

Probabilistic Roadmaps - Putting It All Together *

Lucia K. Dale
ldale@sewanee.edu
Mathematics & Computer Science
University of the South
Sewanee, TN 37383-1000

Nancy M. Amato
amato@cs.tamu.edu
Computer Science
Texas A&M University
College Station, TX 77843-3112

Abstract

Given a robot and a workspace, probabilistic roadmap planners (PRMs) build a roadmap of paths sampled from the workspace. A roadmap node is a single collision-free robot configuration, randomly generated. A roadmap edge is a sequence of collision-free robot configurations which interpolate the path from one roadmap node to another. Queries to the roadmap are (start, goal) pairs. If both the start and goal of a pair can be connected to the same connected component of the roadmap, the query is solved.

Many promising variants of the PRM have been proposed, each with their own strengths and weaknesses. We propose a meta-planner for using many PRMs in such a way that the strengths are combined and the weaknesses offset. Our meta-planner will perform the combination in the following manner. i) Provide a framework in which different motion planners are available and to which new ones are easily added. ii) Characterize subregions (possibly overlapping) based on sample characteristics and connection results. iii) Assign subregions to one or more planners which are judged promising. iv) Provide stopping criteria for roadmap construction.

We present experimental results for four characterization measures. A general technique we call 'filtering' is presented for keeping roadmaps compact.

1 Introduction

The basic motion planning problem is to compute a collision-free path for a moving object through a workspace which may contain obstacles. The moving object can be simple or complex as can the workspace. In robotics, the moving object is referred to as a 'robot'. For convenience we will do the same. The problem however arises in many fields other than robotics - drug design, space exploration, virtual reality, computer aided design, surgical training, and material management to name just a few.

*This research supported in part by NSF CAREER Award CCR-9624315, NSF Grants IIS-9619850, ACI-9872126, EIA-9975018, EIA-9805823, and EIA-9810937, and by the Texas Higher Education Coordinating Board under grant ARP-036327-017. Work completed while Dale at Texas A&M University.

Motion planning is an intractable problem. The lower bound is known to be PSPACE-hard [4, 13, 20, 26, 27] while the upper bound is likely to be exponential in the number of degrees of freedom the robot possesses [10, 11]. In light of these complexity results, our work, as well as that of many other researchers in this area, is experimental and heuristic in approach. The approaches we will focus on are the probabilistic roadmap method (PRM) [17, 19, 24, 25] and probabilistic 'growth' planners [7, 22].

Given a robot and a workspace, (PRMs) build a roadmap, usually stored as a graph, of the robot's workspace. As described by [17, 19, 24, 25] the basic PRM uses uniform sampling to generate, uniformly at random, configurations (a.k.a. nodes) of the robot in the workspace. Configurations which are in collision are discarded. For those collision-free nodes remaining, pair-wise connections are attempted using a local planner which interpolates a path from one node to another. Collision-free connections are retained and the result is a basic PRM roadmap. Attempting exhaustive pair-wise connections is uncommon. Instead, for each node, the remaining nodes are sorted by distance [1] and connections are attempted for a user-selected small constant number k of them. After construction, queries to the roadmap are (start, goal) pairs. If both the start and goal of a pair can be connected to the same connected component of the roadmap graph, the query is solved. Many queries can be rapidly solved or rejected in real-time by a single roadmap. If a single specific query is known beforehand and is the only query of interest for the robot, some motion planning approach other than one which builds a roadmap for the entire workspace should be selected [17, 19, 24, 25].

If the robot can move without collision from point A to point B in the workspace, a connecting path between the two points should exist in a roadmap of that workspace. Good roadmaps accurately reflect the connectivity of workspaces with respect to given robots. A PRM roadmap records a sample of all feasible paths. If allowed to run long enough, a probabilistically complete algorithm returns a correct solution, if one exists. Usually only probabilistically complete [18], PRMs trade complete solutions for faster ones.

Probabilistic growth planners employ a growth paradigm [7, 22] which, starting with one or more collision-free nodes (seeds), attempt to generate new nodes and edges in such a way as to move out from the initial seeds. The goal may be to explore a region or to reach one or more other seeded regions. Every seed initiates a growing connected component for the roadmap.

