A Slicing Connection Strategy for Constructing PRMs in High-Dimensional Cspaces

Abstract - **This paper presents a connection strategy for PRM-based motion planning in high-dimensional configuration spaces. The connection strategy identifies components of motion during the roadmap construction and plans the roadmap edges on a sequence of configuration slices approximating the configuration spaces of the components. The goal of the approach is to extend the dimensionality of the problems PRM planners can solve, and the strategy it deploys is dimensional subdivision. We report experimental results on serpentine robots with up to 150 degrees-of-freedom and multi-robot systems with up to 144 degrees-of-freedom. The test problems are beyond the capability of a representative state-of-the-art PRM based planner, and the approach presented in this paper enables the use of motion planning methods in solving new high-dimensional problems.**

Index Terms – Motion planning, probabilistic roadmaps, connection strategies, configuration space slicing.

I. INTRODUCTION

Probabilistic roadmap (PRM) algorithms have become one of the most successful approaches to motion planning [1]. PRM motion planners have a solid theoretical foundation [2][3], and they are known to perform very well with most realistic problems. The two major strategies used in a PRM planner are the sampling strategy and the connection strategy. Sampling strategy picks configurations to become the nodes in the roadmap. The connection strategy selects prospective sample pairs and attempts to connect them with local paths. A considerable amount of research has addressed the well-known "narrow passage" limitation of PRMs and led to new sampling strategies. The goal of this paper, however, is to address the fundamental challenge in motion planning and extend the dimensionality of the configuration space (*cspace*) that a PRM planner can deal with by proposing a novel connection strategy. In particular, this paper contributes a new approach to generate local paths that is intended to deal with large dimensionality of *cspace*.

The PRM technique was introduced for planning in high-dimensional configuration spaces. The early results included problems with up to 12 degrees-of-freedom (*DOF*) [4]. More recent papers include results on test problems with few dozens of degrees-of-freedom [5]. The complexity results for motion planning state that the problem is exponential in the number of degrees-of-freedom [6]. While the theoretical analysis of PRM is based on ratios of volumes of visibility sets and the ratios do not directly depend on the dimensionality of the *cspace*, a given number of sampled configurations in the roadmap become increasingly sparse as the dimensionality increases. This implies that better visibility properties are necessary for PRMs to remain efficient in high dimensions. But, on the contrary, our experiments indicate that self-collisions cause the visibility in high-dimensional *cspaces* to become less favorable. As a result, PRM algorithms usually succumb to roadmap exhausting the available memory when trying to cope with the compound effect. Sophisticated sampling techniques can remedy the situation up to a point, but eventually they too will have to give in to the samples becoming too sparse even in the relatively high density areas. Despite the fundamental intractability of the problem, development of motion planning techniques that extend the dimensionality of the solvable problems is motivated by many novel applications involving systems with many degrees-of-freedom such as multi-robot systems, humanoid robots, digital actors and virtual characters, and molecular complexes.

The new connection strategy presented in this paper exploits the existence of *components of motion*: relatively decoupled subsets of the degrees-of-freedom that allow efficient local motion planning using predominantly those degrees-of-freedom. In many systems, such components emerge naturally from the kinematic structure of the system. They can present themselves as subsets of degrees-of-freedom which determine different modes of motion such as translation and rotation or different scales of motion such as the arm and finger joints of a humanoid robot. Components of motion may also arise from distinct parts of the system. Multi-robot systems have a component for each individual robot. Humanoid robots have limbs that can be considered individually within the stability constraints. Motion planning for deformable objects has two coupled but distinctive components: planning for the motion of the object and planning of the shape of the object. Molecular complexes have also been observed to have components of motion [7].

In addition to the usual roadmap construction, the connection strategy identifies and caches components of motion. Local planning for the roadmap edges is done on *cspace* slices from an existing sample via subsamples at slice intersections to the new sample. We associate additional information with each sample describing the slices that were used to plan a connection to the sample. During the construction of the roadmap, a space of possible slice sequences is explored in order to find slice sequences that

match the robot's components of motion for an improvement in observed performance.

We report experimental results on robots with up to 150 degrees-of-freedom and multi-robot systems with up to 144 degrees-of-freedom.

II. Previous Related Work

The exponential nature of the motion planning problem has made addressing the dimensionality one of the recurring themes in motion planning research. Several planners are based on the idea of solving lower dimensional problems by searching a submanifold of the *cspace* or its approximation. Lozano-Pérez decouples the degrees-of-freedom of a manipulator by replacing the wrist and end-effector by a bounding volume and planning only for the first three links [8]. Six-dimensional planning is only done from the original start to its three-dimensional projection and from goal's three-dimensional projection to the original goal. He also proposes an approach of planning the path by successively adding a link and searching submanifolds of increasing dimensionality all the way to the DOF-dimensional *cspace*. Gupta and Zhu use similar sequential approach in their motion planner for manipulators [10]. They consider the links of the manipulator sequentially and use a numerical potential field for solving only a two-dimensional subproblem when adding a link. One of the first massively parallel motion planners used wrist-decoupling and rotational symmetry to compute 3-dimensional *cspace* representations for the arm and wrist degrees-of-freedom [9]. Donald's motion planning algorithm uses heuristic local experts which restrict the search on *cspace* obstacle boundaries and their intersections [11].

