Computers & Graphics 36 (2012) 70–79

Contents lists available at SciVerse ScienceDirect

Computers & Graphics

journal homepage: www.elsevier.com/locate/cag

Virtual Reality in Brazil 2011

Simulating crowds based on a space colonization algorithm

Alessandro de Lima Bicho [a], Rafael Araújo Rodrigues [b], Soraia Raupp Musse [b], Cláudio Rosito Jung [c,*], Marcelo Paravisi [b], Léo Pini Magalhães [d]

[a] Universidade Federal do Rio Grande—FURG, RS, Brazil
[b] Pontifícia Universidade Católica do Rio Grande do Sul—PUCRS, RS, Brazil
[c] Universidade Federal do Rio Grande do Sul—UFRGS, RS, Brazil
[d] Universidade Estadual de Campinas—UNICAMP, SP, Brazil

ARTICLE INFO

Article history:
Received 9 June 2011
Received in revised form
30 November 2011
Accepted 5 December 2011
Available online 23 December 2011

Keywords:
Crowd simulation
Virtual humans
Space colonization
Leaf venation
Tree development

ABSTRACT

This paper presents a method for crowd simulation based on a biologically motivated space colonization algorithm. This algorithm was originally introduced to model leaf venation patterns and the branching architecture of trees. It operates by simulating the competition for space between growing veins or branches. Adapted to crowd modeling, the space colonization algorithm focuses on the competition for space among moving agents. Several behaviors observed in real crowds, including collision avoidance, relationship of crowd density and speed of agents, and the formation of lanes in which people follow each other, are emergent properties of the algorithm. The proposed crowd modeling method is free-of-collision, simple to implement, robust, computationally efficient, and suited to the interactive control of simulated crowds.

1. Introduction

Animation of crowds finds applications in many areas, including entertainment (e.g., animation of great numbers of persons in movies and games), creation of immersive virtual environments, and evaluation of crowd management techniques (for instance, simulation of the flow of people leaving a football stadium after a match). Several techniques for modeling crowd dynamics already exist, but important aspects of crowd simulation have remained open for further research. Specifically, (i) the existing approaches are often focused on panic situations rather than usual (normal) behavior, in which people in the crowd have goals to seek; (ii) not integrated techniques are usually needed to calibrate the movement of people in low or high density crowds, and to affect local and global motion planning; (iii) existing crowd-modeling methods are often complex, and they require careful parameter tuning to obtain visually convincing results [1,2]; and (iv) collision-free motion, particularly in high densities is still a problem [3].

In this paper, we propose a novel crowd simulation algorithm that addresses some of the shortcomings of previous methods. Our model is based on the idea that individual agents affect each other by competing for the space where they move, which is represented explicitly by a set of marker points. In contrast to most methods proposed so far, the motion of each agent is thus affected directly not by the presence of neighboring agents, but by their absence, indicated by available markers. This change of perspective leads to a crowd model that is simple and robust, and it recreates emergently several aspects of real crowd behavior. These include collision avoidance (mathematically guaranteed by our algorithm), goal seeking, the dependency of the agents' speed and the smoothness of their trajectories on the density of crowds, and the tendency of people with similar goals in dense crowds to follow each other (form lanes). Furthermore, users can control crowd motion by interactively "spraying" or erasing free-point markers in selected areas of the scene.

In contrast to previous techniques for modeling crowd dynamics, which drew inspiration from psychology (behavioral models) or physics (e.g. models based on particle systems, force fields, or fluid dynamics), our method is inspired by a model of biological patterning. Specifically, it is derived from the space colonization algorithm introduced by Runions et al. [4,5] to simulate the development of leaf veins and trees. The simulated veins compete for access to sources of the plant hormone auxin, assumed to be distributed throughout the leaf blade. In the case of trees, individual branches compete for access to abstract markers of unoccupied space, which are distributed in the space of future tree crowns. In our work, each agent exploits the local availability of space in order to create an efficient path toward its goal while

[*] Corresponding author. Tel.: +55 51 3308 6231.
E-mail addresses: dmtbicho@furg.br (A.L. Bicho), rafael.rodrigues@pucrs.br (R.A. Rodrigues), soraia.musse@pucrs.br (S.R. Musse), crjung@inf.ufrgs.br (C.R. Jung), marcelo.paravisi@pucrs.br (M. Paravisi), leopini@fee.unicamp.br (L.P. Magalhães).

0097-8493/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cag.2011.12.004

avoiding other agents. Thus, the simulated motion of human-like agents in crowds is governed by competition for marker points similar to that used to model veins and trees. As such, this work demonstrates a surprising connection between methods used previously to generate biological patterns and crowd dynamics, since human beings also search for available space in order to move.

The remainder of the paper is organized as follows. Previous work on crowd dynamics, crowd simulation, and the space colonization algorithm are reviewed in Section 2. The proposed method for crowd simulation is described in Section 3, with mathematical details left to the appendices. Results of simulated experiments are presented and evaluated in Section 4. Finally, in Section 5, we present conclusions and suggest directions for future work.

2. Previous work

2.1. Crowd dynamics

The behavior of real crowds was analyzed in the late 1970s and 1990s [6–9]; results of their analysis provide a useful reference for simulation and animation of crowds. Two important aspects that guide the motion of real people are: goal seeking, reflecting the target destination of each individual; and the least-effort strategy, reflecting the tendency of people to reach the goal along a path requiring the least effort [9]. According to these strategies, people travel along smooth trajectories, since this requires less energy than frequent changes of direction or speed. In particular, adjustments of direction and speed, required to avoid collisions, are minimized. Further consequences of the least-effort strategy are the formation of lanes and the speed reduction effect. The first term refers to the tendency of people walking in the same directions to reduce their effort by closely following each other, while the second one refers to the reduction of speed in dense crowds.

The concept of personal space, the subject of study of interpersonal interactions in a spatial context (proxemics) [10], also plays an important role in population dynamics. Personal space can be thought of as an area with invisible boundaries, surrounding each individual, which should not be penetrated by other individuals in order for interpersonal interactions to occur comfortably. The size of this zone depends on the environment as well as the people culture, and decreases as the crowd density gets higher. In the context of simulations, the personal space determines the minimum distance that should be maintained among the agents.

2.2. Crowd simulation

Virtual crowds are usually modeled as collections of interacting agents, although treating a crowd as a continuum (for example, obeying laws of fluid dynamics) is also possible [11–13]. In *behavioral models*, movements of a group of agents are an emergent property of individual agents, which are both influenced by and influencing their neighbors. These individual behaviors are defined using sets of simple goal-oriented rules, such as ‘‘move with the average speed of your neighbors’’ or ‘‘keep an optimal distance to your neighbors’’. Behavioral animation was pioneered by Reynolds [14], who simulated flocks of birds and schools of fish assuming that each agent has direct access to the motion characteristics (position and velocity) of other agents. Tu and Terzopoulos [15] improved the conceptual realism of this work by endowing artificial fish with synthetic vision and perception of the environment. Both the original results of Reynolds and the models of Tu and Terzopoulos were confined to relatively small, low-density groups of animals.

The rules governing the movements of agents in behavioral models may be viewed as an abstract representation of the ‘‘psychology’’ of modeled individuals. In contrast, in *force-field models*, interactions among agents (in this case, often referred to as particles) are based on analogies with physics. For example, Helbing et al. [16,17] introduced abstract attraction and repulsion forces to simulate groups of people in panic situations. Braun et al. [18] extended this model by endowing agents with individual characteristics and including the concept of groups, which improved the realism of simulations. Ondřej and collaborators [19] explored the concept of synthetic vision to tackle the problem of collision avoidance in crowd simulation, while other sensorial attributes (such as sound and touch) were also explored in [20].

