The Gaussian PRM Sampling for Dynamic Configuration Spaces

Yu-Te Lin
Computer Science Department
Stanford University
yutelin@stanford.edu

Abstract— **Probabilistic roadmap planners (PRMs) are widely used in high dimensional motion planning problems. However, they are less effective in solving narrow passages because feasible configurations in a thin space can rarely be sampled by random. Although some approaches have been proposed, they are either involved in complicated geometrical computations or requiring much information of obstacles. Moreover, if the configuration spaces are dynamic instead of fixed, some solutions may be failure in some unexpected situations. In this work, we provide a novel approach to replace the randomized sampler with the Gaussian PRM sampler. For dynamic configuration spaces, we also invite a machine learning technique to dynamically classify the pre-built samplers for different spaces.**

Keywords—**Probabilistic roadmap (PRM), sampling strategy, narrow passages, path planning and dynamic configuration spaces**

I. Introduction

PRM is a widely used path planning algorithm, which works efficiently in high dimensional spaces; moreover, it could be integrated in a variety of robot control programs. However, there is a critical problem - narrow passages. Since this problem comes from the randomized sampling in roadmap construction; intuitively, if one can just sample in the narrow passages, this problem could be solve directly. In this work, we use a simple data structure, Gaussian distribution set, to model the free space. The algorithm could improve the connectivity of roadmap, about 25% in our experiments, and success sampling rate, about 50% in our experiments. On the other hand, for dynamic configuration spaces (C-spaces), to enumerate samplers for all C-spaces is computationally intractable; therefore, we generate some representative samplers and training data according to the target application. By using the machine learning technique, Support Vector Machines (SVM) in our experiments, one can produce a classifier off-line and use it before every roadmap construction.

The rest of this paper is organized in five sections. Section II describes the related work of PRM. Section III and Section IV respectively describes the construction and reuse of Gaussian PRM samplers. Section V shows the detail of implementation and experimental results. Section VI summarizes this research.

II. Related Work

A. Motion Planning

Motion planing is an essential sub-field of robotics and has been researched for decades [1]. Nowadays, it has been widely used in applications, such as CAD/CAM, virtual environment systems, motion simulations, computational biology, intelligent user interface, and radio surgery equipment. Generally, motion planning algorithms are composed of two parts, modeling and searching. Although workspace is usually defined in two dimension or three dimension, C-space is frequently mapped to a higher dimensional space with respect to the degrees of freedom (DOFs). In the searching phrase, collision detections and planning algorithms will be adopted. Planning algorithms are designed in many differen manners, such as potential field and cell decomposition, but recently, the algorithms tend to be devised with randomized sampling concept [2].

B. Probabilistic roadmap path planner

In robotics, motion planning usually confronts the problem of high dimensionality which makes the search space become extremely large. A general approach to solve this high-dimensional problem is to use the sampling based methods. The well-known technique, PRM [3], divides the planning process into learning phrase and query phrase. In the learning phrase, it randomly samples the free space and uses the local planner to construct a roadmap. In the query phrase, it tries to connect the start and goal configurations to the roadmap respectively. If both of the end configurations are able to be connected to the same connected component, a continuous collision-free path could be found. Actually, there are several algorithms in this family, such as rapidly exploring random trees (RRTs) [4] [5] which builds the roadmap from known configurations, Fuzzy PRM [6], which builds the roadmap with loosely checking local planner and validates the roadmap at last, and the predictive PRM [7], which uses active sampling to reduce expansive configuration validations.

C. Sampling Strategies for PRM

This family of research is derived from the general narrow passage problem in PRM. In the learning phrase of roadmap building, this algorithm usually samples for free configurations in a randomized manner. However, by randomized sampling, one can build a roadmap covering the narrow passages only by probing almost everywhere in the C-space, which extremely reduces the benefit of PRM. Consequently, there are several sampling strategies proposed to overcome this problem. Most of these approaches assume that narrow passages appear near the obstacles in a workspace. Therefore, some of them use the heuristic methods based on the obstacle surface properties [8]

Algorithm 1 GaussianSetGenerator

```
1:  $\mu \leftarrow$ RandomFreeConfiguration()
2:  $\Sigma \leftarrow$ IdentityCovarianceMatrix()
3:  **while** GaussianSampleError($\Sigma$) $< \tau$ **do**
4:      $\Sigma \leftarrow$ ExtendAllElements($\Sigma$, $step$)
5:  **end while**
6:  **for all** $i$ in diagonal **do**
7:      **while** GaussianSampleError($\Sigma$) $< \tau_i$ **do**
8:          $\Sigma \leftarrow$ ExtendOneElement($\Sigma$, $step$, $i$)
9:      **end while**
10: **end for**
```

or the shrinking and growing of obstacles [9]. There are also some approaches, such as Gaussian sampling [10] and Bridge-Test sampling [11] which directly replace the randomized sampler to create samples near obstacles.

