Evaluation of the K-closest Neighbor Selection Strategy for PRM Construction

Troy McMahon, Sam Jacobs, Bryan Boyd, Lydia Tapia, Nancy M. Amato

***Abstract*— Probabilistic Roadmap Methods (PRMs) are one of the most used classes of motion planning methods. These sampling-based methods generate robot configurations (nodes) and then connect them to form a graph (roadmap) containing representative feasible pathways. A key step in PRM roadmap construction involves identifying a set of candidate neighbors for each node. Traditionally, these candidates are chosen to be the *k*-closest nodes based on a given distance metric. In this paper, we evaluate the *k*-closest neighbor selection strategy and compare it to other strategies for identifying candidate neighbors, including an all-pairs connection strategy, a random connection strategy and a distance-based strategy that selects all nodes within a certain distance of the base node. To our knowledge, ours is the first study to examine the effectiveness of the *k*-closest strategy and to compare it to potential alternatives. We evaluate *k*-closest in comparison to other methods in a set of homogeneous and heterogeneous environments with rigid and articulated robots. We show that *k*-closest is able to find neighbors that are more connectible than randomized methods, particularly in difficult environments and that *k*-closest is better at adapting to differences in sample density than the distance method.**

I. Introduction

The general *motion planning* problem involves finding a valid path for a movable object (robot) from a start to a goal configuration in a given environment. Motion planning is an important component of many applications, including computer-aided design [2], robotics [21], virtual reality simulations [10], and bioinformatics [4] [9]. The motion planning problem is regarded intractable, as the complexity of an exact method grows exponentially with the complexity of the robot [28]. Randomized sampling-based methods are able to solve many motion planning problems exact methods cannot.

One widely used sampling-based method is the Probabilistic Roadmap Method (PRM) [18]. PRMs sample motions in *configuration space* (C-space), in which points correspond to robot configurations. PRMs are constructed by randomly generating free samples in C-space then connecting them using a local planner. One of the key steps in PRM construction is node connection. Ideally, roadmap connectivity should reflect the connectivity of the underlying C-space. From this perspective, the best strategy would be to attempt to connect all $\theta(n^2)$ pairs of nodes. However, the cost of all these connection attempts is not feasible for any but the simplest of problems. Hence, the selection of the candidate neighbors is crucial to both roadmap quality and efficiency.

The objective of a good neighbor selection strategy is to identify a set of candidate neighbors that have a high probability of being connectible by the local planner and that are useful in terms of roadmap connectivity. The most commonly used method for neighbor selection in PRMs uses nearest-neighbor search to select the k nodes that are closest to the node in question, where k is typically some relatively small, fixed constant.

In this paper, we compare *k*-closest with several alternatives, including an all-pairs connection strategy, a random connection strategy and a distance-based strategy that selects all nodes within a certain distance of the base node. While alternatives to *k*-closest have been proposed and studied individually [22], [25], [23], [18], to our knowledge, there has not been a study examining the effectiveness of the *k*-closest strategy or comparing it to other potential strategies for identifying candidate neighbors. We conduct this study across a set of homogeneous and heterogeneous environments of varying difficulty with both rigid and articulated robots. Our study evaluates how different neighbor selection methods perform across a variety of environments, and distance metrics. We evaluate the different methods using a set of evaluation metrics that assess how good the methods are at locating connectible neighbors and producing well connected roadmaps.

Our findings confirm that *k*-closest is well suited to PRMs, outperforming the other strategies on difficult problems. We show that distance based methods perform comparably to *k*-closest in homogeneous environments if the right distance threshold is used but that *k*-closest is superior in heterogeneous environments. We also show that methods with a small amount of randomness can produce roadmaps with more edges and a greater connectivity than *k*-closest at the expense of having a higher connection cost. The main reason for this is that they test more unique neighbor pairs than *k*-closest does. Due to space constraints we were not able to include all of our results, these can be found in [24].

II. Preliminaries and Related work

Neighbor selection is a key step in the success of a PRM strategy. In this section we explain the role of various neighbor selection strategies in PRMs and also explore how they have been used previously to achieve specific results.

A. Probabilistic Roadmap Methods

PRMs [18] are a class of sampling-based motion planners that build a graph (roadmap) in which vertices are valid robot configurations and edges represent feasible transitions between configurations. This graph then encodes representative, feasible pathways that can be used to connect given start and goal configurations.

PRMs have been applied to a wide range of problems [32]. They have been used for path planning with mobile robots [20] [8] [5], humanoid robots [19] and reconfigurable robots [1]. They have been applied to biological problems including analysis of biological structures [11] and protein folding [4]. They have been used in industrial automation and path planning with robotic manipulators [30] [34] [17] [26].

In [12], Geraerts and Overmars perform a reachability-based analysis of the PRM method. In this study they evaluate existing methods based on how well they cover environments and how connected the roadmaps they produce are. This study shows that existing methods are capable of generating node sets that cover the environment well and that the major difficulty in roadmap construction is connecting these nodes, especially in difficult narrow passage problems. The study concludes that the major hurdle in roadmap construction is not covering the environment but generating a connected roadmap. In [13], Geraerts and Overmars show experimentally that the main difficulty in PRM construction is constructing a roadmap whose connectivity represents the connectivity of C-space.

This research supported in part by NSF awards CRI-0551685, CCF-0833199, CCF-0830753, IIS-096053, IIS-0917266, NSF/DNDO award 2008-DN-077-ARI018-02, by the DOE NNSA under the Predictive Science Academic Alliances Program by grant DE-FC52-08NA28616, by THECB NHARP award 000512-0097-2009, by Chevron, IBM, Intel, Oracle/Sun and by Award KUS-C1-016-04, made by King Abdullah University of Science and Technology (KAUST). This research used resources of the National Energy Research Scientific Computing Center, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231.

B. Candidate Neighbor Selection Approaches

There have been variants proposed for identifying neighbors during the connection phase of PRM construction [22]. The most common strategy is the so-called *k*-closest, which selects the set of *k* points that are nearest the query sample, i.e., its *k*-nearest neighbors, where *k* is typically some small constant. This simple to apply strategy was used in many PRMs, including the original PRM [18], OBPRM [33], and GaussPRM [7]. The intuition behind the use of the set of closest points is that the costs for verifying the validity of the connection are reduced and, depending on the problem, shorter connections are more likely to be collision-free [22].

