2009 IEEE International Conference on Robotics and Automation
Kobe International Conference Center
Kobe, Japan, May 12-17, 2009

An Unsupervised Adaptive Strategy for Constructing Probabilistic Roadmaps*

Lydia Tapia, Shawna Thomas, Bryan Boyd, and Nancy M. Amato
Parasol Lab., Dept. of Computer Science and Engineering, Texas A&M Univ., College Station, Texas, 77843-3112, USA
{*ltapia, sthomas, bcb2913, amato*}*@cse.tamu.edu*

***Abstract*— Since planning environments are complex and no single planner exists that is best for all problems, much work has been done to explore methods for selecting where and when to apply particular planners. However, these two questions have been difficult to answer, even when adaptive methods meant to facilitate a solution are applied. For example, adaptive solutions such as setting learning rates, hand-classifying spaces, and defining parameters for a library of planners have all been proposed. We demonstrate a strategy based on unsupervised learning methods that makes adaptive planning more practical. The unsupervised strategies require less user intervention, model the topology of the problem in a reasonable and efficient manner, can adapt the sampler depending on characteristics of the problem, and can easily accept new samplers as they become available. Through a series of experiments, we demonstrate that in a wide variety of environments, the regions automatically identified by our technique represent the planning space well both in number and placement. We also show that our technique has little overhead and that it out-performs two existing adaptive methods in all complex cases studied.**

I. Introduction

This paper explores the problem of finding a sequence of valid (collision-free) states that take a moving object, referred to as a robot, from an initial state to a goal state. These robot states, or configurations, are represented by a set of parameters that describe the placement and pose of the robot; these parameters are commonly known as the robot's degrees of freedom (DOF). This problem, often referred to as *motion planning* (MP), has application in domains such as robotics, gaming/virtual reality, computer-aided design (CAD), virtual prototyping, and bioinformatics.

Even in a known environment, MP is not an easy problem to solve. There is strong evidence that any complete planner will have complexity that grows exponentially in the DOF of the robot [21]. To combat this complexity, the Randomized Path Planner [2] was proposed to address MP with approximate, randomized methods. Remarkable results were achieved with this method, including solutions for previously unsolved MP problems. Subsequently, numerous randomized approaches have been proposed [1], [4], [5], [12], [14], [20]. The efficiency and effectiveness of these planners has been seen to be highly correlated with the planning space and the problem construction [8].

With such an extensive library of planning choices there are many decisions to be made when solving a particular problem instance. For example, which planner should be used, what parameters would be best, or even should a combination of planners be used? Adaptivity has been proposed as a solution [3], [7], [11], [17], [19], [22]. While significant improvement has been shown over non-adaptive approaches, these methods have all been seen to have serious drawbacks that limit their usefulness such as requiring significant user intervention (e.g., manual classifications of training instances for supervised machine learning methods, parameter tuning to set learning rates and learning weights) or restricting the types of problems they are able to solve.

In this paper, we explore a combination of two previously introduced adaptive methods, the feature-sensitive motion planning framework [17] and the Hybrid PRM planner [11]. We use unsupervised learning to minimize user intervention typically required for manual training and parameter tuning, one of the main drawbacks of the previous approaches. Our unsupervised adaptive strategy (UAS) first uses the feature-sensitive framework to identify regions, except it replaces the tedious manual creation and labeling of training examples with unsupervised clustering. It then applies the adaptive strategy from Hybrid PRM in each of these semi-homogeneous regions. UAS assigns and adjusts sampler rewards based on the structural improvement the sampler makes to the roadmap [18]. Our experimental results demonstrate that the combination of methods better automates, with minimal human intervention, the questions of where and when to apply which planning solutions. In these complex spaces, we compare the contribution of each of these adaptation methods individually with the combined planner. We show that in a variety of environments, the regions automatically identified by UAS represent the planning space well both in number and placement. Our results show that UAS has low overhead and that it out-performs two existing adaptive methods in all complex cases studied.

* This research supported in part by NSF Grants EIA-0103742, ACR-0081510, ACR-0113971, CCR-0113974, ACI-0326350, CRI-0551685, CCF-0833199, CCF-0830753, by the DOE, Chevron, IBM, Intel, HP, and by King Abdullah University of Science and Technology (KAUST) Award KUS-C1-016-04. Tapia supported in part by an NIH Molecular Biophysics Training Grant (T32GM065088), a PEO scholarship, and by a Department of Education (GAANN) Fellowship. Thomas supported in part by an NSF Graduate Research Fellowship, a PEO scholarship, a Dept. of Education Graduate Fellowship (GAANN), and an IBM TJ Watson PhD Fellowship.

II. PRELIMINARIES

A robot is a movable object whose position and orientation can be described by n parameters, or degrees of freedom (DOFs), each corresponding to an object component (e.g., object positions, object orientations, link angles, link displacements). Hence, a robot's placement, or configuration, can be uniquely described by a point $(x_1, x_2, ..., x_n)$ in an n dimensional space (x_i being the ith DOF). This space, consisting of all possible robot configurations (feasible or not) is called *configuration space* (C-space) [16]. The subset of all feasible configurations is the *free C-space* (C-free), while the union of the unfeasible configurations is the *blocked C-space* (C-obstacles). Thus, the MP problem becomes that of finding a continuous trajectory for a point in C-free connecting the start and the goal configurations. In general, it is intractable to compute explicit C-obstacle boundaries, but we can often determine whether a configuration is feasible or not quite efficiently, e.g., by performing a collision detection (CD) test in the *workspace*, the robot's natural space.

