Multi-Agent Simulation Using Continuum Crowds and the ClearPath Method

By Miri Kim and Luis Ángel Larios Cárdenas

1. Objectives

General

Evaluate the performance of a crowd of artificial agents in a virtual environment using two approaches for modeling steering behavior: ***Continuum Crowds*** and the ***ClearPath*** methodology.

Specific

- Model a three-dimensional virtual world, together with dynamic entities, represented by disc-based artificial agents navigating the terrain while avoiding one another and various static obstacles that may be encountered.

- Perform a space-time assessment of the efficiency of both ***Continuum Crowds*** and ***ClearPath*** approaches, accounting for the time at which agents reach a given goal spot (such as a safe place in an evacuation simulation), and for the number of collisions throughout the execution.

- Analyze and judge the emergence of crowd behavior as well as which method produces the most natural resemblance to human steering in a series of test cases.

2. Related Work

The renovated work of Reynolds in [REY99] opened the possibility for computer graphics scientists to develop applications routed to model emergent collective behaviors of artificial crowds. Since then, interest in such simulations has called for attention not only in the animation industry, but also within other disciplines, like building modeling and sociology, which found these techniques useful to measure and observe human behavior at a macroscopic level, with the ability to interact with an application that eases the analysis of possible population responses to global stimuli.

Crowd simulation requires not only deploying independent agents and their personal features, but also implementing efficient procedures to simulate the natural way in which people translate from their current positions to pre-computed (or known) goals [SHA05]. This field is broadly known as path-planning and implies that agents must reach their goal spot watching their step at all time in order to avoid collisions with static (like walls) and dynamic (like other persons) obstacles. On this respect, [SUD07] makes a succinct classification of these methods:

Discrete Methods

These procedures sample either the environment or the agents and are classified as follows:

a) *Agent-based methods*, which implement a series of local rules for each agent, fundamentally from the guidelines stated in [REY99], facilitating the emergence of flocking behavior by resorting to a simple collision avoidance rule-based method. When these basic concepts are integrated to a local path planner, the combination produces an improved technique as in [GUY09], who uses a discretization method to efficiently compute the motion of each person by reformulating the conditions for collision-free navigation as an optimization problem that extends the notion of velocity obstacles. This last development, known as ***ClearPath***, is one of two approaches recreated in the present work to synthesize real-time crowd simulation.
b) *Cellular automata methods*, where the motion of each agent derives from the idea of Conway's cellular automaton and depends on the occupancy of neighboring cells without worrying about any dynamics in the process.
c) *Particle dynamics methods*, that account for the forces that take place in modeling the flow of motion. In their work, [SUD07] used a novel path planning data structure called "Adaptive Elastic Roadmaps" or AERO, which is a connectivity graph of

milestones and links, used to compute the collision-free guiding path for every agent, who previously has determined their goal spot in a 2D representation of a virtual world. AERO turns out to be an enhanced Corridor Map Method as of [GER07], which was developed having in mind the construction of a real-time planner for robotic agents.

Continuous Methods

These procedures consider a set of agents as fluid, and therefore, their motion resembles gas, fluid, or granular flow depending on the density of the crowd at the different locations on a discretized environment. In [TRE06], the second method modeled in the present project, the authors exclude the agent-based dynamics and develop a global planner that yields dynamic potential and velocity fields over the domain, which guide all individual motion simultaneously in a group fashion, integrating "discomfort zones" and population splatted into density levels as kernels of the collision-avoidance process. On the same line, [NAR09] attempts to produce crowd aggregation by resorting to a combination of both global and local planning, trying to overcome the weaknesses in [TRE06], and keeping interactive times while maintaining thousands of agents in virtual scenarios. Accordingly, [NAR09] introduces a new constraint called "unilateral incompressibility," which aids in modeling the large-scale individual and collective interactions when populations are very dense.

3. Challenges

Some situations, ranging from the utilized methodology to the rational use of computational resources, affected the appropriate development of the current system. The following critical points give a brief idea about the challenges that developers faced and that hindered at a certain level the work as it was planned.

