Proceedings of the 2002 IEEE
International Conference on Robotics & Automation
Washington, DC • May 2002

Using a PRM Planner to Compare Centralized and Decoupled Planning for Multi-Robot Systems

Gildardo Sánchez
Computer Science Department
ITESM, Campus Cuernavaca
Cuernavaca, México

Jean-Claude Latombe
Computer Science Department
Stanford University
Stanford, CA, USA

Abstract

This paper describes experiments with a Probabilistic Roadmap Planner on a spot-welding station with 2 to 6 robot manipulators combining 12 to 36 degrees of freedom. When performing centralized planning, the planner has proven to be reliable and fast. When performing decoupled planning, it was not significantly faster, but it was much less reliable, failing to find a solution 30 to 75% of the times in 6-robot examples. This is an important result as it invalidates the assumption that the loss of completeness in performing decoupled planning is not very substantial in practice and indicates that centralized planning is a more desirable approach – at least in applications like spot-welding, which requires rather tight robot coordination.

1 Introduction

Probabilistic roadmaps (PRM) are an effective tool to capture the connectivity of a robot's collision-free space and solve path-planning problems with many degrees of freedom (dofs) [3, 11, 13, 14]. An especially interesting application for PRM planners is to compute coordinated paths of manipulator robots in spot-welding stations such as those found in automotive body shops (Figure 1). Such stations include 4 to 10 robots, each sharing workspace with 1 to 4 other robots. The number of robot degrees of freedom ranges from 20 to 60. This does not account for the dofs of the clamping devices, which may add several dozen dofs. The CAD model of the environment may consist of several 100,000 triangles. The task of manually programming a station using a graphic simulator and/or teach pendants is long and prone to errors. Late adjustments in weld points (these changes may be motivated by data collected during crash-tests) result in interruptions of the manufacturing line. A fast and reliable planner could greatly expedite the re-programming task and shorten interruptions.

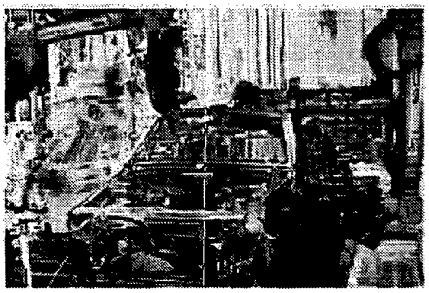


Figure 1: Spot-welding station.

There are two well established approaches to multi-robot motion planning, independent of the core path planning technique used: centralized and decoupled [16]. So far, the prevalent approach has been decoupled planning. In most cases, centralized planning has been beyond the practical capabilities of existing core planning techniques, as it requires searching configuration spaces with many dimensions. Decoupled planning breaks the original planning problem into more tractable sub-problems. Though decoupled planning is inherently incomplete – that is, it is not guaranteed to find a solution whenever one exists – it has been assumed that, in practice, the loss of completeness is relatively small and worth the computational gain.

We have recently developed a new PRM planner that combines a single-query bi-directional sampling strategy with a lazy collision-checking connection strategy. The later postpones collision tests until they are absolutely necessary. This planner, called SBL (for Single-query, Bi-directional, Lazy in checking collision), is described in [19, 20]. It is significantly more efficient than previous PRM planners, which enables its application to problems involving multiple robots interacting in geometrically complex environments. Here, we describe the use of SBL to implement both the

0-7803-7272-7/02/$17.00 © 2002 IEEE

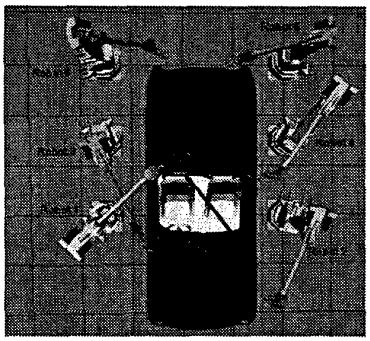


Figure 2: Model of six-robot welding station.

centralized and the decoupled approaches. We give experimental results obtained for the model of a 6-robot welding station shown in Figure 2, comparing the performance and reliability of both approaches. The results reveal that decoupled planning is too unreliable to be practical for this application. This is an important observation, since it invalidates the assumption that the loss of completeness caused by decoupled planning is not very significant in practice and indicates that centralized planning is a more desirable approach. By no means, however, does this imply that decoupled planning is useless. First, it may be reasonably reliable for other applications where interactions among robots are less constraining. Second, there are distributed robot systems where centralized planning is not possible because no robot or processor knows the global state of the system or the goals of all robots. Finally, even in cases where decoupled planning is possible but unreliable, it may still be useful if the planner receives interactive hints from the user.

