BIT*: Sampling-based Optimal Planning via Batch Informed Trees

Jonathan D. Gammell
Institute for Aerospace Studies
University of Toronto
Toronto, ON, Canada
jon.gammell@utoronto.ca

Siddhartha S. Srinivasa
The Robotics Institute
Carnegie Mellon University
Pittsburgh, PA, USA
siddh@cs.cmu.edu

Timothy D. Barfoot
Institute for Aerospace Studies
University of Toronto
Toronto, ON, Canada
tim.barfoot@utoronto.ca

***Abstract*—In this paper, we introduce initial work on an anytime optimal sampling-based planning algorithm, Batch Informed Trees (BIT*). BIT* unifies the developments of Optimal RRT (RRT*) and Fast Marching Trees (FMT*) while extending them with a heuristic. An overview of the algorithm and some initial results are presented, along with a discussion of ongoing future work. As is demonstrated, this new algorithm shows promise compared to RRT* and FMT* in terms of computational cost required to find equivalent solutions.**

I. Introduction

There has been a renewed interest in sampling-based planning algorithms that probabilistically find the *optimal* solution to a given planning problem. A recent major contribution was by Karaman and Frazzoli [7], who developed optimal versions of Probabilistic Roadmaps (PRMs) [8] (PRM*) and Rapidly-exploring Random Trees (RRTs) [9] (RRT*) by leveraging the statistical properties of random geometric graphs (RGGs) [10].

RRT* is an *anytime* algorithm that finds an initial solution and then asymptotically converges to the optimal solution. Janson and Pavone [6] use similar algorithmic principles in their Fast Marching Trees (FMT*) algorithm to process a *batch* of samples in order of increasing cost-to-come. This order removes the need for rewiring and is an example of dynamic programming [1] or Dijkstra's algorithm [2] for a Euclidean cost-function on an RGG. While FMT* calculates the optimal path for the given samples, it is not anytime and further improvement of the solution requires calculating a new tree through a denser set of samples.

In this paper, we present a novel sampling-based algorithm that unifies the anytime nature of RRT* with the ordered processing of FMT*. It does this in an efficient manner by using heuristics, as found in A* [5], and multiple batches of samples. While analysis is preliminary, initial results in $\mathbb{R}^2$ (Figs. 1, 2) and $\mathbb{R}^8$ (Fig. 3) show that it outperforms RRT* and FMT* in terms of computational cost to find equivalent results while still providing an anytime solution.

The remainder of this abstract is organized as follows. Section II informally presents the Batch Informed Trees (BIT*) algorithm, while Section III discusses ongoing work. A more detailed explanation of BIT* is available in [4].

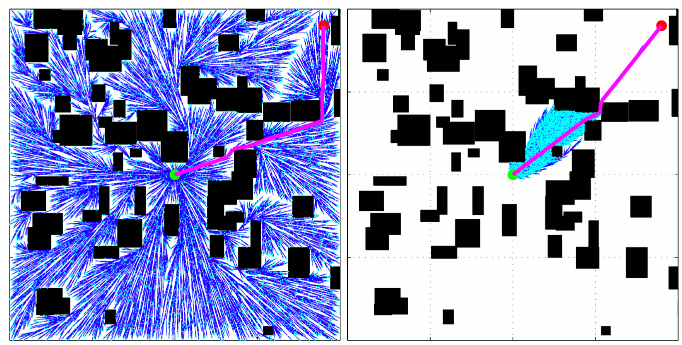

Fig. 1. An example of RRT* and BIT* run on a difficult random $\mathbb{R}^2$ world for the same arbitrary computational time. BIT*'s use of heuristics allows it to find a better solution ($c = 1.30$) faster than RRT* ($c = 1.54$) by performing its search in a principled manner that initially prioritizes low-cost solutions and then focuses on improving them. Cyan dots and blue lines represent vertices and edges, respectively, in the resulting tree.

II. Batch-informed Trees (BIT*)

BIT* combines the anytime nature of RRT* with the ordered expansion of FMT* with a heuristic multiple-batch approach. By processing the vertices in batches of size greater than one, BIT* is able to search in an efficient manner, like FMT*. By processing multiple batches, it is able to return solutions in an anytime manner, like RRT*.

