DEGREE PROJECT IN COMPUTER SCIENCE AND ENGINEERING,
SECOND CYCLE, 30 CREDITS
STOCKHOLM, SWEDEN 2017

High Density Simulation of Crowds with Groups in Real-Time

JACK SHABO

KTH ROYAL INSTITUTE OF TECHNOLOGY
SCHOOL OF COMPUTER SCIENCE AND COMMUNICATION

High Density Simulation of Crowds with Groups in Real-Time

JACK SHABO

Master in Computer Science
Date: July 2, 2017
Supervisor: Christopher Peters
Examiner: Johan Hoffman
Swedish title: Högdensitetssimulering av folkmassor med grupper i realtid
School of Computer Science and Communication

Abstract

To simulate crowds of people is of great social interest and is also believed to be useful when analyzing situations involving denser crowds. Many simulators seen over the years have however been struggling with simulating larger number of people, often due to a computationally expensive collision avoidance step. Furthermore, many simulators seems to forget the fact that people tend to stick together in smaller social groups rather than walking alone.

A simulator for high density crowds has nevertheless been implemented through modeling crowds as a unilateral incompressible fluid. Together with an integration of groups onto this approach, the obtained solution allows for real time simulation of up to 3000 virtual people.

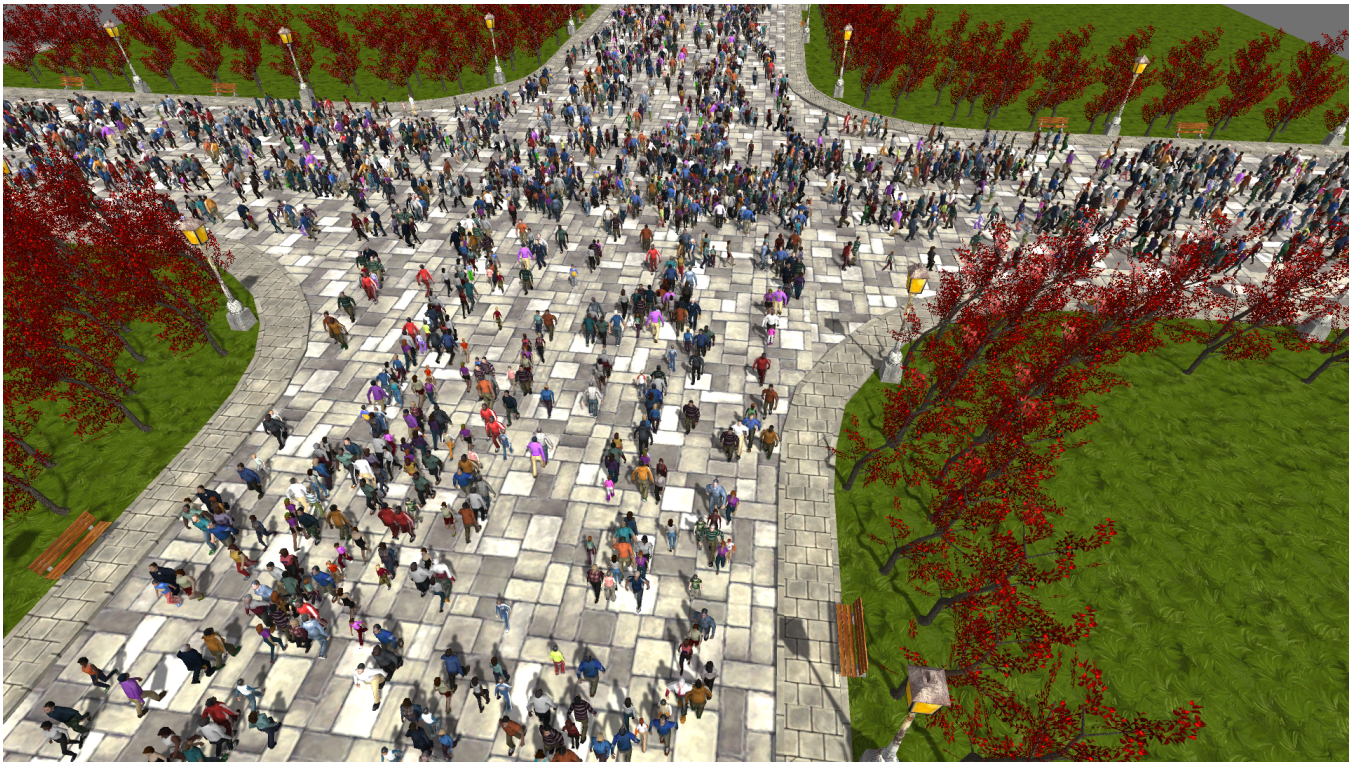
The impact of having groups in simulations has furthermore been set as the overall goal of the thesis and has been analyzed through observing the effects of groups in various scenarios. A smaller user study has also been conducted in order to gain perceptual insights of groups in various crowd densities. These have shown that groups have a smaller impact on the crowd flow, and do not put a larger strain on the performance of the simulation. Groups are further proved to be perceived differently in different densities, with a possible difficulty for scenarios in higher density.

Sammanfattning

Att simulera folkmassor är av stort socialt intresse och tros även vara till nytta när man analyserar situationer som berör mer kompaktare folkmassor. Många tidigare simulationer har dock kämpat med att simulera större folkmassor, oftast på grund av höga beräkningskostnader mot förhindrandet av att två eller fler virtuella människor kolliderar in i varandra. Dessutom verkar många simulationer glömma bort att människor oftast går i mindre sociala grupper snarare än att gå var och för sig hela tiden.

En simulering har trots detta gjorts genom att modellera folkmassor som en unilateral inkompressibel vätska. Tillsammans med en integration av grupper på detta tillvägagångssätt har lösningen visat sig alltsomallt ge simulation i realtid för uppemot 3000 virtuella människor.

Effekten av att ha grupper i simulationer har vidare analyserats i en rad olika scenarion. En mindre användarstudie har också gett insikter i hur grupper uppfattas i olika kompakta folkmassor. Resultat har visat att grupper har en mindre effekt på folkmassor i det stora hela, och lägger inte en alltför stor påfrestning på simulationers prestanda. Det har också bevisats att grupper uppfattas annorlunda i olika kompakta folkmassor, med en viss möjlighet till svårare uppfattning i högre, mer kompakta folkmassor.



All simulations were done through the game engine Unity (version 5.6.0). Furthermore, the hardware used in all simulations were on a PC machine with a 3.5GHz Intel Core i7-4770K CPU, 8.00 GB of RAM, and the NVIDIA GeForce GTX 770 GPU.

Contents

Contents **iv**

1 Introduction **1**
1.1 Simulating Crowds . . . 1
1.1.1 Crowds . . . 1
1.1.2 Crowd Properties . . . 1
1.2 Problem Description . . . 2
1.3 Research Questions . . . 2
1.4 Limitations . . . 3
1.5 Thesis Outline . . . 3

2 Background **4**
2.1 Crowd Simulation . . . 4
2.1.1 Realistic Behavior . . . 4
2.1.2 State of the Art . . . 6
2.2 Global Planning and Roadmap . . . 8
2.2.1 Sampling . . . 8
2.2.2 Grids . . . 9
2.2.3 Adaptive Roadmaps . . . 9
2.2.4 Finding a Path . . . 10
2.2.5 Homing . . . 10
2.3 Unilateral Incompressibility Constraint . . . 11
2.3.1 Eulerian vs Lagrangian . . . 11
2.3.2 The Grid . . . 11
2.3.3 UIC Concept . . . 12
2.3.4 Solving the Constraint . . . 13
2.3.5 Agent Motion . . . 14
2.3.6 Pairwise Separation . . . 15
2.3.7 Obstacles . . . 15
2.4 Evaluation . . . 15

3 Implementation **17**
3.1 Simulation Loop . . . 17
3.2 Global Planning . . . 18
3.2.1 Sampling and Placing Nodes . . . 18
3.2.2 Finding Available Paths . . . 19
3.3 Interpretation of UIC solution . . . 21

3.3.1 The Grid and the Agent . . . 21
3.3.2 Creating the LCP Equation . . . 22
3.3.3 LCP Solvers . . . 23
3.4 Agent Movement . . . 27
3.4.1 Minimum Distance Enforcement . . . 28
3.4.2 Obstacle Collisions . . . 28
3.4.3 Back/Forward Tracing . . . 28
3.4.4 Steering Towards Nodes . . . 29
3.4.5 Smoothed Turns . . . 30
3.5 Groups . . . 31
3.5.1 Requirements . . . 31
3.5.2 Implementation . . . 32
3.5.3 Calculating the Slot Positions . . . 33

4 Evaluation **35**
4.1 Scenarios . . . 36
4.1.1 Environments . . . 36
4.1.2 The Simulator's Properties . . . 36
4.1.3 Performance . . . 37
4.1.4 Group Distributions . . . 37
4.2 User Study: Perceived Groups . . . 37
4.2.1 Overview . . . 38
4.2.2 Limitations . . . 38
4.2.3 Specifications . . . 40
4.2.4 Procedure . . . 42
4.2.5 Hypothesis . . . 42
4.2.6 Evaluation . . . 43

5 Results **44**
5.1 Global Planning . . . 44
5.1.1 Scenarios . . . 44
5.1.2 Parameters . . . 44
5.1.3 Simulation . . . 45
5.1.4 Outcome . . . 47
5.2 Density Influence . . . 48
5.2.1 Scenarios . . . 48
5.2.2 Parameters . . . 49
5.2.3 Simulation . . . 49
5.2.4 Outcome . . . 52
5.3 LCP Solvers . . . 54
5.3.1 Scenarios . . . 54
5.3.2 Parameters . . . 54
5.3.3 Simulation . . . 54
5.3.4 Outcome . . . 56
5.4 Rendering Models . . . 57
5.4.1 Scenarios . . . 57
5.4.2 Parameters . . . 57

5.4.3 Simulation . . . 58
5.4.4 Outcome . . . 58
5.5 Simulation Steps . . . 59
5.5.1 Parameters . . . 59
5.5.2 Simulation . . . 60
5.5.3 Outcome . . . 62
5.6 User Study: Perceived Groups . . . 63
5.6.1 Perceived Realism . . . 64
5.6.2 Perceived Groups . . . 65
5.6.3 Outcome . . . 68
5.7 Movement Options . . . 72
5.7.1 Scenarios . . . 72
5.7.2 Parameters . . . 72
5.7.3 Simulation . . . 73
5.7.4 Outcome . . . 76

6 Discussion **78**
6.1 High Density Crowd Simulation . . . 78
6.1.1 Effects and Performance . . . 78
6.1.2 Flaws and Improvements . . . 79
6.2 Groups . . . 80
6.2.1 Performance . . . 80
6.2.2 Perception . . . 81
6.3 Social Ties . . . 82
6.3.1 Real Crowds . . . 82
6.3.2 Using Crowds . . . 82
6.3.3 Ethical Decisions . . . 82

7 Conclusion **84**
7.1 The Simulator . . . 84
7.2 Groups . . . 84

Bibliography **85**

Appendix A MPRGP Functions **89**

Chapter 1

Introduction

1.1 Simulating Crowds

1.1.1 Crowds

People have the social habit to gather into larger masses when faced with similar routes and obstacles. This results in a crowd of moving people with a density limited by the environment and the number of people alike. Crowds of larger densities, therein a higher number of people, have proven to exhibit complex behavior [20] as individuals in the mass have unique properties. Walking people have goals, varying velocities and paths in which they strive to avoid obstacles, therein other individuals, to reach their goal. Additionally, with people and larger crowds existing naturally in the everyday life, it has become of high interest to simulate crowds . Architectural planning, the design of evacuation plans, effective pedestrian walkways in high-traffic areas, and more general studies of social behavior are some example of where simulations of crowds could yield interesting aspects [28], at the least. Nevertheless, practical use often require the simulation to be as realistic as possible in order to yield results that can correlate to real situations.

1.1.2 Crowd Properties

In essence, a crowd consist of individuals, hereon referred to as "agents", who share three novel properties: A position in the world, a velocity determining its speed and consequently direction, as well as a need to avoid obstacles. The velocity's direction is governed by a particular goal, which could be expressed in general terms such as "home", "to the store", "to work" or more local and specific such as "past this road", "around the next corner and then 100 meters ahead", while its magnitude is set by the system or user. Nevertheless these properties, a crowd is a sentient collection with psychological behaviors [25] primarily sharing the property of having a certain distance between agents; other properties such as sharing the goal with other agents is an unknown factor unless there has been some form of communication exposing the contrary. An empirical study [32] made in a Chinese university shows that the preferred distance to other agents is at least one meter. This distance does however most likely differ depending on cultural and situations factors. For instance, in less denser crowds it might be preferable to maintain a comfortable distance, while at higher densities this value is somewhat forcefully lower: having many agents

walking in the same direction in a limited space leaves less free space for each surrounding agent. Furthermore, it has been shown that people usually tend to have an adjusted, smaller, distance to people sharing the particular property of being in the same *group* as oneself. Groups are a natural element in real life crowds, but have nevertheless been left out in most simulations.

1.2 Problem Description

To simulate large, realistic crowds ranging up to many thousands of agents has proven to be a computational difficulty [20], especially when real-time simulation is sought. A major contributor to this is the constraint that for realistic simulations, agents should not collide into each other or surrounding obstacles; in denser crowds this quickly becomes unfeasible to calculate efficiently. Many different approaches to this problem have been seen over the years, often with either an agent-based approach that consider the movement and collision for each individual agent or a more general continuum of crowds. The former is more computationally expensive but has proven to yield more realistic behavior, while the latter is precisely the opposite. The loss of individualism when treating individuals as part of a larger crowd is apparent.

A recent and promising combination of the two approaches proposed by Narain et al. [20] claims to achieve the desired en masse simulation with up to 1 million agents by enforcing an unilateral constraint. This approach forms the basis of this thesis, whose aim is not only for a scalable, real-time simulation of crowds with realistic and complex behavior, but furthermore to extend the existing algorithm, not in performance but instead onto the level-of-detail, amongst which the amount of useful data to be extracted from having groups in virtual crowds. The presence of groups within simulated crowds, which desire to walk together in close proximity is the challenge to be addressed and analyzed in an overall focus of high density settings.

1.3 Research Questions

This paper will investigate how adding the element of groups affects the performance, effects and overall plausibility of crowd simulations based on Narain et al. [20] algorithm. While it might be desireable to gain plausibility in the grade of realism, the thesis will limit itself to investigate whether groups can be perceived in crowds of different densities, having an overall focus of higher density crowds. An additional goal will be to investigate the various components of the simulator in order to see how these affect perception as well as performance and, in some sense, realism.

The following question state the core research question of the thesis:

What are the perceptions and effects of having groups in a high density, unilateral crowd?

This question will be answered throughout the thesis in various sections. To ease with interpretations and further conclusions, the research question is divided into the following sub-questions:

1. How is the simulation of a high density crowd affected by adding groups, with respect to both computational and simulated performance?
2. Can groups of various sizes be perceived in a simulated crowd of higher density?
3. How are components necessary for plausible high density simulations?

These questions will be answered through seperate sections, with a performance and density analysis for the first, a user study for the second, and an analysis of experimental extensions for the third. There will however be certain limitations in place throughout the thesis as a whole with respect to these, as described in the next section.

1.4 Limitations

This paper will aim to analyze and implement Narain et al. [20] solution as a basis for a crowd simulation and then attempt further extension, mainly with groups. Nevertheless, the thesis will be limited to not strive to simulate as many agents as Narain et al. [20]; real time simulation of up to some thousand agents will be sought but not exceeded. Furthermore, while crowds in various densities will be used, there will be more effort in analyzing how crowds behave in a higher density setting. Also, realistic crowd behavior and plausible environments will be considered but not prioritized over performance and group behaviors; the simulations will feature simpler characters with only a walking animation and a constant maximum velocity.

1.5 Thesis Outline

The thesis will start with a longer presentation of relevant background for crowd simulations, including important properties, components and previous approaches. Furthermore, the approach proposed by Narain et al. [20] will be introduced in the latter part of the background chapter. Following this is a chapter describing the made implementation of the simulator, based on said authors' approach with further extensions.

Then three chapters follow, which are formed to specifically answer the sought goals of this thesis. An introductory evaluation chapter will describe how evaluations are made along with a longer description of a user study that will be conducted to gain empirical insights into the plausibility of groups. Specific scenarios for the simulator and the results of these follows in a separate chapter, along with discussions relevant for each specific case; the results from the user study are also presented here. A more general discussion that correlate the overall given results with the research goals are then made in another chapter.

Finally, the most important findings are summarized in a concluding chapter.

Chapter 2

Background

This chapter presents relevant theory for the intended simulation, along with earlier work done in these. The method to simulate high density crowds according to the method of Narain et al. [20] will be introduced in section 2.3.

2.1 Crowd Simulation

2.1.1 Realistic Behavior

One of the ultimate goals of simulating the behavior of crowds is to precisely do that: to emulate the behavior of enough people such that the presence of other people become a factor in the path of individual agents. Indeed, it is assumed throughout this thesis that simulated agents have the goal to traverse from point A to point B, and that there is a particular shortest path for each pair (A, B). In reality however planning for such a path is often limited, as regular people only have a limited field of vision and cannot anticipate further crowd density ahead - therein an optimal shorter path. Despite this, Treuille et al. [29] claim that performing a global planning for all agents' preferred motion actually yields more realistic behavior.

In fact, several properties found in real crowds have been found in simulations running with this assumption. One property is the formation of lanes and vortexes [13][29] while another could be that agents handle special obstacles, such as narrow passes, in a smoothed movement [2]. In another turn of thoughts, these general properties and global planning steps have a greater challenge yet to tackle: the randomness of real crowds. Real people have a number of different factors that distinguish them from each other in some way: individuals have different velocities, height, width, clothing, mindset etc. All in all, there are a lot of different affecting factors of various levels that would be hard to generalize. Thus it would perhaps be of interest to emulate certain scenarios which attempt to stimulate these properties in some way for individual agents. Would the crowd behave in a different manner if some agents were in fact larger, or perhaps even decides to turn around to another destination since it had a change of heart on its destination? It has been proven in previous state-of-the-art that the calculations of each individuals decision and especially collision is unfeasible for denser crowds, thus the number of individual properties feasible to simulate is limited compared to that of real crowds.

Another rather important property that surprisingly enough is hardly mentioned

in many papers is the concept of groups. Being one main goal in the aspired simulation of high density crowds, groups could seemingly be generated when enough simulated individuals share a similar space and velocity, especially in narrow regions. Nevertheless, groups throughout this paper will be considered to be individuals sharing a social bond [18] along with a similar velocity and proximate positions. In reality, these could consist of families, friends, couples etc.; the number of combinations is many. Realizing this after their recordings of crowds in the United Kingdom, Singh et al. [25] modify an existing simulator to include groups of up to 4 agents.

Each group, created at the start of the simulation with a common goal, consist of a leader and followers. The followers adjust their position in a fixed pattern relative to the leader's position, while all members regulate their velocities should any member fall behind; the members ahead would slow down while the ones behind would accelerate towards some set value. Such situations have been identified when a group is intersecting with another group, or with individuals, onto which groups pave way for the colliding agents and move to avoid each other. A further interesting fact about the work of Singh et al. [25] is that they use psychological forces to handle group/individual intersections along with many other forces affecting the individuals. Such forces however might require further compatibility with the simulation technique intended in this project, namely Narain et al. [20] method. Then again, following similar approaches as seen in Singh et al. [25] paper could provide interesting behavior in the intended high density environment. Other group implementations [18][24] are also interesting to consider, where different formations of the groups are identified along with novel approaches for simulating these groups. At a more simpler approach for having groups in the crowds, one could also consider the group to be a single agent but then render the entity with multiple characters and animations, hence giving the impression of grouped agents within a crowd. This technique has been identified in modern games.

Figure 2.1: High density crowd from the game Planet Coaster[1], where groups marked in red show signs of being rendered as multiple people, but are moving as one entity altogether.

Interesting to mention is also that results in Moussaïd et al. [18] empirical analysis of pedestrians show that the amount of groups in a crowd is almost equal to the number of individuals. Of 2100 individuals, around 45% consist of groups of two and around 5% of groups of three. Similar results have been found by Wei et al. [32]. Furthermore, these observations show that the distance between members in groups is lower than the distance to other, unknown individuals. These inter-group distances range from 0.4 to 0.8 meters depending on the group type. Couples might stay closer, while friends maintain a slightly further distance. Taking logical deductions of real crowds in mind, these results agree well with what one can expect in general.

2.1.2 State of the Art

To simulate crowds and the like is not relatively new. Many novel approaches have been made, and are mentioned throughout this chapter. Nevertheless, following is a slightly more detailed description of some historical and modern ideas.

Reynolds Boids

The first and oldest of the examples is Reynolds [22] behavioral model of flocking birds - also known as "boids". While not dealing with crowds per say, the model, all in all a collection of individual actors, is a classical example renowned in many papers of fluid dynamics, crowd simulation and the like. The key objective of the work is to emulate behavior of a flock through identifying existing patterns in how real birds behave in groups and then program these into a graphical simulation. With the key observation that real birds do not seem affected by entities in the flock other than those within a local distance, Reynolds [22] define three fundamental behaviors for flocking:

- **Collision avoidance / Seperation**: Boids actively try to not fly into each other, driven by an "[...] urge to steer away from an imminent impact" [22].
- **Velocity Matching / Alignment**: Complementary to collision avoidance, velocity matching between boids decreases the risk of colliding into one another and simultaneously merges the individuals into one larger entity.
- **Flock Centering / Cohesion**: Being part of a flock, boids are driven to stay together as a flock in some respect.

These are prioritized in decreasing order, but could nevertheless be freely modified in both order and implementation in order to achieve different behaviors [26].

[1] `https://www.planetcoaster.com/`

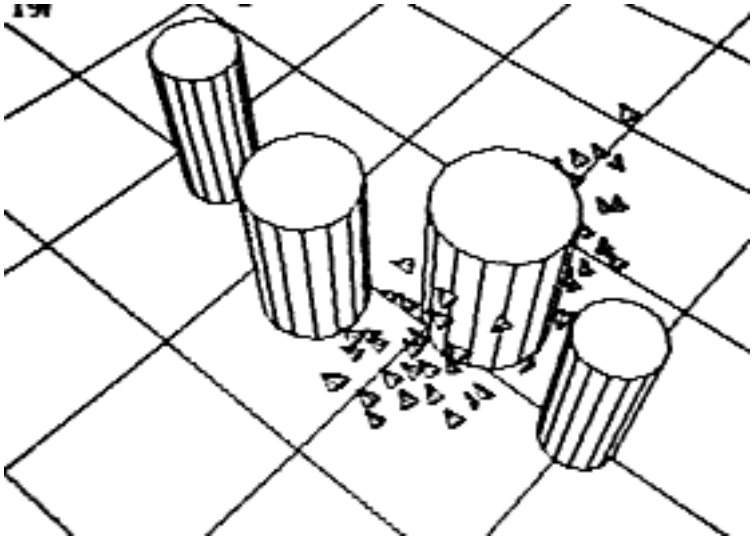


Figure 2.2: A snapshot of the boid simulation from Reynolds [22], 1987.

Social Forces

In 1995, Helbing and Molnár [13] proposed their interpretation of a social force model for pedestrian dynamics. With the movement and goal-driven behavior of pedestrians in mind, this model define an interesting and yet another classical approach for multiple-agent systems. Although being a crude description, the model defines steering behavior for agents under the assumption that pedestrians have a certain repulsive force to obstacles and nearby agents. Furthermore, agents experience an attractive force to objectives and in some part to other agents as well.

These forces and more are motivated by psychological / social behaviors observed in real crowds , therein the name Social forces. Results have yielded very plausible and interesting behaviors, such as the forming of lanes for pedestrians sharing a general direction. Nevertheless, Helbing and Molnár [13] propose and encourage extensions of the model with this behavioral model in mind *"[...] since the variables of pedestrian motion are easily measured so that corresponding models are comparable with empirical data"*. Indeed, the social forces model is familiar in modern papers, which then have their own novel approaches for a variety of situations and behaviors, such as the work of Singh et al. [25].

Reciprocal Velocity Obstacle (RVO)

A more modern approach for multi-agent simulation and navigation is the Reciprocal Velocity Obstacle (RVO) method suggested by Van Berg et al. [30] in 2008. Much like other papers [20], the authors identify the need for feasible local collision avoidance for the agents such that the seen behavior is (somewhat) realistic. Their solution for such avoidance is the RVO model which does not enforce communication between agents per say, as is common in crowd simulations. Instead RVO relies on a form of averaging velocities based on the entire set of agents. The result is a computed "oscillation-free" motion for every agent, such that they reach their intended destination with a smoothed and possibly complex route while at the same time

effectively avoiding other agents.

RVO provides very visually pleasing simulation of crowds, which except for other agents also avoid static obstacles through a more general implementation of the model. Nevertheless, shown results [30] indicate that RVO achieve real-time simulation of up to 1000 agents with 10 frames per seconds. Hence, further high density simulation would require another alternative. In spite of this, RVO is an easily-understood method that has very plausible behavior and performance well within the limits for hundreds of agents.

2.2 Global Planning and Roadmap

An important aspect of crowd simulation is the deciding of an agent's individual goal in the virtual environment. In effect, this goal governs the agent's preferred velocity at any point in time. These two factors, an agent's goal and preferred velocity, can however be decided in a variety of ways, some of which even decouple the goal from an agent's preferred velocity [33]. The manner of how the factors are set depend on the high-level model used for an agent's general steering behavior: the global planning of agents.

A proven approach [20][33] is to set an agent's goal and then determine a path for the agent to follow. Such a path could consist of several sub-goals, each of which govern an agent's preferred velocity. The path itself can be derived from a roadmap, an indication of positioned points in the virtual environment connected through a "road". Graph-structures with vertices and edges usually fulfill this representation.

2.2.1 Sampling

The roadmap of a virtual environment can be easily and swiftly constructed by performing uniform sampling of points in the environment and then connecting pairs of points free from any obstacles in between; a straight "road" connects these two points if such is the case. Said approach does however have a nature of randomness that proves to be infeasible for a more complex environment with narrow-passages, as stated by Wilmarth et al. [33]. They propose a more advanced algorithm called *Medial Axis Probabilistic Road Map* (MAPRM) that not only performs samples on the environment but on its *Medial Axis* (MA) [1]. The MA consist of all the (2D) points of the environment's obstacle-free space which have more than one closest point to the boundary of the non-free space. Sampling onto the MA is achieved through a process of uniformly selecting a point and retracting it away from its unique nearest border point on the environment until that point is no longer an unique nearest border point. The border is defined to be the environment's border as well as the border to any obstacle(s).

As seen in figure 2.3, MAPRM outperforms uniform sampling in narrow passages. Wilmarth et al. [33] does nevertheless state that such is the case when obstacles around narrow-passage are sufficiently large; obstacles of smaller area yield less good sampling on the MA.

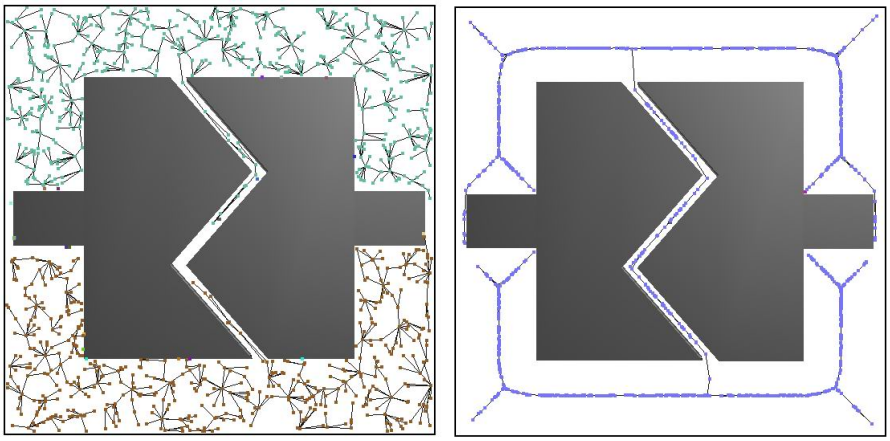


Figure 2.3: Uniform sampling (left) and medial axis sampling (right) [15]

2.2.2 Grids

Yet another approach for constructing a roadmap would be to divide the entire environment into a set of square cells of fixed size, interconnected to adjacent cells [2]. Naturally, such cells are to be interpreted as vertices in a graph with edges to neighbouring cells. The approach does evidently scale with the size of the environment quadratically, and thus perform slower than sampled roadmaps when it come to finding a path along the constructed graph.

There is however another way of using the idea of grids that is quite revolutionary to the path-finding part of roadmaps. Introduced by Chenney [6], *flow tiles* present a technique for designing velocity fields. Being placed similarly to cells in a grid, these fields remove the need for path-finding altogether by defining an agent's velocity to be precisely that of the set velocity field itself. For instance, if a velocity field allows for forward velocity in its center but not on its sides which are set to produce turning motions, an agent stepping onto that flow tile has its velocity entirely governed by those forces. Multiple placement of such tiles form 'streamlines' [6] of velocity for the crowd. That fact does reveal a limitation of flow tiles, namely that streamlines never cross; agent's in streamlines cannot cross each other in the middle of an intersection. Hence this approach could prove to be unfeasible in a high density setting.