Continuum Crowds

a) [TRE06]'s paper exposes a very intuitive methodology for implementing C***ontinuum Crowds***; however, most of the details are not quite defined when it comes to a simulation construction. This issue caused the developers to make assumptions that did not work appropriately at a first time (like where to assign the personal discomfort field so that it would not affect the avatar itself but others around it), and attempting to find the right approach consumed valuable timetable slots that were reserved for other activities, preventing, thus, to recreate additional functionalities that may have been very useful (such as an automatic parameter-tuning).
b) The most expensive task of ***Continuum Crowds*** is computing the potential field, which implies to manage wisely several dynamic data structures to avoid slowing the general process. This required being very careful, when allocating and dis-allocating memory, and the use of heaps and binning techniques to shorten the computation time and space for picking quickly the elements to work with during a certain part of the general algorithm.
c) Even the chariness for avoiding over space-time consumption is not enough when the computational resources are not suited for lots of calculations for a real-time application. This issue made developers reduce their expectation on observed results if they wanted "to observe" experimental outputs. Including hundreds of agents was very prohibitive when the system was tested on a common PC with conventional memory and processor power available, so it was necessary to find representative examples of the behavior intended for particular scenarios.
d) Parameter-tuning was largely a time consuming task, especially before the limitations of memory and processor speed. The process became hard to tackle since there is a hidden relationship among parameters that permits the ***Fast Marching Method*** to behave appropriately. In many cases, arbitrary choices for weights (such as density exponent, discomfort, time, and length of path) led to early stops in the algorithm since it started to diverge and output complex potential values. Mainly, it was observed that the inclusion of discomfort zones with a very high cost associated (such as modeling walls), caused the system to teeter on the brick of failure when agents concentrated in corridors, which increased the density field and

closed paths for others that needed to go through them to reach their goal (in evacuation simulation). Usually, after twenty minutes of simulation, just before concluding it, the system reported an error, and the whole task required to tune the parameters again to begin simulating one more time.

ClearPath

a) Understand related work which this method is based on, such as *Velocity Obstacles* and *Reciprocal Velocity Obstacles* (RVO). Mainly, they are available online, but using them on a real implementation required grasping the fundamentals for they provided the desired velocity on which ***ClearPath*** had an effect.
b) ***ClearPath*** comes from a very abstract formulation that includes geometry and other vector concepts that are not easily digested. Ultimately, knowing them helped picture the general methodology behind this algorithm.
c) Adapting RVO to an existing template code for animation took a long time and turned out to be quite complicated due to the duplications of symbols and linking issues with additional libraries.
d) Vague sections of the ***ClearPath*** algorithm elicited developers' own assumptions, which most of the time resulted inadequate and because of that a fully implementation of the method was not possible in the time allowed.

4. METHODOLOGY

The system consists of a finite number of artificial agents immerse in a virtual world that characterizes a series of test-case scenarios:
a) Actions and computations take place in a plane parallel to the XY one.
b) The environment consists in an enclosed, non-toroid grid, which number of cells can vary according to the scenario to be tested.
c) The user can specify the simulation time, and, when desired, the time at which agents must stop their current activity and head towards a common goal spot, simulating this way an evacuation.

