Asymptotically-optimal Path Planning for Manipulation using Incremental Sampling-based Algorithms

Alejandro Perez Sertac Karaman Emilio Frazzoli Seth Teller Matthew R. Walter

I. Introduction

Most manipulation platforms are composed of robotic arms with several degrees of freedom, for which the motion planning problem involves exploring a high-dimensional configuration space. Moreover, planning problems for grasping and manipulation of complex objects require invoking computationally-expensive collision checking procedures several times. Consequently, existing planners are unable to identify high-quality (low-cost) solutions in a reasonable amount of computation time.

In this talk, we present an algorithm that overcomes these difficulties by augmenting the asymptotically-optimal RRT* algorithm with a sparse sampling procedure, called the Ball Tree algorithm, and a memoization technique that speeds up the collision checking procedure. The proposed algorithm is specifically tailored for anytime computation. That is, the method quickly yields a feasible initial solution, and utilizes remaining computation time to improve the solution with the guarantee of almost-sure convergence to a globally-optimal solution.

We evaluate the algorithm through a series of Monte-Carlo simulation studies involving seven and fourteen degree of freedom manipulation planning problems using a realistic simulation environment. Simulation results suggest that the proposed algorithm provides significant improvements in both the quality of the first solution found as well as the final path that is executed by the robot, while incurring no substantial computational cost when compared to the RRT algorithm. We further demonstrate the algorithm on the PR2 platform for single-arm and dual-arm planning problems. A more elaborate discussion of the algorithms and the results presented in this talk is given by Perez et al. [1].

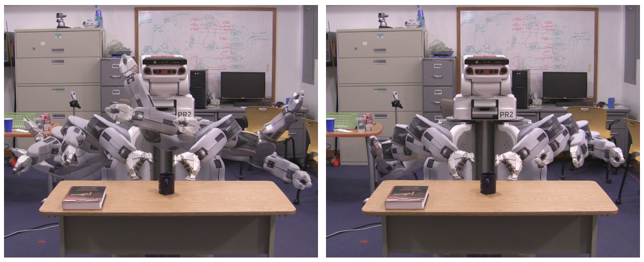
II. Algorithm

The RRT* algorithm, introduced by Karaman and Frazzoli [2], is an incremental sampling-based motion planning algorithm that offers the asymptotic optimality guarantee, i.e., almost-sure convergence to globally optimal solutions, which the RRT algorithm lacks, without incurring substantial computational overhead when compared to the RRT.

We implement the RRT* algorithm by delaying calls to the collision checking procedure until absolutely necessary.

Alejandro Perez, Seth Teller, and Matthew R. Walter are with the Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, USA {aperez, teller, mwalter}@csail.mit.edu

Sertac Karaman and Emilio Frazzoli are with the Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, MA, USA {sertac, frazzoli}@mit.edu



(a) RRT (b) BT+RRT*

Fig. 1. Given the goal of taking both arms of the PR2 from an initial pose underneath the table to the pre-grasp pose with the end effectors near the mug, (a) the RRT typically results a plan involving unnecessary actuation of several joints while (b) our method identifies more efficient plans.

During the extension phase of the RRT* algorithm, we sort each path in order of increasing cost and proceed to check the paths for collision in this order until a collision-free path is found. During the rewiring phase of the RRT* algorithm, we invoke the collision checking procedure only if the cost of the rewiring path is low enough to improve the cost of the rewiring vertex. See Perez et al. [1]. Although, in the worst case, this approach will result in checking all trajectories for collision, the authors have found in the experimental studies that on average only a few paths are checked for collision, significantly improving the running time of the RRT* algorithm in problems in which collision checking is computationally expensive.

The Ball Tree algorithm, presented by Shkolnik and Tedrake [3], is a sampling-based method similar to the RRT that approximates connected regions of free space with balls instead of points. Treated as sets of reachable points, the algorithm uses these balls to perform rejection sampling, resulting in trees that are sparser than those of the standard RRT while maintaining probabilistic completeness.

We propose a manipulation planning algorithm that offers two compelling advantages. Firstly, it is noticeably faster than conventional planners at identifying an initial, low-cost, feasible path to the goal in configuration space. Secondly, the algorithm is able to take advantage of available computation time to refine this solution towards an optimal one. We achieve these characteristics by combining the Ball Tree algorithm, which maintains sparse trees to efficiently reach the goal, the RRT* algorithm, which provides the anytime refinement of the tree, and a memoization method, which speeds up the collision checking procedure.

The proposed algorithm is called the BT+RRT* in this text. The details of this algorithm are given in Perez et al. [1].

III. Results

We evaluate the effectiveness of our algorithm through both simulation studies and experiments on the PR2 robotic platform. We first perform a Monte Carlo study to analyze the algorithm's performance in comparison to that of the RRT and RRT* on two different planning problems for the PR2 robot. The first involves solving for a configuration space path that brings a single, seven degree of freedom arm to a pre-grasp pose. In the second scenario, we consider jointly planning trajectories for both arms (see Fig. 1). The experiments utilize the OpenRAVE simulation environment [4].

