Asymptotically Near-Optimal Planning with Probabilistic Roadmap Spanners

James D. Marble and Kostas E. Bekris

***Abstract*— Asymptotically optimal motion planning algorithms guarantee solutions that approach optimal as more iterations are performed. Nevertheless, roadmaps with this property can grow too large and unwieldy for fast online query resolution. In graph theory there are algorithms that produce subgraphs, known as spanners, which have guarantees about path quality. Applying such an algorithm to a dense, asymptotically optimal roadmap produces a sparse, asymptotically near optimal roadmap. Experiments performed on geometric problems in SE(3) show that a large reduction in roadmap edges can be achieved with a small increase in path length (Fig. 1). Online queries can be answered faster with similar results in terms of path quality. Additional time savings result from applying the spanner algorithm incrementally so edges that do not increase path quality will never be added to the roadmap and will not be checked for collisions.**

I. Introduction

Roadmap planners [1] utilize an off-line phase to build up knowledge about the configuration space ($\mathcal{C}$-space). `PRM` attempts to connect sampled configurations to either a fixed number, k, of nearest neighbors or all configurations within a fixed radius ball centered at the new configuration. Higher quality paths can result from constructing larger, denser roadmaps that better sample $\mathcal{C}$ by investing more preprocessing time. However, it has been shown that a `PRM` using a fixed k will not converge to the optimal path [2].

The k-`PRM`* algorithm minimizes the number of neighbors each new sample has to be connected to while increasing path quality over time (i.e., asymptotic optimality) by making k a logarithmic function of the roadmap size. Roadmaps constructed with this variation will almost surely converge to an optimal solution [2]. While such roadmaps are desirable for their high path quality, their large size is problematic. Large roadmaps impose significant costs during storage, transmission, and query resolution. In order to relax optimality requirements while producing sparser roadmaps it is possible to use graph spanners, a tool from graph theory [3].

A graph spanner is a sparse subgraph. Given a weighted graph $G = (V, E)$, a subgraph $G_S = (V, E_S \subset E)$ is a t-spanner if, for all pairs of vertices $(v_1, v_2) \in V$, the shortest path between them in G_S is no longer than t times their shortest path in G. Because t specifies the amount of additional length allowed, it is known as the *stretch factor* of the spanner. There are many algorithms in the literature for constructing graph spanners, each with their own properties.

This work presents two approaches for integrating k-`PRM`* with graph spanners. In the first approach, the randomized $(2k-1)$-spanner algorithm [4] was chosen for application after a dense roadmap was constructed. This alternative provides optimal stretch along with reasonable size and linear time complexity. However, this algorithm operates on the entire graph and can not be applied incrementally as the dense graph is being constructed. In the second approach, an algorithm with slightly worse time complexity is interleaved with roadmap construction so that the entire dense roadmap need not be held in memory all at once.

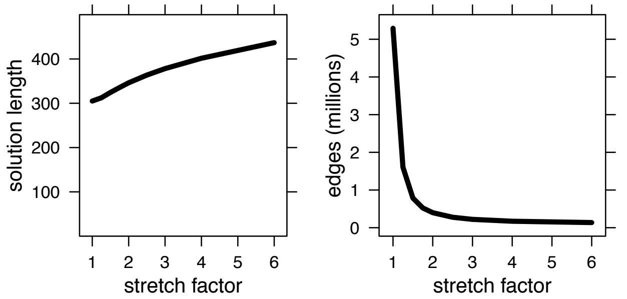

Fig. 1. An asymptotically near optimal solution provides a large decrease in roadmap density and solution query time. The larger the stretch factor, the sparser the roadmap. These results are averaged over 1000 random queries on 10 runs of 50,000 vertex roadmaps. The k-`PRM`* algorithm corresponds to a stretch factor equal to 1.