2.2.3 Adaptive Roadmaps

Instead of constructing a roadmap depending on the environment, the work of Sud et al. [28] suggests a graph-based adaptive roadmap that changes with the movement of obstacles and crowds of agents alike. This is done by applying spring-physical forces onto the graph, represented as a number of particles dynamically updated through internal and external forces. While this novel idea could yield interesting collision-avoidance behavior through these forces, experimental results [28] show that such an approach can yield quite high update-times even for 1000 agents. In sought high density scenarios, a similar approach could defeat the struggle for real-time simulation.

2.2.4 Finding a Path

With a graph-based representation of the traversal areas in the environment, one is sure to either use an adjacency matrix or adjacency list as a underlying data-structure. The former would be more fitting to the cell-based approaches while the latter is more appropriate for sparser graphs such as the ones created by sampling. Regardless of the data structure, at some point during the simulation agents will need to compute a path in the graph from their relevant position in it, to a goal as close to a graph node as possible. It is assumed throughout this thesis that the shortest path between one such start and goal is sought.

If potentially long pre-processing time for the simulation is not an issue, it would be advantageous to pre-compute the shortest path using e.g. the all-pairs shortest path Floyd-Warshall algorithm with time-complexity O(n^3), where n = number of vertices. With a simple extension, the created structure of Floyd-Warshall can reconstruct the shortest path. This would mean that agents spawned in the running simulation would know its path in constant time.

With an increased map-size, and need for more nodes, the pre-processing could however take far too much time and in the extreme case too much memory as well to be feasible for the used machine. In this case, the path for each agent would have to be calculated dynamically as each individual agent is spawned - a common procedure [2][3][28]. This can be achieved with common path-finding algorithms such as Breadth-First Search, A* or Dijkstra's algorithm. The path-finding does nevertheless come with an extra cost. Hence Li et al. [16] propose a leader-follower model of the crowd which could prove to be useful for performance.

In the proposed method, individuals would be divided into groups with an abstract leader that carries out all global planning. With a path at hand, the followers would then consider their preferred velocity to be that of the leader. This abstraction indirectly amplifies an existing issue of having agents trapped in a local minimum state, which is known to occur when agents are trapped in a state due to obstacles and various repelling forces [17]. Nevertheless, by having each agent record their last "seen" position of the abstract leader one can divert the problem and have agents escape local minimum states [16].

2.2.5 Homing

Each agent's path to a particular goal is composed of a number of subgoals, corresponding to a number of nodes to traverse. While accurately passing by a subgoal might be infeasible with denser crowds as it presumably would yield unrealistic effects, one has to take a more free approach in the movement. A homing behavior has been suggested by Bayazit et al. [2] in which agents are provided with a "sensory range" that detects upcoming goals. Subgoals within this range cause agent's to change their target to be the next subgoal in the path, which agents then would steer towards. To avoid local minima traps, it is suggested to keep this steering, ergo the preferred velocity, of the agent to the lowest priority in which forces from obstacles and adjacent agents are considered first.

2.3 Unilateral Incompressibility Constraint

Introduced in 2007 by Narain et al. [20], the Unilateral Incompressiblity Constraint (UIC) method for crowd simulation seems like an optimal choice for high density simulation, having agents ranging up to the thousands. Being the core method of choice for this paper's simulation technique, following is a short section describing relevant methodology and then a thorough summary over their work. See the next chapter for a full specification of this paper's interpretation and done implementation.

2.3.1 Eulerian vs Lagrangian

The collection of individuals in a (larger) crowd can be seen as a continuous fluid bound by densities, velocities and the unique human capability to "think" [23][20]. In the scope of this, simulated crowds are represented as agents with extra properties that concern their movement and interaction towards other agents and the environment alike on a macro scale. Before providing examples of such properties, one would need to define a transformation from individual agents to the more collective mass with fluid-like behavior. In other words, one have to take both *Lagrangian* and *Eulerian* viewpoints of their motion into account.

Bridson and Müller-Fischer [4] describe the Lagrangian view to be that of individual particles, with a position $\overrightarrow{x}$ and velocity $\overrightarrow{u}$.Together the collective particles are seen as a changing system of particles over time. In quite contrary terms, the Eulerian view of the system is not individual particles but rather as certain pieces of quantities observed at a particular point in time. These can be density and velocity, both interchanging over time, but nevertheless measured at fixed points in time in a larger-scale observation. With further analysis to methods running under these views, Harlow [12] describe some advantages and disadvantages of the two. In summary of the latter, Lagrangian models are said to be less effective with (fluid) systems of lower distortions while Eulerian methods force properties of the fluid to be *"uniformly mixed with those of all other materials in the cell"*. With the term "cell", Harlow [12] refer to the imagined subdivision of the fluid into separate cells in a larger mesh defining the continuous form, a common aspect in numerical techniques for solving fluid dynamics.

With respect to both, Harlow [12] proposes a certain method that combines the Lagrangian and Eulerian views in order to overcome some of these disadvantages: The Particle-In-Cell (PIC) method. In short terms, the PIC method describes a Lagrangian mesh of particles that are combined and moved through an Eulerian mesh of cells; the velocity and mass of particles are saved on such cells. At the same time individual positions (and masses) are kept separately. This results in combined calculations for the fluid dynamic.

2.3.2 The Grid

The PIC method and its special properties been accounted for by Narain et al. [20] through their proposed transfer of information between discrete agents to a special continuous representation, saved on a variant of the aforementioned (Eulerian)

grid consisting of cells. Values such as density and velocity are saved at the center of such cells and are then typically used in numerical methods specific to the simulation method of choice. Typically, the values are used in calculations of gradient and/or divergence operations where approximations of derivatives are used to propagate faster solutions. In the set of available derivative approximations, the central difference (2.1) is a strong candidate due to its accuracy for smoother functions. Nevertheless, having a grid structure along with central differences approximations have proven to be flawed [12].

$$\left(\frac{\partial q}{\partial x}\right)_i \approx \frac{q_{i+1} - q_{i-1}}{2\Delta x}[12] \tag{2.1}$$

The formula exposes a flaw if the value in the cell q_{i+1} is the same as in q_{i-1}; the value of q_i, the "in-between" cell, would not only be ignored (as is the case with (2.1) in any way), but also be estimated to zero. This is commonly known as the "null-space" [12] or "pressure checkerboard" [7] problem. One common solution is the introduction of the *staggered grid* where samples of q are also saved at the middle of each cell's edge faces $q_{i+1/2}$ for certain quantities. With relevance to the quantities of crowd agents, densities are saved on the center of cells while velocities are saved on the edge faces [7][12] along with the densities of neighbouring cells [20]. Further practise is to have vertical faces save horizontal velocity and the horizontal faces save the vertical velocity of particles. These faces then allow for further approximation of derivatives using e.g. the forward difference method (2.2).

$$\left(\frac{\partial q}{\partial x}\right)_i \approx \frac{q_{i+1/2} - q_{i-1/2}}{\Delta x} \tag{2.2}$$

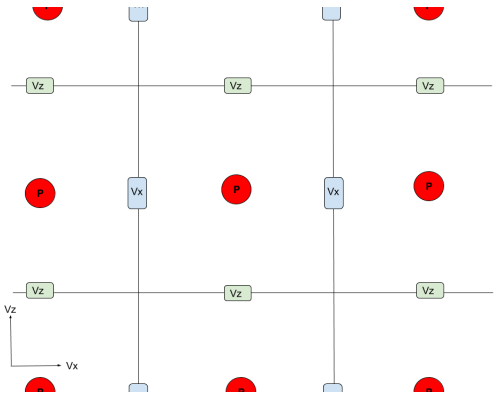


Figure 2.4: The staggered grid with density stored on the center of cells, and the 2D velocities saved on cell faces.

2.3.3 UIC Concept

Based on the observation that individuals in denser crowds experience less freedom of movement, the method proposed by Narain et al. [20] generalizes local collision avoidance. This is done through, as described, keeping the crowd in a continuous representation characterized as a continuum fluid. Nevertheless, the authors do not treat is as an ordinary physical fluid since the crowd is neither fully compressible,

since people take up space, nor incompressible as the crowd is ever changing in position and therein density. Their defined term "unilateral" describes a hybrid version that is bound by the following relationship:

$$\rho \leq \rho_{max} = \frac{2\alpha}{(\sqrt{3}d_{min}^2)} \tag{2.3}$$

Here, ρ describes the density while d_{min} is the minimal preferred distance for each agent towards other individuals. The term p_{max} in the equation enforce a maximum allowed value of the crowd's density. To restrain ρ from increasing beyond this maximum value makes up the goal of the method by Narain et al. [20]. An interpretation constraint is that a low value on the density indicates a sparser crowd, while a higher value closer to the upper limit p_{max} show signs of a denser crowd. Moreover, in the equation $\alpha \leq 1$ denotes a constant factor that allows for more perfect packing of the crowd ($\alpha = 1$) or more constrained settings ($\alpha < 1$).

As agents move closer than the set value of d_{min}, the equation shows that the upper limit constraint is then immediately violated. Thus to uphold the constraint, the authors derive a "pressure" p that corrects the velocity $\tilde{v}$ fields of the continuous flows such that (2.3) holds for the crowd. In other words, the authors define a collision avoidance step on the macro scale such that agents are kept separate in relationship to a set minimum distance.

2.3.4 Solving the Constraint

With the consideration of the (continuous) crowd as a constrained dynamical system, the authors define a relationship between the density p and velocity v of the system.

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho v) = 0 \tag{2.4}$$

This relationship defines the cornerstone for the solution of the constraint correction. Through the assumption that agents seek to advance to their destination as much as possible, within the limits of a maximum walking speed v_{max} and the density constraint, the authors suggest the following correction of $\tilde{v}$:

$$v = v_{max} \frac{\tilde{v} - \nabla p}{||\tilde{v} - \nabla p||} \tag{2.5}$$

The pressure p and density ρ satisfy:

$$\rho < \rho_{max} \Rightarrow p = 0 \text{ thus } p > 0 \Rightarrow \rho = \rho_{max} \tag{2.6}$$

$$p > 0 \Rightarrow \nabla \cdot v = 0 \tag{2.7}$$

These formulations essentially mean that when the crowd flow has a density below the maximum, there is no correcting pressure affecting the crowd and its velocity. In this situation, the crowd flows freely with each agents' preferred velocity. On the other hand, if the crowd experience a pressure then the crowd is subjugated to a maximum density with adjusted velocities until the density constraint is once again satisfied.

To cope with the non-linearity of (2.5) Narain et al. [20] split the equation into the two parts $renorm(psolve(\tilde{v}))$ where $psolve(\tilde{v}) = \tilde{v} - \nabla p$ and $renorm(\hat{v}) = v_{max}\frac{\hat{v}}{||\hat{v}||}$. This allows for the substitution of *psolve* into (2.4) such that the correcting pressure p is found:

$$\frac{\partial \rho}{\partial t} = -\nabla \cdot (\rho \tilde{v}) + \nabla \cdot (\rho \nabla p) \tag{2.8}$$

Finally by calculus involving time integration with the used timestep Δt, the following equation describes the density of the $n+1$th timestep with the previous density values ρ^n and pre-projection velocities (the velocities saved in the system) $\tilde{v}^n$:

$$\rho^{n+1} = \rho^n - \nabla \cdot (\rho^n \tilde{v}^n)\Delta t + \nabla \cdot (\rho^n \nabla p^n)\Delta t \tag{2.9}$$

By applying the constraint (2.6) to this, one is given a linear complementarity problem (LCP)

$$\mathbf{z} = \mathbf{A}\mathbf{x} + \mathbf{b}$$

where **x** is sought such that

$$\mathbf{z}, \mathbf{x}, \geq 0, \quad \mathbf{z}^{\mathbf{T}}\mathbf{x} = 0 \tag{2.10}$$

With further substitution of $\sigma = \rho_{max} - \rho$ Narain et al. [20] define the LCP to be:

$$\mathbf{z} = \sigma^{n+1} \tag{2.11}$$

$$\mathbf{b} = \sigma^n + \nabla \cdot (\rho^n \tilde{v}^n)\Delta t \tag{2.12}$$

$$\mathbf{A}\mathbf{x} = -\nabla \cdot (\rho^n \nabla x)\Delta t \tag{2.13}$$

$$\mathbf{x} = p^n \tag{2.14}$$

This was then solved by the authors using the *Modified proportioning with reduced gradient projections (MPRGP)* algorithm presented by Dostal and Schoberl [9]. With shorter description, this algorithm is an adaption of the conjugate gradient method with an active set of found solutions to the solution vector x, which is expanded through exploration of *"[...] the face of the feasible region defined by the current iterate and the reduced gradient projection with the fixed steplength"*[9]. In essence, the solution is given through exploiting the fact that the matrix A is symmetric as well as positive definite, and that various modifications allow for finite convergence of x.

To further speed up this algorithm, the *Modified Incomplete Cholesky Level-0, MIC(0)*, preconditioner [4] was applied. Being an iterative algorithm, MPRGP was warm-started with previously calculated pressure values to yield faster convergence rates.

2.3.5 Agent Motion

With the solution matrix (2.14) given after the LCP solver has run its course, the next step to uphold the UIC constraint would be to compute the corrected velocity (2.5) and then apply this to each agent. However, noting that velocity normalization was neglected when solving the pressure, Narain et al. [20] inform that *renorm($\hat{v}$)* could possibly not satisfy (2.6). Hence to amend this and some other problems, the

psolve($\tilde{v}$) is performed a second time (essentially running the solver again) such that the corrected velocity v is

$$v = psolve(renorm(psolve(\tilde{v}))) \tag{2.15}$$

with the inner *psolve*'s b-matrix clamped to positive values.

The final step is to enforce the UIC, and that is done by considering both the agents preferred velocity v_i, and corrected velocity $v(x_i)$ which depend on each agent's position x_i on the staggered grid. With $\rho(x_i)$ denoting the density at the agents position, the following formula is suggested for calculating the final velocity of agents:

$$v_i = \tilde{v}_i + \frac{\rho(x_i)}{\rho_{max}}(v(x_i) - \tilde{v}_i) \tag{2.16}$$

2.3.6 Pairwise Separation

In order for agents to maintain the minimum distance enforced mentioned in (2.3), the UIC solution step is followed with a pairwise local collision avoidance step. While the sole purpose of UIC is to enforce just that, it is unfortunately only done on a macro scale. Thus, with pairwise iteration two agents are moved away if they violate the minimum distance d_{min}[20][29]. This is not sufficient if complete separation is sought as moving apart one pair might create further violations in denser crowds. Nevertheless, it has been deemed sufficient for providing plausible simulations.

2.3.7 Obstacles

Simulation environments often contain static obstacles which agents must avoid to not cause the unrealistic effect of walking through them. This is accounted for in the UIC step by, for each grid cell, computing the percentage of free space versus occupied space of obstacles, labeled with *f*. The fraction is then integrated into the LCP solver by the modification $\sigma = f\rho_{max} - \rho$. While this modification does not prevent agents from passing through obstacles per say, it does reflect increased density when agents are in the proximity of a cell with static obstacles and hence agents would have its velocity adjusted from that cell in the given equations.

2.4 Evaluation

To evaluate a crowd simulation in certain scenarios can help highlighting interesting properties or faults of the undertaken approach. Some examples of these from Narain et al. [20] are that their agents tend to form lanes and vortices's to resolve certain situations, or that they only avoid each other when they are within a close distance. Measuring the performance of the simulation is also relevant, especially for said author and their achievement of 1 million simulated agents in 3 frames per second on a regular desktop computer. Indeed, the process of evaluating and analyzing made simulations can yield suggestions for future approaches.

However, debatably more relevant is using the evaluated data for use on real-life crowds. Scenarios such as building evacuations, street planning and regulating the flow of crowds in larger buildings can all benefit from a simulation, yielding

for example the walking behavior of people or how dense certain regions of the environment can get over time; evaluations can provide explanations of a crowd's properties and in the best case give clues for possible optimizations of the specific environments. By analyzing real-life footage directly without any prior simulation, Helbing et al. [14] explain how a tragic event involving high density crowds was analyzed and then deduced to have been caused by the panicked crowd subject to turbulence-like behavior. As a means to further process the evaluated data and behaviors in this scenario, could perhaps a simulation enlighten unseen means for preventing any future similar tragedies?

Furthermore, observing and analyzing real crowds is a definite means to gain insights about general behaviors and patterns alike. One means of observing crowds for further processing is through video recordings of pedestrians. Then as explained in Peters and Ennis [21] such recordings, collectively defined as a corpus, can be analyzed automatically (with e.g. image processing) or manually to gain some form of data that can be saved or queried on a computer. Real cases of this procedure is the tracking of pedestrian trajectories [31], the various groupings found in crowds [18][25][32] and direct statistical similarity measurements of real-world and simulated crowd data [11].

Another good way of gaining more empirical data is by conducting user studies where real people are shown imagery onto which they then share their perceptions and assessments in some controlled manner, such as filling in a survey. The provided interpretations can then be used to enhance existing crowd simulations to be more accurate real-life models [18][21] or to gain insights into human perception itself [27]. There is however a greater risk when performing a user study since the given data is not measured through a computational process; humans interpret differently and is influenced by many factors. For instance, displaying imagery of a walking crowd with variable distances between individuals, walking speeds and animated models can exhibit a very different focus for individuals rather than having the same model for each simulated agent. In the latter scenario, the focus might be of the crowd as a whole, while in the former the focus is shifted towards the variable walking speeds and distances. There is furthermore the possibility that such a study can yield unwanted remarks about having the same simulated model, rather than receiving perceptions of the variable walking speeds. The lesson to be learned here is that user studies can be great for gaining empirical data, as long as they are conducted with great care and planning as in to what is sought and where the indented focus should lie for the participants.

Contrary to the empirical data received by observations and perceptions, by directly gaining measurable, quantitative data one is left with not as many variables to consider, but rather actual "proof" that can be quantified and compared mathematically. Velocities, spatial positions at certain time indexes, the running times of simulations with set parameters down to the performance of a crowd simulation on the computer's CPU and GPU is all quantifiable data that can be compared between different simulations, or with visual inspection as done in Musse et al. [19]. The freedom, or challenge depending on the perspective, of having this type of data allows for multiple interpretations, studies and improvements alike in real and simulated crowds. Nevertheless, the risk of making misinterpretations due to having too little quantified data is as apparent here as having too few participants in a user study.

Chapter 3

Implementation

The overall approach for constructing the simulation is presented in this chapter. Initially, the approach and interpretation of Narain et al. [20] solution is presented. Following this is a presentation of made adjustments that affect the movement of the agents. Finally, the approach made for integrating groups of agents into the unilateral crowd is shown at the end of this chapter.

3.1 Simulation Loop

Following is the main simulation loop which governs all calculations made in the simulation. The loop order does not strictly follow the one used in Narain et al. [20] algorithm, mainly due to the overall simulation containing various changes. Subsequent sections describe each step in more detail.

1. All cells and nodes on the staggered grid splat their accumulated density and velocity, updating the continuous representation of the crowd.

2. The Unilateral Incompressibility Constraint is solved through constructing a LCP of the current grid values. Following this, one would acquire a correcting velocity v = psolve(renorm(psolve())).

3. For each agent, acquire its preferred velocity from the global planner. Correct its velocity, adding the respective collision avoidance velocity (see section 3.4.1) as well.

4. Update the agents position $x_{i+1} := x_i + v_i \Delta t$ where v_i is their actual velocity, x_i the previous position and Δt the timestep. Then translate each agent's new position and current velocity to the grid.

5. Perform the pairwise minimum distance enforcement, acquiring the collision avoidance velocity for each agent.

3.2 Global Planning

3.2.1 Sampling and Placing Nodes

In order to minimize the global planning for determining the preferred velocity for each agent during the simulation, all possible paths were computed beforehand. This was done by constructing a roadmap using either custom placed nodes on the map, or taking help from an optional sampling of nodes on the medial axis using an adaption of MAPRM, mentioned in Section (2.2.1). Sampling with the MAPRM adaption is believed to prevent agents from walking closer to obstacles in the scene, which might be perceived as unrealistic. Nevertheless, sampling a finite number of points open up the possibility of not sampling enough points in the environment. Such could be the case when having e.g. intersections, hard turns or objects obstructing parts of the walkable area, where nodes connecting two or more areas fail to be sampled. Having custom placed nodes on important interconnections, and relying on sampled nodes in more open areas is thought to yield both a connected environment and less predictable walking patterns in-between simulations; the option to sample a larger number of nodes is however thought to diminish the problem.

The custom placing of nodes was done through simple drag-and-drop in the environment while MAPRM was adapted as specified in Algorithm 3.1. A simplification of the original algorithm was found necessary, having a constant x determining the minimum distance between nodes k and q in the algorithm. The constant x is unique for each environment. Furthermore, the made MAPRM adaption does not fully sample on the medial axis but does yield a sample with feasible accuracy.

Algorithm 3.1. Sampling nodes using an MAPRM adaption in 2D

Input & Initialization:
*Let **F** denote the points in the environment free from obstacles.*
N ← Numbers of nodes to sample
for *i = 1 to N* **do**
  p ← *Uniformly random point in the 2D environment*
  q ← *Distance to the nearest point in **F** from p*
  $s = p,\ \overrightarrow{v} = \frac{\overrightarrow{qp}}{|\overrightarrow{qp}|}$
  if *p is not free from an obstacle* **then**
    $s = q,\ \overrightarrow{v} = \frac{\overrightarrow{pq}}{|\overrightarrow{pq}|}$
  end if
  while *s is within the environments borders ∧ s is free* **do**
    $s = s + \overrightarrow{v}$
    k ← *Distance to the nearest point in **F** from s*
    //For any minimal distance x
    if $q \neq k \wedge distance(q, k) > x$ **then**
      break
    end if
  end while
end for

3.2.2 Finding Available Paths

When having sampled and/or placed the number of desired nodes, the first step towards having a pre-computed global planner is to accumulate these into a set of vertices. The following step is to create a roadmap with all vertices connected through edges. It is however not the case that all vertices can be connected with the presence of obstacles. Thus the roadmap was created through connecting two vertices if and only if a line between them was free of any obstacles.

With the vertices connected, traversable paths are computed using an adapted version of Floyd-Warshall's algorithm. In its original form, the algorithm calculates the shortest available path between all pairs of these vertices in $O(n^3)$ time providing constant time ($O(1)$) access for each pair. With a slight modification however, the algorithm can be used to not only yield a shortest path between a pair of nodes, but also from all nodes to all other (connected) nodes. This is calculated at the same computational complexity as in the original algorithm, along with the benefit of constant time access for the (shortest) path from one node to any other node. See Algorithm 3.2 for the adapted algorithm. Along with the resulting structure depicting the shortest paths, agents in the simulation would now have a pre-computed global planner, giving constant time access to any upcoming node in their paths. The price of time-complexity $O(n^3)$ does however result in a longer pre-computational time for the simulation if n is large.

Algorithm 3.2. Floyd-Warshall All Pair Shortest Path

```
Input & Initialization:
distances ← Shortest distances between each pair of nodes (∞ if pair not connected)
maxLen ← Longest side-length of the environment, as seen in 2D
next ← Next node in path for each pair of nodes
shortestPaths ← To calculate: The shortest path from each node to another
N ← Number of nodes

{Culling out nodes too far away and initializing next}
for i from 0 to N-1 do
    for j from 0 to N-1 do
        if dist[i][j] > maxLen then
            next[i].Add(−1)
        else
            next[i].Add(i)
        end if
    end for
end for

{Main sequence}
for k from 0 to N-1 do
    for i from 0 to N-1 do
        for j from 0 to N-1 do
            //If there is a shorter path from i to j through a node k
            if dist[i][k] + dist[k][j] < dist[i][j] then
                dist[i][j] = dist[i][k] + dist[k][j]
                next[i][j] = next[k][j]
            end if
        end for
    end for
end for

{Path Reconstruction}
for i from 0 to N-1 do
    for j from 0 to N-1 do
        shortestPath[i][j] = getPath(next, i, j)
    end for
end for

procedure GETPATH(NEXT, I, J):
    p ← ∅
    if i == j then
        p.Add(i)
    else
        if next[i][j] ≥ 0 then
            p.AddAll(getPath(next, i, next[i][j])
            p.Add(j)
        end if
    end if
    return p
end procedure
```

3.3 Interpretation of UIC solution

The following section describes how the solution by Narain et al. [20] for real-time, high density crowds was implemented. In summary, the methodology follows the said authors' at a majority but does contain certain deviations.

3.3.1 The Grid and the Agent

With an approach similar to figure 2.4, the staggered grid was used as a basis for correcting the preferred velocity of agents. As described in section 2.3.3, the pressure caused by a dense crowd act as a macro-local collision avoidance step which introduces a secondary velocity that together with each agent's preferred direction corrects each agent's velocity. This secondary velocity derived from the crowds representation as a continuous unilateral fluid is created by at first translating each agent's position and velocity towards the continuous form. Having a staggered grid in the present scenario involves saving the density at the middle of each cell and the two directed velocities at the cell edge's faces. While Narain et al. [20] make use of such a staggered grid, they also use the extension of saving the density of cells at the faces as well. In their approach as well as the approach of this work, this is done by the following two formulae with each agent's mass $m_i = 1$:

$$\rho(x) = \sum_i w_x(x_i) m_i, \tag{3.1}$$

$$v(x) = \frac{\sum_i w_x(x_i) v_i}{\sum_i w_x(x_i)} \tag{3.2}$$

Here, each agent's position x_i is weighted onto the grid's cells using an inverse bilinear interpolation weight $w_x(x_i)$. In practise, this was achieved by, for each agent, considering a centered 2D area as large as a cell. Then, the inverse interpolation was achieved through distributing the percentage of the centered area overlapping the current cell (x,y) and neighbouring cells in the x, z, and xz direction (if any). Similarly, the nodes positioned on the grid cell's faces (see figure 2.4) are considered to have an area as large as a cell onto which agent's provide their intersecting percentage. Along with the nearby nodes at the same cell, the agent contributes to nodes on the neighbouring cells. See figure 3.1 for an illustration of this. Collectively, the translation caused by all agents via Eq 3.1 and 3.2 generate a continuous form, saved onto the grid through a final "splatting" of the values at each cell, as well as each of its faces, with the area of a cell. Following is the construction of a LCP, which then provide means to get the sought correcting velocity.

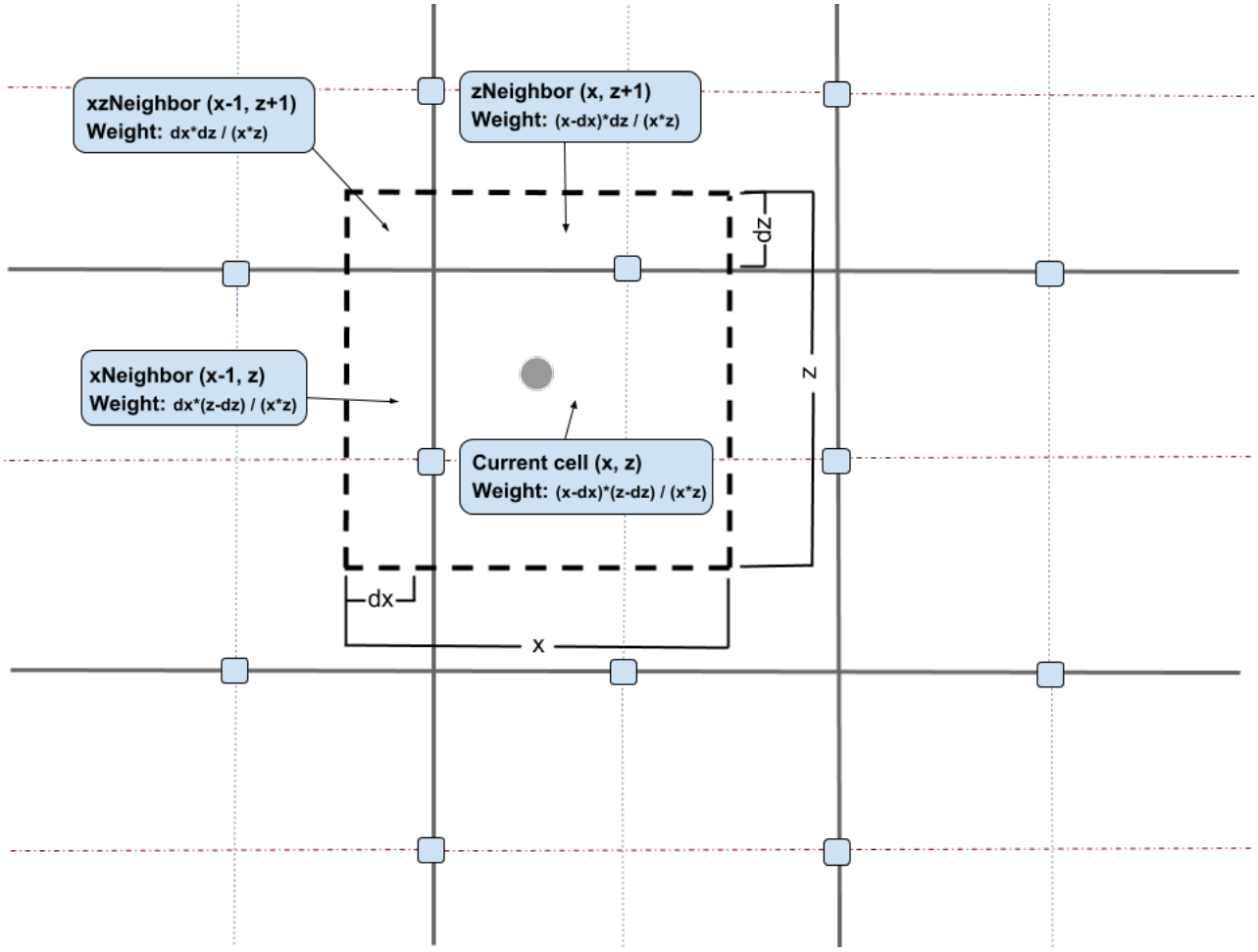


Figure 3.1: Inverse bilinear interpolation from an agent which contributes its weight to overlapping cells. Similar contributions are done to overlapping nodes, also with a cell's size interval.

