CGI2022 manuscript No.
(will be inserted by the editor)

Velocity-based Dynamic Crowd Simulation by Data-Driven Optimization

Pengfei Liu* · **Qianwen Chao*** · **Henwei Huang** · **Qiongyan Wang** · **Zhongyuan Zhao** · **Qi Peng** · **Milo K. Yip** · **Elvis S. Liu** · **Xiaogang Jin**✉

Abstract A crowd simulator that generates realistic crowds with various movement patterns and environmental adaptability is urgently desired but underdeveloped for the applications of video games, urban visualization, autonomous driving, and robot navigation test. In this work, we present a novel velocity-based framework based on data-driven optimization to build dynamic crowd simulation that allows interactive control of global navigation, local collision avoidance, and group formation. An agent's adaptive decision-making regarding its goals and dynamic local environment is formulated as an optimization problem which is solved by finding an optimal velocity from the real-world crowd velocity dataset. Each component that affects an agent's movement is integrated into a velocity-based crowd energy metric to measure the similarity between the agent's required simulated velocity and a given velocity sample. The proposed model can simulate thousands of agents at interactive rates. In addition, the framework is general and scalable to be integrated with various crowd simulation methods to meet the requirements of various kinds of robot navigation test. We validate our approach through simulation experiments in robot navigation scenarios, as well as comparisons to real-world crowd data and popular crowd simulation methods.

Keywords Crowd Animation · Data Driven · Motion Control · Optimization

Xiaogang Jin, jin@cad.zju.edu.cn

*Indicates equal contribution
Pengfei Liu, Zhongyuan Zhao, Qi Peng, and Xiaogang Jin
State Key Lab of CAD&CG, Zhejiang University, Hangzhou 310058, China

Qianwen Chao, Henwei Huang
Harvard Medical School, Boston, the United States

Qiongyan Wang
Xidian University, Xian, China

Milo K. Yip, Elvis S. Liu
MoreFun Studios, Tencent, Shenzhen, China

1 Introduction

Incorporating realistic crowds into virtual environments has received increasing attention from a variety of research communities in recent years, including, but not limited to computer graphics, visual reality, urban planning, emergency simulation, virtual robotic navigation test, and behavioral science. Human crowds exhibit highly complex behaviors driven by adaptive individual decisions of agents with respect to their goals, environmental obstacles, and other nearby agents. Simulating such unconsciously self-organized crowd movement in a dynamically changing environment is highly desired.

A number of crowd simulation methods, including macroscopic [35,25], and microscopic approaches [11, 37,34], have been developed to model and simulate crowd dynamics. However, these methods focus on modeling the individual behaviors without referring to real-world crowd trajectory data, which results in simulations that may not realistically resemble real-world crowd scenarios.

To date, several data-driven approaches have been proposed to enhance the realism of crowd movements in simulations. These methods [17,12,9,20,21,16] mainly train models for specific scenarios and apply them to similar scenarios, usually resulting in poor scenario adaptability. To address this problem, deep learning-based methods [18,36] have been proposed for scene-agnostic crowd simulation. However, convincing simulation results require large amounts of real-world data

Fig. 1 Examples of various virtual crowds with different movement patterns simulated by our framework: walking on zebra crossing (the first), aggregation and queuing at the building entrance (the second) , indoor evacuation (the third), and army formation transform (the fourth).

from multiple scenarios, and model training is computationally expensive. More recently, Ren et al.[32] proposed a data-driven optimization method to generate plausible behaviors for heterogeneous multi-agent scenarios. Since the model simulates the general behavior of different types of individuals by using basic collision avoidance and simple global navigation mechanisms, it cannot be applied to crowd simulations in complex dynamic environments and to simulate versatile crowd movements, such as indoor evacuation, gathering, queuing, zoo/museum tours, and square walks.

In order to generate realistic crowds with various movement patterns and environmental adaptability, the movement of virtual pedestrians in a crowded environment needs to consider several constraints, including global navigation towards the target, local collision avoidance with surrounding pedestrians and static obstacles, and social interactions that lead to self-organization and emergent phenomena in crowds. In this paper, we present a novel velocity-based crowd simulation framework through data-driven optimization by taking all the constraints for making decision into account.

Specifically, the decision-making process of each agent is formulated as an optimization problem, which can be solved by selecting a velocity from the real-world crowd dataset that tends to minimize a newly defined *crowd energy metric*. Several essential energy terms are developed in the crowd energy formula. These terms consider the agent's movement continuity, global navigation, collision avoidance, and group formation control simultaneously. Each energy term is defined as a factor related to a local optimal velocity, and the agent's decision-making is to weigh these local optimal velocities. In order to make individual behavior in the simulation as realistic as that in real crowd, we calibrate the model parameters using real-world crowd data. Fig. 1 shows several crowd examples generated by our approach, including walking on zebra crossing, queuing and aggregation in front of the building gate, indoor evacuation, and army formation transform.

The main contributions of this work are as follows:

- A novel, unified, and calibrated approach based on data-driven optimization to simulate versatile crowd movements with environmental adaptability.
- A velocity-based crowd energy metric for similarity measurement, by considering movement continuity, global navigation, local collision avoidance, and group formation control.
- A group formation control mechanism based on mean-shift clustering to guide the self-organized crowd movement in a dynamically changing environment.

2 Related Work

2.1 Crowd Simulation Models

In crowd simulation, there are two kinds of widely used crowd control models according to the expressive level of simulation details: fluid-based macroscopic and agent-based microscopic methods. Macroscopic models [35, 25] use an analogy with fluid or gas dynamics to describe how crowd density and velocity change over time using partial differential equations. They are ignore detail of crowd, so these kind of methods are not suitable for low-density crowds. In contrast, the microscopic model treats each person in the crowd as an intelligent agent with its own properties and goals. Each agent makes a decision individually from its neighborhood information for every time-step. Researchers have developed a variety of microscopic control models, including force-based [10] and velocity-based [26, 8, 2] models.

In a force-based model, each agent receives virtual forces generated from the spatial or social relationship between the agent and its neighbors. Helbing et al. [11, 10] proposed the Social Force Model (SFM) for normal and panic situations. Pelechano et al. [29] proposed an individual control in dense environments. Karamouzas et al. [14] defined a time-to-collision dependent potential energy whose derivatives generate forces. Another stream of researchers proposed the velocity-based model, where each agent selects a velocity that

