# Centroidal Particles for Interactive Crowd Simulation

**Omar Hesham** **Gabriel Wainer**
Department of Systems and Computer Engineering
Carleton University, Ottawa, Canada
omar.hesham@carleton.ca, gwainer@sce.carleton.ca

**ABSTRACT**
Real-time crowd simulation is a challenging task that demands a careful consideration of the classic trade-off between accuracy and efficiency. Existing particle-based methods have seen success in simulating crowd scenarios for various applications in the architecture, military, urban planning, robotics, and entertainment (film and gaming) industries. In this paper we focus on local dynamics and present an area-based penalty force that captures the infringement of each entity's personal space. This method does not necessitate a costly nearest-neighbor search and allows for an inherently data-parallel implementation that is capable of simulating thousands of entities at interactive frame-rates. The algorithm successfully reproduces personal space compression around motion barriers for moving crowds and around points of interest for static crowds.

**Author Keywords**
Crowd; penalty force; interactive; GPGPU

**ACM Classification Keywords**
I.6.5 [Simulation and Modeling]: Model Development; I.3.7 [Computer Graphics]: Animation; I.2.11 [Artificial Intelligence]: Distributed Artificial Intelligence.

## 1. INTRODUCTION

The Modeling and Simulation (M&S) of virtual crowds has found a certain appeal as an illustrative tool in urban and architectural projects. It allows designers and engineers to visualize the utility of their project's space and facilitates the feedback, discussion, and decision-making among all stakeholders, be they technical, business-oriented, or otherwise.

A more demanding tier of crowd simulation is seen in the entertainment and serious gaming industries. While theoretical accuracy is desired in this context, the practical concerns favour other objectives: visual believability, stability in an unpredictable environment, and the performance to meet the real-time requirements of interactive experiences. Creative applications further demand a certain degree of artistic control. In a similar vein, serious gaming, which often involves interactive scenarios for training, education, and social purposes, tends to aim for model accuracy but ends up favoring performance that maintains a fluid simulated scenario for the learner. Advances in parallel computing and ever increasing hardware affordability are helping close the gap towards accuracy in interactive simulation, including on low-power and mobile devices.

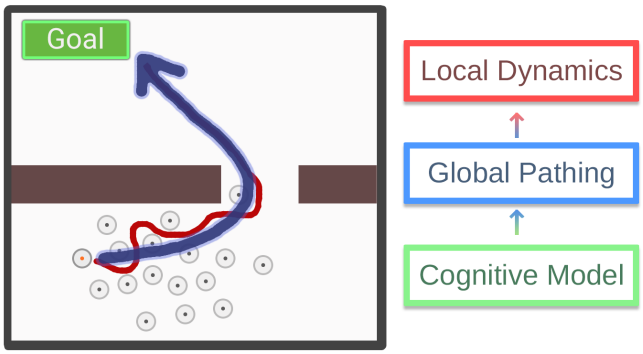


**Figure 1**. Crowd abstraction hierarchy.

The most demanding tier of crowd simulation applications comprises civil safety analysis tools, contingency planning, and military applications. The simulation of emergency evacuation procedures, prediction of traffic bottlenecks, and shaping of pedestrian flow by manipulating access point allocations and doorway designs, are all examples of design-stage activities in this tier. They help the authorities assess threats and plan preventative measures that minimize the risk of disasters in large crowded events such as outdoor music festivals, public forums, and sporting events [1]. Due to their critical nature, the models typically require a high degree of validated accuracy, data-driven calibration, and conformity with existing modeling standards such as High-Level Architecture (HLA) or with industrial formats such as Building Information Modeling (BIM) [2].

This diversity of application requirements induced an equally rich variety of approaches to the modeling and abstraction of crowd behaviour. A fairly common hierarchy of systems divides the process into three interoperating levels: a cognitive model, a global pathing model, and a local interaction model. As illustrated in Figure 1, the cognitive model is typically tasked with broad decision-making, such as deciding where the target location is, and interacting with entity goals and personality traits that could alter such decisions (e.g. taking the stairs instead of the elevator, or following a parent during an evacuation instead of taking the nearest exit). Once the target location is decided, the global pathing module analyzes the spacial structure of the mostly

SCSC 2016, July 24-27, 2016, Montreal, Quebec, Canada

static environment and finds an optimal path according to some cost minimization; typically the shortest path or the least congested one. Finally, the local interaction module further modifies the path to navigate around and avoid collision with minor dynamic obstacles, which include other pedestrians, gates, and doorways, while still generally following the optimal path and not straying too far from it.

While there are methods that blur the lines a bit and attempt to solve more than one level at once (e.g. [3]), the modular hierarchy approach encourages the separation of concerns and allows further experimentation and mixing of components and solutions from various sources.

Our contribution focuses on local dynamics in an interactive entertainment and serious gaming context. To this end, we propose an area-based penalty force and demonstrate promising results obtained from consumer-grade graphics hardware. The model successfully reproduces empirically known crowd phenomenon such as lane formation in bidirectional flow and regular banding around areas of congestion [4]. In addition, our model is capable of reproducing a still compression effect, which is the accumulative reduction of personal space near areas of high congestion and around points of interest (e.g. near the stage at a concert).

