A comparison of crowd simulation techniques.

Submitted in partial fulfilment
of the requirements of the degree of

BACHELOR OF SCIENCE (HONOURS)

of Rhodes University

Author: Matthew Funcke. Supervisor: Professor George Wells.

Grahamstown, South Africa
30/10/2012

Abstract

There is a great deal of literature available on the topic of crowd simulation, but far less focusing on model comparisons. This paper takes the most prominent models, compares them to one another, and then makes generalisations based on the results. Models discussed include Cellular Automata models, the Social Forces Model, Rule-Based models and Hybrid models. The primary goal of this paper is to determine what model types are best suited to what applications. It was found that cellular automata are best suited to teaching, social forces models generally give the most accurate results, rule based models are most suited to applications with a heavy focus on graphics and hybrid models(both commercial and non-commercial) tend to generally do well for all applications. Additionally direct relationships between model cost, complexity and simulation accuracy were found.

Acknowledgements

First and foremost I would like to thank my fiance for her continual support while working on this project, especially when I felt like I simply couldn't keep going. I would also like to thank my supervisor for his valuable advise and opinions. Thanks also to Professor Andreas Schadschneider, for valuable pointers regarding several aspects of the paper. Finally I would like to acknowledge the financial and technical support of Telkom, Tellabs, Stortech, Genband, Easttel, Bright Ideas 39 and THRIP through the Telkom Centre of Excellence in the Department of Computer Science at Rhodes University.

CONTENTS:

1. Introduction

The psychology and behaviour of crowds of people has been a topic of study for a long time[1], in part due to the interesting emergent behaviour that gets exhibited by calm, agitated and even stampeding crowds of people[2]. For example, in high densities crowds tend to form bi-directional lanes to maximise flow speed; and in certain situations, effects such as fear and agitation can propagate from person to person with only a few people knowing what the original danger actually was.

In the field of computer science, crowd simulations have been the subject of studies for over 30 years due to several factors, including: crowd data (density, speed and direction) is all empirical and therefore makes comparing simulations to real world scenarios easier and more accurate; there are interesting physical similarities between crowd behaviour and both gas and fluid dynamics; a large quantity of data already exists from psychological studies that can be used for comparisons; finally, and most importantly, there are a large number of real world applications for crowd simulation[3].

These practical and academic applications include: evacuation simulations to determine the safety of a facility[1, 4, 5] and maximise the efficiency of its evacuation[1, 5, 6], architectural planning and optimization[3, 6], training programs[1, 5], pedestrian management systems for guiding real-time crowds such as in airports[7], generation of realistic crowds for games[5, 8, 9] and movies[5, 8], psychological studies[9], risk evaluation for insurance companies[6] and more. One of the more unusual uses of virtual crowds is as an audio manipulation tool, a sound is given to a group of agents who then echo it back to the user with variations dependent on their positions[10].

One might ask the question "why not simply use real world people for the various applications, rather than simulations which might not be as accurate?". This is a valid question, however there are several reasons why the use of real people is often infeasible as explained by Klüpfel et al.[11]: there are multiple ethical, financial, and logical constraints on performing a real world evacuation exercise, for example you can not endanger the test subjects therefore the results obtained would only show evacuation results under ideal circumstances, which in turn might result in inaccurate data. Other constraints include large populations and testing of unbuilt structures where a simulation would be the only option. Figure 1 shows the limitations of real-world exercises and how with a few assumptions a simulation can be as accurate as a real world crowd in an emergency[11]. Zhou et al.[12] says that "in general, pure mathematical approaches or analytic models

are not adequate in characterizing the dynamics of a crowd", which agrees with the points made by Klüpfel et al.[11].

The goal of this paper is to compare the various techniques used to simulate crowds in order to determine what techniques are best suited to what applications, for example a cellular automaton would be more suited to teaching programming logic than creating high definition CGI crowds for movies, while the inverse would be true for a complex commercial hybrid model.

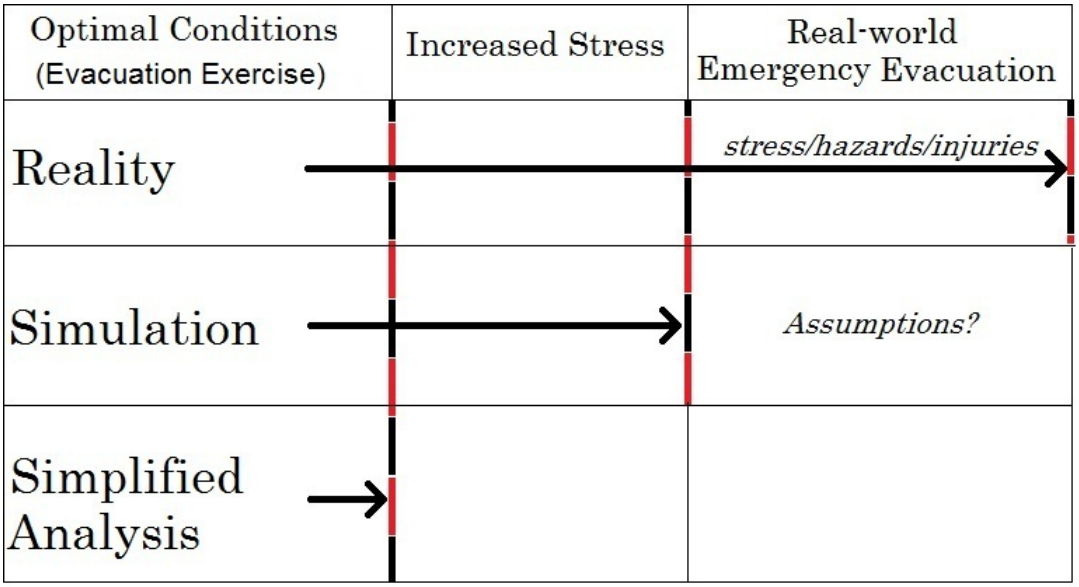

Figure 1: Demonstration of the limitations of real-world crowd exercises as illustrated by Klüpfel et al.[11]This figure shows that a real world exercise (due to the lack of realistic stresses and hazards) is only as good as a simplified mathematical model in terms of accuracy.

2. Literature Review

Due to the survey style nature of this paper the following section is by far the largest, as most of the later results rely heavily on the related literature. Section 2.1 defines certain terms so as to avoid ambiguity later. Section 2.2 explains some examples of work similar to this: where crowd simulation models have been compared. In 2.3 various factors are listed that have been used by other authors to compare and even score crowd simulation models and techniques, examples of these factors are: scalability, cost and realism. The different methods of classifying models are listed in section 2.4 along with summaries of certain specific models, this is the largest and most important section. Section 2.5 explains work done by other authors that compares models but which does not focus on grading them in any way. The final section in the literature review summarizes proposed techniques for analysing crowds, both simulated and real.

2.1 Preliminary Definitions

For the sake of clarity it is necessary to define certain terms such as crowd, mob and panic.

The terms mob and panic are both highly controversial terms in the field of crowd simulation, evacuation analysis and safety science in general [13, 14, 15]. According to Keating[16] and Klingsch et al.[17] true 'panic' only occurs in less than 10% of all cases. In this paper when the word 'panic', is used, it will be referring to crowds that are fearful or agitated, rather than truly panicked, unless otherwise stated.

Figure 2 shows the classification of crowds according to Forsyth[18] and extended by Klüpfel et al.[11]. The only 'mob' that this paper will be dealing with is a crowd attempting to evacuate a building.

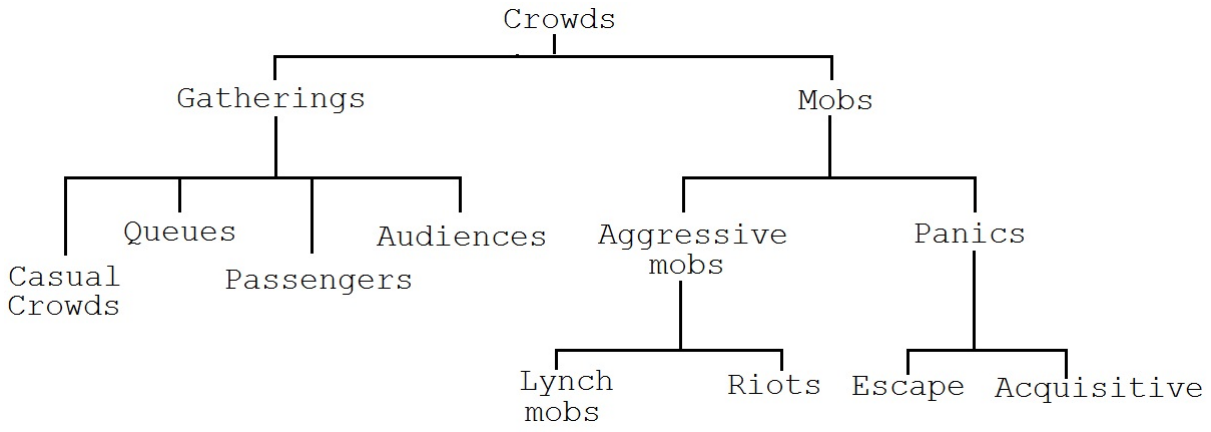

Figure 2: Classification of crowds according to Forsyth[18] and extended by Klüpfel et al.[11].

2.2 Similar Work.

Rogsch and Klingsch 2012[19] establish a guide to evaluating *commercial* models based on these conditions:

- Agent movement (overtaking, acceleration, and deceleration).
- Agent movement around corners.
- Potential fields (how agents approach an exit).
- Path selection.
- Exit selection.
- Velocity-density relationship.
- Update schemes.

They also show the results of a few comparisons between the commercial models: Exodus, Aseri, Simulex and PedGo; figure 3 shows one of their evaluations. However their work is meant as a guide to allow end-users to establish what fundamental algorithm a model uses rather than its appropriateness for a particular task. Thus no scores are given to individual models. As their work is primarily focused on commercial software and algorithm identification it has only limited applicability to the work done in this paper.

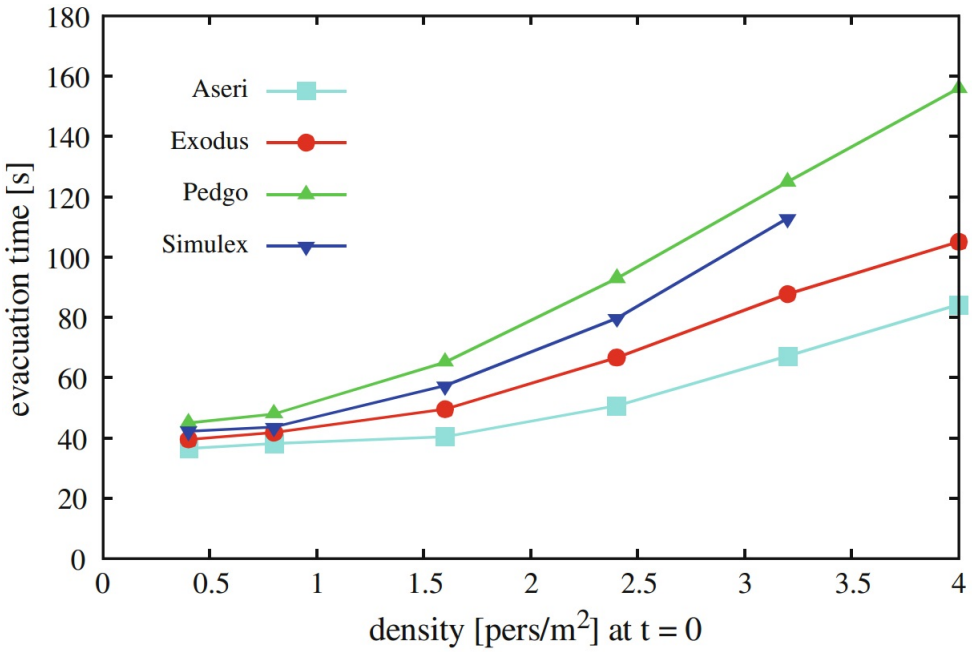

Figure 3: Comparison between four commercial models (Exodus, Aseri, Simulex and PedGo) simulating an evacuation in a 2m wide corridor. As performed by Rogsch and Klingsch 2012[19].

Zhou et al.[12] performed a comparison of crowd simulation techniques with a different focus to both this paper and to that by Rogsch and Klingsch 2012[19]: instead of scoring and grading individual models they establish a technique to classify models, which will be further explained in Section 2.4.

2.3 Comparison Factors Between Models

Each application of crowd simulation technology values certain aspects more than others. For example a safety evaluation consultant will value accurate agent behaviour more than realistic 3D graphics while the converse might be true for a Hollywood CGI team. Here is a list of the major factors to consider according (primarily) to Castle [20] that need to be taken into consideration when determining the suitability of a model for a particular application:

- Accuracy of agent behaviour.
- Emergent behaviours such as lane formation, the edge effect, or shock-waves. As described by Fell [2].
- Have the results of the models simulations been validated in any way.
- Are simulations real-time?
- How scalable is the model (can it support hundreds, thousands or hundreds of thousands of agents.)
- Financial cost to obtain a license or implement the model.
- Time requirements or costs to setup or implement the model.
- Availability of software (download-able, off-the-shelf, on a consultancy basis through the developer).
- What hardware requirements does the model specify (High end graphics card, large hard drive requirements, lots of RAM, multiple parallel processors).
- What operating systems can it run on (Linux, windows, Mac OS X).
- Does the model work for 2D or 3D environments?
- What format (Image file, GIS, CAD, etc.) does the model require in order to run.
- Graphic quality of agent/crowd representation.
- How are agents represented logically (each agent is unique, agent are all identical, groups of people rather than individuals agents are considered etc.)
- What environmental information is available to each agent (the whole environment, a field of vision, nothing but what the currently occupied cell tells the agent etc.)
- How are agents movements limited (is speed set at a constant rate, can individuals move at different speeds, can agents fall and obstacles etc.)
- Ease of use / Learning curve (Is it simple click and drag or is training needed?)
- What was the original purpose of the software or model.
- Is there support available for the model or software.

