Probabilistic Path Planning for Multiple Robots with Subdimensional Expansion

Glenn Wagner, Minsu Kang, Howie Choset

Abstract— **Probabilistic planners such as Rapidly-Exploring Random Trees (RRTs) and Probabilistic Roadmaps (PRMs) are powerful path planning algorithms for high dimensional systems, but even these potent techniques suffer from the curse of dimensionality, as can be seen in multirobot systems. In this paper, we apply a technique called subdimensional expansion in order to enhance the performance of probabilistic planners for multirobot path planning. We accomplish this by exploiting the structure inherent to such problems. Subdimensional expansion initially plans in each individual robot's configuration space separately. It then couples those spaces when robots come into close proximity with one another. In this way, we constrain a probabilistic planner to search a low dimensional space, while dynamically generating a higher dimensional space where necessary. We show in simulation that subdimensional expansion enhanced PRMs can solve problems involving 32 robots and 128 total degrees of freedom in less than 10 minutes. We also demonstrate that enhancing RRTs and PRMs with subdimensional expansion can decrease the time required to find a solution by more than an order of magnitude.**

I. Introduction

Multirobot systems have great promise for surveillance and search and rescue, as well as warehousing applications, thanks to their inherent redundancy and flexibility. Unfortunately, the flexibility and power of multirobot systems comes at the price of a high dimensional configuration space. Handling such high dimensional configuration spaces is one of the fundamental problems of multirobot path planning.

Probabilistic planners such as Rapidly-Exploring Random Trees (RRTs) [11] and Probabilistic Roadmaps (PRMs) [9] have demonstrated the ability to handle high dimensional systems [1], including multirobot systems [3], [6], [15], [17], [18]. Unfortunately, multirobot systems can reach sufficiently high dimensionality where even these approaches have difficulty. We seek to enhance the performance of RRTs and PRMs for multirobot path planning by exploiting the structure of

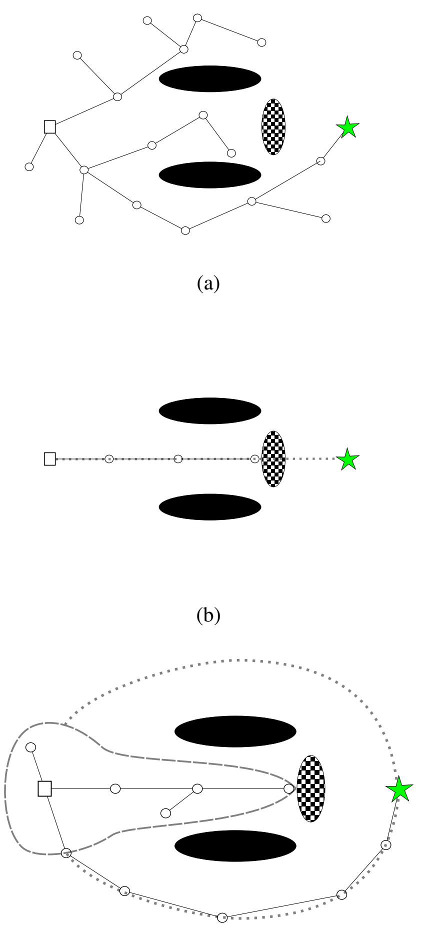


Fig. 1: **(a)** A standard probabilistic planner searches the full configuration space of a multirobot system to find a path from the initial configuration (square) to the goal (star) which avoids configuration space obstacles (ovals). Black ovals represent robot-obstacle collisions, while the checkered oval represents a robot-robot collision. **(b)** We seek to improve the performance of probabilistic planners for multirobot systems by restricting their search space to a one dimensional structure (dotted line). This search space is generated by combining paths found by planning for each robot separately, so the search space may initially be blocked by robot-robot obstacles. **(c)** When the search space is determined to be blocked, its dimensionality is increased around the states leading to the obstacle (area enclosed by dashed line). We then use independent planning in the single robot configuration spaces to generate new one-dimensional structures to reconnect the search space to the goal (dotted lines).

Glenn Wagner is a graduate student at the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, 15213 `gswagner@andrew.cmu.edu`

Howie Choset is a professor at the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, 15213 `choset@cs.cmu.edu`

Minsu Kang is an undergraduate at Seoul National University, Seoul, South Korea `minsuya@gmail.com`

the problem, to allow us to search a space smaller than the full configuration space of the the system (Figure 1).

