# Interactive Crowd Content Generation and Analysis using Trajectory-level Behavior Learning

Sujeong Kim, Aniket Bera and Dinesh Manocha
*University of North Carolina at Chapel Hill*
*https://gamma.cs.unc.edu/PDL*

***Abstract*—We present an interactive approach for analyzing crowd videos and generating content for multimedia applications. Our formulation combines online tracking algorithms from computer vision, non-linear pedestrian motion models from computer graphics, and machine learning techniques to automatically compute the trajectory-level pedestrian behaviors for each agent in the video. These learned behaviors are used to detect anomalous behaviors, perform crowd replication, augment crowd videos with virtual agents, and segment the motion of pedestrians. We demonstrate the performance of these tasks using indoor and outdoor crowd video benchmarks consisting of tens of human agents; moreover, our algorithm takes less than a tenth of a second per frame on a multi-core PC. The overall approach can handle dense and heterogeneous crowd behaviors and is useful for realtime crowd scene analysis applications.**

## 1. Introduction

The widespread use of commodity cameras and sensing devices has led to a considerable increase in videos of crowd scenes. Cameras are frequently used for surveillance of large groups of people in public places, walking on outdoor streets, attending sporting events, participating in religious and political gatherings, etc. In addition, it is important to analyze live streams of crowd videos for a variety of abnormal activities or behaviors.

One of the key challenges in this field is to devise methods to automatically analyze the behavior and movement patterns in crowd videos. For example, human experts can extract such information but can accommodate only a few crowd videos in any given instance. Crowd scene analysis has been studied extensively in computer vision, pattern recognition, multimedia and signal processing for more than a decade [1]. Many algorithms have been designed to track individual agents and/or to recognize their behavior and movements (e.g., by detecting abnormal behaviors) [2]. However, current methods are typically limited to sparse crowds or are designed for offline or non-realtime applications.

In this paper, we address the problem of interactive crowd content generation and scene analysis from videos. Our main goal is to extract the behavior and/or movement patterns of crowds at interactive frame rates, i.e., at tens of milliseconds on commodity desktop systems and ensure. One of our main motivations is to facilitate the development of interactive surveillance applications, which entails automatically recognizing normal or abnormal behaviors for live video streams.

**Main Results:** We present an interactive approach to analyzing crowd videos and generating content for multimedia applications. The key idea is to learn trajectory-level behaviors for each agent by combining techniques from online tracking in computer vision, motion simulation models in computer graphics, and machine learning. Given a video stream, we extract the trajectory of each agent using a realtime multi-person tracking algorithm and a non-linear motion model. Next, we use a Bayesian inferencing technique to compute the most likely state of each pedestrian and use the state information to compute the trajectory behavior feature for each agent. We use these trajectory behavior features for the following multimedia applications: crowd replication, anomaly detection, and motion segmentation.

We have implemented our system on a multi-core PC and have applied it to both indoor and outdoor crowd videos containing up to tens of pedestrians. We are able to compute crowd agents' trajectories and behavioral features in less than a tenth of a second. We demonstrate the benefit of these techniques by isolating pedestrians with unique or atypical behaviors, inserting automatically simulated virtual pedestrians with specific behaviors, and performing motion segmentation in structured and unstructured crowd video datasets. Compared with prior methods, our approach offers the following *benefits*:

**General:** Our approach is general and can be applied to heterogeneous videos of indoor and outdoor crowds. Furthermore, the trajectory-level behavior features can be used in a variety of applications.

**Interactive Performance:** Our approach can be used on a realtime video stream and to compute trajectory-level behavior features for each agent during each frame. As a result, we are able to capture and extract local movement changes in a crowd for each agent or groups of agents. As opposed to techniques in the prior literature, we do not require a large video dataset for offline learning.

**Dense Crowds:** Our approach can accommodate large crowds of moderate densities (e.g., 1-3 agents per square meter) because we use a non-linear motion model for multi-person tracking, state estimation, and automatic computation of the collision-free trajectories of virtual agents.