Agents, on the other hand, are represented as mass-points, drawn as circles, with colors and other direction indicatives that aid tracking the development of the simulation throughout its several stages. The avatars have the next features:
a) They have only two possible states:
    - **Navigating**. If the agent has not reached a goal and is moving around the terrain.
    - **At goal**. If the agent has reached an established goal location (one or more cells).
b) Agents are generated randomly in several locations of their world and are assigned a goal or group of goal cells initially. Depending on the scenario and the test case, they are able to choose dynamically a next goal or stay indefinitely at the place they reached.
    - **Continuum crowds**. Agents are grouped in up to 4 teams; each of them, even when avatars can be generated in different locations with respect to their teammates, shares the same goal spots and, therefore, the same potential field. Agents are thought "to have reached" their goal if and only if all of the members of that group lie inside the target location.
    - **ClearPath**. Agents are fully independent of others and have the ability to pick their own goal choice, as well as avoiding static and dynamic obstacles purely from computing individual RVO cones that delimit a range of constrained motion.
c) When moving from one place to another, avatars resort to two distinct approaches for performing path planning and collision avoidance:
    - **ClearPath**. As stated in [GUY09], heterogeneous agents define goals and start moving in the direction computed as a result of satisfying velocity-obstacles constraints shaped as cones. The new velocity is, then, demonstrated to be the optimal one lying on the edge of those cones when a collision is foreseen in a short term. This method allows motion in whatever direction as long as it is free, showing thus the construction of smooth paths that are not restricted to a discretized environment.
    - **Continuum Crowds**. The agent computes an optimal global path that

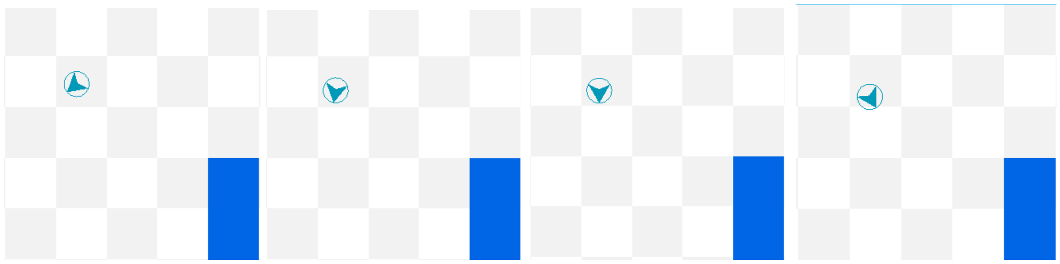

Figure 1. Smoothing velocity field with current orientation.

embeds the collision avoidance behavior at the same time. Although it is possible (and was experimented) to model individual goals and personalities, groups of avatars need only to compute one potential field composed of individual cost units in accordance to the grid discretization in order to ameliorate the expensive task derived from the *fast marching method* that approximates a potential function on the plane. Once this field is available, agents can draw gradients, embrace one out of 4 upwind directions with minimum cost and potential, and consider a desired velocity that is opposite to the respective chosen gradient. The desired velocity is finally *interpolated* with the current agent's orientation, which has a maximum rotation angle, and a final velocity vector is computed as a *smooth* version of the next time-step direction and the current state, creating this way a better impression of realism.

5. Results

Results are described according to the method they belong to, and later as a comparison between both ***Continuum Crowds*** and ***ClearPath***. All systems' capabilities, including the simulator, were programmed in C++ on a common Windows-based PC with 1GB RAM and 1.73 GHz of processor power, using the Microsoft Visual Studio 2008 platform, the OpenGL library, the Eigen matrix calculus open library, and the RVO library (for ***ClearPath***).

Continuum Crowds

Five different scenarios were evaluated to determine the behavior of agents using the ***Continuum Crowds*** algorithm. In a first case, a 10x10 grid contains two goal cells blocked by a three-cell-length "wall", and one agent waiting in a distant location from the spot. The introduction of this simple test case was to model the smoothing of velocity and orientation. Originally, the algorithm considered only four directions to take: north, south, east and west, which in some degree constrained motion to sharp changes of heading - an undesirable effect. One approach for avoiding sudden changes in velocity was to limit the amount of turning for the avatar, so that when a new direction was completely opposite (e.g. 180º) to the current one, the agent could only turn 15º as maximum with an interpolated velocity that did not have the intended length but a proportional scale to the required amount to rotate, giving, thus, a size that was closer to its original value when the turn did not differ largely from the current avatar's orientation. Figure 1 shows this velocity smoothing/interpolation.

