Viewing Motion Planning as Disassembly: A Decomposition-Based Approach for Non-Stationary Robots

Technical Report 04-108

Yuandong Yang   Oliver Brock
Laboratory for Perceptual Robotics
Department of Computer Science
University of Massachusetts Amherst

December 2004

Abstract

An efficient single-query motion planning approach for non-stationary robots is presented. The approach exploits workspace information to identify highly constrained placements of the robot along a potential solution path. Due to the constraints imposed by the environment at these placements, we view them as an assembly, consisting of the robot and the environment. The original motion planning problem is then decomposed into a series of disassembly tasks. Each assembly is disassembled by moving the robot from the narrow passage into adjacent open regions. This disassembly can be performed efficiently by exploiting the geometric constraints imposed by the environment. To obtain a solution to the original motion planning problem, the resulting disassembly paths for two consecutive disassemblies along the solution path can be connected efficiently, as they meet in an open, unconstrained region of the workspace. Experimental evidence shows that the proposed approach results in a significant reduction in computational cost for motion planning in high-dimensional configuration spaces.

1 Introduction

Global motion planning has been shown to be PSPACE-complete in the general case [9, 17]. The fastest known complete motion planning algorithm requires computation time exponential in the number of degrees of freedom of the robot [5]. Intuitively, the exponential computational complexity of motion planning in configuration space can be explained by the need to compute a description of configuration space obstacles in a space of exponential size, followed by a search in this exponential space for a collision-free path.

The most efficient motion planners for high-dimensional spaces found in the literature sample configuration space to construct a so-called roadmap [11]. Rather than computing a detailed, explicit description of the boundary between free space and obstacles [13], the roadmap determined by these planners captures an approximation of that boundary. The space to be represented and searched, however, remains exponential in size.

The motion planning approach presented in this paper is based on the observation that geometric constraints in the workspace can be exploited to restrict the search in configuration space to a small, relevant subset. This observation is particularly apparent in disassembly tasks: in an assembly, geometric constraints restrict the motion of individual parts [20]. These constraints can be exploited to efficiently determine the allowed directions of motion of each part, and thus a disassembly motion.

We propose to view the general motion planning problem as a series of disassembly tasks. Along a given path, we regard those configurations that are highly constrained by their environment as *assemblies*. In the sampling-based motion planning literature, these configurations are said to be inside *narrow passages*. The disassembly can take advantage of the geometric constraints imposed by the environment to direct search in configuration space. If a motion planning problem requires a solution that traverses multiple narrow passages (or contains multiple assemblies), the overall problem is decomposed into a sequence of disassembly problems. The solutions to these disassembly problems can be connected efficiently, as motion planning between the assemblies is much easier. The proposed approach effectively decomposes the overall motion planning problem into a number of difficult and easy sub-problems.

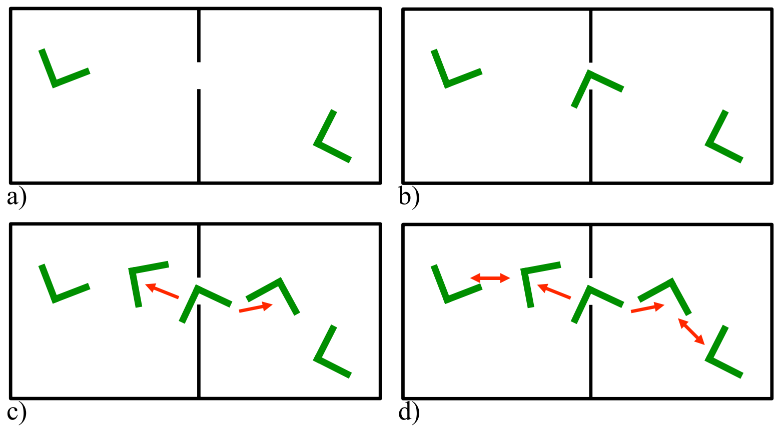

Figure 1: Decomposition of a path planning problems into two disassembly problems: a) initial and final configurations of a robot; b) "assembled" state; c) constraints imposed by the environment are used to solve two disassembly problems; d) the motion planning problem is solved by connecting the disassembly sequence with the initial and final configurations.