2 Philosophy

An example of a motion planning problem environment (workspace and robot) is shown in Figure 1. In this workspace the robot, lower left hand corner, is free to move about within the white areas. The shaded regions represent physical barriers to the robot’s movement. At no time may the robot enter or cross a shaded region. Types of regions we are interested in include:

- *free* - The region is free from obstacles.
- *cluttered* - The region is cluttered with obstacles.
- *narrow passage* - The region provides a passage between other types of regions.
- *blocked passage* - The region suggests a passage but does not in fact provide one between other types of regions.
- *blocked* - The region is completely filled with obstacles.

Figure 1 provides examples of each. Some areas, A & E, are relatively free from obstacles. The robot can move freely in these areas which are easily located and explored using relatively simple and inexpensive planners. Other areas are more problematic. Label B is near an opening of a narrow passage connecting the free areas of A and E. The robot can crawl through from one to the other. Differentiating a narrow passage from a blocked passage such as that near label C can be quite challenging. Area D contains a clutter of closely spaced obstacles. This can be viewed as many intersecting narrow passages.

The characterizations of different subregions, “free”, “cluttered”, “narrow”, and so on, depend on the robot as well as the workspace. When the robot is smaller or more maneuverable (more degrees of freedom) a workspace will seem more open than when the reverse is true.

Many different motion planning algorithms have been proposed. The basic *PRM* can discover and map the free regions of the robot’s navigable space [14, 17, 19, 24, 25]. Variants of the *PRM* have been proposed that propose, with varying degrees of success, to map narrow passages and cluttered environments

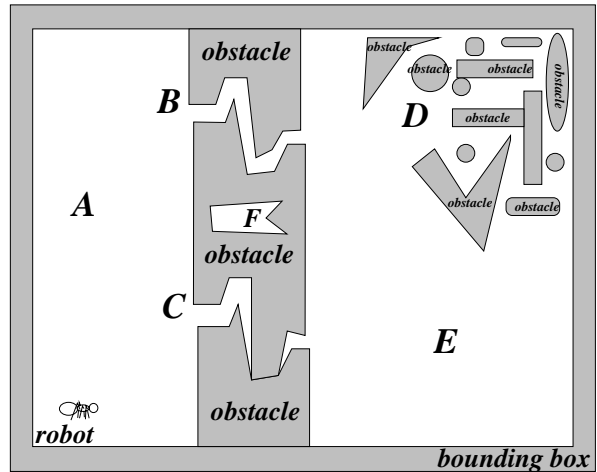


Figure 1: Work space exhibiting areas which allow the robot, located at lower left, differing amounts of maneuverability among the shaded obstacles.

[3, 6, 15, 25, 28, 29]. The distinction between passages (Fig. 1B) and promising but ultimately blocked passages (Fig. 1C) or isolated regions (Fig. 1F) is often impossible.

In addition to the basic *PRM* and variant methods, planning methods have been proposed which are also based on probabilistic approaches but are unique in some way [7, 22]. Other recent work aims to improve the efficiency of *PRMs* by postponing most of the validation to the query stage [8, 23]. Each has, as described by its authors, a unique set of strengths and weaknesses.

We propose a synthesis. If the original problem can be broken down into subproblems which match or approximate the various sets of strengths and weaknesses, then the appropriate motion planner can be assigned to work on the appropriate subproblem. A summary of planners and our understanding of their strengths is provided in Table 1.

Breaking the original problem into subregions is problem dependent, as is the assignment of appropriate planners. We are aware of no work being done to make such subdivisions or to assign the planners automatically. Currently, problem subdivision is done manually or, more often, not at all, while choice of planner is made by users, based on non-deterministic criteria such as intuition, estimates of gross computational expense, and/or software availability.

We propose a meta-planner which will do the following.

- Provide a framework in which different motion planners are available and to which new ones are easily added.
- Characterize subregions (possibly overlapping)

Algm	open	clutter		
		less	more	most
Basic PRM [19]	✓	✓		
Fuzzy PRM [23]	✓	✓		
Lazy PRM [8]	✓	✓		
RNG [30]	✓	✓		
OBPRM [3]		✓	✓	✓
Gauss PRM [9]		✓	✓	✓
Visibility Rtmp[21]	✓	✓	✓	✓
MAPRM [28]		✓	✓	✓
Ariadne’s Clew [7]	✓	✓	✓	✓
RRT [22]			✓	✓
Dilated Spaces [15]			✓	✓
Closest VE (surfaces) [12]		✓	✓	✓
User Input [6]				✓

Table 1: Many more shaded graduations between ‘less’ and ‘most’ clutter exist than could be shown here.

based on sample characteristics and connection results.

- Assign subregions to one or more planners which are judged promising.
- Provide stopping criteria for roadmap construction.

