Position-Based Multi-Agent Dynamics for Real-Time Crowd Simulation

Tomer Weiss
University of California, Los Angeles
tweiss@cs.ucla.edu

Alan Litteneker
University of California, Los Angeles
alitteneker@cs.ucla.edu

Chenfanfu Jiang
University of Pennsylvania
cffjiang@seas.upenn.edu

Demetri Terzopoulos
University of California, Los Angeles
dt@cs.ucla.edu

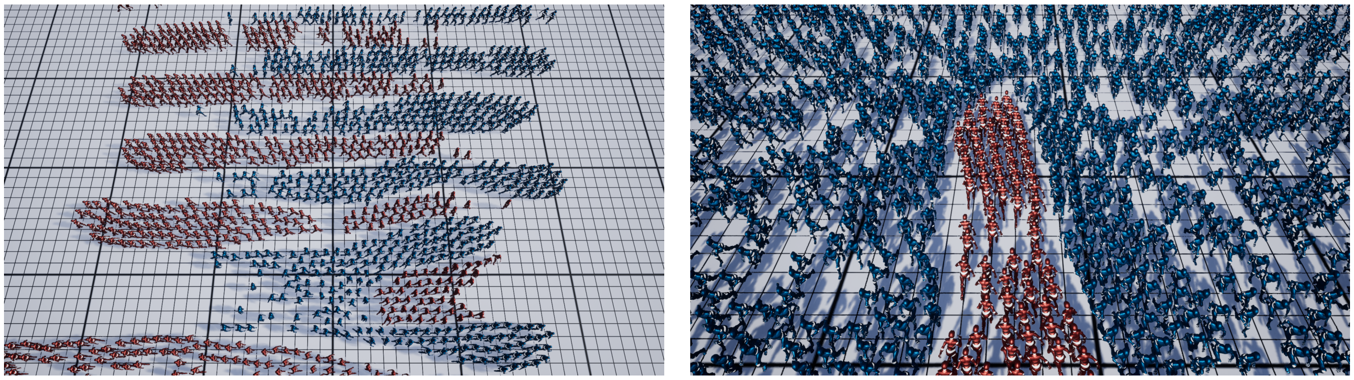

Figure 1: Our PBD-based crowd simulation method animates both sparse and dense groups of agents at interactive rates.

ABSTRACT

Exploiting the efficiency and stability of Position-Based Dynamics (PBD), we introduce a novel crowd simulation method that runs at interactive rates for hundreds of thousands of agents. Our method enables the detailed modeling of per-agent behavior in a Lagrangian formulation. We model short-range and long-range collision avoidance to simulate both sparse and dense crowds. On the particles representing agents, we formulate a set of positional constraints that can be readily integrated into a standard PBD solver. We augment the tentative particle motions with planning velocities to determine the preferred velocities of agents, and project the positions onto the constraint manifold to eliminate colliding configurations. The local short-range interaction is represented with collision and frictional contact between agents, as in the discrete simulation of granular materials. We incorporate a cohesion model for modeling collective behaviors and propose a new constraint for dealing with potential future collisions. Our new method is suitable for use in interactive games.

CCS CONCEPTS

• **Computing methodologies** → **Animation**; *Real-time simulation*;

KEYWORDS

position-based dynamics, crowd simulation, collision avoidance

ACM Reference format:
Tomer Weiss, Alan Litteneker, Chenfanfu Jiang, and Demetri Terzopoulos. 2017. Position-Based Multi-Agent Dynamics for Real-Time Crowd Simulation. In *Proceedings of MiG '17, Barcelona, Spain, November 8–10, 2017,* 8 pages.
DOI: 10.1145/3136457.3136462

1 INTRODUCTION

Crowd simulation is ubiquitous in visual effects, animations, and games. Efficiently simulating the motions of numerous agents with realistic interactions among them has been a major focus of research in recent decades [Thalmann 2007]. Among various modeling considerations, collision avoidance remains challenging and time consuming. Collision avoidance algorithms can be classified into discrete and continuum approaches [Golas et al. 2013]. Continuum approaches, such as the technique proposed by Narain et al. [2009], have proven efficient for large-scale dense crowds, but are less suitable for sparse crowds. Force-based discrete approaches, such as the recently proposed power-law model [Karamouzas et al. 2014], are well suited for sparse crowds, but can be computationally expensive and may require smaller time steps due to explicit time integration.

In this paper, we use Position-Based Dynamics (PBD) [Müller et al. 2007], as an alternative discrete algorithm for simulating both dense and sparse crowds. While more carefully designed models, such as the social force model [Helbing and Molnar 1995] and the power law model [Karamouzas et al. 2014], can yield some realistic crowd behaviors, they occasionally require elaborate numerical treatments to remain stable and robust. Given the success of PBD in simulating various solid and fluid materials in real-time physics, our work further extends the idea to crowd simulation.

Our objective is therefore to offer a numerical framework for crowd simulation that is robust, stable, and easy to implement, ideally for use in interactive games. Due to the flexibility of PBD in defining positional constraints among particles, our proposed framework provides a new platform for artistic design and control of agent behaviors in crowd modeling and animation. Furthermore, we adopt the PBD approach since it is an unconditionally stable implicit scheme. Even though it may not always converge to the solution manifold, a nonlinear Gauss-Seidel-like constraint projection enables the algorithm to produce satisfactory results with modest computational cost suitable for real-time applications. Additionally, the resulting solution scheme is easy to implement and does not require any linear solves.

1.1 Contributions

This paper, which extends [Weiss et al. 2017], makes the following contributions:

- We show how crowds can be simulated within the PBD framework by augmenting it with non-passive agent-based planning velocity.
- We adopt position-based frictional contact constraints of granular materials to model local collision avoidance among nearby agents. An XSPH viscosity term is also added to approximate coherent and collective group behavior.
- We develop a novel long-range collision avoidance constraint to deal with anticipatory collisions. Our model permits the natural development of agent groups.
- We demonstrate multi-species crowd coupling by supporting spatially varying Lagrangian physical properties.

1.2 Overview

The remainder of the paper is organized as follows: Section 2 surveys relevant prior work on crowd simulation and PBD. Section 3 overviews our algorithmic approach. Section 4 discusses algorithmic details and detailed constraint design. We present our simulation results in Section 5. Section 6 concludes the paper with a discussion of our method's limitations and future work.

