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

Troy McMahon, Sam Jacobs, Bryan Boyd, Lydia Tapia, Nancy M. Amato
Parasol Laboratory, Dept. of Computer Science and Engineering, Texas A&M University, College Station, Texas, 77843-3112, USA
{tmcmahon,sjacobs,bcb2893,ltapia,amato}@cse.tamu.edu

***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 potential 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. We also study how adding small amounts of randomness to neighbor selection affects the structure of the resulting roadmap. 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 free environment and in a set of tunnel and clutter environments of varying difficulty. We also evaluate these methods in a free/tunnel homogeneous environment and a free/clutter homogeneous environment. In these environments we conducted experiments using a variety of node generation methods and distance metrics. We also evaluate the methods in a set of articulated linkage environments to determine how the methods perform in higher dimensions. 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 a distance method. We also evaluate the graph structure of roadmaps produced by the methods.**

I. Introduction

The general *motion planning* problem involves finding a valid path for a movable object (e.g., robot, vehicle) from a start to a goal configuration in a given environment. The notion of validity depends on the problem – traditionally, it refers to a collision-free path, i.e., avoiding collisions with self and with obstacles in the environment. Motion planning is an important component of many applications, including computer-aided design [**?**], robotics [**?**], virtual reality simulations [**?**], and bioinformatics [**?**] [**?**]. The motion planning problem is regarded intractable, as the complexity of an exact method grows exponentially with the complexity of the robot [**?**]. Research on sampling-based approaches has produced methods that can solve important motion planning problems that were once considered unsolvable [**?**].

One widely used sampling-based method is the Probabilistic Roadmap Method (PRM) [**?**]. 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 makes it infeasible for any but the simplest of problems. Hence, the selection of the candidate neighbors is crucial to both roadmap quality and performance.

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 [**?**], [**?**], [**?**], [**?**], 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, node generation methods, 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.

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 node generation methods and distance metrics. We evaluate the different methods using a set of different evaluation metrics. These metrics evaluate the ability of each method to generate connectible pairs and assess the quality and connectivity of the roadmaps they produce. They also evaluate the structure of the roadmaps produced by the different methods.

Our findings confirm that k-closest is well suited to PRMs, outperforming the other strategies particularly in difficult environments. We show that distance based methods perform

comparably to k-closest in homogeneous environments if the distance d is selected appropriately, though determining such a d *a priori* is not always straightforward. While in heterogeneous environments, there is no such d — if d is tuned for the difficult region, then too many connections will be made in the freer regions, while if d is tuned for the relatively free regions, then not enough connections will be made in the difficult regions. In comparison, the k-closest method is better able to adapt to differences in sample density and is better suited for heterogeneous environments. 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. We also illustrate that in many cases including some randomness in the selection process has a noticeable impact on roadmap structure. We show that these methods produce roadmaps that have longer edges, a smaller diameter and are more scale-free.

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 [**?**] 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 [**?**]. They have been used for path planning with mobile robots [**?**] [**?**] [**?**], humanoid robots [**?**] and reconfigurable robots [**?**]. They have also been applied to biological problems including analysis of biological structures [**?**] and protein folding [**?**]. They have been used in industrial automation, particularly for path planning with robotic manipulators [**?**] [**?**] [**?**] [**?**].

Alg. 1 provides an overview of the PRM method. Roadmap creation in PRMs is typically performed in two phases. In the *Node Generation* phase, a set of n robot configurations in C-space is generated using a specific sampling strategy (or set of sampling strategies), and then they are added to the roadmap graph as vertices. The *Node Connection* phase attempts to add edges to this graph, where edges represent a valid motion between the configurations corresponding to the two roadmap vertices.

The node connection step is the most computationally expensive step of PRM construction. During this step we find a set of candidate neighbors for each of the nodes in the roadmap then attempts to connect each node to its candidate neighbors using a local planner. The cost of finding candidate neighbors, c_n, depends on what method is being used. If you are using k-closest then the candidate neighbors of a node can be found in $O(nlog(k))$ and the total neighbor selection time is $O(n^2log(k))$. During this step we also make a call to the local planner for every candidate neighbor pair that is found. The number of pairs depends on what methods is being used, if you are using the k-closest method then the total number of connections will be $O(kn)$ and the total connection time will be $O(knc_c)$

Algorithm 1 PRMs: Probabilistic Roadmap Methods

Generate Nodes {find a set of n collision-free configurations V - $O(nc_s)$}
Connect Nodes {connect nodes to form roadmap}
  for each configuration in V: (candidates for connection)
    select a set of Candidate neighbors, V_c - $O(c_n)$
    for each Candidate neighbor in V_c
      Attempt to Connect Candidate Neighbors Using Local Planner - $O(c_c)$
Process Query
  Connect start/goal to roadmap
  Find path in roadmap between connection nodes

In this algorithm n is the number of nodes, V is a set of n free nodes, V_c is the set of candidate neighbors for a given node, c_s is the cost of generating a sample, c_n is the cost finding the neighbors of a node and c_c is the cost of attempting to connect 2 nodes. Depending on what local planner you use c_c may depend on the distance between the nodes being connected.

Several PRM variants have been proposed in the literature that use different methods of node generation to handle different classes of problems. The original PRM (Basic PRM) [**?**] uses uniform random sampling to produce a distribution that is symmetry invariant. This method performs well when the environment is relatively free. Deterministic alternatives such as grid-based samplers have also been proposed [**?**]. Obstacle-Based PRM (OBPRM) [**?**] is a PRM that samples near the boundary of C-space obstacles. Gaussian-Sampling PRM (GaussPRM) [**?**] samples configurations in difficult regions by using techniques associated with blurring in image processing. Medial-Axis PRM (MAPRM) [**?**] biases sampling along the medial axis of the valid C-space. Resulting paths generally have a wide clearance from obstacles. In [**?**] Hsu et. al. present an adaptive method which identifies different regions of the environment and selects sampling methods that are appropriate for each region. In [**?**], Thomas et. al. explore applying different combinations of existing sampler methods. They show that some combinations of methods outperform individual methods.

The node connection phase itself contains two separate operations. First, pairs of candidates for connection are chosen using some criteria. In most applications of PRMs, the selection criteria is simply the k configurations that are closest, in distance, to the source configuration, i.e., the k-closest neighbor selection strategy. After these candidates are found, a *local planner* is used to attempt to find a valid path connecting the two configurations; the local planner is generally a deterministic planner, e.g., a straight-line collision detection check in C-space. For most motion planning problems, PRM construction time is dominated by the connection validation step [**?**].

