go out of the narrow tunnel, reorient, and then go back to the tunnel again. Regardless of which feature point f is chosen, it may potentially mislead the planner, because all short paths in $\mathcal{W}$ between $P_f(s)$ and $P_f(g)$ give little information on the correct configuration-space path that connects s and g. Nevertheless, WCO performed well here, because it combines information from both $\mathcal{W}$ and $\mathcal{C}$. It dynamically updates the workspace channels, which provide information for connecting distinct roadmap components. By doing

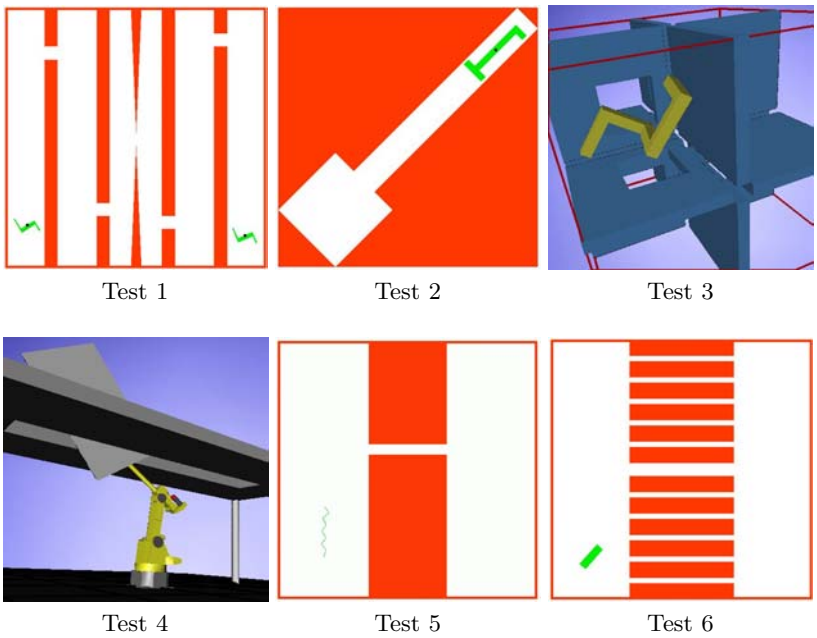


Fig. 3. Test 1: A 3-DOFs rigid-body robot moves from the lower left corner to the lower right corner by passing through five narrow openings. **Test 2**: A 3-DOFs rigid-body robot turns 180 degrees in a narrow deadend. **Test 3**: A 6-DOFs rigid-body robot must pass through 6 out of 7 narrow openings in order to answer the given query. **Test 4**: A 6-DOFs robot manipulator with its end-effector holding a large plate maneuvers through a narrow slot. **Test 5**: The robot is a planar articulated arm with a free-flying base. The dimensionality of $\mathcal{C}$ is increased by adding up to 8 links to the robot, resulting in a maximum of 10 DOFs. The robot must move through the narrow passage in the middle. **Test 6**: A 3-DOFs rigid-body robot moves from the left to the right wide-open space. It only fits through the passage in the middle. The number of false passages increases from 2 to 10.

so, as soon as $\mathcal{F}_c$ is covered adequately by $\mathcal{R}$, WCO can potentially identify the correct regions of $\mathcal{C}$ to sample.

Compared with WIS, WCO performed significantly better except for Test 2. This is expected, because WIS uses a single feature point (MP) and a static sampling distribution, which does not respond to changes in $\mathcal{R}$ and wastes lots of effort in sampling regions of $\mathcal{C}$ already well covered. In Test 3 and 4, WIS performed as badly as uniform sampling. The reason is that in both cases, the solution path requires the robot to rotate and translate in a coordinated fashion. A single feature, used by WIS, is incapable of representing such complex motion and generates a suitable sampling distribution for solving the problem. Furthermore, WIS does not update the distribution dynamically to improve the performance. In Test 2, to solve the query, the entire $\mathcal{F}_c$ must be adequately covered, whether a static or a dynamic sam-