The decomposition of the overall motion planning problem into disassembly tasks is illustrated in Figure 1. Figure 1 a) shows the original motion planning problem with the initial and the final configuration of an L-shaped robot. Motion planning inside the left or right part of the environment is trivial, only the traversal of the narrow passage is difficult. In Figure 1 b) the assembly is shown: a placement of the robot inside the narrow passage. Using the geometric constrains imposed by the environment, the narrow passage can be solved by iteratively moving the robot to either side. Once the robot has cleared the narrow passage, the remaining motion planning problem is trivial, as shown in Figures 1 c) and d).

We present a single-query, sampling-based motion planning approach for non-stationary robots based on these ideas and demonstrate that it outperforms other sampling-based motion planners by a large margin.

2 Related Work

The original probabilistic roadmap (PRM) approach [11] uniformly samples the entire configuration space to build a representation of free space. Using the resulting representation, any motion planning query can be answered efficiently. Such approaches are referred to as multi-query approaches. Many motion planners presented in the literature aim to improve the performance of motion planning based on uniform sampling by limiting the amount of configuration space exploration required to solve a motion planning problem.

Some approaches improve performance by replacing the uniform sampling scheme with sampling informed by workspace information. By incrementally modifying a uniformly placed sample so that the robot is close to the medial axis of workspace, the quality of generated samples can be improved [8]. Information obtained in the workspace can also be exploited to locally adapt the sampling density [19, 21]. The resulting methods sample less densely in configuration space regions that correspond to open spaces in the work space, resulting in a reduced cost of the overall exploration. Such an adaptation of sampling density can also be accomplished directly in the configuration space by exploiting models and active learning methods from machine learning [3].

Single-query planners answer a single motion planning query [1, 10, 12, 14, 16, 18]. Exploration ends when a solution is found. The different methods vary in the heuristics they use to guide the exploration. These heuristics are designed to minimize exploration based on insights about the planning problem. For example, a bias towards areas of configuration space in the proximity of initial and final configuration of the planning problem has been proposed [1]. In another approach, the Voronoi bias directs exploration towards unexplored regions [14]. The use of local information about the environment obtained from collision checks has also been proposed to efficiently solve industrial disassembly problems [6]. Another single-query motion planning heuristic is based on an information theoretic framework [4].

Decomposition-based motion planning methods [2, 7] decompose the motion planning problem into a low-dimensional and a high-dimensional subproblem. The low-dimensional motion planning problem can be solved efficiently in the workspace. The solution to this problem captures connectivity information in the workspace relevant to the high-dimensional planning problem. By exploiting this information, a solution to the overall motion planning problem can be computed efficiently. Decomposition-based motion planning methods inherently are single-query methods, because they exploit the workspace connectivity information related to a particular planning problem. The motion planning method presented in this paper falls into the category of decomposition-based motion planners.

3 Motion Planning as Disassembly

The goal of the proposed motion planning method is to exploit information obtained in the workspace to reduce the amount of configuration space exploration required to solve a given motion planning problem. We now give an overview of the three main steps of the algorithm.

Determining Workspace Connectivity The proposed motion planning approach can be classified as decomposition-based [2]. As such, it decomposes the planning problem into two subproblems. The first subproblem is solved by determining workspace connectivity information. The result is a continuous tunnel of obstacle-free workspace, connecting the initial and final positions of the robot. This tunnel will be used to limit configuration space exploration to only those configurations for which the robot is partially inside the tunnel.

Finding Assemblies We regard the width of the tunnel as an indication of difficulty of the motion planning problem. Consequently, a collision-free placement of the robot inside a narrow area can be viewed as an assembly. Sampling is used to find a collision-free placement of the robot in such a narrow area. We will argue that the resulting highly constrained placement of the robot effectively represents a seed point in configuration space, from which a solution to the disassembly problem can be easily found. The assemblies decompose the original motion planning problem into a series of disassembly problems.

