2007 IEEE International Conference on
Robotics and Automation
Roma, Italy, 10-14 April 2007

Biasing Samplers to Improve Motion Planning Performance

Shawna Thomas, Marco Morales, Xinyu Tang, and Nancy M. Amato

Abstract— **With the success of randomized sampling-based motion planners such as Probabilistic Roadmap Methods, much work has been done to design new sampling techniques and distributions. To date, there is no sampling technique that out-performs all other techniques for all motion planning problems. Instead, each proposed technique has different strengths and weaknesses. However, little work has been done to combine these techniques to create new distributions. In this paper, we propose to bias one sampling distribution with another such that the resulting distribution out-performs either of its parent distributions. We present a general framework for biasing samplers that is easily extendable to new distributions and can handle an arbitrary number of parent distributions by chaining them together. Our experimental results show that by combining distributions, we can out-perform existing planners. Our results also indicate that not one single distribution combination performs the best in all problems, and we identify which perform better for the specific application domains studied.**

I. Introduction

Motion planning has many applications outside of robotics such as computer animation [16], [4], drug design [21], computational biology [3], and computer aided design [9], [22]. Complete motion planning algorithms are rarely used in practice because they are computationally infeasible for all but the simplest problems [20]. Instead, attention has turned to randomized algorithms that sacrifice completeness for computational feasibility. In particular, Probabilistic Roadmap Methods (PRMs) [15] have been highly successful in solving difficult problems in different application domains.

PRMs require two basic operations: (1) the ability to randomly sample a configuration of the robot and (2) a feasibility/validity test for samples (e.g., collision detection). While PRMs are simple to apply and highly successful, they perform poorly in some situations, particularly when the robot must pass through a narrow passage. To overcome this, much work has been done to design new sampling techniques to increase the probability of sampling these narrow passages. Techniques include biasing sampling around obstacles [2], [6], [13], [19], biasing sampling toward the medial axis of the free space [23], [10], biasing sampling toward unknown/uncertain areas [8], [7], etc.

None of these sampling techniques has been shown to out-perform all other existing techniques for all problem instances. Instead, each sampling technique has different strengths and weaknesses. In fact, that is the motivation behind most recent research. However, many of these sampling methods start from a uniform random distribution and filter or perturb samples to create a new distribution. Little work has been done to combine existing sampling techniques.

We propose a general framework of biasing one sampling distribution with another such that the resulting distribution out-performs either of its parent distributions. Instead of filtering/perturbing samples from a uniform random distribution (as with many existing sampling schemes), we filter/perturb samples from some other existing distribution.

Consider the 2D configuration space example in Figure 1(a). Assume that roadmaps are built by sampling nodes and for each node only connecting a small number of neighboring nodes, as is typical of many planners. A distribution that samples obstacle surfaces (Figure 1(b)) may have trouble bridging the obstacle gaps because connections are attempted between nodes that are likely to be near the same obstacle. Conversely, a distribution that samples the medial axis (Figure 1(c)) may have trouble finding samples in the narrow passages because the surrounding obstacles are "thin". However, if we combine these distributions by first sampling near obstacles and then retracting these samples to the medial axis (instead of retracting samples from the uniform random distribution), we can sample the narrow passages and bridge the large gaps between them (Figure 1(d)).

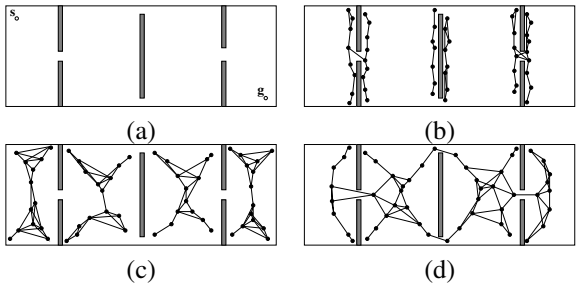

Fig. 1. (a) A 2D example where biased sampling is useful. Existing techniques such as sampling around obstacles (b) or the medial axis (c) may have difficulty capturing the connectivity. Combining these distributions, e.g., using the distribution (b) to bias the distribution (c), may overcome their weaknesses (d).