There have been many previous attempts to understand the behavior of probabilistic roadmaps. In [**?**] and [**?**], Hsu et. al. define the concept of (ε, α, β)-Expansive environments and prove a lower bound on the probability of the PRM method solving an environment. In [**?**], Hsu et. al. uses the concept of (ε, α, β)-Expansiveness to explain why PRMs are well suited for motion planning problems. In [**?**] and [**?**], Kavraki et. al. study how the probability that the PRM method won't find an existing path is 'effected' by the length and clearance of the path and the number of nodes in the roadmap. In [**?**] and [**?**], Ladd and Kavraki formulate the PRM algorithm as a transitive closure problem and prove a bound on the expected probability that two points will be connected as a function of the length and clearance of the path between them.

In [**?**] and [**?**], Geraerts and Overmars preform 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 [**?**], Geraerts and Overmars show experimentally that the main difficulty in PRM construction is constricting a roadmap whose connectivity represents the connectivity of C-space. They also show that using a more complicated local planner can greatly improve the performance of the PRM method.

A number of variations and optimizations of the PRM method have been presented. One such method is the PRM method [**?**], [**?**] in which you only add edges that will help in connecting the query nodes. Lazy PRMs have been shown to reduce the time required to solve single query problems. A similar approach is taken with customizable PRMs where they preform an approximate validity check during roadmap construction then preform an exact validity check during the query phase [**?**]. In [**?**] Bohlin presents a variation of the lazy PRM method that uses nodes that are arranged in a hierarchical grid in C-space. This method connects nodes to their neighbors in the grid then adds nodes at a smaller resolution if a connection is not found.

B. Candidate Neighbor Selection Approaches

There have been variants proposed for identifying neighbors during the connection phase of PRM construction (step 2a in Alg. 1) [**?**]. 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 [**?**], OBPRM [**?**], MAPRM [**?**], and GaussPRM [**?**]. 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 [**?**].

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, possibly resulting in a very expensive connection phase. On the other hand, a distance that is too small may exclude many connections, resulting in poor connection. The original PRM implementation[**?**] included a variation of the distance method with an upper bound on the number of neighbors. The distance method is commonly used is for grid-based PRM methods [**?**]. It is also used in biophysical simulations where certain cutoffs are standard, e.g., a RMSD of atom positions between molecular conformations [**?**]. Geraerts and Overmars [**?**], [**?**] use a radius or 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 pretty well suited for generating connected roadmaps. Our study differs from this work in that 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 effect the performance of the methods. We also differ in that we study and evaluate the structure of the roadmaps produced by each of the methods.

In [**?**], Karaman and Frazzoli study the asymptotic behavior of sampling-based motion planning methods as the number of nodes increases. They show that most widely used methods do not converge to an optimal value. They show that the standard PRM, which uses a distance neighbor selection policy but only tests neighbors that are not part of the same connection component, is not asymptotically optimal. The also show that the 'simplified' PRM method, which uses a distance based method, is asymptotically optimal but computationally expensive. They also show that the *k*-closest PRM is not asymptotically optimal and that it is not always probabilistically complete. As part of their work they propose a new algorithm called PRM*. This algorithm uses a distance method with a distance that varies based on the number of samples, volume of C-free, and dimensionality of the problem. They also present a method called k-PRM* which uses the *k*-closest method with a k value that depends on the number of samples. They show theoretically and experimentally that the PRM* and k-PRM* methods are asymptotically optimal and that they require minimal computation time and memory.

In [**?**], 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 [**?**].

Geraerts and Overmars [**?**] also explore a *Component*-based selection strategy. This strategy attempts *k* connections to each connected component [**?**]. 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 [**?**], Bohlin presents a grid-based neighbor selection policy. The nodes in this method are arranged in a grid in C-space, and connections are attempted between adjacent nodes on the grid. If a path is not found than 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[**?**]. Researchers have proposed improvements to the brute-force nearest-neighbor computation that use data structures (e.g. KD-trees [**?**], [**?**], Metric/Spill Trees [**?**], [**?**], dimensionality reduction [**?**] [**?**]) to efficiently provide solutions to the neighbor selection problem. These methods use an approximation parameter that tolerates a certain amount of error in neighbor selection. In [**?**] Plaku et. al. perform an extensive analysis of 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 (which is used as a base-line for comparing the methods), 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 value.

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. Also, note that when $k = n$, where n is the total number of sampled nodes, then both k-closest and k-random actually implement the *all-pairs* strategy.

- **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.
- **k-dist**(v_s, V_c, d, k)**.** Select all $v \in V_c$ that are within distance d of v_s. If there are more than k vertices within d of v_s than select the k-closest vertices.
  k-dist(v_s, V_c, d, k)=k-closest$(v_s$,dist$(v_s, V_c, d), k)$
- **kr-kc:** 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 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 show how introducing some randomness into neighbor selection will affect roadmap quality.

As we increase the upper bound on the rank of the neighbors the nodes we select will overall become more distant. By increasing the bound on the neighbors we can select we can decrease the proximity of the samples we select in a controlled manner. It is expected that roadmap quality (i.e., connectivity) will decrease as proximity decreases because fewer successful connections are made. By varying proximity in a controlled manner we study how drastically roadmap quality decreases. We are particularly interested in what effect environment type and difficulty and the number of dofs has on this decrease. We also use the kr-kc methods to study how the structure of the roadmap is affected by decreasing the proximity of candidate neighbors.

IV. Evaluation Metrics

We assess the neighbor selection methods using a set of evaluation metrics. We chose metrics that indicate the ability of a method to find connectable neighbors and produce well-connected roadmaps. We also selected metrics that measure the cost of each method and the structural properties of the roadmaps produced.

A. Ideal Roadmap

To determine the relative quality of roadmap as the different neighbor selection parameters are changed, it is important to have an ideal case with which to normalize the results. Thus, an **ideal roadmap** is generated for each connection method. The ideal map will be connected using an all-pairs method: every node in the roadmap will attempt a connection to every other node. This method, while computationally expensive, will produce the best-possible connected roadmap for a set of nodes. This roadmap is used as a baseline against which to compare the roadmaps generated by other methods.