Randomized motion planners explore C-space and produce a data structure containing feasible configurations and some information about the connectivity of C-free. One of the most notable planners, PRMs [12], builds a roadmap (graph) of the free C-space. The first phase in this process, *node generation*, is where collision-free configurations are sampled and added as nodes to the roadmap. In the second phase, *node connection*, neighboring nodes are selected by a *distance metric* as potential candidates for connection. Then, simple *local planners* attempt connections between the selected nodes. Successful connections are roadmap edges.

Although the initial PRMs were successful in solving many problems previously thought unsolvable, they were not successful in problems where the solution path must pass through a narrow passage in the C-space. In order to address this deficit, many PRM variants have been introduced. For example, OBPRM [1] generates samples near C-obstacle surfaces by first generating a random sample and searching along a random direction until the sample's collision state changes. Another variant, Gaussian PRM [5], generates pairs of samples that are a distance d apart, where d has a Gaussian distribution, until one sample is collision-free and the other is not, and retains the free sample as a roadmap node. Many other heuristics have been proposed [4], [14], [20]. The performance of these methods is dependent on the problem instance, e.g., OBPRM performs better in cluttered areas and uniform random sampling is quite efficient in open areas.

III. RELATED WORK

Adaptive Planning Methods. This section provides an introduction to many of the adaptive planning methods that have been proposed. The methods are summarized in Table I. A discussion of their strengths and weaknesses for adaption is given below.

Feature Sensitive Motion Planning Framework. This approach was introduced as a method that used machine learning to characterize and partition a planning problem [17]. In this approach, the planning space is recursively subdivided until a machine learning method is able to classify a subdivision as appropriate for a planner from a given library. This topology mapping may be defined in either workspace or C-space. The strength of this method lies in its ability to identify a model of a problem's topology that make certain regions appropriate for certain planners. However, other than recursive subdivision calls, it is not able to adapt planner applications over time. Another drawback of this approach is that it requires a mapping of samplers to regions, typically generated by machine learning techniques that require an "expert" to label hundreds of examples of training data. Such a mapping must be repeated as new planners are developed.

Hybrid PRM. Here, a reinforcement-learning approach provides sampler adaption by selecting a node generation method that is expected to be the most effective at the current time in the planning process [11]. Variations of this method that changed the learning process [23] and employed workspace information [13] have also been explored. The theory behind this method is that as the space becomes oversampled by simple samplers, more complex samplers will be able to take over. However, these samplers are applied globally over the whole problem, and the features of the planning space, such as topology, are not used when deciding where to apply the selected method. Also, there are many parameters that need to be set for optimal application of the Hybrid PRM method such as initial sampler weights, sampler reward/cost assignment, how weights are adjusted during learning, and how long before beginning adaptation, to name a few. As new samplers become available, it is straightforward to add them to Hybrid PRM.

Information Theory Approaches. Burns and Brock [6], [7] demonstrated the applicability of ideas from information theory (e.g., information gain and entropy) to guide sampling to regions where it is predicted to be useful. This guidance helps explore the spatial constraints of the space, and the implicit modeling of spatial regions helps guide future sampling.

RESAMPL [22], uses local region information (e.g., entropy of neighboring samples) to make decisions about both how and where to sample, which samples to connect together, and to find paths through the environment. This use of spatial information about the planning space enables RESAMPL to increase sampling in regions identified as "narrow" and decrease sampling in regions identified as "free".

Workspace Adaptation Methods. Many methods have been proposed to explore the impact of adaptation in response to the features of the planning workspace. A recent adaptation of the Hybrid PRM method [13] uses workspace information, extracted from a cell decomposition, to define locations where samplers should be applied. Another workspace-based approach applies the watershed method (previously applied in image processing) to identify narrow passageways in the workspace [3]. After such features are identified, the planning can be adapted based on the characterization of the region.

	Method	Characteristics				
		User Intervention	**Topology Adaption**	**Sampler Adaption**	**C-space Type**	**Add New Sampler**
	UAS	*little*	*yes, modeled*	*yes*	*any*	*easy*
	Traditional PRM	little	none	none	any	N/A
	Basic Feature Sensitive MP [17]	supervised planner training	yes, modeled	yes, fixed mapping	any	difficult
	Hybrid PRM [11]	manual parameter tuning	none	yes	any	easy
Information	IG/Entropy-based [6], [7]	manual parameter tuning	yes, implicit	N/A	any	N/A
Theory-Based	RESAMPLE [22]	manual parameter	yes, implicit	N/A	any	N/A
Workspace	Workspace Hybrid PRM [13]	little	yes, mapped	yes	restricted	easy
Adaption-Based	Watershed-based Method [3]	manual parameter tuning	yes, mapped	yes, fixed mapping	restricted	N/A