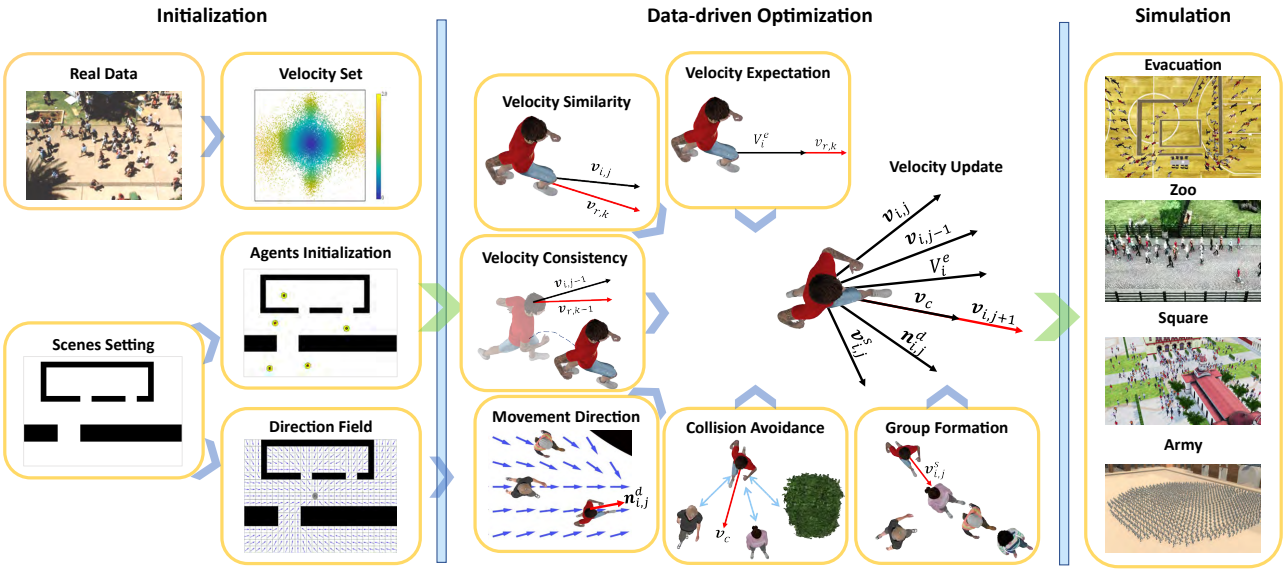


Fig. 2 The pipeline of our approach. In the initialization stage (left), we create a velocity dataset based on real data. For a virtual scene, we initialize the positions of agents of a crowd and calculate the direction field according to the scene geometry. In the simulation stage (right), we simulate crowds with various movement patterns in diverse environments. The decision-making of each agent is formulated as an optimization problem (see the data-driven optimization stage, middle), whose energy function considers velocity similarity, velocity consistency, velocity expectation, movement direction, collision avoidance, and group formation jointly. Our approach can control crowd patterns, such as queue, aggregation, and walking in tows in the same framework.

minimizes a given cost function [2]. The velocity-based model is usually less sensitive to parameter choice and more stable in a large time-step than the force-based model.

2.2 Data-Driven Multi-Agent Simulation

To enhance the visual realism of crowd simulation, there have been several attempts to introduce real captured crowd data into multi-agent simulation. These methods try to simulate virtual crowds by learning behavior patterns from real-world samples [17,12]. Seemingly natural crowd behaviors can be produced by directly coping the real-world trajectories or pre-computed patches [21,16,15]. Although these methods can generate crowd movements similar to those observed in crowd samples, they lack consideration of scene adaptability. Ren et al. [32] simulated the heterogeneous multi-agent systems by combining data-driven with physics-based simulation methods. Due to the model versatility for different agent types, the model only uses parameters to adjust crowd. It is difficult to simulate frequently changing crowds.

Recently, techniques based on neural networks have received increasing attention in the crowd simulation community. [18] presented an agent-based deep reinforcement learning approach for crowd navigation, which learns a single unified policy that can adapt all the scenarios. Generative Adversarial Networks can be used to learn the properties of real-world crowd datasets [1] and generate new trajectories with matching patterns. However, convincing simulation results rely on a large quantity of real-world crowd data.

2.3 Group simulation

A group in a crowd is defined as a subset of agents which desire to move together [33,13]. Musse et al. [23] simulated the emergent group behavior with the consideration of the relationship between groups, and later presented a hierarchical model to control groups with different degrees of autonomy [24]. The Social Force Model [11] was extended to simulate groups by adding several attractive forces [28] or including social interactions among people walking in groups [22]. The Velocity Obstacle approach can also be extended to model group behaviors in crowd simulation [9,33]. In addition, some common human behaviors, such as leader-follower behaviors [31] and following behaviors [19] were explored and simulated through experimental studies.

3 Method Overview

The pipeline of our approach is illustrated in Fig. 2. In the initialization stage, we preprocess different

types of real-world crowd datasets by uniformly converting them into velocity representations and dividing the velocity set into several subsets according to the magnitude of the velocity. We establish an underlying goal-directed direction field over the free space in an environment, which could be used for directing agents in a simulation. Each agent is initialized by randomly setting an initial position and randomly selecting an initial velocity from the velocity dataset. In each step of the simulation, we treat the motion decision-making or local navigation process of each agent as an optimization problem, which can be solved by selecting a velocity from the dataset that tends to minimize a newly defined *crowd energy metric*. The energy metric is defined based on the locomotion and dynamics rules of agents, including velocity continuity and consistency, velocity expectation, collision avoidance, movement direction control, and group formation in the surrounding environment for group formation control. The influence of each energy term is modeled as a local optimal velocity, and the final individual's decision-making is a trade-off among different local optimal velocities.

3.1 Initialization

Velocity Set Preparation: From the real-world crowd trajectories, we calculate the velocity of each agent in each frame and obtain the velocity set $V = \{v_{i,t}, v_{i,t-1}\}$, where $v_{i,t}$ is the velocity of an arbitrary agent i at frame t, and $v_{i,t-1}$ is the velocity in the previous frame. Considering that the velocity between two adjacent frames can only change within a small range due to the temporal continuity of human movement, we therefore sort the set V in ascending order according to velocity magnitude $|v_{i,t}|$, and divide it into several velocity subsets. In each search process, only the subset where $v_{i,t}$ is located and its adjacent subsets are searched, which can greatly improve the search efficiency of the algorithm.

In our implementation, the real-world crowd dataset for data-driven optimization is provided by the ETH [30] and UCY Pedestrian datasets [20], which contain more than 1,600 pedestrian trajectories in five scenes (ETH, Hotel, Univ, Zara1 and Zara2). We divide the velocity set V containing 61,995 individual velocities into 100 subsets, and in each search, the search range is 6 adjacent subsets.

Direction Field Computation: Our method computes the direction field for each distinct group of agents based on the static description of the environment and specified goal positions. These direction fields are smooth with no local minima, and used to guide agents to their corresponding goals. Similar to [27, 5], the computation of the direction field requires a discretization of free space in the environment. Here, we use regular grids (Fig. 3), in which each cell of the grid stores a vector representing the ideal moving direction, the distance from the current grid to the target, and the crowd density information in the current grid. Then, we use a variant of Dijkstra's algorithm to propagate costs throughout the grid, starting with zero costs associated with the goal position. At each instance of the computation, the path cost at a given cell is evaluated through a linear combination of the calculated path costs of two neighboring cells. In this way, keeping track of the direction of the optimal path taken at each grid cell yields a smooth navigation vector field over the entire free space in the environment. For more implementation details, we refer to [27]. It is worth noting that the direction field will not be recomputed at each time step of the simulation, but only when the goal position changes.

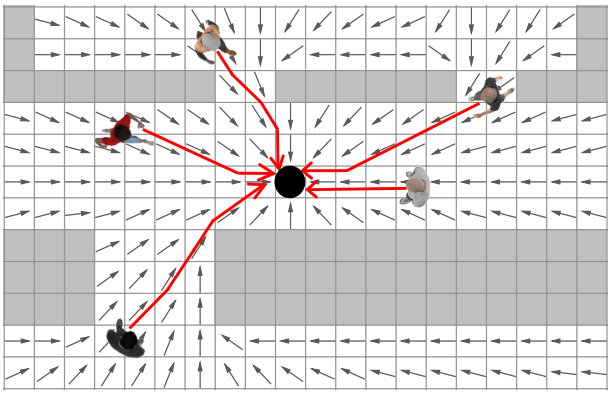


Fig. 3 Illustration of the direction field for the agent's global navigation. The black dot represents the agents' movement target and the gray grids denotes the static obstacles in the environment.

