Practical local planning in the contact space

Stephane Redon, Ming C Lin

Practical Local Planning in the Contact Space*

Stephane Redon and Ming C. Lin
Department of Computer Science
University of North Carolina, Chapel Hill, NC 27599-3175, USA
{*redon,lin*}*@cs.unc.edu*

***Abstract*— Proximity query is an integral part of any motion planning algorithm and takes up the majority of planning time. Due to performance issues, most existing planners perform queries at fixed sampled configurations, sometimes resulting in missed collisions. Moreover, randomly determining collision-free configurations makes it difficult to obtain samples close to, or on, the surface of C-obstacles in the configuration space. In this paper, we present an efficient and practical local planning method in contact space which uses "continuous collision detection" (CCD). We show how, using the precise contact information provided by a CCD algorithm, a randomized planner can be enhanced by efficiently sampling the contact space, as well as by constraining the sampling when the roadmap is expanded. We have included our contact-space planning methods in a freely available state-of-the-art planning library - the Stanford MPK library. We have been able to observe that in complex scenarios involving cluttered and narrow passages, which are typically difficult for randomized planners, the enhanced planner offers up to 70 times performance improvement when our contact-space sampling and constrained sampling methods are enabled.**

***Index Terms*— Motion Planning, Collision Detection, Contact Space.**

I. Introduction

Motion planning, one of the most important classical problems in algorithmic robotics, has been extensively investigated over the last three decades. Besides robotics, motion planning is also used in many other applications, including virtual prototyping, computational biology, computer animation, and surgical procedures.

One of the key steps in motion planning is proximity query, including collision detection or distance computation, which can take up to 90% of the planning time [6]. Due to performance issues, most existing planners use *discrete* proximity query algorithms, performing queries at fixed sampled configurations. Therefore, the resulting planning algorithms are subject to the possibility of generating a path that consists of non-collision-free segments, due to missed collisions between sampled nodes. Such phenomena typically occur when a robot moves rapidly into obstacles or when a robot moves through some very thin objects in the workspace. Moreover, randomly sampling the configuration space at discrete positions makes it difficult to sample the *contact space* (i.e. the surface of the C-obstacles in the configuration space [21]) and its neighborhood.

Recently some fast *continuous collision detection* (CCD) algorithms have been proposed for rigid objects (e.g. [28], [18]) and articulated models (e.g. [30]), and a clever *exact collision checking* (ECC) of robot paths has also been suggested for robot motion planning [33]. Our CCD algorithm achieves a roughly comparable runtime performance as the ECC method introduced by Schwarzer et al. [33]; therefore, we can effectively determine the connectivity of two sampled configurations. However, our CCD algorithm can further compute a precise contact information when a collision occurs (i.e. the time of first contact, the contact points, and the contact normals).

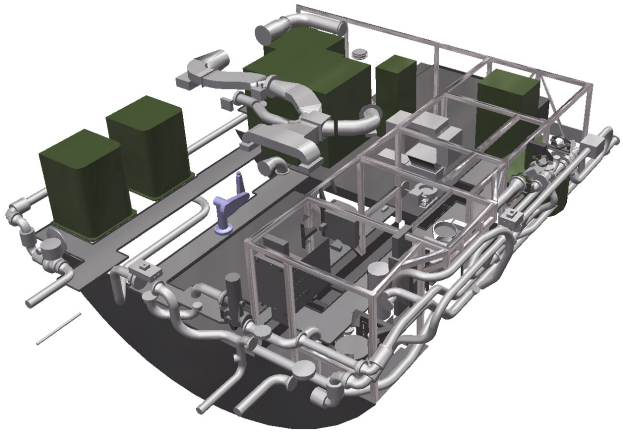

Fig. 1. *Planning the motion of a Puma Robot (800 triangles, 7 dofs) in a partial Auxiliary Machine Room model (117,000 triangles). Complex environments and tasks involving cluttered or narrow passages are typically difficult for randomized planning methods. Our local planning method efficiently samples the contact space and constrains the sampling to accelerate planning on these challenging scenarios. Our preliminary results indicate up to 70 times performance improvement when our contact-space planning method is incorporated with a state-of-the-art planning library.*

In this paper we show how, using this contact information, we are able to enhance a randomized planning method by not only efficiently sampling the contact space, but also by constraining the sampling when the roadmap is expanded. We show how such efficient contact-space local planning allows us to achieve up to 70 times performance improvement in complex scenarios involving cluttered or narrow passages, which are typically difficult for randomized planning methods. Moreover, as with the ECC algorithm, we are able to guarantee that the *entire* computed path is penetration-free.