## 2. Related Work

Pedestrian tracking has been extensively studied in computer vision, pattern recognition, robotics, and related fields. At a broad level, pedestrian tracking algorithms can be classified as either online or offline trackers. Online trackers use only the present, previous or recent frames for tracking each pedestrian in the video in realtime. These trackers are based on non-adaptive random projections that model the structure of the image feature space of objects [3] or on learning semantic motion patterns for dynamic scenes by improved coding [4]. Other tracking algorithms use pedestrian motion features to compute the trajectory of the agents. These algorithms include clustering methods based on the assumption that pedestrians only appear and/or disappear at entry and/or exit [5], flow-field based methods to determine the probability of motion in densely crowded scenes [6]. Some of the work focused on generating smooth trajectories for use in data-driven simulation but is limited to tracking and doesn't learn overall pedestrian behaviors [7]. Overall, most of these methods are useful for offline applications but cannot accommodate dense crowd videos for use in interactive applications.

Extensive research has been previously conducted with respect to analyzing various crowd behaviors and movements from videos [1]. The main goals of these studies typically include extracting useful information regarding either behavior patterns, performing crowd activity recognition or detecting abnormal behavior or situations for surveillance analysis. Certain methods focus on classifying the most common behavior patterns in a given scene. However, most of these methods are designed for offline applications and tend to use a large number of training videos to learn the patterns offline for the following purposes: detecting common crowd behavior patterns [8], detecting normal and abnormal interactions [9], [10], or human group activities [11]; and developing approaches that use a large selection of videos on the web as examples of certain types of motion [12]. However, these techniques employ either manual selection or offline learning techniques to estimate the goal positions of all the pedestrians in a video, which limits their use with respect to realtime applications.

## 3. Interactive Trajectory Behavior Learning

In this section, we present our interactive trajectory-level behavior computation algorithm.

### 3.1. Terminology and Notation

We first introduce the notation used in the remainder of the paper and offer an overview of our approach.

**Pedestrians:** We use the term *pedestrian* to refer to independent individuals or human-like agents in the crowd. Their trajectories and movements are extracted by our algorithm using multi-person tracking.

**State representation:** A key aspect of our approach is to compute the state of each pedestrian in the video. Intuitively, the state corresponds to the low-level motion features that are used to compute the trajectory-level behavior features. In the remainder of the paper, we assume that all the agents are moving on a 2D plane. Realtime tracking of pedestrians is performed in a 2D image space and provides an approximate position, i.e., $(x, y)$ coordinates, of each pedestrian for each input frame. In addition, we infer the velocities and intermediate goal positions of each pedestrian from the sequence of its past trajectory. We encode this information regarding a pedestrian's movement at a time instance as a state vector. In particular, we use the vector $\mathbf{x} = [\mathbf{p}\ \mathbf{v}\ \mathbf{g}]^{\mathbf{T}}$, $\mathbf{x} \in \mathbb{R}^6$ to refer to a pedestrian's state. The state vector consists of three 2-dimensional vectors: $\mathbf{p}$ is the pedestrian's position, $\mathbf{v}$ is its current velocity, and $\mathbf{g}$ is the intermediate goal position. The intermediate goal position is used to compute the optimal velocity that the pedestrian would have taken had there been no other pedestrians or obstacles in the scene. As a result, the goal position provides information about the pedestrian's intent. In practice, this optimal velocity tends to be different from $\mathbf{v}$ for a given pedestrian. The state of the entire crowd, which consists of individual pedestrians, is the union of the set of each pedestrian's state $\mathbf{X} = \bigcup_i \mathbf{x_i}$.

**Trajectory behavior feature:** The pedestrians in a crowd are typically in motion, and their individual trajectories change as a function of time. In this paper, we restrict ourselves to trajectory-level behaviors or movement patterns, including current position, average velocity (including speed and direction), and the intermediate goal position. These features change dynamically.

