Torso Crowds

S.A. Stüvel

N. Magnenat-Thalmann

D. Thalmann

A.F. van der Stappen

A. Egges

Technical Report UU-CS-2015-018
November 2015

Department of Information and Computing Sciences
Utrecht University, Utrecht, The Netherlands
www.cs.uu.nl

ISSN: 0924-3275

Department of Information and Computing Sciences
Utrecht University
P.O. Box 80.089
3508 TB Utrecht
The Netherlands

Torso Crowds

S.A. Stüvel[1], N. Magnenat-Thalmann[2], D. Thalmann[2], A.F. van der Stappen[1] and A. Egges[1]

[1]Virtual Human Technology lab, Utrecht University, The Netherlands
[2]Institute for Media Innovation, Nanyang Technological University, Singapore

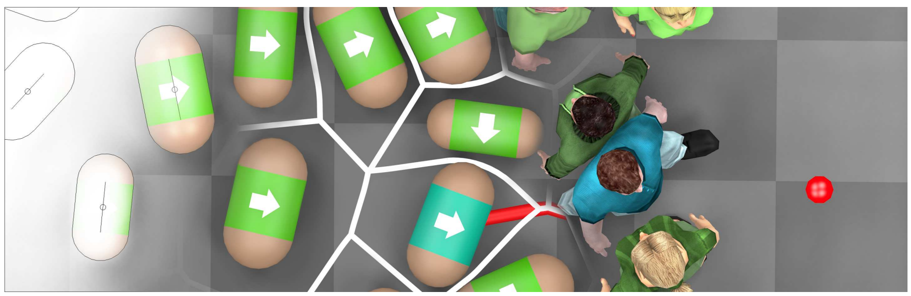


Figure 1: *Our crowd simulation model, showing different stages of the simulation. From left to right: the agent representation, calculation of the Voronoi diagram, planning a path towards a goal position, and finally the animation of virtual characters.*

Abstract

We present a novel dense crowd simulation method. In real crowds of high density, people manoeuvring the crowd need to twist their torso to pass between others. Our proposed method employs capsule-shaped agents, which enables us to plan such torso orientations. Contrary to other crowd simulation systems, which often focus on the movement of the entire crowd, our method distinguishes between active agents that try to manoeuvre through the crowd, and passive agents that have no incentive to move. We introduce the concept of a *focus point* to influence crowd agent orientation. Recorded data from real human crowds are used for validation, which shows that our proposed model produces equivalent paths for 85% of the validation set. Furthermore, we present a character animation technique that uses the results from our crowd model to generate torso-twisting and side-stepping characters.

1 Introduction

The shapes most often used to represent characters in crowd simulations are points and discs. In sparse crowd simulations, such a simple shape works well; the chosen representation does not have a large impact on the behaviour, as there is ample space around the agents. However, when the agents move very close to each other, motion follows shape. A common motion in dense crowds is the twisting of the torso, to squeeze through an opening between people. This is commonly seen at band performances, busy cocktail parties, or crammed lifts. Points and discs are ill suited for such situations, as the rotational symmetry prohibits planning of such twist. Instead, in this paper, we investigate an agent representation based on the torso. By employing a representation that is closer to the human shape, we expect to obtain more realistic human-like motions than disc-based crowd simulation methods.

[1]e-mail:{s.a.stuvel,a.f.vanderstappen,j.egges}@uu.nl
[2]e-mail:{nadiathalmann,danielthalmann}@ntu.edu.sg

Main contribution In this article we introduce the Torso Crowd model for dense crowd manoeuvring, based on a novel capsule-shaped agent representation modelling the characters' torsos (see Figure 1). Contrary to other crowd simulation systems, which often focus on the movement of the entire crowd, our method distinguishes between passive agents that have no incentive to move from their present location, and active agents that try to manoeuvre through the crowd towards a goal position. We introduce the concept of a *focus point* for crowd agents, that allows for more control and more realistic, social and complex behaviour. Furthermore, we validate the active agent behaviour using ground truth data, obtained by motion capturing a real crowd.

We use the term *agent* to indicate an abstract crowd agent such as a point, disc, or capsule. The term *character* designates a humanoid virtual character, whereas the term *people* refers to real humans. We mostly consider the motions of the upper body, i.e. the torso. The lower body is only considered at the final visualization step, where humanoid body animation is generated. Torso *twist* is not defined as a rotation relative to the lower body, but as a rotation relative to the agent's trajectory.

Organization The rest of the paper is organized as follows. Section 2 discusses related work. The overall setting of our crowd model is described in Section 3, followed by a description of the design of active (Section 4) and passive (Section 5) agents. We show our results, compare with a cylinder-based simulation method and with ground truth obtained from motion capture, and describe several simulated scenarios in Section 7. Section 8 describes the animation technique used to display the moving crowd agents as walking humanoid figures. Section 9 discusses future work, and concludes the article. The accompanying video can be found online at `http://stuvel.eu/video/torso-crowds`.

2 Related work

For a general overview of crowd simulation techniques and topics, we refer to the books by Thalmann and Musse [1], and Pelechano et al. [2]. In the remainder of this section, we focus on work that is related to the simulation of dense crowds.

There are many approaches to simulating crowds, each leading to different behaviour. *Flow-based* methods are macroscopic, focusing on the crowd as a whole. They support very large crowds, as high-level coordination prevents obstructions. Common examples are fluid dynamics [3] or gas kinetics [4], which can be applied to particle-based crowds, and are particularly suitable for high-density simulations. However, such approaches model a global optimum, whereas humans generally behave less optimally and can even get stuck in very dense situations. *Cellular automaton* approaches such as the works by Chenney [5] and Alizadeh [6] discretize floor space into cells, where every character can occupy exactly one cell, and vice versa. Although such approaches are computationally inexpensive and thus support large crowds, they also result in even spacing between agents, which will appear unnatural when densities are high. Both flow-based and cellular automaton systems do not consider how people move, and thus generally do not result in believable human crowds. *Agent-based* methods employing social forces, such as the HiDAC model [7], or planning in velocity space, such as the RVO model [8] generally allow for high-density crowds while supporting individual behaviour of agents, and can be extended to support physical interaction with obstacles and the environment [9]. These approaches avoid agent collisions at all costs, even to the point where all agents stop moving. In contrast, in actual dense crowds people frequently bump into each other. This is reflected in our method, as our agents prioritize motion over collision avoidance.

Common representations used in path planning and crowd simulation are points [3, 4] and discs [8, 10] Discs are the most commonly used representation due to their computational simplicity, and have proven to be suitable to simulate abstract (i.e. non-humanoid) crowds of any density. However, when using such a simple, rotationally symmetric representation, it becomes hard to animate more detailed human motion. This results in artefacts such as interpenetration of characters, unnatural distances between characters, and a lack of torso rotations. Furthermore, it has been shown that a disc does not accurately represent the actual volume occupied by the character in 3D space [11]. In Section 7 we show the importance of the agent shape in dense crowds, not just for realism of the motions, but also to support higher densities without getting stuck. Singh et al. use multiple discs [12] to represent an agent, and plan their motion using a footstep model. This approach allows for denser crowds than body-enclosing discs, and offers realistic walking animations. However, the ability to simulate dense crowds remains to be investigated. Finally, some crowd animation methods directly use animation data to drive the characters [13,14], also producing realistically animated characters. However, due to the high-level planning of these methods, planning the motion of individuals, such as specific characters moving towards their respective goal positions, is much harder to do. To our knowledge, our method is the first to present the capsule as agent representation in crowd simulations.

Our method employs Voronoi diagrams for planning motions through the crowd. The edges of such a diagram represent the path of maximum clearance between agents; intuitively this corresponds well with the desire of people to minimize perceived effort when walking [15, 16]. Stüvel et al. [17] showed that in a dense crowd people indeed move along such paths. The Explicit Corridor Map method by Geraerts [18] uses city-scale generalized Voronoi diagrams for path planning. Sud et al. [19] perform path planning based on 1st and 2nd order Voronoi diagrams, containing information about respectively the closest agent and the closest pair of agents. They employ a path scoring technique slightly resembling our proposed method. While their article promotes speed of computation, our approach focuses on a richer character representation, and validation against real crowd data.

3 Setting and Problem Formulation

Our crowd simulation system considers the torso as the main moving element. The algorithm is based on the findings by Stüvel et al. [17,20], who observed and recorded dense crowd behaviour. In their experiment, participants were given the task of manoeuvring through the crowd to predefined points. The movement of the crowd was recorded using a motion capture system, and these data serve as a ground truth for the behaviour of people actively manoeuvring through dense crowds. Our Torso Crowds model is designed to support the observed motions of the crowd-escaping participants, and believable simulation of essentially stationary people.

When observing dense crowds in general, and the previously mentioned recordings in specific, it is clear that torso rotations are critical when manoeuvring through a dense crowd. To support such rotations, our agents extend the common disc-based crowd agents, as shown in Figure 2. The common agent is defined as a point with a radius r'; our agent model extends this point to a line segment of length ℓ, with a (probably different) radius r. This extension eliminates the rotational

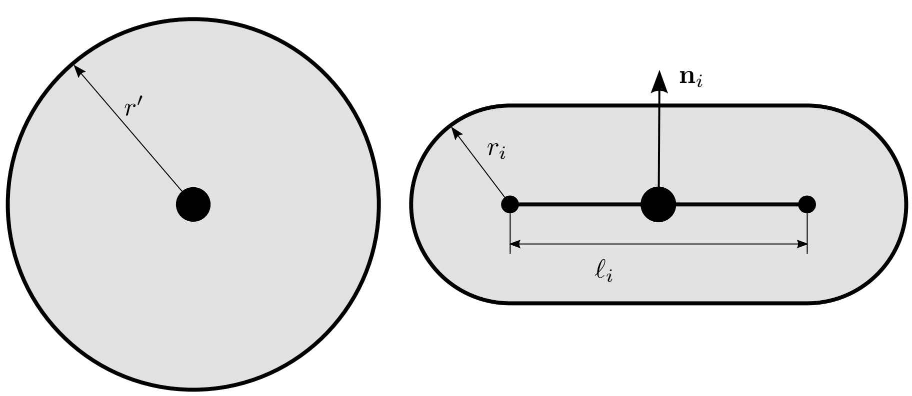


Figure 2: *Two types of crowd agent representation. On the left a common crowd agent: a point with a radius. On the right our crowd agent: a line segment with a radius.*

symmetry, thereby making it possible to plan torso rotations.