The rest of the paper is organized as follows. Section 2 gives a brief survey of related work. Section 3 presents an overview of our CCD algorithm. Section 4 describes our algorithm for local planning in contact space using our CCD method. Section 5 presents an initial validation of our approach, which demonstrates the performance of our method on several challenging benchmarks. Finally,

* This project is supported in part by ARO, AMSO, DARPA, NSF, ONR/VIRTE and Intel.

we conclude in Section 6 with possible future research directions.

II. Related Work

In this section, we briefly survey prior work in continuous proximity queries and motion planning.

A. Continuous Collision Detection

A few algorithms have been proposed for continuous collision detection (CCD) between a rigid object and the simulated environment. These algorithms model the trajectory of the object between successive discrete time instances and check the resulting path for collisions. More specifically, there are four different approaches presented in the literature. The *algebraic equation solving* approach attempts to solve the CCD problem by explicitly solving the underlying CCD equations [4], [18], [27]. The *swept volume* approach is based on calculating the SV of moving objects explicitly and checking for collisions between the SV and the rest of the environment [1]. The *kinetic data structures* (KDS) approach is a scheduling scheme based on the usage of certificates, which signal when a collision might occur. By consuming certificates one by one, KDS determines whether a collision actually occurs [19]. The adaptive bisection approach employs a conservative separation test which ensures separation over time intervals, and it selectively subdivides the time interval that fails the test until the subdivided interval becomes smaller than some threshold along the time dimension [28], [33].

B. Motion Planning

The PRM method [17] was developed independently by Kavraki and Latombe [16] and Overmars and Švestka [24], [25]. The simplest PRM algorithms generate a set of configurations in the free space. The roadmap graph is used to generate a path between the start and goal configurations. However, the efficiency of these planners can degrade in configurations containing narrow passages or cluttered environments. In such cases, a significant fraction of randomly generated configurations are not in the free space. Moreover, it is difficult to generate a sufficient number of samples in the narrow passages or connect all the nodes in the free space using local planning methods. Many approaches have been proposed in the literature to overcome these problems. These include:

Sampling near the Obstacle Boundaries [2]: It samples the nodes from the contact space, i.e. the set of the configurations where the robot just touches one of the obstacles. It works well in many cases, but its performance is difficult to analyze.

Dilating the Free Space: The basic approach is to dilate the free space by allowing the rigid body to penetrate the obstacles by a small amount. The regions near these nodes are then re-sampled to find collision free configurations [15]. Recent approaches include iteratively generating new paths while progressively reducing the allowed penetration threshold [11].

Information about the Environment [14], [25]: These algorithms make use of information known about the environment, including executing random reflections at the C-obstacle surfaces [14] and adding "geometric" nodes for non-articulated robots near the boundaries of the obstacles in the workspace [25].

Sampling Based on the Medial Axis: The key idea of some recent work is to generate nodes that lie on the medial axis of the workspace or the free space [13], [26], [35], [36]. The resulting path typically corresponds to a set of points that are furthest from the obstacle boundaries and have maximum clearance. Medial axis also has been used for many earlier motion planning algorithms [3], [7], [8], [21], [23], though its practical use has been limited because of the difficulties in accurately computing the medial axis or generalized Voronoi diagrams.

Our local planning method takes advantage of the contact information provided by our CCD algorithm, thereby sharing a similar philosophy with sampling configurations near or on obstacle boundaries [2], [14], [25]. As in the SBL planner, we test paths at runtime while building the roadmap, and thus add nodes in the contact space at runtime. Furthermore, we use the precise contact information to *constrain* the sampling (cf. Section IV). Although previous approaches have proposed to locally constrain the planning using the neighboring geometry [10], the efficient continuous collision detection method that we use makes such an idea practical in complex scenarios, in particular by adding the possibility to efficiently constrain the motion of robots which come *into contact* with the environment.