In this paper, we present a general framework for biasing samplers that is easily extendable to new distributions and can handle an arbitrary number of parent distributions by chaining them together. To our knowledge, this has not been done before in the motion planning community. Our experimental results show that by combining distributions, we can out-perform existing planners. Our results also indicate that not one single combination of distributions performs the best in all problems, and we identify which combinations perform better for the specific application domains studied.

This work was supported by NSF Grants EIA-0103742, ACR-0081510, ACR-0113971, CCR-0113974, ACI-0326350, by the DOE, and by Hewlett-Packard. Thomas supported in part by an NSF Graduate Research Fellowship, a PEO Scholarship, and a DOE GAANN Fellowship. Morales supported in part by a Fulbright/Garcia Robles (CONACYT) Fellowship.

II. Related Work

The motion planning problem is to find a valid path for a movable object/robot between some start configuration and some goal configuration. A useful abstraction is *configuration space* (C-space), the set of all possible configurations, valid or not. Thus, motion planning algorithms can handle arbitrary robots by simply planning the motion in the robot's C-space instead of in its workspace.

A. Sample-based Motion Planning

One class of randomized algorithms builds a roadmap, or graph, that can be used to answer many different queries like the Probabilistic Roadmap Methods (PRMs) [15]. To construct a roadmap, configurations are randomly sampled from C-space and kept if they satisfy feasibility requirements (e.g., collision-free). Then connections between neighboring configurations are attempted using a simple local planner (e.g., a straight-line in C-space). Valid connections become edges in the roadmap. To process a given query, or start and goal configuration pair, first the start and goal are connected to the roadmap using a local planner, and then a graph search returns the shortest path in the roadmap between the start and the goal, or reports that none exists.

Another class of randomized algorithms explores C-space beginning from a start configuration to produce a tree such as Rapidly-exploring Randomized Trees (RRTs) [17], Ariadne's Clew [5], and Hsu's expansive planner [14]. For example, RRTs bias tree growth toward unexplored regions of C-space stopping when the tree has reached the goal configuration.

B. Overcoming the Narrow Passage Problem

Although randomized motion planning methods have been successful in solving many difficult problems, they perform poorly when the solution path must go through a narrow passage in C-space. These methods are based in uniform random sampling, so the probability of sampling a configuration in a particular passage is equal to the percentage of C-space volume which that passage occupies. Thus, research has focused on sampling strategies that increase this probability.

Many sampling strategies attempt to generate samples near the surface of C-space obstacles. For example, [2] generates samples near these surfaces by first generating a random colliding (resp., collision-free) sample and searching along a random direction until the sample becomes collision-free (resp., colliding). In [6], pairs of samples are created that are a distance d apart, where d has a Gaussian distribution, until one sample is collision-free and the other is not. The collision-free sample is kept in the roadmap. [13] is similar to Gaussian sampling in that it samples pairs of configurations a distance d apart. The difference is pairs are sampled until they are both colliding and their midpoint is collision-free. This collision-free sample is stored in the roadmap. Finally, [19] creates local paths in the robot's contact space by using contact information from continuous collision detection to generate samples in the robot's contact space.

Other strategies bias sampling toward the medial axis of the free space since the medial axis encodes the C-space topology and samples on it have maximal clearance. In [23], both colliding and collision-free samples are retracted to the medial axis of the free C-space. By retracting colliding samples, the probability of sampling narrow passages is increased. In [10], the medial axis of the workspace is used to bias sampling in the C-space. The workspace medial axis is quickly computed using graphics hardware.

Finally, [8], [7], bias sampling toward unknown/uncertain areas. As they sample, they build a model of C-space that records the entropy of each sample. To generate a new sample, they compute the expected information gain over a set of random samples. The sample with the greatest information gain is added to the model and also added to the roadmap if it is valid.

C. Measuring Performance

Recent research on C-space model evolution provides metrics to evaluate the performance of proposed sampling strategies. An ideal roadmap should reflect C-space coverage and connectivity. Some studies have compared C-space coverage and connectivity with those achieved by different sampling strategies [11]. This would require a discretization that is not always feasible to obtain.