As described in [12], we have already developed the framework. To date it includes our Obstacle-Based *PRM* (*OBPRM*) as well as our interpretations of several other *PRM*s and *PRM* variants. We have made a proto-type of this framework available to selected researchers and expect to have a publicly available version by December 2000. Thus much of the basic functionality for our meta-planner already exists. The control flow of our meta-planner is described by the following pseudo-code.

META-PLANNER PSEUDO-CODE

1. BasicPRM “to distinguish free regions”
2. **while** Stopping conditions NOT satisfied,
3. Partition regions into subregions
4. Assign subregions to more sophisticated planners
5. **endwhile**

In Step 1 we always use a basic *PRM* to discover and map all the open areas of C_{free} , the union of all collision-free robot configurations. Basic *PRM* methods perform this function quickly and efficiently and they are very simple to implement.

In Step 2 the results of the previous step are used to determine if stopping conditions are satisfied or not. A stopping condition in Step 2 may be any one of the following. *i*) Evidence exists that the entire workspace is satisfactorily mapped. *ii*) Mapping progress has ceased and all motion planners which show promise have already been applied to all appropriate subregions. *iii*) Maximum computation time has expired.

Step 3 is responsible for division of the entire problem into subproblems (subregions).

In Step 4, we assign subregions to more sophisticated planners than those to which they have previously been subjected. We describe as ‘more sophisticated’ those methods which are algorithmically more complex as well as certain instantiations of relatively simple methods. Since *PRM*s are, in general, probabilistically complete, we refer to them as ‘more sophisticated’ if we can limit them to smaller regions and run them longer.

An important issue when working with probabilistic roadmaps is controlling the size of roadmaps. We prefer to represent a workspace with as small a map as possible. Nodes and edges represent redundant information when they lie in areas which are already well-covered by other nodes and edges in the graph. Roadmaps with many redundant nodes and edges are more expensive to build (preprocessing), search (run-time), and maintain (re-use). Oversampling a workspace can be avoided *i*) by sampling the workspace in a limited way, sampling more only as necessary, *ii*) by confining the operation of a motion planner to its regions of strength, *iii*) and by filtering out redundant information as it is recognized. The first technique is utilized by many existing *PRM*s, the remaining two we will explore in more detail.

3 Characterizing Regions

Analyzing *PRM*s and characterizing configuration spaces and roadmaps has proven remarkably difficult. Under the assumption that properties assuring a certain ‘goodness’ of the space hold, it has been shown that the running time of the basic *PRM* grows slowly with the probability of tolerable failure [16, 18]. As noted [5], this is less than practically useful because the quantification of ‘goodness’ is assumed, rather than perceived or calculated.

In order to confine the operation of a motion planner to its regions of strength, we must be able to characterize regions. We would like to do so based on readily available generation and connection phase results of roadmap construction. Because a motion planner can collect this information during its regular execution, it is computationally free. In this paper, we present four characterizing measures and discuss the collection and interpretation of each.

First, the ratio of non-colliding nodes to all nodes sampled is one simple measure which we call the FreeSpace Ratio.

- $FreeSpace\ Ratio = \frac{\#free\ nodes}{\#sampled\ nodes}$

In regions where the robot is allowed completely “free” movement, this ratio will be one. The more “cluttered” the region, the closer the ratio approaches zero.

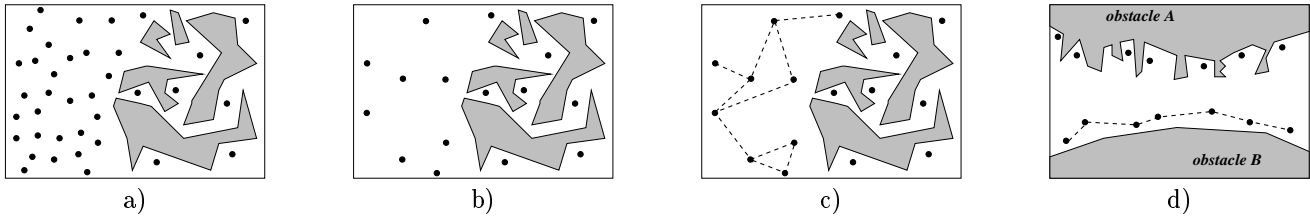


Figure 2: Dots represent robot configurations (nodes). Shaded areas are obstacles. Dashed lines are straightline connections. *a)* FreeSpace Ratio varies left to right. *b)* FreeSpace Ratio ambiguous. *c)* Number and size of connected components varies left to right. *d)* A is too “rough” for connections between near neighbors (adjacent dots). B is “smooth”.