This is a challenging effect to simulate for static/near-static crowds, where previous methods relied mostly on the relative velocities between entities and would have a difficult time distinguishing between static entities at varying distances from a barrier or an area of collective interest [5, 6]. As demonstrated in Figure 2, our area-based force is able to aggregate such a compression of personal space in high density stationary crowds. Furthermore, the area force is customizable through graphical parameters that are designer-friendly for creative control and for the introduction of non-homogeneity into the system. This local interaction force can be integrated with existing global pathing schemes or used to augment the calculations of other local avoidance methods.

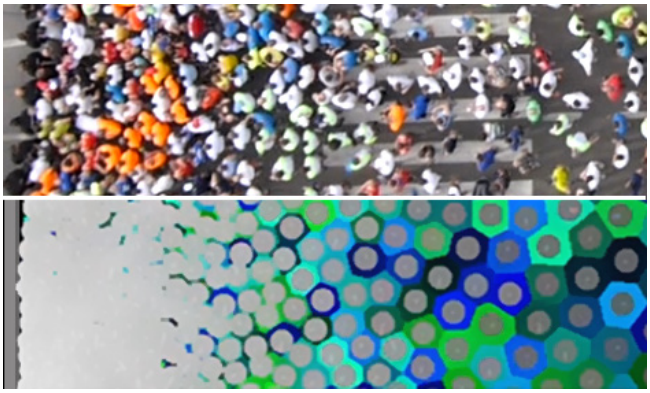

**Figure 2.** An "invisible force": the gradual compression of personal space among a static crowd near areas of interest or barriers to desired motion. Marathon start line (top); our simulated result (bottom).

We discuss the relevant background next, followed by a geometric description of our proposed method and a discretized graphics pipeline implementation. Lastly, we highlight some key results before briefly discussing opportunities for improvement, including thoughts on the conversion from a discrete-time evaluation to a more efficient discrete-event model.

## 2. RELATED WORK

Replicating human decision-making is a highly ambitious endeavor, never mind simulating an entire crowd of them. To this end, the abstraction of motion dynamics based on the generalization of observed phenomenon is necessary to achieving a computable result. This section presents a brief overview of the multitude of methods developed to tackle this problem. For a more detailed analysis, the reader is referred to the critical assessment done in [4, 7].

Early efforts to simulate crowd motion took a macroscopic approach that modeled the crowd collectively as a singular fluid-like object which exhibited interactions over a continuum [8, 9]. Flow-based methods have since evolved, with notable contributions such as Continuum Crowds [3] delivering visually convincing large-scale results at interactive frame rates, suitable for animation, gaming, training, and educational media. Others take an operational research approach, where the simulation space is reduced to a graph on which network optimization methods could be applied [10]. For instance, in a building evacuation scenario, rooms are mapped to graph nodes, and the hallways connecting them are mapped onto graph edges. The graph is solved for the shortest path to identify optimal exit routes. Computing the graph's maximum flow would help identify potential bottlenecks and prioritize future developments around those nodes.

Overall, macroscopic methods are considered computationally inexpensive, making them suitable for large-scale projects that involve high crowd densities. This level of abstraction is most appropriate when seeking a general sense of a crowd's orientation, density distribution, and collective rate of locomotion. They are also suitable to model the physical interactions that cause a transfer of energy in dense crowds, supporting the pre-emptive contingency planning for preventable physical phenomenon such as crowd waves, crushes, and stampedes. However, they typically incorporate a certain degree of aggregation that hides the details of the individual entity-to-entity interactions.

By contrast, microscopic methods rely on simulating each entity individually in an agent-based fashion, and tuning the local interaction laws so that the emergent global behaviour reproduces real crowd phenomenon. Developments in this area include Reynold's flocking model [11], Helbing's social forces [12], HiDAC [6] which incorporates psychological models and pushing behaviour, and a wide array of Cellular Automata (CA) implementations. CA methods are particularly popular in architectural contexts [13] due to their inherent efficiency and rapid visualization. In CA, a

given 2D or 3D space is discretized into a uniform lattice of non-overlapping cells, typically forming a square grid, but can also be based on any tileable shape that forms a consistent neighbourhood pattern. A global clock periodically triggers a simultaneous update of all entities, where the next state of each cell is determined the by the state of all cells in its neighbourhood. When taken to an extreme, microscopic algorithms could also opt to simulate the individual joints of every entity's anatomy (e.g. legs on a human, pedals on a bike, etc.) in order to generate the mechanically accurate locomotion towards a known target location [14].

More recent efforts have evolved from an observation that humans are anticipatory in their motion, able to predict when and where a collision is likely to occur, and enacting avoidance manoeuvres well in advance of such an event. This behaviour was extensively studied and analyzed to produce predictive models [15] that avoid collision in a manner that closely mimics empirical data. Real-time implementations in this category include the Reciprocal Velocity Obstacles (RVO) [16], and a vision-based approach that renders a depth map of the scene from each entity's perspective to evaluate the appropriate avoidance manoeuvres [17].

