Planning with Reachable Distances: Fast Enforcement of Closure Constraints

Xinyu Tang, Shawna Thomas, and Nancy M. Amato

***Abstract*— Motion planning for closed-chain systems is particularly difficult due to additional *closure constraints* placed on the system. In fact, the probability of randomly selecting a set of joint angles that satisfy the closure constraints is zero. We propose *Planning with Reachable Distance* (PRD) to overcome this challenge by first precomputing the subspace satisfying the closure constraints, then directly sampling in it. To do so, we represent the chain as a hierarchy of sub-chains. Then we calculate the "closure" sub-space as appropriate reachable distance ranges of sub-chains satisfying the closure constraints. This provides two distinct advantages over traditional approaches: (1) configurations are quickly sampled and converted to joint angles using basic trigonometry functions instead of more expensive inverse kinematics solvers, and (2) configurations are guaranteed to be closed.**

In this paper, we describe this hierarchical chain representation and give a sampling algorithm with complexity linear in the number of links. We provide the necessary motion planning primitives for most sampling-based motion planners. Our experimental results show our method is fast, making sampling closed configurations comparable to sampling open chain configurations that ignore closure constraints. Our method is general, easy to implement, and also extends to other distance-related constraints besides the ones demonstrated here.

I. Introduction

Closed-chain systems, as in Figure 1, are involved in many applications in robotics and beyond, such as parallel robots [12], closed molecular chains [16], animation [4], reconfigurable robots [7], [14], and grasping [6]. However, motion planning for closed-chain systems is particularly challenging due to additional constraints, called *closure constraints*, placed on the system. Using only the traditional joint angle representation, the probability that a random set of joint angles lies on the constraint surface is zero [9].

Instead of randomly sampling in the joint angle space to find closed configurations, we propose *Planning with Reachable Distance* (PRD). PRD overcomes this challenge by precomputing the subspace where closed constraints are satisfied and then directly sampling in this subspace. We represent the chain as a hierarchy of sub-chains by recursively breaking down the problem into smaller subproblems. Each sub-chain in the hierarchy may be partitioned into other, smaller sub-chains forming a closed loop. For any sub-chain, we can compute the attainable distance (reachable distance or length) between its two endpoints recursively. With this information, we simply sample distances in these ranges, and then use basic geometry relationships to calculate the joint angles. Thus, we can directly sample in the "reachable" space, which is expensive to compute explicitly. While a linkage's reachable distance has been used in other computations, to our knowledge it has not been used to guide sampling in sampling-based planners. PRD has two distinct advantages over traditional approaches:

- configurations are quickly sampled and converted to joint angles using basic trigonometry functions instead of more expensive inverse kinematics solvers.
- configurations are guaranteed to be closed.

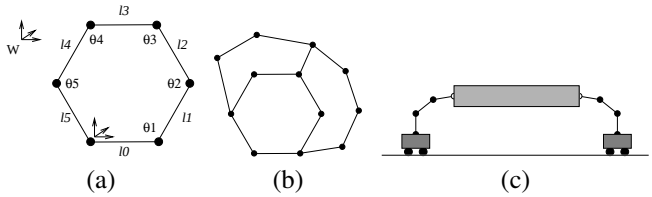


Fig. 1. Examples of different closed-chain systems. Each must satisfy certain closure constraints. (a) Single loop: loop must remain closed. W is the world coordinate frame. (b) Multiple loops: all loops must remain closed. (c) Multiple robot grasping: all robot end-effectors must remain in contact with the grasped object.

In this paper, we formally describe this new chain representation and give a recursive sampling algorithm with complexity linear in the number of links in the chain. This algorithm explores the closure constraint surface by recursively sampling in the feasible reachable distance range for each sub-chain. It can quickly determine whether or not it is possible to satisfy the closure constraints of a sub-chain's children. Our method does not guarantee collision-free configurations; like many methods, they must be checked afterwards.