Instead, we use a set of metrics proposed in [18] that characterize the planner's progress as it constructs the roadmap. These metrics classify nodes in four different categories: (1) *cc-create* when a new component is produced, (2) *cc-merge* when components are combined improving model connectivity, (3) *cc-expand* when the added sample increases the coverage of exactly one component, and (4) *cc-oversample* when the added sample does not increase the model's coverage nor connectivity. Keeping track of how the different node types evolve allows us to observe the stages of the planner's progress. Here, we apply these metrics to compare the quality of the roadmaps achieved by combining sampling distributions.

III. Biasing Samplers

Each sampling technique is known to perform well with certain types of problems and not as well with others. We propose to combine existing sampling techniques to create a new sampling distribution that exploits the strengths of its parent distributions.

Consider the heterogeneous environment in Figure 2(b). In this environment, there are large regions of free space separated by thin obstacles with narrow passages. Samplers that bias sampling toward the medial axis will be able to quickly and efficiently map the free regions of the space. However, to increase sampling in narrow passages, it relies on the volume of C-space obstacles surrounding the passages. In this environment, this volume is still small relative to the entire C-space, thus sampling in these passages may prove difficult. On the other hand, samplers that bias sampling near C-space obstacle surfaces will be more likely to sample the narrow passages. However, they may require many samples near the surface to find connections from one narrow passage to the next. By combining these two types of sampling

distributions, we can exploit the strengths of each. For example, we can first generate samples near the obstacle surfaces, and then use these samples as starting points for a sampler that biases toward the medial axis.

Algorithm III.1 outlines the general strategy of chaining samplers together. It is simple and general. Any component sampler may be used as long as it has a method to output a new sample biased by an input sample (e.g., perturbed, filtered). Note that a few select samplers, such as uniform random sampling, are not biased by definition. These samplers will simply ignore the input sample.

Algorithm III.1 Biased Sampling Framework

Input: List of samplers S, number of samples n.
Output: Set of configurations C.
Assumption: Each sampler in S has an operation "Sample" that outputs a new sample biased with an input sample.

1: **for** i=1 to n **do**
2: Let c be a uniform random configuration.
3: **for** each sampler $s \in S$ **do**
4: $c = s$.Sample(c).
5: **end for**
6: Add c to C.
7: **end forreturn** C.

IV. Results and Discussion

In this section, we present experimental results and analyze the performance of different samplers and sampler combinations. We study both rigid body problems and articulated linkage problems. All results were run on 700 MHz Intel Pentium III Xeon processors using the C++ motion planning library developed in the Parasol Lab at Texas A&M University. Collision detection was performed by RAPID [12].

We study the following component samplers: uniform random sampling, OBPRM [2], Gaussian sampling [6], Bridge test sampling [13], and MAPRM [23]. Because OBPRM can generate samples on either side of an obstacle boundary, we use two versions: one returning free samples and one returning colliding samples. Similarly, Gaussian sampling generates pairs of (free, colliding) samples, so we use two variants: one returning the free sample and one returning the colliding sample. We use the following abbreviations: BT for Bridge Test, G_f for Gauss free, G_c for Gauss collision, MA for Medial-axis based, OB_f for Obstacle-based free, and OB_c for Obstacle-based collision.

For each experiment, we first sample 5000 collision-free nodes and attempt connections between the 20 nearest neighbors for each node. Two different local planners are used to connect nodes together: straight-line and rotate at 0.5 [1]. We compute the following performance metrics: types of samples created, diameter changes (of the entire roadmap and the largest connected component), ability to solve a user-defined witness query, and number of samples in narrow passages (when defined). Results are averaged over 10 runs.

A. Rigid Body Problems

First we consider rigid body problems. We look at the three different environments in Figure 2: (a) the s-tunnel that has a narrow passage surrounded by "thick" obstacles, (b) the walls environment that has narrow passages surrounded by "thin" obstacles, and (c) a random environment with no clearly defined narrow passage. Table I summarizes the results. Note that the percentage of samples in narrow passages is extremely small in the walls environment because the volumes of all the narrow passages combined is only about 1% of the entire volume. We do not report the percentage of narrow passage samples for the random cluttered environment because it has no well-defined narrow passage.