People standing still in a crowd behave differently from people trying to reach a certain goal position. In order to model these differences in behaviour, two types of agents are used in our crowd simulation technique. *Active agents* move to reach their goal position, whereas *passive agents* mostly stay in place, only moving to make room for other agents. Section 4 describes the behaviour of the active agents, while Section 5 describes the passive agents. Our method handles walls, doors and obstacles, as described in Section 6.

The crowd consists of N agents A_i, $i \in [1, \dots, N]$. Each agent A_i in reference placement is defined as the Minkowski sum of a line segment of length ℓ_i centred around the origin, and a disc of radius r_i. The placement of an agent is represented by a pair $(\mathbf{a}_i, \theta_i)$, where $\mathbf{a}_i$ and θ_i are the agent's position and orientation. For ease of discussion, we denote the direction of the forward-facing normal of the torso of agent A_i in placement $(\mathbf{a}_i, \theta_i)$ by $\mathbf{n}_i$, and the continuous set of points covered by its central axis by $\mathbf{s}_i$. These concepts will be detailed in the following sections.

4 Active agents

From observation of the previously mentioned ground truth data and dense crowds in general, we formulate the following assumptions as basis for our active agent model:

- People tend to choose a comfortable path, maximizing clearance by avoiding areas of very high density. Occasionally a less comfortable path is chosen, when it leads to an area of larger clearance.
- People tend to minimize perceived energy use [15], and thus prefer short, straight paths.
- People generally move in the direction of their goal, but divert from the shortest path when it is obstructed or when an alternative path is significantly more comfortable.
- Averaging at 0.4 m/sec, the traversing speed through a dense crowd is relatively low [20]. This, combined with the dynamic nature of crowds and possibly a lack of overview of the situation, makes long-distance planning of exact paths to the goal impractical.

The generalized Voronoi diagram (GVD) is a partitioning of the plane. In our crowd model, a cell is defined for each agent, being the set of all points that is closest to that agent. The GVD is represented as a pair $\{V, E\}$ of vertices V and edges $E \subset V \times V$ that represent the boundaries of those cells, where the edges are arcs (possibly with zero curvature, i.e. line segments). Every point on an edge or a vertex is equidistant to its neighbouring crowd agents. These edges form the medial axis between the agents, and thus represent a more or less comfortable path that maximizes clearance. Stüvel et al. [17] have observed that people in dense crowds indeed follow such paths (also see Figure 3).

In real situations, observing surrounding people, planning a path, and manoeuvring through the crowd, are intertwined in a continuous process. However, people are not continually reconsidering all their options all the time, but rather make more or less discrete decisions. Our agents reflect this behaviour by replanning their actions at a moderate rate.

Similar to Sud et al. [21], all active agents use the same GVD for planning their motion. The passive agents use a slightly altered GVD, as described in Section 5. As a result, the GVD needs to be calculated at most twice per simulation update, regardless of the crowd size and frequency of planning. For simplicity of computation, we use the central axes $\mathbf{s}_i$ to calculate the GVD, rather than the agent shape itself. Due to the nature of the capsule, where the distance from the axis to the edge of the shape is constant, the approximated GVD is very similar to the exact GVD, and the differences are unlikely to cause noticeable changes in the crowd's behaviour.

Our Torso Crowd method is suitable for simulating dense crowd manoeuvring, and based on experimental observations of such behaviour. However, in situations where the crowd is not dense, we have no proof of validity. As a consequence, our implementation switches to the RVO2 crowd simulation algorithm [8] when agents move out of the dense crowd. This can be seen in the

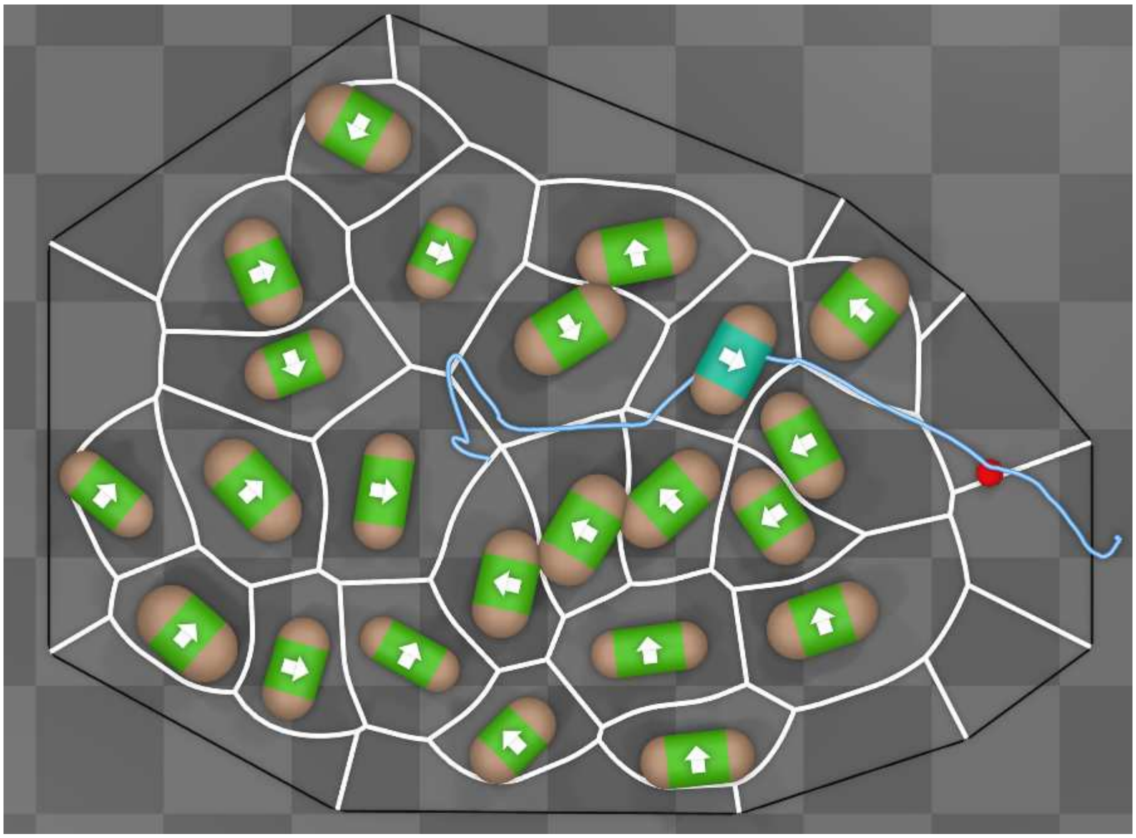


Figure 3: *Top-down view of a real, motion captured crowd, with the generalized Voronoi diagram in white lines.*

accompanying video, when agents walk out of a lift and into an empty hallway. We reuse the definition of *dense* situations by Stüvel et al. [17], namely those situations where there is at least three humans per square metre, as measured by the area of their Voronoi cell. Since we can measure this density on a per-agent basis, this decision is also made for each agent individually.

In the next subsections, we discuss the planning and execution of the agent's movement. Firstly, similar to real people, a desired position is planned, taking into account potential torso twists needed to reach that position. Since the available clearance at the planned position poses a bound on the torso orientation, this orientation is planned in a second step.

4.1 Limited-horizon path planning

To plan the movement of an active agent, the following steps are taken:

1. Find paths by exploring the vicinity in the GVD of the Voronoi cell containing the agent.

2. Calculate a score for each path, and determine the best-scoring path.

3. Calculate the desired agent orientation at the start of the path, accounting for available clearance.

The GVD provides proximity information in a natural way; the cell of the active agent represents its proximity, and the outgoing edges of that cell's vertices form paths between agents in its direct vicinity. The search is initialized by taking these outgoing edges, i.e. the edges that only have a single vertex incident to the active agent's Voronoi cell. This set is then extended,

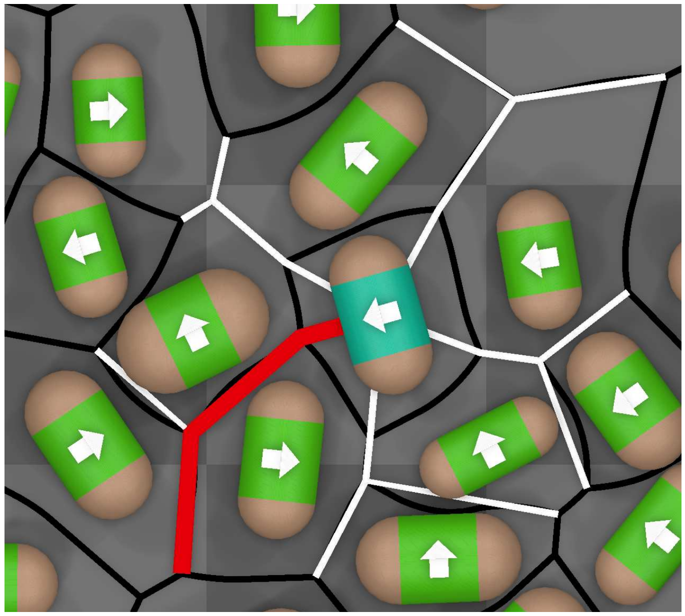


Figure 4: *Candidate paths in white lines, with the best-scoring path as a thick, red line. The paths are extended to the agent position. The goal position is bottom left outside the frame. Note that the paths along curved Voronoi edges are just drawn as straight line segments for simplicity.*

parametrized by given values for Euclidean distances H_D and H_ϵ, and edge count limit H_C, as follows: the outgoing edges are followed until either distance H_D, or edge count limit H_C is reached. For the latter limit, edges shorter than H_ϵ are ignored. Such short edges occur when three agents are almost equidistant, and the clearance between the agents would likely be perceived as a single space. Hence, such edges are unlikely to correspond to human perception. Even though this approach could theoretically lead to a path consisting of an arbitrarily large number of edges, such a situation does not occur in dense crowds when using crowd agents of more or less realistic human-like sizes. The resulting path P consists of a sequence of GVD edges; following the path should bring the agent closer to its goal.