Hybrid methods have also been proposed. For instance, the approaches presented by Pelechano et al. [21] and van der Berg et al. [22] integrate behavioral and force-fields techniques in order to improve crowd control, aiming to minimize the drawbacks of both technologies. However, the negative aspect of these methods is the increase in complexity of the implementation.

Some effort has also been dedicated to GPU implementations of crowd simulation algorithms, aiming to reduce computational time. For instance, Rudomín and colleagues [23] presented an approach where the behavior of agents was specified as a finite state machine with a GPU implementation. Their GPU implementation presented a gain of 10 times when comparing with a CPU implementation for a large number of agents (around one million).

Each category of models presents a tradeoff. Behavioral models are suited for an individualized specification of agents, but global crowd control is more difficult to achieve because of the emergent character of the motions. In contrast, force-field models offer good global crowd control in high-density situations, but tend to generate less realistic motions of individual characters, which reflect their simplistic physical basis. This sets the stage for our method, in which crowds of agents obeying simple behavioral rules can be globally controlled and relatively realistic motions can be obtained without tedious parameterization as emergent properties of the model.

Also, an important drawback of existing approaches is collision treatment. In fact, quoting the statement of Patil et al. in a recent work [3], ‘‘It is important to note that *none* of the collision avoidance methods suggested in the literature can absolutely guarantee collision-free paths for all the agents in the simulation’’. In the proposed approach, collision avoidance for all agents is proved mathematically, being a relevant contribution of the paper.

2.3. The space colonization algorithm

The crowd modeling method proposed in this paper is based on the space colonization algorithm, which was originally proposed to model leaf venation patterns [4]. Variants of this algorithm make it possible to generate branching or reticulate patterns. Here we review the branching venation model, which more directly applies to crowd animation.

The venation model simulates three processes within an iterative loop: leaf blade growth, the placement of markers of free space, and the addition of new veins. The markers correspond to sources of the plant hormone auxin, which, according to a biological hypothesis, emerge in the growing leaf regions not penetrated by veins. A set of markers S interacts with the vein pattern, which consists of a set of points V called *vein nodes*. This pattern is extended iteratively toward the markers of free space. The markers that are approached by the advancing veins are gradually removed, since the space around them is no longer free. As the leaf grows, additional markers of free space are added in the space between existing veins and markers. This process continues until the growth stops, and there are no markers left.

The interplay between markers of free space and vein nodes is at the heart of the space colonization algorithm. During each iteration, a vein node is influenced by all the markers closer to it than any other vein node. Thus, veins compete for markers, and thus space, as they grow. There may be several markers that influence a single vein node $\boldsymbol{x}$: this set of points is denoted by $S(\boldsymbol{x})$. If $S(\boldsymbol{x})$ is not empty, a new vein node $\boldsymbol{x}'$ will be created and attached to $\boldsymbol{x}$ by an edge representing a vein segment. The node $\boldsymbol{x}'$ is positioned at a distance D from $\boldsymbol{x}$, in the direction defined as the average of the normalized vectors toward all the markers $\boldsymbol{s} \in S(\boldsymbol{x})$. Thus, $\boldsymbol{x}' = \boldsymbol{x} + D\hat{\boldsymbol{n}}$, where

$$\hat{\boldsymbol{n}} = \frac{\boldsymbol{n}}{\|\boldsymbol{n}\|} \quad \text{and} \quad \boldsymbol{n} = \sum_{\boldsymbol{s} \in S(\boldsymbol{x})} \frac{\boldsymbol{s}-\boldsymbol{x}}{\|\boldsymbol{s}-\boldsymbol{x}\|}. \tag{1}$$

The distance D serves as the basic unit of distance in the model and provides control over the resolution of the resulting structure. Once the new nodes have been added to V, a check is performed to test which, if any, of the markers of free space should be removed due to the proximity of veins that have grown toward these points.

The space colonization algorithm has subsequently been adapted to model trees [5]. Beyond the extension to 3D structures, the algorithm for trees introduced the notion of the radius of influence, which limits the distance from which markers of free space can attract tree nodes. Furthermore, the set of marker points is usually predefined at the beginning of simulation and no new markers are added afterwards, since, in contrast to the expanding leaf blade, the space in which a tree grows remains fixed.

3. Modeling crowds with the space colonization algorithm

The proposed method for crowd modeling is based on the space colonization algorithm. In its original applications to biological patterning, veins or tree branches could be regarded as paths created by vein or branch tips as they penetrated free space. In crowd simulation, these growing tips are identified with moving agents. Interestingly, an analogous relation between paths and motions can be observed in the development and applications of ideas related to particle systems. While some applications focused on the motion of particles (e.g., simulations of fire and fireworks), others emphasized their paths (e.g., simulation of grass and trees) [24].

The proposed approach preserves many characteristics of the original space colonization algorithm and its extension to trees. The key new elements, underlying the adaptation of the space colonization algorithm to crowd simulation, are listed below:

1. *Persistence of markers*: In contrast to the sources of auxin, which are permanently removed when reached by veins, the markers in crowd simulations are claimed by each agent temporarily, upon entering the agent's personal space, and are released when the agent moves away. The released markers can subsequently be used by other agents.
2. *Goal seeking*: The development of veins is guided locally by the presence of auxin sources in the proximity of a vein. In contrast, the motion of people is also influenced by the intention of each individual to reach a goal.
3. *Speed adjustment*: In the original space colonization algorithm, veins grow at a constant rate. On the other hand, agents vary their speed according to the available space in our crowd model.

To describe the proposed approach, let us consider a 2D simulation environment populated with discrete markers over "walkable" regions, using the dart-throwing algorithm [25]. For each agent I_i, there are assigned individual parameters, namely its current position $\boldsymbol{x}_i(t)$, its current goal $\boldsymbol{g}_i(t)$, its desired maximum speed $s^i_{\max}$, and its perception field (the maximum distance from which an agent can perceive markers), modeled as a circular region with radius R_i.[1] Next we describe the procedure for updating the position of each agent.

3.1. Computation of displacement vector

For a given agent I, let us consider set $S = \{\boldsymbol{a}_1, \boldsymbol{a}_2, \ldots, \boldsymbol{a}_N\}$ of markers that are closer to agent I than any other agent and that are also within its personal space. In the simulation of vein development, the orientation vectors were normalized and simply averaged to define the direction of vein growth (Eq. (1)). However, in the simulation of the agents' motion we also need to take their goal vectors into account. To this end, we weight each orientation vector according to the degree to which it is aligned with the agent's goal. Specifically, the *tentative motion vector* $\boldsymbol{m}$ is computed as

$$\boldsymbol{m} = \sum_{k=1}^{N} w_k (\boldsymbol{a}_k - \boldsymbol{x}), \tag{2}$$

where coefficients w_k are weights given by

$$w_k = \frac{f(\boldsymbol{g}-\boldsymbol{x}, \boldsymbol{a}_k-\boldsymbol{x})}{\sum_{l=1}^{N} f(\boldsymbol{g}-\boldsymbol{x}, \boldsymbol{a}_l-\boldsymbol{x})}. \tag{3}$$

To determine function f, let us first assume that all markers $\boldsymbol{a}_k$ affecting agent I are at the same distance $\|\boldsymbol{a}_k-\boldsymbol{x}\|$ from this agent. Such function should prioritize markers that lead the agent directly to its goal, i.e., it should (i) reach its maximum when the (non-directed) angle θ between $\boldsymbol{g}-\boldsymbol{x}$ and $\boldsymbol{a}_k-\boldsymbol{x}$ is equal to 0°; (ii) reach its minimum when $\theta = 180^\circ$; and (iii) decrease monotonically as θ increases from 0 to 180°. Also, if the distances $\|\boldsymbol{a}_k-\boldsymbol{x}\|$ differ, the markers further from the agent should have relatively smaller weights, to prevent them from dominating the computation of the tentative motion vector $\boldsymbol{m}$.

