

Provided for non-commercial research and education use.  
Not for reproduction, distribution or commercial use.



This article was published in an Elsevier journal. The attached copy is furnished to the author for non-commercial research and education use, including for instruction at the author's institution, sharing with colleagues and providing to institution administration.

Other uses, including reproduction and distribution, or selling or licensing copies, or posting to personal, institutional or third party websites are prohibited.

In most cases authors are permitted to post their version of the article (e.g. in Word or Tex form) to their personal website or institutional repository. Authors requiring further information regarding Elsevier's archiving and manuscript policies are encouraged to visit:

<http://www.elsevier.com/copyright>



# Reachability-based analysis for Probabilistic Roadmap planners<sup>☆</sup>

Roland Geraerts\*, Mark H. Overmars

*Institute of Information and Computing Sciences, Utrecht University, 3584 CH Utrecht, The Netherlands*

Received 1 June 2006; received in revised form 3 May 2007; accepted 22 June 2007

Available online 3 July 2007

---

## Abstract

In the last fifteen years, sampling-based planners like the Probabilistic Roadmap Method (PRM) have proved to be successful in solving complex motion planning problems. While theoretically, the complexity of the motion planning problem is exponential in the number of degrees of freedom, sampling-based planners can successfully handle this curse of dimensionality in practice. We give a reachability-based analysis for these planners which leads to a better understanding of the success of the approach. This analysis compares the techniques based on coverage and connectivity of the free configuration space. The experiments show, contrary to general belief, that the main challenge is not getting the free space covered but getting the nodes connected, especially when the problems get more complicated, e.g. when a narrow passage is present. By using this knowledge, we can tackle the narrow passage problem by incorporating a refined neighbor selection strategy, a hybrid sampling strategy, and a more powerful local planner, leading to a considerable speed-up.

© 2007 Elsevier B.V. All rights reserved.

*Keywords:* Motion planning; PRM; Reachability; Sampling; Node adding; Potential field local planner

---

## 1. Introduction

A central problem in robotics is planning a collision-free path for a moving robot in a rigid environment. In 1979, Reif showed that path planning for a polyhedral robot among a finite set of polyhedral obstacles was PSPACE-hard [1]. Four years later, Schwartz and Sharir proposed a complete general-purpose algorithm based on an algebraic decomposition of the configuration space of any fixed dimension  $d$ . When the space of collision-free placements is a set defined by  $n$  polynomial constraints of maximal degree  $m$ , a path can be computed by an algorithm whose time complexity is doubly exponential in  $d$  and polynomial in both  $n$  (geometrical complexity) and  $m$  (algebraic complexity) [2]. In 1986, this algorithm was improved to a single exponential time algorithm [3]. Due to their complexity, these algorithms have never been used in practice. In 1988, Canny found a PSPACE algorithm for the

general mover's problem and showed that it was PSPACE-complete [4], showing that exact planners have little chance of solving complicated problems. In 1991, Barraquand created the Randomized Path Planner, which was an artificial potential field planner [5]. This planner did not require the construction of an explicit description of the configuration space. Although this planner was much faster than previous approaches, it could get stuck at a local minimum of the potential function. One year later, different researchers independently devised the Probabilistic Roadmap Method (PRM), which was successfully able to deal with many motion planning problems [6–8]. This success is due to its wide applicability and good performance: if a solution to a problem is easy, then it seems that the PRM can solve it very fast.

The PRM consists of two phases: a construction and a query phase. In the construction phase, a roadmap graph  $G = (V, E)$  is constructed, approximating the motions that can be made in the (static and known) environment. Globally speaking, the PRM samples the configuration space  $\mathcal{C}$  (that is, the space of all possible placements for the robot in the environment) for collision-free placements by using a collision checker. These are added as nodes to the graph. Pairs of promising nodes are chosen in the graph and a simple local planner is used to try to connect such placements with a path. If they can be connected,

---

<sup>☆</sup> This research has been supported by the GATE project, funded by the Netherlands Organization for Scientific Research (NWO) and the Netherlands ICT Research and Innovation Authority (ICT Regie). Part of this research has been funded by the Dutch BSIK/BRICKS Project.

\* Corresponding author. Tel.: +31 30 253 3977; fax: +31 30 251 3791.  
E-mail addresses: [roland@cs.uu.nl](mailto:roland@cs.uu.nl) (R. Geraerts), [markov@cs.uu.nl](mailto:markov@cs.uu.nl) (M.H. Overmars).

then an edge between the nodes is added to the graph. This process continues until the graph reflects the connectivity of the free configuration space  $\mathcal{C}_{\text{free}}$  (that is, the space of all possible free placements for the robot). In the query phase, the start and goal placements are connected to the graph. The path is usually obtained by a Dijkstra's shortest path query.

The basic PRM leaves many details to be filled in, in particular how to sample the  $\mathcal{C}$ -space, which local planner to use and how to select promising pairs of nodes. Over the past fifteen years, researchers have investigated these aspects and developed many improvements over the basic scheme. In [9], we compared and analyzed many of these aspects.

The (analysis of the) complexity of a motion planning problem is often expressed in terms of geometric complexity (of the obstacles and robot) and the number of degrees of freedom (DOFs) of the robot. This is reasonable for methods that are based on the geometry of obstacles such as visibility graphs, Voronoi diagrams and exact cell decompositions [10]. In practice, these methods fail when the geometric complexity is high, or when many ( $>4$ ) DOFs or many primitives are involved.

Complexity analysis is also employed for sampling-based planners. Analyses for these planners use the coverage of  $\mathcal{C}_{\text{free}}$  with (hyper)spheres which results in exponential complexity bounds, see e.g. [11]. Yet, in practice, the PRM can successfully handle this curse of dimensionality because it is *reachability*-based, i.e. a sample can often be connected to other samples that are far away because they can be reached by the local planner. For example, if each sample can reach a large part of  $\mathcal{C}_{\text{free}}$  by using a local planner, then  $\mathcal{C}_{\text{free}}$  will be covered and connected quickly. This does not follow from the standard analysis that only allows a sample to be connected to its adjacent neighbors. Another reason why the PRM is fast is because its primitive operations are simple. Checking samples for collisions does not require an explicit representation of the configuration space (whose combinatorial complexity can be very high). When a path or a sample is checked for collisions, only the obstacles in the vicinity are involved. As a result, 'redundant' primitives on the other side of the environment do not affect the performance. These properties lead to a favorable performance that is proportional to some measure of difficulty for the problem to be solved.

Another way to analyze sampling-based planners is to measure the time it takes to solve a relevant witness query [9]. This however does not provide information on the quality of the roadmap, i.e. there is no guarantee that every possible query can be solved with this roadmap. In this paper, we evaluate the planners based on solving every possible query by performing a reachability analysis which emphasizes the notions of *coverage* and *maximal connectivity* of  $\mathcal{C}_{\text{free}}$ . By inspecting the roadmap when  $\mathcal{C}_{\text{free}}$  is covered and when  $\mathcal{C}_{\text{free}}$  is maximally connected, we obtain a better understanding of these techniques. These concepts are introduced in Section 2. In Section 3, we describe the experimental setup. In the following three sections, we analyze neighbor selection, sampling and local planning techniques. The analysis makes it clear that the PRM derives its strength by the ability to make long and many

connections. We observe that the main difficulty is not getting  $\mathcal{C}_{\text{free}}$  covered, but getting the nodes connected, especially when the problem gets more complicated. We tackle the connectivity problem by proposing a local planner that is more powerful than the standard straight-line local planner. We conclude in Section 7 that a refined neighbor selection strategy, a hybrid sampling technique and a new potential field local planner lead to a better performance of PRMs.

## 2. Coverage and maximal connectivity

The PRM was designed to be a multiple shot planner which enables fast querying. This goal can be achieved by creating a graph  $G = (V, E)$  that covers  $\mathcal{C}_{\text{free}}$  and captures its connectivity. We define coverage and maximal connectivity as follows:

**Definition 1 (Coverage).**  $G$  covers  $\mathcal{C}_{\text{free}}$  when each configuration  $c \in \mathcal{C}_{\text{free}}$  can be connected using the local planner to at least one node  $v \in V$ .

