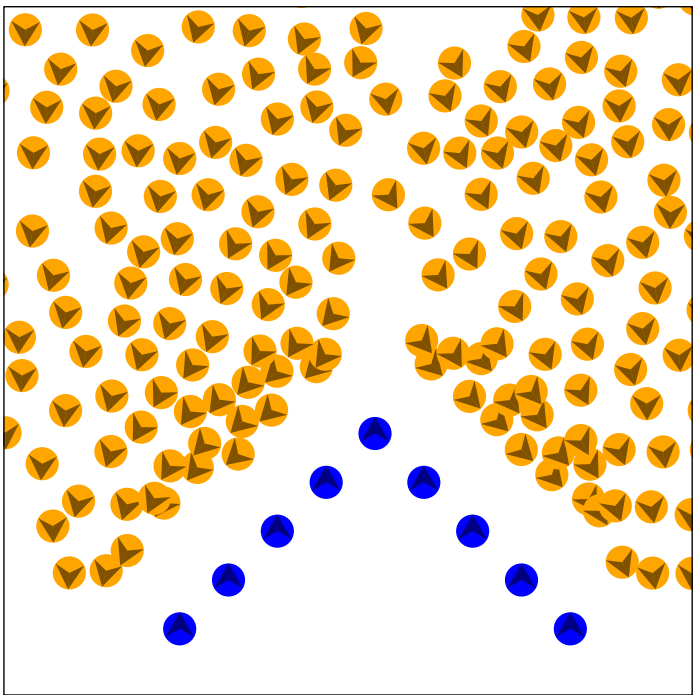
Tessa Verbruggen

Maintaining formations in high-density crowds



Master Thesis
ICA-3173704

supervisors:
Roland Geraerts
Frank van der Stappen
Floris Jan van Brederode

Game and Media Technology
Utrecht University, the Netherlands

July 2014

Abstract

As serious gaming gets applied more and more in our society, algorithms dealing with related problems get more important as well. One such problem is how characters can keep up a formation while interacting with crowds of varying densities in a realistic manner. The goal of this thesis project was to find a way to do this.

Two different methods of modeling formations were proposed. In the 'formations using Relative Positions' method, regular characters take up a position in relation to a leader character. Together, a set of these characters make up a formation. Alternatively, formations can be modeled as a Dynamic Object. An obstacle is placed in and moved through the environment. Characters re-plan their paths to avoid this impassable obstacle. The user does not see this obstacle, but sees a set of characters instead. These characters are purely visual and are not taken into account for any collision avoidance method.

This paper also introduces two methods to steer the crowd away from the formations. Proxy agents can be set in front of formation agents to force crowd agents to choose a different path instead of traversing the area between two formation agents. Directional waves steer the crowd away from the formation over a larger distance. Characters adjust their paths early on.

Finally, the two methods are combined to show a simulation where crowds of low and medium density moving into a formation are routed away from and around the formation.

Contents

1 Introduction

Virtual representations of the world allow us to pretend we are shooting aliens in computer games, watch medieval knights fight a large-scale battle in a movie, or train a firefighter to make the right decisions without the risk of burning down a building. Both in the entertainment industries and in serious simulations realistic projections of the real world are desirable. A sub-area of virtual representation is crowd simulation. The navigation of pedestrians through these virtual worlds is part of this. Pedestrians should take realistic routes without colliding with other beings, buildings or objects. Much work has already been done in this area, but many issues are still open for improvement in terms of ease of use, realism and cost effectiveness.

Figure 1.1: A crowd on Queen's Day 2010

1.1 Project Motivation

When I started my research project, I was working with E-Semble. This company produces simulation software to educate and train operational and tactical safety and security professionals. Trainings based on E-Semble's software *XVR* usually involve one instructor and one or more students. The instructor creates a virtual environment beforehand. During the training, the students decide how they react in the given situation. By giving orders to the fictional characters ('this firetruck should travel here') they manipulate the virtual situation. I will be doing research related to an application for riot and crowd control. From now on, I will use the abbreviation R&CC for 'riot and crowd control'. The Dutch department of R&CC goes by the name 'de Mobiele Eenheid (ME)', and will occasionally be mentioned specifically in this thesis.

For R&CC leaders, it is important to know where to station objects (like barriers [20]) and/or groups of R&CC to optimally influence the crowd. In large-scale situations it is important to have a controlled flow of pedestrians. When dense crowds occur (like in Figure

1.1), the risk of crushing people exists. In the past, several crowd-related disasters have occurred [13]. Examples include the Love Parade stampede [65], the Hillsborough Stadium disaster [64] and the crushing on the Jamarat Bridge in Mecca [17]. Good planning in advance (crowd management) and reactions on the site (crowd control) can avoid this type of disasters, while still making sure no other undesirable behaviour escalates (such as aggression or people reaching off-limits locations). E-Semble's product XVR is a helpful tool in teaching this kind of planning before and control during events. I will now explain my contribution to the related research area.

1.2 Research Proposal

My research consists of two parts. Firstly, I designed a method to model formations in high density crowds. Secondly, I looked at how several parameters could influence the modeled situation and how they are useful to someone creating a scenario.

1.2.1 Formations in dense crowds

If we look at real-world situations, we can see that good use of formations can be very useful in achieving our goals. Examples can be found in many places.

One example can be found in the animal kingdom. Not just humans, but also animals sometimes hunt in formation. Dolphins [14] (as seen in Figure 1.2) and lions [50] have both been observed to not only hunt in groups but to take specific formations while doing so. This increases their efficiency - prides of lions that used formations obtained more food than prides that did not.

Figure 1.2: Dolphins drive sardines into a giant bait ball. Image by Diver Magazine.

Kirkland and Maciejewski [26] looked into some ways to influence pedestrian movement using robots. They discovered that if the robots were placed well, the pedestrians' efficiency went up.

By stationing R&CC in the right locations, it is possible to influence the movement of a crowd. Formations are particularly effective; they can disperse, contain or block a crowd. Because of their versatile nature they can react to a situation and change their tactic (unlike stationary obstacles that can only block off an area and are generally not easily relocated). Formations are especially good in urban areas because they can help R&CC split a crowd into smaller groups [35]. Figure 1.3 illustrates a top-down view of a formation in a crowd.

Many games simulate wars. In real-world wars, units (both human and mechanized) usually work in formation. To be able to model realistic war situations, quite some research has been done on agents keeping formations (for example [3], [12] and [29]). However, most

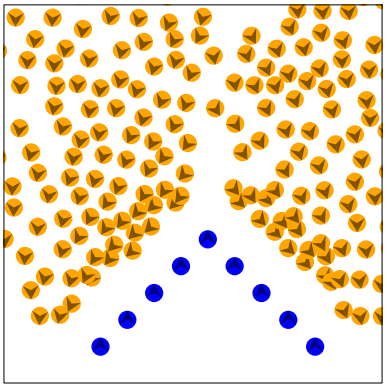


Figure 1.3: A group of agents keeping a formation (in blue) while opposing a crowd.

research assumes a practically static environment with a low density of agents that are not part of the formation. Agents that are part of a formation are able to react to and avoid static objects, but when they encounter other agents, those individual agents will usually just move through the formation in simulations using these methods. This is illustrated in Figure 1.4. Hardly any research has been done on how to keep the formation when encountering dynamic obstacles, and how to keep those other agents from moving through the formation. A related area that is open for research is formations encountering crowds of varying densities.

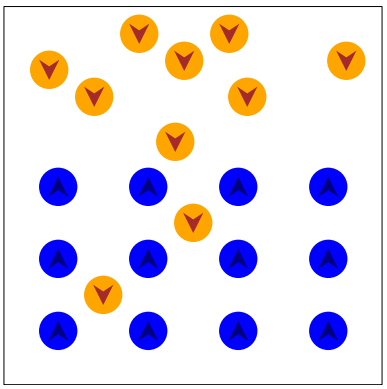


Figure 1.4: Individual agents (orange) slip through a formation (blue).

I have developed a method that models the interaction between a group of agents in formation and a high-density crowd in a realistic manner. My focus in this research has been on situations that R&CC might encounter.

Usually there will be some space right in front of the R&CC squad because people generally avoid getting in close contact with R&CC. It seems likely that a formation might get deformed occasionally. In reality however, R&CC are never breached. When a scenario escalates to a situation where it seems likely that individuals might breach or deform the formation, R&CC retreats and regroups. So it is important that any algorithm dealing with these situations does not allow for formations to be breached.

Both the behaviour of the formation agents and of the crowd agents are important in this research. The focus is on the interaction between these two groups.

Research Questions

The main question I will be trying to answer is 'How can a simulated formation of agents interact with a dense crowd in a realistic manner?' To answer this, I will look at several sub-questions:

- How do we ensure the formation is not breached?
- How do people in dense regions make room for the formation?
- How should the border cases, where people switch between high- and low-density regions, be handled?

1.2.2 Parametrization

Another concept I will be looking into is the parametrization of the situation. The method I design will likely be used in applications for training purposes. It is desirable that the algorithm deals with a wide range of situations. The user needs to have some control over the created situation. Likely examples of desirable variables are the density, the aggressiveness and the pervasiveness of the crowd.

I will be researching which aspects are desirable to influence and what variables in the implementation correspond to these effects.

Research Questions

Here, the overarching question is 'How can we give the user meaningful control over the simulation?'. This can be divided into the following questions:

- What would be useful ways for the user to influence the situation?
- Which parameter(settings) will result in the required behaviour?

1.3 Structure of this work

This work is organized as follows. Chapter 2 will give an overview of related work in the area of crowd simulation. The first section will give a general overview of motion planning methods. The next sections will go into detail on literature that is specifically relevant to my research. These are on high-density crowds, formations, exceptional behaviour and crowd behaviour. The last section of Chapter 2 gives relevant information on how riot control in general, and Dutch riot control in specific works.

Chapter 3 lists some methods that might be adjusted to provide a solution for my research questions. Chapter 4 describes the constraints in which my solution will work, and Chapter 5 gives detailed explanations of the preliminaries to my work.

In Chapter 6, the methods that I developed will be introduced. Section 6.1 describes the two different ways of modeling formations I have used. Sections 6.2 and 6.3 respectively handle the way I have used composite agents to steer crowd agents away from formations, and the Directional Waves method.

The next chapter discusses the results obtained from the proposed solutions, and finally in Chapter 8 some conclusions are drawn from these results. Appendix A contains the report of an interview I had with Martin Klootsema, who works for the *Mobiele Eenheid*.

2 Literature Review

This section will provide an overview of pedestrian navigation in general, and a more detailed account of high-density crowds, formations and exceptional behaviour. The last paragraph contains information on how the Dutch riot & crowd control operates.

2.1 Motion Planning

Navigation consists of many parts. The two parts that are primarily of interest to my research are global navigation (*path planning*) and local navigation (*collision avoidance*). Global navigation deals with selecting an appropriate route to reach the goal. LaValle [27] provides a clear overview of the many different options to approach this problem. Local navigation is about (dynamic) obstacle avoidance while following a given route. Some methods decouple these two parts, while others combine them.

A pioneer in the field of simulated navigation was Reynolds [42]. His *boid model* is the basis for much work in crowd simulation (for an example, see Figure 2.1). Originally intended to model bird-like flocking behaviour, each agent calculates its own path. The basic model consist of the concepts of separation (do not bump into neighbours), alignment (match velocity with that of neighbours) and cohesion (steer towards the average location of neighbours). Later he extended this to include some special behaviour like seek and pursuit [43].

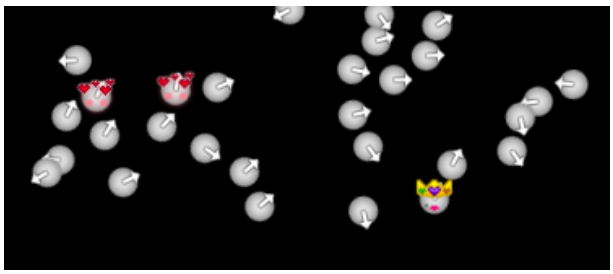


Figure 2.1: Image from the game Disco Fever. As you can see, boids generally have similar directions as the boids around them. Graphics by Joe Kohlmann.

Most global navigation methods can be sorted into three categories: cell decomposition, potential fields and roadmaps/navigation meshes. I will now give an overview with some examples of each of these categories.

2.1.1 Cell decomposition

Methods that use cell decomposition usually combine global navigation and local navigation. Agents usually do not have specific goals, but base their route on local input. When an agent starts moving, it does not know where it will end up yet. These methods can be useful when a visual representation of 'walking people' is required, but the user has no interest in the individual characters.

The environment is divided into a grid. An example of a grid-based method is using *Cellular Automata* (CA). Burstedde et al. [5], Varas et al. [61] and Yue et al. [68] all

developed CA-based methods. I will now describe the general method. For each cell, possible transitions to other cells are stored with a certain probability. Each agent has a preferred direction, and this is combined with the information from his current location. From this probability matrix, a new location is chosen at random. If several agents want to move to the same cell, one is chosen to make the move and the others do not move at all. Figure 2.2 illustrates this.

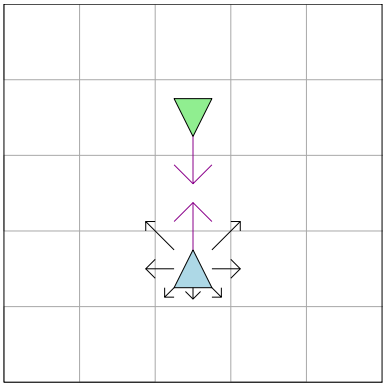


Figure 2.2: Two agents in a grid-based representation. The length of the arrows represents the probability a certain direction is chosen. The purple arrows show the chosen direction. In this case, the green and the blue agent chose the same target cell. One of these will not move in this update.

It can be desirable that agents build on successful routes traversed by other agents. Ants leave trails on the ground to help other ants find food. In CA, similar methods can be applied, as described by Helbing et al. [19]. A counter keeps track of the recent activity in a cell. Taking this counter into account when calculating probabilities results in popular routes getting even more visitors.

Cellular Automata were originally used in research on modeling traffic and led to some successful results, like in the studies of Nagel & Schreckenberg [44] and Wagner et al. [63]. Later they were adapted to handle pedestrians. However, while cars (usually) follow lanes, pedestrians tend to take up all available space. The result is that grid-based methods are much less convincing for steering pedestrians. Another disadvantage is that because cells in both diagonal and straight directions can be chosen, abrupt changes in the absolute velocity can occur. This is another detail that makes visual representations of pedestrians relatively bad. It can give a good indication of the global flow of pedestrians in a specific situation though, so it is often used for simulations where visualization is not important or non-existent. Non-visual simulations are often used to calculate optimization. For example, it is possible to determine good locations for exit doors by looking at pedestrian flow using a CA model. Xiao et al. [66] did some work in this area.

An advantage of CA is that it is fast; no complex calculations are performed, and therefore, each update takes little time. However, it takes much memory as each grid cell has to be stored separately.

2.1.2 Potential Fields

Another approach is the *potential-field* method. One of the first to propose this was Hughes [20] but probably the best known example is *continuum crowds* as described by Treuille et al. [54]. This method combines local and global navigation. It is assumed that

- each person is trying to reach a geographic goal,
- people move at the maximum speed possible,
- a discomfort field exists so that people try to avoid discomfortable regions,
- people try to minimize time, distance and discomfort.

Using continuum dynamics, a potential field is calculated based on these assumptions that steers the people. Crowd density is taken into account because people are more likely to choose paths with low crowd density in order to avoid congestion and collisions. A disadvantage of this method is that it is quite expensive to calculate the potential field. Only a few different fields can be calculated each update, so all agents should be divided into a small amount of groups that share the same potential field. This forces the agents to have the same goals, and in many situations that is unrealistic and thus undesirable.

Chenney uses a similar method called flow tiles [7]. His method mainly focuses on things like water rather than human beings, but can be applied to pedestrians too. Patil et al. [36] propose a method based on a combination of navigation fields and guidance fields. The user can indicate a guidance field at run-time which blends with the original navigation field to form a new potential field. The guidance field allows the user to steer agents in a certain direction at runtime. Jin et al. [21] did something similar using user-specified anchor points. Narain et al. [34] use compressible crowds. Once crowds get denser, a potential field takes over (more in Section 2.2).

Potential fields provide realistic and efficient results when only a few different goals need to be simulated.

2.1.3 Roadmaps and navigation meshes

A common method for global navigation is a *roadmap*. A roadmap consists of vertices and edges that represent collision-free locations. In complex environments, it can be hard to create complete roadmaps. The *Probabilistic Roadmap Method* (PRM) [25] is an influential method that samples collision-free locations and builds a roadmap between them. In theory, you can keep sampling until the complete space is covered. However, in practice it is not realistic to sample infinitely, so the complete space will not be covered. The roads only indicate exact routes and provide no information on the regions around them, so agents have to follow the exact paths. This produces paths with sharp corners and it is not possible for two agents to pass each other. Another disadvantage is that PRM's are expensive to calculate. They have to be pre-computed, so the method is not well-suited for dynamic environments.

Instead of roadmaps, *navigation meshes* are often used. A navigation mesh indicates all traversable areas. This results in much more human-like behaviour as agents are allowed to make small variations like cutting corners and avoiding each other. The two points closest to an agent's start and goal position are determined. A route is then found using a standard planning algorithm such as A*.

Instead of roadmaps, *navigation meshes* are often used. A navigation mesh indicates all traversable areas. This results in much more human-like behaviour as agents are allowed to make small variations like cutting corners and avoiding each other. The two points closest to an agent's start and goal position are determined. A route is then found using a standard planning algorithm such as A*.

An example is the *Explicit Corridor Map* (ECM) by Geraerts and Overmars [15]. This map is created by taking the medial axis (which is a pruned generalized Voronoi diagram) of the environment by making smart use of the GPU.

Navigation meshes are often combined with *social forces* for local navigation. Helbing [18], who coined the term social forces, was the first to apply this. At each point in time an agent has a velocity v. Several forces work on the agent that each influence the velocity, in a Newtonian way. The resulting velocity will be the new velocity. The most important forces that are applied are a force that steers it in its goal direction, forces that make them avoid other pedestrians and obstacles, and forces that keep them within the boundaries of the calculated free-space.

In the *Indicative Route Method* [24], an attraction point is moved over the preferred path based on the ECM. This is considered one of the forces working on the agent, and thus it can easily be combined with social forces.

Another method used for local navigation is *Velocity Obstacles*, first proposed by Fiorini and Schillert [11]. An agent calculates a 'velocity obstacle' for each other agent, given the

current velocity of the other agent. The velocity object is the set of possible velocities that will result in a collision at some point in time, extrapolating the other agents current velocity. Van den Berg et al. [58] [57] introduced the *Reciprocal Velocity Obstacle*, an extension of this method, where agents take into account the fact that other agents are most likely also working to avoid collisions.

2.2 High-Density Crowds

Several works specifically target high-density crowds. A well-known method is *HiDAC* by Pelechano et al. [38] Here, control is split into two levels: a higher level for things like navigation and learning, and a lower level for perception and reactive behaviours. HiDAC is (partly) a rule-based method. An example of a rule would be 'if other agents are close and oscillation might occur, the agent stops'. HiDAC varies many weights to achieve diverse crowd behaviour.

Narain et al. [34] have an elegant solution for crowds of varying density. Agents largely take their own preferred velocity. However, when crowds get denser, a potential field gradually takes over. Once it is really crowded, collision avoidance is basic, which is true in real life too. They take compressibility into account: in denser crowds, agents take up less personal space. As this is an efficient method, large numbers of agents can be simulated.