After a set of candidate paths is found, each path is given a score. The agent will attempt to use the path with the highest score. The composite score function $S(i, P)$ takes agent A_i and path P. It enforces the behaviour of real people in dense crowds, based on the observations by Stüvel et al. [17]. As all score functions should be balanced to make a final decision as to the best possible path, they are combined into a weighted sum:

$$S(i, P) = w_g S_g(i, P) + w_c S_c(P) + w_l S_l(P) + w_m S_m(i, P)$$

where $S_g(i, P)$, $S_c(P)$, $S_l(P)$ and $S_m(i, P)$ are score functions, and w_g, w_c, w_l and w_m are weights given to these sub-scores. Values for these weights are determined in Section 7.1. In the following descriptions of each score function, $\mathbf{p}_0$ and $\mathbf{p}_f$ respectively indicate the initial and final vertex positions of path P.

Score function $S_g(i, P)$ drives the agent towards its goal. It measures how well the path leads to the goal position $\mathbf{g}_i$, expressing the distance, from the end of the path to the goal, as a ratio of the total Euclidean distance to the goal. This normalization ensures that the resulting score is independent of the absolute distance to the goal:

$$S_g(i, P) = 1 - \frac{|\mathbf{g}_i - \mathbf{p}_f|}{|\mathbf{g}_i - \mathbf{a}_i|}$$

Score function $S_c(P)$ measures the clearance radius along the path, ensuring that the agent prefers comfortable routes with large clearances. It consists of two components. The first component stems from the moderate rate replanning principle. It assumes that a person plans a motion towards a more spacious area; when this area is reached, a new decision can be made. The second component prefers motion along paths with as much clearance as possible. The GVD structure enables efficient calculation of clearance radius $C(\mathbf{x})$ at any point $\mathbf{x} \in \mathbb{R}^2$.

$$S_c(P) = w_c^F C(\mathbf{p}_f) + w_c^A \frac{1}{|P|} \int_{\mathbf{x} \in P} C(\mathbf{x})\ d\mathbf{x}$$

where $|P|$ indicates the total arc length of P, $\mathbf{x} \in P$ are the collection of all points along path P, and w_c^F and w_c^A are weights for respectively the final and the average clearance of the path. Due to implementation details of our GVD library, we only had access to the minimal clearance of edges, and approximated the integral using discretized summation. This score function also serves as a term to minimize the relative rotation of the torso with respect to the motion trajectory, due to the way the clearance information is used to plan torso orientations (see Section 4.2)

The conservation of energy can be broken down into two components: the minimization of the distance travelled, and the effort required to travel that distance. Score function $S_l(P)$ models the first component, and measures path length. This function combines with $S_g(i, P)$ into the preference of short paths leading to the goal.

$$S_l(P) = -\sum_{e \in P} |e|$$

Score function $S_m(i, P)$ represents the second component of energy conservation, by penalizing changes in momentum, i.e. sharp turns. Since we can safely assume that the mass of the agent is constant, any change in momentum is explained by a change (in the direction of) the velocity vector, which in turn can be modelled by the cosine similarity of the current velocity and the direction towards the path:

$$S_m(i, P) = \frac{\dot{\mathbf{a}}_i \cdot \mathbf{e}_0}{|\dot{\mathbf{a}}_i||\mathbf{e}_0|}$$

where $\mathbf{e}_0 = \mathbf{p}_0 - \mathbf{a}_i$, the vector connecting the agent to the starting point of the path.

A more elaborate alternative for $S_m(i, P)$ could calculate the weighted integral of the curvature along $\mathbf{e}_0$ and P, with the weight inversely proportional to the distance from the agent. This would take the curvature of the entire path into account, emphasizing more immediate momentum changes. However, our proposed approach is simpler, and seems to be sufficient in practice. Furthermore, due to the agent replanning while it is en route to its goal, effectively the curvature of the entire path is taken into account.

4.2 Torso rotation planning

Once the best path P has been chosen, which determines the next torso position, the torso *orientation* T_o is determined. For this we use the torso normal $\mathbf{n_i}$ of agent A_i. Torso orientation T_o consists of two components: heading T_h and torso twist T_t. The first component, T_h, represents a common nonholonomic walking motion along the start of the path. Its computation is trivial and not described here. The second component, torso twist T_t, adjusts for the minimal clearance along the start of the path. The clearance at later parts of the path is of less importance for the current torso twist planning, due to the moderate-rate replanning principle. The first edge of the path lies between two neighbouring agents, and ends at a point of local maximum clearance behind those two agents. It is this part of the path that is used for the planning of the torso twist. The clearance at a point indicates the distance from that point to the nearest agent capsule. To maximize the time for the agent to smoothly change its torso orientation towards the desired twist, we calculate the minimal clearance c along the first edge of path P. The torso twist T_t can then be expressed in radians as

$$T_t = \begin{cases} 0 & : c - r_i \geq w_i \\ \cos^{-1}\left(\frac{c - r_i}{w_i}\right) & : 0 < c - r_i < w_i \\ \frac{\pi}{2} & : c - r_i \leq 0 \end{cases}$$

where $w_i = \ell_i/2 + r_i$ is the half-width of the capsule. This results in two possible orientations, both of which will fit the available clearance equally well: $T_h + T_t$ and $T_h - T_t$ The choice for the final orientation is based on the findings by Stüvel et al. [20]. They observe that, while manoeuvring through a dense crowd, people tend to aim their torso normal towards their goal position. The absolute angle between the torso normal and the vector to their goal is limited to 90^o for 90% of the time, and never more than 120^o. Consequently, we choose $T_o = T_h \pm T_t$ such that this angle is minimized. Once the desired position $\mathbf{p}_0$ and orientation T_o have been calculated, each agent employs proportional-derivative controllers to steer towards the planned configuration.

So far we have discussed the general approach for an active agent. Based on observations from real crowds, we deviate from this approach when a character starts to move towards a goal. Stüvel et al. [17] observed that, before they start manoeuvring, people orient their torso towards their goal. Similar behaviour is incorporated into our crowd model. When an agent becomes active and starts planning its movements, it performs the same planning steps as described in the previous subsections. However, it discards the planned position $\mathbf{p}_0$, and rotates on the spot towards the planned orientation T_o. Subsequent planning steps are performed as described earlier.

5 Passive agents

In this section, we discuss the behaviour of *passive* crowd agents, which, in contrast to active agents, do not have an explicit target to navigate to. We consider two sometimes contradictory motivations for their placement: finding local comfort, and rotation towards a focus point. Our passive agents locally optimize their placement, making themselves as comfortable as possible, i.e. maximize the clearance around them, given the constraints of their immediate surroundings. The translation $\mathbf{t}_S$ to reach a more comfortable placement is described in Section 5.1. When the geometry of the environment and the configuration of the crowd allow for it, a trade-off is made between rotating to a comfortable orientation and a rotation towards a focus point. This can be the centre of a chatting group of people, the charismatic front man of a performing band, or simply the floor number display of the lift. To our knowledge, we are the first to use such a focus point in a crowd simulation system. The rotation ϕ_S from the current to the desired orientation is described in Section 5.2. Passive members of a crowd temporarily accept a less comfortable position in order to make way for someone else to pass; this avoidance by translating ($\mathbf{t}_A$) and rotating (ϕ_A) is described in Section 5.3. In Section 5.4 we show how these desires are combined into the agent's motion.

5.1 Space finding

Passive agents try to coarsely maintain their position. For example, even when a lift is crowded, the door is open, and outside the lift is a plethora of space, agents waiting in the lift will remain in that lift. Manoeuvring to a different area, such as stepping out of the lift, is considered active behaviour, and is described in the previous section. We use walls and doors (see Section 6) to delineate areas in the environment. To restrict the space finding algorithm to the agents' current area, our passive agents consider all doors as closed, regardless of their actual state. However, the agents do search for a better place to stand in their direct vicinity; this is what we call *space finding* behaviour. This results in a translation vector $\mathbf{t}_S$ from their current position to a more spacious position. Effectively it is a combination of comfort optimization and avoidance of passive agents.

Whether the space finding algorithm is engaged depends on the space around the agents. We assume that our passive agents like to stand in a spot where there is enough space surrounding them. When that is the case, i.e. the distance to the nearest neighbouring agent or obstacle is larger than a certain threshold, they remain stationary, even though there may be even more space available to them; the agent is marked as *happy* with its current placement, and will not engage the space finding algorithm (so $\mathbf{t}_S = \mathbf{0}$). This threshold can be configured individually for each agent, and can be a function of culture, scenario, or the geometry of the surroundings.

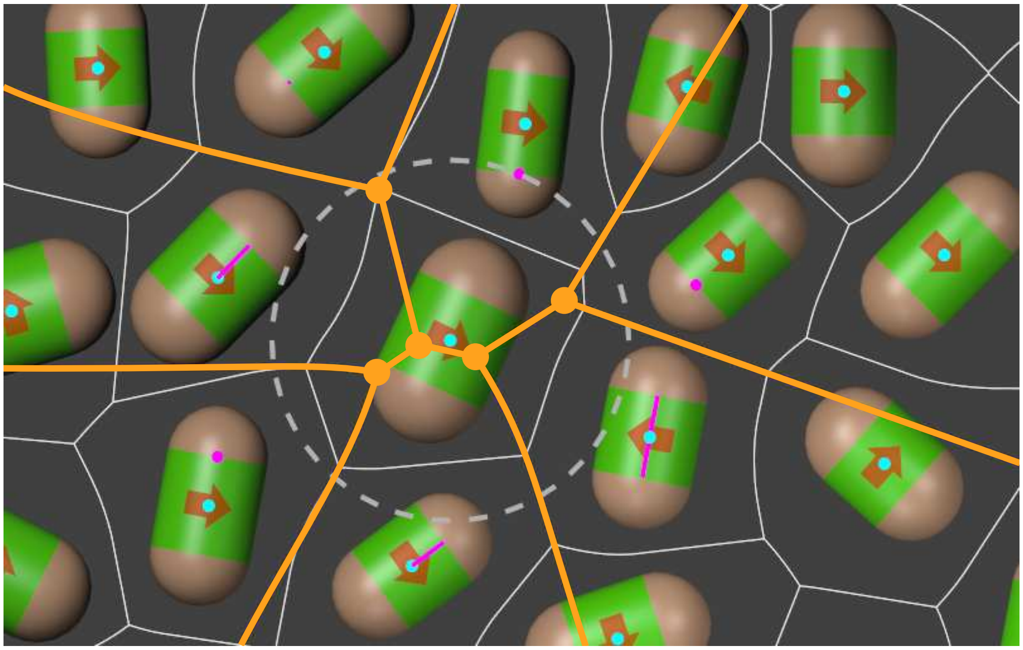


Figure 5: *Example of a local Generalized Voronoi Diagram (GVD), with points of maximal local clearance, in orange. The GVD of the crowd is shown in white. The features that define the local GVD are shown in magenta. The dashed circle shows the clearance of the agent.*

In tighter situations, our passive agents move to maximize their comfort. To obtain nearby candidate positions of maximal comfort, agents consider points of maximum clearance between their surrounding neighbours. By definition, such points correspond with vertices of a Generalized Voronoi Diagram (GVD, see Section 4) of those neighbours. Such a *local Generalized Voronoi Diagram* $\mathcal{L}_i$ of agent A_i is the GVD defined by $\mathcal{N}_i$, where $\mathcal{N}_i$ is the set of neighbouring agents and obstacles of agent A_i. $\mathcal{N}_i$ can be efficiently extracted from the GVD of the entire crowd, by iterating over the edges of the cell containing A_i, and taking the agents or obstacles on the opposite side of the edges. Note that agent A_i itself is *not* included in $\mathcal{L}_i$ (see Figure 5). The vertices of $\mathcal{L}_i$ correspond to local clearance maxima, and thus potentially comfortable positions for the agent to move to.