A possible choice for f that satisfies these assumptions is

$$f(\boldsymbol{x}, \boldsymbol{y}) = \frac{1+\cos\theta}{1+\|\boldsymbol{y}\|} = \frac{1}{1+\|\boldsymbol{y}\|}\left(1 + \frac{\langle \boldsymbol{x}, \boldsymbol{y} \rangle}{\|\boldsymbol{x}\|\|\boldsymbol{y}\|}\right), \tag{4}$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product. In Appendix A, we demonstrate that, if markers claimed by an agent are continuously and uniformly distributed in its personal space and within the agent's perception field, this choice of f will guarantee that the tentative motion vector $\boldsymbol{m}$ will point in the direction of the agent's goal. Obviously, for other marker distributions (e.g. when the user inserts obstacles) or when the perception field of an agent does not lie in the interior of its personal space (e.g. when there are neighbors at a short distance), the agent's direction may deviate somewhat from its goal. We also show that (i) the position $\boldsymbol{x}$ of agent I, displaced by vector $\boldsymbol{m}$, remains within the current personal space of agent I, and (ii) the magnitude of vector $\boldsymbol{m}$ increases with the size of this space. Taken together, these properties make vector $\boldsymbol{m}$ a good candidate for specifying next-step movement of the agent, guaranteeing a collision-free trajectory and capturing the increase of speed in larger spaces. However, in calculating the actual displacement, we have to also consider the maximum speed (displacement per

[1] For clarity, indices i and t will be omitted from now on, unless necessary.

simulation step[2]) $s_{\max}$ of the agent. Consequently, we calculate the actual displacement $\boldsymbol{v}$ as

$$\boldsymbol{v} = s\frac{\boldsymbol{m}}{\|\boldsymbol{m}\|}, \quad \text{where } s = \min\{\|\boldsymbol{m}\|, s_{\max}\}, \tag{5}$$

so that the position of the agent is updated through

$$\boldsymbol{x}(t+1) = \boldsymbol{x}(t) + \boldsymbol{v}. \tag{6}$$

Eq. (5) implies that if $\|\boldsymbol{m}\| > s_{\max}$, the speed of the agent is limited by $s_{\max}$. Otherwise, the speed is given by $\|\boldsymbol{m}\|$. It is important to notice that, if the radius R of the perception field is too small, the magnitude of the direction vector calculated in Eq. (2) will always be smaller than $s_{\max}$, making it impossible for an agent to achieve its maximum speed. In Appendix A, we present the relation between R and the expected value of the magnitude of displacement per step $\|\boldsymbol{m}\|$, which may serve as a guidance for setting an appropriate value for R.

It is important to note that the proposed approach generates collision-free motion if the agents are infinitesimal (i.e., present no area). Otherwise, the center of the agents does not collide, but their bodies may interpenetrate. However, as shown in the next section, a visual inspection indicates that very few collisions occur when using the proposed approach. In any case, an extension that copes with collision-free motion for finite-size agents is presented in Appendix A.3, and a preliminary comparison between infinitesimal versus finite-sized agents is presented in Section 4.4.

4. Experimental results

In this section we present several examples that illustrate various features of the proposed crowd simulation method. In particular, we show that different aspects of the crowd dynamics outlined in Section 2.1 are emergent properties of our model. It is important to mention that, unless otherwise indicated, all results were obtained using the same set of parameters, regardless of the density of simulated crowds. The density of markers was set to 15 markers/m^2, and the personal space R surrounding each agent had radius of 1.25 m. Since every person in real life has their own preferred maximum speed, the individual maximum speed $s^i_{\max}$ for each virtual agent I_i was drawn at random within the interval 0.03–0.05 m per time step of 1/30 s, corresponding to the range 0.9 m/s–1.5 m/s. See Appendix A.2 for a further discussion of the relation between $s_{\max}$ and R. The default simulated environment was a square of dimensions 50 × 50 m^2, and all experiments were performed using an Intel® Core$^{\text{TM}}$ 2 Duo T7500 2.2 GHz mobile processor, 3 GB DDR2 memory and 128 MB NVIDIA® GeForce® 8400M GS video card.

We used two methods to visualize simulation results. In 2D visualizations, agents are shown as moving dots associated with line segments, and these lines point to the markers influencing each agent. An example of such a visualization is given in Fig. 2(b). It reveals that, in this case, the agents in the center of the crowd have fewer markers available to them, and therefore a more limited range of movements than the agents near the boundaries. In 3D visualizations, agents are represented as articulated virtual humans. An example of such a visualization is shown in Fig. 3.

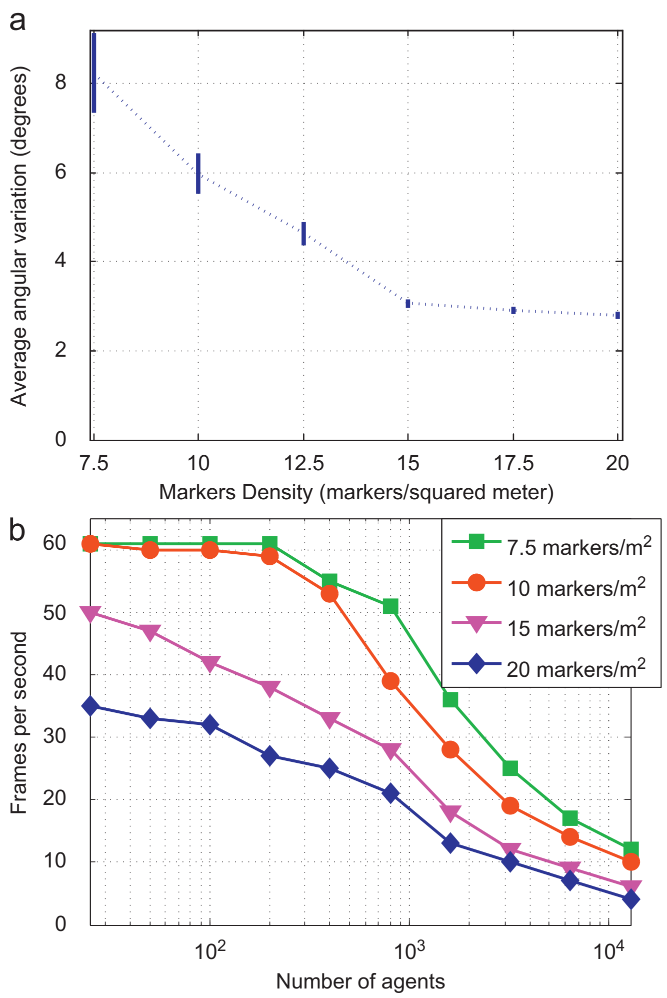

Fig. 1. (a) Average variation of the agents direction per simulation step as a function of the density of markers. Vertical bars indicate standard error. (b) Simulation speed as a function of the number of agents, evaluated for four different densities of markers. All standard deviations were less than 1 and thus are not shown.