B. Connectivity Metrics

Our first set of metrics evaluate how good the methods are at finding connectable 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 connectable neighbors. The local planner success is different from the number of edges because the total number of edges attempted is not always the same for all of the methods. **NOTE: We should use a word other than "different" here. What is happening when the total number of edges attempted is different between different methods?**

- **Size of the Largest Connected Component**: The number of nodes contained in the largest connected component (CC) of the roadmap. This method indicates how successful the methods are at generating connected roadmaps.
- **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 roadmap is normalized over the connectivity of an all-pairs roadmap using the same set of nodes. A connectivity of 1 indicates that a roadmap captures the connectivity of the ideal roadmap while a connectivity of less than 1 indicates that there are nodes that are connected in the ideal map but not in the roadmap. Normalizing the connectivity allows us to compare the relative ability of each method to produce connected roadmaps.

C. Roadmap Structure Metrics

Our next set of metrics evaluate the structure of roadmaps produced by each method, and will show the impact of each connection method on roadmap structure.

- **Average Edge Length** and **Average Maximum Edge Length**: measures the length of edges that exist in the roadmap. **Average Edge Length** is the average length of all edges in the roadmap. This metric will be used to compare the relative edge lengths of the different methods and to show which methods produce longer edges. **Average Maximum Edge Length** is the average length of the longest edge of each node. This metric will tell us how long the longest edges created by each method are which will indicate how successful the randomized methods are at making longer connections. When calculating these metrics we used the Euclidean distance metric.
- **Diameter of Largest Connected Component**: The diameter is the length of the longest shortest path in a graph. This is important because it tells us about the structure of the roadmap and indicates how long paths in the roadmap will be.
- **Scale-free metric:** Scale-free graphs are graphs where the degree of each node follows a power law distribution. They are characterized by having *hub* nodes with a degree that is significantly greater than the average degree. Roadmaps that exhibit scale-free qualities will produce paths with less edges, as the diameter of the graph will be shortened by the number of high-degree nodes. We will use the scale-free metric proposed in [**?**] to evaluate how scale-free the roadmaps are. In this metric, G is a graph with edges E, and the degree of vertex i is d_i:

$$s(G) = \sum_{(i,j) \in E} d_i d_j$$

D. Cost Metrics

Our last metric will evaluate the cost of the metrics. This 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 [**?**] 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 of varying degrees of freedom. This will allow us to study the performance of the different methods across a variety of different environment types. These environments also allow us to study how changing the difficulty of the problem will impact the different methods. A full list of the environments we used is included in Table I along with the notation we use for each of 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 these environments. 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: This environment was a free environment with no obstacles. The environment was 18.5x18.5x18.5 units in size (these dimensions were chosen so this environment would have the same free volume as the other environments).

Tunnel Environment: The Tunnel environment (Figure 1(b)) consists of a tunnel that is 1x1 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 as shown in (Figure 1(c)) consists of 216 cube obstacles that were arranged in a 6x6x6 grid. These obstacles were 3x3x3 units in size and were separated by 1 unit of free space. This environment was 23x23x23 units in size.

Free+Cluttered Heterogeneous Environment: The first heterogeneous environment was made up of free and cluttered regions. The environment was 25% free to the left of the cluttered regions and 25% to the right as shown in

Fig. 1. Environments Studied:(a) Robots Used (with a section of the Tunnel for size comparison), (b) Tunnel, (c) Cluttered Homogeneous, (d) Cluttered Heterogeneous, (e) Tunnel Heterogeneous, (f) Gridworld, (h) Linkage and Manipulator Robots.

(Figure 1(d)) The cluttered part of this environment included 97 obstacles. Like in the cluttered environment the obstacles are 3x3x3 cubes and are separated by 1 unit of free space.

Free+Tunnel Heterogeneous Environment: The second heterogeneous environment (Figure 1(e)) consisted of a tunnel connecting 2 regions of free space. The tunnel was 1x1 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.

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 robots are shown in Figure 1(a,g).

Small Cube Robot: 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.

Medium Cube Robot: 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.

Large Cube Robot: 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 which was .8x.1x.1 units in size.

Articulated Linkages: In our experiments we also used a set of 4 articulated linkages 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 a total of 8 degrees of freedom.
- **16 Link Articulated Linkage:** This linkage consisted of 16 links of length .4375. This robot had a total of 16 degrees of freedom.
- **32 Link Articulated Linkage:** This linkage consisted of 32 links of length .2188. This robot had a total of 32 degrees of freedom.
- **64 Link Articulated Linkage:** This linkage consisted of 64 links of length .1094. This robot had a total of 64 degrees of freedom.

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 amount the links in the arm. The hand part of the manipulator consisted of 2 graspers of length 3. Like with the articulated linkage we always 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.

In the rigid body experiments the environment difficulty is controlled by varying the size of robot used. The use of small robots (a fraction of passage width) results in a higher local planner success rate, while using large robots (a tight fit in passages) results in a low local planner success rate.

In the articulated linkage experiments the problem dimensionality is controlled by varying the number of links in the linkage. This allows us to study method performance in a variety of dimensionalities, and will tell us how changes to problem dimensionality impacts each of the methods. **NOTE: I'm not quite sure what this means. It would be better to use something more specific than "complicated robots". Do manipulator robots allow us to see environments where only rotational DOF are used?** The manipulator robots will help us to determine how well the different method perform in environments with more complicated robots that include links of differing length.

C. Roadmap Construction Methods and Parameters

When running our experiments we varied the node generation strategies and distance metric methods. Uniform, Obstacle-based and Medial-Axis sampling strategies were used as node generation methods. For our distance metrics we used a Scaled Euclidean as well as a swept distance metric. For a local planner we use a binary straight-line local planner.

(a) (b)

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.

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

- **Uniform Random Sampling**: Also called Basic PRM [**?**], the uniform sampling strategy generates configurations uniformly at random and retains the free samples as roadmap nodes.
- **Obstacle Based Sampling**: Obstacle-based sampling, or OBPRM [**?**], uses a bisection search strategy along random rays in C-space to converge on a sample near the C-obstacle surface.
- **Medial Axis Sampling** Medial-Axis Sampling or MAPRM[**?**] biases sampling along the medial axis of the valid C-space.
- **Binary Straight-Line Local Planner** A straight line local planner [**?**] connects nodes along a straight line path in C-space. A binary straight-line local planner test the validity of this path by testing the validity of points along the path in a binary fashion.
- **Scaled Euclidean Distance Metric**: The Scaled Euclidean distance metric is one of the most commonly used distance metrics in motion planning [**?**]. In the Tunnel and Free+Tunnel environments we used a scaling factor of .9 an in the Free, Clutter and Free+Clutter environments we used a scaling factor of .5.
- **Local Planner Swept Distance Metric**: The local planner swept distance between two configurations is the total volume the robot sweeps in work space when moving between the configurations. This method is generally considered a very accurate measure of how difficult configurations will be to connect. The main disadvantage of swept distance is that it takes a long time to compute making it impractical for many motion planning applications. [**?**]

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 to sparse and we won't be able to connect them.