2 RELATED WORK

Position-Based Dynamics (PBD) was first introduced by Müller et al. [2007] for the fast simulation of deformable objects through the Gauss-Seidel projection of positional constraints. Since then, PBD and Nucleus, a closely-related constraint solver by Stam [2009], have become popular in physics-based animation for their simplicity and robustness. Macklin et al. [2014] presented a unified PBD solver for various natural phenomena. XPBD was proposed recently to eliminate the iteration count and time step dependence of PBD [Macklin et al. 2016]. Even though PBD traditionally defines geometric constraints among particles, it can also approximate force responses from continuum mechanics. Bender et al. [2014a] formulated continuum energies as PBD constraints. The close relationship between PBD and popular continuum-mechanics-based discretization was further explored in the recent work on optimization-based methods for real-time animation [Bouaziz et al. 2014; Liu et al. 2013; Narain et al. 2016; Wang 2015]. A more complete survey of PBD is provided by Bender et al. [2014b].

Efficient, natural, and stable collision avoidance in sparse and dense distributions of agents remains a very active area of research in crowd simulation. In multi-agent simulations, it is essential to capture both individual local behaviors and aggregate collective behaviors. Continuum approaches—such as 'Continuum Crowds' [Treuille et al. 2006], a crowd model that uses continuum dynamics to simulate pedestrian flow—are particularly suitable for dense crowds and complex environments [Jiang et al. 2010]. Unfortunately, the traditional regime of pure continuum models tends to smooth out local agent behaviors, motivating research on hybrid methods. For example, Narain et al. [2009] simulated dense crowds with a hybrid, Eulerian-Lagrangian particle-in-cell approach and the unilateral incompressibility constraint (UIC), which has proven to be an effective assumption for crowds. Subsequently, frictional forces were taken into account in modeling crowd turbulence [Golas et al. 2014; Helbing et al. 2007], which is essential in extra high-density scenarios. This also inspired us to treat dense agent collisions with a frictional contact model similar to PBD dry sand simulation in [Macklin et al. 2014]. To robustly model multiple densities, Golas et al. [2013] proposed a hybrid scheme for simulating high-density and low-density crowds with seamless transitions.

Other techniques for collision avoidance have been proposed. Many researchers adopted force-based models [Helbing et al. 2000; Reynolds 1987, 1999]. An interaction energy between pedestrians was modeled with a power law in recent and concurrent work by Karamouzas et al. [2014; 2017]. As an alternative to forces, the reciprocal velocity obstacle was proposed in robotics for multi-agent navigation [Van den Berg et al. 2008]. Guy et al. [2009] extended velocity obstacles and used a parallel optimization framework for collision avoidance. Ren et al. [2016] augment velocity obstacles with velocity connections to keep agents moving together, thus allowing more coherent behaviors. He et al. [2016] simulated dynamic group behaviors based on the least effort principle. Guy et al. [2010] simulated large-scale crowds by optimization based on the Principle of Least Effort. Bruneau and Pettré [2015] presented a mid-term planning system to fill in the gap between long-term planning and short-term collision avoidance.

3 ALGORITHM OVERVIEW

Our simulation loop per time step is similar to that for PBD, with several modifications. We outline our procedure in Algorithm 1 and highlight the different steps.

Assume that we have N agents. Each agent i, where $i = 1, 2, \ldots, N$, is represented with a fixed-sized particle with position $\boldsymbol{x}_i \in \mathbb{R}^2$ and velocity $\boldsymbol{v}_i \in \mathbb{R}^2$. For multi-species considerations, we treat each

Algorithm 1 Position-Based Crowd simulation loop

```
1: for all agent i do                                          ▷ §4.1
2:     calculate $\boldsymbol{v}_i^p$ from a velocity planner
3:     calculate a blending velocity $\boldsymbol{v}_i^b$ from $\boldsymbol{v}_i^p$ and $\boldsymbol{v}_i^n$
4:     $\boldsymbol{x}_i^* \leftarrow \boldsymbol{x}_i^n + \Delta t \boldsymbol{v}_i^b$
5: end for
6: for all agent i do
7:     find neighboring agents $S_i = \{s_{i1}, s_{i2}, ..., s_{im}\}$
8: end for
9: while iteration count< max stability iterations do
10:    for all agent i do
11:        compute position correction $\Delta \boldsymbol{x}_i$                  ▷ §4.2
12:        $\boldsymbol{x}_i^n \leftarrow \boldsymbol{x}_i^n + \Delta \boldsymbol{x}_i$
13:        $\boldsymbol{x}_i^* \leftarrow \boldsymbol{x}_i^* + \Delta \boldsymbol{x}_i$
14:    end for
15: end while
16: while iteration count< max iterations do
17:    for all agent i do
18:        compute position correction $\Delta \boldsymbol{x}_i$        ▷ §4.2, §4.4, §4.5
19:        $\boldsymbol{x}_i^* \leftarrow \boldsymbol{x}_i^* + \Delta \boldsymbol{x}_i$
20:    end for
21: end while
22: for all agent i do
23:    $\boldsymbol{v}_i^{n+1} \leftarrow (\boldsymbol{x}_i^* - \boldsymbol{x}_i^n)/\Delta t$
24:    Add XSPH viscosity to $\boldsymbol{v}_i^{n+1}$                         ▷ §4.3
25:    Clamp $\boldsymbol{v}_i^{n+1}$                                        ▷ §4.6
26:    $\boldsymbol{x}_i^{n+1} \leftarrow \boldsymbol{x}_i^*$
27: end for
```

particle as a circle with radius r_i and mass m_i. When we are stepping from time n to time $n+1$ in a traditional PBD simulation loop for passive physical simulations, a forward Euler position prediction is first performed as $\boldsymbol{x}_i^* = \boldsymbol{x}_i^n + \Delta t(\boldsymbol{v}_i^n + \Delta t f_{\text{ext}}(\boldsymbol{x}_i^n))$, where f_{ext} represents external forces such as gravity. In position-based crowds, $\boldsymbol{x}_i^*$ needs to be computed differently to take into account the velocity planning of each agent. In particular, we compute $\boldsymbol{x}_i^*$ based on a blending scheme between a preferred velocity and the current velocity $\boldsymbol{v}_i^n$ (see Section 4.1). It is apparent that the predicted $\boldsymbol{x}_i^*$ for a particle completely ignores the existence of any other particles and just passively advects in the velocity field. To resolve this, PBD defines constraint functions on the desired location of the particles. Both equality and inequality constraints are supported, and they can be expressed as $C_k(\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_N) = 0$ and $C_k(\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_N) \geq 0$ respectively. Hence, the task is to search for a correction $\Delta \boldsymbol{x}_i$ such that $\boldsymbol{x}_i^{n+1} = \boldsymbol{x}_i^* + \Delta \boldsymbol{x}_i$ satisfies the constraints. Once the new positions are computed, agent velocities can be updated as $\boldsymbol{v}_i^{n+1} = (\boldsymbol{x}_i^{n+1} - \boldsymbol{x}_i^n)/\Delta t$. This update guarantees stable agent velocities as long as the constraint projection is stable.