Performing Disassembly Once assembled states for the robot have been determined, the resulting disassembly problems can be solved by exploiting the geometric constraints imposed by the environment on the robot's motion. Given the assembled state of the robot, the constraints generally will only allow small incremental motions. Therefore, sampling can be focused in the proximity of the current configuration, reducing the amount of configuration space to be explored. In addition, the workspace tunnel provides a general direction for the workspace motion of the robot, resulting in an additional reduction of configuration space exploration.

3.1 Determining Workspace Connectivity

A robot sweeps out a workspace volume as it moves from its initial to its final configuration. If this workspace volume were known, the exploration of configuration space could be restricted to the subset of configurations for which the robot is contained within this volume. Decomposition-based approaches to motion planning compute an approximation of this workspace volume, called a workspace tunnel [2]. The exploration of configuration space can then be restricted to those configurations for which the robot partially overlaps with the tunnel. The set of qualifying configurations represents a small subset of the overall configuration space and consequently exploration can be performed much more efficiently.

The workspace tunnel is computed using a sphere expansion algorithm. The details of this algorithm are given elsewhere [2]; here, we outline the general idea. The purpose of the sphere expansion algorithm is to determine a contiguous workspace volume through which the robot might be able to move from its initial position to its final position. To compute this volume we use a wavefront of free space spheres with maximum radius. Initially, the largest sphere of free space centered at a reference point of the robot in its initial position is computed. The radius of this sphere is determined by the distance of the reference point to the closest obstacle. The surface of the sphere is sampled and maximal spheres of free space centered at the sample points are determined. If the size of a sphere does not allow the robot to move through it, it is discarded. The remaining spheres are kept in a priority queue, with the highest priority assigned to the sphere closest to the final position of the robot. This process is referred to as sphere expansion. Expansion of the highest-priority sphere is performed until a sphere contains the reference point of the robot in its final position. Figure 2 shows the resulting free space tunnel.

During sphere expansion, the parent/child relationship of spheres is maintained. The resulting data structure is a tree of spheres. The root sphere contains the reference point of the robot in its initial position. The spheres of the tunnel represent a path from the root of the tree to the leave containing the reference point in the final position of the robot. The line segments connecting the centers of the spheres along the tunnel are referred to as the spine of the tunnel.

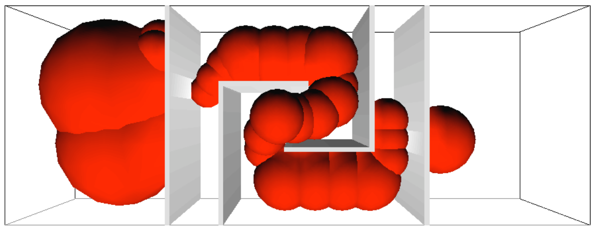

Figure 2: Workspace tunnel computed using sphere expansion, connecting the initial position of the robot in the bottom, left part of the environment and the final position in the right part.

The sphere expansion method cannot guarantee that the computed tunnel contains a valid solution to the planning problem. By exploring multiple tunnels in workspace, however, the probabilistic completeness of other sampling-based motion planners can be achieved for the planner proposed here. For brevity, we restrict our discussion to a single tunnel. The extension of the described method to multiple tunnels is straightforward. It performs a complete exploration of workspace, as described in [21], followed by the repeated selection of tunnels and an application of the method described here, until a solution has been found or no unexplored tunnel exists.

3.2 Finding Assemblies

Highly constrained, collision-free placements of the robot are referred to as an assembly. They represent a placement of the robot inside a narrow passage in the work space. To find assemblies, narrow sections of the computed tunnel are considered. A section is narrow, if the radius of spheres in this section is below a threshold. This threshold is a parameter of the proposed approach. It can easily be estimated based on the geometry of the robot.