People try not to spend too much energy [15], and will accept a marginally more cramped situation when walking to a better spot would take a significant effort. We use the following energy minimization function to balance the gain (more available space) with the expended energy (the distance to travel to that space). All vertices $\mathbf{v}_j \in \mathcal{L}_i$ are considered potential better positions, and are given an energy cost

$$\begin{aligned} E(\mathbf{a}_i, \mathbf{v}_j) &= \frac{|\mathbf{v}_j - \mathbf{a}_i|}{C(\mathbf{v}_j) - C(\mathbf{a}_i)} \\ \mathbf{v}_d &= \operatorname*{argmin}_{\mathbf{v}_j \in \mathcal{L}_i} E(\mathbf{a}_i, \mathbf{v}_j) \\ \mathbf{t}_S &= \mathbf{v}_d - \mathbf{a}_i \end{aligned}$$

where $C(\mathbf{x})$ indicates the clearance around $\mathbf{x}$; $\mathbf{v}_d$ denotes the vertex with the lowest energy cost, and determines the agent's space finding translation vector $\mathbf{t}_S$. This scoring is efficient; we have found that, in practice, 89% of the time $\mathcal{L}_i$ contains no more than three vertices, with an average of 2.7 vertices.

5.2 Orientation finding

When agents are squeezed into a small area, they rotate themselves to fit the available space. However, if the constraints allow for it, the agents focus on a given point (a performing band on a stage, floor number display of a lift, etc.). This results in a rotation ϕ_S from the current orientation of the agent towards a desired orientation. The *focus point* is environment- and scenario-dependent, and can of course change over time and be different for each person or agent. It is denoted as $\mathbf{f}_i$ for agent A_i. The accompanying video shows the effect of this focus point. A group of agents have a focus point in the centre, and the video demonstrates the effect of increased density on this group: the group stays together, even though the focus point has no direct influence on the position of the agents (see Figure 9).

In the remainder of this section, p is the passive agent's index number, so $\mathbf{a}_p$ indicates its position. The angle between the agent's torso normal $\mathbf{n}_p$ (see Section 3) and the vector to its focus point $\mathbf{f}_p$ is defined as

$$\alpha_f = \angle(\mathbf{f}_p - \mathbf{a}_p, \mathbf{n}_p)\,,$$

where $\angle(\mathbf{x}, \mathbf{y})$ indicates the signed angle between two vectors on the interval $(-\pi, \pi]$.

When marked as *happy* with their current placement (which depends on the available clearance, as described in Section 5.1), our crowd agents rotate such that their torso normal points towards their focus point. In this case, we take $\phi_S = \alpha_f$.

The shape of the available space is the dominant factor in someone's orientation when that space is tight; one rotates to fit the little space available. The narrower the space, the less important any focus point becomes. To include this behaviour in our model, we inspect the shape of the agent's Voronoi cell. Since this cell contains all points that are closer to the agent than to any other agent, it is a good model for their available space. The *width* of the cell is defined as the minimal distance between two parallel lines that enclose the cell. The direction of these lines are a common measure for the oblong direction of the cell. However, this direction is not stable under small variations in agent configurations, so we use a more elaborate approach. To obtain a vector that indicates the overall orientation of the space, a Principal Component Analysis (PCA) [22] is applied. Such an analysis is applied to a point cloud, to determine its dominant direction. Since it cannot be applied to continuous shapes, intuitively we could sample the interior of the Voronoi cell to obtain such a point cloud. However, to increase computational performance, we limit this approach to the sampled cell edges; considering the results this is sufficient. The result of the PCA consists of a covariance matrix; the eigenvectors of this matrix, when ordered by their absolute eigenvalues c_1 and c_2, indicate the first and second principal components $\mathbf{C}_1$ and $\mathbf{C}_2$. In the remainder of this section, we assume that the absolute eigenvalues are ordered by magnitude, i.e. c_1 belongs to $\mathbf{C}_1$.

When the Voronoi cell of a passive agent has no clear orientation, the eigenvectors hold little information, and the eigenvalues will be more or less equal. In this case, the agent rotates towards the focus point. When the shape of the cell is elongated, and thus relevant for the orientation of the crowd agent, the first principal component aligns with the cell's shape. This relevance is indicated by a large difference between the first and second eigenvalue of the covariance matrix, i.e. $c_1 - c_2 \geq \epsilon_2$. In this case, there are two possible orientations for the agent, in which the agent's central axis $\mathbf{s}_p$ aligns with either $\mathbf{C}_1$ or $-\mathbf{C}_1$; the agent chooses the orientation that minimizes α_f. If there is no focus point, α_f is not defined, and the agent chooses the orientation that requires the smallest rotation from its current orientation.

$$\alpha_c = \angle(\pm\mathbf{C}_1, \mathbf{s}_p)$$

To ensure smooth transition between α_c and α_f, we blend between them depending on the eigenvalue difference:

$$\phi_S = \begin{cases} \alpha_c & \text{if } \ \epsilon_2 \leq c_1 - c_2 \\ I(\alpha_c, \alpha_f, t) & \text{if } \ \epsilon_1 \leq c_1 - c_2 < \epsilon_2 \\ \alpha_f & \text{if } \ \qquad c_1 - c_2 < \epsilon_1 \end{cases}$$

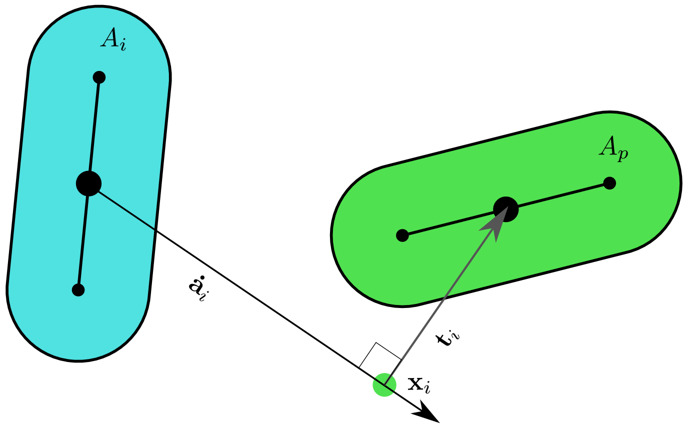


Figure 6: *Active agent avoidance; the passive agent A_p (green) will move to avoid the active agent A_i (cyan). Arrow $\mathbf{t}_i$ indicates the resulting avoidance vector.*

where $\epsilon_1 < \epsilon_2$, $I(\alpha_c, \alpha_f, t)$ indicates angular linear interpolation along the shortest arc for $t = (c_1 - \epsilon_1)/(\epsilon_1 - \epsilon_2)$. In our implementation, we use $\epsilon_1 = 0.015$ and $\epsilon_2 = 0.045$.

5.3 Avoidance of active agents

The behaviour of passive and active agents is quite different. Passive agents move slower, and try to divide the available space between them. Active agents move faster (when allowed by the constrained environment), and, more importantly, try to reach a specific goal. These differences are also reflected in the way that passive agents perform agent avoidance. This section describes how they avoid *active* agents[1].

Since far away agents have negligible chance of colliding with the passive agent, only those nearby are avoided. Of the active agents that are within an avoidance distance d_i of the passive agent, measuring distance between the agents' capsules, the nearest K are considered for avoidance. In our implementation we used $d_i = 0.4r_i$ and $K = 4$. The avoidance distance d_i can be varied to model observant (larger) or unaware (smaller) behaviour, and is not necessarily related to the agent's radius. In the following description of the avoidance behaviour, we denote the index of the active agent that is to be avoided as $i \in \{i_1, \ldots, i_K\}$, and the index of the passive agent as p. Agents that move away from the avoiding agent, i.e. where $(\mathbf{a}_i - \mathbf{a}_p) \cdot \dot{\mathbf{a}}_i > 0$, are safely ignored, as their motion is sufficient to avoid any collisions.

The avoidance behaviour consists of two components, a rotation ϕ_A and a translation $\mathbf{t}_A$. The passive agent rotates to minimize its width in the active agent's direction of movement, and it translates to move out of the way. The active agent's position $\mathbf{a}_i$ and velocity vector $\dot{\mathbf{a}}_i$ are used to determine a first-order approximation of its future trajectory.

Passive agent A_p rotates to reduce its width perpendicular to $\dot{\mathbf{a}}_i$, allowing A_i as much space as possible to pass. ϕ_A is chosen such that the central axis $\mathbf{s}_p$ aligns with to either $\dot{\mathbf{a}}_i$ or $-\dot{\mathbf{a}}_i$, depending on which produces the smallest rotation:

$$\phi_i = \angle(\pm\dot{\mathbf{a}}_i, \mathbf{s}_p)$$

The final rotation ϕ_A is the sum of the individual rotations ϕ_i. This summation is very simple; we are interested in a more refined approach, such as computing the rotation to avoid the one agent that is most likely to collide, based on its position and velocity. The avoidance of other agents could then be performed once that agent has been avoided. The investigation of more elaborate methods is left as future work.