4 METHOD

Our position-based formulation includes several modifications to the standard PBD scheme as well as additional constraints for short-range and long-range collision avoidance between agents.

4.1 Velocity Blending

Agent level roadmap velocity planning describes high-level agent behaviors. Local behavior may be influenced by factors such as social or cognitive goals, while global behavior may be specified by a particular walking path. We note that the roadmap planning is an orthogonal component to our constraint based scheme.

In the physics-based simulation of solids and fluids, particles generally retain their existing velocities. In particular, as demonstrated in [Bouaziz et al. 2014], the implicit Euler time integration of a physical system can be formulated as an minimization problem that balances the 'momentum potential' $\|\boldsymbol{M}^{1/2}(\boldsymbol{x}-(\boldsymbol{x}^n+\Delta t\boldsymbol{v}^n))\|_F^2/2\Delta t^2$ and other potential energies, where $\boldsymbol{M}$ is the mass matrix. In a multi-agent crowd simulation, it is similarly more desirable to include the inertia effect before predicting an agent's desired velocity. Denoting the preferred velocity given the planner with $\boldsymbol{v}_i^p$, we calculate the agent velocity $\boldsymbol{v}_i^b$ as a linear blending between $\boldsymbol{v}_i^p$ and the current velocity $\boldsymbol{v}_i^n$, as follows:

$$\boldsymbol{v}_i^b = (1-\alpha)\boldsymbol{v}_i^n + \alpha \boldsymbol{v}_i^p, \tag{1}$$

where $\alpha \in [0,1]$. We set $\alpha = 0.0385$ in all our simulations. A more adaptive choice, such as the density-based blending factor as in [Narain et al. 2009], can also be used in our framework.

4.2 Frictional Contact

We model local particle contacts with an inequality distance constraint as in standard position-based methods:

$$C(\boldsymbol{x}_i, \boldsymbol{x}_j) = \|\boldsymbol{x}_i - \boldsymbol{x}_j\| - (r_i + r_j) \geq 0, \tag{2}$$

where r_i and r_j are the radii of agents i and j. To model frictional behavior between neighboring agents, we further adopt kinematic frictions as described in [Macklin et al. 2014].

4.3 Cohesion

To encourage more coherent agent motions, we add the artificial XSPH viscosity [Macklin and Müller 2013; Schechter and Bridson 2012] to the updated agent velocities. Specifically,

$$\boldsymbol{v}_i \leftarrow \boldsymbol{v}_i + c\sum_j (\boldsymbol{v}_i - \boldsymbol{v}_j) W(\boldsymbol{x}_i - \boldsymbol{x}_j, h), \tag{3}$$

where $W(\boldsymbol{r}, h)$ is the Poly6 kernel for SPH [Macklin and Müller 2013]. For our simulations, with particles with radius 1, we use $h = 7$ and $c = 217$.

4.4 Long Range Collision

Karamouzas et al. [2014] describe an explicit force-based scheme for modeling crowds. We design a similar scheme as a position-based constraint. As in their power law setting, the leading term is the time to collision τ, defined as the time when two disks representing particles i and j touch each other in the future. As in [Karamouzas et al. 2014], it can be shown that

$$\tau = \frac{b - \sqrt{b^2 - ac}}{a}, \tag{4}$$

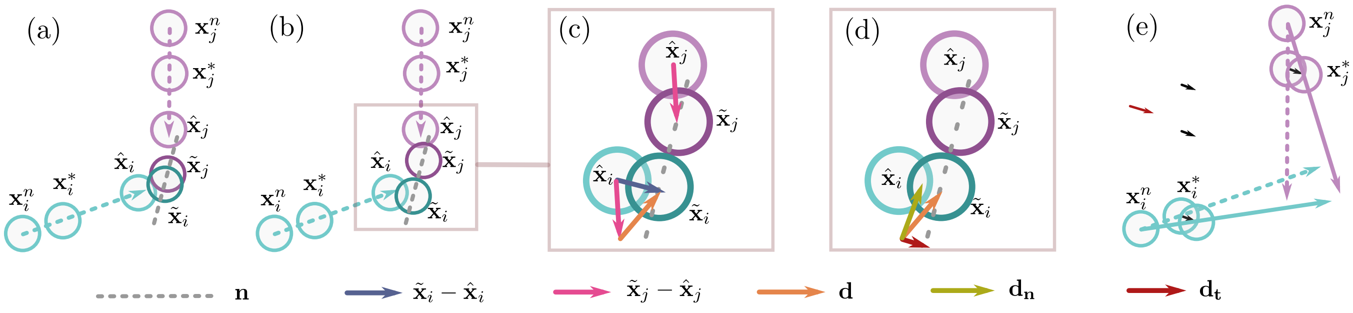

Figure 2: Avoidance model for predictive collision avoidance. (a) Starting with particles in current positions x_i^n and x_j^n, PBD estimates their positions x_i^* and x_j^* at the next time step. To further predict behaviors in the future, we estimate a discrete time to collision $\hat{\tau}$ using their trajectories. This results in $\hat{x}_{i,j} = x_{i,j}^n + \hat{\tau} v_{i,j}$. When further advanced in time by Δt, particles collide at $\tilde{x}_i$ and $\tilde{x}_j$. (b) Projecting these collision constraints resolves the collision between $\tilde{x}_i$ and $\tilde{x}_j$. (c) We compute the relative displacement d from time $\hat{\tau}$ to $\tilde{\tau}$. (d) d is decomposed into contact normal (d_n) and tangential (d_t) components. (e) The tangential contribution of the relative displacement is distributed to x_i^* and x_j^*, which results in an avoidance resolution of future contacts.