Recently, Garber and Lin [12] proposed a new approach called "constraint-based planning", that transforms the motion planning problem into "constraint-based dynamic simulation". Typical geometric constraints in constraint-based planning include non-penetration with obstacles. Our approach can also be readily integrated into constraint-based planning [12], where constraint-based dynamics is used. Finally, we note that several authors have studied motion planning with contact constraints [9], [37], [22]. Our approach differs from this line of work in that we do not attempt to determine a formal or complete representation of the contact space and its topology. Instead, we locally sample and constrain in the contact space as needed. This allows us to efficiently handle challenging scenarios which involve articulated robots and complex environments with cluttered and narrow passages, with any number of degrees of freedom and any number of simultaneous contact points.

III. Continuous Collision Detection for Articulated Models

The main idea of our approach is the following. As a preprocessing step, we compute for each link a single line swept sphere (LSS) which bounds the link, as well as a *hierarchy* of oriented bounding boxes (OBBs) which enclose the actual rigid geometry of the link (an unstructured soup of triangles). Similarly, we compute one axis-aligned bounding box and one OBB-hierarchy per object in the environment. At runtime, we detect collisions between the moving robot and the environment in two stages:

Dynamic Bounding-Volume Hierarchies Culling: Given two successive configurations of the articulated model, we compute an arbitrary in-between path from the initial to the final configuration. Given the continuous path for each link, we use interval arithmetic to compute an axis-aligned bounding box (AABB) which encloses the volume swept by the link during its in-between motion. These dynamic AABBs are used to efficiently cull away some of the links-objects pairs that do not collide during the robot motion, using the pre-computed AABBs for the environment.

Exact Contact Computation: For each potentially colliding link-object pair identified by the previous stage, we further cull away portions of the geometries of the link and the environment object by using a continuous OBB overlap test based on interval arithmetic. For the remaining portions of geometry (i.e. triangles), we generate and solve a list of elementary collision detection equations between polyhedral features.

Overall, our algorithm can perform continuous collision detection at nearly interactive rates for articulated models in complex environments. When a collision occurs, the algorithm provides the contact features efficiently and robustly, and can thus guarantee collision-free paths. For more details, we refer the reader to [30].

IV. Local Planning in the Contact-Space

A. Overview

The PRM algorithms typically involve three main steps: (a) generate a list of samples in the configuration space; (b) use collision detection or distance computation algorithms to select the samples lying in the free space; (c) try connecting the samples in the free space by local planning methods and build a roadmap graph. Traditionally, connecting two sampled configuration in C-space has often been performed using either discrete collision detection methods, which sample a path connecting the two nodes (most often a straight line) with some finite resolution, or exact collision checking [4], [33]. To the best of our knowledge, a practical continuous collision detection method with contact information has never been applied to the motion planning problem. Given the contact information provided by our CCD algorithm, we are able to enhance the local planning step in the following ways:

Contact-space sampling: Since continuous collision detection allows us to compute the time of impact as well as the contact features (contact position and contact normal), we are able to efficiently and robustly sample the contact space, and not only the free space.

Constrained sampling: Given the precise contact information provided by our CCD algorithm, we are able to determine a piecewise-linear characterization of the local tangent space of C-obstacles around the nodes in the contact space. We use this piecewise-linear characterization to constrain the search of a new node when expanding the roadmap from a node belonging to the contact space.

B. Contact-Space Sampling

Whenever a path connecting a pair of sampled nodes is examined during local planning, we test each edge (q_i, q_{i+1}) connecting two successive configurations on the path using our CCD algorithm. Note that the nodes themselves have already been tested by the discrete collision checker, and are known to be in free space.

Whenever an edge (q_i, q_{i+1}) is invalid because it does not lie entirely in free space, *i.e.* because there happens to be a contacting configuration q_c on the path used to perform CCD between q_i and q_{i+1}, we delete the edge (q_i, q_{i+1}), but add the new node q_c to the roadmap, as well as the edge (q_i, q_c). Except for the newly added node q_c, the edge (q_i, q_c) lies entirely in the free space. Note that, since our CCD algorithm can detect simultaneous contact points, we are able to sample the lower-dimensionality portions of the contact space as well.

C. Constrained Sampling

Assume a configuration q_c lying in the contact space has been added to the roadmap. This configuration might involve several contact points $\mathbf{P}^1(\mathbf{q}), \ldots, \mathbf{P}^m(\mathbf{q})$ between the robot and the environment, which depend on the configuration $\mathbf{q}$ of the robot. Let $\mathbf{n}^1, \ldots, \mathbf{n}^m$ denote the corresponding 3-dimensional contact normals in the workspace, and let us assume that these normals are directed towards the exterior of the obstacles.