The second test case allowed tuning the weights for computing the unit cost field: discomfort, length, and time of the path, as well as the density effect over the speed field for each cell, given by the density exponent, the maximum and minimum densities, and the proportion of density that the discomfort represents. Two agents inhabited an open environment (no obstacles) that encompassed four goal locations, where three of them were available to be chosen dynamically with the possibility of overlapping, which means that agents could share the same goal at the same time, for the sake of observing if they were able to avoid themselves smoothly once they faced one another on a common lane. As a result, the minimum number of collisions or invasions to a "personal radius" was achieved by having a high weight for discomfort field in both the unit cost computation and in density equivalent proportion (personal future projection). After 25 simulated seconds, both agents' goals were suppressed as if they were

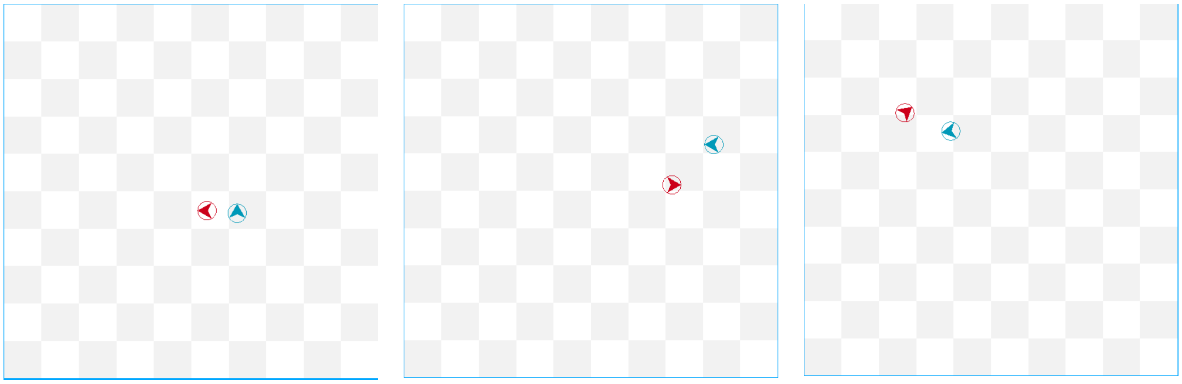

Figure 2. Two agents avoiding each other after setting parameters.

ordered to head immediately to a common location (called *safe goal* thereafter), introducing this way the developers' model of evacuation with a simple example. Snapshots for the experiment are displayed in figure 2.

In a third scenario, 40 agents, teamed up in two groups, were located in two opposite places of a grid with 16 cells per side. Their goals were exactly where the other team was generated, so a front encounter was intentionally modeled to evaluate the behavior of avatars when dynamic obstacles are heading towards them. In consequence, there were observed lane formations as reported by [TRE06], product of the high ratio between discomfort and the cost associated with length and time spent for path choosing. Moreover, in order to keep the simulation alive, agents were given the ability to move to the remaining goal-cell set shortly after all members of the same team had reached the common goal. In this occasion no "evacuation" was programmed (time for it to happen was set beyond maximum simulation time), but agents could share the same goal spots, provided only two sets of them existed on the whole terrain. Results on this experiment are shown in figure 3.

Another interesting phenomenon described in [TRE06] is the vortices formation when 4 streams of agents located at the corners of the terrain attempt to reach their opposite corner. This experiment was also included in the present reconstruction of ***Continuum Crowds***, where it was observed that having a discomfort field weight very high with respect to other parameters affected the generation of this sort of behavior; therefore, it was necessary to reduce the former weight, so that the groups of avatars remained compact, resembling fluid motion from one spot to another. Although the vortex was not remarkably defined due to the limitation in the number of agents, it was noticeable a gyration of the 120-unit crowd in order to make it to the desired location. At this point, no evacuation was emulated in the 30x30 terrain, and agents were not allowed to choose new goals once all members of their