Another simple method is a *Distance* method that identifies neighbors within some fixed distance of the query point. A drawback of this method is that some knowledge of the problem is required in order to determine an appropriate distance. If the distance is too big, too many connections will be attempted, resulting in a very expensive connection phase. If it is too small, then many connections will be excluded and the roadmap will be poorly connected. The original PRM implementation[18] included a variation of the distance method with an upper bound on the number of neighbors. The distance method is commonly used for grid-based PRM methods [29]. It is also commonly used in biophysical simulations where certain cutoffs are standard, e.g., a RMSD of atom positions between molecular conformations [23]. Geraerts and Overmars [15], [14] include a distance method in a study of the impact of sampling, node selection/adding strategy and local planning on coverage and connectivity of PRMs. This study shows that the *k*-closest method is fairly well suited for generating connected roadmaps. Our study differs from their work in the following aspects. First, we study how introducing randomness and how decreasing the proximity of neighbors effects roadmap connectivity/quality. We also differ in that we study how environment difficulty and node density affect the performance of the methods.

In [13], Geraerts and Overmars explore a *Visibility*-based connection strategy. Visibility neighbors are those that are visible (connectible with a straight-line) from the point. However, this often requires special placement of the points as in the variant, Visibility Roadmaps [25].

Geraerts and Overmars[13] also explore a *Component*-based selection strategy. This strategy attempts *k* connections to each connected component [22]. Depending on the number of components and the value of *k*, there could be a large number of attempts.

As part of his work in [6], Boden presents a grid based neighbor selection policy. The nodes in this method are are arranged in a grid in C-space and connections are attempted between adjacent nodes on the grid. If a path is not found, then additional grid points are introduced at a higher resolution in areas that could not be connected and connections are attempted between adjacent nodes in the higher resolution grid.

Another direction in neighbor selection that has been explored is the examination of the impact of using approximate and more efficient strategies for computing *k*-closest. In [27] Plaku et. al. study how approximate neighborhood finders perform when applied to to the motion planning problem.

III. Candidate Neighbor Selection Policies

In this paper we compare several strategies for selecting neighbor candidate sets for the PRM connection phase. In addition to the traditionally used *k*-closest strategy, we consider an all-pairs connection strategy (used as a baseline comparison for each method), random connection strategies, and distance-based strategies that select nodes within a certain distance.

In our discussion, we will define selection policies that operate on a candidate set of neighbor cfgs (V_c) and a source configuration (v_s). The set of all configurations in the roadmap is V, with $V_c \subseteq V$ and $v_s \in V$. Most selection policies choose a maximum of k configurations, where k is a user provided constant.

Listed below are a set of basic strategies for selecting candidates from the candidate set V_c. Note that all strategies that use distances (i.e., *k*-closest and distance) depend on the distance metric used.

- **all-pairs**(v_s, V_c)**:** Select all configurations from V_c as connection candidates for v_s.
- *k*-**closest**(v_s, V_c, k)**:** Select the k configurations $v \in V_c$ that are the closest to v_s.
- *k*-**random**(v_s, V_c, k)**.** Select k configurations at random from V_c.
- **distance**(v_s, V_c, d)**.** Select all $v \in V_c$ that are within distance d of v_s.
- **kr-kc**(v_s, V_c, k_1, k_2) This policy selects the closest k_1 nodes then returns k_2 of these nodes at random (where $k_1 \geq k_2$).
  kr-kc(v_s, V_c, k_1, k_2) = *k*-**random**$(v_s,$ *k*-**closest**$(v_s, V_c, k_1), k_2)$

In our study we use the following variations of the kr-kc method:

- **kr-2kc**(v_s, V_c, k) = kr-kc$(v_s, V_c, 2k, k)$
- **kr-4kc**(v_s, V_c, k) = kr-kc$(v_s, V_c, 4k, k)$

The kr-kc methods are a useful tool for studying connectivity because they allow us to control the amount of randomness in candidate neighbor selection. These methods first select a set of more than k nodes and then they randomly select k of these nodes. They allow us to put an upper bound on the rank of the nodes we select. For example if we use kr-2kc we are limiting ourselves to the first $2k$ nodes. Comparing different kr-kc methods will help us to study how reducing the proximity of the candidate neighbors will effect roadmap quality.

IV. Evaluation Metrics

We assess the neighbor selection methods using a set of evaluation metrics. We selected metrics that would allow us to evaluate how effective the methods are at finding connectible neighbors and how good they are at producing well connected road maps. We also chose metrics that will tell us how expensive each of the methods are.

A. All-Pairs Roadmap

The all-pairs roadmap captures the best possible connectivity for a node set. We use the all-pairs roadmap as a baseline for comparing the other methods.

B. Connectivity Metrics

Our first set of metrics evaluate how good the methods are at finding connectible neighbors and how successful they are at producing connected roadmaps. This will help us to determine how the methods affect roadmap quality.

- **Number of Edges**: measures of the number of roadmap edges. This metric will tell us how many edges the methods are able to produce. In general, roadmaps with more edges tend to be better connected.
- **Local Planner Success(%)**: percentage of successful local planner connections. This metric will tell us how effective each method is at finding connectible neighbors. The local planner success is different from the number of edges because the total number of edges attempted is not always the same across the different methods.
- **Connectivity**: The connectivity of a roadmap $R = (N, E)$ is the number of pairs of nodes p, q for which there is a path from p to q in R. This metric indicates how connected a roadmap is. The connectivity of the

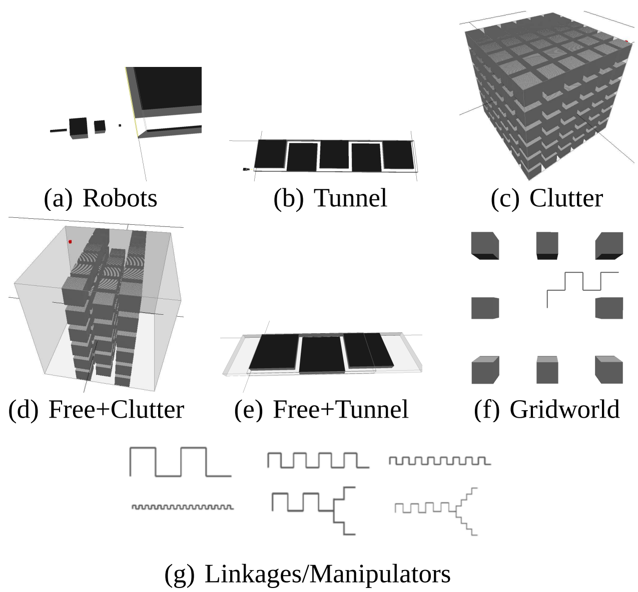


Fig. 1. Environments Studied:(a) Robots Used, (b) Tunnel,(c) Cluttered Homogeneous, (d) Cluttered Heterogeneous, (e) Tunnel Heterogeneous, (f) Gridworld, (g) Linkage and Manipulator Robots

roadmap is normalized over the connectivity of an all-pairs roadmap using the same set of nodes.