Figure 2a shows a workspace which we assume to have been sampled uniformly. Each of the small black dots represents a collision-free robot configuration (node) in the sample. The FreeSpace Ratio for the “free” left half of the figure is one. The FreeSpace Ratio for the “cluttered” right half, is approximately $1/3$. The value approximates the amount of free space there.

Referring to Figure 2b, we observe the FreeSpace Ratio is not unambiguous. Figure 2b shows, sampled much less densely, the same workspace as in Figure 2a. In this case, assume all nodes were collision-free. That is, although collisions were possible on the right, by chance, the sample did not contain any. The number of collision-free nodes on the left and right half of the space is equivalent. Therefore, although the two halves of the figure are quite different, for the node sample shown, the difference cannot be detected using the FreeSpace Ratio. Other characterization measures are necessary.

Second and third, we consider two more measures:

- *number of connected components*
- *size of connected components*

In Figure 2b, the FreeSpace Ratio was insufficient to show the difference between the two halves of the example region. However, connection results quickly disambiguate the situation. Between any pair of collision free nodes, in a truly free region, a collision free edge exists. The same does not hold in a cluttered region. In Figure 2c we observe a marked difference in the number of collision-free straightline connections between members of the sample. On the left, the number of connected components is small and their size is large. On the right, the number of connected components is large and their size is small. The difference allows us to again characterize the left half of the figure as “free” and the right half as “cluttered”.

Finally, given that nodes can be obtained arbitrarily close to configuration obstacle surfaces [2, 9], the results of connection attempts between nodes near the surface of an obstacle can be used to provide a surface characterization for the obstacle.

- $SurfRatio = \frac{\#surface\ connections}{\#surface\ connections\ attempted}$

For a “rough” obstacle, the Surf Ratio will be near zero. For a smooth one, Surf Ratio will be nearer to one. In Figure 2d, the surfaces of obstacle A and obstacle B have been sampled as shown by the black dots. The dashed lines are roadmap edges made by straightline local planners. The lack of successful straightline connections between near neighbors located near the surface of obstacle A indicate its surface is “rougher” than that of obstacle B where the surface presents fewer impediments to near neighbors connections.

4 Filtering

A good roadmap will accurately reflect the connectivity of the workspace. An important issue when working with probabilistic roadmaps is controlling their size. It is preferable to represent a workspace with as small a map as possible. Nodes and edges represent redundant information when they lie in areas which are already well-covered by other nodes and edges in the graph. Roadmaps with many redundant nodes and edges are more expensive to build (pre-processing), search (run-time), and maintain (re-use). Applying filters which recognize and delete, or prevent the addition of, redundant information from a roadmap allows us to keep the roadmaps compact.

The benefits of intelligent filtering at each stage of roadmap construction (e.g. node generation and node connection) are not limited to any specific motion planning algorithm but rather can be generally applied.

In a basic *PRM*, the computational cost of building the roadmap is limited from the outset. The method generates no more than n nodes (robot configurations) and attempts to connect each to no more than k of its nearest neighbors. The size of the roadmap is limited by the user’s choice of n and k . Limiting the sampling density to n is a primitive filter. The limitation of connection attempts to the k closest neighbors is another form of filtering – on inter-node distance. For each node, its neighbors are sorted by distance and connection will be attempted with no more than k of them.

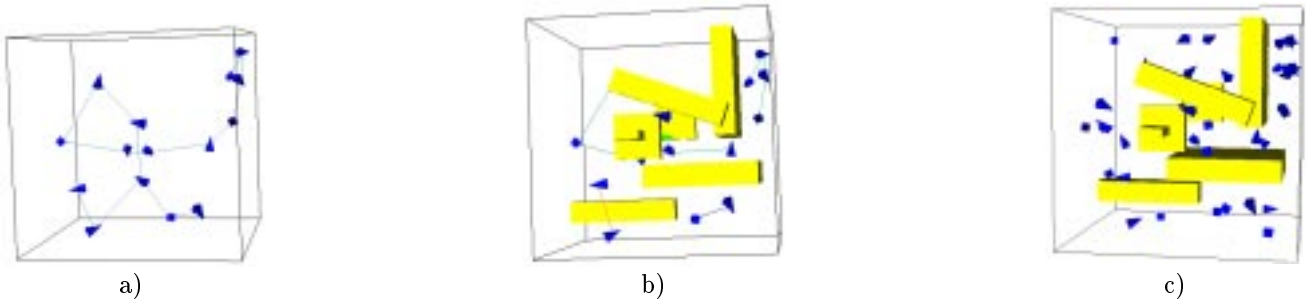


Figure 3: *a)* Suspect ‘free’ by FreeSpace Ratio. *b)* Suspect ‘free’ by FreeSpace Ratio. Detect ‘clutter’ by connected components info. *c)* Detect ‘clutter’ by FreeSpace Ratio.