II. Sequential Approach

The sequential approach [5] can be broken down into two parts. First, a dense roadmap is constructed using k-`PRM`* and then passed through the randomized $(2k-1)$-spanner algorithm. This makes the roadmap sparse by removing edges that do not contribute much to path quality. Without the path quality guarantees provided by k-`PRM`*, there would be no guarantees on the path quality of the sparse roadmap.

Results for 100 random queries on roadmaps with 20,000 vertices show significant reduction in roadmap density and query resolution time with small increases in path quality.

1) Space Requirements: While each spanner contains the same vertices as the original roadmap, the space required for connectivity is reduced by up to 85%. Environments with a more connected free space had a larger reduction in the number of edges because fewer are needed for connectivity.

2) Path Quality: The length of the resulting paths increases as the number of edges in the spanner is reduced. For random start and goal configurations, the extra cost is smaller than the worst case guaranteed by the spanner algorithm.

Path quality degradation in the spanner roadmap is plotted in Fig. 3 as a function of the original path length and k. The worst degradation happens for short paths, where taking a detour of even a single vertex can increase the path length

J. D. Marble and K. E. Bekris are with the Department of Computer Science and Engineering, University of Nevada, Reno, 1664 N. Virginia St., MS 171, Reno, NV, 89557 `{jmarble,bekris}@cse.unr.edu`

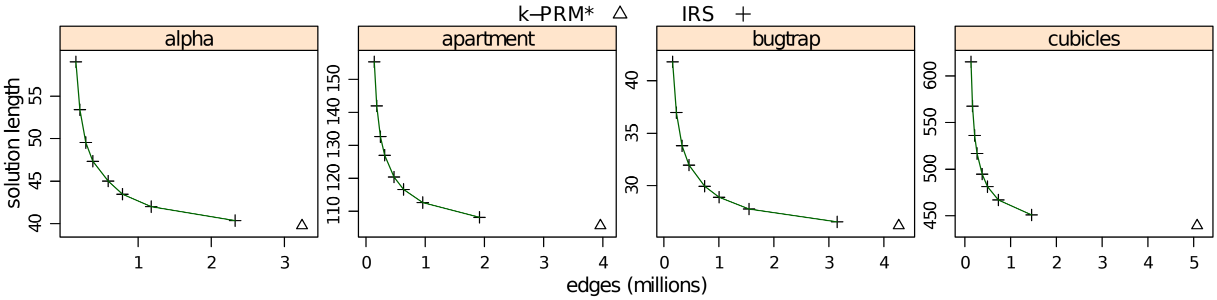

Fig. 2. By adjusting the stretch factor parameter, `IRS` provides a range of trade-offs between roadmap density and solution quality. From left to right, the stretch factors shown are 6, 4, 3, 2.5, 2, 1.75, 1.5, and 1.25. Note that `IRS` with a stretch factor of 1 is equivalent to k-`PRM`*.

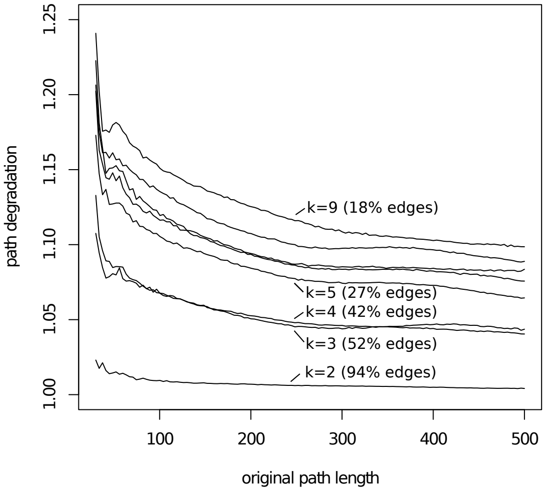

Fig. 3. Mean path length degradation as a function path length with 20,000 vertices. Larger stretch factors result in fewer edges and longer paths. The largest path degradation occurs at smaller path lengths.

by a large factor. The path length for the 17–spanner ($k = 9$) in this experiment increases by up to 200% for paths that were originally short. The mean increase is less than 25%.

