Computer Vision and Image Understanding 134 (2015) 48–63

Contents lists available at ScienceDirect

Computer Vision and Image Understanding

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

Analysis-by-synthesis: Pedestrian tracking with crowd simulation models in a multi-camera video network

Zhixing Jin *, Bir Bhanu

Center for Research in Intelligent Systems, University of California, Riverside, CA 92521, United States

ARTICLE INFO

Article history:
Received 2 February 2014
Accepted 6 October 2014

Keywords:
Pedestrian tracking
Multi-camera systems
Crowd simulation

ABSTRACT

For tracking systems consisting of multiple cameras with overlapping field-of-views, homography-based approaches are widely adopted to significantly reduce occlusions among pedestrians by sharing information among multiple views. However, in these approaches, the usage of information under real-world coordinates is only at a preliminary level. Therefore, in this paper, a multi-camera tracking system with integrated crowd simulation is proposed in order to explore the possibility to make homography information more helpful. Two crowd simulators with different simulation strategies are used to investigate the influence of the simulation strategy on the final tracking performance. The performance is evaluated by multiple object tracking precision and accuracy (MOTP and MOTA) metrics, for all the camera views and the results obtained under real-world coordinates. The experimental results demonstrate that crowd simulators boost the tracking performance significantly, especially for crowded scenes with higher density. In addition, a more realistic simulation strategy helps to further improve the overall tracking result.

1. Introduction

Tracking pedestrians has been an active research topic in computer vision. Although many sophisticated techniques have been proposed, it is still a challenging problem that requires further advances to track people in a camera network in real-world applications. There are many reasons that make tracking a difficult problem. For example, the illumination conditions are changing continuously; the appearances for a single pedestrian are not the same from different perspectives; the amount of occlusions among pedestrians is significant when people form a crowd; and moreover, the human behaviors are sometimes unpredictable. To overcome these problems, researchers have proposed different approaches, among which are the approaches that aim to use multiple overlapping cameras [1]. By fusing information from cameras with overlapped field-of-views, the tracking accuracy could be improved due to the reduction of the influence from occlusion, the separation of crowded people from one another in large foreground blobs, etc. [2,3].

In this paper, we are focused on the setting where several cameras with overlapped field-of-views are used to track pedestrians [4,5]. To take advantage of this setting, many approaches make use of homography-related methods [3,6,7], which are able to model the relationship among different views in order to estimate the actual position of each pedestrian in the real-world ground plane. These methods are efficient since the only extra information that we need to know is the set of camera parameters and the calculation of the perspective transformation is computationally light.

However, in almost all the current trackers that use multiple cameras, the estimated real-world positions for pedestrians are only used in data association across cameras, while their relationships with one another are somehow ignored. In fact, the real-world positions of pedestrians are capable of providing more constraints and predictions, which can be quite helpful in addition to a traditional frame-based tracking approach. From this perspective, crowd simulation is a good example of methods that integrate extra information brought by the real-world positions of pedestrians. In the area of computer graphics, crowd simulation is a very popular topic, with various applications such as designing emergency evacuation routes and introducing special effects in movies. It is used for simulating the behavior of either every individual or the whole group under certain constraints (e.g., to avoid collisions). Nowadays, one of the most popular crowd simulation approaches is mainly focused on simulating walking trajectories (direction and velocity) of each individual given the starting and ending locations. As in a multi-camera system, the direction and velocity information for each pedestrian can be acquired based on the estimated real-world positions at each frame, integrating crowd simulation algorithms with image analysis will be useful for accurate tracking.

In this paper, we propose an approach to improve the performance of a multi-camera tracking system by combining

* Corresponding author.
E-mail addresses: jinz@cs.ucr.edu (Z. Jin), bhanu@cris.ucr.edu (B. Bhanu).

http://dx.doi.org/10.1016/j.cviu.2014.10.001
1077-3142/

vision-based tracking results with the output from a crowd simulation under the real-world coordinates. The system diagram of the proposed approach is illustrated in Fig. 1. In addition to the frame trackers for different cameras which are independent from each other, a crowd simulator runs separately and provides predictions for all the views by projecting the real-world positions back to frame positions. This is a further extension of using the location and velocity information of pedestrians in a tracking system. The frame trackers for each view are designed based on the recent tracking-by-detection approach [8]. Compared to our previous work [9], the manner in which we integrate the information from crowd simulation is different. In addition, the crowd simulator in the proposed approach has two candidates: the RVO2 library [10] and the Social Behavior Model (SBM) [11,12]. Their simulation strategies are different, which may lead to different overall tracking performance. Therefore, besides the utilization of crowd simulation approaches to provide extra predictions for traditional vision-based tracking, a second purpose of this paper is to investigate whether the theoretically better simulation could be more valuable when integrated to the tracking system.

The rest of the paper is organized as follows. Section 2 gives a brief description of related work, including vision based tracking approaches (those use frame information only) and crowd simulation methods, as well as the contributions of this paper. Section 3 presents our proposed approach in detail. Section 4 gives the experimental results and provides a discussion on experiments. Finally Section 5 concludes the paper.

2. Related work and contributions

2.1. Pedestrian tracking

Most of the *state-of-the-art* tracking approaches belong to the category called tracking-by-classification or tracking-by-detection. They usually work in an online manner, e.g., the Online Ada-Boosting [13], Semi-Boosting [14] tracker and Online Multiple Instance Learning [15] tracker. The general idea for this category of trackers is to train a classifier based on the object's appearance features extracted from the initial patch (Region-of-Interest, ROI) on the first frame or the first several frames. Later at each time step, based on the evaluation from this classifier, a patch that maximizes the likelihood in the search window is located, and is further used to update the classifier itself. By repeating these steps, the classifier has the capability to adapt itself to the most recent tracking environment when video continues, as well as maintain a good performance on distinguishing target objects from the surroundings. However, such an evolving classifier may slowly drift and becomes off-target finally. Therefore, some tracking-by-detection approaches perform tracking purely based on detection results from a human detector, which are obtained independently from frame to frame [16–19]. There are also tracking approaches that integrate both evolving classifiers and human detectors [8,20]. Since a human detector is generally more confident, and works independently from the tracker, the input from a human detector is an effective way to initialize the classifier and/or provide the "ground-truth" to correct a drifted tracker. In the proposed approach, the pedestrian tracking method used is based on [8], which is composed of a particle filter, a human detector and an online boosting classifier. This approach is mainly based on particle filtering, while the human detector and online boosting classifier are used to adjust weights for the particles. In addition to the good performance shown in [8], another advantage of this approach is that the crowd simulation can be easily integrated into the system as the extra simulation information can be treated as an additional factor that influences particle weights.

As the ability for tracking in a single-camera is often limited, more cameras are added into a surveillance system to solve the problem, which is known as multi-camera tracking. Basically, there are three settings that a multi-camera tracking system can employ: overlapping cameras [2–5,21], non-overlapping cameras [22,23], and the mixture of them [24]. Related to our research focus in this

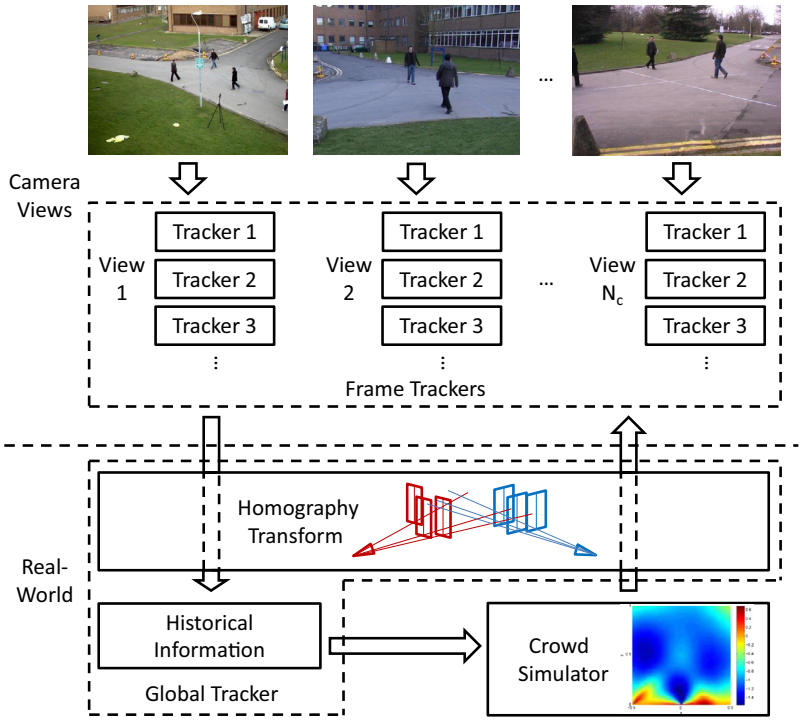


Fig. 1. The system diagram. This example is a system with N_c different views. During the initialization, the information from different camera views is integrated and saved as the historical information. Later at each time step: (1) the crowd simulator first generates the predicted locations for pedestrians; (2) then the simulated locations are passed to different camera views to help update the frame trackers; (3) finally the new position and velocity information for each pedestrian on the ground plane is calculated and used to update the historical information.

paper, the main challenge of a single-camera tracking system is the presence of occlusion in a crowded scene, therefore, we use an overlapped multi-camera system which can significantly relieve the problem. But as the number of cameras in a system becomes larger, the complexity of the system also increases and many other problems occur. For example, the data correspondence among cameras is one of the most important but not yet perfectly solved problem. There are many different methods reported for the solution of this problem, which can be generally divided into three categories: region-based approaches, point-based approaches, and principal axis-based ones [2,21]. However in this paper, data correspondence is not our focus, so we associate data from different cameras manually to avoid potential errors. We use the principal axis-based intersection method to calculate the position of each pedestrian on the ground plane.

2.2. Crowd simulation

The purpose of crowd simulation is to model human behaviors, particularly human walking, and to compare the output from these models to scenarios in real life. It can be considered as opposite to tracking [25]. Due to the uncertainty of human behaviors, many approaches have been proposed that try to model the problem from different perspectives. For example, the social force model is inspired by physics and social-psychology and has been successfully used in many applications such as evacuation simulation [26,27], with many further variants such as the Social Behavior Model [11,12]; the RVO2 (Reciprocal Velocity Obstacles) library assumes that all the pedestrians take the same collision avoidance strategy which leads to the global optimal solution and uses linear programming to have an efficient solution [10]; the rule-based approach provides simple but effective collision avoidance, which is only based on local information "observed" by a pedestrian in the crowd [28]; the continuum dynamics model treats the crowd at a macroscopic level, thus, it is highly efficient in simulating crowd behaviors at an extremely large and dense scale [29].

In most cases, crowd simulation approaches are point (individual) based. For example, the social force model (Social Behavior Model) considers each pedestrian in the crowd as a particle and the rules (e.g., collision avoidance) as forces added to the particle. Then the whole crowd simulation problem is transformed to a particle system, which is solved mathematically. In our previous work [9], the RVO2 library [10] is integrated as the crowd simulator. But in this paper, another simulator, Social Behavior Model [11] is combined to the tracking system as well.