To step out of the way of agent A_i, the passive agent translates perpendicular to the velocity vector $\dot{\mathbf{a}}_i$ (see Figure 6). For the active agent, we determine the line through $\mathbf{a}_i$ and oriented

[1] Avoidance of passive agents is handled by the space-finding algorithm, which is described in Section 5.1.

along $\dot{\mathbf{a}}_i$. For the passive agent, we determine the line orthogonal to $\dot{\mathbf{a}}_i$ and intersecting $\mathbf{a}_p$. The intersection point $\mathbf{x}_i$ of those lines determines translation vector $\mathbf{t}_i$:

$$\mathbf{t}_i = \frac{1}{\delta_i} \frac{\mathbf{a}_p - \mathbf{x}_i}{|\mathbf{a}_p - \mathbf{x}_i|}$$

with dampening factor $\delta_i > 0$. The dampening factor can be agent-specific, to allow for different personality traits. A high dampening factor will make the agent slower to respond than a low dampening factor. In the accompanying video we used $\delta_i = 200$ for all agents. The final agent avoidance translation vector $\mathbf{t}_A$ is the sum of individual translations $\mathbf{t}_i$.

5.4 Turning desire into action

The previous subsections described methods to obtain a vector towards more space $\mathbf{t}_S$, a rotation ϕ_S towards a focus point or to align with the available space, and translation $\mathbf{t}_A$ and rotation ϕ_A to avoid active agents. This section describes how our method selects which translation and rotation to use to produce the agent's motion.

The space finding translation vector $\mathbf{t}_S$ is only applied when certain conditions are met. Firstly, based on the principle of energy minimization, we assume that people accept a marginally worse situation when manoeuvring into a better spot would use significantly more effort than standing still. In our algorithm, the clearance at the found point must be significantly better than the agent's current situation; we use a threshold value of 125% of the agent's current clearance. Not only does this produce more natural results (an irregular distribution of free space among the crowd), it also prevents oscillation between points of similar clearance. Secondly, when making space for someone to pass (see Section 5.3), people generally accept a worse situation, as it will only be temporarily. However, people try to move towards an open space if one is available and can be reached while still allowing someone to pass, since this will make it both easier for the passing person and more comfortable for the avoiding person. To model this, space finding vector $\mathbf{t}_S$ is only applied when agent avoidance and space finding result in a translation in roughly the same direction; in other words, when the dot product $\mathbf{t}_S \cdot \mathbf{t}_A > 0$. When these are more or less opposite, only the agent avoidance is performed. The same approach is taken for ϕ_S and ϕ_A; if both rotate in the same direction, they are combined, otherwise only ϕ_A is applied.

$$\begin{aligned} \mathbf{p}_0 &= \mathbf{a}_p + \mathbf{t}_A + \begin{cases} \mathbf{t}_S & \text{if } \mathbf{t}_S \cdot \mathbf{t}_A > 0 \\ \mathbf{0} & \text{otherwise} \end{cases} \\ T_o &= \theta_p + \phi_A + \begin{cases} \phi_S & \text{if } \phi_A \phi_S > 0 \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

where θ_p is the passive agent's current orientation, and $\mathbf{p}_0$ and T_o are respectively the planned position and torso orientation as described in Section 4. The movement of the agent is controlled in the same way as described in that section.

6 Walls, doors, and other obstacles

In order to model realistic scenarios, our method supports walls, doors and polygonal obstacles. To integrate these into the crowd behaviour, they are all modeled as line segments and included as additional sites in the generalized Voronoi diagram (GVD). As a result, the GVD contains line segment sites for agents, walls, doors and obstacles. All these are interpreted by the crowd agents as impenetrable obstacles.

Doors are modeled as special wall segments that can be enabled when the door is closed, and disabled when the door is opened. As described in Section 5.1, doors are interpreted differently by active and passive crowd members. When a door is open, its line segment simply is not inserted into the active agents' GVD at the next simulation update. The GVD for the passive agents always inserts door line segments, to ensure that the space finding algorithm does not cross area boundaries.

In real life, people anticipate the movement of others. Anticipation in crowd simulation has been studied before [23–25]; in these works, crowd agents predict other agents' movements, and adapt their own motion to avoid collisions. Our crowd model takes the opposite approach; our active agents place information in the environment to notify passive agents of their intentions. As a real-life example of the intended behaviour, consider a person entering a lift; people appear to mentally model the space required for that person, and make space accordingly. Since the final orientation of the person is not known a-priori, a point would be sufficient to model this. In our simulation, active agents insert point obstacles in the passive GVD, at their goal position $\mathbf{g}_i$. As a result, the passive agents make space around this position, sooner than the avoidance behaviour would. This is only applicable in situations where the active agent's behaviour is predictable, such as when entering or exiting a lift or bus, which is why it is an optional feature of our crowd simulation method.

7 Results

In this section, we validate our Torso Crowds model against a real crowd, in order to find values for parameters that result in human-like behaviour. Furthermore, we investigate our model by looking at several scenarios. We also compare our model with a disc-based crowd simulation: Reciprocal Velocity Obstacles [8].

7.1 Validation and parameter optimization using a real crowd

To validate our crowd model behaviour, we used motion capture data of a real crowd [17]. This data set contains the torso width and thickness of each participant, and a recording of their locations and torso orientations during each of 47 trials. These recorded motions represent human behaviour in a real situation, and thus form suitable ground truth for our parameter optimization and model verification. We do note that the recordings were performed in a controlled environment, and thus may not be a faithful representation of day to day scenarios. We leave evaluation using real crowds, for example using video analysis, to future work. The data set is used to validate the behaviour of the active agents, and the passive agents in the interior of the crowd. In the experiment, the active participants had the concrete, reasonably realistic task of manoeuvring through a crowd to a given point. The rest of the crowd had to stand still in a dense configuration, which was necessarily synthetic for the participants at the edge of the crowd due to the set-up of the experiment. To compare the behaviour of the active participants with our crowd simulation system, we look at topological equivalence, rather than Euclidean distance between paths, as the exact positions of the paths are highly dependent on the behaviour of the passive crowd members. The parameters for the passive agents are simpler and more intuitive than those for the active agents, and were chosen based on visual inspection of the simulation results of the scenarios described in Section 7.2.

Our crowd model uses a number of parameters that determine the behaviour of the active agents, as described in Section 3. These parameters, with their optimized values, are shown in Table 1. To optimize these parameters, we used the following approach:

1. *Conversion:* The motion capture data is converted to our abstract agent representation, enabling us to input captured situations into our crowd simulation method.

2. *Test sets:* We choose N random frames from the recorded motion capture data. We ensure that each of the chosen frames represents a different situation. The set of frames is separated into two distinct, equally sized, randomly chosen subsets $\mathcal{T}$ for parameter tweaking and $\mathcal{V}$ for verification.

Table 1: *The path planner parameters obtained from our comparison with our ground truth data. All values were obtained by manual optimization.*

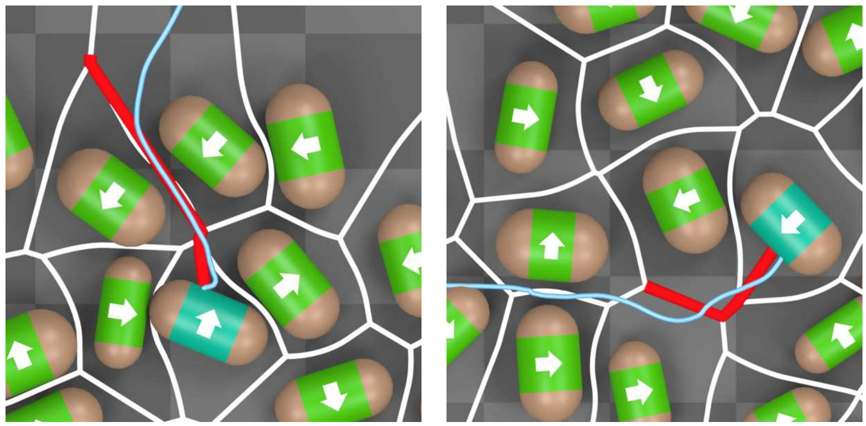
category	parameter	value	parameter	value
Planner horizon	H_C	3	H_D	1.50 m
	H_ϵ	0.05 m		
Score function weights	w_c	2.30	w_g	1.41
	w_l	0.21	w_m	1.00
Clearance weights	w_c^F	0.1	w_c^A	0.9

3. *Parameter tweaking:* For each frame in $\mathcal{T}$, the choices of the path planning algorithm are compared with the choices of the participant. We adjust parameters and repeat the comparison, until either all choices made by the path planner are equal to the choices made by the participants, or no more improvements can be made. When the planned path passes between the same agents as the participant's motion, they are considered equal.

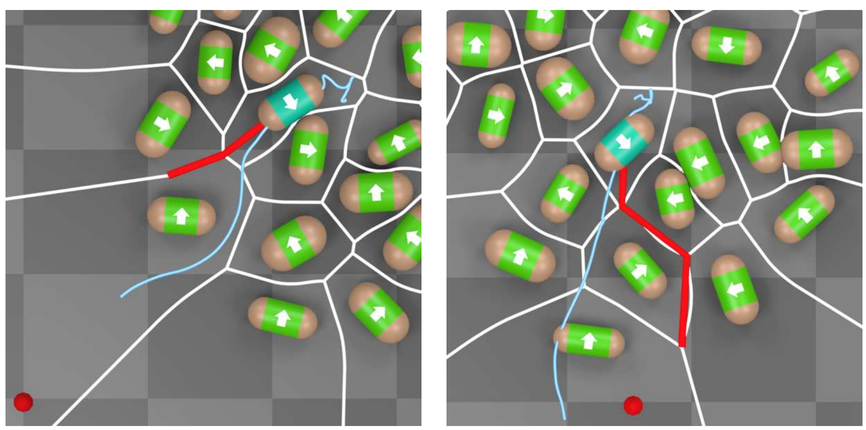
4. *Verification:* For each frame in $\mathcal{V}$, the same type of comparison is performed, as a verification of the parameters. We also measure the difference in planned and recorded torso twist.