In the workspace, the robot motion has to satisfy m simultaneous local non-penetration constraints:

$$\mathbf{dP}^i(\mathbf{q_c}) \cdot \mathbf{n}^i \geq 0, \qquad 1 \leq i \leq m,$$

where $\mathbf{dP}^i(\mathbf{q_c})$ is the small variation in the position of $\mathbf{P}^i$ corresponding to a small variation $\mathbf{dq}$ in the configuration $\mathbf{q_c}$ of the robot. Each of these m non-penetration constraints can be easily expressed in the configuration space as $\mathbf{dq} \cdot \tilde{\mathbf{n}}^i \geq 0$, where $\tilde{\mathbf{n}}^i$ is the n-dimensional outward normal defining the orientation of the local tangent hyperplane in the configuration space, at the position $\mathbf{q_c}$. The components of the $\tilde{\mathbf{n}}^i$ vectors correspond to the components of the well-known $m \times n$ robot Jacobian $\mathbf{J}$ [31], [29], and the global constraint on the variation $\mathbf{dq}$ is $\mathbf{Jdq} \geq \mathbf{0}$.

The m non-penetration constraints define a polyhedral *set of valid variations*, *i.e.* the set of variations which satisfy all the constraints simultaneously. Provided the robot in the configuration q_c is in a consistent state with no interpenetration, this set is non-empty. Because we use linear constraints, the set of valid variations is only a local piecewise-linear characterization of the *valid space* (i.e. the union of the free space and the contact space [21]) around the configuration $\mathbf{q_c}$. However, this set of valid variations can be advantageously used for local planning. At runtime, whenever one tree is to be expanded from one configuration q, we check whether this configuration belongs to the contact space or not. When it does, we use the set of valid variations to help determine a new node q_{new}, by projecting the *intentional variation* on the set of valid variations. Precisely, whenever a new *tentative* node q_t is randomly chosen in the ρ-neighborhood of the contact-space node q, we perform the following operations *before* checking its validity with a discrete collision checker: (a) compute the tentative variation $\mathbf{dq_t} = \mathbf{q_t} - \mathbf{q}$; (b) project

the tentative variation $\mathbf{dq_t}$ on the set of valid variations to obtain the new variation $\mathbf{dq}$; (c) set the new node to $\mathbf{q_{new}} = \mathbf{q} + \mathbf{dq}$. The projection of the intentional variation $\mathbf{dq_t}$ onto the polyhedral set of constraints can be performed using various metrics, including the Euclidean and kinematic ones [29]. Whichever metric is chosen, we use the Wilhelmsen algorithm to compute the projection [34].

We note that the combination of contact-space sampling and constrained sampling allows us to *accumulate* contact-space constraints: whenever a new contact-space node q_c is determined by extending a node q_i, we check whether the contact-space constraints of q_i, if any, are still valid in the new configuration q_c. This occurs for example when the robot comes in contact with an obstacle while sliding along another one. This contributes to further sample the difficult, lower-dimensionality portions of the contact space.

V. Results

In this section, we present two sets of benchmarks that demonstrate the potential of our approach. While the first benchmark is rather simple from a collision detection point of view [30], both benchmarks are actually complex in a motion planning context.

We have integrated our continuous collision algorithm with MPK, the motion planning kit from Stanford which includes SBL, a single-query path planner with lazy collision checking [32], [33]. Although collision detection in the original MPK involves checking *entire* paths (and not only discrete configurations), no precise contact information is determined, as a "yes or no" answer only is provided. We have replaced the exact dynamic checking with our computation of the contact features described in Section 3 to enable contact-space sampling. We have also included an efficient projection algorithm [29] to handle the simultaneous contact-space constraints, when performing the constrained sampling. As the original MPK planner, the contact-space methods have been implemented in C++. The benchmarks have been performed on a 2.8 GHz dual-processor Pentium 4 PC with 2 Gb of RAM. We note that both sets of benchmarks involve the possibility for the robot to reach configurations with simultaneous contact points and constraints, which are handled adequately by our contact-space planning methods.