PRD can significantly improve the performance of sampling-based planners for closed-chain systems, such as *Probabilistic Roadmap Methods* (PRMs) [5] and *Rapidly-exploring Randomized Trees* (RRTs) [10], which have been widely successful in solving other high degree of freedom (dof) problems. However, the traditional joint angle representation makes it difficult for these methods to sample closed configurations. While several strategies use heuristics to improve the probability of sampling closed configurations [9], [19], [2], [18], [1], [17], sampling is still difficult and expensive for large systems. Here, we provide two necessary motion planning primitives (sampling and local planning) to implement most of these sampling-based planners.

Our experimental results show that our method is fast and efficient in practice making sampling configurations with closure constraints comparable to sampling open chain

Research supported in part by NSF Grants EIA-0103742, ACR-0081510, ACR-0113971, CCR-0113974, ACI-0326350, by the DOE, and by Hewlett-Packard. Thomas supported in part by an NSF Graduate Research Fellowship, a PEO Scholarship, and a DOE GAANN Fellowship.

configurations that ignore closure constraints entirely. Our method is easy to implement and general — it can be applied to other articulated systems. It is also extendible to more distance-related constraints besides the ones here. For example, it can be used to sample configurations with the end effector in specific regions.

II. Preliminaries

This paper focuses on closed-chain systems. A closed-chain system differs from other systems in that it must also satisfy certain closure constraints. Figure 1 gives examples of different types of closed-chain systems. Each system has a different set of closure constraints to satisfy.

Closed-chain systems are traditionally represented by the position and orientation of the base or a base link and one or more angles for each joint (corresponding to the joint's dof). For example, the single loop in Figure 1(a) is represented by 11 values: 6 for the position and orientation of link $l0$ and 5 for the angles $\theta 1 \ldots \theta 5$. While this representation sufficiently describes a configuration, it is extremely difficult to sample these parameters randomly while satisfying the closure constraints. This is because this joint angle representation does not also encode the closure constraints — they are handled separately. In fact, it has been shown that the probability of randomly sampling a set of joint angles that satisfy the closure constraints is zero [9].

III. Related Work

In theory, exact motion planning algorithms [15], [8] can handle systems with closure constraints. However, since exact algorithms are exponential in the dimension of C-space (the set of all possible robot configurations, valid or not), they are generally impractical for large closed-chain systems.

Randomized algorithms such as Probabilistic Roadmap Methods (PRMs) [5] and Rapidly-exploring Randomized Trees (RRTs) [11] are widely used today for many applications. They first construct a roadmap (graph or tree) that represents the connectivity of the robot's free C-space, and then query the roadmap to find a valid path for a given motion planning task. Initially, they were mainly limited to rigid bodies and articulated objects without closure constraints.

Kavraki et al. [9], [19] randomly samples configurations and then applies an iterative random gradient descent method to push the configuration to the constraint surface. They solved planar chains with up to 8 links and 2 loops in several hours with PRM and several minutes with RRT. Han et al. [2] uses a kinematics-based PRM approach by first building a roadmap ignoring all obstacles, and then populating the environment with copies of this kinematics roadmap, removing invalid portions. This method can solve chain problems with 7 – 9 links in under a minute. This work is extended to increase the number of links the method can handle in [1], [18]. Trinkle and Milgram proposed a path planning algorithm [17] based on configuration space analysis [13]. Their method does not consider self-collision but still may be applied as a local planner. Recently, Han et al. [3] proposed a set of geometric parameters for closed-chain systems such that the problem can be reformulated as a system of linear inequalities. Then, linear programming and other algorithms can be used for sampling and local planning. However, they do not discuss the algorithm's complexity or consider collisions in planning.

Our method is very fast and has complexity linear in the number of links. It guarantees that a closed configuration will be generated or reports that one cannot be obtained. Moreover, our method is general and easy to implement for sampling-based planners. It is extendible to more distance-related constraints than the ones considered here.

IV. Reachable Distance Representation

Here we describe our hierarchical representation based on reachable distances. The main advantage of this representation over the traditional joint angle representation is that it also encodes the closure constraints. This new representation allows us to easily randomly sample closed configurations.