While some individual samplers perform well, they do not in general out-perform certain sampler combinations. Sampler performance is reflected in the percentage of samples that are not cc-oversample, the percentage of samples in narrow passages, and how fast and consistently it can solve the given query. In the s-tunnel for example, OBPRM and MAPRM perform the best individually, but OB_cG_f and OB_fMA are able to solve the query quicker and more consistently. In the walls environment, the best performing component samplers, Gauss and Bridge Test, are out-performed by G_cBT. In general, combining more successful individual samplers leads to better performance. In addition, combining a poorly performing sampler with a more successful one sometimes, but not always, leads to improved performance.

Figure 3 shows in detail how four different samplers perform over time in the s-tunnel environment for a single random seed. Roadmap diameters reflect how the roadmap topology evolves. All of the samplers quickly map the two free portions of the environment, as indicated by the rapid increase in the largest connected component's diameter. In addition, the sum of the diameters during this phase is roughly twice the maximum diameter suggesting the roadmap has two large connected components. As more samples are added, the diameter measures converge signaling the emergence of one large connected component. For this environment, this also signifies when the witness query can be solved. Uniform sampling (a) and OBPRM (b) fail to solve the query in this particular run, while MAPRM (c) solves it after 2500 samples and OB_fMA (d) solves it after 1100 samples. Notice too that the diameter measures for these two samplers have the same shape, but the diameters for OB_fMA grow, converge, and stabilize much faster, also suggesting the sampler is learning the C-space topology quicker.

These plots also explain why OB_fMA performs better than the individual component samplers. First, OBPRM generates more cc-expand samples than MAPRM early on (see both the % of cc-expand and the expansion ratio). Thus, OB_fMA can boost the performance of MAPRM simply by using OBPRM samples as starting points. Second, OBPRM has more cc-create samples than MAPRM implying that it generates samples closer to constrained areas that are difficult to connect while MAPRM generates samples with larger clearances that are easier to connect. OB_fMA uses this strength of MAPRM

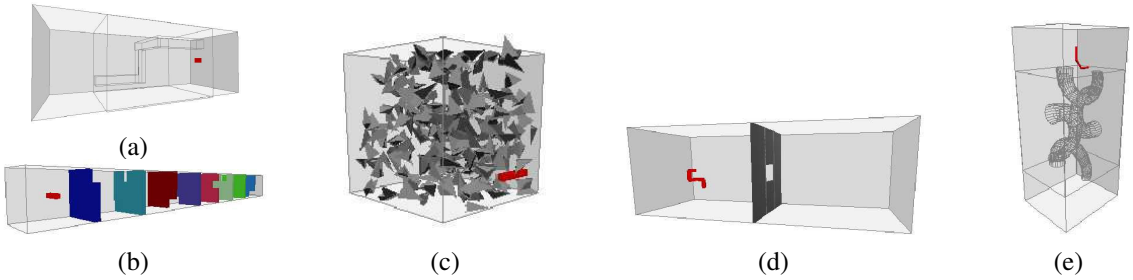

Fig. 2. Environments Studied. (a) S-tunnel environment. The robot must pass through the narrow tunnel. (b) Walls environment. The robot must pass through each narrow passage. (c) Environment with randomly placed obstacles. The stick robot must move from one corner to the opposite corner. (d) Hook environment. The articulated linkage must find the hole in the obstacle to solve the query. (e) Maze environment. The articulated linkage travels through the tunnels to move from the top to the bottom.