We used $N = 80$ to tweak and verify our parameters. Little adjustment was needed during the tweaking phase, resulting in the parameters displayed in Table 1. To prevent over-fitting to our motion capture data data, we also validated against the behaviour observed in the simulations seen in the accompanying video. During the verification phase, the path planner chose a path that was topologically equivalent to the participants in 85% of the cases. Figure 7a shows examples of such correctly planned paths. In four of the six cases where the planner diverted from the recorded data, the planned path was equally plausible (see Figure 7b). In the recordings of the other two cases, at the exact frame used for validation, the participant shifted weight from one foot to the other while otherwise stationary, which resulted in a large change in the instantaneous momentum vector and thus in a different path being chosen (see Figure 7c); within 1/30 second after the test frame, the planner chose the same path as the participant in both cases. Of course this is not an issue when using simulated data, as our system does not model this weight shifting.

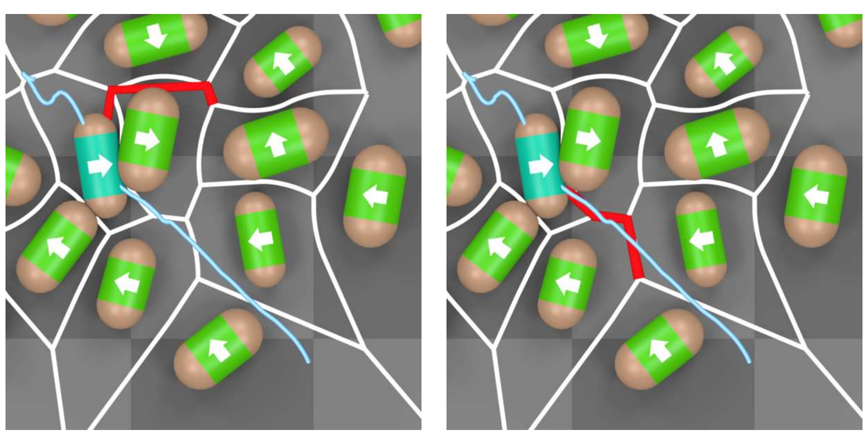
The verification of our model also includes a comparison between the planned and recorded torso orientations for the 34 test cases where the predicted path was topologically equivalent to the path of the recorded participant. To remove the influence of local path variations, we compare the torso *twists*, since these are relative to the agent's and participant's own paths. The twist is defined as the angle between the torso normal and the torso's instantaneous velocity vector (as described in Section 4.2). For each verification frame, our Torso Crowd method is used to plan the agent's next short-term target position $\mathbf{p}_0$ and torso twist T_t. The recording is then forwarded to the time where the participant reaches $\mathbf{p}_0$, after which his/her torso twist T'_t is determined. The error is then expressed as the signed difference between the twists: $E = T_t - T'_t$ where the sign of error E indicates whether our planner over-estimates (positive) or under-estimates (negative) the required twist. In 10 cases we under-estimated the required twist. We classify one of those cases as outlier; it showed a -42^o difference due to the participant moving at that angle even though it was not needed given the available space. In the other under-estimated cases the average error was small at -12^o, and the error was never more than -16^o. In 25 of the 34 cases, we over-estimated the required twist. This is easily explained by the fact that the plan is based on the GVD, which represents the current situation. In the recorded data, it is clear to see that the passive participants make space for the active participant, resulting in more available space, hence less torso twist is required. The average error when over-estimating was 22^o. The largest error in the predicted twist was 76^o. However, in this case the planned global torso orientation was reached within 0.8 seconds after reaching the planned position. Note that we compare the torsos at the moment in time where the distance between the recorded participant and the planned position is minimal. In 16 of the 34 test cases, either the planned torso twist T_t or the global torso orientation T_o is approached (within a 2^o error margin) within 0.5 seconds from that moment in time. This indicates that the



(a) *We consider the blue and red paths as topologically equal, since they choose the same route between the same agents.*



(b) *We consider those two paths as topologically unequal, since they choose a different route towards the goal, but equally plausible routes towards the goal position.*



(c) *In the left image, the planner took an unnatural decision, due to noise in the recorded velocity vector. However, 1/60 second later (right image) the planner made the same choice as the participant.*

Figure 7: *Comparison between the planned paths (thick red line) and the recorded motion capture data (thin blue line).*

recorded participant rotates at a slightly different rate, but still assumes the planned configuration shortly before or after. The average of the absolute error is quite small at 19^o, and the median

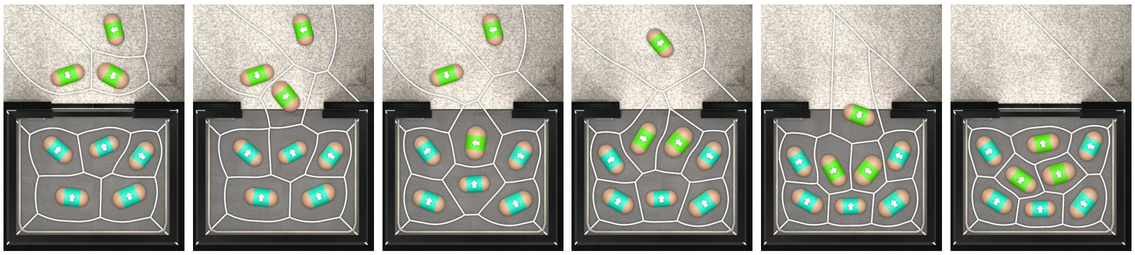


Figure 8: *Stills of the "small lift" scenario. Three agents enter the lift, while the others make space.*

of 16^o indicates that more than half of the predictions have a smaller-than-average error. We can conclude that our method for simulating active agents corresponds well with the ground truth data.

The avoidance behaviour of the passive participants was also investigated, to confirm that they show the alignment behaviour we model in Section 5.3. Since our aim is the simulation of dense crowds, we discarded the participants at the edge of the crowd, and limited this analysis to those that are in a dense situation as per the metric described by Stüvel et al. [17]. Their continuous motion was segmented into *avoidance actions*, which are defined as a period in which the participant shows a translation and/or rotation in order to make way for the active participant. In our data set, all avoidance actions consisted of at least a translation (average 0.09 m, $\sigma = 0.07$ m), which allowed us to find the peak in translation speed, and use the local minima around this peak to define the start and end timekeys of each avoidance action. At both timekeys, we measured the angle between the passive participant's torso segment $\mathbf{s}_p$ and the active participant's velocity vector $\dot{\mathbf{a}}_i$. By analysing 94 avoidance actions, we found that at the start of the avoidance action, the average angle was 42^o ($\sigma = 25^o$), and at the end timekey it was 30^o ($\sigma = 22^o$). A paired-samples T-test on the angles shows that this is a strong significant difference ($p < 0.0001$), indicating that there is indeed a trend to align with the active agent's velocity vector. The specific values of the observed averages are of relative importance, as we did not account for any anticipation or other temporal effects. Doing so may produce stronger results, which is left for future research. The simulated avoidance behaviour is parametrized, and can be adjusted to mimic these findings.

7.2 Examples and comparison with disc-based simulation

We have modelled several scenarios to test our crowd simulation method. As we focus on situations where a large part of the crowd stands still, typical tests where the entire crowd moves do not suffice. Furthermore, in dense crowds people often bump into each other, so a benchmarking method that penalizes collisions, such as SteerBench [26], will produce unrealistic scores. Instead, we have chosen to use a lift and a hallway to model crowded spaces. All scenarios are simulated at real-time on a single CPU core of a modern PC. The scenarios are shown in the accompanying video. For each scenario, we first show the simulated agents, and then animated characters that follow the motions of those agents.

Small lift In this scenario, the lift visits various floors, and on each floor agents get in or out of the lift (see Figure 8). This scenario shows the typical division of the available space seen in lifts: one person by itself stands more or less in the middle of the lift, while the space gets divided when more people enter. While waiting for their floor, the agents turn towards a common focus point: the floor indicator panel above the door. When agents leave the lift, the remaining space is used by the remaining agents. Note that, mimicking real life, the space is *not* optimally divided amongst the agents. Instead, agents around a gap, where an agent stood before leaving the lift, benefit most from the newly available space. The effect of the insertion of the immediate goal of active agents, as described in Section 6, can clearly be seen when passive agents make space as an

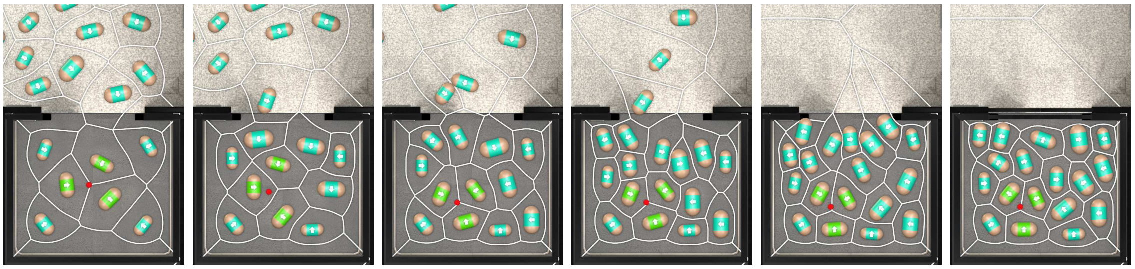
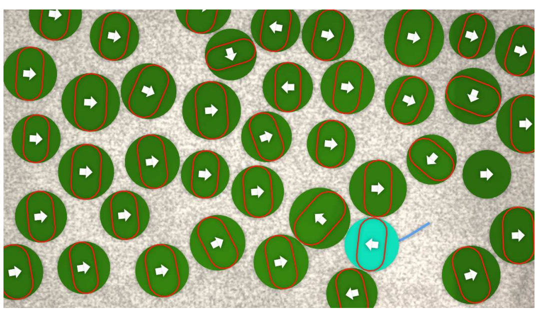


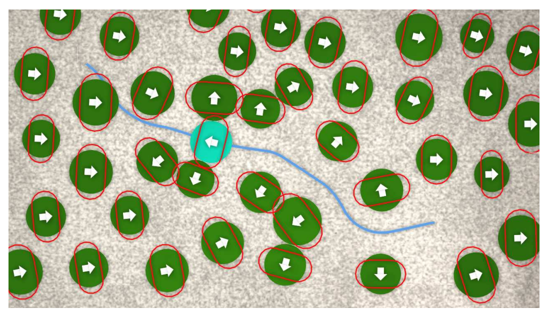
Figure 9: *Stills of the "large lift" scenario. The three green agents have a common focus point (the red dot).*

active agent enters the lift.