## 3. METHOD

We propose a personal area-based microscopic method that at dense enough scenarios produces physical interactions, pressure propagation waves, and compression of personal space due to the crowd's collective attraction to an area of interest, where it behaves as a compressible fluid, even when the crowd is stationary. We start with an overview of the processing pipeline followed by a breakdown of the collision penalty force and its parameterization.

### 3.1. Overview

To simulate local pedestrian interactions, we first compute a combined personal space violation map, from which we are able to compute reactive penalty forces. The way each entity contributes to the personal space can be modified and parameterized visually through weight maps. Finally, the new location for each entity is computed by integrating the net acceleration force iteratively over several frames.

### 3.2. Personal Space

Each entity is represented by a particle in the 2D plane surrounded by a contiguous area of personal space (PS) footprint. Based on the experiments done in [15], the personal space is about ~0.8m evenly around the center of the entity. In addition, as the entity gains velocity, there is a further elongation of the personal space in the direction of travel that is directly proportional to the speed. That is, humans typically expect an increasingly empty area ahead of them as they gain speed.

As entities get within proximity of each other, we assume that the personal space becomes shared equidistantly between them. The closer those entities are to one another, the more they equally violate the other's personal space. We also assume, as experiments have shown in [15], that there's a short delay to human response, requiring a brief time (~150-350ms) to react to their surroundings and enact their collision avoidance manoeuvres. From this view, we propose an area-based penalty force that reacts to the PS violation in an iterative manner, attempting to restore the preferred PS area over several frames.

For any point in a given entity's PS to be considered unviolated, it has to be closer to that entity than any other entity. The concept of sharing a space equidistantly as such evokes the tessellations produced by the Voronoi Diagram. In fact, we start by subjecting our entities and their personal space to an analysis similar to the one found in the Centroidal Voronoi Tessellation (CVT) relaxation algorithm [18]. We overlay the combined entity personal spaces onto a shared map, called the personal space map (PSM) that produces a tessellation that clearly aggregates and outlines all PS violations.

Given the PSM, each entity can independently compute the current unviolated area's centroid (i.e. its new geometric center). In CVT relaxation, the particle is simply moved to the new centroid, whereas in this calculation we treat the vector from the center of the original footprint to this new centroid as the basis for the penalty force calculation.

The force is illustrated in Figure 3. Typically, there's a primary directional force that moves the entity towards its destination, and it's typically given by the global pathing scheme. The assumption is that the entity would reach its goal if it follows that vector, given that there are no obstacles in its path. The penalty force is added to the global pathing force, resulting in the reactive behaviour of our crowd. The magnitudes of those forces and their stochastic variations are adjusted to create a smooth transition to the entity's comfort speed, *cs*. On average, a comfort speed for walking is $1.4 \pm 0.24$m/s [15], but can be further calibrated as described in the next sections.

Due to the iterative nature of the algorithm, we end up with a slight transfer of energy, causing the gradual compression of personal space around areas of congestion for moving crowds and areas of interest for stationary ones. This is a welcomed effect as demonstrated in the results of Section 5.

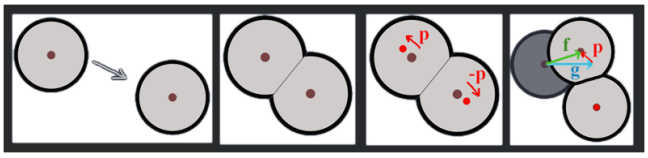


**Figure 3.** The net force (*f*) experienced by an entity is a linear combination of the global pathing force (*g*), and the penalty force (*p*) which falls along the direction of the new centroid.

### 3.3. Particle Parameterization

So far, we've been treating the personal space as a uniformly weighted homogenous area. However, we can further modify the local dynamics by changing the footprint's geometry or using a map to influence its weight. The con-

cept is briefly illustrated in Figure 4. The personal space footprint can be artificially varied over time or in response to global events (e.g. a fire alarm evacuation) or in proximity to points of interest (e.g. slowing down when window shopping or near interesting booths at a busy exhibit hall). The footprint shape can also change to reflect the entity type (e.g. adult, child, stroller, etc.).

The footprint can also accept a weight map which, through simple convolution, varies the impact of the PS infringement. For instance, placing a slightly heavier weight on the right side of the footprint to indicate a preference for taking the "right lane" when encountering oncoming traffic.

These are all very intuitive and graphical parameterizations that are artist/designer-friendly, without requiring deep technical knowledge of the implementation or the code involved. This makes it suitable for manipulation under art direction in a studio setting (e.g. film, gaming) and for educational scenarios that don't conform to ideal conditions, such as having "difficult" or aggressive crowd members, or culling the footprint of a distracted pedestrian (e.g. on their cell phone) to simulate the increased likelihood of such an entity's repeated collision and possible injury.