Van Toll et al. [60] designed a method where agents take density into account while planning their global path. Agents try to avoid congestions, and this results in a more realistic representation. This is illustrated in Figure 2.3.

Figure 2.3: With density information, characters spread out more. Red circles indicate under-used regions. Images by van Toll et al. [60]

2.3 Formations

Quite a bit of work has been done on formations in the field of robotics. Some of this research can be applied in simulation. However, robots are usually expensive items and have a specific function. So the focus has usually been on avoiding collisions (to spare the expensive machinery) and achieving the robot's task rather than simulating natural pedestrian behaviour. This means that quite a few methods in robotics are not directly applicable on human agents. An example is the method by Lin et al. [29]: their robots spiral around until they find their spot in the formation, which has nothing to do with human behaviour. However, some methods do work well with humans too.

Balch and Hybinette [3] propose creating attraction points in relation to agents. Depending on the formation, certain points around an agent attract other agents. For example, for a linear formation, each agent would be attracted to a position to the side of its neighbouring agent. A downside of this method is that when the formation does get deformed (for example because an object has been passed) it can be hard for agents to snap back. This is illustrated in Figure 2.4. Fredslund et al. [12] developed a similar method. They assign each robot a 'friend' - the robot will try to reach a specific position in relation to its friend.

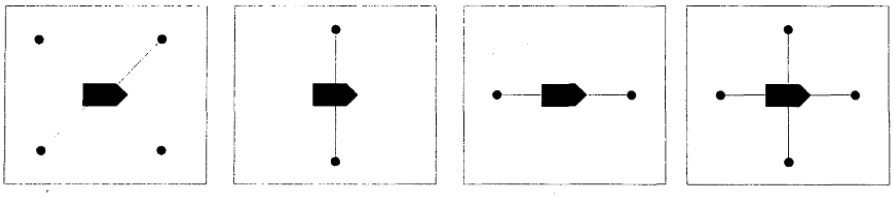


Figure 2.4: Depending on the formation, agents project attraction points for other agents in relation to their own location. These other agents manoeuvre to these attraction points. In these images, the main agent is shown in the middle of each image. The circular images depict the following agents' positions. Images by Balch and Hybinette [3].

Kamphuis and Overmars designed two methods for keeping groups coherent, but they do not take formations into account. One method [23] works with the Corridor Map Method and attraction points. The other [22] plans the path for a group shape using Probabilistic Roadmaps and the motions for the individual agents within the shape with potential fields.

Bochardo et al. [1] propose an extension of the boid model. A leader is assigned that gives commands; the other agents take a location in relation to the leaders location, based on the issued command.

In Li [28] agents are conceived as being part of a 2D elastic shape modeled using the boundary element method. This shape can be deformed by displacing a few control nodes. An extension of the continuum crowds method is used for navigation. Silveira et al. [49] [48] proposed a similar method using potential fields and modeling it as a boundary value problem. This also allows for formations. Chang and Li [6] use a cell based approach to fill a certain region with agents. The user specifies a shape, which is divided into cells and filled.

Quite some work has been done on formations and some good results have been found. However, hardly any work has been done in the area of formations in environments of varying density. Additional research in this area is desirable.

2.4 Exceptional Behaviour

Several papers have proposed methods for achieving certain behaviour that does not usually occur in crowd simulation methods. Exceptional behaviour is an interesting research area as it could be used for many applications.

The first option that comes to mind is *scripting* behaviour. Several examples of scripted behaviour in crowd simulation can be found, like by Ulicny et al. [56] or by Schweiss et al. [46]. Scripting is a good method to achieve specific behaviour, but can be a lot of work. Each type of behaviour requires its own script.

Pelechano et al. [37] have developed an extensive model where agents have 11 emotions. They base their decisions on how to get results as close as possible to their characteristics. They also incorporate a leader model where leaders explicitly exhibit different behaviour from non-leaders according to a rule-based approach. However, I think it would be hard to generalize this method to obtain different behaviour than described in this paper: his model is used in quite a specific case.

Musse and Thalmann [33] divide all behaviour in three levels in the *ViCrowd* method: crowd behaviour, group behaviour and individual behaviour. If some crowd behaviour should occur, it is decided which groups should perform it, and then how the agents should perform it. At certain locations in the environment 'Action Points' are placed. These points trigger certain behaviour (mostly shown in animations). As the points are static, they can not be applied in a dynamic environment.

Sung [53] proposes scalable behaviour. Among all agents in a certain situation (such as at a specific location or in a specific state), some number will perform a certain action. Which agents will perform it is decided probabilisticly. In our problem space, the interaction between formation and crowd is important, which means that specific characters should be

performing specific actions. Assigning actions probabilisticly will not do.

Another example are the *situation agents* as described by Schuerman [45]. These agents are a different set of agents from the crowd agents. In certain regions, a situation agent is placed to enforce specific behaviour. For example, at narrow passages the situation agent can force one group to wait for the other by overriding their velocity. This results in a quicker passage.

Yeh et al. [67] propose using *proxy agents*. These agents are fake agents that belong to a real agent. They are not visible to the user and do not update their own location. The owner updates the fake agent's size and location. However, the other agents treat this agent as a real agent and try to avoid collisions with it. In this way, behaviour like aggression and giving certain agents priority can be imposed. This can be seen in Figure 2.5. A more extensive description of this method is given in Section 5.3.

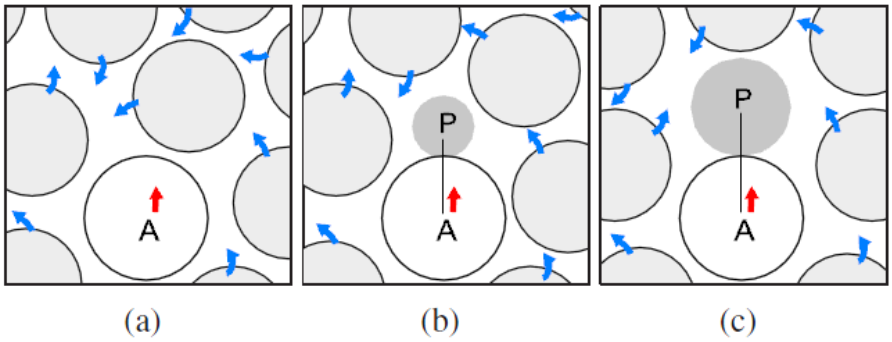


Figure 2.5: Agent A produces a proxy agent to force other agents to make way for him. Image by Yeh et al. [67]

In very dense environments, this method does not work well. The agents do not have the opportunity to make room for the proxy agents, as other agents are behind them. In low density environments, agents could easily avoid proxy agents without being influenced by them as much.

Another category of methods that deal with special behaviour are the methods that mainly work by showing certain animations. An example is Shao's paper [47], that focuses on New York Pennsylvania station. These methods are of no interest to this research as I will not look at animation.

In short, there are quite a few different approaches to modelling behaviour not captured in general path planning. Some interesting starts have been made but some follow-up work is desired.

2.5 Crowd Behaviour

To be able to check if the behaviour created by my algorithm is realistic, it is important to be sure what exactly is realistic behaviour. In order to determine this, I have looked at several papers and video's.

2.5.1 Crowd Psychology

Some papers have been written on behaviour in large crowds. Often the focus is on crowd psychology, as in the papers by Reicher [41]. He has done extensive research on what makes people behave differently in a crowd, taking specific occurences as examples. He has interviewed many people involved in the St. Paul's riots in Bristol in 1980 [39], and in the aggressive 'Battle of Westminster' in 1988 [40].

All the researched situations show that being part of a crowd, doesn't suddenly make people behave aggressive. A very important factor in whether or not the crowd turns aggressive, is a sense of fairness. Crowds will only fight their opponents (often, including in

the two described cases, the police) if they believe they are treated unjustly (examples can be when the police blocks off a road they were told to follow due to miscommunication, or when the police treats regular police members as villains along with the actual aggressive crowd members). They feel their aggressive behaviour against the police is justified, but do not extend it to outsiders. Even when the crowd, for example, throw stones at police cars, they might boo at crowd members that throw stones at uninvolved bystander's cars.

When the crowd behaves aggressively, this often makes the police feel treated unjustly in turn. They are just there to follow orders. This results in more aggressive behaviour on the police's side as well. This is how situations escalate.

At the UEFA Football Championships in Portugal in 2004, they applied these new insights [52]. Deployed tactics from the police involved not coming off as aggressive (by working in pairs instead of in squads) and treating aggressive people as individuals and singling them out from the crowd. This Football Cup was particularly effective in avoiding hooliganism.

2.5.2 Crowd Movement

When searching on YouTube, many video's taken at the scene of a riot can be found. Most of these look at sub-scenes. The focus is on one extreme act of aggression (lighting a car on fire, destroying a trash can). What is interesting to see is that there are often just a few main aggressors. Most people are just onlookers. They follow what happens with interest. They might join in on the shouting, but they do not destroy property themselves.

It is much harder to find videos that show a global picture though.

Low [30] has found that crowds often show herding behaviour. In evacuation situations, even though there are several different exits, most people will take the same exit. If they see others taking that route, people expect it to be a good direction. Another interesting detail is that people generally try to exit a building through the entrance they take. They know from experience that this is a good route. Therefore, they try to take it again. These two details result in the fact that people do not work like fluids as they are a group of individuals rather than one mass.

Moussaïd et al. [32] have done extensive studies on the effects of groups on the movement of a crowd. Up to 70% of people in a crowd walk in groups within a crowd. This severally effects the movement of the crowd. It would be interesting to see how this is influenced by the emotions of a crowd. In a situation where a large part of the crowd is scared, do they still try to walk side by side?

2.6 Riot Control

In the Netherlands, riot & crowd control is called *Mobiele Eenheid* (ME). Members of the ME all belong to the police force. In June 2012 I interviewed Martin Klootsema, who works as Inspector/Specialist Conflicts and Crisismanagement for police unit Amsterdam-Amstelland. This interview is documented in Appendix A. I will now give a brief overview of insights obtained from this interview that are relevant to my research. I will end this section with information regarding what types of formations are commonly used.

2.6.1 ME

The smallest unit in the ME is called a group, which consists of 10 members (including a trained chauffeur) and a group commander. Two groups form a section, two sections form a platoon, and two or more platoons form a company. Each unit (groups, sections, platoons and companies) has its own commander [2]. In Figure 2.6 this structure is illustrated.

The goal of any ME action is to make the opponents realise they don't stand a chance and to make them leave voluntarily because of this realisation. Crowds are always asked twice to leave before the ME takes any action. If at any moment the ME seems to be on the losing side, they retreat and regroup. They pull back to the vehicles under cover and come back with additional units. As a result *formations are never breached*.

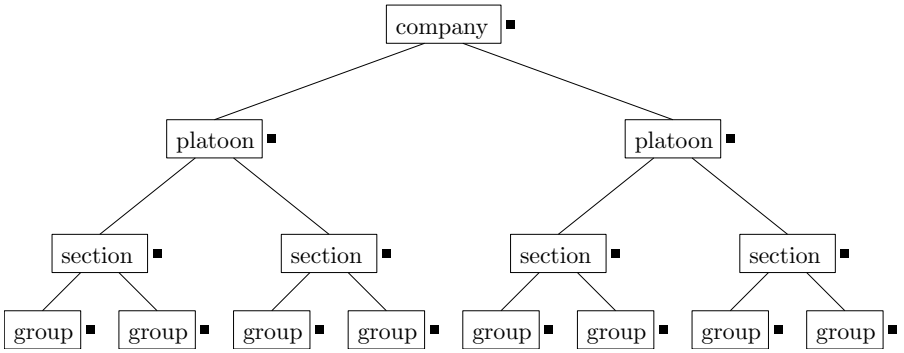
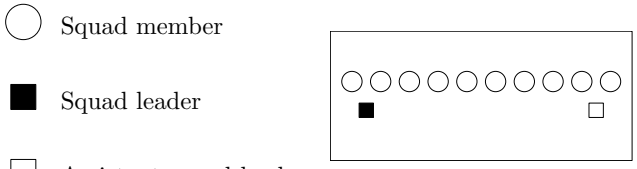


Figure 2.6: Dutch riot control. Black squares indicate commanders. A company can consist of more than two platoons.



(a) Legend

(b) Legend

Figure 2.7: Squad Line Formation

Whenever the crowd throws up barricades, the ME removes these. This is incorporated in the ME's training. The ME never creates its own barricades.

At any time four platoons can immediately be deployed without influencing regular police work. Occasionally platoons from other regions are borrowed, for example when the Netherlands reached second place in the World Cup Football in 2010. Some additional types of units exist for special occasions, such as water cannons, horses or shovels. I will not take these special units into account in my research, as they are not deployed in a formation (except for the horses, but they fall under the general case).

At the time of this interview, all training of members of the ME took place in real life. The ME did not use virtual environments as a part of their training yet. Training of basic personnel focusses on the shaping of and moving in formations, the irregularity of the crowd, and trusting each other. The application of this research would be more suitable to the training of commanders than of basic personnel, as commanders have to apply tactics by judging situations and issueing commands on where the formations will move.

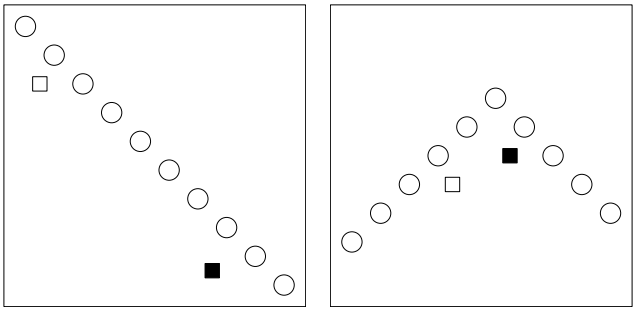
I will now give an overview of the formations used in the US Army.

2.6.2 Formations

According to US Field Manual 19-15 [35], the US army uses 5 basic types of formations in crowd control. Information on which specific formations the Dutch ME uses is not available publicly. It is highly likely they are similar to the formations the US army uses and therefore I will now describe those formations.

The US army squads (similar to the groups of the ME) consist of 10 members, a squad leader and an assistant squad leader. Only three formations are commonly used. They are the line, echelon and wedge formations. These are illustrated in Figures 2.7 and 2.8. Occasionally two other formations are used. These are a diamond and a circle. These two shapes are often used by *snatch squads*. A snatch squad is deployed to pick up one or two people in a crowd to arrest them.

It is possible to scale these formations when platoons come into play. Besides just scaling these formations, some extra options appear. Most of these options can be divided into



(a) Echelon **(b)** Echelon

Figure 2.8: Squad Echelon and Wedge formation

close support (an extra line right behind the existing line), general support (the additional squad(s) lines up behind the main formation, waiting) and lateral support (as shown in Figure 2.9).

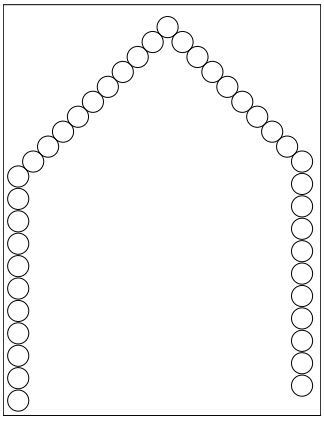


Figure 2.9: A platoon wedge with two squads in lateral support.

From these formations, we can conclude that the positions shown in Figure 2.10 are the possible positions one agent can take in relation to another agent. When a formation is modeled as agents snapping into places relative to other agents, these are the relevant locations.

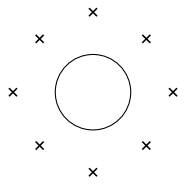


Figure 2.10: Locations of interest in relation to a squad member. The crosses indicate potential locations of other squad members.

3 Possible Solutions

To solve the problems stated in Chapter 1.2, I thought of several (partial) solutions. The goal is a solution where a formation can interact with a dense crowd, where the formation is never breached, and the crowd makes way for the formation in a realistic manner. I will now mention some promising possible approaches, along with their intended advantages and potential problems.

3.1 Attractors and Repulsors

In this approach, the user would be able to place attractors and repulsors in the situation (or attach them to dynamic objects, like the formation). These would exert a force that is taken into account by agents in the collision avoidance calculations. This method is closely related to Helbing's social forces [18]. This repulsor should work over large distances and weaken with larger distance.

Intended advantages

- The system can be used widely (beyond just formations).
- I expect there are not that many special cases.
- The crowd feels the influence of the formation even if their members can not see them.
- The system is easy to use.

Potential problems

- What happens with the crowd that is located around a corner? Do they feel the formation's influence in the same direction?
- This method is probably not the most realistic solution.
- Is it possible to only repulse in a certain direction? Is that still easy to use?

3.2 Composite Agents

Agents place fake agents (proxy agents) to create space. The user does not see these agents, but other agents do try to avoid the space taken up by these. Agents in formations can use some proxy agents in front of them to create the required space.

Intended advantages

- Creates extra space for the formation members.
- Crowd agents cannot traverse open areas between formation members.

Potential problems

- Most commonly used collision-avoidance methods take into account the fact that other agents also apply collision avoidance. However, proxies do not apply collision avoidance. The result is that when using one of these methods, agents do not completely avoid collisions with proxies.
- Agents trying to avoid proxies don't have space to flee to the back.
- The system is not strict enough, i.e. agents still slide between the formation members.

3.3 Compressibility

This method is largely based on the paper "Aggregate Dynamics in Dense Crowds" by Narain et al. [34]. In high density situations, a flow tile approach takes over path planning. In low density regions, agents follow their own paths. A weighted approach is used so that the denser the region, the more influential the flow tiles are. In this case, formations could influence the tiles close to them, to make agents leave the area close to them. Tiles experiencing a high flow into them, would flow away from this source as well. In this way, formations could indirectly influence regions relatively far away.

Intended advantages

- Medium density regions are no special case, so they solve themselves.
- People are steered away from the formation, even if it is against their wishes.
- It is an efficient approach, especially at high-density regions.

Potential problems

- It might be hard to create different characteristics for different crowd members.

3.4 Finite Elements Method

Another approach would be to use something related to the *finite elements method* (FEM) [8], [4], [69]. FEM is commonly used in physics, but has not been adapted to use in crowd simulation yet.

In FEM, a grid (2D or 3D) is laid out over the area. When forces are applied on a point, these are transferred through the grid. These forces deform the grid. Usually, this is used to calculate impact on materials. However, I think it could be adapted to calculate pedestrian locations. This method is similar to the compressibility method. These could probably be combined. For example, pedestrians could base their behaviour on a weighted mix between the grid and their own goals, just like in the compressibility method.

Intended advantages

- Easy to transfer forces to far away.

Potential problems

- How does the grid influence the pedestrians?
- Where should the grid be placed?
- Traditional FEM methods require some locations in the grid to be 'pinned down' If not, the whole object would just move when a force was applied. How do you decide which points should be pinned down?

3.5 Fluid Dynamics

Quite a bit of research has been done on fluid dynamics, for example by Tritton [55] or Ferziger et al. [10]. It might be possible to model the crowd as a fluid, and the formation as an object of solid material that interacts with the crowd.

Intended advantages

- Probably a realistic result for high-density crowds.
- Much work has been done in this area, including some on using fluid dynamics to model crowds. It should be relatively easy to adapt an existing method to work with crowds.