Due to the large number of factors that need to be considered and the limited scale of this project only certain factors will be taken into consideration. In the discussion section the selected factors will be analysed in terms of each model, when possible, in order to determine what function/s each model is suitable for.

Zhou et al.[12] suggest four criteria for the evaluation of models, both from the perspective of the developer and the end user: flexibility, extensibility, execution efficiency and scalability, the use of these will be discussed in more detail in the 3.3 of the discussion section.

2.4 The Fundamental Architectures and Classifications

Many, many models exist for simulating crowds [21, 22], for an extensive(but still incomplete) list of the available crowd simulation models see appendix 1, table 14. It would be impossible to analyse each of these models individually or even to read all the literature available on each one, thus this work will focus on the more common models. There are three basic model architectures that most literature agrees upon as being the foundation of most models, these fundamental architectures are: Cellular automata (CA), social forces models, and rule based models[23, 24].

All of the literature reviewed appears to regard cellular automata as one of the most basic methods of simulating crowds, additionally there are appear to be more cellular automata models available than any other type, potential reasons for this are discussed later . Cellular automata are defined by Keßel et al.[6], Burstedde et al.[25], and Pelechano [5] as a model where both the environment and the contained agents are discretised in space, time and the state variable of the agent or cell.

Helbing [3] suggests an alternative fundamental model, the Social forces model. This model proposes that both real and simulated humans are subject to certain "social forces", for example we are repelled by strangers and walls, and attracted by friends and our destinations. Helbing's model takes these forces, converts them into vector quantities and sums them up in order to determine the direction that an agent will move in a continuous (non-discretised) space.

Reynolds [26] proposed a rule based model, which works not only on human simulations but also on groups of animals such as flocks of birds and schools of fishes. This model works by applying distancing rules between agents so that they try to keep a constant/comfortable distance between one another.

Not all literature agrees with this breakdown, for example Parent[27] breaks down systems into being particle, flocking or behavioural systems as seen in table 1. When examined more closely it can be seen that this break down is in fact similar to the one given above where: flocking system equate to rule based models, particle systems equate the social forces model and behavioural systems partially equate to cellular automata.

All of the basic models have areas in which they can be improved, either in realism or behavioural accuracy [23], this has lead to multiple enhancements and alterations being made by multiple parties, such as [4, 8, 9, 22] to name a few. A more in depth look at each of these implementations follows.

Table 1: Classification of models according to [27]

Aspects	Particle Systems	Flocking Systems	Behavioural Systems
Structure	Non-hierarchical	Levels: flocks, agents	Can present hierarchy
Participants	many	some	few
Intelligence	none	some	high
Physics-based	yes	some	no
Collision	Detect & respond	avoidance	avoidance
Control	Force field, global tendency	Local Tendency	rules

Zhou et al.[12] propose a technique for classifying crowd models rather than real world crowds: their technique is based on four factors, the size of the crowd being simulated and the time-scale of the crowd phenomena concerned are the primary factors, while the model approach and behavioural factors being considered are the secondary factors. Figure 4 shows the classification technique proposed by Zhou et al.[12].

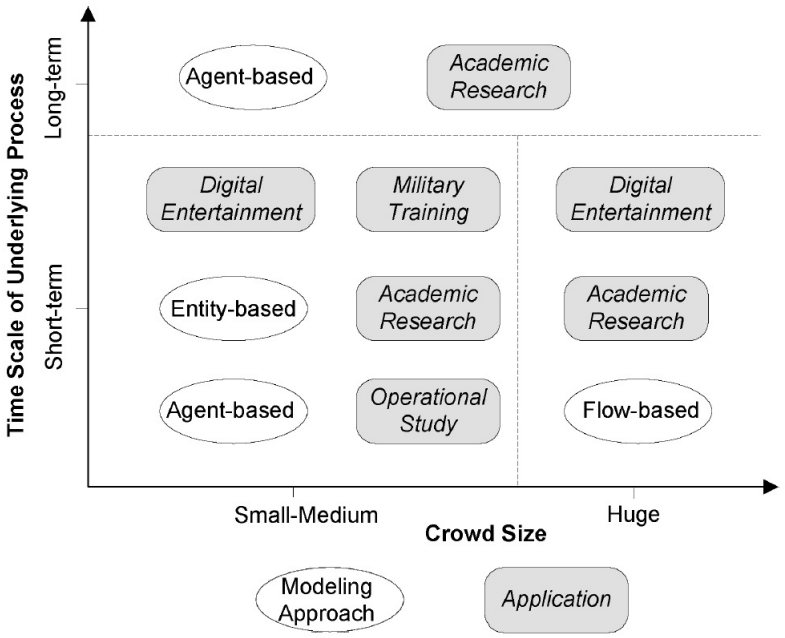

Figure 4: Classification of crowds according to Zhou et al.[12] Small crowds contain tens of agents, medium crowds contains hundreds of agent, and huge crowds contain thousands of agents or more. Time scale varies from a few minutes to several hours based on the phenomenon concerned: for example a short-term time-scale phenomenon could be concerned with agents' reactions to a threat, while a long-term phenomenon could be the spreading of an opinion through a crowd.

There are still other methods by which models get classified. For example Pelechano et al.[22], Keßel et al.[6] and Wagoum et al. [28]classify models as being microscopic or macroscopic in focus, where macroscopic models focus on the system as a whole rather than on individual agents and microscopic models which focus on the behaviour of individual agents in order to create more realistic agent autonomy. Musse et al.[1]classifies models as either being focused on accurate behaviour or on realistic graphics, there are however certain models which can do both [23, 29, 30]. According to Wagoum et al. [28] cellular automata models, rule-based models, social forces models, agent-based models and hybrid models are all generally macroscopic in focus.

Several models do not fit with any of the above classifications, and instead take aspects of multiple models and attempt to combine them [22], for some examples see the Behaviour Methods column of table 2 . These hybrid models are often complex or expensive but usually succeed in creating highly realistic simulations. The hybrid models that this study elaborates on are Massive[29], MassMotion[30] and HiDAC[23].

2.4.1 Cellular Automata Models

2.4.1.1 Floor Field Model, Burstedde et al. 2001[25]

The cellular automaton created by Burstedde et al.[25] utilized a unique method to determine what direction an agent should move on the 2D grid of the world: every cell of the grid contained a static floor field and a dynamic floor field. The static floor field would remain unchanged regardless of the situation and would simply become "more attractive" the closer to an exit the cell was. The dynamic floor field changed value with both time and how many people travel over it, the more agents that travel over a particular path to an exit the more attractive that path becomes to other agents.

This alteration to the standard cellular automaton led to three different types of behaviour: when agents relied more on the dynamic floor variable their behaviour would become more disordered; when agents relied equally on the dynamic and static floor field to navigate, cooperative emerged between agents; however when the agent followed the static field more than anything else the behaviour became deterministic. It is also important to note that this model updates the positions of each agent serially rather than in parallel, thus competition over a space never occurs, for this reason the model is limited in its realism when simulating evacuations.

The floor fields model does demonstrate some emergent behaviours, for example figure 5 demonstrates lane formation. As agents rely on the floor fields to determine their behaviour, individuality and uniqueness are not possible using this model which leads to a more macroscopic perspective when using this model. Wagoum et al.[28] refer the floor fields model as "a more sophisticated approach" to cellular automata, "that leads to more realistic results".

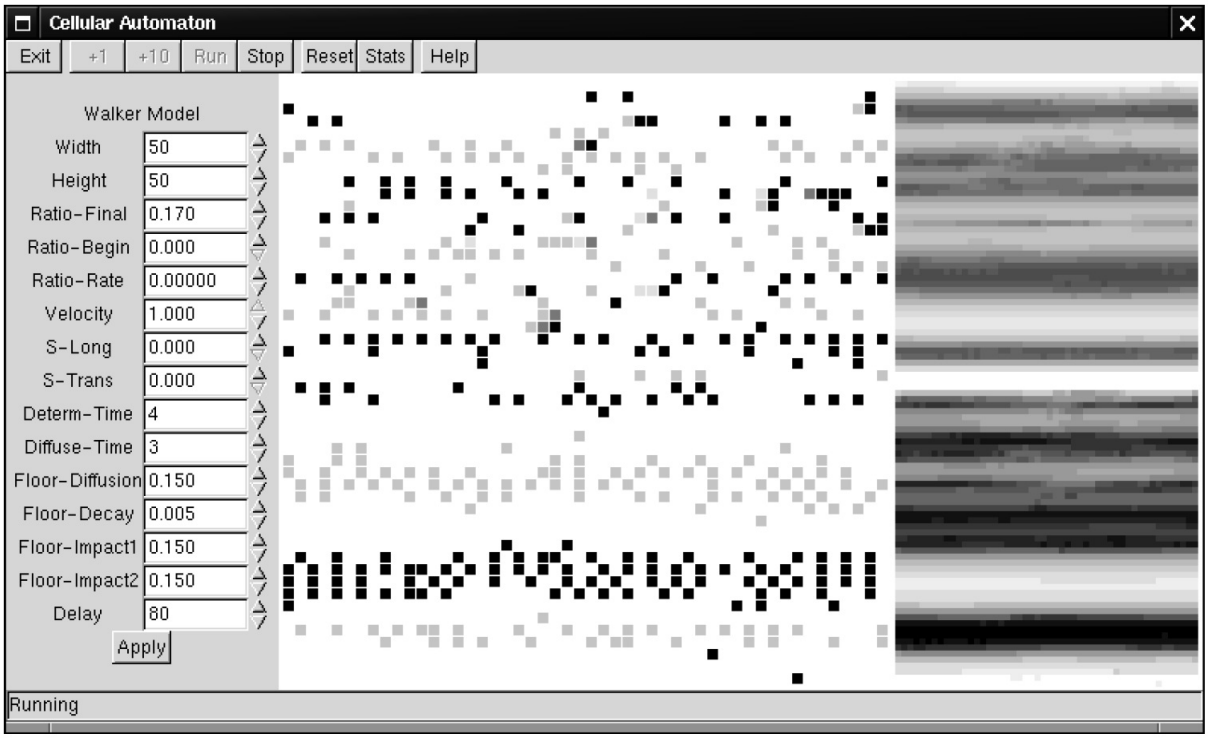

Figure 5: A screen-shot of the floor fields model demonstrating lane formation between agents moving left(black) and right(grey) down a corridor.

Burstedde et al. briefly mention the number of updates per second that the floor fields model can handle as being 0.24 million updates per second. By assuming that a good number of updates per agent is 30 per second, one can conclude that their model would easily handle crowds of around 8000 agents.

2.4.1.2 Tecchia et al. 2001 (ABS) [8]

The Agent Behaviour Simulator (ABS) by Tecchia [8], does not fit perfectly into any one category and *could* be considered a hybrid. However it discretises time, space and agent placement logically, while graphically each agent has a position within the cell it occupies. As previously explained cellular automata rely on discrete space and time while social forces and rule based models treat space as continuous. Therefore the ABS model will be treated logically as an advanced cellular automaton due to discretisation of space and time, but graphically as though the agents were being represented in a real 3D world.

It uses a four layered behavioural approach to create and display, in real time, in excess of 1000 agents at once (at 5000 or more agents the fps drop). The four behaviour layers each

determine different levels of behavioural complexity: the first layer simply says whether a cell is vacant, while higher layers determine things such as attractions in the environments or complex behaviours like waiting for a bus. Each agent is represented by a 3D triangle and direction on a 2D grid, however while the agents are each given a discrete cell to occupy, they can position themselves at different locations inside their cell, see figure 6.

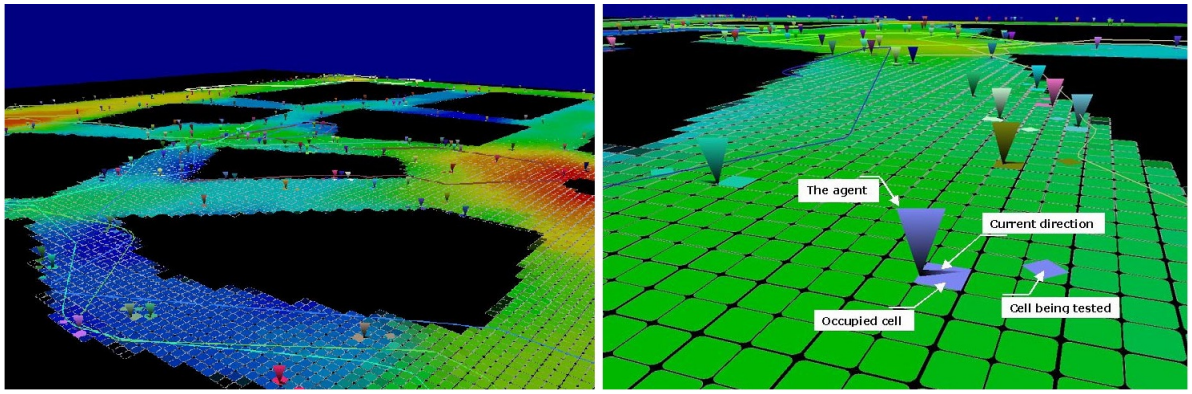

Figure 6: Demonstration of Tecchias ABS system and the non-discrete placement of agents within their discrete cells. Images courtesy of [8].

2.4.1.3 Keßel et al. 2002

Keßel [6] took the standard cellular automaton and further limited what agents were allowed to do by never allowing an agent to move backwards. This limitation was imposed because the primary focus of the work was on how crowds of different densities behave in corridors of various widths and under what conditions the simulation becomes jammed. Forward motion was made more probable than motion from side to side.

This combination of simple rules and restrictions produced emergent behaviours such as bi-directional lane formation and the edge effect (agents moving faster at the edges). As an aside it also demonstrates that certain mathematical models for agent behaviour can and do stop working in extreme situations, such as 1 cell wide corridors. Keßels model is able to handle dense crowds of "individual" agents, however there is no uniqueness between each agents, for example all the agents move at the same speed and with the same destination in mind. Also as it is a bare minimum cellular automaton it has only very simple graphical capabilities.