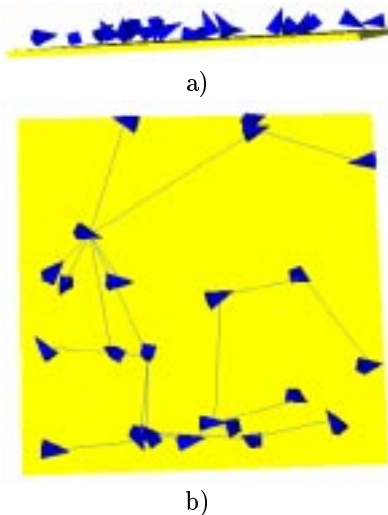


Figure 4: Smooth surface obstacle. Dark triangular polyhedrons are roadmap nodes. Lines are roadmap edges. *a)* side view *b)* top view

More sophisticated methods filter nodes based on their distance from obstacle surfaces rather than from each other. In cluttered environments and narrow passages, the enclosing obstacles are what limits the robot’s movement. Filtering the sample to retain only those nodes close to obstacle surfaces makes the discovery of narrow corridors delineated by those surfaces more likely. Gaussian *PRM* provides such a filter [9] while *OBPRM* generates such nodes directly [2, 3].

Without filtering of some kind, growth planners [7, 22] can oversample an area. Filtering also serves to keep the growth planner limited to the difficult area. If excessive growth seems possible from a sample seed, the seed may not be in a difficult region or the planner may have grown away from one. Growth can be pruned away without loss of information. If only limited growth is possible, we can assume that exploration is taking place in a truly difficult region.

Filters can be simple or complex. Some are gen-

eral, such as limiting sampling density, while others are more specialized, such as pruning growth algorithms back into their areas of strength. Nodes and edges represent redundant information when they lie in areas which are already explored and represented in the graph. Filtering applied to every phase of *PRM* roadmap construction keeps roadmap sizes reasonable by preventing the addition of redundant information.

5 Results

In this section we demonstrate the results of filtering one *PRM* method (*OBPRM*, [2]) to obtain good seeds for a growth planner (RRT, [22]). We also demonstrate the characterization of various spaces and surfaces using the measures discussed in Section 3. Knowing what kinds of regions we are dealing with allows us to take advantage of the summary provided in Table 1. There we list many probabilistically based motion planners and indicate the types of regions where each can expect strong performance. When selecting any planner, points which must also be considered are the usual ones of cost, complexity and availability.

All experiments were conducted on a PC i686 running Linux. The code, written in C++, implements a 3D motion planning testbed for rigid bodies and general articulated linkages.

5.1 Free vs. Cluttered

In our first set of experiments, we consider characterizing “free” versus “cluttered” regions. Results are reported in Table 2, where the FreeSpace Ratio values are recorded in the column labeled ‘ $\frac{V}{rV}$ ’. For the three experiments, **free**, **clutterA**, and **clutterB**, we use the same elongated pyramidal robot. Two different workspaces, one free of obstacles, Figure 3a, and the other somewhat cluttered, Figure 3b, are used. In the figures, the wire frame bounding box delimits the workspace. Obstacles are large light-shaded polyhedrons. The small dark objects are robot configura-

CHARACTERIZE - Free vs. Cluttered								
	$\frac{V}{rV}$	V	rV	cc	1	2	3	iso
free	1	15	15	1	15	0	0	0
clutterA	1	15	15	7	4	3	2	2
clutterB	.88	44	50	8	21	10	5	4

Table 2: FreeSpace Ratio values are recorded in column labeled $\frac{V}{rV}$. ‘rV’ is #nodes requested. ‘V’ is #nodes sampled. ‘cc’ values indicate #number of connected components in the roadmap while those labeled by numbers give the size in nodes for each of the largest 3 cc’s. ‘iso’ is #isolated nodes.

tions (nodes). Lines between nodes represent roadmap edges made by a straightline local planner.