To improve the understanding of the workspace around narrow regions, local sphere expansion is performed, starting from the center of spheres in narrow regions of the tunnel. This expansion explores the free workspace reachable by the robot when affixed to the center of the spheres. To divide the motion planning problem into disassembly problems, the most difficult regions along the tunnel have to be identified. This can be accomplished using a watershed algorithm applied to the union of spheres contained in the tunnel and the additional spheres obtained in the additional exploration [19]. The result will be a set of spheres representing each narrow passage, labeled by the narrow passage they belong to. Spheres between two adjacent narrow passages are also labeled distinctly, likewise the spheres connecting the initial and the final position to the first and last narrow passage along the tunnel.

An example of this such a labeling is given in Figure 3. Figure 3 a) shows the initial and final position of the robot and the environment with a narrow passage. Figure 3 b) illustrates the tunnel computed by sphere expansion, as described in Section 3.1. The spheres obtained by the local expansion and the labeling determined by the watershed algorithm are shown in Figure 3 c).

An assembly is found by sampling robot configurations in the proximity of the narrow passage. Uniform sampling is adequate in this situation, because the entire region has uniform complexity. To generate a sample inside the narrow passage, a value for the rotational degrees of freedom of the robot is generated uniformly at random. In addition, a random point on the robot and a random point inside the volume of the narrow passage are chosen. The translational component of the robot's configuration is determined such that these two points coincide.

Since an assembly should represent a highly constrained, collision-free placement of the robot, we require that the robot reaches across a narrow passage. For any collision-free configuration, we require that two so-called handle points (see Figure 4) are contained in spheres with different labels outside the narrow region. An example of such a placement can be seen in Figure 3 d). Collision-free samples that do not fulfill this criterion are rejected. Sampling ends, if a small number of assemblies have been determined. If no assembly can be found, a new tunnel has to be computed.

3.3 Performing Disassembly

So far only a very small fraction of the configuration space has been explored by uniform sampling. These regions correspond to narrow passages along a workspace tunnel connecting the initial and the final goal configuration. This computational expense was necessary, however, to solve the most difficult parts of

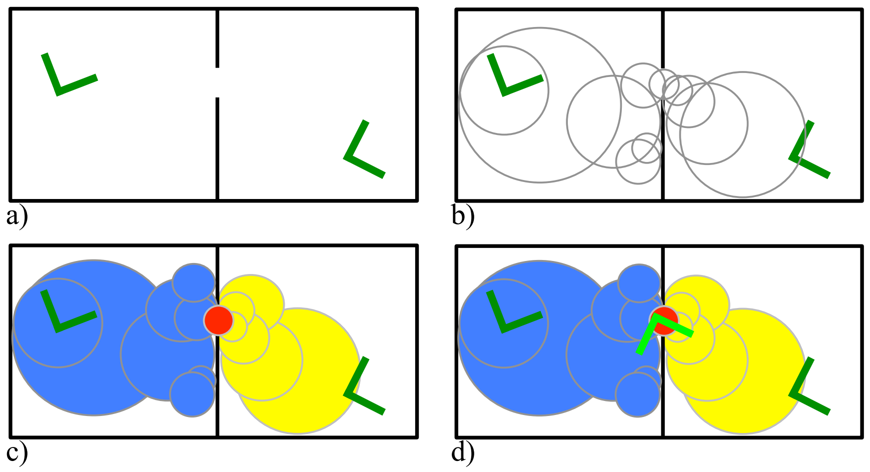

Figure 3: Decomposition of the motion planning problem based on assemblies: a) the initial and final position of a robot; b) workspace tunnel connecting the initial and final positions; c) spheres are added to improve workspace understanding around narrow passages, spheres are labeled as narrow passage and open region; d) the assembled state of the robot decomposes the motion planning into two subproblems: moving from the assembled state to the initial and final configurations.