The common advantages of these two crowd simulators (RVO2 library and Social Behavior Model) are: (1) They only require basic information for each pedestrian, such as position, velocity, and the desired velocity, which is quite easy to obtain; (2) The calculation for both models is relatively efficient. However, there are several essential differences between them: (a) Their collision avoidance strategies follow different ideas, which is similar to the different solutions in Prisoner's Dilemma in the game theory [30]. The RVO2 library uses a strategy which could result in a global optimal solution, and the Social Behavior Model adopts a much safer strategy but in most cases it is not the best from a holistic perspective. (b) The RVO2 library sets a safe time for each pedestrian, within which the pedestrian is absolutely safe from any kind of collision, but in the Social Behavior Model, the safety of a pedestrian is decided by the "forces" on him/her, which is elastic to a certain extent and not guaranteed. (c) When one pedestrian tries to prevent collisions with others, the RVO2 library treats all the other pedestrians within his/her observation range equally weighted, but the Social Behavior Model puts lower weights for pedestrians farther away from the focus of his/her field-of-view. In conclusion, the RVO2 library simulates pedestrians in a more robotic way, while the Social Behavior Model provides a simulation strategy more similar to human nature in real life.

2.3. Contributions

As compared to *state-of-the-art* tracking systems, the contributions of the paper are:

1. It integrates a crowd simulator with a tracker for the better usage of the pedestrian information under real-world coordinates. At each time step, the simulator generates a set of possible locations for all pedestrians, according to their previous positions, velocities, and the estimated desired velocities. When this additional information is combined with a video frame for a pedestrian, it serves as a better prediction for this pedestrian's location so as to improve the tracking performance.
2. Two crowd simulators are adopted and their influences on the tracking performance are investigated through extensive experiments. The two crowd simulators include the RVO2 library [9,10] and the Social Behavior Model (SBM) [11,12], where the Social Behavior Model follows a more realistic strategy for collision avoidance. The experiments are conducted on different sequences from a challenging multi-camera dataset, with different crowd densities.
3. We introduce a new framework to integrate the information from crowd simulation that is different from out previous work [9]. It discards the sampling step in our previous work when we calculate the contribution from simulation results to particle weights. In this manner, the particle weights in the frame trackers are evaluated without loss of any information. This new framework allows the system to outperform our previous tracking system in [9].

3. Technical approach

As shown in Fig. 1, the proposed system consists of three components: (1) a crowd simulator (RVO2 library or Social Behavior Model) working on the ground plane to provide predictions for the positions of each pedestrian; (2) a frame tracker based on the *state-of-the-art* tracking-by-detection, which is able to track pedestrians based on the visual information, corresponding to each pedestrian in each camera view; and (3) a global tracker that inter-

Table 1
Summary of important notations.

Notation	Description
t	Time step
tr	Frame tracker and/or its position
c_{tr}	Evaluation based on the boosting classifier of the frame tracker
d_o	Detected patch and/or its position
d_c	Detection confidence map
p	Particle in a frame tracker and/or its position
w	Weight for a particle
$\beta,\ \gamma,\ \eta,\ \delta$	Coefficients in observation model
N_p	Number of particles used in the particle filter
m	Matching score between frame trackers and detections
g	Gating score between frame trackers and detections
τ	Threshold for associating frame trackers and detections
$\mathbf{v}$	Pedestrian velocity on the ground plane
$\mathbf{p}$	Pedestrian position on the ground plane
$\mathbf{r}$	Pedestrian radius
$\mathbf{E^I},\ \mathbf{E^S},\ \mathbf{E^D}$	Velocity energy evaluations in Social Behavior Model
$\mathbf{W}$	Weight for interactive energy between pedestrians
$\mathbf{a}$	Pedestrian acceleration on the ground plane
$\mathbf{w}$	Weight for acceleration during velocity estimation
s_{tr}	Evaluation from crowd simulation results
L_{tr}	Simulated positions for a frame tracker

acts between the frame trackers working for different camera views and the crowd simulator working on the ground plane, as well as maintains the pedestrian information on the ground plane. In the following, we first describe the details of the existing tracking-by-detection and crowd simulation approaches used in our system, and then propose our advancements that integrate them together.

Table 1 summarizes the important notations used in the following description. Note that in this section, the subscript and superscript (e.g., the i and t indicating particle index and time step in w_i^t) are explicitly expressed only when necessary.

3.1. Analysis and synthesis approaches

This section is mainly focused on the description for the current analysis and synthesis approaches, i.e., the tracking-by-detection and crowd simulation approaches that serve as foundations of the proposed approach.

3.1.1. Video analysis – tracking-by-detection

The frame tracker in this approach is a tracking-by-detection method that combines a particle filter, a boosting classifier, and a human detector [8]. The main components that are used in our system are described below. For readers who are interested in further details, please refer to [8].

Each frame tracker corresponds to one pedestrian, and is constructed mainly based on a *bootstrap filter*, with N_p particles. For each particle, the state $\mathbf{x} = \{x, y, u, v\}$ is maintained by position (x, y) and velocity (u, v). This particle filter adopts a simple constant velocity motion model

$$(x,y)^t = (x,y)^{t-1} + (u,v)^{t-1} + \varepsilon_{(x,y)} \tag{1}$$

$$(u,v)^t = (u,v)^{t-1} + \varepsilon_{(u,v)} \tag{2}$$

where $\varepsilon_{(x,y)}$ and $\varepsilon_{(u,v)}$ are two zero-mean normal distributions as noise terms. The variances $\sigma^2_{(x,y)}$ and $\sigma^2_{(u,v)}$ for these two noise functions are initially set proportional to the size of the tracking patch, and then gradually decrease as the number of successfully tracked frames increases. At every time step t, importance resampling is carried out which makes the particle weights $w_i^{t-1} = 1/N$, so w_i^t only depends on the observation from the current frame, which can be calculated based on the observation model

$$w_{tr,p} = \beta \mathcal{I}(tr)\mathcal{N}(p - d_o^*) + \gamma d_c(p) P_0(tr) + \eta c_{tr}(p) \tag{3}$$

For each particle p in the tracker tr, the first term in Eq. (3) contributes the zero mean normal estimation based on the distance between the particle and the associated detection d_o^* if there is one. The second term is the confidence value, which is obtained at the particle position (x, y) from the confidence map d_c. The third term is the classifier evaluation result for the patch located at p. After the weight for each particle is calculated, the weights for all particles are normalized so that they sum to 1.

The human detector provides detection output and confidence to the first and second terms in Eq. (3), respectively. For detection results, since they are usually calculated with high confidence, we will try to associate each of them to a frame tracker, which is represented as the indicator function $\mathcal{I}(tr)$ in the equation. For a tracker tr, the indicator function is equal to 1 if this tracker is associated to a detected patch and is set to 0 otherwise. These associations are determined based on the pair-wise matching scores between frame trackers and detections using the following equation

$$m(tr, d_o) = g(tr, d_o) \cdot \left(c_{tr}(d_o) + \alpha \cdot \sum_{p \in tr}^{N_p} \mathcal{N}(d_o - p) \right) \tag{4}$$

where tr and d_o denote the positions of the frame tracker and the detection output, respectively, $\mathcal{N}(d_o - p)$ is a normal distribution based on the distance from the detection to each particle in the frame tracker, $c_{tr}(d_o)$ is the classifier evaluation for the detected patch (described later), α is a parameter balancing the contribution from classifier evaluation and distance based evaluation, and $g(tr, d_o)$ is a gating function estimating the relationships between the detection and the frame tracker based on their spatial relationship. It is calculated as

$$\begin{aligned} g(tr, d_o) &= P(size_{d_o} | size_{tr}) \cdot P(pos_{d_o} | pos_{tr}) \\ &= \begin{cases} \mathcal{N}\left(\frac{size_{tr} - size_{d_o}}{size_{tr}}\right) \cdot \mathcal{N}(|d_o - tr|), & \text{if } |V_{tr}| < \tau_v \\ \mathcal{N}\left(\frac{size_{tr} - size_{d_o}}{size_{tr}}\right) \cdot \mathcal{N}(dist(d_o, V_{tr})), & \text{otherwise} \end{cases} \end{aligned} \tag{5}$$

where *size* is measured as the height of the bounding box, V_{tr} is the velocity of the pedestrian estimated by the frame tracker and $dist(d_o, V_{tr})$ is the distance from the detected position (a point) to the velocity vector (a line). The distribution $\mathcal{N}(dist(d_o, V_{tr}))$ has a shape similar to a 2D cone, which means that the future position of the tracker should generally follow the current direction when its speed has exceeded a certain threshold τ_v. Accordingly, the matching score is related to the spatial relationship between the frame tracker (including its particles) and the detection, as well as their similarity in feature space. With all the pair-wise matching scores computed for every tracker–detection pair, each detected patch is associated to a frame tracker by a greedy algorithm, as long as the pair has a matching score greater than a certain threshold τ.

At each time step, the confidence map is computed as the intermediate result by the detector and scaled to $[0, 1]$ (e.g., Support Vector Machine output in the traditional Histogram of Oriented Gradient (HOG) based human detector [31]). In addition, for the situation when the detection fails because of occlusions, P_0 is used as the *interobject occlusion reasoning*, which is defined as

$$P_0(tr) = \begin{cases} 1, & \text{if } \mathcal{I}(tr) = 1 \\ \max\limits_{tr': \mathcal{I}(tr')=1} \mathcal{N}(tr - tr'), & \text{else if } \exists \mathcal{I}(tr') = 1 \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

The classifier in this frame tracker is based on the online Ada-Boost classifier in [13], which contributes as the third term in Eq. (3). For each frame tracker (pedestrian), an associated classifier is trained and maintained based on the features of the tracked patch. The classifier is initialized when the tracker is created, using the patch information at that frame. The positive sample comes from the initial patch and the negative samples are the nearby patches within a certain range (including background). During tracking, a pool of weak classifiers are maintained and updated, and the ones with best performance are selected to form the current strong classifier. Given the features extracted from a particular patch, the classifier is able to evaluate this patch based on the trained model for the corresponding pedestrian. At each time step, when the new patch location for the pedestrian has been computed, the classifier updates itself using the features extracted from the new patch.

3.1.2. Computer synthesis – crowd simulation

We have used two crowd simulators in this approach. The first one is a global optimization approach called RVO2 library [10], and the second one is based on the modeling of social behavior for a single pedestrian [11,12]. As mentioned in Section 2.2, the Social Behavior Model provides a more realistic strategy for collision avoidance, which is expected to yield better tracking performance. In the following we explain the differences of the two simulators in detail, and examine how they impact the overall performance of the multi-camera tracking system.

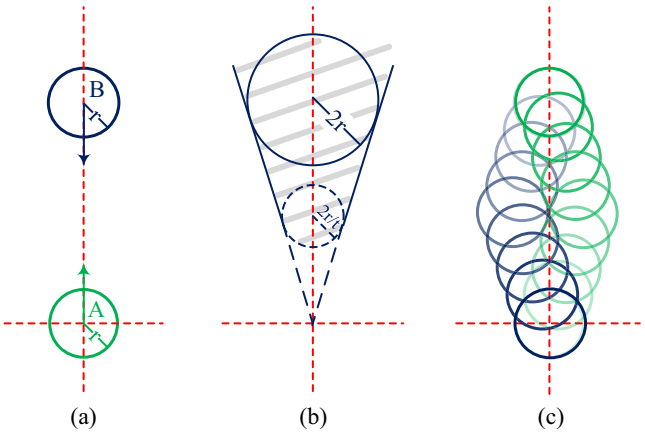


Fig. 2. A simple example for the RVO2 library. (a) The system contains two pedestrians *A* and *B*, walking face-to-face along a line. (b) The velocity obstacle $VO^{T}_{A|B}$ of pedestrian *A* when *B* is not moving (shadow part). (c) One of the possible trajectories that the RVO2 library generates.