Section 2 presents the SBL planner. Section 3 describes the centralized and decoupled approaches to multi-robot planning and their implementation using SBL. Section 4 discusses experimental results comparing the two approaches.

2 SBL Planner

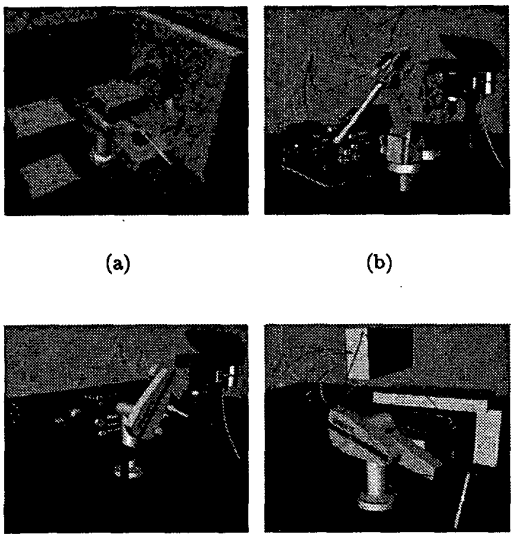
Overview. The PRM approach was proposed to plan collision-free paths for robots with "many" – 6 or more – dofs [13, 14]. A PRM planner operates as follows. It samples the robot's configuration space at random, tests each sample for collision, and retains the collision-free samples as *milestones*. It connects pairs of milestones that are relatively close apart by simple paths (typically, straight line segments in configuration space) and retains the collision-free ones as *local paths*. The milestones and local paths form the *probabilistic roadmap*. The planner connects the initial and goal configurations of the robot to this roadmap and invokes a search algorithm to extract a collision-free path between them, if there exists one.

The motivation for this approach is that, while it is often impractical to compute an explicit geometric representation of the collision-free subset (the *free space*) of a configuration space, algorithms exist to efficiently check whether a given configuration or local path is collision-free [3]. In particular, hierarchical algorithms pre-compute a multi-level bounding approximation of every object in an environment (e.g., [6, 8, 18]). At each collision query, they use the pre-computed approximations to quickly discard large subsets of the objects that cannot possibly collide. They scale up well to environments where object surfaces are described by 100,000's of triangles [10].

A PRM planner cannot usually recognize that no collision-free path exists. So, it returns that no path exists if it has not found one after some specified computational time. But it has been shown that, under broad assumptions, the probability that a PRM planner finds a collision-free path, if one exists, goes to 1 exponentially in the number of milestones (which in practice is roughly proportional to the running time) [10, 9]. Said otherwise, if the planner returns that no path exists, the probability that this outcome is incorrect converges quickly to 0. This property is called *probabilistic completeness*.

Sampling strategies. PRM planners differ among them mainly by their sampling strategies. Some planners – called *multi-query* planners – pre-compute a roadmap that they later use to answer multiple path-planning queries [13, 14]. Others – *single-query planners* – compute a new roadmap for each query [10, 15]. Multi-query planners are appropriate when the relatively high cost of pre-computing a roadmap can be amortized over many queries. Single-query planners are more suitable when the queries are few.

A single-query planner uses the query configurations to explore restricted subsets of the free-space components that are reachable from these configurations. This is done either by growing one tree of milestones rooted at one query configuration, until a connection is found with the other query configuration (*single-directional* sampling), or by growing two trees concurrently, respectively rooted at one of the two query configurations, until a connection is found between the two trees (*bi-directional* sampling) [9]. In both cases, milestones are iteratively added to the roadmap. Each



(a) (b)

(c) (d)

Figure 3: Path planning environments.