III. Gaussian PRM Samplers

In this section, a pre-computation method for building Gaussian PRM Sampler is introduced. Basically, this method learns a set of Gaussian distributions to model the free space off-line. In other words, the objective is to construct a set of Gaussian distributions to approximate the free space by exploring the geography of the C-space in advance. In the exploring process, both of the randomized sampling and Bridge-Test sampling are used for μ selections. As a result, one can use the Gaussian PRM Sampler, a set of Gaussians, to efficiently generate collision free configurations and narrow passage configurations in the learning phrase of PRM.

A. *Free Space Learning*

Initially, there is no assumption of C-space; therefore, the randomized algorithm, Algorithm 1, is used for generating the Gaussian sets. As the algorithm shows, this method expands each Gaussian gradually to model the free space. First of all, a free configuration is randomly sampled and be treated as the mean (μ) of a Gaussian. Then the diagonal elements of the corresponding covariance matrix (Σ) are extended simultaneously until the current Gaussian's error sampling rate meets the tolerance (τ). Another set of tolerances (τ_i) are also adopted for those diagonal elements to extend respectively, which will make the Gaussian model the free space better. After several iterations, one could get a set of Gaussians by this approach. Once the the number of Gaussians reaches the threshold, an appropriate number that represents there are sufficient Gaussians to approximate the free space, one could get a Gaussian PRM sampler. Please note that the elements off the diagonal of the covariance matrix are ignored since it will enormously increase the computation complexity.

In stead of using a single Gaussian, an adequate number of Gaussian distributions should be generated to model the free space. Based on Fig. 1, it is assumed that the free space could be well-approximated by these Gaussians as long as the number of Gaussians is sufficient even though only the diagonal elements of covariance matrix are taken into account.

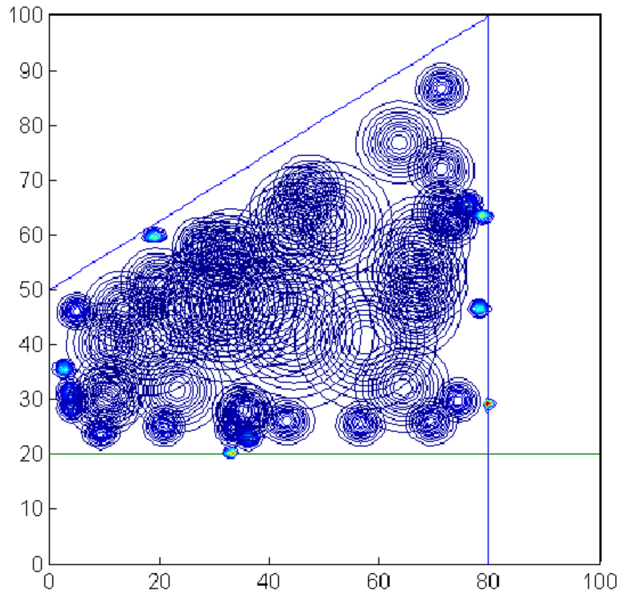

Fig. 1. Gaussian PRM Sampler. This figure is drawn from a set of Gaussians. The circles model the free space of a C-space.

Algorithm 2 BinaryGaussianSetGenerator

```
1:  $\mu \leftarrow$ RandomFreeConfiguration()
2:  $\Sigma \leftarrow$ IdentityCovarianceMatrix()
3:  $lowerBound \leftarrow 1$
4:  $upperBound \leftarrow bound$
5:  **repeat**
6:      $\Sigma \leftarrow$ SetVariance($(upperBound + lowerBound)/2$)
7:      **if** GaussianSampleError($\Sigma$) $> \tau$ **then**
8:          $upperBound \leftarrow (upperBound + lowerBound)/2$
9:      **else**
10:         $lowerBound \leftarrow (upperBound + lowerBond)/2$
11:     **end if**
12: **until** convergence
13: **for all** $i$ in diagonal **do**
14:     **while** GaussianSampleError($\Sigma$) $< \tau_i$ **do**
15:         $\Sigma \leftarrow$ ExtendOneElement($\Sigma$, $step$, $i$)
16:     **end while**
17: **end for**
```

In fact, if the distributions are over-fitting to a specific free space, the usefulness of the corresponding sampler will be seriously limited.