For each set of benchmarks, we have varied the difficulty of the problem by modifying the scale of the mobile robot, so as to increase the difficulty of sampling the narrow passages (i.e. increasing the narrowness in configuration space). For each scale tested, we have conducted ten tests and measured the time required to plan the motion for each test, as well as the total number of nodes present in the roadmap when the planner terminates. We then average the results over the ten tests for each scale. For each test, the planner is queried twice on the same pair of initial and final configurations: the first time with the default MPK settings, and the second time with contact-space planning enabled.

Tuning the parameters of a PRM algorithm to achieve optimal performance is not simple. In particular, a good choice of the size of the sampling window ρ depends heavily on the nature of the obstacles in the environment, especially when narrow passages are present. Because MPK contains an adaptive sampling strategy, and because our method is intrinsically adaptive (since we determine configurations on the boundaries of the C-obstacles) and requires no such tuning, we have not measured the impact of varying ρ in the benchmarks in this initial validation. The impact of parameter tuning in MPK will be measured in future benchmarking.

A. Single Rigid Body

This benchmark is derived from the classic *peg-in-a-hole* benchmark. A single rigid body has to go from one side of a wall to the other side (cf. Fig. 4). The rigid body is a L-shaped object contained in a $3 \times 3 \times 3$ cube. The dimensions of each of the three branches of the shape are $0.5 \times 0.5 \times 3$ for the robot scale of 1.0. The width of the square hole is 6, while the thickness of the obstacle is 11.5. The task becomes difficult when the robot scale tends towards 2, as the rigid body dimensions tend to the width of the hole. Table I provides a performance comparison when the scale of the rigid body varies. For each scale, the total time (in seconds) to perform the query and the final number of nodes in the roadmap are given for both methods (with and without contact-space planning). The results are also plotted in graphs shown in Figure 2.

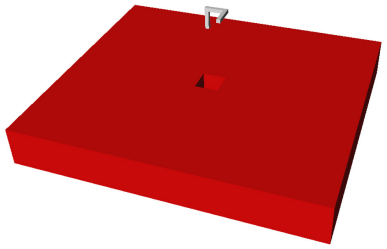

Fig. 4. *A single rigid body must go from one side of the obstacle to the other through the opening (robot scale: 1.85).*

In the most difficult benchmark, enabling both contact-space sampling and constrained sampling allows us to determine a path within a few minutes, providing a factor of up to 20 times speed-up. As expected, much fewer nodes are required to determine a path when contact-space planning is enabled.

B. Puma Robot and Auxiliary Machine Room

The second benchmark involves a Puma robot (800 triangles, 7 dofs) moving in a partial Auxiliary Machine Room model (117,000 triangles, cf. Fig. 1 and Fig. 3). While the initial configuration is in uncluttered free space, the final configuration lies in a narrow passage of the configuration space (*cf* Fig. 3).

We have used the same benchmarking procedure as with the rigid body. Table II provides a performance comparison as the scale of the Puma robot varies. For each scale, the total time (in seconds) to perform the query and the final