We first looked at the probability that a node set would be able to solve the environments. In the Tunnel environment the beta-lookout regions are the joints of the tunnel (the regions connecting 2 tunnel segments). To solve the Tunnel environment at least 1 sample in each joint is needed. If v(J) is the volume of a single joint, m is the number of joints and v(F) is the total area of free space then the probability of having a node in all joints is given by the equation:

$$\text{P}= \prod_{0}^{i=m-1} (1-(1-v(J)/v(F))^{(n-i)})$$

We computed this probability for a point robot as well as for a set of spheres with the same radius as the cubes used. For our probabilistic analysis we use these spheres as an approximation for the cube robots. For the sphere robots we computed this value by expanding all obstacles in the environment by the radius of the sphere. The probabilities of solving the Tunnel environment with each of these robots is included in Figure 3.

Based on these results we selected an n value of 1000. We chose this value because it gives a high probability of producing node sets that are capable of solving the Tunnel environment with the .1 and .5 robot. This n value is too small to ensure that it will solve the Tunnel environment with the .8 robot however our initial set of experiments shows that all-pairs connectivity with 1000 nodes will produce roadmaps where the largest connected component contains around 50 nodes (Figure 4). While this is less than the other environments it is still large enough for use to study. If we did use an n value that was large enough to solve the Tunnel environment, then the samples would be too dense in the other environments. We also considered using different n values for each of the environments however this would make it difficult to compare the results across different environments. For the purpose of this study we are only interested in evaluating how the different methods do in each

of the different environments and this can be done even if the set is not large enough to solve all queries in the environment.

We also confirmed experimentally that 1000 nodes was appropriate for the problems we studied. Figure 4 shows that with the small and medium robots, using 1000 nodes with all-pairs sampling will produce roadmaps where the largest connected component contains nearly all 1000 nodes. With the large robot all-pairs connection produced roadmaps where the largest CC contained about 800 of the 1000 nodes. From these initial experiments, we can conclude that 1000 nodes is sufficient to produce well connected roadmaps in allow the Clutter environments. Based on these initial experiments and on the probabilistic analysis we performed in the Tunnel environment we decided that this was an acceptable n value for our experiments.

Fig. 4. 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.

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 these k values because they were were sufficiently large to produce large connected components in all of our environments. For the distance method, we selected a distance d that was based on the edge length statistics from the roadmaps produced using the k-closest policy.

From what we saw in our results a k value of 8 was sufficient for this study. We therefore focus most of our attention on the sets with a k value of 8 when presenting our results. 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 also how adding small amounts of randomness to candidate neighbor selection affects 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 connectable 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. We also show that the connection method has an impact on the length of the edges in a roadmap and on the overall structure of the roadmap.

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 each method adapts to differences in sample density.

In each homogeneous environment, we chose to compare k-closest with a distance-based method that produced a similar number of CD-calls to k-closest for $k = 8$. This is sensible, because roadmaps generated required equivalent work to produce. For the Tunnel environment, a value of $d = 0.125$ was used; for the Clutter environments, a value of $d = 0.4$ was used. The resulting CD-calls for the distance method and k-closest can be found in Figure 15. In the roadmaps generated by these two methods, both the local planner success rate (Figure 7(a)) and number of edges connected (Figure 8) were similar. The connectivity of the roadmaps produced by the distance method (Figure 9(a)) was similar to k-closest in the Tunnel environments (tun-1, tun-5, tun-8), but slightly lower in the clutter environments (clt-1, clt-5, clt-8).

In Figure 5, we study the sensitivity of the distance-based methods to the value of d in the heterogeneous environments Free+Tunnel(a-d), and Free+Cluttered(e-h). 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 produces roadmaps with the same connectivity as k-closest (or higher in the case of Clutter environment). However, these roadmaps have many more edges than k-closest and require many more CD-calls to construct. This is a problem because additional edges increase 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 cl-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 summary, 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.

In Figure 6 we study the k-dist method. As with the distance method, k-dist produces unconnected roadmaps when the d value used was too small, and roadmaps with the same connectivity as k-closest for larger d values ($d >$ 1.25 in the Tunnel environments and $d >$ .45 in the Clutter environments). Because we limit the number of connection attempts to a maximum of k for each node, this method does not over-connect free regions in the environment, and the roadmaps it produces have a similar number of edges to k-closest and requires a comparable number of CD-calls. For some distances, this method produces less edges than k-closest and makes less CD-calls but still achieves the same connectivity. In the tun-het-1 environment the k-dist method with $d = 0.125$ consistently produces roadmaps that are as connected as k-closest but with less edges and using fewer CD-calls. In the cl-het-5 environment the k-dist method with $d = 0.45$ produces roadmaps that were as connected as k-closest yet with slightly less edges and requiring slightly fewer CD-calls.

We also noticed that the local planner success for this method was consistently higher than for k-closest. This is because the k-dist method does not attempt any relatively

(a)

Fig. 3. Probability that the all-pairs method will solve the Tunnel environment as a function of the number of nodes used.

distant connections (which have a minimal chance of succeeding). From these experiments, we can conclude that the k-dist method is capable adapting to differences in sample density, and is able to produce similar results to *k*-closest.

B. Effects of Proximity

Next, we will study the effect of neighbor proximity on roadmap quality; this tells us how important it is for methods to find neighbors that are close to a node. We study how introducing randomness impacts the ability to connect candidate neighbors, and how it affects the number of edges in a roadmap. We also study how this randomness affects the overall connectivity the roadmap. **NOTE: The following sentence is confusing:** We show that more local methods find neighbors that are more connectable but slightly random methods can produce roadmaps that are better 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 the *k* closest nodes, kr-2kc and kr-4kc give neighbors from the closest 2*k* and closest 4*k* nodes and *k*-random can return any 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 first studied how introducing randomness impacts the ability to connect candidate neighbors and how it affects the resulting number of edges in the roadmap. We also study how this randomness affects the overall connectivity of the roadmap. Our findings show that the *k*-closest method produces neighbors that are more connectable than other methods, and that the connectibility of neighbors decreases as proximity decreases. We also show that reducing the proximity of neighbors by a small amount can produce roadmaps that have more edges and higher connectivity.