Potential problems

- How do you handle border regions? In fluid dynamics surface tension is an important issue. Pedestrians behave differently.
- How do you keep individual's plans in mind?
- The current fluid dynamics methods for pedestrians don't work all that well.

3.6 Decision

Of all the stated possible approaches, it seems that a method that has elements of the proposed finite elements method and/or the compressibility method is the most promising for a realistic solution.

These methods smoothly blend high- and low-density regions. They combine personal behaviour with crowd behaviour, which is an important aspect in this case. By assigning different weights to different characters it is easy to assert different behaviour and keep a sense of individualism. Individuals who strongly pursue their own goals could show 'special behaviour', like throwing bottles at riot control.

Also, these methods have elegant solutions to transfer forces to locations further on in the crowd. Especially in dense regions, this is an important aspect. The crowd should influence the crowd behind it, but not everyone should move the exact same distance at the exact same time, as that would show unrealistic behaviour. These methods have an elegant solution to slowly influence the crowd over larger distances.

My research will focus on finding a solution using this compressibility approach. An extensive description of the aggregate dynamics method by Narain et al. [34] can be found in Section 5.4.

For XVR, a simpler, less realistic method might be a good solution. In this case, the Repulsor/Attractor method might work well. Users would have a high level of control of the simulation. Another advantage is that it could easily be applied in settings other than with formations too. However, I feel this is not a very elegant or realistic solution. For me, these are reasons not to work on such a solution: unrealistic, crude methods with a high ease-of-use already exist. I do not feel I need to add another implementation to these.

Another solution that might work well for XVR is the composite agents method. It seems a highly versatile, simple and elegant solution. I have decided to try and find a solution using these composite agents as well. The implementation and its advantages and disadvantages are described in Section 6.2.

In conclusion, I will be exploring options based on composite agents, and on the compressibility and/or FEM ideas. In this second approach a grid will be laid over the scene. Formations influence the forces working in this grid. The grid is used to calculate pedestrians movement. How exactly the grid is used to calculate the velocities, and how the formation's forces are transferred between grid cells, will be explained in Section 6.3.

4 Constraints and Limitations

4.1 Assumptions

I will build a 2D, top-down view application to test my theories about keeping formations. This application will have a low focus on graphics. It should only show an abstract (but clear) representation of agents.

Some functionality will be implemented according to existing methods. I will not personally do any research on those subjects. Examples are the Explicit Corridor Map (ECM) by Geraerts and Overmars [15], used to represent the environment, the Indicative Route Method (IRM) by Karamouzas et al. [24], used for path planning, and a vision-based approach to collision avoidance by Moussaïd et al [31]. An extensive description of these methods can be found in Chapter 5.

My research will focus on dense city areas. The tested environments will be simple, such as the one shown in Figure 4.1. This means performing experiments in narrow streets (easily covered by the width of one formation) and possibly streets crossing.

Figure 4.1: An example of an environment to be used.

4.1.1 Scenarios

The method I will develop can be deployed in several situations. On a high level, examples within XVR that use some kind of formation interacting with a dense crowd can be:

- Riot & Crowd Control working in formations,
- ambulance staff carrying a stretcher between them,
- fireman carrying a large hose,
- any other situation in which people try to keep a position relative to another person.

In my research, I will focus on Riot & Crowd Control. This comes down to larger formations and an authoritarian influence. Within this setting, several scenarios can be thought of. These scenarios were based on the ones described by Visschedijk [62]. They were written around several emotional settings.

Aggression

In a football stadium, a match recently started between two rival clubs. At previous matches, riots have started between some groups of supporters. Riot Control should prevent fighting and riots breaking out.
Current situation: The match has finished a while ago. Supporters went downtown. Now, both groups of supporters are assembling at a large square.

Panic

A peace demonstration takes place in The Hague. People are demonstrating against discrimination of a minority in an African country. The demonstration will travel to this country's embassy and offer a petition. About 10,000 people showed up, including women and children. Riot Control should keep the embassy safe from attacks.
Current situation: After offering the petition peacefully, the demonstration starts to move on. However, at the end of the street, people sympathizing with the majority of the African country show up and start throwing stones at the demonstrators.

Anger

A peace demonstration takes place in The Hague. People are demonstrating against discrimination of a minority in an African country. The demonstration will travel to this country's embassy and offer a petition. About 10,000 people showed up, including women and children. Riot Control should keep the embassy safe from attacks.
Current situation: The ambassador refuses to come out and receive the petition. The crowd grows angry.

Neutral

It is Queensday in Amsterdam. Many people have come to the capital to celebrate.
Current situation: Some streets should be kept free from the crowd for better circulation. R&CC should make sure people do not enter a specific street.

Elation

In a large city in Afghanistan, election results have just been announced. One of the candidates won with a small majority. Many civilians are out on the streets to listen to the celebratory speech and celebrate victory. The crowd consists of both old and young people from different backgrounds. R&CC should avoid riots and destruction.
Current situation: A large group of people just reached the square where the winner will give a speech in a few minutes.

Scared

The location is a military camp in a war-stricken area. Lately, there has been some unrest and many civilians would like to find safety at the camp.
Current situation: A group of about 500 civilians (consisting of men, women and children) is approaching the camp. The group keeps growing and the civilians would like to enter the camp.

I will focus on aggressive settings. The behaviour that occurs in this type of situation is described in the next section.

5 Preliminaries

My implementation makes use of some functionality (that is still open for research) according to existing methods, but I will not personally do any research on those subjects. These are the Enhanced Corridor Map to map the environment and the Indicative Route Method for local navigation. I will briefly describe these in Sections 5.1 and 5.2.

This chapter also describes the two methods I have used as a bases for my research. These are "Composite Agents", described in Section and "Aggregate Dynamics in Dense Crowds". An overview of how these methods work is given in Section 5.3 and 5.4.

5.1 ECM

To map the environment, the Enhanced Corridor Map as described by Geraerts et al. [15] was used. The ECM describes the free space in the environment. ECMs make use of the medial axis.

To determine the medial axis, we look at the closest point on the boundary of the environment from any point in the environment. The Voronoi Diagram is the set of all points where the number of closest points on the boundary is 2 or more. Figure 5.1 shows such a diagram.

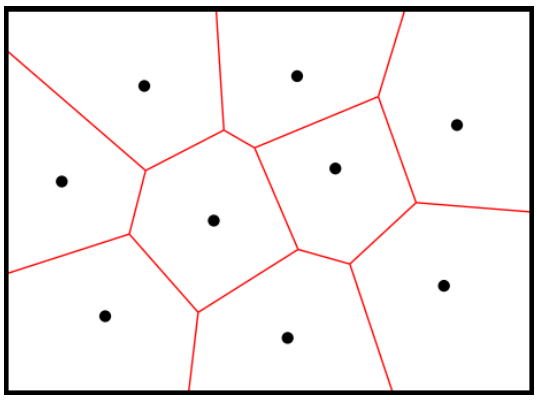
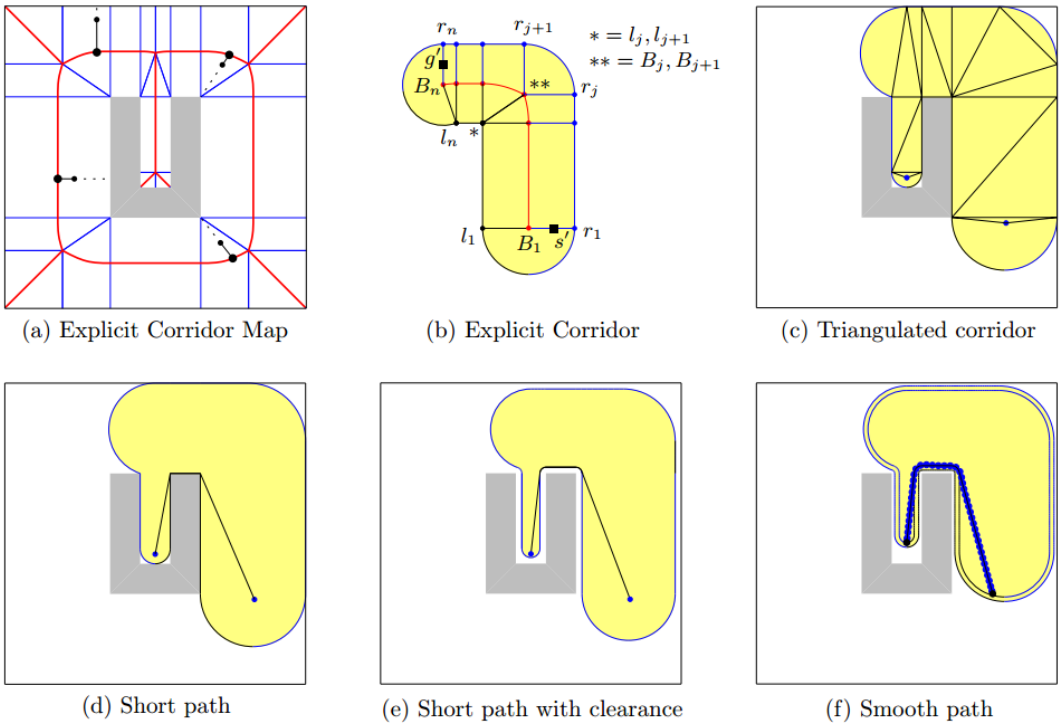


Figure 5.1: The medial axis of a set of points. The boundaries of the image are not taken into account.

The ECM uses the GPU to calculate the medial axis efficiently. As it is not relevant to my research, we will not go into the details here.

Along with the medial axis, *event points* are found. These are the points where the shape of the medial axis changes. Together, they form the Explicit Corridor Map (Figure 5.2a).

The ECM can be used for path-planning. To find a path from a start position s to a goal position g, two points s' and g' close to the original points are found on the ECM. A graph-search algorithm such as A* [16] can then be used to find a shortest path. This path is called the backbone path. The walking space around this path can be described as an Explicit Corridor (Figure 5.2b). A so-called funnel algorithm can be used to find a short and smooth path through this corridor (5.2d).



(a) Explicit Corridor Map (b) Explicit Corridor (c) Triangulated corridor

(d) Short path (e) Short path with clearance (f) Smooth path

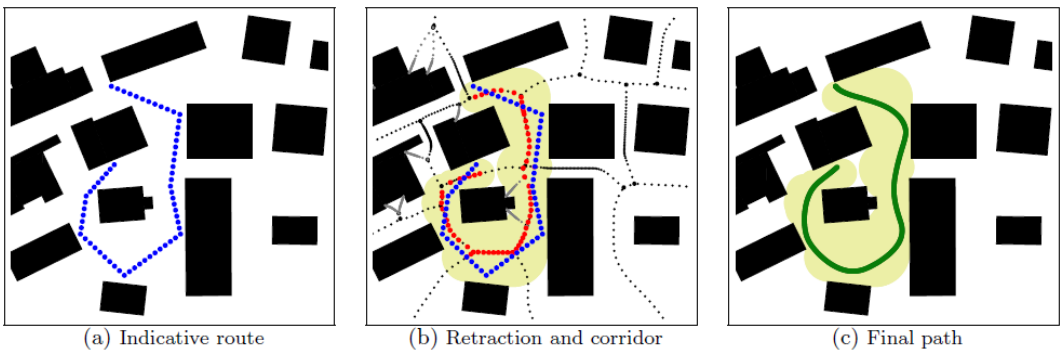
Figure 5.2: Using the ECM for path planning. (a) Examples of query points (small dots) and their retractions (large dots) in an ECM. The medial axis is shown in red; closest-point annotations are shown in blue. (b) Example of an Explicit Corridor. (c) Triangulating a corridor. (d) Shortest path within the corridor, obtained by using a triangulation. (e) Shortest path with clearance, obtained by triangulating a shrunk corridor. (f) Smooth path, obtained with a force model. Description and image (a) by van Toll [59] and images (b)-(f) by Geraerts [15].

For a more extensive description of the ECM, we refer the reader to [15]. I will now describe how the Indicative Route Method can be used for steering.

5.2 IRM

The Indicative Route Method as described by Karamouzas et al. [24] is a general framework for steering a character through a corridor.

The *indicate route* is an indication of the route that will be followed by the character. There are many ways how such a route can be found. Examples are by applying an algorithm such as A* on a roadmap (such as the ECM), or by user input. This route is then discretized and retracted onto the medial axis.



(a) Indicative route (b) Retraction and corridor (c) Final path

Figure 5.3: The Indicative Route Method. Images by Karamouzas et al. [24]

The character is considered a disk-shaped object that can be moved by attractive and repulsive forces (much like in Helbing's social forces method [18]). At each point in time, an attraction point along the original indicative route is chosen, which applies a pulling force on the agent. As these points are frequently moved, this results in a smoothed version of the indicate route. Figure 5.3 illustrates the Indicative Route Method.

On top of the IRM, a collision avoidance method can be applied. In my implementation, a vision-based approach by Moussaïd et al. [31] was used.

5.3 Composite Agents

Yeh et al. [67] have developed a method called composite agents. An agent A_i can have one or more proxy agents $P_{i,j}$. The combination of a regular agent with its proxy agents is called a composite agent. Proxy agents do not have their own goals and behaviours, but are steered by their parent agent. They are not visualized, so the user does not see them. However, other agents do take the proxy agents into account when calculating collision avoidance. The result is that the agents avoid some seemingly empty spaces. As the parent agents do not take their own proxy agents into account, they can use this space to fulfill their goals.

Yeh et al. describe the relationship between agents and proxies as:

$$proxy(A_i) = \begin{cases} \emptyset & \text{for basic agents} \\ \{P_{i,1}, P_{i,2}, ..., P_{i,m}\} & \text{for composite agents} \end{cases}$$
$$parent(P_{i,j}) = A_i.$$

In the original paper, several applications were proposed. They have used proxy agents to model aggression, social priority, authority, protection and guidance in complex scenes.

Aggression To model aggression, a proxy agent is placed directly ahead of the parent agent. This forces agents to leave some space right in front of the agent and gives him a chance to move faster. See Figure 5.4.

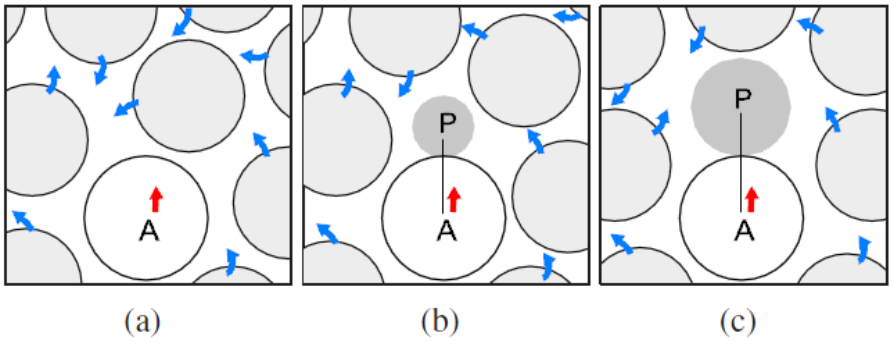


Figure 5.4: Agent A produces a proxy agent to force other agents to make way for him. As A's urgency increases, P grows. Image by Yeh et al. [67]

Social Priority In certain cases, some people get priority over others. An example is when exiting a train. To model this, a proxy is placed in the doorway. The agents exiting the train are allowed to move out, while the agents entering will have to wait for the proxy to disappear. This is illustrated in Figure 5.5.

Authority When several authority figures try to pass a crowd while following each other, people automatically move aside. To model this, passing agents leave proxies behind. These slowly shrink over time, so the proxy right behind an agent is larger than the proxies further away. This is illustrated in Figure 5.6.

Protection & Guidance Proxy agents can also be used to show protection and/or guidance behaviour. For example, a mother steers her child in a dense crowd. She guides

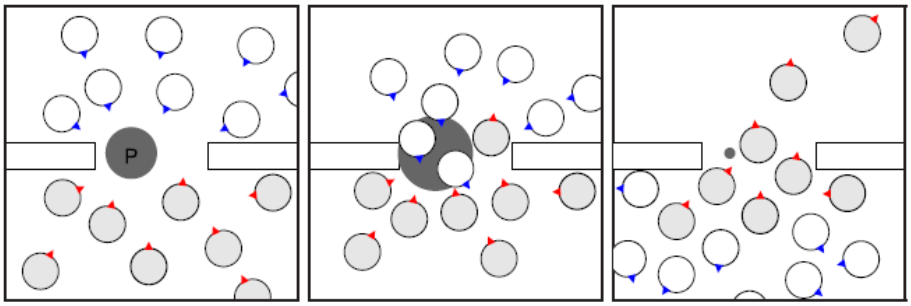


Figure 5.5: The white agents should be allowed to pass the doorway first. They place proxies in the doorway. As the white agents approach, the proxy grows. When they have passed, the proxy shrinks and the grey agents can pass. Image by Yeh et al. [67]

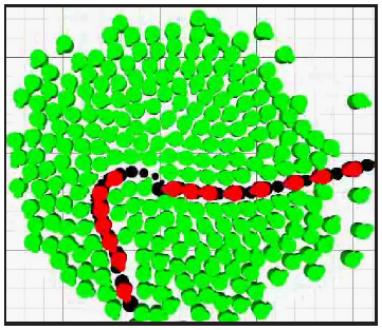


Figure 5.6: The red agents are allowed to step on the trail (in black). The green agents keep these spaces clear. Image by Yeh et al. [67]

the child in the right direction and keeps him away from collisions and other dangerous locations. Figure 5.7a and 5.7b show protection and guidance behaviour respectively.

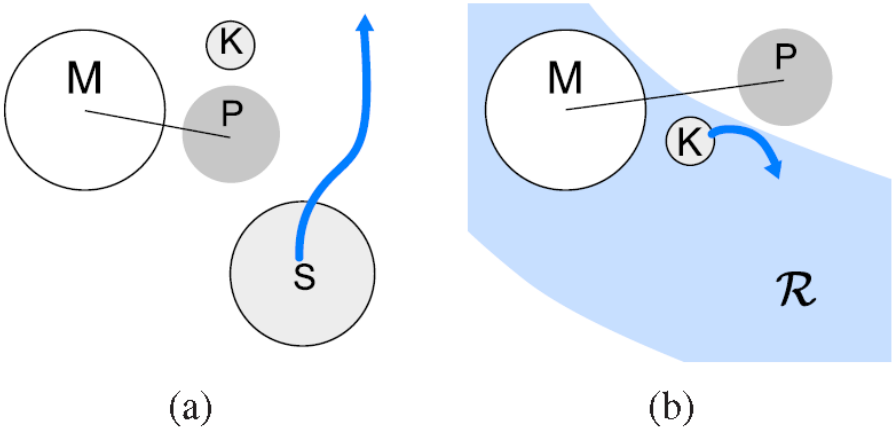


Figure 5.7: In (a) a mother places a proxy between her child and an approaching stranger. In (b) she uses a proxy to keep the child on the right path. Image by Yeh et al. [67]

I have applied the principal of proxy agents to create space in front of formations. Section 6.2 describes how this was applied.

5.4 Aggregate Dynamics for Dense Crowd Simulation

Rahul Narain et al. have developed "Aggregate Dynamics for Dense Crowd Simulation" [34]. This paper proposes an approach to simulate very large, dense crowds. This section

will describe the method designed by Narain et al. All formulas in Section 5.4 were taken from this paper.

5.4.1 Overview

The aggregate dynamics approach has two significant results.

