



Minerva Access is the Institutional Repository of The University of Melbourne

Author/s:

Yang, M;Feng, Y;Rao, AS;Rajasegarar, S;Tian, S;Zhou, Z

Title:

Evolving graph-based video crowd anomaly detection

Date:

2024

Citation:

Yang, M., Feng, Y., Rao, A. S., Rajasegarar, S., Tian, S. & Zhou, Z. (2024). Evolving graph-based video crowd anomaly detection. *The Visual Computer*, 40 (1), pp.303-318. <https://doi.org/10.1007/s00371-023-02783-4>.

Persistent Link:

<https://hdl.handle.net/11343/339535>

Evolving graph-based video crowd anomaly detection

Cite this Accepted Manuscript (AM) as Accepted Manuscript (AM) version of Meng Yang, Yanghe Feng, AravindaS. Rao, Sutharshan Rajasegarar, Shucong Tian, Zhengchun Zhou, Evolving graph-based video crowd anomaly detection, The Visual Computer <https://doi.org/10.1007/s00371-023-02783-4>

This Accepted Manuscript (AM) is a PDF file of the manuscript accepted for publication after peer review, when applicable, but does not reflect post-acceptance improvements, or any corrections. Use of this AM is subject to the publisher's embargo period and AM terms of use. Under no circumstances may this AM be shared or distributed under a Creative Commons or other form of open access license, nor may it be reformatted or enhanced, whether by the Author or third parties. By using this AM (for example, by accessing or downloading) you agree to abide by Springer Nature's terms of use for AM versions of subscription articles: <https://www.springernature.com/gp/open-research/policies/accepted-manuscript-terms>

The Version of Record (VOR) of this article, as published and maintained by the publisher, is available online at: <https://doi.org/10.1007/s00371-023-02783-4>. The VOR is the version of the article after copy-editing and typesetting, and connected to open research data, open protocols, and open code where available. Any supplementary information can be found on the journal website, connected to the VOR.

For research integrity purposes it is best practice to cite the published Version of Record (VOR), where available (for example, see ICMJE's guidelines on overlapping publications). Where users do not have access to the VOR, any citation must clearly indicate that the reference is to an Accepted Manuscript (AM) version.

Evolving Graphs based Video Crowd Anomaly Detection

Meng Yang¹, Yanghe Feng^{2*}, Aravinda S. Rao³, Sutharshan
Rajasegarar⁴, Shucong Tian¹ and Zhengchun Zhou¹

¹School of Mathematics, Southwest Jiaotong University,
Chengdu, 610000, China.

^{2*}School of Systems Engineering, National University of Defense
Technology, Changsha, 410073, China.

³Dept. of Electrical and Electronic Eng., The University of
Melbourne, Melbourne, 3010, Australia.

⁴School of Information Technology, Deakin University, Geelong,
3125, Australia.

*Corresponding author(s). E-mail(s): fengyanghe@nudt.edu.cn;

Contributing authors: yangmeng@swjtu.edu.cn;

aravinda.rao@unimelb.edu.au;

sutharshan.rajasegarar@deakin.edu.au; sc Tian@my.swjtu.edu.cn;

zcc@swjtu.edu.cn;

Abstract

Detecting anomalous crowd behavioral patterns from videos is an important task in video surveillance and maintaining public safety. In this work, we propose a novel architecture to detect anomalous patterns of crowd movements via graph networks. We represent individuals as nodes and individual movements with respect to other people as the node-edge relationship of an evolving graph network. We then extract the motion information of individuals using optical flow between video frames and represent their motion patterns using graph edge weights. In particular, we detect the anomalies in crowded videos by modeling pedestrian movements as graphs and then by identifying the network bottlenecks through a max-flow/min-cut pedestrian flow optimization scheme (MFMCPoS). The experiment demonstrates that the proposed framework achieves superior detection

2 *Evolving Graphs based Video Crowd Anomaly Detection*

performance compared to other recently published state-of-the-art methods. Considering that our proposed approach has relatively low computational complexity and can be used in real-time environments, which is crucial for present day video analytics for automated surveillance.

Keywords: Abnormal Detection, Signal Processing, Video Surveillance, Graph Understanding

1 Introduction

Anomalous events often occur in a monitored environment due to random or untoward events, such as fire in a building, an attack on a group of people, a major accident and emergency events. Automated detection of such anomalous movement patterns assists in real-time crowd monitoring and related video surveillance applications [1]. Some of the examples of anomalous events in crowd videos include: (i) a person's movement different to other people in the crowd (for example, standing still while other people are moving); (ii) a person loitering around a place for a long time with no specific reason; (iii) an individual running for something while others in the crowd are walking; and (iv) a sudden influx or efflux of people at a location (example, at an entrance or exit in public places, such as stadium and cinema). By identifying and detecting these events, law enforcement agencies and event managers will be able to take effective measures to maintain security and safety.

Nowadays, crowded public places are increasingly equipped with a large number of surveillance cameras for recognizing human activities. However, the omnipresence of these cameras causes problems in terms of streaming, storing and processing video footage in a timely manner. Further, manual assessment of videos for recognizing certain visual activities suffers from observer fatigue, which makes the detection and recognition of crowd activities to be less accurate in practice. Therefore, monitoring an individual is a challenging task in a crowded scenario, especially when we need to detect events from patterns of movements from the entire scene. Moreover, it is challenging to achieve high accuracy in identifying an individual from the crowded scene because of the presence of complex and dynamic scene backgrounds, overlapping objects, and inter and intra-object occlusions [2].

In order to address these challenges, several graph mining-based approaches have been proposed in the literature [3, 4]. The basic idea is to translate the video into a graph sequence so that the graph mining methods can be applied on these streams. In particular, hyper-spherical clustering [5] based approaches was used to automatically identify the interesting regions in the scene, and then the density of the pedestrian flows between every pair of these interesting regions is analyzed over consecutive time intervals. This information is then represented as a directed graph, where the identified interesting regions are represented as nodes, and the direction and density of flows/movements of



Fig. 1: Anomalous crowd behavior scenarios: (a)(b) sample frames from UCSD dataset—red blocks denote that the cyclists move in an abnormal manner; (c)(d) sample frames from MCG dataset—red blocks denote people standing statically or loitering.

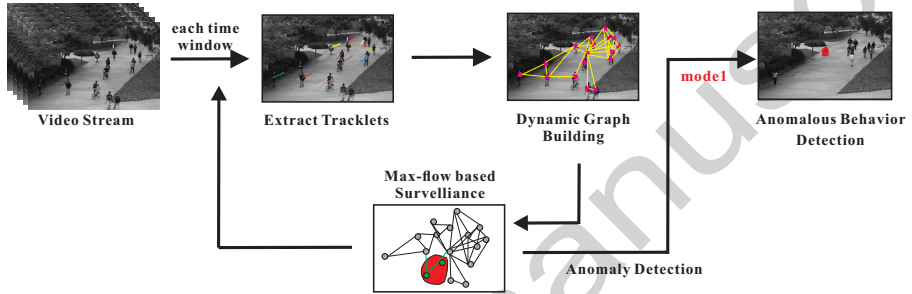


Fig. 2: An overview of our proposed anomaly detection framework. (1) At first, the moving objects in the input video are detected, followed by extraction of short motion cues forming tracklets for each fixed time window. (2) Time-variant graph sequences are built to represent the crowd motion activity, where the objects are identified as nodes, and the spatio-temporal features of the tracklets are used as the weights of the edges. (3) Then, a max-flow based surveillance approach is proposed to find the max/min cut of each graph, and the output is used to identify any anomalies. (4) Finally, based on different dominant features, the framework finds the anomalous behaviors.

people/objects between a pair of nodes over a time interval is represented as the corresponding edge weight in the graph. A sequence of adjacency matrices are obtained for a video sequence using subgraphs [3] and a change points to identify the various anomalous movements.