Barraquand and Ferbach present a variational approach to motion planning that uses dynamic programming on randomly selected ruled submanifolds to iteratively improve a possibly colliding path [12]. The dimensionality of the submanifolds is at most 4 in order to keep dynamic programming tractable. They solve problems with up to 16 DOF. McHenry's slice-based path planning algorithm uses a slice optimization process to produce a possibly non-planar, two-dimensional slice which minimizes a criterion combining terms for distance to the obstacles, joint limits, and continuity and smoothness of the slice [13]. Dynamic programming is used for searching a path on the slice.

The original PRM method is an extreme form of a dimensionality reduction method since the roadmap edges are searched on one-dimensional submanifolds. The various edge checking algorithms introduced for use with PRMs are essentially line search algorithms. Only few higher dimensional submanifold local planners have been proposed. The *rotate-at-S* and "*A* like*" local planners presented for rigid bodies decouple planning for translational and rotational degrees-of-freedom [14].

Motion planning for multi-robot systems can take advantage of the decoupling between the robots by planning for each robot independently and enforcing co-ordination between the robots by searching their velocity space for a path that avoids collisions [15]. The method may result in deadlock situations and is incomplete. Decoupling between robots can be used also in sampling-based planners. One can increase the probability of obtaining a collision-free *cspace* sample by moving and testing for collision one robot at a time [16][5].

The motion planner for free-climbing robot presented in [17] plans motions on sequences of constraint manifolds. However, the primary goal of the approach is to enable a two-stage search strategy rather than address the dimensionality.

III. The Connection Strategy

A. Overview

This section describes a connection strategy that develops the idea of submanifold search for use as a connection strategy in a PRM motion planner. The core idea is to have the connection strategy of a PRM planner to search not only a one-dimensional submanifold of the *cspace* or the DOF-dimensional *cspace* as proposed before, but a submanifold of a dimensionality up to d, the largest dimensionality the local planner can handle efficiently. The selection of the dimensionality and the active degrees-of-freedom during each phase of local planning become a point of injection of application specific knowledge into the planner or another search problem that must be solved while planning the path. Another novel feature of the connection strategy is that it performs randomized exploration of submanifold sequences while constructing the roadmap.

Unlike in most connection strategies, the paths between the samples of the global roadmap are not straight-line, but can have arbitrary shapes. Furthermore, each path is divided into segments, each segment restricted to some submanifold of the *cspace*. Each submanifold corresponds roughly to a component of motion of the system and each segment to a motion of that component. While connecting pairs of the global samples, the planner identifies sequences of prospective submanifolds of the *cspace* and places subsamples at their intersections. Pairs of subsamples are connected by building local trees of additional configurations sampled at the submanifold and straight-line paths between them.

B. Local Dimensionality Reduction

The two goals of restricting the planning of the roadmap paths to submanifolds are to cap the dimensionality of the search space that the local trees have to cover, and to take advantage of the system's components of motion. Taking advantage of the submanifold structure of the *cspace* in planning involves developing techniques for representing the submanifold structure and finding the submanifolds. The connection strategy introduced in this paper represents the submanifold structure as a set of hyperplanes. Let C denote the DOF-dimension *cspace* and C' a lower-dimensional submanifold of C. No reparametrization of the *cspace* is done here, but the submanifold is a slice through C and represented as a subset of the degrees-of-freedom of the robot. A *slicing sequence* is a list of non-intersecting subsets S_i (an ordered set

partition) of the degrees-of-freedom of the robot. A procedure used to generate a slicing sequence $(S_1, S_2, \ldots S_n)$ is called a *slicing strategy* and defines the slices C_k', $k=1...n$.