B. *Binary Gaussian Generator*

In Algorithm 1, it incrementally extends the Gaussians to fit the free space; however, this approach spends too much time on growing the Σ which is initialized to the value of one. Actually, the most time consuming part is to gather the error rate from a given covariance matrix. Base on experimental experience, the simultaneously extension of diagonal elements dominates the construction time. In other words, whenever a circle shaped

Algorithm 3 BridgeTestSampling

```
1: repeat
2:     $c_1 \leftarrow$ RandomCollidedConfiguration()
3:     $c_2 \leftarrow$ RandomCollidedConfiguration()
4:     $c \leftarrow$ MiddleConfiguration($c_1$,$c_2$)
5: until CheckFreeConfiguration($c$)
```

Gaussian has been created, it could be quickly deformed to a ellipse one to fit the free space. Therefore, a modified algorithm which works like a binary search, Algorithm 2, is proposed to replace the linear search for the diagonal elements extension. And the convergence criterion could be loosely set for further reducing the computation time, but it will trade with the accuracy of free space modeling.

C. Bridge Test Gaussian Generator

To solve the narrow passage problem efficiently, the Bridge-Test method for generating the Gaussians is invited. Actually, it has the same extension process as in Algorithm 1 or Algorithm 2, but it has a different μ selection method. This method assumes the narrow passages appear between obstacles; therefore, it will first randomly sample two infeasible configurations and calculates the middle configuration of them. If the middle configuration is feasible, then it will be set as the mean of a Gaussian. Consequently, those Gaussians spread in the narrow passages will be created. The algorithm of Bridge-Test sampling is listed in Algorithm 3.

D. Composite Usage

Based on the experimental experience, it is suggested to have a composite use of the random Gaussian sampler and the Bridge-Test Gaussian sampler. Intuitively, with nodes only sampled in narrow passages cannot build a widespread roadmap; moreover, if a Gaussian sampler is over-fitting to a specific C-sapce, the sampler's reusability will be reduced. More specifically, each sampling method should have a weight; the higher the method is weighted, the more configurations will be generated from it. Therefore, in the learning phrase of PRM, there may be some configurations generated from the general Gaussian sampler, some from the Bridge-Test Gaussian sampler and even some from the pure random sampler.

IV. Reuse of Gaussian Samplers

A. Multi-step Single Query PRM

Multi-step single query PRM needs to rebuild the roadmap in each single query. Take the climbing robot [12] for example, the planner first plans a sequence of goals for the climbing robot to grasp. In successive configurations, a single query PRM planner is responsible for generating a continuous path for the robot motion. Obviously, it is a multi-step single query PRM problem. In this kind of problem, one cannot reuse any roadmap built during query since this problem is composed of several different C-spaces. Therefore, for each query, the planner has to build a new roadmap for planning. In case a C-space contains narrow passages, the planner will either spend much time on sampling new configurations to complete the roadmap or simply fail to find a path. To solve this problem, the pre-built Gaussian PRM samplers described before can help. However, it is impractical to build all Gaussian PRM samplers for all C-spaces. Therefore, the machine learning approaches and pattern classification techniques can be applied to match similar C-spaces to a common sampling strategy.

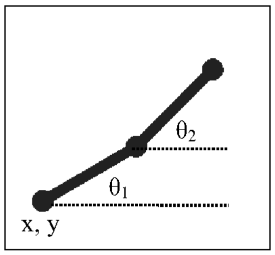

Fig. 2. An articulated robot which is parameterized with four variables (x, y, θ_1, θ_2).

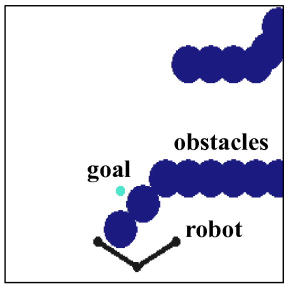

Fig. 3. First experiment. There is only one goal.

B. C-space Patterns

Since it is impossible to enumerate all Gaussian PRM samplers, we should first generate several samplers for future use. Intuitively, if the most representative samplers can be generated, the multi-step planner can have the best candidate strategies to choose from. Nevertheless, it is difficult to define the representativeness in most planning problems. Accordingly, the method here is to generate more Gaussians than they are actually needed and then prune them to a reasonable size.

Another important and difficult problem is the acquisition of training data. Since training examples are difficult to generate, the target application is applied to produce them. One simple way to collect training data is to try many multi-step planning to test which sampler is best for each specific C-space. Regarding testing a sampler, the intuition is to use it to create several free configurations and compare the error sampling rates. In addition, the area coverage and roadmap connectivity of configurations generated by the sampler are also important to validate the training data.