One standard approach to planning for high dimensional multirobot systems is to apply a decoupled algorithm such as trajectory coordination or priority planning, in which the planning for each robot is done separately [5], [8], [12], [13], [14]. Sánchez and Latomb show that a decoupled algorithm (path coordination on graphs built by PRMs) often fails to find a solution in realistic environments [16]. We conclude from their work that even in the context of probabilistic planners, intelligently varying the dimensionality of the search space is necessary in order to combine the speed of decoupled approaches with the reliability of coupled approaches.

There are several algorithms which combine probabilistic planners with variable coupling. Clark *et al.* [4] have introduced dynamic networks, a run-time algorithm which uses communication constraints to determine which robots must couple their planning. Švestka and Overmars [19] have shown how to construct a roadmap in the full configuration space using a PRM constructed in a single robot configuration space. Furthermore, the authors show that decomposing the full configuration space roadmap into a hierarchy of mutually independent subgraphs allows planning for multiple robots to be partially decoupled without losing probabilistic completeness. Sucan and Kavraki developed Task Motion Multigraphs (TMMs) for mobile manipulation, which provide multiple spaces of varying dimensionality in which the planning for a specific task can be performed [18][17]. TMMs can be adapted for use in multirobot path planning as they can implicitly perform decoupled planning, while still being able to search higher dimensional spaces when necessary.

In our previous work [20], we introduced a technique for multirobot path planning that we call *subdimensional expansion*. Subdimensional expansion exploits the structure of the multirobot path planning problem by first planning for each robot separately, in its own configuration space. We then combine the paths produced by the individual plans to form a one-dimensional search space, embedded in the full configuration space, for the full system. As the search space is explored, robot-robot collisions are found. These collisions are used to guide the expansion of the search space, producing a search space whose dimensionality is dependent on the interactions between robots, rather than on the total number of robots in the system. While subdimensional expansion has distinct similarities with TMMs, subdimensional expansion can increase and especially decrease the dimensionality of the search space with greater flexibility and locality than TMMs, and has a more principled way of determining how the dimensionality of the search space must be augmented. We previously applied subdimensional expansion to graph search, resulting in the *M** algorithm [20].

In this paper, we apply subdimensional expansion to probabilistic path planning techniques, namely RRTs and PRMs. Basing subdimensional expansion on probabilistic planners improves its performance for robots with higher dimensional individual configuration spaces, allowing us to plan for systems of more complex robots, such as those with dynamic constraints. While the basic concept of subdimensional expansion remains constant, the manner in which the search space is constructed depends upon the planner being used.

II. Subdimensional Expansion

Subdimensional expansion is a technique for constructing a search space while being explored by a planner, the partial results of which are used to guide the construction of the search space. Subdimensional expansion is built on two primary concepts; the *individual policy* and the *collision set*.

The individual policy ϕ^i maps points in the configuration space Q^i of the ith robot r^i, to the optimal action for r^i at that location in the absence of other robots. Each robot obeys its respective individual policy until there is evidence that doing so may result in a collision with another robot (Figure 2).

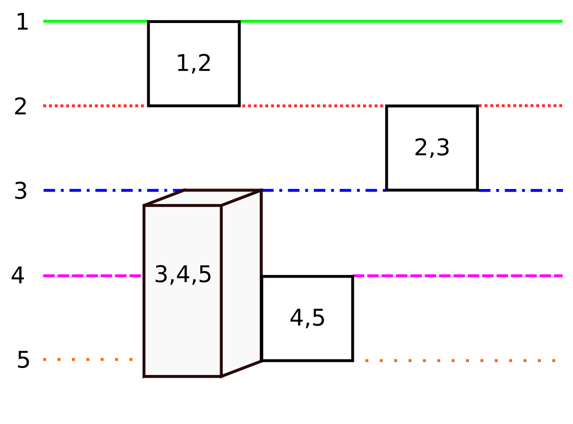
The collision set records the robots for which the individual policy is insufficient. The collision set C_k for a given point in the joint configuration space $q_k \in Q = \prod_i Q^i$ is the set of robots r^i for which the planner has found a path through q_k to a collision between r^i and another robot. C_k is updated as the planner explores more paths and discovers more collision states.

When the planner seeks to extend a path from q_k, each robot not in C_k can safely obey its individual policy, but all possible paths for robots in C_k must be considered. A search space of variable dimensionality is constructed by following this procedure. While theoretical guarantees will depend on the details of the implementation, we were able to show that M* [20], an implementation of subdimensional expansion for graph search, is complete and produces optimal paths. When the path optimality is relaxed to ϵ-suboptimality[1] we showed empirically that M* has sub-exponential growth in computational cost with the number of robots.