### 3.2. System Overview

The overall system consists of multiple components: a real-time multi-person tracker, state estimation and behavior feature learning. Fig. 1 highlights these components. The input into our system is one frame of real-world crowd video at a time, and our goal is to compute these behavior features for each agent. An adaptive multi-person tracker is used to compute the observed position of each pedestrian on a 2D plane, denoted as $(\mathbf{z}_0 \cdots \mathbf{z}_t)$. Our online state estimation and behavior-learning algorithm is used to compensate for the tracking noise.

Because we do not know the dynamics of each agent (such as its velocity) and its true state, we estimate state $\mathbf{x}$ from the recent observations for each pedestrian. We use a Bayesian inferencing technique to estimate the most likely state of each pedestrian in an online manner and thereby compute the state of the overall crowd, $\mathbf{X}$. Based on estimated real crowd states, we extract the trajectory behavior feature of each agent. These features are grouped together to analyze the behavior or movement patterns, and used for various multimedia applications.

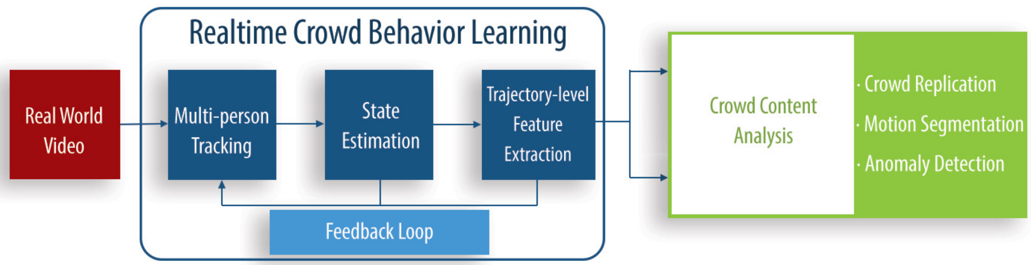



Figure 1: **Overview of our approach.** We highlight the different stages of our realtime algorithm: tracking, pedestrian state estimation and behavior learning. These computations are performed at realtime rates for each input frame. In addition, we highlight various multimedia applications of our approach.

### 3.3. Realtime Multi-person Tracking

Our approach employs a realtime multi-person tracker. There is a considerable amount of research in the computer vision literature regarding online or realtime tracking. In our case, any online tracker that requires the knowledge of motion-prior can be used. In particular, we use particle filters as the underlying tracking algorithm.

To predict the position of each agent, we use the appropriate motion model $m_t$ and the error $Q_t$ in the prediction that this motion model has generated. Additionally, the observations in our tracker are represented by a function, $h()$, that projects the state $\mathbf{X_t}$ to a previously computed state $S_t$. We denote the error between the observed states and the ground truth as $R_t$. We can now phrase these formally in terms of a standard particle filter as follows:

$$S_{t+1} = m_t(\mathbf{x}^t) + Q_t, \quad (1)$$
$$S_t = h(\hat{\mathbf{x}}^t) + R_t. \quad (2)$$

To reliably estimate the motion trajectory in a dense crowd setting, we use RVO (reciprocal velocity obstacle) – a local collision-avoidance and navigation algorithm – as the non-linear motion model. Given each agent's state at a particular time-step, RVO computes a collision-free state for the next time-step. More details and mathematical formulations of the ORCA constraints are provided in [13].

### 3.4. State estimation

We estimate the state of each real agent during each frame. The state estimation is performed in world-space coordinates by transforming the observations, $\mathbf{z}_t \in \mathbb{R}^2$, from the multi-person tracker. In this way, we are able to minimize the distortion in the trajectory, which eventually improves the accuracy of our local navigation motion model. Moreover, the state information computed for each agent can then be used in a different setting or crowd video (see section 4).