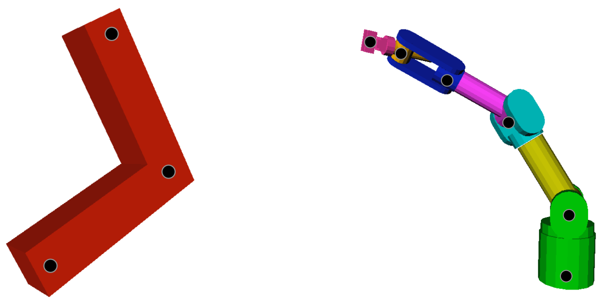

Figure 4: Robots used in the experiment. Left: six degree-of-freedom, L-shaped rigid robot, composed of two overlapping blocks ($0.85m \times 0.2m \times 0.2m$); right: free-flying Mitsubishi PA-10 with 13 degrees of freedom, total length $1.317\ m$. Black dots indicate handle points.

the motion planning problem. The resulting assemblies give us much information about solutions for the remaining disassembly problems. We will use the configuration of the assembly to bias our sampling scheme for disassembly. The sampling scheme is motivated by two insights:

1. the translational component of the disassembly motion should be biased to move the robot into an open regions and out of the narrow passage, and
2. due to the constraints imposed by the environment, only small incremental motions should be attempted.

Pseudo code for the proposed sampling procedure for disassembly is shown in Figure 5. Given a set A of assembled placements of the robot inside the narrow passage, this procedure builds a roadmap for each of the $a \in A$. Samples are obtained by small perturbations of milestones already present in the roadmap. This effectively uses the information represented by the initial assembly and subsequently the entire roadmap to reduce configuration space exploration by biasing it towards regions likely to contain collision-free placements of the robot. The translational component of the configuration is perturbed with a bias towards the open regions of workspace represented by the tunnel. This bias exploits workspace information to further reduce configuration space exploration, since promising directions for translational motion are sampled more densely.

```
DISASSEMBLE (assemblies $A$, tunnel $T$)
  initialize roadmap $R$ to contain assemblies $A$
  while robot has not left narrow passage
    randomly select milestone $m$ from $R$
    randomly select direction $d$ towards open region in $T$
    perturb translation of $m$ biased by $d$ to obtain $m'$
    perturb rotational components of $m'$ to obtain $m''$
    if $m''$ is collision free
      if $r \in R$ exists so that $m''$ can be connected to $r$
        insert $m''$ and edge $(r, m'')$ into $R$
```

Figure 5: Pseudo code for disassembly; A represents a list of assembled states, T refers to the workspace tunnel, R designates a roadmap; m, m', m'', r are configurations.

By selecting configurations for perturbation uniformly at random the roadmap R, rather than in a biased fashion as in other approaches [10, 14], we effectively achieve a bias towards promising regions of configuration space. This can be seen as follows. Since the configuration space region under consideration represents a narrow passage, collision-free samples will be rare. Once a collision-free sample has been found, its neighborhood is likely to contain additional free placements of the robot. This probability increases, as configurations move away from the assembled state, effectively introducing a bias from constrained regions towards less constrained regions. This bias is desired in the case of a disassembly task.

An assembly is considered to be disassembled, once the robot has been removed from the narrow section of the tunnel. Consecutive disassembled configurations along the tunnel are connected using the traditional PRM framework with uniform sampling [11]. Only samples in the proximity of the relevant section of the workspace are retained. Consequently, the PRM framework is only applied to a small and open region of the configuration space.

By concatenating disassembly sequences obtained by the proposed sampling scheme and intermediate paths obtained using a localized PRM planner, a solution to the initial motion planning problem can be determined. A localized PRM planner uses the sampling scheme described in Section 3.2, instead of a uniform sampling scheme. This restricts the generation of samples to the region of interest by ensuring that the robot intersects with the relevant workspace region.

4 Experimental Results

We evaluate the performance of the proposed planning method by comparison with four other sampling-based approaches: a PRM planner with uniform sampling [11], the RRT method [14], a LazyPRM planner [1], and a single-query version of the aMAPRM method based on medial axis sampling [21]. The PRM planner with uniform sampling represents a multi-query method and thus the comparison to the proposed multi-query method has to be seen as a reference point. As we will see, however, two of the single-query methods are unable to solve some of the experimental scenarios used here.