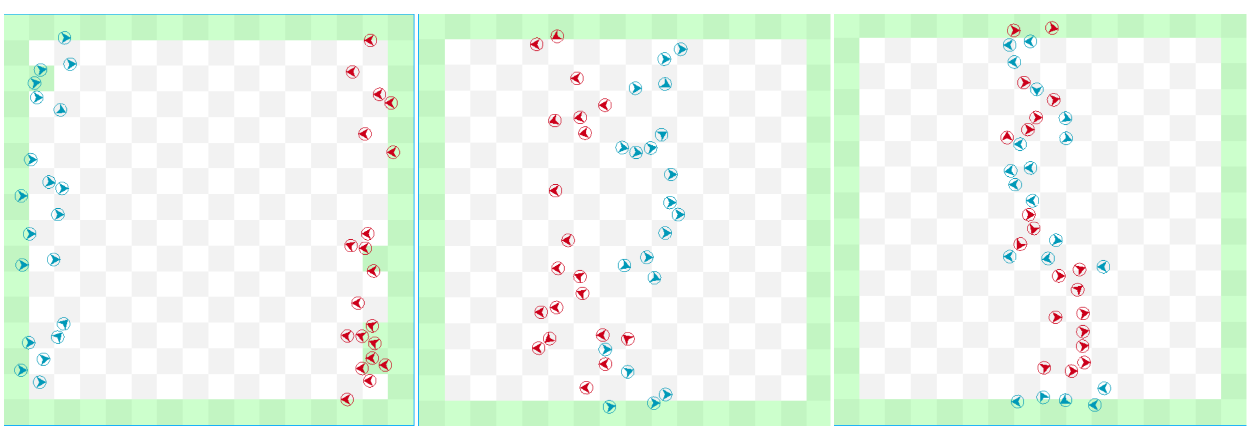

Figure 3. Experimenting with agents facing one another on their way to their goal spots.