In this work, we propose a novel crowd anomaly detection architecture based on time evolving graph theory. In particular, we use a time varying graph sequence in which we treat each individual object as a node, and propose the use of similarity measures as the edge weights. Therefore, more dynamic anomalies can be detected. In other words, the video is represented as a dynamic graph stream with flexible numbers of nodes and edges. Briefly describing, we first use (Kanade-Lucas-Tomasi) KLT tracker [6] to efficiently capture the points of interests of the objects moving in the scene throughout a small window of consecutive frames. This is followed by using similarity metrics that capture movement characteristics, such as direction, magnitude, velocity

4 *Evolving Graphs based Video Crowd Anomaly Detection*

and geographic distance of the objects to construct the global dynamic graph sequences. Subsequently, we introduce a max-flow/min cut based algorithm to detect the component(s) of the network (communities) that are far from the rest of the network, which are then identified as anomalous events. Fig. 2 shows the proposed architecture.

As the experimental result shows, the proposed framework can be used in UCSD dataset as shown in Fig. 1-a and Fig. 1-b, where two cyclists move in an abnormal way when compared with other pedestrians. The example frames from MCG dataset are as described in Fig. 1-c and Fig. 1-d, where some people are standing still or loitering around a place. We can perform anomaly detection, provide information on the time frame when an event has occurred, as well as detecting the culprit.

The proposed approach can detect different anomalies present in video surveillance applications, which are generic and can be re-deployed to other locations with relatively little re-training. We perform the model training offline and infer the anomalies online for real-time applications. Considering our small model size compared to deep networks, our proposed approach can detect anomalies in real-time and requires little re-training to deploy to other locations.

This paragraph reflects the following structure of our paper. In Section 2, the literature review is discussed, followed by giving the problem statement in Section 3. The algorithm of extracting Motion Cues with Building Dynamic Graphs of Crowd Movements, representing Activities as Dynamic Graphs from Tracklets, Max-flow/Min Cut Pedestrian Flow Optimization for Anomaly Detection and final anomaly detection are demonstrated in Section 4, 5, 6 and 7, respectively. In Section 8, we present our experimental evaluation and results on the three benchmark datasets and one private dataset, and then conclude in Section 9.

2 Literature Review

Several methods have been proposed in the literature for detection of anomalous crowd behavior. Broadly, these methods can be categorized based on (i) tracking and modeling of motion using spatiotemporal features, and (ii) using deep learning.

Tracking methods [7] first obtain the trajectory of object movements and then represent these trajectories based on different feature representation schemes, as the inputs for anomaly detection. In [8], a social force model was proposed to describe crowd behavior, followed by the use of optical flow to extract motion features, and a Latent Dirichlet Allocation (LDA) model to detect abnormal crowd behavior. In [9], pedestrians' motion patterns were extracted from crowd behavior and represented using a mixture of dynamic textures model (MDT), which is a joint model that captures both the dynamics and the appearance of the scenes. Kim and Grauman [10] combined optical flow patterns with a mixture of Probabilistic Principal Component Analyzers

(PCA) as a model, and proposed a spatio-temporal graph model based on a Markov Random Field (MRF) for guaranteeing global consistency. However, achieving high accuracy of anomaly detection for crowded scenes or in complex behavioral settings remains a challenge for these approaches.

Deep learning based approaches have gained increasing attention recently with recent breakthroughs [11–14], in object detection and localization [15] using Convolutional Neural Networks (CNNs) [16]. Deep learning architectures, such as **Stacked Denoising Auto-Encoder (SDAE)** [17] and Fully Convolutional Networks (FCN) [18] have been used to learn feature representation automatically. **Xu et al.** [17] proposed an architecture called Appearance and Motion DeepNet (AMDN), that can automatically learn and capture features for the appearance, motion and joint correlation of objects in input image/video sequences. **As video data is** a high-dimensional space, obtaining efficient anomaly detections can be **challengeable**. In a recent development, high level deep learning frameworks have also been used to characterize crowd behavior. **Convolutional neural network based model is quite popular. For example, in [12], and in [13],** convolutional neural network and bidirectional LSTM are combined to solve this task. Other high level deep neural frameworks like using FCNs with temporal data [18] for automatic feature learning. Morias *et al.* [19] extract the dynamic skeleton features using a message-passing encoder-decoder recurrent network for anomaly detection. Dubey *et al.* [20] studied the joint learning approach to extract appearance and motion features using context-dependency called the Deep-network with Multiple Ranking Measures (DMRMs).

Although there are several approaches that have been proposed to detect anomalous crowd events, existing tracking-based and motion modeling-based methods differ in handling practical issues, such as crowded and complex scenarios. In addition, the computational complexity of training deep learning networks still face challenges, especially when computational resources are constrained. Thus, it is still an open challenge to achieve effective and efficient anomalous crowd monitoring in resource constrained environments **where training data is limited** [21, 22].

3 Problem Statement

In this paper, we propose a general yet elegantly simple graph based algorithm, that can achieve high accuracy in an efficient and effective manner. An overview of our framework is illustrated in Fig. 2. At first, we extract movement features as tracklets by applying a KLT feature point tracker [23], where the **tracklet** is a fragment of a trajectory, which is obtained over a short period [24]. These tracklets are then used to build dynamic graph sequences based on spatial and temporal features in order to model crowd behavior. Finally, a max-flow/min-cut pedestrian flow optimization scheme (MFMC-POS) is employed to perform anomaly detection. Our proposed method can be

6 *Evolving Graphs based Video Crowd Anomaly Detection*

used for different types of crowd event detection problems, where it can perform anomalous motion detection at both frame-level (identifying anomalous frames) and pixel-level (recognizing anomalous objects). The overall process of our proposed approach is shown in Algorithm 1.

In detail, the abnormal crowd detection is performed as follows: (1) First, the objects in the video are detected and tracked for a fixed time window Δ . We denote these objects as $O = \{o_i, i = 1 \dots n\}$, and the associated tracklets of each object are denoted as $\Gamma = \{\tau_i, i = 1 \dots n\}$, where n is the maximum number of objects and tracklets. Then, each pair of tracklets τ_i and τ_j in Γ are used to calculate the similarity measurement between the corresponding objects o_i and o_j , $S = \{s_{\phi, m, v, geo}, \phi = \text{direction}, m = \text{magnitude}, v = \text{velocity}, geo = \text{geodistance}\}$, where the dominant feature is chosen based on the application. (2) Further, the crowd movement information is represented as a time evolving graph $G = \{G_t, t = \Delta, 2\Delta, \dots\}$, where we use $G_t = (V, E)$ to denote the graph at time t .

