3D Behaviors for Multi-Agent Simulations with Position-Based Dynamics

Ritesh Sharma[1] Tomer Weiss[2] Marcelo Kallmann[1]

[1]University of California Merced [2]New Jersey Institute of Technology

Abstract

Position-Based Dynamics (PBD) has been shown to provide a flexible framework for modeling per-agent collision avoidance behavior for crowd and multi-agent simulations in planar scenarios. In this work we propose to extend the approach such that collision avoidance reactions can utilize the entire volumetric 3D space around each agent when deciding how to avoid collisions with other agents. We propose to use separation planes for collision avoidance, using either preferred or algorithmically determined planes. Our results demonstrate the ability to control spatial 3D behavior of simulated agents. We also compare with a different 3D collision avoidance approach based on Reciprocal Velocity Obstacles (RVOs).

Introduction

Collision avoidance is a key component of multi-agent simulations for both 2D and 3D environments. Most multi-agent and crowd simulation frameworks can be extended to 3D. However, if no specific extensions are incorporated, when agents move together in the same plane, the collision avoidance behavior will essentially remain planar when trivially embedded in a 3D environment. Different approaches can be taken for incorporating collision avoidance reactions to fully utilize free 3D space around agents. One motivation for utilizing such free space is, for instance, the simulation of Unmanned Aerial Vehicle (UAV) agents. When simulating multiple autonomous drone agents, drones would need to avoid going on top of each other when performing a collision avoidance maneuver, in order to prevent downward forces from the stream of air coming from an UAV's rotor. Another motivation is in case of heavy-density scenarios, where the planar space is not sufficient for collision avoidance navigation.

The well-known Reciprocal Velocity Obstacles (RVOs) does not include behavior customization but is efficient and has been extended to 3D `http://gamma.cs.unc.edu/RVO2/downloads`. Position-Based Dynamics (PBD) has become a popular approach in computer graphics for animating different types of dynamical systems. Recently, PBD has been applied to crowd animation [2]; however, only 2D planar scenarios were explored. In order to fully utilize the 3D space around the agents, the proposed approach specifies separation planes to be considered when handling collision avoidance, taking into account both preferred planes and automatic plane determination. We present initial results demonstrating the ability to control the spatial behavior of the simulated agents, and also comparing the overall behavior with the result of using Reciprocal Velocity Obstacles (RVOs) [1].

Baseline PBD Solver

We implemented the algorithmic steps described by Weiss and colleagues [2], except for long range collision resolution. The main steps in a simulation time step are summarized below:

Step 1: Set up initial position and velocities

Step 2: Find neighboring agents

Step 3: Solve short range interaction

Step 4: Solve long range collision

Step 5: Update new position and velocities

Instead of finding a tangential direction in Step 4, we define a separation plane and solve long range constraints based on such plane and contact normal. Once the position correction is determined, we multiply the stiffness constraint of $k * e^{\left(\frac{-\hat{\tau}^2}{\tau_0}\right)}$ and $||d||$ to the position correction, where k is user-defined. Scalar d is the relative displacement of the position of the agents $\tilde{x_i}$ and $\tilde{x_j}$ at the predicted point of contact, and $\hat{x_i}$ and $\hat{x_j}$ are the positions at the discrete time step $\hat{\tau}$, which is the time step slightly before the predicted contact. Scalar τ_0 is a fixed constant.