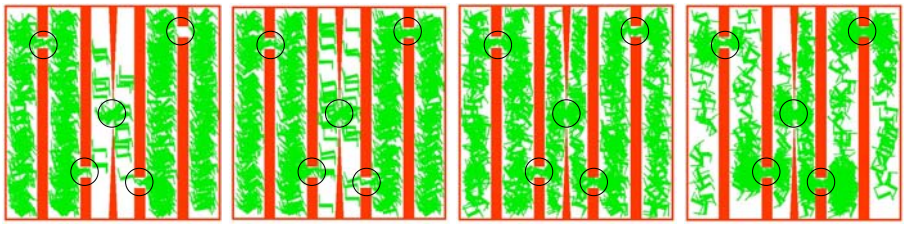


Fig. 4. 700 milestones generated by different planners. The pictures show that WCO increases the number of milestones in important regions that improve the connectivity of the roadmap, without generating too many unnecessary milestones in unimportant regions.

Table 1. Performance comparison of several PRM planners. All times are measured in seconds.

Sampler	**Test 1**					**Test 2**				
	T_{pre}	T_{upd}	T_{tol}	N_{mil}	N_{sam}	T_{pre}	T_{upd}	T_{tol}	N_{mil}	N_{sam}
Uniform			75.9	13,540	52,687			4.1	601	53,616
Original AHS			23.0	3,477	164,776			3.3	163	76,742
WIS	0.034		6.6	1,660	7,024	0.007		0.7	154	11,521
WCO	0.045	0.072	2.7	650	2,448	0.008	0.012	0.8	170	5,531
Sampler	**Test 3**					**Test 4**				
	T_{pre}	T_{upd}	T_{tol}	N_{mil}	N_{sam}	T_{pre}	T_{upd}	T_{tol}	N_{mil}	N_{sam}
Uniform			94.6	9,011	36,594			69.8	9,246	35,878
Original AHS			56.7	1,669	198,313			56.0	2,672	168,013
WIS	0.607		80.3	5,723	160,686	0.071		200.7	14,423	961,613
WCO	0.942	2.408	25.9	2,080	22,811	0.244	0.993	31.1	3,211	62,405

T_{pre}: time for triangulating $F_{\mathcal{W}}$. T_{upd}: time for updating component sampling distributions (Algorithm 1, lines 8–13). T_{tot}: total running time. N_{mil}: number of milestones required for answering the query. N_{sam}: number of configurations sampled.

Table 2. The effect of feature points on the running times of WCO. All times are measured in seconds. |CH| denotes the number of feature points in CH.

Test Env.	|CH|	MP	CH	CH ∪MP
Test 1	6	2.2	4.7	2.7
Test 2	5	1.3	0.7	0.8
Test 3	13	40.8	28.9	25.9
Test 4	8	154.3	62.0	31.1

pling distribution is used. WIS has an advantage, because it is simpler and does not incur the cost of updating the sampling distribution. Even so, the performance of WCO is comparable.

6.2 The Choice of Feature Points

Different feature points are good for sampling different regions of $\mathcal{C}$. To assess the benefits of multiple feature points, we ran WCO on Tests 1–4 with different feature sets. The experimental results show that although the performance of CH and MP varies across the test environments, the combined feature set CH ∪MP has consistently good performance (see Table 2). This corroborates the

theoretical result that the ensemble sampler is almost as good as the best component sampler and demonstrates the effectiveness of the AHS approach.

6.3 Other Experiments

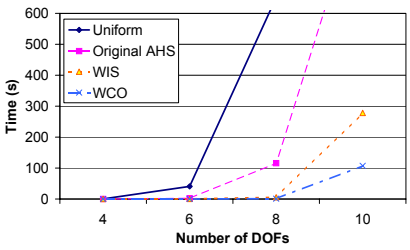


Fig. 5. The performance of WCO, as $\dim(\mathcal{C})$ increases (Test 5).