TABLE I

COMPARISON OF ADAPTIVE METHODS. "USER INTERVENTION" REFERS TO THE AMOUNT OF USER INPUT NEEDED FOR SUCCESSFUL APPLICATION. "TOPOLOGY ADAPTION" REFLECTS IF A METHOD IS ABLE TO MAP OR MODEL THE PLANNING SPACE. "SAMPLER ADAPTATION" REFERS TO WHETHER DIFFERENT PLANNERS CAN BE APPLIED DURING THE PLANNING PROCESS. "C-SPACE TYPE" CONSIDERS THE TYPES OF C-SPACES THAT CAN BE ADDRESSED BY THE METHOD. IF NEW SAMPLING METHODS CAN EASILY BE BE APPLIED, IT IS REFLECTED IN "ADD NEW SAMPLER".

While these approaches rely heavily on spatial characteristics in order to decide where to apply a planner, they do not consider the change in the topology that is discovered as a space is explored. Their performance also degrades in more complex problems (different C-space types) where difficult (e.g., narrow) regions of C-space can no longer be identified from difficult regions of the workspace (such as with articulated linkages and other constrained robots).

Metrics for Planner Performance. Adaptive MP strategies require metrics to evaluate the performance of planning approaches. While many common metrics have been used for evaluation (e.g., solution of a query, time, CD counts), it is often still difficult to get a clear measure of planner performance. Methods that require a discretization of the planning space have been proposed [8]. In the problems studied here, we applied a group of metrics that have been explored on non-discrete spaces in order to classify the contribution of samples produced by planners [18]. We use these metrics, defined below, for our planner evaluation.

If two configurations, q_1 and q_2, can be connected by a sequence of valid motions, they are considered *visible* to each other. For example, the straight-line local planner will decide that q is visible to q' if the straight segment from q to q' is composed of only valid configurations. In [18], a *visibility ratio* is assigned to each configuration to approximate the visibility of a single configuration to its neighbors. This ratio is defined in terms of the number of successful connections over the number of connection attempts involving that configuration.

In [18], a method is introduced that provides a classification for every node as it is inserted into the roadmap. A node is classified as: *cc-create* if it cannot be connected to any existing roadmap component, *cc-merge* if it connects to more than one connected component in the roadmap, *cc-expand* if it connects to exactly one component in the roadmap and satisfies a visibility expansion criterion as defined in [23], and *cc-oversample* otherwise.

IV. METHODS

One of the major drawbacks of the feature-sensitive framework and Hybrid PRM is the reliance on manual intervention and sensitivity to parameter tuning. In our method outlined in Algorithm 1, we combine the two approaches and eliminate much of this user burden. First, we replace the requirement of manual training data creation and labeling with unsupervised learning for region identification. Then, we exchange the manual mapping of region types to samplers with the adaptive strategy provided in Hybrid PRM. This allows our method to continue to perform well as new sampling strategies are developed without requiring any additional input from the user. Also, the homogeneity of the region allows Hybrid PRM to quickly assess the space and select optimal samplers. This property reduces sensitivity to parameters such as learning rate and number of samplers. Finally, we use roadmap structure improvement metrics to automatically assign rewards/costs to the various samplers instead of tuning those parameters by hand, thus further eliminating parameter sensitivity. In the following subsections, we describe each step of the algorithm in more detail.

A. Unsupervised Region Identification

To identify semi-homogeneous regions in the environment, we first construct a small roadmap using each of the different samplers. Next, we partition the nodes in the roadmap into c clusters using k-means clustering, for a given number of clusters c. There are many features that have been previously explored for region identification [17]. In the results shown here, clustering is based on a set of features that are independent of robot type: visibility, X-position, Y-position, and Z-position. Then, we define each region as the bounding box of each node set. Due to the use of positional values as features, clusters may result in overlapping regions.

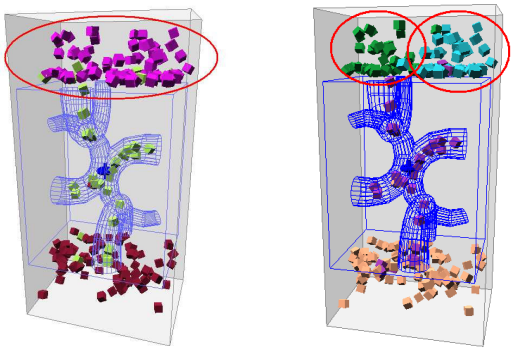
The choice of number of clusters c is often difficult to select under the k-means framework. In our motion planning application, the use of positional values (X, Y, Z) as features only complicates this selection because additional clusters will always provide an improvement. For example, consider

Algorithm 1 Unsupervised Adaptation Method (UAS).