We use the Ensemble Kalman Filter (EnKF), which is an extension of Kalman Filtering, to compute the most likely state of each agent, $\mathbf{x_t}$, based on previous observations, $(\mathbf{z}_1, \cdots \mathbf{z}_t)$. Each agent's state constitutes a part of the overall crowd state $\mathbf{X}_t$. Per-agent inferencing permits us to easily accommodate entering and leaving agents in the environment, which is important in dynamic scenarios. The crowd state and interactions among pedestrians are still approximated by our state-transition model, $m_t$, as shown in Eqn. 1. In our case, we again use the RVO as the motion model $m_t$ for local navigation. First, EnKF predicts the next state based on the transition model and $Q_t$. When a new observation is available, $R_t$, is updated based on the difference between the observations and the prediction, which is used to compute the state of the real pedestrian. In addition, we run the Expectation Maximization (EM) step to compute the covariance matrix, $Q_t$, and to maximize the likelihood of the state estimation.

### 3.5. Trajectory Behavior Feature Extraction

The state estimation provides the position, velocity, and intermediate goal position at a given time. Based on the series of states, we compute the trajectory behavior features, which describe the past and future trajectory characteristics at the current location.

The trajectory behavior feature describes the characteristics of the trajectory behavior during a certain time window corresponding to the last $w$ seconds. The behavior feature vector consists of the current position, the average velocity during the time window, and the intermediate goal of an agent. We encode the connection between the recent velocity, $v^{avg}$, of a pedestrian and the intermediate goal position, $g$, at the current position. We denote the behavior feature vector, $\mathbf{b}$, which is a six-dimensional vector, as follows:

$$\mathbf{b} = [\mathbf{p}\ \mathbf{v}^{avg}\ \mathbf{g}]^T, \quad (3)$$

where $\mathbf{p}$, $\mathbf{v}^{avg}$, and $\mathbf{g}$ are each two-dimensional vectors that represent the current position, average velocity during the time window $t - w$ through $t$, and the estimated intermediate goal position computed as part of state estimation, respectively.

The duration of the time window can be set based on the characteristics of a scene. Small time windows are effective at capturing details in dynamically changing scenes with many rapid velocity changes that are caused by some agents moving quickly. Larger time windows, which tend to smooth out abrupt changes in motion, are more suitable for scenes that have fewer changes in pedestrian movement. In our case, we maintain the time window between 0.5 and 1.0 seconds in our current benchmarks.

## 4. Multimedia Applications and Evaluation

Our formulation computes trajectory behavior features at each time step. We can use these features to analyze the input crowd video and generate crowd contents. In this section, we demonstrate how these features and/or characteristics can be used to detect anomalies and for motion segmentation, and further to replicate the crowd behaviors. We employ the unsupervised classification method, which runs in an online manner and does not require offline training. Thus, the analysis is performed completely based on the input video from the known frames, such as a set of recent frames.

We use a K-means data-clustering algorithm to group the trajectories' behavior features observed during a certain time window. We classify these features into $K$ groups of flows, which we call behavior clusters. $K$ and $N$ are user-defined values that represent the total number of the clusters and the total number of collected behavior features, respectively, and $K \leq N$. A set of behavior clusters $B = \{B_1, B_2, ..., B_K\}$ is thus computed as follows:

$$\underset{B}{\operatorname{argmin}} \sum_{k=1}^{K} \sum_{b_i \in B_k} dist(b_i, \mu_k), \tag{4}$$

where $b_i$ is a behavior feature vector, $\mu_k$ is a centroid of each cluster, and $dist(b_i, \mu_k)$ is a distance measure between the arguments. Further details about the behavior feature extraction and classification can be found in [18].

### 4.1. Motion segmentation

Our trajectory-level behavior features can also be used for motion pattern segmentation. Typically, motion pattern segmentation techniques segment spatial regions on an image/video based on the similarity of the pedestrians' movement patterns.