3.2 Data-driven Optimization

At each time step of the simulation, the velocity of an agent is updated by finding the velocity in the input velocity sample sets that is most similar to its state in the synthesized crowd. Formally, let $\mathbf{v}_{i,j}$ denote the velocity of agent i at frame j to be updated in simulation, and $\mathbf{v}_{r,k}$ denote the velocity candidate in dataset V, indicating the velocity of agent r at frame k, its crowd energy E is defined as follows:

$$E = w_v E_v + w_p E_p + w_e E_e + w_c E_c + w_d E_d + w_g E_g, \quad (1)$$

where the *velocity similarity term* E_v measures the current velocity similarity between the agent i and r, the *velocity consistency term* E_p measures the similarity between the two agents' velocities at their respective

previous frames, the *velocity expectation energy term* E_e measures the similarity between $\mathbf{v}_{r,k}$ and the desired velocity of the agent i, the *collision avoidance energy term* E_c is introduced to preserve the agent i's safe distance with its neighbors and static obstacles during movement, the *movement direction energy term* E_d measures the similarity between the direction of $\mathbf{v}_{r,k}$ and the expected movement direction of the agent i, which is obtained from the precomputed direction field, and the *group formation energy term* E_g is further introduced to describe the interaction of agent i's interactions with surrounding neighbors and generate various group movement patterns in crowds. w_v, w_p, w_e, w_c, w_d and w_g are normalized weight parameters of these energy terms.

4 Energy Term Calculation

4.1 Velocity-related energy terms

E_v, E_p, and E_e are three energy terms related to agent i's velocity, which are used to ensure the individual's smooth continuous movement and the expectation of moving at the desired velocity.

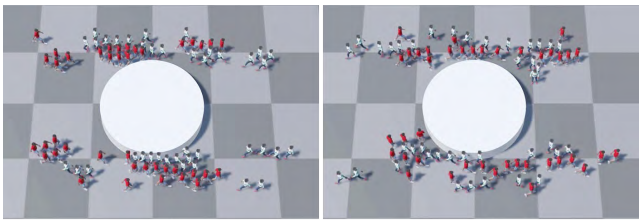
In the real-world crowd, humans cannot change their motion state frequently or suddenly within a time step. Inspired by this observation, in the simulation, we design the velocity similarity term E_v to represent the tendency of individual velocity changes within a reasonable range. This energy term measures the similarity between $\mathbf{v}_{i,j}$ and $\mathbf{v}_{r,k}$ in terms of movement direction and velocity magnitude, which is defined as:

$$E_v = w_{v1} \| \|\mathbf{v}_{i,j}\| - \|\mathbf{v}_{r,k}\| \|_2 + w_{v2} \left\| \mathbf{n}_{i,j} - \mathbf{n}_{r,k} \right\|_2 , \quad (2)$$

where $\left\| \mathbf{n}_{i,j} - \mathbf{n}_{r,k} \right\|_2$ is designed for direction similarity, and $\| \|\mathbf{v}_{i,j}\| - \|\mathbf{v}_{r,k}\| \|_2$ is designed for velocity magnitude similarity. $\mathbf{n}_{i,j}$ and $\mathbf{n}_{r,k}$ are respectively the unit vector representing the moving direction of agent i at frame j and that of the candidate r at frame k in the velocity dataset. w_{v1} and w_{v2} are weight parameters. It is noteworthy that measuring the similarity of velocity magnitude and direction separately with different weight parameters can expand the search range in the velocity set, thereby effectively avoiding individual movement concussion in the simulation.

In addition to keeping the individual's velocity change within a reasonable range, the individual's velocity change is also required to be continuous to ensure the continuity of movement and the smoothness of the trajectory in the simulation. As defined in Eq. 3, we take the individual's previous frame speed into account and introduce the velocity consistency term E_p:

$$E_p = \left\| \mathbf{v}_{i,j-1} - \mathbf{v}_{r,k-1} \right\|_2 , \quad (3)$$



(a) (b)

Fig. 4 Different crowd movement patterns generated by our approach combined with different collision avoidance algorithms ((a) SFM, (b) ORCA).

where $\mathbf{v}_{i,j-1}$ and $\mathbf{v}_{r,k-1}$ are the velocities of the agent i and r at the previous frame, respectively.

The velocity expectation energy term E_e describes individual i's motivation to move with an expected velocity V_i^e, which only measures the similarity of velocity magnitude:

$$E_e = \| V_i^e - \|\mathbf{v}_{r,k}\| \|_2 , \quad (4)$$

where V_i^e is a constant used as a personalized attribute of the agent i, defined during the initialization step.

4.2 Collision avoidance energy term

Collision avoidance is achieved through the energy term E_c. Taking the neighboring agents (as dynamic obstacles) and the environment (as static obstacles) as input, we compute the collision-free velocity $\mathbf{v}_{i,j}^c$ for the agent i by using reliable local collision avoidance algorithms, such as Social Force Model (SFM) [11] and Optimal Reciprocal Collision Avoidance (ORCA) [3]. Then the velocity $\mathbf{v}_{i,j}^c$ is compared with the candidate velocity $\mathbf{v}_{r,k}$ in the velocity set by measuring the similarity, which leads to the following formula for E_c:

$$E_c = \left\| \mathbf{v}_{i,j}^c - \mathbf{v}_{r,k} \right\|_2 . \quad (5)$$

Our approach can be easily integrated with any local collision avoidance algorithm to obtain $\mathbf{v}_{i,j}^c$. Different collision avoidance mechanisms lead to different crowd movement patterns. As shown in Fig. 4, in a scene where two groups (red and white) face each other, the agents that use SFM to avoid collisions tend to gather in small piles, while the agents using ORCA are scattered to avoid collisions.

4.3 Movement direction energy term

The movement direction energy term E_d is introduced for global navigation, which imitates the agent's movement toward its goal. Through the constructed direction field over the entire environment during initialization, the ideal (fastest) moving direction of any

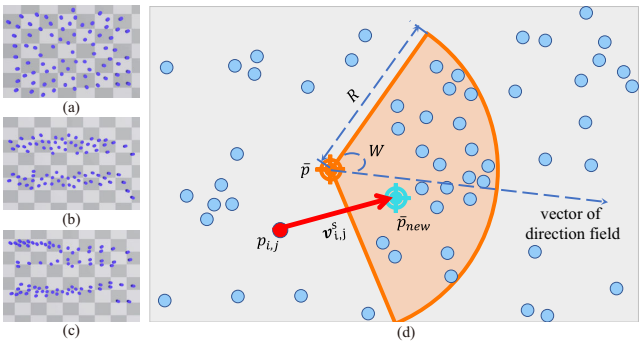


Fig. 5 Various crowd movements simulated by our approach (a) without the group formation energy term and (b)-(c) with different settings of positional attractions: (b) two subgroup formed by manually specified attractors, and (c) two dynamic subgroup formed using automatically specified attractors by mean-shift. In (d), we give an illustration of the mean-shifting process applied on the feature space (blue dots) of all agents for crowd aggregation in a square.

agent can be obtained from the movement direction stored in the grid cell where it is located (Fig. 3). Correspondingly, the energy for movement direction control is presented as:

$$E_d = \left\| \mathbf{n}^d_{i,j} - \mathbf{n}_{r,k} \right\|_2 , \tag{6}$$

where $\mathbf{n}^d_{i,j}$ is the unit vector denoting the desired moving direction of agent i at frame j, $\mathbf{n}_{r,k}$ is the unit vector representing the moving direction of agent candidate r at frame k in the velocity dataset.