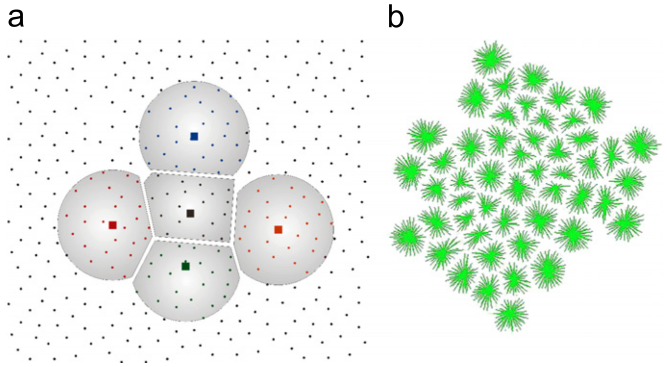

Fig. 2. (a) Personal space (shaded regions) and markers (dots) associated with five sample agents (squares). The markers captured by each agent are shown in the same color as the agent. (b) Line segments connect agents and corresponding markers in 2D visualization. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

4.1. Impact of the density of markers

In order to properly set parameters for further experiments, we analyzed the impact of the density of markers on the trajectory of agents and the computational efficiency of simulations. According to the least effort hypothesis, the ideal trajectory in the simplest case of a single agent is a straight line between its initial position and the goal. Fig. 1(a) shows the average and

[2] Since the speed is given in "displacement per simulation step", it in fact relates to the distance traveled by the agent at each time step.

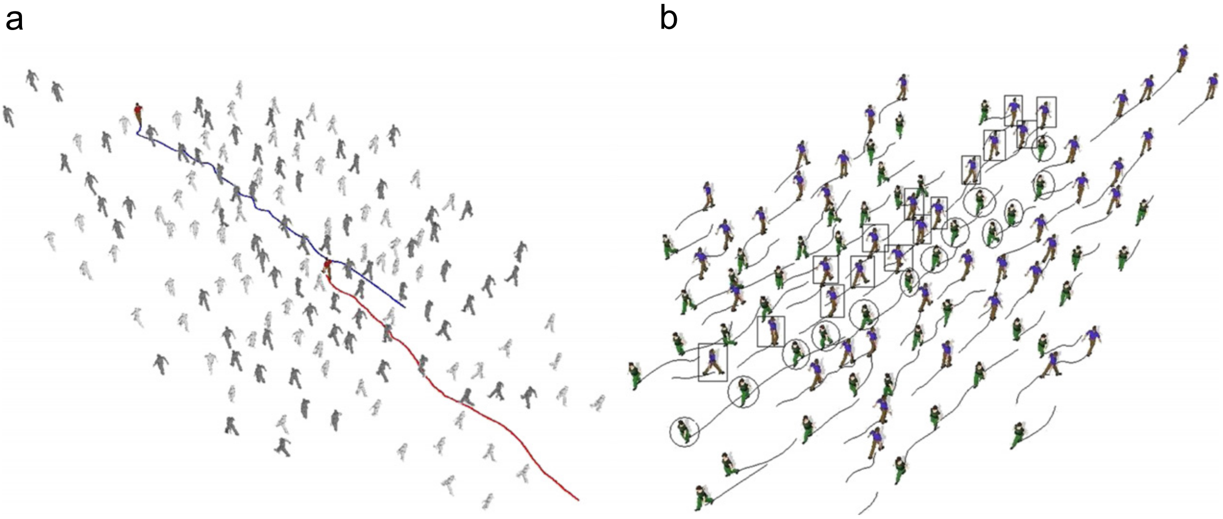

Fig. 3. (a) Trajectory smoothness of a leading agent (blue) and an agent in the middle of the group (red). (b) Formation of lanes (indicated by circles and squares) in two groups of 50 people moving in opposite directions. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

standard deviation (vertical line segments) of the angular variation of the agent's direction for several densities of markers. As it can be observed, the average angular variation (as well as the standard deviation) decreases rapidly for marker densities increasing from 7.5 to 15 per m^2. The decrease is slower for marker densities exceeding 15 per m^2.

We also analyzed the relation between the density of markers and simulation speed. The results were obtained for markers distributed over a square of dimension $80 \times 80\ m^2$ at four different densities. As shown in Fig. 1(b), the simulation speed decreases with the number of markers, as expected.

Based on the results shown in Fig. 1, we used 15 markers per m^2 in all subsequent experiments. This value represents a compromise between computational time (simulation of 800 agents can be performed in real-time, 30 frames per second) and trajectory smoothness (increases in marker density above 15 markers per m^2 do not reduce the angular variation significantly).

It is interesting to notice that a continuum distribution of markers into Voronoi polygons could be easily formulated, just replacing the summations in Eqs. (2) and (3) by integrals. However, such integrals would have to be solved numerically, which would lead to the discrete formulation proposed in this work (and higher densities of markers would correspond to a more accurate numerical solution).

4.2. *The shape of trajectories*

The smoothness of the trajectories depends not only on the density of markers but also on the density of agents. In fact, when two groups of people move through the same space in opposite directions, people at the front of their group have to change direction to a larger extent than people who walk behind, to avoid frontal collisions with the members of the other group [9]. This behavior is consistent with the least effort hypothesis, applied to each individual agent. To verify whether the same behavior emerges in our simulations, we considered two groups of agents moving in opposite directions. We then selected two agents from the same group: one at front and the other in the middle of the group, marked as blue and red trajectories, respectively, in Fig. 3(a). Visually, it can be observed that the red trajectory is smoother. Such smoothness is also evaluated quantitatively by computing the average angular change (in absolute value) of both agents. The agent at the front presented an angular change of 19.19°, with standard deviation 13.69° per simulation step. The agent in the middle of the crowd presented and average angular variation of 15.11°, with standard deviation 12.99°. For the sake of comparison, an isolated agent moving in the same field of markers would change its direction with average 3.86° and standard deviation 4.64°. Thus, as expected, a follower in a crowd changes its direction to a smaller extent than a leading agent, but to a larger extent than an isolated agent.

4.2.1. *The emergence of lanes*

Minimization of effort leads to a spontaneous formation of lanes, or chains of people who walk behind each other (cf. Section 2.1). Such lanes also readily emerge in our simulations, as it can be visually observed in Fig. 3(b). Once again, these results were obtained for two groups of agents moving in opposite directions.

4.2.2. *Collision avoidance*

A very important behavior is collision avoidance within the crowd. While our algorithm guarantees collision-free motion (Appendix A.1), it is nevertheless interesting to evaluate it in the simulations. To this end, we considered four groups, each with 50 agents, which originated at the corners of a square scene and moved toward the opposite corners. These groups create a dense crowd of people moving in different directions near the center of the scene. Fig. 4 shows four frames from the resulting simulation. While the set of markers allocated to each agent decreased near the center, these sets do not interpenetrate. For each agent, the next step will thus be collision free.

4.2.3. *The speed reduction effect*

As the density of crowd increases, the number of markers associated with each agent decrease, suggesting that the agents' speed is reduced as a function of crowd density. This reduction is the essence of the "speed reduction effect". To analyze its emergence in our simulations, we performed a series of experiments with different groups and numbers of agents now moving in a corridor of dimensions $10\ m \times 40\ m$. The maximum desired speed of all agents was fixed at $s_{\max} = 1.2\ m/s$. In the first two experiments, one group of 25 or 50 agents moved from one end of