The experiments are performed with a free-flying, L-shaped rigid-body robot (six degrees of freedom) and a free-flying Mitsubishi PA-10 manipulator arm (thirteen degrees of freedom), both shown in Figure 4. The three experimental environments are shown in Figure 6. An example motion is shown in Figure 7. The implementation of all motion planners is based on the Motion Strategy Library (MSL) [15]. The implementations of the PRM and RRT planners provided by the MSL library are used in the experiments reported here. The experimental results are summarized in Table 1. The proposed algorithm, labeled DPRM for disassembly-based PRM planner, outperforms all other motion planners by several orders of magnitude. Other authors have also observed significantly reduced planning times for a diffusion-based motion planner applied to part disassembly problems [6].

Table 1 shows that the roadmap constructed by the disassembly-based PRM planner contains very few milestones. This implies that the assemblies determined during motion planning effectively capture the difficulty of the overall motion planning problem. Once the assembly has been generated (assemblies are milestones in the roadmap), very few additional samples suffice to solve the remainder of the motion planning problem. Consequently, the efficiency of the proposed planner results from the fact that exploration of configuration space is focused on finding those assemblies. The information represented by assemblies can be used to solve the remainder of the motion planning problem with little exploration.

5 Conclusion

We propose to view single-query motion planning problems for non-stationary robots as a sequence of disassembly tasks. A given motion planning problem is decomposed into a series of sub-problems. This decomposition is determined using workspace information. Difficult regions, corresponding to narrow passages in the workspace, are solved efficiently by regarding them as a disassembly task. To solve such a disassembly task, an assembled (highly constrained) placement of the robot is determined by uniform sampling of a small region of configuration space. The constraints imposed by the environment on the resulting assembly are used to guide the sampling-based process of disassembly. Using a PRM planner with uniform sampling, the path segments obtained by the disassembly are connected to yield a solution to the original motion planning problem.

In the experiments presented, the proposed motion planning approach outperforms other sampling-based methods by several orders of magnitude. The efficiency of the proposed motion planner results from the effective use of workspace information to reduce the amount of configuration space that has to be explored during the planning process. Based on the connectivity information about the workspace and the information represented by geometric constraints imposed on assemblies by the environment, a very small fraction of

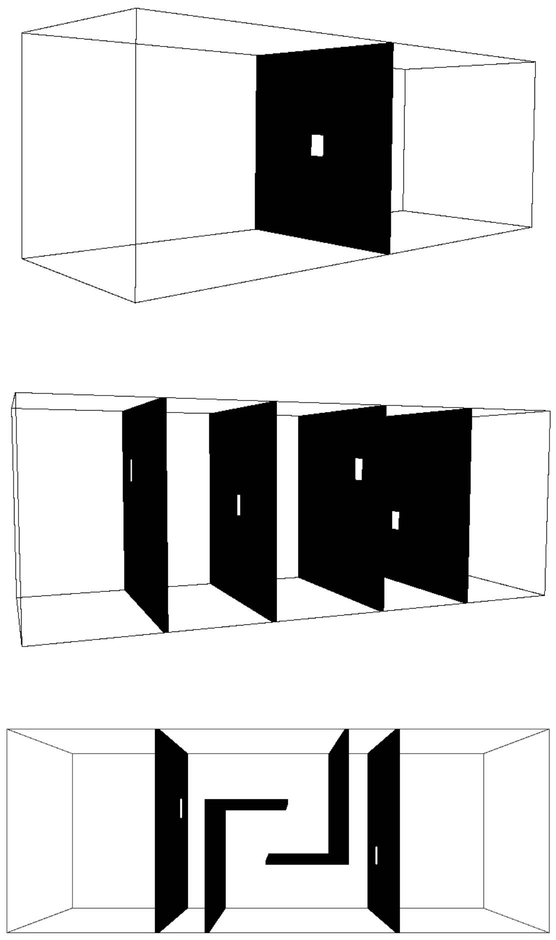

Figure 6: All experimental environments are bounded by a box ($12m \times 4.5m \times 4.5m$). In the first environment (board, left) to two parts of the free space are connected by a narrow passage ($0.1m \times 0.5m \times 0.5m$). The second environment (4 boards, middle) has five free space regions connected by narrow passages of the same size. The third environment (S-tunnel, right) contains three free space regions connected by narrow passages as before; an S-shaped obstacle is placed in one of the free space regions.