3) Query Resolution Time: This time includes the effort to connect the start and goal to the roadmap and perform an A* search. The connection time is only affected by the number of vertices in the roadmap. Since each spanner has the same number of vertices as the original roadmap, the discrete search dominates the query resolution time for large roadmaps. Reducing the number of edges in the roadmap lowers the query resolution time by up to 75%.

III. Incremental Approach

The Incremental Roadmap Spanner (`IRS`) [6] takes the idea of the sequential approach one step further by interleaving roadmap and spanner construction. When the algorithm adds an edge, the spanner algorithm can reject it before the edge is checked for collision and added to the roadmap.

The method used to filter edges is the same used by the the naïve greedy spanner algorithm [3] and Useful Cycles [7]. A length-limited graph search is used to determine if an existing path between two neighbors is no more than t times the weight of the proposed edge. If so, the edge is rejected. This criterion is applied during k-`PRM`* construction before an edge is checked for collision. If an edge is filtered, the expensive collision check step for this edge is avoided. Otherwise, the k-`PRM`* algorithm is unchanged. The density of the roadmap is reduced, resulting in lower space requirements and query resolution time. Additionally, construction time is reduced by up to 50 percent. Fig. 1 shows the result of applying this approach in one environment.

For each query, there was a simple attempt at path smoothing. Nonconsecutive vertices on the path that were near each other were tested for connectivity. If they could be connected, then the path is shortened by removing intervening vertices. This greedy and local method for smoothing brought the quality of paths on the sparse graph closer to that on the dense graph making the `IRS` approach even more favorable.

IV. Discussion

This work shows that it is practical to compute sparse roadmaps in $\mathcal{C}$-spaces that guarantee asymptotically near optimal paths. The experiments suggest that these roadmaps have considerably fewer edges than k-`PRM`*, while resulting in little degradation in path quality. The stretch factor parameter allows this trade-off to be tuned (Fig. 2).

Vertices that do not improve path quality could be removed, but previous work on this idea provides no guarantees on the resulting path quality [8]. Future work will investigate how to filter nodes while providing path quality guarantees. Finally, it is important to study whether roadmap spanners can guarantee the preservation of the homotopic path classes [9]. Homotopic classes tend to be preserved by the spanner because their removal has significant effects in path quality.

References

[1] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. Overmars, “Probabilistic roadmaps for path planning in high-dimensional configuration spaces,” *TRA*, vol. 12, no. 4, pp. 566–580, 1996.

[2] S. Karaman and E. Frazzoli, “Sampling-based algorithms for optimal motion planning,” *IJRR*, vol. 30, no. 7, pp. 846–894, June 2011.

[3] D. Peleg and A. Schäffer, “Graph Spanners,” *Journal of Graph Theory*, vol. 13, no. 1, pp. 99–116, 1989.

[4] S. Baswana and S. Sen, “A simple and linear time randomized algorithm for computing sparse spanners in weighted graphs,” *Random Structures and Algorithms*, vol. 30, no. 4, pp. 532–563, July 2007.

[5] J. D. Marble and K. E. Bekris, “Computing Spanners of Asympotically Optimal Probabilistic Roadmaps,” in *IROS*, 2011.

[6] ——, “Asymptotically Near-Optimal is Good Enough for Motion Planning,” in *ISRR*, 2011.

[7] D. Nieuwenhuisen and M. H. Overmars, “Using cycles in probabilistic roadmap graphs,” in *ICRA*, 2004, pp. 446–452.

[8] R. Pearce, M. Morales, and N. Amato, “Structural Improvement Filtering Strategy for PRM,” in *RSS*, 2008.

[9] L. Jaillet and T. Simeon, “Path deformation roadmaps,” in *WAFR*, New York City, NY, 2006.