Informally, BIT* works as follows. We start with a tree, $\mathcal{T}$, consisting of the initial states, $\{\mathbf{x}_{\mathrm{s}}\}$. A batch of $n_i = n'$ uniformly-distributed random samples are drawn from the obstacle-free problem space, X_{free}. These samples, the vertices in the tree, and the connection condition described by Karaman and Frazzoli [7] (i.e., a radius of connection or a number of near vertices) describe an RGG. We then build a shortest-path spanning tree through this RGG by incrementally adding edges between vertices on the existing tree and unconnected samples. The use of a heuristic allows each iteration to process the edge belonging to the best potential solution given our information. This prioritizes both growth towards the goal and the exploration of high-quality paths.

If the processed edge is collision-free, we add it to the tree and add any edges between it and nearby unconnected samples to our queue of edges. Depending on the heuristic used, or if this is not the first batch, we must also search for rewirings

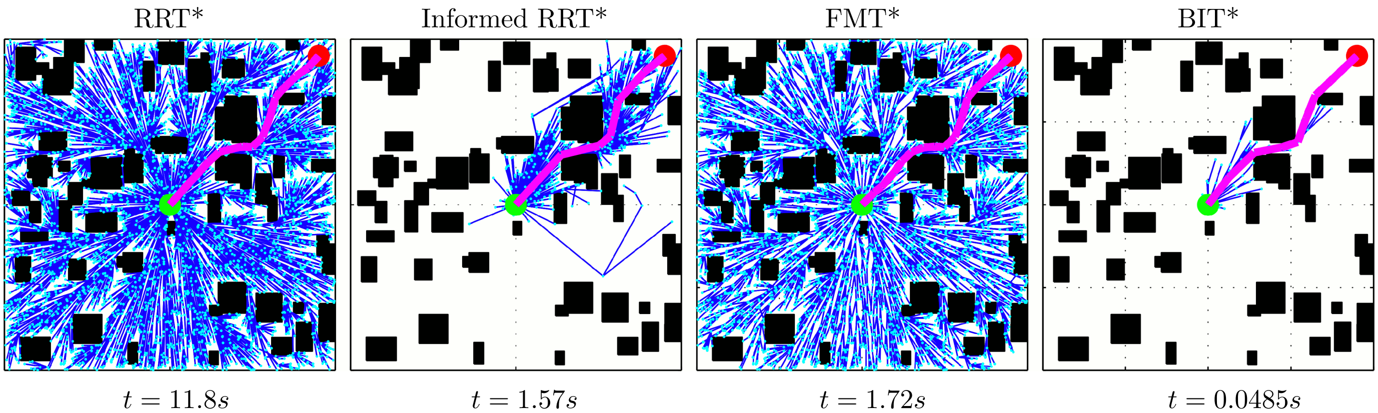

Fig. 2. An example of RRT*, Informed RRT* [3], FMT* ($n = 5000$) and BIT* run on a random $\mathbb{R}^2$ world. Each algorithm was run until it found a equivalent solution to FMT* ($c = 1.34$). BIT*'s use of heuristics allows it to find such a solution significantly faster ($t = 0.0485s$) than RRT* ($t = 11.8s$), FMT* ($t = 1.72s$) and Informed RRT* ($t = 1.57s$) by performing its search in a principled manner that initially prioritizes low-cost solutions and then focuses on improving them. Cyan dots and blue lines represent vertices and edges, respectively, in the resulting tree.