2.4.1.4 Extended Floor Field Model, Kirchner et al. 2003

Kirchner et al. [4] takes an already unusual cellular automaton, that created by Burstedde et al. 2001[25], and enhances it to determine if the accuracy of the simulation under evacuation conditions can be increased. This enhancement can be summarized as the addition of a friction variable, which caused contention over empty cells that two agents both wanted to occupy.

Kirchner believed that while the original model was a good general purpose simulator it did not perform realistically when simulating panicked or evacuating crowds: due to the cooperation between agents, venues would be evacuated in optimal times and there would be no tension between agents attempting to exit through the same door.

In order to make this model function better when simulating evacuations Kirchner took the cellular automaton model created by Burstedde et al. 2001 [25], made the agents update their positions in parallel and added a "friction" variable which appeared to increase the accuracy of agent behaviour when the crowds being simulated became dense, by causing contention between which agent would occupy an empty cell. According to Kirchner the addition of the friction variable causes agents to exhibit behaviour "observed in panic situations and also in simulations using the social forces model".

Kirchner et al. note that the variance in their results when simulating evacuations is significant in terms of time from fastest to slowest evacuation, and thus the average evacuation time may in fact not be a meaningful quantity. Regardless this extension of the floor fields model allows for more realistic simulations, especially for situations that require dense crowds.

Further variations of the Floor Fields model (such as F.A.S.T.) exist, however due to the scope of this paper they will not be elaborated on.

Kirchner, Burstedde and Keßels' models each implement movement differently, more specifically regarding how many directions an agent should be able to move. Kirchners extended Floor Fields model [4] says that cellular automata should be able to move in 4 directions (left, right, up, and down) whereas the original Floor Fields model[25] says that agents should be able to move in 8 directions. Keßels model [6] does not allow agents to move backwards and thus only allows them forward, left and right steps, which is particularly useful for simulating the behaviour of crowds in a bi-directional corridor. Each set of movement rules has its advantages and disadvantages, for example the two

reasons why the Floor Fields model uses four directions instead of eight is that four directions simplifies things and eight directions showed no significant effects on the nature of observed phenomena when compared to four directions[31]. According to Wagoum et al[28] having either four or eight possible directions has very little effect on the final results. See Figure 7 for an illustration of the different cellular automata motion rules.

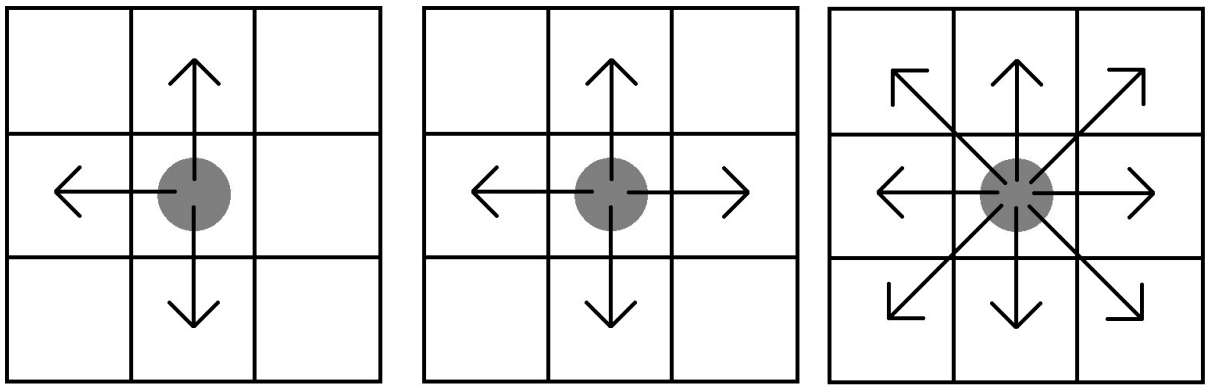

Figure 7: From left to right, a comparison of cell movement between Keßel, Kirchner et al. and Bursttedde et al. models respectively.

Most cellular automata, despite being limited to behaviour on a 2D grid, can in theory be enhanced to work in '3D' environments using stairs and floors as explained by Musse et al. [1]. This could potentially make cellular automata models more useful when analysing crowd movement in a 3D space.

2.4.2 Social Forces Models

2.4.2.1 Helbing and Molnar 1995

While this is one of the fundamental models it is important to understand it in slightly greater detail due to the number of enhancement and alterations that have been added to it and due to several similarities and differences that show up in more advanced models.

In Helbings basic Social Forces Model [3] agents are represented as particles directed by a velocity vector. This velocity vector is determined by combining several equations that convert environmental conditions into speed and direction values. These environmental conditions include repulsive forces from 'stranger' agents and barriers, attractive forces from 'friend' agents, shop windows and the agents desired destination.

Helbing suggests that both real life decisions and agent decisions can be broken down into 'instinctive' or automatic decisions that do not require much thought which are usually predictable, and into complex behaviours/decisions that are harder to simulate. The social forces model focuses on the more instinctive decisions that agents have to make, and does not attempt to include more complex behaviour such as waiting at a bus stop.

This division of simple and complex decision making should not be confused with the brain/body breakdown as used by Maïm et al. [9] which broke the brain (and thus *all* decisions) away from the graphics of the body.

Despite agent behaviour being governed by a single (albeit complex) equation, emergent behaviours still manifest: during simulation agents form lanes and when two groups try to pass in opposite directions through the same doorway a fairly accurate representation of reality occurs where pressure on one side builds up until its agents are forced through, demonstration of these effects can be seen in figure 8. While not explicitly stated by Helbing, his model also allows for manifestation of the edge effect, where people near the edges of a crowd or corridor tend to move faster. This is due to barriers being repellant thus only agents who want to move quickly are forced closer to the barrier as a 'fast lane'.

The model does not elaborate on anything graphical and it would appear as if the primary focus is on behaviour, this does not mean that the model could not be applied to a 3d space, but instead that many additional features would have to be added in order to make it "look good". However Pelechano et al.[22] notes that the realism is somewhat decreased due to agents appearing to vibrate when crowds become dense due to their particle nature. This particle shaking effect is not limited to only the social forces model as shown in table 2.

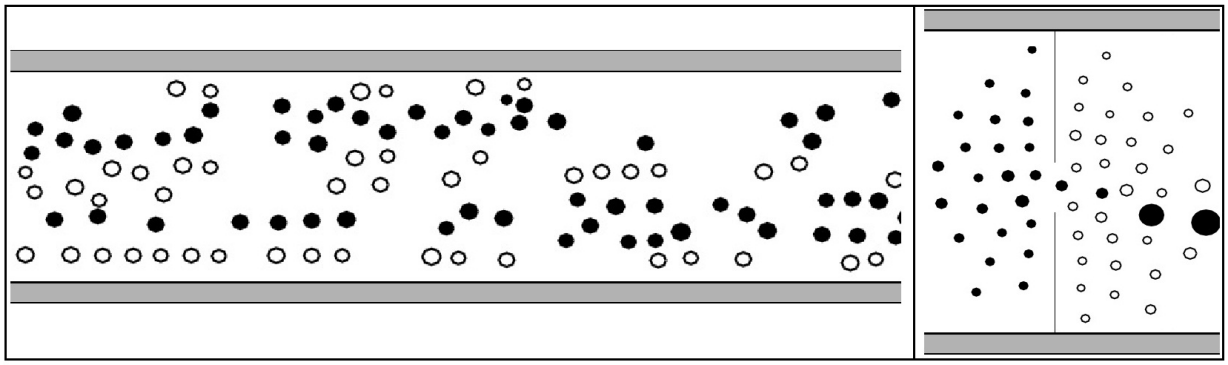

Figure 8: Emergent behaviours of agents in a corridor as shown by Helbing in the Social Forces Model: lanes forming (left), doors clogging (right). Full circles are agents moving from left to right, while empty circles are agents moving from right to left, the size of the circles is relative to the speed of the agents.

2.4.2.2 Quinn et al. 2003 [24]

This model is an enhancement of the original social forces model by Helbing and Molnar [3] which aims to re-create the social forces model on a large scale while allowing for real-time running and graphical rendering. Just like the original social forces model all agents are autonomous and unique, in order to add to the realism, agent data was assigned using crowd data trends gathered by [32]. Unfortunately nothing has been done in this implementation to rectify the vibrations exhibited by agents in dense crowds as explained by both [23] and [33]. The primary difference between this model and the original social forces model is in its parallelisation, by spreading the computational load between one manager process and ten worker processes this model is able to update the locations of 10,000 unique agents nearly fifty time a seconds. This level of parallelism is achieved by using "parallel computers constructed out of commodity microprocessors", which means that while the hardware could be considered high-end it is still widely available and relatively low cost.

The use of the Message Passing Interface (MPI) library adds to both the parallelism, and the cross platform compatibility. The authors of this model also intended it to be highly user friendly by allowing real-time alterations to terrain and obstacles via a simple user interface.

The implementation explained by [24] aims to produce both a 2D and 3D version, for top down control and immersive training respectively. However as the 3D representation is

listed as proposed future work, one will have to assume that the model is only available in its 2D implementation for comparative purposes. Figure 9 shows the scale and 2D graphics of this model: each black dot is a single person, everyone is attempting to evacuate through the centre.

Figure 9: Highly parallel social forces model [24] demonstrating the evacuation of 10,000 people from a 1.05km long airport terminal.

2.4.3 Rule Based Models

2.4.3.1 Reynolds 1987 & 1999 [26, 34]

Reynolds [26] described one of the earliest rule based models, he explains that his rule based model is essentially an extended particle model. He describes rule-based agents as following a hierarchy of motion, as shown in figure 10. Each section of the hierarchy should be independent of the other sections, and each one should be governed by its own rules. This allows for independent sets of rules to be easily exchanged with one another[34].

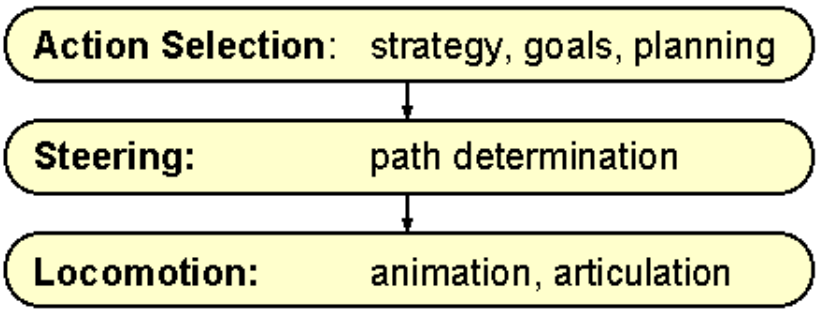

Figure 10: Hierarchy of motion for rule-based agent as described by Reynolds[34].

Various objectives are defined for individual agents such as flee, seek, pursue, follow, wander, explore and evade. These objectives determine what rules will be followed by an agent regarding other agents in the environment. For example an agent with the objective of following other 'friend' agents might use the associated 'follow' rule when it encounters

a 'friend', thus said agent would attempt to maintain a constant distant from the other agents while traveling in the same direction. An alternative example might be one agent seeing another 'hostile' agent, its objective could then be flee or evade, which in turn would cause the first agent to create as much distance between the hostile agent and itself as possible. Figure 11 shows an example of an agents potential steering behaviour under different circumstances.

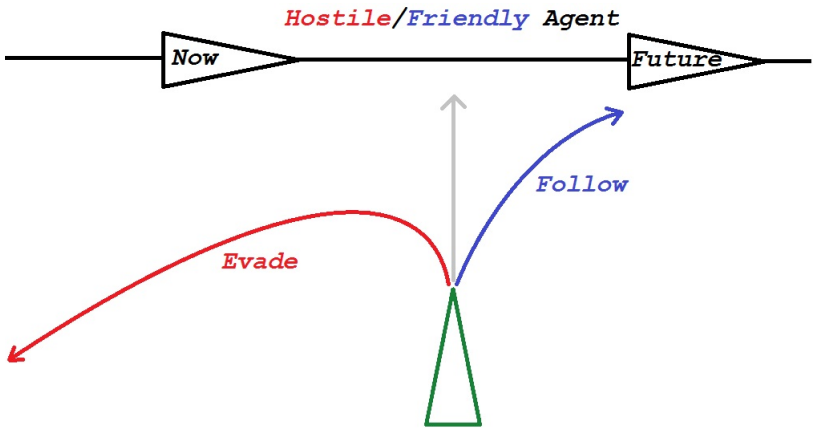

Figure 11: Rule-based agents potential steering behaviours[34], the grey arrow is its current trajectory.

This kind of steering behaviour results in realistic movements in crowds of low to medium densities[22], but when crowds become dense clipping issues between agents arise resulting in unrealistic graphics and unpredictable behaviours[23]. Inter-agent contact is not possible with this kind of model thus behaviours such as pushing are also not possible, which might decrease the accuracy of evacuations or similar simulations.

No mention is made by Reynolds[26, 34] as to the number of agents that can be simulated at once using this model, however algorithmic scalability is explained as varying from $O(n)$ to $O(n^2)$[26], for the purposes of scoring this model the best case scenario will be assumed.

2.4.3.2 OpenSteer 2007

As shown by table 2 there are in fact not a large number of crowd simulation models that rely purely on rule-based behaviour. Therefore rather than elaborating on a specific model or method for creating rule-based systems this section will explain OpenSteer.

OpenSteer [35] by itself is in fact not a stand alone model for rule-based crowd simulation, but is instead an "open source library of components to help build steering behaviours". This means that OpenSteer can be used to build rule-based models of varying complexities by providing pre-made methods to accomplish agent tasks such as Wander, Seek, Flee, Follow Path, Avoid Obstacle and more.

One of the most attractive features of OpenSteer is its open source nature. While some models provide working software [29, 30] and others provide the theoretical knowledge required in order to re-make the model[3, 23], these methods do not allow you take a look at the working logic behind the model. This makes OpenSteer particularly attractive as a teaching tool or for any task that is on a tight budget.