3.3.2 Creating the LCP Equation

Having calculated the crowd's density and velocity contributions in a timestep Δt, equations 2.11 to 2.14 define the LCP to solve for acquiring the pressure in 2.14. Using the aforementioned derivative approximations with forward differencing, both **b** and **A** are calculated through a cell-wise iteration. In the implementation, the grid's cells had a similar width as well as height and thus, let Δs denote the width and height of cells hereon.

The calculation of b was done through transforming the cell's indices (i, j) on the grid to a one-dimensional indexed representation starting at cell (0,0) and then calculating column-wise along the x axis until the end of the row; the calculation of the next row's cells was appended to b after this. A direct interpretation of Eq 2.12 using horizontal and vertical nodes on the cell's faces along with forward differencing

yields:

$$\mathbf{b_{i,j}} = f\rho_{max} - \rho_{i,j} + (\frac{\rho_{i,j+\frac{1}{2}} v^x_{i,j+\frac{1}{2}} + \rho_{i+\frac{1}{2},j} v^z_{i+\frac{1}{2},j} - \rho_{i,j-\frac{1}{2}} v^x_{i,j-\frac{1}{2}} - \rho_{i-\frac{1}{2},j} v^z_{i-\frac{1}{2},j}}{\Delta s})\Delta t \tag{3.3}$$

With similar conditions, A was calculated as a one-dimensional representation along the diagonal of a square matrix. There, each diagonal entry held the average density of cell (i, j), based on its four surrounding faces in the staggered grid. Furthermore, these surrounding nodes were calculated to the same row in A. The nodes to the left and right of cell (i, j) were set to the left and right to the diagonal entry (Eq 3.5, 3.6), while the upper and lower nodes were positioned to the left and right at a distance of "the number of cells per row" (Eq 3.7, 3.8); each row in A accounted for one cell's 5 values. Being sparse, A was represented as a (list of lists) sparse matrix throughout the calculations for performance. Hence with a direct interpretation from Eq 2.13, A was set to be all zeroes except for:

$$\mathbf{A_{i,j;i,j}} = \frac{\rho_{i-\frac{1}{2},j} + \rho_{i+\frac{1}{2},j} + \rho_{i,j-\frac{1}{2}} + \rho_{i,j+\frac{1}{2}}}{\Delta s^2}\Delta t \tag{3.4}$$

$$\mathbf{A_{i,j;i,j+1}} = -\frac{\rho_{i,j+\frac{1}{2}}}{\Delta s^2}\Delta t \tag{3.5}$$

$$\mathbf{A_{i,j;i,j-1}} = -\frac{\rho_{i,j-\frac{1}{2}}}{\Delta s^2}\Delta t \tag{3.6}$$

$$\mathbf{A_{i,j;i+1,j}} = -\frac{\rho_{i+\frac{1}{2},j}}{\Delta s^2}\Delta t \tag{3.7}$$

$$\mathbf{A_{i,j;i-1,j}} = -\frac{\rho_{i-\frac{1}{2},j}}{\Delta s^2}\Delta t \tag{3.8}$$

This interpretation can be extracted by forward differencing Eq 2.13 such that A can be extrapolated from:

$$\begin{aligned}\mathbf{Ax} = \mathbf{Ap} = -\nabla \cdot (\rho^n \nabla p)\Delta t = -\left(\frac{\partial}{\partial x}\left(\rho\frac{\partial p}{\partial x}\right) + \frac{\partial}{\partial y}\left(\rho\frac{\partial p}{\partial y}\right)\right)\Delta t \\ = ... = \left(p_{i,j}\left(\frac{\rho_{i-\frac{1}{2},j} + \rho_{i+\frac{1}{2},j} + \rho_{i,j-\frac{1}{2}} + \rho_{i,j+\frac{1}{2}}}{\Delta s^2}\right)\right. \\ \left. - p_{i,j+1}\frac{\rho_{i,j+\frac{1}{2}}}{\Delta s^2} - p_{i,j-1}\frac{\rho_{i,j-\frac{1}{2}}}{\Delta s^2} - p_{i+1,j}\frac{\rho_{i+\frac{1}{2},j}}{\Delta s^2} - p_{i-1,j}\frac{\rho_{i-\frac{1}{2},j}}{\Delta s^2}\right)\Delta t\end{aligned} \tag{3.9}$$

3.3.3 LCP Solvers

The implementation considered three iterative LCP solving algorithms which were adapted using the aforementioned data and structures. These were the MPRGP algorithm [9], an MPRGP variant with preconditioning in the face [4][8] and then a different algorithm called *Projected Gauss-Seidel with Projected Successive Over Relaxation (PSOR)* by Erleben [10]. Worth to mention is that since Dostal [8] require some additional steps for the preconditioner matrix M for the preconditioned MPRGP variant, such as converting M to a Moore-Penrose generalized inverse, simplifications for acquiring M was possible due to the properties of UIC seen in Eq 2.6.

While these were then generally implemented as specified by the respective author, a couple of modifications were made to speed up the solving process. All except

one modification were algorithm specific in which the common modification were to check whether the LCP terminating conditions (2.10) were satisfied, which is shown in algorithm 3.3.

Algorithm 3.3. Terminal condition for all LCP solvers. Returns true or false depending on whether x have converged to a solution.

Input & Initialization:
$A \leftarrow$ *Symmetric positive definite matrix of order n*
$b \leftarrow$ *Vector of order n, in the format of (2.12) and (!Insert Ref!)*
$x \leftarrow$ *Vector of order n, the currently calculated pressure values (2.14)*
$\epsilon \in (0, 1)$

```
z = Ax + b
sum ← 0
for i = 0 to len(z)-1 do
    if z[i] < 0 || x[i] < 0 then
        return false
    end if
    sum += z[i]*x[i]
end for
return |sum| < ϵ
```

The MPRGP algorithms were used since the preconditioned variant was used in the solving of Narain et al. [20] LCP, and since it was in the authors interest to compare this preconditioned variant with the original. Hence Algorithms 3.4 and 3.5 specify these as they were implemented in this project. As seen there, the MPRGP approaches both have the significant modification of having a maximum number of allowed iterations for the solver. This was deemed necessary as some cases of the simulations (and parameters) can cause far too long convergence for x. Except for having the aforementioned terminal conditioner common for all LCP solvers, the two MPRGP variants follow the original specification without any further modifications.

Terms and symbols used in the algorithms by the original author are present in these without further explanation in this chapter; for a more complete specification, see Appendix A.

Algorithm 3.4. Adapted MPRGP algorithm

Input & Initialization:
$A \leftarrow$ Symmetric positive definite matrix of order n
$b \leftarrow$ Vector of order n, in the format of equation 2.12
$x \leftarrow$ Vector of order n, the previously calculated pressure values (2.14)
$\ell \leftarrow$ Vector of order n, consisting of zeroes
maxIt $\leftarrow$ Set maximum number of iterations
$\epsilon \in (0, 1)$
$\bar{\alpha} \leftarrow \frac{1}{2\|A\|}$
$\Gamma \leftarrow 1$
$g = Ax^0 + b,\ p = \varphi(x^0),\ cnt = 0$

while $\|V(x)\| > \epsilon \wedge cnt < maxIt \wedge \neg EndCondition(A, x, b)$ **do**
  $cnt = cnt + 1$
  if $\|\boldsymbol{\beta}(x)\|^2 \leq \Gamma\tilde{\boldsymbol{\varphi}}(x^k)^\intercal\boldsymbol{\varphi}(x^k)$ **then**
    {Trial conjugate gradient step}
    $\alpha_{cg} = \frac{\mathbf{g}^\intercal\mathbf{p}}{\mathbf{p}^\intercal\mathbf{A}\mathbf{p}},\ \mathbf{y} = \mathbf{x^k} - \alpha_{cg}\mathbf{p}$
    $\alpha_f = min\{\frac{(x_i^k - \ell_i)}{p_i} : p_i > 0\}\ (\infty\ if\ all\ p_i \leq 0)$
    if $\alpha_{cg} \leq \alpha_f$ **then**
      {Conjugate gradient step}
      $\mathbf{x^{k+1}} = \mathbf{y},\ \mathbf{g} = \mathbf{g} - \alpha_{cg}\mathbf{A}\mathbf{p}$
      $\beta = \frac{\boldsymbol{\varphi}(\mathbf{y})^\intercal\mathbf{A}\mathbf{p}}{\mathbf{p}^\intercal\mathbf{A}\mathbf{p}},\ \mathbf{p} = \boldsymbol{\varphi}(\mathbf{y}) - \beta\mathbf{p}$
    else
      {Expansion step}
      $\mathbf{x^{k+\frac{1}{2}}} = \mathbf{x^k} - \alpha_f\mathbf{p},\ \mathbf{g} = \mathbf{g} - \alpha_f\mathbf{A}\mathbf{p}$
      $\mathbf{x^{x+1}} = P_{\Omega_B}(\mathbf{x^{k+\frac{1}{2}}} - \bar{\alpha}\boldsymbol{\varphi}(\mathbf{x^{k+\frac{1}{2}}})$
      $\mathbf{g} = \mathbf{A}\mathbf{x^{k+1}} + \mathbf{b},\ \mathbf{p} = \boldsymbol{\varphi}(\mathbf{x^{k+1}})$
    end if
  else
    {Proportioning step}
    $\mathbf{d} = \boldsymbol{\beta}(\mathbf{x^k}),\ \alpha_{cg} = \frac{\mathbf{g}^\intercal\mathbf{d}}{\mathbf{d}^\intercal\mathbf{A}\mathbf{d}}$
    $\mathbf{x^{x+1}} = \mathbf{x^k} - \alpha_{cg}\mathbf{d},\ \mathbf{g} = \mathbf{g} - \alpha_{cg}\mathbf{A}\mathbf{d},\ \mathbf{p} = \boldsymbol{\varphi}(\mathbf{x^{x+1}})$
  end if
end while
{Return solution x^k}

Algorithm 3.5. Adapted MPRGP algorithm with MIC(0) preconditioning

Input & Initialization:
A ← SPD matrix of order n
b ← Vector of order n, in the format of (2.12) and (!Insert Ref!)
x ← Vector of order n, the previously calculated pressure values (2.14)
ℓ ← Vector of order n, consisting of zeroes
M ← MIC(0) preconditioner, SPD in the face of indices's for x where $x_i \neq \ell_i$
maxIt ← Set maximum number of iterations
$\epsilon \in (0, 1)$
$\bar{\alpha} \leftarrow \frac{1}{2\|A\|}$
$\Gamma \leftarrow 1$
$g = Ax^0 + b,\ p = \varphi(x^0),\ z = Mg,\ p = z,\ \ cnt = 0$

while $\|V(x)\| > \epsilon \wedge cnt < maxIt \wedge \neg EndCondition(A, x, b)$ **do**
  $cnt = cnt + 1$
  if $\|\boldsymbol{\beta}(x)\|^2 \leq \Gamma \tilde{\boldsymbol{\varphi}}(x^k)^\intercal \boldsymbol{\varphi}(x^k)$ **then**
    {Trial conjugate gradient step}
    $\alpha_{cg} = \frac{\mathbf{z}^\intercal \mathbf{g}}{\mathbf{p}^\intercal \mathbf{A}\mathbf{p}},\ \mathbf{y} = \mathbf{x^k} - \alpha_{cg}\mathbf{p}$
    $\alpha_f = min\{\frac{(x_i^k - \ell_i)}{p_i} : p_i > 0\}\ (\infty\ if\ \ all\ p_i \leq 0)$
    if $\alpha_{cg} \leq \alpha_f$ **then**
      {Conjugate gradient step}
      $\mathbf{x^{k+1}} = \mathbf{y},\ \mathbf{g} = \mathbf{g} - \alpha_{cg}\mathbf{A}\mathbf{p},\ \mathbf{z} = \mathbf{M}\mathbf{g}$
      $\beta = \frac{\mathbf{z}^\intercal \mathbf{A}\mathbf{p}}{\mathbf{p}^\intercal \mathbf{A}\mathbf{p}},\ \mathbf{p} = \mathbf{z} - \beta\mathbf{p}$
    else
      {Expansion step}
      $\mathbf{x^{k+\frac{1}{2}}} = \mathbf{x^k} - \alpha_f\mathbf{p},\ \mathbf{g} = \mathbf{g} - \alpha_f\mathbf{A}\mathbf{p}$
      $\mathbf{x^{x+1}} = P_{\Omega_B}(\mathbf{x^{k+\frac{1}{2}}} - \bar{\alpha}\boldsymbol{\varphi}(\mathbf{x^{k+\frac{1}{2}}})$
      $\mathbf{g} = \mathbf{A}\mathbf{x^{k+1}} + \mathbf{b},\ \mathbf{z} = \mathbf{M}\mathbf{g},\ \mathbf{p} = \mathbf{z}$
    end if
  else
    {Proportioning step}
    $\mathbf{d} = \boldsymbol{\beta}(\mathbf{x^k}),\ \alpha_{cg} = \frac{\mathbf{g}^\intercal \mathbf{d}}{\mathbf{d}^\intercal \mathbf{A}\mathbf{d}}$
    $\mathbf{x^{x+1}} = \mathbf{x^k} - \alpha_{cg}\mathbf{d},\ \mathbf{g} = \mathbf{g} - \alpha_{cg}\mathbf{A}\mathbf{d},\ \mathbf{z} = \mathbf{M}\mathbf{g},\ \mathbf{p} = \mathbf{z}$
  end if
end while
{Return solution x^k*}*

The third and different LCP solver, PSOR, was added due to its simplicity and efficiency. In practise, the algorithm is an iterative algorithm that improve the solution x for a set number of iterations. An improvement per say is done through a process of successive over-relaxation as defined in Erleben [10]; in essence it performs controlled "guesses" of the solution to the LCP. Termination of the PSOR algorithm is done through checking whether the constraints hold as specified before, but also whether the values in x have made a significant change between two iteration steps. Here, a significant change is defined to occur when the change of the maximum values of x in two iterations, $|xi_{max} - x^{i+1}_{max}|$, is greater than ϵ, a small value; the algorithm would continue if such were the case and terminate if vice versa. This makes Algorithm 3.6 susceptible to more inexact solutions, but is nevertheless deemed to be advantageous as repeating the guesses up to the maximum number of iterations can be expensive if the solution converges slowly. In any case, the given solution would still be somewhat exact and would likely be very different at the next time step in any case.

Algorithm 3.6. Adapted PSOR algorithm

Input & Initialization:
A ← SPD matrix of order n
b ← Vector of order n, in the format of (2.12) and (!Insert Ref!)
x ← Vector of order n, the previously calculated pressure values (2.14)
maxIt ← Set maximum number of iterations
$\theta \in (1, 2)$
$\epsilon \in (0, 1)$

for $k = 1\, to\, maxIt$ **do**
  if EndCondition(A, x, b) **then**
    break
  end if
  $oldMax = max(x)$
  for $i = 0\, to\, (|A| - 1)$ **do**
    $\mathbf{x_i = max(0, x_i - \theta \frac{A_i x + b_i}{A_{ii}})}$
  end for
  if $||max(x) - oldMax|| < \epsilon$ **then**
    break
  end if
end for
{Return solution x^k}

3.4 Agent Movement

Having acquired the preferred and correcting velocity for each agent, it would now be possible to insert these into Eq 2.16 for gaining an actual velocity followed by an agent moving. Nevertheless, there are some settings and features of the simulation that could affect the overall movement. Some, such as the minimum distance enforcement presented next, are mandatory for the simulation while some settings are completely optional but does affect the plausibility of the seen agents.

3.4.1 Minimum Distance Enforcement

The pairwise separation step of the process presented by Narain et al. [20] (See section 2.3.6) was done through introducing a *collision avoidance velocity* v_c for each agent [3]. This velocity was appended to Eq 2.16 for each agent, as in Eq 3.10. While the procedure for deriving this velocity was done pair-wise, it was not necessarily done between each pair of agents. An optimization was made by introducing a 2D neighbour-list of set size in which agents were "binned" depending on their current position and then only compared with the agents in the same and adjacent bins. While the worst-case time complexity for this approach remain the same as in a strict pair-wise comparison, having a high-resolution neighbour-list and the agents well-spread out (as is the purpose of the collision avoidance steps in this simulation), the made optimization yielded good performance for this step. As for the calculation of the collision avoidance velocity per say, it was done through considering pairs of agents within a distance less than twice the preferred minimum distance d_{min}. The first agent of these thus appended a velocity equal to the opposite direction of the second agent to its v_c.

$$v_i = \tilde{v_i} + \frac{\rho(x_i)}{\rho_{max}}(v(x_i) - \tilde{v_i}) + v_c \tag{3.10}$$

3.4.2 Obstacle Collisions

With a certain percentage of a cell marked as obstructed in the calculations, section 2.3.7 defines how agents would avoid cells with higher pressures if they had obstacles on top of them. This was deemed to be insufficient as agent's would still partly walk through obstacles at e.g. sharp turns or through being "pushed" by other agents. Hence, the developing environment for this simulation, Unity, and its built-in collider system was (optionally) used between agents and objects marked as obstacles. This prevented agents from moving into obstacles, but could result in them "sliding" alongside the obstacles instead. The effect can especially be seen in narrow environments and/or high density settings. Furthermore, using the built-in system results in a slight decrease of performance in the simulation which scales with the number of agents. Also, frequent collision with obstacles, which is more likely with many agents, cause a higher performance drop. It was nevertheless deemed to be a reasonable trade-off for the plausibility gain.

3.4.3 Back/Forward Tracing

While the global planner provides a preferred velocity for each agent, these are affected by both a correcting velocity and a minimum distance enforcement. Hence the placement of objects and nodes in the environment could lead agents astray from their intended path, or could create a movement that does not quite reflect how real people would walk. Agents trapped in a local minima state is a known issue in crowd simulation and could be the case for the former, while agents slavishly along a calculated path could be the case for the latter. To amend these two cases, the simulation offer two optional settings which does affect performance and also the plausibility of the agent's movement, however not necessarily for the better.

Through the help of ray-surface intersection tests, also known as raycasting, two different features were extracted for each agent. Originating from an agent's world position, a raycast made the agent:

- Walk back to its previously visited node if the "Walk back" setting was used and the raycast, having the agent's current goal in the path as destination, intersected with at least one obstacle along the way. In effect, this setting makes agents retrace their steps if they cannot "see" their goal.
- Move towards the goal after the current goal if the "Skip Node" setting was used and the raycast, having the agents subsequent goal in the path as destination, was free from intersection with any obstacle. This can be seen as agents walking towards its primary goal faster if it can "see" ahead of the path.

Both settings require exactly one raycast of variable length in each timestep, affecting the performance of the simulation slightly individually but notably en masse. These does also affect the plausibility of the simulation in various ways. The "Walk back" feature can be really good for agents stuck, but can yield the unrealistic effect of agents walking back to where they came from rather than moving forward towards its intended final goal. With respect to the extreme cases of being stuck in a local minima where agents hardly change position at all and instead continue to walk into obstacles, it is believed that the trade-off is fair. Nevertheless it should be stated that this setting does not guarantee escapes from local minima situations; these could theoretically still happen with complicated environments and/or crowd densities. In contrast however, the "Skip Node" setting does effectively guarantee that the agent will find it possible to move towards its subsequent goal. Sharp edges of the environment could despite this cause a momentous "slide" effect on a collision if the agent walk straight towards its new goal after "seeing" it. The reason for this effect is due to the one-dimensionality of a raycast, which only having a length, compared to an agent which also has a width.

3.4.4 Steering Towards Nodes

The agents are walking along a precomputed path with sampled and placed nodes. These could be either densely or carefully placed in the simulation, yet in the end agents would receive a precomputed path that minimizes the distance from its start to its goal; the minimal set of nodes required for the path is considered while all other nodes are ignored. Hence agents will walk along the path, possibly passing through some subsequent nodes, until the agent reaches the node set as a goal upon which the agent would be removed from the simulation. However, one question remains untold: How would the agent know when it has passed through any node in its path?

A node was defined as a circle with a certain radius and position. Thus for an agent to pass through a node in its path and change its velocity towards the subsequent node, the initial preferred velocity of the agent had the direction from the agents position towards the circle's center. Then, an agent was set to have visited its target node once it was within it's radius. Having a larger radius means a larger "area of contact" for the agents which, opposite to having a smaller radius, means

that more agents can pass through the nodes without struggling to essentially touch the node.

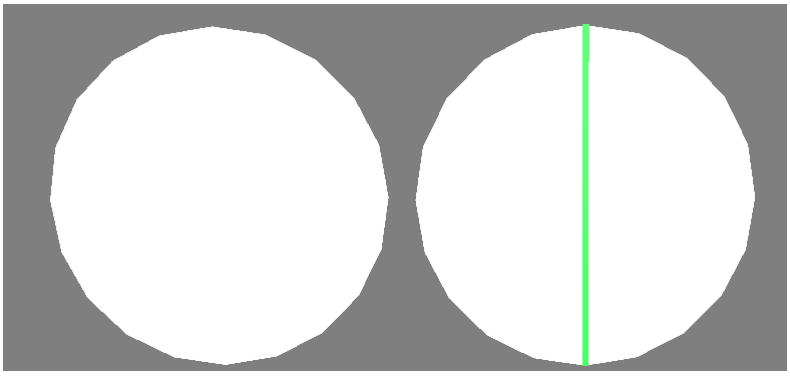


Figure 3.2: Left: A regular node on which agents steer towards its center. Right: Another node with agents steering towards their closest point along the green line.

Having agents walk towards a certain node's center does however introduce a focused steering early on in the path, causing agents to become more packed as they walk. To amend this behavior, if not sought with e.g. narrow passages or doorways, a second type of node was introduced. This new node had all the properties earlier mentioned, with the exception that agents would steer towards the nearest point on a line across its diameter as illustrated in figure 3.3. With this, the agents can maintain their original preferred direction relative to its start position from its previous node and the nearest point on the node diameter. A side effect of using this node is that agents steering towards a point no other than the node center could "touch" the node on a distance longer than the radius from the node's center; the radius of the entire node is used for distance comparison as of yet. This is especially noted on points closer towards the node's border.

3.4.5 Smoothed Turns

Essentially, a path in the simulation consist of a minimum of two nodes with a start and a goal. If it consists of more nodes then the intermediate nodes must at all times result in a change of direction for the agent, or in other words, the agent turning. If such were not the case the goal node would be reachable in a straight line right from the start, hence having no need for any additional nodes in the path. Furthermore, if the agent would make a turn they would without any prior modification perform a direct rotation towards the next node in the path. This minimizes the distance for the agent along the path, but is quite unrealistic for normal people. In reality, people make turns in a smoothed manner through a gradual rotation occurring over a smaller time frame. As an optional setting, the simulation translated this behavior by rotating the agent a small amount if the angle between the agents direction and target direction was larger than 20 degrees and if the agent had not walked past the node activating the turn. The rotation was done towards the target direction with a value (in degrees) equal to the following formula, with Δt as the used time step:

$$\Delta t((35.0 - 400\Delta t) \tag{3.11}$$

The values were chosen based on trial and error with different time steps ($0.01 \leq \Delta t \leq 0.05$).

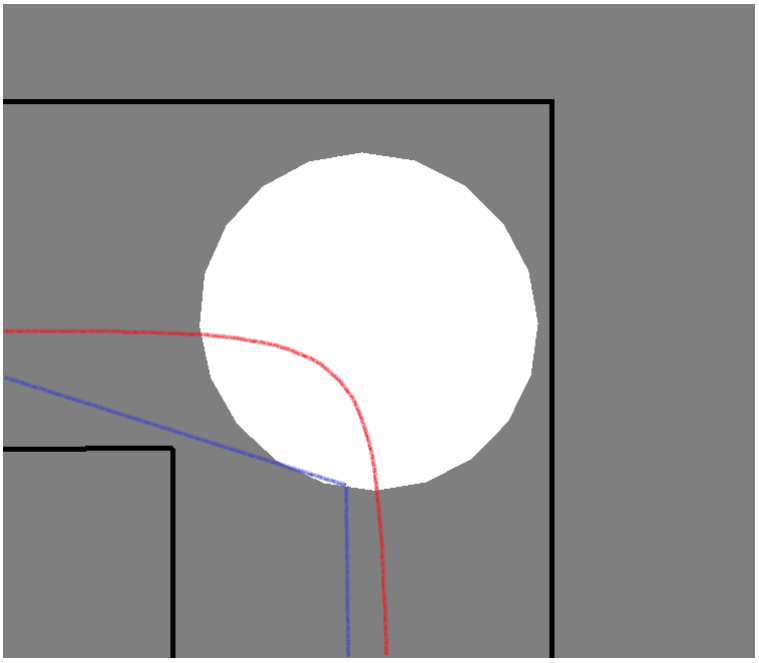


Figure 3.3: Minimal distance turning (blue) and smoothed turning (red)

3.5 Groups

Groups of sizes 2-4 were successfully integrated to the implementation. In short, this was done through an interpretation of the approach presented by Rojas et al. [24] combined with mechanisms used by Singh et al. [25]

3.5.1 Requirements

In order to integrate groups into the described simulation, the respective group members must be subject to the same requirements set onto individuals. Mainly, this entails maintaining a minimum distance and being subject to the unilateral constraint in order to achieve collision avoidance with other agents. Furthermore, group members should also utilize the same global planner as individual agents, having the same property of walking along a precomputed path. Being subject to the minimum distance requirement, groups must consequently also have the option of separating and rejoining in the cases where other agents intersect into the group; if groups were unable to separate they would produce an in-plausible local blocking behavior. It was further discovered through a simpler implementation of groups, by rendering one agent as a group of four and adapting minimum distance values therein, that the blocking behavior affected the simulation globally over time. Another reason for having groups separating and then rejoining is the fact that a group is not considered to be a couple of individuals that happens to be near each other, but instead a collection of individuals with some form of social connection. The re-

joining behavior indicate the fundamental will to try and stick together and move towards a goal without leaving any members of the group behind.

3.5.2 Implementation

Following Rojas et al. [24] dynamic social formation of groups, a group in the made simulations consists of five positioned "slots" that divide agents movement into a certain two-level steering. As seen in Figure 3.4, these five are positioned ahead of the agents. Having each agent move towards their closest slot then determines an individual steering, which is influenced by a designated leader agent. Rojas et al. [24] groups consist of rendered agents who move towards individual slots, and one invisible, colliderless "group agent" that determine all global planning and consequently updates the slot positions based on its current position in world space. With a low density crowd, this approach would cause group members to move close to each other and towards a shared direction under constant change. Nevertheless, an immediate flaw in higher densities is that the group agent would move ahead freely while denser crowds would cause the actual group to be left behind; agents in high density crowds have reduced freedom of movement and could potentially be slowed down due to collision avoidance velocities or the correcting "pressured" velocity in even higher densities.

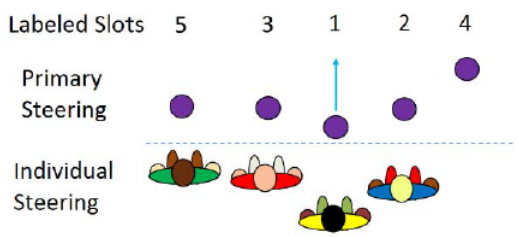
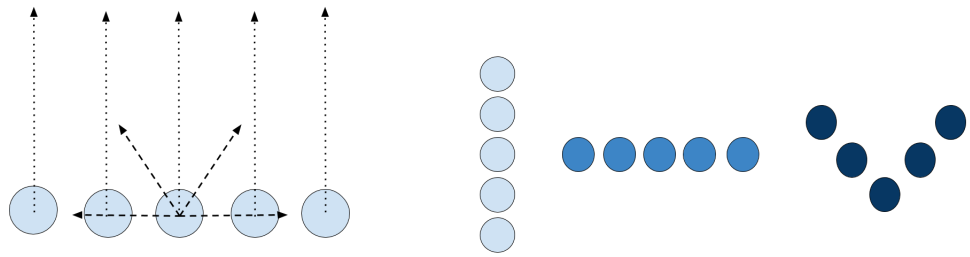


Figure 3.4: A group and their available slot positions. [24]

A better adaption was to assign one actual group member to be the leader that updates the five slot positions based on its position. All other agent's in the group would then individually steer towards the closest slot. Should they already be near their respective slot, they would be close enough to simply follow the global planner's path as any other agent, which effectively fulfills the "primary steering" of the group. The leader was left out of this movement system since the slots were positioned around his centered position; the leader was treated as a individual agent having some extra calculations. Effectively, the described movement constituted group member's preferred velocity. All members in a group were further bound by the same correcting velocity as in Eq (3.10). This integrates the groups into the simulation with ease and in accordance with mentioned simulation steps, but can cause unwanted behavior when considering the other affecting velocities. For instance, it was observed that if other agents caused grouped agents to be pushed forward or if the group member assigned to be the leader was slowed down, agents ahead could turn around and move back towards its desired slot position. The described behavior is unrealistic, and causes similar blocking behavior identified before. Thus, much like Singh et al. [25] did in their work, group members were set to adjust their

preferred velocity if they were at a distance larger than 3 meters away from their slot's position. Should they be behind the slot, they would increase their preferred velocity in the same direction by 30%. Similarly, agents would decrease their preferred velocity by the same amount but then have their preferred velocity set to its primary steering direction, given by the global planner. This results in that agents slow down or increase their pace depending on their distance towards the group, without ever moving towards the opposite direction due to a preferred velocity.