When given two global *cspace* samples $\boldsymbol{q}$ and $\boldsymbol{q}'$ from the roadmap, the local planner builds n path segments e_k on slices C_k' between subsamples $(\boldsymbol{q} = \boldsymbol{s}^1), \ldots, \boldsymbol{s}^k, \ldots, (\boldsymbol{s}^{n+1} = \boldsymbol{q}')$. The segments e_k between pairs of consequential subsamples $(\boldsymbol{s}^k, \boldsymbol{s}^{k+1})$ are local motions on slices C_k', which, when concatenated, produce a motion from $\boldsymbol{q}$ to $\boldsymbol{q}'$ and a roadmap edge. Thus, each produced path segment is a motion that changes a set S_k of degrees-of-freedom of the system from the values of the sample $\boldsymbol{q}$ to those of $\boldsymbol{q}'$. More formally, let q_i be the value of the ith degree-of-freedom or equally ith element in the *cspace* vector $\boldsymbol{q}$. Also, let $\boldsymbol{s}^k$, $1<k<n+1$ to be the kth subsample. Now, the elements s_j^k of $\boldsymbol{s}^k$ are

$$s_j^k = \begin{cases} q_j, j \notin \bigcup_{i=1}^{k-1} S_i \\ q'_j, j \in \bigcup_{i=1}^{k-1} S_i \end{cases} \quad (1)$$

C. Slicing Strategy

When planning for a known multi-robot system, an obvious slicing strategy is to plan for each individual robot or groups of robots sequentially. Each S_k is thus the set of degrees-of-freedom of the robot k in the system. While this strongly resembles decentralized multi-robot motion planning with the deadlocking problem, it should be noted that we use the approach for constructing edges for the roadmap, and, therefore, failures can be tolerated without compromising the probabilistic completeness.

For a more general algorithm one has to be able to represent a more varied set of components of motion. As will be discussed in the next section, preliminary experiments indicated that the obvious slicing strategy of taking C_k's as uniformly distributed random d-subsets of the degrees-of-freedom results in abysmal performance. Instead, our slicing strategy is to take cyclically $n = \lceil DOF/d \rceil$ sequential slices of C starting at some random degree-of-freedom and randomize the order of those slices. The slicing strategy produces slices of dimensionality d except one can be of dimensionality d or of dimensionality $(DOF \bmod d)$.

D. Slice Sequence Caching

The randomized slicing strategy used here samples the space of possible slice sequences. The connection strategy makes an effort to reuse good samples from that space. The slicing sequence that is successfully used to connect a newly generated global *cspace* sample to the roadmap is stored with the sample in the roadmap node. When connecting the newly generated sample to an existing close sample from the roadmap, the slice sequence stored with the existing sample is first used assuming that it would be successful again. If local planning with the reused slice sequence fails, a new one is generated according to the slicing strategy, and a new connection attempt is made with it. The motivation for caching and reusing slice sequences is to take advantage of the non-changing kinematic structure of the device and possible spatial coherence in the *cspace*, which can induce exploitable structure available in the form of repeatedly successful slicing sequences that match the components of motion of the system.

E. Building Local Trees

Any suitable motion planning technique restricted to the slice could be used to generate the motions between the subsamples. Since the components of motions of high-dimensional systems can be complex and require substantial planning, we use SBL [18]. It builds two trees of milestones rooted at the subsamples. Two trees are built at the time to construct the local path segment by segment. For determining the size of the trees built with SBL, the number of unsuccessful connection attempts to each global roadmap sample is counted and the maximum value of the two samples is taken and trees of up-to $1000+failures \times 100$ milestones are built. If the trees connect at some milestone, the path between the subsamples is retained from the trees and the other milestones are discarded. If all the segments can be produced in turn, the segments are concatenated and added to the global roadmap as an edge between the samples, otherwise the segments and subsamples are discarded. Pruning the local trees is necessary to keep them from exhausting the memory. The connectivity information is stored compactly in the global roadmap in the form of the complex paths extracted from the local trees. In the case of the test problems presented in the next section, the connectivity of the *cspace* is captured with a roadmap of at most few thousands of samples.

IV. Experimental Results

A. The Experimental Set-Up

We embed the strategy into a simple PRM planner and experiment with versions of the planner with connection strategies that either hard-code the known kinematic structure of a multi-robot system or search for the slice sequences. The testbed motion planner is a single query planner and initializes by inserting the start and goal configurations into the roadmap. The planner continues by adding uniformly sampled random configurations to the roadmap and connecting the samples to the previous samples with local paths until the start and goal configurations become connected, or a computational limit is exceeded. The connection is attempted to $k=20$ closest samples from each connected component of the roadmap. The metric used to select the closest samples is Euclidean distance. Based on preliminary experiments, the dimensionality of the slices for the tree-based planner we use is capped at $d=24$.