The detected objects $O = \{o_i, i = 1 \dots n\}$ from the previous step are used as the vertices $V = \{v_i, i = 1 \dots n\}$ of G , and the obtained similarity values $S = \{s_{\phi m v}\}$ are used as the edge weights $E = \{e_{ij}, i, j \in [1 \dots n]\}$, where i and j denote a pair of vertices. After the first two steps, the crowd activity during one time window is converted into a time-variant graph, upon which we apply (3) an optimization approach (MFMCPoS) to find the suspicious nodes `suspicious_nodes` and maximum flow value mf after applying maximum flow/minimum cut partitioning. (4) Finally, `suspicious_nodes` is used as the input of the last step in the anomaly detection. This is done to do anomalous behavior detection (as shown in Fig. 1), where `suspicious_nodes` are used for the final anomalous object detection.

Algorithm 1 Anomalous Event Detection Framework

Require: Video Input $F = \{f_t, t = \Delta, 2\Delta, \dots\}$

Ensure: anomalous nodes `anom_nodes`, or crowd event change point t_{anom}

```

1: while  $t \leftarrow \Delta$  to video.length do
2:   (tracklet  $\Gamma_t$ , objects  $O_t$ )  $\leftarrow$  PeopleMotionExtraction( $f_t$ )
3:    $G_t(V_t, E_t) \leftarrow$  TimevariantGraphBuilding( $\Gamma_t, O_t$ )
4:   suspicious_nodes  $\leftarrow$  MFMCPoS( $G_t$ )
5:   anom_nodes,  $t_{anom} \leftarrow$  AnomalyDetection(suspicious_nodes)
6: end while

```

4 Extracting Motion Cues and Building Dynamic Graphs of Crowd Movements

The video data is first pre-processed using a two-dimensional Gaussian low-pass filter to eliminate high frequency noise. The parameters of the Gaussian filter are set as $\sigma = 0.5$ and each frame is divided into an array of 5×5 blocks.

The parameters are chosen in such a way that the low-frequency information and edge information is retained in the filtered output.

The motion features are then extracted and represented as tracklets, where a tracklet is a fragment of the trajectory obtained over a short period [24], using a KLT tracker [25] over a small window Δ of consecutive frames. The feature detectors we have considered include Scale Invariant Features Transformation (SIFT) and Speeded Up Robust Features (SURF). Since our aim is to find the group behavior at the whole frame level, we do not need to use state-of-art object detection algorithms to achieve highly accurate individual detection performance (such as would be required for people counting tasks). Therefore, **we use SIFT and SURF detectors for KLT tracker**. As SURF is suitable for illumination changes in images [26], and can be more efficient in the feature selection stage through a fast approximation of the Hessian when compared with SIFT, we choose SURF as the feature descriptor in this stage. SURF has better performance in terms of rotation invariance, warping transforms and blur when compared with SIFT. Furthermore, SURF is three times faster than SIFT, so we choose SURF as the feature detector to be used with the KLT tracker for extracting tracklets.

4.1 Tracking Motion Features using KLT Tracker

The KLT tracker allows tracking of objects using optical flow [25]. We use KLT to extract motion features from the moving objects that appear in the first frame and then track them in the rest of the frames. The image motion constraint is calculated using the following equation:

$$I(x, y, t) = I'(x + \Delta_x, y + \Delta_y, t + \Delta_t), \quad (1)$$

where I denotes the pixel intensity at location (x, y) in the t -th frame, and I, I' are consecutive frames with time difference Δ_t . Tracking objects over several frames suffers from high computational complexity and also introduces noise. Therefore, we choose tracklets instead of full trajectories. We take $O = \{o_i, i = 1 \dots n\}$ to represent the moving objects, where n is the maximum number of objects in the frame. The associated tracklets of objects O are denoted as $\Gamma = \{\tau_i, i = 1, \dots, n\}$.

We improve the optical flow using the features extracted by the SURF descriptor, which helps the detection and tracking scheme to work in a more effective and efficient way.

5 Representing Activities as Dynamic Graphs from Tracklets

Having obtained the tracklets $\Gamma = \{\tau_i\}_{i=1, \dots, n}$ from the previous step, each pair of tracklets τ_i and τ_j in Γ are used to calculate the similarity features that are used to model the crowd movement behavior. These metrics are defined in

8 *Evolving Graphs based Video Crowd Anomaly Detection*

the following subsections. These metrics will help to translate crowd activity into time-varying graph sequences.

5.1 Behavioral Similarity Measures

The similarity measurement scheme we use is proposed in [27, 28], where the similarity measure is calculated from tracklet features. For a tracklet τ_i , the direction is calculated as $\phi_{\tau_i} = (\Delta x_{\tau_i}, \Delta y_{\tau_i}) = (x_{\tau_i, \text{begin}} - x_{\tau_i, \text{end}}, y_{\tau_i, \text{begin}} - y_{\tau_i, \text{end}})$, i.e., the difference between the tracklet's start and end coordinates. The magnitude features m_{τ_i} denotes the distance between the start and end spatial coordinates of ϕ_{τ_i} , where $m_{\tau_i} = \sqrt{(\Delta x_{\tau_i})^2 + (\Delta y_{\tau_i})^2}$, i.e., the L_2 norm of ϕ_{τ_i} . Since spatial and directional features are robust to perturbations and invariant to location changes, they provide a suitable representation for movement behaviors. In this paper we construct the following types of similarity features based on the above features: direction s_ϕ , magnitude s_m , velocity s_v and geographic distance s_{geo} .

5.1.1 Directional Similarity Measure

Given two tracklets τ_i and τ_j , their directional features are represented as ϕ_{τ_i} and ϕ_{τ_j} , respectively. The cosine metric is used to compute the angle between the two tracklets τ_i and τ_j , which is denoted as the directional similarity measure s_ϕ :

$$s_\phi^{\tau_i, \tau_j} = \frac{1}{2} \cdot \left(\frac{\phi_{\tau_i} \cdot \phi_{\tau_j}}{\|\phi_{\tau_i}\| \cdot \|\phi_{\tau_j}\|} + 1 \right) \quad (2)$$

5.1.2 Magnitudinal Similarity Measure

Given the magnitude features m_{τ_i} and m_{τ_j} for the two tracklets τ_i and τ_j , we define s_m as the magnitude similarity measure to capture the normalized difference between the two tracklets' traversed distances:

$$s_m^{\tau_i, \tau_j} = 1 - \frac{|m_{\tau_i} - m_{\tau_j}|}{\max(m_{\tau_i}, m_{\tau_j})} \quad (3)$$

5.1.3 Velocity Similarity Measure

The velocity similarity measure $s_v^{\tau_i, \tau_j}$ is calculated based on Dynamic Time Warping (DTW) for each pair of tracklets τ_i and τ_j [28].

$$s_v^{\tau_i, \tau_j} = \frac{e^{\frac{-d_{vx}(\tau_i, \tau_j)}{\sigma_{vx}}} + e^{\frac{-d_{vy}(\tau_i, \tau_j)}{\sigma_{vy}}}}{2} \quad (4)$$

where σ_{vx} and σ_{vy} correspond to the overall standard deviation of object velocities along the x-axis and y-axis, respectively, and $d_{vx}(\tau_i, \tau_j)$ and $d_{vy}(\tau_i, \tau_j)$ denote the distance on the x-axis and y-axis between the two tracklets.

