Disproving the Optimality of a String-Pulling Approach for Path Smoothing on Grid Graphs

Taoan Huang, Jihee Han, Tansel Uras, Sven Koenig
University of Southern California
taoanhua@usc.edu, jrjrhan@gmail.com, turas@usc.edu, skoenig@usc.edu

October 2020

A grid is a discretization of a continuous 2D environoment using square cells where each cell is either blocked or unblocked. Grids allow one to use discretize optimization approaches for a large variety of path-planning problems in robotics. A grid graph $G = (V, E)$ is constructed by placing vertices at the corners of unblocked cells and connecting two vertices of the same unblocked cell with an edge. However, shortest paths found on grid graphs are often longer than those found in continuous settings. [1] focuses on path smoothing on grid graphs and proposes a string-pulling approach. A grid graph $G = (V, E)$ and a shortest grid path $p = (s = p_1, p_2, \ldots, t = p_n)$ from s to t on G are given as input to the algorithm. Imagine p is a piece of string placed on the 2D plane on which the grid graph lies. The algorithm aims to output the path corresponding to the configuration of the string after pulling the string tight between s and t. We refer to such a path as the optimal path. The string-pulling algorithm proposed in [1] shows improvement over several baseline algorithms experimentally but lacks theoretical guarantees. We refer the readers to [1] for the details and pseudocode of the algorithm.

In this report, we disprove the optimality of the algorithm by providing a counterexample. In particular, we prove by contradiction that the algorithm does not output the optimal path for this counterexample.

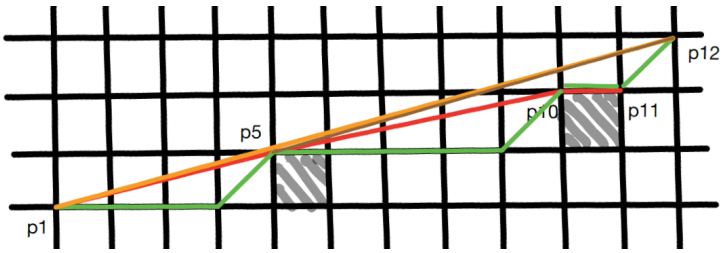

Figure 1: The counterexample: A grid with two blocked cells. The green path $(p_1, \ldots, p_{12})$ is the input path. The red path $(p_1, p_5, p_{10}, p_{11})$ is the optimal path for the input path $(p_1, \ldots, p_{11})$. The orange path (p_1, p_{12}) is the optimal path for the input path $(p_1, \ldots, p_{12})$.

Consider the grid shown in Figure 1 and two different input paths. In the first case, the input path (green) is $(p_1, ...p_{11})$ and the optimal path (red) is $(p_1, p_5, p_{10}, p_{11})$. In the second case, the input path (green) is $(p_1, ...p_{12})$ and the optimal path (orange) is (p_1, p_{12}).

We prove by contradiction that the algorithm does not always output the optimal path: Assume that the algorithm always outputs the optimal path. In the first case, after finishing the for loop, since the algorithm is optimal, the queue contains the vertices p_1, p_5 and p_{10} and then p_{11} is added to the queue (line 32 of Algorithm 1 in [1]), giving us the optimal path. In the second case, at the moment when i increases from 10 to 11 in the for loop, the queue contains the same vertices p_1, p_5 and p_{10} in the queue as at the time when the for loop terminates in the first case. Since p_{12} and sp_{end} (p_{10}) have line-of-sight and `cur_turn` $\neq$ `turn`, p_{10} is removed from the queue (line 23). The algorithm terminates after doing nothing in the next iteration since p_{12} and sp_{end} (p_{10}) have line-of-sight and `cur_turn` = `turn`. Thus, the algorithm outputs (p_1, p_5, p_{12})

(the red segment from p_1 to p_5 and the brown segment from p_5 to p_{12}), which is longer than the optimal path, leading to a contradiction.

Funding Acknowledgement This research was supported by grants funded by the National Science Foundation (NSF) under grant numbers 1724392, 1409987, 1817189, 1837779 and 1935712.

References

[1] J. Han, T. Uras, and S. Koenig. Toward a string-pulling approach to path smoothing on grid graphs. In *Symposium on Combinatorial Search (SoCS)*, 2020.