Intuitively, in a closed-chain system, the length of each link has to be in an appropriate range to satisfy the closure constraints. For example, Figure 2 shows a simple triangular closed-chain with 3 links a, b and c of variable length. To sample a closed configuration, we have to make sure the length of each link is in an "appropriate" (feasible) range. In other words, the length of each link needs to satisfy the triangle inequality: $|c| >= |a + b|$ and $|c| <= |a - b|$. To sample a closed configuration, we first can calculate the link's remaining available range based on the available ranges of the other links. In this way, we can eventually sample a valid length for each link to satisfy the closure constraints or report that it is impossible to find a closed configuration. For the triangular case in Figure 2, once we get a valid length for each link, we get a triangular configuration that is guaranteed to be closed. Below we show how we extend this sampling strategy to handle a general closed-chain system. This scheme only ensures that the closure constraints are satisfied. Collision checking must still be performed to determine the configuration's validity. Note that this representation can also support joint angle limits by restricting the feasible range of the links. For simplicity, we ignore joint angle limits in this paper.

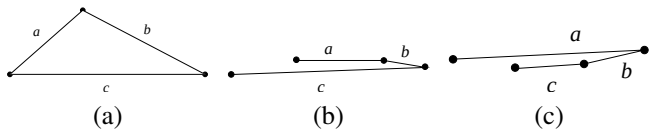


Fig. 2. Configurations of an articulated system with 3 links. a, b and c each has variable length. (a) Each link has an appropriate length to close the configuration. (b) c is too long. (c) c is too short.

A. Closed-Chain as a Hierarchy of Sub-Chains

For simplicity, here we consider a single loop closed-chain. However, this representation is general and may be applied to other closed-chain systems. The chain (called the *parent*) may be partitioned into several sub-chains (called

the *children*) where each sub-chain is a set of consecutive links and the union of the children represents all the links in the parent. For example, the *virtual link* 16, see Figure 3(a), connects the two actual links 0 and 1. In this way, the parent is closed if and only if the virtual links and its children form a closed-chain. We recursively partition sub-chains until all the children have only 1 link. This defines a hierarchy of sub-chains and virtual links, as displayed in Figure 3(b).

B. Reachable Range of a Sub-Chain

The *reachable distance* of a sub-chain is the distance between the endpoints of its virtual link. The set of all possible reachable distances for a sub-chain is called its *reachable range*. For example, the reachable range of a single link is its length. A parent link's reachable range depends on the reachable ranges of its children. If we build the sub-chain hierarchy such that each sub-chain has only 0 or 2 children, then the reachable ranges are computed as follows.

Consider a sub-chain with no children. Let l_{min} and l_{max} be the minimum and maximum link length, respectively. (For a non-prismatic link, $l_{min} = l_{max}$.) The reachable range is then $[l_{min}, l_{max}]$. Now consider the sub-chain with 2 children in Figure 4(a). The sub-chain and its two children form a triangle. Let $[a_{min}, a_{max}]$ and $[b_{min}, b_{max}]$ be the reachable range of the first child and the second child respectively. The reachable range of the parent is then $[l_{min}, l_{max}]$ where

$$l_{min} = \begin{cases} max(0, b_{min} - a_{max}), & a_{min} < b_{min} \\ 0, & a_{min} = b_{min} \\ max(0, a_{min} - b_{max}), & a_{min} > b_{min} \end{cases} \quad (1)$$

and $l_{max} = a_{max} + b_{max}$. Note that Equation 1 is general. Given the reachable ranges of any two links in the same triangular sub-chain, it calculates the reachable range of the third one to satisfy the triangle inequality.