	Running time (s)	Milestones in roadmap	Total Nr. of CC	Std. Dev.
a	4.45	1609	11211	2.48
b	4.42	1405	7267	1.86
c	0.17	33	406	0.07
d	6.99	4160	12228	3.55

Table 1: Statistics for environments in Figure 3

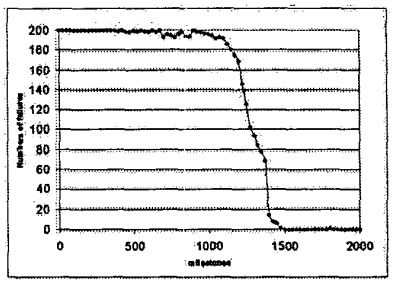


Figure 4: Experimental rate of convergence of SBL.

new milestone m' is selected in a neighborhood of a milestone m already in a tree and is connected to m by a local path (hence, m' becomes a child of m). Bi-directional planners are usually more efficient than single-directional ones.

SBL planner. Previous PRM planners test each connection between milestones for collision before inserting it in a roadmap. It was shown in [5] that a *lazy collision-checking* strategy that postpones tests until they are absolutely needed save considerable time. We have refined this strategy and integrated it with a single-query bi-directional sampling strategy, yielding the SBL planner [19, 20]. SBL uses the PQP collision checker described in [8]. It also includes a simple path optimizer that shortens the generated paths.

Our experiments showed that SBL is between 4 and 40 times faster than a classical single-query bi-directional PRM planner. Figure 3 presents four single-robot examples on which we have run the planner. The path of the end-effector before optimization is shown in red; the path after optimization is in yellow. Table 1 lists performance measures of the planner; for each example, these measures are averages over 100 runs with different seeds of the random-number generator. The longer running times in the geometrically simpler examples *a* and *d* result from the fact that in these two examples the obstacles create "narrow passages" in free space. The running times do not include path optimization, which takes a small fraction of the planning time.

What is a failure? The results of Table 1 are based on a total of 500 runs of SBL. The maximal number of milestones that SBL was allowed to generate in a single run was set to 50,000. Had this number been reached during a run, the planner would have reported failure (i.e., that no path exists) for this run. Not only SBL did not fail in any of the 500 runs, but the number of milestones actually needed in each run was much smaller than 50,000. This is a good indication of the planner's reliability. But since the notion of a failure will be important later, a discussion is appropriate here.

The diagram in Figure 4 was established for the example of Figure 3(b) by running SBL with increasing values of the maximal number of milestones (horizontal axis), from very small ones to larger ones. For each maximal number S of milestones, we ran the planner 200 times with different seeds, and we counted the number of failures (vertical axis). The curve indicates that for a given problem there exist two thresholds S_{min} and S_{max}. If $S < S_{min}$, the planner fails consistently; if $S > S_{max}$, it succeeds consistently; if $S_{min} < S < S_{max}$, it has mixed successes and failures. In Figure 4, S_{min} and S_{max} are roughly equal to 700 and 1500, respectively. Similar diagrams were obtained with other examples. They are coherent with the theoretical result that a PRM planner has a fast

convergence rate in the number of milestones.

This leads to the following conclusion. Suppose that we consider a new planning problem that is too complex for us to know in advance if it admits a solution, or not. If we run SBL on this problem several times and the planner never finds a solution path, then either there exists no such path, or $S < S_{min}$ (for the S_{min} of this problem, which is unknown). If the planner succeeds at least once, then we know that there exists a solution path. Then, by increasing the maximal number of milestones, we can easily achieve a 100% success rate. This reasoning will be critical to properly analyze the experimental results of Section 4.

3 Multi-Robot Planning

We now turn to the application of SBL to multi-robot systems. An initial and a goal configuration are input for each robot. A coordinated path is one that indicates the configuration of every robot at each instant. Collisions must be avoided between each pair of robot and workspace obstacle, and between each pair of robots.

Centralized planning. It consists of considering all the robots as if they were forming a single multi-arm robot, by encoding their dofs in a single "composite" configuration space $\mathcal{C}$ and searching that space for a free path between the initial and goal configurations. $\mathcal{C} = \mathcal{C}_1 \times \mathcal{C}_2 \times ... \times \mathcal{C}_p$, where p is the number of robots and $\mathcal{C}_i$ is the configuration space of the ith robot ($i \in [1,p]$). Thus, the number of dimensions of $\mathcal{C}$ is equal to the total number of dofs of the robots. In the example of Figure 2, each robot has 6 dofs and $\mathcal{C}$ has 36 dimensions.