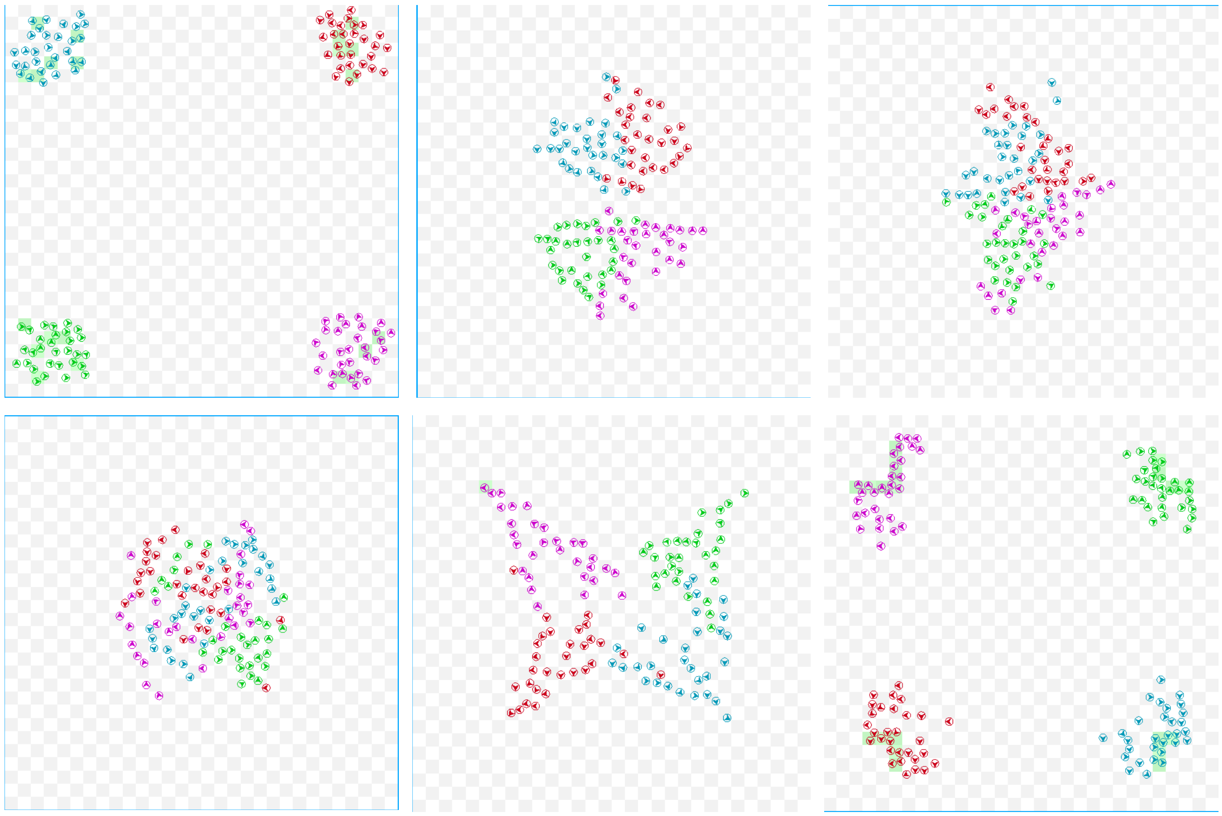

Figure 4. Reconstructing the vortex formation when 4 streams of agents face in the center of the grid.

groups reached the assigned spot. This experiment took about twenty minutes to complete, considering a maximum speed of 5 cells per second (which is the same value for all experiment here described). Some key phases of the simulation are exhibited in figure 4.

Finally, with the purpose of simulating an evacuation, after the preparation of previous experiments, the developers set up a 20x20 grid with 6 sets of goal locations (plus the safe goal) and 4 groups of 15 agents each with the ability to pick goals dynamically that are not currently assigned to other group (no overlapping, in contrast to scenario 2 and 3). As described before, it was not possible to use directly the parameters from other scenarios because the existence of "walls" and path obstructions destabilized the ***Fast Marching Method*** computation and soon complex values started being generated as indication of divergence. Instead, several trial-and-error attempts were needed to acquire a collection of parameters that, for this circumstance, had much more to do to the density exponent and the discomfort weight than to any other parameter. Of course, some drawbacks were observed, such as agents being pushed inside walls, and choosing to go through them after not free viable path was found. The experiment took up to 35 minutes to have all avatars at the safe goal, taking into account that the simulated time for rising the "emergency" (overwriting everybody's goals) happened at t = 20.00s. Some screen captures of the experiment are available at figure 5.

In all experiments, goal cells were defined with a maximum capacity, so that they did not become overpopulated and create zones of high density. To tackle this issue, when a certain amount of agents stood on a goal cell, a nominal discomfort was added, preventing others from going there and encouraging picking available spots even if they were farther.

ClearPath

Because of overtime spent on debugging RVO as core for ***ClearPath***, adequate results could not be generated as in the case of previous method. Some intended simple test cases

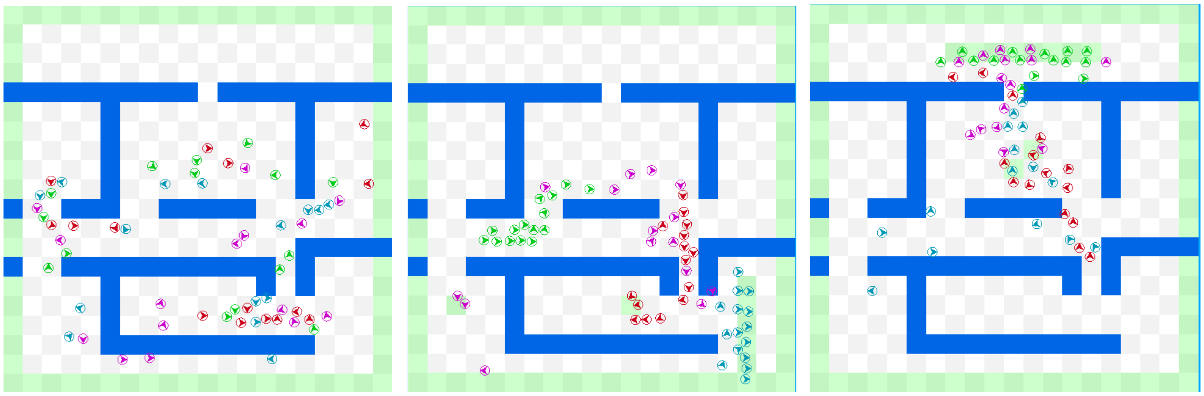

Figure 5. Evolution of simulating an evacuation.