C. Cost Metric

This metric will show us how computationally expensive the different methods are which is important in determining the tradeoffs associated with the methods.

- **CD-Calls**: We measure the number of collision detection (CD) calls made during the **connection** phase of roadmap construction. CD-calls are a platform independent metric that is used to measure roadmap construction time.

V. Experiments

We implemented all planners using the C++ motion planning library developed by the Parasol Lab at Texas A&M University. RAPID [16] is used for collision detection computations.

A. Environments

We selected a range of environments to study including an extensive set of rigid body experiments and a set of articulated linkages experiments with varying degrees of freedom. This will allow us to evaluate the performance of the methods across a variety of environment types and difficulties. A list of the environments we used is included in Table I along with the notation we use for them.

1) **Environments:** We chose three homogeneous environments including a narrow passage (Figure 1(b)), a cluttered environment (Figure 1(c)), and a free environment (not shown). Most problems in motion planning can be considered to be comprised of free regions, narrow passages and cluttered regions. The homogeneous environments allow us to draw conclusions about specific types of environments. We also included two heterogeneous environments, a free+cluttered environment (Figure 1(d)) and a free+tunnel environment (Figure 1(e)). These were included to study how the different methods perform in problems with multiple types of regions.

In designing these environments we did as much as we could to allow for meaningful comparisons across them. In order to ensure that the sample density was the same in all the environments we designed all of our environments to have the same volume of free workspace. We also designed the tunnel and clutter environments so that the clearance between the robot and the obstacles will be the same (see Figure 2).

Free Environment: Free environment with no obstacles.

Tunnel Environment: The tunnel environment (Figure 1(b)) consists of a tunnel that is 1x1 unit wide. The tunnel is divided into 9 segments of length 10, these passages are separated by 90 degree turns in the tunnel.

Cluttered Environment: This environment (Figure 1(c)) consists of 216 3x3x3 cube obstacles that were arranged in a 6x6x6 grid and separated by 1 unit of free space.

Free+Cluttered Heterogeneous Environment: The first heterogeneous environment(Figure 1(d)) consisted of 2 regions of free space separated by a cluttered region. The cluttered part of this environment included 97 3x3x3 cubes obstacles that were separated by 1 unit of free space. The free volume in this environment was divided evenly between the cluttered region and the two free regions.

Free+Tunnel Heterogeneous Environment: The second heterogeneous environment (Figure 1(e)) consisted of a tunnel connecting 2 regions of free space. The tunnel was 1 by 1 wide and consisted of 5 segments of length 10. The free volume in this environment was divided evenly between the tunnel and the free regions.

Gridworld Environment: The gridworld environment was a 2 dimensional environment designed for use with the manipulator and articulated linkage robots. In this environment the base of the manipulator/linkage was fixed at the point (0,0). This environment included 8 1x1 square obstacles that are separated by 2 units of free space.

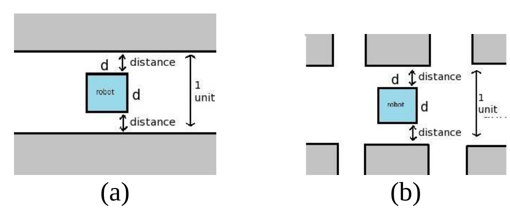


Fig. 2. Here we look at the cross section of the tunnel (a) and clutter (b) environment. The width of the tunnel and the spacing between blocks in the clutter are both the same (1 unit). This means that if we use the same robot in both environments than the clearance between the robot and the tunnel in the tunnel environment will be the same as the clearance between the robot and the cubes in the clutter environment.

B. Robots

In our rigid body experiments we used a set of three cube robots of varying sizes. By changing the size of the robot we can increase or decrease the clearance between the robot and the obstacles of the environment which will change the difficulty of the environment. With smaller robots there will be more clearance and the environment will be easier. With larger robots there will be less clearance and the environment will be more difficult. We also included in our experiments a stick robot that was intended to illustrate the differences between the Euclidean and swept volume distance metrics. A set of linkages and manipulators was used to study how the methods perform with more complicated, higher dimensional robots. These are shown in Figure 1(a,g).

Cube Robots: Our smallest cube robot was a .1x.1x.1 cube. This robot was 1/10 the size of the spacing between obstacles in the tunnel and clutter environments. Our mid-size cube robot was .5x.5x.5 units in size. This robot was 1/2 the size of the spacing between obstacles in the tunnel and clutter environments. Our largest cube robot was a .8x.8.x.8 cube. This robot was large enough that it would barely fit in the tunnel or in between the obstacles in the clutter environment.

Stick Robot: The stick robot was .8x.1x.1 units in size.

Articulated Linkages: We also studied a set of 4 articulated linkage robots (see Figure 1(g)). These robots consisted of a set of linkages that are joined by articulated joints. The base of this linkage was fixed at the point (0,0) in the environment with the first linkage being able to rotate about this point. The linkages were a total of 7 units long with this length distributed equally among the links in the linkage. In all cases the links were were .01 units wide.

- **8 Link Articulated Linkage:** This linkage consisted of 8 links of length .875 units. This robot had 8 degrees of freedom (Dof).
- **16 Link Articulated Linkage:** This linkage consisted of 16 links of length .4375. This robot had 16 Dof.
- **32 Link Articulated Linkage:** This linkage consisted of 32 links of length .2188. This robot had 32 Dof.
- **64 Link Articulated Linkage:** This linkage consisted of 64 links of length .1094. This robot had 64 Dof.

Manipulator: We also included a set of manipulator robots (see Figure 1(g)). The manipulators consisted of an arm and a hand for grasping objects. The arm of the manipulator was a total of $3\sqrt{2}$ (or 4.2426) units long with this length being distributed equally among the links in the arm. The hand part of the manipulator consisted of 2 graspers of length 3. Like the articulated linkage, we used links that were .01 units wide.

- **16 Dof Manipulator:** This manipulator consisted of a total of 16 links and had 16 degrees of freedom. 8 links were in the arm of the robot and 4 were in each grasper of the hand. The links in the arm were of length .5304 while the links in the hand were of length .375.
- **32 Dof Manipulator:** This manipulator consisted of a total of 32 links and had 32 degrees of freedom. 16 links were in the arm of the robot and 8 were in each grasper of the hand. The links in the arm were of length .265 while the links in the hand were of length .1875.