The downside to OpenSteer is that "non-programmers" may have trouble interacting with it even in simple ways such as integrating it into existing software [35]. Additionally the results obtained when using this 'behaviour-rule library' will differ depending on how it is used, for example: when used poorly no emergent behaviours may occur and the system may scale poorly, while the converse could be true if the system utilised OpenSteer to its full potential. In theory OpenSteer could be combined with fuzzy logic and result in a model that would rival Massive, thus for the purposes of scoring later, it will be assumed that OpenSteer is being used to make an 'average' rule-based model. Figure 12 shows an example of the OpenSteer library being combined with another model.

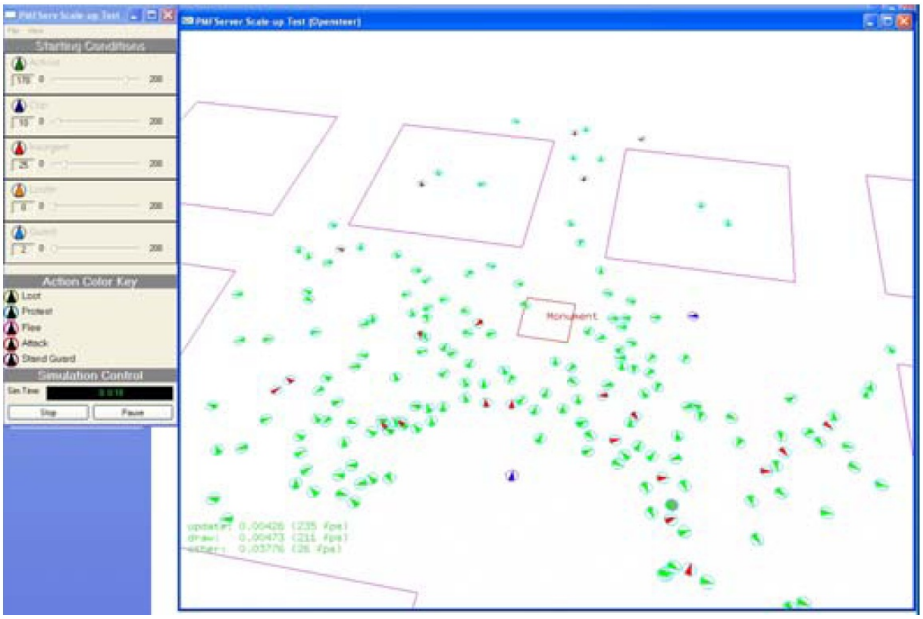

Figure 12: A protest scene being simulated using a combination of OpenSteer[35] and PMFServ[36, 37], as shown by Pelechano [5].

2.4.4 Hybrid and More Advanced Models

2.4.4.2 Pelechano et al. 2007 (HiDAC) [23]

Like the ABS model[8] the HiDAC model divides behavioural decisions based on their complexity, however while ABS [8] breaks behaviours into four layers of complexity the HiDAC model simply uses two: high-level behaviour and low-level behaviour. This division of behaviour is similar to that discussed by Helbing and Molnar[3] as being 'instinctive' or automatic (low-level) and complex behaviours/decisions (high-level). Maïm et al.[22] notes that the HiDAC model is an extended social forces model and aims to rectify some of the major drawbacks of said model as well as add new features. For example:

- Elimination of the 'vibration' of agents at high densities.
- Environmental perception at reasonable speeds.
- Formation of more natural bi-directional lanes (although these do form in the original Social Forces Model).
- Crowd and agent impatience.
- Pushing behavior between impatient or panicked agents.
- Agents falling and becoming obstacles when pushed too hard.
- Queuing behaviour near doors and gateways where bottle-necking may occur.
- Panic propagating realistically from agent to agent.

Many of these enhancements are achieved by giving agents unique personalities and statistics: for example level of impatience, level of panic, and comfort zones.

In order to rectify the shaking behaviour of the social forces model, Maïm et al.[23] implements a "stopping rule", when crowds paths are blocked in the desired direction agents will stop attempting to move in that direction until a personality dependent timer runs out. Once the timer expires the agent will try and move in the same direction again.

In order to allow for the formation of queues in normal (non-panic) scenarios a "waiting rule" is also implemented where agents attempting to navigate a bottleneck will form orderly queues. Impatient agents may attempt to avoid the bottleneck entirely by finding an alternative path.

In order to allow agents to fall and become obstacles pushing behaviour had to be implemented, this pushing behaviour is implemented by giving each agent a "personal space threshold" which when entered would cause them to push against other agents. When the sum of external forces on an agent becomes too great the agent loses equilibrium and falls over thus becoming an obstacle for other agents. See figure 13 for an example of impatient and panicked behaviours as shown by the HiDAC model.

The author has been unable to find comparisons to the HiDAC model and thus shortcomings that other authors have found cannot be listed. However Maïm et al.[23] does mention that in order to achieve the behaviours that the HiDAC model is capable of, the user would currently have to "set a few low level parameters", which might cause problems for someone trying to use it who does not understand the model thoroughly.

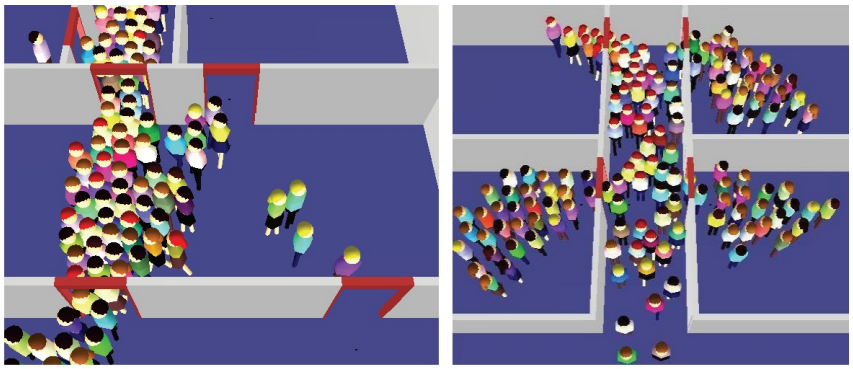

Figure 13: HiDAC generated crowds demonstrating impatience (left), and panicked red-heads (right).

2.4.4.1 Maïm et al. 2008 [9]

This complex model does not fit neatly within the boundaries of any of the basic models due to its divided nature. The goal of the Maïm et al. model was to create a highly scalable real-time simulator, in order to achieve this, computational cycles are limited for 'less important' tasks.

In order to maximally utilize computation time Maïm et al.[9] changes how agents are drawn in the world and how they behave based on where they are relative to the camera. This model uses four different methods in order to draw the Agents based on where they are in relation to the camera: Deformable Meshes for close important agents, Rigid Meshes for medium distance agents, Imposters for far agents and no drawing when not in the line of sight.

It also uses three different methods of behavioural computation: agents in important areas use an accurate potential field-based algorithm, visible agents that are unimportant or uninteresting use navigation graphs with a basic obstacle avoidance algorithm, while non-visible agents use a navigation graph with no obstacle avoidance, shown in figure 14 (a). According to Maïm et al[9] this minimal requirement method minimises strain per agent on the CPU, however it could potentially decrease global accuracy of a simulation.

This combination of rendering techniques and behavioural computation methods results in a model that is highly scalable, graphically realistic, and real-time, as shown in figure 14 (b).

When Maïm et al. tested the model emergent behaviours occurred, more specifically lane formation and panic effects [9]. Dense areas are noted as being potentially problematic, and due to its group-based approach one cannot simulate individual agents with individual goals but must instead assign groups of agents a goal.

According to Maïm et al. this model is best suited to rendering games due to its real-time nature and high quality graphics[9].

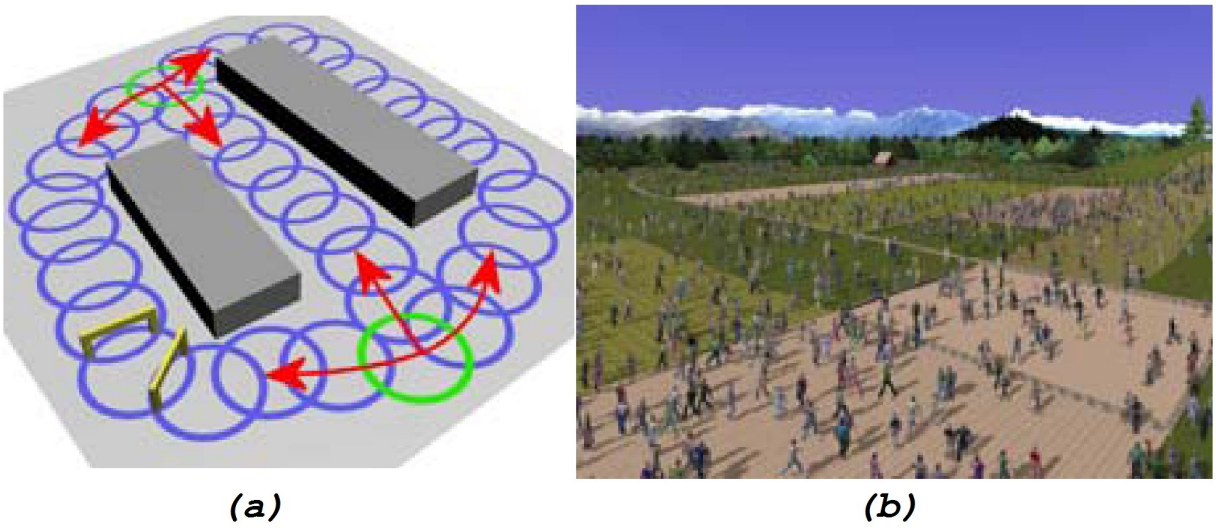

Figure 14: (a) A navigation graph showing how an agent can move from one distant vertex to another (in green) via three different paths (red arrows), across a single navigation flow (blue circles). The edges of two vertexes are represented as gates (in yellow). [9] (b) 10,000 pedestrians moving in a large field.

2.4.4.3 Massive

Massive (created by Massive Software) has applications in film development, game development, education, architectural visualization, and engineering simulation[29] . This unusual model was the only one the author was able to find that combines swarm intelligence[1] with fuzzy logic [22], this unique behavioural technique allows for many 'identical' agents to be placed in an environment and then behave uniquely due to the fuzzy logic aspect of their behavioural technique.

Something else that Massive does differently, is that no agent has a view of the entire environment available to them and must instead rely on a ray based method of perceiving (through artificial vision, hearing and touch) and interpreting their environments in order to decide what behaviour is most appropriate [22, 29, 39].

Noted flaws for this model are that agents cannot communicate with one another and the simulation does not run in real time [22]. According to Massive Software[29] the non-real-time nature of the software is unimportant as the primary focus is on creating graphically realistic simulations; for an example of Massive's realistic crowd rendering see Figure 15. Something that did not appear to be mentioned by Massive Software[29] but is mentioned by Pelechano et al.[22], is that this model aims to create "realistic simulation for short periods (under five seconds)", this means that agents do not handle global navigation or long-term goals well, which would in theory limit its use an evacuation or architectural analysis tool.

[1] A rule based system that applies only a few very simple rules, akin to those governing insects, to many agents in order to create emergent intelligent behaviour: a fairly modern and powerful addition to the field of Artificial Intelligence [38].

Figure 15: Battle scene from the movie I, Robot - animated using Massive [29]

2.4.4.4 MassMotion

It should be noted that as this is one of the newest models available, and thus there is very little documentation outside of what its creators say about it. According to the MassMotion manual [40] this model is based on a modified Social Forces algorithm, much like the HiDAC model. Agents obtain information from a navigation map, as well as the speed and positions of other nearby agents. Each agents then uses this information in real-time to alter its speed and direction in the next frame.

MassMotion [30] like Maim et al. 2008 [9] is able to produce very large crowds (tens of thousands of individual agents) that are rendered in real time, however while Maim et al. 2008 is limited to assigning goals to groups of agents MassMotion is able to create unique objective for each individual agent.

According to Oasys Software , simulations are "rendered with gaming quality graphics" as shown in figure 16, this means that the model could potentially be used to create CGI scenes for movies [41]. It also purports to run a an ordinary desktop PC, meaning that it has relatively low hardware requirements.

Figure 16: A demonstration of the rendering quality of MassMotion[30].

A feature found in MassMotion that could not be found by the author in any other software or literature is a built in simulation analysis tool that can point out potential problem areas, this would be a valuable tool for both architectural planning and evacuation simulations.

The largest drawback for this model appears to be its cost: £20,000.00 for a standalone license and £30,000.00 for a shared license (equivalent to ±R278,000 and ±R417,000 respectively, prices accurate at time of paper completion).

2.4.4.5 Wagoum et al. 2012

This evacuation assistant model by Wagoum et al.[28] varies from most other hybrid models, in that rather than combining aspects of different fundamental models in order to create more advanced behaviour, it simulates the same crowd twice using two different methods. One of the key features of this model is that it allows the user to automatically keep track of real crowds while continuously running faster than real-time evacuation simulation to determine how safe a venue would be in the event of an emergency.

It uses a version of the Floor Fields model for its cellular automaton component, Wagoum et al.[28] state that there are several reasons for this: the Floor Fields model produces accurate evacuation simulation results, it demonstrates several emergent behaviours such

as lane formation, and it scales well. A 'generalized centrifugal forces model'[42] is used as the social forces model because its results correlate well with the empirical data they used from [43], [44], and [45].

This model has been tested on a real world facility with a capacity of 60,000 spectators, which means it is highly scalable. Figure 17 shows the breakdown of the system.