**Definition 2 (Maximal Connectivity).**  $G$  is maximally connected when for all nodes  $v, v' \in V$ , if there exists a path in  $\mathcal{C}_{\text{free}}$  between  $v$  and  $v'$ , there exists a path in  $G$  between  $v$  and  $v'$ .

Coverage ensures that every query (which consists of a start and goal configuration) can be connected directly to the graph, as is required to solve the query. If there exists a path (in  $\mathcal{C}_{\text{free}}$ ) between the start and goal configuration, then maximal connectivity ensures that a path between them can be found in the graph. Note that the path in the graph and the path in  $\mathcal{C}_{\text{free}}$  do not have to be in the same homotopic class.

If both the criteria are satisfied, then a path can always be found for every query. Additional criteria are of course imaginable, for example creating a graph that optimizes path quality (see e.g. [12]), but we will not consider them here. Several authors have studied the use of PRM for solving single motion planning queries. For single shot techniques such as [13, 14], coverage does not play an important role, and, our analysis is less relevant.

We use coverage and connectivity as an analysis tool to gain insight into sampling-based methods. Our goal is to determine for various techniques how long it takes before  $\mathcal{C}_{\text{free}}$  has been covered and connected. Because this would be rather complex for a continuous (high-dimensional) configuration space  $\mathcal{C}$ , we discretize  $\mathcal{C}$  (for problems that have 2D and 3D  $\mathcal{C}$ -spaces): for each cell (whose dimensions are determined by the step size used by the local planner) in  $\mathcal{C}$ , we check whether the placement of the robot for that cell is free and store this information in an array. When a node  $v$  is added to  $V$ , its discretized *reachability* region is calculated by checking for each free cell  $c$  in the array whether there exists a local path between  $v$  and  $c$ . All free cells that can be connected by the local planner are labeled with a unique region number. If each free cell has been covered by at least one region, the coverage criterion has been met. The connectivity criterion is verified as follows: for each added node  $v \in V$  we calculate the set of nodes  $W \subseteq V$  to which it

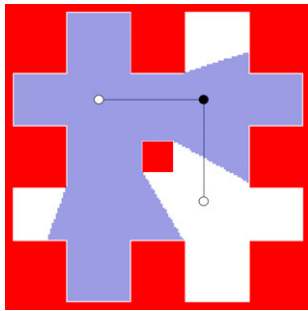


Fig. 1. The coverage and maximal connectivity criteria have been met. The reachability regions of the white nodes cover the complete free space and are connected via the black node.

can be connected through the grid of free cells. Then we add all combinations of  $v$  with  $W$  to a connectivity list. If there exists a path in graph  $G$  for each connection in the connectivity list then  $\mathcal{C}_{\text{free}}$  is maximally connected. Please realize that these calculations are only done to compare planning techniques. They are not part of the actual motion planning algorithm, and, hence, are not taken into account when reporting the running times.

As an example, Fig. 1 shows an environment whose free space is covered by two (white) nodes and is connected via one extra (black) node. Hence, this three-node graph suffices to solve this problem. The reachability region for the upper left node has been drawn. Each configuration in this region can be connected with a straight-line local planner to the node. The shape of a reachability region can be complicated. Fig. 2(a) shows a region for a 2D environment with many small obstacles and Fig. 2(b) shows a 3D region for the manipulator arm depicted in Fig. 3(f) with three rotational DOFs.

### 3. Experimental setup

All experiments were conducted in a single motion planning system called SAMPLE (System for Advanced Motion Planning Experiments), implemented in Visual C++ under Windows XP. SAMPLE automates the conduction of experiments, i.e. statistics are automatically generated and processed, decreasing the chance for errors. The experiments were run on a 3 GHz Pentium 4 processor with 1 GB internal memory. We used Solid as basic collision checking package [15].

The statistics were all averaged over 100 independent runs.

We used the six environments depicted in Fig. 3. The environments have the following properties:

**Clutter 1** The 2D cluttered environment consists of 16 polygons through which a small robot must navigate. It should be easy to create a maximally connected roadmap for this environment, because each sample will cover a large portion of the free space. The robot is a square with two translational degrees of freedom (DOFs).

**Clutter 2** We added 24 polygons to the first environment to reduce the average size of the regions. Again, the robot is a translating square.

Table 1  
Information on the configuration spaces of the environments

	DOF range	Step size	Number of cells
Clutter 1	$40 \times 40$	0.5	$80 \times 80$
Clutter 2	$40 \times 40$	0.5	$80 \times 80$
Narrow passage	$40 \times 40$	0.5	$80 \times 80$
Hole	$40 \times 40 \times 40$	2.0	$20 \times 20 \times 20$
Corridor	$40 \times 8 \times 40$	2.0	$20 \times 4 \times 20$
Manipulator	$6.1 \times 0.6 \times 0.7$	0.05	$122 \times 12 \times 14$

**Narrow passage** This environment has been designed to be more difficult. It contains a narrow passage through which a square has to move. The passage is surrounded by two large open spaces. The robot is a translating square.

**Hole** The Hole environment has two large open spaces separated by a wall with a narrow hole in it. The robot is a small cube that can only translate. Each sample will cover a large portion of  $\mathcal{C}_{\text{free}}$ .

**Corridor** A small translating cube has to move through a 3D winding corridor consisting of four hairpins. The walls of the corridor will limit the size of the reachability regions.

**Manipulator** This 3D environment features a robot arm with three rotational DOFs which operates in a constrained workspace. The  $\mathcal{C}$ -space has a long passage.

We discretized the  $\mathcal{C}$ -space of each environment.<sup>1</sup> The level of discretization can be found in Table 1. Consider for example the Clutter 1 environment: The ranges of the translational DOFs of the robot are  $[0 : 40] \times [0 : 40]$ . The step size used by the local planner is 0.5. Hence, the  $\mathcal{C}$ -space is discretized with  $80 \times 80$  cells.

We calculate the distance between two configurations  $q$  and  $r$  by summing the weighted partial distances for each DOF  $0 \leq i < n$  that describes the configurations, i.e.

$$d(q, r) = \sqrt{\sum_{i=0}^{n-1} (w_i |q_i - r_i|)^2}.$$

We set the weights  $w_i$  for the translational DOFs to 1 and set the weights for the rotational DOFs to 6.

For each experiment, we ran the PRM until both coverage and maximal connectivity had been achieved and recorded the following statistical data.

**Definition 3 (Number of Regions).** Each node  $v \in V$  in the graph implies a new region. The number of regions then equals to the cardinality of  $V$ .

**Definition 4 (Average Size of the Regions).** Let  $k$  be the number of regions  $r_i$  discovered so far. Furthermore, let  $|r_i|$  be the number of cells in region  $i$  and  $|\mathcal{C}_{\text{free}}|$  the total number

<sup>1</sup> We did not use problems with more than three DOFs because the analysis of coverage and connectivity would require too much memory. However, as the  $\mathcal{C}$ -spaces of the environments are representative of many problems, we think that the results also apply to problems involving more than three DOFs.

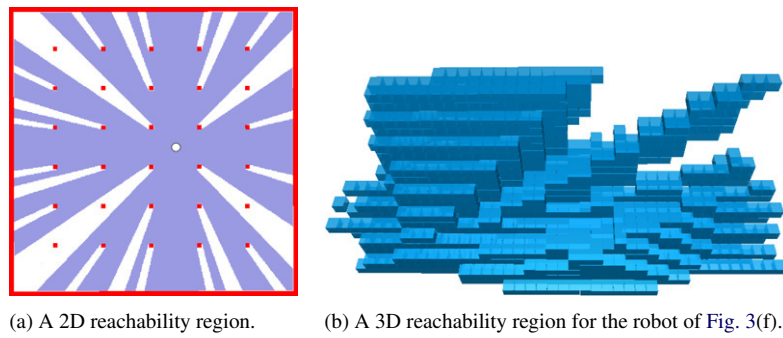


Fig. 2. Complicated 2D and 3D reachability regions.