Let $\tau : s \in [0,1] \mapsto \tau(s) \in \mathcal{F}$ be a path in the free subset $\mathcal{F}$ of $\mathcal{C}$. The projection τ_i of τ into $\mathcal{C}_i$ is the path to be followed by the ith robot. For each $s \in [0,1]$, $\tau(s)$ is of the form $(\tau_1(s), \tau_2(s), ..., \tau_p(s))$, which describes the configurations of the p robots at a single point along the path τ. Hence, a path in $\mathcal{F}$, if one exists, not only describes the individual path to be followed by each robot, but also how the robots are to be coordinated.

In principle, any sufficiently general path-planning algorithm can be used to implement centralized planning, by applying this algorithm to the composite space $\mathcal{C}$. But, in the past, centralized planning has not been considered practical because it usually leads to searching large-dimensional configuration spaces that are beyond the practical capabilities of existing planning techniques. Most proposed centralized planners have been based on incomplete heuristics, for example potential field techniques [4, 23, 24]. Complete algorithms have only been proposed only for very simple robotic systems, e.g., the coordination of two discs among polygonal obstacles [21]. PRM-based centralized planning has recently been applied to disassembly planning [22].

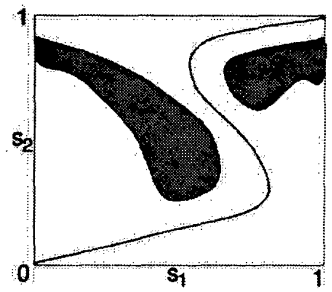
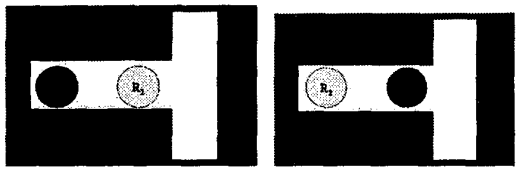


Figure 5: Coordination space of two robots.

Decoupled planning. This is a two-phase approach. In the first phase, a collision-free path is generated for each robot by considering only the obstacles in the environment and ignoring the other robots. In the second phase, called *velocity tuning*, the relative velocities of the robots along their respective paths are selected to avoid collision among them [2, 12, 17]. Variants of this scheme have been proposed [1, 7].

Velocity tuning consists of searching a *coordination space*. Consider two robots, and let τ_1 and τ_2 be their respective paths computed in the first phase. By forcing the robots to move along these paths, we reduce the number of dofs of each robot to 1, hence the dimension of their composite configuration space – now called the coordination space – to 2. Let each path $\tau_i (i = 1, 2)$ be parameterized by some $s_i \in [0,1]$. The set $P = [0,1]^2$ represents the coordination space of the two robots (Figure 5). Each point $(s_1, s_2) \in P$ defines a placement of the robots at their configurations $\tau_1(s_1)$ and $\tau_2(s_2)$. This point is collision-free if at this placement the two robots do not collide with each other. A path joining the point $(0,0)$ - where both robots are at their initial configurations – to $(1,1)$ – where they are at their goal configurations – in the collision-free subset of P defines a valid coordination of the robots along τ_1 and τ_2. This path may not be monotonic along any of the dimensions of P, meaning that for a while the ith robot may move backward along τ_i, e.g., to provide maneuvering space to the second robot. An unfortunate choice of τ_1 and τ_2 in the first planning phase may lead the points $(0,0)$ and $(1,1)$ to lie in two distinct connected components of the free subset



(a) Initial configuration (b) Goal configuration

Figure 6: A "bad" example for decoupled planning.

of P. Figure 6 shows a classical example, in which the two robots are discs that must switch positions, where decoupled planning is most likely to fail.