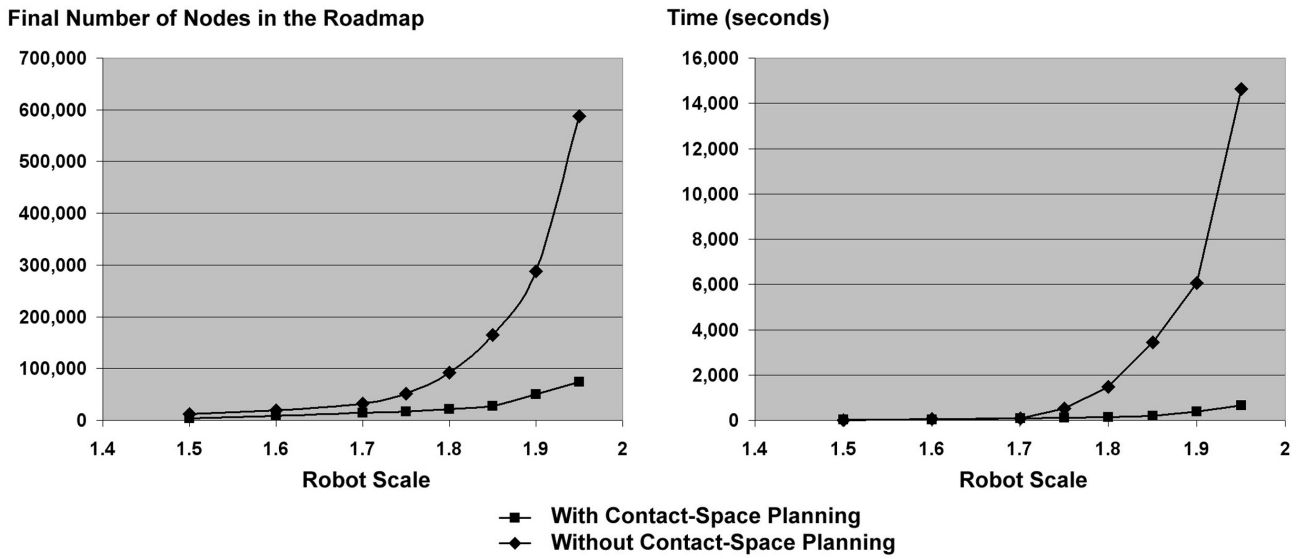

Fig. 2. *Benefits of our contact-space planning method when the scale of the rigid body increases. In the most difficult benchmark, enabling both contact-space sampling and constrained sampling allows us to determine a path within a few minutes, providing up to 20 times performance gain. As expected, much fewer nodes are required to determine a path when contact-space planning is enabled.*

TABLE I

SINGLE RIGID BODY

Robot Scale		1.5	1.6	1.7	1.75	1.8	1.85	1.9	1.95
Nodes	With Contact-Space Planning	3,820	8,476	14,490	16,889	22,032	27,942	50,621	73,677
	Without Contact-Space Planning	11,736	19,554	32,424	51,124	92,263	164,337	288,242	587,794
	Ratio	3	2.3	2.3	3	4	5.8	5.7	8
Time (seconds)	With Contact-Space Planning	15	37	78	104	144	194	393	650
	Without Contact-Space Planning	13	55	88	529	1,473	3,437	6,060	14,640
	Ratio	0.85	1.5	1.1	5	10	19	16	23

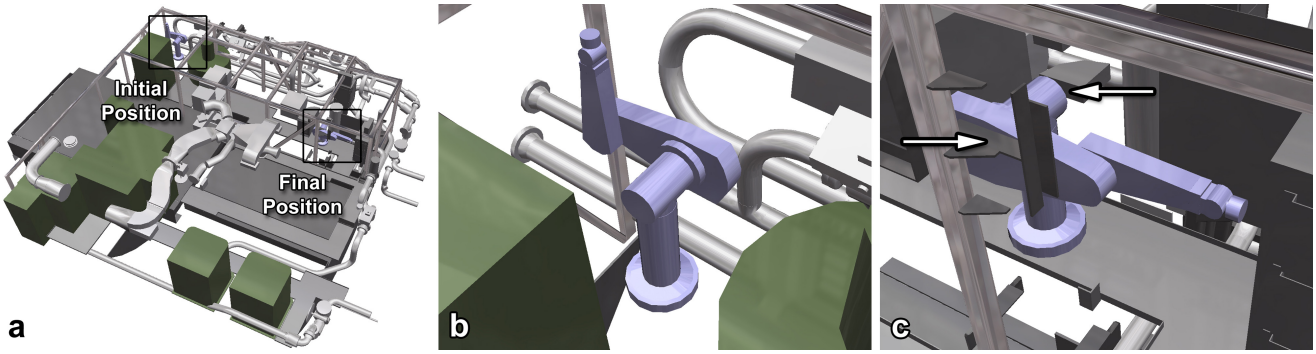

Fig. 3. *Planning the motion of a Puma Robot (800 triangles, 7 dofs) in a partial Auxiliary Machine Room model (117,000 triangles). The Puma robot must go from an initial uncluttered position (b) to a final position close to environment obstacles (c, close-by obstacles indicated by arrows). The scale of the Puma robot in this figure is 1.095, in which case enabling contact-space planning leads to about 70 times performance improvement.*