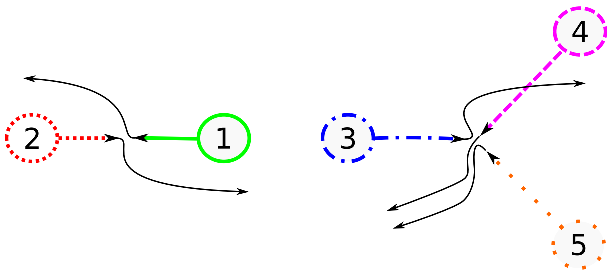
III. sRRT

RRTs were introduced by LaValle and Kuffner as an efficient single-query planner for high dimensional systems [11]. RRTs construct a search tree in the full configuration space of a system whose root is the

[1]A guarantee that the returned path will cost no more than ϵ times the cost of the optimal path



(a)



(b) (c)

Fig. 2: **(a)** A conceptual visualization of a variable dimensionality search space for five robots. Initially each robot is constrained to its individual policy, represented by a single line. **(b)** When robots 1 and 2 collide, the local dimensionality of the search space must be increased, as represented by a square. **(c)** When three robots collide while following their individual policy, the local dimensionality of the search space must be increased further, represented by the cube, to include all local paths of the three robots. Once robot 3 clears robots 4 and 5, it no longer must remain coupled with them, even though robots 4 and 5 continue to interact. Depending on the desired path optimality, the interaction between robots 2 and 3 may be resolved without coupling their planning with robots with which they had previously interacted.

initial configuration. The tree is extended in a three step process. First, a random sample is drawn from the full configuration space. A local planner then finds a local path from the nearest node in the search tree towards the random sample. Finally, the local path is tested for collisions. If no collisions are found, the final configuration of the local path, which may not be the same as the random sample depending on the local planner, is added to the search tree. Once a configuration sufficiently close to the goal is added to the tree, a path to the goal from the initial configuration can be retrieved.

Subdimensional expansion can be implemented with RRTs, resulting in the *sRRT* algorithm. To do so, we must modify the sample generation step, so samples are drawn from the search space constructed by subdimensional expansion instead, of the full configuration space. We must also add a step to update the collision sets based on the results of collision checking each local path.

We denote an RRT tree as a set of nodes, $\mathcal{T} = \{(q_k, p_k, C_k)\}$, where each node contains a configuration q_k, a pointer to its predecessor node p_k, and a collision set C_k.

While we prefer to use optimal individual policies, constructing such policies is infeasible when the dimensionality of the individual robot configuration space is large. Therefore, we use RRTs to construct the individual policies. The individual policy for r^i is constructed by building a tree $\mathcal{T}^i$ in Q^i. We grow the tree back from the goal configuration q_F^i so that we can efficiently find a path to the goal from many different configurations. Since the tree is grown backwards from the goal configuration, the predecessor of q_k in $\mathcal{T}^i$ is the next step on the path defined by ϕ^i to the goal from q_k^i. We can therefore use p_k^i to define the individual policy

$$\phi^i(q_k^i) = p_k^i. \tag{1}$$

If $q_k^i \notin \mathcal{T}^i$, we first extend $\mathcal{T}^i$ until $q_k^i \in \mathcal{T}^i$.

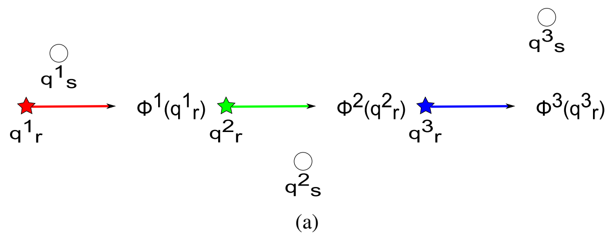
Path planning for the full system is done by growing a tree $\mathcal{T}_f$ forwards in the full configuration space, from a root node at the initial system configuration q_I. The expansion of $\mathcal{T}_f$ is restricted to the search space $Q^\#$ determined by subdimensional expansion. We construct the search space by identifying each node $q_k \in \mathcal{T}_f$ with a local search space $Q_k^\# = \{q \mid q^i = \phi^i(q_r^i) \forall i \notin C_r\}$. The local search space is the set of configurations which can be reached from q_r when the robots not in C_r obey their individual policy.