Local planner success: We looked at the local planner success of these methods in the homogeneous environments (Figure 7(a)). In all cases (excluding the Free environment) *k*-closest has the highest local planner success followed by kr-2kc, kr-4kc and *k*-random. From this we conclude that the probability of finding connectable neighbors decreases with proximity. This is entirely expected because the distance metric is a heuristic for the difficulty of node connection: as long as an appropriate distance metric is chosen, we expect local planner success to decrease with neighbor proximity.

Of more interest is the relative effect of neighbor proximity in the different environments. In both the Tunnel and Clutter environments, the effect of reducing neighbor proximity increases significantly as the problems become more difficult. In the tun-1 environment the difference between *k*-closest, kr-2kc and kr-4kc is barely noticeable, while in the tun-8 environment the local planner success for kr-4kc is nearly half that of *k*-closest. In the clt-1 environment, the local planner success decreases by about a fifth as you go from *k*-closest to kr-4kc while in the clt-8 environment it decreases by over 50%.

In the Free environment and the easier Tunnel and Clutter environments (free-5, tun-1, clt-1, tun-5, and clt-5) the number of edges increases as we go from *k*-closest to kr-2kc and kr-4kc (Figure 8(a)). 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 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 each other 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 most difficult Tunnel and Clutter environments (tun-8 and clt-8) the number of edges decreases as we go from *k*-closest to kr-2kc and kr-4kc. Even though the number of candidate edges we are testing is larger for the 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: **NOTE: In this section (and in the rest of the results), it would be helpful to present the conclusions/trends first, followed by a few individual observations that prove these trends. As it is now, most of the text in the results section just describes the contents of the graphs shown elsewhere in the paper. These complex and nuanced results will become easier to understand when we highlight the trends first, and allow the reader to refer to the graphs more for results.** We next looked at the connectivity of the roadmaps produced by the different methods (Figure 9(a)). As a general trend, we observed that the kr-kc methods produced roadmaps that were more connected than *k*-closest while the *k*-random method produced roadmaps that were less connected than the other methods. This trend held across the various clutter and tunnel environments that we studied and for the different parameter setting that we used.

In the Free environment we noticed that all methods produced roadmaps with a connectivity of 1. Since the Free environment has no obstacles, it is expected that all methods will produce completely connected roadmaps. In the tun-1 environment, *k*-closest kr-2kc and kr-4kc all produced roadmaps with a connectivity of 1, meaning that they are producing roadmaps that are as connected as the all-pairs roadmap. *k*-random produces roadmaps that are considerably less connected than the other methods. In the clt-1 environment, *k*-closest, kr-2kc, and kr-4kc produced roadmaps that were nearly as connected as the all-pairs method. The *k*-closest method produced roadmaps with a connectivity of .85 while the kr-2kc and kr-4kc method produced roadmaps with connectivity of .9. We notice here that the kr-2kc, and kr-4kc produce roadmaps that are slightly more connected than *k*-closest. As in the tun-1 environment, *k*-random performs very poorly compared to the other methods. These

(a) (e)
(b) (f)
(c) (g)
(d) (h)

Fig. 5. Metrics across two heterogeneous environments comparing Distance method (with different a variety of different distances) to *k*-closest. These experiments were run using Uniform sampling and a Scaled Euclidean distance metric.

(a) (e)
(b) (f)
(c) (g)
(d) (h)

Fig. 6. Metrics across two heterogeneous environments comparing k-dist method (with different a variety of different distances) to *k*-closest. These experiments were run using Uniform sampling and a Scaled Euclidean distance metric. In this plot the k-dist method is referred to as dist-kmax.

Fig. 10. Size of the largest connected components in the roadmaps produced by each of the methods (using Uniform samples, Scaled Euclidean distance, and k=8).

results indicate that in easier environments, methods with a controlled amount of randomness produce roadmaps that are as connected as *k*-closest. These results also indicate that methods with too much randomness (*k*-random) cannot generate well-connected roadmaps, even in relatively easy environments.

In tun-5 environment, we noticed that *k*-closest and kr-4kc produced roadmaps with a connectivity of 1 while kr-2kc had a connectivity of nearly 1. In the clt-5 environment, we noticed that none of the methods had a connectivity of 1 (which means that none of these methods had the same connectivity as the all-pairs roadmap). We also noticed that the connectivity increased substantially as we went from *k*-closest to kr-2kc. The connectivity of the kr-4kc method was similar to the kr-4kc method. In the tun-8 environment, we noticed that the connectivity increased as we went from *k*-closest to kr-2kc then dropped as we went from kr-2kc to kr-4kc (although the connectivity of kr-4kc was still higher than *k*-closest). The connectivity in the clt-8 environment was significantly less than 1, however it did display a similar trend to the tun-8 environment. The connectivity of kr-2kc was greater than *k*-closest, while the connectivity of kr-4kc was less than that of kr-2kc but still greater than the connectivity of *k*-closest. In the clt-1, clt-5, tun-8 and clt-8 environments, we saw that the connectivity increased as we introduced a small amount of randomness to neighbor selection then decreased as we introduced additional randomness. 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.

2) **Effects on Roadmap Structure:** In order to study the effects of the methods on the structure of roadmaps, we looked at the length of the edges in the roadmaps the produced. We also looked at the diameter of the resulting roadmaps and how scale-free they are. This will show us what impact the methods have on the overall structure of roadmaps and help us to determine how different the roadmaps are. Our results show that the choice of connection method has a noticeable impact on roadmap structure. Our results show that reducing the proximity of neighbor selection results in roadmaps that have longer edges, a smaller diameter and are more scale-free.

Edge length: In this section we study the lengths of the edges generated by the different methods. We first looked at the average length of the edges produced by the different methods (Figure 11). Our first observation is that in all cases *k*-random has the largest average length kr-4kc, kr-2kc and *k*-closest. This is expected because the more randomized methods are allowed to select more distant candidate neighbors. The fact that the edge length increases also helps us to confirm that the randomized methods are successfully connecting to some of the more distant neighbors.

In the Free environment we noticed that the *k*-random method had a substantially larger average edge length (it was three times larger than the other methods). This is because it is able to connect very distant pairs of nodes while the other methods are only connecting local nodes. We also noticed that the variation in the average edge length was very large in the Free environment (which can bee seen it the size of the error bars).

In the Tunnel environment the increase in edge length as we went from *k*-closest to kr-2kc and kr-4kc was similar to the Free environment. One thing that we noticed is that the average edge lengths of the different methods were similar in the different tunnel environments. There is a slight decrease in average edge length as the environments become more difficult however the choice of method has a greater impact on the average edge length than the difficulty of the environment.