A. *Single-Arm Scenario (Seven Degrees of Freedom)*

The results for the seven degree of freedom single-arm planning scenario are summarized in Fig. 2. As a result of the sparse tree structure provided by the Ball Tree method, the algorithm requires less time than the RRT and RRT* to identify an initial solution that, nonetheless, exhibits a low-cost nearly identical to that of the first RRT*solution. The BT+RRT* algorithm utilizes the remaining computation time to refine the solution, with a convergence very similar to that of the RRT*as evident in the lower plot.

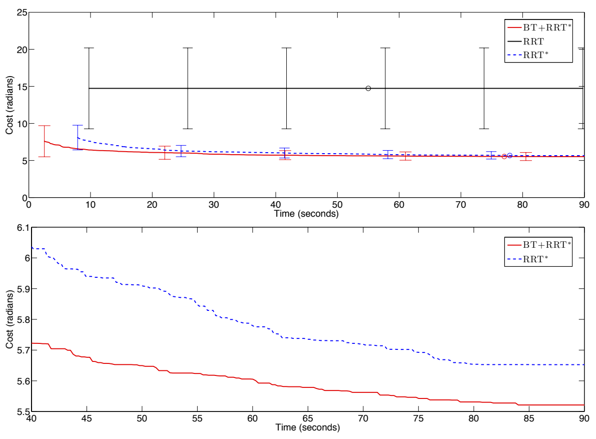


Fig. 2. Solution cost as a function of computation time, averaged over the set of single-arm Monte Carlo simulations for the three algorithms. Vertical bars indicate standard deviation over the 100 runs while open circles denote the average completion time. The bottom figure presents an inset view that compares the mean behavior of our algorithm with that of the RRT*.

B. *Dual-Arm Scenario (Fourteen Degrees of Freedom)*

TABLE I

Fourteen Degree of Freedom Monte Carlo Results

		BT+RRT*	RRT	RRT*
Success Rate (100 runs)		100.00%	25.00%	59.00%
First Solution	Time (s)	34.76 (60.06)	70.78 (82.90)	106.20 (108.65)
	Cost (rad)	9.82 (2.94)	21.00 (7.69)	10.03 (2.61)
Final Solution	Time (s)	374.65 (46.46)	263.16 (30.40)	380.82 (34.20)
	Cost (rad)	8.64 (1.95)	21.00 (7.69)	9.28 (2.14)
Time per Iteration (ms)		37.50 (4.65)	26.34 (3.04)	38.12 (3.42)

The results for this scenario are summarized in Table I. Allowing a maximum of 10,000 iterations, the RRT* was able to find a solution in 59 of the runs and the RRT was successful in only 25. The BT+RRT* algorithm identified a trajectory every time and, much like the seven degree of freedom simulations, returned an initial solution much sooner than the RRT and RRT*. The algorithm then refines the solution, exhibiting an improvement in cost that resembles that of the RRT*, both in terms of mean cost and variance. After 10,000 iterations, the BT+RRT* yields an average trajectory cost slightly better than that of the RRT*.

C. *PR2 Experimental Validation*

In addition to the Monte Carlo simulations, we utilized our algorithm to execute both the single-arm and dual-arm scenarios on the experimental PR2 platform. We demonstrated our planner together with the standard RRT approximately a dozen times for each of the two cases. Both algorithms were allowed 1000 iterations in the single-arm scenario and 2000 iterations in the dual-arm scenario. Fig. 1(b) presents a time lapse image that shows the typical trajectories that result from our planner. We compare this with the RRT solutions that typically require excessive arm motion. The consistency with which our algorithm plans efficient paths through configuration space supports the small variance in the lower cost solutions found in the Monte Carlo simulations. Videos that show single-arm and dual-arm planning with our algorithm on the PR2 robot are available at `http://ares.lids.mit.edu/manipulation_planning/`

IV. Conclusion

Incremental sampling-based motion planners such as the RRT are able to identify feasible motion plans quickly, making them appealing for manipulation. However, the resulting solutions are often far from optimal and the exploration of the space is commonly sacrificed to avoid computationally-expensive collision checking. This paper described a sampling-based planning algorithm that leverages the efficient planning capabilities of the Ball Tree algorithm together with the asymptotic optimality provided by the RRT*. Moreover, the algorithm delays checking paths for collision until it is absolutely necessary and leverages memoization to reduce its computation time. We employed Monte Carlo simulations to evaluate the algorithm's ability to provide low-cost solutions for high-dimensional planning problems in a timely fashion. We further demonstrated the algorithm's effectiveness on the PR2 robot.

Acknowledgements

The authors are grateful to Professors L.P. Kaelbling and T. Lozano-Perez and to Willow Garage Inc. for providing access to the PR2 experimental platform.

References

[1] A. Perez, S. Karaman, A. Shkolnik, E. Frazzoli, S. Teller, and M. Walter, "Asymptotically-optimal path planning for manipulation using incremental sampling-based algorithms," in *Proc. IEEE/RSJ Int'l Conf. on Intelligent Robots and Systems*, 2011.

[2] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *Int'l J. of Robotics Research*, vol. 30, no. 7, pp. 846–894, June 2011.

[3] A. Shkolnik and R. Tedrake, "Sample-based planning with volumes in configuration space," http://groups.csail.mit.edu/robotics-center/public_papers/Shkolnik11.pdf, 2011 (To be submitted).

[4] R. Diankov, "Automated construction of robotic manipulation programs," Ph.D. dissertation, Carnegie Mellon University, Robotics Institute, August 2010.