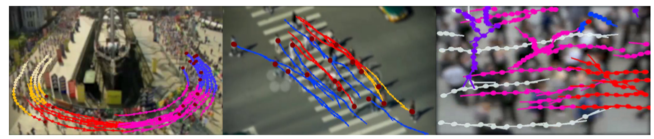
Flow-based methods are often used to segment crowd movements in videos [6]. These techniques mostly work well for structured scenes. Coherent filtering [19] uses tracklets instead of trajectories; thus, it can accommodate unstructured scenarios. Meta-tracking [20] tracks sets of particles and is effective for unstructured scenarios with high density crowds. See, e.g., [1]. In terms of segmentation results, our method yields similar results as meta-tracking, in terms of handling both structured and unstructured scenarios with low or high densities.

In our case, the distance between two feature vectors is computed as

$$\begin{aligned} dist(b_i, b_j) = {} & c_1 \left\| \mathbf{p}_i - \mathbf{p}_j \right\| \\ & + c_2 \left\| (\mathbf{p}_i - \mathbf{v}_i^{avg} wdt) - (\mathbf{p}_j - \mathbf{v}_j^{avg} wdt) \right\| \\ & + c_3 \left\| \mathbf{g}_i - \mathbf{g}_j \right\|, \end{aligned} \tag{5}$$

which corresponds to the weighted sum of distance between three points: current positions, previous positions and future positions. $c_1$, $c_2$, and $c_3$ are the weights.

Each behavior cluster is visualized with eight different colors based on the direction of the velocity components of its centroid. Figures 2 show the segmentation examples in structured, unstructured, and highly unstructured videos. For the Marathon video, we show that the segmentation from the sparse samples matches the behavior patterns of entire crowds. In terms of computation, our algorithm takes only tens of miliseconds for clustering computation during each frame.



(a) Marathon [6] (b) Crossing [14] (c) 879-38 [12]

Figure 2: **Motion segmentation of structured and unstructured scenarios:** Different colors indicate clusters grouped by similarity of behavior or movement features. We use eight discrete colors for visualization of the results in these benchmark.

We compared the accuracy of our motion segmentation and anomaly detection methods, using the quantitative metrics presented in Table 1 and Table V in Li et al. [15]. Table 1 in [15] provides true detection rate for motion pattern segmentation. It is based on the criteria that the method successfully detected the regions containing moving pedestrians. Although we cannot directly compare the numbers with pixel-based performance measures, MOTP values (Table 1) can be an indirect measure for the true detection rate. Compared to the values range **0.4-1.0** in [15], our approach corresponding values are in the range **0.7-0.8** in terms of detecting moving pedestrians, even for unstructured videos.

### 4.2. Anomaly detection

Anomaly detection is an important problem that has been the focus of research in diverse research areas and applications and corresponds to the identification of pedestrians, events or observations that do not conform to an expected pattern or other pedestrians in a crowd dataset. Typically, detection of anomalous items or agents will translate into better surveillance. Anomaly detection can be categorized into two classes based on the scale of the behavior that is being extracted [10]: global anomaly detection and local anomaly detection. We primarily use our trajectory-based behavior characteristics for local anomaly detection. In other words, we detect a few behaviors that are rarely observed in the video during certain periods. The periods can be as long as the length of the video or as short as a few hundred frames. In other words, we distinguish abnormality as temporally uncommon behavior. For example, a person's