4.4 Group formation energy term

Using the five types of energy items introduced above can generate natural-looking macroscopic crowd behaviors. But in addition to the basic collision avoidance between individuals (Fig. 5(a)), it is difficult to describe the complex interaction behaviors between individuals and simulate various group behavior patterns, such as array, queue, aggregation, and evacuation. Furthermore, people often change their behavior states in the real world. For example, people come out of a crowded subway station, cross the square, and then walk to the entrance of a shopping mall to queue. However, previous methods usually use uniform behavior control rules for the entire crowd, which leads to consistent behaviors of individuals in the crowd and lack of environmental adaptability [32]. In order to generate various movement patterns of the crowd in a dynamically changing environment, we define a *group formation velocity* $\mathbf{v}^s_{i,j}$ to imitate the agent i's decision-making under the influence of the surrounding local environment at frame j, and correspondingly, the group formation energy term E_g is given as:

$$E_g = \left\| \mathbf{v}^s_{i,j} - \mathbf{v}_{r,k} \right\|_2 . \tag{7}$$

Positional attraction: Specifically, the influence of the surrounding local environment on individual decision-making is often manifested as position attractiveness [33]. Different definitions of position attractiveness lead to different formulations of group formation velocity $\mathbf{v}^s_{i,j}$, thereby generating different crowd behavior patterns in simulation. Basically, a certain point in the environment or a certain individual in the crowd can be set as a source of attraction to guide individuals, thereby generating the aggregation and guide-follower crowd movement patterns. For the queuing crowd (Fig. 5(b)), the individual in front of each individual can be regarded as the source of positional attraction, so that they follow the individual in front of them and make decisions based on the distance between them. Formally, we define the group formation velocity of the agent as:

$$\mathbf{v}^s_{i,j} = \delta_i(\mathbf{p}^a_{i,j} - \mathbf{p}_{i,j}), \tag{8}$$

where $\mathbf{p}^a_{i,j}$ is the positional attraction source, $\mathbf{p}_{i,j}$ is the position of agent i at frame j, δ_i is the weight of distance influence.

Mean-shift clustering: By specifying the attraction points, rich agent interactions and various group movement patterns can be realized in the crowd. However, it is difficult to manually specify the attraction points for the scenarios where the positional attraction sources of agents are different or dynamically changing, such as crowds walking in twos in a square, and crowds dynamically gathering at multiple exits during evacuation. In such a movement pattern where a self-organized crowd gathers into small groups (Fig. 5(b)), people usually tend to walk to areas with high crowd density around them.

Inspired by this observation, we introduce the mean-shift clustering method [5,6] to automatically compute the dynamic attraction point. The feature space is spanned by the position of each agent in crowd. As illustrated in Fig. 5(d), a window function is selected as a kernel and the mean-shift algorithm iteratively shifts this kernel to a higher density region until convergence. Each shift is defined by a mean-shift vector pointing toward the maximal increasing direction in terms of density. Here, the kernel is shifted to the centroid (or the mean) of all points falling in the local support of the window function. The centroid is called the mean point and is used as the attraction point for group formation in crowd simulation.

Specifically, the mean point of agent i can be computed as follows:

1) Initialize the mean point $\bar{\mathbf{p}}$ with the agent i's position $\mathbf{p}_{i,j}$ at frame j.

2) Apply a window function $\omega(\mathbf{p}, \bar{\mathbf{p}})$ to control the size of agents considered in mean-shift iteration:

$$\omega(\mathbf{p}, \bar{\mathbf{p}}) = \begin{cases} 1, & \theta(\mathbf{p}, \bar{\mathbf{p}}) \leq W \text{ and } L(\mathbf{p}, \bar{\mathbf{p}}) \leq R, \\ 0, & \text{otherwise,} \end{cases} \quad (9)$$

where W and R control the window size ($W = 60^\circ$ and $R = [1.5, 6]$ in our implementation). $\mathbf{p}$ is the position of an agent in crowd, $L(\mathbf{p}, \bar{\mathbf{p}})$ is the distance between the directions of the mean point $\bar{\mathbf{p}}$ and agent, and $\theta(\mathbf{p}, \bar{\mathbf{p}})$ is the angle between the direction field of the grid (where the mean point $\bar{\mathbf{p}}$ is located) and the vector pointing from $\bar{\mathbf{p}}$ to $\mathbf{p}$. If $\omega(\mathbf{p}, \bar{\mathbf{p}})$ equals 0, the effect of this agent will not be considered in the next simulation step.
For each agent i within the window, we compute its weight for re-estimating the mean using a kernel function $k(\mathbf{p}_i, \bar{\mathbf{p}})$:

$$k(\mathbf{p}_i, \bar{\mathbf{p}}) = e^{-0.5 g_i} e^{-0.05 l_i} e^{-2 d_i}, \quad (10)$$

$$g_i = \left\| \frac{\theta(\mathbf{p}_i, \bar{\mathbf{p}})}{W} \right\|^2, l_i = \left\| \frac{L(\mathbf{p}_i, \bar{\mathbf{p}})}{R} \right\|^2, d_i = \left\| \frac{s(\mathbf{p}_i)}{s_i^0} \right\|^2, \quad (11)$$

where $s(\mathbf{p}_i)$ represents the length of the rest path from the agent i's current position $\mathbf{p}_i$ to the target, and s_i^0 gives the length of agent i's path from its initial position to the target.

3) Compute the new mean $\bar{\mathbf{p}}_{new}$ using the weighted mean of the density in the window as follows:

$$\bar{\mathbf{p}}_{new} = \frac{\sum_{i \in A} k(\mathbf{p}_i, \bar{\mathbf{p}}) \mathbf{p}_i}{\sum_{i \in A} k(\mathbf{p}_i, \bar{\mathbf{p}})}, \quad (12)$$

where A is the set of agents, and $\mathbf{p}_{new}$ means the new position in the mean-shift algorithm.

4) Shift the window center to $\bar{\mathbf{p}}_{new}$ and go back to Step 2 if the terminal condition has not been met. If the number of iterations exceeds 100 times or the deviation of $\bar{\mathbf{p}}_{new}$ is less than 0.01, the algorithm terminates. $\bar{\mathbf{p}}_{new}$ is employed as the positional attraction and we compute the group formation velocity $\mathbf{v}_{i,j}^s$ for agent i at frame j by Eq. 8.

4.5 Parameter Calibration

In our proposed method, there are totally 8 weight parameters $(w_{v1}, w_{v2}, w_p, w_e, w_c, w_d, w_g, \delta_i)$ related to the energy terms. In order to avoid the tedious non-trivial tuning of these parameters in a trial-and-error manner, we use the *Simulated Annealing* algorithm [7] to calibrate parameters. Given the real-world traffic data, the calibration task is to determine the specific optimal parameter set of the proposed model that best fits the given trajectory for each crowd agent.

The simulated annealing algorithm works as follows. First, the 8 parameters are randomly initialized within the range of $[1, 10]$. We set the initial temperature as $1,000,000^\circ C$ and define the temperature reduction function as $T_{new} = 0.25 * T_{current}$ according to the geometric reduction rule, where $T_{current}$ is the current temperature, and T_{new} is the new temperature. For each temperature, the number of iterations is set to 500, which leads to 500 parameter sets for our simulation model.