3.1.2.1. RVO2 library. This model solves an optimization problem based on the strategy named Optimal Reciprocal Collision Avoidance (ORCA) [10]. It is computationally efficient and only requires the information about the current position and desired velocity of each pedestrian.

The RVO2 library introduces a concept, namely velocity obstacles, with its definition as

$$VO^{T}_{A|B} = \{\mathbf{v} | \exists t_{\mathbf{v}} \in [0,T] : \mathbf{v} \cdot t_{\mathbf{v}} \in C(\mathbf{p}_B - \mathbf{p}_A, \mathbf{r}_A + \mathbf{r}_B)\} \tag{7}$$

where $\mathbf{p}_A$ and $\mathbf{p}_B$ are the positions for two pedestrians and $\mathbf{r}_A$, $\mathbf{r}_B$ are their radii. $C(\mathbf{p},\mathbf{r})$ indicates a circle centered at $\mathbf{p}$ with radius $\mathbf{r}$. Basically, $VO^{T}_{A|B}$ defines the set of relative velocities of *A* with respect to *B* which will cause a collision in the future within a time period $[0,T]$. Therefore, finding a solution for the collision avoidance equals to finding a set of velocities of pedestrians that are closest to the desired velocities, while none of them falls into the velocity obstacle set. Fig. 2 shows a simple example with two pedestrians. Fig. 2a illustrates the system configuration: two pedestrians with radius $\mathbf{r}$ are walking face-to-face along a line and want to swap their positions. The cone-like shape in shadow in Fig. 2b is the velocity obstacle of pedestrian *A* when *B* is not moving, which contains the velocities that pedestrian *A* cannot take to avoid potential collisions within time *T* (that is why it is called "obstacle"). This velocity obstacle moves as *B* moves. To avoid collision, the RVO2 library will find a solution such that each of the two pedestrians needs to take some effort but it is globally optimal. In this case, each of them turns right to a certain degree (or even changes speed as well, depends on the setting), so that their circles will be tangent to each other at some time in the future, as shown in Fig. 2c. This global optimal solution is computed efficiently using linear programming.

3.1.2.2. Social Behavior Model. The Social Behavior Model computes an energy function for each pedestrian, according to the current status (position and velocity) of all pedestrians. The first version of the Social Behavior Model is proposed by Pellegrini et al. [11], which comes from the social force model proposed in [26]. Yamaguchi et al. [12] further add group information to the model, which makes it more sophisticated. In our proposed approach, we use the version without group constraints [11] so that it is comparable to the RVO2 library.

In this model, each pedestrian *i* is treated as a particle with real-world position $\mathbf{p}_i^t$ and velocity $\mathbf{v}_i^t$ ($\mathbf{p}_i^t$ and $\mathbf{v}_i^t$ here are vectors). The whole crowd is then considered as a particle system. The main factors that influence the trajectory for each pedestrian are represented by energy functions. The collision avoidance for a particular pedestrian *i* is achieved by the interaction energy between this particle and all other particles. The interaction energy between particles *i* and *j* is calculated as

$$\mathbf{E}^{\mathbf{I}}_{ij}(\mathbf{v}) = e^{-\frac{d^{*2}_{ij}(\mathbf{v})}{2\sigma_d^2}} \tag{8}$$

where $\mathbf{v}$ is the candidate velocity of pedestrian *i*, and $d^{*2}_{ij}(\mathbf{v})$ denotes the smallest squared distance between pedestrians *i* and *j*. The squared distance between two pedestrians can be computed as

$$d^2_{ij}(\mathbf{t},\mathbf{v}) = \left\| \left(\mathbf{p}_i^t - \mathbf{p}_j^t\right) + \mathbf{t} \cdot \left(\mathbf{v} - \mathbf{v}_j^t\right) \right\|^2 \tag{9}$$

Thus, the time for the minimum squared distance can be obtained by computing the derivative of d^2_{ij} with respect to $\mathbf{t}$ and set it to zero, resulting in

$$\mathbf{t}^* = \max\left(-\frac{\left(\mathbf{p}_i^t - \mathbf{p}_j^t\right) \cdot \left(\mathbf{v} - \mathbf{v}_j^t\right)}{\left\|\mathbf{v} - \mathbf{v}_j^t\right\|^2}, 0 \right) \tag{10}$$

Therefore, the minimum squared distance is

$$d^{*2}_{ij}(\mathbf{v}) = \left\| \left(\mathbf{p}_i^t - \mathbf{p}_j^t\right) + \mathbf{t}^* \left(\mathbf{v} - \mathbf{v}_j^t\right) \right\|^2 \tag{11}$$

In addition, for different pedestrians, different weights are assigned to them depending on their spatial relationship to pedestrian *i*:

$$\mathbf{W}_{ij} = e^{-\frac{\left\|\mathbf{p}_i^t - \mathbf{p}_j^t\right\|^2}{2\sigma_w^2}} \left(\frac{1}{2}(1+\cos\phi)\right)^{\psi} \tag{12}$$

where ϕ is the angle between the vector $\mathbf{v}$ and the vector $\Delta\mathbf{p}_{ji} = \mathbf{p}_j^t - \mathbf{p}_i^t$. So the first half of the weighting function represents the spatial relationship between pedestrians *j* and *i* in distance, and the second half denotes the relationship between pedestrian *j* and the field-of-view of pedestrian *i*. Here ψ controls the "peakiness" of weighting function in the field-of-view. Note, the field-of-view of each pedestrian is restricted to ±90°, that is, the weight $\mathbf{W}_{ij}$ is set to 0 when $|\phi| > \pi/2$. Therefore, for pedestrian *i*, pedestrians that are close to him/her as well as located right on his/her walking way will obtain higher weights.

The overall interaction energy for the pedestrian *i* is the weighted sum of the pair-wise interaction energy:

$$\mathbf{E}^{\mathbf{I}}_i(\mathbf{v}) = \sum_{j \neq i} \mathbf{W}_{ij} \mathbf{E}^{\mathbf{I}}_{ij}(\mathbf{v}) \tag{13}$$

Another two factors that control the trajectory are related to the desired velocity $\mathbf{v}^*$: the speed energy and the direction energy. They are defined as

$$\mathbf{E}^{\mathbf{S}}_i(\mathbf{v}) = \left(\|\mathbf{v}^*\| - \|\mathbf{v}\|\right)^2 \tag{14}$$

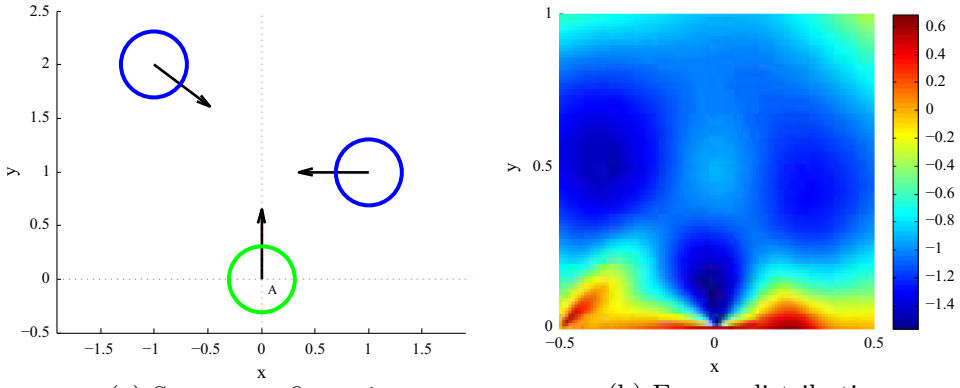
$$\mathbf{E}^{\mathbf{D}}_i(\mathbf{v}) = -\frac{\mathbf{v}^*}{\|\mathbf{v}^*\|} \cdot \frac{\mathbf{v}}{\|\mathbf{v}\|} \tag{15}$$

So the total energy of a pedestrian *i* with velocity $\mathbf{v}$ is

$$\mathbf{E}_i(\mathbf{v}) = \mathbf{E}^{\mathbf{I}}_i(v) + \lambda_s \mathbf{E}^{\mathbf{S}}_i(v) + \lambda_d \mathbf{E}^{\mathbf{D}}_i(v) \tag{16}$$

The optimal velocity $\mathbf{v}'$ is the one that minimizes the above equation. To make smooth pedestrian trajectories, the update of velocity is a weighted sum of the old and new velocities: $\mathbf{v}_i^{t+1} = \alpha_s \mathbf{v}_i^t + (1-\alpha_s)\mathbf{v}_i'$. Fig. 3 demonstrates a system with three pedestrians as well as the energy distribution of pedestrian *A*. It is revealed that the optimal velocity $\mathbf{v}'$ is not the current velocity of *A*.

In the idea of Social Behavior Model, each pedestrian makes a decision upon the assumption that all other pedestrians keep their previous trajectories. In addition, the pedestrian treats other pedestrians that may cause collisions differently, depending on their distances to the pedestrian as well as the displacements between them. A nearby pedestrian located right in front of him/her will



(a) System configuration. (b) Energy distribution.

Fig. 3. A figure illustrating the result of the energy function in the Social Behavior Model. (a) A system contains three pedestrians, with positions $(0,0)$, $(1,1)$, $(-1,2)$, and velocities $(0,0.5)$, $(-0.5,0)$, $(0.4,-0.3)$ respectively. (b) The energy calculated for pedestrian A, with velocity range as $\mathbf{v}_u = [-0.5, 0.5]$ and $\mathbf{v}_v = [0,1]$.

obtain the most attention compared to others. So if two pedestrians walking face-to-face along a line, their trajectories generated by Social Behavior Model will not be as straight as those generated by the RVO2 library, and the circles within which the two pedestrians are located may not be tangent at some time in the future.

3.2. Integrating analysis and synthesis for tracking

As mentioned in Section 1, the integration of analysis and synthesis better utilizes the temporal and spatial relationship between pedestrians during the tracking process. To achieve this, these procedures have to be added at each time step: (1) Generates the input for the crowd simulator, including the estimation of desired velocity for each pedestrian; (2) Project the simulated real-world locations for each pedestrian to all camera views based on homography matrices; (3) After the tracking is done for each camera view, the real-world location for each pedestrian is obtained and updated. As a result, a global tracker is designed with two functions: (1) communicate between frame trackers and the crowd simulator; (2) estimate the desired velocity for each pedestrian. In addition, the frame tracker needs modification in order to use information from the crowd simulator. In the following, the two functions of the global tracker as well as the modification for the frame tracker are described.

3.2.1. Interactions between frame trackers and crowd simulator

An important function of the global tracker is to communicate between the frame trackers and the crowd simulator. Therefore, the transformation between the points on the ground plane and the points on each camera view needs to be recovered. This is done based on the homography matrix between the ground plane each view (image plane), which is defined as a 3×3 matrix

$$H_v = \begin{bmatrix} h_{11}^v & h_{12}^v & h_{13}^v \\ h_{21}^v & h_{22}^v & h_{23}^v \\ h_{31}^v & h_{32}^v & 1 \end{bmatrix} \tag{17}$$

The computation for the homography matrices is done by manually selecting four corresponding points from different views, as well as from the real-world ground plane. With the homography matrix H_v, each point (x_g, y_g) on the ground plane can be transformed to (x, y) on the image plane following the equation

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = H_v \begin{bmatrix} x_g \\ y_g \\ 1 \end{bmatrix} \tag{18}$$

To obtain the real-world location for each pedestrian, the inverse of the homography matrix H_v is necessary. Based on the positions of one pedestrian on different camera views, the position of this pedestrian on the ground plane is determined using the principal-axis based integration. The principal axis of a pedestrian is defined as the line connecting the head and the feet. For simplicity, however, in our approach, we use the vertical line in the middle of the patch instead. The principal-axis based approach projects the principal axis of the same pedestrian from each view onto the ground plane and calculates the intersection of them. If the principal axes of the pedestrian in all the views are perfect, then the intersection points of their projection should converge to a single point, which is exactly the position of the pedestrian on the ground plane. Although in most cases this perfect intersection cannot be achieved, it can be proven that the principal axis-based integration is very robust and useful [2,21].

