High-density Real-time Virtual Crowds via Unilaterally Incompressible Fluid Simulation

Richard Ristic          Johan Berglund
KTH Royal Institute of Technology  |  Supervisor: Christopher Peters

Abstract
Human crowds experience limited individual freedom of motion and tend to exhibit aggregate behavior at high densities. This observation can be exploited in simulations by representing virtual crowds as both distinct agents and continua. We introduce a global planning algorithm suitable for navigating large virtual crowds of agents. These crowds are in turn modeled as \textit{unilaterally incompressible} fluids to effectively simulate large-scale behavior. The algorithm is ultimately tested on certain specific scenarios with the intention of detecting common emergent behaviors, to finally draw conclusions on its ability to simulate real-life situations.

1 Introduction
Crowds of humans show highly complex individual-driven behavior affected by numerous factors. There are a lot of challenges involved in recreating these behaviors virtually and a lot of effort is being put into developing realistic and effective models. The benefits of these simulations are clear, as there are several fields where they are applicable, including traffic engineering, urban planning, architectural design and emergency training.

There are circumstances, during evacuations for instance, where panic can induce catastrophic forms of human behavior, where the rush of people cause injuries and even casualties. Crowd simulations could be used to predict disasters and more importantly prevent these from ever ensuing, something that has been investigated by [Helbing et al. (2000)]

The scenarios described above often involve really large dense crowds. Simulating the behavior of such crowds can become really computationally demanding. The problem is that there is no simple way of making sure that all the agents avoid each other without computing for every single one of them. With up to thousands of agents, carrying out individual computations while maintaining interactive frame rates becomes unreasonable. Therefore, we need a way of simulating large dense crowds at lower computational costs.

By observing that large crowds of agents tend to show aggregate behavior at high densities, one can simulate these as unilaterally incompressible fluids, thus treating these as single aggregates. Such an approach has been proposed by [Narain et al. (2009)]. As much as it is an effective way of simulating a large crowd as a whole, it probably has its limitations in recreating certain natural behaviors, arising from individual decision-making for instance. On account of this, we will re-implement the proposed method
and evaluate it in terms of its ability to recreate realistic human behavior. Furthermore, we will touch upon possible improvements, with the purpose of enhancing realism.

2 Related Work

The simulation of crowds consists of several components that act towards reproducing the behavior of a real-life human crowd. Many techniques have been presented, tackling this problem from various angles. In general, it ultimately comes down to global planning and local behavior.

In order for the agents to be able to navigate through more complex environments, a global planning component is necessary to implement. There are numerous ways to achieve this, one of which is the introduction of guiding vector fields, as proposed by Jin et al. (2008). Another method, providing path planning based on topological information, is presented by Barnett et al. (2013) and one of the most basic solutions would be to represent the traversable paths with a roadmap, similar to the work of Bayazit et al. (2002). Eventually, for efficiency, one could consider implementing a leader-follower model (Li et al., 2001).

Local models for agent behavior show a wide range of variety and complexity. Some methods allow for sociological factors (Musse and Thalmann, 1997), while other methods account for psychological effects (Pelechano et al., 2005).

A different approach to crowd simulation is the use of continuum-based models, originating from the continuum theory for the flow of pedestrians, developed by Hughes (2003). Such an approach is described by (Treuille et al., 2006). The implementation by Narain et al. (2009), which exploits the aggregate behavior of such crowds, further expands upon this.

3 Model

The model used exploits the fact that large crowds tend to show aggregate behavior at high densities, with loss of individual freedom of movement. Consequently, the crowd is interpreted as a unilateral incompressible fluid, in other words a fluid that is infinitely expandable but compressible only to a certain point.

In practice, a discrete set of agents, with individual positions and velocities, is represented as a continuum with a continuous density field \( \rho \) and velocity field \( \tilde{v} \). Here, the tilde symbol is used to denote the agents’ own preferred velocities.

3.1 Global Planning

Initially, a way for the agents to navigate is required, in order for them to eventually reach their designated goals. It is a critical component when dealing with more complex environments, thus many implementations already exist. In this implementation, since it has to deal with large crowds consisting of many individual agents, several methods have to be combined in order to arrive at a solution which balances both realism and effectiveness.

The global planning is carried out using a basic roadmap-approach, similar to the one used by Bayazit et al. (2002), where a graph structure represents the traversable areas. Path planning simply consists of a node-to-node breadth-first search, where the shortest path on the graph represents the actual physical path then followed by the agents.