One concern of using workspace information to guide sampling in $\mathcal{C}$ is that as the dimensionality of $\mathcal{C}$ increases, workspace information becomes less useful. For this, we constructed a test environment with increasing dimensionality of $\mathcal{C}$ (Test 5). The results indicate that workspace information still has its merit (Fig. 5). The usefulness of workspace information does not directly depend on the dimensionality of $\mathcal{C}$ [11]. Instead, it depends more on whether there are workspace channels with good (n, p)-property.

One drawback of WCO is that it may find false workspace passages as channels, i.e., workspace passages that the robot can not pass through. It seems plausible that as the number of false passages grows, the performance of WCO will keep on worsening. So, we performed a test with an increasing number of false workspace passages (Test 6). The results indicate that this trend happens, but only to a limited extent (Fig. 6). The reason is that by construction, the number of channels in a channel graph G' is linear in the number of terminals in G'. Hence, after a certain limit, increasing the number of invalid workspace passages does not increase the number of channels or affect WCO's performance.

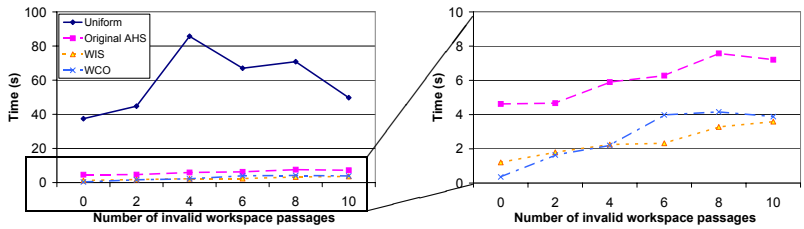


Fig. 6. The performance of WCO, as the number of false passages increases (Test 6).

7 Discussion

The WCO component samplers treat the feature points independently. Only the ensemble sampler implicitly accounts for the robot's kinematic constraints on the feature points. To further improve sampling efficiency, we may explicitly incorporate such constraints. We start with the simplest type, namely, the distance between a pair of feature points, as the distance between two feature points of a rigid robot or a rigid link of an articulated robot is fixed. One way of imposing such a constraint is to find the workspace channels for all

feature points, as explained in Section 4.2. However, instead of immediately updating the sampling distribution, we check whether two channels violate the distance constraint for the corresponding feature points and ignore the channels if they do. One way to check distance constraint is by enlarging each channel according to the given distance constraint and check whether the enlarged channels intersect. There are also many other types of kinematic constraints, which can possibly be incorporated through more sophisticated geometric features, such as line segments and surface triangles.

Although the above idea is simple and intuitively should improve WCO's performance, more thoughts and experiments are needed for finding good methods to incorporate and combine the information from multiple kinematic constraints. Treating each geometric feature as a "robot" and viewing WCO as multi-robot planning give us a spectrum of options. One extreme is decoupled planning with no coordination among robots. This approach is computationally efficient, because it ignores all constraints, but is not complete. WCO is somewhat similar to this approach, though it actually handles kinematic constraints implicitly. The other extreme is centralized configuration-space planning, which accounts for all the constraints, but is slower. Between the extremes, there are decoupled planning with pairwise coordination, global coordination, and prioritized planning. Each can translate to a method for incorporating kinematic constraints. For instance, the idea in the previous paragraph is similar to pairwise coordination.

8 Conclusion

This paper presents WCO, an adaptive sampling strategy for PRM planning. WCO is composed of many component samplers, each based on a geometric feature of a robot. Using the adaptive hybrid sampling approach, it combines information from both workspace geometry and sampling history to construct a dynamic sampling distribution. In our experiments, WCO significantly outperformed two recently proposed sampling strategies, which use, respectively, workspace information and dynamic sampling alone.

For future work, it would be interesting to extend WCO by relaxing the independence assumption on the component samplers and developing good methods to incorporate a robot's kinematic constraints *explicitly*. Viewing WCO as multi-robot planning is a promising direction. Another interesting extension is to use sampling history to improve the search for workspace channels, instead of just relying on the channel graph.