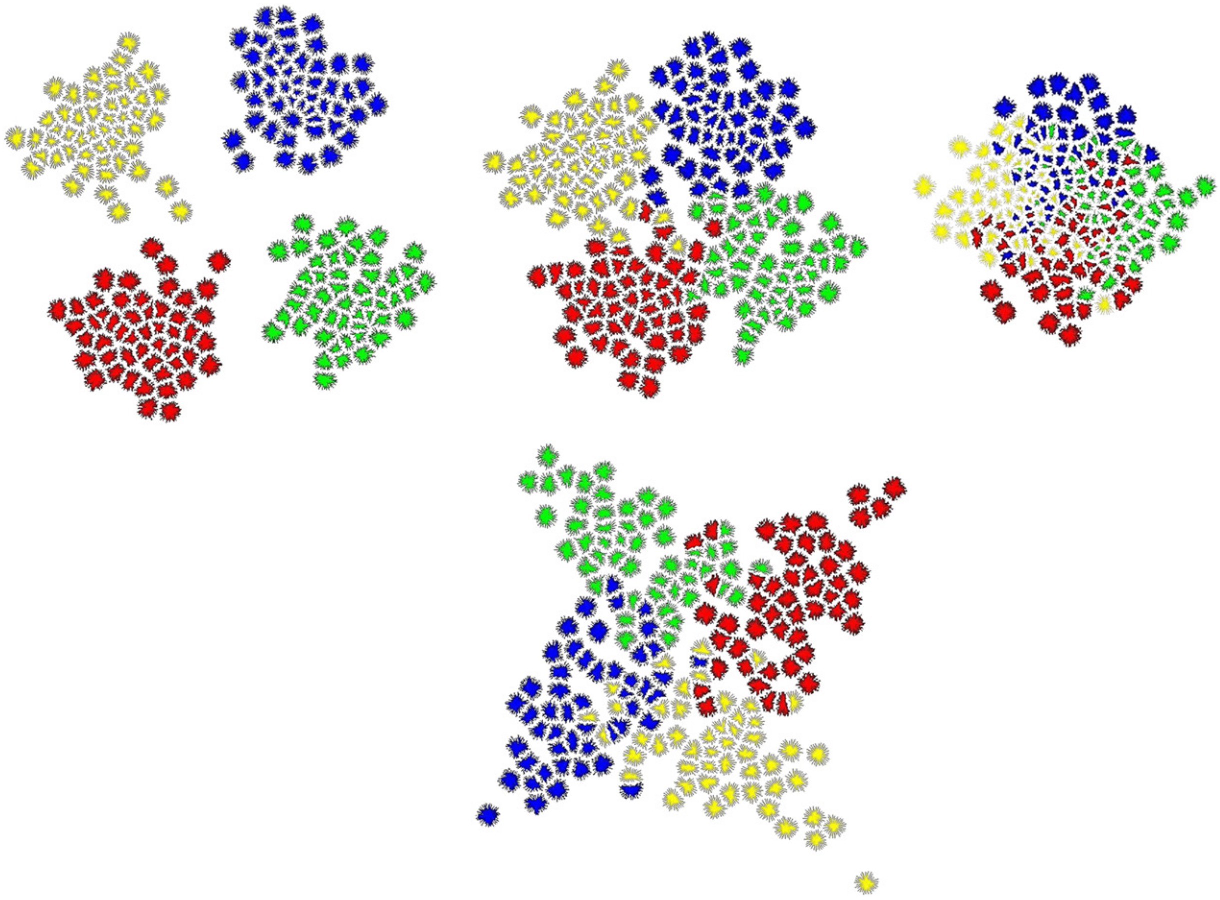

Fig. 4. 2D visualization of the collision-avoidance behavior. Four groups cross each other in the center of the scene.

Table 1
Speed reduction effect: reduction of realized speed as a function of the number of agents. Maximum agent speed in each case is $s_{\max} = 1.2$ m/s.

Agents (groups)	25 (1)	50 (1)	50 (2)	100 (2)	200 (2)	400 (2)	800 (2)
Mean realized speed (m/s)	1.19	1.19	1.17	1.16	1.14	1.11	1.09
Standard deviation	0.0006	0.0006	0.0021	0.0045	0.0096	0.0206	0.0319

the corridor to the other. In the remaining experiments, two groups of 25, 50, 100, 200 and 400 agents moved in opposite directions. Each experiment was repeated 20 times for different randomized configurations of marker points and initial positions of the agents. The average speeds of agents realized in these experiments are shown in Table 1. As expected, in the absence of flow in the opposite direction (the first two experiments), the realized speed is almost equal to the maximum speed allowed for the agents (1.2 m/s). In the simulations with groups of agents moving in opposite directions, the realized speeds were progressively smaller as the number of agents increased.

These results can be expressed in terms of the global density of agents. For instance, in the last experiment (last column in Table 1) there are on average 2 agents per m^2 (800 agents/400 m^2). However, global density is not a very informative measure of crowding, since it may vary spatially. In our simulations, the groups crossed each other near the center of the corridor, forming a high-density area there, while other areas were relatively empty. To take these differences into account, we divided the corridor into cells of dimensions $1\,m \times 1\,m$, and computed the number and mean velocity of agents in each cell. The results are shown in Fig. 5, which compares the distribution of speeds generated using our method (here labeled BioCrowds) with the average speeds reported for

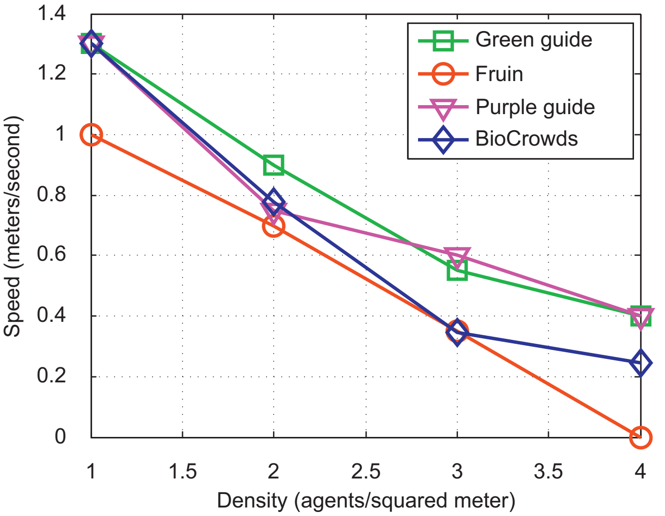

Fig. 5. Mean agent speed as a function of local crowd density. Plots labeled green guide, fruin, and purple guide represent measured data from real life, while plot labeled BioCrowds describes emergent results of our method. In these simulations, we assumed that the maximum velocity of agents was 1.2 m/s. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

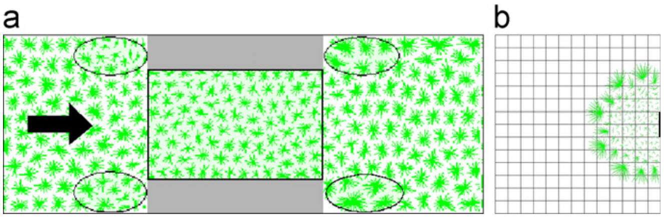

Fig. 6. (a) Bottleneck effect: ellipses highlight the regions where agents stop due to the environment. (b) Arc formation: agents have the same goal (e.g. location of a door—black square) and stop forming an arc.

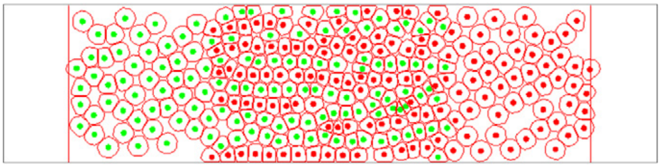

Fig. 7. The *Convex Hull* of an agent is the red polygon that circumscribes it. In this simulation, we used 60 markers/m^2. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

various densities of real crowds [7].[3,4] As it can be observed, the emergent speeds of crowds simulated using our method are consistent with the measured data.

4.2.4. The stopping effects

Virtual agents move when they have available markers in their perception fields and personal spaces. It is the essence of competition for space strategy in which agents fight for space and, then, move accordingly. However, when there are no markers available, stopping effects can happen, as it can also happen in real life. Two stopping effects are illustrated in this paper: (i) Bottleneck effect and (ii) Arc formation.

