Path Planning on Complex Terrain

Daniel M. Tracy
Rensselaer Polytechnic Institute
tracyd@cs.rpi.edu

W. Randolph Franklin
Rensselaer Polytechnic Institute
frankwr@rpi.edu

Barbara Cutler
Rensselaer Polytechnic Institute
cutler@cs.rpi.edu

Franklin T. Luk
Rensselaer Polytechnic Institute
luk@cs.rpi.edu

Marcus Andrade
Federal University of Viçosa, Brazil
marcus.ufv@gmail.com

Jared Stookey
Rensselaer Polytechnic Institute
jstookey@jstookey.com

1. Introduction

We present an algorithm for efficient path planning on complex terrain with an arbitrary cost metric given by a 2D array corresponding to edge weights.

The agent is allowed the full range of Euclidean motion on the 2-dimensional plane, unlike alternate path planning schemes that strictly avoid obstacles, such as the Lee [3] and Hightower [2] algorithms or performing a graph search on the Voronoi diagram of the obstacle boundaries [1]. We use two runs of the A* algorithm to efficiently compute this path.

We assume that the agent has complete knowledge of the terrain. The agent's goal is to plan a path between two given endpoints that minimizes a given cost metric. For example, the metric may penalize the agent for entering an area that is being observed by an opponent, or it may account for the slope of the terrain.

Our path finding routine is an adaptation of the A* algorithm. The first cost metric we evaluated was simply the number of grid points visited (similar to [3]). For the A* algorithm, the terrain is represented as an $n \times n$ grid, each point on the terrain is a separate node, and each node has up to eight children in the search tree, corresponding to eight neighboring points (see Figure 1a). This method has the limitation of minimizing only the Chebyshev distance between the end points [4]. The Chebyshev distance between points (x_1, y_1) and (x_2, y_2) is defined as $max(|x_1 - x_2|, |y_1 - y_2|)$. Our implementation deviates slightly from the actual Chebyshev formula, in that the path length is computed via the Euclidean formula, but only movement in eight directions is considered while planning the path.

2. Chebyshev vs. Euclidean Distance

We start by considering a simple case, where many points on the terrain are obstacles and are marked as untraversable. The agent would like to take the shortest path through the terrain that avoids the obstacles. The cost metric is simply the total distance traversed. A naive method to allow for a full range of Euclidean motion, as shown in Figure 1b, would be to include edges between all grid point pairs in the search space. However, this increases the size of the search space from $O(n^2)$ to $O(n^4)$, as there are $O(n^4)$ possible edges to consider. Also, to compute the true cost of each edge requires $O(n)$ time, because the edge must be segmented as described below. This is clearly too expensive for larger terrains.

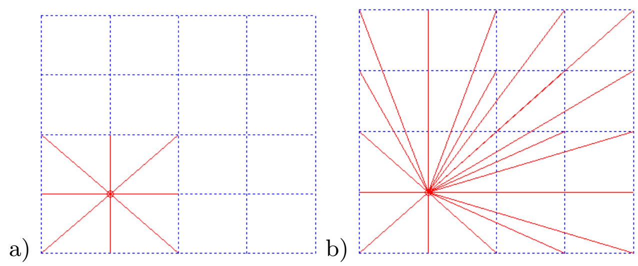

Figure 1: In the first pass of our path-planning algorithm, only movement in eight directions (Chebyshev movement) is considered. Next we would like to plan a path that considers movement in all directions.

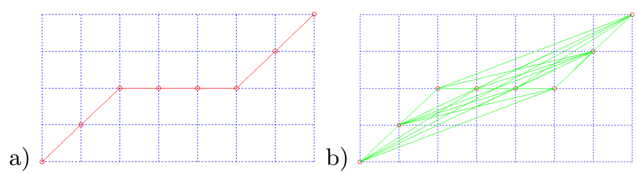

Figure 2: A sample result of the first pass of our path planning algorithm. For the second pass, we consider movement along the green lines.

To speed up the algorithm, we designed a two-pass system. On the first pass, all points on the terrain are included in the search space, and the Chebyshev path is computed as described in the previous section. On the second pass, the only nodes in the search space are the points that are retained in the first path, and an edge is aded to every other node in the search tree (see Figure 2). Thus, for any pair of points in the search space, the smuggler may traverse a straight line connecting them. In practice, computing this second pass is more efficient than the first pass.

Although our 2-pass algorithm is not guaranteed to be the optimal Euclidean path, the output from our heuristic 2-pass system does very well in practice. Our approach will never produce a path worse than the original Chebyshev path. The

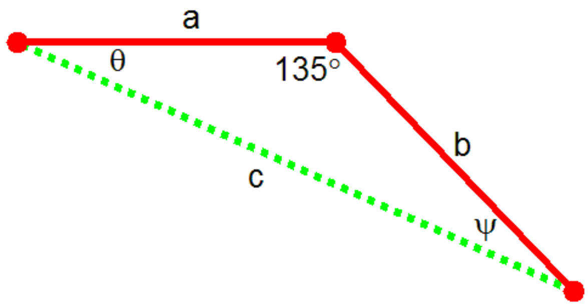

Figure 3: Largest possible difference between the Euclidean and Chebyshev paths.