As the temperature drops, the range of parameters gradually changes from $[0.01, 0.1]$ to $[0.01, 0.01]$ with each iteration r. The created parameter set is applied to our simulation model to generate the virtual movement of the crowd agent. Then, we measure the error F_r^v between simulated behavior and the real-world agent trajectory in terms of velocity, defined as follows:

$$F_r^v = \frac{1}{N} \sum_{j=1}^{N} \left\| \mathbf{v}_j^{sim} - \mathbf{v}_j^{data} \right\|_2, \quad (13)$$

where $\mathbf{v}_j^{sim}$ and $\mathbf{v}_j^{data}$ are the simulated velocity and the given velocity of the agent at an arbitrary frame j, respectively. N is the total number of frames of agent movement. We compare F_r^v with the error in the previous iteration F_{r-1}^v by $\Delta F = F_r^v - F_{r-1}^v$. According to the value of ΔF, the parameters generated in this iteration will be accepted with a certain probability ρ:

$$\rho = \begin{cases} 1, & \text{if } \Delta F \leq 0; \\ e^{-\Delta F / T}, & \text{if } \Delta F > 0, \end{cases} \quad (14)$$

where T is the current temperature.

Repeat the above procedures and decrease the temperature according to the reduction function, the annealing process stops until the termination conditions are met. The termination criterion is either the end temperature is less than 1.0 or the change of error F_r^v is less than 0.01. The calibration process takes about eight hours for 40 trajectories.

Using the calibrated parameter set, we can achieve a simulated crowd similar to the given real-world crowd data. In order to further simulate complex crowd movements in various scenarios, fine-tuning can be further made based on these calibrated parameters.

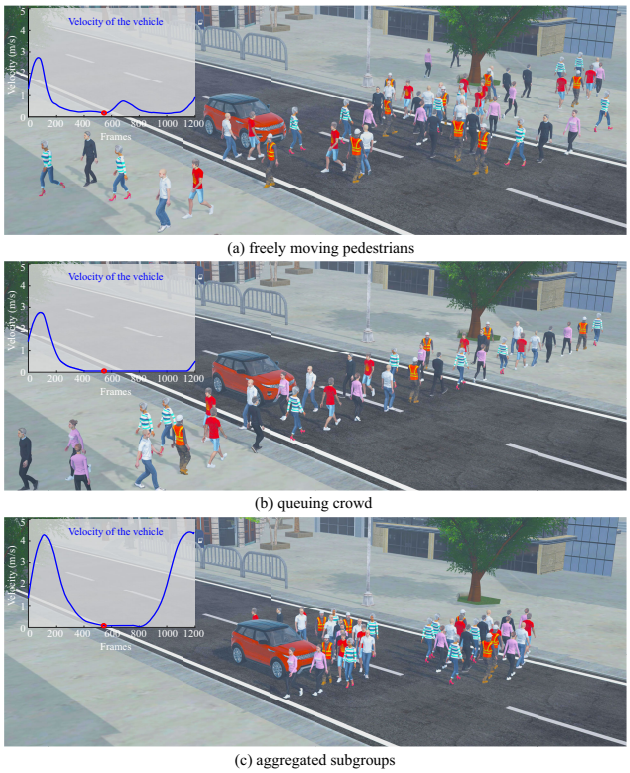
5 Dynamic Crowd Simulation

In order to exhibit various movement patterns in the same crowd scenario, the positional attraction sources can be defined in different ways for different agent groups. It is also possible to simulate the agent movement in a dynamically changing environment by switching the direction field according to its dynamically changing target and the positional attraction

computing ways. Specifically, in a multi-exit crowd evacuation scenario, we separately set up a direction field for the entire environment with each exit as the target position during initialization, where each grid stores the distance from the current position to the target and the crowd density in the grid. During the evacuation process, the agent will dynamically select the appropriate exit and switch the direction field according to the density of the crowd around each exit and the distance from each exit.

The general velocity-based framework we proposed can be easily integrated with other crowd simulation methods. Specifically, the computation method of group formation velocity can be replaced by an arbitrary microscopic group modeling method. The desired moving direction in the movement direction energy term can be obtained from any global navigation mechanism. Similarly, any collision avoidance approach can be used to obtain the collision-free velocity in the collision avoidance energy term.

We first validated our approach in an autonomous vehicle navigation test scenario, where a virtual crowd of 40 pedestrians crosses an urban road without traffic lights. In order to test the decision-making ability of the autonomous vehicle in different crowd situations, we generated three different behavior patterns of crowds by defining positional attraction sources in different ways: freely moving pedestrians, queuing crowd, and aggregated subgroups. When describing the mutual interaction between the car and pedestrians, we employ the method of Chao et al. [4] for car decision-making. Fig. 6 shows the simulation result and the velocity curve of the autonomous vehicle in each case. It can be seen that the decision-making of the autonomous vehicle varies when facing crowds with different movement patterns. This proves the necessity and effectiveness of our method to simulate various crowd behaviors for application in autonomous vehicle testing.



(a) freely moving pedestrians

(b) queuing crowd

(c) aggregated subgroups

Fig. 6 Example crowd simulation results with different movement patterns generated by our approach for autonomous vehicle testing: (a) freely moving pedestrians, (b) queuing crowd, and (c) aggregated subgroups. The blue curve in each image shows the velocity of the autonomous vehicle.

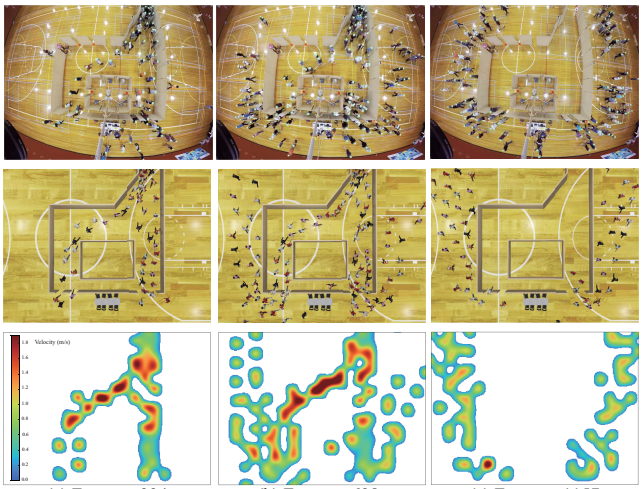
6 Simulation Results

6.1 Comparison with real-world crowd dynamics

Evacuation: We use virtual crowds to reproduce the real-world crowd evacuation experiment organized by the University of Melbourne. The escape layout is designed to build four panels in the basketball stadium, one of which has four narrow exits (only one person can pass through at the same time). In this scenario, people dynamically choose the exit according to the dynamically changing environment, and after escaping, they will also choose to go left or right. Fig. 7 show the snapshots of the experimental video (top) and our simulation result (middle). It can be seen that the various movement patterns, such as self-organized queuing, aggregation at the exit, and spreading movement pattern after escaping, can all be reproduced by our approach. This can be achieved by assigning different definitions of positional attraction sources to different groups of agents.

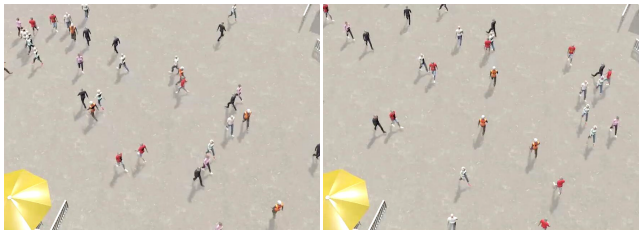
In addition, an agent's movement in this dynamically changing environment is simulated by switching the direction field according to its dynamically changing target and the positional attraction computing ways. Specifically, we set up multiple direction fields for the entire environment with each exit as the target position during initialization, where each grid stores the distance from the current position to the target and the crowd density in the grid. During the evacuation, the agent will dynamically select the appropriate exit and switch the direction field according to the density of the crowd around each exit and the distance from each exit.