In the Clutter environment we noticed the same trends as Tunnel environment. The average edge length increased as randomness increased and decreased slightly as the environment became more difficult however the choice of connection method had a larger impact on average edge length than the the difficulty of the environment.

In looking at the average max edge length we notice some

(a) Local Planner Success, Uniform Euclidean, k=8

(b) Local Planner Success, Uniform Euclidean, k=16

(c) Local Planner Success, Uniform Swept, k=8

(d) Local Planner Success, OBPRM Euclidean, k=8

(e) Local Planner Success, MAPRM Euclidean, k=8

Fig. 7. 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

Fig. 8. Number of edges 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

(d) Connectivity, OBPRM Euclidean, k=8

(e) Connectivity, MAPRM Euclidean, k=8

Fig. 9. Roadmap connectivity across a variety of connection methods and environments.

Fig. 11. Average Edge Length, Uniform Euclidean, k=8.

Fig. 12. Average Max Edge Length, Uniform Euclidean, k=8.

of the same trends as the average edge length. The average max edge length increases as we go from *k*-closest to kr-2kc, kr-4kc and *k*-random. Like with the average edge length, the average max edge length depends more on the type of method used than the difficulty of the environment. We also noticed that the difference between *k*-random and the other methods was less in the tunnel environments than in the other environments. This is because nodes can only be connected if they are in the same tunnel segment so *k*-random cannot make any really distant connections in this environment.

Based on the edge length and the average edge length we can conclude that the different methods are behaving differently. The more randomized methods are attempting to make longer connections than the more localized methods. The more randomized methods are successfully making longer connections and they are generating maps with longer edges than the more localized methods. These results also indicate that the kr-kc are working the way we designed them to. They are making connections that are on average longer than *k*-closest but shorter than *k*-random and the length of the connections increases as we go from kr-2kc to kr-4kc. These results also indicate that the choice of connection methods is one of the dominant factors in determining the length of the edges in a roadmap.

Diameter: We noticed that the diameter decreased as we went from *k*-closest to kr-2kc and kr-4kc. This is because the randomized methods are producing longer edges which means that you can take direct paths to distant parts of the roadmap as opposed to taking a roundabout path that includes many shorter edges. This means that the overall path lengths will tend to be shorter in more randomized roadmaps and the diameter will be smaller. We also noticed that this decrease was more noticeable in the sets with a k value of 8 than the sets with a k value of 16. The rate at which the diameter decreases is similar for the different problem difficulties. We also noticed that the tunnel environments consistently had a larger diameter than the clutter environments. This is because paths in the Tunnel environment are constrained within the tunnel region and a path between nodes at the opposite ends of the tunnel will need to traverse the entire distance of the tunnel.

We noticed that in the Free environment the diameter of *k*-random was smaller than the diameter of the other methods. This is because *k*-random is producing roadmaps with much longer edges that the other methods. In the easier tunnel and clutter environments the diameter of *k*-random was larger than the other methods. This is because *k*-random is producing roadmaps with less edges and the resulting paths are more circuitous. In the harder environments, the diameter of the *k*-random sets is smaller than the other methods however this is only because the size of the largest cc (Figure 10) is much smaller for these roadmaps

Scale-free metric: In the Free environment, decreasing the proximity of the sampling method made the resulting roadmaps more scale-free. This is because methods that select closer nodes are only connecting nodes to their immediate neighbors. This means that the degree of the nodes in the roadmap will be more uniform and that there will be no 'hub' vertices that have a large number of neighbors (this is a property of scale-free graphs). In graphs generated using more randomized methods, there will be a greater variation in the degree of vertices which means that the resulting graphs will be more scale-free.

In the tun-1 and tun-5 environments, the roadmaps also became more scale-free as we went from *k*-closest to kr-2kc and kr-4kc. This increase became less dramatic as the environment became more difficult. This is because the randomized methods are producing less edges and this is offsetting the advantages of having less uniform degrees. In the tun-8 environment, the randomized methods produce so few edges that they are actually less scale-free than *k*-closest.

In the Clutter environments, we are less able to make longer connections than the Tunnel environments. This means that in all cases the randomized methods produced less edges than *k*-closest and in all cases the roadmaps produced by kr-2kc and kr-4kc were less scale-free that *k*-closest.

In both the tunnel and the clutter environments, the *k*-random method produces roadmaps that are less scale-free than the other methods. In the tunnel and clutter environment *k*-random roadmaps had considerably fewer edges than the

other methods. The benefits to the degree distribution that we saw in the Free environment are completely offset by the fact that there are fewer edges.

From looking at the scale-free metric we can conclude that the connection method used has an impact on the overall structure of roadmaps. We can also conclude that the randomized methods produce roadmaps that are more scale-free than roadmaps than *k*-closest.

3) **Effects on Roadmap Construction Cost:** In this section we study the computational cost associated with each of the methods. Our results indicate that PRM construction cost is greater for methods that attempt more connections, for methods that test longer edges and for methods that create more edges. Our results show that the cost associated with *k*-closest is less than the cost associated with the kr-kc methods.

Collision Detection Calls: In the Free environment, the number of local planner collision CD-calls (Figure 15) 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. Part of this is because the more localized methods are testing fewer unique edges. As we saw with the number of edges, more localized methods tend to put pairs on nodes in each-others candidate neighbor set. Since we only test an edge once the overall number of edges will be less for the less randomized methods. 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 the Tunnel and Clutter environments the number of CD-call 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 less CD-calls than ones that don.t. In these environments the number of CD-calls increases as we went from *k*-closest to kr-2kc and kr-4kc however the difference decreases as the environments become more difference. Like in the Free environment, the more randomized methods are trying longer edges that will 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 less CD-calls than the other methods because is making so few successful connections in comparison to the other methods. We also noticed that the number of CD-calls for *k*-random is less in the Tunnel environment than in the Clutter environment. This is because the Tunnel environment consists of thicker obstacles (the blocks between the tunnel segments) and binary search will require fewer samples to detect a collision.

Fig. 15. Number of collision detection calls made by each of the methods (With Uniform Samples, Scaled Euclidean Distance, and k=8).

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*, the effects of using a different distance metric and the effects of using different node generation methods. 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 next looked at how increasing the value of *k* from 8 to 16 would affect the different methods. Overall, the local planner success (Figure 7(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. The connectivity (Figure 9(b)) followed the same overall trend. In the tun-8, clt-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. Like in the other sets, the connectivity increases as we introduce a small amount of randomness 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 the tun-1 and clt-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 7(c)) is 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 decreases as we introduce randomness, however this decrease is more drastic with the swept distance metric. This is particularly noticeable in the clt-st environment and the clt-8 and tun-8 environments. The swept distance metric was also producing roadmaps that had more edges than the Scaled Euclidean distance metric.