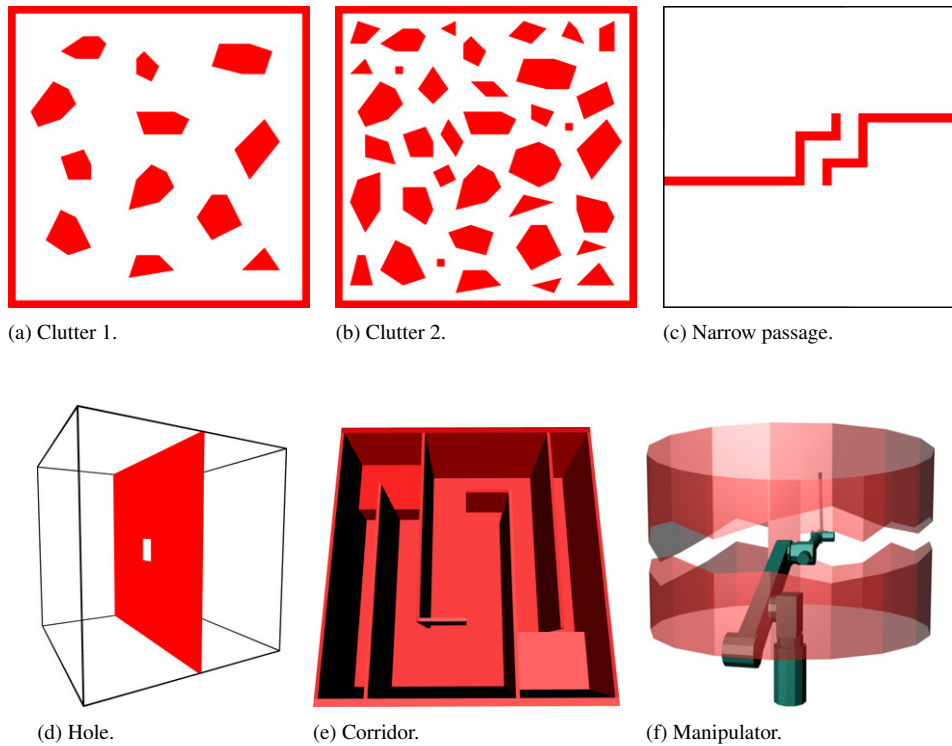


Fig. 3. The six test environments.

of free cells. Then the *average size of the regions* equals:  $\frac{1}{k} \sum_{i=1}^k |r_i| / |\mathcal{C}_{\text{free}}|$ .

We recorded the number of regions and the average size of the regions at two moments: at the moment that  $\mathcal{C}_{\text{free}}$  was covered and at the moment that  $\mathcal{C}_{\text{free}}$  was maximally connected (after  $\mathcal{C}_{\text{free}}$  was covered). We also recorded the running time after both the criteria had been satisfied. We give an indication of the dispersion of the running times by a *box plot*. Each plot consists of a box, one large and two small horizontal lines and a vertical line. The box represents the middle 50% of the data, the large horizontal line represents the average, the small lines represent the average  $\pm$  the standard deviation and the vertical line represents the minimum and maximum value.

#### 4. Neighbor selection strategy

The neighbor selection strategy specifies for a particular sample how a set of neighbor samples is chosen to which

it is connected. The goal of the strategy is to make the graph connected as fast as possible. A strategy usually selects neighbors based on a combination of the following criteria: the *maximum connection distance*, the *maximum number of connections* tried, and the *node adding strategy*. We chose the *nearest-k* node adding strategy as this method performed reasonably well in different environments [9]. This strategy tries to connect the new configuration to the nearest  $k$  nodes in the graph that lie close enough. We used the optimal sampling strategy for each environment (see below). That is, we used Bridge sampling for the Narrow passage environment. For the other ones, we used Halton sampling. We study the effects of different choices for the first two criteria on the six test environments.

##### 4.1. Maximum connection distance

The maximum connection distance should not be too small or too large. A very small connection distance will

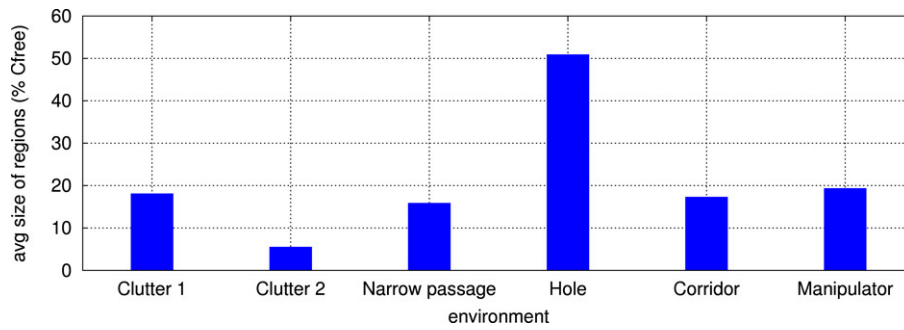


Fig. 4. The average size of the regions corresponding to the optimal neighbor selection parameters.

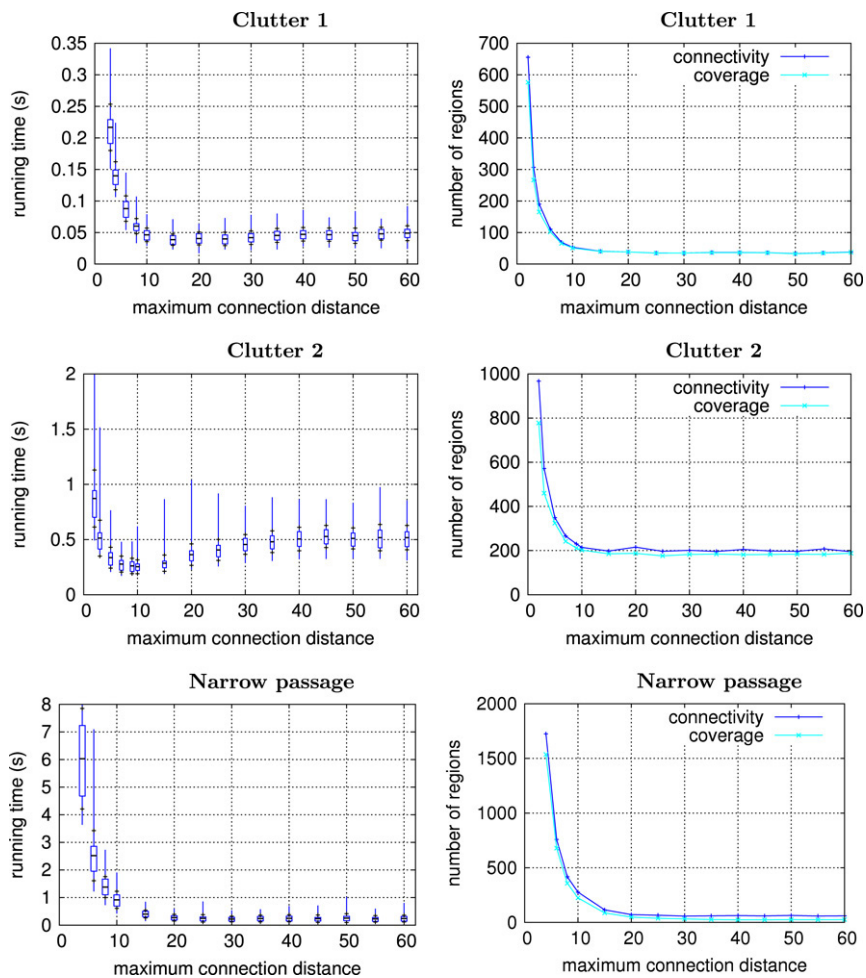


Fig. 5. Influence of the *maximum connection distance* on the running time and the number of regions required to get the free space covered and maximally connected in the 2D environments.

always require an exponential number of samples (in the number of DOFs) which increases the running times a lot. In addition, it is not useful to try too long connections since the chance of success for such connections is small while the collision checks required for testing the local path are expensive.

In each of the following experiments, we varied the maximum connection distance from a small value (close to zero) to a large value. The maximum number of connections was set to 75 (see below). Fig. 5 shows the results for the 2D environments and Fig. 6 shows the results for the 3D environments.