References

1. N. Amenta, S. Choi, and R. Kolluri. The power crust. In *Proc. ACM Symp. on Solid Modeling*, pages 249–260, 2001.
2. P. Auer, N. Cesa-Bianchi, Y. Freund, and R. Schapire. The nonstochastic multiarmed bandit problem. *SIAM J. Computing*, 32(1):48–77, 2002.
3. C. Barber, D. Dobkin, and H. Huhdanpaa. The quickhull algorithm for convex hulls. *ACM Trans. on Mathematical Software*, 22(4):469–483, 1996.

4. V. Boor, M. Overmars, and A. van der Stappen. The gaussian sampling strategy for probabilistic roadmap planners. In *Proc. IEEE Int. Conf. on Robotics & Automation*, pages 1018–1023, 1999.
5. B. Burns and O. Brock. Toward optimal configuration space sampling. In *Robotics: Science and Systems*, 2005.
6. H. Choset, K. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. Kavraki, and S. Thrun. *Principles of Robot Motion : Theory, Algorithms, and Implementations*, chapter 7. The MIT Press, 2005.
7. T. Cormen, C. Leiserson, R. Rivest, and C. Stein. *Introduction to Algorithms*. The MIT Press, 2nd edition, 2001.
8. M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. *Computational Geometry: Algorithms and Applications*. Springer, 2nd edition, 2000.
9. M. Foskey, M. Garber, M. Lin, and D. Manocha. A Voronoi-based hybrid motion planner. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots & Systems*, pages 55–60, 2001.
10. C. Holleman and L. Kavraki. A framework for using the workspace medial axis in PRM planners. In *Proc. IEEE Int. Conf. on Robotics & Automation*, pages 1408–1413, 2000.
11. D. Hsu, J.-C. Latombe, and H. Kurniawati. On the probabilistic foundations of probabilistic roadmap planning. *Int. J. Robotics Research*, 2006. To appear.
12. D. Hsu, J.-C. Latombe, and R. Motwani. Path planning in expansive configuration spaces. *Int. J. Comp. Geometry & Applications*, 9(4–5):495–512, 1999.
13. D. Hsu, G. Sánchez-Ante, and Z. Sun. Hybrid PRM sampling with a cost-sensitive adaptive strategy. In *Proc. IEEE Int. Conf. on Robotics & Automation*, pages 3885–3891, 2005.
14. L. Kavraki, M. Kolountzakis, and J.-C. Latombe. Analysis of probabilistic roadmaps for path planning. In *Proc. IEEE Int. Conf. on Robotics & Automation*, pages 3020–3025, 1996.
15. L. Kavraki, J.-C. Latombe, R. Motwani, and P. Raghavan. Randomized query processing in robot path planning. *J. Comp. & Syst. Sci.*, 57(1):50–60, 1998.
16. L. Kavraki, P. Švetska, J.-C. Latombe, and M. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *Int. J. Robotics Research*, 12(4):566–580, 1996.
17. H. Kurniawati and D. Hsu. Workspace importance sampling for probabilistic roadmap planning. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots & Systems*, pages 1618–1623, 2004.
18. S. LaValle and J. Kuffner. Randomized kinodynamic planning. *Int. J. Robotics Research*, 20(5):378–400, 2001.
19. M. Morales, L. Tapia, R. Pearce, S. Rodriguez, and N. Amato. A machine learning approach for feature-sensitive motion planning. In *Algorithmic Foundations of Robotics VI*, pages 361–376, 2004.
20. P. Švestka. On probabilistic completeness and expected complexity for probabilistic path planning. Technical Report UU-CS-1996-08, Utrecht University, Dept. of Information & Computing Sciences, Utrecht, the Netherlands, 1996.
21. J. van den Berg and M. Overmars. Using workspace information as a guide to non-uniform sampling in probabilistic roadmap planners. *Int. J. Robotics Research*, 24(12):1055–1071, 2005.
22. Y. Yang and O. Brock. Adapting the sampling distribution in PRM planners based on an approximated medial axis. In *Proc. IEEE Int. Conf. on Robotics & Automation*, pages 4405– 4410, 2004.