For each pair of views c_i and c_j, the intersection point $\mathbf{I}_{ij}$ of the principal axes is obtained. If there are more than two views, a set of intersection points $\mathbb{I}$ can be collected. Since $\mathbf{I}_{ij} = \mathbf{I}_{ji}$, only the intersections from views c_i and c_j with $i < j$ are computed. So finally the set $\mathbb{I} = \{\mathbf{I}_{ij}\}(i < j)$ has $N_c(N_c - 1)/2$ points in total (with N_c views). When the intersection points do not converge to a single point, we simply average all the $\mathbf{I}_{ij}$'s to estimate the pedestrian position $\mathbf{p}$. The same strategy applies to the velocity estimation on the ground plane.

3.2.2. Velocity estimation for crowd simulation

Since crowd simulation requires information for the desired velocity for each pedestrian when it predicts pedestrians' future locations, it is important for the global tracker to provide this information to the crowd simulator. However, compared to the real-world position and velocity information which is quite easy to compute in a multi-camera tracking system, it is difficult to obtain the desired velocity directly due to the uncertainty of human behavior in the future. That is, for most of the pedestrians, their preferences of walking direction and speed (especially the direction) cannot be exactly obtained based on the current information. Therefore, we have to find an alternative way to estimate the desired velocity. Our solution is based on using the historical information instead. For each pedestrian, we use Monte Carlo simulation based on his/her trajectory to estimate the desired velocity in the near future. The first step is to calculate the derivatives (accelerations) and their weights from the last N_h frames for a pedestrian k. The calculated set is defined as $\mathbf{A}_k^t = \left\{\mathbf{a}_k^{t-N_h}, \mathbf{a}_k^{t-N_h+1}, \ldots, \mathbf{a}_k^{t-1}\right\}$ ($\mathbf{a}_k^{t-i}$ here is the velocity differentiation for pedestrian k at the last ith frame), as well as the corresponding weights

$$\mathbf{w}_k^{t-i} = \frac{N_h - i + 1}{\sum_{j=1}^{N_h} j}, \quad i = 1, \ldots, N_h \tag{19}$$

Table 2
The MOTP and MOTA evaluations on camera views and ground plane for S2.L1.

	Evaluation criterion	Approach [9] with RVO2 (%)	With simulator		Without simulator (%)
			RVO2 (%)	SBM (%)	
View 1	MOTP	75.25	76.69	76.49	75.43
	MOTA	81.93	85.60	88.86	84.69
View 5	MOTP	71.98	72.06	72.89	72.48
	MOTA	82.36	84.74	85.82	82.68
View 7	MOTP	73.87	74.61	74.98	73.54
	MOTA	82.72	85.43	85.77	79.49
Ground plane	MOTP	75.03	79.92	80.46	78.49
	MOTA	80.62	83.90	86.36	81.22

Table 3
The MOTP and MOTA evaluations on camera views and ground plane for S2.L2.

	Evaluation criterion	Approach [9] with RVO2 (%)	With simulator		Without simulator (%)
			RVO2 (%)	SBM (%)	
View 1	MOTP	70.31	71.18	71.53	67.83
	MOTA	58.60	62.41	67.56	41.77
View 2	MOTP	70.76	70.43	71.39	68.73
	MOTA	54.69	59.78	63.79	49.31
Ground plane	MOTP	69.85	78.39	78.43	71.92
	MOTA	32.01	35.67	37.77	12.92

The assignment of the weights follows a strategy that the more recent acceleration is more important, thus it gets a higher weight (Eq. (19)). After that, a Monte Carlo simulation process is carried out on the acceleration set using these weights, which results in a set of N_a accelerations $\mathbf{A}_k^{t'} = \left\{\mathbf{a}'_{k,1}, \mathbf{a}'_{k,2}, \ldots, \mathbf{a}'_{k,N_a}\right\}$ $\left(\mathbf{a}'_{k,i}$ is one of $\mathbf{A}_k^t\right)$. Then the desired velocity set is calculated as

$$\mathbf{v}'_{k,i} = \mathbf{v}_k + \mathbf{a}'_{k,i} + \varepsilon_{\mathbf{a},k} \tag{20}$$

where $\mathbf{v}'_{k_i}$ is the estimation of desired velocity and $\mathbf{v}_k$ is the current velocity. $\varepsilon_{\mathbf{a},k}$ is a zero-mean normal distribution with variance equal to $\sigma\left(\mathbf{A}_k^{t'}\right)$.

Finally, to reduce the computation of the crowd simulation, not all possible combinations of velocities are exploited. Instead, we pick up the velocities with the same index (i.e., i) from the estimated velocity set for each pedestrian so that N_v sets of velocities $\mathbf{V}_i = \{\mathbf{v}_{1,i}, \mathbf{v}_{2,i}, \ldots\}$ are formed. We then use these N_v sets of desired velocities as the input to the crowd simulator, so the total number of possible locations estimated, i.e. the number that the crowd simulation repeats at each time step, is N_v. Thereafter, the crowd simulator repeats N_v times with the same initial position and velocity information to generate N_v sets of predictions of future positions for all the pedestrians. For each pedestrian, these positions are projected back to each view c_i as a distribution of possible frame locations L_{tr}^i for the particular frame tracker.

3.2.3. Integrated observation model for tracking-by-detection

To integrate the crowd simulation predictions into the frame tracker described in Section 3.1.1, we modify the observation model for the tracking-by-detection. Basically, at each time step, after the global tracker transforms the predicted positions for each pedestrian from the ground plane to each camera view, a distribution of these predicted positions can be obtained on each camera view. This distribution is represented by a set of possible locations L_{tr}, which are transformed from the ground plane based on the homography matrix. It is then used in the integrated observation model to estimate particle weights. For each particle p in a tracker tr, its weight with respect to the simulated predictions can be computed as

$$s_{tr}(p) = \frac{1}{|L_{tr}|}\sum_{l\in L_{tr}} \mathcal{N}(p-l) \tag{21}$$

where l is one of the simulated position on the frame that has been transformed from the ground plane.

Thereafter, the observation model in Eq. (3) is modified such that it has an additional term that represents the influence from the crowd simulation

$$w_{tr,p} = \beta \mathcal{I}(tr)\mathcal{N}(p-d_o^*) + \gamma d_c(p)P_0(tr) + \eta c_{tr}(p) + \delta s_{tr}(p) \tag{22}$$

where δ is the coefficient for this extra simulation term.

According to this observation model, the weight of each particle for each frame tracker depends on the detection result and the confidence map generated by the human detector, the evaluation result from the boosting classifier, as well as the predicted locations from the crowd simulator. Therefore, the modified observation model takes advantage of both existing tracking-by-detection approaches based on detection results [16–19] and tracking-by-classification approaches based on classifier evaluation [13–15], as well as further help from the detection confidence map and crowd simulation. To integrate and balance the contributions from the four components, parameters β, γ, η, and δ are introduced (the selection of their values is described later in Section 4.1.2.1). Moreover, compared to our previous work in [9], the modification for the observation model further avoids the possible loss of information brought by the sampling for the candidate locations on the ground plane. This might increase the performance even under the same frame tracker and crowd simulator

Table 4
The influences of different δ values.

$\delta : \eta$		0.5 (%)	1 (%)	2 (%)	5 (%)
RVO2	MOTP	81.80	80.97	82.06	81.07
	MOTA	82.43	84.17	85.91	84.36
SBM	MOTP	81.69	81.68	81.98	81.41
	MOTA	83.20	85.18	87.57	86.70

implementation. The pseudo-code for the updated frame trackers is summarized in Algorithm 1.

Algorithm 1. Frame tracker update

Data: Frame F, Detections D_o, Confidence d_c, Simulations L_v
foreach *tracker* tr **do**
  foreach *detection* $d_o \in D_o$ **do**
    Calculate matching score $m(tr, d_o)$;
  end
end
Detection association;
foreach *tracker* tr **do**
  Particle transition;
  foreach *particle* $p \in tr$ **do**
    Update particle weight based on d_o, d_c, L_{tr} (Equation (22));
  end
  if *associated*$(tr,\ d_o \in D_o)$ **then**
    Update tracker size;
    Update classifier c_{tr};
  end
  Particle resampling;
end

3.2.4. Pseudocode of the proposed approach

The pseudo-code for the entire simulation integrated tracking system is summarized as Algorithm 2.

Algorithm 2. Multi-camera tracking

while *tracking continues* **do**
  {Position set $\mathbb{P}$, velocity set $\mathbb{V}$}←Historical info;
  Desired velocity set $\mathbb{V}_d$ ←Estimation from historical info;
  Simulated location set $\mathbb{L}$ ←Simulation$(\mathbb{P}, \mathbb{V}, \mathbb{V}_d)$;
  foreach *View* c_i *(frame* F*)* **do**
    L_v ←Perspective transformation$(\mathbb{L})$;
    {Detection set D_o, confidence map d_c}←Detection(F);
    Update frame tracker(F, D_o, d_c, L_v) (Algorithm 1);
  end
  {Position $\mathbb{P}'$, velocity $\mathbb{V}'$}←Intersection based on frame trackers;
  Update historical info$(\mathbb{P}', \mathbb{V}')$; Initialize pedestrians if necessary;
  Remove pedestrians if necessary;
end

4. Experimental results

In this section, we first provide the details of our experimental setting, including the datasets tested, the implementation details, and the parameter values. Then we present the experimental

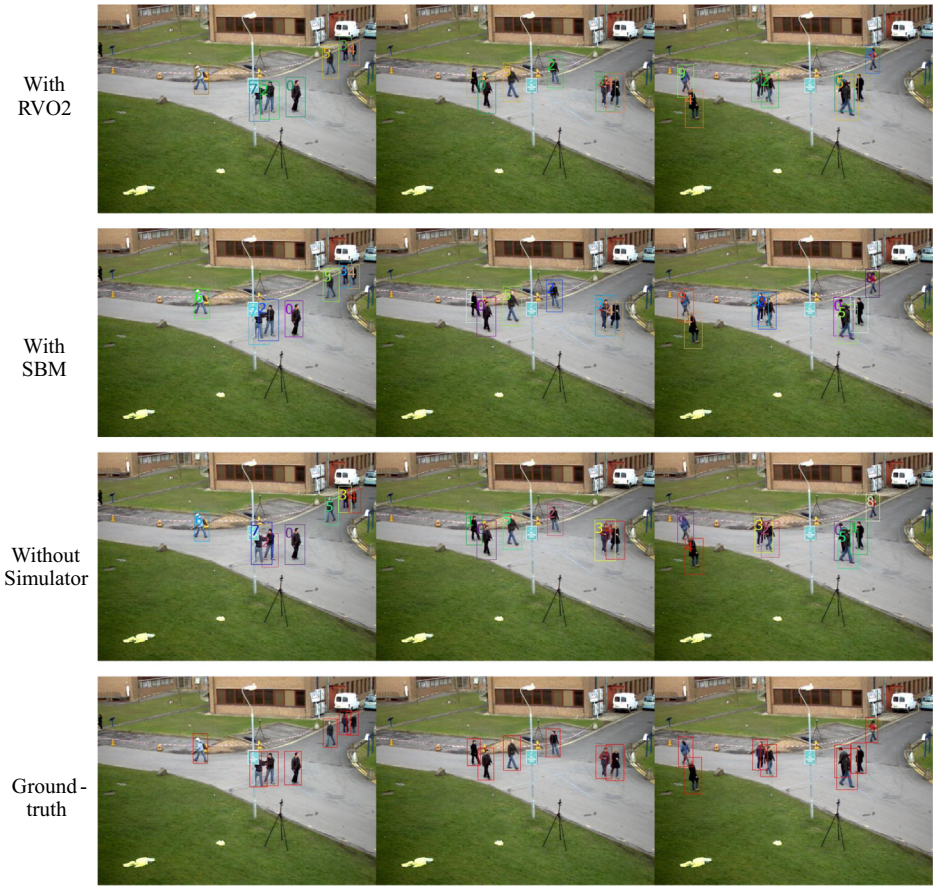


Fig. 4. Some sample frames from S2.L1 View 1. The three columns correspond to frames #171, #330, and #729. The first row shows the results from the multi-camera tracker with RVO2 library, the second row shows the results with Social Behavior Model, the third row shows the results without simulator, and the last row shows the ground-truth. Different trackers are shown in different colors. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