The number of edges(Figure 8(b)) in the roadmaps was also larger with the swept distance metric than the Scaled Euclidean distance metric. Like 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-5, 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 less unique edges.

(a) Diameter of Largest CC, Uniform Euclidean , k=8

(b) Diameter of Largest CC, Uniform Euclidean, k=16

(c) Diameter of Largest CC, Uniform Swept, k=8

(d) Diameter of Largest CC, OBPRM Euclidean, k=8

(e) Diameter of Largest CC, MAPRM Euclidean, k=8

Fig. 13. Diameter across a variety of connection methods and environments.

(a) Scale-Free Metric, Uniform Euclidean, k=8

(b) Scale-Free Metric, Uniform Euclidean, k=16

(c) Scale-Free Metric, Uniform Swept k=8

(d) Scale-Free Metric, OBPRM Euclidean, k=8

(e) Scale-Free Metric, MAPRM Euclidean, k=8

Fig. 14. Scale-free metrics across a variety of connection methods and environments. clt-8 values are not visible due to the difference in scale

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. Like with the Scaled Euclidean distance metric, the number of edges decreases as we go from *k*-closest to kr-2kc and kr-4kc (Figure 8(a,b)).

In both the Tunnel and the Clutter environments, the connectivity of the swept experiments increased slightly when we introduced 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 in the Scaled Euclidean environments. Overall, the swept experiments seem to be less affected by randomness than the Scaled Euclidean experiments.

3) **Effects of Node Generation Method:** In order to study how the different node types affect the performance of the methods, we looked at a set of roadmaps consisting exclusively of OBPRM nodes, and a set of roadmaps consisting exclusively of of MAPRM nodes. In practice, both of these methods are usually used in combination with other node generation methods [**?**], however we elected to use roadmaps with only one node type. We did this in order to study how the connection methods worked in combination with the different node types and to determine which connection methods match up well with each node generation method. We also wanted to study how important proximity was for nodes that are close to obstacles (OBPRM nodes) as opposed to nodes that are far from the obstacles (MAPRM nodes). The OBPRM and MAPRM node generation methods are also commonly used with different local planner methods. In particular, the OBPRM method is often used with a 'Rotated-S' local planner instead of a strait line local planner. We chose to used the start line local planner in these experiments to make it easier to compare them to our other results.

When studying different node generation methods, we paid particularly close attention to the environments with the small and medium cube robots (tun-1, tun-5, clt-1, and clt-5), because the difference between the different node types will be most noticeable in these environments. The local planner success in these environments tends to be higher with MAPRM nodes than uniform sampling and lower with OBPRM nodes. This is because the MAPRM samples are further from obstacles and are easier to connect while the OBPRM samples are closer to the obstacles and are more difficult to connect. Like with uniform samples, the local planner success decreased as we introduced randomness to neighbor selection. The relative local planner success of *k*-closest and the kr-kc methods did not vary for the different node generation methods

In the environments with the large cube robots (tun-8 andd clt-8), the local planner success with uniform and OBPRM nodes is similar. In these environments the robot is nearly as wide as the tunnel so there will not be much difference between Uniform and OBPRM samples. The MAPRM samples gave use a significantly higher local planner success in these environments. In the tun-8 environment the local planner success was 2 times higher with MAPRM nodes that with uniform and in the clt-8 environment it was almost 5 times larger. This might be because the MAPRM sampler is producing nodes that are aligned with the sides of the tunnel. The local planner success of MAPRM in these environments was also affected more by introducing randomness that the other methods in these environments.

In the tun-1, tun-5, clt-1, and clt-5 environments OBPRM produced roadmaps that were less connected than uniform sampling. This most likely because OBPRM produced nodes that were harder to connect and the resulting roadmaps will have fewer edges and therefore be less connected. The connectivity of OBPRM roadmaps increased as we introduced randomness however this increase was not as large as with uniform sampling. These results seem to contradict previous work such as [**?**] which indicates that OBPRM produces more connected roadmaps than uniform sampling. The difference between this work and ours is that they perform their experiments in heterogeneous environments. The advantage of OBPRM nodes is that they produce samples in more difficult regions of an environment (such as narrow passages). In a heterogeneous environment this allows you to solve the difficult regions without oversampling the easy regions. In our environments the difficulty is the same everywhere so there is no benefit to using OBPRM sampling.

The connectivity of the MAPRM roadmaps was higher than the other methods especially in the tun-8 and clt-8 environments. This is most likely a result of the higher local planner success of MAPRM roadmaps which is leading to roadmaps that have more edges and are more connected. Like with the other methods, the connectivity of MAPRM increased as we introduced a controlled amount of randomness to candidate neighbor selection (kr-2kc). This increase is particularly noticeable in the clt-8 and tun-8 environments.

Proportionally the difference between the local planner success of MAPRM, Uniform Sampling and OBPRM is larger for more difficult environments. In the easier environ-

ments (tun-1, clt-1 and tun-5) the decrease in local planner success as we go from *k*-closest to kr-2kc and kr-4kc is more noticeable with OBPRM nodes and less noticeable with MAPRM nodes. Because there is a grater clearance between MAPRM nodes an obstacles it is easier to make longer connections and the effects of using a less localized method will not be as large. Likewise, there is less clearance between OBPRM nodes and obstacles which means that it will be more difficult to make longer connections and the impact of using a less localized method will be larger.

In the more difficult environments (tun-8 and clt-8) the decrease in local planner success is larger for MAPRM than any of the other methods although the local planner success of MAPRM with (kr-4kc) is still higher than the local planner success of the other methods.

The diameter of the OBPRM roadmaps was larger than the diameter of the uniform roadmaps in the Free environment and the tunnel and clutter environments with the small cube robot. This is because the nodes in the OBPRM roadmaps are all on the surface of obstacles which means that paths in these roadmaps are confined to the surface of obstacles. Compared to a uniform roadmap we would expect paths in an OBPRM roadmap to be longer and we would expect the diameter of an OBPRM roadmap to be larger. Introducing randomness to neighbor selection has the same effect on OBPRM roadmaps that it had on uniform roadmaps. The diameter of the MAPRM roadmaps was larger than the diameter of uniform roadmaps in the tunnel environments with the small and medium cube however in the Clutter environments (clt-1, clt-5, clt-8) the diameter of MAPRM roadmaps was smaller than uniform roadmaps. Introducing randomness to neighbor selection had a similar effect on the diameter uniform. OBPRM and MAPRM roadmaps. In all cases randomness reduced the diameter of roadmaps by a considerable amount.