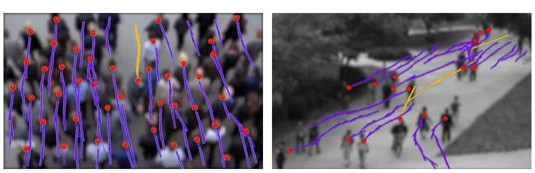
| Video | Real Peds | Virtual Peds | Density | Num. Frames | Time (sec) Tracking | Time (sec) Learning | MOTP | MOTA | Application |
|---|---|---|---|---|---|---|---|---|---|
| Crossing [14] | 19 | n/a | Medium | 238 | 0.04 | 0.03 | 71.9% | 51.9% | Motion Segmentation |
| Hotel [15] | 7 | 20 | Low | 137 | 0.029 | 0.005 | 79.2% | 64.1% | Crowd Replication |
| Marathon [6] | 18 | n/a | High | 450 | 0.04 | 0.02 | 35.1% | 21.7% | Motion Segmentation |
| 879-38 [12] | 23 | n/a | High | 349 | 0.042 | 0.01 | 73.9% | 51.2% | Motion Segmentation |
| 879-44 [9] | 63 | n/a | High | 349 | 0.048 | 0.05 | 81.8% | 69.4% | Anomaly Detection |
| UCSD-Peds1-Cart [16] | 13 | n/a | Low | 200 | 0.04 | 0.004 | 74.5% | 57.4% | Anomaly Detection |
| UCSD-Peds1-Biker [16] | 21 | n/a | Low | 200 | 0.041 | 0.009 | 76.1% | 53.2% | Anomaly Detection |

TABLE 1: Performance on a single core for different benchmarks: We highlight the number of real and virtual pedestrians, the number of static obstacles, the number of frames of extracted trajectories, the time spent in different stages of our algorithm, and the MOTA and MOTP values (metrics for measuring tracking accuracy) [17]. Our learning and trajectory computation algorithms can be used for different applications that are highlighted in the rightmost column.

behavior going against the flow of crowds may be detected as an anomaly at one point, but the same motion may not be detected as an anomaly later in the frame if many other pedestrians are moving in the same direction. By contrast, methods that use offline training may have consistent results regarding anomaly detection based on the training examples classified as *normal* behaviors.

As with the motion pattern segmentation application, we can reuse the clustering method for anomaly detection. When an anomaly appears in a scene, the anomaly features typically tend to be isolated as a separate cluster. We compute the differences of each cluster against all other clusters. If the difference value is higher than the threshold value, we detect the cluster as an anomaly.

Fig. 3 shows the results of anomaly detection in different crowd videos. **879-38 video dataset [9]**: The trajectories of 63 pedestrians are extracted from the video. One person in the middle is walking against the flow of pedistrians through a dense crowd. Our method can distinguish the unique behavior of this pedestrian by comparing its behavior features with those of other methods. In **UCSD-Peds1-Biker** and **UCSD-Peds1-Cart**, our method distinguishes parts of the trajectories of the biker and the cart because their speeds were noticeably different from other pedestrians.



(a) 879-44 [9] (b) UCSD-Peds1-Biker [16]

Figure 3: **Anomaly Detection:** Abnormal (yellow) and normal (purple) trajectories of motion patterns.

We evaluated the accuracy of anomaly detection algorithm on UCSD PEDS1 dataset [16] and the 879-44 Dataset[9], and compared with Table V in Li et al. [1] in Table 2.

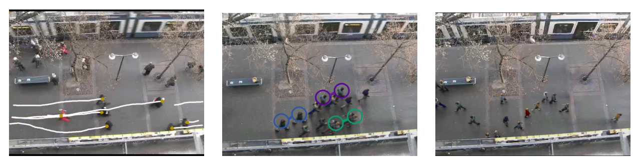
## 4.3. Crowd Replication

In many content creation and multimedia applications, crowd replication is widely used as a visual effects technique. In movies, artists may wish to increase the realism of a scene by virtually increasing the number of people in the scene. For example, a motion picture director may decide to shoot a war scene with only a few hundred people due to limitations in the production budget, knowing that the visual impact can be increased by 'duplicating' the pedestrians to look like a substantially larger crowd. Typically, replication is performed manually by copying and pasting existing segments of the given video to populate the crowd using video post-processing software. Techniques for populating crowds for purposes of animation, urban simulation or related applications have been widely studied [21]. However, these methods require manual scripting of the behaviors or offline data collection and motion computation.