In **free**, all sampled nodes are, trivially, collision-free. The FreeSpace Ratio is 1. The roadmap is a single large connected component of nodes as shown in Figure 3a.

For the **clutterA** experiment, we fit obstacles into the **free** workspace in such a way as to not collide with any of the nodes generated in the **free** experiment. Figure 3b shows a basic *PRM* map built for the region. Because of the adversarial design of the experiment, the FreeSpace Ratio value is misleading. However, we can refer to the number and size of the connected components in the roadmap and correctly conclude that the region is “cluttered”.

In **clutterB**, we sample more densely than we did for the same workspace in **clutterA**. Figure 3c shows the sample. In this case the FreeSpace Ratio value is sufficient to determine the region is “cluttered”.

5.2 Rough vs. Smooth

In our second set of experiments, we characterize smooth versus rough surfaces and suggest a method to connect, into the roadmap, nodes which are located close to a rough surface. Results are reported in Table 3, where the Surf Ratio values are recorded in the column labeled $\frac{E}{rE}$. For the three experiments, **smooth**, **roughA**, and **roughB**, we again use the pyramidal robot. After running the *OBPRM* planner, we consider the Surf Ratio for three different surfaces.

In **smooth**, shown in Figure 4, the surface is completely smooth. Connection between near neighbor configurations should be simple. The Surf Ratio value (0.58) is not 1 because at times nodes were so near the surface that the straightline path between them collided with the surface. Because the surface was smooth, however, these failures did not prevent the planner from connecting all nodes to the same connected component in the roadmap.

For **roughA**, shown in Figure 5, the obstacle surface contained many protrusions which blocked the simple local planner from making connections between

CHARACTERIZE - Smooth vs. Rough								
	$\frac{E}{rE}$	E	rE	cc	1	2	3	iso
smooth	.58	22	38	1	23	0	0	0
roughA	.15	20	134	4	18	3	2	1
roughB	.03	5	163	15	4	3	1	13

Table 3: Surf Ratio values are recorded in column labeled $\frac{E}{rE}$. ‘rE’ is #surface connections. ‘E’ is #surface connections attempted. Other values are the same as described for Table 2.

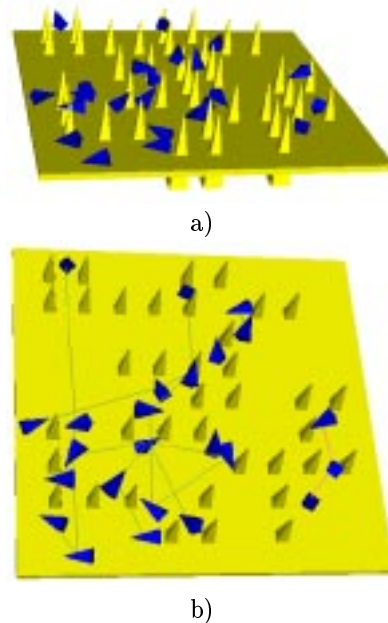


Figure 5: RoughA surface. a) side view b) top view

some near neighbors. This is reflected in a correspondingly smaller Surf Ratio (0.15) and the presence of several connected components in the roadmap. For **roughB** the obstacle surface is identical to that of



Figure 6: A wireframe bounding box encloses the workspace. *ClosestVE* uses a straightline local planner to connect small robot block configurations perpendicularly to the single pre-existing roadmap edge passing from left to right over the large vertical obstacle walls.

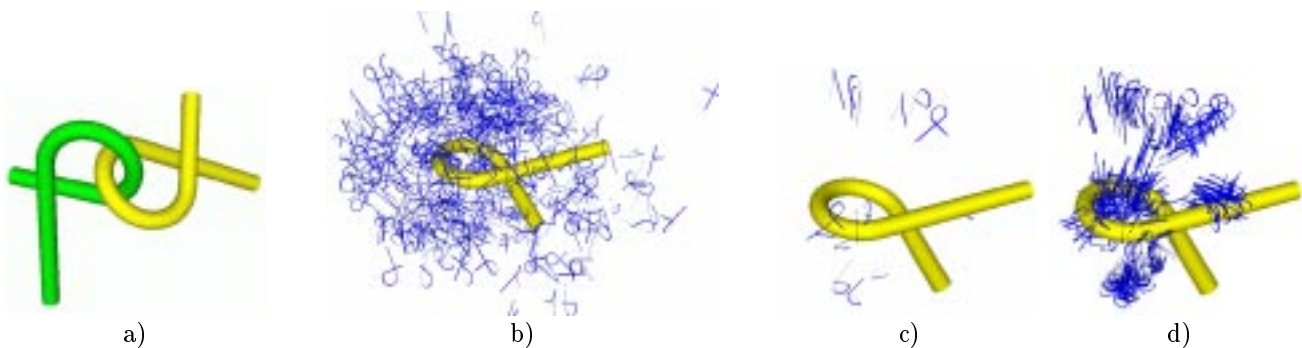


Figure 7: a) Alpha Puzzle with full size robot. b) *OBPRM* generated nodes, each scaled (.1). c) Filter on connected components. d) Use RRT to grow seed nodes.