The connection strategy embedded in the testbed motion planner is assessed with two sets of problems. The first set of problems is comprised of multi-robot problems with obvious submanifold structure due to decoupling of the robots. The problems are geometrically easy having no significant narrow passages. Welding station problems WeldingStation and DoubleWeldingStation are similar to the ones used to

demonstrate SBL [18]. The relatively low-dimensional problem WeldingStation has 36 degrees-of-freedom, and a sample of 10000 uniform random configurations indicates that 42% of the *cspace* is collision-free. Combining two cells in close arrangement in DoubleWeldingStation makes a considerably more difficult problem with 72 degrees-of-freedom and 14% free *cspace*. A test problem with 144 DOFs and 13% free *cspace* is constructed by coupling 24 free-flying rigid bodies into one particle system (Particles). The start and goal configurations are randomly generated. A test problem comprised of six robots with 10 degrees-of-freedom each makes it possible to test sensitivity to the dimensionality of the submanifolds (MultiRobot). This problem has less than 1% of free *cspace*.

The second set of problems is more difficult. The test problem MultiRobotNarrow is a multi-robot problem with obvious decoupling of kinematic structures, but significant need for co-ordination of the motions due to the narrow passage. The problem has also less than 1% of free *cspace*. Three linear linkages of different length, Serpentine20, Serpentine40, and Serpentine75, are high-dimensional problems with no obvious submanifold structure. Each segment of the linkage has two degrees-of-freedom, one for rotation of the link around its base and one for rotation of the link along its longer axis. One end of the linkage is fixed in place. Lower dimensional problem Serpentine20 has 20 links and, therefore, 40 DOFs. It has 31% of free *cspace*. Serpentine40 has 40 links, 80 DOFs, and 7% of free *cspace*. The most difficult problem of the study, Serpentine75, has 75 links and 150 DOFs. Due to huge size of the *cspace*, it is difficult to accurately estimate the fraction of free *cspace* but it's less than 1%. For all serpentines, the task is to untangle the linkage into the fully extended configuration from the start configurations given in Fig. 6.

B. Results

We use the number of collision checks as the performance metric for three reasons: A collision check is know to be the most expensive single operation in PRM planners, the number of collision checks estimates the size of *cspace* explored during planning, and, due to the similarity of the algorithms compared here, it has a strong correlation with running time but is machine-independent.

The left section of Table I presents the average numbers of collision checks for the first set of test problems. With the exception of the easiest problem of the set, slice sequence caching provides a substantial performance improvement over slicing only and indicates that the planner is indeed capable of exploiting the existence of components of motion in the systems. The data presented in the right section of the table demonstrates that the planner is capable of solving very challenging problems also when no exploitable structure exists and slice sequence caching does not have significant detrimental effect in such cases.

Table II presents some results for SBL and a PRM planner with SBL as the local planner as described in the previous section but without slicing (PRM-SBL). Only the welding station problems can be solved consistently with these planners. On the other problems SBL exhausted the 1 GB RAM the workstation has, and PRM-SBL exceeded our time limit. The lower bounds on the planning cost in the table for the other problems are an order of magnitude larger than the averages over 30 successful runs of the slicing PRM planner. Table II also presents results for the deterministic slicing strategy of calling local planner sequentially for each sequential slice with dimensionality of 6 or 10.

C. The Effect of Slicing

The difference between the above PRM planners with and without slicing is that slicing makes the local tree-based planner to move only a subset of the degrees-of-freedom at each particular iteration of the tree construction. The effect of slicing can be directly investigated in an experiment by generating two close collision-free samples on a slice and testing the short line segment connecting them for collisions and observing the fraction of collision-free segments over a large number of segments. The experiment simulates the basic operation of tree-based planning, the extension of the tree from some sample with a new collision-free sample and a collision-free segment between them. The left panel of Fig. 7 presents a graph of the fraction of collision-free segments for the test problem Particles as a function of the number of changing degrees-of-freedom in the segment. It shows a sharp decrease in the fraction of collision-free segments as the number of changing degrees-of-freedom increase. The graph suggest that in addition to reducing the dimensionality of the problem, slicing also increases the probability of making progress by extending the tree with a collision-free sample and segment.

The right panel of Fig. 7 illustrates data from another of our experiments and helps to understand our decision to use consecutive degrees-of-freedom for slices. The four graphs in the right panel are similar experiments on the fraction of collision-free extensions from a collision-free sample. Let's call the number of consecutive degrees-of-freedom that are changed simultaneously the local block size of a slice. Each graph in the panel gives the fraction of collision-free extensions at some number of changing degrees-of-freedom, d, as a function of the local block size. Each graph extends only to the value of d, since the local block size cannot exceed the slice dimensionality. It can be observed that at each level of d, the larger the local block size, the larger the fraction of collision-free extensions. This is an empirical observation that is explained by the coupling between adjacent degrees-of-freedom of most physical kinematic systems, which tend to cause the components of motion to be comprised of degrees-of-freedom that are consecutive in the kinematic chain. This observation prompted us to use maximal local block size of d in the slicing strategy. The other test problems exhibit similar behavior in both experiments even if less pronounced.