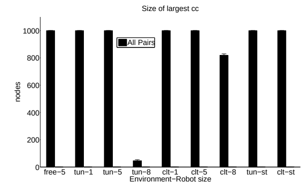
With the manipulator and articulated linkages we varied the dimensionality of the problem by varying the number of links in the linkage. This allows us to study how problem dimensionality impacts each of the methods.

C. Roadmap Construction Methods and Parameters

When running our experiments we varied the node generation strategies and distance metric methods. In our rigid body experiments we used Uniform sampling [18]. In our articulated linkage and manipulator experiments we used the reachable sampler presented in [31]. For a local planner we used a simple straight-line local planner. For our distance metric we chose a Scaled Euclidean distance metric and a swept distance metric. We chose the Scaled Euclidean distance metric because it is one of the most commonly used distance metrics and it has been shown to produce reasonable results [18]. The swept distance between two configurations is the total volume the robot sweeps in work space when moving between them using the specified local planner. This distance metric is generally considered to be very accurate at the expense of being costly to compute[3].

To perform this study we need to select an appropriate number of nodes (n). If the number of nodes is too large then the samples will be too dense and the differences between the difference methods will not be as noticeable. If it is too small, then the samples will be too sparse and we won't be able to connect them. We selected an n value of 1000 because our initial experiments indicated that it was sufficient to produce node sets that were capable of solving most of the environments and capable of producing a meaningful number of connections in all of the environments.

We confirmed experimentally that 1000 nodes was appropriate for the problems we studied. Figure 3 shows that in all of the environments except for the tun-8, using 1000 nodes with all-pairs connection will produce roadmaps where the



(a)

Fig. 3. Size of the largest cc when using all-pairs connection with 1000 nodes and uniform sampling. These results were averaged over 10 runs using 10 different seeds.

largest connected component contains nearly all 1000 nodes. In the tun-8 environment, all-pairs connection produced roadmaps where the largest CC was still a significant size (200 nodes). From these initial experiments we concluded that 1000 nodes is sufficient to produce well connected roadmaps in all of our environments and that it is an acceptable n value for our experiments.

Each experiment was repeated 10 times (using 10 different sets of nodes) and the values shown are averages of the 10 runs. For the k-closest, k-random, and kr-kc methods we used k values of 8 and 16.

Our choice of k values was also based on our initial set of experiments. We selected k values of 8 and 16 because they were sufficiently large to produce large connected components in all of our environments. From what we saw in our results, a k value of 8 was sufficient for this study so we focus most of our attention on this set. We do however include some results from the sets with a k value of 16 for the sake of comparison.

VI. Results

In this section, we evaluate the k-closest neighbor selection strategy and compare it to the other approaches for identifying candidate neighbors. We study the impact of the difficulty of the problem and the homogeneity/heterogeneity of the environment on these issues, and we study how adding small amounts of randomness to candidate neighbor selection effects the structure of the resulting roadmap. Our results show that the distance method performs worse than k-closest in heterogeneous environments. They show that the randomized methods find neighbors that are less connectible than k-closest but that the kr-2kc and kr-4kc methods produce roadmaps that have more edges and are more connected. Our results also show that k-closest roadmaps require fewer CD-calls.

A. Distance-based Neighbor Selection

We evaluate the distance method in comparison to k-closest in the homogeneous and heterogeneous environments. The heterogeneous environments are of particular interest because they show how the methods adapt to differences in sample density.

In each of the homogeneous environments, we chose the version of the distance method that made the closest number of CD-calls to k-closest with a k value of 8. To do this we ran the distance method with many different distances and for each environment we selected the distance that had the closest number of CD-calls to k-closest. Since these versions of the distance method do the same amount of work as k-closest, it is sensible to compare the quality of the resulting roadmaps. The local planner success of this version of the distance method (Figure 4(a)) was similar to k-closest with a k value of 8. The total number of edges generated by the distance metric (Figure 5(a)) was also similar to the number of edges generated by k-closest. The connectivity of the roadmaps produced by the distance method (Figure 6(a)) was similar to k-closest in the Tunnel environments(tun-1, tun-5, tun-8). However, it was slightly lower in the Clutter environments (clt-1, clt-5, clt-8).

Notation	Environment	Robot	Dimensions	Obstacles
free-5	Free	.5x.5x.5 cube	18.5x18.5x18.5	None
tun-1	Tunnel	.1x.1x.1 cube	50x10x1	1x1 tunnel with 8 turns and 9 segments of length 10
tun-5	Tunnel	.5x.5x.5 cube	50x10x1	1x1 tunnel with 8 turns and 9 segments of length 10
tun-8	Tunnel	.8x.8x.8 cube	50x10x1	1x1 tunnel with 8 turns and 9 segments of length 10
tun-st	Tunnel	.8x.1x.1 stick	50x10x1	1x1 tunnel with 8 turns and 9 segments of length 10
clt-1	Clutter	.1x.1x.1 cube	23x23x23	216 3x3x3 cube obstacles arranged in 6x6x6 grid
clt-5	Clutter	.5x.5x.5 cube	23x23x23	216 3x3x3 cube obstacles arranged in 6x6x6 grid
clt-8	Clutter	.8x.8x.8 cube	23x23x23	216 3x3x3 cube obstacles arranged in 6x6x6 grid
clt-st	Clutter	.8x.1x.1 stick	23x23x23	216 3x3x3 cube obstacles arranged in 6x6x6 grid
tun-het-1	Free+Tunnel	.1x.1x.1 cube	30x10x1	1x1 tunnel with 4 turns and 5 segments of length 10
tun-het-5	Free+Tunnel	.5x.5x.5 cube	30x10x1	1x1 tunnel with 4 turns and 5 segments of length 10
cl-het-5	Free+Clutter	.5x.5x.5 cube	23x23x23.1	97 3x3x3 cube obstacles arranged in 3 plains
G-8	Gridworld	8 link articulated linkage	14x14	8 1x1 obstacles
G-16	Gridworld	16 link articulated linkage	14x14	8 1x1 obstacles
G-32	Gridworld	32 link articulated linkage	14x14	8 1x1 obstacles
G-64	Gridworld	64 link articulated linkage	14x14	8 1x1 obstacles
G-man-16	Gridworld	16 Dof manipulator	14x14	8 1x1 obstacles
G-man-32	Gridworld	32 Dof manipulator	14x14	8 1x1 obstacles