If there are $p > 2$ robots, one may coordinate all the robots by generating a collision-free path in the p-dimensional space P where the ith axis encodes the parameter s_i of the path of the ith robot, from the point $(0, ..., 0)$ to the point $(1, ..., 1)$. We call this approach to velocity tuning *global coordination*. An alternative, *pairwise coordination*, consists of planning $p-1$ paths in $p-1$ two-dimensional spaces $P_2, ..., P_p$. The axes of P_2 encode the parameters s_1 and s_2 along the paths of the 1st and 2nd robots, and a collision-free path $\tau_{1,2} : s_{1,2} \in [0,1] \mapsto \tau_{1,2}(s_{1,2}) \in P_2$ defines a valid coordination of these two robots. One axis of P_3 encodes the parameter s_3 along the path of the 3rd robot, while the other axis encodes the parameter $s_{1,2}$ along the coordinated path of the 1st and 2nd robots. Hence, each point in P_3 determines a placement of the first three robots, and a collision-free path in P_3 defines a valid coordination of these robots. Etc.

Decoupled planning leads to searching lower-dimensional spaces than centralized planning. But it is inherently incomplete, even if the core planning algorithms used in the first and second phases are complete. Velocity tuning may fail because the paths generated in the first phase cannot be coordinated without collision between robots, while such coordination would have been possible had other paths been selected. A decoupled planner based on global coordination is less incomplete than one based on pairwise coordination, since a specific path selected in the path space P_i may result into a space P_{i+1} with no collision-free path between $(0, ..., 0)$ and $(1, ..., 1)$. Nevertheless, pairwise coordination has been more widely used than global coordination.

Implemented planners. We have developed three multi-robot planners respectively based on centralized planning, decoupled planning with global coordination, and decoupled planning with pairwise coordination. They all use SBL as the core planner. We call these planners C-SBL, DG-SBL, and DP-SBL.

C-SBL is exactly SBL running in the composite configuration space $\mathcal{C}$ of the robots. The collision test of a point in $\mathcal{C}$ is done by calling the collision checker on every pair consisting of a rigid body of a robot and an environment obstacle, and on every pair of rigid bodies belonging to two distinct robots.

DG-SBL makes $p+1$ calls to SBL, where p is the number of robots: p calls to plan the path of each robot (ignoring the other robots); then, another call to plan a collision-free path in the space $P = [0,1]^p$. The roadmap in P is built by growing two trees of milestones respectively rooted at $(0, ..., 0)$ and $(1, ..., 1)$. The collision test of a point in P is done by calling the checker on every pair of rigid bodies belonging to two distinct robots.

DP-SBL makes $2p-1$ calls to SBL: p calls to plan the path of each robot; then, $p-1$ calls to plan paths in the spaces P_2 through P_p. The collision test of a point in P_i is done by calling the checker on every pair of bodies consisting of one rigid body of the ith robot and one rigid body of one of the robots 1 to $i-1$.

All three planners use the same optimizer. In DG-SBL and DP-SBL, this optimizer is called after the robot paths have been coordinated. Indeed, optimizing individual paths before they are coordinated would reduce the flexibility left for coordinating them, and hence increase the incompleteness of the planners.

4 Experimental results

Experimental setting. We have conducted experiments with the three planners in the environment of Figure 2. The models contain 5,000 triangles for each robot and 21,000 triangles for the car. Figures 7–9 show the initial and goal configurations for three problems (identified as problems I, II, and III). The robots are identified as robots 1, 2, ..., 6, in the figures. Problem I for 2 (or 4) robots is defined as shown in Figure 7, but restricted to robots 1 and 2 (or 1, 2, 3, and 4). Problems II and III for 2 and 4 robots are defined in the same ways.

To keep our planners general, we did not take advantage of certain obvious properties of the environment. For example, the planners assume that collisions may occur between any two bodies of any two robots, while it is clear that many pairs of bodies cannot collide. So, in C-SBL and DG-SBL, since each robot consists of 6 rigid bodies, testing that no two robots collide

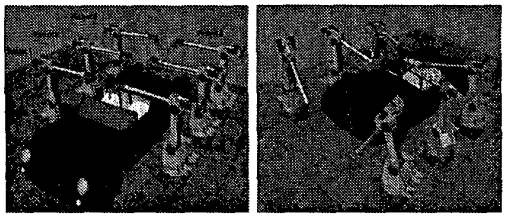


Figure 7: Initial and goal configurations, problem I.