Input: An environment E, a query Q, a set of samplers S, and an increment size m.
Output: A roadmap R.

```
Identify (homogeneous) regions for planning in E.
Set Pr(s) = 1/|S| for each sampler s ∈ S.
while Q not solved with R do
    for all regions identified in E do
        for i = 1 .. m do
            Select sampler s according to probabilities Pr.
            Generate a sample with s and add it to R.
            Update Pr(s) according to the structural improvement of R.
        end for
    end for
end while
Return R.
```

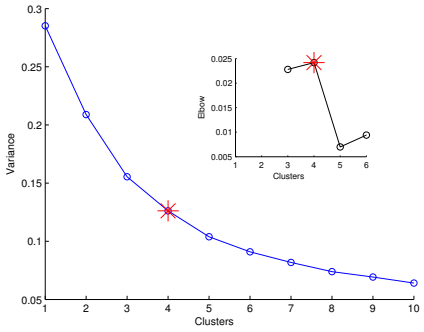
the set of samples in Figure 1 for the Maze environment (Figure 3(a)). Partitioning the samples into 3 clusters (a) intuitively splits the environment into a single constrained region in the middle and two free regions on each end (samples are colored according to cluster membership). Increasing the number of clusters to 4 (b) begins to partition the already homogeneous regions. For example, the one circled region in (a) becomes the two circled regions in (b).



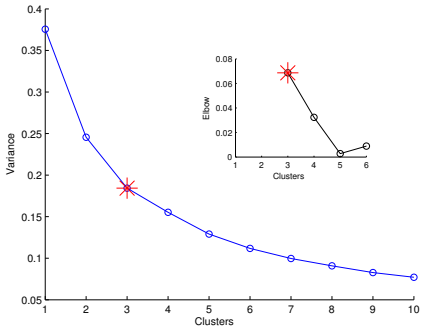
(a) 3 clusters (b) 4 clusters

Fig. 1. Clustering based on a small roadmap in the Maze.

In order to overcome this limitation and automate the process, we examine the percentage of the variance explained, $(\sum_{i=1}^{k} \sigma_i^2)/\sigma^2$ where σ^2 is the variance of the data set, for each c. Recall that the data set is defined by the features used for clustering (Section IV-A). We select the c that maximizes the second derivative of this function. This is commonly known as the elbow criterion [10], [15]. Intuitively, this criterion selects c such that adding additional clusters does not add sufficient information. In Figure 2, the average variance is plotted against the number of clusters for two environments, the L-tunnel and Maze. The "elbow" is indicated with a red star in each plot, and its calculation is shown in the inset. As noted above, the Maze is best represented by 3 clusters (see Figure 1(a)) instead of 4 (Figure 1(b)) which splits a region of homogeneous visibility.



(a) L-tunnel (elbow at 4 clusters)



(b) Maze (elbow at 3 clusters)

Fig. 2. Change in variance as the number of clusters increases.

B. Unsupervised Sampler Reward Assignment

Another area typically requiring user intervention is tuning the learning rate and the rewards/costs for each sampler. In addition, as new samplers are added to the set, these values may have to be adjusted. Similar to Hybrid PRM [11], we use an exponential function to update sampler rewards. However, we define the individual sample rewards differently. We reward samplers on the range [0,1] as follows: cc-create and cc-merge nodes have a reward of 1 (since they always improve the roadmap) and all other nodes (e.g., cc-expand and cc-oversample) have a reward of $e^{-\alpha v_t{}^2}$, where v_t is the visibility ratio of the node generated at time step t and $\alpha > 0$. This gives nodes with low visibility a large reward and nodes with high visibility a small reward. We found that $\alpha = 4$ works well in practice because it weights the rewards on either end of the visibility spectrum (i.e., nearly 1 for the lowest visibilities and nearly 0 for the highest). We assign equal weight to past performance and random selection when setting sampler probabilities.

V. Experiments

In this section, we explore the performance of two existing adaptive planning strategies, the Feature Sensitive Motion

Planning Framework [17] and Hybrid PRM [11], and compare them to our unsupervised adaptive strategy, UAS. We study both rigid body problems and articulated linkages in environments of varying heterogeneity.

A. Experimental Setup

We implemented all planners using the C++ motion planning library developed by the Parasol Lab at Texas A&M University. RAPID [9] is used for collision detection computations. Connections are attempted between k "nearby" nodes according to some distance metric; here we use $k = 20$, C-space Euclidean distance, and a simple straight-line local planner. The Feature Sensitive Motion Planning Framework is implemented as described in [17]. For a given region, we select a sampler based on the region's average visibility. Our experiments map the regions as follows: in low visibility regions OBPRM is chosen, in medium visibility regions we use Gaussian sampling, and in high visibility regions uniform random sampling is used. Hybrid PRM is implemented as discussed in [11]. Sampler probabilities are initialized to the uniform distribution.

We explore several different rigid body and articulated linkage environments of varying topology, see Figure 3. In these environments, the witness queries have been designed to force the robot to traverse the entire problem space. This ensures that they capture the problem complexity.