The first one describes the increase of density (and reduction of speed) that happens in environment presenting bottleneck due to walls (see Fig. 6(a)). In this figure, the ellipses on the left indicate bottleneck regions that present a higher density of virtual agents (4 persons/m^2), lower speed (average 0.31 m/s, standard deviation 0.056), and a smaller amount of available markers. The ellipses on the right indicate the corresponding regions after the bottleneck, with a lower density of agents (2 persons/m^2), higher speed (average 1.18 m/s, standard deviation 0.03) and more available markers. The middle of the corridor (rectangular region) presents intermediate results, with an average density of three persons/m^2, mean speed 0.99 m/s and standard deviation 0.004.

The second behavior was firstly proposed by Helbing et al. [17], and it describes the phenomena that occur when people stop due to an exit door, including stopping effect and also the emerging geometrical arc formation (see Fig. 6(b)).

4.2.5. Interactive crowd control

The motion of agents can be controlled by interactively spraying or erasing markers. Fig. 8(a) shows a screenshot of our prototype system that implements such interaction. Agents tend to follow paths with higher density of markers, so that local control can be achieved by increasing the number of markers along preferred paths. When markers are removed, agents immediately adjust their paths as shown in Fig. 8(b).

4.3. Crowd models: a comparative analysis

This section presents a comparative analysis of the proposed approach and some existing crowd simulation models [17,11,21,22,12]. Table 2 presents the computational complexity, performance and hardware related to such models. It is important to point out that the presented computational cost consider only the simulation process, ignoring the rendering step. Regarding to our BioCrowds model, frame rates refer to those obtained for a density of 15 markers/m^2, as shown in Fig. 1(b).

According to Table 2, BioCrowds presents a computational performance comparable to the other models. It is also interesting to note that the results of BioCrowds were obtained using a monothread implementation, whereas, for example, the model presented by Berg [22] used 16 processing cores to parallelize their code.

Table 3 presents a qualitative analysis regarding the main emergent behaviors produced by the analyzed models. The symbols indicate whether the behavior occurs (✓), does not occur (×) or could not be verified (?), through the analysis of the specified literature and related videos. It is possible to verify that BioCrowds model presents the main expected behaviors for crowd simulation.

Another interesting effect provided in BioCrowds is the stopping effect. In other models [17,11,21,22], the agents present low-amplitude oscillations when they have no space to move. It happens mainly in force-fields models due to the balance of forces. In our model, since the available markers for each agent do not change, agents may really stop, as illustrated in Fig. 6(b). On the other hand, if the application requires an oscillation in the crowd motion (e.g., impatient people), we can easily produce it by including a random parameter in Eq. (2). The resulting effect is that for each frame the motion direction should be different, liberating and taking into account other markers.

4.4. Infinitesimal versus finite agents in collision-free simulations

As discussed in Appendix A, the motion of the agents is collision free for infinitesimal agents, but agents occupying a finite space may interpenetrate. A possible solution for collision-free motion for finite-area agents as well is presented in Appendix A.3, and this section shows a brief comparison between both versions (infinitesimal versus finite-area agents).

As described in Appendix A.3, the body of a finite-sized agent is approximated by a circular region, which radius r should be based on information of anthropomorphic sizes of populations of the world. Still [9] reported that the average width between the shoulders is around 0.4558 m, leading to our choice of $r = 0.2279$ m for the radius r.

The scenario for this simulation is a corridor of dimensions 10 m × 40 m (the same simulation scenario used in Section 4.2.3), populated with two groups of 200 agents moving in opposite directions. The value for the agent maximum speed and the radius of the perception region will remain the same as the other experiments, i.e., $\mathbf{s}_{\max} = 1.2$ m/s and $R = 1.25$ m.

Running the finite-area version of the simulator with a density of 15 markers/m^2 yields an average speed of 0.47 m/s for the agents, which is significantly lower than the analogous simulation with infinitesimal agents shown in Table 1, sixth column. In fact, such difference is explained by the fact that we approximate the Voronoi polygons around each agent by the convex hull of its markers, and a density of 15 markers/m^2 may lead to very irregular polygons. To alleviate this problem, we have increased

[3] GreenGuide: Department of National Heritage, *Guide to safety at sports grounds (The green guide)*, fourth ed. London, UK: HMSO, 1997.

[4] PurpleGuide: Health and Safety Executive, *Guide to health, safety and welfare at pop concerts and similar events (The purple guide)*, first ed. London, UK: HMSO, 1993.

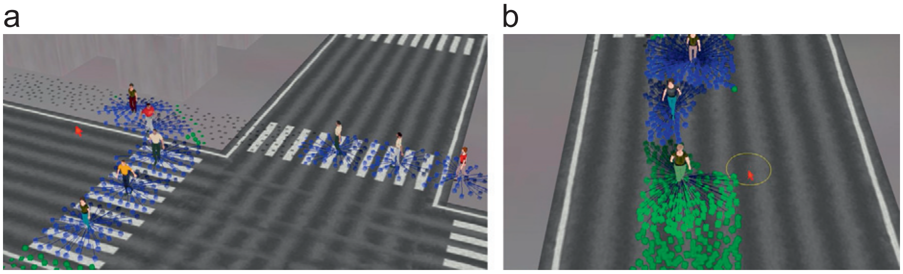

Fig. 8. Snapshots of the proposed interface, illustrating (a) the "sprayed" markers (green dots) on the floor, and (b) the possibility of erasing markers (the yellow circle represents the marker eraser, and it has been used to narrow down the region where the agents can walk). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 2
Computational complexity and performance of analyzed crowd models, where n denotes the number of agents. Additionally, m is the number of markers for BioCrowds.

Model	Hardware	Complexity/performance
Helbing et al. [17]	Not specified	$\mathcal{O}(n^2)$ / _
Treuille et al. [11]	Intel Pentium 3.4 GHz Processor	_ / 10 K agents (2 to 5 FPS)
Pelechano et al. [21]	Intel Xeon 2.99 GHz Processor	_ / 1.8 K agents (25 FPS)
Berg et al. [22]	Intel Xeon 2.93 GHz Processor	_ / 2.5 K agents (20 FPS) to 20 K agents (2 FPS)
Narain et al. [12]	Intel Core i7-965 3.2 GHz Processor	_ / 2 K agents (62 FPS) to 100 K agents (4 FPS)
BioCrowds	Intel Core 2 Duo 2.2 GHz Mobile Processor	$\mathcal{O}(nm)$ / 800 agents (30 FPS) to 13 K agents (6 FPS)

Table 3
Emergent behaviors presented by analyzed crowd models.

Model	Emergent crowd behaviors			
	Lanes	Bottleneck effect	Arc formation	Guaranteed free-of-collision
Helbing et al. [17]	✓	✓	✓	✓—However, can generate low-amplitude oscillations
Treuille et al. [11]	✓	✓	?	×—Proposes a final test in case of collision situation
Pelechano et al. [21]	✓	✓	✓	✓—However, is based on a series of specific proximity rules
Berg et al. [22]	✓	✓	✓	×—Authors mention that collisions can occur
Narain et al. [12]	✓	✓	✓	×—Authors mention that collisions can occur
BioCrowds	✓	✓	✓	✓—Free-of-collision behaviors are inherent to the model, and proved mathematically

the density to 60 markers/m^2, achieving an average speed of 1.08 m/s, with standard deviation of 0.06. As expected, the average speed is a little lower than the infinitesimal version for the same scenario, and the standard deviation a little larger, since infinitesimal agents may be as close to each other as possible, whereas the minimum distance between finite-sized agents is limited by their bodies. For the sake of illustration, a screenshot of the simulation illustrating the convex hull around each agent is shown in Fig. 7.

5. Final considerations

In this work, we proposed a new model for crowd simulation. Important aspects of people's motion in a crowd (collision avoidance, goal seeking, relationship between density/speed and smoothness of trajectories on the local density of the crowd, and lane formation) are some of emergent properties of the model. The model also provides a convenient method for interactively controlling the movements of crowds.