The OBPRM roadmaps were overall less scale-free than uniform roadmaps. This is partially because the OBPRM roadmaps had a lower local planner success. This means that OBPRM roadmaps will have fewer edges and consequently tend to be less scale-free. The OBPRM roadmaps became more scale-free as we introduced randomness to candidate neighbor selection however this increase is not as large with OBPRM roadmaps as with uniform roadmaps. This is because it is more difficult to connect more distant OBPRM nodes. As we have seen the local planner success of OBPRM roadmaps decreases more with randomness than the other methods. This means that the kr-2kc and kr-4kc OBPRM roadmaps will have less edges than the corresponding uniform roadmaps and will consequently be less scale-free. Overall OBPRM nodes are not very well suited to produce scale-free roadmaps because they tend to have smaller visible volumes than uniform or MAPRM samples. This means that an OBPRM node will have less other nodes that it can connect to and it is less likely that there will be any vertices with a large number of neighbors (hub nodes).

The MAPRM roadmaps were more scale-free than uniform or OBPRM roadmaps. This is because the medial axis roadmaps had a higher local planner success rate which mean they have more edges and will consequentially be more scale-free. The MAPRM roadmaps also seem to be affected more by introducing randomness than uniform or OBPRM roadmaps. As we saw the local planner success of MAPRM roadmaps decreases less with randomness. This means that the kr-2kc and kr-4kc roadmaps will have more edges than uniform or OBPRM roadmaps and will tend to be more scale-free. Overall MAPRM nodes are well suited to creating scale-free graphs because they have larger visible volumes than uniform or OBPRM nodes. This means that a given MAPRM node will see more other nodes than a uniform or OBPRM node would. This makes it easier to form hub nodes that have a large number of neighbors.

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 effect of the method on roadmap connectivity and construction cost decreases as we increase the dimensionality of the problem. They also indicate that none of the methods capture the connectivity of the all-pairs roadmap. We also study the structure of the roadmaps produced in the linkages environments. Our findings indicate that in lower dimensional environments the kr-kc methods produce roadmaps that are more scale-free than *k*-closest, and that had a smaller diameter. In the higher dimensional environments the kr-kc methods produced roadmaps that were less scale free than *k*-closest and that had a larger diameter.

1) **Effects on Connectivity:** We first looked at the local planner success and the number of edges for the linkage experiments (Figure 16(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. Like in the rigid body environments, the *k*-random method always performed worse than the other methods. It 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 7(a), 8(a), 16(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

(a) Local Planner Success (b) Number of edges

(c) Connectivity (d) Diameter of Largest CC

(e) Scale Free Metric (f) CD-calls

Fig. 16. Various metrics for experiments run in the different linkage environments.

difference between k-closest and the kr-kc methods in the G-32 and G-64 environments. This indicates that the $2k$ and the $4k$ 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.

The connectivity of the linkage and manipulator robots (Figure 16(c)) was significantly less than 1, particularly in the G-64, G-man-16, and G-man-32 environments. This indicates that none of the methods captured the connectivity of the all-pairs roadmaps. Moreover, the connectivity of the methods decreases as the dimensionality of the problem increases. This indicates that in higher dimensional problems there are many connectable pairs that the methods do not find, and that these pairs contribute significantly to the connectivity.

The relative connectivity of the methods followed the same trend as with the rigid bodies, it increased as we went from k-closest to kr-2kc and kr-4kc. The difference in the connectivity of the methods was significant even though the difference in the number of edges (Figure 16(b)) was small. Because there are so few edges in these roadmaps, even the addition of a small number of edges can have a considerable effect on the connectivity. As with the number of edges, the connectivity of the kr-kc methods decreased relative to k-closest, although the connectivity of the kr-kc methods was never less than k-closest. Note that the connectivity of the kr-kc methods was greater than that of k-closest in the G-32, G-64 environments even though the number of edges was less. This is similar to the trend observed in the tun-8 and clt-8 environments (see section VI-B.1) and indicates that the longer edges produced by the kr-kc methods contribute more to roadmap connectivity than the shorter edges produced by k-closest.

Overall, the connectivity metrics indicate that the performance of the methods decreases as the dimensionality of the problem becomes larger. The local planner success is lower in higher dimensions, the number of edges produces is smaller, and the connectivity of the roadmpas produced is less. At the same time, the local planner success of k-closest about the same as kr-2kc and kr-4kc, which means that the k closest nodes are not any more connectable than the $2k$ or $4k$ closest nodes. This may indicate that the Scaled Euclidean distance metric is less able to identify connectable node pairs in higher dimensions. This may also indicate that the straight line local planner is not well suited higher dimensional problems. It might be necessary to develop a set of specialized distance metrics and local planners for higher dimensional problems.

2) **Effects on Graph Structure:** We next studied the structure of the linkages roadmaps. We compared the diameter (Figure 16(d)) of the roadmaps produced by the different methods and we looked at how scale-free the roadmaps were (Figure 16(e)). In the G-8 and G-16 environments, these metrics followed the same trend as the rigid body experiments. The diameter decreased as we went from k-closest to kr-2kc and kr-4kc, and the roadmaps became more scale free. As the dimensionality of the problem increase, the diameter of the kr-kc methods increased with respect to$k$-closest. Likewise, the kr-kc roadmaps became less scale-free with respect to k-closest. In the G-32 and G-64 environments, the kr-kc methods produced roadmaps that had a larger diameter than k-closest, and were less scale-free.

3) **Effects on Construction Cost:** The number of collision detection calls (Figure 16(f)) decreased as the dimensionality increased. This is because the number of successful local planner attempts decreases with dimensionality. As we 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. Like 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 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 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 study also indicates that a distance method with a limit on the number of neighbors

is capable of similar performance to k-closest it the correct distance values are selected.

Our experiments demonstrate that more randomized methods produce candidate neighbors that are less likely to be connected than k-closest. This was most noticeable in more difficult environments and when using nodes that are more difficult to connect (OBPRM nodes). 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. The overall performance of the methods also decreased as the dimentionality of the problem became larger.

The results of this study also indicate that the neighbor selection method has an impact on the structure of roadmaps. We show that decreasing the proximity of neighbor selection results in roadmaps that are more scale free than k-closest and that have longer edges and a shorter diameter.

REFERENCES