When we make the connection distance very small, the PRM starts looking like grid-based techniques in which samples are only connected to their direct neighbors. The figures show that this considerably increases the (average and variance of the) running time. Furthermore, there is a large difference in the moment of coverage and the moment of maximal connectivity. This shows that a small connection distance complicates the making of connections.

It is clear, as the maximum distance gets larger, that the average size of the reachability regions will increase (up to some value), and hence, the number of samples needed to

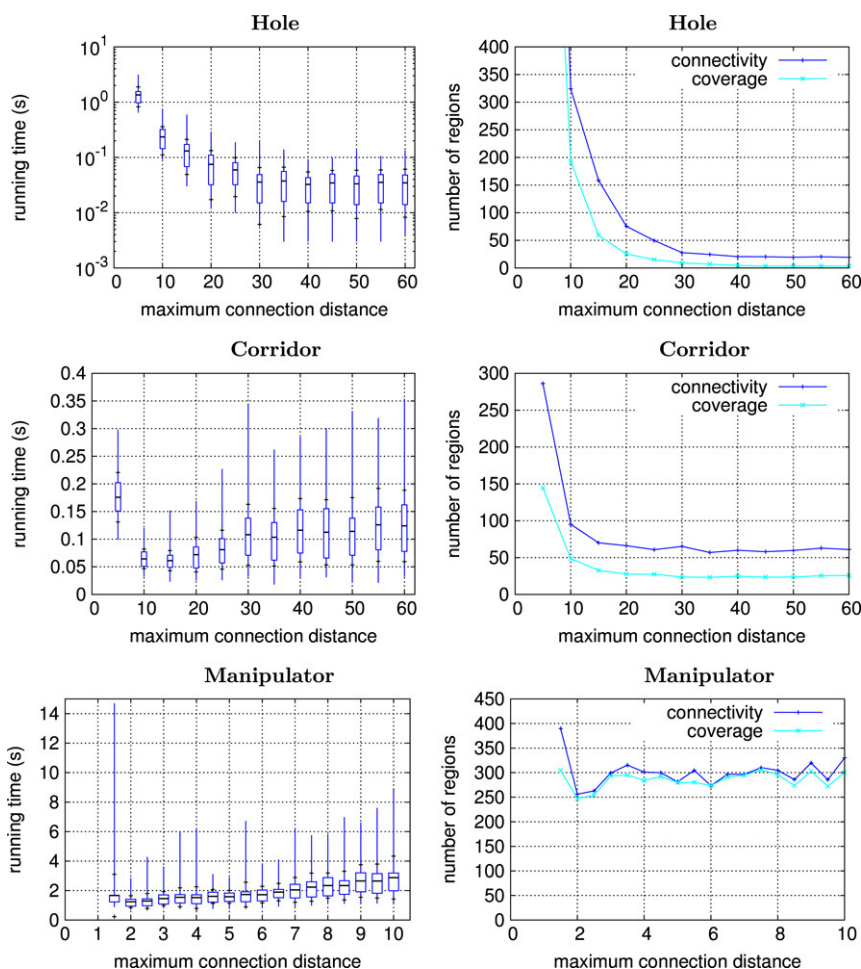


Fig. 6. Influence of the *maximum connection distance* on the running time and the number of regions required to get the free space covered and maximally connected in the 3D environments.

solve the problem will decrease. In other words, the number of samples required to solve the problem decreases when the time required per sample increases. The results show that there is some optimal trade-off which is dependent on the environment (and metric). If the size of a region (that corresponds to a particular sample) is small, then the sample can be connected to other samples to which the distance is small, and *vice versa*. In general, if the average reachability of the samples is low, then a small connection distance is preferable and *vice versa*. See Fig. 4 which shows the average size of the regions for the six environments corresponding to the optimal neighbor selection parameters. We can make two observations. First, the larger the average size of the regions, the smaller the growth of the running time when the maximum connection distance increases. Second, the larger the average size of the regions, the larger the value for the optimal maximum connection distance. This information can be used to estimate the (local) optimal maximum connection distance. For example, when a sample can be connected to other samples at a large distance, then the average size of its reachability region is large. Hence, using a large maximum connection distance is a good choice for choosing neighbors. In addition, few samples should be created near the sample.

The results confirm that the PRM derives its strength from making long connections. While a very small connection distance has a dramatic negative impact on the running time, a large value has only little impact, especially when the average size of the reachability regions is large.

#### 4.2. Maximum number of connections

The second analyzed criterion for selecting neighbors is the *maximal number of connections* attempted to connect a node. In this section we will show the relation between the maximal number of connections and the coverage and connectivity.

The number of attempted connections does not influence the coverage, but has a clear influence on the connectivity. If the number of connections is too small, it might be hard to get the free space maximally connected because the chance is small that those few samples are selected to which a connection is possible. If connections are attempted with (too) many nodes,  $C_{\text{free}}$  will become maximally connected using less regions. Nevertheless, this might negatively influence the running time since testing those connections is expensive.

In our experiments, we varied the maximum number of connections (and used the optimal values for the maximum connection distance). Fig. 7 shows the results for the 2D

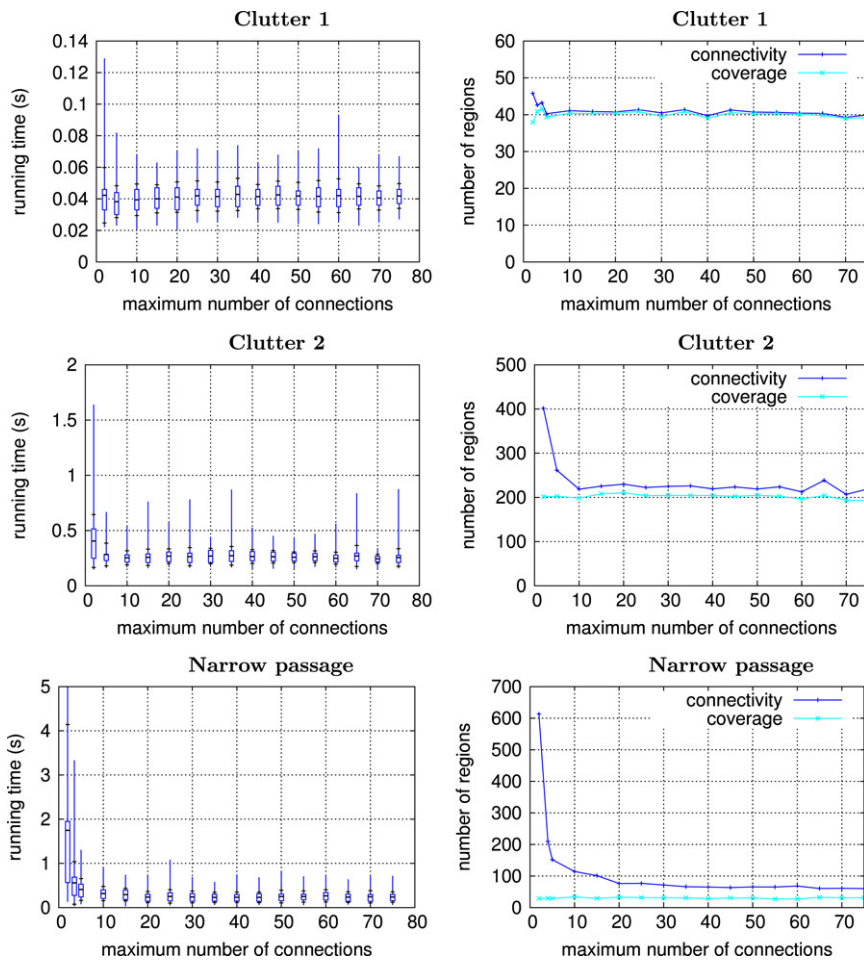


Fig. 7. Influence of the *maximum number of connections* on the running time and the number of regions in the 2D environments.

environments and Fig. 8 shows the results for the 3D environments. Indeed, when only a few connections are tried, more regions are needed to get the roadmap maximally connected. Making more and more connections does not seem to be useful because the number of regions needed to get  $\mathcal{C}_{\text{free}}$  covered and maximally connected remains constant at a certain maximum number of connections. However, as a large maximum number of connections (e.g. 75) does not affect the running time much, we recommend to use such a high value. This can be explained as follows. First, we observe that setting the maximum number of connections larger than the expected number of regions (that satisfies the coverage and connectivity criteria) has no effect on the running time. Consider for example the Clutter 1 environment. As it uses a random sampling strategy, the expected number of samples to which a particular sample can be connected is the connection area, divided by the area of the free space, multiplied by the average number of regions that corresponds to satisfying both criteria: the expected number of samples equals  $\pi * 15^2 / 40^2 * 40 = 17.7$ . This means, if the samples are uniformly random spread and a maximum connection distance of 15 is used, then the expected number of samples to which a connection is tried (per sample) is 17.7. Setting the maximum connection distance higher than this value will have little effect on the running time. Setting the parameter slightly smaller than this value will also not have much effect

on the running time as the chances are high that the discarded samples do already belong to the same connected component.