Methodologically, the key innovation is the simple way in which the agents monitor their environment, by "observing" free space, rather than each other directly. This space is represented using a set of marker points, which leads to a simple yet computationally effective implementation of the competition for space. To this end, each agent captures the markers that are within its perceptive field and that are closer to it than to any other agent. These markers locally guide the motion of the agents. Global goal-seeking is modeled by biasing the influence of the captured marker points according to their agreement with each agent's direction to its goal, which can be assigned to individual agents or groups.

The use of markers also provides a conceptually clean metaphor for interacting with the simulated crowds, which can be directed by spraying or erasing the markers. To find the set of markers that lie within the personal space of each agent, the Voronoi tessellation could be computed. However, since the markers are fixed, our implementation consisted of analyzing each marker, and finding which agent is the closest one (a grid structure is also used to reduce the search space, so that exhaustive search for all agents in not necessary).

Regarding validation, it is still a challenge to evaluate crowd simulation models qualitatively in terms of attained realism. In fact, there are no objective metrics that quantify the similarity between two simulations, or to evaluate quantitatively how realistic a given result is. Nevertheless, in terms of the "speed reduction effect", the decay of speed in terms of crowd density achieved by virtual humans in the proposed approach was coherent with measured data in real life, as shown in Fig. 5.

Finally, it is important to point out that the proposed approach generates guaranteed collision-free motion, which is a challenging problem in crowd simulation [3]. The formulation considering infinitesimal agents produces few interpenetrations when displaying agents as finite-area regions, and the extension to provide collision-free motion for finite-sized agents presented in Appendix A.3 and Section 4.4 can be used to avoid interpenetrations (at the cost of increasing the complexity of the algorithm).

As future work, we plan to extend the proposed biologically motivated model to cope with groups, and further dedicate to obtaining quantitative metrics to evaluate crowd simulation results. We also plan to perform a more thorough evaluation for the proposed simulation algorithm using finite-area agents.

Acknowledgment

Authors Claudio Jung and Soraia Musse would like to thank Brazilian Agency CNPq for partially funding this work. Authors Alessandro Bicho and Léo Pini Magalhães would like to thank the Brazilian Agency CAPES for partially funding this work. Authors would like to thank Prof. Przemyslaw Prusinkiewicz and M.Sc. Adam Runions for discussions about biological model that inspired the proposal of this work.

Appendix A

A.1. Collision avoidance

Before proving the collision free nature of the proposed method it is necessary to introduce the properties of Voronoi diagrams and convex sets used in the proof. First, given a set of agents with associated positions, dividing the space into the regions of points that are closer to one agent than any other is exactly the Voronoi tessellation of the point set containing the positions of all agents [26]. Second, in such a partition, the region associated with each agent is a convex polygon [26]. Finally, a convex polygon is a special type of convex set, which can be defined as follows [27]:

Definition 1. Let C be a set, then C is convex provided that for any set of points $\{\boldsymbol{x}_1,\ldots,\boldsymbol{x}_n\} \in C$, and any $w_1,\ldots,w_n \in \mathbb{R}$ such that $w_1+w_2+\cdots+w_n=1$, $0 \le w_i \le 1$, it follows that $\sum_{i=1}^{n} w_i\boldsymbol{x}_i$ is also in C.

In the proposed formulation, it is important to notice that the set of markers related to an agent lie within the intersection of its circular perception field and its personal space, defined as the corresponding Voronoi polygon. Since both regions are convex, their intersection remains convex (so that all markers lie within a convex set).

The displacement vector $\boldsymbol{v}$ given by Eq. (5) can be either $\boldsymbol{m}$ (if $s_{\max} > \|\boldsymbol{m}\|$) or $s_{\max}\boldsymbol{m}/\|\boldsymbol{m}\|$ (if $s_{\max} \le \|\boldsymbol{m}\|$). In the first case, $\boldsymbol{m}$ is computed using a weighted average of the displacement vectors $\boldsymbol{a}_k - \boldsymbol{x}$ that satisfy the conditions of Definition 1 (as the weights sum to 1). Thus $\boldsymbol{v}=\boldsymbol{m}$ must be a point in the translated Voronoi polygon of agent I. In the second case, as $s_{\max}/\|\boldsymbol{m}\| \le 1$ (as $s_{\max} \le \|\boldsymbol{m}\|$) it follows that $\boldsymbol{v}=s_{\max}\boldsymbol{m}/\|\boldsymbol{m}\|$ is a point on the line segment connecting the origin and $\boldsymbol{m}$. As the origin and $\boldsymbol{m}$ are both in I's translated Voronoi polygon it follows that all points on the line segment connecting the points are also in the polygon (by Definition 1, as the region is convex). Thus $\boldsymbol{v}$ must be a point within the translated Voronoi polygon. Hence, in both cases $\boldsymbol{v}$ must belong to the Voronoi polygon of agent I (translated to the origin). Finally, as the Voronoi tessellation provides a disjoint partition of space, the trajectory of each agent is guaranteed to be collision-free.

A.2. Relation between the perception field and the displacement vector

In this section, we show the relation between the radius R of the perception field and the expected value for the magnitude of the motion vector $\|\boldsymbol{m}\|$ given by Eq. (2).

Let us consider a given agent, with a goal such that $\boldsymbol{g}-\boldsymbol{x}=(1,0)$ (without loss of generality, we have chosen the unit vector in the x direction), and a circular perceptive field with radius R. Assuming that the markers are continuously distributed around the agent according to a uniform distribution, the expected value for the direction vector $\boldsymbol{m}$ is

$$E[\boldsymbol{m}] = \frac{\int_{\Omega} \frac{1}{1+\|\boldsymbol{x}\|}\left(1+\frac{\langle \boldsymbol{g},\boldsymbol{x}\rangle}{\|\boldsymbol{x}\|}\right)\boldsymbol{x}\,d\boldsymbol{x}}{\int_{\Omega} \frac{1}{1+\|\boldsymbol{x}\|}\left(1+\frac{\langle \boldsymbol{g},\boldsymbol{x}\rangle}{\|\boldsymbol{x}\|}\right)d\boldsymbol{x}}, \tag{7}$$

where Ω is the circular perceptive field. Using polar coordinates (r,θ), a direct computation leads to

$$E[\boldsymbol{m}] = \left(\frac{1}{4}\frac{R^2-2R+2\ln(1+R)}{R-\ln(1+R)},0\right), \tag{8}$$

so that $\boldsymbol{m}$ points to the goal vector $\boldsymbol{g}-\boldsymbol{x}$. Furthermore, its magnitude $\|\boldsymbol{m}\|$ is given by $\frac{1}{4}(R^2-2R+2\ln(1+R))/(R-\ln(1+R))$. To ensure that the radius R of the perceptive field is large enough to allow agents to move $s_{\max}$ during a time step we must choose R and $s_{\max}$ such that the condition $s_{\max} < \frac{1}{4}(R^2-2R+2\ln(1+R))/(R-\ln(1+R))$ is satisfied. It should be noticed that the values for $s_{\max}$ and R defined in Section 4 satisfy this relation.

A.3. Elimination of collision for finite-sized agents

As discussed before, the motion of the agents is collision free, irrespective of the density of the crowd or markers, since their personal spaces are always non-intersecting. However, this observation only applies to infinitesimally small agents, and agents occupying a finite space may theoretically collide (if they approach from opposite sides the same point on an edge of a Voronoi polygon).