Our method overcomes these limitations by inserting automatically simulated virtual pedestrians. The behaviors of virtual pedestrians are computed to exhibit similar trajectory behaviors as the real pedestrians in the original video. Fig. 4 shows a series of images taken from our results from populating the hotel video from BIWI video dataset [15]. We extracted seven pedestrian trajectories from the original

| Reference | Dataset | Performance | | | | |
|---|---|---|---|---|---|---|
| | | *Area under ROC Curve* | *Accuracy* | *DR* | *Equal Error Rate* | *Online/Offline* |
| **Our Method** | UCSD | **0.873** | **85%** | **-** | **20%** | **Online** |
| Wang 2012 | | 0.9 | - | 85% | - | Offline |
| Cong 2013 | | 0.86 | - | - | 23.9 | Offline |
| Cong 2012 | | 0.98-0.47 | 46% | 46% | 20% | Offline |
| Thida 2013 | | 0.977 | - | | 17.8% | Offline |
| Our Method | 879-44 | **0.97** | **80%** | **-** | **13%** | **Online** |

TABLE 2: Comparison of Anomaly Detection Techniques. Our methods has comparable results with the state of the art offline methods in anomaly detection.



(a) (b) (c)

Figure 4: **Crowd Replication:** (a) Original video with tracked trajectories. (b) Scene rendered with manually duplicated trajectories; the motion looks unrealistic because the walking motion is symmetric. (c) The scene rendered using our approach; the motion looks realistic because there are asymmetric walking motions and other behaviors that resemble the original crowd.

video. With our behavior learning algorithm, we added 20 virtual agents to the scene which have behavior patterns that are similar to the original video. We compared our results with the replication output using a manual copy and paste method (traditional copying) (Fig. 4 (b)). In the latter output, the pedestrians' trajectories are duplicated and the result looks unnatural (See similar-colored circles in figure). In contrast, our method can generate natural looking crowd behaviors, while continuing to follow the characteristics of the real pedestrians. As with the augmented crowd application, replicated crowds can be augmented by 3D reconstructed scenes.

## 5. Conclusion and Future Work

We present an interactive system and its applications that are related to crowd content generation and analysis. Our system runs in realtime and computes the trajectory and movement behavior for each agent during each time step. The behavior features extracted are used to analyze video content or to compute collision-free trajectories of virtual pedestrians, whose movement patterns resemble those of real pedestrians. Our approach is automatic and online, and it can capture the dynamically changing movement behaviors of real pedestrians. We demonstrate its application for scene analysis techniques, including anomaly detection and motion segmentation. We further use the behavior analysis for crowd replication application, where we can easily add tens to hundreds of virtual pedestrians and generate dense crowds.

**Limitations:** The performance and accuracy of our algorithm is governed by the tracking algorithm, which can be noisy, sparse or may lose tracks. Furthermore, current realtime methods may not work well in very dense crowd videos. Our online learning algorithm is useful only for capturing local pedestrian trajectory characteristics, whereas offline learning methods can compute many global characteristics. For example, our anomaly detection and motion segmentation algorithms will only capture unique/rare behaviors observed in temporally adjacent frames.

**Future Work:** There are many avenues for future work. In addition to overcoming the limitations of our work, we would like to combine the characteristics of pedestrian dynamics with other techniques that can model complex crowd behaviors. We would like to extend them for intelligent surveillance applications and also predict future crowd states.

## Acknowledgment

This work was supported by NSF awards 1117127, 1305286 and a grant from The Boeing Company.

## References

[1] T. Li, H. Chang, M. Wang, B. Ni, R. Hong, and S. Yan, "Crowded scene analysis: A survey," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 25, no. 3, pp. 367–386, March 2015.

[2] S. J. Junior *et al.*, "Crowd analysis using computer vision techniques," *IEEE Signal Processing Magazine*, vol. 27, no. 5, pp. 66–77, 2010.

[3] K. Zhang, L. Zhang, and M.-H. Yang, "Real-time compressive tracking," in *ECCV*, 2012, pp. 864–877.