TABLE I

ENVIRONMENT AND ROBOT COMBINATIONS WE USED IN THIS PAPER ALONG WITH THE NOTATION WE USED FOR EACH OF THEM

In Figure 7 we study the sensitivity of the distance based methods to the value of *d* in the heterogeneous environments Free+Tunnel (a-c) and Free+Cluttered (d-f). In particular we study the performance of the distance method for a variety of *d* values. For smaller *d* values, the distance method produces roadmaps with few edges and poor connectivity. For larger values it produced roadmaps with the same connectivity as *k*-closest (or higher in the case of clutter environment). However, these roadmaps had many more edges than *k*-closest and required many more CD-calls to construct. This is a problem because it increases the storage size and the query time of the roadmap. The distance method did not produce roadmaps that were as connected as *k*-closest for *d* values that were smaller than .2 in the tun-het-5 environment and .4 in the clt-het-5 environment. At these *d* values, the distance method is already producing many more edges than *k*-closest because it is over connecting the free regions. In heterogeneous environments, the distance method is unable to obtain the same connectivity as *k*-closest without producing many more edges and making many more CD-calls.

B. Effects of Proximity

One topic that we study in this paper is the effect of neighbor proximity on roadmap quality. This will tell us how important it is for methods to find neighbors that are close to a node. We study how introducing randomness will impact our ability to connect candidate neighbors, and how it will effect the number of edges in a roadmap. We study how this randomness will effect the overall connectivity the roadmap. We show that more local methods produce neighbors that are more connectible but slightly random methods can produce roadmaps that are more connected. We also show that reducing the proximity of the nodes we select has an impact on the structure of the roadmaps.

For this we looked at the *k*-closest method, *k*-random method and the kr-kc methods. Recall that *k*-closest gives us the *k* closest nodes, kr-2kc and kr-4kc give us neighbors from the closest 2*k* and closest 4*k* nodes and *k*-random can return any of the nodes. We can therefore say that *k*-closest gives us neighbors that are in the closest proximity to a node followed by kr-2kc, kr-4kc and *k*-random.

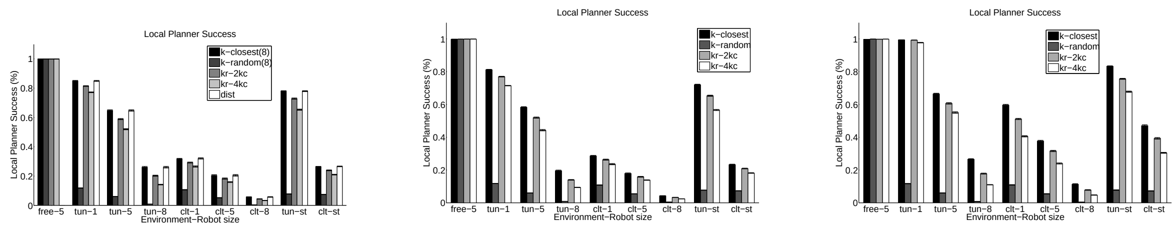
1) **Effects on Roadmap Connectivity:** We studied how introducing randomness will impact our ability to connect candidate neighbors and how it will effect the number of edges in a roadmap. We also study how this randomness will effect the overall connectivity of the roadmap. Our findings show that the *k*-closest method produces neighbors that are more connectible than other methods, and that the connectability of neighbors decreases as proximity decreases. We also show that reducing the proximity of neighbors slightly results in roadmaps that have more edges and are better connected.

Local planner success: We looked at the local planner success of these methods in the free-5, tun-1, clt-1, tun5, clt-5, tun-8 and clt-8 environments ((Figure 4(a)). In all cases (except for the free environment) *k*-closest has the highest local planner success followed by kr-2kc, kr-4kc and *k*-random. From this we can conclude that the probability of finding neighbors decreases with proximity. This is expected because the distance metric is a heuristic for how difficult nodes will be to connect. As long as we choose an appropriate distance metric, we would expect local planner success to decrease with neighbor proximity.

Of more interest is the relative effect of neighbor proximity on the different environments. In both the Tunnel and Clutter environments, the effects of reducing neighbor proximity increases significantly as the problems become more difficult. In the free environment and the easier Tunnel and Clutter environments, we see that the number of edges (Figure 5(a)) increases as we go from *k*-closest to kr-2kc and kr-4kc. With the more localized methods, it is very likely that a node's candidate neighbors will also have the node in their candidate neighbor set. Since our roadmaps are undirected, we only add one edge between each pair of nodes, the total number of unique edges that are tested will be less for the more localized methods. As neighbor proximity decreases, so does the likelihood that two nodes will contain eachother in their candidate neighbor set. Consequently, the number of edges tested increases as neighbor proximity decreases and the number of edges will also increase. It is important to realize that *k*-closest is in reality sampling fewer than *k* edges per node. In the tun-8 and clt-8 environments we see that the number of edges decreases as we go from *k*-closest to kr-2kc and kr-4kc(Figure 5(a)). Even though the number of candidate edges we are testing is larger for the more randomized methods, the local planner success is smaller and the number of edges is less. The *k*-random method produces the most edges in the free environment. However, it produces considerably fewer edges than the other methods in the tunnel and clutter environments.

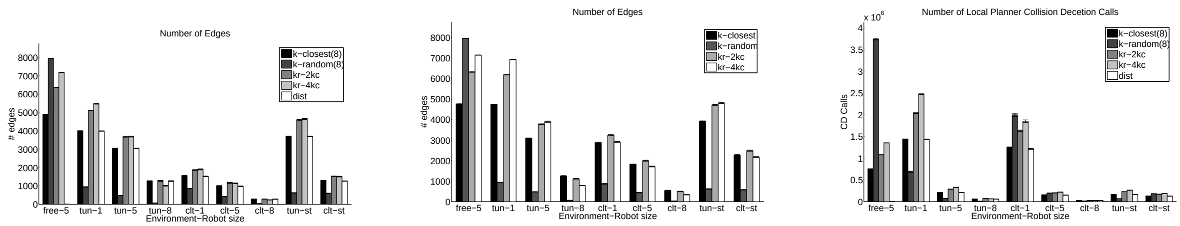
Connectivity: We next looked at the connectivity of the roadmaps produced by the different methods (see Figure 6(a)). The connectivity of *k*-closest and the kr-kc methods was either 1 or close to 1 in the free, tun-1, and tun-5 environments. Since the connectivity is normalized over the connectivity of the all-pairs roadmaps, this means that *k*-closest and the kr-kc methods are producing roadmaps that are as connected as the all-pairs roadmaps. At the same time, the connectivity of *k*-closest and the kr-kc methods was less than 1 in the tun-8 environment and all of the Clutter environments (especially the clt-8 environment). The *k*-random method performed poorly in all of the Tunnel and Clutter environments consistently producing roadmaps that were less connected than the other methods. This shows that methods with too much randomness cannot generate connected roadmaps even in relatively easy environments such as the tun-1 and clt-1.