Figure 8: Initial and goal configurations, problem II.

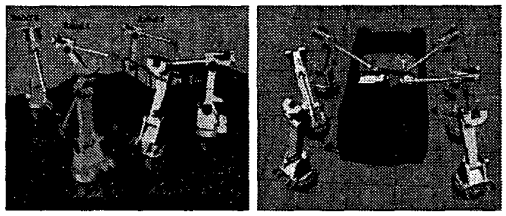


Figure 9: Initial and goal configurations, problem III.

	Running time (s)	Milestones in roadmap	Std. Dev.
PI-2 Robs	0.26	11	0.52
PII-2 Robs	0.25	11	0.17
PIII-2 Robs	2.44	191	1.57
PI-4 Robs	3.97	62	5.67
PII-4 Robs	3.94	56	2.40
PIII-4 Robs	30.82	841	15.55
PI-6 Robs	28.91	322	28.91
PII-6 Robs	59.65	882	31.08
PIII-6 Robs	442.85	5648	170.46

Table 2: Average data for centralized planner.

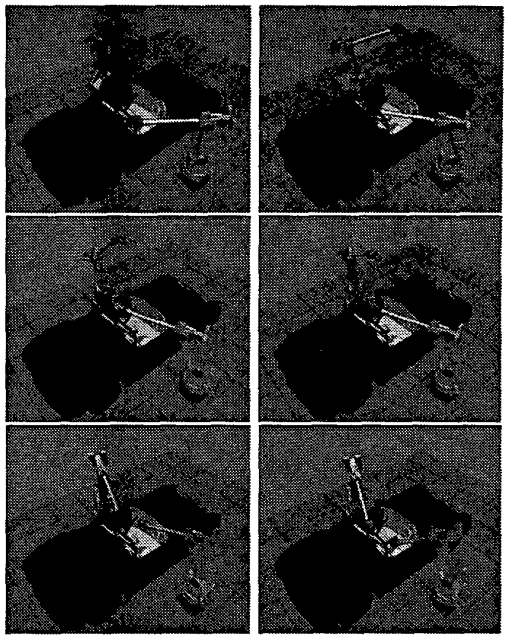


Figure 10: Path computed by C-SBL.

at a given configuration requires running the checker on 540 pairs of bodies (for DP-SBL the number is smaller).

Evaluation of C-SBL. Table 2 shows average data collected over 100 runs of C-SBL on problems I, II, and III, with 2, 4, and 6 robots. In all $100 \times 3 \times 3 = 900$ runs, C-SBL successfully returned a path in a satisfactory amount of time. In all runs, the planner generated much fewer milestones than the maximum allowed (50,000). The largest number of milestones, which was achieved for a run on problem III-6, was 6917. Figure 10 shows a series of snapshots along a path computed for problem III with two robots.

The increase in the running times when the number of robots grows is caused by both the quadratic increase in the number of pairs of bodies that may have to be tested at each collision check and the greater difficulty of the problems due to the constraints imposed by the additional robots upon the motions of the other robots. The time per collision check increases from 0.001 seconds for 2 robots, to less than 0.004 seconds for 4 robots, to about 0.0085 seconds for 6 robots. So, it increases more slowly than the number pairs of bodies tested. The reason is that each collision check involves testing each robot body against the environment and testing every pair of bodies from two different robots against each other. On average, the

second type of test is less costly than the first. Indeed, the environment contains more triangles than a robot body. Moreover, many pairs of robot bodies are far apart, especially when the number of robots increases. This allows the hierarchical checker to rapidly decide that they cannot possibly collide. While the number of body-environment tests is linear in the number of robots, the number of body-body tests is quadratic in that number.

Comparison of C-SBL, DG-SBL, and DP-SBL. Unlile C-SBL, the two decoupled planners failed a number of times. Table 3 lists the average running times in seconds (T) and numbers of failures (F) of DG-SBL and DP-SBL over 20 runs on each of the 9 problems. The average times are computed only over the successful runs. For comparison, we also include the averages for C-SBL (established over 100 runs). The maximal number of milestones allowed at each invocation of SBL was 50,000. This means that DG-SBL and DP-SBL had up to 50,000 milestones to generate each individual robot path. DG-SBL (global coordination) had up to 50,000 milestones to coordinate the robot paths. DP-SBL (pairwise coordination) had up to 50,000 milestones to coordinate each pair of paths.