[4] W. Fu, J. Wang, Z. Li, H. Lu, and S. Ma, "Learning semantic motion patterns for dynamic scenes by improved sparse topical coding," in *Multimedia and Expo (ICME), IEEE International Conference on*, 2012, pp. 296–301.

[5] X. Song, X. Shao, Q. Zhang, R. Shibasaki, H. Zhao, J. Cui, and H. Zha, "A fully online and unsupervised system for large and high-density area surveillance: Tracking, semantic scene learning and abnormality detection," *TIST*, 2013.

[6] S. Ali and M. Shah, "A lagrangian particle dynamics approach for crowd flow segmentation and stability analysis," in *Computer Vision and Pattern Recognition. IEEE Conference on*, June 2007, pp. 1–6.

[7] A. Bera, S. Kim, and D. Manocha, "Efficient trajectory extraction and parameter learning for data-driven crowd simulation," in *Graphics Interface*, 2015.

[8] B. Solmaz, B. E. Moore, and M. Shah, "Identifying behaviors in crowd scenes using stability analysis for dynamical systems," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 34, no. 10, pp. 2064–2070, 2012.

[9] R. Mehran, A. Oyama, and M. Shah, "Abnormal crowd behavior detection using social force model," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, June 2009, pp. 935–942.

[10] L. Kratz and K. Nishino, "Anomaly detection in extremely crowded scenes using spatio-temporal motion pattern models," in *Computer Vision and Pattern Recognition. IEEE Conference on*. IEEE, 2009, pp. 1446–1453.

[11] B. Ni, S. Yan, and A. Kassim, "Recognizing human group activities with localized causalities," in *Computer Vision and Pattern Recognition, 2009. IEEE Conference on*. IEEE, 2009, pp. 1470–1477.

[12] M. Rodriguez, J. Sivic, I. Laptev, and J.-Y. Audibert, "Data-driven crowd analysis in videos," in *Computer Vision (ICCV), 2011 IEEE International Conference on*, Nov 2011, pp. 1235–1242.

[13] J. Van Den Berg, S. J. Guy, M. Lin, and D. Manocha, "Reciprocal n-body collision avoidance," in *Robotics Research*, 2011.

[14] J. Shao, C. Loy, and X. Wang, "Scene-independent group profiling in crowd," in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, June 2014, pp. 2227–2234.

[15] S. Pellegrini, A. Ess, K. Schindler, and L. Van Gool, "You'll never walk alone: Modeling social behavior for multi-target tracking," in *Proc. of the (IEEE) 12th International Conference on Computer Vision*, Sept 2009, pp. 261–268.

[16] V. Mahadevan, W. Li, V. Bhalodia, and N. Vasconcelos, "Anomaly detection in crowded scenes," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2010, pp. 1975–1981.

[17] B. Keni and S. Rainer, "Evaluating multiple object tracking performance: the clear mot metrics," *EURASIP Journal on Image and Video Processing*, vol. 2008, 2008.

[18] S. Kim, A. Bera, A. Best, R. Chabra, and D. Manocha, "Adaptive data-driven crowd simulation for interactive virtual environments," University of North Carolina at Chapel Hill, Tech. Rep., 2015.

[19] B. Zhou, X. Tang, and X. Wang, "Coherent filtering: Detecting coherent motions from crowd clutters," in *Proc. of the European Conference on Computer Vision - Part II*, 2012, pp. 857–871.

[20] P.-M. Jodoin, Y. Benezeth, and Y. Wang, "Meta-tracking for video scene understanding," in *Advanced Video and Signal Based Surveillance, IEEE International Conference on*, Aug 2013, pp. 1–6.

[21] B. Yersin, J. Maïm, J. Pettré, and D. Thalmann, "Crowd patches: populating large-scale virtual environments for real-time applications," in *Interactive 3D graphics and games*, 2009, pp. 207–214.