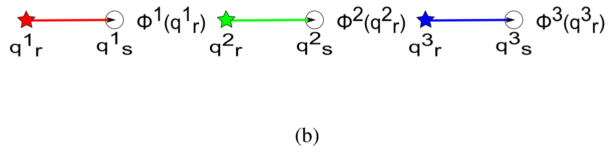
We can bring a randomly generated sample $q_s \in Q$ into $Q^\#$ by a simple projection operation. Let q_r be the nearest node to q_s in $\mathcal{T}_f$. We can project q_s onto $Q_r^\#$ by replacing the coordinates for each robot not in C_r with the coordinate for that robot's next step along its individual policy from q_r (Figure 3).

$$q_s' = \prod_i \begin{cases} \phi^i(q_r^i) & r^i \notin C_r \\ q_s^i & r^i \in C_r \end{cases} \tag{2}$$

Once a sample is generated, and the local planner finds a path from q_r to q_s', the collision checker searches for robot-robot and robot-obstacle collisions. If r^i is found to be involved in a robot-robot collision, r^i is added to C_r, then recursively added to the collision sets of the predecessor nodes of q_k.

The collision set update process is necessary to determine when $Q^\#$ is blocked by robot-robot collisions, and serves to increase the local dimensionality of $Q^\#$ to allow paths around said collisions to be found. As a result, while our search space is low dimensional, like





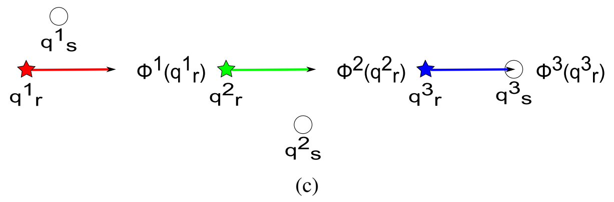
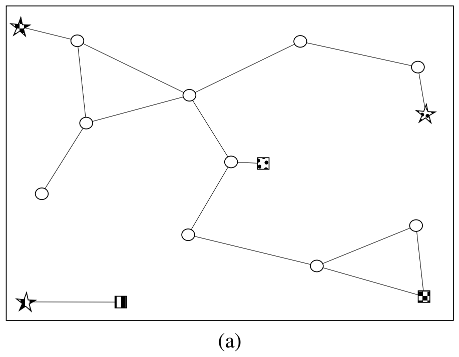


Fig. 3: **(a)** We show how a random sample q_s gets projected onto the search space in the Voronoi region of the node q_r. We visualize a three robot configuration space by showing the coordinates of each robot side by side. q_r^i gives the location of the ith robot in q_r. The arrow points along the individual policy from q_r to the next configuration $\phi(q_r)$. In the circles show the random sample q_s before any projection. **(b)** $C_r = \emptyset$, so each robot is restricted to its individual policy, resulting in the projected sample q_s'. **(c)** $C_r = \{r^1, r^2\}$. Therefore, only robot r^3 is restricted to its individual policy, resulting in the projected sample q_s''



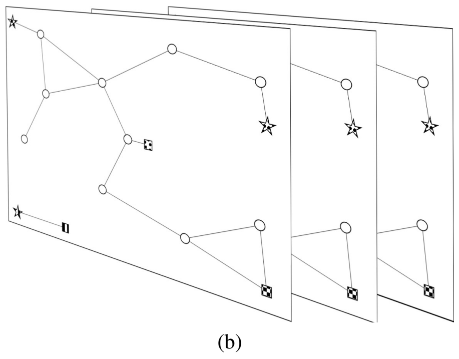


Fig. 4: **(a)** We show a PRM constructed in a single robot configuration space that connects the initial configuration (box) to the the goal configuration (star) for three homogeneous robots, indicated by pattern. Circles are randomly generated samples used to construct the PRM. **(b)** We construct a PRM in the full configuration space of the three robot system by taking the Cartesian product of three copies of single robot PRM.

that of a decoupled algorithm, it can dynamically grow when it has been shown to be insufficient, avoiding the failures that may be inevitable with a fully decoupled approach.

IV. Subdimensional Expansion With PRMs

A PRM generates a roadmap covering the configuration space, which is then passed to a graph search algorithm to extract the desired path. M* is sufficiently general that we can apply it to the graphs generated by a PRM, resulting in the *sPRM* algorithm. sPRM allows for path planning of multirobot systems while only constructing PRMs in the single robot configuration spaces. Our approach to constructing PRMs follows that of Švestka and Overmars [19], who used multiple copies of a single robot PRM to construct a roadmap in the full configuration space (Figure 4).