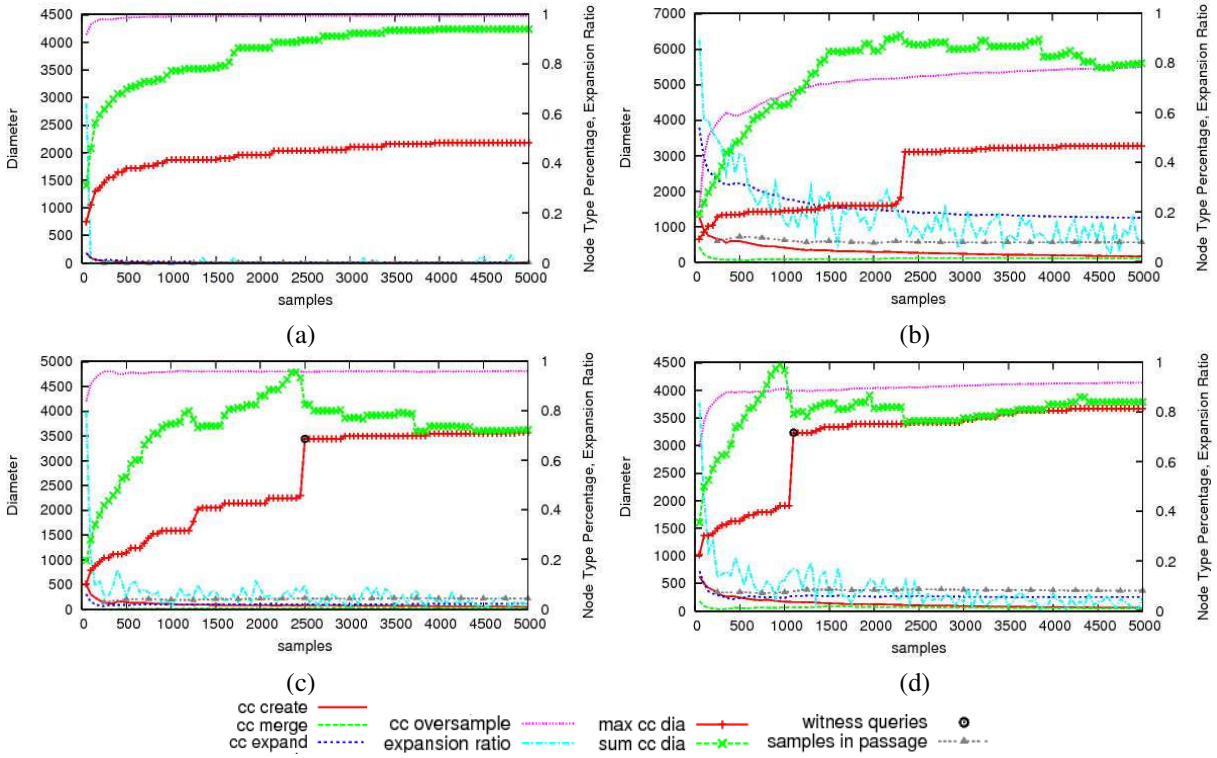

Fig. 3. Detailed s-tunnel results of a single run comparing (a) uniform sampling, (b) OBPRM, (c) MAPRM, and (d) OB$_f$MA.

to move the original OBPRM samples to more connectable areas solving the query faster.

Finally, note that the best performing samplers are not the same across all environments. For example, OB$_f$MA performs well in the s-tunnel, but not well in the walls environment. Also, there is a sharp performance difference in Gauss, Bridge, and combinations involving them between the s-tunnel and the walls environment. This may be attributed to the relation between the sampling parameter d and obstacle "thickness". The selection of d in the walls environment with "thin" obstacles was much better for building roadmaps to solve the query than in the s-tunnel with "thick" obstacles. Observe that there are not as many samplers who outperform uniform random sampling in the random cluttered environment due to the unordered nature of its C-space. In general, sophisticated samplers like OBPRM and MAPRM suffer when there is little structure to exploit.

B. Articulated Linkage Problems

We study two different articulated linkage problems: the hook environment (Figure 2(d)) that has a narrow passage surrounded by "thin" obstacles and the maze environment (Figure 2(e)) that has a narrow passage surrounded by "thick" obstacles. Table II summarizes the results.

Unlike the rigid body problems studied, no samplers were able to solve the witness query. We believe that due to the higher dimensionality of C-space, more samples or more sophisticated connection strategies are needed. However, there are still some trends in the other performance metrics. In the hook environment, G$_c$MA generates the most samples in the narrow passage, the most cc-create and cc-merge samples, and the fewest cc-oversample samples. This suggests that it