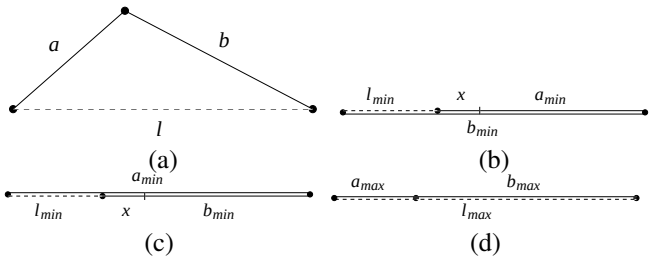


Fig. 4. (a) A sub-chain with 2 children a and b and its virtual link l. (b) A configuration where l is minimum when $a_{min} < b_{min}$ and $x = min(a_{max} - a_{min}, b_{min} - a_{min})$. (c) A configuration where l is minimum when $a_{min} > b_{min}$ and $x = min(b_{max} - b_{min}, a_{min} - b_{min})$. (d) A configuration where l is maximum l_{max}.

A closed configuration is then a set of distances for each virtual link in the hierarchy such that each distance is within the virtual link's reachable range. We can then compute its joint angles using only basic trigonometry relationships instead of more expensive inverse kinematics solvers. We discuss sampling in more detail in Section V-A.

C. Available Range of a Sub-Chain

The available reachable range (*available range*) of a virtual link is set of distances/lengths which allow it to close with the other links in the same sub-chain (i.e., satisfy the triangle inequality). The available range is a subset of the reachable range that is a function of the available ranges (or lengths) of other links in the same sub-chain loop. For example, in the beginning, for each sub-chain with 2 children, the reachable ranges of all 3 links can satisfy the triangle inequality and thus the available range is the same as the reachable range. However, once we fix the length of a link, portions of the reachable ranges of other links in the same sub-chain may no longer be valid. When this happens, we need to recalculate their available ranges using Equation 1. Note that Equation 1 is used in a more general way here. a, b and l can be any link in the same triangular sub-chain.

V. Application to Sampling-Based Planning

Here we describe two primitive operations for most randomized sampling-based motion planners such as PRMs and RRTs: sampling and local planning (i.e., finding a valid path between two samples). We show that these operations are fast and efficient, thereby allowing sample-based motion planners to be directly applied to large closed-chain problems.

A. Sampling

To sample a closed configuration and determine its joint angles, we first recursively sample the lengths of each virtual link. We then sample the orientation of each sub-chain (e.g., concave or convex for chains in the plane, dihedral angles for non-planar chains). Finally, we use the virtual link lengths and orientations to compute the appropriate joint angles. Note that this sampling only ensures that the configuration is closed — it will still need to be checked for collision. We describe each step in more detail below.

1) Recursively sample link lengths: Because we know the reachable range for each sub-chain in the hierarchy, sampling a *closed* configuration simply becomes sampling distances in the *available* reachable ranges of each sub-chain. Once we fix the length or reachable distance of a parent sub-chain, portions of its children's reachable ranges may become invalid. We define the remaining valid portions of their reachable range as their *available reachable range*. Note that an available reachable range may never become empty, at the very least its minimum and maximum may become equal. Thus, we sample reachable distances and update available reachable ranges starting at the the root of the hierarchy until all sub-chain reachable distances are sampled. Algorithm V.1 describes this recursive sampling strategy.

Recall that the sub-chain hierarchy is a binary tree and that there is one internal node for each virtual link. The sampling algorithm is called once on each virtual link. Because there are $O(n)$ internal nodes in the binary tree (n is the number of actual links), the sampling algorithm runs in $O(n)$ time.