V. Implementation

A. Problem Definition

We constructed a simplified test bed for the dynamic C-space problem. In the setting, there is an articulated robot

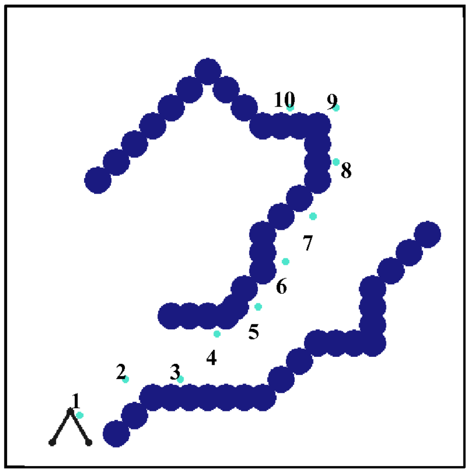

Fig. 4. Second experiment. There are ten goals in this setting. The planner will conduct ten times of single-step path planning. In each planning, the planner uses a classifier to match the best Gaussian PRM sampler.

TABLE I
ONE STEP PLANNING RESULT.

error rate	graphs number	variance
15%	208	9776

The result of randomized sampling in the first experiment. While the error rate is the percentage of samples in the infeasible area, the graph number shows how many graphs in a roadmap and the variance shows the average variance from the nodes in the roadmap.

with one revolute joint as Fig. 2 shows. The robot could be parameterized by four variables (x, y, θ_1, θ_2). There will be several holds for the robot to grasp, and the robot has to grasp at least one hold all the time. Consequently, in a continuous path, the robot can only transit by changing θ_1 and θ_2 to reach another hold and after the transition, the x and y will be switched to the other end. Therefore, for each one-step planning, there will be a different C-space. Since the C-space is not predefined, whenever there is a path to be planned, the planner has to build a new roadmap. The workspace is like Fig. 3 and Fig. 4. In order to generate the training data, we first choose 100 different C-spaces to generate more than 10000 most representative samplers and create a classifier by the SVM package [13]. Then, we use the classifier to select samplers for our experiments. There are two experiments in this work, the first is visualized in Fig. 3, which contains only one C-space, and the second is in Fig. 4, which involves in ten dynamic C-spaces.

B. Results

TABLE I shows the one-step planning result where the error rate is the percentage of random sampler's erroneous sampling, which means failure in sampling the free space, the graphs denotes the number of connected components in the roadmap, and the variance is the average node variance of the roadmap. One could estimate the accuracy of the sampler's ability in sampling the free space by the error rate, while one could also judge the connectivity of the roadmap by the number of graphs. Variance, on the other hand, represents how the area covered by a roadmap is. Generally speaking, lower percentage in error rate, fewer number in graphs and higher value in variances are indicative of a better sampler. In TABLE II, there are ten more results. The weight column denotes the percentage of samples generated by the Gaussian PRM sampler and the methods column indicates either only the general sampler (denoted by G) is used or both of the general and Bridge-Test (denoted by G+B) ones are used. As one can see from TABLE II, it is obvious that a prefect sampler could not be reached, i.e. lower error rate and fewer graphs will inevitably lead to lower variances.

TABLE II
GAUSSIAN PRM SAMPLING RESULT IN THE FIRST EXPERIMENT.

weight	sampler	error rate	graphs	variance
20%	G	12.4%	198	8499
20%	G+B	12.5%	195	8275
40%	G	11.7%	173	8382
40%	G+B	11.8%	162	8106
60%	G	9.20%	162	7960
60%	G+B	10.0%	159	7774
80%	G	0.61%	166	6882
80%	G+B	0.62%	154	6619
100%	G	0.25%	135	5906
100%	G+B	0.40%	133	5841

G: general Gaussian PRM Sampler.
G+B: both general and Bridge-Test Gaussian PRM Samplers.

TABLE III
RANDOMIZED SAMPLING RESULT.

error rate	graphs number	variance
26%	306	11600

The result for randomized sampling in the second experiment, which is composed of ten dynamic C-spaces.

TABLE IV
RESULT OF GAUSSIAN PRM SAMPLING IN THE SECOND EXPERIMENT.

weight	sampler	error rate	graphs	variance
20%	G	25.9%	328	10743
20%	G+B	27.2%	322	10536
40%	G	20.9%	298	8802
40%	G+B	22.0%	291	8658
60%	G	20.0%	304	7909
60%	G+B	20.5%	298	7586
80%	G	0.40%	109	7190
80%	G+B	0.60%	104	6981
100%	G	0.00%	61	5888
100%	G+B	0.00%	60	5673

G: general Gaussian PRM Sampler.
G+B: both general and Bridge-Test Gaussian PRM Samplers.