- Agents modify their speed in such a way as to match the speed of the agents around them.
- When the density of agents in a certain region is too high, no other agents enter this region.

I will now give a short description of how this result is obtained. Section 5.4.2 gives a more detailed description of the implementation.

An environment with agents A exists. Each simulation loop consists of the following steps.

1. Each agent i plans its path and calculates its own preferred velocity $\tilde{\mathbf{v}}_i$. This can be done using any global planner. In my application, ECM and IRM were used to calculate global navigation.
2. A grid is laid out over the area. Each grid edge stores the average velocity $\tilde{\mathbf{v}}$ of the agents around it. Each grid cell stores the agent density ρ in it. In fluid simulation this is called a staggered grid.
3. The stored average velocities are modified so that in grid cells where the density ρ would exceed the maximum density ρ_{max}, no extra agents enter the region.
4. Agents calculate their final velocity based on a combination of their own preferred velocity and the velocity field $\mathbf{v}$. In denser regions, the velocity field is weighted higher.

The result is a fluid simulation where agents follow streams of agents around them, which gives realistic results. Figure 5.8 shows two agents and the velocity field. The following section will give a more in-depth explanation.

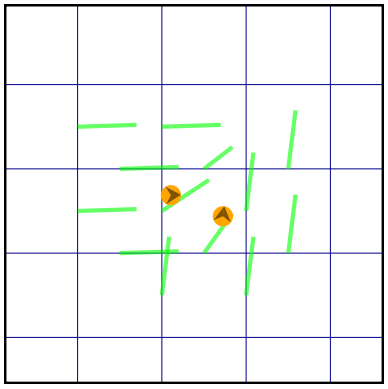


Figure 5.8: The two agents have perpendicular directions. The green lines show the scaled average velocities of the agents close to them.

5.4.2 In-depth explanation

Calculating the grid

The algorithm starts with agents calculating their preferred velocity according to any global planning mechanism. Narain et al. [34] used a roadmap-based approach but in my implementation, ECM and IRM were used. The approach allows for any global planner to be used. From these preferred velocities, the velocity field is calculated.

The continuous fields ρ and $\mathbf{v}$ are stored on a staggered grid. The density ρ is stored in each cell, and the average velocities $\tilde{\mathbf{v}}$ are stored on the edges. To obtain these values, the values of nearby agents are weighted based on the agent's position $\mathbf{x}_i$ and summed.

$$\rho(\mathbf{x}) = \sum_i w_{\mathbf{x}}(\mathbf{x}_i) m_i \tag{5.1}$$

$$\tilde{\mathbf{v}}(\mathbf{x}) = \frac{\sum_i w_{\mathbf{x}}(\mathbf{x}_i)\tilde{\mathbf{v}}_i}{\sum_i w_{\mathbf{x}}(\mathbf{x}_i)} \tag{5.2}$$

The mass m_i is set at 1. Agents are considered nearby if their horizontal or vertical projection on the middle of the grid edge under consideration is less than the width of a cell. This is illustrated in Figure 5.9.

This procedure converts the crowd to a continuous representation.

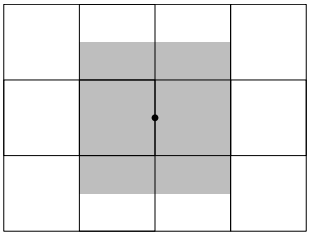


Figure 5.9: All agents in the grey region are considered for the average velocity at the indicated point.

The UIC projection

The crowd's state and preferred motion are represented as continuous fields of density ρ and preferred velocity $\tilde{\mathbf{v}}$. Their evolution over time can be seen as a fluid-like system. However, when looking at compressibility it does not correspond directly to a physical fluid. The crowd is not purely incompressible (when many people are close, 'personal space' is much smaller than in empty regions), but it is not purely compressible either (when the density gets too high, people start dying). Narain et al. consider a crowd as a *unilaterally incompressible* fluid, which is a hybrid of the two. A constraint is imposed on ρ, so that it can never increase beyond a maximum value ρ_{max}. If $\rho = \rho_{max}$, agents are packed as closely together as possible.

This can be described as the *Unilateral Incompressibility Constraint* (UIC).

$$\rho \leq \rho_{max} = 2\alpha/(\sqrt{3}d_{min}^2) \tag{5.3}$$

It is assumed agents never come closer together than a specified distance d_{min}. The constant factor $\alpha \leq 1$ is used to represent the fact that people are rarely perfectly packed.

Gradients and Divergence

The concepts of gradient, divergence and flux will now be explained, as they are used to calculate the UIC.

The gradient $\bigtriangledown$ of a given point in a scalar field is the direction of the greatest rate of increase at that location. So in our 2D vector field, this is the following vector:

$$\bigtriangledown \mathbf{v} = (\frac{\delta \mathbf{v}}{\delta x}, \frac{\delta \mathbf{v}}{\delta y}). \tag{5.4}$$

The divergence, noted as $\bigtriangledown \cdot$, is the magnitude of that increase. This is calculated as follows:

$$\bigtriangledown \cdot \mathbf{v} = (\frac{\delta \mathbf{v}}{\delta x} + \frac{\delta \mathbf{v}}{\delta y}). \tag{5.5}$$

Divergence can be used to calculate flux. Flux is defined as the rate of flow of a property per unit area. So in this case, this would be the rate of agents (the property) leaving or entering (the flow) a cell (the unit area).

Density constraints

The field $\tilde{\mathbf{v}}$ directly influences ρ. The relationship between ρ and $\tilde{\mathbf{v}}$ can be described as

$$\frac{\delta\rho}{\delta t} + \bigtriangledown \cdot (\rho \mathbf{v}) = 0 \tag{5.6}$$

If ρ will exceed ρ_{max}, $\tilde{\mathbf{v}}$ is corrected to ensure that ρ remains below ρ_{max}. This ensures that the UIC is maintained at all times.

The complicated part is to determine the corrected velocity $\mathbf{v}$ which stays as close as possible to $\tilde{\mathbf{v}}$ while making sure $\rho \leq \rho_{max}$.

To calculate this, a new scalar 'pressure' p is introduced. If the maximum density is not reached, there is no pressure on the agents. When there is any pressure, no additional agents can enter the area. $\mathbf{v}$ is adjusted to ensure that. This can be written as follows:

$$\rho < \rho_{max} \Rightarrow p = 0, \tag{5.7}$$

$$p > 0 \Rightarrow \bigtriangledown \cdot \mathbf{v} = 0. \tag{5.8}$$

This means pressure only acts in regions where density is at the maximum. It is assumed that agents try to reach their maximum speed. From these conditions, it can be shown that the optimal solution is of the form

$$\mathbf{v} = v_{max} \frac{\tilde{\mathbf{v}} - \bigtriangledown p}{\|\tilde{\mathbf{v}} - \bigtriangledown p\|}. \tag{5.9}$$

Numerical method

Performing the UIC projection on the simulation grid is split into two operations.

$$v_{max} \frac{\tilde{\mathbf{v}} - \bigtriangledown p}{\|\tilde{\mathbf{v}} - \bigtriangledown p\|} = \text{renorm}(\text{psolve}(\tilde{\mathbf{v}})), \tag{5.10}$$

where

$$\text{psolve}(\tilde{\mathbf{v}}) = \tilde{\mathbf{v}} - \bigtriangledown p, \tag{5.11}$$

$$\text{renorm}(\tilde{\mathbf{v}}) = v_{max} \hat{\mathbf{v}} / \|\hat{\mathbf{v}}\|. \tag{5.12}$$

Collision avoidance is performed by psolve. Renorm just normalizes the velocity to speed up the flow.

To find p, $\tilde{\mathbf{v}}$ is substituted by psolve($\tilde{\mathbf{v}}$) in Equation 5.6. This results in

$$\frac{\delta\rho}{\delta t} = \bigtriangledown \cdot (\rho \mathbf{v}) + \bigtriangledown \cdot (\rho \bigtriangledown p). \tag{5.13}$$

In the discrete setting with cell size c, the gradient of pressure $\bigtriangledown p$ is defined at cell edges by the difference between pressure at adjacent cells. The divergence $\bigtriangledown \cdot \mathbf{u}$ of some vector field $\mathbf{u}$ is found by summing the flux over the cell edges.

$$\bigtriangledown \cdot \mathbf{u} c^2 \approx \sum\nolimits_{edges} \mathbf{u} \cdot \mathbf{n} c, \tag{5.14}$$

where $\mathbf{n}$ is the normal of the relevant cell edge.

At any timestep n the density vallues ρ^n and pre-projection velocities $\tilde{\mathbf{v}}^n$ are known. The density at the $(n+1)$th timestep is then given by

$$\rho^{n+1} = \rho^n - \bigtriangledown \cdot (\rho^n \tilde{\mathbf{v}}^n)\Delta t + \bigtriangledown \cdot (\rho^n \bigtriangledown p^n)\Delta t. \tag{5.15}$$

From this, a *linear complementarity problem* can be deduced. An LCP is of the form

$$\begin{aligned} \mathbf{z} &= \mathbf{A}\mathbf{x} + \mathbf{b}, \\ \mathbf{z} \leq 0, \qquad \mathbf{x} &\leq 0, \qquad \mathbf{z}^T\mathbf{x} = 0. \end{aligned} \tag{5.16}$$

When the substitution $\sigma = \rho_{max} - \rho$ is used, this problem can be modelled as an LCP with

$$\mathbf{z} = \sigma^{n+1}, \tag{5.17}$$
$$\mathbf{b} = \sigma^n + \bigtriangledown \cdot (\rho^n \tilde{\mathbf{v}}^n)\Delta t, \tag{5.18}$$
$$\mathbf{A}\mathbf{x} = - \bigtriangledown \cdot (\rho^n \bigtriangledown \mathbf{x})\Delta t, \tag{5.19}$$
$$\mathbf{x} = p^n. \tag{5.20}$$

Narain et al. used an LCP solver by Dostál and Schöberl [9]. In my implementation, I do not solve the LCP. Instead, I use a simplified version. When $\rho \geq \rho_{max}$, I simply do not allow any more characters to enter the cell.

Agent Motion

In dense regions, the agents are constrained by the flow of the crowd. In these regions, the interpolated grid velocity at the agent position $\mathbf{v}(\mathbf{x}_i)$ is used as the velocity of agent i. However, in regions of low density, agents' own preferred velocity plays a bigger part. To find the final velocity, an interpolation between the continuum velocity $\mathbf{v}(\mathbf{x}_i)$ and the agents preferred velocity $\tilde{\mathbf{v}}_i$, depending on the crowd density $\rho(\mathbf{x}_i)$ at its location, is calculated:

$$\mathbf{v}_i = \tilde{\mathbf{v}}_i + \frac{\rho(\mathbf{x}_i)}{\rho_{max}}(\mathbf{v}(\mathbf{x}_i) - \tilde{\mathbf{v}}_i). \tag{5.21}$$

The UIC projection only enforces separation between agents on a macroscopic scale. An additional step is performed to enforce minimum distances for each pair of individual agents. This is done by simply moving agents apart if they are too close to each other. This procedure is not guaranteed to separate all pairs to the preferred distance, but it gives good results because agent overcrowding is already prevented by the UIC.

Section 6.3 describes how the aggregate dynamics method has been adjusted in the Directional Waves method to work with formations.

6 Methods

This chapter describes the different methods to model formations in dense crowds I have compared. The first section describes the two different ways of modelling formations, the second section describes how I have used proxy agents and the third section shows my adjustment of the aggregate dynamics method to create the directional waves method.

6.1 Formations

Two different ways of modeling formations were used. The next sections will go into detail on how these were implemented.

6.1.1 Relative positions

As a starting point, a basic implementation of a formation was used. This is similar to the method proposed by Balch and Hybinette [3], briefly described in Section 2.3. The formation is modeled using *relative positions*. This type of formation will now be called an *RP-formation*.

The formation consists of a set of agents. One agent near the middle of the formation is appointed the leaderships position. This leader pursues its own goals just like a regular agent. The leader chooses its path with a large amount of clearance, in order to leave enough space for the other agents to keep their positions at the side of the leader. This is especially important when moving around corners.

All other formation members are assigned a location relative to the leader. This relative position consists of a distance d from the leader and an angle α with the leader's direction (as illustrated in Figure 6.1). Followers plan paths with that relative location as their goal. Each timestep they look at the leaders velocity, and calculate which position p he will be at in time t from now. They then calculate a path to their current goal location relative to p. Followers move at a slightly higher maximum velocity than the leader, to allow them to catch up if they fall behind for any reason. Algorithm 1 describes the process in pseudocode.

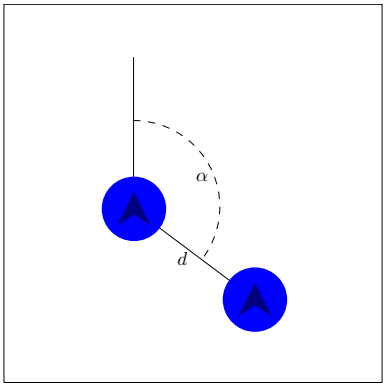


Figure 6.1: Followers take up the position at distance d and angle α of their leader.

Algorithm 1 Follower agents taking up relative positions

Input: Relative positions α and d, time t, leader's velocity v
Output: A moving formation
  Create leader character l;
  Create follower characters f;
  Assign relative positions (α and d) to the followers;
  while simulation runs **do**
    for each timestep s **do**
      Leader takes one step along his path;
      for all Followers f **do**
        Calculate p = (leader.position + $v * t$);
        Calculate path to location at α and d from p;
        Take one step along this path;

Symbol	Value	Explanation
-	9	Number of formation members
-	1.0	Distance between formation members
t	0.3	Followers strive for a location relative to the leaders position at time t from now
v	1.6	The formation's velocity

Table 6.1: Chosen values for formation-method 'relative positions'

This way of modeling formations is similar to how ME-members take up formations (as described in Appendix A). The two people at the ends of the line take up position, and the other agents take up positions between these, relative to the outer agents' positions.

Most variables values were set by trial and error. I will now explain some of my choices.

In my implementation, each formation consists of 9 members. While this implementation would work with even numbers of agents, uneven numbers are easier to handle. Nine members works well because it is as close as possible to the 10 members that an ME formation would consist of, while using an uneven number. The space between agents is set at 1.0. This was arbitrarily chosen as a distance that looks visually pleasing. Time t was chosen to be 0.3 seconds. At lower times (and therefore smaller distances to their goal), followers fall behind as they cannot see far enough into the future to keep up. At larger times (and therefore larger distances to their goal), followers run ahead of their leader as they walk slightly faster. A formations velocity is limited by the leader's velocity v. In my implementation, this was set to 1.6. This is slightly higher than the velocity of a regular character (1.4) to represent a steady pace. Table 6.1 shows an overview of the chosen values.

6.1.2 Formation as a dynamic object

An alternative implementation of formations was also used. To ensure the formation is never breached, it is modeled as a single object. As a complete formation does not behave in the same way as one person (and has an entirely different shape), it is no longer modeled as a set of characters or even as one large character. The formation is represented by an impassable obstacle, named the *formation obstacle*. The shape of the formation obstacle is dependent on the shape of the formation. When the formation is shown as a line, a rectangular obstacle is used; when the formation is shown in a v-shape, a triangle is used. This type of formation will now be called a *dynamic object (DO) formation*.

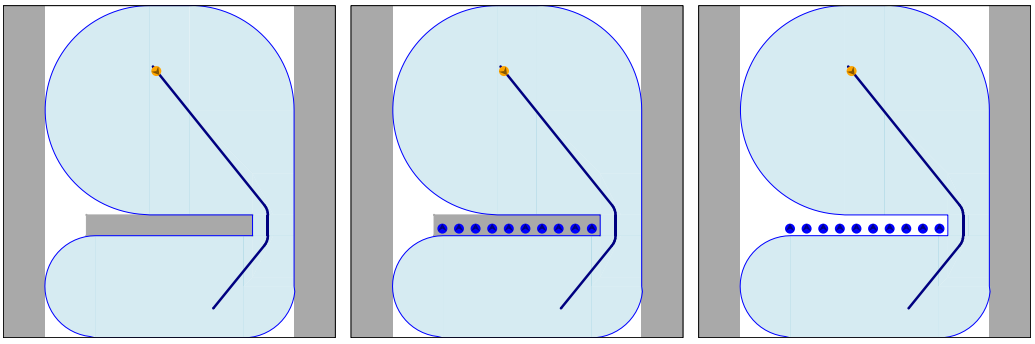
The obstacle moves through the location at a steady pace. In this implementation, the obstacles path is hard-coded. It cannot be adjusted at run-time and is dependent on the chosen scenario. When choosing the path, it is important to ensure no collisions with other obstacles occur.

Every t seconds, the formation obstacle moves forward. The explicit corridor map (see Section 5.1 for a more extensive explanation of the ECM) is then recalculated, and non-formation characters adjust their path accordingly. Algorithm 2 describes this process in pseudocode. The formation obstacle's location is not adjusted at every update, to avoid having to recalculate the ECM in every step.

Algorithm 2 Formation modelled as obstacle

Input: time between formation obstacle updates t, formation velocity v, time of last update u
Output: a moving formation
Create a non-visualised obstacle f with a shape based on the formations shape;
Create formation characters c, but do not add them to the list of regular characters;
while simulation runs **do**
  for each timestep s **do**
    time lapsed Δ = current time $-u$
    if $\Delta > t$ **then**
      Move obstacle $\Delta * v$;
      Set u = current time;
    Δ = current time $-u$;
    for all Followers m **do**
      Move m $(\Delta * v)$;

In order to visualize the simulation more realistically to the end user, instead of the rectangular or triangular object, characters are shown. The object that is used for path-planning is not rendered. A set of characters is added to the simulation that adjusts their position to travel over the same path as the obstacle. These characters are rendered, but are not taken into account by the other characters in the simulation. Figure 6.2 illustrates what the simulation takes into account, what is actually there, and what the user sees.



(a) Obstacle **(b)** Obstacle and Characters **(c)** Characters

Figure 6.2: (a) The simulation as used for path planning; (b) Both the obstacle and the characters are shown; (c) The end user only sees the characters.

In this implementation, the time t until the formation obstacle moves, was set at 0.25 seconds. At $t = 0.3$, the formation looks to be moving smoothly, without having to recalculate the ECM at each timestep. Although this formation works just as well with an even number of characters as with an odd number of characters, I have chosen to set this at the same number (9) as the formation as relative positions, in order to compare the two different methods in a setting as similar as possible. The size of the obstacle is a quite tight fit around the characters. As the distance between formation characters was chosen at 1.0, this value was taken as the required clearance for a formation member. A rectangular obstacle of size 9 x 1 and a triangular obstacle has sides of lengths $\sqrt{30}$, $\sqrt{30}$ and $2 * \sqrt{15}$. These sizes cover the formation without being too much bigger. The formation's velocity was set to 1.6 to resemble a steady pace. Table 6.2 gives an overview of these values.

Symbol	Value	Explanation
-	9	Number of formation members
-	1.0	Distance between formation members
t	0.3	Seconds between formation obstacle moves
v	1.6	The formation's velocity
-	9 x 1	Size of rectangular formation obstacle
-	$\sqrt{30}$ x $\sqrt{30}$ x $2 * \sqrt{15}$	Size of triangular formation obstacle

Table 6.2: Chosen values for formation-method 'dynamic object'

6.2 Proxy Agents

I have applied the Composite Agents method [67] to achieve empty space in front of formations. The 'aggression' method described in Section 5.3 was applied to show the formation's sense of urgency; each formation member has one large proxy right ahead of it. For a formation of 9 agents, this would mean 9 proxy agents. Figure 6.3 illustrates this.