## 5. Sampling

The PRM has been expressed as a sampling-based motion planning method. In this section we will study the behavior of different sampling techniques. They can be classified into three categories: uniform, non-uniform and hybrid techniques.

The first category comprises the uniform techniques such as random, grid, cell-based and Halton sampling [16]. It is well known that these techniques can have difficulties dealing with the narrow passage problem. The second category tackles the narrow passage problem by biasing the sampling distribution. That is, more samples are added in ‘difficult’ regions of the environment. A region is difficult if the size of the region (which corresponds to a particular sample) is small compared to the total free space. The number of samples that are generated within these regions can be increased by filtering out samples that probably do not contribute to the coverage and maximal connectivity of the roadmap. Examples include Gaussian [17] and obstacle-based [18] sampling. The third category combines the strengths of the previous two categories. The Bridge test [19] for example concentrates samples in difficult areas but it also generates some samples in open areas. Several



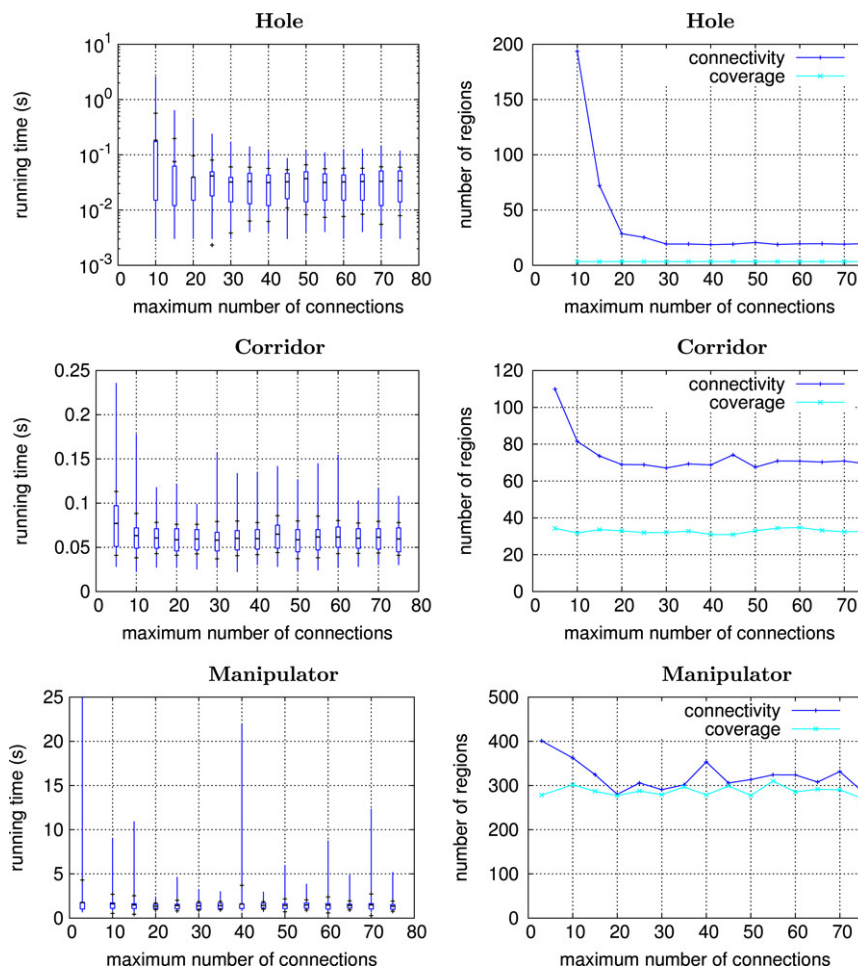


Fig. 8. Influence of the *maximum number of connections* on the running time and the number of regions in the 3D environments.

combinations of existing sampling strategies are suited to serve as a hybrid technique, see the paper of Hsu et al. [20] for an elaboration.

### Experiments

For each category we choose a representative method. For the uniform technique we choose *Halton*, for non-uniform *Gaussian* and for hybrid we choose *Bridge test*.

**Halton** In [16], it has been suggested to use the so-called Halton point sets as samples. Halton point sets have been used in discrepancy theory to obtain a coverage of a region that is better than using a grid (see e.g. [21]). To avoid this deterministic method being lucky or unlucky, we choose a random initial seed instead of setting the seed to 0 [22].

**Gaussian** Gaussian sampling is intended to add more samples near obstacles. The idea is to take two random samples, where the distance  $\sigma$  between the samples is chosen according to a Gaussian distribution. Only if one of the samples is in  $C_{\text{free}}$  and the other one collides we add the free sample. This leads to a favorable sample distribution [17]. We conducted preliminary experiments to find the optimal values for

$\sigma$ . We set  $\sigma$  to {4.0, 4.0, 1.2, 4.0, 4.0, 1.2} for the six environments, respectively.

**Bridge test** The Bridge test is a hybrid technique that aims at better coverage of the free space [19]. The idea is to take two random samples, where the distance  $\sigma$  between the samples is chosen according to a Gaussian distribution. Only if *both* the samples collide and the point in the middle of them lies in  $C_{\text{free}}$  the free sample is added. Also to get points in open space, every sixth sample is chosen randomly. We set  $\sigma$  to {4.8, 4.8, 2.4, 5.6, 5.6, 4.0} for the six environments, respectively.

We will compare their behavior by considering the experiments we performed on the six environments. These environments are representative of many different motion planning problems so we expect the observations to apply rather generally.

Fig. 9 shows the results for the 2D environments and Fig. 10 shows the results for the 3D environments.

Halton sampling resulted in relatively low running times for all environments, except for the Narrow passage environment. This is consistent with the results from [9]. Halton's uniform distribution created too many samples in the two wide open