5.1.4 Geographic Distance Measure

The geographic distance $s_{geo}^{\tau_i, \tau_j}$ represents the Euclidean geo-distance between the centroid of each pair of tracklets τ_i and τ_j and can be computed by Equation (5):

$$s_{geo}^{\tau_i, \tau_j} = \sqrt{(Cent\tau_i^x - Cent\tau_j^x)^2 + (Cent\tau_i^y - Cent\tau_j^y)^2} \quad (5)$$

where $(Cent\tau_i^x, Cent\tau_i^y)$ and $(Cent\tau_j^x, Cent\tau_j^y)$ are the coordinates of the centroid of tracklet τ_i and τ_j , respectively, i.e.,

$$(Cent\tau_i^x, Cent\tau_i^y) = \left(\frac{x_{\tau_i, begin} + x_{\tau_i, end}}{2}, \frac{y_{\tau_i, begin} + y_{\tau_i, end}}{2} \right) \quad (6)$$

5.2 Combinational Similarity Measure

The four types of similarity measures defined (before) are combined into three combinational similarity measures to determine the edge weights between two given tracklets (i.e., of objects). The intuition behind this is that (1) the directional similarity is used for detecting pedestrians who are moving in an anomalous direction; (2) the magnitude similarity is used to capture movements with the same direction but different lengths; (3) the velocity similarity measure is somewhat similar to the magnitude similarity, in that it helps us determine whether a pair of tracklets that are moving along the same direction and have the same lengths, follow the same pattern of movement.

5.2.1 Direction-Magnitude Similarity Measure

The combined direction-magnitude similarity measure is defined as a linear combination of the direction similarity s_ϕ and magnitude similarity s_m using Equation (7). Note that tracklets τ_i and τ_j represent two distinct objects:

$$s_{\phi m}^{\tau_i, \tau_j} = \alpha \cdot s_\phi^{\tau_i, \tau_j} + (1 - \alpha) \cdot s_m^{\tau_i, \tau_j} \quad (7)$$

where α is a parameter within the range $[0, 1]$ that is used to control the influence of direction and magnitude for a pair of tracklets.

5.2.2 Velocity-Direction-Magnitude Similarity Measure

The combined velocity-direction-magnitude similarity measure $s_{\phi mv}$ is the combination of the Direction-Magnitude combination similarity measure $s_{\phi m}$ and the velocity similarity measure s_v . For a pair of tracklets (τ_i, τ_j) , this similarity measure is given by

$$s_{\phi mv}^{\tau_i, \tau_j} = \beta \cdot s_{\phi m}^{\tau_i, \tau_j} + (1 - \beta) \cdot s_v^{\tau_i, \tau_j} \quad (8)$$

where β is a parameter within the range $[0, 1]$ that is used to control the influence of direction-magnitude combination and the velocity similarity measures.

5.2.3 Direction-GeoDistance Similarity Measure

The combined direction-geodistance similarity measure $s_{\phi, geo}$ is the combination of the direction similarity s_{ϕ} and the geographic distance s_{geo} , which is defined as:

$$s_{\phi, geo}^{\tau_i, \tau_j} = \begin{cases} s_{\phi}^{\tau_i, \tau_j}, & s_{geo} < \delta_{geo} \\ 0, & s_{geo} \geq \delta_{geo} \end{cases} \quad (9)$$

where the direction similarity measure s_{ϕ} is calculated using Equation (2) and δ_{geo} is a threshold representing the maximum difference between the location of two tracklets. This combined similarity measure aims to distinguish the case where two objects are moving in the same direction while being far away from each other. In such a scenario, we do not consider them to be similar as they do not form a cluster of objects.

5.3 Building Dynamic Graphs

We represent crowd movement information in terms of a time evolving graph $G = \{G_t, t = \Delta, 2\Delta, \dots\}$, such that $G_t = (V, E)$ denotes the graph at time t . The detected objects $O = \{o_k, k = 1 \dots n\}$ from the previous step are used as the vertex set $V = \{v_k, k = 1 \dots n\}$ of G , and the edge between a pair of vertices represents the relationship between them, where the similarity measurement value (from Equations (7), (8) and (9)) as $S = \{s_{\phi, m, v, geo}\}$. These similarity measures are assigned as the weights of the edges $E = \{e_{ij}, i, j \in [1 \dots n]\}$, where i and j specify a pair of nodes. The process of building a graph with the time window Δ is shown in Algorithm 2.

In Algorithm 2, the threshold δ_a is used as the minimum similarity between two objects for them to be considered as neighboring nodes in the graph. This value is later used to determine the sensitivity of our approach by producing ROC curves based on multiplicative changes in the values of δ_a . We choose $s_{\phi m}^{\tau_i, \tau_j}$, $s_{\phi mv}^{\tau_i, \tau_j}$ or $s_{\phi, geo}^{\tau_i, \tau_j}$ based on the features that are dominant in a given dataset and that can help us in identifying anomalous objects or anomalous events. For instance, determining an abnormal movement pattern within a crowd will be best captured by the direction-magnitude and velocity-direction-magnitude similarity measure.

6 Max-flow/Min Cut Pedestrian Flow Optimization for Anomaly Detection

After building the dynamic graph using Algorithm 2, we apply our proposed method called the max-flow/min-cut pedestrian flow optimization scheme (MFMCPoS) for determining anomalous incidents or objects in the given video. The classic community detection tasks focus on partitioning the whole

Algorithm 2 Time-variant Graph Building

Require: Tracklets $\Gamma = \{\tau_i, i = 1 \dots n\}$ and Detected objects $O = \{o_k, k = 1 \dots n\}$

- 1: **while** $k = 1$ to n **do**
- 2: $v_k = o_k$
- 3: **end while**
- 4: vertex $V_t = \{v_k^t, k = 1 \dots n\}$ of graph G_t
- 5: **while** $i = 1$ to $n - 1$ **do**
- 6: **while** $j = i + 1$ to n **do**
- 7: **if** $S_i \geq \delta_a$ **then**
- 8: $e_{ij} = S_i$, where $S_i = \{S_{\phi_m}, S_{\phi_{mv}}, S_{\phi_{geo}}\}$
- 9: **else**
- 10: $e_{ij} = 0$
- 11: **end if**
- 12: **end while**
- 13: **end while**
- 14: $E_t = \{e_{ij}^t, i, j \in [1 \dots n]\}$

Ensure: graph $G_t = (V_t, E_t)$, where $t = \Delta$

graph, whereas the focus in our problem is on detecting the members (or communities) that are far from (more dissimilar to) the rest of the network.

