Integrating Local Collision Avoidance with Shortest Path Maps

Ritesh Sharma, Renato Farias and Marcelo Kallmann
Computer Graphics Lab, University of California Merced, California, United States
rsharma39@ucmerced.edu, rfarias2@ucmerced.edu, mkallmann@ucmerced.edu

1. Introduction

Local collision avoidance is an integral part of making multiple agents able to interact with each other in a fluid and realistic way. However, being restricted to only local knowledge of the environment means that agents may get trapped in obstacles. In such cases global path planning techniques need to be integrated.

A SPM [1, 2] encodes all globally-optimal shortest paths from defined goal locations to every point in an environment. While SPMs are well-known in the computational geometry area, the complexity in computing them has limited their adoption in applications. Based on our previous work [3, 4] relying on GPU rasterization procedures we are able to efficiently produce a SPM representation that integrates well with multi-agent simulations. Our SPMs can in addition handle polygonal lines as goal locations. A single SPM is therefore able to optimally route multiple agents around obstacles and towards their closest points in their closest goal polygonal line.

This work presents our first results exploring the new approach of integrating Shortest Path Maps (SPMs) with local collision avoidance in order to provide optimal paths for agents to navigate around obstacles toward their goal locations. Our SPM implementation is available.

2. Shortest Path Maps

SPMs are structures computed with respect to "sources", which in our case are arbitrary polygonal lines representing goal locations in our scenarios. A SPM computed for a particular planar environment encodes a globally-optimal Euclidean Shortest Path for *all* points in that environment.

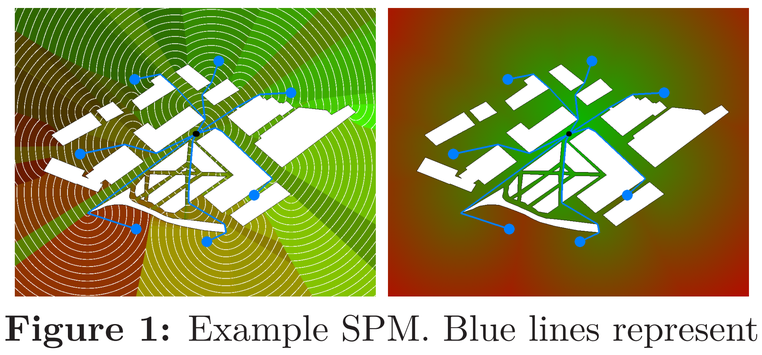

Figure 1: Example SPM. Blue lines represent shortest paths from each blue agent to a source point in the center of the environment. SPM regions (left) share a same parent vertex towards the source. A distance field is also naturally represented (right).

3.Collision Avoidance Integration

We have integrated our SPM implementation with the well-known Reciprocal Velocity Obstacle (RVO) [5] approach for collision avoidance.

At the beginning of every time step each agent makes a query from its current position to the SPM in order to retrieve in constant time the preferred velocity vector, which is the vector leading to the next vertex along their shortest path to the goal.

Once this vector is retrieved, a collision-free velocity vector is determined from the RVO method. First, the set of velocity regions using the preferred velocity vector for each agent in the neighborhood of the current agent is computed. Then the velocity obstacles for every pair of agents is computed and the boundary of the union of all velocity obstacles is determined. Finally each agent is routed to their nearest point on that boundary.

The result naturally balances following the shortest path while avoiding agents nearby.

4. Results

To evaluate our approach we have implemented two scenarios: one with a single group of agents, and the other with four groups of agents. In both scenarios the groups of agents have to navigate from a source region to a goal region. Figure 2 reveals that despite using an effective collision avoidance technique, the agents cannot navigate around the obstacles and are easily stuck in traps. This does not happen when the agents are guided by the velocity vector extracted from the SPM of the environment. When multiple groups of agents are defined, as shown in Figure 3, each group relies on a SPM specific to its goal region. In each case, when agents rely on their SPM they do not get stuck in the "pockets" of the environment.