The rate of failure of each decoupled planner is relatively small for the three problems with 2 robots, but it increases sharply as the number of robots grows to 4 and 6. For 6 robots, it ranges between 30% and 75%, with pairwise coordination being more unreliable than global coordination. A finer analysis showed that DG-SBL and DP-SBL never failed to generate individual robot paths. All their failures happened when velocity tuning in a coordination space returned failure after generating 50,000 milestones. In every successful run of a decoupled planner, the number of milestones in each of the roadmaps built by this planner was much smaller than 50,000 (at least 10 times smaller), indicating that with very high probability (recall Figure 4) the planner's failures in other runs were caused by the incompleteness of decoupled planning, not by SBL's probabilistic completeness.

In the most difficult problem (III-6), DG-SBL and DP-SBL failed 13 and 17 times, respectively, out of 20 runs. These numbers indicate that in a problem as complex as III-6, relatively few individual paths can be coordinated. We are not aware of any technique that would allow a decoupled planner to reliably find them in the first planning phase.

5 Conclusion

This paper describes the application of a PRM planner, SBL, to plan collision-free paths for several interacting robots. It introduces three planners that respectively implement centralized planning, decoupled planning with global coordination, and decoupled planning with pairwise coordination. These implementations were made possible by the fact that SBL efficiently handles many degrees of freedom.

We experimented with these planners on a multi-robot welding station, with 2, 4, and 6 robots. We made the important observation that decoupled planning is substantially incomplete for this type of application, which requires tight robot coordination. The rate of failures of the decoupled planners for problems with 6 robots ranged from 30 to 75%, which makes them of little practical value. On the other hand, the decoupled planners were at best only marginally faster than the centralized planner when they were successful. This observation invalidates the prevailing assumption that the loss of completeness in performing decoupled planning is not very significant in practice. It indicates that centralized planning may be a more desirable approach, and that the existence of efficient PRM planners such as SBL makes this approach technically feasible. Clearly, there are problems where centralized planning is not suitable, or even impossible - e.g., problems where multiple mobile robots with onboard computing and limited communication channels move in the same environment. For such problems, decoupled planning may remain the preferred approach.

Acknowledgments

This research was funded by grants from General Motors Research and ABB. G. Sánchez's stay at Stanford was partially supported by ITESM (Campus Guadalajara) and a fellowship from CONACyT. This paper benefited from discussions with H. González, C. Guestrin, D. Hsu, L. Kavraki, and F. Prinz. PQP was made available to us by S. Gottschalk, M. Lin, and D. Manocha from the CS Dept. at UNC.

References

[1] R. Alami, F. Robert, F.F. Ingrand, and S. Suzuki. Multi-robot cooperation through incremental plan-merging. In *Proc. IEEE Int. Conf. Rob. & Autom.*, pages 2573–2578, 1995.

[2] B. Aronov, M. de Berg, A. F. Van der Stappen, P. Švestka, and J. Vleugels. Motion planning for multiple robots. *Discrete and Computational Geometry*, 22:505–525, 1999.

Planner	Problem I						Problem II						Problem III					
	2 Robs		4 Robs		6 Robs		2 Robs		4 Robs		6 Robs		2 Robs		4 Robs		6 Robs	
	T	F	T	F	T	F	T	F	T	F	T	F	T	F	T	F	T	F
C-SBL	0.26	0	3.97	0	28.91	0	0.25	0	3.94	0	59.65	0	2.44	0	30.81	0	442.85	0
DG-SBL	0.22	1	2.74	3	29.53	7	0.37	2	6.59	4	65.45	6	4.32	5	16.23	6	267.81	13
GP-SBL	0.30	3	4.85	5	19.23	9	0.42	3	5.63	7	28.92	6	3.42	9	25.35	13	182.63	17

Table 3: Comparison of centralized and decoupled planners.

[3] J. Barraquand, L. E. Kavraki, J. C. Latombe, T. Y. Li, R. Motwani, and P. Raghavan. A random sampling scheme for path planning. *Int. J. of Robotics Research*, 16(6):759–774, 1997.