The bottom images in Fig. 7 shows the heatmap of the pedestrian velocity distribution in different areas during the simulation. At the $334th$ frame, the individual slows down after passing the right exit, resulting in congestion and slower agent velocities at the right exit. At the same time, there is more space near the



(a) Frame = 334 (b) Frame = 628 (c) Frame = 1157

Fig. 7 The snapshots of video of the real-world crowd evacuation experiment (top), our simulation result (middle), and the heatmap of pedestrian velocity distribution (bottom).



(a) (b)

Fig. 8 The comparison between (a) the rendering of real-world crowd movements from ETH-Univ dataset, and (b) our simulation result with this dataset as input.

left exit, and the crowd leaves in an orderly manner at a faster speed (at the 628th frame). Moreover, it can be observed that some people change their target exit from the right to the left in order to leave more quickly. After the evacuation is completed (at the 1157th frame), the crowd in the open area moves at the desired speed, resulting in a more even distribution of crowd velocity.

Square: We simulate the crowd behaviors in a square using the real-world square crowd dataset ETH-Univ [30] as input, and compare the generated crowd dynamics with that in the real-world sample. As shown in Fig. 8, the virtual crowd presents similar behaviors as the ones in the real-world scene, especially the movement pattern of walking in twos. Fig. 9 also gives the velocity probability distribution of the simulated crowd, which matches that of the ETH-Univ dataset.

6.2 Runtime Performance

Table 1 gives the detailed parameter settings used in the above three scenarios. It is worth noting that in the square scene, we increase the range of the mean shift

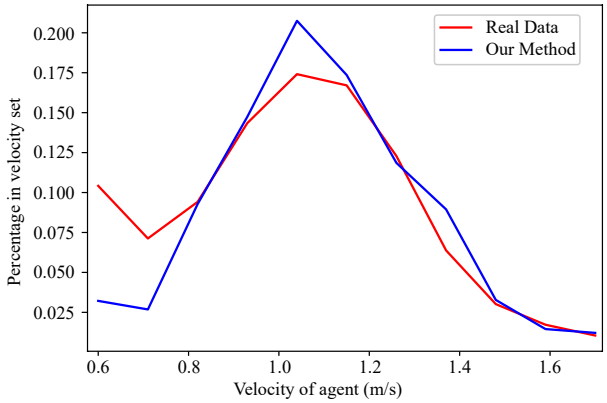


Fig. 9 Comparison of crowd speed distributions between real crowd data from the ETH-Univ dataset (red), and our simulated crowd (blue) with this dataset as input .

Table 1 Parameter values for different scenarios.

Scence		w_{v1}	w_{v2}	w_p	w_e	w_c	w_d	w_g	δ_i	z
Vehicle interaction		2.9	1.0	2.5	1.0	3.0	1.9	3.0	1.0	15
Evacuation	indoor	3.9	1.0	2.5	0.2	3.2	3.5	3.0	1.5	10
	outdoor	3.9	1.0	2.5	0.2	6.0	1.0	1.9	1.0	
Square		2.9	1.0	2.5	1.2	8.0	2.6	4.2	1.0	8

and the weight of group formation term w_g to make more agents walk in pairs. At the same time, in order to keep the crowd collision-free, we increase the weight of collision avoidance w_c.

The timing performance of our approach is related to the number of agents and the search range z (representing the number of adjacent subsets to be searched) in the velocity set. Fig. 10 shows the runtime performance of our method in terms of agent number and search range. It can be seen that the runtime of our data-driven optimization process is approximately linear with the crowd size. When the search range is set to 3, our method can simulate about 750 agents in real time (30 fps) and about 1,750 agents at interactive simulation rates (10 fps) . In addition, the runtime at $z = 6$ is twice that at $z = 3$, which indicates that when the search range z of the dataset is expanded, the time spent is also doubled. All the reported times were obtained on a 64bit desktop machine with a 2.30GHz Inter Core CPU i5-8300H processor and 8GB memory.

6.3 Comparison with typical approaches

We compare our method with that of Ren et al. [32] in a benchmark scenario with a circular obstacle in the middle. Two groups (80 agents each) are initialized on opposite sides. The positions of targets are given in the vicinity of the initial location of the opposite group, where a lot of collisions and interactions could happen in the middle. As can bee seen from Fig. 11(a),

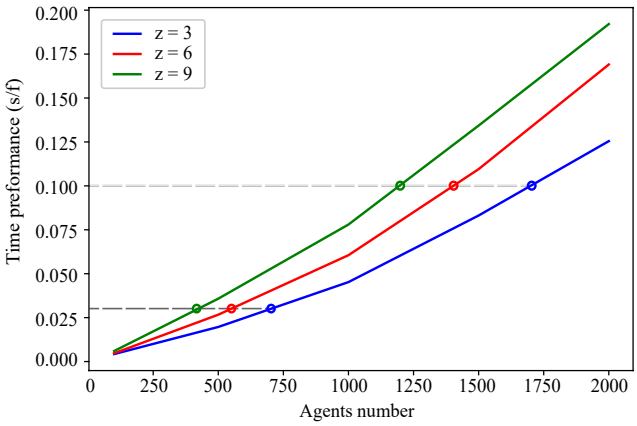


Fig. 10 The runtime performance of our approach with different values of search range z of the velocity set.

the agents simulated by Ren et al.'s method avoid the obstacle and other agents in the process of moving towards the target. The direction of movement is controlled simply by pointing from the current position to the target position, without considering the static obstacle in the environment. In contrast, our method builds a direction field for the entire environment as the agent's global navigation at any location. In addition to basic collision avoidance, the agents simulated by our approach exhibit various interactive behaviors, which leads to different crowd movement patterns, such as queuing, aggregation, and the switching of the two formations as shown in Fig. 11(b-d).

As a comprehensive framework, our method can be incorporated into any crowd simulation system to achieve more realistic virtual crowd movements by replacing the calculation method of the local optimal velocity in each energy item. Fig. 12 compares our approach with that of Patil et al. [27], which uses navigation fields to direct crowd simulation with the Social Force Model for local collision avoidance. The scenario comprises of four groups of 25 agents each in each corner of the environment that need to move to the opposite corner (Fig. 12 (a)). In the middle, there are four square-shaped static obstacles that form narrow passages. As shown in Fig. 12 (b), the agent simulated by Patil et al.'s approach can smoothly move to the target position while avoiding collisions with other agents and obstacles. In contrast, due to the introduction of the group formation mechanism, the crowd generated by our approach can exhibit different group movement patterns in the same scenario (Fig. 12 (c)), such as aggregation, queue, array, and basic collision avoidance similar to that in Fig. 12 (b).