TABLE II

PUMA ROBOT

Robot Scale		0.1	0.4	0.7	1.0	1.03	1.06	1.075	1.095
Nodes	With Contact-Space Planning	109	244	591	4,717	5,246	12,232	11,785	22,202
	Without Contact-Space Planning	142	260	416	5,802	11,952	37,513	162,636	1,060,191
	Ratio	1.30	1.06	0.70	1.23	2.27	3.07	13.30	47.75
Time (seconds)	With Contact-Space Planning	0.80	1.33	3.86	45	52	128	122	244
	Without Contact-Space Planning	1.29	1.95	8.85	62	75	135	889	17,265
	Ratio	1.61	1.47	2.29	1.38	1.44	1.05	7.28	70.75

number of nodes in the roadmap are given for both methods (with and without contact-space planning). We observe that, when the scale of the robot is small and few passages seem narrow (the "general" case), the planner performs similarly whether contact-space planning is enabled or not. As the scale of the robot increases, however, and the robot in its final configuration is very close to the environment, our contact-space planning method enables us to greatly improve the performance of the planner and determine a path within just a few minutes, providing up to 70 times performance gain in the most difficult benchmark. Once again, much fewer nodes are required to determine a path when our contact-space planning is enabled.

VI. Conclusion and Future Work

We have presented a practical contact-space planning method using continuous collision detection (CCD). We have shown how an efficient CCD algorithm allows us to enhance a randomized planner by not only effectively sampling the contact space, but also by constraining the sampling when the roadmap is expanded. We have tested the enhanced planner in scenarios involving cluttered and narrow passages, which are typically difficult for randomized planners, and observed up to 70 times performance improvement when adding our contact-space sampling and constrained sampling methods.

Future work includes performing extensive benchmarks and a formal analysis to better evaluate the potential of the approach. Canny showed that the complexity of the basic motion planning problem is a single exponential function of the dimension of the configuration space [5]. By planning in the contact space, our method allows us to locally lower the dimensionality of the problem. We would like to further study the complexity of such a technique. Also, we would like to investigate how contact-space planning can complement other randomized approaches, such as Rapidly-exploring Random Trees [20]. Finally, we would like to analyze the potential of our contact-space planning method in other applications, including virtual prototyping, manipulation tasks, and dynamic scenarios. In particular, being able to rapidly compute the contact information potentially allows us to efficiently handle friction in planning scenarios.

Acknowledgment

The authors wish to thank Stanford University for the MPK library.

References

[1] K. Abdel-Malekl, D. Blackmore, and K. Joy. Swept volumes: Foundations, perspectives, and applications. *International Journal of Shape Modeling*, 2002.

[2] N. Amato, O. Bayazit, L. Dale, C. Jones, and D. Vallejo. Obprm: An obstacle-based prm for 3d workspaces. *Proceedings of WAFR98*, pages 197–204, 1998.

[3] J. Canny and B. R. Donald. Simplified Voronoi diagrams. In *ACM Symposium on Computational Geometry*, pages 153–161, 1987.

[4] J. F. Canny. Collision detection for moving polyhedra. *IEEE Trans. PAMI*, 8:200–209, 1986.

[5] J. F. Canny. The complexity of robot motion planning. *MIT Press*, 1988.

[6] H. Chang and T. Li. Assembly maintainability study with motion planning. In *Proceedings of International Conference on Robotics and Automation*, 1995.

[7] H. Choset. Nonsmooth analysis, convex analysis and their applications to motion planning. *International Journal of Computational Geometry and Applications*, 1997.

[8] H. Choset, I. Konukseven, and A. Rizzi. Sensor based planning: Using a honing strategy and local map method to implement the generalized voronoi graph. *SPIE Mobile Robotics*, 1997.

[9] G Dakin. Fine-motion planning for robotic assembly in local contact space. *Ph.D. Dissertation, University of Massachusetts at Amherst*, 1994.

[10] B. Faverjon and P. Tournassoud. A local based approach for path planning of manipulators with a high number of degrees of freedom. *In Proc. IEEE Int'l Conf. on Robotics and Automation (ICRA'1987), pages 995-1001, Raleigh, NC*, March-April 1987.

[11] E. Ferré and J.-P. Laumond. An iterative diffusion algorithm for part disassembly. *In Proc. IEEE Int'l Conf. on Robotics and Automation (ICRA'2004), pages 3149-3154*, April-May 2004.

[12] M. Garber and M. Lin. Constraint-based motion planning for virtual prototyping. *UNC Technical Report*, 2002.

[13] L. Guibas, C. Holleman, and L. Kavraki. A probabilistic roadmap planner for flexible objects with a workspace medial-axis-based sampling approach. In *Proc. of IROS*, 1999.

[14] T. Horsch, F. Schwarz, and H. Tolle. Motion planning for many degrees of freedom - random reflections at c-space obstacles. *Proc. of IEEE International Conf. on Robotics and Automation*, pages 3318–3323, 1994.