We construct a PRM in the configuration space of each robot r^i, resulting in the roadmaps G_S^i. We use the initial and goal configurations of the robot, q_I^i and q_F^i as samples in addition to random samples. We can construct a single PRM to be shared by multiple homogeneous robots, in which case the initial and goal configurations of all homogeneous robots are used as additional samples. We assume that q_I^i and q_F^i lie in the same connected component of G_s^i. We acknowledge that this may be a computationally difficult problem, especially for workspaces with narrows corridors [2], [7]; this is a problem common to all PRMs. For the sake of explanation, we assume that PRMs can be successfully constructed in single robot configuration spaces.

We then run rM*, a more efficient variant of M* [20], on the implicitly defined graph $G = \prod_i G_S^i$ to find a joint path. Since G is constructed incrementally, robot-robot collisions are checked in a lazy manner giving some of the benefits of the lazy collision checking performed by Bohlin and Kavraki [1]. Assuming a collision free joint path exists, Švestka and Overmars [19] showed that the probability of G containing a collision free path goes to 1 as the time spent generating G_S goes to infinity. Since M* is complete [20], the procedure described above is probabilistically complete. While RRTs can be used to generate roadmaps, such

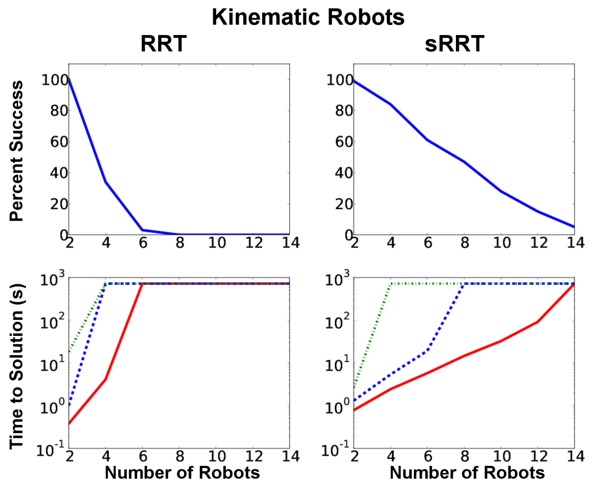


Fig. 5: We plot the percent of trials in which each algorithm was able to find a solution within 12 minutes, and the 10th (solid), 50th (dashed), and 90th (dots) percentile of times required to find a solution using RRT and sRRT. The plateauing that is apparent in the time plots is the result of the algorithms timing out in increasingly large percentages of trials.

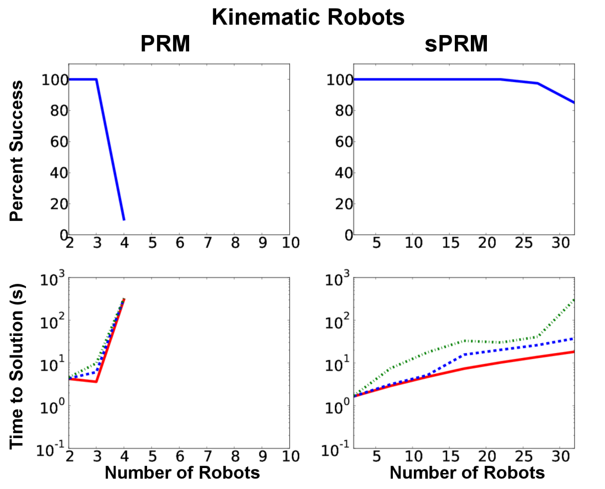


Fig. 6: We plot the percent of trials in which each algorithm was able to find a solution within 5 minutes, and the 10th (solid), 50th (dashed), and 90th (dots) percentile of times required to find a solution using subdimensional PRM. Note that the plot for the standard PRM goes to 10 robots, while sPRM reaches 32 robots

roadmaps are inappropriate for this approach, since trees do not contain efficient alternate paths between two points.

V. Results

To test our algorithms, we ran simulations on an AMD X6 at 2.8 GHz with 8 GB of RAM. We simulated planar circular robots of radius 0.5 in the presence of circular obstacles of radius 0.5. We generated square environments with 1 robot per 100 units area, and 5 obstacles per 100 units area. All coding was done in unoptimized python. Our PRM implementations are parallelized across the six available cores.