In the tun-5, clt-5, tun-8 and clt-8 environments, the kr-kc methods consistently produced more connected roadmaps than *k*-closest. As we saw in the previous section, the kr-kc methods produced more edges that *k*-closest. The connectivity results show that these additional edges can contribute to



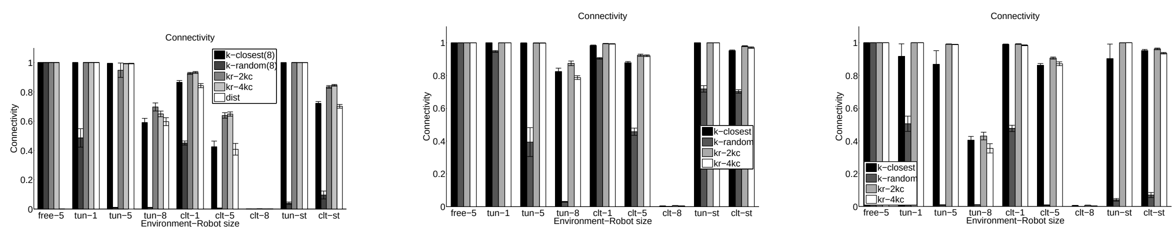
(a) Local Planner Success, Uniform Euclidean, k=8 (b) Local Planner Success, Uniform Euclidean, k=16 (c) Local Planner Success, Uniform Swept, k=8

Fig. 4. Local planner success across a variety of connection methods and environments.



(a) Number of Edges, Uniform Euclidean, k=8 (b) Number of Edges, Uniform Swept, k=8 (c) CD-calls, Uniform Euclidean, k=8

Fig. 5. Number of edges and Number of CD-calls across a variety of connection methods and environments.



(a) Connectivity, Uniform Euclidean, k=8 (b) Connectivity, Uniform Euclidean, k=16 (c) Connectivity, Uniform swept, k=8

Fig. 6. Roadmap connectivity across a variety of connection methods and environments.

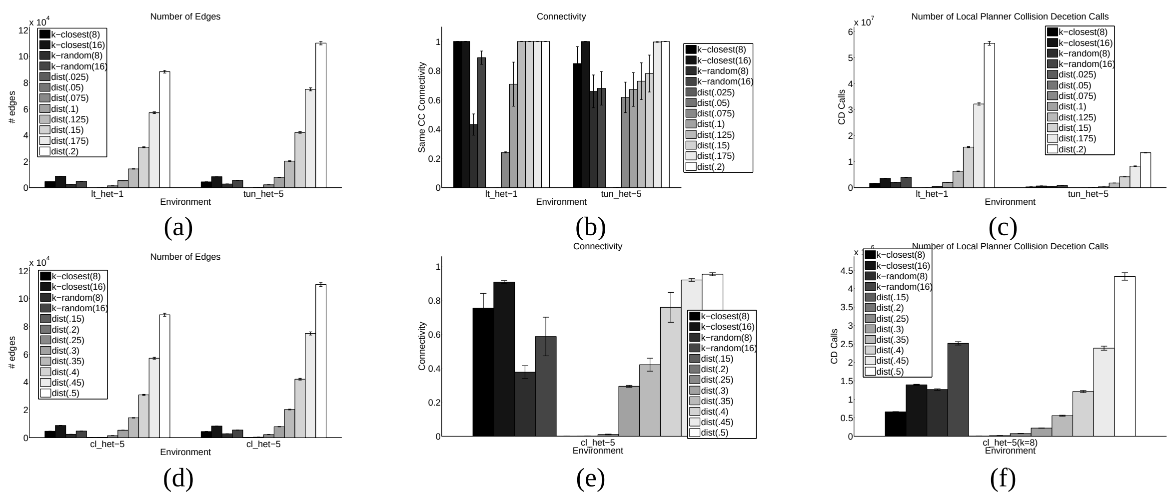


Fig. 7. Metrics across two heterogeneous environments comparing Distance method (with different a variety of different distances) with *k*-closest. These experiments were run using Uniform sampling and a Scaled Euclidean distance metric.

the connectivity of the roadmap. It is particularly interesting that the connectivity increases for the kr-kc methods in the tun-8 and the clt-8 environment because the kr-kc methods actually produced fewer edges than *k*-closest in these environments. This indicates that edge for edge, the longer edges produced by the kr-kc methods contribute more to roadmap connectivity than the shorter edges produced by *k*-closest.

Collision Detection Calls: In the free environment, the number of local planner collision CD-calls (Figure 5(c)) increased as randomness was introduced. The *k*-closest method required the fewest collision detection calls followed by kr-2kc and kr-4kc. *k*-random required far more collision detection calls than any of the other methods. This is because the more localized methods are testing fewer edges. Also, the edge length increases as randomness is introduced and the number of CD-calls required to test an edge is proportional to the length of the edge. In the Tunnel and Clutter environments, the number of CD-calls decreases as the environments become more difficult. This is a result of the decrease in local planner success in these environments. The local planner uses a binary search along an edge and it stops when it encounters a collision. This means that local planner attempts that detect a collision will require fewer CD-calls than others. In these environments *k*-closest, kr-2kcand kr-4kc have increasing numbers of CD-calls. However, the difference decreases as the environments become more difficult. As in the free environment, the more randomized methods are trying longer edges that require more CD-calls to test which is why the more randomized methods require more CD-calls than *k*-closest. At the same time, the local planner success of the randomized methods is less than *k*-closest, particularly in the more difficult environments; this is causing the number of CD-calls of the randomized methods to drop relative to *k*-closest as the environments become more difficult. In the Tunnel and Clutter environments, *k*-random requires fewer CD-calls than the other methods because is making fewer successful connections.

C. Effects of Roadmap Construction Metrics

We next analyze the effects of changing the roadmap construction metrics on the performance of the different candidate neighbor methods. We explore the effects of using a higher *k* and the effects of using a different distance metric. In general we show that the trends we observed in the previous sections are also present when the construction metrics are changed. This is important because it shows that our observations are not unique to the settings that we used in the previous sections.