We also compare our method to two popular local collision avoidance methods, SFM and ORCA, in a benchmark scenario with 200 agents evacuating

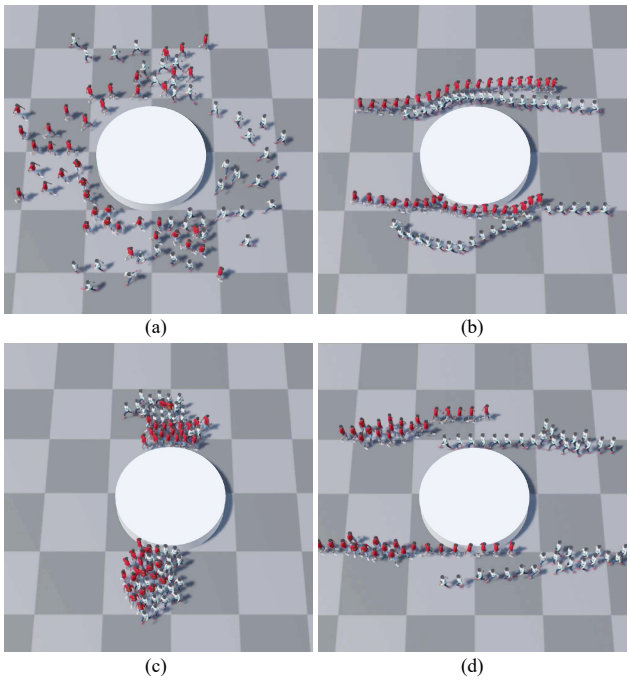


Fig. 11 Snapshots of crowd movements in a scenario with a circular obstacle in the middle: (a) basic collision avoidance using the method of [32], and (b) queue, (c) aggregation, (d) aggregation-queue switching by our approach.

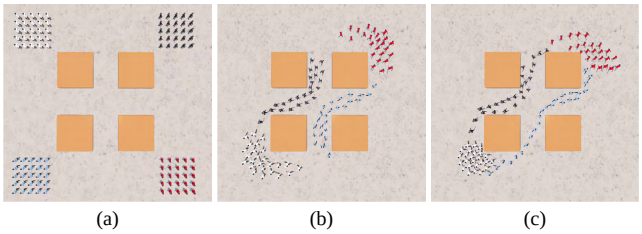


Fig. 12 (a) The initial position of the crowd in the scenario, and intermediate positions of the agents in the simulation (b) using the method of Patil et al. [27], and (c) by our approach.

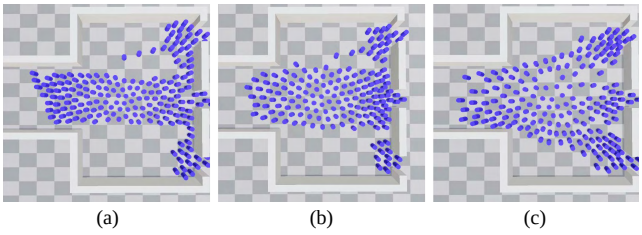


Fig. 13 Snapshots of crowd simulation in an evacuation scenario by (a) ORCA, (b) SFM, and (c) our approach with the introduction of SFM.

through three narrow exits. It can be seen from Fig. 13 (a-b) that the simulated crowd using only ORCA/SFM will be blocked at the middle exit, while the other two exits have very few people. In contrast, by introducing SFM into our framework, agents in the simulated crowd will dynamically choose exits with relatively few

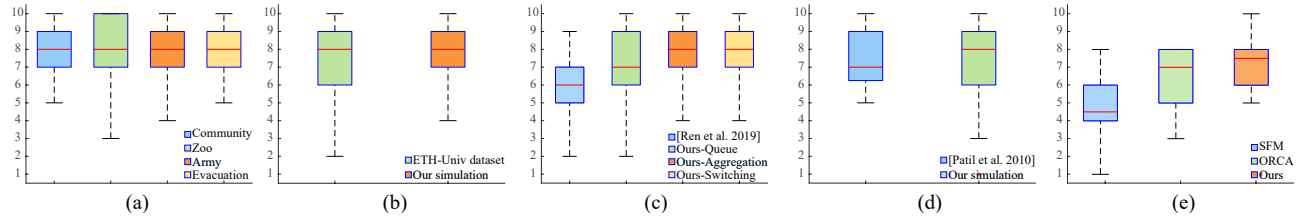


Fig. 14 Plausibility scores of the user study. (a) Statistics of the simulated crowd in various scenarios generated by our approach. (b) Comparison statistics between our simulated crowd dynamics and the real-world crowd samples. (c) Statistics for comparison between Ren et al's approach [32] and ours. (d) Experimental outcomes comparing our method with the method of Patil et al. [32]. (e) Comparative statistics between SFM, ORCA, and our approach.

people, which is also in line with the real-world crowd dynamics.

6.4 Perceptual Study

We conducted a user study to understand and analyze whether the virtual crowd simulated by our method meets people's perception of realistic crowds. We rendered all the 14 simulated crowd animations and 1 ETH-Univ crowd dataset shown in the figures above. Then we recruited 39 participants to participate in this user study. All the participants are graduate students aged between 20 and 30, with normal visions. At each time they were asked to watch one crowd animation stimulus and then rate it in terms of its perceived fidelity. The score range is from 1 (not at all realistic) to 10 (very realistic). To counterbalance the order of the visual stimuli, the stimuli were displayed in a random order for each participant. The participants were allowed to view an animation stimulus many times before scoring it.

The outcomes of the user study are reported in Fig. 14. The various scenarios simulated by our method have got an average score of 8, which shows that people feel that our generated crowd behaviors are very realistic. However, the queuing scene shown in Fig. 11(b) has a slightly lower mean score 7, because users feel that the queuing movement pattern does not match this scene. Fig. 14(b) shows the fidelity score comparison between our simulated crowd and the real captured data from ETH-Univ. In the same rendering environment (Fig. 8), the mean scores of the two are very close (p-value $= 0.146 > 0.05$), which is enough to prove that our simulated crowd is so realistic that it is indistinguishable from the real crowd. Furthermore, in the score comparison with other typical simulation methods (Ren et al's approach[32], Patil et al's approach[27], SFM and ORCA), the crowd animation by our method obtains higher mean scores, due to the richer crowd movement patterns and more vivid crowd behaviors (Fig. 14(c)-(e)).

7 Conclusion

We present a comprehensive framework for crowd simulation that allows interactive control of global navigation, local collision avoidance, and group formation. Each control is integrated into a velocity-based crowd energy metric to measure the similarity between the agent's required simulated velocity and a given velocity sample. Through a data-driven optimization process, our approach can simulate crowd behaviors in a dynamically changing environment and generate various movement patterns in the same scenario. Furthermore, our approach can be applied to new scenarios beyond the input dataset and simulate agent behaviors that may differ from those captured by the input data.

As a common problem of data-driven methods, the composition of the input real-world crowd trajectory data directly affects the simulation results and time efficiency of our method. Although our method can deal with the presence of noise in the input data, it has to expand the velocity search range to find the local optimal speed, this will slow down the update efficiency of the crowd. Moreover, if the velocity distribution of the input samples is sparse, the simulated agent is prone to abnormal behaviors with discontinuous speed, such as jitter and drift.

In the future, we plan to extend our method to introduce full-body motions in the animation of crowd characters. Various types of spatio-temporal information about the ongoing character interactions (e.g. distances, relative positions or velocities, time to collision) can be calculated during the simulation process, and used to trigger specific character animations. Also, observations show that different people will achieve the same goal in different manners in terms of the underlying personality, we plan to emulate personality traits of individuals within a crowd. Last but not least, we are interested in integrating our approach with various crowd editing tools and algorithms. In this way, users can flexibly refine the quality of crowd behaviors directly synthesized by our approach.

Acknowledgements Xiaogang Jin was supported by the National Natural Science Foundation of China (Grant No. 62036010), the Key Research and Development Program of Zhejiang Province (Grant No. 2020C03096), and the Ningbo Major Special Projects of the "Science and Technology Innovation 2025" (Grant No. 2020Z007).