A. sRRT

For sRRT, we simulated kinematic circular robots in the plane. Each trial was given five minutes to find a solution, or the trial would be considered a failure. We ran 100 trials for each algorithm for each number of robots. Our simulations show (Figure 5) that sRRT offers substantial performance advantages over basic RRT. sRRT can successfully find a path within 12 minutes twice as often as RRT at 4 robots, and sRRT still has a 50% success rate at 8 robots, where basic RRT almost never finds a solution within the time constraints.

B. sPRM

We first implement sPRM for simple kinematic robots. For a given number of robots, sPRM was tested on 40 environments, while basic PRM was tested on 10 environments. Our basic PRM implementation has great difficulty solving problems involving 4 robots within our 5 minute time constraints, while sPRM can effectively solve 32 robot problems (Figure 6).

We then show the ability of sPRM to handle robots with dynamic constraints. Our simulated robots used the following dynamics.

$$\begin{bmatrix}\dot{x}\\ \ddot{x}\\ \dot{y}\\ \ddot{y}\end{bmatrix} = \begin{bmatrix}0&1&0&0\\0&0&0&0\\0&0&0&1\\0&0&0&0\end{bmatrix}\begin{bmatrix}x\\ \dot{x}\\ y\\ \dot{y}\end{bmatrix} + \begin{bmatrix}0&0\\1&0\\0&0\\0&1\end{bmatrix}\begin{bmatrix}a_x\\ a_y\end{bmatrix}, \qquad (3)$$

where $a_x, a_y \in [-5, 5]$ are the commanded acceleration. Our local planner uses cubic basis functions to find a solution to the system of equations (3) which connects two desired points in the configuration space

$$\begin{bmatrix}x(t)\\ \dot{x}(t)\\ y(t)\\ \dot{y}(t)\end{bmatrix} = \begin{bmatrix}a_x&b_x&c_x&d_x\\0&3a_x&2b_x&c_x\\a_y&b_y&c_y&d_y\\0&3a_y&2b_y&c_y\end{bmatrix}\begin{bmatrix}t^3\\ t^2\\ t\\ 1\end{bmatrix}, \qquad (4)$$

$$t \in [0, 1].$$

The boundary conditions for (4) can be written as

$$\begin{bmatrix}x(0)\\ x(1)\\ \dot{x}(0)\\ \dot{x}(1)\\ y(0)\\ y(1)\\ \dot{y}(0)\\ \dot{y}(1)\end{bmatrix} = \begin{bmatrix}0&0&0&1&0&0&0&0\\1&1&1&1&0&0&0&0\\0&0&1&0&0&0&0&0\\3&2&1&0&0&0&0&0\\0&0&0&0&0&0&0&1\\0&0&0&0&1&1&1&1\\0&0&0&0&0&0&1&0\\0&0&0&0&3&2&1&0\end{bmatrix}\begin{bmatrix}a_x\\ b_x\\ c_x\\ d_x\\ a_y\\ b_y\\ c_y\\ d_y\end{bmatrix}. \qquad (5)$$

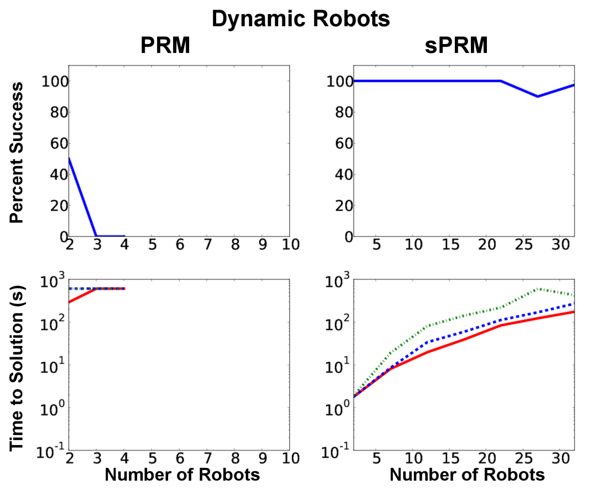


Fig. 7: We plot the percent of trials in which each algorithm was able to find a solution within 10 minutes, and the 10th (solid), 50th (dashed), and 90th (dots) percentile of times required to find a solution using a PRM in the full configuration space and subdimensional PRM. The plateauing that is apparent in the time plots is the result of the algorithms timing out in increasingly large percentages of trials. Note that the full configuration space PRM results are only plotted to 10 robots, while the sPRM results are plotted to 32 robots.