1) **Effects of Higher *k* Value:** We evaluated how increasing the value of *k* from 8 to 16 would effect the different methods. Overall, the local planner success (Figure 4(a, b)) was lower with a *k* of 16 than with a *k* of 8 because we are attempting to make longer connections. Similar to this effect, the local planner success decreased as the proximity of neighbors decreased and the connectivity (Figure 6(b)) followed the same overall trends. In the tun-8, ctl-5 and clt-8 environments, the connectivity of kr-2kc is consistently greater than the connectivity of *k*-closest, and the connectivity of kr-4kc is consistently less than kr-2kc. As in the other sets, the connectivity increases as we introduce a small amount of randomness and then decreases as we introduce more randomness. The difference between these methods was less noticeable with the higher *k* value. Another difference was that the *k*-random method performed far better with a larger *k* value, particularly in the easy environments. In clt-1 and tun-1 environments, the *k*-random method produced roadmaps that were nearly as connected as the other methods.

2) **Effects of Distance Metric:** The local planner success for the swept experiments (Figure 4(c)) was generally higher than the local planner success of the Scaled Euclidean experiments. This is a natural result of using a more accurate distance metric. In both sets, the local planner success decreased as we introduce randomness, however this decrease was more drastic with the swept metric. This was particularly noticeable in the clt-8, tun-8 and clt-s environments.

The number of edges (Figure 5(b)) in the roadmaps was also larger with the swept distance metric than the Scaled Euclidean distance metric. As with the Scaled Euclidean distance metric, the number of edges increases as we go from *k*-closest to kr-2kc and kr-4kc in the free, tun-1, tun-5, clt-1 and clt-5 environments. As we showed in Section IV(B), this is because the less randomized methods are selecting many node pairs twice and are attempting fewer unique edges. The fact that this trend occurs in both the swept and Scaled Euclidean plots indicates that it is not something that is unique to the Scaled Euclidean distance metric. As with the Scaled Euclidean distance metric, the number of edges in the tun-8 and clt-8 environments decreases as we go from *k*-closest to kr-2kc and kr-4kc.

In both the Tunnel and the Clutter environments, the connectivity of the swept experiments (Figure 6(b)) increased slightly when we added a small amount of randomness to candidate neighbor selection (which can be seen by comparing the *k*-closest sets to the kr-2kc sets). However, this increase was far less than with the Scaled Euclidean distance metric. Overall, the swept experiments seem to be less effected by randomness than the Euclidean experiments.

D. Effects of Dimensionality

We next studied how the different methods performed in environments of varying dimensionality. Here we look at the articulated linkage and the manipulator robots which have between 8 and 64 degrees of freedom. Our results indicate that the impact of the methods is less in higher dimensional problems, and the the effect of the method on roadmap connectivity and construction cost decreases as we increase the dimensionality of the problem.

1) **Effects on Connectivity:** We first looked at the local planner success and the number of edges for the linkage experiments (Figure 8(a,b)). The local planner success and the number of edges decreased as we increased the dimensionality of the problem. This is because we are more likely to encounter a self collision while attempting to connect higher dimensional linkages. We also noticed that the local planner success and the number of edges was greater for the linkage than for the manipulator. As in the rigid body environments, the *k*-random method consistently produced fewer edges than the other methods and had a lower local planner success.

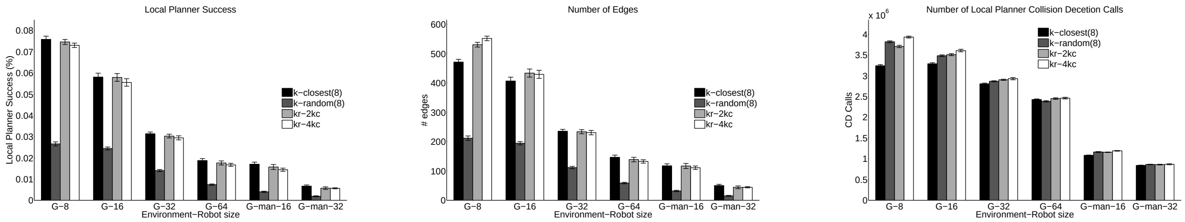
In the G-8 and G-16 environments, the number of edges increased slightly as we went from *k*-closest to kr-2kc and kr-4kc. This is similar to the trend that we observed in the sequential environments. The relative performance of the kr-kc methods decreased with respect to *k*-closest as we increased the dimensionality, and in the G-32, G-64 and G-man-32 environments the kr-kc methods produced fewer edges than *k*-closest.

As in the rigid body environments, the local planner success decreased as we introduced randomness. However, the difference between the local planner success of *k*-closest and the kr-kc methods was smaller than with the rigid bodies. Overall, the difference in local planner success and in number of edges between *k*-closest and the kr-kc methods was less for the linkages than for the rigid bodies (Figures 4(a), 5(a), 8(a,b)). Moreover, the difference between these methods decreased as we increased the dimensionality of the problem to the point where there is practically no difference between *k*-closest and the kr-kc methods in the G-32 and G-64 environments. This indicates that the 2*k* and the 4*k* closest nodes are nearly as connectible as the *k* closest nodes. This also indicates that it is more difficult to find the best candidate neighbors in higher dimensions and the Scaled Euclidean distance metric is not able to locate the best candidates for connection.

2) **Construction Cost:** The number of collision detection calls (Figure 8(c)) decreased as the dimensionality increased. This is because the number of successful local planner attempts decreases with dimensionality. As stated earlier, the cost of a failed local planner call is less than the cost of a successful local planner call because you do not need to compute the entire edge. As in the rigid body environments, the number of CD-calls increased as we went from *k*-closest to kr-2kc and kr-4kc. This is because the kr-kc methods are attempting longer connections. We also noticed that the difference in the cost of the different methods was smaller for the linkages than for the rigid bodies (Figures 5(c), 8(c)) and that this difference decreased as dimensionality increased.

VII. Conclusion

Based on the results of this study, we concluded that the *k*-closest method was well suited for roadmap construction. The *k*-closest method produced candidate neighbors that have



(a) Local Planner Success

(b) Number of edges

(c) CD-Calls

Fig. 8. Various metrics for experiments run in the different linkage environments.

a high probably of being connectible. More generally, we confirmed that more localized methods produce candidate neighbors that are more likely to be connectible. We also confirmed that *k*-closest was an effective optimization in that it reduced the cost of roadmap connection. The *k*-closest method generally required fewer collision detection calls than the other methods while still producing similar quality roadmaps. We also observed that *k*-closest is well suited for heterogeneous environments, and that it is able to adapt well to differences in sample density.