- **Maze**: This environment consists of two open areas connected by a maze of narrow tunnels. The tunnels vary from smaller than the robot (impassable) to just slightly wider than the robot. The robot is in the shape of a spinning top, and it is often very difficult for a planner to find feasible motions in the maze.
- **L-Tunnel**: The L-shaped robot must rotate and translate in between three large obstacles to traverse an L-shaped maze.
- **Hook**: The Hook environment has two walls with slits between them. The robot, a hook, must rotate and translate between the two slits to move from one side of the environment to the other.
- **Cluttered**: The Cluttered environment has 27 randomly placed cube obstacles. The robot, a box, must traverse from one side of the environment to the other. This environment was designed to be homogeneous.
- **Regions**: The Regions environment has four distinct regions: a long narrow tunnel followed by a cluttered region with free regions on either side. The robot, a 4 link articulated linkage, must elongate itself to pass through the tunnel and then change to a more compact form to navigate the cluttered region.
- **Walls**: The Walls environment has several chambers with small holes connecting them. Each chamber is either cluttered or free. The robot must traverse each chamber to solve the query.

To have a variety of sampling techniques in our available library of samplers, we have chosen samplers known to work well with varied amounts of obstacles. The samplers selected are: uniform random sampling [12], Gaussian sampling [5], and OBPRM [1]. For Gaussian sampling, we use two values of the Gaussian distance d: the robot's minimum diameter, r_d, and $2r_d$.

We use the following metrics to evaluate planner performance: ability to solve a user-defined witness query, changes in types of nodes generated (e.g., cc-create, cc-merge, cc-expand, and cc-oversample), and collision detection calls as a measure of time.

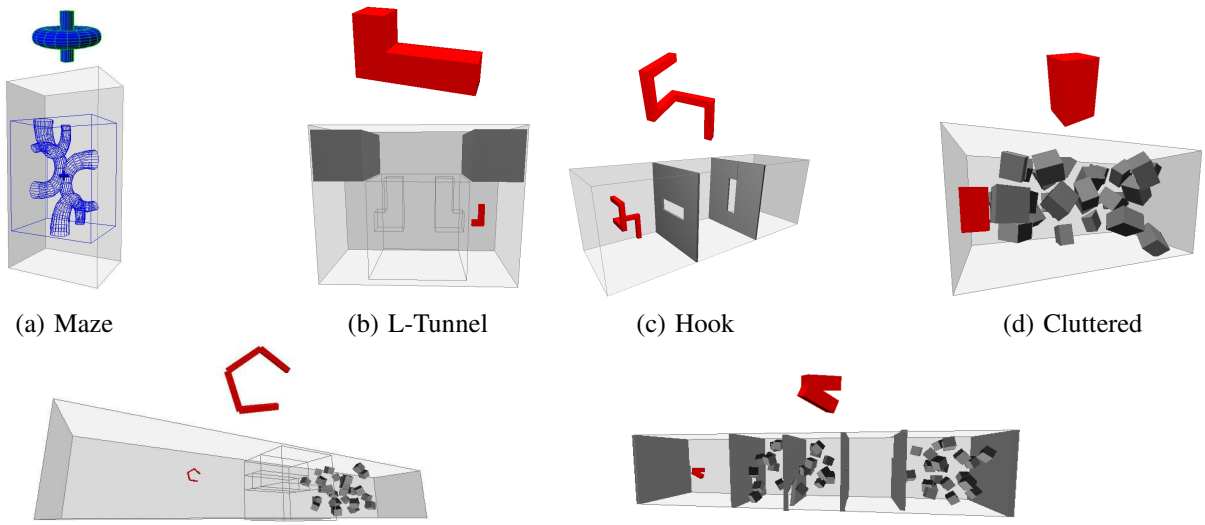
B. Cluster Study

As demonstrated in [17], the training set (initial roadmap) must be cheap, fast, and represent the main features of the space. To define a good initial roadmap size, we chose to construct our small roadmap with fixed proportions of uniform random, Gauss, and OBPRM nodes. Other sampling techniques could be used, but this set mirrored the planners used for full map-building. We also used a low connection parameter ($k = 5$) to reduce cost. For example, roadmaps of 100 to 1000 nodes required from 48,664 to 400,848 CD calls. For the experiments here and those done in previous studies [17], low values of k capture the topology of the space. A larger value of k is used when maps are generated in the regions (Section V-A).

After the roadmaps were constructed, we studied the effect of the number of nodes on cluster quality. Recall that while all features, positional and visibility, are used in clustering, the positions are used to define region boundaries and visibility is used to define region homogeneity. For example, in the Maze environment, we found that with 200 nodes, 3 clear clusters were formed (Figure 1(a)). When the training set size was reduced to 100, three clear clusters were still able to be formed. However, the min/max visibility ranges that each cluster represented became more encompassing (changing one cluster from 0.40, 1.0 to 0.25, 1.0). On the other hand, increasing the number of nodes to 400 changed the cluster to represent visibilities from 0.38 to 1.0.

The visibility changes made two clear facts. First, an increased number of training samples increases the chance of finding clear, homogeneous regions. This was reflected in the difference in min/max ranges for different data set sizes. Second, while more samples are useful, they are not necessary to obtain good regions. This was made clear by the average visibility values, variance of visibility, and size and placement of the regions across all data set sizes. These results are shown in Table II, where clusters are grouped by the relative positions of their members. Due to these facts, we chose a low, set number of samples (200) for clustering in the environments shown. However, as new problems are explored, the effects of sample size on cluster identification can be evaluated as shown.