roughA with the exception that the protrusions have greater volume. The Surf Ratio is very small (0.03) and there are not only many more connected components, but also many isolated nodes, those which could not be connected to any of their neighbors.

We have developed a connection method for which a “rough” surface is a strength area. *ClosestVE* can be used to connect nodes generated near the surface of “rough” obstacles to nearby pre-existing roadmap edges and nodes.

Figure 6 demonstrates the abilities of *ClosestVE* [12]. The small rectangular objects shown are different robot configurations (nodes) which are in the roadmap. Roadmap edges, shown as lines between roadmap nodes (small blocks), are made by a straight-line local planner.

Initially the roadmap consists of all the nodes shown and a single edge between two of them. That edge connects two nodes located at the extreme left and right of the wireframe bounding box. It lies above the top edges of the larger parallel obstacle walls below. The 86 nodes shown were connected to the pre-existing roadmap edge using *ClosestVE*. *ClosestVE* can connect new nodes to roadmap nodes and the implicit nodes on edges. Other connection methods only consider inter-node connections.

5.3 Filtering

In our final set of experiments, we apply a filter to nodes generated for our hardest example problem. The alpha puzzle, as shown in Figure 7a, consists of two identical twisted tubes. One tube is the robot, the other an obstacle. Going from linked to unlinked requires the robot to simultaneously translate and rotate in a very specific sequence which has so far eluded all *PRM* attempts to capture. In our final set of experiments we use *OBPRM* to generate a preliminary map. Robot nodes generated near the outer circumference of the obstacle are easily connected to one another. Those on the inside are more difficult. We want the

ones on the inside. So the map is filtered. Only nodes which could not be connected to any other roadmap node are retained. We can use the filtered set of nodes as seeds for one of the growth paradigm planners, our interpretation of the Rapidly-Exploring Random Tree (RRT) [22]. Figure 7b-d shows the result. In all the figures, the large tube is the obstacle. The small dark tubes are scaled (.1) versions of the robot tube. The scaled versions of the robot sometimes appear in collision but the full sized versions never are. Figure 7b shows the map nodes resulting from the initial application of *OBPRM*. Figure 7c shows the the isolated nodes remaining in the roadmap after all the larger connected components have been removed. Figure 7d shows the progress that RRT is able to make with the “good” seed nodes provided by *OBPRM* after filtering.

6 Conclusion

We have proposed an iterative adaptive meta-planner for motion planning. Many components of such a system are available today. Our group has developed a basic framework which provides implementations of our own algorithms and those (our interpretations) of others.

To implement our meta-planner we need to subdivide the problem environment and and characterize the subregions. Towards that end, we use and recommend the basic *PRM* when confronted with an unexplored instance of the motion planning problem. A basic *PRM* builds a nice roadmap for open areas and allows us to gather information with which to characterize “free” versus “cluttered” regions. Such information can drive the next iteration towards building a better roadmap.

We highly recommend an Obstacle-Based *PRM* (*OBPRM*) in areas identified as “cluttered”. For obstacle surfaces which can be identified as “rough”, we present *ClosestVE* as a connection method for nodes generated near the surface. Such nodes cannot easily

connect to one another because of the surface irregularities lying between them.

Finally, to keep computational costs down, the roadmap must be kept compact. We propose the filtering of probabilistically generated information which is redundant and discuss several different types of filters.

Acknowledgements

We would like to thank the robotics group at Texas A&M and the designer of the alpha puzzle, Boris Yamrom of the Computer Graphics & Systems Group at GE's Corporate Research & Development Center.