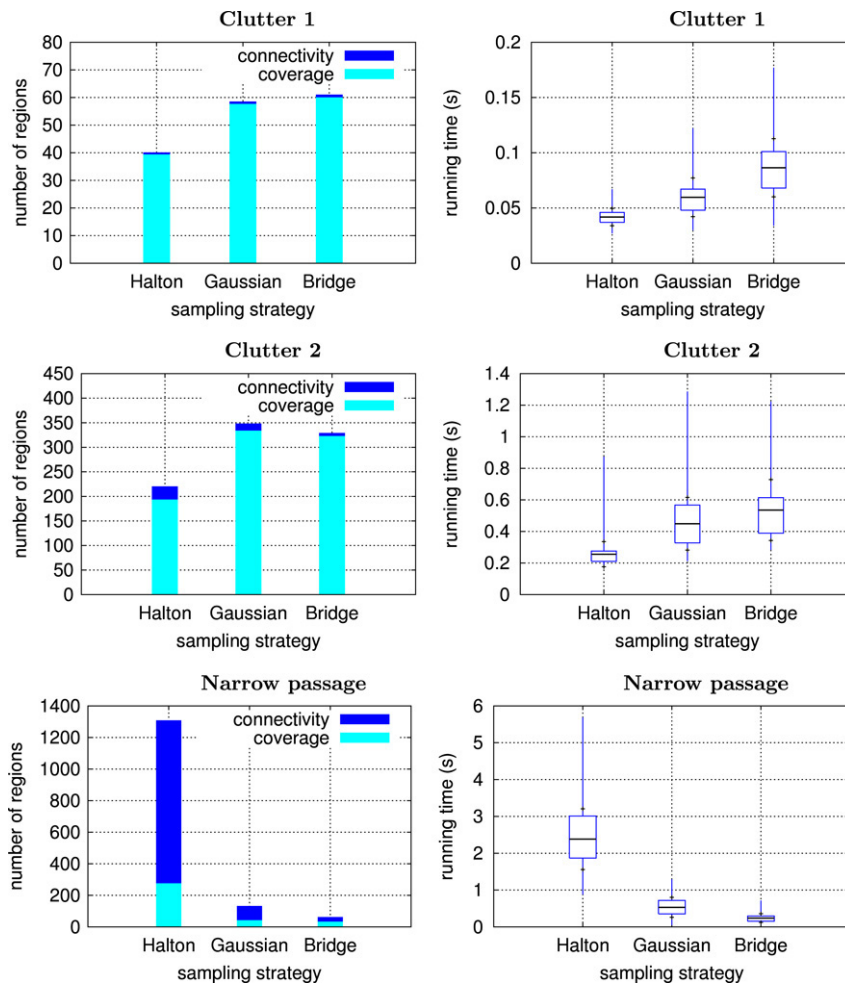


Fig. 9. Sampling statistics for the 2D environments. The charts in the left column show the average number of regions required to cover and connect the free space. In the Narrow passage environment, for example, Halton sampling required 275 regions to cover the free space and 1300 regions to connect the free space. The right column shows box plots corresponding to the time needed to satisfy both criteria.

areas and few in the narrow passage. Since the chance is small to obtain a set of samples that covers the space in the narrow passage, the total number of regions required for coverage and maximal connectivity was higher for this method than for the other two methods.

The Gaussian technique needed fewer samples than Halton in the Narrow passage environment. The reason for this is that relatively more samples are concentrated in the difficult areas of the  $C$ -space, which resulted in faster coverage. In addition, the ample free space was covered fewer times (due to the Gaussian distribution of the samples). As this distribution generates fewer samples in ample free space and more near obstacles, it can be difficult to connect them. This explains why connecting  $C_{free}$  involved more than three times as many regions compared to covering  $C_{free}$ .

The Bridge test technique has been designed to combine the strengths of the previous two categories by concentrating samples in difficult areas while some samples are also generated in open areas. This resulted in the lowest average running time and the lowest number of regions needed to get  $C_{free}$  connected in the Narrow passage environment. However, the technique performed moderately in the other environments. The charts

show that it had difficulties in getting  $C_{free}$  covered. This can be explained by looking at the properties of these environments. As they have no narrow passages, a technique that is designed to create samples in narrow passages will spend time uselessly. In the Hole environment, many Bridge samples were created near the walls while most of them did not increase the coverage of  $C_{free}$ . The same argument holds for the Manipulator environment.

By looking at the charts of Figs. 9 and 10, we can make an important observation. For the Clutter 1 and Clutter 2 environments, the difference between the moment that  $C_{free}$  was covered and the moment that  $C_{free}$  was maximally connected is very small. In contrast, for the Narrow passage environment this difference was much larger. Hence, covering  $C_{free}$  is not the problem, but getting  $C_{free}$  maximally connected is more difficult when the environment contains a narrow passage. To clarify this, we considered two versions of the Manipulator environment. The first variant is the one depicted in Fig. 3(f). The second variant is the same as the first one, except that we made the passages narrower by scaling the workspace in the  $y$ -direction, i.e. the workspace became 25% less high. In the first variant, the Halton sampling strategy needed 271 samples to

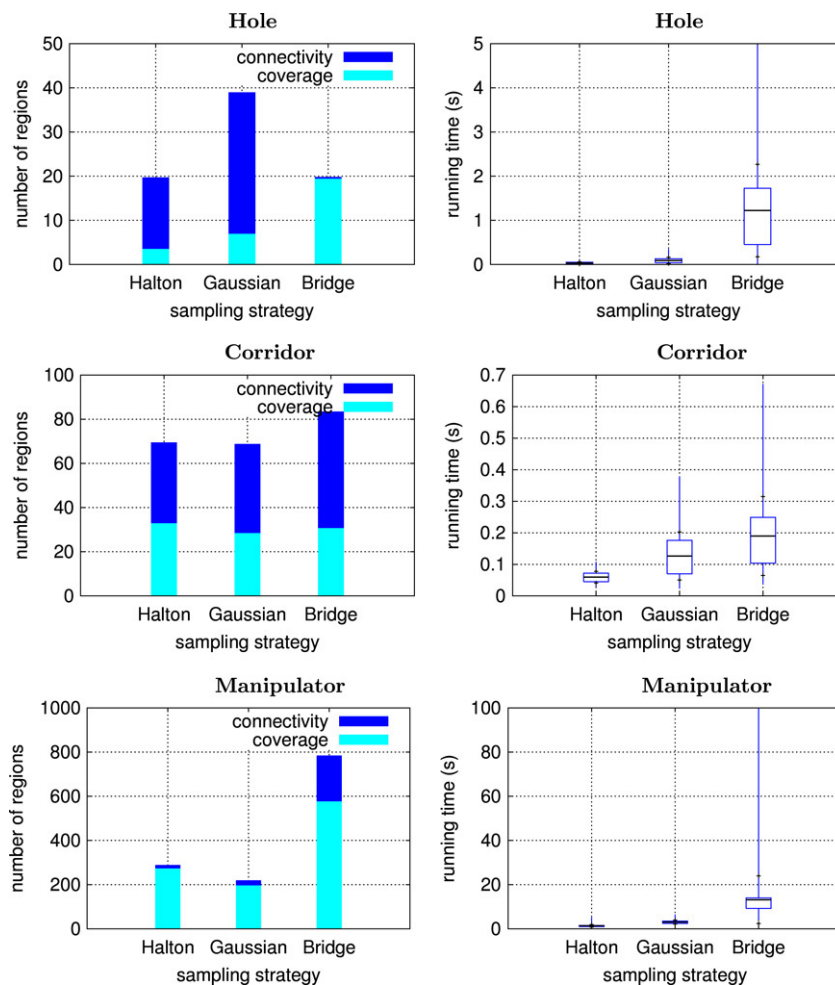


Fig. 10. Sampling statistics for the 3D environments.

cover the space. Only 6% more samples were needed to connect the space. While the second variant needed the same number of samples to cover the space, connecting the space required five times as many samples. Hence, connecting samples near or in narrow passages is more difficult. The main observation that can be made is that the narrow passage problem is not so much caused by coverage but by connecting the nodes. Rather than concentrating on more clever sampling, it may be beneficial to spend more effort on connecting nodes in difficult regions. Actually, already one of the first papers on PRM did this by trying to connect difficult nodes in a second phase using a bouncing strategy [8]. In Section 6, we will show how more powerful local planners can be used for this as well. The challenge is to apply such a connection strategy only when and where it is necessary.

#### Ideal sampling strategy

An ideal sampling strategy should create few samples that covers and connects  $C_{free}$ . The smaller the number of samples, the lesser the time needed to connect those samples which is the most time-consuming step in the PRM. However, some overlap between the regions that belong to the samples is required because this simplifies creating connections between them. This

can be achieved by creating a hybrid technique which filters out samples that do not contribute to extra coverage or maximal connectivity. The visibility sampling technique tries to achieve this by throwing away nodes [23]. It is shown in [24] that this technique has difficulties satisfying the coverage and maximal connectivity criteria. The approach is too strict and should probably be combined with other sampling techniques.

## 6. Local planners

In the previous sections we showed that it can be difficult to connect certain nodes while the coverage criterion has already been met. If we were able to create a local planner that is more powerful than the straight-line local planner (SLLP) described below, then we could decrease the gap between the moment of coverage and maximal connectivity, improving the total running time. Although this new planner might be more time-consuming, a careful trade-off between the power and speed of the planner should lead to a better performance of the PRM.

*sllp* The planner takes small steps along the path from start to goal. Let  $d$  be the distance between the start and goal, then the number of steps equals to  $d$  divided by an appropriate step size. We check each step

(i.e. the placement of the robot) for collisions with the environment.