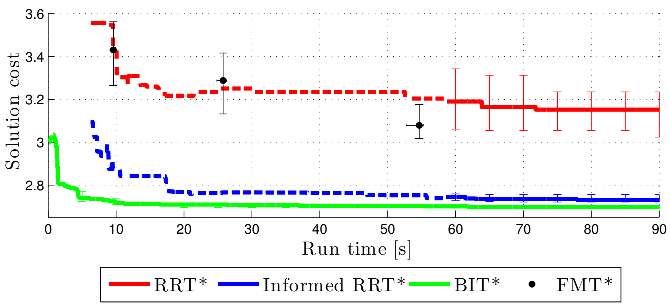

Fig. 3. The median solution cost versus run time for 20 different experiments on a typical random world in $\mathbb{R}^8$ for RRT*, Informed RRT*, BIT* with a batch size of 5000, and FMT* with $n = 1000$, 2500, & 5000. The dashed line represents a median calculated from a number of solutions between 10 and 20 and the solid line the median once all 20 runs had found a solution with error bars denoting a non-parametric 95% confidence interval. Note that BIT* outperforms all the other planners.

that improve the tree. As in RRT*, we only perform these rewiring *locally* and do not reevaluate the descendents. We continue processing this queue until it is empty.

If we exhaust the edge queue without finding a path to the goal, we generate n' new samples in X_{free} and update the existing tree from the combined set of samples $n_{i+1} = n_i + n'$ with an updated radius of connection.

We find a first solution when we add a goal state, $\mathbf{x}_{\text{g}}$, to the tree, but we continue to process the edge queue for as long as it could provide a better solution. If further improvement is still required when the edge queue is empty, we add a new batch of samples and search the resulting combined set of samples, $n_{i+1} = n_i + n'$. The new samples are not drawn from the entire planning domain but from the subset of the problem that contains possibly better solutions. This heuristically informed subset, $X_{\widehat{f}} \subseteq X_{\text{free}}$, is defined by the cost of the solution path [3]. This method maintains a uniform density over subdomain of the problem being searched, a common assumption for RGG properties.

Details are omitted here for brevity, but in practice we extend the basic BIT* algorithm described above with methods to generate the free samples and the edge queue in a *just-in-time* manner as well as graph and sample pruning [4].

III. Discussion & Conclusion

Initial qualitative results suggest that BIT* effectively combines the benefits of FMT* and RRT*. It does this for problems seeking to minimize path length by using Euclidean distance as a heuristic to process batches of samples.

Using a heuristic not only prioritizes better initial solutions, but it also helps focus future refinement to the subset of states that can improve the solution. For high-dimensional problems (e.g., manipulation planning) seeking to minimize path length, direct sampling of this subset is generally required as the probability of improving the solution with uniform global sampling goes to zero as the size of the problem increases or the quality of the solution improves [3].

Ongoing work focuses on performing thorough experimental and theoretical comparisons of BIT* to existing optimal sampling-based planning algorithms [4].

Acknowledgments

This research was funded by contributions from the Natural Sciences and Engineering Research Council of Canada (NSERC) through the NSERC Canadian Field Robotics Network (NCFRN), the Ontario Ministry of Research and Innovation's Early Researcher Award Program, and the Office of Naval Research (ONR) Young Investigator Program.

References

[1] R. E. Bellman. The Theory of Dynamic Programming. *Bulletin of the AMS*, 60(6):503–516, 1954.

[2] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1):269–271, 1959.

[3] J. D. Gammell, S. S. Srinivasa, and T. D. Barfoot. Informed RRT*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic. *IROS*, 2014. arXiv:1404.2334 [cs.RO].

[4] J. D. Gammell, S. S. Srinivasa, and T. D. Barfoot. BIT*: Batch informed trees for optimal sampling-based planning via dynamic programming on implicit random geometric graphs. Tech. Report TR-2014-JDG006, ASRL, University of Toronto, 2014. arXiv:1405.5848 [cs.RO].

[5] P. E. Hart, N. J. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *TSSC*, 4(2):100–107, Jul. 1968.

[6] L. Janson and M. Pavone. Fast marching trees: a fast marching sampling-based method for optimal motion planning in many dimensions. *ISRR*, Dec. 2013.

[7] S. Karaman and E. Frazzoli. Sampling-based algorithms for optimal Motion Planning. *IJRR*, 30(7):846–894, 2011.

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

[9] S. M. LaValle and J. J. Kuffner Jr. Randomized kinodynamic planning. *IJRR*, 20(5):378–400, 2001.

[10] M. Penrose. *Random Geometric Graphs*. Oxford University Press, 2003.