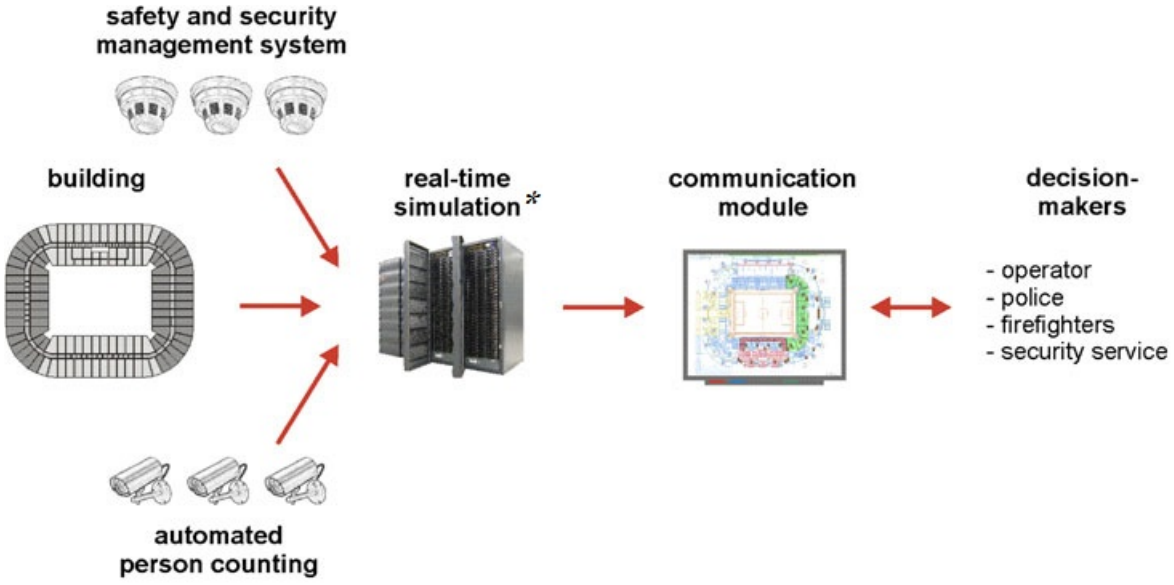

Figure 17: Wagoum et al.[28] evacuation assistant schematic architecture. The real-time simulation section is where the cellular automaton and social forces models perform their work.

2.5 Other Crowd Model Comparisons

According to Pelechano et al. [23] most (but not all) cellular automata are subject to unrealistic animations and only function accurately at medium to low crowd densities. Unlike cellular automata Pelechano et al. [23] says that social forces models operate well at high densities, while realism is still an issue, due in part to agents appearing to vibrate in high density crowds. Pelechano et al.[23] goes on to say that while rule based models can produce realistic animations they stop functioning properly at high densities due to issues such as agent clipping.

[22] performs a detailed comparison of how various models work or can be classified, for details see tables 2 and 3. An example of a comparison factor is inter-agent communication: according to Pelechano[5] most basic models do not allow agents to share information of any kind with one another, the information that a real crowd might share varies from

positive (sharing the location of the nearest exit) to negative (spreading feelings of fear and agitation), thus models that allow for inter-agent communication tend to show more realistic emergent behaviours[5] . In order to avoid bias towards any one system it should be noted that this comparison and the HiDAC model were both created by the same people.

Table 2: Comparison of different crowds simulation models according to Pelechano et al.[22]

Model	Collision Response	Particle Shaking Corrected	Behaviour Method	Spatial Structure for Motion
Social forces (SF)	Yes	No	Forces	Continuous
Rule based	No	Not Needed	Rules	Continuous
Cellular Automata	No	Not Needed	CA	2D grid
Simulex	Yes	No	Distance maps	Continuous
Egress	No	Not Needed	CA	Hexagonal Grid
ViCrowd	No	Not Needed	Rules + FSM	Continuous
OpenSteer	No	Not Needed	Rules	Continuous
Legion	No	Not Needed	Least-Effort	Continuous
Exodus	No	Not Needed	CA	2D grid
Steps	No	Not Needed	CA	2D Grid
Massive SW	No	Not Needed	Rules + fuzzy logic	Continuous
Reactive Navigation	No	Not Needed	Rules	Continuous
Artificial fishes	No	Not Needed	Rules	Continuous
ACUMEN	Yes	No	Particle simulation	Continuous
Crosses	No	Not Needed	Rules + FSM	Continuous
Autonomous Pedestrians	No	Not Needed	Artificial life approach	Continuous
Space Syntax	No	Not Needed	Visibility graphs	Continuous
MACES + HiDAC	Yes	Yes	Extended SF	Continuous
MassMotion	Yes	Yes	Extended SF	Continuous

Table 3: Further comparison between models according to Pelechano et al.[22]

Model	Communication or Signals	Learning	Real Time	Individuals or Roles
Social forces (Helbing)	No	No	Yes	Some
Rule based	No	No	Yes	No
Cellular Automata (CA)	No	No	Yes	No
Simulex	No	No	No	Some
Egress	No	No	No	Some
ViCrowd	No	No	Yes	Yes
OpenSteer	No	No	Yes	Some
Legion	Some	No	No	Some
Exodus	No	No	No	Some
Steps	No	No	No	Some
Massive SW	No	No	No	Yes
Reactive Navigation	No	No	Yes	Yes
Artificial fishes	No	No	Yes	Yes
ACUMEN	Yes	No	Yes	Yes
Crosses	No	No	Yes	Yes
Autonomous Pedestrians	No	No	Yes	Yes
Space Syntax	No	No	Yes	No
MACES + HiDAC	Yes	Yes	Yes	Yes
MassMotion	Some	No	Yes	Yes

Musse[1] lists some of the major uses of simulating crowds and then gives one or more models that can be used for each purpose, for a breakdown of the models and their uses as explained by Musse et al.[1, 46]see table 4. Please note that the 'comparisons' performed by Musse et al.[1] do not say how well each model performs in each and every task, which is the goal of this study.

Table 4: Uses of various models according to Musse et al.[1, 46].

Model	Use	Category
Simulex	Crowd evacuation simulation	Distance maps[47]
Legion	Crowd evacuation simulation	Least-Effort[48]
CACTUS	Crowd management training systems	Agent-based[46, 49]
SULNTS	Crowd management training systems	Unspecified[50]
GATHERING	Sociological Models	Social Forces[46, 51]
Jager 2001	Sociological Models	Cellular Automaton[52]
Reynolds 1987	Computer Graphics	Rule based[26, 34]
Bouvier 1996	Computer Graphics	Particle System [53, 54]
Hodgins 1994	Computer Graphics	Behavioural System[46]

2.6 Methods for Measuring Model Accuracy and Agent Movement

According to Klüpfel et al. [11] empirical data for crowd and pedestrian movement can be broadly broken up onto four categories: accident reports, evacuation exercises, observations and movement experiments, as illustrated in figure 18. This classification does not take into account reliability, validity or objectivity, however it is still a nice method for grouping data. All four types of data can be used to evaluate the accuracy of a models results, most models can also simulate scenarios that fit into each of the four categories.

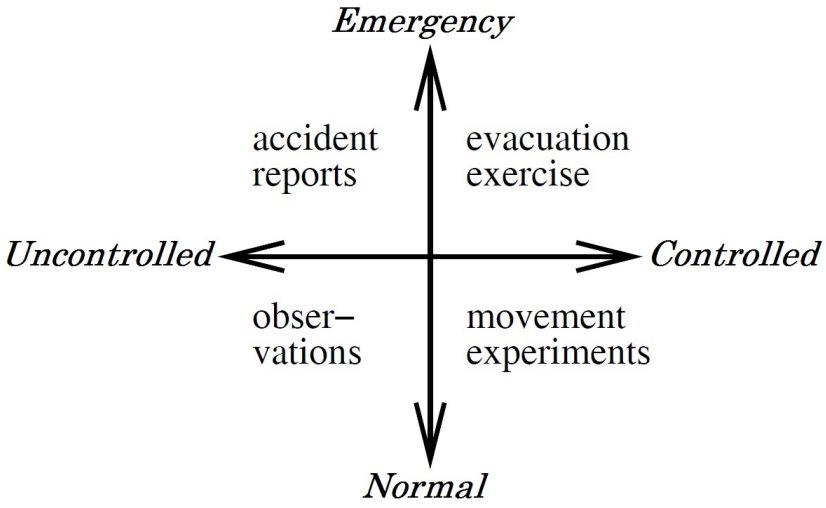

Figure 18: Classification of empirical data according to Klüpfel et al.[11], based on the situation being normal/emergency and uncontrolled/controlled.

Teknomo et al.[55] gives a detailed method for the analysis of crowd movement using a single arbitrarily positioned camera. This model would be very useful for comparing simulations to real world footage, however it is a totally manual method and requires eight hours of work from a single person in order to determine the paths of 40-60 pedestrians in a 150 second time frame.

In a separate paper Teknomo[32] shows some potentially useful crowd data, for example in a normal crowd: mean velocity is 1.38 meters per second with a standard deviation of 0.37 meters per second, while agent mean acceleration is 0.68 meters per second per second.

The only software that the author was able to find that can automatically analyze crowd behaviour was the in-built tool of MassMotion[30], thus based on the work done by Teknomo et al.[55] an in depth empirical analysis of every model's accuracy is not feasible.

Pelechano[5] and Schadschneider[31] note that simulation results, especially evacuation times, depend strongly on the parameters used for that simulation. Thus simulations, even on the same model, will vary regarding accuracy until 'ideal' parameter values are found. Figure 19, from Pelechano[5] , shows an example of such parameter variation: a high percentage of leaders results in faster exploration of an area and thus faster evacuation times while a low percentage of leaders results in slower evacuation times.

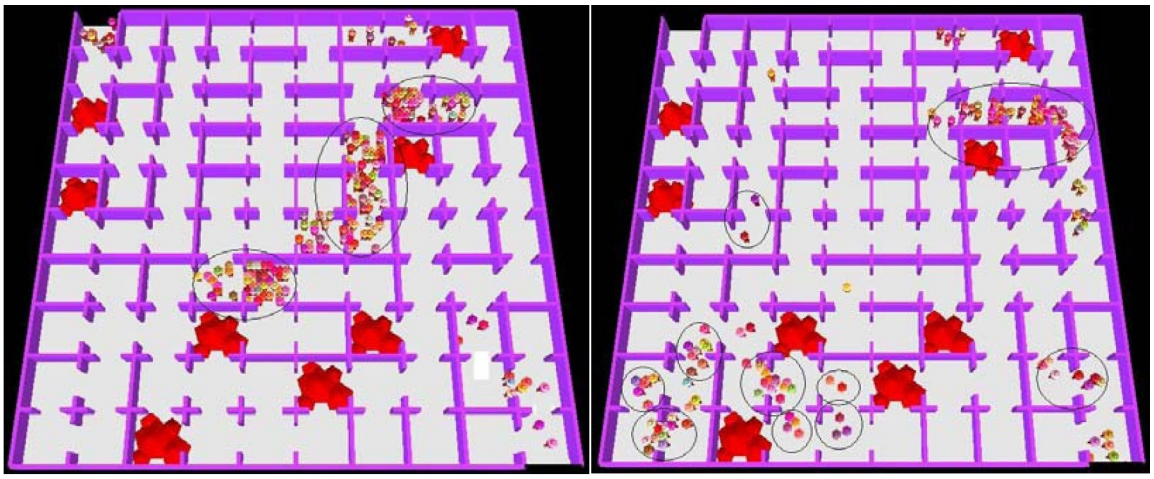

Figure 19: The MACES model showing a difference in evacuation dynamics due to variance in the number of leader agents[5]: low percentage of leaders on the left and high percentage on the right.

According to Musse et al.[56] there is very little documentation on methods for quantifiably comparing crowd simulations, and most comparisons are done on a visual basis,

Teknomo et al.[55] is an example of such a visual comparison. This method of visual comparison is also used by Guy et al.[57]. However according to Zhou et al.[12] the results of a purely visual analysis are highly subjective and are therefore this technique is unsatisfactory. This point leads on to the more quantifiable method described by Musse et al.[56] for simulation comparisons: it utilises a four dimensional histogram to represent data about a crowd such as mean velocities and density. An important point about this comparison technique is that it provides no score or grade for individual models or observed behaviours, but instead its primary goal is the categorisation of crowds based on observed behaviours[56] .

2.7 Summary of the Literature Reviewed

As shown above there are a great deal of methods and models available for simulating crowds, as well as varied opinions about what makes a good simulator, how to classify models, and which factors matter for which applications. Due to the scale of the study it will thus be necessary to compare a few select models, for example the fundamental three, one or two enhancements of each of these, plus select hybrid models such as HiDAC, MassMotion or Massive.

The literature reviewed thus far will later be critically analysed and discussed regarding each of the models to be compared. For example when multiple authors have a particular view about a model, that will be factored into my eventual conclusions. Once that has been done each model will be analysed according to the factors mentioned in the Comparison Factors Between Models section.

Preliminary conclusions based entirely on the content of the literature indicates a direct relationship between price and quality of animation, while accuracy of behaviour is very model specific and potentially erratic even between similar models, and thus individual model usefulness can be expected to vary significantly from application to application.

Further analysis of the literature reviewed can be found in the following section, where view points will be compared and interesting phenomena will be discussed.

3. Discussion

This discussion will address several issues brought up in the literature review, including:

- The overabundance of models and the need for a representative sample.
- Which models should be selected for the representative sample and why.
- Why cellular automata are so common compared to other model types.
- The large number of comparison factors (nearly twenty) and which ones will be used.
- How can performance, for certain factors, be scored, whether quantitatively or qualitatively? For example cost and scalability can be scored based on clear numbers, while graphics quality and usability are more subjective.
- The general advantages and disadvantages of the three fundamental models.

3.1 Overabundance of Models

As shown in the section 2.4 and in Table 14 of appendix 1 there are no less than 64 different *documented* crowd simulation models. If it took only two hours to research each model in depth and one additional hour to determine what scores it deserves it would take no less than 192 hours of work just to obtain preliminary scores. Additionally after comparing factor scores to application weights the end results would have 576 different scores that would then need to be compared to one another. This demonstrates an obvious need to select and analyse a representative sample of models.

The models used in the representative sample need to demonstrate the variability between models, represent each of the fundamental models, and give a comprehensive 'quality/complexity' distribution. For this reason two non-hybrid models from each of the fundamental model categories need to be selected, along with three hybrid models. This distribution should result in the representative sample containing: three high quality/complexity models, three medium quality/complexity models and three low quality/complexity models. The models selected are shown in table 5, along with what fundamental category they best fit into and a preliminary estimation of their quality/complexity.