worked well in the presence of one goal and a couple of agents; however, as soon as geometric obstacles were added to the environment, the avatars started oscillating or, in other circumstances, remained stationary with no apparent reason. In a first instance, the problem appeared to be due to narrow passages and exits (like in figure 5), but despite of removing those restrictions, only a pair of agents were able to reach a circle-based goal while others took an opposite direction. Moreover, the main objective of ***ClearPath***, avoiding collisions, failed in most cases, since agents could not perceive the presence of their partners and ended up overlapping once they were at the same goal spot.

Continuum Crowds versus ClearPath

From the perspective of the general description of ***ClearPath*** and ***Continuum Crowds***, both methods were made for different simulation configurations. On one hand, with ***Continuum Crowds*** it was possible to observe that agents behaved as fluid because they shared the same potential field that guided them towards their common goal. Having this advantage of homogenizing the avatars allowed reducing the computation costs of the expensive ***Fast Marching Method***; however, the individuals recreated more a flock-like behavior rather than a human one, which did not resemble appropriately the natural motion of people in that respect (putting aside the vertex and lines formations reported on the present project and [TRE06] experiments).

On the other hand, ***ClearPath***, theoretically, defines an improved approach that does not steal individuality from agents. Even obstacles, have their own definition, and are the key in constructing urban enclosed scenarios where agents must respond accordingly in order to reach their goal and avoid one another. This was a remarkable drawback in ***Continuum Crowds*** because the procedure was not meant to explicitly rely on agents' ability to avoid collisions. Despite the lack of a real model to compare with, the ***ClearPath***'s formulation suffices to prove that in an arranged simulation, the latter approach would outnumber the ***Continuum Crowds***' performance, which reported thousands of collisions and other artifacts derived from the sparse discretization of the world.

6. Limitations

Some general limitations obtained from both ***Continuum Crowds*** and ***ClearPath*** are as follows:

a) Simulations were geared to model simple behaviors of a population of artificial agents with individual or group goals while displacing on a plane embedded in a 3D world.
b) The system did not model deliberative formation of groups of agents, but it showed the emergence of collective behavior, such as jamming and flowing (lanes, vortices).
c) There are some restrictions inherited by the underlying implemented methods:
    - The ***Continuum Crowds*** method sacrifices individuality in order to produce simulations at interactive frame rates, where the level of discretization of the plane determines the maximum population that the environment can support. In particular,

the present implementation of Continuum Crowds is limited as described:

- In order to allow up to one hundred agents at the same time in the environment, the size of the grid had to be also in proportion to that fact. Thus, incrementing the number of cells yielded a lower number of steps per unit of time, and simulations took longer to show an acceptable result or to indicate failure because of the parameterization of cost weights. In consequence, most of test cases were at low scale, with around 100 avatars in a medium-sized environment.
- The simplification of the terrain assumes a constant slope along the plane, so one part of computing the speed field in the general algorithm was truncated to the maximum speed, and later interpolated with flow speed for areas with intermediate density levels.
- Avatars do not have any intelligence but following the direction implied by the velocity field after it has been *smoothed* with their current *orientation*. Additionally, agents choose randomly their next goal only once all of their teammates have reached the same attained location.
- For indoor environments, modeling walls is restricted to define cells with high discomfort values. However, arbitrarily large discomfort costs associated to them led to incongruences in the potential field and the ***Fast Marching Method*** failed to approximate correctly the global function. Therefore, after finding out working parameters and costs, simulations ran smoothly but having sometimes agents that crossed “walls”, which was a consequence of picking the least expensive route to a goal once it found itself obstructed in all possible directions by high density concentrations.
- ***Continuum Crowds*** is highly dependent in the degree of discretization of the environment, or, in other words, the size of the cells that store information about costs. Agents are blind to detect obstacles and rely directly upon data they can extract from their current cell and the costs heading in four upwind directions. Sometimes, avatars unavoidably ended up at the same cell (like a goal) and collisions appeared immediately; nonetheless they were disguised with “symmetric minimum distance enforcement,” which in most cases introduced artifacts easy to detect because of agents sliding or being pushed inside a wall (that was later used for their own benefit of avoiding a corridor, for instance).
- System parameterization differs accordingly to the kind of environment to simulate. In cases of open areas (no “walls”), agents could avoid themselves smoothly, but, on the other hand, the same parameters in very crowded experiments with obstacles included usually not only did not smoothen motion but also made the ***Fast Marching Method*** fail. Therefore, there was a tradeoff between the intended behavior and the potential computation integrity, which was easily compared in the evacuation simulation when having just avatars attempting to reach opposite goal sides with no obstacles in between.