References

1. Alahi, A., Goel, K., Ramanathan, V., Robicquet, A., Fei-Fei, L., Savarese, S.: Social lstm: Human trajectory prediction in crowded spaces. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 961–971 (2016)
2. Van den Berg, J., Lin, M., Manocha, D.: Reciprocal velocity obstacles for real-time multi-agent navigation. In: 2008 IEEE International Conference on Robotics and Automation, pp. 1928–1935. IEEE (2008)
3. Berg, J.v.d., Guy, S.J., Lin, M., Manocha, D.: Reciprocal n-body collision avoidance. In: Robotics research, pp. 3–19. Springer (2011)
4. Chao, Q., Liu, P., Han, Y., Lin, Y., Li, C., Miao, Q., Jin, X.: A calibrated force-based model for mixed traffic simulation. IEEE transactions on visualization and computer graphics (2021)
5. Chao, Q., Yu, J., Dai, C., Xu, T., Zhang, L., Wang, C.C., Jin, X.: Steering micro-robotic swarm by dynamic actuating fields. In: 2016 IEEE International Conference on Robotics and Automation (ICRA), pp. 5230–5235. IEEE (2016)
6. Comaniciu, D., Meer, P.: Mean shift: A robust approach toward feature space analysis. IEEE Transactions on Pattern Analysis and Machine Intelligence **24**(5), 603–619 (2002)
7. Dowsland, K.A., Thompson, J.: Simulated annealing. Handbook of Natural Computing pp. 1623–1655 (2012)
8. Guy, S.J., Curtis, S., Lin, M.C., Manocha, D.: Least-effort trajectories lead to emergent crowd behaviors. Physical Review E **85**(1), 016,110 (2012)
9. He, L., Pan, J., Narang, S., Wang, W., Manocha, D.: Dynamic group behaviors for interactive crowd simulation. arXiv preprint arXiv:1602.03623 (2016)
10. Helbing, D., Farkas, I., Vicsek, T.: Simulating dynamical features of escape panic. Nature **407**(6803), 487–490 (2000)
11. Helbing, D., Molnar, P.: Social force model for pedestrian dynamics. Physical Review E **51**(5), 4282 (1995)
12. Ju, E., Choi, M.G., Park, M., Lee, J., Lee, K.H., Takahashi, S.: Morphable crowds. ACM Transactions on Graphics **29**(6), 1–10 (2010)
13. Karamouzas, I., Overmars, M.: Simulating and evaluating the local behavior of small pedestrian groups. IEEE Transactions on Visualization & Computer Graphics **18**(3), 394–406 (2012)
14. Karamouzas, I., Skinner, B., Guy, S.J.: Universal power law governing pedestrian interactions. Physical Review Letters **113**(23), 238,701 (2014)
15. Karamouzas, I., Sohre, N., Hu, R., Guy, S.J.: Crowd space: a predictive crowd analysis technique. ACM Transactions on Graphics (TOG) **37**(6), 1–14 (2018)
16. Kim, M., Hwang, Y., Hyun, K., Lee, J.: Tiling motion patches. In: Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation, pp. 117–126 (2012)
17. Kim, S., Bera, A., Best, A., Chabra, R., Manocha, D.: Interactive and adaptive data-driven crowd simulation. In: 2016 IEEE Virtual Reality (VR), pp. 29–38. IEEE (2016)
18. Lee, J., Won, J., Lee, J.: Crowd simulation by deep reinforcement learning. In: Proceedings of the 11th Annual International Conference on Motion, Interaction, and Games, pp. 1–7 (2018)
19. Lemercier, S., Jelic, A., Kulpa, R., Hua, J., Fehrenbach, J., Degond, P., Appert-Rolland, C., Donikian, S., Pettré, J.: Realistic following behaviors for crowd simulation. Computer Graphics Forum **31**(2pt2), 489–498 (2012)
20. Lerner, A., Chrysanthou, Y., Lischinski, D.: Crowds by example. Computer Graphics Forum **26**(3), 655–664 (2007)
21. Li, Y., Christie, M., Siret, O., Kulpa, R., Pettré, J.: Cloning crowd motions. In: Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation, pp. 201–210. Citeseer (2012)
22. Moussaïd, M., Perozo, N., Garnier, S., Helbing, D., Theraulaz, G.: The walking behaviour of pedestrian social groups and its impact on crowd dynamics. PloS one **5**(4), e10,047 (2010)
23. Musse, S.R., Thalmann, D.: A model of human crowd behavior: Group inter-relationship and collision detection analysis. In: Computer Animation and Simulation'97, pp. 39–51. Springer (1997)
24. Musse, S.R., Thalmann, D.: Hierarchical model for real time simulation of virtual human crowds. IEEE Transactions on Visualization and Computer Graphics **7**(2), 152–164 (2001)
25. Narain, R., Golas, A., Curtis, S., Lin, M.C.: Aggregate dynamics for dense crowd simulation. ACM Transactions on Graphics **28**(5), 122 (2009)
26. Ondřej, J., Pettré, J., Olivier, A.H., Donikian, S.: A synthetic-vision based steering approach for crowd simulation. ACM Transactions on Graphics (TOG) **29**(4), 123 (2010)
27. Patil, S., Van Den Berg, J., Curtis, S., Lin, M.C., Manocha, D.: Directing crowd simulations using navigation fields. IEEE Transactions on Visualization and Computer Graphics **17**(2), 244–254 (2010)
28. Pedica, C., Vilhjálmsson, H.: Social perception and steering for online avatars. In: International Workshop on Intelligent Virtual Agents, pp. 104–116. Springer (2008)
29. Pelechano, N., Allbeck, J.M., Badler, N.I.: Controlling individual agents in high-density crowd simulation. SCA '07, p. 99–108. Eurographics Association, Goslar, DEU (2007)
30. Pellegrini, S., Ess, A., Gool, L.V.: Improving data association by joint modeling of pedestrian trajectories and groupings. In: European conference on computer vision, pp. 452–465. Springer (2010)
31. Qiu, F., Hu, X.: Modeling group structures in pedestrian crowd simulation. Simulation Modelling Practice and Theory **18**(2), 190–205 (2010)
32. Ren, J., Xiang, W., Xiao, Y., Yang, R., Manocha, D., Jin, X.: Heter-sim: Heterogeneous multi-agent systems simulation by interactive data-driven optimization. IEEE Transactions on Visualization and Computer Graphics **27**(3), 1953–1966 (2021)
33. Ren, Z., Charalambous, P., Bruneau, J., Peng, Q., Pettré, J.: Group modeling: A unified velocity-based approach. Computer Graphics Forum **36**(8), 45–56 (2017)
34. van Toll, W., Grzeskowiak, F., Gandía, A.L., Amirian, J., Berton, F., Bruneau, J., Daniel, B.C., Jovane, A., Pettré, J.: Generalized microscropic crowd simulation using costs in velocity space. In: Symposium on Interactive 3D Graphics and Games, pp. 1–9 (2020)

35. Treuille, A., Cooper, S., Popović, Z.: Continuum crowds. ACM Transactions on Graphics (TOG) **25**(3), 1160–1168 (2006)
36. Vemula, A., Muelling, K., Oh, J.: Social attention: Modeling attention in human crowds. In: 2018 IEEE international Conference on Robotics and Automation (ICRA), pp. 4601–4607. IEEE (2018)
37. Wolinski, D., Lin, M.C., Pettré, J.: Warpdriver: context-aware probabilistic motion prediction for crowd simulation. ACM Transactions on Graphics (TOG) **35**(6), 164 (2016)