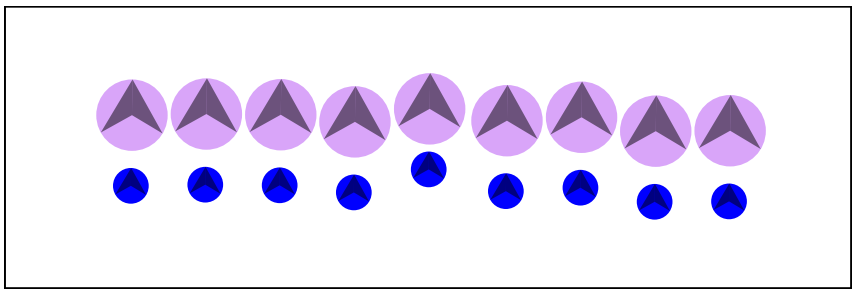


Figure 6.3: Formation agents (in blue), each having a proxy agent in front of it (in purple).

Each timestep, formation members m first move as described in 6.1.1. Then, each formation member sets its own proxies to the location where m will be in time t from now, if m does not change its velocity. Algorithm 3 details the steps.

Algorithm 3 Formation agents with proxies

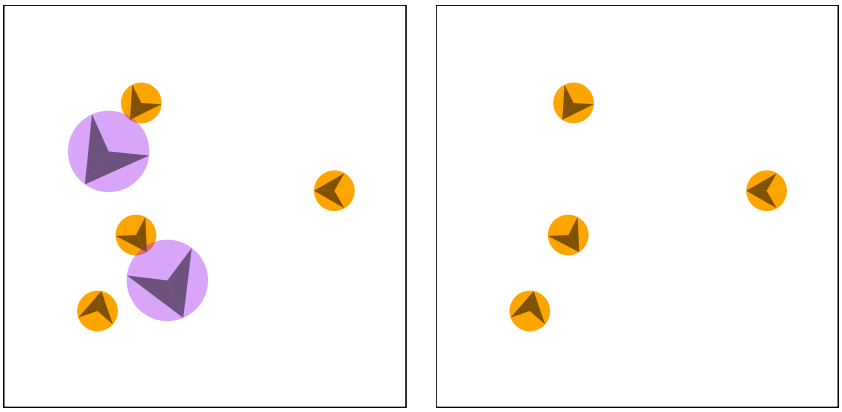
```
Output: A moving formation with proxy agents
  Create leader character;
  Create follower characters;
  Create a proxy agent for each formation member;
  while simulation runs do
    for each timestep s do
      for all formation members m do
        Move m along its path;
        for all linked proxy agents p do
          Move p to (m.position + m.velocity * time t);
          Calculate and store p's velocity.
```

In my implementation, t is set to 0.5. The distance resulting from this time keeps the proxies close enough to minimize crowd agents slipping between the formation agent and its proxy, while also far enough away to keep the crowd members at a realistic distance. The proxy's radius is set to 2.1 times the radius of a regular character. As in this implementation regular characters have a radius of 0.24, this sets the radius of proxy agents to 0.504. This radius was chosen because at the chosen distance between formation members, space between proxies is minimized without too much overlap. Table 6.3 repeats these values.

Symbol	Value	Explanation
t	0.5	Proxies are located where its leading agent will be at time t from now
-	0.504	Size of a proxy's radius

Table 6.3: Chosen values for formation-method 'relative positions'

As stated in Section 5.3, the end user does not see the proxy agents. Figure 6.4 illustrates the difference between the situation as perceived by the characters in the simulation, and as perceived by the user.



(a) Character perception **(b)** Viewer perception

Figure 6.4: (a) Characters will avoid collisions with proxy agents; (b) End users do not see the proxy agents.

6.3 Directional Waves

A second method to force the crowd to make space for the formation, is using *Directional Waves* (DW). Directional Waves make use of the velocity field that was calculated to apply aggregate dynamics by Narain et al. [34] as described in Section 5.4. This section describes how I have adjusted the velocity field to suit formations. The directional waves method only works with formations that are modeled as dynamic objects.

In order to have the formation influence the crowd agents through this grid, an extra step is introduced between steps 3 and 4 as listed in Section 5.4.1. The formation sends out directional waves to warn crowd agents it is coming. Figure 6.5 shows how this field of directional waves looks.

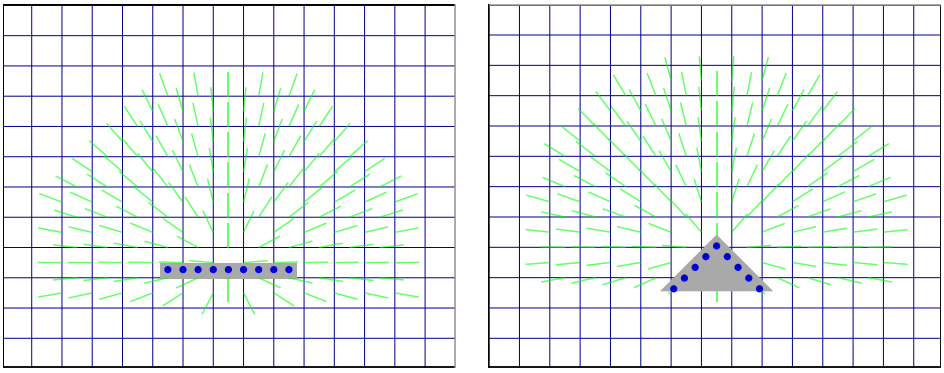
An environment with agents A exists. Each simulation loop consists of the following steps.

1. Each agent i plans its path and calculates its own preferred velocity $\tilde{\mathbf{v}}_i$. This can be done using any global planner.

2. A grid is laid out over the area. Each grid edge stores the average velocity $\bar{\mathbf{v}}$ of the agents around it. Each grid cell stores the agent density ρ in it.

3. Adjust the velocity field to accommodate formations.

4. The stored average velocities are modified so that in grid cells where the density ρ would exceed the maximum density ρ_{max}, no extra agents enter the region.



(a) Line-shaped formation **(b)** V-shaped formation

Figure 6.5: The directional waves send out by (a) a line-shaped formation, and (b) a V-shaped formation.

5. Agents calculate their final velocity based on a combination of their own preferred velocity and the velocity field **v**. In denser regions, the velocity field is weighted higher.

I will now explain how the velocity field is adjusted to accommodate formations. Algorithm 4 describes the process in detail.

Algorithm 4 Directional Waves

Input: the velocity the formation influences the grid by v_f, the weight this velocity is applied by w_f, the distance within which the formation influences the grid d_f, the location of the formation l_f, maximum cell density ρ_{max}.

Output: a staggered grid with average velocities on edges and densities within cells, adjusted to accommodate a formation.

for all grid cells c **do**
  for all grid's cell edge points e **do**
    $\tilde{\mathbf{v}}_e$ = the sum of the weighted velocities of characters close to e.
    w_e = the sum of the weights.
    if e lies in the half-plane that is in the formation's direction **then**
      if e lies within d_f from the formation **then**
        direction $\vec{d} = l_e - l_f$.
        inverted distance $x = d_f - |\vec{d}|$.
        $\tilde{\mathbf{v}}_e = \tilde{\mathbf{v}}_e + \vec{d} * x * v_f * w_f$.
        $w_e = w_e + w_f * x$.
    $\tilde{\mathbf{v}}_e = \tilde{\mathbf{v}}_e / w_e$.
  if c lies within d_f from the formation **then**
    inverted distance $x = |l_c - l_f|$.
    $\rho_c = \min(\rho_c * a * \max(x^2 * \rho_{max}/d_f^2, 1.0), \rho_{max})$.

Each timestep, the program looks at all cells to apply steps 2 and 3. For each cell edge e, $\tilde{\mathbf{v}}_e$ stores the sum of the weighted velocities of the agents that are close (where closer agents are weighted higher). The total weights are stored in w_e. If the cell is within influencing distance d_f of the formation, an additional velocity gets added to $\tilde{\mathbf{v}}_e$. This velocity consists of several parts.

$$\tilde{\mathbf{v}}_e = \tilde{\mathbf{v}}_e + \vec{d} * x * v_f * w_f. \tag{6.1}$$

where

Symbol	Value	Explanation
v_f	1.6	The velocity the formation adjusts $\tilde{\mathbf{v}}_e$ by
w_f	3.0	The weight the formation adjusts $\tilde{\mathbf{v}}_e$ by
d_f	12.0	The distance over which the formation has influence
a	3.0	The factor by which the density is adjusted
ρ_max	7.0	The maximum amount of people in an area of 1 x 1
-	2.0	The size of a cell edge

Table 6.4: Chosen values for formation-method 'relative positions'

$$\vec{d} = \text{the direction from the formation to the middle of the cell edge,} \tag{6.2}$$
$$x = d_f - \text{the distance between } e \text{ and the formation,} \tag{6.3}$$
$$v_f = \text{the velocity the formation influences it with,} \tag{6.4}$$
$$w_f = \text{the weight the formation influences it with.} \tag{6.5}$$

To make sure closer locations are influenced heavier, x gets introduced. Of course, w_e also gets adjusted so that $w_e = w_f * x$. When all character's velocities and the formation's influence have been summed, $\tilde{\mathbf{v}}_e = \tilde{\mathbf{v}}_e / w_e$ to obtain an average velocity.

In the aggregate dynamics method by Narain et al., characters in denser regions take the averaged velocities into account more. To be able to give the formation a heavier influence, the density is also adjusted in cells within influencing distance from the formation.

$$\rho_c = \min(\rho_c * a * \max(x^2 * \rho_{max}/d_f^2, 1.0), \rho_{max}). \tag{6.6}$$

Again, this is scaled so closer characters are influenced heavier than characters further away. In this case, it's scaled exponentially; closer characters are influenced much heavier. The new density is a factor of the old density, because in high density situations, the formation should have a heavier influence on the crowd. In a crowd, it is important for characters further away to already start moving away to make space for the characters closer by.

The density is never adjusted to smaller than it was, which is why ρ_c is always scaled by a factor of at least 1.0. The density can never go over ρ_{max} either. Factor a is introduced in order to represent how much the crowd is influenced by the formation, and f_w has the same goal. A scared formation would flee quicker, and would require higher values of a and f_w than an angry formation.

This method has many different variables that can be set. Table 6.4 shows them all. v_f was set to 1.6. This was set at that value because it is the same velocity at which a formation is travelling. If a character would only be influenced by the formation, it should travel at this speed when it is directly in front of the formation. The factors w_f and a have been set at values that seem to achieve good results.

The maximum density (people per square meter) was set to 7.0. According to G.K. Still [51], five people per square meter is the maximum safe density. At six or seven people, the risk of shockwaves occurs. As it is important to be able to model these dangerous situations, we allow for such a high density in this simulation. The density could easily be set to a lower number, to model different scenarios.

The next section will go into detail how well the methods proposed in this chapter perform.

7 Experiments & Testing

7.1 Plan

The proposed methods from Chapter 6 have been extensively tested. Emphasis is on the emerging behaviour with the main question being 'Does this look realistic?'. This section describes the different situations I have looked at.

7.1.1 Comparison

I will be comparing six different situations to each other. I will be comparing two types of modeling a formation in combination with three different types of helping the formation influence the crowd.

Formations

- Formation as relative positions (RP-formation),
- Formation as dynamic object (DO-formation).

Additional measures

- No additional measures to avoid collisions with or breaching of the formation were taken.
- Proxy agents are used to create space for the formation members.
- Directional waves are used to create space for the formation members.

Some additional variables will be introduced to better judge the methods. These will be varied to illustrate distinctive behaviour. Probably the most important parameter I will be varying is the crowd density. I will test in low, medium and high density environments.

I will be using three different test environments. The formation will be tested in two different shapes.

Test environments

- An empty location with no obstacles;
- A street with the same width as the formatio;,
- A street that is slightly wider than the formation.

Formation shapes

- Line-shaped;
- V-shaped.

My implementation uses of the Explicit Corridor Map to map the environment, the Indicative Route Method for local navigation, and a vision-based approach by Moussaïd et al. for collision avoidance. Chapter 5 gives a description of these. As my methods work independently from these, they could all be replaced by different methods. A number of variables will not vary during testing. An overview of the relevant variables and their values can be found in Tables 6.1, 6.2, 6.3 and 6.4. Sections 6.1.1, 6.1.2, 6.2 and 6.3 explain how these values were chosen.

7.1.2 System

All experiments will be run on a single machine with an AMD Phenom II X4 955 Processor at 3.20 GHz with 4 Gb of RAM and a Sapphire HD5750 GPU. The operating system on this machine was Windows 7 64-bit. Visual Studio 2010 Professional was used as the development environment.

The next sections will show and explain the results. The first sections will assess the emerging behaviour, and the last section talks about the efficiency. In all images, formation members are shown in blue, proxies in purple, and regular agents in orange.

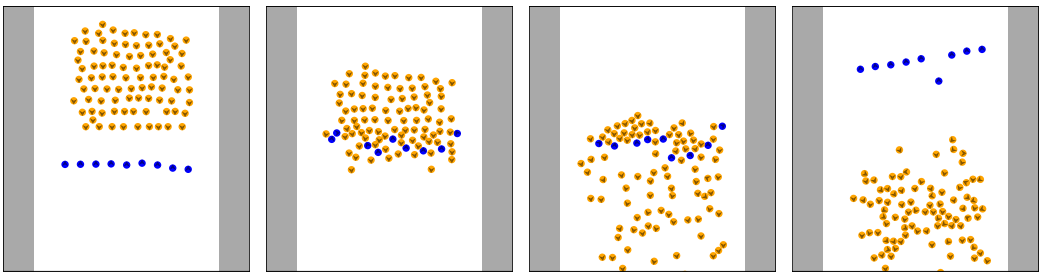
7.2 Establishing a baseline

To establish a baseline, I have looked at the two different ways formations are modeled. No additional measures were taken to help the formation obtain room to move or to keep the crowd away from the formation. I will now describe what behaviour occurs in this situation.

7.2.1 Formation as relative positions

When modeling the formation as relative positions, the behaviour that occurs is very much as expected. In a low density situation, regular agents pass through the formation without any hesitation. There is enough space to walk between neighbouring formation characters without collisions, and no reason not to.

In medium- and very high-density situations, the regular agents still pass through. However, because there are so many oncoming pedestrians, the formation does not have the ability to move forward any longer. It stays in a more-or-less stable position in the crowd, much like a rock in a stream of water. Figure 7.1 shows what happens when a formation meets a crowd of medium density. In a high density situation or with a V-shaped formation, similar behaviour occurs.

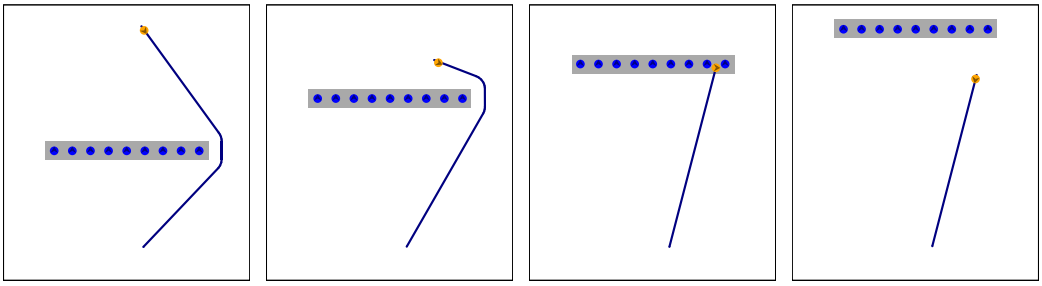


(a) Approaching **(b)** Stable position **(c)** Stable position **(d)** Final Situation

Figure 7.1: When an RP-formation runs into a crowd with no additional measures taken, it stays in a more-or-less stable position in the crowd, much like a tree in the wind.

7.2.2 Formation as dynamic object

The formation modeled as dynamic object does not work well at all without additional measures. When a single character is in an environment with a formation, it will immediately start (re)planning its route while taking the formation into account. The character will take



(a) Approaching **(b)** Rerouting **(c)** Meeting **(d)** Final Situation

Figure 7.2: (a) A character approaches a line-formation; (b) The character re-routes to avoid the obstacle; (c) The character can not get out of the way fast enough and freezes when overlapping the formation; (d) The formation has passed and the character continues on its way.

a route that does not collide with the formation in its current position. However, it does not take into account that the formation is moving. As the formation keeps approaching, the character has to keep adjusting its path. The character tries to stay as close to the corner as possible to plan a route as short as possible. When it takes longer for the character to reach the corner of the object than it takes for the formation to reach the character, the character is not able to get out of the formation's way in time. As the distance the formation has to travel will always be smaller than the distance traveled by the character (see Figure 7.3), at equal speeds the formation will always reach the character first. In this implementation, the formation travels even faster than the crowd characters.

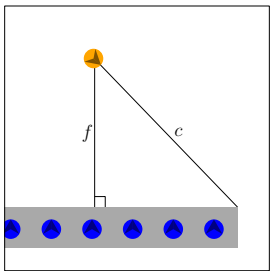


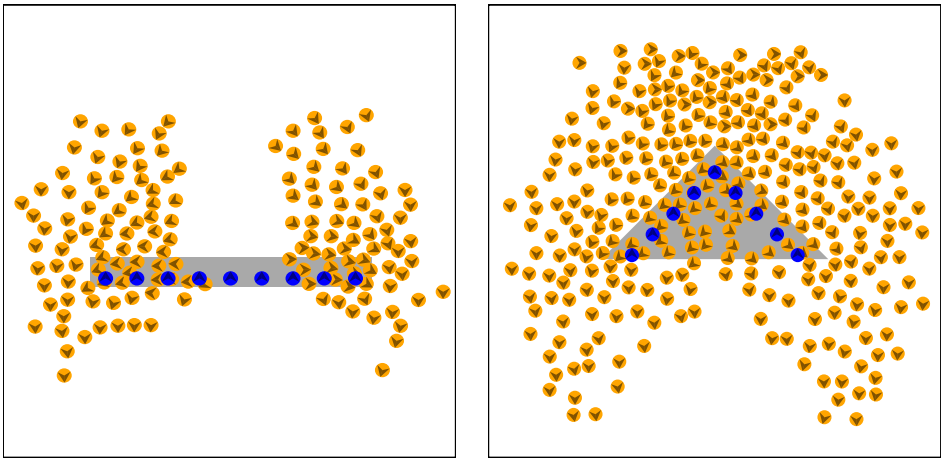
Figure 7.3: The distanced traversed by the formation to reach the character (f) will always be smaller than the distance traveled by the character (c) to reach the corner of the formation.

When the formation and the character meet, the formation continues along its path. If that path leads over the character, the formation walks over the character. The character is now stuck in the formation obstacle, and unable to find a traversable route. The character freezes until the formation has past. When the formation has past, the character once again reroutes its path and continues along its way. Figure 7.2 shows a character's interaction with a formation as dynamic object. In medium and high density crowds, the same effect occurs on a larger scale, as can be seen in 7.4.

As the end user does not see the object that models the formation, this looks like a breach in the formation at best (when the character stops between two formation agents) and an inexplicable freeze and collision at worst (when the character intersects a formation agent).