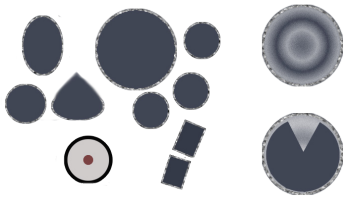

**Figure 4.** Variations of PS footprints via shape and weight maps.

Furthermore, a simple web of 1D springs can be attached between entities that should remain together, such as a family or a row of friends. This allows each entity to still interact with and be affected by its environment independently, yet adds a delayed constraint (the spring) that eventually maintains the group's cohesion whenever possible. Those springs can have strain limits at which point the spring is broken, simulating a member of the group getting lost or stuck amidst a dense crowd away from its group.

### 3.4. Comfort Speed

We employ the notion of the fundamental diagram [19], where pedestrian speeds, on average, vary inversely to their local density. Empirical data has shown that the fundamental diagram differs depending on the context of the crowd (e.g. indoors mall vs. crosswalk) and also across cultures. We use it to determine the comfort speed for each individual entity dynamically throughout the simulation. The density can be computed in a uniform grid, or as we will cover in the implementation section, utilize the personal space area to directly estimate the local density.

## 4. IMPLEMENTATION

The PSM presented in Section 3.2 can be treated as a truncated Voronoi diagram, whose cells are bounded by a certain distance from their sites.

To achieve the high performance desired in interactive applications, we take a cue from the GPU-accelerated computation of Voronoi diagrams in [20] and develop a similar pipeline here.

Textured 3D cones are used to represent the entities and their personal space, with the tip of the cone representing the footprint center, and the base representing the outer edges of the personal space. In effect, the height along the surface of the cone encodes the distance to the center of the entity. When rendered from an orthographic top view (free of any perspective distortion) facing the tips, two cones will overlap at precisely the points that are equidistant to both entities. Figure 5 visualizes this calculation.

By encoding the entities as geometric primitives and using the GPU's depth buffer to quickly obtain the PSM tessellation as discrete pixels, we are left with a shared data structure that allows each entity to compute its relative centroid and resulting penalty forces in a data-parallel fashion. The entities do not need to conduct a costly nearest neighbor search, as they simply consume and interact with the set of pixels representing their personal space in the PSM.

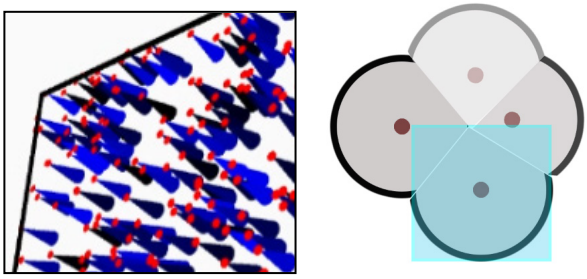

**Figure 5.** Side view of 3D Cones (left); PSM allows each entity to compute forces in a highly localized data-parallel fashion (right).

The mass of the individuals can be modeled by adjusting the height of the cones. The heavier the individual is (less likely to be affected by force), the closer the cone should be to the camera. Essentially, the lighter individuals would have to exert more force in order to make up for their increased distance from the camera and infringe on the heavier individual's space.

In order to differentiate between the rendered cones, they're colored using a reversible hash map that is a function of the unique entity IDs. For our purposes, we reserve the first dozen colors in the 24-bit RGB space for obstacles and debugging purposes, and utilize the rest to uniquely color code each entity. The reverse lookup (which is also a constant cost function) enables any entity to directly identify the ID of another entity infringing on its pixel space.

Hence we can theoretically simulate more than 16 million entities on a 24-bit RGB rendering surface, and potentially billions on higher bit-depth surfaces. Though, practically speaking, 16 million entities would end up using a single pixel each on a 4k×4k PSM map, as a best case-scenario. More on such scalability issues is discussed in Section 4.1.

To compute the true center of PS footprints that have an influence map, vary by time, or are proportional to the entity's speed (as discussed in the Section 3.3), we first calibrate the shape for bias and then adjust the cone tip and texture accordingly. Without this step, an asymmetrical PS area that, for instance, elongates with speed will experience an ever increasing force in the direction of travel. Instead the collision-free bias should be taken into account in order to detect true violations of the PS footprint.

There are two ways to compute the centroid from the PSM:

a) Per-pixel aggregation - if there is overcrowding resulting in significant PS overlap, and then it is more economical to only count the visible pixels and aggregate the results using the reverse hash lookup.

b) Per-entity aggregation - in sparser scenarios, it's more efficient to skip the PSM's empty pixels and simply count the valid pixels in the vicinity of each entity.

A simple threshold we used in our implementation checks: if ($|\text{entities}| \times \pi r^2$) > (1.2×PSM area) perform (a); else perform (b), where $r$ is the footprint radius. We take each PSM pixel to represent 10×10cm, so our PS radii range from 6 to 10 pixels, as sizes vary in a crowd.

### 4.1. Scalability

GPUs are highly efficient when it comes to massively data-parallel processing, but they also have built-in memory limitations on texture sizes, and by extension, the size of the PSM maps we can compute for a given scene. A possible route is to use a courser grid to represent the scene, at the cost of simulation fidelity. The more reasonable approach in this instance is tiling. Regardless of the shape and complexity of the simulation plane, it can be divided into tiles. A straightforward approach that minimizes communication across tiles is to have them overlap their computation space by an amount equal to the average entity PS radius. Similar to how we skip empty pixels in sparse crowds, we extend the concept by skipping entire tiles if they're empty, and only processing the occupied ones.