D. Discussion

Comparing the proposed planner to SBL and a PRM with SBL as local planner indicates that slicing can provide a substantial performance improvement and make it possible to solve problems of unprecedented complexity. This is to be expected for problems which have obvious submanifold structure that the planner is designed to exploit. In such cases, the slice sequence caching provides additional performance improvement. However, if the solution cost of the problem is dominated by the cost of a path through a narrow passage, slice sequence caching is incapable of reducing the cost of finding the one-off slice sequence having the necessary sequence of subsets of the degrees-of-freedom for the co-ordinated motion through the passage. Unsurprisingly, best performance is observed when the knowledge of components of motion is hard-coded in the planner, but incorrect knowledge can have very detrimental effect. The serpentine problems demonstrate that the planner solves complex problems also when no obvious submanifold structure appears to be present.

We used SBL as a representative modern PRM planner but several others [1] could be used instead. Slicing can be expected to improve each of them, since it is rooted in the current theoretical understanding of PRM planners as argued in the introduction and demonstrated by the experiment investigating the effect of slicing. A deployable motion planner should incorporate adaptive sampling, e.g. [19], to take advantage of the non-uniform visibility properties of the *cspace*, and perhaps geometric transformations, e.g. [20], to globally improve the visibility.

V. Conclusions

The motion planner presented here can solve high-dimensional problems well beyond the capability of the established tree-based planner used as the baseline here. The improvement comes from storing the connectivity captured by the tree-based planner compactly in a global roadmap of complex paths thus avoiding exhausting the memory and performing tree-based local planning on a submanifold where the local planner has higher probability of extending the search front. As a PRM, the planner employs divide-and-conquer approach by using spatial subdivision of the problem to relatively inexpensive local motions. Furthermore, it divides the problem also dimension-wise by planning the local motions by subsets of the degrees-of-freedom and conquers by reusing sets that are successful.

For multi-robot planning one can combine aspects of centralized and decentralized approaches by using a slicing strategy that matches exactly the kinematics of the robots. For more a general planner, however, one must use a slicing strategy that can adapt to larger variety of kinematic structures. The slicing strategy presented in this paper relies on partial randomization to provide that adaptivity.

The algorithm presented here expends minimal amount of computation for slice generation and uses a very capable tree-based planner to search the slices for the local motions. This is in contrast to McHenry's slice-based path planner which spends most computation in optimizing the slice and uses relatively fast planner on the slice. Due to the curse of dimensionality, the cost of slice generation can be expected to increase with the dimensionality of the slice. Still, it is an interesting open question how to best balance the computational effort spent on slice generation, local planning, and roadmap construction.

References

[1] H. Choset et al., *Principles of Robot Motion: Theory, Algorithms, and Implementations*, Chapter 7, MIT Press, 2005.

[2] D. Hsu, L. E. Kavraki, J.-C. Latombe, and R. Motwani, "Capturing the connectivity of high-dimensional geometric spaces by parallelizable random sampling techniques," in P. M. Pardalos and S. Rajasekaran Eds., Advances in Randomized Parallel Computing, Boston: Kluwer Academic Publishers, 1999, pp. 159-182.

[3] A. M. Ladd and L. E Kavraki, "Measure theoretic analysis of probabilistic path planning," IEEE Trans. Rob. Aut., vol. 20, no. 2, pp. 229–242, April 2004.

[4] L. E. Kavraki, P. Švestka, J.-C. Latombe, M. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," IEEE Trans. Rob. Aut., vol. 12, no. 4, pp. 566-580, August 1996.

[5] M. Akinc, et al., "Probabilistic roadmaps of trees for parallel computation of multiple query roadmaps," Int. Symp. Rob. Res., Sienna, Italy, 2003.

[6] M. Sharir, "Algorithmic motion planning," in J.E. Goodman and J. O'Rourke, Eds., Handbook of Discrete and Computational Geometry, Boca Ratona: CRC Press, 1997, pp. 733-754.

[7] M. Teodoro, G. N. Phillips Jr., and L. E. Kavraki, "Singular value decomposition of protein conformational motions: Application to HIV-1 protease," in Currents in Computational Molecular Biology, Tokyo: Universal Academy Press Inc. , 2000, pp. 198–199.

[8] T. Lozano-Pérez, "A simple motion planning algorithm for general robot manipulators," IEEE J. Rob. Aut., vol. 3, no. 3, pp. 224 - 238, June 1987.

[9] T. Lozano-Pérez, and P. O'Donnell, "Parallel robot motion planning," IEEE Int. Conf. Rob. Aut., 1991, pp. 1000–1007.

[10] K. Gupta, and X. Zhu, "A novel approach to motion planning for many degrees of freedom manipulators: Numerical potential fields within sequential framework," IEEE Int. Conf. Rob. Aut., 1994, pp. 2038-2043.