![Illustration of a simple roadmap.](image)
When implementing a leader-follower approach, like the one presented here, one has to consider the possibility that agents can get stuck behind obstacles when blindly following the leader. This issue is addressed by letting the agents look for the leader at a fixed frequency and keep track of the position where it was last seen, as proposed by Li et al. (2001). That way, if an agent loses track of the leader, it will head towards the last seen position and look for the leader again, as illustrated in figure (3). In fact, a similar approach is used by the leader while traversing its path, as it regularly looks for subsequent nodes in order to detect possible shortcuts.

Finally, to further improve realism, the leader is essentially extended into an abstract line. When spawned, each agent is assigned a point on the leader to follow based on the projection of the relative vector between these two, as seen in figure (4). As a result, the agents tend to spread out and do not form narrow lines, as would be the case if the leader retained the shape of a single point.

### 3.2 Staggered Grid

In most models in fluid dynamics, a staggered grid is used to discretize the problem and it is also used here. In the simulation, a quadratic grid with a variable resolution is implemented, where density values are stored in the cell centers and velocity components at the cell edges, as is shown in figure (5).

#### 3.2.1 Particle-in-cell Method

When projecting velocity and density values onto the grid, it is important that the representation is as continuous as possible. Practically, the agents need to contribute not only to the cells in which they find themselves, but also to adjacent cells. This is achieved by using the particle-in-cell method of fluid simulation described by Evans and Harlow (1957).

Basically, an abstract area is considered, centered on each agent and of the same size as one cell. Depending on the position of the agent in its current cell, parts of the area will coincide with adjacent cells which in turn will decide the fraction of the total density- and velocity values that are distributed to those cells. See figure (6) for further clarification.
3.3 The Unilateral Incompressibility Constraint

Fundamentally, the unilateral incompressibility constraint (UIC), introduced by Narain et al. (2009), implies that the density in each cell can never exceed the maximum allowed density. By making the assumption that the agents cannot come closer than a specified minimum distance $d$, the maximum density is given by:

$$\rho_{\text{max}} = \frac{2\alpha}{\sqrt{3}d^2}$$

(1)

where $\alpha \leq 1$ is a constant used to account for the fact that crowds are seldom perfectly packed. The UIC, which must hold at all times, then becomes:

$$\rho \leq \rho_{\text{max}} = \frac{2\alpha}{\sqrt{3}d^2}$$

(2)

3.3.1 The continuity equation

By taking into account that the number of agents is always conserved, the relationship between the density and the velocity can be given by the continuity equation below:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{v}) = 0$$

(3)

where the corrected velocity $\mathbf{v}$ has to be determined, preferably as close to $\mathbf{\tilde{v}}$ as possible, such that $\rho$ now satisfies the UIC. The task of finding $\mathbf{v}$ is equivalent to solving the optimization problem (4).

$$\max_{\mathbf{v}} \int \rho \mathbf{v} \cdot \mathbf{\tilde{v}} \quad \text{subject to } \|\mathbf{v}\| \leq v_{\text{max}}$$

Using variational calculus, which is left out here, one arrives at the solution which takes on the following form:

$$\mathbf{v} = v_{\text{max}} \frac{\mathbf{\tilde{v}} - \nabla p}{\|\mathbf{\tilde{v}} - \nabla p\|}$$

(5)

where $p$ denotes a scalar pressure which obeys under the following set of conditions:

$$\rho < \rho_{\text{max}} \Rightarrow p = 0$$

(6)

$$\rho = \rho_{\text{max}} \Rightarrow p > 0 \Rightarrow \nabla \cdot \mathbf{v} = 0$$

(7)

The physical implication is that whenever the flow at a certain point reaches the maximum density allowed, a pressure is applied outwards from that specific point to counteract the compression. By subtracting the pressure gradient from the velocities, the agents now tend to move in the opposite direction to that of increasing pressure (Narain et al., 2009).

3.4 Numerical Method

When solving equation (5), it is divided in two parts and the two tasks are carried out separately.

$$\mathbf{\tilde{v}} = \mathbf{\tilde{v}} - \nabla p$$

(8)

$$\mathbf{v} = v_{\text{max}} \mathbf{\tilde{v}}$$

(9)

It is in equation (8) that the use of a numerical method is required, as the actual values of the pressure $p$ need to be determined. Finally, it is equation (9) that yields the corrected velocity $\mathbf{v}$ which is later to be applied to every agent.