[4] J. Barraquand, B. Langlois, and J. C. Latombe. Numerical potential field techniques for robot path planning. *IEEE Trans. on Systems, Man and Cybernetics*, 22(2):224–241, 1992.

[5] R. Bohlin and L. E. Kavraki. Path planning using lazy PRM. In *Proc. IEEE Int. Conf. Rob. & Autom.*, pages 521–528, 2000.

[6] J. Cohen, M. Lin, D. Manocha, and M. Ponamgi. I-Collide: An interactive and exact collision detection system for large scale environments. In *Proc. ACM Interactive 3D Graphics Conf.*, pages 189–196, 1995.

[7] M. Erdmann and R. Lozano-Pérez. On multiple moving objects. In *Proc. IEEE Int. Conf. Rob. & Autom.*, pages 1419–1424, 1986.

[8] S. Gottschalk and M. Lin. OOBTree: A hierarchical structure for rapid interference detection. In *Proc. SIGGRAPH 96*, pages 171–180, 1996.

[9] D. Hsu. *Randomized Single-Query Motion Planning in Expansive Spaces.* PhD thesis, Stanford University, Stanford, CA, USA, 2000.

[10] D. Hsu, J. C. Latombe, and R. Motwani. Path planning in expansive configuration spaces. In *Proc. IEEE Int. Conf. Rob. & Autom.*, pages 2719–2726, 1997.

[11] D. Hsu, J. C. Latombe, R. Motwani, and L. E. Kavraki. Capturing the Connectivity of High-Dimensional Geometric Spaces by Parallelizable Random Sampling Techniques. In *Advances in Randomized Parallel Computing*, P.M. Pardalos and S. Rajasekaran (eds.), Combinatorial Optimization Series, Kluwer, Boston, MA, pages 159-182, 1999.

[12] K. G. Kant and S.W. Zucker. Toward efficient trajectory planning: Path velocity decomposition. *Int. J. of Robotics Research*, 5:72–89, 1986.

[13] L. E. Kavraki. *Random Networks in Configuration Space for Fast Path Planning.* PhD thesis, Stanford University, Stanford, CA, USA, 1994.

[14] L. E. Kavraki, P. Švestka, J. C. Latombe, and M. H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Trans. on Robotics and Automation*, 12(4):566–580, 1996.

[15] J.J. Kuffner, Jr. *Autonomous Agents for Real-Time Animation.* PhD thesis, Stanford University, Stanford, CA, USA, 1999.

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

[17] P. A. O'Donnell and T. Lozano-Pérez. Deadlock-free and collision-free coordination of two robot manipulators. *Proc. IEEE Int. Conf. Rob. & Autom.*, pages 484–489, 1989.

[18] S. Quinlan. Efficient distance computation between non-convex objects. In *Proc. IEEE Int. Conf. Rob. & Autom.*, pages 3324–3329, 1994.

[19] G. Sánchez-Ante. *Single-Query Bi-Directional Motion Planning with Lazy Collision Checking.* PhD thesis, ITESM-Campus Cuernavaca, México, 2002.

[20] G. Sánchez-Ante and J. C. Latombe. On delaying collision-checking in PRM planning – application to multi-robot coordination. *Int. J. of Robotics Research*, to appear, 2002.

[21] J. T. Schwartz and M. Sharir. On the piano movers' problem III: Coordinating the motion of several independent bodies: The special case of circular bodies moving amidst polygonal obstacles. *Int. J. of Robotics Research*, 2(3):46–75, 1983.

[22] S. Sundaram, I. Remmler, and N.M. Amato. Disassembly sequencing using a motion planning approach. In *Proc. IEEE Int. Conf. Rob. & Autom.*, pages 1475–1480, 2001.

[23] P. Tournassoud. A strategy for obstacle avoidance and its applications to multi-robot systems. In *Proc. IEEE Int. Conf. Rob. & Autom.*, pages 1224–1229, 1986.

[24] C. W. Warren. Multiple robot path coordination using artificial potential fields. In *Proc. IEEE Int. Conf. Rob. & Autom.*, pages 500–505, 1990.