The max-flow/min-cut algorithm was originally devised to identify the maximum flow that a pipeline network can support at any given time [29]. This maximum flow was determined by the narrowest pipe's capacity, i.e., the bottleneck of the network. This algorithm has been used for graph partitioning, where the aim is to find partitions of nodes such that the number of edges between these partitions is minimized. The intuition behind using max-flow/min-cut in our scenario is that the anomalous objects behave abnormally with the rest of the moving objects based on our similarity measures, therefore they would correspond to nodes that are furthest away from the rest of the nodes and impose a bottleneck constraint on the pedestrian flow of the network. These nodes are unlikely to belong to the well-connected community of normal pedestrians. We employ MFMCPOS to detect anomalous objects which tend to have different velocity, direction, and distance from the objects within their reference block.

We now provide the necessary notation and preliminaries on the MFMCPOS problem. Given a graph $G = (V, E)$, where V and E represent the sets of nodes and edges of the graph, our aim is to find the best cut in the graph. Consider a flow network (G, s, t, c) , where $s, t \in V$ denote the source and destination nodes respectively, and c represents the maximum amount of flow, (i.e., capacity, that passes through an edge). The capacity of an edge e_{ij} is a mapping $c: E \rightarrow \mathbf{R}^+$, denoted by c_{ij} or $c(i, j)$. A flow f is a mapping $f: E \rightarrow \mathbf{R}^+$, denoted by f_{ij} or $f(i, j)$, and subjected to some constraints like $f_{ij} \leq c_{ij}$.

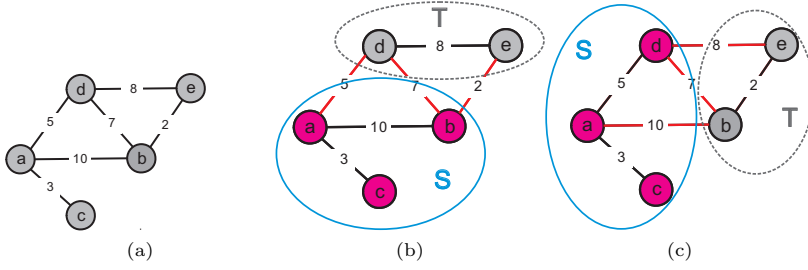


Fig. 3: (a) undirected graph G ; (b) An example of a cut $C(S, T)$ where $S = \{a, b, c\}$ and $T = \{d, e\}$; and (c) a second example of a cut $C(S, T)$ where $S = \{a, d, c\}$ and $T = \{b, e\}$.

We formulate the problem [29, 30] as follows. Let (G, s, t, c) be a flow network, then finding a $s - t$ cut involves partitioning V into two sets S and T , such that $S \cup T = V$, $S \cap T = \emptyset$, and $s \in S$ and $t \in T$. Given nodes s and t as our source and destination with the capacities of the graph shown by the edge weights, we determine the best partition that separates s from t and provides us with the maximum flow from source to destination. Fig. 3 shows an example cut $C(S, T)$ in an undirected graph G and also two possible partitions of the graph.

Algorithm 3 Max-flow/Min-cut Pedestrian Flow Optimization Scheme (MFMCPoS)

Require: graph $G = (V, E)$

```

1:  $n \leftarrow \text{num of nodes in } V, \text{best\_mf} \leftarrow \text{num of edges in } E$ 
2: while  $i \leftarrow 1$  to  $n - 1$  do
3:   while  $j \leftarrow 1$  to  $n$  do
4:      $[mf, cs, ct] = \text{maxflow}(G, i, j)$ 
5:     if  $\text{best\_mf} > mf$  and  $mf > 0$  then
6:        $\text{best\_mf} \leftarrow mf, \text{best\_cs} \leftarrow cs, \text{best\_ct} \leftarrow ct$ 
7:     end if
8:   end while
9: end while
10: if  $\text{length}(\text{best\_ct}) < \text{length}(\text{best\_cs})$  then
11:    $\text{suspicious\_node} \leftarrow \text{best\_ct}$ 
12: else
13:    $\text{suspicious\_node} \leftarrow \text{best\_cs}$ 
14: end if
```

Ensure: max flow value and suspicious node sets : $[mf, cs, ct]$

The capacity $c(S, T)$ of cut $C(S, T)$ is the sum of the edge weights $e(i, j)$ between the two partitions, i.e., $i \in S$ and $j \in T$.

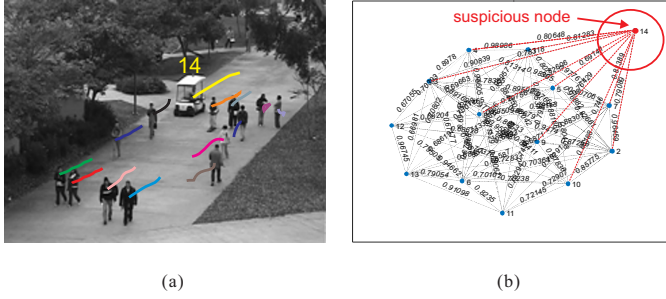


Fig. 4: Example of using MFMCPPOS on UCSD dataset, test24, time window 9 (frame 81-90), where node id 14 is detected as the suspicious node.

In the example shown in Fig. 3-b, $S = \{a, b, c\}$ and $T = \{d, e\}$, $c(S, T) = 14$. Then the flow of a cut $C(S, T)$ is denoted by $f(S, T)$, as defined in Equation (10), with the constraints $f(i, j) \leq c(i, j)$,

$$f(S, T) = \left(\sum_{i \in S} \sum_{j \in T} f(i, j) - \sum_{j \in T} \sum_{i \in S} f(j, i) \right), \quad (10)$$

and the value of flow f defined as $|f| = \sum_{i \in V} f(s, i) = \sum_{i \in V} f(i, t)$.

The residual capacity of the pair (i, j) , which is called $c_f(i, j)$ is given as follows:

$$c_f(i, j) = c(i, j) - f(i, j), \quad (11)$$

And the residual network of a graph G that is induced by the flow f is shown as the directed graph $G_f = (V, E_f)$, where:

$$E_f = \{(i, j) \in V \times V \mid c_f(i, j) > 0\}, \quad (12)$$

Therefore the max-flow/min-cut problem corresponds to maximizing the value $|f|$, finding the maximum flow from source s to destination t in graph G . The algorithm is based on computing the max-flow iteratively by finding an augmented path in a residual digraph G_f . The digraph G_f must follow the constraints that it does not contain a two-cycle formed by two edges with non-zero weights. The proposed MFMCPPOS give us maximum pedestrian flow and suspicious nodes (corresponding to people) detailed in Algorithm 3. Fig. 4 shows an example of using MFMCPPOS on the UCSD testing dataset, sequence 24, time window 9 (frame 81-90), where node id 14 is detected as a suspicious node.