To determine the pressure values $p$, equation (8) is substituted into the continuity equation (3), completely disregarding the normalization done in equation (5) for now (Narain et al., 2009).

$$\nabla \cdot \left[ \rho (\mathbf{\tilde{v}} - \nabla p) \right] = 0 \Rightarrow \frac{\partial \rho}{\partial t} = -\nabla \cdot (\rho \mathbf{v}) + \nabla \cdot (\rho \nabla p)$$

(10)

Now, the finite difference method is used to approximate the time derivative of $\rho$.

$$\frac{\rho_{n+1} - \rho_n}{\Delta t} = -\nabla \cdot (\rho_n \mathbf{\tilde{v}}_n) + \nabla \cdot (\rho_n \nabla p_n) \Rightarrow \rho_{n+1} = \rho_n - \nabla \cdot (\rho_n \mathbf{\tilde{v}}_n) \Delta t + \nabla \cdot (\rho_n \nabla p_n) \Delta t$$

(11)
where \( n \) denotes the current time step. The UIC \((2)\) is now applied to \( \rho_{n+1} \), which finally yields:

\[
\rho_{\max} - \rho_{n+1} \geq 0 \Rightarrow \\
\rho_{\max} - \rho_n + \nabla \cdot (\rho_n \tilde{v}_n) \Delta t - \nabla \cdot (\rho_n \nabla p_n) \Delta t \geq 0
\]  

\( 12 \)

3.4.1 Linear Complementarity Problem

A general linear complementarity problem (LCP) can be written in the following form:

\[
Ax + b \geq 0, \quad x \geq 0, \quad (Ax + b) \cdot x = 0
\]  

\( 13 \)

By identifying the different terms, it is concluded that the inequality \( 12 \) is equivalent to the following LCP:

\[
x_i = p_n
\]  

\( 14 \)

\[
(Ax)_i = -\nabla \cdot (\rho_n \nabla x) \Delta t
\]  

\( 15 \)

\[
b_i = \rho_{\max} - \rho_n + \nabla \cdot (\rho_n \tilde{v}_n) \Delta t
\]  

\( 16 \)

where \( i \) denotes the components corresponding to the different cells. According to Narain et al. (2009), the matrix \( A \) has some specific properties that are taken into account when choosing the solver for the LCP. It is a sparse, symmetric and positive semidefinite matrix which makes the MPRGP algorithm (Modified Proportioning with Reduced Gradient Projections), proposed by Dostál and Schöberl (2005), appropriate to use.

When the algorithm, which is iterative, is initialized it is supplied with an initial guess \( x_0 = 0 \) and subsequently with pressures from the previous iterations, that is \( x_{n+1} = x_n \).

3.5 Obstacles and the Modified UIC

A cell that is covered by an obstacle will have a much smaller available area for the agents to traverse (Narain et al., 2009). This is accounted for by simply modifying the UIC \((2)\) as follows:

\[
\rho \leq f \rho_{\max}
\]  

\( 17 \)

where \( f \) is the fraction of the available area in the specific cell. Applying the modified UIC to the LCP only yields a change in \( 16 \).

3.6 Correcting Agent Velocities

The corrected velocity \( v \), which is now stored in the staggered grid, has to be distributed back to the individual agents. This is done by using the particle-in-cell method from section \( 3.2.1 \) again, but in reverse. The final velocity \( v_f \) of the agent is then given by the interpolation below:

\[
v_f = \tilde{v} + \frac{\rho}{\rho_{\max}}(v - \tilde{v}) + v_c
\]  

\( 19 \)

where \( v_c \) is the velocity contribution arising from the pairwise collision handling described in section \( 3.7 \). Closer inspection shows that when the density is really low in comparison to the maximum density, the agent can move freely at its own preferred velocity. On the other hand, if the density is close to its maximum value, the agent’s freedom of motion is completely lost (Narain et al., 2009).

\[
\rho \ll \rho_{\max} \Rightarrow v_f \approx \tilde{v} + v_c
\]  

\( 20 \)

\[
\rho = \rho_{\max} \Rightarrow v_f = v + v_c
\]  

\( 21 \)

Finally, the new agent position is simply updated according to:

\[
r_{n+1} = r_n + v_f \Delta t
\]  

\( 22 \)

where the \( \Delta t \) used is the exact same time step as the one in the LCP.

3.7 Pairwise Collision Handling

The accuracy of the corrected velocity obtained, when solving the LCP, is limited to the resolution of the grid and can not resolve collisions on a local level by itself. A pairwise collision handling of some sort is required to account for that and a simple way of doing this is to just move the agents apart upon collision.