3.5.3 Calculating the Slot Positions



(a) Raycast directions (b) River, Abreast and UV formations

Figure 3.5: Implementation details for groups.

Calculating the position for each group's five slots was done according to Rojas et al. [24] descriptions and the underlying template based interpolation approach. In short, at each instance in time the slots were set to be aligned according to one of three set formations. These were the abreast formation, the UV formation and the river formation, see figure 3.5 b). Each timestep either resulted in an update using the same formation, or an interpolation towards another formation, as described by the following slot calculating process:

1. Unassign all slots in the group, and then for each agent assign it to its closest slot. The chosen slot would be locked for further agents. A slight modification was made so that agents chose slots closer to the group leader (slots 2 and 3 in Figure 3.4).

2. Using ray casting, send out one ray from each slot and 4 rays from the group leader's position. The rays shooting from the slots was 10 units in length and had the same direction as the group leader's forward direction. Furthermore, the 4 rays were 5 units long and were directed to the group leader's left and right direction as well as +-45 degrees off from the leaders forward direction. See figure 3.5 a). These would then have their shooting distance recorded, with only being smaller than 10 or 5 units long if they hit a static obstacle along its direction. For the 5 rays emitting from the slots, the ray with the smallest distance was recorded as m_o, and for the 4 smaller rays from the group leader's position the smallest distance of these was labeled m_f.

3. Using the gained values m_o, m_f, the following thresholds define the current group formation to be assumed:
   (a) if $m_o < 1$ or $m_f < 5$ then go for a river formation
   (b) else if $m_o < 2$ or $m_f < 10$ then go for a uv formation
   (c) else go for an abreast formation
4. Given the formation to be used, the desired slot positions $D_{i,j}$ was gained from the values in figure 3.6. Here P_i denote the group leader's world position and V_f / V_r its forward/right vector directions.
5. The slot's new position $R_{i,j}$ is then calculated with $R_{i,j} = (1-s)*C_{i,j} + s*D_{i,j}$. Here, $C_{i,j}$ is a slot's current position and s is a small value , clamped to less than one, incremented by 0.001 and then reset to 0 once a new formation desired. Effectively, the equation defines simpler interpolation that causes the group agents to gradually change formation when near obstacles.

ABREAST-TEMPLATE

$D_{i1} = P_i + 0.0 * V_f + 0.0 * V_r$
$D_{i2} = P_i + 0.0 * V_f + 0.6 * V_r$
$D_{i3} = P_i + 0.0 * V_f + -0.6 * V_r$
$D_{i4} = P_i + 0.0 * V_f + 1.2 * V_r$
$D_{i5} = P_i + 0.0 * V_f + -1.2 * V_r$

RIVER-TEMPLATE

$D_{i1} = P_i + -1.2 * V_f + 0.0 * V_r$
$D_{i2} = P_i + -0.6 * V_f + 0.0 * V_r$
$D_{i3} = P_i + 0.0 * V_f + 0.0 * V_r$
$D_{i4} = P_i + 0.6 * V_f + 0.0 * V_r$
$D_{i5} = P_i + 1.2 * V_f + 0.0 * V_r$

U/V-TEMPLATE

$D_{i1} = P_i + -0.2 * V_f + 0.0 * V_r$
$D_{i2} = P_i + 0.0 * V_f + 0.5 * V_r$
$D_{i3} = P_i + 0.0 * V_f + -0.5 * V_r$
$D_{i4} = P_i + 0.3 * V_f + 0.9 * V_r$
$D_{i5} = P_i + 0.3 * V_f + -0.9 * V_r$

Figure 3.6: Template values for the used group formations [24]

Some final remarks about grouped agents is that they had a smaller preferred minimum distance towards the other members of the same group, namely values derived from Moussaïd et al. [18] empirical observations. Also, since all group members not being the leader in their respective group were bound by a different preferred velocity than individual agents, at a distance greater than 3 meters from respective group slot, they were not affected by the aforementioned back/forward tracing and smoothed turns options. It was instead left for the leader to perform the necessary calculations and adjustments when running simulations with these options - the other group members would simply have their individual steering adjusted accordingly.

Chapter 4

Evaluation

This chapter will describe how the made simulation will be evaluated and how the upcoming chapter, 'Results', will be structured. Furthermore, it will be explained how the research questions for this paper will be answered. In general, all simulations to make will be limited to situations of high density, or increasing density that converges to higher levels. However, one exception of this is the user study, defined at the end of this chapter.

Abbreviations used throughout the evaluation, results and discussions chapters follows:

Alpha - The alpha value defined in section 2.3.3, used as a tuning parameter for the minimum distance d_{min}

Cells - A number n indicating that a n x n staggered grid is used in the simulation

FPS - The number of rendered Frames Per Second

ms - Milliseconds

Min Dis - The preferred minimum distance that each agent generally has towards all other agents; inter-group specific distances were used as specified in Section 4.2.3

Neighbor Bins - The number of "bins" used for maintaining agents of proximate distances

Scene - The environment to simulate in (see below)

Solver(Iterations, Epsilon) - The LCP solver to be used in the simulation, along with its maximum allowed iterations and a precision index Epsilon (Lower value means having more accurate answers)

Velocity(m/s) - The set maximum velocity for all agents, in relation to preferred velocity and collision avoidance velocity

4.1 Scenarios

Various scenarios are set up in order to evaluate and measure the simulator in a number of ways. The environments used along with the content to be tested are specified here in broader words. Specific scenarios and their settings are then defined in the next chapter.

4.1.1 Environments

The following environments will be used for all evaluated scenarios and measurements. While simulation specific settings will be variable in these, the environment will be constant with regard to size, obstacles, global planning and other environmental related settings unless specified otherwise. For easier reference, all environments are labeled with a shorter abbreviation.

Environment	Description	Area Size (Units)	Abbreviation
Subway Passage	A long underground passageway for regular commuters, with multiple columns, entrances and exits serving as obstacles. Based on an observed passageway in Stockholm Central Station.	50x400	SUB
Park Passage	A bright park with two larger roads crossing each other in the middle. All off-road areas are set with invisible obstacles, restricting agent movement onto the roads. Agent movement are only set to occur on one of the roads here.	50x300	PARK
Park Passage (Two-Way)	Same as PARK but with agent movement alongside the second road as well.	100x300	PARK2
Office Complex	Corridors and rooms separated by wall obstructions make up this slice of an office complex. Not based on any known office; it was made for showcasing an environment with lots of turns and restricted boundaries.	220x220	OFFICE
Empty Plane Space	An abstract environment without neither a defined theme nor obstacles.	300x300	PLANE

Table 4.1: Environments for all scenarios

4.1.2 The Simulator's Properties

With regard to the made interpretation of the unilateral solution proposed by Narain et al. [20], and the expansion of certain components in order to gain plausibility, the overall properties of the simulation will be thoroughly analyzed and initially presented in the next chapter. This ranges from effects found in said author's paper to exposing the plausibility of various options. The testing will be done through running simulations that stimulate or have the focus on various components. Visual

and functional gains, or losses, through adding groups to various scenarios will be investigated as well. Set scenes and parameters might not reflect actual behavior found in real-life situations; all settings are solely used to highlight the behavior of the simulator. Specifications of the components-to-test will be given at the beginning of each section, which themselves are found in the initial part of the next chapter.

4.1.3 Performance

Simulating large, denser crowds has been a computational challenge for many novel approaches. As the made approach intends to tackle this hindrance, making measurements on the performance of the simulation is deemed appropriate. This will be done by stress-testing all of the simulation steps in various densities and environments. Additionally, since a primary goal of the made work is adding groups to the crowd, the performance of mixed groups compared to individuals will be measured in a variety of scenarios. In general, performance is defined as both the time it takes for an iteration through the simulation steps as well as the number of FPS for each scenario. Measurements of these will be made from the start of the simulation with only a few exceptions; some scenarios will only have measurements done once the simulation is deemed stabilized, that is, when the number of agents entering and leaving the simulation is unchanging.

4.1.4 Group Distributions

All simulations having groups will feature pre-defined group distributions. Note however that only some of these will be used in the user study, specified in the next section; some distributions will also only be used outside the study - see the "user study" and "general" labeling in the following table.

Distribution	Individuals (I)	Groups of 2 (P)	Groups of 3 (T)	Groups of 4 (Q)	User Study	General
1	0.48	0.43	0.08	0.01	Yes	Yes
2	1.0	0.0	0.0	0.0	Yes	Yes
3	0.1	0.3	0.3	0.3	Yes	No
4	0.25	0.25	0.25	0.25	Yes	No
5	0.0	0.7	0.2	0.1	No	Yes

Table 4.2: Group distributions to be used throughout the thesis. Note the abbreviations (I)ndividuals, (P)airs, (T)rios, (Q)uadruples of the distributions. All of the values are in decimal percentage form

4.2 User Study: Perceived Groups

In order to answer the research question of whether groups of various sizes can be perceived in crowds of higher density, one must realize that perception is a trait shared by actual living beings. Hence, an insight to the answer will be given through a smaller user study.

4.2.1 Overview

The overall purpose of the study is to gain empirical data in how groups of various sizes are perceived in different crowd densities. This will be investigated in two different environments, PARK as well as SUBWAY, and will be done through having simulations with group distributions of various group ratios in various densities. Some of these ratios are backed up by empirical data while some are deliberately made to expose the perception of groups compared to that of individuals. Moreover, the study will be conducted at the Royal Institute of Technology in Sweden, with its students of various nationalities as a target group.

4.2.2 Limitations

There are many factors that could be considered in a study, since they might directly or indirectly affect participants perception and judgment ability. The following specifies what will be considered variable and constant throughout the study to narrow down the overall context and possibly affecting factors.

Variables

- **Group Ratios** - The distribution of groups of 2-4 agents and individuals will be used in this study. Each group distribution will feature a certain ratio, and then be simulated, recorded and finally shown to the participant, who will give their perception of the ratio. Hence providing insights in how various groups are perceived.

- **Densities** - There will be group distributions of various group ratios, but also various densities of crowds specific for each simulated environment. Interpretations of high/medium/low-densities will be done through trial and error for each. In general, low is considered a very scarce crowd, whereas medium is a larger crowd without much "exterior" effect on an agent's movement, and finally a high density setting consist of a very large, packed crowd which affects the movement of every agent considerably. The density of a distribution will be generated through spawning agents continuously with a particular spawn rate. Spawning groups of agent in sizes 2-4 with each set rate will naturally yield more agents than only spawning individual ones, hence affecting the perception. As stated earlier, trial and error will determine balanced spawn rates for each density.

- **Environments** - All group distributions will be set in the two different environments, PARK and SUBWAY. These will have a different camera angle for recording the crowd, along with a lighting/contrast specific to the respective environment.

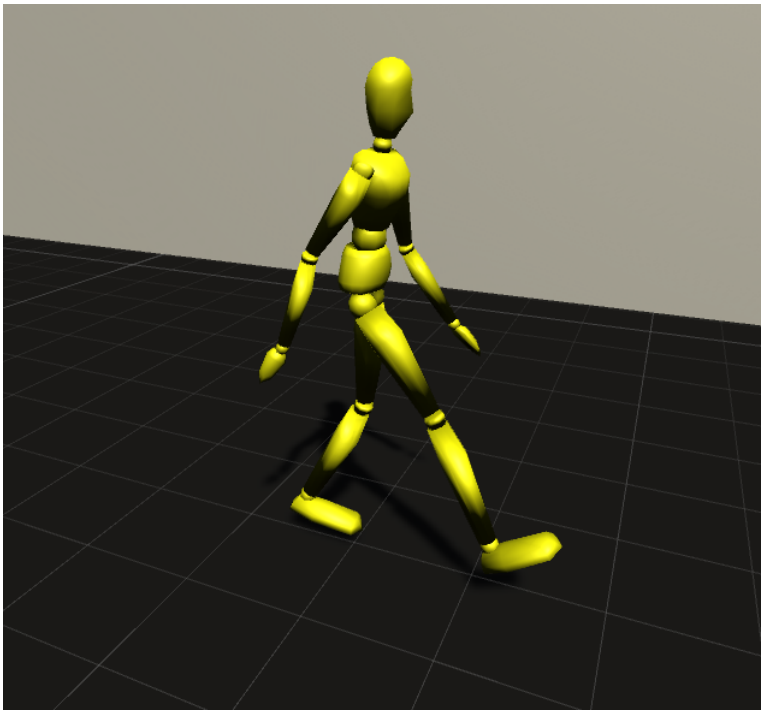


Figure 4.1: Mannequin agent model used in the study.

Constants

- **Appearances** - Following Carretero et al. [5] paper on crowd perceptions, throughout the study all agents will be rendered with a yellow mannequin model. This in order to eliminate possible perceptive variances between models (see figure 4.1).
- **Minimum Distance** - Crowd agents will maintain a fixed preferred minimum distance to individuals, but a fixed lower distance to members of the same group.
- **Velocities** - To simplify interpretations, all agents will have the same maximum velocity whereas the minimum velocity will be variable in accordance with the pressure correcting approach and collision avoidance velocities. An exception however is group agents, as aforementioned they will either increase or decrease their preferred velocity with a factor of 0.3 should they be at a distance 3 or greater from their respective group leader.
- **Scene Settings** - Since there will be two distinct scenes, they will differ with both global planning and other properties. Nevertheless, these will remain unchanged in-between each simulation. The only thing that might differ is the initial spawning position of each agent; the scenes will feature certain smaller, specific areas in which agents can randomly spawn. Nevertheless, the direction of these will remain the same as with all other agents within each area.
- **Group Distributions** A total of 4 unique group ratios will be used, along with the three mentioned densities: low, medium and high. Each simulation sequence will be 15 seconds long, and will be recorded when the simulation has

reached a stable state with regards to agent movement and density. Furthermore, the sequences will be run in the two mentioned environments, providing a total of 24 sequences (6 min). Moreover in the spirit of randomized block design, participants will be shown two "blocks" of sequences. Each of these will contain the 24 sequences in a shuffled order. The shuffling will be done using python's standard random library, and will be done for each new block.

4.2.3 Specifications

Preferred Minimum Distances

The following specifies the preferred minimum distances for each agent, which depend on whether they are in a group or not. Should they be in a group, they will have their preferred minimum distance changed for all other members of the same group. Given preferred distance to individuals are interpreted from Wei et al. [32], whereas group members have their distances averaged from empirical data collected by Moussaïd et al. [18].

Social Category	Preferred Minimum Distances
Individuals / Not in the same group	1.0
Group of 2	0.54
Group of 3	0.585
Group of 4	0.656

Table 4.3: Preferred minimum distances between individuals and group members

Spawn Ratios of the Environments

The following two images illustrate the roadmaps and spawn rates for each environment in the various densities. The spawn rates are derived from trial and error within the different densities. One smaller exception is of the SUBWAY environment, where the spawns and routes is an estimation of behavior observed at the corresponding subway intersection in Stockholm Central Station. Groups spawned in the environments will increase the overall number of agents as they spawn with the same ratios as individuals. This is a simplification that increases the perceived density of the simulation, but was deemed feasible given the derived ratios.

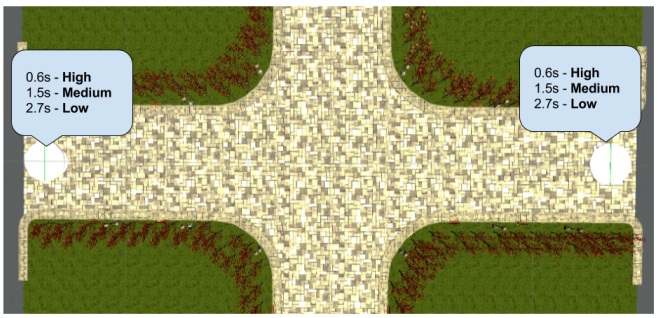


Figure 4.2: PARK scene spawn rates on two different spawn areas, with agents walking between these.

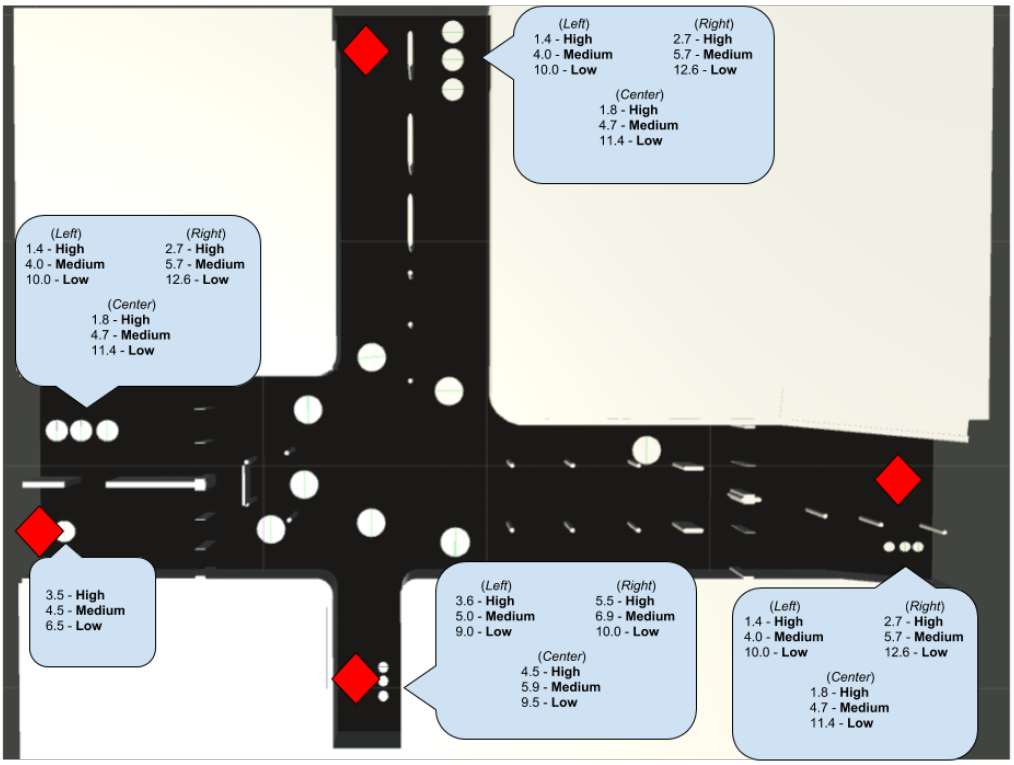


Figure 4.3: SUBWAY spawn rates, with 4 exits (red) and 5 spawn areas. Agents spawning at these proceed to the three furthest exits except for the bottom-left spawn, that move agents to the top exit.

Simulation Specific Settings

All sequences were run with the following values set to the simulator.

Scene	Grid Cells	Neighbour Bins	Velocity(m/s)	Alpha	Solver(Iterations, Epsilon)
PARK	25	180	1.8	1.0	PSOR(40, 0.001)
SUBWAY	30	240	2.5	1.0	PSOR(40, 0.001)

Table 4.4: Settings used for the environments in the user study

Group Distributions

The group distributions used in the study is specified in Table 4.2. The following summarizes the interpretation of these:

Distribution 1, "Natural Ratio" - Based on empirical data[18]

Distribution 2 - "Individuals Only" - With only individuals, can groups still be detected?

Distribution 3 - "Large Groups" - Having mostly groups, can the individuals be detected?

Distribution 4 - "Uniform" - Uniform distribution of the groupings.

As mentioned, each group distribution will be present in each environment, with one repetition in three different densities. This leads to a total of 2x2x3 = 12 stimulï per distribution. Having 15s/sequence and 4 group distributions gives a total of 4x15x12 seconds, or 12 minutes of total video which will yield perceptions on 48 stimuli.

4.2.4 Procedure

Given to the people participating in the study are the instructions to fill in their age and preferred sex, and then view a number of video sequences of crowds in two different environments. These are said to consists of individuals, and groups of 2-4. Their task will be to guess the perceived ratio of groups / individuals for each sequence. This will be done by having the students fill in the following questions after viewing a sequence (possible answers are provided, and answers for the first four questions will be translated to percentages):

Question / Statement	Answer
Number of perceived groups of one	500
Number of perceived groups of two	200
Number of perceived groups of three	100
Number of perceived groups of four	100
Degree to which the ratios seems lifelike (1-10)	5/10
Degree to which the movement seemed lifelike	3/10
Degree to which the groups seemed realistic	6/10

Table 4.5: Questions for the user study, with fictional answers

The study finishes up by noting any thoughts the participant had on the viewed material. Furthermore, they are given a more detailed explanation of the study but are never notified on the real distribution ratios.

4.2.5 Hypothesis

1. Groups and their various distributions will generally be easier to detect in low and medium densities. The guesses on the group distribution ratios will thus be somewhat accurate.

2. Groups will be harder to detect in high density crowds, leading to guesses of either incorrect uniform distributions or one particular category of groups due to the uncertainty.

3. The natural order in Distribution 1 will not be evident in the average guesses.
4. At medium to high densities, Distribution 2 will yield more perceptions on agents being in a group than by themselves.
5. Distribution 4 will yield the most lifelike guesses.
6. The groups will have an average realism of around 5/10; they will not appear to be quite lifelike due to the abreast walking formation.

4.2.6 Evaluation

Although being a smaller one, the user study will hopefully yield some form of answer of how groups are perceived. Given guesses will be averaged and then presented in graphs, showing the perceived groupings compared to the actual values for each distribution. Any unforeseen or interesting remarks will be mentioned in the following discussion.

Chapter 5

Results

In this chapter results from the crowd simulations are presented. Initially, in this chapter the effects of various components of the simulator are tested by defining scenarios for these along with parameter specifications and presentation of results. Following each of them is a smaller discussion of interesting findings and overall use-ability. The second part of the chapter then in a similar manner presents and discusses made measurements. Finally, results from the made user study are shown and analyzed.

5.1 Global Planning

5.1.1 Scenarios

The effects of sampled versus custom placed nodes are investigated here in the OFFICE environment. Two sorts of scenarios will be considered, where the first one is agents walking in and out of the complex and the second is an evacuation of the entire floor. Each scenario will be run with and without sampled nodes, having all the path planning calculated beforehand; time spent on pre-processing will be ignored.

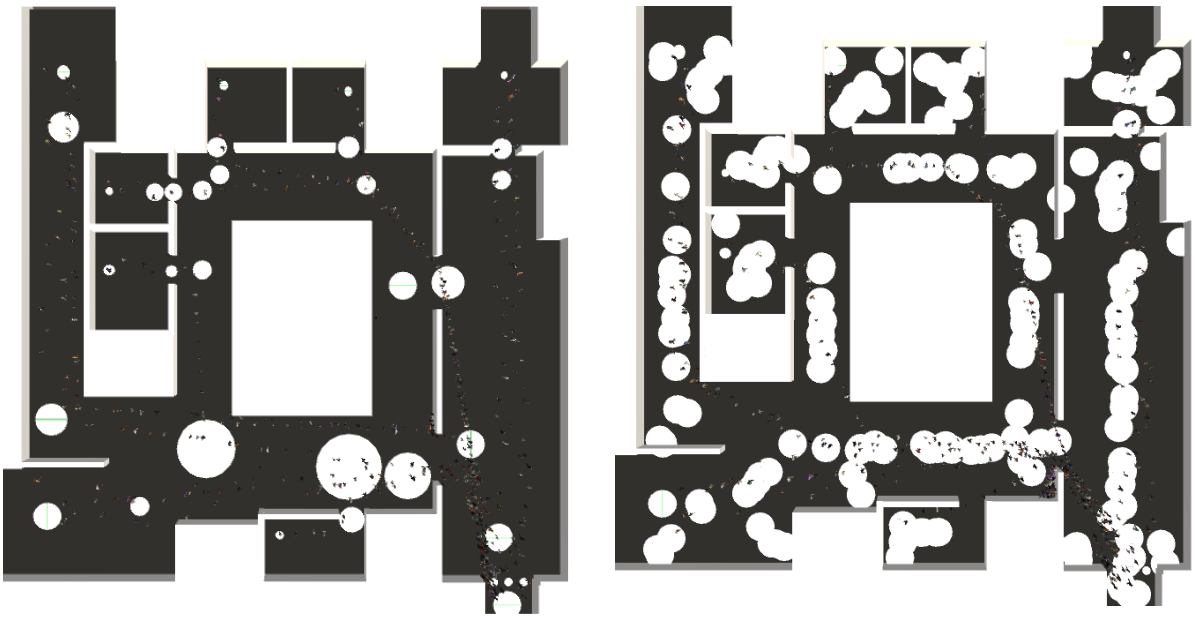
5.1.2 Parameters

Both scenarios featured in OFFICE will have the same custom placed nodes as seen in figure 5.1. When sampling nodes, 200 uniformly distributed nodes will be sampled across the environment's (approximate) medial axis. The first scenario will feature continuously spawning individual agents whereas in the second scenario, 2500 individual agents are set to be evacuating from the OFFICE environment.

Scene	Cells	Neighbor Bins	Velocity (m/s)	Alpha	Solver(Iterations, Epsilon)	Min Dis
OFFICE	40	150	2.5	1	PSOR(40, 0.001)	0.8

Table 5.1: Scenario settings for testing the effect of sampled versus custom placed nodes

5.1.3 Simulation



(a) Custom Placed Nodes

(b) Sampled Nodes

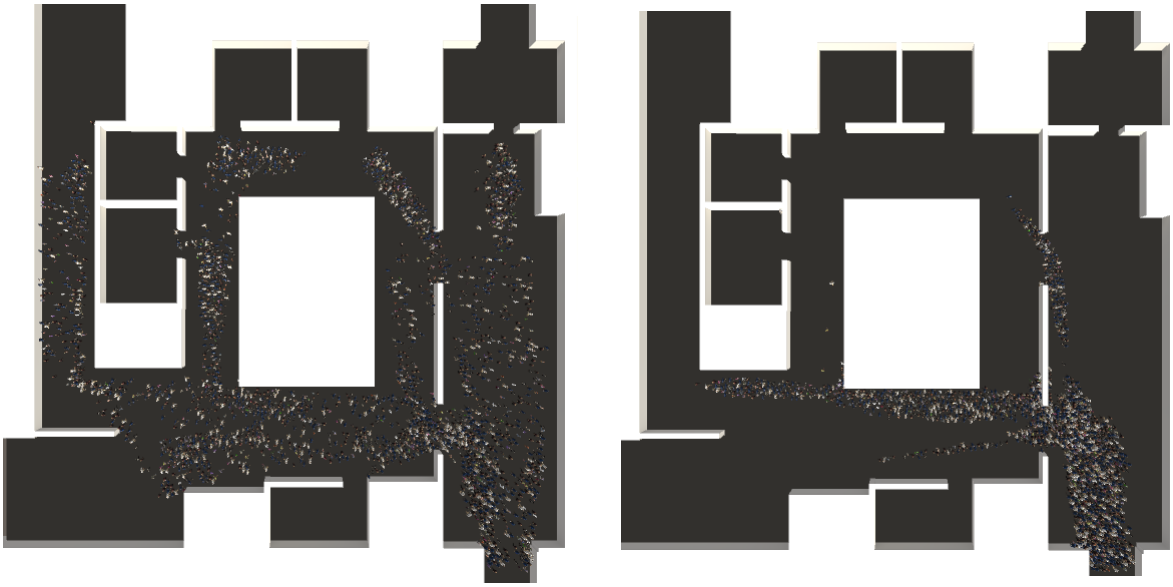
Figure 5.1: Various nodes settings in the OFFICE scenario with agents walking between the nodes.

Running the first scenario with agents walking in and out of the scene was done smoothly without agents making unexpected collisions or being trapped in any local minima. As seen in figure 5.1, orderly lines of agents are formed in-between two nodes in their paths. Occasional collision avoidance caused agents to spread out a small amount. More diverse paths are seen to the left and right of the environment due to the use of the special nodes mentioned in figure 3.3.

Sampling the nodes in the first scenario does not yield overall different walking patterns. The left and right parts of the environment still maintain some spreading of walking agents, while the center section shows more aligned movements. Some paths do however cause agents to walk very close to obstacles, causing smaller collisions when combined with the seen diversity. Furthermore, this is seen causing agents to enter local minima states when the density is high, as seen in figure 5.2.

Figure 5.2: Agents trapped in a local minima.