These tiles can be dispatched for simultaneous processing across multiple GPUs if available. Furthermore, the recent development of low-overhead graphics APIs like Khronos Vulkan and Microsoft DirectX12 encouraging the multi-threaded dispatching of render-calls [21] supports the scalability potential of our presented method.

### 4.2. Density Estimation

To compute the per-entity comfort speed discussed in Section 3.4, we need to obtain an estimation of the local crowd density near each entity. We leverage the existing PSM to estimate the value $d_i$ (entities/m$^2$) as the reciprocal of its visible (unviolated) PS area.

Once known, we render another PSM pass, but this time color every entity $i$ using the estimated $d_i$, where brighter values correspond to higher densities. We then smooth the discontinuities using a Gaussian blur filter; again a very standard and parallel-friendly utilization of the graphics pipeline. The resulting density field, seen in Figure 6, is sampled for the comfort speed calculation in the following time step.

It might be argued that better density estimation methods exist, such as those found in physics simulations using Soft Particle Hydrodynamics (SPH) that rely on cubic kernels to accurately converge to a true density continuum [22]. However, we opted for the simpler Gaussian kernel across our PSM with a radius of half a footprint's radius in effective pixels. Not only is the result sufficient for our purposes (an estimation of density), but it also helps that it *isn't* too precise since the fundamental diagram is only an aggregate of human behaviour that hides individual variations and human inaccuracies that would naturally occur from an entity subconsciously assessing its surrounding. Furthermore, the Gaussian filter is linearly separable meaning we can blur all the rows first using a 1D Gaussian, then blur the columns to obtain the final 2D Gaussian blur. This optimizes the computation of the convolution from a quadratic O($nm^2$) computation to a more linear O($nm$), where $n$ is the number of pixels in space, and $m$ is the radius of the filtering kernel.

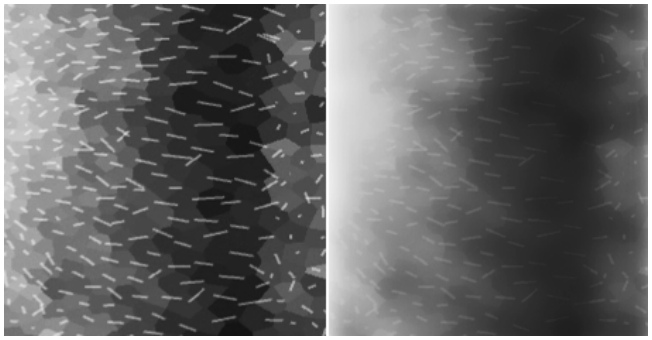

**Figure 6**. Local density is estimated per entity (left) then a global low-pass filter (right) is applied to obtain a smooth field.

## 5. SIMULATION RESULTS

### 5.1. Setup

For each scenario, we initialize entity positions and orientations, and randomize a few parameters such as weight, size, and a comfortable speed base. For demonstration and testing purposes we use simple global pathing schemes such as A*, floor maps, or global vectors to target positions [13]. The simulation time step used is 50ms, as we did not notice an improvement in quality by going with a finer time delta.

Videos and accompanying source code can be obtained from http://vs3.sce.carleton.ca/wordpress/ .

### 5.2. Results

Figures 7-11 illustrate the results, including a variety of emergent phenomenon reproduced by our algorithm. Lane formation in bidirectional flow is an innate pedestrian optimization strategy that is observed in narrow hallways to minimize resistance to their otherwise opposing directions of motion. Figure 7 illustrates this scenario, while Figure 8 displays the performance achievable with two different test

systems. The first is a mobile-class Intel Core i5-3337U 2.7GHz processor with integrated graphics (that is, the CPU is also responsible for all graphical processing), and the second is a desktop AMD A10-7850 3.7 GHz CPU with a dedicated Nvidia GTX970 graphics card. We consider simulations running at 12fps to be reasonably interactive, 8fps a bare minimum experience, and 20fps essentially real-time (given the 50ms simulation time step).

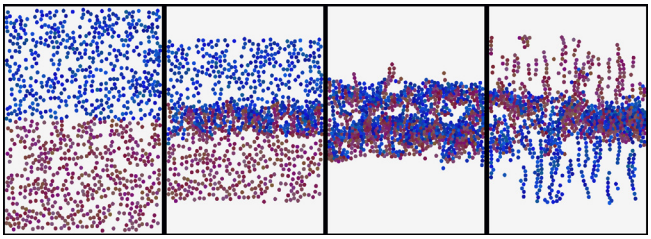

**Figure 7**. Natural lane formation during a bidirectional flow simulation of 1000 entities on a 600×800 PSM grid (blue entities headed south; red headed north).