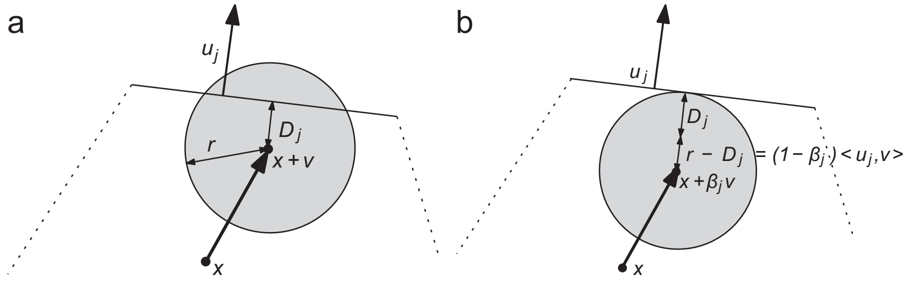

Fig. 9. Collision avoidance for non-punctual agents. (a) Situation of possible collision. (b) Enforcement of collision avoidance by using a speed reduction factor.

Let us consider that a virtual agent is represented in two dimensions by a circle with radius r. If the whole circle lies within the personal space of the agent in the subsequent iteration, then collision-free motion is also guaranteed for non-punctual agents. To accomplish that, we include a reduction factor $0<\beta\leq 1$ in Eq. (6), leading to

$$\boldsymbol{x}(t+1)=\boldsymbol{x}(t)+\beta\boldsymbol{v}. \tag{9}$$

To determine β, we first compute the distances D_j from $\boldsymbol{x}(t)+\boldsymbol{v}$ to each edge e_j of the Voronoi polygon related to the agent. If $D_j>r$, the agent lies entirely within its personal space, and there is no chance of collision. Otherwise, its body invades the personal space of a neighboring agent, and collision (at that particular Voronoi edge) can be avoided by selecting an adequate reduction factor β_j. More precisely, the reduction factor that leads the agent the closest to e_j without crossing it is given by

$$\beta_j=\begin{cases}1-\dfrac{r-D_j}{\langle \boldsymbol{u}_j,\boldsymbol{v}\rangle} & \text{if } D_j\leq r,\\ 1 & \text{if } D_j>r,\end{cases} \tag{10}$$

where $\boldsymbol{u}_j$ is a unit vector normal to edge e_j pointing outwards from the polygon, as shown in Fig. 9. To avoid crossing any edge e_j of its personal space, we select $\beta=\min \beta_j$ as the reduction factor for the agent, as required for Eq. (9). In practice, Voronoi edges for which $\langle \boldsymbol{u}_j,\boldsymbol{v}\rangle$ is negative are not considered, since the agent is moving away from them. Additionally, if we want to enforce a minimum distance ϵ between two agents, we can simply replace r with $r+\epsilon/2$ when computing β_j in Eq. (10). In terms of implementation, if we know all the markers assigned to a certain agent, its corresponding Voronoi polygon may be approximated by the Convex Hull of such markers [28].

References

[1] Thalmann D, Musse SR. Crowd simulation. London, UK: Springer-Verlag London Ltd.; 2007.

[2] Ulicny B, de Heras Ciechomski P, Musse SR, Thalmann D. State-of-the-art: real-time crowd simulation. In: EG 2006 course on populating virtual environments with crowds, eurographics, 2006.

[3] Patil S, van den Berg J, Curtis S, Lin MC, Manocha D. Directing crowd simulations using navigation fields. IEEE Trans Visual Comput Graph 2011;17(2):244–54.

[4] Runions A, Fuhrer M, Lane B, Federl P, Rolland-Lagan A-G, Prusinkiewicz P. Modeling and visualization of leaf venation patterns. ACM Trans Graph 2005;24(3):702–11.

[5] Runions A, Lane B, Prusinkiewicz P. Modeling trees with a space colonization algorithm. In: Ebert SMD, editor. Proceedings of the eurographics workshop on natural phenomena. Aire-la-Ville, Switzerland: Eurographics Association; 2007. p. 63–70.

[6] Henderson LF. The statistics of crowd fluids. Nature 1971;229(5284):381–3.

[7] Fruin JJ. Pedestrian and planning design. New York, NY, USA: Metropolitan Association of Urban Designers and Environmental Planners, Elevator World Inc.; 1971.

[8] Helbing D. Pedestrian dynamics and trail formation. In: Schreckenberg M, Wolf DE, editors. Traffic and granular flow '97. Singapore: Springer-Verlag; 1997. p. 21–36.

[9] Still GK. Crowd dynamics. PhD thesis. University of Warwick, Coventry, UK; August 2000.

[10] Hall ET. The hidden dimension. Garden City, NY, USA: Doubleday; 1966.

[11] Treuille A, Cooper S, Popović Z. Continuum crowds. ACM Trans Graph 2006;25(3):1160–8.

[12] Narain R, Golas A, Curtis S, Lin MC. Aggregate dynamics for dense crowd simulation. ACM Trans Graph 2009;28(5):122:1–8.

[13] Jiang H, Xu W, Mao T, Li C, Xia S, Wang Z. Continuum crowd simulation in complex environments. Comput Graph 2010;34:537–44.

[14] Reynolds CW. Flocks, herds and schools: a distributed behavioral model. In: Proceedings of the annual conference on computer graphics and interactive techniques (SIGGRAPH '87). New York, NY, USA: ACM Press; 1987. p. 25–34.

[15] Tu X, Terzopoulos D. Artificial fishes: physics, locomotion, perception, behavior. In: Proceedings of the annual conference on computer graphics and interactive techniques (SIGGRAPH '94). New York, NY, USA: ACM Press; 1994. p. 43–50.

[16] Helbing D, Molnar P. Self-organization phenomena in pedestrian crowds. In: Self-organization of complex structures: from individual to collective dynamics, Gordon and Breach, London, UK; 1997. p. 569–77.

[17] Helbing D, Farkas I, Vicsek T. Simulating dynamical features of escape panic. Nature 2000;407(6803):487–90.

[18] Braun A, Musse SR, de Oliveira LPL, Bodmann BEJ. Modeling individual behaviors in crowd simulation. In: Proceedings of international conference on computer animation and social agents 2003. New Brunswick, USA: IEEE Computer Society; 2003. p. 143–8.

[19] Ondřej J, Pettré J, Olivier A-H, Donikian S. A synthetic-vision based steering approach for crowd simulation. ACM Trans Graph 2010;29(4):123:1–9.

[20] Koh WL, Zhou S. Modeling and simulation of pedestrian behaviors in crowded places. ACM Trans Model Comput Simul 2011;21(3):20:1–23.

[21] Pelechano N, Allbeck JM, Badler NI. Controlling individual agents in high-density crowd simulation. In: Proceedings of ACM SIGGRAPH/eurographics symposium on computer animation (SCA '07). Aire-la-Ville, Switzerland: Eurographics Association; 2007. p. 99–108.

[22] van den Berg J, Patil S, Sewall J, Manocha D, Lin M. Interactive navigation of multiple agents in crowded environments. In: SI3D '08: proceedings of the 2008 symposium on interactive 3D graphics and games. New York, NY, USA: ACM; 2008. p. 139–47.

[23] Rudomín I, Hernández MEB. Fragment shaders for agent animation using finite state machines. Simulation modelling practice and theory. Simul Model Pract Theor J 2005;13(8):741–51.

[24] Reeves WT. Particle systems—a technique for modeling a class of fuzzy objects. ACM Trans Graph 1983;2(2):91–108.

[25] Cook RL. Stochastic sampling in computer graphics. ACM Trans Graph 1986;5(1):51–72.

[26] Aurenhammer F. Voronoi diagrams—a survey of a fundamental geometric data structure. ACM Comput Surv 1991;23(3):345–405.

[27] Rockafellar RT. Convex analysis (Princeton landmarks in mathematics and physics). Princeton University Press; 1996.

[28] O'Rourke J. Computational geometry in C. Second ed. Cambridge, UK: Cambridge University Press; 1998.