[11] B.R. Donald, "A search algorithm for motion planning with six degrees of freedom," Artificial Intelligence, vol. 31, no. 3, pp. 295-353, March 1987.

[12] J. Barraquand, and P. Ferbach, "Path planning through variational dynamic programming," IEEE Int. Conf. Rob. Aut., 1994, pp. 1839-1846.

[13] M. C. McHenry, "Slice-based path planning," PhD dissertation, University of Southern California, Los Angeles, CA,1998.

[14] N. M. Amato, O. B. Bayazit, L. K. Dale, C. Jones, and D. Vallejo, "Choosing good distance metrics and local planners for probabilistic roadmap methods," IEEE Trans. Rob. Aut., vol. 16, no. 4, pp. 442 –447, August 2000.

[15] K. G. Kant, and S. W. Zucker, "Toward efficient trajectory planning: Path velocity decomposition," Int. J. Rob. Res., vol. 5, no. 3, pp. 72–89, Fall 1986.

[16] C. M. Clark, S. M. Rock, and J.C. Latombe, "Dynamic networks for motion planning in multi-robot space systems," Int. Symp. Artificial Intelligence, Robotics and Automation in Space, Nara, Japan, May 2003.

[17] T. Bretl, "Multi-step motion planning: Application to free-climbing robots," PhD Dissertation, Stanford University, Stanford, CA, 2005.

[18] G. Sanchez-Ante, and J.C. Latombe, "A single-query bi-directional probabilistic roadmap planner with lazy collision checking," Int. Symp. Rob. Res., 2003, pp. 403-417.

[19] D. Hsu, G. Sánchez-Ante, and Z. Sun, "Hybrid PRM sampling with a cost-Sensitive adaptive strategy. IEEE Int. Conf. Rob. Aut., 2005, pp. 3885–3891.

[20] M. Saha, and J.C Latombe, "Finding narrow passages with probabilistic roadmaps: The small-step retraction method," IEEE/RSJ Int. Conf. Int. Rob. Sys, 2005.

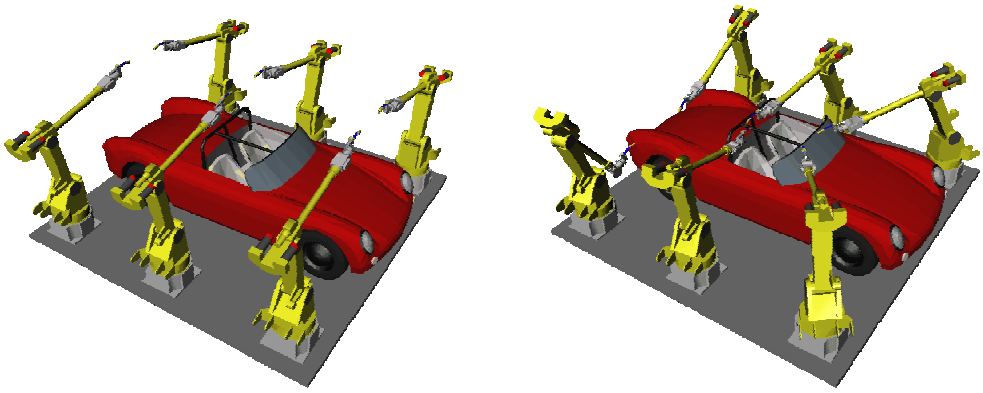

Fig. 1 Start and goal configurations for the test problem WeldingStation (36 DOF).

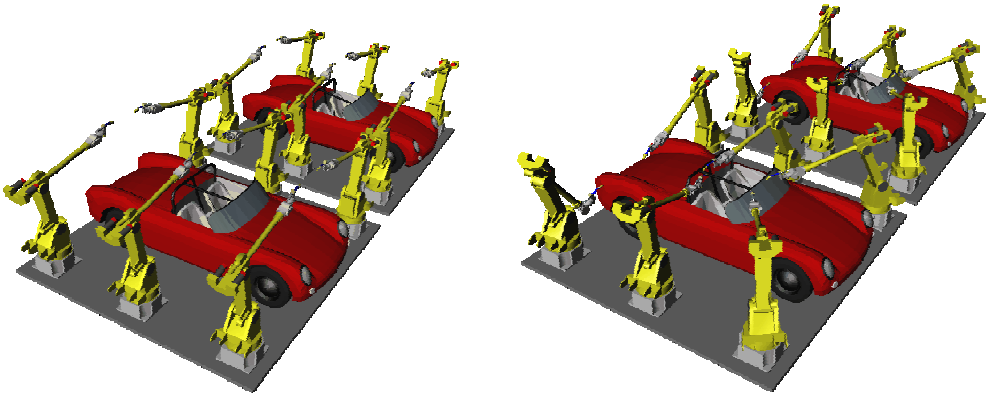

Fig. 2 Start and goal configurations for the test problem DoubleWeldingStation (72 DOF).