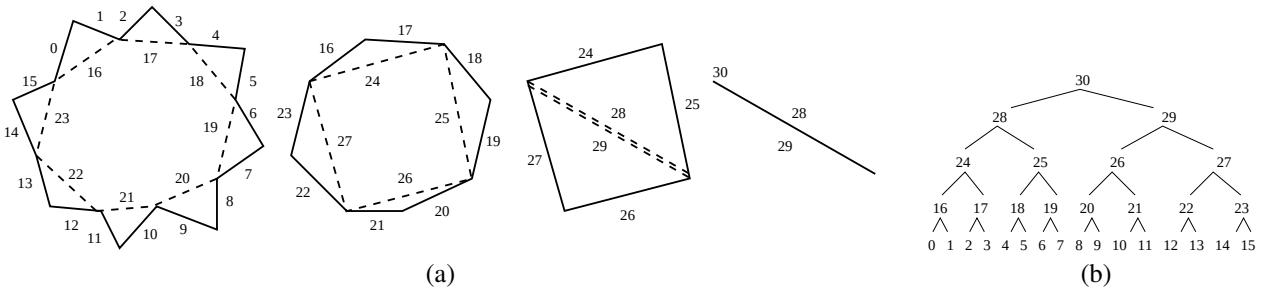


Fig. 3. (a) A chain (indicated by solid links in the first image) may be partitioned into a set of sub-chains represented by virtual links (indicated by dashed links). This partitioning repeats to form a hierarchy where the virtual links in one level become the actual links of the next level. (b) The tree represents the entire sub-chain hierarchy where nodes correspond to virtual links.

Algorithm V.1 Sample

Input. A sub-chain c. Let $c.arr$ be c's available reachable range, $c.left$ and $c.right$ be c's children, and $c.len$ be the length of c's virtual link. Let p be c's parent and s be c's sibling.

1: Update $c.arr$ from $p.arr$ and $s.arr$.
2: Randomly sample $c.len$ from $c.arr$.
3: Set $c.arr$ to $[c.len, c.len]$.
4: **if** c has children **then**
5: Sample($c.left$).
6: Sample($c.right$).
7: **end if**

2) Sample dihedral or concave/convex orientation: Each sub-chain and virtual link forms a triangle. In 2D, two different configurations have the same virtual link length: a concave orientation and a convex orientation (see Figure 5(a)). In 3D, a virtual link length can represent many configurations depending on the dihedral angle between its triangle and its parent's triangle (see Figure 5(b)). Thus, we also sample the orientation of the virtual link.

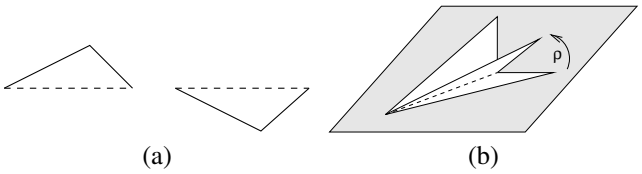


Fig. 5. (a) In 2D, the same virtual link represents two configurations: a concave triangle and a convex triangle. (b) In 3D, the same virtual link represents many configurations with different dihedral angles ρ between the sub-chain's plane and its parent's plane.

3) Calculate joint angles: Consider the joint angle θ between links a and b. Links a and b are connected to a virtual link c to form a triangle. Let l_a, l_b, and l_c be the lengths of links a, b, and c, respectively. The joint angle can be computed using the law of cosines:

$$\theta = acos(\frac{l_a^2 + l_b^2 - l_c^2)}{2l_a l_b}). \quad (2)$$

B. Local Planning

Given two configurations, q_s and q_g, a local planner attempts to find a sequence $\{q_s, q_1, q_2, \ldots, q_g\}$ of valid configurations to transform q_s into q_g at some user-defined resolution. Here we describe a simple local planner that uses a straight-line interpolation in the reachable space to determine the sequence of configurations. While this is certainly not the only local planning scheme possible, we find it performs well in practice.

For each virtual link, this local planner interpolates between its length in q_s and its length in q_g to get a sequence of lengths. We then must determine the virtual link's orientation sequence (i.e., concave/convex in 2D and the dihedral angle in 3D) to fully describe the sequence of configurations. In 2D, if the orientation is the same in q_s and q_g, we keep it constant in the sequence. If it is not the same, we must "flip" the links as described below. In 3D, we simply interpolate between the dihedral angle in q_s and the dihedral angle in q_g. We determine the validity of the sequence by checking that each configuration is closed (i.e., each virtual link's length is in its available reachable range) and collision-free.

1) Concave/convex flipping for 2D chains: Consider the first and last configurations of sub-chains in Figure 6. When the orientation is different, as in this example, we need to find a transformation from one to the other.

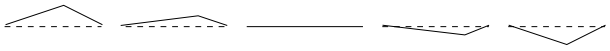


Fig. 6. Flip the links to change its orientation.