What is important to note from Figure 2 is that this is not a static distribution or placement algorithm, but rather the emergent behaviour of a fully dynamic local interaction-based force for stationary crowds and pedestrians in motion. In sparse scenarios, the penalty force acts like a microscopic method whose goal is to avoid collision with nearby entities, while in denser cases where collision and physical contact are likely, it starts displaying macroscopic qualities that collectively treat the crowd as a singular fluidic object, exhibiting waves and energy propagation among its individuals.

At less than $0.5m^2$ of personal space per entity, the crowd reaches a critical density [23], where entities are subject to enough pressure to cause significant discomfort, if not outright injury or worse. This is a common concern in large-event contingency planning, and simulation can help identify pockets of potentially unsafe accumulation and overcrowding of attendees [24]. In Figure 9, a crowd of 1000 entities in a confined space is reacting to the fact that they are over-crowded. Requiring neither a cognitive model nor a global pathing scheme, the penalty force is sufficiently able to naturally and gradually restore the compressed crowd to a more comfortable distribution, filling the room if necessary until every entity reaches a suitable state. This can be seen in real life when crowds are held behind barriers and a gate opens allowing the otherwise compressed crowd to diffuse through.

Those on the outer edge of the crowd have less density to deal with, so naturally, as we would expect in reality, they have the greatest freedom to accelerate to higher speeds compared to those that are relatively slowed or still trapped near the middle. We utilize this effect to simulate the emergent behaviour at a marathon race start. As visualized in Figure 10, the wave of delayed acceleration is the result of those at the front of the race having the advantage of lower density ahead of them, allowing them to sprint ahead sooner while those behind are stuck to less competitive speeds, until the delayed wave catches up to them eventually and the speeds equalize. Unlike the compressed room in Figure 9, this scenario has a global goal vector along the marathon track applied evenly to all contestants; but the penalty force is able to reproduce the observed acceleration wave effect.

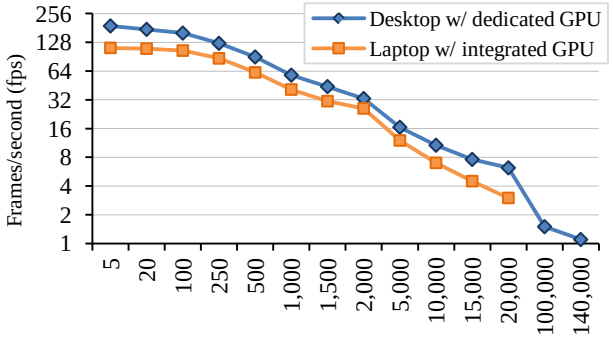


**Figure 8**. Simulation performance of the bidirectional scenario illustrated in Figure 7, varied by number of pedestrians.

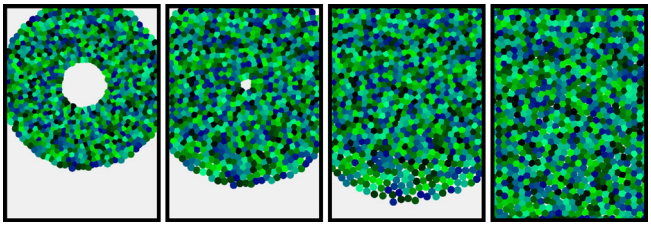

**Figure 9.** Penalty forces are sufficient to cause an overcrowded room (left) to diffuse to a more comfortable equilibrium (right), where any further motion would not improve the situation.

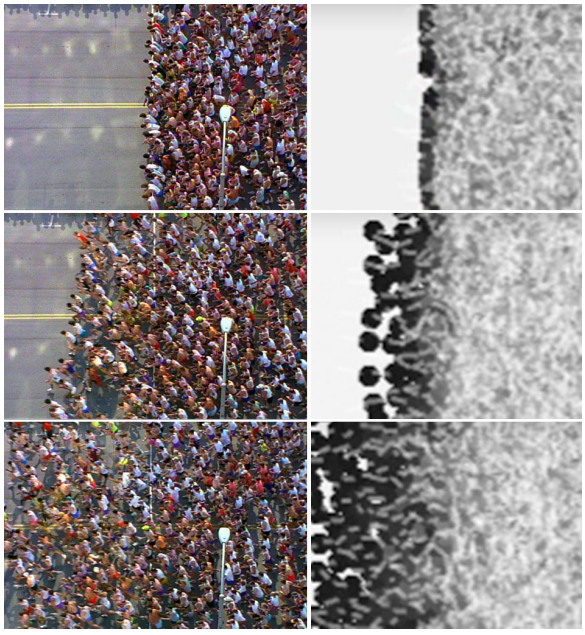

**Figure 10**. The gradual release of density-dependent velocity. Left: snapshots of marathon start; Right: our simulated result.

The compression of personal space in static crowds is rarely seen in political protests or general gatherings of people where there isn't a focal point of interest or a barrier to motion in a desired direction. In such cases, a more even distribution emerges, such as the one in Figure 9.

Huddled crowds, as seen in Figures 2 and 11, have a tendency to fill the space semi-regularly, equalizing the number of neighbours surrounding each entity. They also exhibit a petal-like pattern, with entities preferring to stand between the shoulders of those ahead, instead of directly behind them.