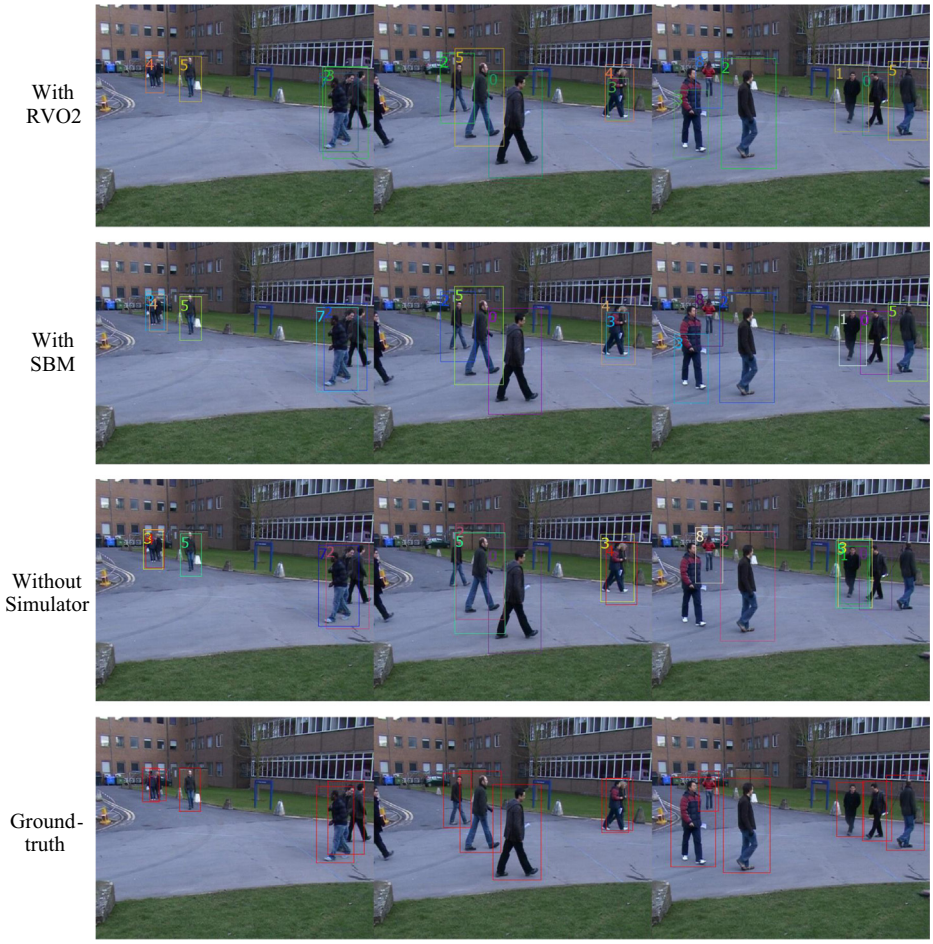


Fig. 5. Some sample frames from S2.L1 View 5. The three columns correspond to frames #171, #330, and #729. The first row shows the results from the multi-camera tracker with RVO2 library, the second row shows the results with Social Behavior Model, the third row shows the results without simulator, and the last row shows the ground-truth. Different trackers are shown in different colors. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

results and related discussions. We implement the entire system using C++ in conjunction with OpenCV.

4.1. Experimental setting

4.1.1. Dataset

Our experiments are conducted on the PETS 2009 dataset since it has the most challenging scenarios. The sequences that we choose are S2.L1 and S2.L2 with different crowd densities.

For sequence S2.L1, the number of people in the scene varies from 3 to 9 (low crowd density). Among the 7 views provided, View 1, 5, and 7 are used for testing (795 frames for each view). For sequence S2.L2, the number of people in the scene is between 10 and 36 at different times (medium crowd density). There are data from four different views for this sequence. We decided to use only the images from View 1 and View 2 (436 frames for each view).

To calculate the homography matrices, we manually mark four points on the ground plane as well as the corresponding points on each view and use the existing function called *findHomography* in OpenCV. The image of the ground plane is obtained from Google Maps, which is provided as part of the data.

The ground-truth is obtained by manual annotation. For each pedestrian, the bounding-box is annotated every five frames, and the annotations for in-between frames are obtained through linear interpolation and validated manually later. Subsequently, the principal axis is determined as the vertical line in the middle of the bounding box. The ground-truth of the position of a pedestrian on the ground plane is computed by intersecting the projections of principal axis on the ground plane. It is done for all the frames in both sequences for all the pedestrians.

4.1.2. Implementation details

Since we are more focused on investigating the effectiveness of the integration of crowd simulation, we have made some modifications in the original frame tracker approach to simplify the system. In the original tracker design [8], there is a procedure called *Iterative Likelihood Weighting* which deals with abrupt and fast camera motion. It is not adopted in our implementation because the scenes of PETS 2009 are all captured using steady cameras. In addition, the same pedestrians appear in the scene again are treated as new pedestrians (re-initialized) for simplicity. Furthermore, the size and the position of the initial patch for each tracker are assigned directly from manual annotation to ensure that the initialization does not bring any potential errors, since human detectors usually do not work very well in a crowded scene, especially for occluded pedestrians. The termination of a tracker occurs when it is not associated to any detection results for more than 10 frames, which still follows the original design. The original approach [8] applied two different human detectors: Implicit Shape Model (ISM) [32] and Histogram of Oriented Gradient (HOG) [31]. In our implementation, only HOG detector is kept because ISM is trained for side-views [8] and HOG is more generalized, thus more suitable for the PETS 2009 dataset.

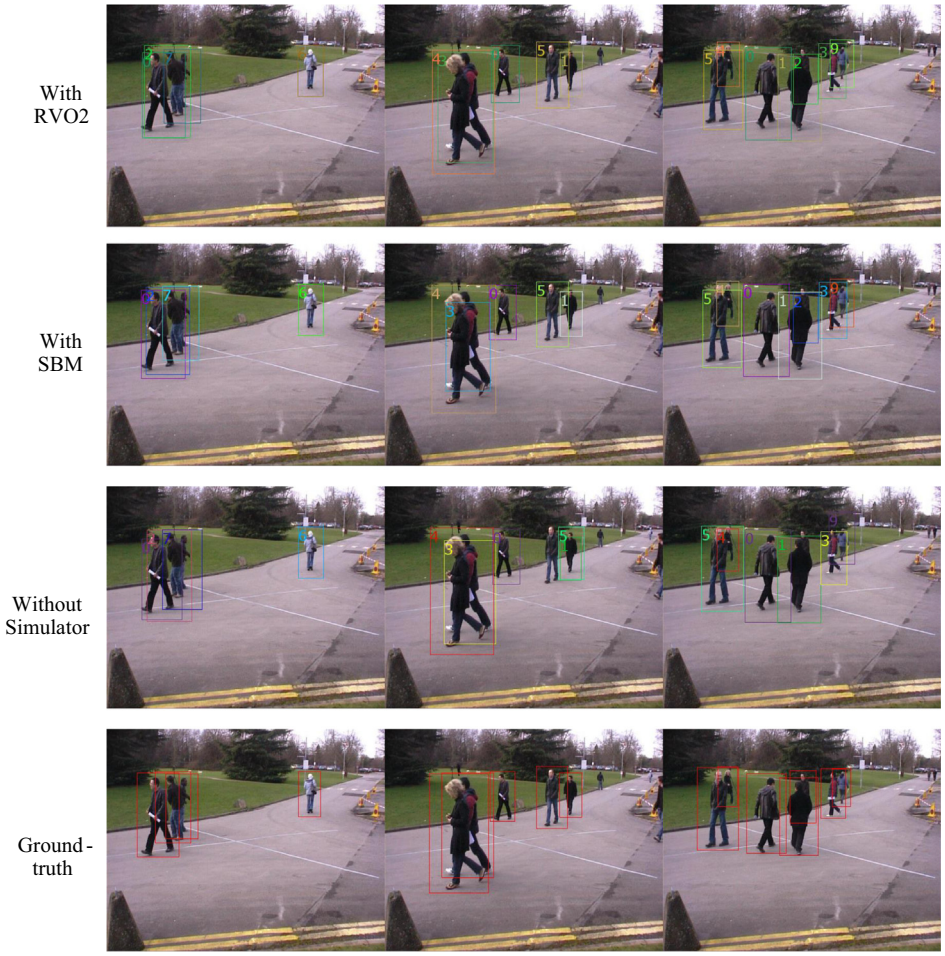


Fig. 6. Some sample frames from S2.L1 View 7. The three columns correspond to frames #171, #330, and #729. The first row shows the results from the multi-camera tracker with RVO2 library, the second row shows the results with Social Behavior Model, the third row shows the results without simulator, and the last row shows the ground-truth. Different trackers are shown in different colors. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Another difference from the original approach [8] is the association between frame trakers and detections. After the calculation of the matching scores between the detections and frame trackers, the pair-wise association does not follow the original work [8]. Instead, we first set the scores smaller than a certain threshold τ to 0, and then determine the optimal association using the Hungarian algorithm [33]. The Hungarian algorithm is a standard approach that is able to find the globally optimal matching solution in $O(n^3)$ running time where n is the size of the tracker set. Compared to the greedy strategy mentioned in the original approach [8], it is able to provide a better performance since it gives a globally optimal solution with just a little more computational cost.

One last point worth mentioning concerns the communication between frame trackers and crowd simulator. Since the ground location for each pedestrian is computed based on multiple views, the small offset for the tracking in one view may result in a dramatic change of the pedestrian location on the ground plane. So during the estimation of desired velocity for each pedestrian, those dramatic acceleration changes will then have significant influence. To solve this problem, an acceleration of a particular pedestrian with its norm greater than the standard deviation of all the historical accelerations of this pedestrian will not be taken into account (i.e., set their weights to 0) in the Monte Carlo simulation process.

In the following, we provide a detailed description on all the parameters in this system. Note that these parameters are kept the same when the program runs on different datasets (sequences).