7 Anomaly Detection

The previous step provides us with a set of suspicious nodes and the corresponding max-flow value mf , which is then used as the input of our anomaly

detection step. As shown in our proposed framework in Fig. 2, our scheme can perform anomalous behavior detection, which corresponds to cyclists, cars and so on in the UCSD dataset, or loitering people in the MCG dataset. The proposed anomaly detection scheme is shown in Algorithm 4. It is used as $\text{anomalous_nodes} = \text{AnomalyDetection}(\text{suspicious_nodes})$ per time window.

Algorithm 4 Anomaly Detection

Require: suspicious node sets

Ensure: anomalous nodes

```

1: if suspicious_node is a new incoming node  $v_n$  then
2:   anomalous_nodes  $\leftarrow$  suspicious_node
3: else
4:   anomalous_nodes  $\leftarrow$  None
5: end if

```

In details, *suspicious_nodes* denote the final detected set of anomalies as shown in the following definition.

Definition 1: When a new node v_n appears in the current graph G_n , if it is detected as one of the *suspicious_nodes* in the subsequent graph sequence, we identify it as the anomalous node v_{ano} , otherwise it is identified as a normal node.

8 Results and Discussion

8.1 Experimental Setup and Datasets

Experimental Setup - The proposed approach is implemented in Matlab 2013b, and all experiments are conducted on Windows 10 Intel® i7 with a Geforce GTX 1060 NVIDIA graphics card.

Datasets - We conduct experiments on two datasets - **UCSD Ped1 dataset**, **UCSD Ped2 dataset**, **Avenue dataset**, and **MCG dataset**.

(1) **UCSD Ped1 dataset** [31] are obtained from a surveillance camera on a pedestrian walkway that contains 34 video sequences in the training dataset (containing purely normal activity) and 36 videos in the test dataset (which contains abnormal events) with 238×158 pixel resolution.

(2) UCSD Ped2 dataset contains crowds of people moving in parallel to the camera plane, and comprises 16 video sequences in the training dataset and 12 videos in the test dataset (which contains abnormal events) with 360×240 pixel resolution. Because we do not require pre-training, we only use the test sequences of these two datasets. For both UCSD Ped1 and Ped2, the normal behavior is people walking, and the abnormal behavior includes anomalous motion patterns and non-pedestrian objects like cyclists, people with wheelchairs, cars, skaters, etc.

(3) **The Avenue Dataset** [32] contains 16 training and 21 testing video clips, collected by The Chinese University of Hong Kong (CUHK) on their campus.

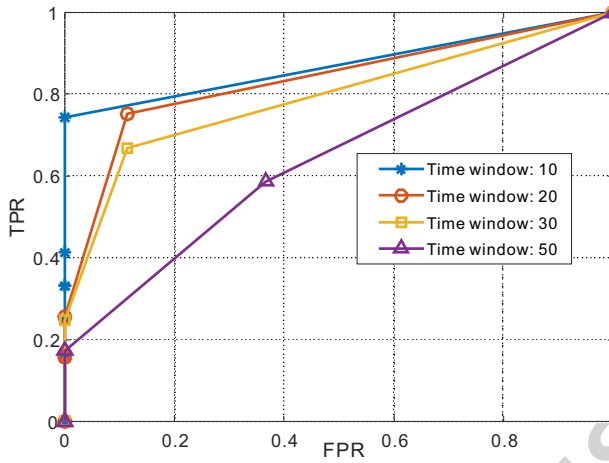


Fig. 5: Time window experiment, where time window $\Delta \in \{10, 20, 30, 50\}$ frames.

The total dataset comprises of 30652 frames, where the training data is with 15328 frames the testing data is 15324 frames.

(4) **MCG dataset:** We also conduct experiments on real-world video sequences from the Melbourne Cricket Ground (MCG), which is a large sports stadium in Melbourne, Australia. We used the video data of six cameras named C1 to C6. C1 is installed at the top of venue, and C2-C6 are installed at different corridor locations that provide diverse perspectives in the video data. We detect anomalous behaviors such as standing or loitering, which are important categories of movement anomalies in the MCG dataset. The video data is used as image sequences with 640×480 pixel resolution.

Based on the type of activity in each dataset, the details of the dominant features are listed in Table 1.

8.2 Time Window Experiment

In order to find a suitable time window, we conduct experiments on the UCSD Ped1 Test sequence 24 (Test24). The ROC curve is generated by changing the threshold δ_a . The results are shown in Fig. 5, and summarized in Table 2. When the time window is equal to 10 frames, the best performance is achieved. The reason why performance degrades for larger time windows is because each time window is encoded as one graph. If some of the frames within a large time window are abnormal, but the whole graph is identified as normal, the accuracy will be decreased. As the size of the time window decreases, the number of graphs that are generated increases, thus increasing the computation time. So we finally chose time window $\Delta = 10$, and this setting is used in all of the following experiments.

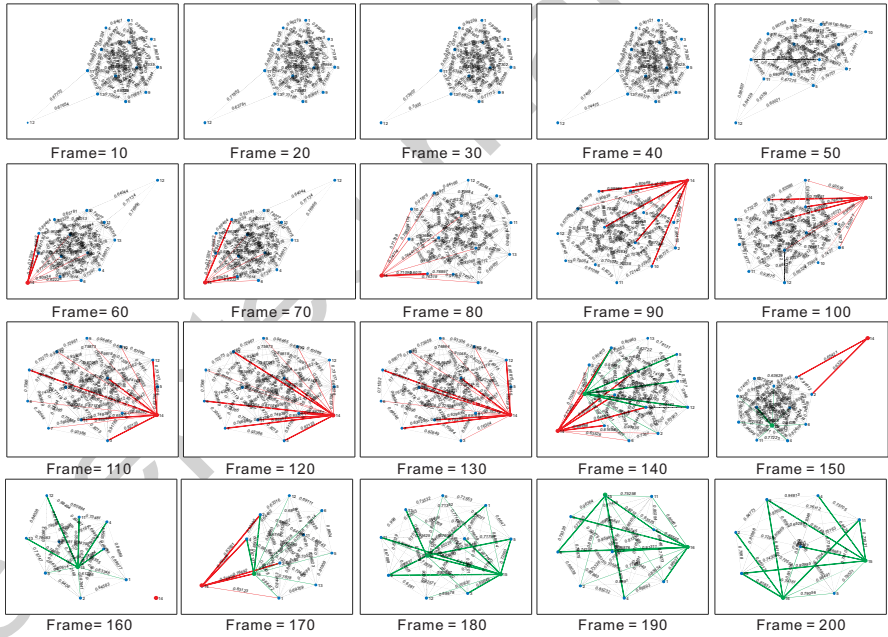
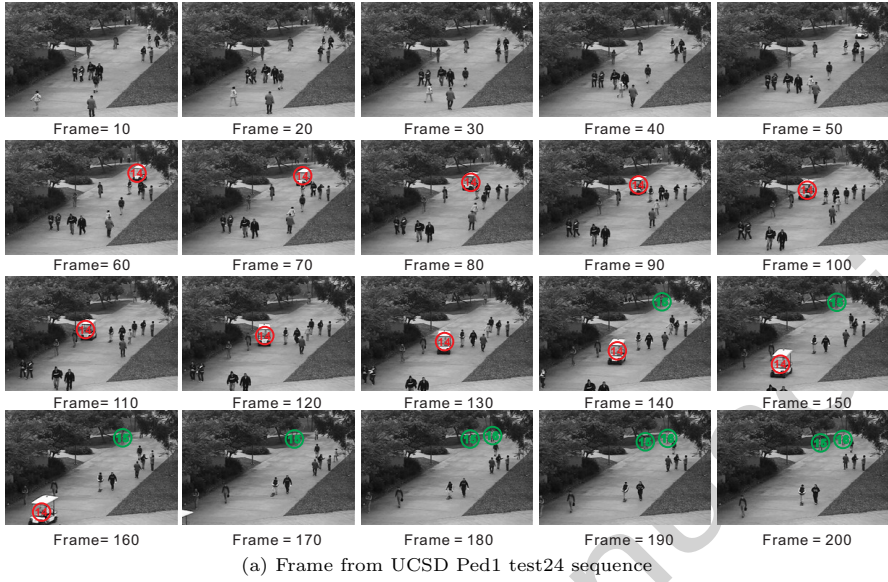