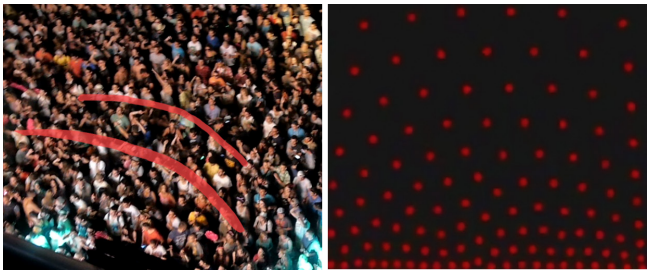

**Figure 11**. At equilibrium, our method reproduces the emergent banding and semi-regular distribution among a static crowd.

### 5.3. A Discrete-Event Approach

The Lagrangian evaluation of crowd dynamics is well-established for discrete-time simulation. But in cases where there is a clear discretization of states and events, it can be worthwhile to pursue a discrete-event implementation of those state variables and their transitions. The key idea here is to perform the state update only when necessary. That is, instead of calculating the motion of the all the entities synchronously at regular time intervals, each entity would instead asynchronously evaluate its own state (e.g. cognitive goals) and the local environment to compute a state transition in an event-driven fashion.

Cellular Automata (CA) and similar Eulerian fixed grid approaches are notably popular in discrete-event crowd simulation [25]. But in order to accommodate freeform and non-axis-aligned structures, we were motivated from the start to instead consider a Lagrangian approach to computing the relevant physical quantities -i.e. the position, orientation, and vision assessment being computed at the vector location of each entity as it moves continuously through space. Even so, we do borrow the fundamental idea from discrete-event Eulerian methods, in that an entity should not update its state unless its local neighbourhood has experienced a significant change that triggered an *event*. This spares a lot of needless computation for entities that have reached equilibrium, and is particularly relevant for this paper where we study stationary crowds.

Our algorithm was designed with this optimization in mind. Specifically, the globally accessible PSM is a convenient way for each entity to independently interact with its neighbourhood; notifying and being notified about changes and relevant events without requiring direct communication with nearby entities. Implementation-wise, each entity is "registered" to listen to the changes in its surrounding *pixels* (not nearest neighbours). Those pixels normally exist in close proximity and exhibit favourable memory access patterns. It's noteworthy that, with careful modelling, an Eulerian grid can be emulated in a Lagrangian engine, maintaining compatibility with existing Eulerian models and tools.

## 6. LIMITATIONS AND OPPORTUNITIES

While this lightweight force is capable of producing a variety of visually convincing emergent crowd behavior on its own, it is equally suitable for integration with existing particle-based methods if desired. For more serious applications that rely on data-driven calibration and rigorous accuracy requirements, the idea of encoding local interactions as geometric primitives requires further study. A worthy pursuit is the geometric encoding of velocity-space collision avoidance schemes, such as RVO optimization, which allow for calibration and the accurate reproduction of kinetic trajectories of microscopic pedestrian interactions. The challenge in describing the RVO scheme geometrically is in its empirically-motivated assumption that collision avoidance is a shared effort between nearby entities, thus requiring that entities know about and share information with their nearest neighbours, something we've been actively avoiding in our performance-minded GPU implementation.

The event-driven approach described earlier in Section 5.3 can be further applied on a per-parameter basis, assigning certain events to trigger specific state variable updates. As the complexity of such event-driven models increases, following a standard modeling formalism such as DEVS (Discrete Event System Specification) can aid in the model's design and management of its complexity [26]. Such formalisms typically provide highly optimized universal simulation engines, allowing the modeller to focus on abstracting the behaviour being studied, without concern for implementation details. Pursuing such an approach would present several advantages compared to plain CA or homebrew event-driven solutions, namely: composability with the existing library of DEVS models and architecture systems (e.g. [27]); interfacing with modeling standards such as HLA; and the ability to submit the model to formal verification, validation, and static analysis techniques [28] (e.g. deadlock detection, unreachable states, etc.).

We began the exploration of this centroidal area force in the context of crowd simulation due to the natural limits of human acceleration and velocity. And while it might be tempting to directly use this force for physics-based simulation of fluids, soft-bodies (such as clay), and granular solids (such as sand), the unpredictable, often chaotic, and extreme accelerations experienced within those bodies require a more careful consideration of the utility of this force. It would be difficult to entirely dismiss the nearest neighbor search if our "implicit collision" response force was to be used for physics-based simulation. However, in its current form, the effort presented in this paper certainly opens the possibility

for less critical applications such as film, gaming, and immersive virtual reality (VR) experiences.

## 7. CONCLUSION

This paper introduces a method to compute a given crowd's local dynamics in an interactive application context, and it demonstrated several emergent patterns of dense crowds in static or near-static conditions. Our main contribution is an area-based penalty force that gradually aggregates the infringement of personal space in order to assess the appropriate response from each entity. We presented an implementation that encodes the proximity information into geometric shapes and textural maps, turns the neighbourhood computation into a matter of simply rendering those shapes through a typical 3D graphics pipeline, and simulates thousands of entities at interactive frame rates.