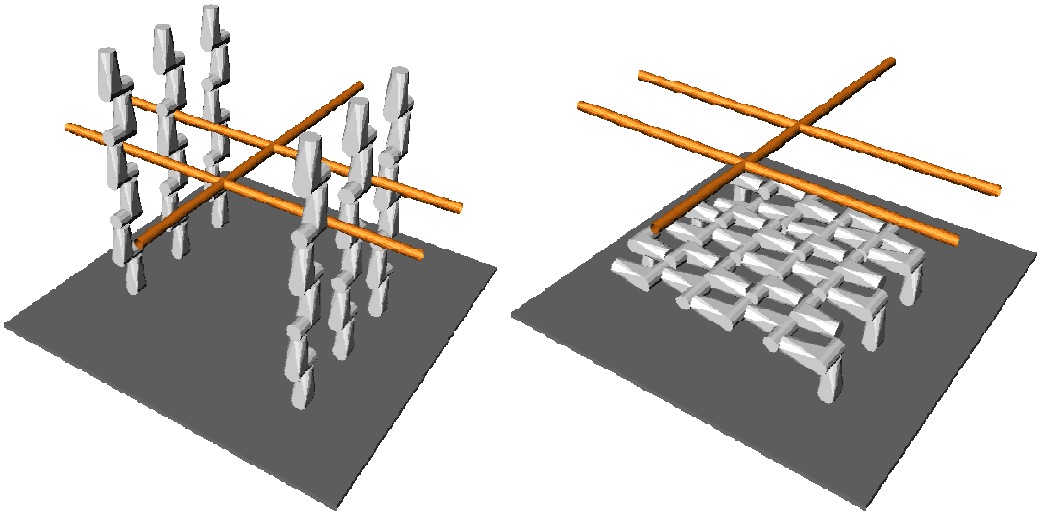

Fig. 3 Start and goal configurations for the test problem MultiRobot (60 DOF).

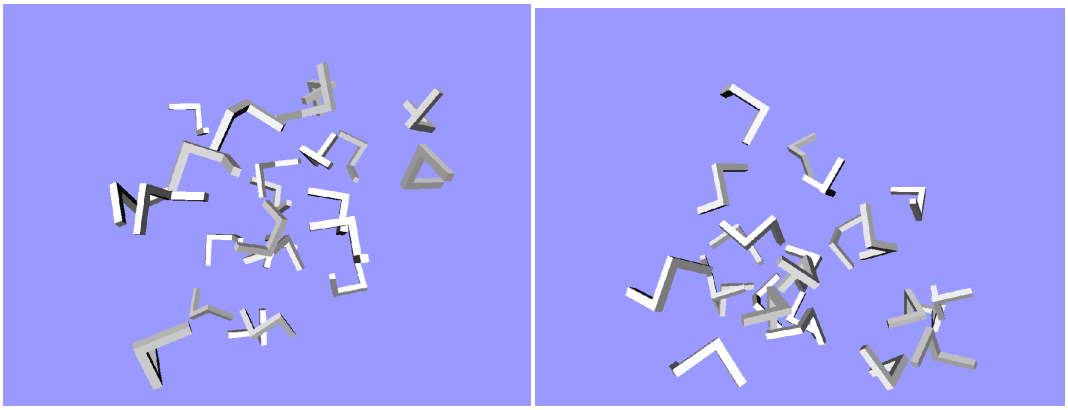

Fig. 4 Start and goal configurations for the test problem Particles (144 DOF).

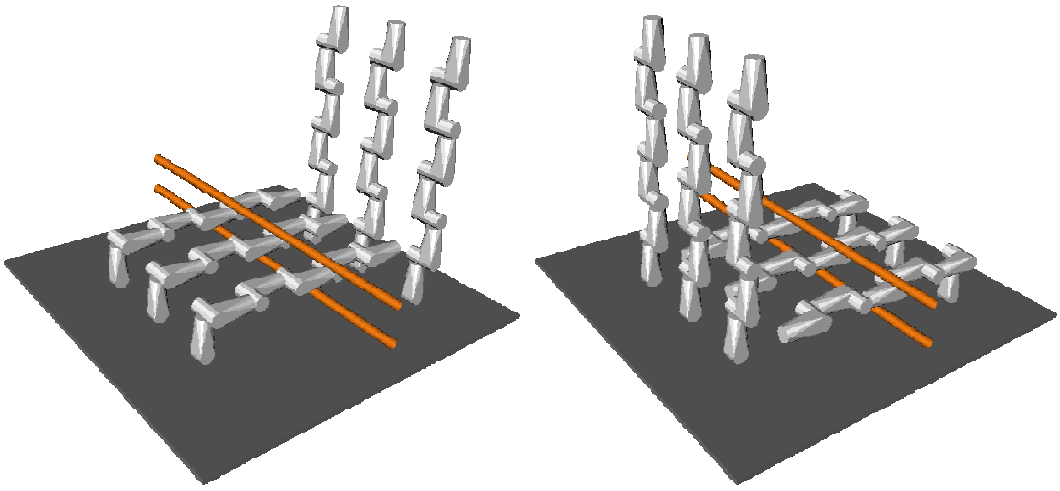

Fig. 5 Start and goal configurations for the test problem MultiRobotNarrow (60 DOF).