where

$$a = \frac{1}{\Delta t^2}\|\boldsymbol{x}_i^* - \boldsymbol{x}_j^*\|^2, \tag{5}$$

$$b = -\frac{1}{\Delta t}(\boldsymbol{x}_i - \boldsymbol{x}_j) \cdot (\boldsymbol{x}_i^* - \boldsymbol{x}_j^*), \tag{6}$$

$$c = \|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2 - (r_i + r_j)^2. \tag{7}$$

No potential energies associated with forces are required in our framework. To facilitate collision-free states in the future, we directly apply a collision-free constraint on future positions. Recall that in our simulation loop, the predicted position of particles i and j in the next time step are

$$\boldsymbol{x}_{i,j}^* = \boldsymbol{x}_{i,j}^n + \Delta t \boldsymbol{v}_{i,j}, \tag{8}$$

where $\boldsymbol{v}_{i,j}^b$ is defined in (1) , and we use the i, j subscripts to denote that the above is defined exclusively in the context of i or j.

We estimate a future collision state between i and j using τ. We first compute the exact time to collision using (4). Valid cases are those with $\tau > 0$ and $\tau < \tau_0$, where τ_0 is a fixed constant. We used $\tau_0 = 20$ in all our experiments. After pruning out invalid cases, we process the remaining colliding pairs in parallel (Section 5.1). We define $\hat{\tau} = \Delta t * \lfloor \tau / \Delta t \rfloor$, where $\lfloor \cdot \rfloor$ denotes the floor operator. This is simply clamping τ to find a discrete time spot slightly before the predicted contact. With $\hat{\tau}$, we have

$$\hat{\boldsymbol{x}}_{i,j} = \boldsymbol{x}_{i,j}^n + \hat{\tau} \boldsymbol{v}_{i,j}. \tag{9}$$

Note $\hat{\boldsymbol{x}}_{i,j}$ are similar to $\boldsymbol{x}_{i,j}^n$ in the traditional collision constraint case (2) and are still in a collision free state. Stepping forward will cause the actual penetration. We define the colliding positions with

$$\tilde{\boldsymbol{x}_{i,j}} = \boldsymbol{x}_{i,j}^n + \tilde{\tau} \boldsymbol{v}_{i,j}, \tag{10}$$

where $\tilde{\tau} = \Delta t + \hat{\tau}$. We enforce a collision free constraint on $\tilde{\boldsymbol{x}}_i$ and $\tilde{\boldsymbol{x}}_j$. Note that $\tilde{\boldsymbol{x}}_{i,j}$ is a function of $\boldsymbol{x}_{i,j}^*$; therefore, it is still essentially a constraint on $\boldsymbol{x}_{i,j}^*$. Due to its anticipatory nature, high stiffness on this constraint is not necessary. To prevent over-stiff behaviors, instead of using the overlap between the predicted particle locations, we define the stiffness to be $k \exp(-\hat{\tau}^2/\tau_0)$, where k is a user-specified constant.

4.5 Avoidance Model

We further present a novel avoidance model for crowd collision. The long-range collision constraint from Section 4.4 will cause agents to slow down due to motion along the contact normal from the collision resolve, which is often not desirable in dense scenarios (Fig. 1). However, we observe that the tangential component of that collision response is often desired, effectively causing the agents to simply slide in response to the predicted collision. Hence, we preserve only the tangential movement in such collisions. We calculate the total relative displacement as

$$\boldsymbol{d} = (\tilde{\boldsymbol{x}}_i - \hat{\boldsymbol{x}}_i) - (\tilde{\boldsymbol{x}}_j - \hat{\boldsymbol{x}}_j), \tag{11}$$

which can be decomposed into contact normal and tangential components as follows:

$$\boldsymbol{d_n} = (\boldsymbol{d} \cdot \boldsymbol{n})\boldsymbol{n}, \tag{12}$$

$$\boldsymbol{d_t} = \boldsymbol{d} - \boldsymbol{d_n}, \tag{13}$$

where $\boldsymbol{n} = (\tilde{\boldsymbol{x}}_i - \tilde{\boldsymbol{x}}_j)/\|\tilde{\boldsymbol{x}}_i - \tilde{\boldsymbol{x}}_j\|$ is the contact normal. To this end, we preserve only the tangential component in the positional correction to $\boldsymbol{x}_{i,j}^*$. This provides an avoidance behavior and prevents agents from being pushed back in a dense flow. Fig. 2 illustrates this process.

4.6 Maximum Speed and Acceleration Limiting

After the constraint solve, we further clamp the maximum speed and acceleration of the agents to better approximate real human capabilities.

4.7 Walls and Obstacles

Agents can interact with walls and other static obstacles in the environment. We prevent agents locomoting into walls and other static obstacles by a traditional collision response (2), between the agent's predicted position and the nearest point on the obstacle. The obstacle's collision point is assigned infinite mass.

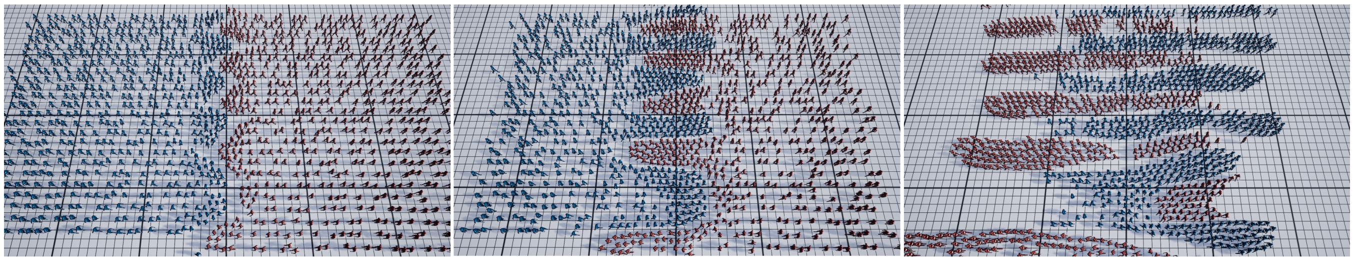

Figure 4: Groups passing each other using the avoidance model. Left: Groups of agents organize into a boundary front in preparation for collision avoidance. Middle: Agents huddle together in noticeable thick lanes. Right: Agents successfully pass each other.