The coefficients that define the trajectory can be found by matrix inversion.

The results of our simulation show the dramatic improvement in performance offered by sPRM over a full configuration space PRM (Figure 7). The full configuration space PRM had only a 50% success rate in solving a 2 robot problem, and never succeeded in solving a 3 robot problem within the 10 minute time limit (which allowed for approximately 20,000 samples). sPRM was able to solve 32 robot problems with high reliability.

We attribute this difference in performance in part to the fact that the volume space that can be reached in a single step by our local planner without violating the acceleration limits is comparatively small. Therefore, the likelihood that all the robots in the system would be able to transition between the states specified by two random samples from the configuration is very low. As a result, building a conventional PRM in the full configuration space for the above combination of dynamics and local planner is very difficult.

C. Narrow Passages

To test the performance of our algorithms in the presence of narrow passages, we constructed an environment in which between two and six robots must swap positions, after passing through a narrow passage (Figure 8). We tested each algorithm 10 times in this environment for two, four, and six robots, using a three hour time-out. To allow for more rapid testing and better comparisons, the PRMs were restricted to a single core. The basic RRT ran out of time in three of the 6 robot trials, while the basic PRM always timed out for the 6 robot trial (Figure 9).

Our results show that subdimensional expansion provides significant benefits when dealing with narrow passages. The individual policies effectively allow the narrow passage problem to be solved in the single robot configuration spaces, leaving only the coordination problem to be resolved in the joint configuration space. On average, sPRM performed substantially worse than sRRT, but this poor performance is mainly due to a small number of trials which took extremely long amounts of time. The solution time of sPRM for 6 robots was dominated by the time required to search the roadmap with rM*, as the actual construction of the roadmap took at most 6 seconds. This suggests that the best collision free path in the roadmap costs much more than the optimal path for which collisions are ignored. Such an environment would force rM* to search a large portion of the joint configuration space, dramatically increasing the time required to find a solution.

VI. Conclusions

We have demonstrated that subdimensional expansion can be applied to probabilistic planners and systems with dynamic constraints. We have further shown that implementing subdimensional expansion can decrease the time required to find solutions by more than an order of magnitude, especially for systems of 4 or more robots. We note that sPRM generates superior results to sRRT for comparable systems. We believe that this is partly due to our use of rM* as the graph search algorithm in sPRM. rM* can decouple groups of robots involved in widely separated by simultaneous collisions[20], substantially reducing the dimensionality of the search space. We intend to implement a similar approach in sRRT as future work.

We also intend to investigate proofs of probabilistic completeness for sRRT. Standard approaches to proving the probabilistic completeness of planners cannot be applied to sRRT, because sRRT will never cover the full configuration space if no collisions occur along the path initially generated by the individual policies [10]. There are also certain pathological cases involving discrete individual policies that must also be addressed in any such proof.

References

[1] R. Bohlin and L.E. Kavraki. Path planning using lazy prm. In *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, volume 1, pages 521–528. IEEE, 2000.

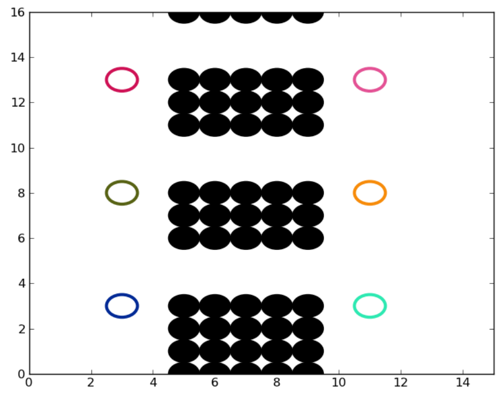


Fig. 8: Our test environment for narrow passages for between two and six robots. The initial positions of all six robots are indicated by the hollow circles. Robots that are horizontally opposed must swap positions, while avoiding the obstacles, shown as filled black circles. Each passage is two robot diameters wide, and five diameters long. Only the bottom most pair of robots were used for the two robot case, while the four robot case used all but the top most pair of robots.