The results obtained using this representative sample can then be used as a basis for a more generalised explanation of what type of models are good for what applications. For example, a potential generalisation might be that whenever simulation accuracy is essential it is better to go with a hybrid model over a social forces model, and better to use a social forces model rather than a cellular automaton. For exceptional cases these generalisations might not remain true.

Table 5: The representative sample of models to be analysed.

Model	Category	Quality/Complexity Estimation
Massive	Hybrid Rule Based	High
MassMotion	Hybrid Social Forces	High
HiDAC	Hybrid Social Forces	High
Quinn	Social Forces	Medium
OpenSteer	Rule Based	Medium
ABS	Cellular Automaton	Medium
Helbing	Social Forces	Low
Reynolds	Rule Based	Low
Generic CA	Cellular Automaton	Low

There is of coarse no model entitled "Generic CA" however due to the vast number of cellular automata available, selecting a single one to represent them all seemed as if it might cause cellular automata to be misrepresented in the final scores. Therefore an amalgamation of cellular automata attributes will be used to represent a single 'average' cellular automaton.

3.2 Numerous Cellular Automata

When you take a more detailed look at the models discussed and listed in the literature review a distinct bias begins to present itself: Cellular automata(CA) models appear to make up at least 50% of the non-commercial models available (table 2 shows mostly *commercial* models). When comparing CA models one common factor present itself: The logic behind each model is generally simple, at both a mathematical and data structure level as seen in all the CAs examined in the literature review[4, 6, 8, 25]. The simplicity of most CAs in turn leads to three reasons why one might choose to develop a CA rather than

a more complex model: first CAs are really cheap to make, second they don't take long to develop, and finally they are easy to extend and enhance once built due to their rule based nature[31]. In order to test the validity of these three points the author developed a CA comparable to the one developed by Keßel et al.[6] only with eight directions of motion instead of three. This CA took only four days to complete, and once done it was easy to add or modify the agent movement rules, thus lending credence to why there are so many CAs available.

3.3 Comparing Models

The list of potential comparison factors listed by Castle[20] and Zhou et al.[12], as shown in section 2.3, is too extensive for the scale of this paper, additionally some of the factors are extremely hard to accurately quantify (such as agent realism, model result validation, and required formats). Thus in order to maintain an appropriate scope for this paper the factors which affect the most applications and are also quantifiable will be used for scoring purposes. For example scalability matters for evacuations, architectural analysis and even movie making; while logical agent realism (from a data representational perspective) is only useful for academic applications[12].

The more refined list of factors to be used is shown in Table 6[2].

These factors were selected because all of them are quantifiable in some way or another, as well as the fact that they matter for most applications (an example of a factor that does not matter in most cases is file format).

[2]For 2D Vs 3D The world could be a simple chessboard, it could be composed of 2D arrays connected by 'stairs', it may be a single story building but the furniture and fittings could be 3D, the world could appear graphically 3D but actually be represented in two dimensions etc...

Table 6: A list of the comparison factors to be used.

Name	Description
Cost	The cost of a model in terms of resources, time and money.
Graphics	The rendering and display quality of the images or video produced.
Usability	How easy a model is to use, how hard is it to learn to use it initially.
Scalability	How large can a models simulations become in terms of crowd size.
Accuracy	How realistic is the behaviour of agents in a simulation at macro and micro level
2D Vs 3D	How is both the logical and graphical world represented.
Real-Time	What speed does a model run at, does it have variable speeds.
Hardware	What are the hardware requirements, e.g. multiple parallel processors.
Support	How much documentation, help, and guidance is available for the model.

Zhou et al.[12]suggest using just flexibility, extensibility, execution efficiency and scalability: scalability and execution efficiency will both be used, except execution efficiency will be referred to as hardware requirements as an inefficient model is likely to have higher hardware requirements and certain models require specialist hardware, for example Quinn et al.s[24] social forces model would require parallel processors. As the goal of this paper is to determine what models are best suited to what applications, the criteria of flexibility and extensibility become less relevant, this is because the final scores will reflect what models are flexible as far as uses are concerned.

Possible methods for inter-model comparisons include: comparing the results obtained when simulating an identical set of starting conditions on each model, and also comparing the results to real-world crowd footage, much like Guy et al.[57] as shown in figure 20. Further inter-model quantifiable comparisons could be performed based on the 4D histogram technique explained by Musse et al. [56]. This paper will be focusing on inter-model comparisons based primarily on the available literature for each model, as copies of every model to be compared could not be obtained. Comparing one model that can be tested, against another model that can not be tested, may lead to one or more models being misrepresented. Further future work could utilise alternative comparison techniques, such as those mentioned above, if copies of all the models could be obtained.

(a) Simulation of Shibuya Station in Tokyo (b) A different view of the simulated crossing (c) A still from a video of the crossing

Figure 20: A comparison of real-world crowd footage (c) to that produced by a simulation (a & b), as shown by Guy et al.[57]

3.4 Comparison of the Fundamental Models

Cellular automata are generally really cheap, both in terms of implementation time and monetary cost. The rule governing the agents are easily extensible which allows for major modifications to be made easily. Depending on implementation CAs are generally highly scalable. Unfortunately they do tend to give poor graphical results due to the discretisation of both space and time. Despite this they are still able to accurately demonstrate emergent behaviours, and their final results *can* be highly accurate, when implemented well. The major downside to CAs is their limitation to 2D environments, with the correct adjustments several 2D environments can be linked using 'stairs' but true 3D can not be obtained using a CA.

Social forces models tend to be result in more accurate simulations as well as realistic emergent behaviours, however in high density crowds (when agents are not treated right) agents may appear to vibrate. Social forces models are computationally intense due to the theoretically infinite range of the forces involved, this problem can be rectified.

Rule based models usually result in good looking simulations at low to medium densities, however clipping between agents starts to become an issue at higher densities. The major selling point for rule based models is that they scale well, this is because each agent is responsible for following its own rules and only relating these rule to its immediate surroundings.

Both Social forces and rule based models are most commonly implemented for 2D environments, however both can be made to work in 3D environments.

All of the material discussed thus far can now be combined in order to perform a comparison of the various models. The following section does just that.

4. Methodology

This section explains how comparison factors, specific models and specific applications relate to each other in order to obtain model-application scores.

The final goal of this paper is to determine what crowd simulation models and model types are best suited to which real world applications. This goal can be broken down into two primary components: how do models score when compared to each other, and how much do certain factors matter to real world applications. In order to quantify these two components each model needs to be given a score for each of the nine factors listed in table 6, additionally each factor needs to be given an importance weighting for each real-world application.

For example: Massive might get a score for its graphics factor of 90%, and the real world application of making movies would value graphics quality above everything else, resulting in a graphics weighting of 100%; if graphics was the only factor being compared this would result in Massive obtaining a final score of 90% for movie making.

In order to give scores for each factor-model pairing, quantifiable scales had to be derived for scoring each of the factors, these scales are shown in table 7. The values for cost and scalability were derived using this: $1.1-\frac{\log(min)}{\log(max)}$, as a base equation. This can be simplified to $ScaleValue = max^{1.1-fs}$, where fs is a value between 0 and 1 that correlates to the scale value you want to calculate. The lowest value for cost was derived from the most expensive model available (MassMotion); while the peak value for scalability was derived from the capacity of the largest stadium in world: the Rungrado May Day Stadium which can contain up to 150,000 people at once[58], it was assumed that models would not need to be able to handle crowds larger than that.

Other values that are more subjective, such as graphics and usability, are derived from more specific sets of requirements.

Table 7: Factor scales for model scoring. Certain factors are hard to give descriptions for, therefore some fields have been left blank, such fields should be interpreted as having values or descriptions in-between the values immediately before and after the field in question.

Factor	Score out of 10									
	1	2	3	4	5	6	7	8	9	10
Cost (Rands)	400K+	110K-400K	30K-110K	8K-30K	2K-8K	632-2K	174-632	48-174	13-48	Free
Graphics (Quality)	Fig. 5	Fig. 8	Fig. 11	Fig. 6	Fig. 13	Fig.	Fig. 14 (b)	Fig. 16	Fig. 15	Photo Realistic
Usability	Unintuitive, 3 Reference books.	Need professional lessons		Several Tutorials	Read a Tutorial	Short Guide	OK point and click interface		10 Mins to learn	Perfectly Intuitive
Scalability (Agent #)	1-10	10-31	31-100	100-316	316-1K	1K-3K	3K-10K	10K-31K	31K-100K	100K+
Accuracy	Single Rule	Insect like		Noticeable Errors	Few Macro Mistakes	Good Macro Behaviour	Several micro mistakes	Few micro anomalies, no macro	Unnoticeable difference (but there)	Real People
2D Vs 3D	100% Flat			2D world 3D models (e.g. ABS[8])		2D Levels connected via 'stairs'			3D environment mostly 3D 'furnishings'	Real World
Real-Time (Speed)	Only see results once complete			Noticeably above or below real-time	Slightly above or below real-time	Real Time	Real time +1 extra speed	Real time +2 extra speeds	Real time +4 extra speeds	Fully Customizable
Hardware	NASA Grade	Parallel PCs	Custom PC	High end PC	Above Average PC	Average PC	Below Average PC	Cheap PC		Smart Phone
Support	None	Little	Scattered literature	Tutorial or two			Online community		Lots of Documents	24/7 help line

There are a few simple equations that are used to combine the model scores and application weights in order to obtain a weighted average as a final score. The necessary equations follow:

$$R_s = (S \times W)/M \tag{1}$$

Where R_s is a single real score, S is the model's score for a particular comparison factor and W is the weight of said factor for the application being used. M is the maximum score, usually 10.

$$I_s = (I \times W)/M \tag{2}$$

Where I_s is a single ideal score, I is the fictional ideal score for a particular comparison factor. I is usually 10 but if you want to compare against a slightly more realistic high end model, values between 7 and 9 can be used.

$$T_r = \sum R_s \tag{3}$$

Where T_r is the total real score for a single model and application pairing.

$$T_i = \sum I_s \tag{4}$$

Where T_I is the ideal score for a single model and application pairing.

$$S_f = (T_r/T_i) \times 100 \tag{5}$$

Where S_f is the final adjusted percent score of a single model-application pair, not including critical factor penalties.

As an example here table 8 shows how the scores for Massive being used for movie making are calculated if one uses the factor scores and application weights as shown in tables 9 and 10 with an M value of 10:

	Cost	Graphics	Usability	Scalability	Accuracy	2D Vs 3D	Real-Time	Hardware	Support
$R_s =$	$\frac{2\times0.1}{10}$	$\frac{10\times1.0}{10}$	$\frac{3\times0.2}{10}$	$\frac{7\times0.7}{10}$	$\frac{7\times0.7}{10}$	$\frac{10\times1.0}{10}$	$\frac{2\times0.2}{10}$	$\frac{3\times0.3}{10}$	$\frac{3\times0.4}{10}$
	0.02	1.0	0.06	0.49	0.49	1.0	0.04	0.09	0.12
$I_s =$	$\frac{10\times0.1}{10}$	$\frac{10\times1.0}{10}$	$\frac{10\times0.2}{10}$	$\frac{10\times0.7}{10}$	$\frac{10\times0.7}{10}$	$\frac{10\times1.0}{10}$	$\frac{10\times0.2}{10}$	$\frac{10\times0.3}{10}$	$\frac{10\times0.4}{10}$
	0.1	1.0	0.2	0.7	0.7	1.0	0.2	0.3	0.4
$T_r = 3.31$			$T_i = 4.6$			$S_f = \frac{3.31\times100}{4.6} = 71.95\%$			

Table 8: Steps involved in calculating the final score for Massive regarding movie making.

One might be tempted to use only the equations described thus far, however when using only said equations anomalies begin to present themselves in the final scores (as shown in table 12 of section 5). These inaccuracies are due to the fact that some applications have one or more critical factors: when a certain minimum value is not met by the factor score, the overall score should be severely penalized. For example, when considering models for making movies, Massive which was designed for CGI movie making is outscored by MassMotion; and other models such as HiDAC and OpenSteer score unexpectedly well, despite not having a high graphics score. The reason for this is that while these other models do not score particularly well with graphics they make up for it with high scores in other areas.

To mitigate this problem the following equation needs to be applied to the final score for each factor that has a critical value:

$$P_t = ((C_v - S) \div (10/P_f)) \times P_s \tag{6}$$

$$S_{fp} = \begin{cases} P_t \leq 0 & S_{fp} = S_f \\ P_t > 0 & S_{fp} = S_f - P_t \end{cases} \tag{7}$$

WhereP_t is the total penalty size, S_{fp} is the final score with penalties applied, C_v is the critical value for a factor (any whole number from 1 to 10), P_f is the penalty frequency or how often the penalty is applied (10% was used for this investigation), and finally $P_s$is the penalty size (10% was also used for this value). Negative values for P_t indicate that the model has scored above the minimum value, and thus if one were to subtract this negative value the score in question would increase (this could be a valid method if you wanted to

reward models that meet the requirements, but this was not the case in this paper). It should be noted that certain weight, score and critical factor penalty combinations *will* result in negative scores, a negative score is still valid it simply means that a model should never even be considered for a particular application.

If one wanted to compare model scores to the real world the M value (maximum score) could be altered from being a constant 10 to more realistic value for each factor. For example the cost of paying a real crowd of people to act in a movie be around R50 per person per hour (R40,000 to rent a crowd of 100 people for eight hours) resulting in a real world maximum score of only three according to the cost scale in table 7. A similar example would be in the real-time nature of real world crowd only ever scoring six on the scale shown in table 7. Modifications of the M value generally result in higher scores per model, and would allow for more varied evaluation of individual models. This kind of modification would also allow one to compare all models against one comparative 'base-model' by substituting the base-models scores into the appropriate M values.

Table 9 shows the scores that have been allocated to each model for individual factors. It would be implausible to explain why each and every model got the scores it did, however a brief explanation of some of the scores follows:

Any model that is academic is unlikely to have a monetary cost associated with purchasing a copy, therefore for models such as Social Forces and Rule Based, cost is derived from how costly it would be to implement ones own instance of a model in terms of time and effort. A generic CA would be less trouble to create than a model using the OpenSteer library, and a from scratch implementation of Reynolds Rule Based Model would be more effort than implementing a one using OpenSteer. Thus the cost scores for a generic cellular automaton, an OpenSteer model and an implementation of Reynolds' rule based model are nine, eight and seven respectively.