3D Avoidance Behavior

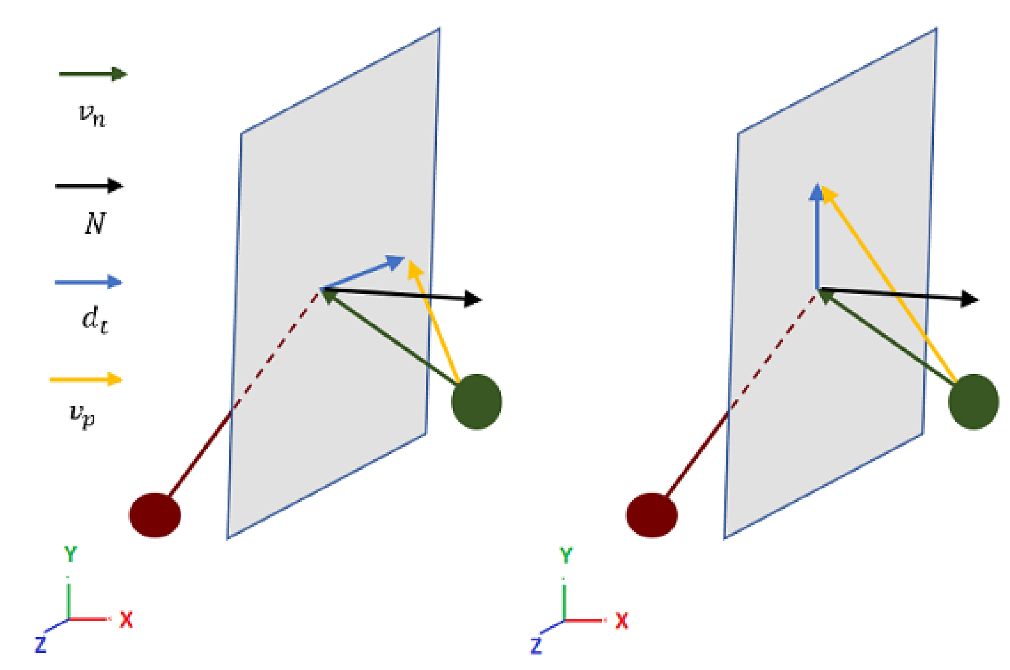

Fig. 1: The two agents are ascending towards their respective goals and will collide at the separation plane defined by normal N. The predicted velocity vector v_p is computed by taking vector addition of v_n and d_t. Left: Horizontal collision avoidance. Right: Vertical collision avoidance.

At each time step using the long range technique discussed in [2], the contact normal n for each agent is determined. A normal N to the plane of collision is then computed as $N = n \times a$, where a is taken as a preferred global axis frame vector. For horizontal avoidance, a is the y-axis of the global frame. For vertical avoidance, a is the z-axis of the global frame. This assumes that agents are moving on the xz plane. Next, the tangential direction of the agent is computed with

$$d = (\hat{x_i} - \tilde{x_i}) - (\hat{x_j} - \tilde{x_j}), \tag{1}$$

$$d_n = (d.n)n, \tag{2}$$

$$d_t = d - d_n, \tag{3}$$

where d, d_n and d_t are the relative displacement, the contact normal and the tangential component respectively. Next, the predicted velocity is computed as