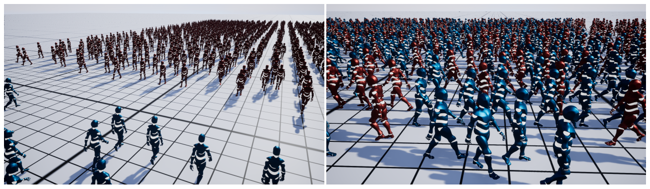

Figure 3: Two groups of agents passing through each other using our long range collision avoidance.

5 EXPERIMENTS AND RESULTS

5.1 Setup and Parameter settings

We implemented our framework in CUDA, using an NVIDIA GeForce GT 750M. We set $\Delta t = 1/48$ sec for all the experiments (2 substeps per frame). We solve each constraint group in parallel, employing a Jacobi solver, with a delta averaging coefficient of 1.2. To find neighboring agents, we use two hash-grids, for short and long range collisions. This is more efficient than using one grid for both, since the long range grid covers a bigger collision radius. Each grid is constructed efficiently and in parallel. See [Green 2008; Macklin et al. 2014] for additional details.

In our simulations, we use 1 stability iteration to resolve contact constraints possibly remaining from the previous time step, and 6 iterations in the constraint solve loop. Additional iterations can increase stability and smoothness, but at increased computational cost.

For agent rendering and locomotion synthesis, we used Unreal Engine 4.15. For smooth locomotion, we clamped the agent's skeletal positional acceleration and rotational velocity. Additionally, we applied a uniform motion scaling of about 30. We rendered the motion at about 5 times the simulation rate.

We demonstrated the robustness of our position-based framework in a variety of scenarios. To simplify the experiment setup and unless otherwise stated, we modeled all agents using a disk with radius 0.5, and use the same width for our humanoid agents in the rendering stage. For smoother motion, we allow an expansion of the agent's disk radius by 5% during collision checks. For each benchmark, we used a simple preferred velocity planner, where the preferred velocity of each agent points to the closest user-scripted goal. We also slightly varied the preferred velocity of each agent around a mean of 1.4, to achieve a more realistic simulation. Table 1 presents timing information.

	# agents	LR	A	ms/frame
Sparse passing	1,600	On	-	11.27
Sparse passing	1,600	-	On	11.61
Dense, low count	1,600	On	-	12.03
Dense, low count	1,600	-	On	11.34
Dense, high count	10,032	On	-	14.06
Dense, high count	10,032	-	On	13.63
Bears and Rabbits	1,152	-	On	11.86
Dense Ellipsoid	1,920	-	On	10.06
Proximal Behavior	50	On	-	10.12
Proximal Behavior	50	-	On	10.13
Target Locomotion	192	On	-	10.42
Bottleneck	480	-	-	11.99
Bottleneck	3,600	-	-	17.76
Bottleneck	100,048	-	-	43.66

Table 1: Timings. LR: long range collision constraint; A: avoidance model constraint. All experiments use $\Delta t = 1/48$, with 6 iterations per time step. These timings do not include rendering times.

5.2 Benchmarks and Analysis

5.2.1 Sparse Passing (Low Count, Long Range Collision): We experimented with two groups of agents locomoting in opposite directions (Fig. 3). The agents in each group are positioned in a loose grid formation with an initial separation distance. To avoid collisions, the agents use the constraint of Section 4.4. In this scenario, the agents organize into narrow lanes, and pass each other easily.

5.2.2 Sparse Passing (Low Count, Avoidance): This scenario is identical to 5.2.1, but the agents employ the constraint of Section 4.5 to avoid collisions. In this scenario, the agents form thicker lanes (Fig. 4), which accumulate into different groups.

5.2.3 Dense Passing (Low Count, Long Range Collision): A total of 1,600 agents are split into two groups, with a separating distance of 2.5 (Fig. 5). We used a higher and denser crowd of agents. To avoid collision, the agents employ the constraint of Section 4.5. Because of the dense agent setting, the two agent groups do not easily pass each other, and some bottleneck groups are formed. Eventually, the agents pass, avoiding unrealistic collisions.

5.2.4 Dense Passing (Low Count, Avoidance): This experimental setup is identical to 5.2.3. To avoid collision, the agents employ the constraint of Section 4.5. In this scenario, the agents form thicker lanes, which form into different groups (Fig. 5).

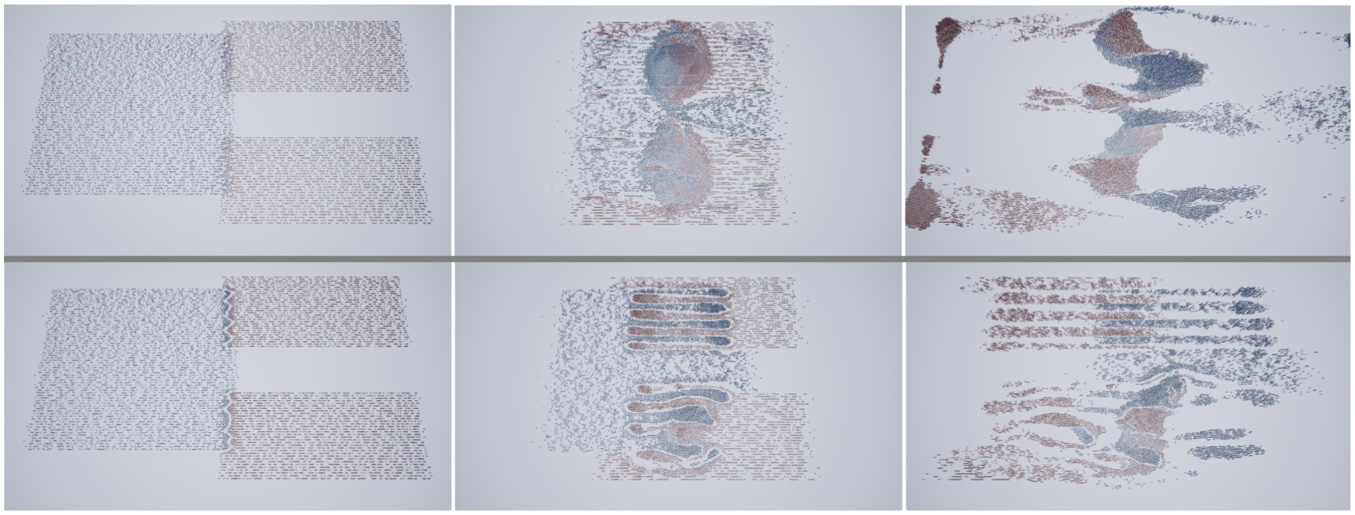

Figure 6: High density and high agent count. Top Row: Agent groups avoid each other using Long Range Collision. Bottom Row: Using the Avoidance model.