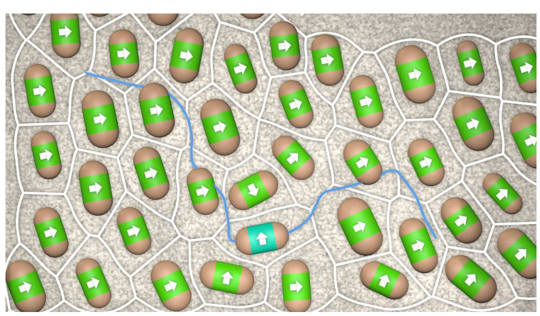
Large lift This scenario demonstrates what happens when a group of agents share a focus point, and the density of the crowd increases. The three green agents (see Figure 9) share a focus point that is positioned at the centroid of their positions. The other agents in the simulation do not have a focus point. The behaviour of the agents entering the lift is not necessarily natural, since half of them have been scripted to move to the back of the lift. This behaviour is more disruptive to the agents already present, and thus forms a more interesting scenario. Even though the focus point has no direct influence on the agents positions, the three agents stay together.



(a) *RVO with the same width as the capsules does not find a path to the goal.*



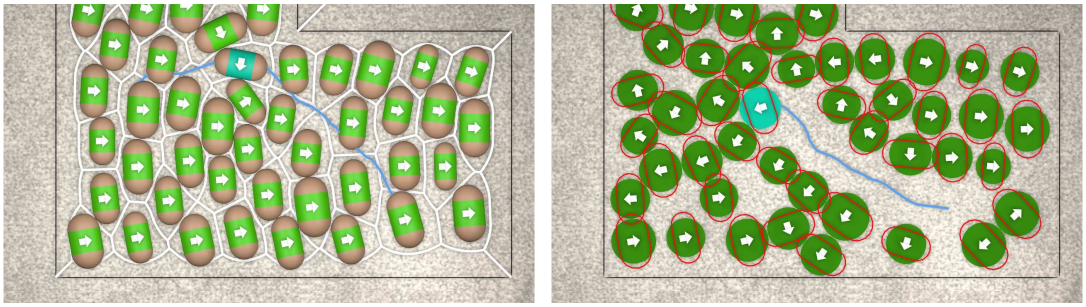
(b) *RVO with the same surface area as the capsules to allow manoeuvring.*



(c) *Torso Crowd finds a path to the goal.*

Figure 10: *Motion paths of a Torso Crowds agent, and RVO approaches. The RVO agents are displayed with a capsule shape overlay, to visualize intersections between agent-driven humanoid characters.*

Hallway In this scenario we show a character manoeuvring through a larger crowd in a hallway. We use this scenario to compare the behaviour of our Torso Crowds model with a widely accepted



(a) *Torso Crowd finds a path to the goal.*

(b) *RVO with the smaller agents, with the same surface area as the capsules, does not find a path to the goal.*

Figure 11: *Motion paths through an even denser crowd. RVO does not find a path to the goal, while Torso Crowds does. The RVO agents are displayed with a capsule shape overlay, to visualize intersections between agent-driven humanoid characters.*

crowd simulation model: Reciprocal Velocity Obstacles [8] (RVO). This comparison does not aim specifically at RVO; we just use RVO as a good example of a disc-based crowd simulation model. In this comparison, the Torso Crowd agents share the same focus point, out of view on the right-hand side. One agent tries to manoeuvre towards its goal position, while the remainder of the crowd is stationary; those agents have a zero preferred velocity.

Since RVO models agents as discs, we need to convert our capsule representation. We keep in mind that the agents actually represent humanoid shapes; making the RVO agents narrower will result in many undetected intersections. Therefore, the radius is chosen such that the disc encloses the torso capsule, as shown in Figure 10a. The blue line shows how far the agent was able to move: in such a dense, stationary crowd, the disc-shaped agents are too big to manoeuvre, while this density is not a problem for Torso Crowds (see Figure 10c). One of the underlying issues is that RVO agents only make space to avoid collisions. When the active agent slows down to avoid a collision, the surrounding agents only move with half the speed necessary to avoid the collision. This forces the active agent to slow down even more, finally forcing it to stand stand still. Its velocity vector then becomes zero and holds no information, and the agents in its surroundings will no longer move.

To give the RVO agents more space, we reduce the agent radii, such that the surface area of the agent's ground projection is equal to that of the capsule. This makes the RVO agents narrower but still thicker than the Torso Crowds agents, and results in an equal ground coverage percentage for RVO and Torso Crowds. The RVO agent can then successfully navigate the crowd, at the expense of intersections between the characters. Figure 10b shows this situation, with red capsules to visualize the character torsos. Statistics on our choice of agent sizes are shown in Table 2; the average width of 0.44 metres matches the average torso width (measured shoulder to shoulder) reported by Stüvel et al. [17]. We can further increase the crowd density; even the smaller RVO agents move slowly, and eventually do not find a path to the goal (Figure 11b). Our Torso Crowds model still handles this situation, and allows the agent to manoeuvre to its goal position (Figure 11a).

We can observe more differences. The Torso Crowds agent takes a longer path through the crowd, as it has been configured to avoid areas of low clearance (i.e. agents that stand close together). The RVO agent tries to maintain the shortest path by preferring velocities directly towards the goal position. Another difference is that RVO agents are limited to nonholonomic behaviour; an agent cannot take a step backward or to the side to make room for a passing agent, resulting in unrealistic instantaneous rotations when a human character is animated in its place. Where the passive Torso Crowds agents fill up the space in the wake of the blue agent to make themselves more comfortable, the green RVO agents remain stationary. These results show that the disc shape is not suitable for the simulation of dense crowds. We can also conclude that our

Table 2: *Agent diameters used in the comparative scenario, in metres. For capsule agents the diameter is defined as $2r_i + \ell_i$, whereas for disc agents this is $2r_i$.*

Simulation	**shape**	**min**	**max**	**mean**
Torso Crowds	capsule	0.382	0.504	0.443
Regular RVO	disc	0.382	0.504	0.443
Same-area RVO	disc	0.280	0.399	0.345

Torso Crowds model shows a wider range of motions.

8 Character animation

In order to display a humanoid crowd, the motions of the crowd agents need to be mapped to humanoid characters. This poses an under-specified problem. Since only the torso motion is simulated, the lower body orientation needs to be reconstructed before further body animation is possible. In this section we first describe our lower body estimation method, and then the proposed skeletal animation method.

8.1 Lower body orientation estimation

The motion data that contains the torso positions and orientations, either from our Torso Crowd simulation or a motion capture recording, is represented as mappings $\mathbf{T}_p : \mathbb{R} \to \mathbb{R}^2$ and $T_o : \mathbb{R} \to \mathbb{R}$, from time to respectively position and orientation in the ground plane. The lower body orientation is estimated based on two observations. Firstly, when manoeuvring, the lower body is oriented more or less in the same direction as the torso, and slightly turned towards the direction of motion. Secondly, the lower body cannot instantly change its orientation. The lower body estimation is expressed as a function $L_o(t)$, representing the angle of movement relative to the torso orientation. Together with $\mathbf{T}_p(t)$ and $T_o(t)$, it is used for the skeletal animation system described in the next section.

Firstly, we smooth the torso orientation $T_o(t)$. Ordinarily, when smoothing a signal, the smoothed signal lags behind the original. Since the lower body should "introduce" the motion, as per the first observation described earlier, we use this "smoothing lag" to our advantage by altering a simple Infinite Impulse Response filter such that it can be evaluated in reversed time:

$$T'_o(t) = T'_o(t+\delta) + (1-\beta)\Big(T_o(t) - T'_o(t+\delta)\Big)$$

where δ is the duration of a simulation frame, 1/60 second in our implementation, and $\beta \in [0, 1]$ determines the amount of smoothing. Here and in the next equation, the minus sign denotes the signed angular difference on the interval $(-\pi, \pi]$ over the shortest arc. In our implementation, we obtained sufficient smoothing using $\beta = 0.9$. The smoothing filter is applied in reverse time, thus $T'_o(t+\delta)$ is evaluated before $T'_o(t)$. As a result, the smoothed signal "lags in front" of the original signal, producing the desired motion.

Secondly, we calculate the angle between the smoothed torso orientation $T'_o(t)$ and the trajectory of the motion $\dot{\mathbf{T}}_p(t)$:

$$L_o(t) = \angle\dot{\mathbf{T}}_p(t) - T'_o(t)$$

where $\angle\dot{\mathbf{T}}_p(t)$ denotes the signed angle of the velocity vector with the world X-axis.

8.2 Skeletal animation

Once the lower body orientation $L_o(t)$ is determined, we can animate the skeletal structure that determines the character's pose. A commonly used technique for crowd animation is the use of a single walk cycle to animate characters at various speeds, where the animation playback

Figure 12: *A crowd of animated, human characters in a lift. The man in the blue clothing is the active character, whose torso twist is clearly visible. The path is shown in blue.*

rate depends on each character's walking speed. Such an approach is simple to implement, but does not support holonomic motion (such as side-stepping). Furthermore, it results in a direct dependency between walking speed and cadence (steps per minute). However, when people change their walking speed, both the cadence and stride length change [27]. This change in stride length cannot be captured in a single walk cycle, producing unnatural results. To address these issues, the basis for our animation technique is two sets of ten gender-specific walk cycles, consisting of an idle animation (0.00 m/sec), eight slow (0.45 m/sec) walk animations in different directions, and a faster (1.00 m/sec) straight forward walk. The eight slow animations consist of straight forward and backward walking, left and right sidestepping, and diagonal steps in four directions. The speed of 0.45 m/sec was chosen for those animations as it was found to be the average speed when manoeuvring through a dense crowd [20].

To produce a character that walks at the correct speed, the joint angles of the animations are blended using weights that depend on the speed of the crowd agent $|\dot{\mathbf{T}}_p(t)|$ and the direction of the motion relative to the torso $L_o(t)$. The speed determines whether we blend between the idle animation and slow walking, and both the speed and direction determine whether we blend between slow and fast walking. The animation's pivot point is positioned at $\mathbf{T}_p(t)$, and oriented at $T_o(t)+L_o(t)$ around the world up-axis. Constraints are placed on the spine bones to incrementally rotate the torso such that it is oriented at $T_o(t)$ around the world up-axis, producing the required torso twists. An example is shown in Figure 12.