For each environment, the number of clusters was identified using the elbow criterion described in Section IV-A. Given a single training roadmap, the clustering was run with



(a) Maze (b) L-Tunnel (c) Hook (d) Cluttered

(e) Regions (f) Walls

Fig. 3. Rigid body (a-d) and articulated linkage (e,f) environments studied. The robot must travel from one end to the opposite end.

1 to 10 clusters. After this, the "elbow" of the cluster variance was used to identify the number of clusters.

Cluster #	Data Set Size	Size (%)	Visibility			
			Avg.	Std. Dev.	Min	Max
0	100	27	0.241	0.191	0.000	0.600
	200	18	0.192	0.169	0.000	0.500
	300	20	0.314	0.210	0.000	0.667
	400	21	0.338	0.202	0.000	0.667
1	100	42	0.813	0.242	0.250	1.000
	200	43	0.907	0.158	0.375	1.000
	300	48	0.885	0.189	0.375	1.000
	400	47	0.886	0.189	0.400	1.000
2	100	31	0.857	0.177	0.500	1.000
	200	40	0.813	0.211	0.400	1.000
	300	32	0.915	0.141	0.500	1.000
	400	31	0.943	0.111	0.667	1.000

TABLE II

CLUSTER STATISTICS ON THE MAZE ENVIRONMENT USING DIFFERENT DATA SET SIZES. CLUSTERS ARE GROUPED BY RELATIVE POSITION.

C. Performance Study

We compare the performance of a Basic Feature Sensitive Motion Planning Framework, Hybrid PRM, and the new UAS in each environment. We allowed each planner to attempt to solve the query with at most 5000 nodes for all environments except L-Tunnel in which we allowed 8000 nodes. Table III provides the overall results, averaged over 5 runs. For Basic Feature Sensitive MP and UAS, which require clustering computation in addition to map building, we break down the statistics into a clustering phase and a mapping phase. In most of the environments studied, the three planners were able to solve the queries 100% of the time. The the environments where solution wasn't possible included: the L-Tunnel environment, where Feature Sensitive failed to solve the query 80% of the time, and the Region environment, where Hybrid PRM never solved the query.

In general, we find that having region identification improves overall performance by allowing a sampler to focus on a particular region. The combined adaptation provided by UAS out-performs both the Basic Feature Sensitive MP and Hybrid PRM. Consider the Maze environment: clustering partitions the environment into 3 distinct regions, two with high visibility where the robot is unobstructed and one with low visibility where the robot must traverse a narrow passage (see Figure 1(a)). By restricting a sampler's focus, we increase its probability of sampling the narrow passage. Hybrid PRM alone requires over twice as many nodes than methods employing Basic Feature Sensitive MP. A similar trend occurs in the Hook environment as well.

Figure 4 demonstrates why this focus improves planner performance. It shows the types of nodes created in an example run of the Maze environment for (a) Basic Feature Sensitive MP alone and (b) UAS adaptation. The query in these two runs is solved with 2102 and 1040 nodes, respectively. The addition of region identification dramatically reduces the number of unproductive cc-oversample nodes and increases the number of productive nodes (e.g.,. cc-create, cc-merge, and cc-expand). Thus, the planner in (b) is able to solve the query using half as many nodes as the one in (a).

We also find that unsupervised sampler selection as provided by UAS relieves the burden of having to identify which sampler to use in a given region without paying much of a performance penalty, if any at all. In most instances, unsupervised planner adaptation performs better than the manually mapped planners to region features as done in the Basic Feature Sensitive MP. UAS has the advantage of being more extensible to new sampling strategies because it does

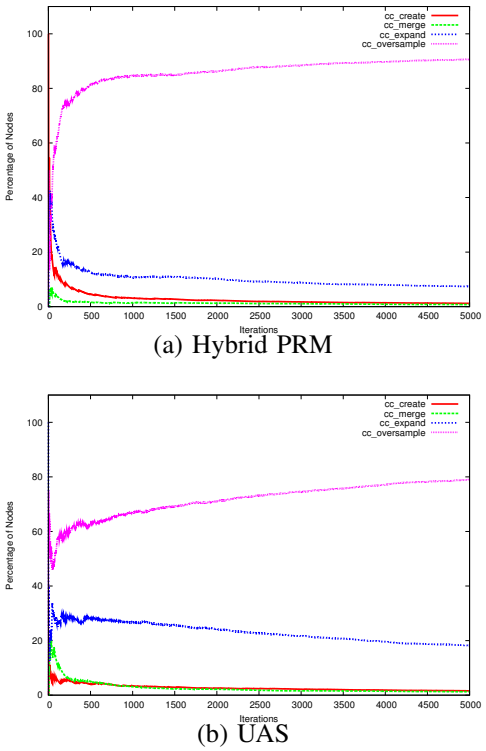
Maze Environment			
Method		**Nodes Required**	**CD Calls**
Basic Feature Sens. MP	clustering	200	41850
	mapping	1022	445571
	totals	1222	487421
Hybrid PRM		3189	2572757
UAS	clustering	200	41850
	mapping	854	708127
	totals	1054	749977
L-Tunnel Environment			
Method		**Nodes Required**	**CD Calls**
Basic Feature Sens. MP	clustering	200	26167
	mapping	3253*	1989134*
	totals	3453*	2015301*
Hybrid PRM		3557	2091587
UAS	clustering	200	26167
	mapping	3395	2027709
	totals	3595	2053876
Hook Environment			
Method		**Nodes Required**	**CD Calls**
Basic Feature Sens. MP	clustering	200	7067
	mapping	1142	208837
	totals	1342	215904
Hybrid PRM		1789	268319
UAS	clustering	200	7067
	mapping	1125	135116
	totals	1325	142183
Cluttered Environment			
Method		**Nodes Required**	**CD Calls**
Basic Feature Sens. MP	clustering	200	12465
	mapping	1761	192636
	totals	1961	205101
Hybrid PRM		2079	395380
UAS	clustering	200	12465
	mapping	2233	474975
	totals	2433	487440
Region Environment			
Method		**Nodes Required**	**CD Calls**
Basic Feature Sens. MP	clustering	200	65842
	mapping	761	485537
	totals	962	551379
Hybrid PRM		Not Solved	Not Solved
UAS	clustering	200	65842
	mapping	485	394613
	totals	685	460455
Walls Environment			
Method		**Nodes Required**	**CD Calls**
Basic Feature Sens. MP	clustering	200	23276
	mapping	2875	566582
	totals	3075	589858
Hybrid PRM		3281	1264543
UAS	clustering	200	23276
	mapping	2293	663891
	totals	2493	687165