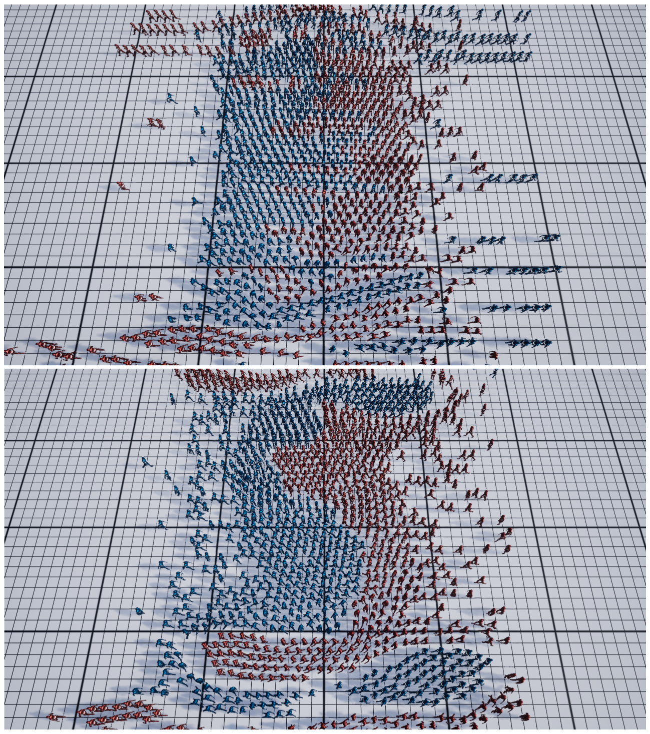

Figure 5: High density agent simulation. Top: Long range collision. Bottom: Avoidance model.

5.2.5 Dense Passing (High Count, Long Range Collision): A total of 10,032 agents are split into two groups (Fig. 6) with a separating distance of 3.5. This experiment setup is identical to 5.2.5.

5.2.6 Dense Passing (High Count, Avoidance): This experiment setup is identical to 5.2.5. To avoid collision, the agents employ the constraint of Section 4.5. In this scenario, the agents form thicker lanes, which form into different groups.

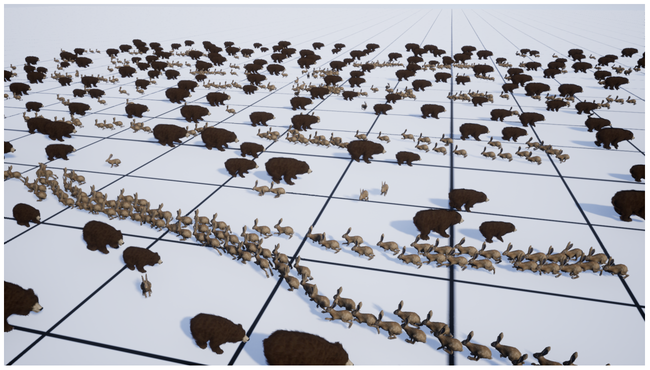

Figure 7: A group of smaller agents (rabbits) passing through a group of larger ones (bears).

5.2.7 Bears and Rabbits: In this experiment, we showcased how a Lagrangian PBD scheme may be employed to model agents of different sizes (Fig. 7). We modeled a group of rabbits passing through a group of bears, totaling 1,152 agents. The rabbits had size 1.0, while the bears had a size ranging from 2.5 to 4.0. To simulate that bears are less prone to change their path than rabbits, we assigned the bears a mass that is approximately 30 times greater than that of the rabbits.

5.2.8 Dense Ellipsoid: This simulation comprises 1,920 agents. To reach their goals, an ellipsoid-shaped group of agents (Fig. 8), with an initial separation distance of 3.3, must locomote through a larger, rectangular group of agents, with a separation distance of 3.0. Throughout the entire simulation, the small group retains its shape and successfully passes the larger group.

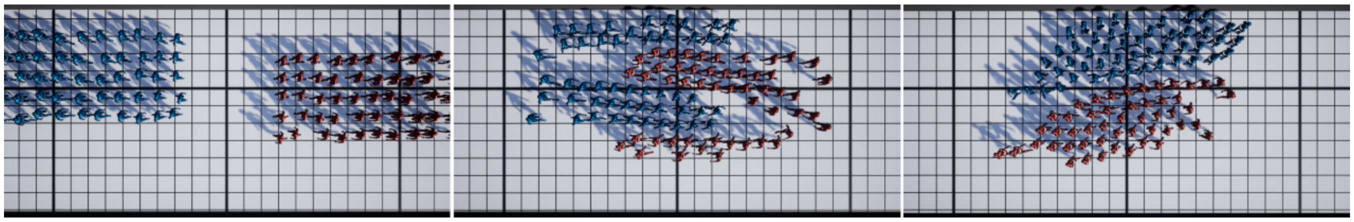

Figure 9: Proxemic group behavior. Left: Initial state. Center: Agents avoid each other using the long-range collision model, while creating lanes. Right: Agents avoid each other using the avoidance model.

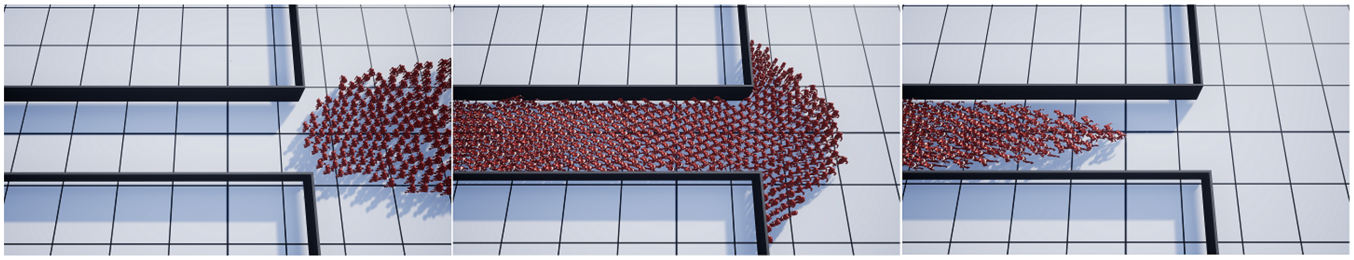

Figure 10: A group of agents passing through a narrow corridor. Left: Agents huddle on approaching corridor's entrance. Middle: A semi-circular arch forms as agents enter a narrow corridor. Right: Agents successfully exit.