Fig. 6: UCSD Ped1 test24 sequence, threshold $\delta_a = 0.6$ (a) Frame=10 illustrates the motion from frame 1 to frame 10, Frame=20 illustrates the motion from frame 11 to frame 20 and so on, until Frame=200. New incoming nodes (nodes 14, 15 and 16) are highlighted in colour. (b) The associated graph, where the vertex is the node id, and each edge is the similarity measure value $s_{\phi mv}$. When the value $s_{\phi mv}$ is smaller than the threshold δ_a , the edge is deleted, in order to eliminate uninformative edges.

Table 1: Details of the used Dataset.

Datasets	Dominant Features
UCSD Ped1	Velocity-Direct-Magni $s_{\phi mv}$
UCSD Ped2	Velocity-Direct-Magni $s_{\phi mv}$
Avenue	Velocity-Direct-Magni $s_{\phi mv}$
MCG	Direct-Magni $s_{\phi m}$

Table 2: Time Window Experiment on UCSD Ped1 Test24 Dataset

Window	AUC
10	91.3%
20	83.4%
30	79.1%
50	64.20%

8.3 Evaluation

To evaluate anomalous behavior detection, we use ROC curves as our evaluation criteria. Each ROC curve is generated by changing the threshold δ_a in Algorithm 2. Next, we use two quantitative indexes, the Area Under the ROC Curve (AUC) and the Equal Error Rate (EER) to analyze the results. AUC is the area under the ROC curve and EER is the value of the ratio of misclassified items, which means the FPR equals the miss rate ($FPR = 1 - \text{true positive rate (TPR)}$).

The UCSD Ped1 and MCG datasets are both tested at the frame-level and pixel-level. Since UCSD Ped2 does not provide pixel-level detection ground truth, it is tested only at the frame-level. In terms of frame-level detection, a frame is labeled as abnormal if there is at least one pixel that is considered to be an anomalous object. So abnormal event detection at the frame-level is used to detect these frames. For pixel-level based detection, if the detected pixels are more than 40% of the real abnormal pixels, this detection will be identified as a true positive. On the other hand, if at least one pixel in a normal frame is detected as anomalous, it is identified as a false positive.

8.3.1 UCSD Ped1 Datasets

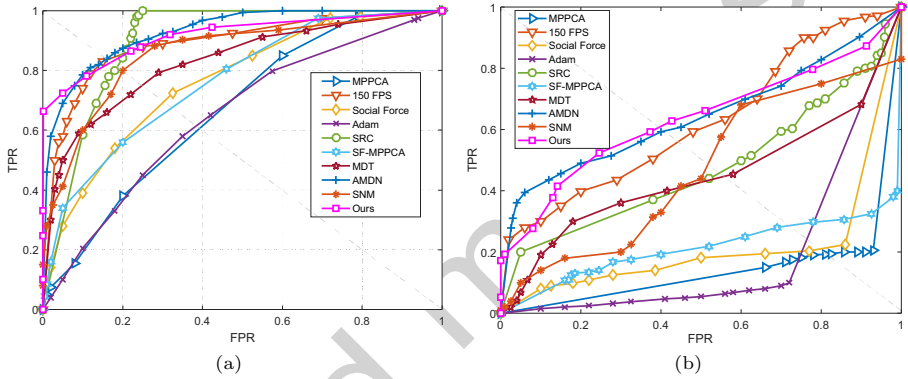
We take the UCSD Ped1 test24 sequence as an example to show how our approach works. UCSD Ped1 test24 is a 200-frame sequence involving a car anomaly. We test on both the frame-level and pixel-level. As discussed above, the UCSD dataset can be characterized as a velocity feature $s_{\phi mv}$ dominant application, so we set the parameter β in Equation (8) to be 0.9.

As shown in Fig. 6-a, Frame=10 illustrates the motion from frame 1 to frame 10, Frame=20 indicates the motion from frame 11 to frame 20, and so on up to Frame=200. The new incoming nodes are highlighted in color as nodes 14, 15 and 16. The corresponding graphs with the threshold $\delta_a = 0.6$ are shown in Fig. 6-b.

The detected suspicious nodes of these frames are listed in Table 3. As we introduced in Section 7 on anomaly detection, **Definition 1** is used to decide the final anomalous nodes. In this case, when a new node appears (nodes 14, 15 and 16), if it becomes a suspicious node in the rest of the graph sequence, we identify it as an anomalous node. Otherwise it is identified as a normal node. For example, node 14 is detected as a suspicious node after it appears, so it is

Table 3: The suspicious nodes detected on the UCSD Ped1 Test24 Dataset

Time Window	Frames	Suspicious Nodes id	Time Window	Frames	Suspicious Nodes id
1	1-10	12	11	101-110	14
2	11-20	12	12	111-120	5
3	21-30	12	13	121-130	14
4	31-40	12	14	131-140	14
5	41-50	12	15	141-150	14
6	51-60	12	16	151-160	14
7	61-70	14	17	161-170	14
8	71-80	14	18	171-180	12
9	81-90	14	19	181-190	11
10	91-100	14	20	191-200	11

**Fig. 7:** ROC Curve for UCSD Ped1 Dataset, (a) Detection at frame-level; (b) Detection at Pixel-level.

identified as abnormal, but node 15 and node 16 are detected as normal. So the detected abnormal frames are the frames that take node 14 as the suspicious node. If the detected suspicious node is an old node (i.e., a node that appears from the beginning), we do not take it as an anomalous node.

For the video sequence we use, the ground truth is: abnormal frames: 51-171 (total: 121), normal frames: 1-50, 172-200 (total: 79). We detect 100 frames of true abnormal frames, with no false detections. Thus, the result for threshold $\delta_a = 0.6$ is $TPR = 82.6\%$, $FPR = 0\%$. This demonstrates how our frame-level detection can achieve good performance. True Positive Rate (TPR) = detected abnormal frames/ground truth abnormal frames = detected abnormal frame / 121 = 100/121 = 82.6%. False Positive Rate (FPR) = falsely detected abnormal frames/ground truth normal frames = falsely detected abnormal frames / 79 = 0/79 = 0%.