7.3 Proxy Agents

In situations where this method was used, each formation member has a proxy agent. These agents cannot be seen by the user, but other agents keep these into account in the collision avoidance phase. The desired result is that agents leave space in front of the formation, so that the formation is free to go where it likes, moving the crowd agents back as it approaches.



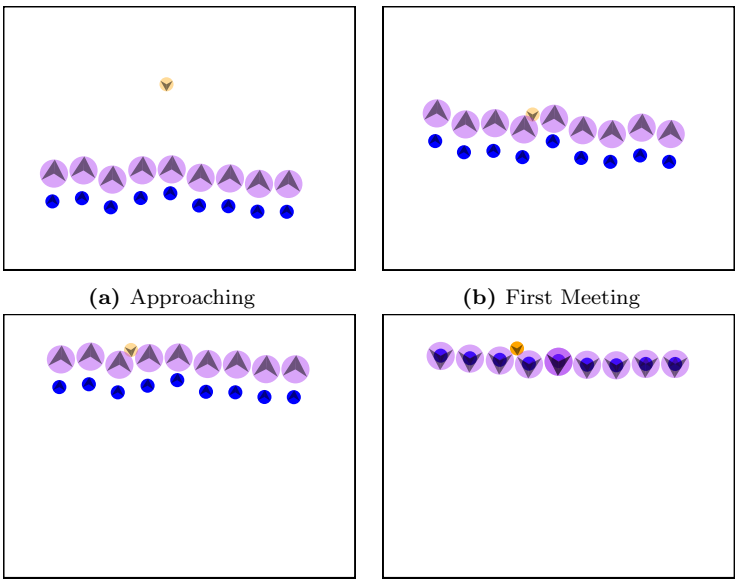
(a) Medium Density **(b)** High Density

Figure 7.4: In a dense formation, characters also slip through and collide with the formation.

Crowd agents should not be able to traverse the space between formation agents or collide with formation agents.

Although there is some success, the method has some problems too. I will now describe how the use of proxy agents influences the simulation and show some illustrating images.

7.3.1 Formation as relative positions



(a) Approaching **(b)** First Meeting

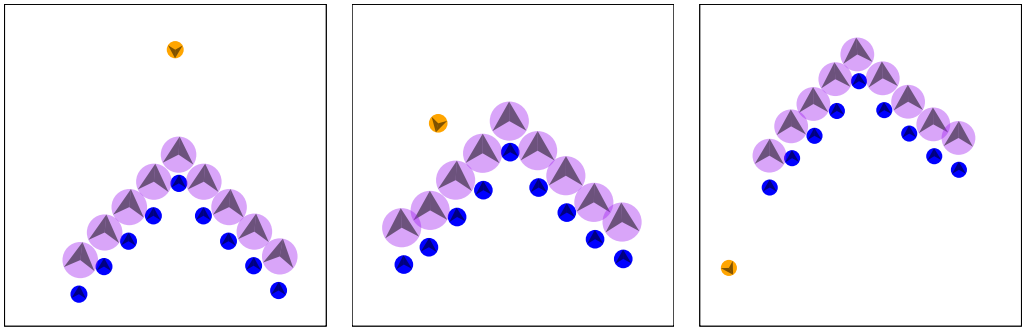
(c) Dragging Along **(d)** Final Situation

Figure 7.5: (a) A character approaches a line-shaped RP-formation with proxy agents; (b) When they first meet, the character slightly overlaps the proxy agents; (c) The character bounces back and is dragged along with the formation; (d) The formation stops walking and the character stays where it is.

In low-density environments, when combining the proxy agent method with formations modeled using relative positions, the results are alright. When a lone character approaches a line formation with proxies at its front head on, the character is stopped by the proxy agents. The formation keeps going (as the formation does not see the proxies, they experience the space in front of them as free), and drags the single character along. When the formation stops, a balance situation occurs. The character seems to give up. This process is illustrated

in Figure 7.5. Ideally the character would re-route, but the character stays at a distance and does not breach the formation.

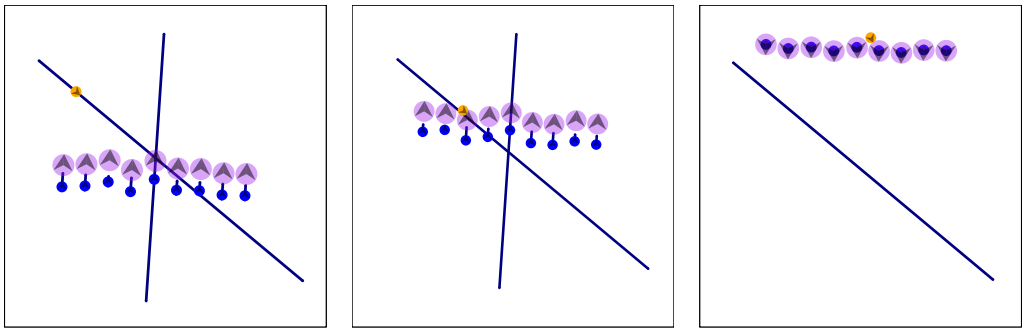
When the formation is V-shaped, the character has an easier time continuing on its route. A collision will usually give it a chance to move to the side of the formation, where it has some extra room to walk. Eventually the character bounces off the triangle and continues on its way. Figure 7.6 shows a meeting between a single character and a V-shaped formation.



(a) Approaching **(b)** While Passing **(c)** Past

Figure 7.6: (a) A character approaches a V-shaped RP-formation with proxy agents; (b) The character moves to the side; (c) The character continues to move to the side until it can pass the formation.

Figure 7.7 shows what happens if the character approaches the line-shaped formation at an angle. The character has a slightly easier time to continue on its route. It will continue travelling along the formation for a while, but as the formation moves on, the angle between the character's route and the formation gets more perpendicular, and approaches the situation shown in Figure 7.5.



(a) Approaching **(b)** First Meeting **(c)** Final Situation

Figure 7.7: (a) A character approaches an RP-formation with proxy agents; (b) As the character's velocity was at an angle to the formation's velocity, it uses its sideways velocity to move aside for the proxy agents; (c) When the character can not pass the formation before the formation moves too far, the angle between the character's intended direction and the formation gets too small and the character is dragged along by the formation.

As we have seen, in situations with a single character the method does what is expected. However, in larger crowds, this method does not work nearly as well. I will now describe some problems that occur.

If there is any space between two proxies, agents will use it. Crowd members keep slipping through the small spaces between two (proxy) agents, even if their radius is too large for this space. Very large proxies help avoid this problem, but this means the distance in front of the formation members that is taken up by proxies is larger too. Figure 7.8 shows an example.

The formation does not stay rigid at all times. When the formation runs into a dense crowd, it loses its shape. This is illustrated in Figure 7.9. The same effect occurs when the formation is V-shaped. It can be explained by two different factors.

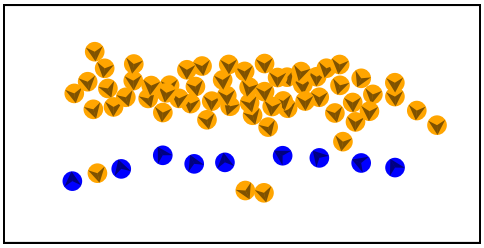


Figure 7.8: Agents slip between formation members.

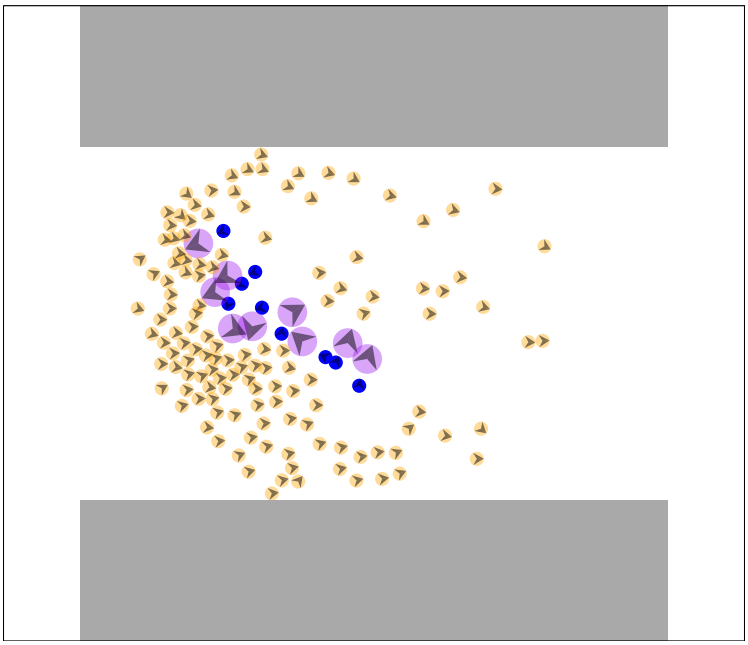


Figure 7.9: When confronted with a crowd, the RP-formation loses its shape. The formation is moving from right to left.

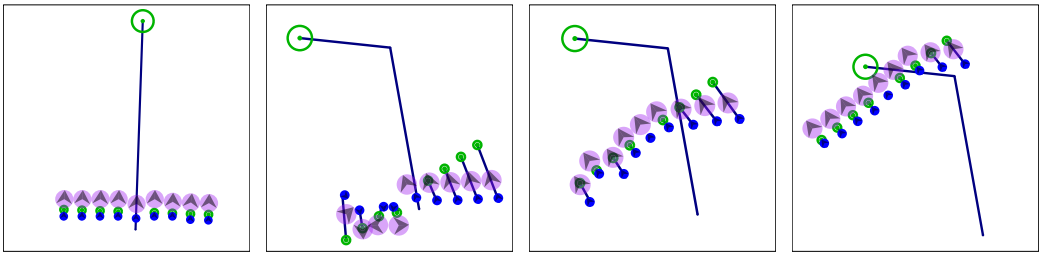
First, the formation members avoid running into the crowd. In order to do so, they sometimes move away from their designated position. This means that the formation does not stay rigid when it reaches the crowd. As the formation members fail to keep their position, crowd members see a chance to slip through and make things worse. If the formation were to stop trying to avoid the crowd, they would intersect the crowd members, which is not desired either.

Second, as the formation leader adjusts its direction to avoid crowd members (by moving slightly left or right) the formation's direction is changed and followers strive for different positions. This is illustrated in Figure 7.10. As this means that followers on one side have to reach a position behind them, the proxy is moved to the other side, leaving room for the crowd members to approach. When the leader adjusts back to its original direction, there is no longer room for the formation members to walk.

Overall, it is clear that a formation modeled using relative positions, does not obtain realistic results using proxy agents to influence the crowd.

7.3.2 Formation as dynamic object

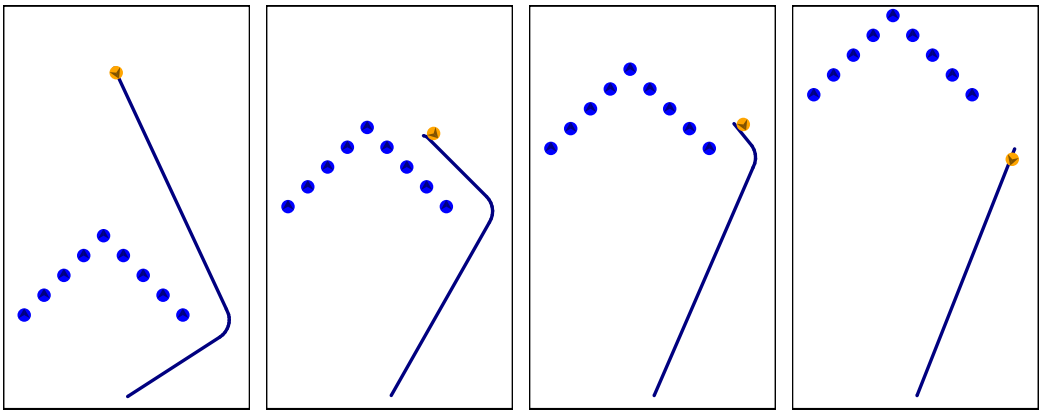
Formations modeled as a dynamic object show better results with proxy agents than formations modeled using relative positions. In low density environments, the results are quite similar. As we could see in the situation with no measures taken, crowd agents already plan their path around the formation. Without extra help though, the characters are not able to get out of the way of the formation and freeze while the formation intersects their location.



(a) Walking ahead **(b)** Turning **(c)** Turning **(d)** Turned

Figure 7.10: A formation leader changes its direction. The green circles indicate the goal positions.

With proxy agents in front of the formation, the character is able to plan ahead and avoid the formation. Figure 7.11 shows a head on collision between a triangular formation and a single character. As you can see, the character moves aside to avoid the formation, and is able to continue along its way.



(a) Walking ahead **(b)** Adjusting route **(c)** Moving aside **(d)** Passed

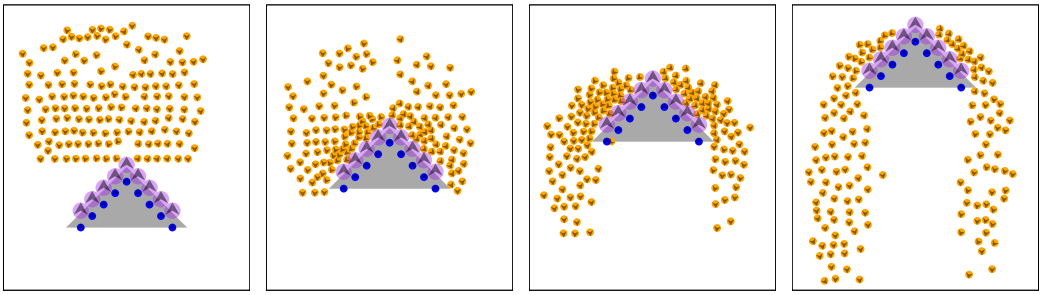
Figure 7.11: A single character runs into a DO-formation that is helped by proxy agents. The formation object and proxy agents are not shown.

When a single character approaches a line-shaped formation, behaviour similar to the same situation with an RP-formation occurs. The character does not collide with the object, but is unable to move aside fast enough and gets dragged along.

Unlike in simulations with RP-formations, the triangular formation object with proxy agents still works well in medium density environments, as illustrated in Figure 7.12. The proxy agents steer the crowd members to the sides. Occasionally an intersection between a crowd member and the formation still occurs, as in Figure 7.12c (an intersection occurs in the left side of the formation object). In high density environments, slightly more intersections occur, but the overall results are still good (Figure 7.13a). In line-shaped formations (Figures 7.13b and 7.13c), the number of intersections goes up again, but it is a great improvement over a scenario with no additional measures. As not all characters can move aside in time, some characters get dragged along. This is the same effect that occurs when an RP-formation with proxy agents meets a line-shaped formation.

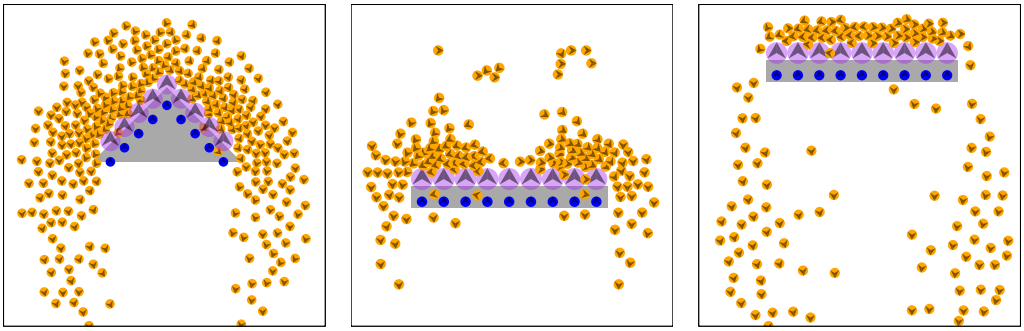
A formation modelled as a dynamic object can achieve acceptable results using proxy agents to influence the crowd. Especially when the formation is V-shaped, it almost meets the requirements.

During testing, it was obvious to see that using DO-formations is far from fast enough in its current implementation. Low density environments run smoothly, but in medium to high density environments, replanning all routes takes too much time and looks shaky. This method requires some heavy optimization before it can be used in real-world applications. It would be a good idea to find a way to only adjust the path locally.



(a) Approaching **(b)** Passing **(c)** Passing **(d)** Passing

Figure 7.12: A medium density crowd runs into a V-shaped DO-formation that is helped by proxy agents. In (c) one character gets overrun by the formation.



(a) V-shape in high density **(b)** Line-shape in med. density **(c)** Line-shape in med. density

Figure 7.13: (a) A V-shaped DO-formation carves through a high density crowd; (b) and (c) A line-shaped DO-formation meets a medium density crowd.

7.4 Directional Waves

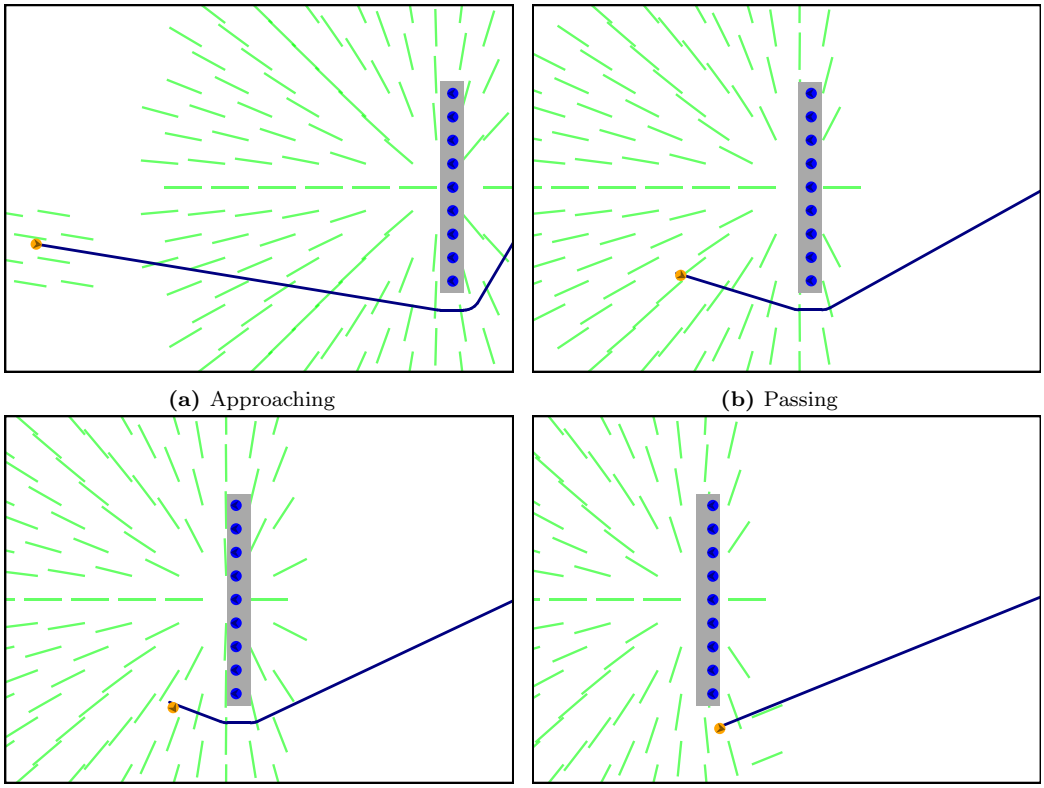
In the directional waves method, the formation uses a grid to steer away characters before they have even reached the formation. Again, the desired result is a situation where agents leave space in front of the formation, so the formation is free to go where it likes, moving the crowd agents back as it approaches. Crowd agents should not be able to traverse the space between formation agents or collide with formation agents. This section shows how well this method works with DO-formations. The method has not been applied to RP-formations.