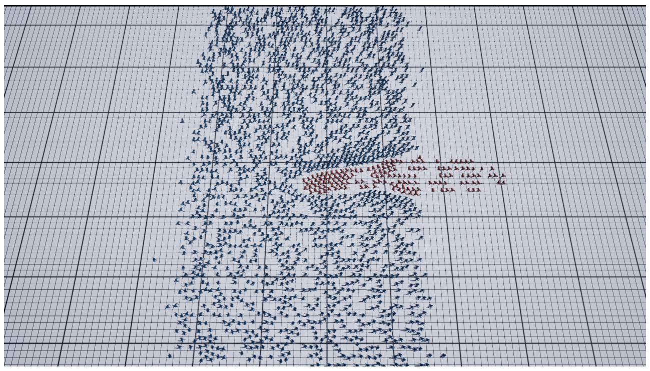

Figure 8: A small ellipsoid shaped group passing through a larger group.

5.2.9 Proximal Behavior, Avoidance Model: Two groups of 50 agents start in tightly packed formations, and must pass each other in a narrow hallway with limited collision avoidance space (Fig. 9). This benchmark demonstrates that our novel avoidance model creates proxemic behavior in agent groups [He et al. 2016].

5.2.10 Proximal Behavior, Long Range Collision: Here, we used the same setting as 5.2.9. We observed lane formation and splitting of the original group.

5.2.11 Target Locomotion, Long Range Collision: 192 agents start in a uniform random grid setting at a separation distance of 5.5. The locomotion targets are in a similar, but in a translated grid pattern, randomly perturbed with additive uniformly distributed random noise. The objective of this benchmark was to show that agents are able to reach their respective goal with minimal interference.

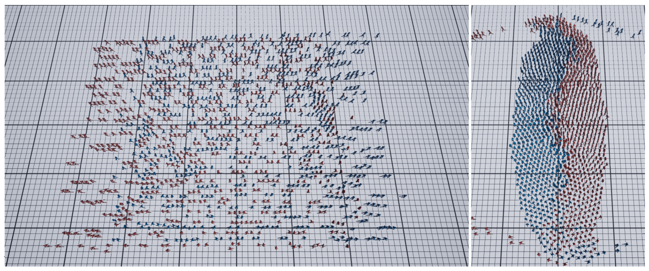

Figure 11: Explicit force-based power law [Karamouzas et al. 2014]. Left: In a sparse setting, the agents successfully avoid collisions. Right: In a dense setting, the agents collide, overlap, and are not able to pass smoothly.

5.2.12 Bottleneck: We demonstrated our method on a bottleneck scenario with varying number of agents. Agents must pass through a narrow corridor to reach their goal (Fig. 10). In this scenario, we observed jamming and arching near the corridor's entrance, as well as the formation of pockets, a phenomena observed in realistic crowds, which was also reported in [Golas et al. 2014; Guy et al. 2010].

5.3 Comparison

The method described in [Karamouzas et al. 2014] is considered the state-of-the-art model for explicit force-based modeling of pedestrian behavior, and it has been validated against human behavior. We implemented this method based on code obtained from the authors. For the comparison, we chose the same parameter settings and time-step as in our method (Section 5.1). Using 1,344 agents, we preformed experiments in the two following settings (Fig. 11):

5.3.1 Crowd Passing (Sparse): For the sparse setting, we used a separating distance of approximately 4.5 between agents. Agents preformed well and avoided collisions, managing to pass with minimal interference to the opposing group. Lane patterns emerged.

5.3.2 Crowd Passing (Dense): In the dense setting, we used a separation distance of approximately 3.3. In this setting, the agents were not able to maintain their trajectory or avoid collisions with the opposing group. Some of these collisions were not resolved, leading to an unrealistic state for almost half of the simulation. Our supplemental video offers additional details.

From the above experiments, we noticed that the power law method does not provide a collision-free model for dense crowds. Nevertheless, careful parameter tuning or increasingly small time steps may help, albeit at the expense of efficiency and ease of use.

6 DISCUSSION

In this paper, we adapted Position-Based Dynamics (PBD) as an alternative discrete algorithm for simulating multi-agent dynamics. Our machinery demonstrated interesting group interactions, such as groups passing each other seamlessly, as well as the formation of traffic lanes and subgroups with minimal interference. We demonstrated our novel PBD method on groups of agents of various sizes, arranged in varying densities, using different mixtures of PBD constraints. We presented novel long range collision constraints with adaptive stiffness, which serve as a realistic preconditioner for the actual collision from frictional contact, with a sufficient stiffness that enforces non penetration. Our solution is flexible and produces interesting patterns and emergent behavior. Compared to existing methods, the advantages of PBD are large time steps, guaranteed stability, and ease of control. In addition, our approach allows simple integration into a preexisting PBD framework. By adding new constraints, our robust, parallel framework can easily incorporate more complex crowd behaviors with minimal run time cost.

Nonetheless, our approach has some limitations. We do not pretend to simulate real pedestrians (cf. [Shao and Terzopoulos 2007; Yu and Terzopoulos 2007]). Designing metrics to evaluate such realism is a problem in and of itself, and it is outside the scope of our present work, but we will investigate this topic in future work, including further quantitative analysis of time-to-collision and other anticipatory position analysis. Even though PBD is a simple and stable framework, it requires a certain amount of parameter tuning. We also plan to explore other constraints, such as clamping the magnitude of turning and backwards motion of agents. We believe that such a constraint will lead to more realistic results. Finally, experimenting with other online locomotion synthesis methods such as motion fields [Lee et al. 2010] can lead to more interesting agent interactions.

ACKNOWLEDGMENTS

We thank Rahul Narain for valuable discussions. Jiang gratefully acknowledges a GPU donation from NVIDIA Corporation and a gift from Awowd, Inc.

REFERENCES

J. Bender, D. Koschier, P. Charrier, and D. Weber. 2014a. Position-based simulation of continuous materials. *Comp Graph* 44 (2014), 1–10.