Support is often hard to score for non-commercial models, thus any model that is based on another similar model generally gets a better score for support than the model it is based on. The reason for this is than the newer model has its own documentation as well as the documents of the model its was based on to act as support. For example Quinn's Social Forces model is based on Helbing which was given a score of seven, thus Quinn's model should be given a score of eight or more.

Remembering that OpenSteer is not a model in and of itself, but is in fact a rule-based library, its scores are based on two assumptions: firstly the model being created using the OpenSteer library is neither a masterpiece of, nor a disaster of, programming (either

case would result in OpenSteer being over or under scored), and secondly OpenSteer will usually outscore Reynolds[26, 34] Rule-based model, as shown in table 9.

2D Vs 3D is quite possibly the hardest factor to score because it is not elaborated on much in the associated literature and it does not have a clear scale, thus it tends to have a direct relationship to the graphics score.

One should note that any future work following this line of investigation could, validly, alter any or all scores and weights used here. In order to avoid potential misinterpretation of the scoring technique, one can download the source code (C#) for the calculator program created to score the models at http://www.filedropper.com/crowdmodelscorecalculator, alternatively one can contact the author at g08f0016@campus.ru.ac.za. The aforementioned calculations are simple enough to be put into a spreadsheet, albeit a large one, if one wanted a more platform independent scoring technique.

The values shown in table 10 for factor importance weights are given according to the perspective of the end user and not the model's developer. For example, the end user of a game requires that his game be usable and reasonably priced, while the game developer would have a very different set of requirements, for example cost would be far less important for a game developer with a big budget.

The weights used are based on what literature for each application deems important, as well as what common sense dictates as being important. For example, Wagoum et al. [28] state that up-to-date information and fast simulations are important for both crowd control and real-time evacuation simulators thus a weight of 0.8 for the real-time factor importance for both crowd control and evacuation simulators. An example of a common sense weight would be graphics when making movies: the visual quality is critical thus an importance weighting of 1.0.

Table 9: The factor scores for each model.

	Massive	MassMotion	HiDAC	Helbing	Quinn	Reynolds	OpenSteer	ABS	Generic
Cost	2	1	6	7	6	7	8	7	9
Graphics	10	8	5	3	4	3	5	4	2
Usability	3	7	6	5	6	6	6	5	6
Scalability	7	8	7	6	8	5	6	6	7
Accuracy	7	9	8	7	7	4	5	6	5
2D Vs 3D	10	9	7	2	2	5	6	4	1
Real-Time	2	7	8	7	8	7	7	8	9
Hardware	3	7	8	9	4	8	7	8	10
Support	3	4	6	7	8	6	9	5	8

Table 10: The factor importance weights for each real-world application.

	Movies	Games	Evacuation	Architecture	Crowd Control	Training	Teaching
Cost	0.1	0.8	0.2	0.1	0.3	0.4	0.8
Graphics	1.0	0.7	0.5	0.5	0.4	0.6	0.3
Usability	0.2	1.0	0.7	0.5	0.7	1.0	0.9
Scalability	0.7	0.5	0.9	0.9	0.8	0.6	0.4
Accuracy	0.7	0.8	1.0	0.8	0.9	0.8	0.6
2D Vs 3D	1.0	1.0	0.9	0.6	0.6	0.4	0.4
Real-Time	0.2	0.8	0.8	0.7	0.8	0.8	0.6
Hardware	0.3	0.9	0.5	0.4	0.4	0.6	0.8
Support	0.4	0.9	0.6	0.5	0.4	0.7	0.7

The critical factor scores shown in table 11 are derived using several steps: firstly, if an application needs to do one particular thing really well it must get a critical score for that one thing; secondly the critical value is then determined by looking at what might be considered a bare minimum based on the scale shown in table 7, finally critical factors and their values are adjusted based on whether the final scores have inexplicable anomalies. For example movies aim to do one thing well, graphics, looking at the scale for graphics in table 11, anything below a value of eight or nine would be looked down upon in the movie industry; if given a value of eight MassMotion outscores Massive by almost 6% which does not make sense, by adjusting the critical value to nine this anomaly is removed and Massive obtains the top score for making movies (which it should as that is what it was originally designed for)[29, 39].

Table 11: The critical factors per model and their minimum values.

	Movies	Games	Evac	Arch	CrowdC	Training	Teaching
Cost	-	6	-	-	-	-	6
Graphics	9	6	-	-	-	-	-
Usability	-	-	-	-	7	8	6
Scalability	-	-	8	8	8	-	-
Accuracy	-	-	9	9	8	7	-
2D Vs 3D	-	-	-	-	-	-	-
Real-Time	-	5	-	-	-	-	-
Hardware	-	7	-	-	-	-	-
Support	-	-	-	-	-	-	6

The methodology described above can now be applied to each model-application pair in order obtain the results as shown in the next section.

5. Results

Using the factor scores per model as shown in table 9 and the factor importance weights per application as shown in table 10, but not yet applying the critical value penalties, you obtain the scores shown in table 12.

Table 12: Final scores without critical factor penalties. Four inaccuracies are highlighted in blue, red, orange and purple respectively: MassMotion should not outscore Massive when making movies; with a cost of over R400,000 is MassMotion really that appropriate for games and teaching which both usually have tight end-user budgets; no movie could ever allow the quality of graphics shown in these orange highlighted models; why are these two social forces models so similarly scored for gaming which usually has hardware restrictions.

	Movies	Games	Evacuation	Architecture	Crowd Control	Training	Teaching
Massive	72.0	50.9	56.9	57.4	54.0	49.3	43.6
MassMotion	77.2	66.4	73.6	74.2	72.3	68.5	62.4
HiDAC	66.7	67.8	69.7	69.6	69.8	68.1	68.2
Social Forces	49.6	59.6	57.5	58.0	59.2	60.8	64.5
Quinn	52.8	57.3	60.0	61.6	61.7	61.7	60.4
Reynolds	49.8	56.9	54.9	54.8	55.5	55.9	59.3
OpenSteer	60.4	66.1	63.3	63.0	63.2	64.9	67.6
ABS	52.4	58.5	57.9	58.2	59.2	59.2	61.1
CA Generic	46.7	62.8	59.3	60.2	61.9	64.7	69.8

After applying the penalties associated with the critical factors in table 11, the final scores as shown in table 13 are obtained. These are the results that will be used to determine how good a model is for a particular application and later they will be used to determine what general model types are best at. This table shows that all the anomalies highlighted in table 12 have been dealt with.

Table 13: Final scores with critical factor penalties applied.

	Movies	Games	Evacuation	Architecture	Crowd Control	Training	Teaching
Massive	72.0	-59.1	26.9	27.4	-6.0	-0.7	-56.4
MassMotion	67.2	16.4	73.6	74.2	72.3	58.5	-7.6
HiDAC	26.7	57.8	49.7	49.6	49.8	48.1	68.2
Social Forces	-10.4	29.6	17.5	18.0	9.2	30.8	54.5
Quinn	2.8	7.3	40.0	41.6	41.7	41.7	60.4
Reynolds	-10.2	26.9	-15.1	-15.2	-24.5	-4.1	49.3
OpenSteer	20.4	56.1	3.3	3.0	3.2	24.9	67.6
ABS	2.4	38.5	7.9	8.2	-0.8	19.2	41.1
CA Generic	-23.3	22.8	9.3	10.2	11.9	24.7	69.8

6. Analysis of Results

This section takes an in depth look at how tables 9, 10 and 11 relate to the values found in table 13 as well as what this means for model applicability. First there will be a discussion of the more model specific scores(6.1), which will then be used to create generalisations for the model types rather than specific models(6.2). Note that because certain models were made as academic investigations (which typically do not require high end graphics or usability) their scores could be increased in alternative implementations that aimed to improved those factors that are not essential for research purposes. Such alterations would probably decrease scores for the hardware and/or cost factors, but would increase the overall scores due to decreased critical factor penalties. For example Quinn's social forces model, if implemented with cutting edge graphics, would score almost as well for making movies as the commercial models. However due to the probable increase in cost it would no longer score well for teaching.

6.1 Specific Applications

On average the top-scoring model is the very expensive piece of commercial software, MassMotion: according to the results in table 13 the only tasks that it would be totally inappropriate for are those that are budget sensitive, i.e. gaming and teaching. One would expect MassMotion to score really well for evacuation and architectural analysis applications as these are the kinds of things that it was originally designed to simulate. The result obtained for crowd control was initially the most surprising result for MassMotion, however when looked at in more detail it makes sense that this model would be good for controlling crowds: it is highly scalable, easy to use, and can be run faster that real-time, all these factors are essential when trying to predict and control crowds in large areas such as airports.

Massive software was originally designed to create CGI for movies, which implies that it should do a good job at creating movies: the results obtained agree with this implication, as Massive outscores all the other models analysed for movie making. With the exception of MassMotion, Massive outscores the other models for making movies by never less than 45%. While Massive does a great job with movies it falls sadly short for all other applications: four out of the seven scores obtained for Massive are negative.

Of the non-commercial models HiDAC outscores every other model for almost every application: this means that if money is an issue your first choice in models for all applications, except teaching, should be HiDAC. However while HiDAC just outscores OpenSteer for making games, it should be noted that due to OpenSteer being a library rather than a model by itself, it should probably be used for game making above all other models including HiDAC.

The top scorer for teaching is the cheap and simple generic cellular automaton, it even outscores HiDAC, there are several reasons for this: cellular automata are simple to understand, simpler ones are generally easy to make even for a beginner programmers, and finally there is a great deal of literature available as support regarding the creation of cellular automata. Unfortunately the generic CA scores poorly for all other applications, this is to be expected as the other applications have specialist requirements that could not be incorporated into something as simple as this model.

The other model that does well when used for teaching is OpenSteer, the reasons for this are similar to those of the generic CA: OpenSteer is free, its a library that aims to simplify the task of programming a model, and it is well documented which makes learning to use it easier.

The two social forces models analysed demonstrate some surprising results:

Quinn's social forces model manages to somehow get three almost identical results for architecture, crowd control and training, this is probably just an interesting numerical coincidence. These three applications, as well as evacuation, all obtain average results. Due to its high scalability and moderately high accuracy (the two critical factors in architecture and evacuation) one might have expected higher results for both applications, the low final score is probably due to other factors scoring poorly thus bringing down the overall result. This is an example of a numerical score potentially misrepresenting what a model would be most suited to. Further evidence of this misrepresentation is shown in Quinn's model's score for teaching being its highest score: it is a more complex implementation of an already maths heavy model that also requires an understanding of parallelism, and would thus probably be better suited to evacuation simulations rather than education. This model might be appropriate for teaching advanced programming or crowd psychology, but it would be inappropriate for teaching an introductory programming course on simulations as it is unnecessarily complex.

Reynolds' rule based model and the ABS model both get their highest scores for teaching, this is to be expected as they are both free models that are also easy to understand.

6.2 General Applications

After analysing the model specific results certain generalisations can be made about what types of model are best suited to what applications:

MassMotion, HiDAC, Helbing and Quinn are all social forces models, and all generally score well compared to the other models, this leads to the generalisation that social forces models are suited to more applications than the other model types. As with all generalisations there are a few exceptions, such as with teaching where simplicity is paramount. This leads on to the next generalization, for teaching: non-commercial models are almost always better for teaching, and amongst these CAs come out as the clear winner. This is due to several factors: academic models are usually cheaper, they are usually simpler to understand, they are usually simpler to create (good as a lesson in programming), and finally, in general, the more support a model has and the simpler it is, the better it is for teaching.

If money is not an issue, then commercial models tend to be the better choice for almost all applications (exceptions include game-making and teaching according to the results

in table 13). But one must not forget that they could be misrepresented here due to the focus typically being on making them look good.

For evacuation and architectural applications social forces based models, both commercial and academic, are the clear winners, while rule-based models tend to score lowest. This is probably because social forces models tended to have higher scores for both accuracy and scalability, which are critical factors for both evacuation and architectural applications.

As far as graphics go you get what you pay for, therefore the most suitable models for making movies are commercial ones. Unlike most other applications where a non-commercial model could be a valid substitute this is not the case for making movies. Rule based models appear to score best for making movies, however social forces models are a close second.

Gaming seems to be a hung vote with all three fundamental models scoring all over the board, because of this the best generalisation that can be offered based on the results is that cheaper models are better suited to making games so long as they have slightly above average graphics.

The more scalable social forces implementations tend to score best at crowd control, this is probably due to their moderately high accuracy and real-time nature. While the various social forces models where best for crowd control, the rule based models scored worst, this could be due to the accuracy of these models being relatively low (with the exception of Massive for which the reason is its low real-time score and useability).

7. Future Work

Future work could be done by anyone who disagrees with any of the values in tables 9,10,or 11: the values can be altered as one sees fit, as many of the them are subjective. Work could be done entirely on developing a method for creating quantifiable scores, weights and critical values.

In order to further validate the general conclusions about application appropriateness for the fundamental model types, a similar investigation could be done that uses a different set of models as the representative sample.

An alternative method to further validate the results obtained, would be to obtain copies of the various models used, and perform practical or quantitative tests on them, as explained in section 3.3.

Further analysis could be performed on the same set of models, however the scores could be calculated against more real world M (maximum score) values, as used in equations 1 and 2 of section 4 resulting in a potentially more realistic set of scores.

8. Conclusion

A clear correlation was found between cost, complexity, and scores: in general as cost increased (either in terms of time or money) so did complexity and vice versa, as either of these factors increased so did the final scores, with anything that has a critical maximum cost value being the exception.