Robot	Workspace Computation			Path Planning					Total
	Environment	N_s	$T_s(s)$	Planner	N_v	N_e	N_c	$T_c(s)$	$T(s)$
L-shaped robot	board	N/A	N/A	PRM	4201	8400	2859848	586.15	586.15
		N/A	N/A	RRT	no path found after 6 hours				
		N/A	N/A	LazyPRM	no path found after 6 hours				
		89	**0.53**	**DPRM**	**56**	**107**	**15105**	**2.87**	**3.40**
	4 boards	N/A	N/A	PRM	no path found after 6 hours				
		N/A	N/A	RRT	no path found after 6 hours				
		N/A	N/A	LazyPRM	no path found after 6 hours				
		310	**2.34**	**DPRM**	**271**	**541**	**59508**	**35.94**	**38.28**
	S-tunnel	N/A	N/A	PRM	no path found after 6 hours				
		N/A	N/A	RRT	no path found after 6 hours				
		N/A	N/A	LazyPRM	no path found after 6 hours				
		1011	**4.81**	**DPRM**	**103**	**203**	**15989**	**5.37**	**10.18**
PA-10	board	N/A	N/A	PRM	18259	36516	29849130	6962.02	6962.02
		N/A	N/A	RRT	56243	56242	3085118	14473.80	14473.80
		N/A	N/A	LazyPRM	no path found after 6 hours				
		377	5.75	aMAPRM	333	662	321129	211.85	217.60
		86	**0.52**	**DPRM**	**21**	**39**	**2558**	**2.14**	**2.66**
	4 boards	N/A	N/A	PRM	18584	37164	4193077	8653.04	8653.04
		N/A	N/A	RRT	no path found after 6 hours				
		N/A	N/A	LazyPRM	no path found after 6 hours				
		448	7.59	aMAPRM	984	1934	2128272	1615.79	1623.38
		435	**2.81**	**DPRM**	**112**	**222**	**11427**	**15.54**	**18.35**
	S-tunnel	N/A	N/A	PRM	27278	54554	5198260	19827.20	19827.20
		N/A	N/A	RRT	no path found after 6 hours				
		N/A	N/A	LazyPRM	no path found after 6 hours				
		598	10.27	aMAPRM	1300	2560	1667714	1625.73	1636.00
		1078	**5.38**	**DPRM**	**219**	**433**	**25600**	**28.30**	**33.68**

TABLE 1: Comparison of a PRM planner with uniform sampling, an RRT planner, a LazyPRM planner, a single-query aMAPRM planner, and a disassembly-based PRM planner (DPRM). N_s is the number of spheres generated during sphere expansion, T_s represents the time to compute the workspace connectivity information; N_v denotes the number of vertices in the roadmap, N_e refers to the number of edges in the roadmap, N_c specifies the total number of collision checks, and T_c gives the duration of roadmap construction. All times are averaged over ten runs are are given in seconds. The experiments were performed on a PentiumIV 3.2GHz PC with 1GB RAM and a 64MB DDR Radeon 300 graphics card.

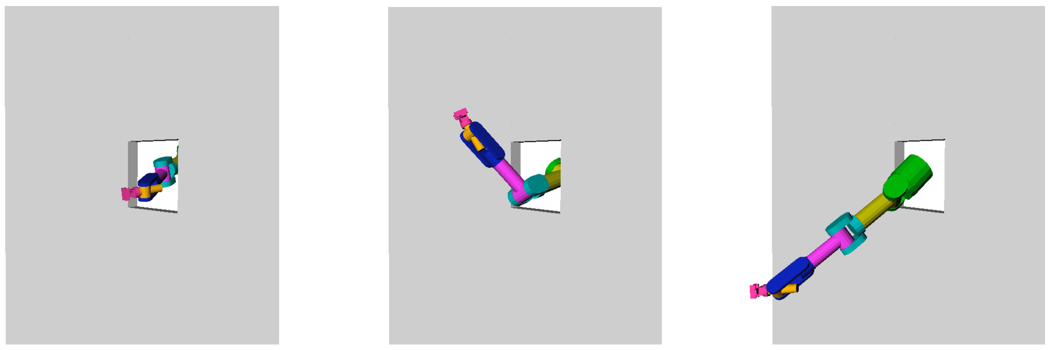

Figure 7: Motion of a PA-10 manipulator passing through a narrow passage. The middle image represents an assembly and the left and right images show intermediate configurations during disassembly into the open regions on either side of the narrow passage.