TABLE III and TABLE IV demonstrate the average results from the second experiment. Results of TABLE III come from randomized sampling, while those of TABLE IV come from Gaussian PRM sampling. In each one-step planning, the planner will use the pre-computed classifier to select the most appropriate Gaussian PRM sampler.

C. Discussion

As described above, there exists an inherent tradeoff between achieving lower error rate with higher connectivity and obtaining wider coverage. Actually, in a PRM planning problem, enhancing connectivity is more important than lowering error rates. The reason is not only a path could be found only if both of the initial and goal configurations have to be able to connected to the same connected component but also the planner has a better connectivity means the roadmap may possess a better coverage in narrow passages. In our experiments, the roadmap always contains more than 100 graphs since our goal is to reveal the samplers' ability to sample the narrow passages. Theoretically, the single and multiple steps planning should lead to similar results, i.e. results from TABLE I should be similar to those of TABLE III while those of TABLE II should be similar to those of TABLE IV. Nevertheless, the dynamic C-spaces in the second experiment may be varied, which means the average result is not necessary to be identical to any single result. Moreover, the generation of training data is another critical issue that should be improved. In our application, we produce the training examples by only inspecting and comparing the error rates. If both the connectivity and the coverage are taken into consideration, the criterion for selecting training examples will certainly be more complete and justified.

VI. Conclusion

This work generates multiple of Gaussians as samplers with respect to the free spaces; in addition, it uses SVM to produce a classifier in order to reuse the pre-built samplers for dynamic configuration spaces. Experiments show that this approach can reduce the error sampling rate from 25% towards 0% and can improve the connectivity. However, there are some topics left for future research, including the training data generation and the automatic parameter tuning. The training data algorithm used in this work is not so perfect to exactly generate the most representative samplers. On the other hand, several parameters, such as τ, τ_i, and weight parameters, in our application are tuned by hand. Therefore, if a self-tuning algorithm is employed, the generosity could also be improved. Moreover, this work is originally designed for assisting the climbing robot referred in [12], so a general implementation is required for supporting other applications using PRM planners.

Acknowledgment

The author would like to thank Jean-Claude Latombe and Timothy Wolfe Bretl for the idea discussions and Andrew Ng for his machine learning suggestions.

References

[1] J. C. Latombe, *Robot Motion Planning,* Kluwer Academic Publishers, Norwell, Mass. 1991.

[2] S. M. LaValle, *Planning Algorithms,* Cambridge University Press, 2006.

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

[4] S. M. LaValle, "Rapidly-exploring random trees: A new tool for path planning," *Iowa State University*, 1998.

[5] J. J. Kuffner and S. M. LaValle, "RRT-Connect: An efficient approach to single-query path planning," *in Proceedings of the International Conference on Robotics and Automation*, vol. 2, pp. 995-1001, 2000.

[6] C. L. Nielsen and L. E. Kavraki, "A two level fuzzy PRM for manipulation planning," *in Proceeding of the International Conference on Intelligent Robots and Systems*, pp. 1716-1722, 2000.

[7] B. Burns, O. Brock, "Sampling-Based Motion Planning Using Predictive Models," *in Proceedings of International Conference on Robotics and Automation*, pp. 1274-1280, 2005.

[8] N. Amato, B. Bayazid, L. Dale, C. Jones, and D. Vallejo. "OBPRM: An obstacle-based PRM for 3D workspaces," *in Robotics: The Algorithmic Perspective. AK Peters*, 1998.

[9] M. Saha and J. C. Latombe, "Finding narrow passages with probabilistic roadmaps: The small step retraction method," *in Proceedings of International Conference of Robots and Systems*, 2005.

[10] V. Boor, M. Overmars and F. van der Stappen, "Gaussian Sampling for Probabilistic Roadmap Planners," *in Proceedings of the International Conference on Robotics and Automation*, 1999.

[11] D. Hsu, T. Jiang, J. Reif and Z. Sun, "The Bridge Test for Sampling Narrow Passages with Probabilistic Roadmap Planners." *in Proceedings of the International Conference on Robotics and Automation*, 2003.

[12] T. Bretl, S. Lall, J. C. Latombe and S. Rock, "Multi-Step Motion Planning for Free-Climbing Robots," *in Workshop on the Algorithmic Foundations of Robotics*, 2004.

[13] Chih-Chung Chang and Chih-Jen Lin, *LIBSVM : a library for support vector machines,* 2001. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm