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

Ritesh Sharma
Computer Graphics Lab
University of California Merced
Merced, CA, United States
rsharma39@ucmerced.edu

Tomer Weiss
Department of Informatics
New Jersey Institute of Technology
Newark, NJ, United States
tweiss@njit.edu

Marcelo Kallmann
Computer Graphics Lab
University of California Merced
Merced, CA, United States
mkallmann@ucmerced.edu

***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 in a controlled way the 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 automatically determined planes. Our results demonstrate the ability to control the spatial 3D behavior of simulated agents by constraining the produced movements according to the separation planes. Our method is generic and can be integrated with different crowd simulation techniques. We also compare our results with a 3D collision avoidance method based on Reciprocal Velocity Obstacles (RVOs).**

***Index Terms*—crowd simulation, 3D multi-agent simulation, 3D collision avoidance**

I. Introduction

Collision avoidance is a key component of multi-agent simulations for both 2D and 3D environments. While most existing multi-agent and crowd simulation frameworks are designed for 2D scenarios, extending a 2D framework to 3D is usually not a difficult task. However, when a 2D framework is trivially embedded in a 3D environment without any specific extensions incorporated, agents will typically move together in the same plane and the collision avoidance behavior will essentially remain planar. This paper addresses the problem of how to incorporate collision avoidance reactions that fully utilize the free 3D space around agents in a customized way according to the application.

One motivation for utilizing the free 3D space around agents is, for instance, the simulation of Unmanned Aerial Vehicle (UAV) agents. When simulating multiple UAV agents, or drones, they 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 situation requiring control of the collision avoidance behavior is when agents have to behave as a group with desired 3D behaviors. For instance, flying vehicles can be piloted by simulated virtual characters. In such case, the scenario is equivalent to a 3D crowd simulation where collision avoidance behavior is important to be customized in order to achieve realistic results. Other relevant scenarios might involve underwater vehicles or the simulation of sea creatures.

Another motivation for exploring techniques for 3D collision avoidance behavior is to achieve efficient behaviors in the case of high-density scenarios. For example, a trivial planar reaction mechanism might be sufficient when the agent concentration in the 3D space is not dense, in high density scenarios the utilization of the full 3D space might become necessary for successful navigation.

In order to fully consider the 3D space around agents, we propose to utilize separation planes for handling collision avoidance behavior, taking into account both preferred planes and planes automatically determined. We implement our approach using Position-Based Dynamics (PBD), which is a flexible approach in computer graphics for producing animations for different types of constrained dynamical systems. PBD has been applied to crowd animation in 2D [12] and in this paper we extend the method by incorporating 3D constraints implementing the proposed plane-based behaviors. We also compare our proposed method with an existing approach based on Reciprocal Velocity Obstacles (RVOs) in 3D, using an implementation publicly available [1][1].

II. Related Work

The motion of groups and crowds of virtual humans and other autonomous agents are an important component of multiple immersive media systems, such as virtual reality, augmented reality, virtual production, and computer games. Computational methods that control the movements of such agents are necessary for interactive and autonomous behavior. Typically, the motion of agents toward their navigation goals needs to be visually realistic, which necessitates agents being able to avoid each other while navigating in a shared space. Hence, collision avoidance is an important component of crowd and multi-agent simulations.

Researchers have proposed multiple methods for simulating the movement of multiple agents or crowds [3], [9], [12], [10]; however, most methods address 2D navigation scenarios for human-like agents moving on a plane. A few other methods have considered 2.5D multi-agent simulation scenarios where

[1]Accessed September 2020.

agents move on a surface of a 3D object, which can have an arbitrary 3D shape [7]. Simulating the movement of agents in 3D is still an emerging area, and is perhaps mostly suitable for the simulation of non-human agents, such as birds, fish, or aerial or underwater vehicles.

An area which has been more extensively explored is related to simulating group-like behavior of animals which can move in 3D. Crowd simulation research in 3D mostly refers to these types of simulation. Early work in 3D crowd simulation was developed by Reynolds [6], and is nicknamed the Boids model. Boids calculates agent velocities based on rules for collision avoidance, relative agent distances and orientations, and agent navigation goals. Simulated agents are mostly animal-like, including fish and birds, and simulate group behaviour such as flocking [2]. Other researchers have built on such work, for example Sato et al. [8] added a unified motion planner based on fish swimming styles. Methods for the simulation of insect swarms have also been proposed [4], [11], [5]. However, while such proposed methods imitate visually realistic group behavior, they do not focus on collision avoidance and in particular they do not focus on the customization of the avoidance behavior in 3D.

Our present work builds on previous work in 2D agent collision avoidance [12] and extends it to 3D with the ability to control how the 3D space is utilized by the obtained collision avoidance behavior. In order to compare our results including avoidance behavior control with an approach without control, we compare our results with the results obtained with an available implementation extending Reciprocal Velocity Obstacles (RVOs) to 3D [1]. While collision avoidance with RVOs is effective, it requires computing a collision-free velocity vector at each step taking into account all agents in a neighborhood, usually with a linear program solver. In contrast, our 3D PBD-based collision avoidance can be distributed, since each PBD constraint enforces collision avoidance independently for each pair of agents. The provided comparisons illustrate the results that can be obtained when controlling plane-based avoidance behavior as we propose in this paper, versus without any control, as provided by the 3D RVO method.

III. Baseline PBD Solver

Algorithm 1 describes the overall algorithm for 3D crowd simulation loop. Algorithm 1 differs from the algorithmic steps described by Weiss and colleagues [12] at step 9. We refer the reader to [12] for full details. Instead of finding a tangential direction we define a separation plane and solve long range constraints based on this plane and use a contact normal which will be discussed in Section IV.

Once the position correction is determined, we multiply the stiffness constraint of $ke^{\left(\frac{-\hat{\tau}^2}{\tau_0}\right)}$ and $||d||$ to the position correction, where k is user-defined. Vector d is the total relative displacement of the position of the agents $\tilde{x_i}$ and $\tilde{x_j}$ at the predicted point of contact:

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

Algorithm 1 PBD based 3D crowd simulation loop

1: **for each** agent i **do**
2: Calculate ${v_i}^b$ from ${v_i}^p$ and v_i
3: ${x_i}^p \leftarrow x_i + \Delta t {v_i}^b$
4: find neighboring agents $N_{ij} = \{n_{i1}, n_{i2}, ..., n_{ik}\}$
5: **while** iteration $<$ maximum stability iterations **do**
6: **for each** agent i **do**
7: Short range position correction $\Delta {x_i}^p$
8: **while** iteration $<$ maximum long-range iterations **do**
9: **for each** agent i **do**
10: Plane-based long range position correction $\Delta {x_i}^p$
11: **for each** agent i **do**
12: ${v_i}^p \leftarrow ({x_i}^p - x_i)/\Delta t$
13: Add XSPH viscosity to ${v_i}^p$
14: Clamp ${v_i}^p$
15: $x_i \leftarrow {x_i}^p$

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.

IV. 3D Avoidance Behavior

We present two methods to control the 3D collision avoidance behavior based on selecting global separation planes. We define a separation plane by its normal d_n.

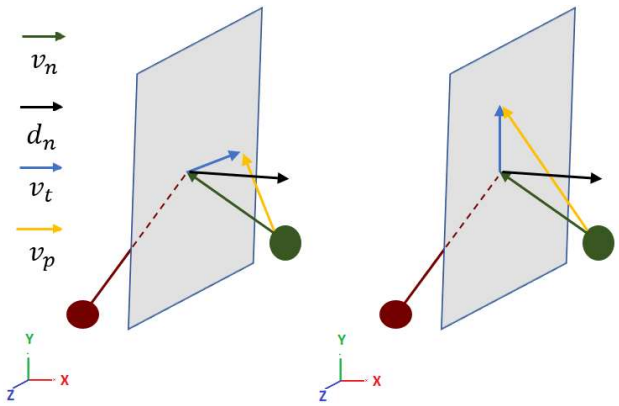

Fig. 1. The two agents are ascending towards their respective goals and will collide at the separation plane passing by the collision point and with normal vector d_n. The predicted velocity vector v_p is computed by taking the vector addition of v_n and v_t. Left: Horizontal collision avoidance. Right: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

At each time step, we determine the contact normal d_n for each agent using the long range technique discussed in [12]. A vector v_t on the plane of collision is then computed as $v_t = d_n \times a$, 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.

The predicted velocity is computed next as:

$$v_p = v_t + v_n. \tag{2}$$

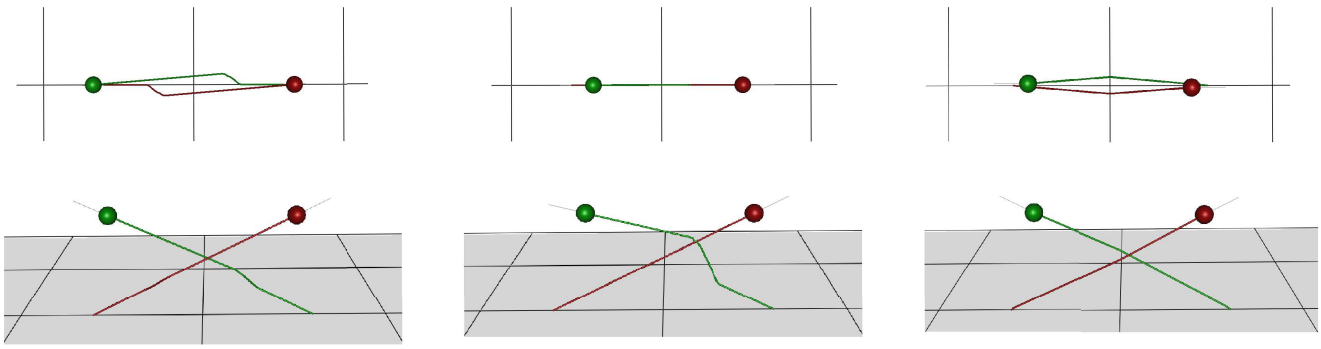

Fig. 2. Trajectories obtained by agents while avoiding collision in 3D. Left: agents avoid each other sideways, Middle: agents avoid each other vertically. Right: agents avoid each other using RVO. In each column, both the top view (top column) and a side view (bottom column) are shown.

Figure 2 illustrates an example of the obtained behavior with vertical and horizontal collision avoidance rules, also comparing results with using RVO without plane based collision avoidance control.

V. Results

Figures 3, 4, and 5 illustrate 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 areas minimizing the number of agents. This can be implemented by measuring the density of agents around the collision avoidance region and selecting a plane that lies in a low-density area. Such approach favors the utilization of the non-used 3D areas around each collision avoidance event, however does not achieve any visible global pattern in the obtained collision avoidance behavior.

VI. Conclusion and Future work

In this work we have proposed a simple yet flexible approach to extend the 2D Position-Based Dynamics approach for crowd simulation to 3D, allowing flexible agent collision avoidance behavior in 3D environments. Furthermore, we have introduced *separation planes* for controlling the collision behavior of the agents. We focused on providing an intuitive manner to customize the 3D collision behavior of agents, which can be clearly noticed in the comparison with the 3D RVO method. We did not observe a difference in the smoothness of agent trajectories.

In the future, we plan to extend our work in several directions. First, we plan to incorporate an automatic rule-based separation plane determination algorithm. Such algorithm will select specific separation planes according to each encountered situation, and will be able to provide a more powerful way to model 3D-specific collision avoidance behavior for different types of agents. Second, we would like to extend our system to include motion model constraints for simulating different types of holonomic and non-holonomic autonomous vehicles. Finally, we are interested in capturing the group-like behavior of non-human animals with PBD constraints. We believe such additions will provide users with improved artistic fine-grained control, in conjunction with other PBD constraints.

References

[1] Optimal reciprocal collision avoidance in three dimensions. *URL: https://github.com/snape/RVO2-3D*, 2020.

[2] M. Anderson, E. McDaniel, and S. Chenney. Constrained animation of flocks. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 286–297, 2003.

[3] D. Helbing and P. Molnar. Social force model for pedestrian dynamics. *Physical review E*, 51(5):4282, 1995.

[4] W. Li, D. Wolinski, J. Pettré, and M. C. Lin. Biologically-inspired visual simulation of insect swarms. In *Computer Graphics Forum*, volume 34, pages 425–434. Wiley Online Library, 2015.

[5] J. Ren, X. Wang, X. Jin, and D. Manocha. Simulating flying insects using dynamics and data-driven noise modeling to generate diverse collective behaviors. *PloS one*, 11(5):e0155698, 2016.

[6] C. W. Reynolds. Flocks, herds and schools: A distributed behavioral model. In *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, pages 25–34, 1987.

[7] B. C. Ricks and P. K. Egbert. A whole surface approach to crowd simulation on arbitrary topologies. *IEEE transactions on visualization and computer graphics*, 20(2):159–171, 2013.

[8] D. Satoi, M. Hagiwara, A. Uemoto, H. Nakadai, and J. Hoshino. Unified motion planner for fishes with various swimming styles. *ACM Transactions on Graphics (TOG)*, 35(4):1–15, 2016.

[9] J. Van den Berg, M. Lin, and D. Manocha. Reciprocal velocity obstacles for real-time multi-agent navigation. In *2008 IEEE International Conference on Robotics and Automation*, pages 1928–1935. IEEE, 2008.

[10] W. van Toll, F. Grzeskowiak, A. L. Gandía, J. Amirian, F. Berton, J. Bruneau, B. C. Daniel, A. Jovane, and J. Pettré. Generalized microscropic crowd simulation using costs in velocity space. In *Symposium on Interactive 3D Graphics and Games*, pages 1–9, 2020.

[11] X. Wang, J. Ren, X. Jin, and D. Manocha. Bswarm: biologically-plausible dynamics model of insect swarms. In *Proceedings of the 14th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 111–118, 2015.

[12] T. Weiss, C. Jiang, A. Litteneker, and D. Terzopoulos. Position-based multi-agent dynamics for real-time crowd simulation. In *Proceedings of the Tenth International Conference on Motion in Games*, MIG '17, New York, NY, USA, 2017. Association for Computing Machinery.

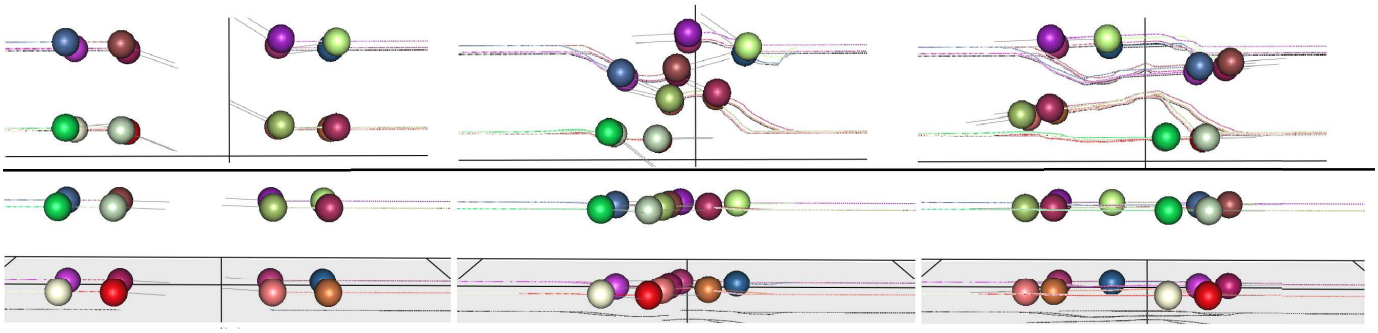

Fig. 3. Simulation snapshots with both top view (top row) and side view (bottom row) of two groups of agents crossing each other based on the horizontal avoidance behavior.