4.1.2.1. Frame tracker parameters. There are several parameters in the tracker [8], including the coefficients β, γ, η, δ in the observation model (Eq. (22)), as well as the variances of the zero mean normal distributions during tracker initialization. For all the parameters coming from the original tracking model, their setting follows the original work. For instance, we set $\beta : \gamma : \eta = 20 : 2 : 1$. Therefore, only the parameter related to the simulation prediction, which is the last term in the observation model (Eq. (22)) needs to be determined in the experiment. Based on empirical tests (Table 4), δ is set to 2η. The variance in Eq. (21) is the same as the variance of $\varepsilon_{(x,y)}$ in Eq. (1).

The HOG detector adopted in our experiments comes from OpenCV. The detector has a trained model with the size as 64×128, which is also the smallest detection size. Therefore, in order to detect the pedestrians with smaller appearances in various frames, we resize (using bicubic interpolation) the original frame to 2560×1920. In this case, the smallest pedestrian in a frame has a size comparable to the model size of the detector.

4.1.2.2. Crowd simulator parameters – RVO2 library. For the crowd simulator, the RVO2 library, there are 10 parameters. Among them, three parameters keep changing during tracking (current position, current velocity, and desired velocity) and are unrelated between any two individuals. The remaining seven parameters which need to be determined beforehand include: the time step of the simulation (δ_t), the maximal number of nearby pedestrians that can be observed by one pedestrian (O_n), the maximal observation distance

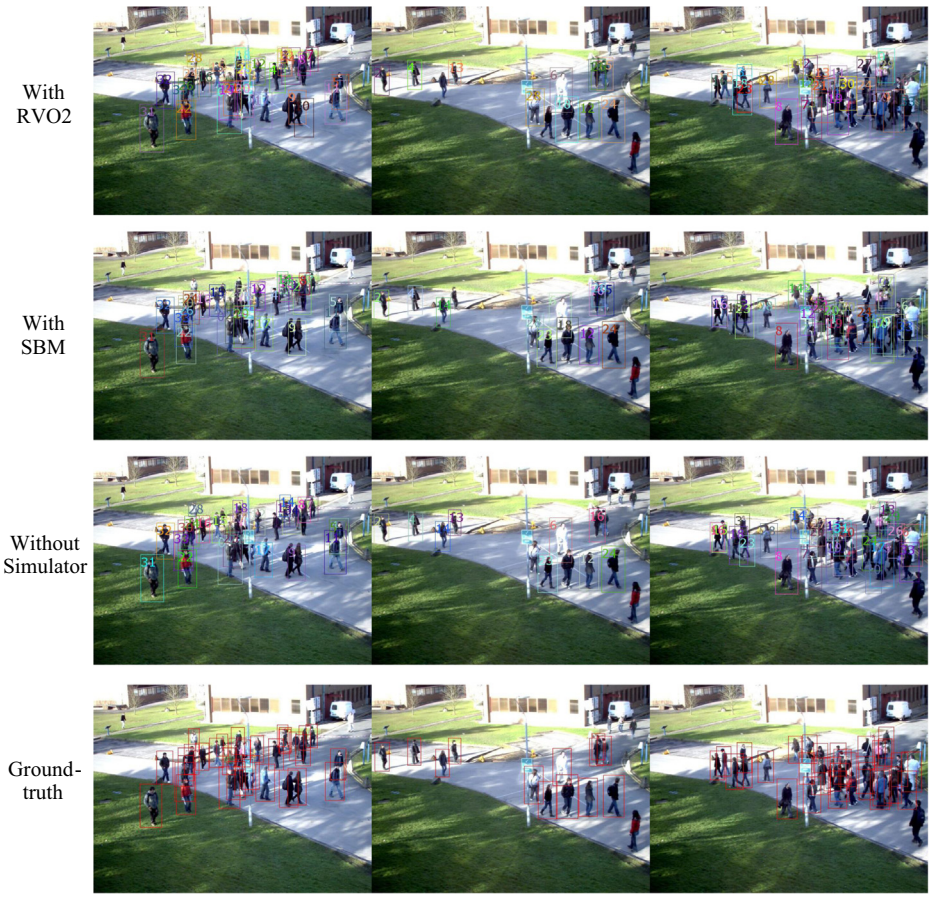


Fig. 7. Some sample frames from S2.L2 View 1. The three columns correspond to frames #65, #200, and #395. The first row shows the results from the multi-camera tracker with RVO2 library, the second row shows the results with Social Behavior Model, the third row shows the results without simulator, and the last row shows the ground-truth. Different trackers are shown in different colors. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

of a pedestrian (O_d), the maximal speed of a pedestrian (S_p), the radius (size) of a pedestrian ($\mathbf{r}_p$), the minimal amount of time set to be safe for any pedestrian w.r.t. others (T_p), and the minimal amount of time set to be safe for any pedestrian w.r.t. static obstacles (T_o). Except for the time step that needs to be the same for all individuals, the other six parameters can actually vary from individual to individual. However since most pedestrians should have similar behavior of walking, in our experiment we simply set all these parameters to be the same for all individuals. The time step δ_t is set to 0.14, which is the reciprocal of the frame rate. Similar to [11], the value of the remaining six parameters are optimized using a Genetic Algorithm [34], but the optimization is conducted on the UCSD crowd dataset [35]. The values used for these parameters in our experiments are $O_n = 23$, $O_d = 757.56$, $S_p = 10.07$, $\mathbf{r}_p = 18.06$, $T_p = 9.57$ and $T_o = 6.91$, which are the direct output from the Genetic Algorithm. According to the characteristics of Genetic Algorithm, these parameter values may not be the globally optimal, and actually small changes of these values are not able to result in large differences in tracking performance.

4.1.2.3. Crowd simulator parameters – Social Behavior Model. The parameter setting for this model follows the original work [11]. To elaborate, there are six parameters as discussed earlier in Section 3.1.2.2 and they are set as: $\sigma_d = 0.361$, $\sigma_w = 2.088$, $\lambda_s = 2.33$, $\lambda_d = 2.073$, $\psi = 1.462$, and $\alpha_s = 0.730$.

Since the parameters for the two crowd simulators are trained under two coordinate settings which use different measurement units, the scale of the ground plane in our experiments need to be decided appropriately. Therefore, we set the scales of the ground plane for the two crowd simulators according to their parameters related to the pedestrian size, so that the two simulators are able to work under the same coordinate settings as they are trained.

4.2. Quantitative results

4.2.1. Performance metrics

The metrics for quantitative evaluation of the performance are MOTP (multi-object tracking precision) and MOTA (multi-object tracking accuracy) [36]. The evaluations are conducted on View 1, View 2, as well as on the ground plane. However, the details differ from frame-based evaluation and the evaluation on the ground plane.

For each frame in each view, since we are tracking pedestrians using bounding boxes (rectangles), the *accuracy* of a tracker is defined as the overlapping ratio between the tracked patch and the corresponding ground-truth bounding box. This overlapping ratio is naturally a number in the range $[0, 1]$. Similar to the evaluation for the original frame tracker [8], we consider the tracking result as accurate when the overlapping ratio is above 0.5.

On the ground plane, however, both the tracking result and the ground-truth are not represented by rectangles. Therefore, although we still use MOTP and MOTA to measure the tracking performance, the definition for accuracy is different from the aforementioned frame-based evaluation. Instead, we consider the *accuracy* based on the distance between the two points. Let $\mathbf{r}_p$ denote

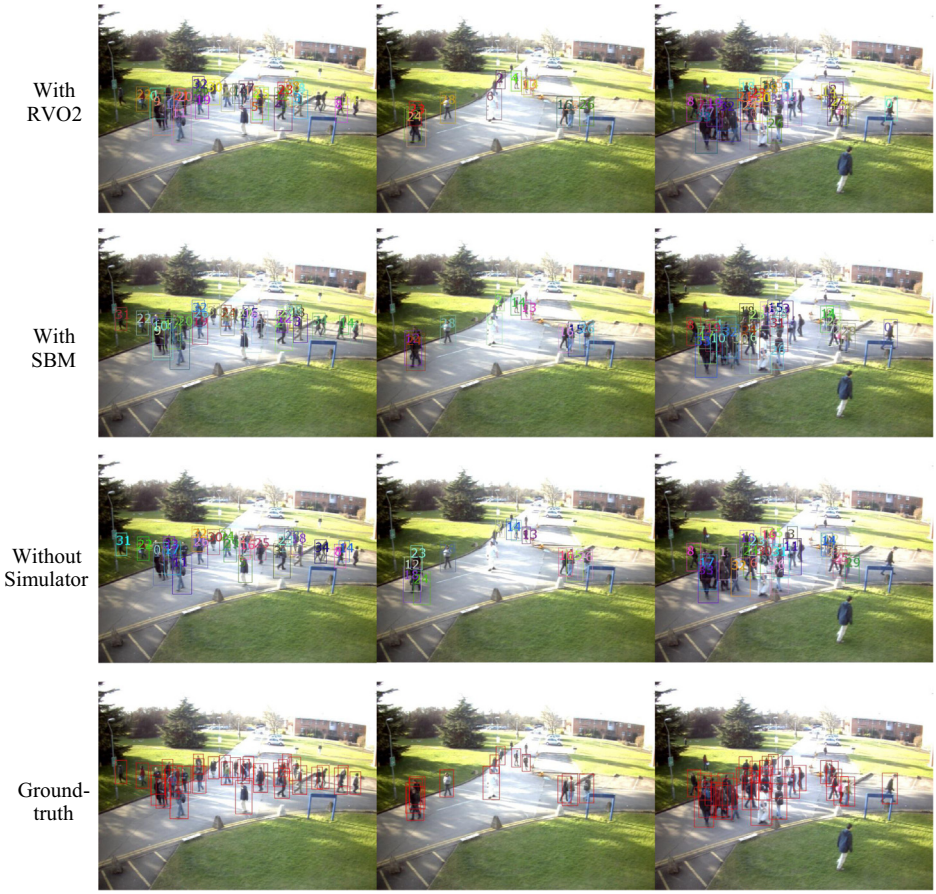


Fig. 8. Some sample frames from S2.L2 View 2. The three columns correspond to frames #65, #200, and #395. The first row shows the results from the multi-camera tracker with RVO2 library, the second row shows the results with Social Behavior Model, the third row shows the results without simulator, and the last row shows the ground-truth. Different trackers are shown in different colors. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

the radius within which a pedestrian is located on the ground plane. Let $\mathbf{p}_{tr}$ be the pedestrian position estimated from the tracking system and $\mathbf{p}_{gt}$ be the position obtained from the ground-truth, then the distance between the tracked point (tr) and the ground-truth (gt) is defined as

$$d(tr,gt) = \max\left(0, 1 - \frac{\|\mathbf{p}_{tr} - \mathbf{p}_{gt}\|}{4\mathbf{r}_{\mathbf{p}}}\right) \tag{23}$$

which is a number between 0 and 1. We then consider the tracking to be accurate when $d(tr,gt) > 0.5$, that is, when the distance between the tracked point and the ground-truth $\|\mathbf{p}_{tr} - \mathbf{p}_{gt}\|$ is smaller than the size of a pedestrian $2\mathbf{r}_{\mathbf{p}}$.

4.2.2. Results

Compared to other *state-of-the-art* multi-camera tracking approaches, our experimental setting has the following difference: (1) we annotate the ground-truth by ourselves since there is no public available annotations except for View 1; (2) the evaluation criteria for all views and ground plane are not the same; (3) the detector used in our work may be different or have different parameters from those in other work. Based on the experiments in [37], these differences in experiments may result in significantly different performance. Therefore, we only compare the tracking performance with our previous work [9]. In addition, to ensure reliable comparisons, we change the evaluation criteria in the previous work [9] to be the same as in this work.