In a situation where a single character starts far ahead from a formation, the method works well. This is illustrated in Figure 7.14. The directional waves steer the character away from the formation. This gives the character an extra push to the side, and pushes the character away from the corner. Both line-shaped and V-shaped formations work well.

In a medium- or high-density situation, it does not work as well. When a V-shaped DO-formation meets a medium-density crowd, the crowd starts to move aside. When agents get in range, they start adjusting their velocity to flee from the formation. Most agents are able to move aside from the formation and walk around it, but the agents directly in front of the formation don't get a chance to move aside. The formation mainly steers these in the same direction as it is moving, but the agents don't walk fast enough. When the formation catches up with these characters, they get overrun. Figure 7.15 illustrates this.

A similar situation occurs when a line-shaped DO-formation interacts with a medium density crowd, as shown in Figure 7.16.

Directional waves solve the biggest problem that occurred in situations with proxy agents (characters getting dragged along by line formations), but they introduce a new problem: in dense situations, some characters get overrun by the formation.



(a) Approaching **(b)** Passing

(c) Passing **(d)** Passed

Figure 7.14: A single character runs into a line-shaped DO-formation that is being helped by directional waves. (a) The character is approaching and is not influenced by the formation yet; (b) the character is bent slightly away from its intended route by the directional waves; (c) the character clearly bends away from its intended route; (d) the character has passed the formation and continues on its way.

7.5 Combining Directional Waves and Proxy Agents

Proxy agents and directional waves both solve some problems that occurred in the baseline situation where a formation approaches a crowd with no additional measures taken. Both methods have an explicit problem though:

- **Proxy Agents** don't leave characters a chance to escape a formation and drag them along as long as the formation is traveling.
- **Directional waves** don't influence the characters in the very dense areas enough to keep them from being overrun.

Because each method had a different problems that seemed to be solved by the other method, the methods were combined to be able to solve both problems. In this section, we will see what happens when Directional Waves are combined with Proxy Agents. As directional waves have only been applied to DO-formations, the combination can also only be applied to DO-formations.

Combining the two methods works very well. In single character situations, the V-shaped formation already worked well. The line-shaped formations had some problems though. When using proxy agents, a character approaching a formation head on would get dragged along. When using directional waves, the character would occasionally intersect the side of the object, and momentarily freeze. The combined methods solve both problems. The proxy agents keep the formation from colliding with any agents. The directional waves steer the character aside, so it moves around the object instead of getting caught.

Figure 7.17 shows a medium density situation with a V-shaped formation. Again, this works very well. Using just directional waves, the proxy agents near the middle would get

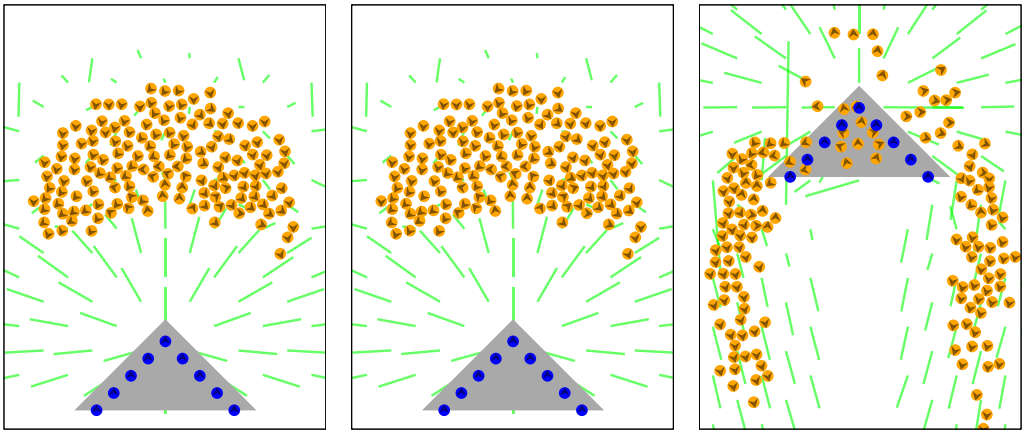


Figure 7.15: A V-shaped DO-formation meets a medium density crowd; (a) The crowd starts to move aside. The closer agents start adjusting their velocity to flee from the formation. (b) The agents directly in front of the formation don't get a chance to move aside as they are only steered ahead; (c) The agents that did not get a chance to flee get overrun.

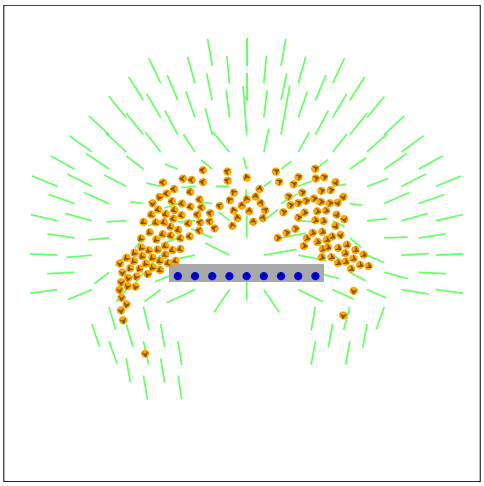


Figure 7.16: Agents slip between formation members.

overrun by the formation. In this situation, the proxy agents keep the characters out of the direct vicinity of the formation, while the directional waves force characters to start moving away at a further distance, giving the agents the chance to move to their locations.

The improvements are largest when looking at the line formation in medium density situations, as shown in Figure 7.18. No intersections occur (as with only directional waves), and the characters do not get dragged along by the formation (as with only proxy agents). The combination does exactly what we want it to do.

When crowds get larger, occasionally some intersections do occur, but the amount is greatly decreased from the method using directional waves without proxy agents. Figure 7.19 shows how the end user experiences a medium-density crowd running into a line-shaped formation. Overall, the combined methods work quite well.

7.6 Efficiency

This section lists the runtimes of some of my algorithms. As the runtimes of moving formations modeled as Relative Positions are only determined by the path-planning algorithms used, these have not been calculated.

Table 7.1 lists the average time spent moving a formation modeled as a Dynamic Object each timestep. The first column lists the average over a regular run. As the formation only

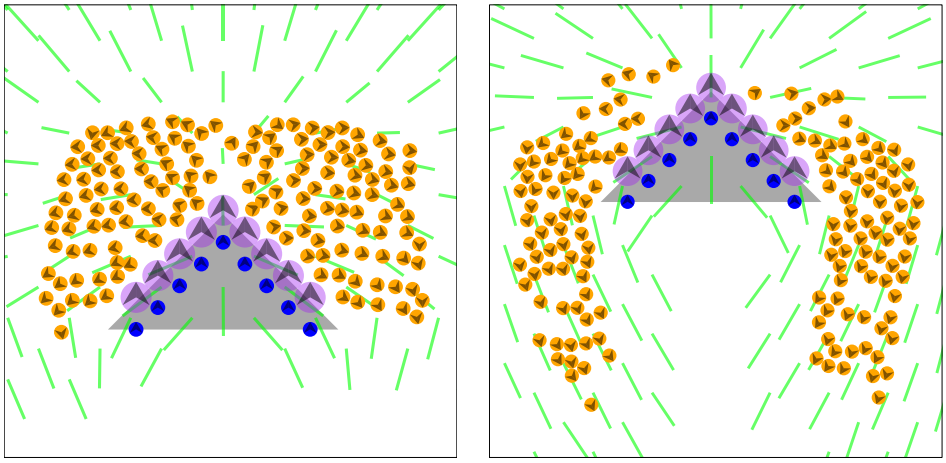


Figure 7.17: When combining the proxy agents method with directional waves, no intersections occur.

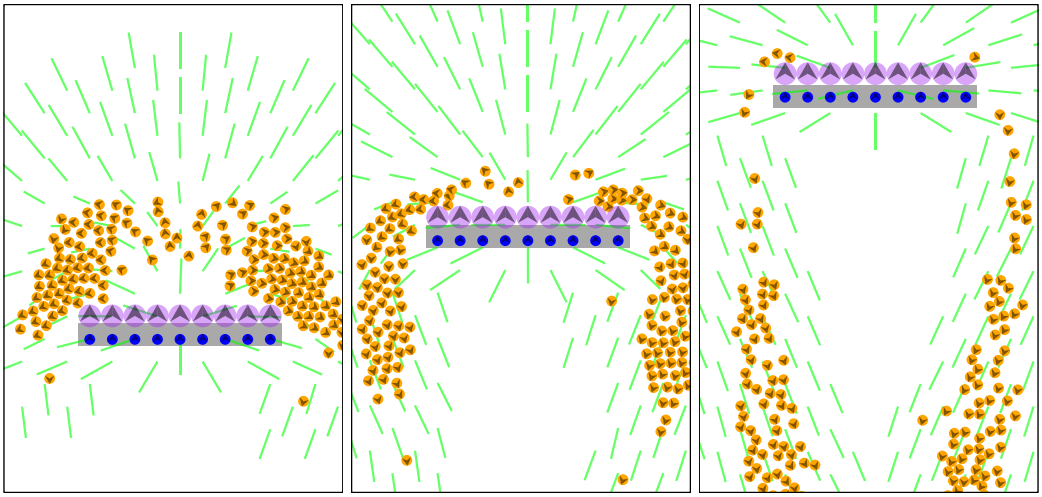


Figure 7.18: When combining the proxy agents method with directional waves at medium density, no intersections occur, and the characters do not get dragged along.

gets moved every 0.3 seconds and the characters get updated every 0.1 seconds, the total time each timestep varies. The second column lists only the time it takes to the move the formation obstacle. As there is only one DO-formation in the environment at any one time, these times are quite alright. Even in simulations that are updated more frequently, moving the formation obstacle only has to happen every few updates, and moving the characters is not expensive.

The tests were conducted in an empty environment of 20 x 20. The time listed is the average of moving the formation over 3500 times. This time includes updating the ECM. It does not include the time spent replanning the crowd agents' routes. Replanning is an important time sink, but as it is independent from my methods, I have not timed it.

Shape	Average Runtime (ms)	Formation obstacle Runtime (ms)
Line	13.9	38.8
V	13.2	37.6

Table 7.1: The time it takes to move a DO-formation. The first column lists the average of updating the characters each step and the formation obstacle only every 3 steps. The second column lists the time it takes to move the formation obstacle, without moving the characters.

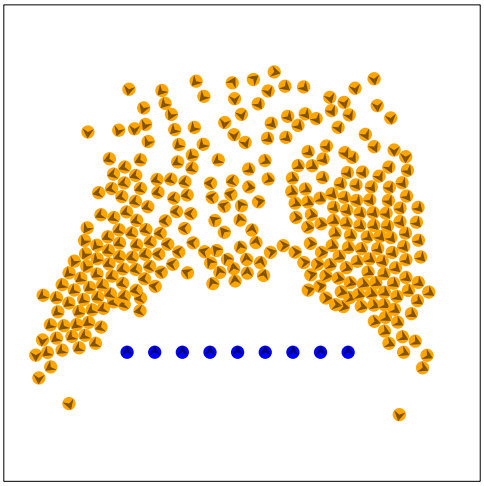


Figure 7.19: A line formation runs into a medium density crowd, as seen by an end user.

Size of environment	Average Runtime (ms)	No. of cells	Runtime per 1000 cells
20 x 20	0.371	100	3.71
30 x 30	0.830	225	3.69
100 x 100	8.89	2500	3.56

Table 7.2: The time it takes to move a DO-formation. The first column lists the average of updating the characters each step and the formation obstacle only every 3 steps. The second column lists the time it takes to move the formation obstacle, without moving the characters.

Moving a proxy agent takes an average of 0.0010 ms. This makes proxy agents an extremely inexpensive method.

The runtimes of the Directional Waves method were also measured. The results can be seen in 7.2. The test was run in an empty environment of several sizes with no crowd characters. The chosen cellsize was 2.0 x 2.0. As we can see, runtimes go up with a larger number of cells, but the time taken per cell stays relatively stable. Having other characters in the environment would increase the runtime non-trivially, as more velocities would need to be summed.

These times are low enough to be able to simulate even large environments efficiently.

8 Conclusion and Future Work

8.1 Conclusion

As serious gaming gets applied more and more in our society, algorithms dealing with related problems get more important as well. One such problem is how characters can keep up a formation while interacting with crowds of varying density in a realistic manner. The goal of this thesis project was to find a way to do this.

Two different methods of modeling formations (using Relative Positions (RP) and modeling the formation as a Dynamic Object (DO)) have been applied and compared. These algorithms do not allow for a formation to meet a crowd without being breached and/or deformed. Therefore I have designed two different methods to help steer the crowd away from the formation and solve some of the problems that occur.

Proxy Agents achieve great results in keeping the formation from being breached. By placing an invisible, large agent in front of formation agents, crowd characters apply collision avoidance at a larger distance from the formation and adjust their directions accordingly. However, both types of formations have some different problems.

When using an RP-formation with proxy agents in a dense crowd, the formation is unable to keep its position. The formation agents adjust their direction to avoid collisions and in doing so they allow crowd agents to pass. The formation is breached and deformed.

Using a DO-formation solves the problem of deforming formations. By modeling the formation as an object instead of a set of agents, it is not possible for the formation to take a different shape. However, when a crowd collides with a line-shaped formation, characters are unable to re-plan their routes and get dragged along by the formation.

As formations modeled using relative positions did not work nearly as well as the formation as a dynamic object, Dynamic Waves were only applied in combination with DO-formations. In this situation, characters no longer get dragged along by the formation. Instead, in the crowded locations at the side of the formation and in the middle of the formation where the agents do not get steered sideways as much, the formation overruns the characters. So once again, the formation is breached.

As both methods had different problems, they were combined in hopes of solving all problems. This resulted in good situations. Characters are no longer dragged along by the line-shaped formation as they were when using proxy agents. Neither does the formation steamroll over the crowd agents, although in very dense situations it still occurs. As currently no methods exist to have a formation interact with a crowd, this is a great improvement over the previous situation.

In conclusion, when modeling a formation as a dynamic object and steering the crowd away from it by using both proxy agents and dynamic waves some good results are acquired. However, the method is far from perfect and requires some extensive work before it is ready to be applied.

8.2 Future Work

8.2.1 Optimization

The first area that could greatly use work is optimization. When modeling the formation as a dynamic object, the simulation runs far from smooth. When testing a small environment with a low density in release mode (number of characters <30), the simulation runs nicely. However, when exceeding this number the simulation starts to halter. Re-planning the routes for all characters every n timesteps is just to heavy to handle. The current implementation is definitely not good enough to be applied in any program.

It is likely that using different general path-planning methods that are better suited to work with moving objects would increase efficiency, as the moving object is the greatest cause of slowness.

When modeling the formation using relative positions, the simulation runs smooth enough, even for larger numbers of characters.

8.2.2 Avoiding collisions

The current solution still allows for collisions and intersections between crowd members and the formation in dense environments. A solution should be found to eliminate these before it works as intended and would look well enough in an application.

8.2.3 Testing with different general path-planning methods

I have only tested my solutions using ECM, IRM and Moussaïd's vision-based collision-avoidance method. It is possible that different methods give different results. Especially a different collision-avoidance method could greatly influence the results. It would be very interesting to see how these methods work with for example RVO [57], which is a collision-avoidance method that is generally held in high-regard.

8.2.4 Steering the DO-formation

In the current implementation, DO-formations are hard-coded to move ahead at a steady pace. This does not allow for a very flexible application. Before a formation modeled as dynamic object could be seriously used, the user needs to be able to steer the obstacle. We could think of user-controlled steering, or assigning a start- and goal-position involving path-planning to steer the object.

8.2.5 Influencing the DO-formation

In the current implementation, DO-formations are hard-coded to move ahead at a steady pace. It could be desirable to have the formation pay more attention to the crowd. When a formation approaches a crowd that is moving away at a slower pace then the formation itself, the formation should slow down to avoid collisions. Currently the RP-formation does slow down, but slows down too much. Ideally, a situation in between these two should be found: the formation does not just steamroll along, but does not give the crowd too much attention either.

8.2.6 Scaling the formation

The results shown al work with a formation with nine members. It would be interesting to see what happens when small formations (for example, groups of two or three people walking together in a crowd) or large formations (for example in a military setting) are used. It is highly likely that different sizes result in different behaviour.

8.2.7 Testing different variable settings

Although I have tested several settings for all variables and chosen the ones that seemed to achieve most realistic results, some time should be spend to explicitly find out which settings lead to which behaviour. Each variable was set by trial and error when it was introduced to the implementation, but it is possible that variables that were set at a later moment in time would incur different behaviour if the previously set variables had different values. This is especially important for the Directional Waves, as this method uses and combines quite many different variables.

8.2.8 Attractors and Repulsors

Steering away characters through directional waves can be seen as a form of a repulsor, as mentioned in Section 3.1. There is no reason why this could not be reversed - the velocity field could just as well work as an attractor. It would be interesting to see if the method can be adjusted to find a more global approach to attractors and repulsors.

9 Bibliography

[1] R. de Chiara A. Boccardo and V. Scarano. Massive battle: Coordinated movements of autonomous agents. In *Proceedings of Workshop on 3D Advanced Media In Gaming and Simulation*, pages 35–42, 2009.

[2] Arrestatieteam. De mobiele eenheid (ME). URL: `http://www.arrestatieteam.nl/eenhedenbinnenland/mobiele_eenheid.php`. 28-04-2012.

[3] T.R. Balch and M. Hybinette. Social potentials for scalable multi-robot formations. *International Conference on Robotics and Automation*, 1:730–80, 2000.

[4] K. Bathe. *Finite element method*. Wiley Online Library, 2008.

[5] C. Burstedde, K. Klauck, A. Schadschneider, and J. Zittartz. Simulation of pedestrian dynamics using a 2-dimensional cellular automaton. *Physica A: Statistical Mechanics and its Applications*, 295:507–525, 2001.

[6] J. Chang and T. Li. Simulating crowd motion with shape preference and fuzzy rules. *IEEE Conference on Cybernetics and Intelligent Systems, 2008*, pages 131–136, 2008.

[7] S. Chenney. Flow tiles. *SCA '04 Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animations*, 389:233–0242, 2004.

[8] G. Dhatt, E. Lefrançois, and G. Touzot. *Finite element method*. John Wiley & Sons, 2012.

[9] Z. Dostál and J. Schöberl. Minimizing quadratic functions subject to bound constraints with the rate of convergence and finite termination. *Computational Optimization and Applications*, 30(1):23–43, 2005.

[10] J.H. Ferziger and M. Perić. *Computational methods for fluid dynamics*, volume 3. Springer Berlin, 2002.

[11] P. Fiorini and Z. Shillert. Motion planning in dynamic environments using velocity obstacles. *International Journal of Robotics Research*, 17:760–772, 1998.

[12] J. Fredslund and M.J. Mataric. A general algorithm for robot formations using local sensing and minimal communication. *IEEE Transactions on Robotics and Automation*, 18:837–846, 2002.

[13] J.J. Fruin. The causes and prevention of crowd disasters. *First International Conference on Engineering for Crowd Safety*, pages 99–108, 1993.