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 *Proc. Int. Workshop on Algorithmic Foundations of Robotics (WAFR)*, pages 155–168, 1998.
- [3] N. M. Amato and Y. Wu. A randomized roadmap method for path and manipulation planning. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 113–120, 1996.
- [4] Ta. Asano, Te. Asano, Leonidas J. Guibas, J. Hershberger, and H. Imai. Visibility-polygon search and Euclidean shortest paths. In *Proc. 26th Annu. IEEE Sympos. Found. Comput. Sci.*, pages 155–164, 1985.
- [5] J. Barraquand, L.E. Kavraki and J.C. Latombe, T.Y. Li, R. Motwani, and P. Raghavan. A random sampling scheme for path planning. *Int. J. of Rob. Res.*, 16(6):759–774, 1997.
- [6] O. B. Bayazit, G. Song, and N. M. Amato. Enhancing randomized motion planners: Exploring with haptic hints. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 529–536, 2000.
- [7] 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.
- [8] R. Bohlin and L. E. Kavraki. Path planning using lazy prm. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 521–528, 2000.
- [9] 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)*, pages 1018–1023, 1999.
- [10] J. Canny. *The Complexity of Robot Motion Planning*. ACM – MIT Press Doctoral Dissertation Award Series. MIT Press, Cambridge, MA, 1987.
- [11] J. Canny. A new algebraic method for robot motion planning and real geometry. In *Proc. 28th Annu. IEEE Sympos. Found. Comput. Sci.*, pages 39–48, 1987.
- [12] L. K. Dale. *Optimization Techniques for Probabilistic Roadmaps*. PhD thesis, Texas A&M University, College Station, Texas, 1999.
- [13] J. E. Hopcroft, J. T. Schwartz, and Micha Sharir. On the complexity of motion planning for multiple independent objects: P-space hardness of the “Warehouseman's Problem”. *Internat. J. Robot. Res.*, 3(4):76–88, 1984.
- [14] T. Horsch, F. Schwarz, and H. Tolle. Motion planning for many degrees of freedom – random reflections at c-space obstacles. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 3318–3323, 1994.
- [15] D. Hsu, L. Kavraki, J.-C. Latombe, R. Motwani, and S. Sorkin. On finding narrow passages with probabilistic roadmap planners. In *Proc. Int. Workshop on Algorithmic Foundations of Robotics (WAFR)*, 1998.
- [16] L. Kavraki, M. Kolountzakis, and J.-C. Latombe. Analysis of probabilistic roadmaps for path planning. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, volume 4, pages 3020–3025, 1996.
- [17] L. Kavraki and J. C. Latombe. Randomized preprocessing of configuration space for fast path planning. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 2138–2145, 1994.
- [18] L. Kavraki, J. C. Latombe, R. Motwani, and P. Raghavan. Randomized query preprocessing in robot path planning. In *Proc. ACM Symp. Theory of Computing (STOC)*, pages 353–362, 1995.
- [19] L. Kavraki, P. Svestka, J. C. Latombe, and M. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Trans. Robot. Automat.*, 12(4):566–580, August 1996.
- [20] Y. Ke and J. O'Rourke. An algorithm for moving a ladder in three dimensions. Technical Report JHU-87/17, Dept. Comput. Sci., Johns Hopkins Univ., Baltimore, MD, September 1987.
- [21] J.-P. Laumond and T. Siméon. Notes on visibility roadmaps and path planning. In *Proc. Int. Workshop on Algorithmic Foundations of Robotics (WAFR)*, 2000.
- [22] S. M. LaValle and J. J. Kuffner. Randomized kinodynamic planning. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 473–479, 1999.
- [23] 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.
- [24] M. Overmars. A random approach to path planning. Technical Report RUU-CS-92-32, Computer Science, Utrecht University, The Netherlands, 1992.
- [25] M. Overmars and P. Svestka. A probabilistic learning approach to motion planning. In *Proc. Workshop on Algorithmic Foundations of Robotics*, pages 19–37, 1994.
- [26] J. H. Reif. Complexity of the mover's problem and generalizations. In *Proc. 20th Annu. IEEE Sympos. Found. Comput. Sci.*, pages 421–427, 1979.
- [27] A. Tannenbaum and Y. Yomdin. Robotic manipulators and the geometry of real semialgebraic sets. *IEEE Trans. Robot. Automat.*, RA-3(4):301–307, 1987.
- [28] 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)*, pages 1024–1031, 1999.
- [29] S. A. Wilmarth, N. M. Amato, and P. F. Stiller. Motion planning for a rigid body using random networks on the medial axis of the free space. In *Proc. ACM Symp. on Computational Geometry (SoCG)*, pages 173–180, 1999.
- [30] L. Yang and S. M. LaValle. A framework for planning feedback motion strategies based on a random neighborhood graph. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 554–549, 2000.