To be successful, this planner should preferably satisfy the following two criteria. First, it must cover at least the same volume as the SLLP does, i.e. it must subsume each reachability region that is created using the SLLP. If the regions are larger we expect that the space is covered faster. More importantly, because of the larger expected overlap between the regions, they will sooner become maximally connected. Second, the planner should be fast enough to be useful in practice. This can be achieved by letting the planner behave as a SLLP if the connections can be made in a straight line; if the straight-line connection results in a collision, then a more clever approach should be employed. These criteria are satisfied by the simple potential field local planner (PFLP) we will describe below.

In general, a potential field method calculates distances between the robot and obstacles to define a force vector on the robot [10]. These operations make a PFLP expensive in comparison to the simple SLLP. To mitigate this effect, we use a modified version of the potential field planner used in [8].

*pflp* The planner tries to make small steps on the straight line toward the goal, as does the SLLP. This assures that the region reached by the PFLP subsumes the region reached by the SLLP. When the robot walks into an obstacle, the planner checks a step from the last collision-free configuration in several directions on the hemisphere in configuration space oriented toward the goal. The most promising step is considered first. A local minimum is easily detected when all possible steps fail in which case the local planner stops and reports failure.

Although the PFLP is more powerful, it will be more expensive than the SLLP in terms of consumed time. A second drawback is that a new parameter is introduced that has to be optimized, i.e. the number of directions on the unit sphere has to be chosen. We choose the axes and diagonals as directions, i.e. in 2D we choose 8 directions and in 3D we choose 26 directions on the unit sphere and select only those that bring the robot closer to the goal. It is a trade-off between the accuracy and speed: the higher this number, the larger the reachability region, but the slower the planner. The number of directions we chose seems to work reasonably, but it is in essence arbitrary.

Fig. 11 shows the Clutter 2 and Narrow passage environments for each of which a reachability region is drawn. The left pictures show the area that can be reached by the SLLP from a particular sample. The right pictures correspond to the PFLP. While reasonably long connections can be made by the SLLP, the reachability region of the PFLP significantly extends the area to which connections can be made. Besides the advantage of covering larger regions, the PFLP, in contrast to the SLLP, is able to find its way through the narrow passage. This allows connections to be made from one side of the passage to the other.

We expect that the PFLP outperforms the SLLP in all environments except the Corridor environment, because the

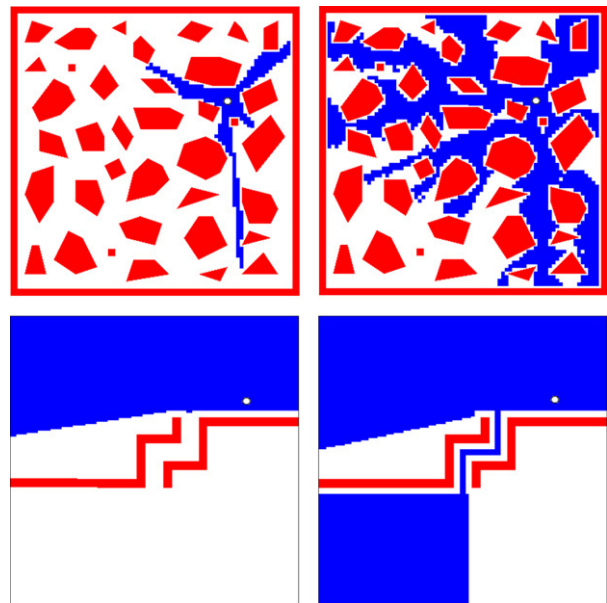


Fig. 11. Reachability regions for straight-line (left column) and potential field local planner (right column) in the Clutter 2 environment (top row) as well as the Narrow passage environment (bottom row).

reachability regions will be larger than those created with the SLLP. The PFLP will be able to easily connect the two ample free spaces in the Narrow passage and Hole environments. In contrast, in the Corridor environment, the PFLP may only have an advantage in the hairpins; much ineffectual work might be done elsewhere before it is concluded that no connections exist. This is expected to have a negative effect on the running time.

### Experiments

We used the optimal sampling strategy in all the experiments. That is, we used Halton for each environment except the Narrow passage environment where we used Bridge sampling. As neighbor selection strategy we used nearest- $k$ , where  $k$  was set to 75.

In the first experiment we study the effect of using the SLLP or PFLP on the average size of the regions. We set the maximum connection distance to infinity to reveal the full potential of the planners. Fig. 12 shows the results. In all the cases, the PFLP covered larger regions than the SLLP. For the Clutter 1 environment, the PFLP created regions that were on an average 86% of  $C_{\text{free}}$  compared to 27% for the SLLP. In the Clutter 2 environment, the regions of the PFLP were even ten times as large as the regions of the SLLP. Even in the Corridor environment, the differences were large.

Next, we conduct experiments to find out whether a PFLP (which creates larger regions, but takes more time per region) outperforms a SLLP (which creates smaller regions, but takes less time per region). To make a fair comparison possible, we use for both planners the optimal choices. The SLLP uses the neighbor selection parameters from Table 2. The maximum connection distances for the PFLP are set to  $\{\infty, \infty, 30, \infty, 20, 2.5\}$  for the different environments. The PFLP generally requires larger maximum connection distances as the regions will be larger.

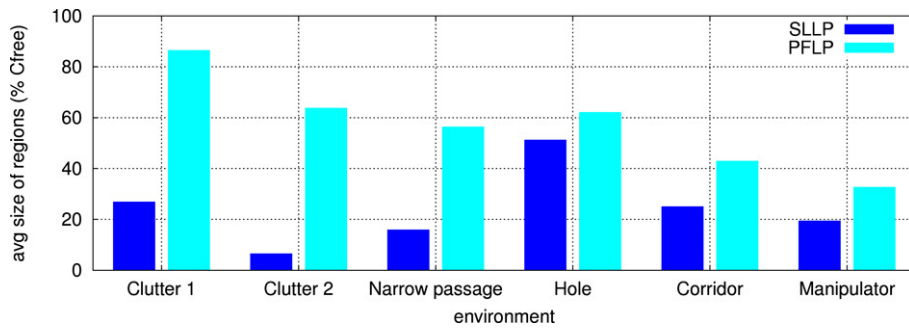


Fig. 12. The average size of the regions corresponding to the two local planners. The maximum connection distance has been set to infinity.

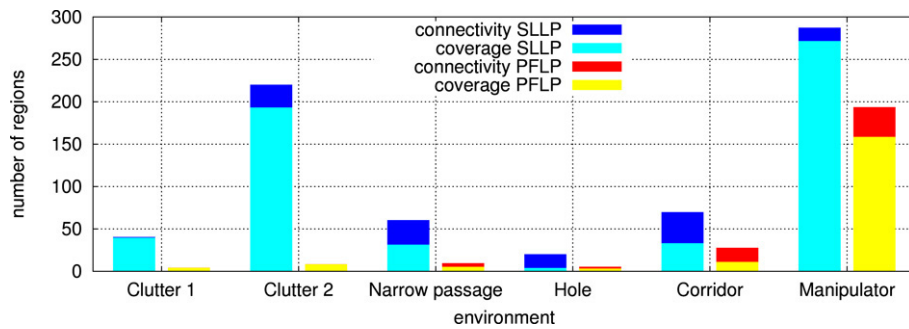


Fig. 13. Influence of the two local planners on the number of regions.

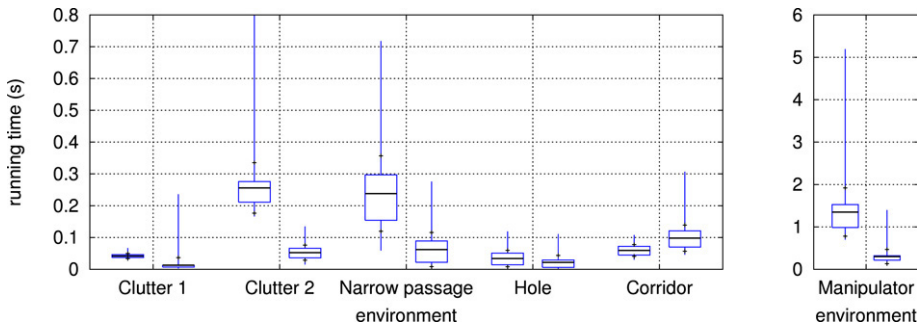


Fig. 14. Influence of the two local planners on the running time. Each environment was tested with the SLLP (left box plots) and the PFLP (right box plots).