TABLE III

PERFORMANCE COMPARISON OF BASIC FEATURE SENSITIVE MP , HYBRID PRM , AND UAS ON DIFFERENT ENVIRONMENTS TO SOLVE THE QUERY. RESULTS ARE AVERAGED OVER 5 RUNS. *RESULTS FOR SPATIAL ADAPTATION IN THE L-TUNNEL SOLVED THE QUERY 20% OF THE TIME AND ARE ONLY AVERAGED OVER SUCCESSFUL RUNS.

not need the intervention of an "expert" to determine which regions new samplers should be applied in.

Additionally, we find that adding unsupervised sampler



(a) Hybrid PRM

(b) UAS

Fig. 4. Comparison of node types created in an example run in the Maze environment. UAS dramatically reduces the number of unproductive cc-oversample nodes and increases the number of productive nodes (e.g.,. cc-create, cc-merge, and cc-expand).

adaptation to unsupervised region identification can overcome a sub-optimal sampler choice dictated by the fixed sampler/feature mapping. For example, in the L-Tunnel environment, only spatial adaptation failed to solve the query. Clustering successfully identified the central obstacle containing the two narrow passages and thus focused sampling inside them (see Figure 5). However, OBPRM was chosen because the cluster had low visibility. OBPRM had the unfortunate tendency to generate many nodes deep inside the narrow passages and few nodes near the openings. Thus, the planner was unable to find a path from inside a passage outside to a free area. Unsupervised sampling adaptation inside the region was able to overcome this by switching the sampler selection from OBPRM to Gaussian sampling.

Even in more complex planning spaces, such as Regions (4 link robot) or Walls (2 link robot), UAS is able to outperform Basic Feature Sensitive MP and Hybrid PRM. For example, in the region environment, Hybrid PRM was unable to solve the query. However, the topology adaptation provided by Basic Feature Sensitive MP and UAS allowed them to solve the problem 100% of the time. In the Region environment, UAS solved the query with fewer nodes and fewer CD calls. In the Maze environment, UAS was able to solve the query with fewer nodes and fewer CD calls than Hybrid PRM. Even though Basic Feature Sensitive MP was manually trained to

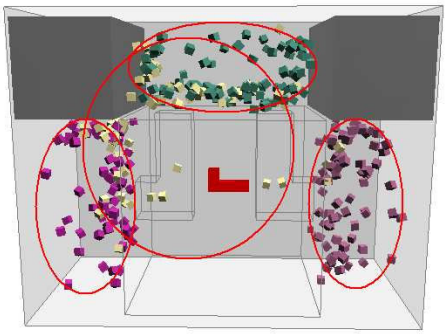


Fig. 5. Regions identified by clustering in the L-Tunnel. The central low visibility region (yellow) successfully identifies the two narrow passages. However, it does not include the opening on the right passage creating challenges for spatial adaption alone.

use certain planners in regions with certain features, it solved the query with only a few less CD calls and more nodes than UAS.

Another interesting test case was the Cluttered environment where twenty seven cube obstacles are randomly placed in a small space. Because of the homogeneous nature of this space, it would be expected that a single sampling method might be suited to perform well. We compared the performance of the three methods in this homogeneous problem. The first distinct result was that the elbow criterion dictated that there were 3 clusters. As defined previously, this is the minimum number of clusters that are able to be identified with this method. Also, the resulting clusters were divided mostly by positional values. This was expected due to the homogeneity of the space. The second distinct result was that Basic Feature Sensitive MP outperformed Hybrid PRM who outperformed UAS. This also was not surprising. Basic Feature Sensitive MP was allowed to use a single method that was known to perform well in low-visibility spaces, Hybrid PRM had to learn the single method to apply, and UAS had to learn this method in each of the three regions. However, this overhead (about 500 nodes and 100000 CD calls) is very little considering the amount of automation provided by UAS.