Conclusions for specific model suitability are as follows: Massive was found to be best suited to making movies, OpenSteer and HiDAC are the best choices for making games, MassMotion scored best for evacuations, architectural analysis, crowd control and training, finally the best choice for teaching crowd psychology and/or programming is a simple cellular automaton. Generalisation of these results show that: commercial models tend to obtain the highest scores, social forces models are suited to more applications than the other model types especially evacuation and architectural applications, CAs are best suited to teaching, and rule based models tend to be best for making movies. Another key conclusion is that purely numeric scores might not always be a good measure of a models application appropriateness, due to certain values being hard to quantify.

Finally it is also clear that comparing academic models to commercial products is not fair: the commercial products are typically made to look good, and have whole teams working on them, thus resulting in higher scores despite the fundamental architecture of the commercial models being almost identical to more academic models.

Appendix 1: List of Models

Table 14: This list is an amalgamation of various models, taken from [21] and Pelechano et al. [22] as well as any others that the author discovered that were not originally listed in these two sources.

Model	Owner/creator		Model	Owner/creator
ABS	Tecchia et al.		HiDAC	Pelechano et al.
ALLSAFE	Heskestad et al.		Lamarche & Donikian	Lamarche & Donikian
Aseri	Schneider		Legion	
BFires	Stahl		Magnetic Model	Okazaki & Matsushita
BGRAF	Ozel		MASCM	Murakami et al
Blue & Adler	Blue & Adler		Massive	Massive Software
Brogan & Hodgins	Brogan & Hodgins		MassMotion	Oasys Software
Brogan & Johnson	Brogan & Johnson		Micro PedSim	Teknomo
Continuum Crowds	Treuille et al.		Musse & Thalmann	Musse & Thalmann
Cost Function Model	Hoogendoorn		Myriad/VeGas	Still
Dijkstra et al	Dijkstra et al		Ohi et al.	Ohi et al.
EARM			PathFinder	Capuccio
EESCAPE	Kendik		Paxport	
Egress	Ketchel		PEDFLOW	Kerridge & McNair
ENTROPY	Donegan et al.		PedGo/Aeneas	TraffGo HT GmbH
E-Scape	Reisser-Weston		Schadschneider	
Evacnet14			Schultz et al.	
EvacSim	Poon		SGEM	Do et al.
Evi	Vassalos		Shao & Terzopoulos	
Exit89	Fahy		Simulex	Thompson
EXITT	Levin		Simwalk	Savannah Simulations
Exodus	Galea		Social Distances	Was et al.
F.A.S.T.	Kretz		Social Forces	Helbing
FDS + Evac	VTT		STEPS	Hoffmann & Henson
Firescap	Feinberg & Norris		Sugiyama et al.	
Floor Fields model	Burstedde et al. 2001		Sung et al.	
Floor Fields, extended	Kirchner et al 2003		Swarm Information	Henein & White
FPETool	NIST		Takahashi et al.	
Fridman Kamimka			TIMTEX	Harrington
Funge et al.	Funge et al.		Wagoum et al. 2012	
Gridflow and CRISP			WayOut	Shestopal & Grubits
Group Psychology	Katuhara et al.		Yamamoto et al.	
Helios	Majumder			

Bibliography

[1] Soraia Raupp Musse, Branislav Ulicny, and Amaury Aubel. *The handbook of virtual humans.* John Wiley & Sons, 2004.

[2] Andrew Fell. A study of modelling crowd dynamics. Master's thesis, Carleton University, 2003.

[3] Dirk Helbing and Peter Molnar. A social force model for pedestrian dynamics. *Phys. Rev. E*, 51:4284–4286, 1995.

[4] Ansgar Kirchner, Katsuhiro Nishinari, and Andreas Schadschneider. Friction effects and clogging in a cellular automaton model for pedestrian dynamics. *Phys. Rev. E*, 67(5):056122, May 2003.

[5] Nuria Pelechano, Kevin O'Brian, Barry G. Silverman, and Norman I. Badler. Crowd simulation incorporating agent psychological models, roles and communication. In *First International Workshop on Crowd Simulation*, pages 21–30, 2005.

[6] A. Keßel, H. Klüpfel, J. Wahle, and M. Schrechenberg. Microscopic simulation of pedestrian crowd motion. In M. Schreckenberg and S.D. Sharma, editors, *Pedestrian and Evacuation Dynamics*, pages 193–202, Berlin, 2002. Springer.

[7] Peter Miller. *The Smart Swarm: How understanding flocks, schools, and colonies can make us better at communicating, decision making, and getting things done.* New York: Avery, 2010.

[8] Franco Tecchia, Celine Loscos, Ruth Conroy, and Yiorgos Chrystanthou. Agent behaviour simulator (abs): A platform for urban behaviour development. In *Games Technology (GTEC 2001)*, 2001.

[9] Jonathan Maïm, Barbara Yersin, and Daniel Thalmann. Real-time crowds: architecture, variety, and motion planning. In *ACM SIGGRAPH ASIA 2008 courses*, SIGGRAPH Asia '08, pages 56:1–56:16, New York, NY, USA, 2008. ACM.

[10] QuikQuak Audio Plug-ins, http://www.quikquak.com/media/UserGuides/-Crowd_Chamber_User_Guide_2_00.pdf. *Crowd Chamber User Guide*, 2 edition, 2009.

[11] Hubert Klüpfel, Michael Schreckenberg, Tim Meyer-könig, and Traffgo Gmbh. A cellular automaton model for crowd movement and egress simulation.

[12] Suiping Zhou, Dan Chen, Wentong Cai, Linbo Luo, Malcolm Yoke Hean Low, Feng Tian, Victor Su-Han Tay, Darren Wee Sze Ong, and Benjamin D. Hamilton. Crowd modeling and simulation technologies. *ACM Transactions on Modeling and Computer Simulation*, 20(4):20:1–20:35, November 2010.

[13] J. Harbst and F. Madsen. The behaviour of passengers in a critical situation on board a passenger vessel or ferry. Technical report, Danish Investment Foundation, Kopenhagen, 1996.

[14] J. Sime. *Fires and Human Behaviour*. David Fulton, 2nd edition, 1990.

[15] Guylène Proulx. Cool under fire. *Fire Protection Engineering*, 16:23–25, 2002.

[16] J. P. Keating. The myth of panic. *Fire Joural*, 76(3):57–62, May 1982.

[17] Wolram.W.F. Klingsch, C. Rogsch, A. Schadschneider, and M. Schrechenberg. *Pedestrian and Evacutaion Dynamics 2008*. Springer, 2008.

[18] Donelson R. Forsyth. *Group Dynamics*. Wadsworth Publishing, 5rd edition, 2010.

[19] C. Rogsch and W. Klingsch. Basics of software-tools for pedestrian movement–identiffication and results. *Fire Technology*, 48:105–125, 2012.

[20] Christian J. E. Castle. Guidelines for assessing pedestrian evacuation software applications. Technical report, University College London, 2007.

[21] Hubert Klüpfel, Tobias Kretz, Christian Rogsch, Andreas Schadschneider, and Armin Seyfried. List of models. Online: http://www.ped-net.org/index.php?id=33&ID=311, March 2007.

[22] Nuria Pelechano, Jan Allbeck, and Norman I. Badler. *Virtual Crowds: Methods, Simulation, and Control*. Morgan & Claypool Publishers, 2008.

[23] Nuria Pelechano, Jan Allbeck, and Norman I. Badler. Controlling individual agents in high-density crowd simulation. In *Symposium on Computer Animation*, pages 99–108. University of Pennsylvania, USA, 2007.

[24] Michael J. Quinn, Ronald A. Metoyer, and Katharine Hunter-zaworski. Parallel implementation of the social forces model. In *in Proceedings of the Second International Conference in Pedestrian and Evacuation Dynamics*, pages 63–74, 2003.

[25] Carsten Burstedde, Kai Klauck, Andreas Schadschneider, and Johannes Zittartz. Simulation of pedestrian dynamics using a two-dimensional cellular automaton. *Physica A*, 295:507–525, 2001.

[26] Craig W. Reynolds. Flocks, herds, and schools: A distributed behavioral model. *Proceedings of ACM SIGGRAPH*, 21:25–34, 1987.

[27] Rick Parent. *Computer Animation, Algorithms and Techniques*. Morgan Kaufmann Publishers, 2002.

[28] Armel Ulrich Kemloh Wagoum, Mohcine Chraibi, Jonas Mehlich, Armin Seyfried, and Andreas Schadschneider. Efficient and validated simulation of crowds for an evacuation assistant. *Computer Animation and Virtual Worlds*, 23(1):3–15, February 2012.

[29] Massive, simulating life. Online: http://www.massivesoftware.com/, 2011.

[30] Massmotion. Online: http://www.oasys-software.com/massmotion.html, 2011.

[31] Andreas Schadschneider. Private Correspondence, July 2012. Email correnspondence between author and Andreas Schadschneider.

[32] Kardi Teknomo. *Microscopic Pedestrian Flow Characteristics: Development of an Image Processing Data Collection and Simulation Model*. PhD thesis, Tohoku University Japan, Sendai, 2002.

[33] Dirk Helbing, Illes Farkas, and Tamas Vicsek. Simulating dynamical features of escape panic. *Nature*, 407:487–490, 2000.

[34] Craig W. Reynolds. Steering behaviors for autonomous characters. In *Game Developers Conference*, pages 763–782, 1999.

[35] Craig Reynolds. *OpenSteer Documentation*. Sony Computer Entertainment America, June 2007.

[36] Barry G. Silverman, Michael Johns, Jason Cornwell, and Kevin O'Brien. Human behavior models for agents in simulators and games: Part i: Enabling science with pmfserv. *Presence: Teleoper. Virtual Environ.*, 15(2):139–162, April 2006.

[37] Barry G. Silverman, Gnana Bharathy, Kevin O'Brien, and Jason Cornwell. Human behavior models for agents in simulators and games: Part ii: Gamebot engineering with pmfserv. *Presence: Teleoper. Virtual Environ.*, 15(2):163–185, April 2006.

[38] Peter Tarasewich and Patrick R. McMullen. Swarm inteligence: Power in numbers. *Communications of the ACM*, 45:62–67, 2002.

[39] Massive software 2.0. Online: http://www.creativeplanetnetwork.com/dcp/reviews/-massivesoftware20/14255, 2011.

[40] Oasys Software, http://www.oasys-software.com/media/Manuals/Latest_Manuals/-MassMotion_5-0_UserGuide.pdf. *MassMotion v5.0.5 Manual: Design Simulate Optimize*, June 2011.

[41] Pete Lamb. Create realistic crowds and more with massmotion. Online: http://www.jigsawbroadcast.com/news/create-realistic-crowds-and-more-with-massmotion, June 2011.

[42] Mohcine Chraibi, Armin Seyfried, and Andreas Schadschneider. Generalized centrifugal-force model for pedestrian dynamics. *Physical Review E*, 82:046111, 2010.

[43] M. Mori and H. Tsukaguchi. A new method for evaluation of level of service in pedestrian facilities. *Transportation Research Part A: Policy and Practice*, 21(3):223–234, 1987.

[44] Dirk Helbing, Anders Johansson, and Habib Z. Al-Abideen. The dynamics of crowd disasters: An empirical study. 75, February 2007.

[45] B. D. Hankin and R. A. Wright. Passenger flow in subways. *Operational Research Quarterly*, 9(2):81–88, 1958.

[46] Soraia Raupp Musse, Branislav Ulicny, Amaury Aubel, and Daniel Thalmann. Groups and crowd simulation. In *ACM SIGGRAPH 2005 Courses*, SIGGRAPH '05, New York, NY, USA, 2005. ACM.

[47] Simulex - technology. Online: http://www.simulexinc.com/products/technology/, 2012.

[48] J. Berrou, J. Beecham, P. Quaglia, M. Kagarlis, and A. Gerodimos. Calibration and validation of the legion simulation model using empirical data. In Nathalie Waldau, Peter Gattermann, Hermann Knoflacher, and Michael Schreckenberg, editors, *Pedestrian and Evacuation Dynamics 2005*, chapter 15, pages 167–181. Springer, Berlin, Heidelberg, 2007.

[49] Roderick J. Williams. *A Simulation Environment to Support Training for Large Scale Command and Control Task.* PhD thesis, University of Leeds, 1995.

[50] Denise Varner, Scott D. Royse, and John Micheletti. USMC Small Unit Leader Non-Lethals Training System (SULNT). Online: http://vault.swri.org/cms/papers/3776_Presentation_1998ITEC_T033.pdf.

[51] Clark McPhail, William T. Powers, and Charles W. Tucker. Simulating individual and collective action in temporary gatherings. *Social Science Computer Review*, 10(1):1–28, April 1992.

[52] Wander Jager, Roel Popping, and Hans van de Sande. Clustering and fighting in two-party crowds: Simulating the approach-avoidance conflict. *Journal of Artificial Societies and Social Simulation*, 4(3), 2001.

[53] Eric Bouvier and Pascal Guilloteau. Crowd simulation in immersive space management. In *Proceedings of the Eurographics Workshop on Virtual Environments and Scientific Visualization '96*, pages 104–110, London, UK, UK, 1996. Springer-Verlag.

[54] Eric Bouvier, Eyal Cohen, and Laurent Najman. From crowd simulation to airbag deployment: Particle systems, a new paradigm of simulation. *Journal of Electrical Imaging*, 6(1):94–107, January 1997.

[55] Kardi Teknomo, Yasushi Takeyama, and Hajime Inamura. Data collection method for pedestrian movement variables. *Civil Engineering Dimension*, 2(1):43–48, March 2000.

[56] Soraia R. Musse, Vinicius J. Cassol, and Cláudio R. Jung. Towards a quantitative approach for comparing crowds. *Computer Animation and Virtual Worlds*, 23(1):49–57, February 2012.

[57] Stephen J. Guy, Jatin Chhugani, Sean Curtis, Pradeep Dubey, Ming Lin, and Dinesh Manocha. Pledestrians: a least-effort approach to crowd simulation. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '10, pages 119–128, Aire-la-Ville, Switzerland, Switzerland, 2010. Eurographics Association.

[58] Rungrado May Day Stadium. Online: http://www.worldstadiums.com/stadium_menu/architecture/stadium_design/pyongyang_may_day.shtml, September 2012.