9 Conclusion

In this article we have introduced a novel crowd simulation method, based on the manoeuvring of people in dense crowds. By extending the common disc-based agent representation to capsules, we are able to plan upper body twisting based on available clearance. Such torso twisting is critical for believable dense crowd manoeuvring.

Our method has been validated against data obtained from real crowd behaviour. The active agent behaviour matches paths chosen by humans in 85% of the cases, and produces different but equally plausible paths in 10% of the cases. The method's parameter values were manually optimized; it would be interesting to investigate automatic parameter tuning such as proposed by Wolinski et al. [28]. Even though we used a simplified Voronoi diagram, the resulting behaviour is a close match to the ground truth. Our majority of our validation focused on the behaviour of active agents; further comparison, with different ground truth data, could improve realism of the passive crowd members as well, and could strengthen our design decisions, such as the space-finding behaviour, the assumption that passive agents do not move to different rooms, and show whether the focus point we introduced actually exists as such in real crowds.

Regardless of the method to obtain the parameters, it is likely that their scope is limited to high-density situations. Since the planning of torso twist is no longer a necessity in lower-density crowds, our crowd simulation system switches between our proposed method and a different agent-based method aimed at regular locomotion, depending on the density of the crowd. Alternatively, our system could be extended to handle lower densities, by employing density-dependent parameter values; such a system should be relatively easy to add, since our density metric is agent-oriented, and the parameters are already adjustable for each agent. It would also be interesting to add velocity-based path planning to our method; for example, the change of clearance over time could be used to prefer small-but-growing openings in the crowd over larger-but-shrinking ones.

The passive agents use a generalized Voronoi diagram to find comfortable places to stand. By definition, such a diagram is symmetric, in that there is no distinction between agents and walls, or the front or rear of agents. This symmetry results in artefacts, such as agents standing too far away from walls. A possible solution may be found in a multiplicatively or additively weighted generalized Voronoi diagram [29], which might also be useful to model the asymmetrical nature of people' personal space [30]. However, since there are no suitable, robust implementations available, we are unable to implement such an approach at this time, and leave this to future work.

In our scenarios, each active agent was appointed a fixed, scenario-specific goal position. When that goal is reached, the agent switches to passive behaviour. Due to the dynamic nature of the crowd, the scripted goal position may not be the most comfortable (see Section 5.1, and the agent will move to a desirable point after reaching the goal. When approaching the goal, the active agent could use a local GVD to find a comfortable position in the goal area, and actively move there before switching to passive behaviour.

We presented an animation system that shows walking characters in a crowd using the motions obtained from the simulation. Our system uses a kinematic approach, hence it does not respond to inter-character collisions. Due to the density of the crowd, however, such collisions are likely to occur. We are currently investigating a method employing physics-based characters that follows our torso planning method [31]. Such a system would be able to respond to collisions in a physically correct way, and be used to plan lower-body motion. Another interesting way to extend our model is based on the observation that in dense crowds people often use their arms for navigation. Not only are they used to physically make space, but also for notification as to the intent to pass between people, and as a tactile addition to visual information about one's neighbours in the crowd.

Further research could employ the Torso Crowd representation to reduce the energy needed to manoeuvre a crowd for other crowd simulations. For example, our passive agents anticipate the motions of the active agents, and move aside and twist their torso to make space. Such behaviour can also be observed in less dense crowds, in cases where making twisting the torso is not a geometric necessity for someone to pass, but does provide them with a more energy-efficient path. This happens, for example, when making space for someone running towards a train. This shows that torso planning is not limited to dense crowds.

Acknowledgement

This research is supported by the Dutch nationally funded project COMMIT/. Part of this work has been performed during the stay of Sybren A. Stüvel at Nanyang Technological University, Singapore. We thank the group of Roland Geraerts for the use of their Explicit Corridor Map framework, which we used to calculate the Voronoi diagrams. We also thank the Blender community for their help.

References

[1] D. Thalmann and S. Musse, *Crowd Simulation*, 2nd ed. Springer-Verlag London, 2013.

[2] N. Pelechano, J. Allbeck, and N. Badler, *Virtual Crowds: Methods, Simulation, and Control*, ser. Synthesis Lectures on Computer Graphics and Animation. San Rafael: Morgan & Claypool Publishers, 2008.

[3] L. Henderson, "The statistics of crowd fluids," *Nature*, vol. 229, no. 5284, pp. 381–383, 1971.

[4] W. Kerr and D. Spears, "Robotic simulation of gases for a surveillance task," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Aug 2005, pp. 2905–2910.

[5] S. Chenney, "Flow tiles," in *Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation*. Eurographics Association, 2004, pp. 233–242.

[6] R. Alizadeh, "A dynamic cellular automaton model for evacuation process with obstacles," *Safety Science*, vol. 49, no. 2, 2011.

[7] N. Pelechano, J. Allbeck, and N. Badler, "Controlling individual agents in high-density crowd simulation," in *Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation*, 2007.

[8] J. van den Berg, S. Guy, M. Lin, and D. Manocha, *Reciprocal n-Body Collision Avoidance*, ser. Springer Tracts in Advanced Robotics. Springer Berlin Heidelberg, 2011, vol. 70, pp. 3–19.

[9] S. Kim, S. Guy, and D. Manocha, "Velocity-based modeling of physical interactions in multi-agent simulations," in *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2013.

[10] S. Lemercier, A. Jelic, R. Kulpa, J. Hua, J. Fehrenbach, P. Degond, C. Appert-Rolland, S. Donikian, and J. Pettré, "Realistic following behaviors for crowd simulation," *Computer Graphics Forum*, vol. 31, no. 2pt2, pp. 489–498, 2012.

[11] S. Stüvel, N. Magnenat-Thalmann, D. Thalmann, A. Egges, and F. van der Stappen, "Hierarchical structures for collision checking between virtual characters," *Computer Animation and Virtual Worlds*, vol. 25, no. 3-4, pp. 333–342, 2014.

[12] S. Singh, M. Kapadia, G. Reinman, and P. Faloutsos, "Footstep navigation for dynamic crowds," *Computer Animation and Virtual Worlds*, vol. 22, no. 2-3, pp. 151–158, 2011.

[13] K. Hyun, M. Kim, Y. Hwang, and J. Lee, "Tiling motion patches," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 19, no. 11, pp. 1923–1934, 2013.

[14] K. Jordao, J. Pettré, M. Christie, and M.-P. Cani, "Crowd sculpting: A space-time sculpting method for populating virtual environments," *Computer Graphics Forum*, vol. 33, no. 2, pp. 351–360, 2014.

[15] G. Zipf, *Human behavior and the principle of least effort.* Addison-Wesley Press, 1949.

[16] S. Guy, J. Chhugani, S. Curtis, P. Dubey, M. Lin, and D. Manocha, "Pledestrians: a least-effort approach to crowd simulation," in *Proceedings of Symposium on Computer Animation*, 2010.

[17] S. Stüvel, M. de Goeij, F. van der Stappen, and A. Egges, "An analysis of manoeuvring in dense crowds," in *Proceedings of Motion in Games conference*, ser. MIG '15, Nov. 2015.

[18] R. Geraerts, "Planning short paths with clearance using explicit corridors," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on.* IEEE, 2010, pp. 1997–2004.

[19] A. Sud, E. Andersen, S. Curtis, M. Lin, and D. Manocha, "Real-time path planning in dynamic virtual environments using multiagent navigation graphs," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 14, no. 3, pp. 526–538, May 2008.

[20] S. Stüvel, M. de Goeij, F. van der Stappen, and A. Egges, "An analysis of manoeuvring in dense crowds," Dept. of Information and Computing Sciences, Utrecht University, Tech. Rep. UU-CS-2015-006, 2015.

[21] A. Sud, E. Andersen, S. Curtis, M. Lin, and D. Manocha, "Real-time path planning for virtual agents in dynamic environments," in *ACM SIGGRAPH 2008 Classes*, 2008.

[22] L. Hyvärinen, "Principal component analysis," in *Mathematical Modeling for Industrial Processes*, 1970.

[23] S. Paris, J. Pettré, and S. Donikian, "Pedestrian reactive navigation for crowd simulation : a predictive approach," *Computer Graphics Forum, Eurographics 2007*, pp. 665–674, 2007.

[24] I. Karamouzas, P. Heil, P. van Beek, and M. Overmars, "A predictive collision avoidance model for pedestrian simulation," in *Motion in Games*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2009, vol. 5884, pp. 41–52.

[25] Y. Suma, D. Yanagisawa, and K. Nishinari, "Anticipation effect in pedestrian dynamics: modeling and experiments," *Physica A: Statistical Mechanics and its Applications*, vol. 391, no. 1-2, pp. 248–263, 2012.

[26] S. Singh, M. Kapadia, P. Faloutsos, and G. Reinman, "Steerbench: a benchmark suite for evaluating steering behaviors," *Computer Animation and Virtual Worlds*, vol. 20, no. 5-6, pp. 533–548, 2009.

[27] C. Kirtley, M. Whittle, and R. Jefferson, "Influence of walking speed on gait parameters," *Journal of Biomedical Engineering*, vol. 7, no. 4, pp. 282–288, 1985.

[28] D. Wolinski, S. J. Guy, A.-H. Olivier, M. Lin, D. Manocha, and J. Pettré, "Parameter estimation and comparative evaluation of crowd simulations," *Computer Graphics Forum*, vol. 33, no. 2, pp. 303–312, 2014.

[29] A. Okabe, B. Boots, K. Sugihara, and S. Chiu, *Spatial tessellations: concepts and applications of Voronoi diagrams.* John Wiley & Sons, 2009, vol. 501.

[30] L. Hayduk, "The shape of personal space: An experimental investigation." *Canadian Journal of Behavioural Science/Revue canadienne des sciences du comportement*, vol. 13, no. 1, p. 87, 1981.

[31] Z. Kavafoglu, E. Kavafoglu, and A. Egges, "Robust balance shift control with posture optimization," in *Proceedings of the Eighth International Conference on Motion in Games*, ser. MIG '15, forthcoming 2015.