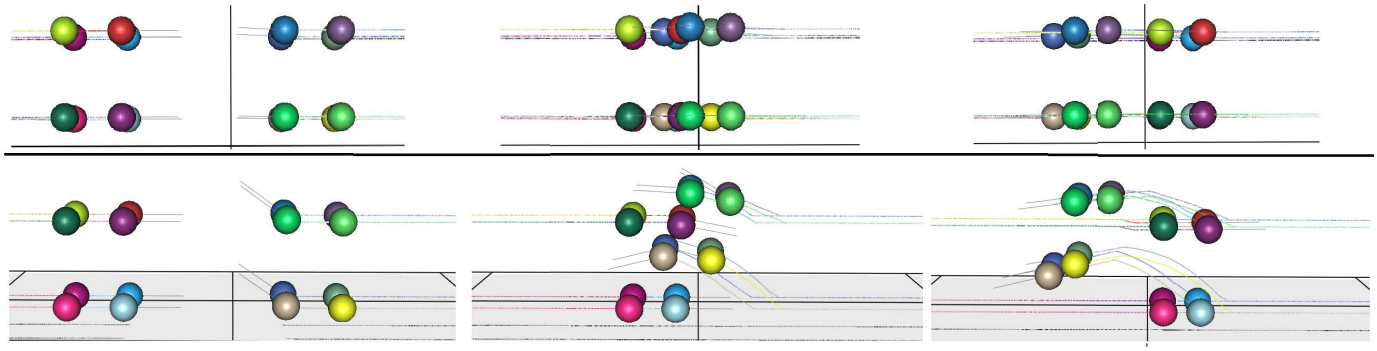

Fig. 4. Same scenario as in Figure 3 but using the vertical collision avoidance behavior.

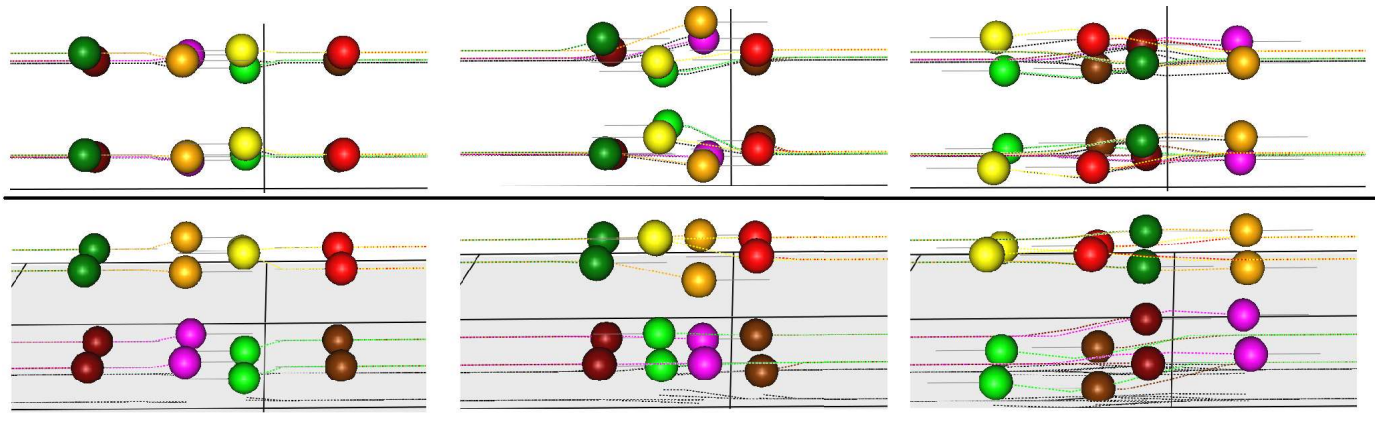

Fig. 5. Same scenario as in Figure 3 but using the collision avoidance behavior provided by the 3D implementation of RVO.