$$v_p = d_n + v_n. \tag{4}$$

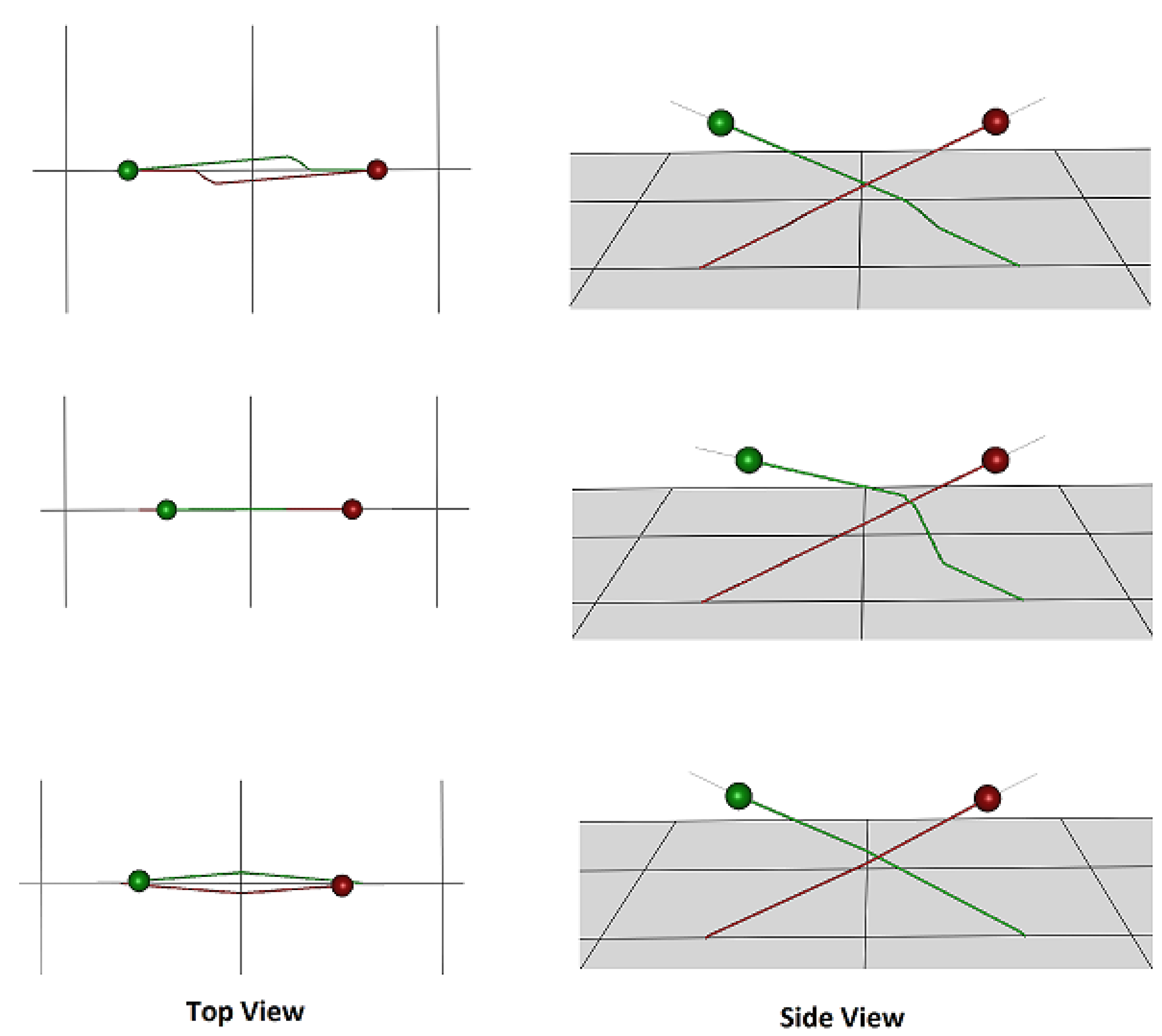

Fig. 2: Trajectories obtained by agents while avoiding collision in 3D. Top row: agents avoid each other sideways, Middle row: agents avoid each other vertically. Bottom row: agents avoid each other using RVO.

Results

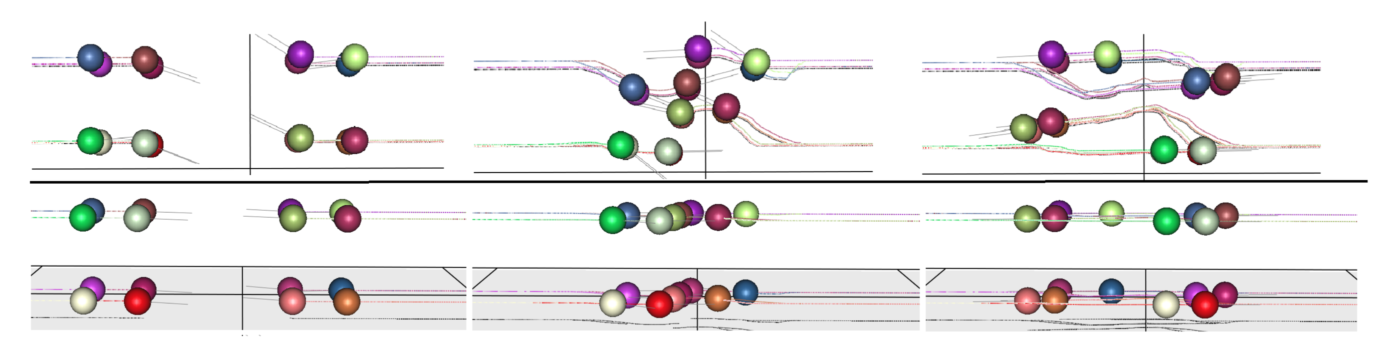

Fig. 3: Simulation snapshots with both top view (top row) and side view (bottom row) of two groups of agents crossing each other based on the horizontal avoidance behavior.