Environment	Sampling Method	Sample Type %				% Narrow Passage	Witness Query		
		cc-create	cc-merge	cc-expand	cc-oversample		% Solved	# Nodes	Time (s)
S-Tunnel	Uniform	0.15	0.00	0.25	99.60	0.14	0	n/a	n/a
	G_f	1.21	0.24	1.30	97.25	2.36	0	n/a	n/a
	BT	0.69	0.17	1.01	98.14	1.48	0	n/a	n/a
	OB_f	2.00	1.25	17.91	78.84	8.21	60	4166	531.59
	MA	1.10	0.69	2.28	95.93	4.42	90	3317	384.15
	G_cOB_f	4.26	1.56	21.39	72.78	8.98	10	1725	212.09
	G_cBT	1.07	0.17	1.61	97.15	1.81	0	n/a	n/a
	G_cMA	2.42	0.82	3.04	93.72	5.30	0	n/a	n/a
	G_fOB_f	1.58	0.36	18.19	79.88	2.30	0	n/a	n/a
	G_fBT	1.11	0.30	1.25	97.34	2.28	0	n/a	n/a
	G_fMA	0.98	0.36	1.08	97.57	2.44	10	3102	289.29
	OB_cG_f	1.50	0.99	6.00	91.51	6.88	100	2840	304.97
	OB_cBT	1.18	0.81	7.08	90.93	6.09	80	1903	184.90
	OB_cMA	1.47	0.94	8.38	89.21	6.72	90	2955	401.72
	OB_fG_f	1.99	1.29	17.89	78.83	8.20	70	3578	429.32
	OB_fBT	2.00	1.26	17.62	79.12	8.03	70	3606	465.43
	OB_fMA	1.35	1.01	5.96	91.68	8.31	100	2140	242.19
	BTG_f	0.72	0.23	0.95	98.10	1.55	0	n/a	n/a
	$BTOB_f$	1.21	0.29	17.45	81.05	1.61	0	n/a	n/a
	BTMA	0.68	0.20	0.71	98.41	1.47	0	n/a	n/a
Walls	Uniform	0.24	0.19	1.88	97.69	0.19	30	3800	168.61
	G_f	0.30	0.23	3.02	96.45	0.43	70	3725	164.00
	BT	0.40	0.32	4.53	94.74	0.88	80	2515	108.59
	OB_f	0.77	0.45	12.57	85.22	0.73	10	4579	278.85
	MA	0.40	0.30	2.65	96.66	0.83	40	3438	188.73
	G_cOB_f	1.73	0.77	16.62	80.88	1.70	0	n/a	n/a
	G_cBT	0.61	0.51	7.73	91.15	1.33	100	1802	70.29
	G_cMA	1.11	0.64	8.01	90.25	1.98	0	n/a	n/a
	G_fOB_f	0.67	0.39	14.33	84.71	0.55	0	n/a	n/a
	G_fBT	0.36	0.27	3.53	95.84	0.50	40	3827	208.42
	G_fMA	0.27	0.20	1.81	97.71	0.46	20	4446	268.96
	OB_cG_f	0.44	0.27	5.67	93.62	0.50	0	n/a	n/a
	OB_cBT	0.44	0.32	6.39	92.86	0.89	50	4106	249.97
	OB_cMA	0.53	0.30	5.51	93.65	0.70	0	n/a	n/a
	OB_fG_f	0.76	0.42	13.79	85.04	0.76	10	4417	299.36
	OB_fBT	0.75	0.40	13.82	85.03	0.80	0	n/a	n/a
	OB_fMA	0.45	0.29	3.87	95.40	0.74	10	2926	154.66
	BTG_f	0.37	0.28	4.46	94.89	0.82	60	2926	138.56
	$BTOB_f$	0.78	0.42	14.90	83.89	0.89	0	n/a	n/a
	BTMA	0.35	0.27	2.54	96.84	0.75	80	2827	155.02
Random Cluttered	Uniform	10.29	5.27	52.76	31.68	n/a	70	2471	182.96
	G_f	14.01	6.97	53.42	25.60	n/a	100	3111	240.58
	BT	19.48	9.12	51.73	19.67	n/a	50	2036	201.00
	OB_f	27.57	11.34	50.06	11.03	n/a	70	3043	308.70
	MA	20.58	9.28	50.23	19.92	n/a	60	2375	352.60
	G_cOB_f	29.61	12.02	48.61	9.77	n/a	50	2921	330.33
	G_cBT	20.32	9.58	51.12	18.98	n/a	40	3107	357.22
	G_cMA	22.21	9.50	49.70	18.60	n/a	10	1484	238.81
	G_fOB_f	20.11	9.55	54.57	15.76	n/a	60	2215	199.88
	G_fBT	14.12	7.13	53.54	25.22	n/a	90	2620	220.38
	G_fMA	13.06	6.66	50.39	29.90	n/a	90	1847	202.47
	OB_cG_f	14.27	7.15	53.12	25.48	n/a	60	3199	339.65
	OB_cBT	17.69	8.61	52.80	20.90	n/a	60	2329	245.91
	OB_cMA	21.62	9.64	51.03	17.71	n/a	70	2279	319.13
	OB_fG_f	27.86	11.71	49.62	10.82	n/a	60	2600	273.22
	OB_fBT	26.84	11.18	50.75	11.24	n/a	60	2925	308.35
	OB_fMA	21.94	9.71	51.30	17.06	n/a	80	1913	233.55
	BTG_f	19.46	9.18	51.46	19.90	n/a	50	1738	166.45
	$BTOB_f$	25.66	11.16	50.83	12.35	n/a	30	2236	247.03
	BTMA	17.87	8.19	50.62	23.32	n/a	60	1241	152.66