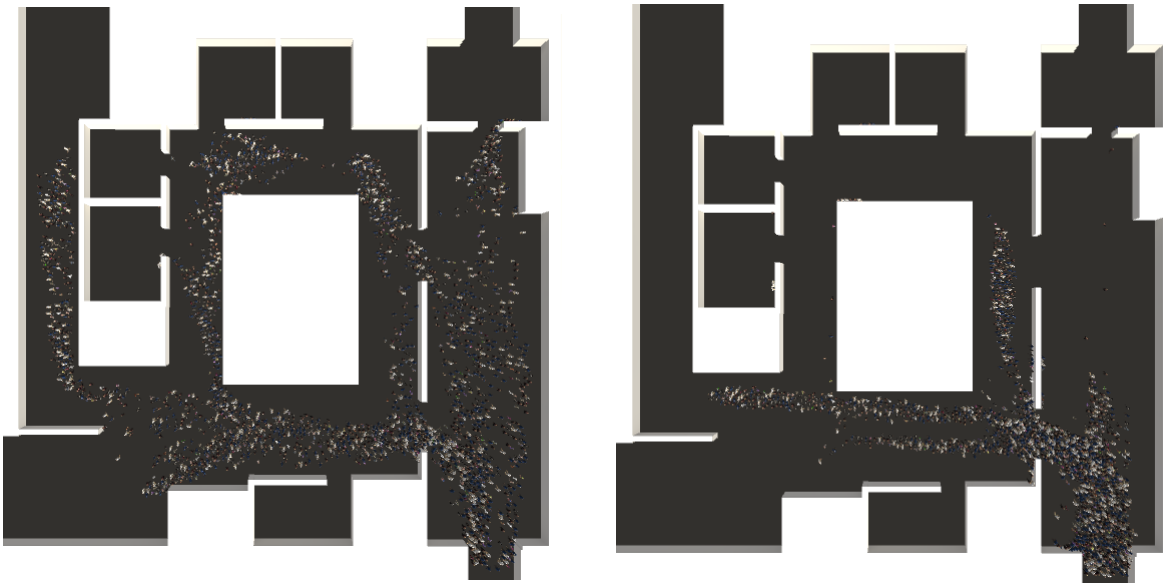
The second scenario showed similar results as before. Evacuating 2500 agents with placed nodes is done without anyone left behind while agents are left behind in various local minima traps using sampled nodes. A somewhat slower evacuation is observed with the latter.



(a) 15 seconds

(b) 50 seconds

Figure 5.3: Custom placed nodes used in an evacuation scenario.



(a) 15 seconds (b) 50 seconds

Figure 5.4: Sampled nodes in the evacuation scenario.

5.1.4 Outcome

Despite the property of sampling nodes relatively far away from obstacles, sampling on the medial axis yielded a worse global planner with local minima states. This is most evident in the evacuation scenario, where the density of agents is really high and causes some agents to be "pushed" away from their intended paths. Using custom placed nodes, not lightly placed but with great consideration, allowed for trial and error with careful placements at more critical zones. For example, in figure 5.1, one can see that nodes being in the risk zone of local minima states are purposely enlarged to have agents make sharper turns more quickly; this might not be the case with the sampled nodes, since all nodes were of the same size. Moreover, the placed nodes featured some special nodes that allow for more diverse paths, and this alone could have been the cause. Hence the possibility of having restricted types and sizes of nodes when sampling could be a cause for the ineffective paths.

Another good reason for the failing MAPRM sampling along with the seen, unrealistic "linings" of people is that the path construction done by the Floyd-Warshall algorithm is limited. The algorithm effectively generates paths that minimize the distance from node A to node B. Thus it does not matter whether smart sampling on the medial axis is done. Even when a large number of nodes are sampled, the path construction will not consider anything but the nodes positions in world space, and the most effective routes will be given.

Judging from what has been seen in these two scenarios, and their respective behaviors with the different roadmaps, it would seem that the somewhat tedious task of placing nodes generates superior, more controlled movement. Nevertheless, it is believed that MAPRM would generate even better and more effective paths if the path construction would consider paths along the medial axis even though these might be longer. Finally, to avoid the "lining" behavior observed in all scenarios, one could consider having a path constructor that makes several, slightly spread out,

paths from a node. Another possibility more feasible in higher densities would be placing the special nodes in figure 3.3, as agents would then "push" each other into more diverse paths. The risk of local minima traps does however remain, and is the price to be paid with pre-computed, constant time global planning.

5.2 Density Influence

5.2.1 Scenarios

Earlier work with the UIC model have shown that as the density of a crowd gets larger, patterns such as lanes and vortices are formed automatically in order to solve movement constrained by collision avoidance. In order to discover how such behavior are present in the made implementation, three scenarios involving PARK, PARK2 and PLANE will be tested with various density-altering settings. Two additional scenarios will also be run, but are distinguished from the other three as these are run with more extreme settings; one will feature extremely high density while the other will feature no density setting at all. The purpose of these two are to underscore both the need, and flaws of using densities for a macro collision avoidance step.

The PARK and PARK2 scenarios will be run with continuously spawning agents of low, medium and high density crowds. Furthermore, these will be run with and without group agents. Very differently, the PLANE scenario will be run with the special setting of spawning individual agents in a disc, and then have each walk to their diagonally opposite point in the scene. This scenario has been seen to expose various properties in previous approaches, and is thus assumed to simulate the density influencing properties more clearly. Approximately 2000 agents will be spawned within several layers of circles, effectively forming a disc (see figure 5.5).

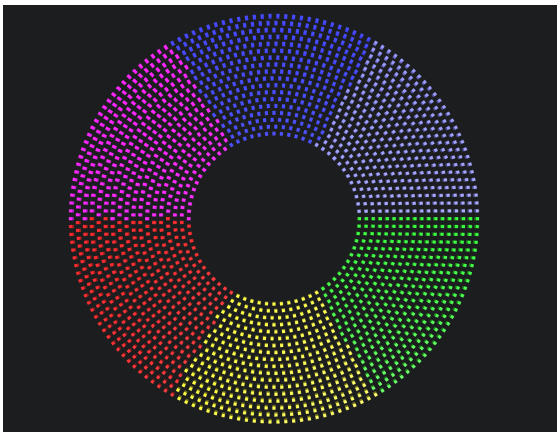


Figure 5.5: Agents in a disc, moving to their diagonally opposite point.

In order to get a clearer view of the seen patterns, all scenarios here have their environments simplified with less textures, along with an abstract agent model that is uniformly colored based on their initial spawning location (PARK, PARK2) or initial direction (PLANE).

5.2.2 Parameters

Scene	Low Density	Medium Density	High Density
PARK	190	500	1300
PARK2	800	1300	3300

Table 5.2: The approximate number of agents present at various densities in PARK and PARK2 (Spawn rates will be adjusted to uphold these numbers when using groups).

Scene	Cells	Neighbor Bins	Velocity (m/s)	Alpha	Solver(Iterations, Epsilon)	Min Dis
PARK	50	180	2.2	1.0/0.4	PSOR(40, 0.000001)	1.3
PARK2	50	200	2.5	1.0/0.4	PSOR(40, 0.000001)	1.4
PLANE	40	40	1.5	0.8/0.5/0.3	PSOR(40, 0.000001)	1.1

Table 5.3: Scenario settings

Scene	Cells	Neighbor Bins	Velocity (m/s)	Alpha	Solver(Iterations, Epsilon)	Min Dis
PARK2	50	50	1.3	0.2	PSOR(40, 0.000001)	1.1
PARK2	-	50	1.3	-	-	1.1

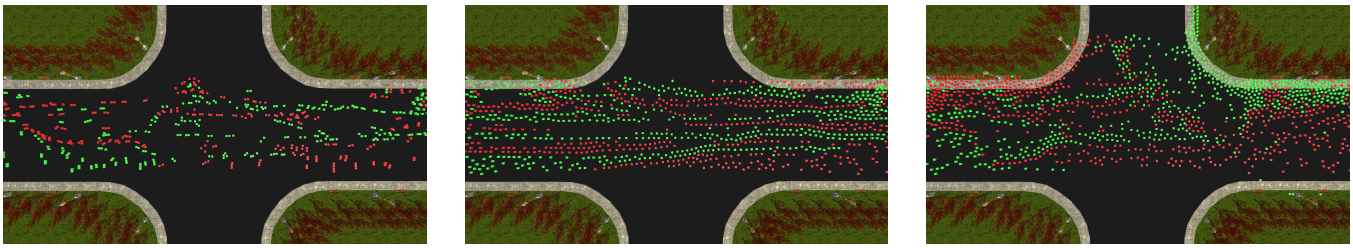
Table 5.4: Extreme scenario settings

5.2.3 Simulation

PARK

When running the PARK scene with lower densities one does not perceive any unwanted behavior. Individual agents walk freely, while occasional "bumps" onto grouped agents of sizes 2-3 cause them to switch from an abreast to a lined river formation. Some groups of size 4 are seen to split up into two closer groups when encountering intersecting agents, but not merge together into an abreast formation afterwards. Changing the alpha to 0.4, hence decreasing the maximum allowed density constant d_{min}, cause no apparent change.

More interesting behaviors are however seen at medium to high densities where agents are rapidly forming lines of agents walking in opposite directions, as illustrated in (a) and (b) in figure 5.6. Groups in these scenarios tend to cause less sharp lines of agents, with occasional clusters digging through the set lines as in to maintain contact with each respective group. At the highest used density, with alpha set to 0.4, the system becomes gradually slower due to agents blocking the path and causing dispersion. This is illustrated in sub-figure (c) along with an occasional "wave" of agents that cause them to rise up towards free space in the middle of the path; this does not last long as it converge to more lined behavior.



(a) Forming lines in medium density (Group setting = 3, Alpha = 0.4)

(b) Converged lines in high density (Group setting = 1, Alpha = 1.0)

(c) High density causes dispersions (Group setting = 1, Alpha = 0.4)

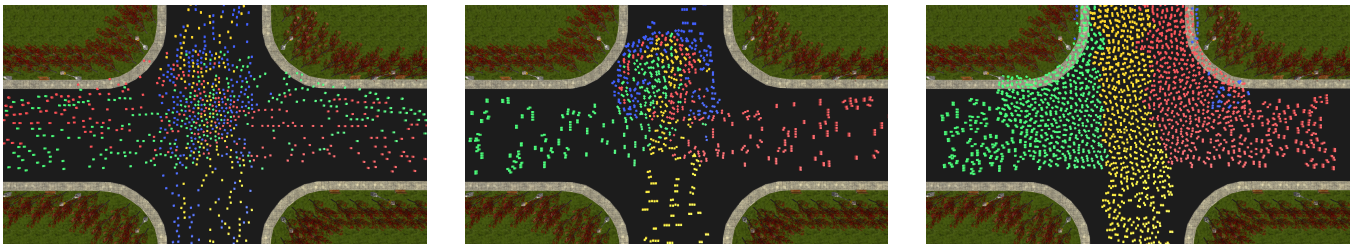
Figure 5.6: Agents crossing a road at various density settings. The group settings mentioned here refers to the distributions in table 4.2

PARK2

The PARK2 scenarios introduce many more agents with some of them walking along both roads in the scene. These cause immediate behavior in low densities with thinner lines of agents that perform small curves in their moving across the intersection at the middle of the scene. As seen in the PARK scene, having groups in the crowd cause disruptions of the lines with smaller clusters. Changing alpha to 0.4 enforces these behaviors slightly.

Medium density in this setting, with roughly 1300 agents moving in the scene, initially causes a smaller turbulence at the intersection that later calms down to lined behavior once again. This is not apparent with groups, which adapt more occasional curves in their movement which sometimes is perceived as walking backwards. With alpha set to 0.4 here, agents from the horizontal road suddenly emerge to a sudden movement upwards upon facing impact at the intersection. Effectively, this causes the agents walking down the vertical road to split up into two and slowly walk along the sides as seen in (b) of figure 5.7. This behavior remains with groups in the system.

More intense behaviors are seen at high density settings. Having alpha set to 1.0, a "normal" maximum density, causes some lining behavior with massive bursts of the "rising" behavior seen before. Turbulence is seen in a majority of the environment in group settings, with increasing number of agents due to the road essentially being blocked by agents trying to determine their path. This is taken to the extreme at the 0.4 setting with very clear vortex's present that shake and "blend" agents upwards as in (c) of figure 5.7. As this grew worse, the simulation eventually could not handle the stress anymore, illustrated by a huge explosion of agents, scattering them well out of bounds for the PARK2 scene.



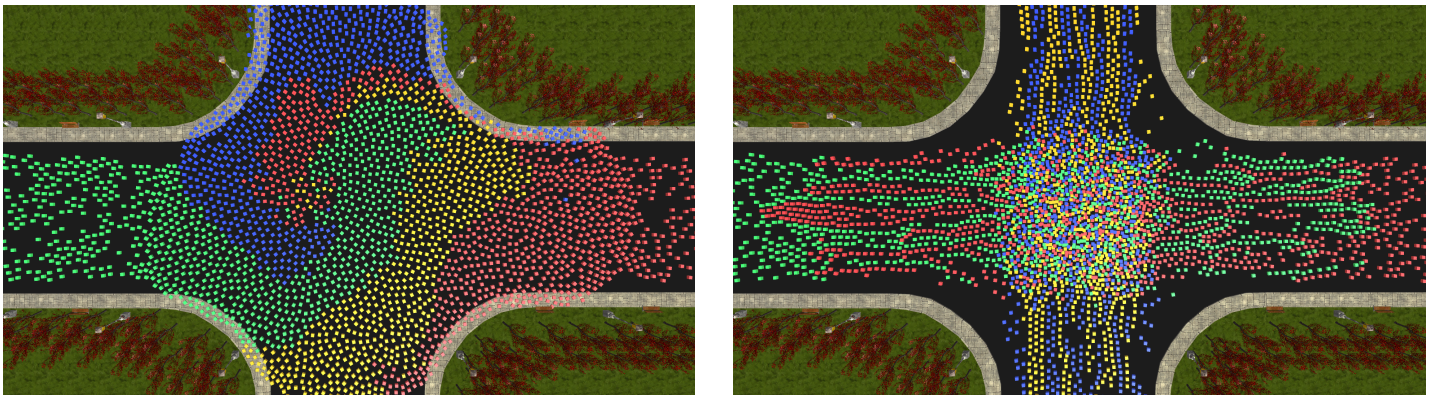
(a) Turbulence in medium density (Group setting = 1, Alpha = 1.0)

(b) Medium density with impacts (Group setting = 3, Alpha = 0.4)

(c) High density causes avoidance (Group setting = 3, Alpha = 0.4)

Figure 5.7: Agents crossing two roads at various density settings.

The two final scenarios in the PARK2 scene, the simulation with initially extreme density and the one with the density setting turned off completely, are shown in figure 5.8. The former causes a large vortex to be formed that blends the agents from all 4 directions, that after a while of convergence manages to allow a larger amount of agents to pass through to their goals. With only very straight lines, a lot of turbulence at the intersection and clear violations of the MDE enforcement, the latter allows all agents to walk through the scene and arrive at the goal.



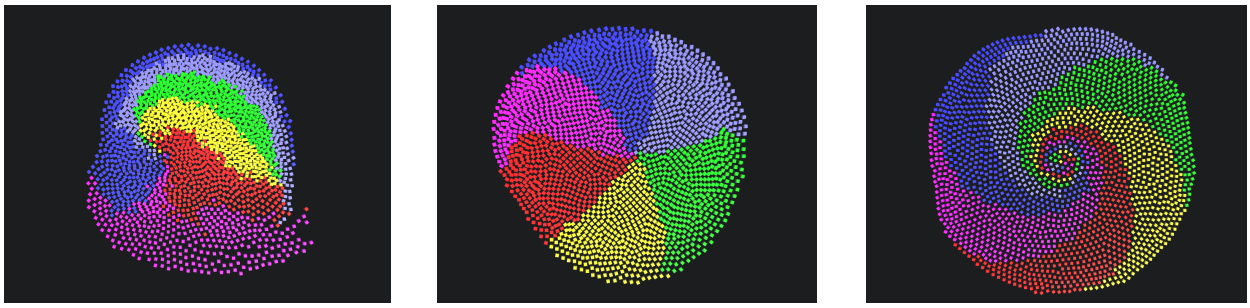
(a) Extremely high density

(b) No continuum velocity or pressure

Figure 5.8: Two extreme cases highlighting good and bad properties.

PLANE

Further simulation in the PLANE scene shows similar behaviors as seen in the higher densities of PARK2. With alpha set to 0.8 in figure 5.9, some agents tend to gather and walk in one direction while other are divided up into two camps moving in the opposite. A lower setting causes the agents to move uniformly as a large cluster, without resolving any movement towards respective goal. With the lowest setting, a large vortex is formed that blends the agents and then effectively solves the system allowing each agent to reach their goal.



(a) 0.8 (b) 0.5 (c) 0.3

Figure 5.9: The disc scenario with various alpha settings.

5.2.4 Outcome

Overall analysis

In general, it can be seen that as the density of the made simulation increases, so does the behavior of agents correcting their velocities to avoid other agents. In the defined low to medium cases, one sees lines of people making quite smooth and small curves from all 4 directions, effectively causing a smaller vortex that immediately corrects the agents onto their set path without causing blocked behavior or turbulence. Similar behaviors has been seen in previous studies and in some real-life situations, with the lining behavior especially observed in various real-life densities. At the defined high density settings, this behavior is eliminated as the movement turns bad with the system trying to resolve larger collision avoidance. With an increased continuum velocity that definitely surpasses each agent's preferred velocity, one ends up having no sensible movement.

With respect to that the continuum behavior integrated into the simulator has ties to a fluid simulation, the behavior seen at these higher densities could be rooted with some similar properties. The "rising" behavior described before could be one of such, effectively emulating an active fluid that pushes its center towards one end while other forces cause the sides to move to other - a similarity would be a drop of water at the moment of impact on the ground. If not anything, the vortex's seen in both PARK2 and PLANE scenes are very akin to some form of fluid whirling around. Nevertheless, this does not have to bode good as seen in the scenario of (a) in figure 5.8.

In another turn of thoughts, one might wonder why the continuum correction is necessary at all since it was observed that lines were formed in the system, effectively running smoothly at lower densities. Indeed, when having a high density (alpha set to 1.0) in the PARK setting it was observed that lines were formed naturally, only having the continuum velocity disturb these for the worse. This need is however clearly seen in the extreme case of turning off the density at PARK2. Agents violating the MDE, walking very effectively but yet paying the price of massive turbulence with contact of other agents call for a need of the macro collision avoidance. The visual plausibility and realism is simply not achieved as well with agents (almost) moving through each other to achieve only a mechanical movement.

Groups

Adding the element of groups to these scenarios has an evident effect. At lower velocities one can see how the groups' properties of changing formations is done collectively with intersecting agents. In addition to having groups of 3-4 split into two subsets, often ending up in a river formation rather than abreast, groups tend to hold together consistently. Groups splitting up into two does however tend to stay that way, even though the behavior of groups rejoining after facing obstacles that causes the split has been observed in real life observations [25]. The reason for this is that the agents split apart due to the collision avoidance velocity, and not due to programmed group behavior. Indeed, the groups will adjust their preferred velocity towards their respective group slot, which always remain in abreast formation unless in the proximity of obstacles. Thus the reason for the agents not moving together as a group cannot be anything else than the 3 given units of "proximity" to respective slot is too long for upholding the sought formation. A quick change of this value to 1.5 units does however yield unwanted turbulent effects when agents are pushed away due to intersecting agents; the group agents perform rapid switches from moving according to its collision avoidance velocity and its preferred velocity, which in these situations makes the agents' overall velocity to shift directions multiple times. Having the value set to 3 has been seen to yield more plausible, smooth transitions although at the cost of more river formations at a worst case.

The effects on groups and formations are less evident in the defined medium density, but identifiable as of yet. Less river formations are seen, and groups take up more place in e.g. lines since the group agents now have people maintaining a minimum distance not only from the sides but from the back as well. A common pattern to see is a UV-formation that is continuously interchanging, though smoothly, with abreast formations. While this is more true in the PARK formation and its densities, they act no differently to the same discussed behavior of higher densities in PARK2. In fact, having groups in the environment with 3000+ agents seems to propagate the higher density effects.

Summary

In summary, having various density settings can cause good or extremely bad, independently from having groups or not. Very interesting properties are exposed at higher densities, which nevertheless are unwanted in a simulation of crowds. It is thus concluded after various experiments that a balance between affecting properties, namely agent velocity, preferred minimum distance, alpha values and (though less mentioned) the number of staggered grid cells in adaption to an environment and its number of agents are needed to get plausible simulations. Some good examples and plausible-causing properties have already been seen, although in scenarios not backed up by any real environment. A good corpora of high density crowds is believed to help with this with real trajectories exposing real behavior, group and agent ratios, all in a real environment.

5.3 LCP Solvers

5.3.1 Scenarios

Three different LCP solvers have been developed for this thesis. With the assumption that each solves the equation $z = Ax + b$ for unknown pressure values x equally and sufficiently accurate, these will be measured for performance. This is achieved by simulating up to 1400-1500 agents in the SUB environment, where measurements of the LCP solvers on each simulation step are made and combined. Measurements are done in ms per solver pair, and the number of required iterations.

5.3.2 Parameters

Scene	Cells	Neighbor Bins	Velocity (m/s)	Alpha	Solver(Iterations, Epsilon)	Min Dis
SUBWAY	60/55/50/45	240	1.8	1.0	variable(100, 0.000001)	1.0

Table 5.5: Scenario settings

5.3.3 Simulation

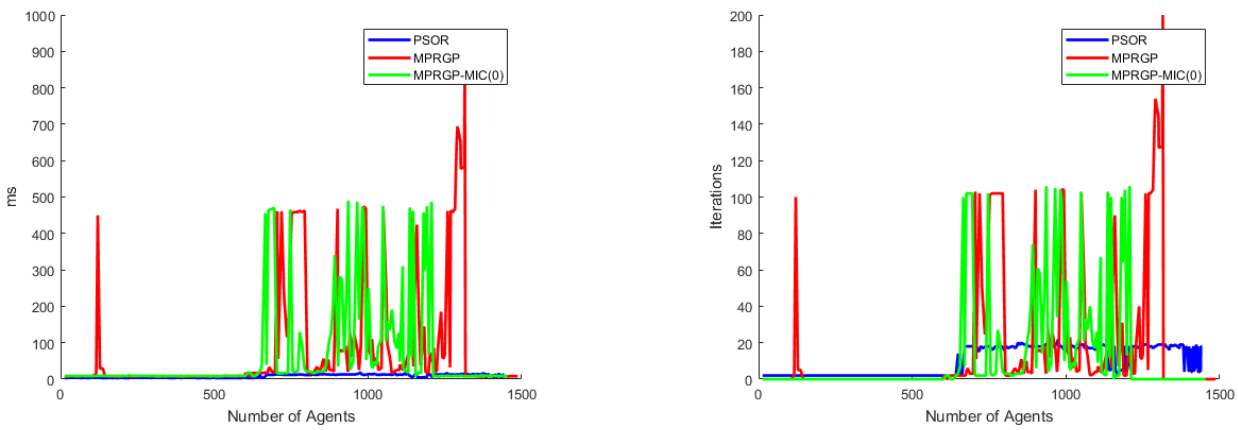


Figure 5.10: Results from having 60 cells

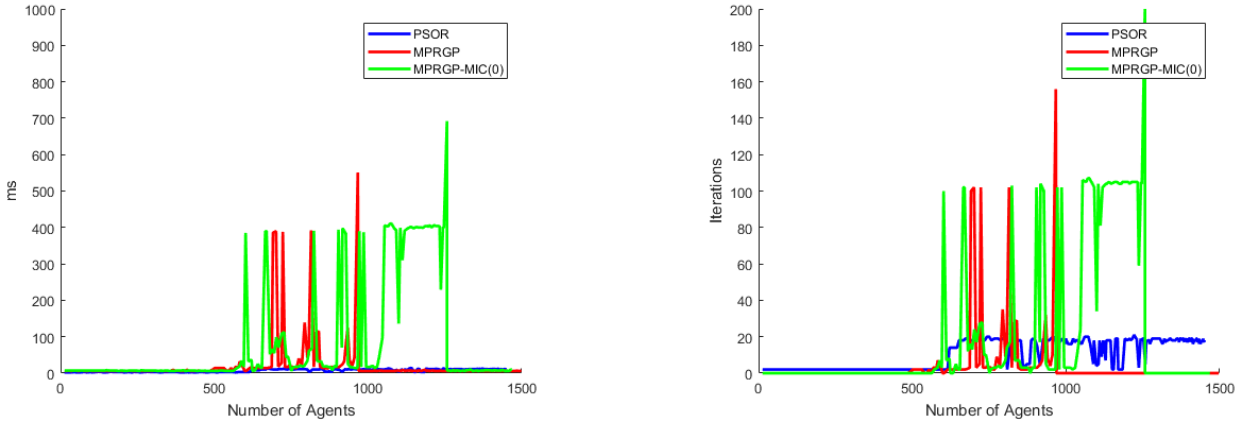


Figure 5.11: Results from having 55 cells

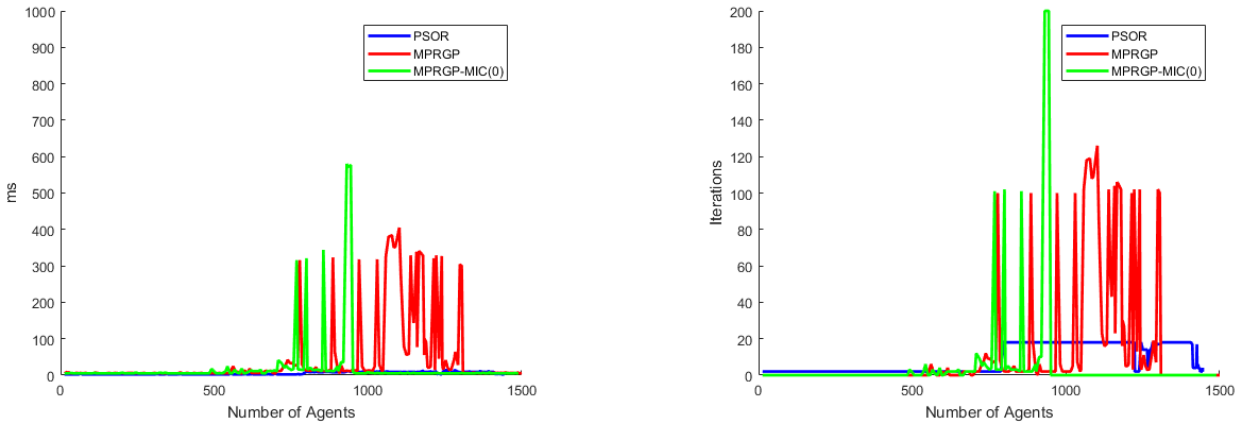


Figure 5.12: Results from having 50 cells

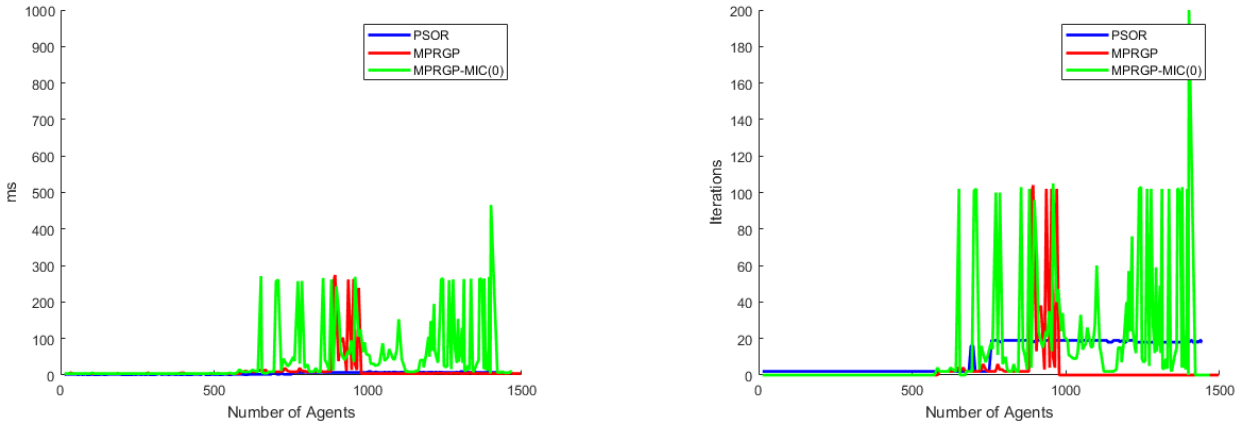


Figure 5.13: Results from having 45 cells

5.3.4 Outcome

Overall analysis

There seems to be a very strong canditate when it comes to solving the defined LCP in both a low and stable manner. PSOR iterates with a seemingly worst-case value of 20 iterations for the SUBWAY environment, barely having any impact on the solution time of around 10 ms. This seems to hold for the descending order of sampled cell sizes.

On the other hand, both MPRGP and the preconditioned MPRGP-MIC(0) perform with great fluctuation in all cases, with either no iterations/significant ms or very high spikes. The sudden, and sometimes holding, spikes of 300-500 ms and 80-100 iterations (or more) slow down the simulation considerably as the number of agent grows. A theory on to why this takes such time is that for starters, both algorithms along with PSOR have the same time complexity of $O(mn^2)$ = $O(n^2)$ for m iterations and n cells in total on the staggered grid. Nevertheless, the MPRGP and its variant both have a lot more calculations bound with this time complexity, making it generally slower in each iteration. Furthermore, the rate of convergance must be the other (main) reason for the worser performance; MPRGP simply requires more iterations to find a feasible solution. The fluctuation behavior per say could expose the behavior of Equation (2.6), namely that there is a correcting pressure whenever the density of the crowd has reached past the maximum d_{max} and none whenever the density satisfies this upper bound. This behavior seems to remain the same in decreasing the number of cells although with somewhat better performance in both iterations and ms. The same behavior can however not be said for PSOR, that provides less fluctuate solutions; it seems to be correcting some density all the time, which is a very interesting behavior nevertheless outside the bounds of this thesis to evaluate.