The MOTP and MOTA results on camera views and ground plane for sequence S2.L1 are shown in Table 2. For sequence S2.L2, the MOTP and MOTA evaluations on camera views and ground plane are shown in Table 3.

4.2.2.1. Results with/without simulator. It can be observed that the performance is better when the information from crowd simulator is integrated. To elaborate, for sequence S2.L1, with the help from crowd simulation, the MOTP evaluations for View 1, 5, and 7 do not reveal significant changes, but the MOTA evaluations for View 1, 5, 7 have improved by 0.91%, 2.06%, and 5.94% when the RVO2 library is integrated as the simulator, and by 4.11%, 3.14%, and 6.28% when Social Behavior Model is applied. The MOTP and MOTA improvements on the ground plane are 1.43% and 2.68% with the RVO2 library, and 1.97% and 5.14% with the Social Behavior Model. For sequence S2.L2, when the RVO2 library is used, the MOTP for View 1 and 2 increases by 3.35% and 1.7%. When the Social Behavior Model is taken as the crowd simulation strategy, the MOTP for View 1 and 2 increases by 3.7% and 2.66%. In terms of MOTA, the performance gains when the RVO2 library is used for View 1 and 2 are 20.64% and 10.47%, respectively. The gains for the Social Behavior Model for the two views are 25.79% and 14.48%. The performances on the ground plane have a similar trend, the MOTP and MOTA increase by 6.47% and 22.75% when the RVO2 library is used as the crowd simulator, and for the Social Behavior Model, the improvements are 6.51% and 24.85%, respectively. Compared to sequence S2.L1 which only contains frames with low crowd density, for sequence S2.L2 with medium crowd density, the tracking performance improvement brought by the integration of crowd simulation is more significant. This means that crowd simulation

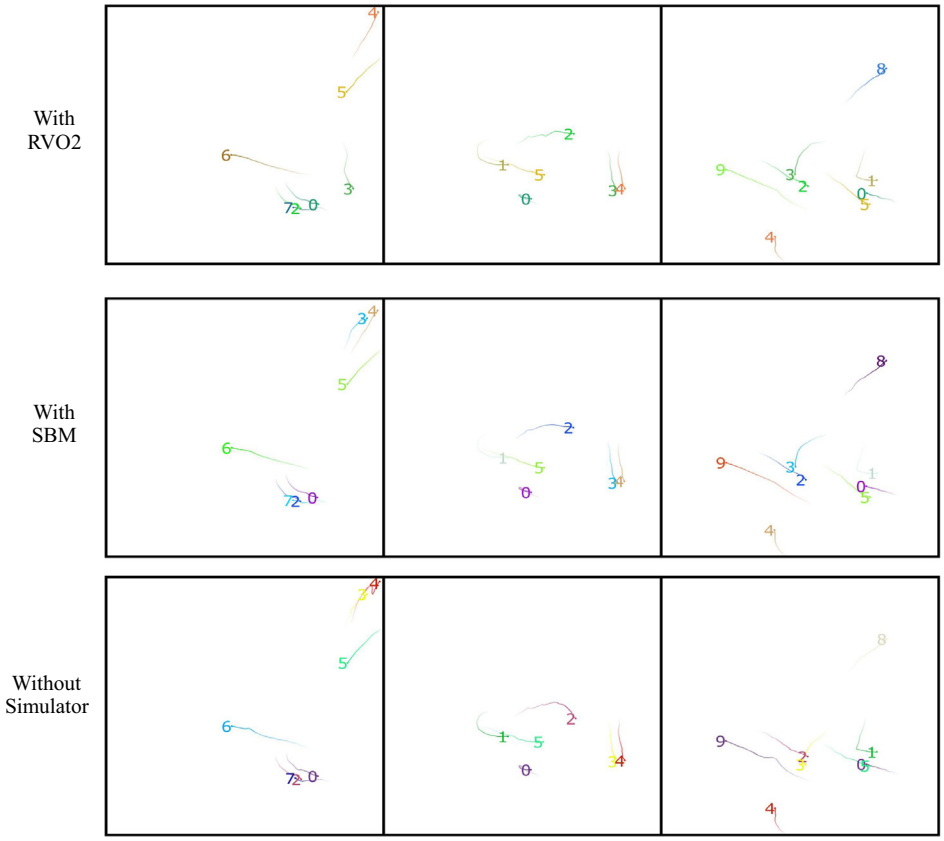


Fig. 9. Tracked pedestrian trajectories on the ground plane for sequence S2.L1. The trajectories are constructed on the most recent 30 frames for better illustration. The three columns correspond to frames #171, #330, and #729. The first row shows the results from the multi-camera tracker with RVO2 library, the second row shows the results with Social Behavior Model, the third row shows the results without simulator. Different trackers are shown in different colors. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

plays a more important role while tracking pedestrians in scenarios that are more crowded, where pedestrian appearances are more unreliable due to severe occlusions.

4.2.2.2. Comparison of two simulators. For the MOTP, different crowd simulation strategies lead to similar results. However, for the MOTA, the Social Behavior Model generates better results than the RVO2 library. For View 1, 5, 7 and the ground plane of sequence S2.L1, the integration of Social Behavior Model results in 3.26%, 1.08%, 0.34% and 2.46% MOTA improvements compared to the integration of the RVO2 library. For sequence S2.L2, the Social Behavior Model outperforms the RVO2 library by 5.15%, 4.01% and 2.1% in the evaluations on View 1, 2, and ground plane. The performance gain is consistent with our expectation that a better simulation strategy brings better motion predictions, and thus leads to better evaluation results. However, the difference between the experimental results corresponding to different simulators is not as large as the gap between the results where there is a crowd simulator or not. There may be several reasons: (1) Although the Social Behavior Model has a more realistic simulation strategy than the RVO2 library, the two strategies are incapable of generating totally different predictions; (2) The result from crowd simulation is not the only factor that decides the particle weights. According to the parameter values, the detection and classification terms together actually contribute a larger portion to the weights of particles for the observation model in Eq. (22).

4.2.2.3. Influence of parameter δ. To investigate and determine the value of parameter δ in Eq. (22), we collect tracking performances when δ takes different values. Since $\beta : \gamma : \eta = 20 : 2 : 1$ according to the original work [8], we tested different ratios between δ and η. Five segments are sampled randomly from each sequence for the test. Each segments contains 50 frames. The MOTP and MOTA results for frame-based evaluations are averaged and shown in Table 4. It reveals that the tracking performance reaches optimal when $\delta : \eta = 2$.

4.2.2.4. Comparison of two frameworks. In addition, the tracking performance difference brought by the integration framework and the previous work [9] is revealed as well. Compared to our previous work [9] with the RVO2 library as the simulator, the RVO2 library integrated tracking system with the new framework in the proposed approach has increased MOTA by 3.67%, 2.38%, and 2.71% for the three views in S2.L1 while the MOTP performance remains the same. But for the ground plane the increments of MOTP and MOTA are 4.89% and 3.28%. For the two views in S2.L2, the MOTP results do not have significant differences between the two frameworks, but the MOTA evaluations increase by 3.81% and 5.09%. In terms of ground plane evaluation for S2.L2, the new framework outperforms the previous work [9] by 8.54% and 3.66% on evaluations based on MOTP and MOTA.

4.3. Visual results

Figs. 4–6 show qualitative results of tracking for View 1, 5, and 7 respectively for sequence S2.L1. For each view, the tracking results with the RVO2 library integrated, with the SBM (Social Behavior Model) integrated, without crowd simulation, and the ground-truth on three sample frames (#171, #330, and #729) are shown for comparison. Figs. 7 and 8 shows qualitative results of tracking

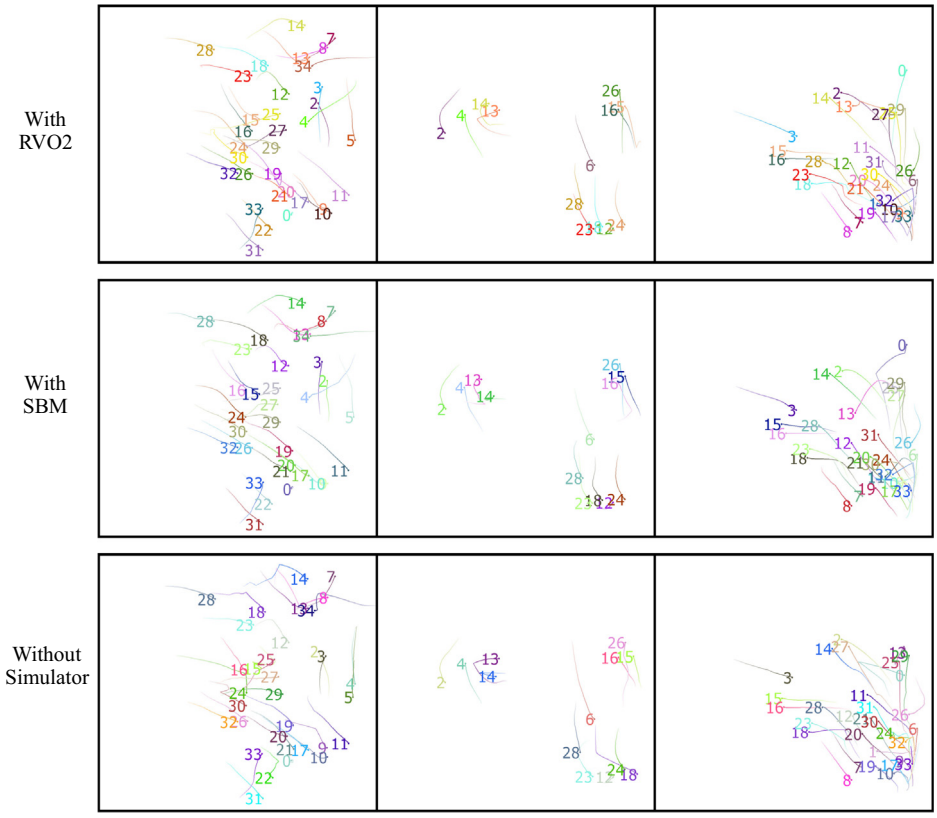


Fig. 10. Tracked pedestrian trajectories on the ground plane for sequence S2.L2. The trajectories are constructed on the most recent 30 frames for better illustration. The three columns correspond to frames #65, #200, and #395. The first row shows the results from the multi-camera tracker with RVO2 library, the second row shows the results with Social Behavior Model, the third row shows the results without simulator. Different trackers are shown in different colors. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 5
Tracking statistics for the sample frames from S2.L1. In each cell, the first number stands for accurately tracked pedestrians, and the second number is the total number of frame trackers.

	Frame #	Ground-truth	With simulator		Without simulator
			RVO2	SBM	
View 1	171	7	7/7	7/7	7/7
	330	6	5/6	6/6	6/6
	729	8	8/8	8/8	8/8
View 5	171	5	4/5	5/5	5/5
	330	5	5/5	5/5	4/5
	729	6	6/6	5/6	4/6
View 7	171	4	4/4	4/4	4/4
	330	5	4/5	5/5	4/5
	729	7	6/7	7/7	6/7

for View 1 and View 2 for sequence S2.L2 on three sample frames (#65, #200, and #395). Figs. 9 and 10 are the tracked trajectories for the two sequences, corresponding to the same frames in the previous figures. In addition, we provide the statistics for the sample frames on the number of tracked and accurately tracked (overlapping rate with ground-truth is over 50%) pedestrians in Tables 5 and 6.