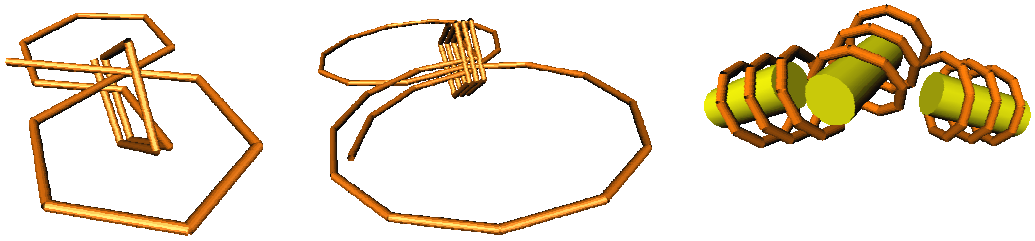

Fig. 6 Start configurations for the test problems Serpentine20 (40 DOF), Serpentine40 (80 DOF), and Serpentine75 (150 DOF).

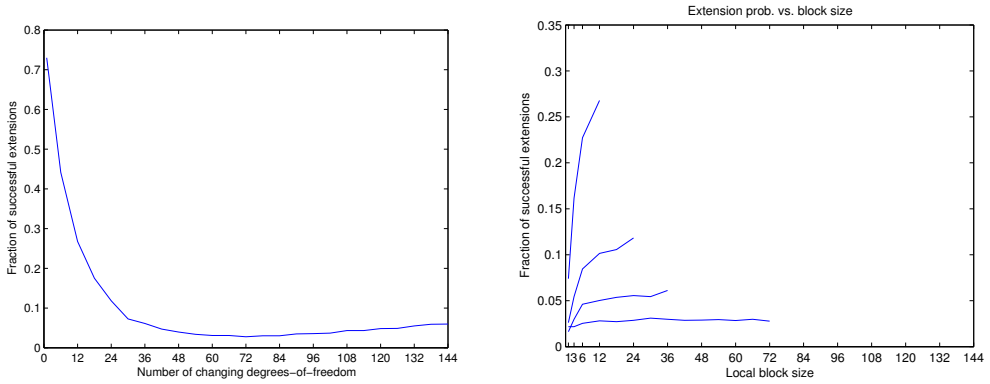

Fig. 7 The fraction of successful extensions from a collision-free sample.

TABLE I

AVERAGE NUMBERS OF COLLISION CHECKS FOR THE TEST PROBLEMS.

	Slicing only	With caching		Slicing only	With caching
WeldingStation	2,258	2,357	Serpentine20	148,038	134,490
DoubleWeldingStation	14,417	6,099	MultiRobotNarrow	5,287,722	6,156,049
MultiRobot	634,255	334,581	Serpentine40	82,329,112	58,467,443
Particles	654,960	314,354	Serpentine75	TBA	125,056,047

TABLE II

RESULTS FOR SBL, A PRM PLANNER WITH SBL AS A LOCAL PLANNER BUT WITHOUT SLICING, AND A PRM PLANNER WITH SBL AS A LOCAL PLANNER WITH DETERMINISTIC SLICING WITH SLICE SIZES OF 6 AND 10. A DATA ENTRY IS EITHER THE AVERAGE NUMBER OF COLLISION CHECKS OVER 30 RUNS, LOWER BOUND FOR THE AVERAGE FROM MINIMUM OF THE SUCCESSFUL RUNS, OR MINIMUM NUMBER OF COLLISION CHECKS PERFORMED IN PARENTHESIS IF ALL THE 30 RUNS FAILED.

	SBL	PRM-SBL	6	10
WeldingStation	1,659	3,362	686	4,233
DoubleWeldingStation	31,329	107,994	1,626	8,180
MultiRobot	>2,248,232	(12,334,299)	>845,429	37,843
MultiRobotNarrow	(5,003,962)	(13,323,631)		1,125,080
Serpentine20	>2,459,774	>1,739,484	(1,448,126)	(2,264,668)
Particles	(1,005,859)	(7,125,272)	50,976	297,525
Serpentine40		(1,679,880)		