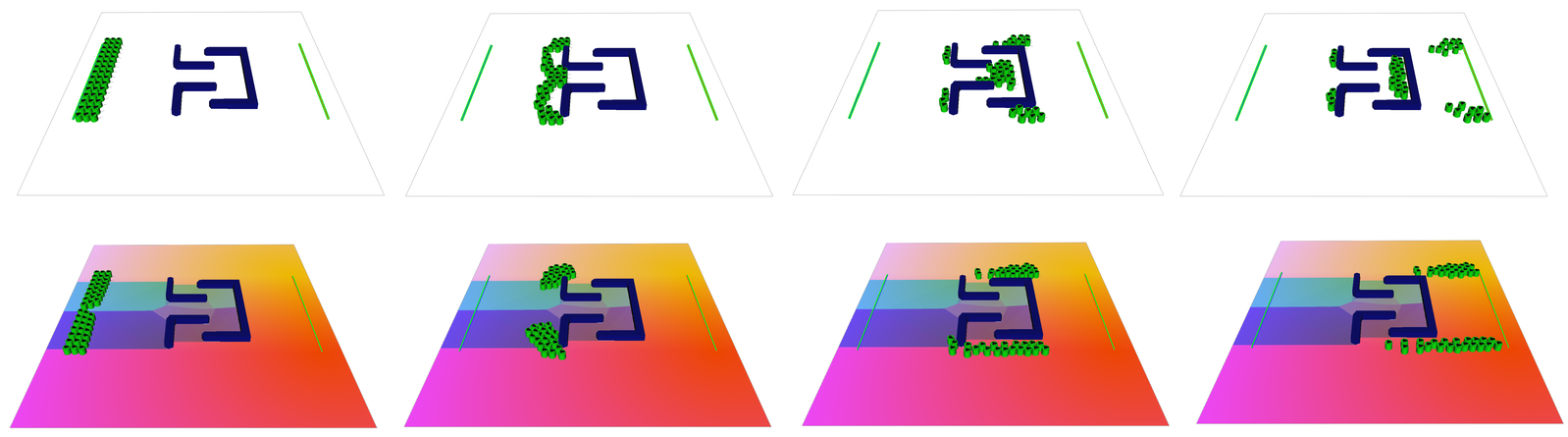

Figure 2: Agents are trapped with only collision avoidance (top) but not with SPMs (bottom).

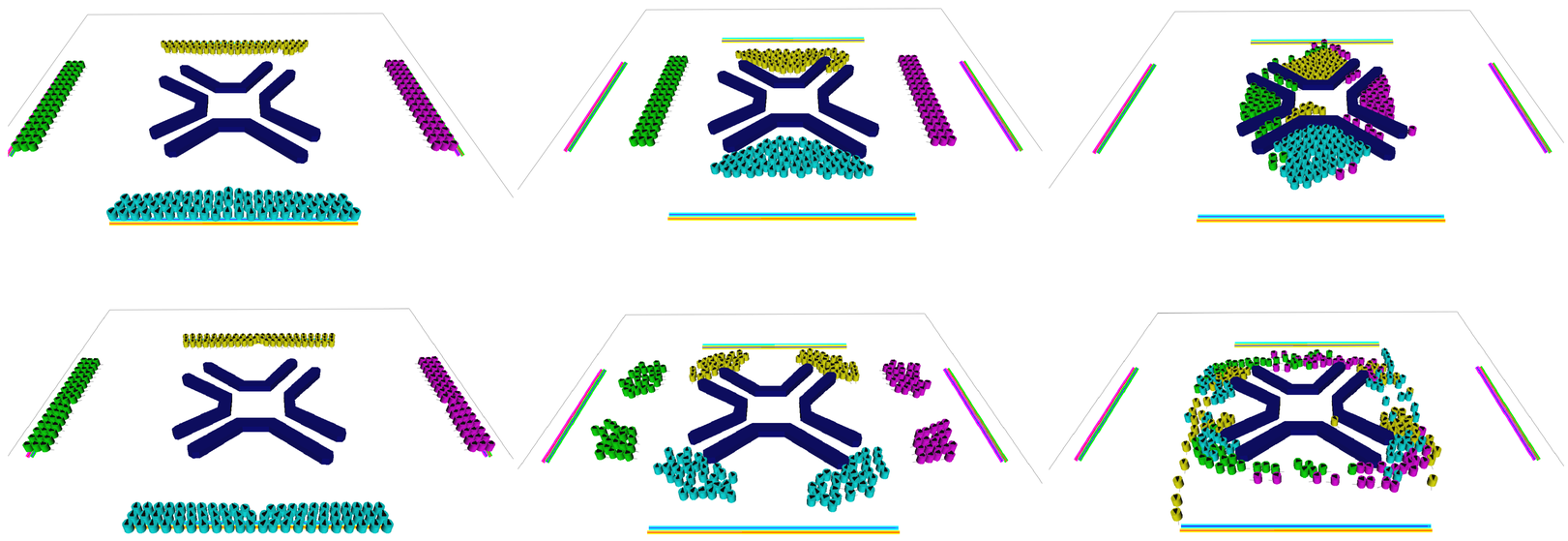

Figure 3: Four groups navigating toward the central area using RVO collision avoidance alone (top) and with SPMs providing preferred velocity vectors aligned with shortest paths (bottom).

5. Code Availability and Acknowledgements

Our SPM implementation is available as the libsigspm module of the SIG toolkit hosted at https://bitbucket.org/mkallmann/sig/.

This research was sponsored by the Army Research Office and was accomplished under Grant Number W911NF-17-1-0463. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Office or the U.S. Government.

References

[1] Joseph S. B. Mitchell. A New Algorithm for Shortest Paths Among Obstacles in the Plane. *Annals of Mathem. and Artif. Intel.*, 3(1):83–105, 1991.
[2] Joseph S. B. Mitchell. Shortest Paths Among Obstacles in the Plane. In *Proc. SOCG*, SCG '93, pages 308–317, New York, NY, USA, 1993. ACM.
[3] Carlo Camporesi and Marcelo Kallmann. Computing shortest path maps with GPU shaders. In *Proc. MIG'14*, pages 97–102. ACM, 2014.
[4] Renato Farias and Marcelo Kallmann. Optimal path maps on the GPU. *IEEE TVCG*, 2019.
[5] Jur van den Berg, Ming C. Lin, and Dinesh Manocha. Reciprocal velocity obstacles for real-time multi-agent navigation. In *IEEE ICRA*, pages 1928–1935. IEEE, 2008.