It can observed that when crowd simulator is integrated, the tracking performance of the whole system improves, while using the Social Behavior Model is a little bit better than using the RVO2 library. This can be also demonstrated from the trajectories on the ground plane (Figs. 9 and 10). When crowd simulator is integrated, the generated trajectories are much smoother than those generated when there is no simulator integrated, because they are determined not only by the frame trackers, but also by the predictions from the crowd simulator that have more restricted temporal and spatial constraints. In addition, the integration of Social Behavior Model results in smoother trajectories than the integration of the RVO2 library, which means that the Social Behavior Model is able to provide more realistic predictions than the RVO2 library. The trajectories are not always smooth because the ground location of each pedestrian depends on all the tracking results from different camera views, and small changes in one of the views may cause significant drift on the ground location.

Table 6
Tracking statistics for the sample frames from S2.L2. In each cell, the first number stands for accurately tracked pedestrians, and the second number is the total number of frame trackers.

	Frame #	Ground-truth	With simulator		Without simulator
			RVO2	SBM	
View 1	65	33	23/28	27/28	21/26
	200	13	11/11	12/12	9/9
	395	31	25/26	25/27	24/29
View 5	65	33	21/26	26/30	23/25
	200	13	11/12	10/11	9/12
	395	31	25/28	23/27	18/24

5. Conclusions

In this paper, we have proposed a multi-camera tracking approach that integrates the crowd simulation approaches to the *state-of-the-art* single camera tracking-by-detection method. The difference between this approach and a traditional multi-camera tracking system is that the crowd simulation provides further usage of homography information. Two different crowd simulation strategies are exploited to observe their influence on the overall tracking performance. The experiments are conducted on crowded scenes from PETS 2009 dataset. The tracking performance is evaluated on different views as well as on the ground plane. Unlike the *state-of-the-art* trackers in which generally no crowd simulators are used, these results and comparisons demonstrate that significant improvement of tracking performance can be achieved when the predictions from crowd simulation are used, especially for scenarios with higher crowd density. For different crowd simulation strategies, difference in performance is also observed. According to the experimental results, the integration of more realistic crowd simulation can further improve the overall tracking performance in a multi-camera video network.

Acknowledgments

This work is supported in part by NSF Grants 0905671 and 1330110.

Appendix A. Supplementary data

Supplementary data associated with this article can be found, in the online version, at http://dx.doi.org/10.1016/j.cviu.2014.10.001.

References

[1] B. Bhanu, C. Ravishankar, A. Roy-Chowdhury, H. Aghajan, D. Terzopoulos, Distributed Video Sensor Networks, Distributed Video Sensor Networks, Springer, 2011.
[2] W. Hu, M. Hu, X. Zhou, T. Tan, J. Lou, S. Maybank, Principal axis-based correspondence between multiple cameras for people tracking, IEEE Trans. Pattern Anal. Machine Intell. 28 (4) (2006) 663–671, http://dx.doi.org/10.1109/TPAMI.2006.80.
[3] R. Eshel, Y. Moses, Tracking in a dense crowd using multiple cameras, Int. J. Comput. Vis. 88 (1) (2010) 129–143, http://dx.doi.org/10.1007/s11263-009-0307-0.
[4] Y. Li, B. Bhanu, Utility-based camera assignment in a video network: a game theoretic framework, IEEE Sensors J. 11 (3) (2011) 676–687, http://dx.doi.org/10.1109/JSEN.2010.2051148.
[5] L. Tessens, M. Morbee, H. Aghajan, W. Philips, Camera selection for tracking in distributed smart camera networks, ACM Trans. Sensor Networks 10 (2) (2014) 1–33, http://dx.doi.org/10.1145/2530281.
[6] K. Kim, L.S. Davis, Multi-camera tracking and segmentation of occluded people on ground plane using search-guided particle filtering, in: European Conference on Computer Vision (ECCV), Graz, Austria, 2006, pp. 98–109. doi:10.1007/11744078_8.
[7] S. Sternig, T. Mauthner, A. Irschara, P. Roth, H. Bischof, Multi-camera multi-object tracking by robust hough-based homography projections, in: International Conference on Computer Vision Workshops (ICCVW), Barcelona, Spain, 2011, pp. 1689–1696. doi:10.1109/ICCVW.2011.6130453.
[8] M. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, L. Van Gool, Online multiperson tracking-by-detection from a single, uncalibrated camera, IEEE Trans. Pattern Anal. Machine Intell. 33 (9) (2011) 1820–1833, http://dx.doi.org/10.1109/TPAMI.2010.232.
[9] Z. Jin, B. Bhanu, Integrating crowd simulation for pedestrian tracking in a multi-camera system, in: ACM/IEEE International Conference on Distributed Smart Cameras (ICDSC), Hong Kong, China, 2012, pp. 1–6.
[10] J. van den Berg, S. Guy, M. Lin, D. Manocha, Reciprocal n-body collision avoidance, in: C. Pradalier, R. Siegwart, G. Hirzinger (Eds.), Robotics Research, Springer Tracts in Advanced Robotics, vol. 70, Springer, Berlin Heidelberg, 2011, pp. 3–19. doi:10.1007/978-3-642-19457-3_1.
[11] S. Pellegrini, A. Ess, K. Schindler, L. Van Gool, You'll never walk alone: modeling social behavior for multi-target tracking, in: International Conference on Computer Vision (ICCV), Kyoto, Japan, 2009, pp. 261–268. doi:10.1109/ICCV.2009.5459260.
[12] K. Yamaguchi, A. Berg, L. Ortiz, T. Berg, Who are you with and where are you going?, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Providence, RI, USA, 2011, pp. 1345–1352. doi:10.1109/CVPR.2011.5995468.
[13] H. Grabner, H. Bischof, On-line boosting and vision, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), New York, NY, USA, 2006, pp. 260–267. doi:10.1109/CVPR.2006.215.
[14] H. Grabner, C. Leistner, H. Bischof, Semi-supervised on-line boosting for robust tracking, in: European Conference on Computer Vision (ECCV), Marseille, France, 2008, pp. 234–247. doi:10.1007/978-3-540-88682-2_19.
[15] B. Babenko, M.-H. Yang, S. Belongie, Robust object tracking with online multiple instance learning, IEEE Trans. Pattern Anal. Machine Intell. 33 (8) (2011) 1619–1632, http://dx.doi.org/10.1109/TPAMI.2010.226.
[16] X. Chen, Z. Qin, L. An, B. Bhanu, An online learned elementary grouping model for multi-target tracking, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Columbus, OH, USA, 2014.
[17] C. Huang, B. Wu, R. Nevatia, Robust object tracking by hierarchical association of detection responses, in: European Conference on Computer Vision (ECCV), Marseille, France, 2008, pp. 788–801. doi:10.1007/978-3-540-88688-4_58.
[18] Z. Qin, Improving multi-target tracking via social grouping, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE Computer Society, Providence, RI, USA, 2012, pp. 1972–1978. doi:10.1109/CVPR.2012.6247899.
[19] B. Yang, R. Nevatia, An online learned crf model for multi-target tracking, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Providence, RI, USA, 2012, pp. 2034–2041. doi:10.1109/CVPR.2012.6247907.
[20] M. Andriluka, S. Roth, B. Schiele, People-tracking-by-detection and people-detection-by-tracking, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Anchorage, AK, USA, 2008, pp. 1–8. doi:10.1109/CVPR.2008.4587583.
[21] W. Du, J. Piater, Multi-camera people tracking by collaborative particle filters and principal axis-based integration, in: Asian Conference on Computer Vision (ACCV), Tokyo, Japan, 2007, pp. 365–374. doi:10.1007/978-3-540-76386-4_34.
[22] C.-T. Chu, J.-N. Hwang, J.-Y. Yu, K.-Z. Lee, Tracking across nonoverlapping cameras based on the unsupervised learning of camera link models, in: ACM/IEEE International Conference on Distributed Smart Cameras (ICDSC), Hong Kong, China, 2012, pp. 1–6.
[23] O. Javed, K. Shafique, Z. Rasheed, M. Shah, Modeling inter-camera space-time and appearance relationships for tracking across non-overlapping views, Comput. Vis. Image Understand. 109 (2) (2008) 146–162, http://dx.doi.org/10.1016/j.cviu.2007.01.003.
[24] H.T. Nguyen, B. Bhanu, A. Patel, R. Diaz, Design and optimization of the videoweb wireless camera network, EURASIP J. Image Video Process. 2010 (2010) 1–12, http://dx.doi.org/10.1155/2010/865803.
[25] S. Zhou, D. Chen, W. Cai, L. Luo, M.Y.H. Low, F. Tian, V.S.-H. Tay, D.W.S. Ong, B.D. Hamilton, Crowd modeling and simulation technologies, ACM Trans. Model. Comput. Simulat. 20 (4) (2010) 1–35, http://dx.doi.org/10.1145/1842722.1842725.
[26] D. Helbing, P. Molnár, Social force model for pedestrian dynamics, Phys. Rev. E: Stat., Nonlinear, Soft Matter Phys. 51 (1995) 4282–4286, http://dx.doi.org/10.1103/PhysRevE.51.4282.
[27] D. Helbing, I. Farkas, T. Vicsek, Simulating dynamical features of escape panic, Nature 407 (2000) 487–490, http://dx.doi.org/10.1038/35035023.
[28] J. Ondřej, J. Pettré, A.-H. Olivier, S. Donikian, A synthetic-vision based steering approach for crowd simulation, in: ACM SIGGRAPH, Los Angeles, CA, USA, 2010, pp. 1–9. doi:10.1145/1833349.1778860.

[29] A. Treuille, S. Cooper, Z. Popović, Continuum crowds, in: ACM SIGGRAPH, Boston, MA, USA, 2006, pp. 1160–1168. doi:10.1145/1179352.1142008.
[30] E. Fehr, U. Fischbacher, The nature of human altruism, Nature 425 (6960) (2003) 785–791, http://dx.doi.org/10.1038/nature02043.
[31] N. Dalal, B. Triggs, Histograms of oriented gradients for human detection, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), vol. 1, San Diego, CA, USA, 2005, pp. 886–893. doi:10.1109/CVPR.2005.177.
[32] B. Leibe, A. Leonardis, B. Schiele, Robust object detection with interleaved categorization and segmentation, Int. J. Comput. Vis. 77 (1–3) (2008) 259–289, http://dx.doi.org/10.1007/s11263-007-0095-3.
[33] R. Jonker, T. Volgenant, Improving the hungarian assignment algorithm, Operat. Res. Lett. 5 (4) (1986) 171–175, http://dx.doi.org/10.1016/0167-6377(86)90073-8.
[34] Z. Jin, B. Bhanu, Optimizing crowd simulation based on real video data, in: IEEE International Conference on Image Processing (ICIP), Melbourne, VIC, Australia, 2013, pp. 3186–3190. doi:10.1109/ICIP.2013.6738656.
[35] A. Chan, N. Vasconcelos, Modeling, clustering, and segmenting video with mixtures of dynamic textures, IEEE Trans. Pattern Anal. Machine Intell. 30 (5) (2008) 909–926, http://dx.doi.org/10.1109/TPAMI.2007.70738.
[36] K. Bernardin, R. Stiefelhagen, Evaluating multiple object tracking performance: the clear mot metrics, EURASIP J. Image Video Process. 2008 (2008) 1–10, http://dx.doi.org/10.1155/2008/246309.
[37] A. Milan, K. Schindler, S. Roth, Challenges of ground truth evaluation of multi-target tracking, in: IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Portland, OR, USA, 2013, pp. 735–742. doi:10.1109/CVPRW.2013.111.