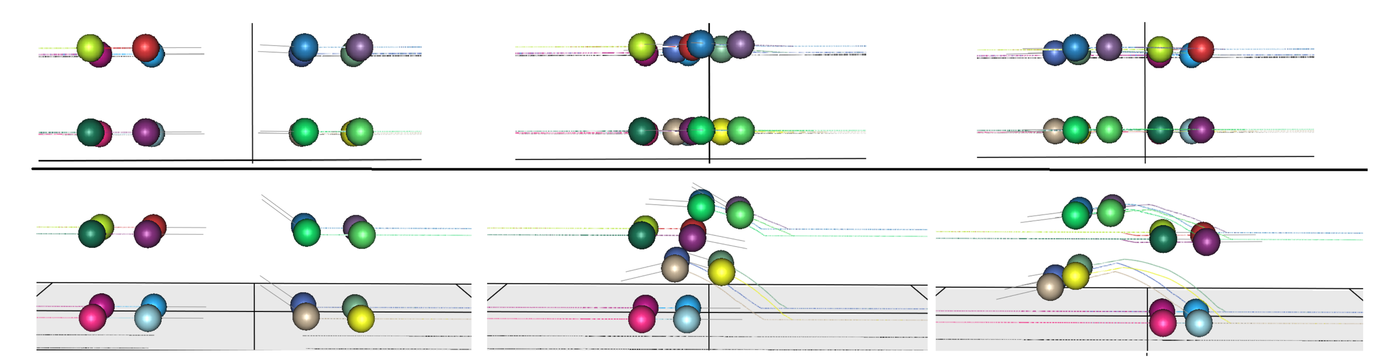

Fig. 4: Same scenario as in Figure 3 but using the vertical collision avoidance behavior.

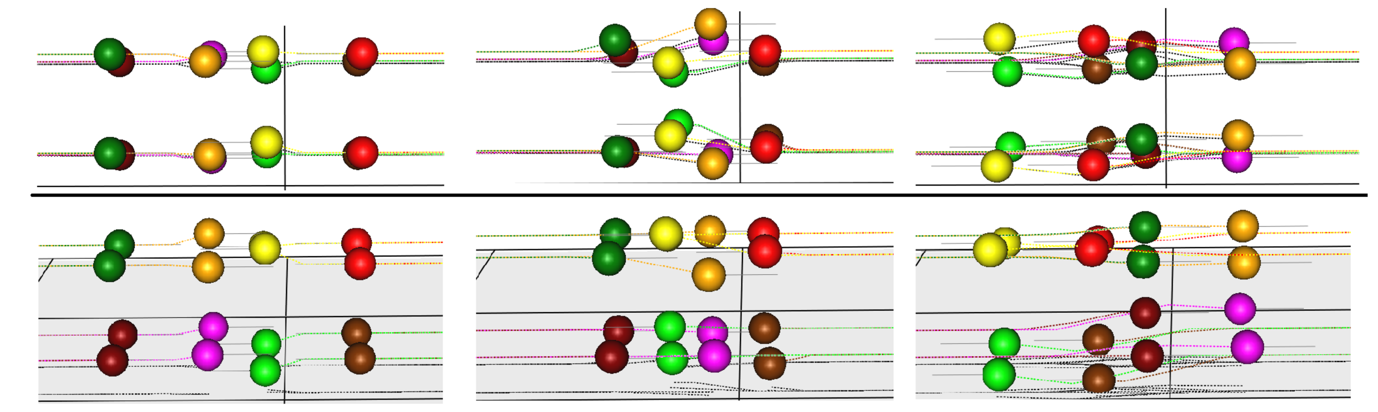

Fig. 5: Same scenario as in Figure 3 but using the collision avoidance behavior provided by the 3D implementation of RVO.

Figures 3, 4, and 5 illustrate additional examples involving several agents. While the presented rules can be extended and applied to generic separation planes, at this time we have only evaluated the presented planes in global coordinates. The automatic selection of a separation plane strategy can be implemented in different ways according to the application. For example we have obtained initial good results by selecting a plane that leads to reactive trajectories occurring in surrounding areas minimizing the number of agents.

References

[1] Jur Van den Berg, Ming Lin, and Dinesh Manocha. "Reciprocal velocity obstacles for real-time multi-agent navigation". In: *2008 IEEE International Conference on Robotics and Automation*. IEEE. 2008, pp. 1928–1935.

[2] Tomer Weiss et al. "Position-Based Multi-Agent Dynamics for Real-Time Crowd Simulation". In: *Proceedings of the Tenth International Conference on Motion in Games*. MIG '17. Barcelona, Spain, 2017.