### Acknowledgements

This research was partially supported by a Queen Elizabeth II Scholarship in Science & Technology. The authors thank Michael Van Schyndel and Chris Joslin for their insights into crowd dynamics, and thank the anonymous reviewers for their critique and valuable feedback.


## REFERENCES


1. Earl, C., Raineri, A. & Others. Crowd management for outdoor music festivals. *Journal of Occupational Health and Safety, Australia and New Zealand* 21, 205 (2005).
2. Azhar, S. Building Information Modeling (BIM): Trends, Benefits, Risks, and Challenges for the AEC Industry. *Leadership and Management in Engineering* 11, 241–252 (2011).
3. Treuille, A., Cooper, S. & Popović, Z. Continuum crowds. *ACM SIGGRAPH 2006 Papers* (2006).
4. Duives, D. C., Daamen, W. & Hoogendoorn, S. P. State-of-the-art crowd motion simulation models. *Transp. Res. Part C: Emerg. Technol.* 37, 193–209 (2013).
5. Lu, C. Analysis of Compressed Force in Crowds. *Journal of Transportation Systems Engineering and Information Technology* 7, 98–102 (2007).
6. Pelechano, N., Allbeck, J. M. & Badler, N. I. Controlling Individual Agents in High-density Crowd Simulation. *SIGGRAPH/Eurographics SCA* (2007).
7. Huerre, S., Lee, J., Lin, M. & O'Sullivan, C. Simulating Believable Crowd and Group Behaviors. *ACM SIGGRAPH ASIA 2010 Courses* (2010).
8. Peschl, I.A.S.Z. Passage Capacity of Door Openings in Panic Situations. *BAUN* (1971).
9. Smith, R. A. Volume Flow Rates of Densely Packed Crowds. *Engineering for Crowd Safety* (1993).
10. Kisko, T. M., Francis, R. L. & Nobel, C. R. EVACNET4 User's guide. *University of Florida* (1998).
11. Reynolds, C. W. Steering behaviors for autonomous characters. *Game Developers Conference* (1999).
12. Helbing, D., Farkas, I. & Vicsek, T. Simulating Dynamical Features of Escape Panic. *Nature* (2000).
13. Bandini, S., Manzoni, S., & Vizzari, G. Crowd Modeling and Simulation. *Innovations in Design & Decision Support Systems in Architecture and Urban Planning*, 105-120 (2006).
14. Brogan, D. C., Metoyer, R. A. & Hodgins, J. K. Dynamically Simulated Characters in Virtual Environments. *IEEE Comput. Graph. Appl.* (1998).
15. Pettré, J., Ondřej, J., Olivier, A.-H., Cretual, A. & Donikian, S. Experiment-based Modeling, Simulation and Validation of Interactions between Virtual Walkers. *ACM SIGGRAPH/Eurographics SCA* 189–198 (2009).
16. den Berg, J. van, Lin, M. & Manocha, D. Reciprocal Velocity Obstacles for real-time multi-agent navigation. *IEEE Intl. Conf. on Robotics and Automation* (2008).
17. Ondřej, J., Pettré, J., Olivier, A.-H. & Donikian, S. A Synthetic-vision Based Steering Approach for Crowd Simulation. *ACM Transactions on Graphics* (2010).
18. Secord, A. Weighted Voronoi Stippling. *International Symposium on Non-photorealistic Animation and Rendering* 37–43 (2002).
19. Chattaraj, U., Seyfried, A. & Chakroborty, P. Comparison of Pedestrian Fundamental Diagram across Cultures. *Adv. Complex Syst.* 12, 393–405 (2009).
20. Hoff, K. E., III, Keyser, J., Lin, M., Manocha, D. & Culver, T. Fast Computation of Generalized Voronoi Diagrams Using Graphics Hardware. *Conference on Computer Graphics and Interactive Techniques* (1999).
21. Shiraef, J. A. An exploratory study of high performance graphics application programming interfaces. *University of Tennessee at Chattanooga* (2016).
22. Pelupessy, F. I., Schaap, W. E. & van de Weygaert, R. Density estimators in particle hydrodynamics. *Astronomy & Astrophysics* 389-298 (2003).
23. Fruin, J. J. The causes and prevention of crowd disasters. *Engineering for Crowd Safety* 1, 10 (1993).
24. Special Events Contingency Planning. *FEMA* (2005).
25. Al-Habashna, A. & Wainer, G. Modeling pedestrian behavior with Cell-DEVS: theory and applications. *Simulation* (2015).
26. Zeigler B, Praehofer H and Kim T. Theory of modeling and simulation. *San Diego, CA: Academic Press* (2000).
27. Schaumann, D., Kalay, Y. E., Hong, S. W. & Simeone, D. Simulating human behavior in not-yet built environments by means of event-based narratives. *SimAUD* (2015).
28. Olamide, S. E. & Kaba, T. M. Formal Verification and Validation of DEVS Simulation Models. *AFRICON* 1-6 (2013).