To flip the virtual link, we need to open the parent's virtual link enough so its children can change orientation while remaining in their available reachable range. In other words, at some point the parent's reachable range must be large enough to accommodate the summation of its children's reachable distance (i.e., allow the children to be "flat"). Such a constraint on reachable-distance can be easily handled by our method. There are other more powerful options. In our current implementation, we first calculate the available range for this minimum "flat" constraint. Then we sample an intermediate configuration where the virtual links are "flat" and try connecting it to both the start and goal configurations.

VI. Results and Discussion

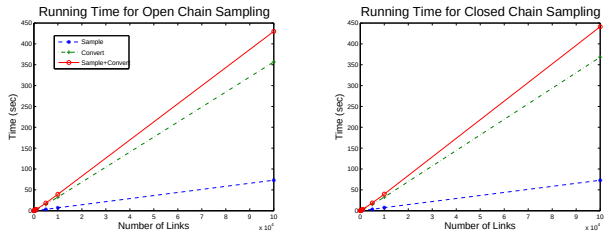
In this section we show how our PRD method performs in practice. We give a performance study in Section VI-A and

present the method in a complete motion planning framework in Section VI-B. All experiments are performed on a 3GHz desktop computer and implementations are compiled using gcc4 under linux. Our current implementation supports planar joints and spherical joints.

A. Performance Study

Here we study and compare the performance of our sampling algorithm both with and without collision detection.

In the first experimental set, we ignore collision detection. We measure the running time to generate 1000 configurations in 10 different environments for chains of size 10, 20, 50, 100, 200, 500, 1000, 5000, 10000, and 100000 links. In each environment, the articulated system is composed of different sized links ranging from 0.1 to 1.0. We compare the performance of generating both open and closed configurations using reachable-distance. As shown in Table I and in Figure 7, the running time has two parts: the time to sample and the time to convert the representation into traditional joint angles. Note that it takes much more time for conversion than for actual sampling. However, the performance of both scales well even for large numbers of links. These results show that sampling closed configurations is comparable to sampling open chain configurations that ignore closure constraints entirely. This demonstrates the advantage of using reachable distances to represent configurations over the traditional joint angle representation.



(a) (b)

Fig. 7. Running time to generate 1000 open (a) or closed (b) configurations in the reachable-distance space without considering collision. Each plot shows the total running time, the pure sampling time in reachable distance space, and the time to convert configurations into joint-angle space. Note that the running time for closed and open configurations are similar.

In the second set of experiments, we measure the performance when self-collision is considered. Again we measure the time to sample and convert 1000 self-collision-free configurations. We studied 6 different environments with chains of size 10, 20, 50, 100, 200 and 500. In each environment, we measured the running time to generate open-chain and closed-chain configurations in reachable-distance space. We also measure the performance using another closed-chain planner, KBPRM [2]. We stop an experiment after 10 hours. Table II and Figure 8 show the performance results.

Note that when considering collision detection, the performance of using PRD to sample an open-chain is comparable to the performance to sample a closed-chain, even though closed configurations involve more self-collision. This again shows that by directly sampling in the reachable space, PRD can sample closed configurations efficiently. This means that our method makes it practical to solve motion planning problems for large articulated systems with distance-related constraints. Also note that the PRD method out-performs the KBPRM method as expected.

Method	CD	10	20	50	100	200	500
PRD Open	N	0.02	0.04	0.11	0.25	0.53	1.50
	Y	0.17	0.56	2.96	14.30	80.54	877.34
PRD Closed	N	0.02	0.04	0.11	0.25	0.52	1.43
	Y	0.18	0.69	4.58	19.25	116.53	1626.17
KBPRM Closed	N	0.30	4.36	N/A	N/A	N/A	N/A
	Y	2.92	902.53	N/A	N/A	N/A	N/A

TABLE II
RUNNING TIMES OF DIFFERENT SAMPLING METHODS BOTH WITH AND WITHOUT COLLISION DETECTION.