TABLE I

RIGID BODY PERFORMANCE RESULTS FOR VARIOUS ENVIRONMENTS.

is able to find "important" samples but lacks sophisticated methods to connect them. This trend repeats in the maze environment. Finally, we can see how obstacle "thickness" affects the performance of samplers sensitive to this like MAPRM. Notice that OB_cMA performs better than OB_fMA in the hook environment while OB_fMA performs better than OB_cMA in the maze environment. This directly follows from the obstacle "thicknesses": obstacles are "thin" in the hook so colliding surface nodes are critical while obstacles are "thick" in the maze so free surface nodes are critical.

Sampling Method	HOOK ENVIRONMENT					MAZE ENVIRONMENT				
	Sample Type %				% Narrow Passage	Sample Type %				% Narrow Passage
	cc-create	cc-merge	cc-expand	cc-oversample		cc-create	cc-merge	cc-expand	cc-oversample	
Uniform	1.87	1.22	50.00	46.92	0.32	1.37	0.99	51.48	46.16	0.03
G_f	2.62	1.59	50.71	45.09	1.28	3.73	2.36	59.09	34.83	0.46
BT	4.50	3.16	44.21	48.12	0.56	3.82	2.71	49.17	44.29	0.26
OB_f	3.07	1.94	48.59	46.40	1.04	5.84	1.95	54.36	37.84	2.95
MA	5.06	2.86	46.97	45.11	1.42	3.57	1.81	55.77	38.85	0.34
G_cOB_f	3.82	2.53	51.29	42.35	1.07	3.64	2.63	59.13	34.60	0.07
G_cBT	5.82	4.14	38.99	51.06	0.60	4.21	3.12	48.82	43.85	0.02
G_cMA	10.27	6.00	48.40	35.33	2.19	9.45	5.31	60.61	24.62	0.18
G_fOB_f	2.83	1.93	49.60	45.64	0.87	3.84	2.46	59.19	34.51	0.43
G_fBT	2.82	1.83	50.72	44.64	0.83	3.67	2.40	59.84	34.09	0.40
G_fMA	2.20	1.30	44.57	51.93	0.64	3.52	2.12	56.54	37.82	0.44
OB_cG_f	2.85	1.85	50.07	45.23	0.77	3.15	1.94	56.13	38.77	0.31
OB_cBT	5.67	3.77	39.52	51.04	0.62	3.33	2.31	44.89	49.47	0.19
OB_cMA	7.87	4.65	47.60	39.88	1.68	5.78	3.24	58.64	32.34	0.12
OB_fG_f	3.00	1.91	49.27	45.83	0.95	5.88	2.00	53.16	38.95	2.94
OB_fBT	3.06	1.98	49.38	45.59	1.00	5.88	2.02	53.61	38.48	2.79
OB_fMA	2.68	1.70	42.71	52.90	0.79	5.26	1.67	51.62	41.45	2.91
BTG_f	4.29	2.99	43.71	49.01	0.60	3.66	2.60	48.28	45.47	0.26
$BTOB_f$	4.19	3.00	43.90	48.91	0.64	3.84	2.80	49.05	44.30	0.22
BTMA	3.71	2.62	38.10	55.57	0.45	3.54	2.52	46.93	47.01	0.20

TABLE II

ARTICULATED LINKAGE PERFORMANCE RESULTS FOR VARIOUS ENVIRONMENTS.