[15] D. Hsu, L. Kavraki, J. Latombe, R. Motwani, and S. Sorkin. On finding narrow passages with probabilistic roadmap planners. *Proc. of 3rd Workshop on Algorithmic Foundations of Robotics*, pages 25–32, 1998.

[16] L. Kavraki and J. C. Latombe. Randomized preprocessing of configuration space for fast path planning. *IEEE Conference on Robotics and Automation*, pages 2138–2145, 1994.

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

[18] B. Kim and J. Rossignac. Collision prediction for polyhedra under screw motions. In *ACM Conference on Solid Modeling and Applications*, June 2003.

[19] D. Kirkpatrick, J. Snoeyink, and Bettina Speckmann. Kinetic collision detection for simple polygons. In *ACM Symposium on Computational Geometry*, pages 322–330, 2000.

[20] J.J. Kuffner and S.M. LaValle. Rrt-connect: An efficient approach to single-query path planning. *In Proc. IEEE Int'l Conf. on Robotics and Automation (ICRA'2000), pages 995-1001, San Francisco, CA*, April 2000.

[21] J.C. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, 1991.

[22] Q. Luo, E. Staffetti, and J. Xiao. Representation of contact states between curved objects. *In Proceedings of IEEE International Conference on Robotics and Automation*, April 2004.

[23] C. Ó'Dúnlaing, M. Sharir, and C. K. Yap. Retraction: A new approach to motion-planning. pages 207–220, 1983.

[24] M. H. Overmars. A random approach to motion planning. Report RUU-CS-92-32, Dept. Comput. Sci., Utrecht Univ., Utrecht, Netherlands, October 1992.

[25] M. H. Overmars and P. Švestka. A probabilistic learning approach to motion planning. In *Algorithmic Foundations of Robotics*. A. K. Peters, Wellesley, MA, 1995.

[26] C. Pisula, K. Hoff, M. Lin, and D. Manocha. Randomized path planning for a rigid body based on hardware accelerated voronoi sampling. In *Proc. of 4th International Workshop on Algorithmic Foundations of Robotics*, 2000.

[27] S. Redon, A. Kheddar, and S. Coquillart. An algebraic solution to the problem of collision detection for rigid polyhedral objects. *Proc. of IEEE Conference on Robotics and Automation*, 2000.

[28] S. Redon, A. Kheddar, and S. Coquillart. Fast continuous collision detection between rigid bodies. *Proc. of Eurographics (Computer Graphics Forum)*, 2002.

[29] S. Redon, A. Kheddar, and S. Coquillart. Gauss' least constraints principle and rigid body simulations. In *IEEE International Conference on Robotics and Automation*, 2002.

[30] S. Redon, Young J. Kim, Ming C. Lin, and Dinesh Manocha. Fast continuous collision detection for articulated models. In *Proceedings of ACM Symposium on Solid Modeling and Applications*, 2004.

[31] D. Ruspini and O. Khatib. Collision/contact models for the dynamic simulation of complex environments. In *IEEE/RSJ IROS97*, 1997.

[32] G. Sanchez and J.C. Latombe. A single-query bi-directional probabilistic roadmap planner with lazy collision checking. In *Int. Symposium on Robotics Research (ISRR'01), Lorne, Victoria, Australia*, 2001.

[33] F. Schwarzer, M. Saha, and J.-C. Latombe. Exact collision checking of robot paths. In *Workshop on Algorithmic Foundations of Robotics (WAFR)*, Dec. 2002.

[34] D. R. Wilhelmsen. A nearest point algorithm for convex polyhedral cones and applications to positive linear approximations. 1976.

[35] Steven A. Wilmarth, Nancy M. Amato, and Peter F. Stiller. Maprm: A probabilistic roadmap planner with sampling on the medial axis of the free space. *IEEE Conference on Robotics and Automation*, pages 1024–1031, 1999.

[36] Steven A. Wilmarth, Nancy M. Amato, and Peter F. Stiller. Motion planning for a rigid body using random networks on the medial axis of the free space. *Proc. of the 15th Annual ACM Symposium on Computational Geometry (SoCG'99)*, pages 173–180, 1999.

[37] J. Xiao and X. Ji. On automatic generation of high-level contact state space. *International Journal of Robotics Research, 20(7):584–606*, July 2001.