VI. Conclusion

This work demonstrates how combining both the topology adaptation and sampler adaptation over the planning process helps answer the "where" and "when" questions of applying different sampling strategies. We have shown a new strategy that simplifies the process of adaption, requires minimal user intervention, can be applied to any MP problem, and supports both topological and sampler adaptation. No previous method has all these characteristics. Augmenting an approach with topology adaptation enhances performance by focusing sampling in important or difficult regions of the problem. Correspondingly, adapting the sampler during the problem solution removes the burden of mapping samplers to region features and by overcoming sub-optimal choices made by such a model. Finally, the results presented here can be easily extended to C-space partitioning as was done in [19].

References

[1] N. M. Amato, O. B. Bayazit, L. K. Dale, C. V. Jones, and D. Vallejo. OBPRM: An obstacle-based PRM for 3D workspaces. In *Robotics: The Algorithmic Perspective*, pages 155–168, Natick, MA, 1998.

[2] J. Barraquand and J. C. Latombe. Robot motion planning: A distributed representation approach. *Int. J. Robot. Res.*, 10(6):628–649, 1991.

[3] J. Berg and M. Overmars. Using workspace information as a guid to non-uniform sampling in probabilistic roadmap planners. *Int. J. Robot. Res.*, 24(12):1055–1072, 2005.

[4] R. Bohlin and L. E. Kavraki. Path planning using Lazy PRM. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 521–528, 2000.

[5] V. Boor, M. H. Overmars, and A. F. van der Stappen. The Gaussian sampling strategy for probabilistic roadmap planners. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, volume 2, pages 1018–1023, May 1999.

[6] B. Burns and O. Brock. Sampling-based motion planning using predictive models. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2005.

[7] B. Burns and O. Brock. Toward optimal configuration space sampling. In *Proc. Robotics: Sci. Sys. (RSS)*, 2005.

[8] R. Geraerts and M. H. Overmars. Reachablility-based analysis for probabilistic roadmap planners. *Robotics and Autonomous Systems*, 55:824–836, 2007.

[9] S. Gottschalk, M. C. Lin, and D. Manocha. OBB-tree: A hierarchical structure for rapid interference detection. *Comput. Graph.*, 30:171–180, 1996. Proc. SIGGRAPH '96.

[10] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2001.

[11] D. Hsu, G. Sánchez-Ante, and Z. Sun. Hybrid PRM sampling with a cost-sensitive adaptive strategy. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 3885–3891, 2005.

[12] L. E. Kavraki, P. Švestka, J. C. Latombe, and M. H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Trans. Robot. Automat.*, 12(4):566–580, August 1996.

[13] H. Kurniawati and D. Hsu. Workspace-based connectivity oracle - an adaptive sampling strategy for prm planning. In *Algorithmic Foundation of Robotics VII*, pages 35–51. Springer, Berlin/Heidelberg, 2008.

[14] S. M. LaValle and J. J. Kuffner. Randomized kinodynamic planning. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 473–479, 1999.

[15] L. Lieu and N. Saito. Automated shapes discrimination in high dimensions. *Proc. of SPIE*, 6701 – Wavelets XII(67011W), 2007.

[16] T. Lozano-Pérez and M. A. Wesley. An algorithm for planning collision-free paths among polyhedral obstacles. *Communications of the ACM*, 22(10):560–570, October 1979.

[17] M. Morales, L. Tapia, R. Pearce, S. Rodriguez, and N. M. Amato. A machine learning approach for feature-sensitive motion planning. In *Algorithmic Foundations of Robotics VI*, pages 361–376. Springer, Berlin/Heidelberg, 2005.

[18] M. A. Morales A., R. Pearce, and N. M. Amato. Metrics for analyzing the evolution of C-Space models. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 1268–1273, May 2006.

[19] M. A. Morales A., L. Tapia, R. Pearce, S. Rodriguez, and N. M. Amato. C-space subdivision and integration in feature-sensitive motion planning. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 3114–3119, April 2005.

[20] C. L. Nielsen and L. E. Kavraki. A two level fuzzy PRM for manipulation planning. Technical Report TR2000-365, Computer Science, Rice University, Houston, TX, 2000.

[21] J. H. Reif. Complexity of the mover's problem and generalizations. In *Proc. 1EEE Symp. Foundations of Computer Science (FOCS)*, pages 421–427, San Juan, Puerto Rico, Oct. 1979.

[22] S. Rodriguez, S. Thomas, R. Pearce, and N. M. Amato. (RESAMPL): A region-sensitive adaptive motion planner. In *Algorithmic Foundation of Robotics VII*, pages 285–300. Springer, Berlin/Heidelberg, 2008.

[23] D. Xie, M. Morales, R. Pearce, S. Thomas, J.-M. Lien, and N. M. Amato. Incremental map generation (IMG). In *Algorithmic Foundation of Robotics VII*, pages 53–68. Springer, Berlin/Heidelberg, 2008.