In our studies, the distance method was not well suited to roadmap construction, because it cannot adapt to differences in sample density. In non-uniform environments, it cannot produce roadmaps that are as connected as *k*-closest without producing many more edges than *k*-closest and making many more collision detection calls.

Our experiments demonstrate that more randomized methods produce candidate neighbors that are less likely to be connected than *k*-closest. We also found that methods with a small amount of randomness attempt more unique edges than *k*-closest and were able to produce roadmaps that had more edges and were better connected. However, this came at the cost of more collision detection calls. We also noticed that the difference in the performance of the methods was less for articulated linkages than rigid bodies, and that the difference in the performance of the methods decreases as the dimensionality of the problem grew.

REFERENCES

[1] F. Aghili and K Parsa. Configuration control and recalibration of a new reconfiburable robot. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 4077–4083, 2007.

[2] N. M. Amato. Equipping CAD/CAM systems with geometric intelligence. *ACM Computing Surveys*, 28(4es), December 1996. Article 17, http://www.acm.org/pubs/citations/journals/surveys/1996-2 8-4es/a17-amato/.

[3] 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. *IEEE Trans. Robot. Automat.*, 16(4):442–447, August 2000.

[4] 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.

[5] K. E. Bekris and L. E. Kavraki. Greedy but safe replanning under kinodynamic constraints. In *IEEE International Conference on Robotics and Automation*, 2007.

[6] Robert Bohlin. Path planning in practice; lazy evaluation on a multi-resolution grid. In *Proc. IEEE Int. Conf. Intel. Rob. Syst. (IROS)*, pages 49–54, 2001.

[7] 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.

[8] J. Bruce and M. Veloso. Real-time multi-robot motion planning with safe dynamics. In *Multi-Robot Systems: From Swarms to Intelligent Automata*, volume 3, 2006.

[9] D. Brutlag, M.S. Apaydin, C. Guestrin, D. Hsu, C. Varma, A. Singh, and J.-C. Latombe. Using robotics to fold proteins and dock ligands. In *Bioinformatics*, page 18:S74.S83, 2002.

[10] Howie Choset, Kevin M. Lynch, Seth Hutchinson, George A. Kantor, Wolfram Burgard, Lydia E. Kavraki, and Sebastian Thrun. *Principles of Robot Motion: Theory, Algorithms, and Implementations*. MIT Press, Cambridge, MA, June 2005.

[11] J. Cortes, T. Simeon, V. Ruiz De Angulo, D. Guieyss, M. Remaud-Simeon, and V. Tran. A path planning approach for computing large amplitude motions of flexiable molecules. In *Bioinformatics*, pages 21(1):116–125, 2005.

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

[13] Roland Geraerts and Mark Overmars. A comparative study of probabilistic roadmap planners. In *Workshop on the Algorithmic Foundations of Robotics*, pages 43–57, 2002.

[14] Roland Geraerts and Mark Overmars. Sampling and node adding in probabilistic roadmap planners. *In Journal of Robotics and Autonomous Systems*, 54:165–173, 2006.

[15] Roland Geraerts and Mark Overmars. Reachability-based analysis for probabilistic roadmap planners. *In Journal of Robotics and Autonomous Systems*, 55:824–836, 2007.

[16] 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.

[17] John Xiao J. Vannoy. Real-time motion planning of multiple mobile manipulators with a common task objective in shared work environments. In *IEEE International Conference on Robotics and Automation*, pages 20.–26, 2007.

[18] 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.

[19] J. Kuffner, K. Nishiwaki, S. Kagami, M. Inaba, , and H. Inoue. Motion planning for humanoid robots. In *Proc. 20th Int.l Symp. Robotics Research*, 2003.

[20] F. Lamiraux and D. Bonnafous. Reactive trajectory deformation for nonholonomic systems:application to mobile robots. In *IEEE International Conference on Robotics and Automation*, volume 2, pages 3099.–3104, 2002.

[21] J.-C. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, Boston, MA, 1991.

[22] S. M. LaValle. *Planning Algorithms*. Cambridge University Press, Cambridge, U.K., 2006. Available at http://planning.cs.uiuc.edu/.

[23] Dawei Li, Hijun Yang, Li Han, and Shuanghong Hou. Predicting the folding pathway of engrailed homedomain with a probablistic roadmap enhanced rection-path algorithm. *Biophysical Journal*, 94:1622–1629, 2008.

[24] Troy McMahon, Sam Jacobs, Bryan Boyd, Lydia Tapia, and Nancy Amato. Evaluation of the k-closest neighbor selection strategy for prm construction. Technical Report, Texas A&M, College Station Tx., 2011.

[25] C. Nissoux, T. Simeon, and J.-P. Laumond. Visibility based probabilistic roadmaps. In *Proc. IEEE Int. Conf. Intel. Rob. Syst. (IROS)*, pages 1316–1321, 1999.

[26] A. L. Olsen and H. G. Petersen. Motion planning for gantry mounted manipulators: A ship-welding application example. In *IEEE International Conference on Robotics and Automation*, pages 4782–4786, 2007.

[27] E. Plaku and L.E. Kavraki. Quantitative analysis of nearest-neighbors search in high-dimensional sampling-based motion planning. In *Proc. Int. Workshop on Algorithmic Foundations of Robotics (WAFR)*, July 2006.

[28] 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, October 1979.

[29] M. S. Branicky S. M. Lavalle and S. R. Lindemann. On the relationship between classical grid search and probabilistic roadmaps. *Int. J. Robot. Res.*, 23(7–8):673–692, 2004.

[30] G. Sanchez and J.-C. Latombe. On delaying collision checking in prm planning application to multi-robot coordination. In *Int. Journal of Robotics Research*, 2002.

[31] X. Tang, S. Thomas, and N. M. Amato. Planning with reachable distances: Fast enforcement of closure constraints. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 2694–2699, Roma, Italy, 2007.

[32] Konstantinous I. Tsianos, Ioan A. Sucan, and Lydia E. Kavraki. Sampling-based robot motion planning: Towards realistic applications. In *Computer Science Review*, volume 1, pages 2–11, 2007.

[33] Y. Wu. An obstacle-based probabilistic roadmap method for path planning. Master's thesis, Department of Computer Science, Texas A&M University, 1996.

[34] Y. Yang and O. Brock. Elastic roadmaps: Globally task-consistent motion for autonomous mobile manipulation. In *Robotics: Science and Systems*, 2006.