J. Bender, M. Müller, M.A. Otaduy, M. Teschner, and M. Macklin. 2014b. A survey on position-based simulation methods in computer graphics. In *Comp Graph Forum*, Vol. 33. 228–251.

S. Bouaziz, S. Martin, T. Liu, L. Kavan, and M. Pauly. 2014. Projective dynamics: Fusing constraint projections for fast simulation. *ACM Trans Graph* 33, 4 (2014), 154:1–154:11.

J. Bruneau and J. Pettré. 2015. Energy-efficient mid-term strategies for collision avoidance in crowd simulation. In *Symp Comp Anim.* 119–127.

A. Golas, R. Narain, and M. Lin. 2013. hybrid long-range collision avoidance for crowd simulation. In *Sym Inter 3D Graph Games (I3D '13)*. 29–36.

A. Golas, R. Narain, and M. Lin. 2014. Continuum modeling of crowd turbulence. *Phys Rev E* 90 (2014), 042816. Issue 4.

S. Green. 2008. Cuda particles. *NVIDIA Whitepaper* 2, 3.2 (2008).

S.J. Guy, J. Chhugani, S. Curtis, P. Dubey, M. Lin, and D. Manocha. 2010. PLEdestrians: A least-effort approach to crowd simulation. In *Symp Comp Anim (SCA '10)*. 119–128.

S.J. Guy, J. Chhugani, C. Kim, N. Satish, M. Lin, D. Manocha, and P. Dubey. 2009. ClearPath: Highly parallel collision avoidance for multi-agent simulation. In *Symp Comp Anim (SCA '09)*. 177–187.

L. He, J. Pan, S. Narang, and D. Manocha. 2016. Dynamic Group Behaviors for Interactive Crowd Simulation. In *Symp Comp Anim.* 139–147.

D. Helbing, I. Farkas, and T. Vicsek. 2000. Simulating dynamical features of escape panic. *Nature* 407, 6803 (2000), 487–490.

D. Helbing, A. Johansson, and H.Z. Al-Abideen. 2007. Dynamics of crowd disasters: An empirical study. *Phys Rev E* 75, 4 (2007), 046109.

D. Helbing and P. Molnar. 1995. Social force model for pedestrian dynamics. *Phys Rev E* 51, 5 (1995), 4282.

H. Jiang, W. Xu, T. Mao, C. Li, S. Xia, and Z. Wang. 2010. Continuum crowd simulation in complex environments. *Comp Graph* 34, 5 (2010), 537 – 544.

I. Karamouzas, B. Skinner, and S.J. Guy. 2014. Universal power law governing pedestrian interactions. *Phys Rev Lett* 113 (Dec 2014), 238701. Issue 23.

I. Karamouzas, N. Sohre, R. Narain, and S.J. Guy. 2017. Implicit crowds: Optimization integrator for robust crowd simulation. *ACM Trans Graph* 36, 4 (2017).

Y. Lee, K. Wampler, G. Bernstein, J. Popović, and Z. Popović. 2010. Motion fields for interactive character locomotion. *ACM Trans Graph* 29, 6 (2010), 138.

T. Liu, A. Bargteil, J. O'Brien, and L. Kavan. 2013. Fast simulation of mass-spring systems. *ACM Trans Graph* 32, 6 (2013), 209:1–7.

M. Macklin and M. Müller. 2013. Position based fluids. *ACM Trans Graph* 32, 4 (2013), 104:1–104:12.

M. Macklin, M. Müller, and N. Chentanez. 2016. XPBD: Position-based simulation of compliant constrained dynamics. In *Motion in Games (MIG '16)*. 49–54.

M. Macklin, M. Müller, N. Chentanez, and T. Kim. 2014. Unified particle physics for real-time applications. *ACM Trans Graph* 33, 4 (2014), 153:1–153:12.

M. Müller, B. Heidelberger, M. Hennix, and J. Ratcliff. 2007. Position based dynamics. *J Vis Comm Imag Repre* 18, 2 (2007), 109–118.

R. Narain, A. Golas, S. Curtis, and M.C. Lin. 2009. Aggregate dynamics for dense crowd simulation. *ACM Trans Graph* 28, 5 (2009), 122:1–122:8.

R. Narain, M. Overby, and G.E. Brown. 2016. ADMM $\supseteq$ projective dynamics: Fast simulation of general constitutive models. In *Symp Comp Anim.* 21–28.

Z. Ren, P. Charalambous, J. Bruneau, Q. Peng, and J. Pettré. 2016. Group modeling: A unified velocity-based approach. In *Comp Graph Forum*. Wiley Online Library.

C.W. Reynolds. 1987. Flocks, herds and schools: A distributed behavioral model. *Comput Graph* 21, 4 (Aug. 1987), 25–34.

C.W. Reynolds. 1999. Steering behaviors for autonomous characters. In *Game Dev Conf*, Vol. 1999. 763–782.

H. Schechter and R. Bridson. 2012. Ghost SPH for animating water. *ACM Trans Graph* 31, 4 (2012), 61:1–61:8.

W. Shao and D. Terzopoulos. 2007. Autonomous pedestrians. *Graph Models* 69, 5-6 (2007), 246–274.

J. Stam. 2009. Nucleus: Towards a unified dynamics solver for computer graphics. In *Comp Design Comp Graph*. IEEE, 1–11.

D. Thalmann. 2007. *Crowd Simulation*. Wiley Online Library.

A. Treuille, S. Cooper, and Z. Popović. 2006. Continuum crowds. *ACM Trans Graph* 25, 3 (July 2006), 1160–1168.

J. Van den Berg, M. Lin, and D. Manocha. 2008. Reciprocal velocity obstacles for real-time multi-agent navigation. In *ICRA*. IEEE, 1928–1935.

H. Wang. 2015. A Chebyshev semi-iterative approach for accelerating projective and position-based dynamics. *ACM Trans Graph* 34, 6 (2015), 246:1–246:9.

T. Weiss, A. Litteneker, C. Jiang, and D. Terzopoulos. 2017. Position-based multi-agent dynamics for real-time crowd simulation. In *Proc ACM SIGGRAPH/Eurograph Symp Comp Anim.* ACM, Article 27, 2 pages. Extended abstract.

Q. Yu and D. Terzopoulos. 2007. A decision network framework for the behavioral animation of virtual humans. In *Symp Comp Anim (SCA '07)*. 119–128.