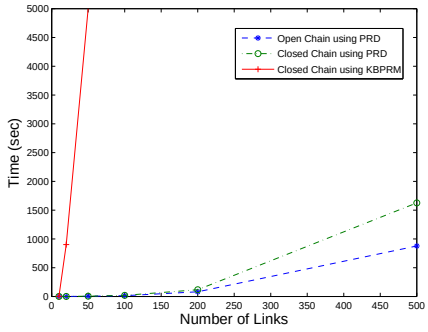


Fig. 8. Running time of open configuration sampling using reachable distances and closed configuration sampling using reachable distances all with collision detection as a function of the number of links. The performance for both methods is comparable since they are each dominated by collision detection.

B. Application in Motion Planning Frameworks

In this section, we show how we apply the sampling and local planning primitives to solve closed-chain problems.

1) Multiple robot grasping: Here, two articulated robotic arms with mobile bases grasp an object and work together to move it from one end of the environment to the other (see Figure 9(a)). Each arm is composed of 3 links. Considering the grasping object and the robot's base, there are 8 dof in total. The robot must maneuver the object underneath the obstacle in the ceiling to solve the query.

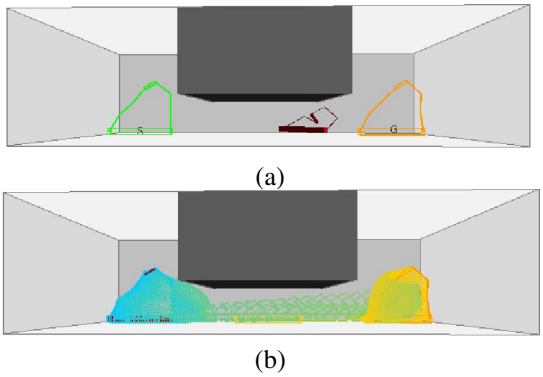
We use an incremental PRM method to construct the roadmap and solve the query. It begins with a small roadmap and incrementally generates more nodes until the query is solved. Our method required 40 nodes to solve the query. It took 58.4 seconds to build the map and found a valid path with 3100 intermediate configurations (see Figure 9(b)).

2) Single-loop closed-chain: A single-loop chain with 10 links of variable length must pass through a series of narrow passages to traverse the entire environment (see Figure 10). Again, we used incremental PRM map generation to build a roadmap until it solves the query. Our method successfully

Number of Links	10	20	50	100	200	500	1,000	5,000	10,000	100,000
Sample Open Chain (sec)	0.006	0.012	0.031	0.061	0.123	0.309	0.595	3.230	7.000	73.202
Sample Open Chain and Convert (sec)	0.017	0.039	0.123	0.268	0.560	1.863	3.219	18.254	39.449	429.958
Sample Closed Chain (sec)	0.006	0.012	0.032	0.061	0.124	0.304	0.599	3.170	7.296	73.008
Sample Closed Chain and Convert (sec)	0.018	0.042	0.130	0.267	0.554	1.512	3.267	18.504	39.823	441.217

TABLE I

TIME TO GENERATE 1000 OPEN/CLOSED CONFIGURATIONS IN THE REACHABLE-DISTANCE SPACE WITHOUT COLLISION DETECTION.



(a) (b)

Fig. 9. (a) Grasping experiment where 2 arms collaborate to transport an object across the environment. (b) Path's swept volume.

found a valid path with only 20 nodes in 96.2 seconds of computation time. Figure 10(b) shows the swept volume of the path with 1680 intermediate configurations.

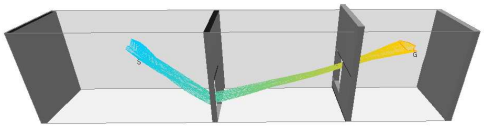


Fig. 10. Single-loop chain that must pass through a series of narrow passages to traverse the entire environment.