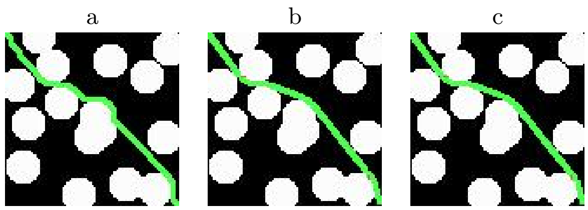

Figure 4: A sample comparison of the results from a) the Chebyshev algorithm, b) our heuristic algorithm, and c) the brute-force algorithm.

largest possible difference between the optimal Euclidean and Chebyshev paths will occur in a situation as in 3. The 135° angle is fixed since we are always using a regularly spaced grid. Then $(a + b) - c$ will be maximized when $\theta = \phi = 22.5^\circ$. Therefore, the optimal Euclidean path should be no less than 92% of the length of the Chebyshev path. Thus, as our heuristic scheme cannot produce a path longer than the Chebyshev path, our heuristic algorithm should produce a path whose length differs by no more than 8% from the optimal Euclidean path. In practice, this difference is usually much less than 8%.

3. Results

For comparison, we compute the optimal Euclidean path with a brute-force application of the A* algorithm. Every pair of grid points, whether adjacent or not, is included in the search space. We compared our heuristic algorithm against the brute-force method by running both algorithms on a hundred 100×100 data sets. We limited the size of the datasets to 100×100, as the brute-force algorithm cannot efficiently handle larger datasets, though our heuristic approach runs quickly on 3200×3200 datasets. The average difference in the lengths of the computed paths was less than 0.1%, while the average speedup was greater than 100. Our heuristic approach is much faster than the brute-force scheme without significantly sacrificing the solution quality. Some sample results are shown in Figures 4 and 5.

4. Alternate Cost Functions

Our scheme can then be generalized to more sophisticated cost metrics. Rather than considering simply Euclidean distance, we are given a 2D array which corresponds to the costs of moving onto a grid point from any adjacent grid point. (The cost of moving from point A onto an adjacent point B need not be the same as the cost of moving from

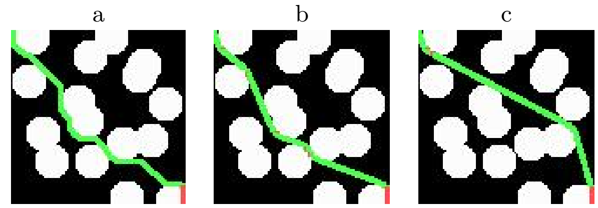

Figure 5: A case where our heuristic algorithm (b), which goes around the central obstacles to the south, produces a different path than the brute-force algorithm (c), which goes around the central obstacles to the north. Here the Chebyshev path (a) would have been equally good going north or south around those obstacles. However, going north allows for more space for a straighter Euclidean path (space that was ignored by the heuristic algorithm).

point B onto point A.) For computing the Chebyshev path, calculating the cost to move between adjacent points is trivial. However, for the second pass which allows a full range of Euclidean motion, the cost to traverse a straight line that connects two distant points must be computed. This line is not likely to pass through grid points exactly. Here the cost metric at several places (not necessarily at grid points) along the line must be interpolated. All the points that lie along gridlines are used, and each point is linearly interpolated from its two closest grid points.

Now our path planning procedure takes a cost function defined on a uniform grid, with non-uniform edge weights, and computes the path that minimizes the cost function while allowing a full range of Euclidean motion.

5. Acknowledgements

This research was supported by NSF grants CCR-0306502 and DMS-0327634, by DARPA/DSO/GeoStar, and by CNPq - the Brazilian Council of Technological and Scientific Development.

6. References

[1] H. Choset. Incremental construction of the generalized voronoi diagram, the generalized voronoi graph, and the hierarchical generalized voronoi graph. In *Proceedings of the First CGC Workshop on Computational Geometry*, October 1997.

[2] D. W. Hightower. A solution to line-routing problems on the continuous plane. In *DAC '69: Proceedings of the 6th Annual Conference on Design Automation*, pages 1–24, New York, NY, USA, 1969. ACM.

[3] C. Lee. An algorithm for path connections and its applications. *IRE Transactions on Electronic Computers, EC-10*, pages 346–365, Sept. 1961.

[4] D. M. Tracy, W. R. Franklin, B. Cutler, M. A. Andrade, F. T. Luk, M. Inanc, and Z. Xie. Multiple observer siting and path planning on lossily compressed terrain. In *Proceedings of SPIE Vol. 6697 Advanced Signal Processing Algorithms, Architectures, and Implementations XVII*, San Diego CA, 27 August 2007. International Society for Optical Engineering. paper 6697-16.