Further interpretation about whether MPRGP or MPRGP-MIC(0) are the better is hard to make as the two perform differently in various cell sizes. Figures 5.10, 5.11 and 5.13 show that MPRGP is the faster while figure 5.14 suggests the contrary. It can nevertheless be said that PSOR is significantly faster. Although one important question to ask oneself is: Is it as accurate? This is assumed since PSOR satisfies Equations (2.11)-(2.14), as it does not exceed its set maximum iterations of 100, or makes less than 10^{-5} small changes in-between iterations. The latter property does however allow for inexact solutions. In spite of this, to further back-up this theory it has been observed that using PSOR for all previous results presented in this chapter has yielded stable simulations that expose properties and density effects just as good as the MPRGP alternatives. In fact, except for being slower, it was observed that if the set maximum number of iterations was too low such that the MPRGP solvers hit the maximum number of iterations, there would sometimes be huge waves of pressure amongst the agents where greater turmoil and implausible explosive effects took place.

Comparisons and Conclusions

The performance of MPRGP-MIC(0) contradicts the performance in Narain et al. [20], which managed less than 40 ms with 40 cells per grid row whereas the findings

here suggest 400 ms for 45 cells per grid row. Any explanation in as to why there is such a huge difference between the two would be hard, as it is not known how the authors truly implemented their algorithm. Some immediate points to be considered are however that both this thesis and theirs had similar hardware, whereas this project was developed in the Unity game engine. This engine could have an affect on the performance, as it is only internally multi-threaded and can thus not fully utilize parallelism on multi-cores with custom-made code; the Unity API is not compatible with custom threads. If the original authors had used an engine of their own, which might be optimized for crowd simulation, that could also had a significant effect on the performance. There is also the possibility that they implemented MPRGP-MIC(0) correctly, whereas the made solution here is flawed, but unless proven otherwise, these are all just speculations.

5.4 Rendering Models

5.4.1 Scenarios

It has been observed that the rendering of agents had a serious impact on overall simulation performance. Thus three different models with meshes of various triangle counts will be compared for overall performance in FPS. This will be done in the OFFICE environment, featuring 3000 agents that will evacuate the scene. To further illustrate how much the performance is affected by plausibility, two of the models will have their animations turned on and off in seperate scenarios.

5.4.2 Parameters

The simulation will have the "Skip Node" option turned on here, allowing agents to faster reach the goal rather than being "pulled along" with the crowd and then try to go back in its tracks to check off previous, unreached, nodes.

Scene	Cells	Neighbor Bins	Velocity (m/s)	Alpha	Solver(Iterations, Epsilon)	Min Dis
OFFICE	40	150	2.5	1	PSOR(40, 0.001)	0.8

Table 5.6: Scenario settings

(a) Mannqeuin: 3400 triangles (b) Standard: 550-650 triangles (c) Cubes: 12 triangles

Figure 5.14: Used agent model's and their mesh's triangle count

5.4.3 Simulation

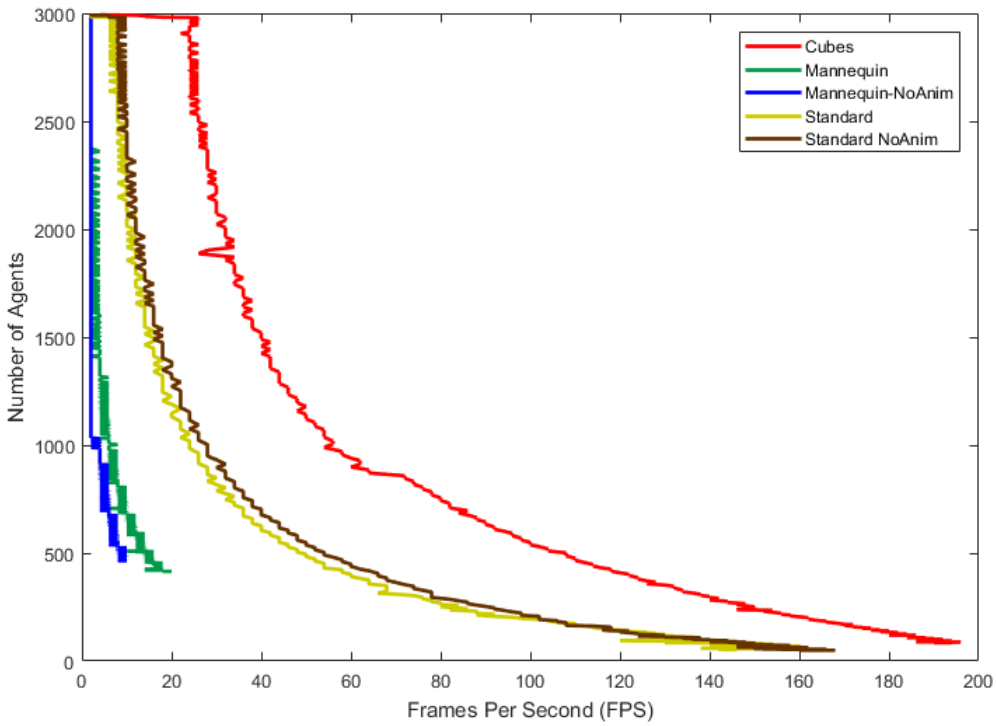


Figure 5.15: The overall performance with different models and animation turned on or off (NoAnim)

5.4.4 Outcome

Judging by the seen overall performance, the choice of agent model does have a significant impact. Moreover, having animations turned on or off yield a smaller improvement on the performance. This is however not the case for using the mannequin model with 3400 triangles per agents. With $3400 * 3000 = 10.2$ million triangles to render at any time in a worst case scenario, where optimizations can be made with occlusion culling, the simulation becomes incredibly slow even as the number of agents are dropping. Moreover, a very peculiar behavior was observed at 2 or lower FPS: Agents in large clusters would suddenly walk backwards and then either "bounce" off the nearest static obstacle, or be rendered standing still. This almost always resulted in agents becoming stuck in a local minima, if they were moving forward again at all. The behavior cannot be truly explained, as it did not occur with using any other "lighter" agent model. It would however seem that once the simulation is below 2 FPS, there are a lot of undefined behavior, possibly caused by the underlying game engine itself; when disabling the collision handling feature with obstacles, the agents would seem to move "properly" again, but nevertheless through obstacles. This maybe once again reinforce the need to overlook the used game engine, or develop less dependence, for performance.

Rendering simple cubes with 6 sides and 12 triangles each seems to yield the

best overall performance, and thus allows for having more agents in real-time simulations. This does however contradict with Narain et al. [20] solution that render much more agents with the same, or better, FPS. Investigations to as why this might be has once again lead to a flaw of using Unity as a game engine. It would seem that rendered agents are not rendered entirely on the computers GPU, which is extremely powerful precisely for this task, but on the CPU. (For the technical reader, SkinnedMeshRenderers were used for all agents except for the Cubes). Writing shader scripts or/and utilizing other rendering optimizations that moves more workload to the GPU is sure to have a significant effect on the rendering performance but maybe also on the overall performance as the CPU has less stress from rendering.

5.5 Simulation Steps

The simulator's sub-steps will be measured here, by making measurements per time step, in three different high density scenarios, with and without groups. Left out of these measurements are the built-in collision handling system of the used game engine, as well as the rendering of agents; these measurements are strictly on the main simulation sequence.

The simulation will be featured in 2 different environments and 4 different groupings, as specified below. For the OFFICE scenarios, agents will be evacuating out of the scene and thus groups will not be present in these. Spawn rates in the SUBWAY scenarios will be modified for the different groupings in as to yield the same maximum number of agents in all settings.

5.5.1 Parameters

The scenarios will feature the same group distributions as those found in table 4.2.

Scene	Cells	Neighbor Bins	Velocity (m/s)	Alpha	Solver(Iterations, Epsilon)	Min Dis
OFFICE	40	150	2.5	1.0	PSOR(40, 0.0001)	0.8
SUBWAY	70	240	1.8	1.0	PSOR(40, 0.0001)	0.8

Table 5.7: Scenario settings

5.5.2 Simulation

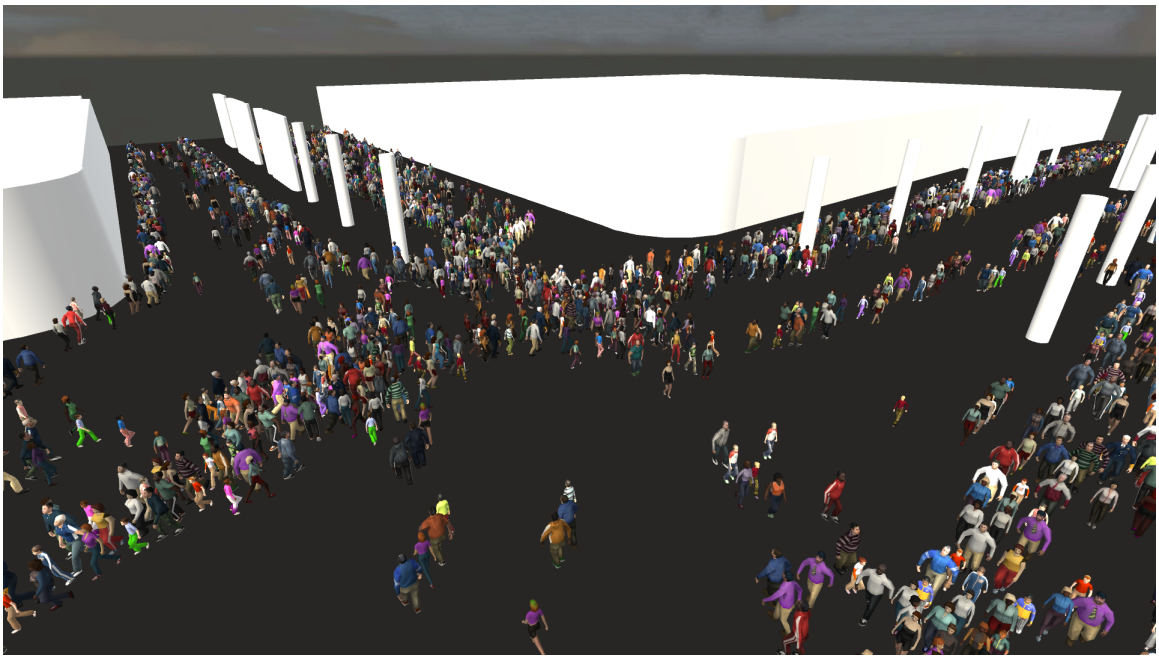


Figure 5.16: 3000 agents in the SUBWAY scenario

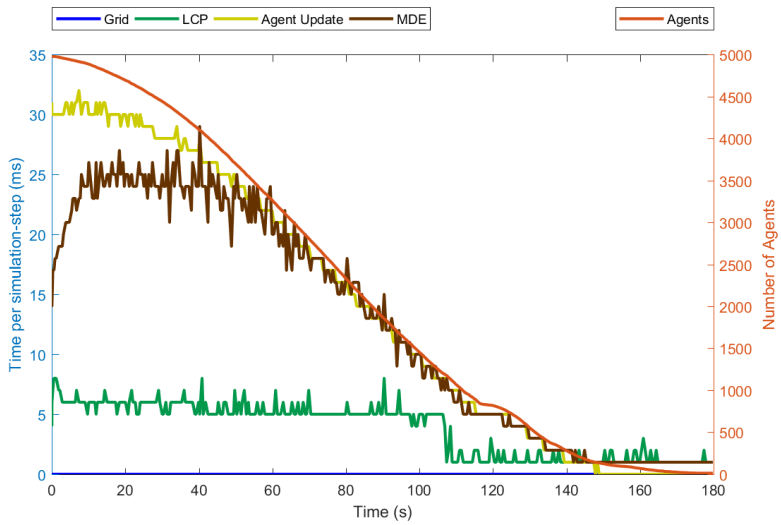


Figure 5.17: Performance with 5000 individual agents evacuating the OFFICE environment

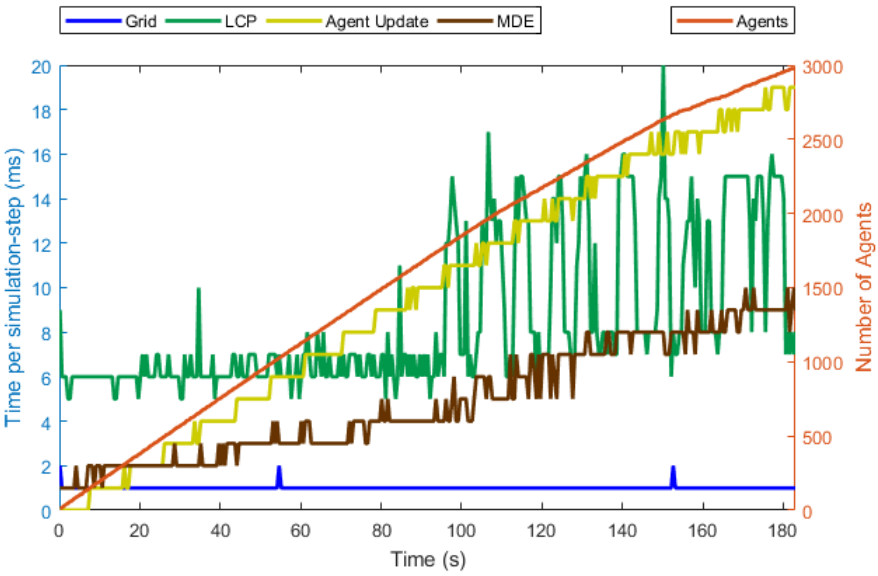


Figure 5.18: Performance with gradually increasing individual agents in the SUBWAY scenario

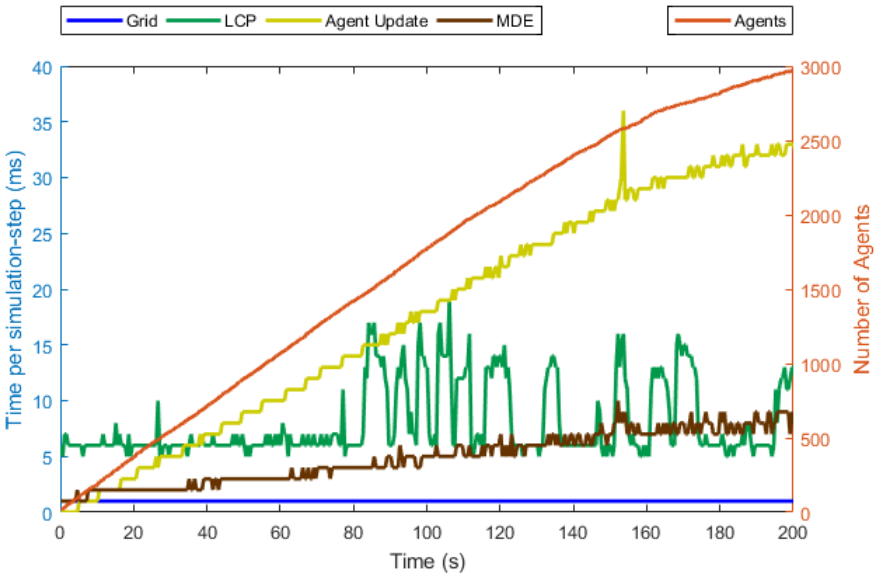


Figure 5.19: Performance with gradually increasing individual and grouped agents in the SUBWAY scenario

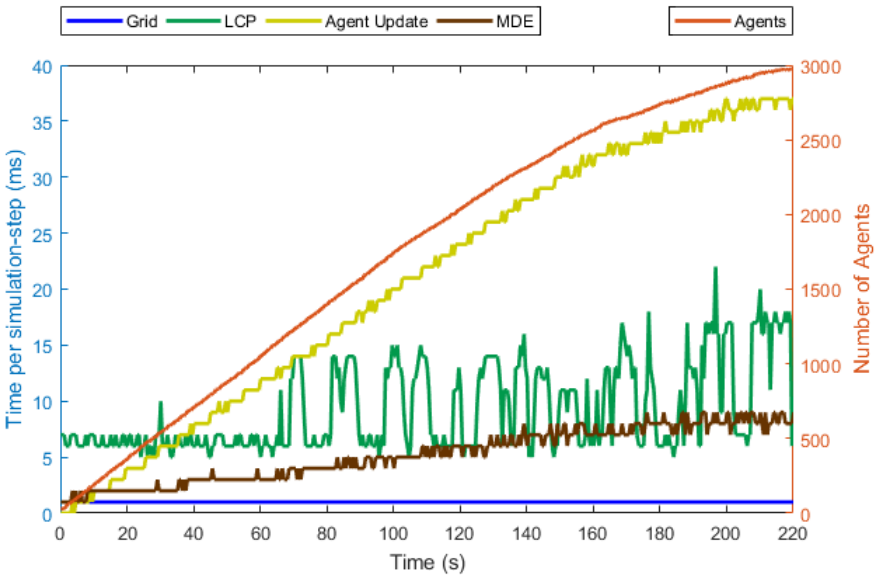


Figure 5.20: Performance with gradually increasing grouped agents in the SUBWAY scenario

5.5.3 Outcome

Overall Analysis

In all 4 scenarios it is evident that maintaining the staggered grid remains constant to 0-2 ms depending on the scenario. This is due to that the number of grid cells and velocity nodes remains at a quite low, constant value.

More variable is the MDE step between the OFFICE and SUBWAY scenarios. At the former it requires as much as 25 ms before gradually decreasing along with the number of agents. In the latter, it rises with the number of agents up to a maximum of 7-10 ms. The difference is quite big when comparing the two, and can most likely be due to that the OFFICE environment is smaller and thus forces the agents together more than the larger SUBWAY scenarios. These are also both scaling along the number of agents, although not linearly in all cases; figures 5.19 and 5.20 show an approximate ratio of $(1 : 0.16)$. Only the MDE performance of the evacuation scenario in figure 5.17 seems to converge to a linear decrease with the number of agents.

Calculating the velocity, the new position as well as contributing to the grid seems to scale linearly for the agents in the Agent Update measurements, seemingly at a maximum effort of 30-35ms with 3000 agents in the scenarios. Although not being fully linear in the SUBWAY scenarios, such scaling is to be expected when increasing the number of agents; each performing the same minimum number of tasks without any exceptions.

The LCP Step

The most fluctuating sub-step of the simulation is the LCP solving and updating step for solving psolve and renorm (Equation (2.15)). In the SUBWAY scenarios it would remain quite constant at around 6 ms until there is a sudden state of fluctuation

rising up to 20 ms. Without doubt, the LCP solver performs more iterations for solving the system of unknown pressures as the density of the crowds increases, hence the decrease in performance. A pattern to be seen here is that over a span of seconds, the LCP step requires gradually more time for solving until reaching a maximum cap, followed by a decrease back to a state where no correcting velocity calculations are needed. The step does not seem to scale as linearly as the previous steps. Instead, a more binary pattern is observed; either there are cells in the grid that require pressure corrections or there are not. This is true also for the evacuation scenario, where the LCP step is at around 6 ms for the first 110 seconds followed by an almost immediate drop to 3 ms throughout the simulation. With respect to the said, it can be concluded that the LCP step does not truly scale with the number of agents indefinitely as there is a defined maximum number of cells to be corrected: Since the number of cells in each scenario is constant there is little room for further effort when all cells are practically full. More effort is however seen when all cells are not full, and in these cases the number of agents does have a say in the performance of this step.

Groups

Having grouped agents in the SUBWAY environment has a clear, but minor, impact on the performance of the simulation. The overall agent update step seems to have an increase of 50% in ms for various number of agents. All other steps do however seem to remain equal in performance to that of individual agents only. This was not expected for the MDE step as it was though that slightly worse performance would be expected when grouped members have a smaller minimum distance towards each other; agents were expected to be more packed in the neighbour bins used for the MDE, and thus provoke more calculations as the calculations of MDE within the bins scales quadratically. What could be expected however is that the scenarios involving groups would require longer time to converge to 3000 agents. This was due to requiring slower spawn ratios when using groups in order to compensate for not spawning 1 agent but 2-4 agents at each spawn tick. The slower spawning ratios is however deduced to not affect the results in any significant manner as the simulation flow is exactly the same throughout all SUBWAY scenarios.

5.6 User Study: Perceived Groups

This section presents the results gained from the user study defined in the previous chapter. The study featured 12 participants of ages 21-34, 11 males and 1 female, with different nationalities. All participants had an academic background with knowledge of basic graphics and interaction.

As a reminder, here is the used group distributions along with the labeling of (I)ndividuals, (P)airs, (T)rios and (Q)uadrupels.

Distribution	Individuals (I)	Groups of 2 (P)	Groups of 3 (T)	Groups of 4 (Q)
1	0.48	0.43	0.08	0.01
2	1.0	0.0	0.0	0.0
3	0.1	0.3	0.3	0.3
4	0.25	0.25	0.25	0.25

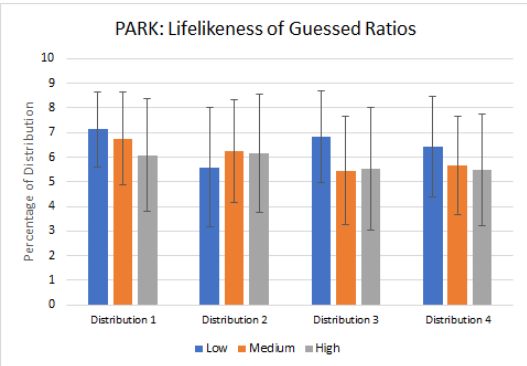
Table 5.8: Group distributions used in the user study. Note the abbreviations (I)ndividuals, (P)airs, (T)rios, (Q)uadruples of the distributions. All of the values are in decimal percentage form

5.6.1 Perceived Realism

The study featured many comments, often of similar nature, regarding a variety of participant perceived features. The most apparent of these were that:

- Streamlines of people were easily perceived at higher densities in PARK scenes
- Sharp turns in all scenes seemed unrealistic with regards to movement
- Clusters of people were more easily seen in the SUBWAY environment, causing participants to perceive groups larger than 4
- Having all agents have the same velocity decreased the overall realism of the movement; a marching behavior was observed

The following graphs display averaged results of the three final questions of the question sheet in table 4.5. Standard deviations are used along with each average.



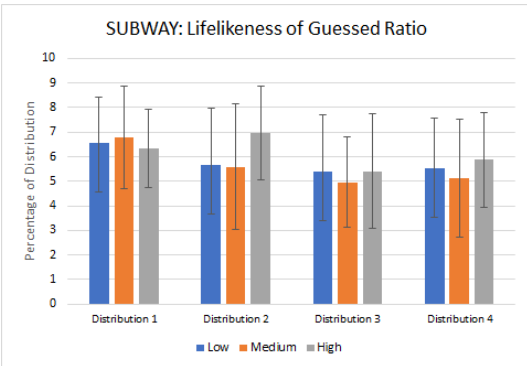


Figure 5.21: Degree to which the ratios seemed lifelike

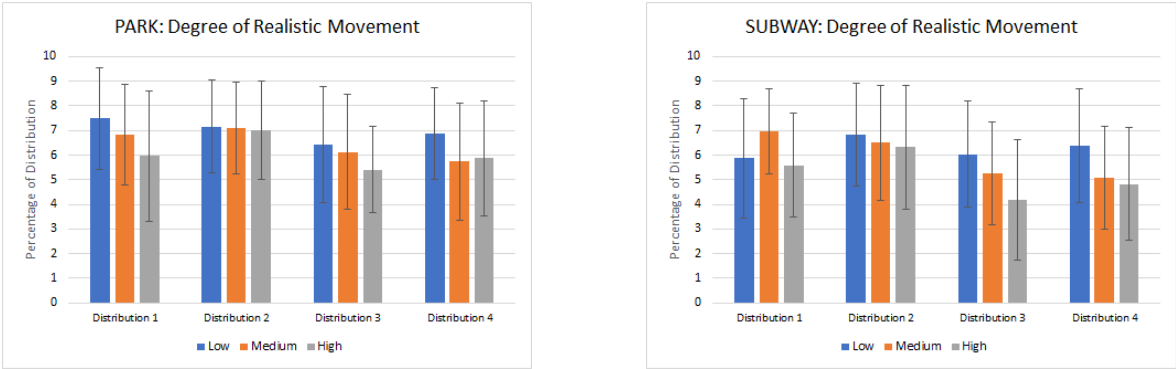


Figure 5.22: Degree to which the movement seemed lifelike

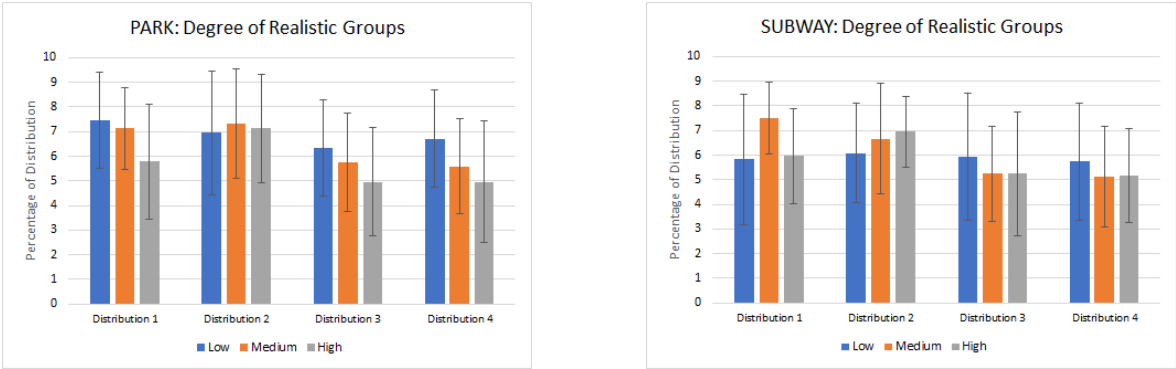


Figure 5.23: Degree to which the groups seemed realistic

5.6.2 Perceived Groups

The following graphs depict the averaged guesses on various group distributions, with variances and standard deviations. Each distribution features guesses in low, medium and high density settings, along with an indication of the actual value.