For two colliding agents \( k \) and \( l \), with positions \( r_k \) and \( r_l \) respectively, the collision avoidance velocity for agent \( k \) is determined by:

\[
v_c = v_{\max} \frac{r_k - r_l}{\lVert r_k - r_l \rVert}
\]  

\( 23 \)
4 Results and Discussion

The algorithm has been tested on certain specific scenarios with the intention of detecting common emergent behaviors. Furthermore, different performance measurements have been carried out in order to discover possible bottlenecks and areas in need of further optimization. The hardware used consistently throughout the evaluation consists of a 3.4 GHz Intel Core i7-2600K CPU, an Nvidia GeForce GTX 580 GPU and 16 GB of RAM.

4.1 Performance

The algorithm is comprised of two major components, the enforcement of the UIC working in coalition with the pairwise collision resolution. Performance-wise, these are both heavily dependent on the resolution of the grid. However, the former benefits from a lower resolution while a higher resolution works in favor of the latter. Figure [9] shows how the computation time for the complete algorithm scales with the total number of cells in the grid. Initially, the computational cost of the pairwise collision handling dominates completely, only to subside as the number of cells are increased. At a certain point, the computation time is at its minimum, and henceforth the gain in performance for the pairwise collision handling pales in comparison to the performance drop caused by the UIC enforcement.
For higher grid resolutions, the MPRGP-algorithm is what can be considered the bottleneck of our algorithm. Figure (10) presents the convergence rate of this algorithm. It is evident that this is a significant contributor to the performance drop when increasing the number of cells, as it scales quadratically with the size of the grid.

An additional factor that has a great impact on the performance is the sole rendering of agents. Even with simple spheres, the rendering itself limits the number of agents that can be simulated to a few thousand. Simulation of really large crowds in real-time is simply not possible with such an ineffective rendering. Studying figure (11), one can conclude that crowds exceeding 6000 agents in size result in severe performance drops, definitely comparable to those originating from the simulation algorithm.

Increasing the density, amplifying the effect of the continuum velocity, gives rise to a more natural behavior. The scenario depicted in figure (7) illustrates this well. When two groups of agents pass each other, moving at opposite directions, one can notice the formation of streamlines. This is a common observation when using a fluid dynamics approach. In real-life situations, people tend to follow those in front, who have already paved the way for them, which is basically what happens in the simulation as well.

An intuitive way of finally maximizing the agent density is to spawn them in a circle and guide them towards their diametrically opposite positions, as seen in figure (8). In this way, the crowd will converge at the circle centre, leaving the innermost agents exposed to immense inter-agent forces. In order to prevent densities from exceeding the limit defined by the UIC (2), outwards pressures arise to counteract the inwards motion. This results in a vortex being formed, causing the agents to revolve around each other until the crowd is finally dissolved. Although this is not a scenario that would arise in real-life situations, handing a group of people the instructions to perform this maneuver, could result in a similar outcome.

### 4.3 Prospective Improvements

At this stage, the simulation is not capable of modeling really large crowds in real-time, as there are several apparent performance issues in the implementation. As stated in section (4.1), one major limiting factor is the enforcement of the UIC, and more specifically the MPRGP-algorithm. This, of course, applies mainly to higher grid resolutions, which is what one should strive for in order to obtain more detailed solutions. In fact, there is an efficient modification that can be applied to the MPRGP-algorithm in order to drastically improve its convergence rate and thus allow for much higher grid resolutions. That is the introduction of the Modified Incomplete Cholesky preconditioner, MIC(0), described by Bridson and Müller-Fischer (2007).

Due to the fact that this method only models the large-scale behavior of a human crowd, the individual decision-making is totally ignored, something
that is otherwise such a fundamental part of how groups of humans interact. One significant aspect in this interaction is the ability of humans to anticipate future collisions and plan ahead, an ability that the virtual crowds in this model lack. As it is now, two groups of agents will not notice each other until they are about to collide, something that could be improved by further modifying the global planning step. Since every group is guided by an abstract leader (See section 3.1), one could make sure so that these leaders act towards separating different groups on a collision course, at an early stage as well. Consequently, this would result in the formation of separate lanes for different directions, which would in turn significantly improve the overall realism of the simulation.

As for the evaluation of the method, improvements could be made to further ensure its objectivity. A way of scientifically quantifying the simulator’s ability to reproduce behaviors observed in recorded real-world data, would be to make use of the metric proposed by Guy et al. (2012), which provides a statistical similarity measure for simulations of this kind by the calculation of entropy. In addition, the scenarios and metrics included in the SteerBench suite, developed by Singh et al. (2009), could be included in the evaluation as an alternative.

5 Conclusion

Simulating large dense crowds tends to be computationally demanding, an issue that has been addressed by introducing a continuum-based approach. This method establishes a solid foundation for crowd simulation, yet further improvements are required to ensure realistic recreation of natural human behavior.

References