the overall configuration space is identified as relevant to the motion planning problem. The reduction of exploration to a small fraction of the overall configuration space, makes the proposed method computationally very efficient.

References

[1] R. Bohlin and L. Kavraki. Path planning using Lazy PRM. In *Proceedings of the International Conference on Robotics and Automation*, volume 1, pages 521–528, 2000.

[2] O. Brock and L. Kavraki. Decomposition-based motion planning: A framework for real-time motion planning in high-dimensional configuration spaces. In *Proceedings of the International Conference on Robotics and Automation*, volume 2, pages 1469–1474, 2001.

[3] B. Burns and O. Brock. Sampling-based motion planning using predictive models. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Barcelona, Spain, 2005. Submitted for review.

[4] B. Burns and O. Brock. Single-query entropy-guided motion planning. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Barcelona, Spain, 2005. Submitted for review.

[5] J. F. Canny. *The Complexity of Robot Motion Planning*. MIT Press, 1988.

[6] E. Ferre and J.-P. Laumond. An iterative diffusion algorithm for part disassembly. In *International Conference on Robotics and Automation(ICRA)*, pages 3149–3154, April 2004.

[7] M. Foskey, M. Garber, M. C. Lin, and D. Manocha. A Voronoi-based hybrid planner. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 55–60, 2001.

[8] C. Holleman and L. E. Kavraki. A framework for using the workspace medial axis in PRM planners. In *Proceedings of the International Conference on Robotics and Automation*, pages 1408–1413, 2000.

[9] J. E. Hopcroft and G. Wilfong. Reducing multiple object motion planning to graph searching. Technical Report TR 84-616, Cornell University, 1984.

[10] D. Hsu, J.-C. Latombe, and R. Motwani. Path planning in expansive configuration spaces. *International Journal of Computational Geometry and Applications*, 9(4):495–512, 1999.

[11] L. Kavraki, P. Svestka, J.-C. Latombe, and M. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. Technical Report CS-TR-94-1519, Rice University, 1994.

[12] J. Kuffner and S. LaValle. RRT-Connect: An efficient approach to single-query path planning. In *Proc. IEEE Int'l Conf. on Robotics and Automation*, pages 995–1001, 2000.

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

[14] S. LaValle. Rapidly-exploring random trees: A new tool for path planning. Technical Report TR 98-11, Computer Science Dept., Iowa State University, 1998.

[15] S. M. LaValle. http://msl.cs.uiuc.edu/msl.

[16] S. R. Lindemann and S. M. Lavalle. Incrementally reducing dispersion by increasing voronoi bias in RRTs. In *International Conference on Robotics and Automation(ICRA)*, pages 3251–3257, April 2004.

[17] J. H. Reif. Complexity of the mover's problem and generations. In *Proceedings of the Symposium on Foundations of Computer Science*, pages 421–427, 1979.

[18] D. Vallejo, I. Remmler, and N. M. Amato. An adaptive framework for single shot motion planning: A self-tuning system for rigid and articulated robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 21–26, Seoul, Korea, 2001.

[19] J. P. van den Berg and M. H. Overmars. Using workspace information as a guide to non-uniform sampling in probabilistic roadmap planners. In *International Conference on Robotics and Automation(ICRA)*, pages 453–460, April 2004.

[20] R. H. Wilson and J.-C. Latombe. Geometric reasoning about mechanical assembly. *Artificial Intelligence*, 71(2):271–396, 1994.

[21] Y. Yang and O. Brock. Adapting the sampling distribution in PRM planners based on an approximated medial axis. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, 2004.