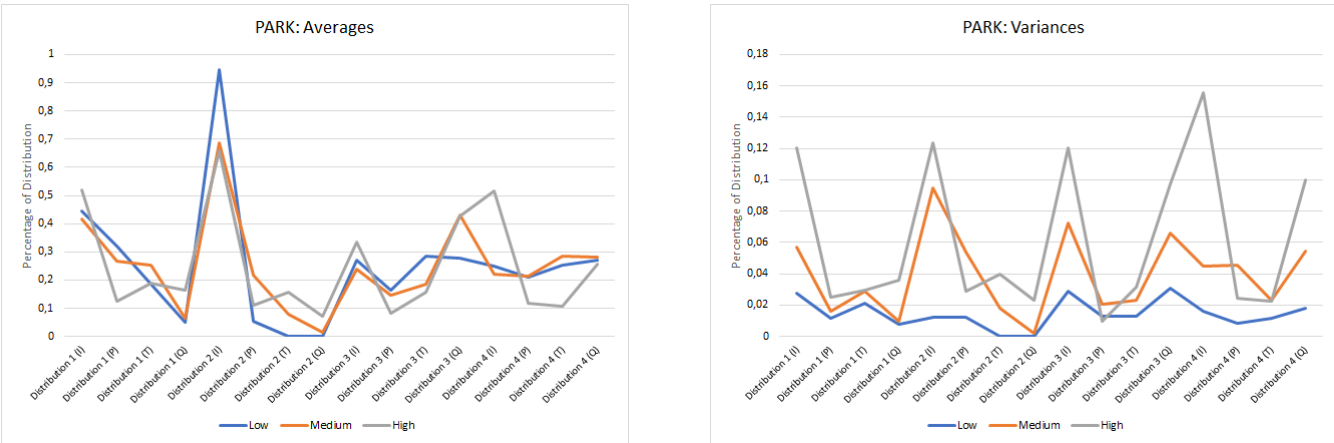


Figure 5.24: Averages and Variances for guessed group distributions of the PARK environment

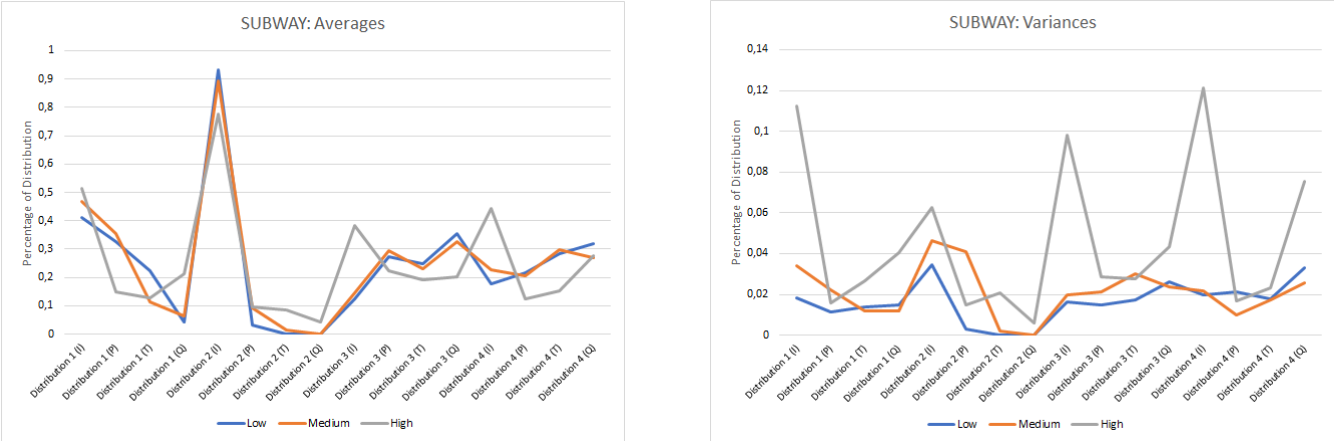


Figure 5.25: Averages and Variances for guessed group distributions of the SUBWAY environment

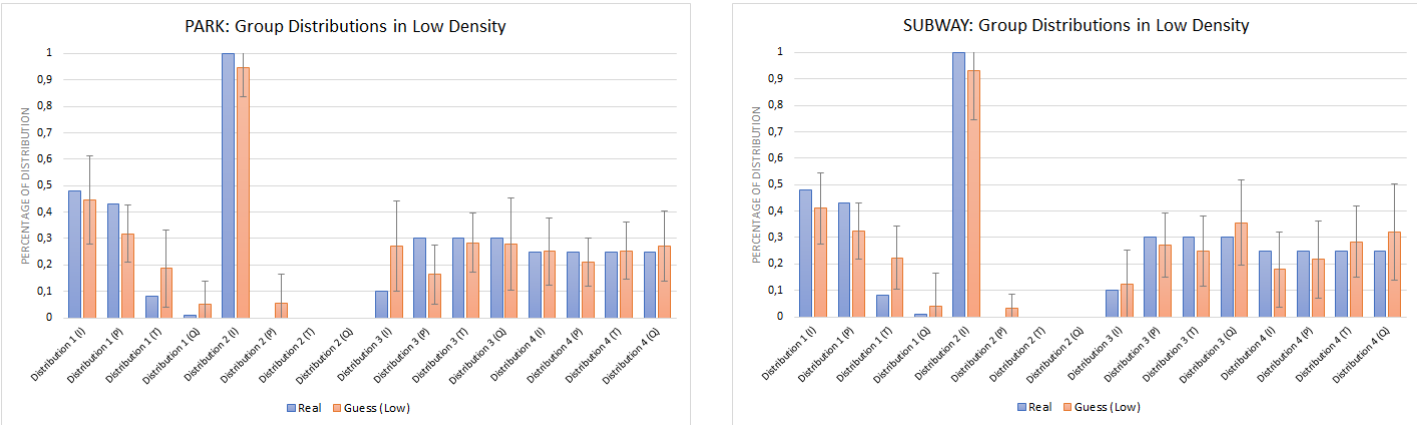


Figure 5.26: Average guesses (red bar) for the group distribution of PARK in low density compared to the real value (blue bar)

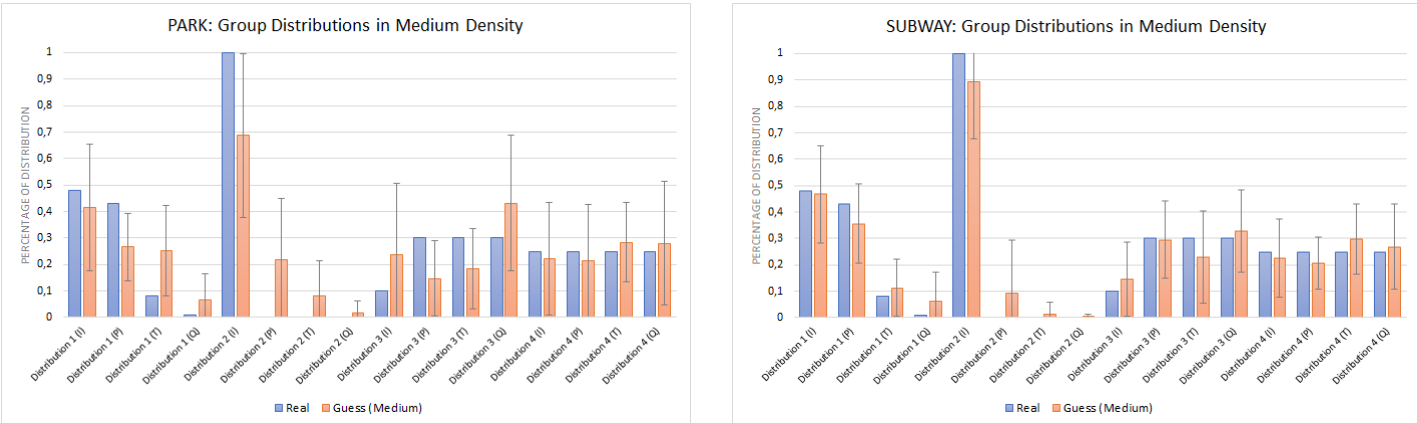


Figure 5.27: Average guesses (red bar) for the group distribution of PARK in medium density compared to the real value (blue bar)

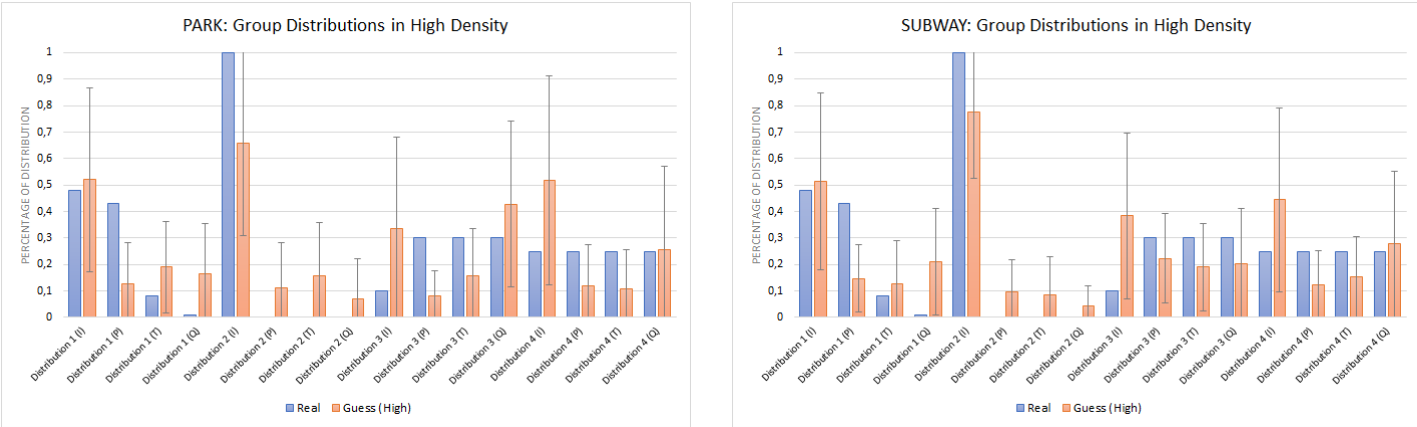


Figure 5.28: Average guesses (red bar) for the group distribution of PARK in high density compared to the real value (blue bar)

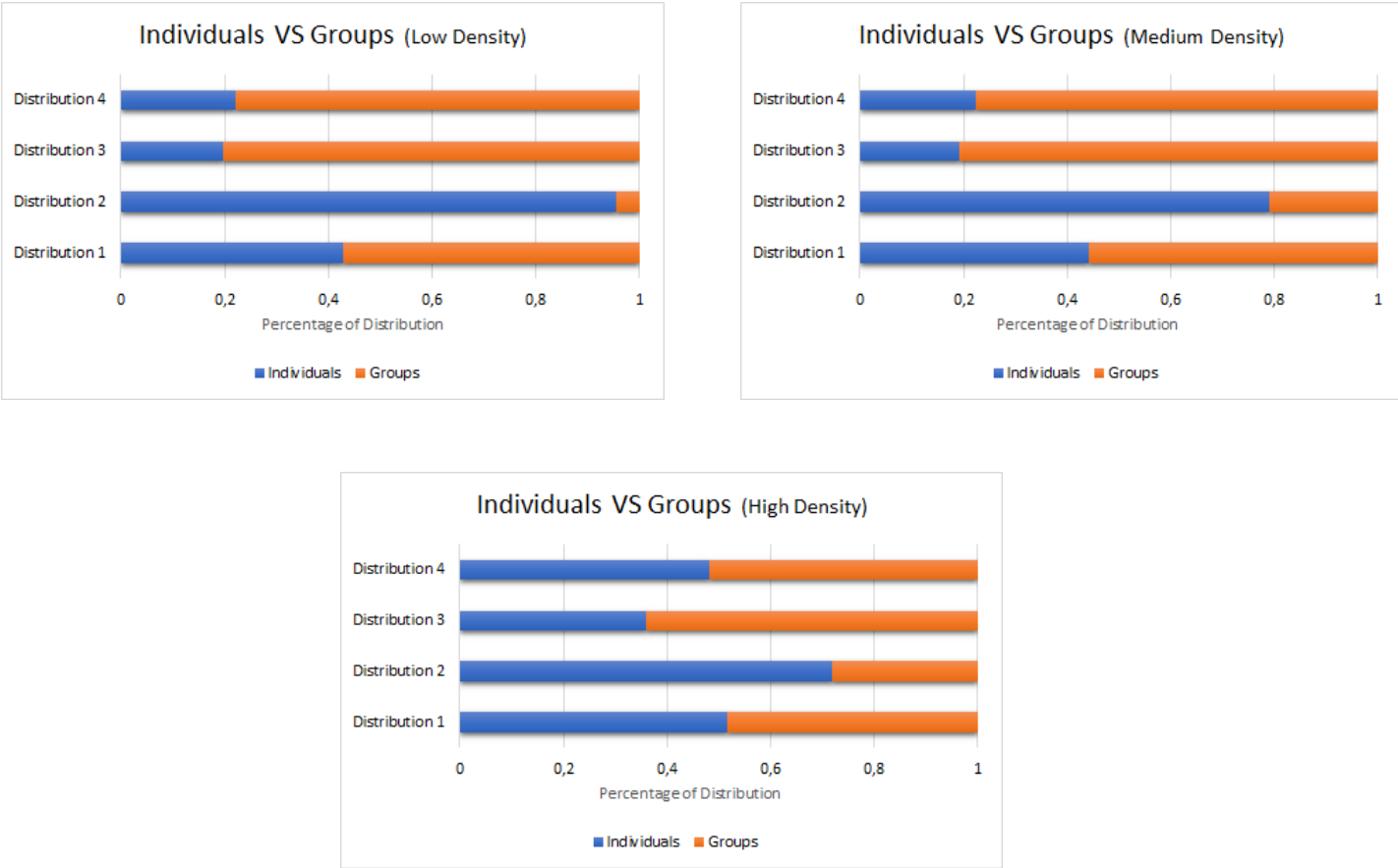


Figure 5.30: The ratio of averaged guessed groups versus individuals in the three different densities

5.6.3 Outcome

Scores on Realism

When observing figures 5.21 - 5.22 one sees not too diverse, and somewhat equally sized standard deviations. On the contrary, a more diverse dispersion is seen in figure 5.23. Furthermore, the averaged scores do not expose any significant deviation for neither density nor group distribution. Instead, they are in a broader sense all averaged between 4 and 8 with relatively equal scores per distribution. Changes are however seen when comparing the results by density.

Lower density scenarios tend to yield a slightly higher score in either graph, with higher values in 16 out of 24 sequences. Furthermore, when comparing the scores on high density scenarios it is seen that a higher score on Distribution 2 is persistent in terms of realistic ratios, movement and groups alike, followed by Distribution 1 as the second highest. What these results could propose is that it is more plausible to simulate individuals in high density scenarios, both in terms of plausible movement and in the presence of groups. Also, if groups are present in high density scenarios the results would suggest that having the distribution in Distribution 1, with a balanced ratio of groups and individuals seen in actual empirical data, yields more plausible results as well. Similar conclusions for the latter have been seen in Peters and Ennis [21] study, in which they conclude that their result *"[...] suggests that adding groups to crowds can increase the scene's plausibility, provided that you're careful in terms of the ratios of individuals and groups.."*.

The given data from the study does however not suggest a larger difference between distribution and density when observing the average scores. This is however not the case when performing Two-Way ANOVA tests for the different scores, which shows significant differences between density and group distribution. A first test on the lifelikeness of the guessed ratios shows that one cannot reject the hypothesis that there is a mean score for each different densities. On the other hand it is shown with 100% cumulative probability that one can reject the possibility of having an equal mean for scores on the different distributions; the participants have thus scored differently between one or more distributions, i.e., found one category of individuals/groups more lifelike than others. It is moreover shown that there is significant interaction, with 100% cumulative probability, for both the density and distributions in the movement scores. Hence scores differ between different densities as well, meaning that some movement is deemed more plausible at one density than another; this outcome does however not tell anything about which density that would be for the category of movement alone. Figure 5.22 does not seem to be consistent with one higher density, but does show that average scores in high density sequences are consistently lower. Movement in high density could thus result in a lower plausibility. The loss in degrees of freedom could be one reason for this as agents in a larger crowd cannot move much at all without having agents in front moving.

Further Two-Way ANOVA tests on the degree of plausible groups in the simulation show significant differences both when considering the density and distributions alone, but also in-between these variables. With 99% cumulative probability there are interactions between the perceived plausibility of groups in various densities and group distributions. Once again, this does not show which combinations of

group distribution and density that yield more realistic group behaviors. The averages in figure 5.23 does not show any definite result either. There are however very different scores for each density between Group Distributions 1 and 2 in the PARK and SUBWAY environment. For instance, Distribution 1 in the PARK environment show a slightly higher score in low density, followed by medium and the high density. In the SUBWAY environment however one can see a 1.5 point averaged higher score for medium density while low and high density scores are somewhat equal. These changes introduce a new hypothesis that the environments per say, with individual lighting, environment and angle of recording, could affect the perceived plausibility of groups.

Comments on Movement

With further respect to given comments on the study, it can be seen that all of them concern the perceived movement in some way. With perceiving streamlines as the only positive quality, as these occur in real crowds, the other comments pointed out something negative about the movement. Making sharp turns, moving at the same velocity and moving into larger clusters were all perceived as negative attributes, as commented by a majority of the participants. The two former could have affected their score on realistic movement, while the latter could not only have affected their score on realistic groups but also group distributions in general. This is however as of yet only speculations.

In defence to the given comments, having agents with the same velocity was a simplification made in order to decrease the number of variables for the study, while sharp turns were used due to lacking any reliable smooth turning mechanism (see the results of this in the next section. Furthermore, agents walking into larger clusters is the result of agents not taking other agents into consideration in its path planning for a preferred velocity. These remarks do however only highlight a need for improving the simulation with smoothed turns and better global planning, as these could affect participants perceptions of realism - hopefully for the better. Having all agents walk with different velocities is an improvement to do as well, but considering that this was made deliberately in order to shift focus from individual agents to perceiving groups it may remain the same for any future study. Real people do nevertheless walk in different paces, and thus a need for more complex studies becomes more apparent.

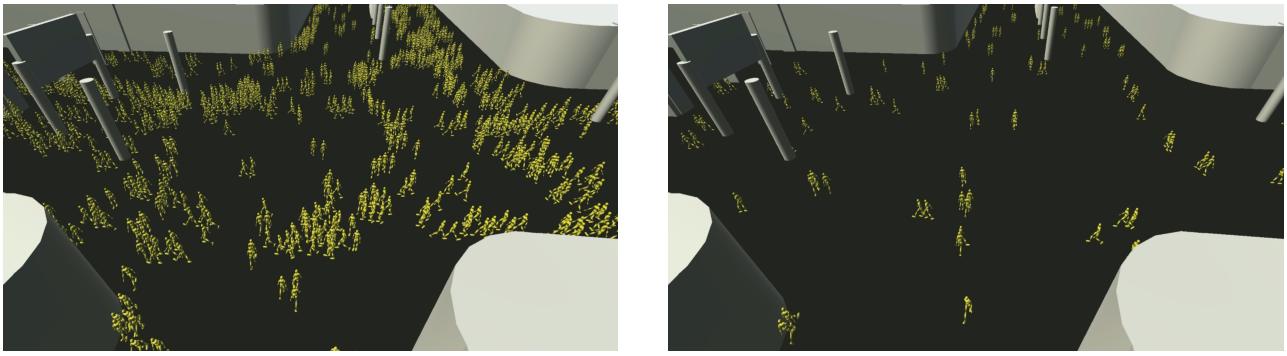
Analyzing the Distributions

There are two apparent patterns in the seen group distributions in both the PARK and SUBWAY environments. The first one is that in most cases, the averaged guesses in low and medium density settings are closer to the actual group distribution ratios than guesses in high density settings. A second pattern is that guesses in lower densities have less dispersion in their standard deviation than higher densities. This can be interpreted that in average, lower densities show more accurate guesses while with consideration to the standard deviation, higher densities cause more dispersed guesses.

This does however not mean that there is a clear difference between the densities per say with regard to having more or fewer groups in the crowd. Nevertheless,

further Two-Way ANOVA tests on the two different environments show that there is a significant difference for the interaction between densities and group distributions. With 100% cumulative probability it is shown that the average guessed ratio of one group distribution is affected by the used density. If one would consider the guess of a group distribution to correspond to the perception of certain groups, it can be said with other words that: The average perception of groups in a crowd are affected by the density of the crowd. To see hints of why this might be, consider the averaged guesses and variances of figure 5.24 and figure 5.25. These show a close relationship on the averages of medium and low density guesses, with a somewhat repeating pattern for medium and high density guesses. The variances plots does however expose increasing variance with increasing density for both environments. More intersections are further seen in the SUBWAY scenarios while a larger spread of variance is seen for the PARK scenarios.

A theory in as to why it seemingly seems harder to detect groups in higher density could be that groups are more invisible in such crowds. As illustrated by figure 5.31, the left image shows a crowd with 90% groups (30% groups of 2, 3 and 4 respectively) and yet, the very same crowd might have only consisted of individuals. It is only when agents are in-between the seen clusters of agents that certain group formations are easily perceived; in all other cases the formations themselves are impossible to distinguish as the agents are very densely packed. The made user study seems to support this claim in some sense: It has only been shown that there are certain interactions between density and group distributions. In order to see exactly which density(ies) this concerns or/and which distributions are affected the most, one would have to perform more thorough tests such as t-tests.



(a) High Density setting with 90% groups

(b) Low Density setting with 52% groups

Figure 5.31: Groups in a high and low density setting

Facing the Hypotheses

In response to the made hypotheses in the previous chapter, the following is said:

Groups and their various distributions will generally be easier to detect in low and medium densities: The guesses on the group ratios will be somewhat accurate

More accurate guesses was seen for lower and medium density settings, and thus groups could be said to be more easily detected in these densities. This does however not mean that they are detected with certainty in these densities

Groups will be harder to detect in high density crowds, leading to guesses of either incorrect uniform distributions or one particular category of groups due to the uncertainty

More dispersed guesses was seen for high density scenarios, but not to the extent that the averaged guesses assumed an incorrect uniform or particular distributions. Further comparison between groups and individuals per say, illustrated in figure 5.30, show that there is a change of averaged guesses between groups and individuals in the high density scenarios compared to medium and low density. In addition to larger standard deviations in high density guesses, there is a strong possibility that the perception of groups in a high density environment differs from other densities, and is possibly also harder in that sense.

The natural order in Distribution 1 will not be evident in the average guesses.

Taking into consideration that there are large standard deviations for Distribution 1 in both of the environments, this hypothesis cannot be rejected

At medium to high densities, Distribution 2 will yield more perceptions on agents being in a group than by themselves.

While there are increased guesses for groups in medium and high densities on Distribution 2, with only individuals in the scene, this hypothesis does not hold. As seen in the graphs plotting the guesses of the group distribution, the majority are in favor of individuals. Hence individuals were spotted and relatively not mistaken for being in a group due to them being near other agents

Distribution 4 will yield the most lifelike guesses

With further consideration to both averaged guesses and standard deviation, figure 5.21 does not seem to show a significantly higher score for Distribution 4, an uniform distribution, or any other Distribution at all. In terms of the highest overall score nevertheless, it would seem that Distribution 1 is the most lifelike guess; an interesting outcome since this distribution is based on group ratios observed in real crowds.

The groups will have an average realism of around 5/10; they will not appear to be quite lifelike due to the abreast walking patterns

There is nothing to suggest the lifelikeness depending on the agents walking formation. The hypothesis for an averaged group realism of 5/10 does however seem to hold in some sense, with stronger relation to the SUBWAY environment.

Conclusions

The perception of groups does seem to depend on the density in the shown crowds. Participants tend to make more disperse guesses at high density scenarios while the range of guessed groupings seems to be somewhat less disperse at lower densities.

While a relationship between density and guesses for group distributions as well as perceptions has been proven by an ANOVA test, further tests will have to be done

in order to distinguish how certain factors correlate. Thus nothing can be said with confidence with regard to whether groups are harder to perceive or are found to be differently plausible in higher densities with an ANOVA test alone. Through observing the variances of the guesses, it can however be seen that there is a larger variance for guesses in high density. This indicates more uncertainty amongst the participants, with a potential for harder perception of groups in higher density. More uncertainty is also seen when observing the scores of group plausibility, where the plausibility of groups varies in different distributions along with density, but also between the used environments for certain group distributions. Hence there are variables affecting the perception of groups, which nevertheless are perceived. Further testing, and preferably a larger number of participants in future studies, could give more definite results.

5.7 Movement Options

5.7.1 Scenarios

Three different options affecting the movement of agents have been developed. These are the Walk Back, Skip Node and Smoothed Turns options, defined in Section 3.4. In order to evaluate their strengths and weaknesses, the OFFICE environment is used and will feature two settings in evaluating these for overall performance and plausibility gain, or loss. The first setting will feature 2500 evacuating agents while the second setting will have continuously spawning agents in parts of the environment, with various goals and increasing density. Each of the settings will be run without, and with each option separately. Furthermore, there will be one scenario of individual agents and two scenarios of grouped agents in the second setting.

5.7.2 Parameters

The scenarios will feature the same group distribution as seen in table 5.8 together with the settings in table **??**.

Scene	Cells	Neighbor Bins	Velocity (m/s)	Alpha	Solver(Iterations, Epsilon)	Min Dis
OFFICE-1	40	150	2.5	1.0	PSOR(40, 0.0001)	0.8
OFFICE-2	40	150	1.8	1.0	PSOR(40, 0.0001)	1.2

Table 5.9: Scenario settings

5.7.3 Simulation

Measured Data

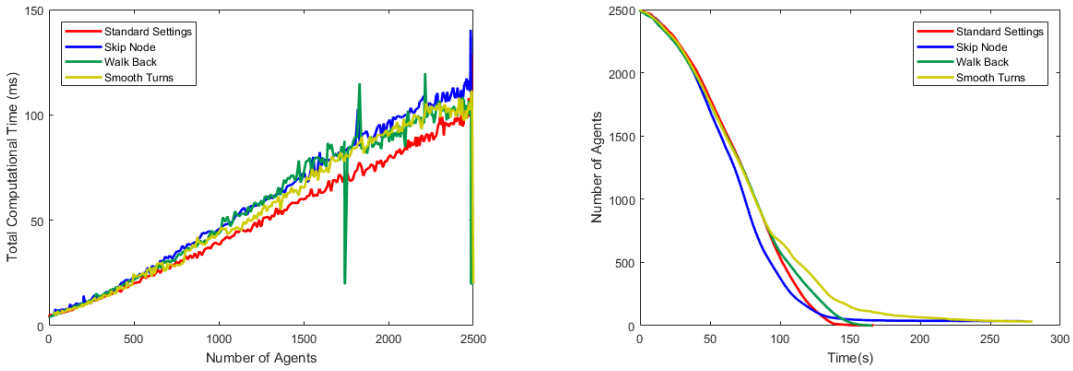
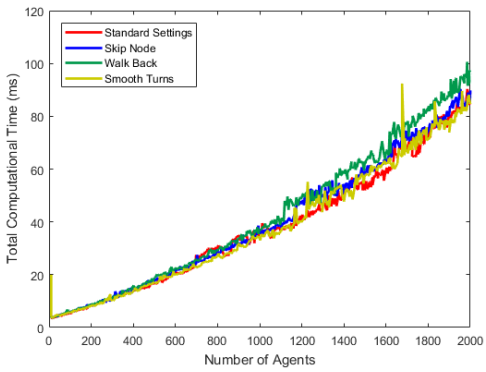
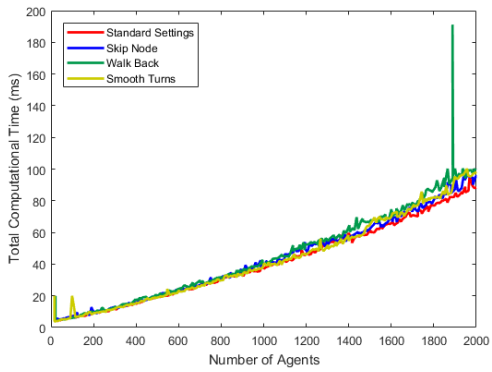


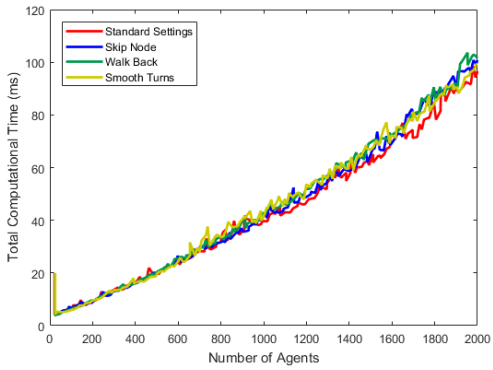
Figure 5.32: Performance of 2500 evacuating agents in the OFFICE-1 environment



(a) Group setting 1



(b) Group setting 2



(c) Group setting 3

Figure 5.34: Performance of group setting 1-3 in the OFFICE-2 environment

Perceived Behavior

The evacuation scenario in the OFFICE-1 environment resulted in a successful evacuation of all agents with only a few agents being stuck in a local minima for a short while. An even faster evacuation was observed with the "Skip Node" option turned on, as seen in figure 5.35. This did however cause some agents to be stuck permanently. Using the "Walk Back" options instead did however result in all agents escaping with no one stuck in a local minima, although at the price of some agents walking back against the escaping stream.

(a) Agents in a local minima (Skip Node)

(b) Some agents walking back (Walk Back)

Figure 5.35: Odd behavior in the evacuation scenario

Having agents walk with smoothed turns gave plausible results when agents were turning without an intersecting stream of agents. In other situations, some agents near the perimeter of a stream of agents performed sudden turns against unexpected areas well off the agents intended paths. The overall behavior was at this time observed to be erratic.

Figure 5.36: Some agents walking erratically (Smoothed Turns)

With agents walking more calmly, in multiple directions, and with a larger pre-

ferred minimum distance in the OFFICE-1 environment amplified the effects of using the options. With only individual agents, it was generally observed that agents tended to get stuck in local minimas more easily as the overall density increased; smaller vortices were thus formed that led agents away from their intended paths. This was more apparent with the "Skip Node" option, along with additional erratic behavior when using "Smoothed Turns". The number of agents stuck in local minima was however almost reduced to none when using the "Walk Back" option in the OFFICE-1 environment.

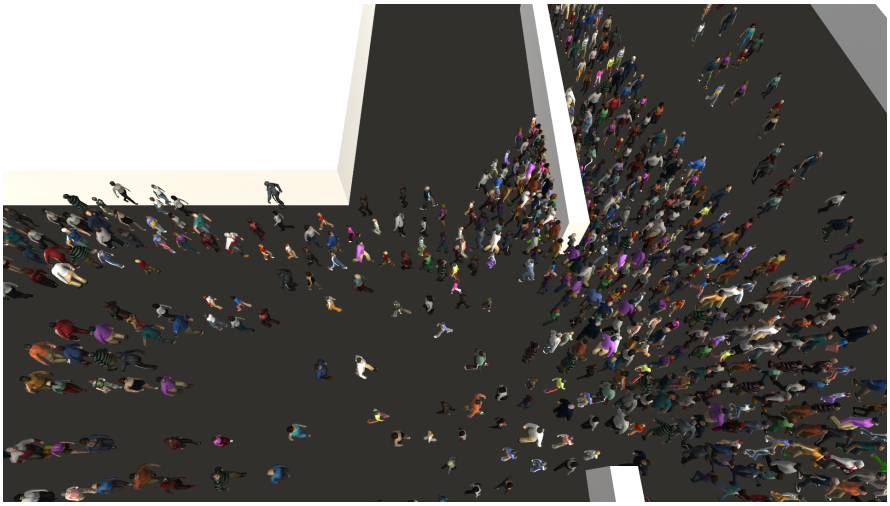


Figure 5.37: Agents stuck due to earlier density corrections

It was observed that adding groups to the crowd did increase the number of agents stuck in local minima. More specifically, it was observed that having group leaders stuck in a minimum caused the group follower(s) to stay stuck with the leader as well. Having group follower(s) did in other cases not cause any additional disruption as they turned just as smoothly as their respective group leader, and walked back when the leader did so as well.