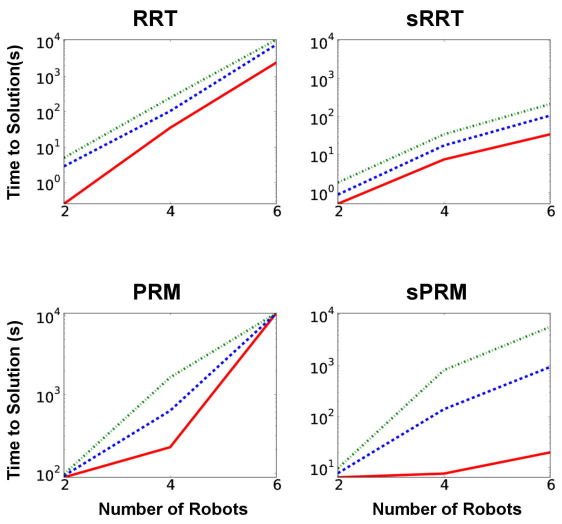


Fig. 9: Results from testing basic RRT, sRRT, basic PRM and sPRM on the environment show in Fig 8. We tested each algorithm 10 times at two, four and six robots. We plot the minimum, average, and maximum time to solution for each algorithm. We used a time limit of 3 hours. The basic RRT timed out in 3 trials in the six robot case, while basic PRM timed out in all 10 trials for the six robot case.

[2] V. Boor, M.H. Overmars, and A.F. van der Stappen. The gaussian sampling strategy for probabilistic roadmap planners. In *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*, volume 2, pages 1018–1023. IEEE, 1999.

[3] S. Carpin and E. Pagello. On parallel RRTs for multi-robot systems. In *Proc. 8th Conf. Italian Association for Artificial Intelligence*, pages 834–841. Citeseer, 2002.

[4] C. M. Clark, S. M. Rock, and J. C. Latombe. Motion planning for multiple robot systems using dynamic networks. In *Proceedings of IEEE International Conference on Robotics and Automation*, pages 4222–4227, 2003.

[5] M. Erdmann and T. Lozano-Perez. On multiple moving objects. *Algorithmica*, 2(1):477–521, 1987.

[6] D. Ferguson, N. Kalra, and A. Stentz. Replanning with rrts. In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pages 1243–1248. IEEE, 2006.

[7] D. Hsu, T. Jiang, J. Reif, and Z. Sun. The bridge test for sampling narrow passages with probabilistic roadmap planners. In *Robotics and Automation, 2003. Proceedings. ICRA'03. IEEE International Conference on*, volume 3, pages 4420–4426. IEEE, 2003.

[8] K. Kant and S.W. Zucker. Toward efficient trajectory planning: The path-velocity decomposition. *The International Journal of Robotics Research*, 5(3):72, 1986.

[9] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configurations spaces. *IEEE Transactions on Robotics and Automation*, 12:566–580, June 1996.

[10] L.E. Kavraki, M.N. Kolountzakis, and J.C. Latombe. Analysis of probabilistic roadmaps for path planning. In *Robotics and Automation, 1996. Proceedings., 1996 IEEE International Conference on*, volume 4, pages 3020–3025. IEEE, 1996.

[11] S. M. LaValle and J. J. Kuffner. Randomized kinodynamic planning. In *Proceedings IEE International Conference on Robotics and Automation*, 1999.

[12] Stephane Leroy, Jean-Paul Laumond, and Thierry Siméon. Multiple path coordination for mobile robots: A geometric algorithm.

In *IJCAI '99: Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, pages 1118–1123, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc.

[13] Ryan Malcom. Multi-robot path-planning with subgraphs. In *Australasian Conference on Robotics and Automation*, 2006.

[14] M. Saha and P. Isto. Multi-robot motion planning by incremental coordination. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 5960–5963, Oct. 2006.

[15] G. Sánchez and J.C. Latombe. On delaying collision checking in prm planning: Application to multi-robot coordination. *The International Journal of Robotics Research*, 21(1):5, 2002.

[16] G. Sanchez and J.C. Latombe. Using a prm planner to compare centralized and decoupled planning for multi-robot systems. In *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, volume 2, pages 2112–2119. IEEE, 2002.

[17] I.A. Sucan and L.E. Kavraki. Mobile manipulation: Encoding motion planning options using task motion multigraphs. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 5492–5498. IEEE, 2011.

[18] I.A. Sucan and L.E. Kavraki. On the advantages of task motion multigraphs for efficient mobile manipulation. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pages 4621–4626. IEEE, 2011.

[19] P. Švestka and M.H. Overmars. Coordinated path planning for multiple robots. *Robotics and Autonomous Systems*, 23(3):125–152, 1998.

[20] Glenn Wagner and Howie Choset. M*: A complete multirobot path planning algorithm with performance bounds. In *Intelligent Robots and Systems, 2011 IEEE/RSJ International Conference on*, September 2011.