Table 2  
The optimal values used in the neighbor selection strategy

	Neighbor selection parameters	
	Max. connection distance	Max. number of connections
Clutter 1	15	75
Clutter 2	10	75
Narrow passage	30	75
Hole	40	75
Corridor	15	75
Manipulator	2	75

Fig. 13 shows the average number of regions and Fig. 14 shows the running times for the two local planners. The results show, as expected, that the number of regions needed to cover the space are considerably lower for the PFLP, which is due to the larger average size of the regions. In addition, the maximal connectivity criterion was satisfied by considerably less regions. The (absolute) difference between the number of nodes to achieve coverage and number of nodes to achieve

maximal connectivity was much smaller for the PFLP than for the SLLP. The PFLP clearly outperformed the SLLP in all environments, except in the Corridor environment. But for this environment, the difference of the average running time is small. Thus, the PFLP turns out to be an efficient local planner.

Ideally, a local planner should be simple in an ‘easy’ part of the  $\mathcal{C}$ -space and more advanced in more ‘difficult’ parts. The potential field local planner combines those requirements: easy connections (i.e. straight-line connections) are made at the expense of a marginal overhead, while difficult connections (i.e. connections that avoid obstacles) can actually be made. Experiments showed that sampling-based methods can benefit from more powerful local planners such as the potential field local planner. Because the local planner is slower, the time improvement in general is less dramatic than the improvement in number of regions required. However, we believe that one could improve the time even further.

In future works, one could investigate techniques that can identify difficult regions in the space. This information can be

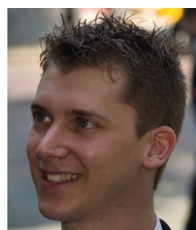
used to select the most appropriate local planner which should lead to an even better performance of the PRM.

## 7. Conclusion

While theoretically, the complexity of the motion planning problem is exponential in the number of degrees of freedom, sampling-based planners can successfully handle this curse of dimensionality in practice because they are reachability-based. We presented a reachability analysis for these planners which focused on coverage and maximal connectivity of the free configuration space  $C_{\text{free}}$ . By inspecting the roadmap when  $C_{\text{free}}$  was covered and when  $C_{\text{free}}$  was maximally connected, we obtained a better understanding of these planners. This led to the insight that not coverage but getting the nodes connected is the main problem, especially when the problems get more complicated, i.e. a narrow passage is present. The narrow passage problem can be tackled by incorporating a hybrid sampling strategy that aims at concentrating samples in difficult areas. The strategy must also generate some samples in large open areas. Other strategies to get the free configuration space faster connected are to use a refined neighbor selection strategy which is able to make long and many connections and a more powerful local planner. We presented a potential field local planner that creates larger reachability regions and accordingly eases making connections. This planner is also better able to find the entry of a narrow passage, decreasing the number of regions needed to get the nodes connected. Experiments showed that this approach leads to a better performance of the sampling-based methods.

## References

- [1] J. Reif, Complexity of the mover's problem and generalizations, in: IEEE Symposium on Foundations of Computer Science, 1979, pp. 421–427.
- [2] J. Schwartz, M. Sharir, On the Piano Movers' Problem II, Ablex Publishing Corporation, Norwood, NJ, 1987.
- [3] M. Ben-Or, D. Kozen, J. Reif, The complexity of elementary algebra and geometry, *Computer and Systems Sciences* 32 (1986) 251–264.
- [4] J. Canny, *The Complexity of Robot Motion Planning*, MIT Press, 1988.
- [5] J. Barraquand, Automatic motion planning for complex articulated bodies, Tech. Rep., Paris Research Laboratory, 1991.
- [6] J. Barraquand, L. Kavraki, J.-C. Latombe, T.-Y. Li, R. Motwani, P. Raghavan, A random sampling scheme for path planning, *International Journal of Robotics Research* 16 (1997) 744–759.
- [7] L. Kavraki, P. Švestka, J.-C. Latombe, M. Overmars, Probabilistic roadmaps for path planning in high-dimensional configuration spaces, *IEEE Transactions on Robotics and Automation* 12 (1996) 566–580.
- [8] M. Overmars, A random approach to motion planning, Tech. Rep. RUU-CS-92-32, Utrecht University, 1992.
- [9] R. Geraerts, M. Overmars, Sampling and node adding in probabilistic roadmap planners, *Robotics and Autonomous System* 54 (2006) 165–173.
- [10] J.-C. Latombe, *Robot Motion Planning*, Kluwer, 1991.
- [11] L. Kavraki, M. Kolountzakis, J.-C. Latombe, Analysis of probabilistic roadmaps for path planning, in: IEEE International Conference Robotics and Automation, 1996, pp. 3020–3025.
- [12] R. Geraerts, M. Overmars, Creating high-quality roadmaps for motion planning in virtual environments, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, 2006, pp. 4355–4361.
- [13] J. Kuffner, S. LaValle, RRT-connect: An efficient approach to single-query path planning, in: IEEE International Conference on Robotics and Automation, 2000, pp. 995–1001.
- [14] C. Nielsen, L. Kavraki, A two level fuzzy prm for manipulation planning, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, 2000, pp. 1716–1721.
- [15] G. Bergen, *Collision Detection in Interactive 3D Environments*, Morgan Kaufmann, 2003.
- [16] M. Branicky, S. LaValle, K. Olson, L. Yang, Quasi-randomized path planning, in: IEEE International Conference on Robotics and Automation, 2001, pp. 1481–1487.
- [17] V. Boor, M. Overmars, A. Stappen, The Gaussian sampling strategy for probabilistic roadmap planners, in: IEEE International Conference on Robotics and Automation, 1999, pp. 1018–1023.
- [18] N. Amato, O. Bayazit, L. Dale, C. Jones, D. Vallejo, OBPRM: An obstacle-based PRM for 3D workspaces, in: International Workshop on the Algorithmic Foundations of Robotics, 1998, pp. 155–168.
- [19] D. Hsu, T. Jiang, J. Reif, Z. Sun, The bridge test for sampling narrow passages with probabilistic roadmap planners, in: IEEE International Conference on Robotics and Automation, 2003, pp. 4420–4426.
- [20] D. Hsu, G. Sánchez-Ante, Z. Sun, Hybrid PRM sampling with a cost-sensitive adaptive strategy, in: IEEE International Conference on Robotics and Automation, 2005, pp. 3885–3891.
- [21] B. Chazelle, *The Discrepancy Method*, Cambridge University Press, Cambridge, 2000.
- [22] X. Wang, F. Hickernell, Randomized Halton sequences, *Mathematics and Computer Modeling* 32 (2000) 887–899.
- [23] C. Nissoux, T. Siméon, J.-P. Laumond, Visibility based probabilistic roadmaps, *Advanced Robotics Journal* 14 (2000) 477–493.
- [24] R. Geraerts, M. Overmars, Creating small roadmaps for solving motion planning problems, in: IEEE International Conference on Methods and Models in Automation and Robotics, 2005, pp. 531–536.



**Roland Geraerts** received his Ph.D. degree in Computer Science in 2006 from Utrecht University in The Netherlands. Currently he is a researcher/lecturer at the Department of Computer Science at the same university. His interests include motion planning and virtual environments.



**Mark H. Overmars** received his Ph.D. degree in Computer Science in 1983 from Utrecht University in The Netherlands. Currently he is a full professor at the Department of Computer Science at the same university, where he is scientific director of the Center for Advanced Gaming and Simulation (AGS; [www.gameresearch.nl](http://www.gameresearch.nl)).

His main research interests include motion planning, virtual environments, and game design. Over the past years he has published over 250 papers in refereed journals and conferences, and he is the author of one of the prime textbooks on computational geometry.