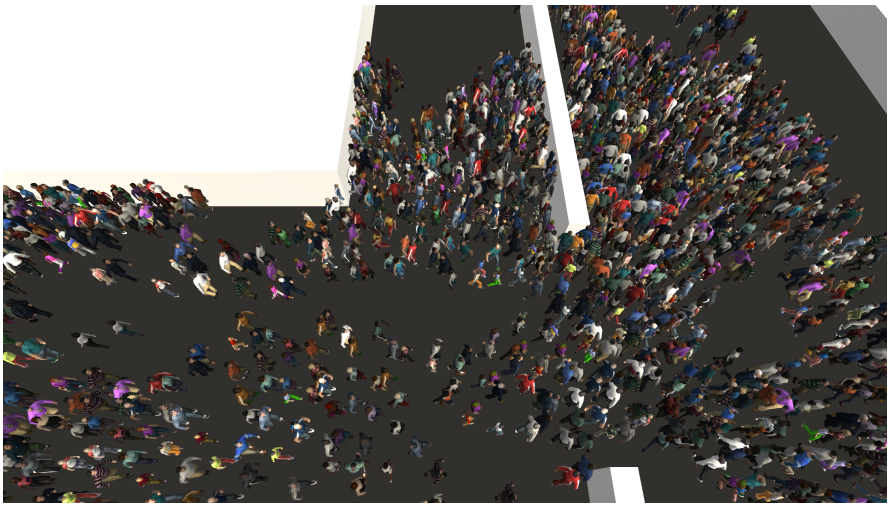


Figure 5.38: Groups increase the number of stuck agents

5.7.4 Outcome

Performance

Using any of the 3 options did decrease the overall performance of the simulation in terms of ms, although not considerably. As seen in figures 5.35 to 5.34 the overall increase is at most 10ms, with only some minor deviations. Furthermore in the terms of performance, figure 5.35, show that agents were evacuated faster with the "Skip Node" scenario while a slower curve is seen when using "Smoothed Turns". The behavior of the former can be explained that agents found a faster way to reach the end than the path laid out with custom placed nodes; the computer found an optimized path other than the custom placed path. The latter behavior with "Smoothed Turns" is thought to be due to the opposite, namely that agents along a path are forced to take longer turns that increase the overall path length. This expose the strength and weakness of having paths that minimize the effort of agents or increases the effort in order to further acquire plausibility gains.

Smoothed Turns

As mentioned in the results, using "Smoothed Turns" was good for undisrupted turns. When intersecting with agents walking in from other directions this quickly turned implausible with erratic behavior. It can thus be deduced that when an agent performing a turn is forced to change its direction, either due to a correcting continuum velocity or collision avoidance velocity, the underlying programming to steer towards the target direction causes the agents to become "lost". In other words, the angle between the target direction and the agents current direction becomes unexpectedly variable and thus steers the agent in a longer and erratically long motion effectively having the agents off-course in the end. Moreover, as the agent never finishes its turning motion, which was done when agents left the node to perform the turn, the agent would perform smoothed turns indefinitely.

In conclusion to this option, it can be said with confidence that the made option of smoothed turns is implausible for a simulation of high density crowds. Agents require more control for making smoothed turns, with respect to not only their current direction, but perhaps also in terms of a better global planner; forcing agents to walk along a set of smaller, proximate nodes laid out for a smooth turn is thought to be one such improvement.

Skip Node

When using the "Skip Node" option it was observed that agents took faster paths that in the evacuation scenario proved to be plausible, but in the other scenario caused local minima problems. Furthermore, having agents take faster paths in the less troubled environment in OFFICE-2 was seen as somewhat implausible as agents tended to walk very close to obstacles in the faster paths. While further empirical data will have to prove the relation, it is believed that a high number of people evacuating from a building tends to take the faster paths in contrast to calmer people that traverse in a more ordered office, even as the density increases. Hence, the option could be more proper to use in the made evacuation scenario. This theory is at least supported by the seen behavior and plausibility in the made scenarios.

Walk Back

Having agents walk back to their last seen node when not seeing their next proved to yield functional gains in all scenarios. Previous instances of agents being stuck in either environment, either temporarily or permanently, was almost omitted with this option. The behavior of walking back is nevertheless a huge letdown in terms of plausibility, although this might be variable for each scenario. Agents walking back in an evacuation scenario is extremely unlikely and yields all in all a huge drop in plausibility. These are however more invisible in the second environment as agents walk in multiple, and sometimes intersecting, directions, hence not resulting in a greater loss of plausibility. Therefore, a balance between functionality and plausibility gains could be something worthwhile to investigate further. It was certainly more visually apparent that something was more wrong with the crowd as they got stuck when not running the "Walk Back" option.

The trade-off is thought to be more reasonable in higher densities, but nevertheless unreasonable at lower densities as well as certain environments and scenarios. In general, such trade-off should be avoided as much as possible; the perfect simulation of reality has no simplifications for functionality.

Groups

Having groups increases the density of certain regions in the environment as these tend to take up more place with their abreast formations. Thus, the behavior of having more agents stuck in a local minima was to be expected. The cause were not due to the groups per say, but due to the continuum velocity causing vortices in the denser regions. Whereas these vortices do perform good and realistic motions in more open environments, they are observed to be a nuisance in smaller corridors such as the one in figure 5.38. Modifying the parameters to have less pressure corrections will solve this, but in essence, this suggests that groups and the underlying unilateral functionality have some compatibility issues. This could however also suggest that the method by Narain et al. [20] works less good in narrow environments, as making the corridor more wider would solve the problem altogether.

Having groups is however thought to be more plausible, not only because there are groups in real life but also because the simulated groups followed up on all movement alterations caused by the various options. This is thought to be true even though some options caused an overall implausible effect, as mentioned above. In the end for the made simulation, one is yet again back to a trade-off, although this time between using more or less groups for narrow environments. The former would be more locally plausible but globally implausible, while the latter would be more globally plausible, less locally plausible, and in overall more functional.

Chapter 6

Discussion

This chapter will summarize and discuss the overall gained results and conclusions made for various components of the simulator. The research questions for this thesis will also be addressed here, with final comments on the made approach and its use for social ties.

6.1 High Density Crowd Simulation

The made simulator has been based on Narain et al. [20] unilateral approach for simulating high density crowds in real time. Similar effects have been seen in the made approach, but also several flaws that limit both the plausibility and performance of simulations.

6.1.1 Effects and Performance

Treating the crowd as a continuous mass allows for a computational cheap collision avoidance step that prevents agents from moving into each other very well, as long as calculations include the minimum distance enforcement step. Simulations have however shown that as the density increases, so does the sensitivity to the simulator's settings. Some "optimal" settings show smaller curved movements for high density crowds when faced with an intersecting stream of agents, effectively causing smoother movement and turns in a macro scale. Many settings have however proven to cause implausible simulations, with extreme vortexes or sudden deviations (section 5.2) leading the agents astray and not really progressing the simulation towards a converged state. A simulation with a high density crowd, with not too density straining density settings, is seen to affect the agent's correcting velocities for the better. In conjunction with behaviors seen in other approaches and real life situations alike, the most plausible effect however is the formation of streamlines. The effect of having agents walk behind other agents if they walk towards the same direction is however believed to not be caused by the continuum representation, but rather due to the minimum distance enforcement. This is motivated by the insight that agents perform less distance corrections, through adding a collision avoidance velocity, for agents not obstructing the path. In a sense, agents are "pushed" into a state where they experience the least movement corrections in front or behind of the current direction; experiencing corrections to the sides is still possible if there for in-

stance are parallel streams in opposite directions. Another insight is that the line of agents are governed by the agent in the very front: This agent experience minimum distance corrections from all sides, effectively "digging" through opposing agents.

While observed effects do share a similarity with those found in the Narain et al. [20] paper, there are also somewhat similar performances in the made approach. Real-time simulations of 3000 agents have been possible with about 20-35 ms per simulation step, therein minimum distance enforcement and solving the linear constraint problem in higher densities. Moreover, upholding the UIC for having a correcting continuum velocity was deemed to be the most expensive step in some scenarios, although in greater fluctuations. This was furthermore done by using a different solver for the LCP, namely the PSOR solver, simply because it yielded faster, but nevertheless less accurate, convergence.

It was however deemed impossible to simulate a much higher number of agents in real time. While the core simulation algorithm per say did not put a larger strain on the system, it was found out that the rendering of agents caused a huge stress on the computer running the simulation. Even with low polygon models of humanoids, the simulation could only achieve roughly 13 frames per seconds with 3000 agents. Using more abstract, simpler models did however increase the FPS to around 30. It was deduced in the rendering results (section 5.4) that the higher strain was caused by the underlying simulation engine, Unity, and its rendering mechanism. In a shorter explanation, it did not utilize the computers GPU when rendering humanoids, and instead put more efforts into the computers CPU. This can however be changed if one would consider writing shader scripts that allow for more communications with the GPU, but is deemed well outside the scope of this thesis. In summary it can be said that in terms of performance in high density scenarios, one would gain the most by performing simulations in an environment that utilize modern hardware components to the fullest. Another option would be to use simpler models, but this would lower the plausibility of the simulation as a result.

6.1.2 Flaws and Improvements

There are many components in the simulation that either did not work as expected, or caused bad behavior, in high density simulations. The largest bottleneck, which is repeatedly mentioned throughout the analysis of various components and scenarios, is the global planner. While it does yield paths for agents in constant time, it is deemed to be insufficient for high density scenarios. The strongest reason for this is that once an agent diverts from its path there is the immediate risk for either being stuck in a local minima or walking into obstacles and then "slide" along them until they are back on track. Both of these behaviors should not exist in a realistic simulation of people, but does unfortunately exist when there are no means for agents to walk back towards their intended routes. While an effort has been made for this with the Walk Back option, it did not fully contribute a lot for the perceived realism. Agents walked back on track with this option, but did nevertheless walk back along their path for this, which is very bad in terms of realistic behavior.

Further effort was made with trying to control the paths with a sampling on the environment's medial axis, such that agents would prefer to walk in-between obstacles and thus minimize the risk for unnatural behaviors. This was however

also proven to be flawed when using pre-calculated paths (section 3.2.2) that minimize the distance between agents start and goal; the global planner did not include enough nodes to fully utilize the centered positions. The global planner did however perform better when used in a simulation with lower densities as they would not have any external forces affecting their paths. In conclusion it can however be said that for plausibility gains in a high density crowd, one must include a global planner that allows for path planning during run time, even if only when agents are astray. This will decrease the performance of the simulation, but will yield a more realistic one.

Similar arguments can be said for having agents make smoother turns. It worked great when there were fewer or no agents obstructing turning agents, but caused very bad effects when that was not the case. Agents were led astray and performed unnatural, long turns away from the intended direction when making turns based on their direction and the direction of their next position in the path. While these conditions can be changed to allow for more "smarter" turns, such as only turning when the angle between the agents direction and the target direction is within a certain interval, or whenever there are no collisions altogether in front of the agent, it yet again falls down to somehow smarter path planning for the agents. The global planner could help in this case for instance by forcing agents to walk along certain smaller, intensely packed nodes whenever turns are needed. This produces more control over the agents.

Indeed, there seem to be a need for more control for the path planning and agent movement altogether, despite that almost all components in the simulation are made for this purpose alone. With further regard to the research question *How are components necessary for plausible high density simulations?* one can in a summarized way draw the conclusion that components are necessary whenever they contribute to the plausibility in a controlled manner. It is only then that controlled simulations of real life scenarios can be made with further effort to tailor-cut each particular scenario with made settings. The very basic principle of simulating crowds is having the same behavior as real people in a crowd, and to the best effort hide the fact that the seen walking characters are virtual.

6.2 Groups

The main contributions from this thesis is adding the element of groups for real time, high density crowds running under Narain et al. [20] unilateral conditions. This has proven to yield very interesting results with regard to both performance and perception gains. Each research question associated with these two factors is answered in the following two sections.

6.2.1 Performance

When comparing simulations run with and without crowds, it has been shown that groups do have an effect on the overall flow of the crowd but do blend in well with individual agents. Groups blend in with streamlines and vortexes alike in higher densities, although causing somewhat higher dispersion due to a will to stay within proximate distances to respective group slots. Furthermore, the ability to stick to-

gether is consistent in a variety of scenarios and does prove to not disturb the movement of other agents not in the group; having the group split or collectively avoid intersecting agents has been observed. Agents in a group have also been seen to follow up on the group leader's turns and pace just as good, or bad, with a smaller increase or decrease in velocity should the agents ever fall behind or ahead of the group.

Having groups in the simulation does furthermore cause a rough 50% decrease in performance when updating each agent's position, but not towards maintaining a minimum distance or calculating the continuum velocity. This was expected, as the only thing separating grouped agents from individual ones is an extra calculation loop for updating the group's slots onto which all grouped agents, except the assigned leader, steer towards (section 3.5.2). The compensation of having a smaller, fixed minimal distance towards members in one's group provides an ad hoc solution for not causing further decrease in performance in maintaining an otherwise longer minimum distance towards group members as well; these have been seen to have a smaller minimal distance in real life. This increase is nevertheless found to be negligible in the long run as the time to update each agent does as of yet grows linearly with the increased number of agents, grouped or not.

6.2.2 Perception

To perceive simulated groups in a crowd of increasing density is the first step towards opinions of whether these behave realistic or not. There has not been a lot of effort in mimicking most of the behaviors found in an actual group, such as the verbal and/or physical communication that might exist in real groups. Efforts have instead been made to keep group members coherent, adjacent and not vulnerable to changes in obstacles or intersecting agents. A smaller user study has then been conducted in order to gain some response to whether groups having these properties can be seen in crowds of various density, and to the best of the participants ability if they are seen as plausible according to each participant's perception on groups. Given results have shown indications that groups are perceived and very differently so in various densities. There are however no direct indications whereas this concerns crowds of higher density or not. A larger dispersion for different group distributions is however seen in simulations of higher density, and could thus suggest for more uncertainty in perceiving groups in denser crowds.

Given feedback does however suggest that the perception of groups is related to more factors than density and the number of agents in a simulation. Most participants believed that implausible movement affected the plausibility *and* perception of groups for the worse. The latter was due to agents clustering together and thus providing the absolute illusion that there were groups of sizes larger than 4. Except for causing misleading perceptions when simulating groups, this behavior has been seen with fewer or none groups as well. A theory in as to why these clusters appear has once again ties to do with the global planner. Agents do not take the positions of other agents into consideration when receiving their pre-calculated paths and thus march straight ahead along its path and "push" other obstructing agents aside. This is true also for groups, onto which the agents either separate the group, is pushed away by the group or for a brief moment gets stuck in front of the group, should

they be of sizes 3-4 and in a tight abreast formation in front of the intersecting agent. The latter is believed to have mislead participants in the study.

It could be said from personal experience that one does not simply walk straight through intersecting groups. The more normal behavior is to steer away from future collisions in good time, and not moments before impact. This supports the need for an even more viable global planner, that surely will require even more computational time in order to also consider the positions of nearby agents when calculating the path for agents in run time. Such time is however thought to yield larger realism to the simulation. Thus the trade-off of performance versus realism is once again apparent.

6.3 Social Ties

6.3.1 Real Crowds

While the given scores on the degree of realistic groups between 1 and 10 was above average in all densities, with clear ties between perceived groups and different densities, it is believed that further plausibility can be gained. One participant made the remark that the seen agents do not seem to ever stop moving. This could be the case for groups in the sense that verbal communication can sometimes occur both within and between groups in real life scenarios, and some of these do involve having the group standing still as a whole. Some examples would be a family deciding where to go next in a fair, a couple stopping for a moment of romance, friends saying goodbye to each other and then walking in separate ways or just having a couple of colleagues chatting in an office. The point to make is that groups often involve social actions, and that simulations of crowds aspiring to simulate groups should include such actions in some sense as well. These could be added in the form of cheap animations that can either be simpler, such as holding hands [24], or more complex as with body language. One must also realize that there is a greater deal of cultural behavior for groups that surely increases the complexity for a generalized simulation of crowds.

6.3.2 Using Crowds

If adding more realistic behavior results in more stress for the system, it is believed that it is worth it if a realistic simulation is sought: A virtual crowd that are few but feels more alive with several actions is thought to be more alike a crowd than a larger number of solely walking agents. In fact, this could be regarded as crucial when using crowds for a practical purpose; you would not want to use a simulation of crowds that does not behave realistically in e.g. an evacuation scenario as that could, in the worst case, cause more harm than good. Hence there is a certain ethical obligation to maximize the realistic movement of simulated crowds for practical purposes.

6.3.3 Ethical Decisions

While this ethical obligation would clearly be violated if one would use the made simulator and its simulated crowds, other ethical decisions has been made through-

out the thesis. These does not concern environmental issues connected with crowds, or sustainability at all, but rather with how information has been used throughout the thesis; a user study has been conducted and with that follows the responsibility to protect the participants' data and anonymity. Hence with such ethics in mind, participants were given a consent form and were informed that they could leave the study at any time without giving reasons to why they would do that. Furthermore, their data is kept save should they ever regret their participation. Such ethical decisions are crucial, not only for user studies, but also along the very long path of simulating realistic crowds.

Chapter 7

Conclusion

7.1 The Simulator

A simulation of crowds has been successfully made with methods suggested by Narain et al. [20] and many other authors. The simulator support high density crowds of up to 3000 agents in real time. Further improvements for performance can and should be made for rendering the agents as humanoid characters. Moreover, the need for a smarter global planner is recognized, even though this might affect performance.

The shown crowds are prone to behave differently when using different settings, sometimes for the good and often for the worse. Effects very similar to that found in fluid dynamics are seen at extreme densities, with no realistic movement in these cases. Nevertheless, fine-tuned settings allow for smoother crowd flow and a somewhat smarter movement when facing intersecting agents. Streamlines are formed naturally in most density settings, causing agents in the line to walk more freely towards its intended goal.

7.2 Groups

Groups have been successfully integrated into the simulation. Adaptations have been made to have group members stay adjacent to one another, but also diverge or change their walking pattern altogether when faced with obstructing forces. In general, having groups causes more but nevertheless negligible stress on the overall simulation performance. More dispersion is also seen when having groups in high density settings.

A user study has shown that groups are harder to perceive at certain densities. It is not determined whether this concerns high density scenarios. Data has however shown that there is more uncertainty in perceiving group settings at higher densities.

Bibliography

[1] F. Aurenhammer. Voronoi diagrams: a survey of a fundamental geometric data structure. *ACM Comput. Surv.*, 23(3):345–405, Sept. 1991. ISSN 0360-0300. doi: 10.1145/116873.116880. URL http://doi.acm.org/10.1145/116873.116880.

[2] O. B. Bayazit, J.-M. Lien, and N. M. Amato. Better group behaviors in complex environments using global roadmaps. In *Proceedings of the Eighth International Conference on Artificial Life*, ICAL 2003, pages 362–370, Cambridge, MA, USA, 2003. MIT Press. ISBN 0-262-69281-3. URL http://dl.acm.org/citation.cfm?id=860295.860352.

[3] J. Berglund and R. Ristic. High-density real-time virtual crowds via unilaterally incompressible fluid simulation, 2015.

[4] R. Bridson and M. Müller-Fischer. Fluid simulation: Siggraph 2007 course notes. In *ACM SIGGRAPH 2007 Courses*, SIGGRAPH '07, pages 1–81, New York, NY, USA, 2007. ACM. ISBN 978-1-4503-1823-5. doi: 10.1145/1281500.1281681. URL http://doi.acm.org/10.1145/1281500.1281681.

[5] M. R. Carretero, A. Qureshi, and C. Peters. Evaluating the perception of group emotion from full body movements in the context of virtual crowds. In *Proceedings of the ACM Symposium on Applied Perception*, SAP '14, pages 7–14, New York, NY, USA, 2014. ACM. ISBN 978-1-4503-3009-1. doi: 10.1145/2628257.2628266. URL http://doi.acm.org/10.1145/2628257.2628266.

[6] S. Chenney. Flow tiles. In *Proceedings of the 2004 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '04, pages 233–242, Aire-la-Ville, Switzerland, Switzerland, 2004. Eurographics Association. ISBN 3-905673-14-2. doi: 10.1145/1028523.1028553. URL http://dx.doi.org/10.1145/1028523.1028553.

[7] A. Date. Fluid dynamical view of pressure checkerboarding problem and smoothing pressure correction on meshes with colocated variables. *International Journal of Heat and Mass Transfer*, 46(25):4885 – 4898, 2003. ISSN 0017-9310. doi: http://dx.doi.org/10.1016/S0017-9310(03)00332-6. URL http://www.sciencedirect.com/science/article/pii/S0017931003003326.

[8] Z. Dostal. *Optimal Quadratic Programming Algorithms: With Applications to Variational Inequalities*. Springer Publishing Company, Incorporated, 1st edition, 2009. ISBN 0387848053, 9780387848051.

[9] Z. Dostal and J. Schoberl. Minimizing quadratic functions subject to bound constraints with the rate of convergence and finite termination. *Computational Optimization and Applications*, 30(1):23–43, 2005. ISSN 1573-2894. doi: 10.1023/B:COAP.0000049888.80264.25. URL http://dx.doi.org/10.1023/B:COAP.0000049888.80264.25.

[10] K. Erleben. Numerical methods for linear complementarity problems in physics-based animation. In *ACM SIGGRAPH 2013 Courses*, SIGGRAPH '13, pages 8:1–8:42, New York, NY, USA, 2013. ACM. ISBN 978-1-4503-2339-0. doi: 10.1145/2504435.2504443. URL http://doi.acm.org/10.1145/2504435.2504443.

[11] S. J. Guy, J. van den Berg, W. Liu, R. Lau, M. C. Lin, and D. Manocha. A statistical similarity measure for aggregate crowd dynamics. *ACM Trans. Graph.*, 31(6):190:1–190:11, Nov. 2012. ISSN 0730-0301. doi: 10.1145/2366145.2366209. URL http://doi.acm.org/10.1145/2366145.2366209.

[12] F. H. Harlow. *The particle-in-cell method for numerical solution of problems in fluid dynamics*. Mar 1962. doi: 10.2172/4769185. URL http://www.osti.gov/scitech/servlets/purl/4769185.

[13] D. Helbing and P. Molnár. Social force model for pedestrian dynamics. *Phys. Rev. E*, 51:4282–4286, May 1995. doi: 10.1103/PhysRevE.51.4282. URL http://link.aps.org/doi/10.1103/PhysRevE.51.4282.

[14] D. Helbing, A. Johansson, and H. Z. Al-Abideen. Crowd turbulence: the physics of crowd disasters. *ArXiv e-prints*, Aug. 2007.

[15] N. A. J Denny, S Thomas. MAPRM: A PRM planner that samples the medial axis of the free c-space. URL https://parasol.tamu.edu/groups/amatogroup/research/MAPRM/. Accessed 2017-03-20.

[16] T.-Y. Li, Y.-J. Jeng, and S.-I. Chang. Simulating virtual human crowds with a leader-follower model. In *Proceedings Computer Animation 2001. Fourteenth Conference on Computer Animation (Cat. No.01TH8596)*, pages 93–102, 2001. doi: 10.1109/CA.2001.982381.

[17] M. Mabrouk and C. McInnes. Solving the potential field local minimum problem using internal agent states. *Robotics and Autonomous Systems*, 56(12):1050 – 1060, 2008. ISSN 0921-8890. doi: http://dx.doi.org/10.1016/j.robot.2008.09.006. URL http://www.sciencedirect.com/science/article/pii/S0921889008001413. Towards Autonomous Robotic Systems 2008: Mobile Robotics in the {UK10th} British Conference on Mobile Robotics - Towards Autonomous Robotic Systems (TAROS 2007).

[18] M. Moussaïd, N. Perozo, S. Garnier, D. Helbing, and G. Theraulaz. The walking behaviour of pedestrian social groups and its impact on crowd dynamics. *PLOS ONE*, 5(4):1–7, 04 2010. doi: 10.1371/journal.pone.0010047. URL https://doi.org/10.1371/journal.pone.0010047.

[19] S. R. Musse, V. J. Cassol, and C. R. Jung. Towards a quantitative approach for comparing crowds. *Comput. Animat. Virtual Worlds*, 23(1):49–57, Feb. 2012. ISSN 1546-4261. doi: 10.1002/cav.1423. URL http://dx.doi.org/10.1002/cav.1423.

[20] R. Narain, A. Golas, S. Curtis, and M. C. Lin. Aggregate dynamics for dense crowd simulation. In *ACM SIGGRAPH Asia 2009 Papers*, SIGGRAPH Asia '09, pages 122:1–122:8, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-858-2. doi: 10.1145/1661412.1618468. URL http://doi.acm.org/10.1145/1661412.1618468.

[21] C. Peters and C. Ennis. Modeling groups of plausible virtual pedestrians. *IEEE Computer Graphics and Applications*, 29(4):54–63, July 2009. ISSN 0272-1716. doi: 10.1109/MCG.2009.69.

[22] C. W. Reynolds. Flocks, herds and schools: A distributed behavioral model. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '87, pages 25–34, New York, NY, USA, 1987. ACM. ISBN 0-89791-227-6. doi: 10.1145/37401.37406. URL http://doi.acm.org/10.1145/37401.37406.

[23] H. R.L. The flow of human crowds. *Annual Reviev of Fluid Mechanics*, 35:169–182, 2003.

[24] F. A. Rojas, H. S. Yang, and F. M. Tarnogol. Safe navigation of pedestrians in social groups in a virtual urban environment. In *2014 International Conference on Cyberworlds*, pages 31–38, Oct 2014. doi: 10.1109/CW.2014.13.

[25] H. Singh, R. Arter, L. Dodd, P. Langston, E. Lester, and J. Drury. Modelling subgroup behaviour in crowd dynamics DEM simulation. *Applied Mathematical Modelling*, 33(12):4408 – 4423, 2009. ISSN 0307-904X. doi: http://dx.doi.org/10.1016/j.apm.2009.03.020. URL http://www.sciencedirect.com/science/article/pii/S0307904X09000808.

[26] A. Skattman. Flock simulation of birds, 2010. URL http://www.csc.kth.se/utbildning/kandidatexjobb/datateknik/2010/rapport/skattman_udd_agnes_K10033.pdf. Accessed 2017-03-19.

[27] S. A. Stüvel, A. F. van der Stappen, and A. Egges. Perception of collisions between virtual characters. *Computer Animation and Virtual Worlds*, pages n/a–n/a, 2016. ISSN 1546-427X. doi: 10.1002/cav.1728. URL http://dx.doi.org/10.1002/cav.1728. cav.1728.

[28] A. Sud, R. Gayle, E. Andersen, S. Guy, M. Lin, and D. Manocha. Real-time navigation of independent agents using adaptive roadmaps. In *Proceedings of the 2007 ACM Symposium on Virtual Reality Software and Technology*, VRST '07, pages 99–106, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-863-3. doi: 10.1145/1315184.1315201. URL http://doi.acm.org/10.1145/1315184.1315201.

[29] A. Treuille, S. Cooper, and Z. Popović. Continuum crowds. *ACM Trans. Graph.*, 25(3):1160–1168, July 2006. ISSN 0730-0301. doi: 10.1145/1141911.1142008. URL http://doi.acm.org/10.1145/1141911.1142008.

[30] J. Van Berg, M. Lin, and D. Manocha. *Reciprocal velocity obstacles for real-time multi-agent navigation*, pages 1928–1935. 2008. ISBN 9781424416479. doi: 10.1109/ROBOT.2008.4543489.

[31] H. Wang, J. Ondřej, and C. O'Sullivan. Path patterns: Analyzing and comparing real and simulated crowds. In *Proceedings of the 20th ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, I3D '16, pages 49–57, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-4043-4. doi: 10.1145/2856400.2856410. URL http://doi.acm.org/10.1145/2856400.2856410.

[32] X. Wei, W. Lv, W. Song, and X. Li. Survey study and experimental investigation on the local behavior of pedestrian groups. *Complexity*, 20(6):87–97, 2015. ISSN 1099-0526. doi: 10.1002/cplx.21633. URL http://dx.doi.org/10.1002/cplx.21633.

[33] S. A. Wilmarth, N. M. Amato, and P. F. Stiller. Maprm: a probabilistic roadmap planner with sampling on the medial axis of the free space. In *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No.99CH36288C)*, volume 2, pages 1024–1031 vol.2, 1999. doi: 10.1109/ROBOT.1999.772448.

Appendix A

MPRGP Functions

Algorithm A.1. g

Input & Initialization:
A ← *Symmetric positive definite matrix of order n*
b ← *Vector of order n*
vec ← *Vector of order n*
return $Avec + b$

Algorithm A.2. φ

Input & Initialization:
A ← *Symmetric positive definite matrix of order n*
b ← *Vector of order n*
l ← *Vector of order n*
vec ← *Vector of order n*
G ← $g(A, b, vec)$
tempPhi ← *Empty vector of order n*
for *i* = *1 to* n **do**
    if $vec[i] == l[i]$ **then** $tempPhi[i] = 0.0$
    else$tempPhi[i] = G[i]$
    end if
end for
return tempPhi

Algorithm A.3. β

Input & Initialization:
A ← *Symmetric positive definite matrix of order n*
b ← *Vector of order n*
l ← *Vector of order n*
vec ← *Vector of order n*
G ← $g(A, b, vec)$
tempB ← *Empty vector of order n*
for *i* = *1 to* n **do**
  if $vec[i] == l[i]$ **then** $tempB[i] = min(G[i], 0.0)$
  else$tempB[i] = 0.0$
  end if
end for
return tempB

Algorithm A.4. v

Input & Initialization:
A ← *Symmetric positive definite matrix of order n*
b ← *Vector of order n*
l ← *Vector of order n*
vec ← *Vector of order n*
G ← $g(A, b, vec)$
return $\varphi(A, b, l, vec) + B(A, b, l, vec)$

Algorithm A.5. $\tilde{\varphi}$

Input & Initialization:
A ← *Symmetric positive definite matrix of order n*
b ← *Vector of order n*
l ← *Vector of order n*
vec ← *Vector of order n*
G ← $g(A, b, vec)$
tempPhiTilde ← *Vector of order n*
α ← *Decimal value*
phi ← $\phi(A, b, l, vec)$
for *i* = *1 to* n **do**
  tempPhiTilde[i] = $\frac{min(vec[i]-l[i]}{\alpha}, phi[i])$
end for
return tempPhiTilde

Algorithm A.6. P_{Ω_B}

Input & Initialization:
l $\leftarrow$ *Vector of order n*
vec $\leftarrow$ *Vector of order n*
diff $\leftarrow vec - l$
for *i* = *1 to* n **do**
    diff[i] = $max(diff[i], 0.0)$
end for
return l + diff