VII. CONCLUSION

We proposed a new method to plan the motion of closed-chain problems based on a hierarchical representation of the chain. It can be used to quickly generate closed configurations or determine that it is impossible to satisfy the closure constraints. We described this new representation and gave a sampling algorithm to generate closed configurations with complexity linear in the number of links. It can be used to significantly improve sampling-based planners for closed-chain systems by overcoming the difficulty of sampling closed configurations. We provide the necessary motion planning primitives (namely sampling and local planning) to implement sampling-based motion planners. Our experimental results show that our method is fast and efficient in practice, making the cost of generating closed and open configurations comparable.

REFERENCES

[1] J. Cortes, T. Simeon, and J. P. Laumond. A random loop generator for planning the motions of closed kinematic chains using PRM methods. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 2141–2146, 2002.

[2] L. Han and N. M. Amato. A kinematics-based probabilistic roadmap method for closed chain systems. In *Robotics:New Directions*, pages 233–246, Natick, MA, 2000. A K Peters. Book containts the proceedings of the International Workshop on the Algorithmic Foundations of Robotics (WAFR), Dartmouth, March 2000.

[3] L. Han, L. Rudolph, J. Blumenthal, and I. Valodzin. Stratified deformation space and path planning for a planar closed chain with revolute joints. In *Proc. Int. Workshop on Algorithmic Foundations of Robotics (WAFR)*, July 2006.

[4] M. Kallmann, A. Aubel, T. Abaci, and D. Thalmann. Planning collision-free reaching motion for interactive object manipulation and grasping. In *Eurographics*, 2003.

[5] L. E. Kavraki, P. Svestka, J. C. Latombe, and M. H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Trans. Robot. Automat.*, 12(4):566–580, August 1996.

[6] O. Khatib, K. Yokoi, K. Chang, D. Ruspini, R. Holmberg, and A. Casal. Vehicle/arm coordination and multiple mobile manipulator decentralized cooperation. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 546–553, 1996.

[7] K. Kotay, D. Rus, M. Vona, and C. McGray. The self-reconfiguring robotic molecule: Design and control algorithms. In *Proc. Int. Workshop on Algorithmic Foundations of Robotics (WAFR)*, pages 375–386, 1998.

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

[9] S. LaValle, J. Yakey, and L. Kavraki. A probabilistic roadmap approach for systems with closed kinematic chains. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 1671–1676, 1999.

[10] S. M. LaValle and J. J. Kuffner. Randomized kinodynamic planning. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 473–479, 1999.

[11] S. M. LaValle and J. J. Kuffner. Rapidly-Exploring Random Trees: Progress and Prospects. In *Proc. Int. Workshop on Algorithmic Foundations of Robotics (WAFR)*, pages SA45–SA59, 2000.

[12] J. P. Merlet. Still a long way to go on the road for parallel mechanisms. In *Proc. ASME Int. Mech. Eng. Congress and Exhibition*, 2002.

[13] R. J. Milgram and J. Trinkle. The geometry of configuration spaces for closed chains in two and three dimensions. *Homology Homotopy Appl.*, 6(1):237–267, 2004.

[14] A. Nguyen, L. J. Guibas, and M. Yim. Controlled module density helps reconfiguration planning. In *Proc. Int. Workshop on Algorithmic Foundations of Robotics (WAFR)*, 2000.

[15] J. H. Reif. Complexity of the mover's problem and generalizations. In *Proc. 20th Annu. IEEE Sympos. Found. Comput. Sci.*, pages 421–427, 1979.

[16] A. Singh, J. Latombe, and D. Brutlag. A motion planning approach to flexible ligand binding. In *7th Int. Conf. on Intelligent Systems for Molecular Biology (ISMB)*, pages 252–261, 1999.

[17] J. C. Trinkle and R. J. Milgram. Complete path planning for closed kinematic chains with spherical joints. *Int. J. Robot. Res.*, 21(9):773–789, 2002.

[18] D. Xie and N. M. Amato. A kinematics-based probabilistic roadmap method for high dof closed chain systems. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 473–478, 2004.

[19] J. H. Yakey, S. M. LaValle, and L. E. Kavraki. Randomized path planning for linkages with closed kinematic chains. *IEEE Transactions on Robotics and Automation*, 17(6):951–958, 2001.