V. CONCLUSION

We presented a simple, general framework for combining existing sampling techniques. The advantage is that we can exploit the strengths of existing sampling methods to create a new sampling method that out-performs it's component samplers. We demonstrated the performance of this framework in various environment types for both rigid bodies and articulated linkages. In all environments, some strategies generated large amounts of samples in narrow passages and were unable to solve the query. This signifies the importance for sophisticated connection methods in addition to sampling.

REFERENCES

[1] N. M. Amato, O. B. Bayazit, L. K. Dale, C. V. Jones, and D. Vallejo. Choosing good distance metrics and local planners for probabilistic roadmap methods. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 630–637, 1998.

[2] 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. A.K. Peters. Proc. Third Workshop on Algorithmic Foundations of Robotics (WAFR), Houston, TX, 1998.

[3] N. M. Amato and G. Song. Using motion planning to study protein folding pathways. *J. Comput. Biol.*, 9(2):149–168, 2002. Special issue of Int. Conf. Comput. Molecular Biology (RECOMB) 2001.

[4] O. B. Bayazit, J.-M. Lien, and N. M. Amato. Better flocking behaviors using rule-based roadmaps. In *Proc. Int. Workshop on Algorithmic Foundations of Robotics (WAFR)*, pages 95–111, Dec 2002.

[5] P. Bessiere, J. M. Ahuactzin, E. G. Talbi, and E. Mazer. The Ariadne's clew algorithm: Global planning with local methods. In *Proc. IEEE Int. Conf. Intel. Rob. Syst. (IROS)*, volume 2, pages 1373–1380, 1993.

[6] 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.

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

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

[9] H. Chang and T. Y. Li. Assembly maintainability study with motion planning. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 1012–1019, 1995.

[10] M. Foskey, M. Garber, M. Lin, and D. Manocha. A voronoi-based hybrid motion planner. In *Proc. IEEE/RSJ International Conf. on Intelligent Robots and Systems (IROS 2001)*, 2001.

[11] R. Geraerts and M. H. Overmars. Reachablility analysis of sampling based planners. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 406–412, 2005.

[12] 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.

[13] D. Hsu, T. Jiang, J. Reif, and Z. Sun. Bridge test for sampling narrow passages with proabilistic roadmap planners. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 4420–4426, 2003.

[14] D. Hsu, R. Kindel, J. C. Latombe, and S. Rock. Randomized kinodynamic motion planning with moving obstacles. In *Proc. Int. Workshop on Algorithmic Foundations of Robotics (WAFR)*, pages SA1–SA18, 2000.

[15] L. E. Kavraki, P. Svestka, 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.

[16] Y. Koga, K. Kondo, J. Kuffner, and J. Latombe. Planning motions with intentions. In *Proc. ACM SIGGRAPH*, pages 395–408, 1995.

[17] S. M. LaValle and J. J. Kuffner. Rapidly-Exploring Random Trees: Progress and Prospects. In *Proc. Int. Workshop on Algorithmic Foundations of Robotics (WAFR)*, pages SA45–SA59, 2000.

[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] S. Redon and M. C. Lin. Practical local planning in the contact space. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, April 2005.

[20] J. H. Reif. Complexity of the mover's problem and generalizations. In *Proc. 20th Annu. IEEE Sympos. Found. Comput. Sci.*, pages 421–427, 1979.

[21] A. Singh, J. Latombe, and D. Brutlag. A motion planning approach to flexible ligand binding. In *7th Int. Conf. on Intelligent Systems for Molecular Biology (ISMB)*, pages 252–261, 1999.

[22] S. Sundaram, I. Remmler, and N. Amato. Disassembly sequencing using a motion planning approach. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 1475–1480, 2001.

[23] S. A. Wilmarth, N. M. Amato, and P. F. Stiller. MAPRM: A probabilistic roadmap planner with sampling on the medial axis of the free space. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, volume 2, pages 1024–1031, 1999.