[14] S.K. Gazda, R.C. Connor, R.K. Edgar, and F.Cox. A division of labour with role specialization in grouphunting bottlenose dolphins (tursiops truncatus) off cedar key, florida. *Proceedings of the Royal Society B*, 272(1559):135–140, 2005.

[15] R. Geraerts and M. Overmars. Enhancing corridor maps for real-time path planning in virtual environments. In *Computer Animation and Social Agents*, pages 64–71, 2008.

[16] P.E. Hart, N.J. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *Systems Science and Cybernetics, IEEE Transactions on*, 4(2):100–107, 1968.

[17] D. Helbing, A. Johansson, and H.Z. Al-Abideen. Crowd turbulence: the physics of crowd disasters. *Physical Review*, ICNM-V(June):967–969, 2007.

[18] D. Helbing and P. Molnár. Social force model for pedestrian dynamics. *Physical Review E*, 51:4282–4286, 1995.

[19] D. Helbing, F. Schweitzer, J. Keltsch, and P. Molnár. Active walker model for the formation of human and animal trail systems. *Physical Review*, 56:2527–2539, 1997.

[20] R.L. Hughes. The flow of human crowds. *Annu. Rev. Fluid Mech.*, 35:169–182, 2003.

[21] X. Jin, J. Xu, C.C.L. Wang, S. Huang, and J. Zhang. Interactive control of large-crowd navigation in virtual environments using vector fields. *IEEE Comput. Graph. Appl.*, 28(6):37–46, 2008.

[22] A. Kamphuis and M. Overmars. Motion planning for coherent groups of entities. *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04.*, 4:3815–3822, 2004.

[23] A. Kamphuis and M.H. Overmars. Finding paths for coherent groups using clearance. In *Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 19–28, 2004.

[24] I. Karamouzas, R. Geraerts, and M. Overmars. Indicative routes for path planning and crowd simulation. In *Proceedings of the 4th International Conference on Foundations of Digital Games*, pages 113–120, 2009.

[25] L.E. Kavraki, P. Švestka, J. Latombe, and M.H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, 12:566–580, 1996.

[26] J.A Kirkland and A.A Maciejewski. A simulation of attempts to influence crowd dynamics 1. *IEEE Computer Engineering*, 5:4328–4333, 2003.

[27] S. LaValle. *Planning Algorithms*. Cambridge University Press, available at `http://msl.cs.uiuc.edu/planning/`, 2006.

[28] Y. Li. *Real-time motion planning of multiple agents and formations in virtual environments*. PhD thesis, Simon Fraser University, 2008.

[29] Z. Lin, M. Broucke, and B. Francis. Local control strategies for groups of mobile autonomous agents. *IEEE Transactions on Automatic Control*, 49:622–629, 2004.

[30] D.J. Low. Statistical physics: Following the crowd. *Nature*, 407(6803):465–466, 2000.

[31] M. Moussaïd, D. Helbing, and G. Theraulaz. How simple rules determine pedestrian behavior and crowd disasters. *Proceedings of the National Academy of Sciences*, 108(17):6884–6888, 2011.

[32] M. Moussaïd, N. Perozo, S. Garnier, D. Helbing, and G. Theraulaz. The walking behaviour of pedestrian social groups and its impact on crowd dynamics. *PLoS ONE*, 5(4):e10047, 2010.

[33] S.R. Musse and D. Thalmann. Hierarchical model for real time simulation of virtual human crowds. *IEEE Transactions on Visualization and Computer Graphics*, 7(2):152–164, 2001.

[34] R. Narain, A. Golas, S. Curtis, and M.C. Lin. Aggregate dynamics for dense crowd simulation. *ACM Transactions on Graphics*, 28(5):122:1–122:8, 2009.

[35] Department of the Army. Civil disturbances. Field Manual 19–15, 11 1985.

[36] S. Patil, J. van den Berg, S. Curtis, M. Lin, and D. Manocha. Directing crowd simulations using navigation fields. *IEEE Transactions on Visualization and Computer Graphics*, 17(2):244–254, 2011.

[37] N. Pelechano. Crowd simulation incorporating agent psychological models, roles and communication. In *First International Workshop on Crowd Simulation*, pages 21–30, 2005.

[38] N. Pelechano, J.M. Allbeck, and N.I. Badler. Controlling individual agents in high-density crowd simulation. In *Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 99–108, 2007.

[39] S. Reicher. The st. pauls' riot: an explanation of the limits of crowd action in terms of a social identity model. *European Journal of Social Psychology*, 14(1):1–21, 1984.

[40] S. Reicher. "the battle of westminster": developing the social identity model of crowd behaviour in order to explain the initiation and development of collective conflict. *European Journal of Social Psychology*, 26(1):115–134, 1996.

[41] S. Reicher. The psychology of crowd dynamics. *Blackwell handbook of social psychology: Group processes*, pages 182–208, 2001.

[42] C. Reynolds. Flocks, herds, and schools: A distributed behavioral model. *SIGGRAPH '87 Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, pages 25–34, 1987.

[43] C. Reynolds. Steering behaviors for autonomous characters. *Game Developers Conference*, pages 763–782, 1999.

[44] M. Rickert, K. Nagel, M. Schreckenberg, and A. Latourd. Two lane traffic simulations using cellular automata. *Physica A: Statistical Mechanics and its Applications*, 231:534–540, 1996.

[45] M. Schuerman, S. Singh, M. Kapadia, and P. Faloutsos. Situation agents: agent-based externalized steering logic. *Computer Animation and Virtual Worlds*, 21(34):267–276, 2010.

[46] E. Schweiss, S.R. Musse, F. Garat, and D. Thalmann. An architecture to guide crowds using a rule-based behavior system. In *International Conference on Autonomous Agents: Proceedings of the third annual conference on Autonomous Agents*, volume 1999, pages 334–335, 1999.

[47] W. Shao and D.S. Terzopoulos. Autonomous pedestrians. *Graphical models*, 69(5-6):246–274, 2007.

[48] R. Silveira, L. Fischer, J.A. Salini Ferreira, E. Prestes, and L. Nedel. Path-planning for rts games based on potential fields. *Lecture Notes in Computer Science*, 6459:410–421, 2010.

[49] R. Silveira, E. Prestes, and L.P. Nedel. Managing coherent groups. *Computer Animation and Virtual Worlds*, 19(3-4):295–305, 2008.

[50] P.E. Stander. Cooperative hunting in lions: The role of the individual. *Behavioral Ecology and Sociobiology*, 29(6):445–454, 1992.

[51] G.K. Still. *Introduction to Crowd Science*. CRC Press, 2014.

[52] C. Stott, O. Adang, A. Livingstone, and M. Schreiber. Tackling football hooliganism: A quantitative study of public order, policing and crowd psychology. *Psychology, Public Policy, and Law*, 14(2):115, 2008.

[53] M. Sung, M. Gleicher, and S. Chenney. Scalable behaviors for crowd simulation. *EUROGRAPHICS*, 23(3), 2004.

[54] A. Treuille, S. Cooper, and Z. Popović. Continuum crowds. *ACM Transactions on Graphics*, 25(3):1160–1168, 2006.

[55] D.J. Tritton. Physical fluid dynamics. *Oxford, Clarendon Press, 1988, 536 p.*, 1, 1988.

[56] B. Ulicny and D. Thalmann. Crowd simulation for interactive virtual environments and vr training systems. In *Computer Animation and Simulation 2001*, pages 163–170, 2001.

[57] J. van den Berg, M.C. Lin, and D. Manocha. Reciprocal velocity obstacles for real-time multi-agent navigation. In *IEEE International Conference on Robotics and Automation*, pages 1928–1935, 2008.

[58] J. van den Berg, S. Patil, J. Sewall, D. Manocha, and M. Lin. Interactive navigation of multiple agents in crowded environments. In *Proceedings of the 2008 symposium on Interactive 3D graphics and games*, pages 139–147, 2008.

[59] W. van Toll. A navigation mesh for efficient density-based crowd simulation in multi-layered environments. Master's thesis, Utrecht University, 2011.

[60] W. van Toll, A. Cook IV, and R. Geraerts. Real-time density-based crowd simulation. *Computer Animation and Virtual Worlds*, 23:59–69, 2012.

[61] A. Varas, M.D. Cornejo, D. Mainemer, B. Toledo, J. Rogan, V. Muñoz, and A.J. Valdivia. Cellular automaton model for evacuation process with obstacles. *Physica A: Statistical Mechanics and its Applications*, 382:631–642, 2007.

[62] G. Visschedijk. The issue of fidelity: What is needed in 3d military serious games? Master's thesis, Universiteit Twente, 2010.

[63] P. Wagner, K. Nagel, and D.E. Wolf. Realistic multi-lane traffic rules for cellular automata. *Physica A: Statistical Mechanics and its Applications*, 234:687–698, 1997.

[64] Wikipedia. Hillsborough disaster. URL: `http://en.wikipedia.org/wiki/Hillsborough_disaster`. 28-04-2012.

[65] Wikipedia. Love parade stampede. URL: `http://en.wikipedia.org/wiki/Love_Parade_stampede`. 28-04-2012.

[66] Z. Xiaoping, L. Wei, and G. Chao. Simulation of evacuation processes in a square with a partition wall using a cellular automaton model for pedestrian dynamics. *Physica A: Statistical Mechanics and its Applications*, 389:2177–2188, 2010.

[67] H. Yeh, S. Curtis, S. Patil, J. van den Berg, D. Manocha, and M. Lin. Composite agents. *Eurographics/ ACM SIGGRAPH Symposium on Computer Animation*, pages 39–47, 2008.

[68] H. Yue, H. Guan, J. Zhang, and C. Shao. Study on bi-direction pedestrian flow using cellular automata simulation. *Physica A: Statistical Mechanics and its Applications*, 389:537–539, 2009.

[69] Olgierd Cecil Zienkiewicz and PB Morice. *The finite element method in engineering science*, volume 1977. McGraw-hill London, 1971.

A Report of interview with Martin Klootsema

On June 22nd 2012 I interviewed Martin Klootsema, who works for police Amsterdam-Amstelland as Inspector/Specialist Conflicts and Crisismanagement. As this interview was conducted in Dutch, I have transcribed the report in Dutch.

Ik heb de informatie die ik heb verkregen in dit interview opgedeeld in een aantal verschillende categorieën. De besproken categorieën zijn hoe er op dit moment wordt getraind, hoe de ME leden zijn onderverdeeld in pelotons, gebruikte tactieken en diverse overige details.

A.1 Hoe wordt er nu getraind?

Onderdeel zijn van de ME is een extra taak naast de gewone politietaken. Op het moment dat iemand geselecteerd wordt moet hij/zij eerst een fysieke test doen. Als die wordt gehaald volgt er een training van 3 maanden. Er wordt voornamelijk getraind op dingen als formaties, onberekenbaarheid van de menigte, en het vertrouwen op andere ME leden. Aan het eind worden ze getoetst in Ossendrecht. De opleidingen tot manschap en tot leidinggevende worden gesplitst gegeven. De commandanten worden bij het Politie Instituut Openbare orde en Gevaarsbeheersing (PIOG) in Ossendrecht opgeleid.

Daarna moeten ME-leden 1 keer per jaar een week trainen om bij te blijven. Daar ligt de nadruk op het groepsdynamische: communicatie, samenwerking en veiligheid.

Amsterdam-Amstelland beschikt over 7 of 8 gecertificeerde instructeurs. Dat is relatief veel in vergelijking met andere regio's.

A.2 Indeling

Er rukt altijd minimaal 1 peloton uit. Een peloton bestaat uit 2 secties die ieder uit 2 groepen gestaan. Elk onderdeel heeft zijn eigen commandant. Iedere groep heeft zijn eigen voertuig en dan is er ook nog een commando-voertuig. Soms gaat er een brand- en traangaseenheid (BraTra) mee. Die heeft een variabele samenstelling van specialisten, afhankelijk van de gelegenheid.

Er kunnen maximaal 4 pelotons worden ingezet zonder dat het invloed heeft op de dagdagelijkse werkzaamheden. Soms worden er extra pelotons van andere regio's geleend (met de inhuldiging na het WK 2010 werden er zelfs pelotons uit België geleend).

Daarnaast zijn er alternatieve troepen die bij sommige gelegenheden worden ingezet. De opties zijn:

- een groep paarden (in Amsterdam-Amstelland 8, in andere regio's gaan bereden troepen per 6),
- shovel (tegen barricades),
- waterwerper (om mensen weg te duwen/jagen),
- waterpeloton (voor binnenvaartschepen en dergelijke),

- hoogwerker (voor ontruimingen).

Bij ontruimingen wordt vaak de hoogwerker gebruikt om via het dak binnen te komen. Dan wordt er gewoon een stuk uit het dak gezaagd en gebruikt als ingang. Krakers leggen vaak overal booby traps en dit is een van de veiligste manieren. Het waterpeloton wordt bijvoorbeeld gebruikt als schepen blokkades op werpen. Dan gaan er ook schippers mee om de boten vervolgens naar de kant te brengen. Lang geleden waren er protesten en waterblokkades. De ontruimingen gingen gepaard met hoge agressie en daarom is toen het waterpeloton opgericht.

Er wordt ook veel gewerkt met Flex ME. Deze mensen gaan deels gekleed in ME kostuum (dus extra bescherming onder de broek, stalen neuzen etc.) maar deels als gewone politie. In principe voeren zij hun dagdagelijkse werkzaamheden uit, maar mocht het nodig zijn kunnen zij zich heel snel omkleden tot ME.

A.3 Tactieken

Verschillende menigten zorgen voor verschillende vormen van tegenstand en vereisen dus verschillende tactieken. Krakers, uitgaansmenigten en Marrokanen gedragen zich bijvoorbeeld zeer verschillend. Daarnaast zijn er natuurlijk verschillende situaties. Zijn er gasmaskers nodig? Is er rook? Enzovoorts! Bovendien kan de politiek invloed hebben. Meestal komt het niet daadwerkelijk tot een confrontatie. Bij voetbalsupporters komt het soms tot een confrontatie, maar bij krakers vrijwel altijd.

Als er barricades zijn opgeworpen door de menigte, worden die verwijderd door de ME. Daar wordt ook op getraind. Verschillende barricades vereisen verschillende technieken (pallets, olievaten, e.d.). De ME werpt zelf nooit barricades op.

Elke vrijdag- en zaterdagavond wordt er flex ME ingezet in de uitgaansregio's in Amsterdam-Amstelland. Van 21.30 tot 07.00 is er ME, bijvoorbeeld op het Rembrandtsplein. Tijdens alle wedstrijden van Ajax wordt er minimaal 1 peloton en een groep paarden ingezet. Toen dat nog niet zo was, liep het erg vaak uit de hand. Het duurt ongeveer een uur voordat een peloton ter plaatse is en dat duurde te lang.

De pelotonscommandant gebruikt geokaarten van de gemeente. Dat zijn kleurenfoto's die met een helikopter zijn gemaakt. Bovendien zijn er camerabeelden uit de commandovoertuigen en (soms) helikopterbeelden. Dit is wat er wordt gebruikt om een overzicht te vormen. In de commandokamer is er ook nog zichtbaar wat er op de lokale TV wordt uitgezonden en dergelijke extra informatie. Als dit relevant is wordt het doorgegeven.

Twee personen hebben een vaste plek in een formatie (de twee uiteinden). De rest vult aan aan de hand van die twee locaties. Hun volgorde kan dus veranderen. De commandant staat altijd achter de linie. Theoretisch zou hij niet bewapend hoeven zijn omdat hij nooit in contact met de menigte komt. Bij het verlaten en betreden van de auto zijn er ook twee vaste mensen die bij de deuren zitten. Zij stappen ook op een bepaalde manier in en uit de auto voor optimale dekking. Eerst verzameld iedereen bij de auto, en dan pas neemt men een formatie aan of stapt men weer in.

Voor de besluitvorming wordt gebruik gemaakt van het BOB model (Beeld, Oordeel & Besluit). Het idee is dat er eerst een goed beeld wordt gevormd, en er vervolgens knelpunten worden gesignaleerd. Dit is een element dat ook veel bij andere instanties in crisisbeheersing terug komt.

In principe is er altijd vooraf al contact met de organisatoren. Dat geldt voor demonstraties, voor feesten, voor voetbalwedstrijden, en zelfs voor ontzettingen. Er worden afspraken gemaakt over wat de kaders zijn. Bij demonstraties wordt er bijvoorbeeld vooraf een wandelroute afgesproken.

Het doel van een ME-actie is dat de opponent inziet dat zij geen kans maken en daarom vertrekken. Er wordt dus ook altijd twee keer omgeroepen dat de menigte moet vertrekken. Pas als de menigte dan blijft staan gaat de ME oprukken. Dat kan stapvoets of als charge (rennend). In beide gevallen gebeurt dat als linie: ieder lid moet dus in de gaten houden dat de rest ook nog mee kan komen.

De ME verliest eigenlijk nooit. Als ze de tegenstander niet aan kunnen, trekken ze zich onder dekking terug in de voertuigen. De groepen en pelotons gaan hergroeperen en komen met meer terug. Linies breken in de praktijk dus ook eigenlijk niet.

A.4 Details & Diverse

Er is geen enkele discussie. De pelotonscommandant overlegt van te voren met de sectiecommandanten. Daarna is er geen overleg. In de helmen zitten oortjes. De commandanten hebben ook microfoontjes. Iedereen kan alleen top-down praten: je kan uitsluitend bevallen horen van boven en geven naar beneden, maar niet overleggen of protesteren.

De kracht van de ME zit in de groep. Als er 1 commando wordt gegeven, is er meteen 1 formatie. Die eenheid in combinatie met de houding straalt heel veel uit. Hun aanwezigheid alleen al heeft een sterke invloed op de menigte. De ME heeft ALTIJD een formatie - ze staan nooit gewoon in een groepje.

Een commando bestaat altijd uit twee delen: het commando ("Charge over 50 meter...") en het uitvoeringscommando ("... nu!").

Krakers bestaan uit verschillende bewegingen. Enerzijds heb je de idealisten - die zijn relatief makkelijk te verwijderen. Maar er zijn ook veel criminelen in het krakerscircuit. En dan heb je ook nog "Black Block" - een groep die zich door heel Europa verhuurd om te vechten tegen de politie. Zij zijn altijd zeer gewelddadig. Zij dragen ook 'harnassen' (PVC buizen e.d.) onder hun kleren en hebben nooit een ID bij zich. S Bij de wedstrijd Ajax-NEC in 2006 of 2007 werd er geëxperimenteerd en werd er maar 1 sectie ingezet. Dat ging helemaal mis en is daarna nooit meer herhaald. De paarden moesten worden gebruikt om de mensen te ontzetten. Paarden zijn zeer krachtig.

Bij de kroningsrellen in 1980 duurde het anderhalve dag voordat het lukte de krakers op te breken. Er vond veel geweld plaats en veel politiemensen hebben daar trauma's aan over gehouden. In die tijd heerste er het idee dat "Alles moest kunnen". De provo's waren tegen de gevestigde orde en zagen de politie dus als rechtstreekse vijand. Dat ging gepaard met slechte communicatie. (Waarom?)

Over het algemeen gooit de menigte met wat er voor handen is, maar blijft ze daarbij op afstand. Er is nauwelijks fysiek contact. Veiligheid (van alle partijen) gaat voor alles! In de BraTra wagen is ook een EHBO-er die aan zowel de ME als haar opponenten eerste hulp biedt.