• Despite developers could not get a working solution from ***ClearPath***, it is still worth to mention its authors’ comments on the method’s limitations:
  - It is possible that there exists a collision-free path for agents, but the approach may not be able to find it. This requires implementing alternative methods that make ***ClearPath*** less restrictive while retaining the same objective.
  - Due to the conservative constraints, it is highly dependent on the time step used to compute

the velocity obstacle cone, which sometimes does not guarantee to find a collision free path. The restriction may be relaxed at expenses of having agents colliding or at the cost of shorter time steps that undoubtedly can slow down the general process of the algorithm.

7. Future Work

Most of problems and limitations already explained represent an open door to further improvements and completion of the comparison between the performance of both ***Continuum Crowds*** and ***ClearPath***.

In particular, for ***Continuum Crowds***, the daunting hand-tuning parameter task should be substituted by an automatic process, which could possibly resort to genetic algorithms to converge to the optimal weight set that guarantees the least possible amount of collisions throughout the simulation. Also, there already exist improved versions of the ***Fast Marching Algorithm*** that behave more robustly before divergence. A combination of both a novel implementation of the level set method and the artificial intelligence technique might transform this approach into a friendlier API to generate optimized scenarios that match the necessities of the user without consuming too much time. Additionally, the realism of the output animation could be enhanced if a human-avatar replaced the current disc-based agent, although this requirement would be applicable if the computational resources available were enough to support rendering and the rest of the process at real-time rates.

In the special case of the ***ClearPath*** implementation, a full development of the technique would aid to give continuity to the original objective of the present project, and, only after that, a more defined vision of prospective improvements could become clear as a consequence of the identification of specific open problems and proposed solutions.

8. References

[GER07] Roland Geraerts, Mark H. Overmars. *The Corridor Map Method: Real-Time High-Quality Path Planning*. ICRA (2007).

[GUY09] Stephen. J. Guy, Jatin Chhugani, Changkyu Kim, Nadathur Satish, Ming Lin, Dinesh Manocha, Pradeep Dubey. *ClearPath: Highly Parallel Collision Avoidance for Multi-Agent Simulation*. ACM SIGGRAPH/EG Symposium on Computer Animation (2009).

[NAR09] Raul Narain, Abhinav Golas, Sean Curtis, Ming C. Lin. *Aggregate Dynamics for Dense Crowd Simulation*. ACM Trans. Graph., 28, 5 (2009).

[REY99] Craig Reynolds. *Steering Behaviors for Autonomous Characters*. Proc. Games Developers Conference (1999).

[SHA05] Wei Shao, Demetri Terzopoulos. *Autonomous Pedestrians*. Graph. Models 69, 5-6 (2005).

[SUD07] Avneesh Sud, Russell Gayle, Erik Andersen, Stephen Guy, Ming Lin, Dinesh Manocha. *Real-Time Navigation of Independent Agents Using Adaptive Roadmaps*. Proc. ACM Symp. Virtual Reality Software and Technology (2007).

[TRE06] Adrien Treuille, Seth Cooper, Zoran Popovic. *Continuum Crowds*. ACM Trans. Graph., 25, 3 (2006).