In terms of pixel-level detection, one of the advantages of our proposed scheme is that when we detect anomalies at the frame-level, it is not difficult to complete the pixel-level detection. This is because we can locate the anomalies

Table 4: State-of-the-art (SOTA) approaches comparison in terms of AUC and EER. Frame-level and pixel-level detection results on UCSD Ped1 dataset.

Method	Frame-level		Pixel-level	
	AUC	EER	AUC	EER
MPPCA [10]	67.0%	40.0%	19.0%	44.1%
150 FPS [32]	91.8%	15.0%	63.8%	43.0%
SF [33]	76.8%	31.0%	21.3%	71.0%
SRC [34]	86.0%	19.0%	45.3%	54.0%
SF- MPPCA [9]	76.9%	32.0%	21.3%	71.0%
MDT [9]	81.8%	25.0%	44.1%	58.0%
AMDN [17]	92.1%	16.0%	67.2%	40.1%
SNM [28]	85.5%	20%	47.7%	51.2%
FFP [35]	90.6%	17.1%	62.1%	43.8%
sRNN-AE [36]	92.0%	15.3%	66.9%	37.4%
Our Method	91.6%	17.5%	63.5%	39.5%

through the detected anomalous node id, and if we pre-define the number of KLT features to be a suitable value, our approach can detect pixel-level anomalies with high accuracy.

Therefore, we conduct experiments on all of the test sequences with UCSD Ped1 datasets, and the results are shown as the ROC curves in Fig. 7, tested at both the frame-level and pixel-level. Table 4 shows a comparison of our proposed method with existing state-of-the-art (SOTA) methods on the UCSD Ped1 dataset.

From the Fig. 7 and Table 4, it can be seen that both of our frame-level ROC curves and pixel-level ROC curves show competitive performance when compare with the existing approaches. In terms of detection results, we achieved 91.6% AUC and 17.5% EER for frame-level detection, and 63.5% AUC and 39.5% EER for pixel-level detection, which outperforms most of the other state-of-the-art (SOTA) methods. Our accuracy is a little lower than 150 FPS [32], AMDN [17] and sRNN-AE [36], where AMDN and sRNN-AE are deep learning methods whose computational cost are higher. Considering our small model size compared to deep networks, the proposed approach can detect anomalies in real-time and requires little re-training to deploy to other locations.

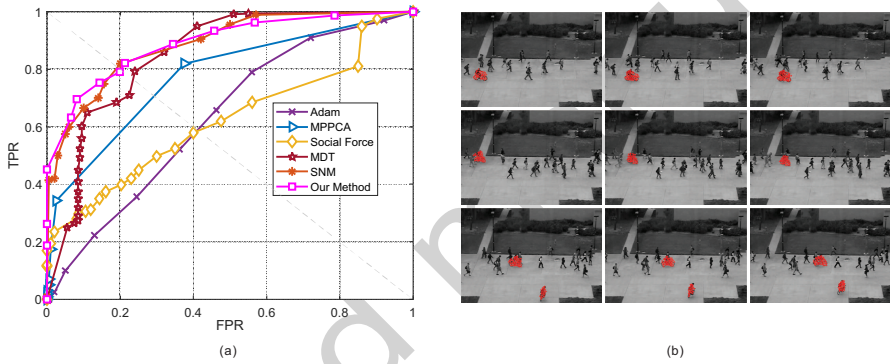
8.3.2 UCSD Ped2 Datasets

Similar to the UCSD Ped1 dataset, the UCSD Ped2 dataset is also a velocity feature $s_{\phi mv}$ dominant application, so we still set the parameter β in Equation (8) to be 0.9. Since the UCSD Ped2 dataset does not provide a ground truth for pixel-level detection, we summarize the results and compare our method with other state-of-the-art (SOTA) approaches based on detection at the frame-level. We conduct experiments and evaluate on all of the test sequences of UCSD Ped2 dataset, and illustrate the results in Fig. 8.

Table 5 shows the results of the comparison of our approach with other SOTA approaches. From these results, it can be seen that the frame-level

Table 5: SOTA approaches comparison in terms of AUC and EER. Frame-level detection results on UCSD Ped2 dataset.

Method	Frame-level	
	<i>AUC</i>	<i>EER</i>
MPPCA [10]	77.0%	29.8%
150 FPS [32]	68.2%	32.6%
SF [33]	63.0%	40.5%
SRC [34]	82.8%	27.0%
SF- MPPCA [9]	71.0%	36.0%
MDT [9]	85.0%	23.5%
AMDN [17]	88.1%	20.4%
SNM [28]	87.9%	20%
FFP [35]	87.4%	22.9%
sRNN-AE [36]	91.23%	18.4%
Our Method	88.3%	17.5%

**Fig. 8:** ROC Curve for UCSD Ped2 Dataset, (a) Detection at frame-level; (b) Example of detection results.**Table 6:** SOTA approaches comparison in terms of computational time on UCSD Ped2 dataset.

Approaches	Computational Time
150 FPS [32]	27FPS
SF [33]	25FPS
AMDN [17]	25FPS
sRNN-AE [36]	25FPS
Our Method	26FPS

based result show competitive performance when compare with the existing approaches. In details, we achieved 88.3% AUC and 17.5% EER for frame-level detection, which outperforms all of the other state-of-the-art (SOTA) methods on the UCSD Ped2 datasets. Only sRNN-AE [36] architecture is competitive when it is compared with our proposed method.

In order to evaluate the computational time used, we did experiment on UCSD Ped2 dataset at frame-level detection, and record the computational time of three existing algorithms, the result is shown in Table 6. It can be seen that most of the existing approaches can achieve the real-time requirement. As the proposed architecture can be used in real-time environments without training, it is still quite competitive.

8.4 Avenue Datasets

In this subsection, we utilized Avenue dataset [32] as evaluation datasets. We evaluated the performance at the frame level, where it does not provide the pixel-level based ground truth. Table 7 shows the quantitative comparison result with the existing state-of-the-art approaches. The experimental parameter used in this dataset is the same as other datasets in previous subsections.

Table 7: Comparison of results of our method with other existing algorithms evaluated on the Avenue dataset. The results include anomaly detection frame-level anomaly detection in terms of AUC and EER.

Approaches	AUC	EER
150 FPS [32]	80.2%	27.7%
CAE [37]	70.2%	25.1%
FFP [35]	84.7%	-
DAF [38]	84.6%	-
AMDN [17]	78.2%	26.5%
LSTM-AE [39]	75.4%	-
ConvLSTM-AE [37]	77.2%	-
NMC [40]	88.7%	-
sRNN [41]	81.6%	-
sRNN-AE [36]	83.2%	-
Our Method	85.6%	23.15%

In details, we achieved 85.6% AUC and 23.15% EER for frame-level detection. We can see that the our method is the second best in terms of AUC, only one approach NMC [40] with 88.7% is better than us. For EER result, our method outperforms all of existing methods.

8.4.1 MCG Dataset

The MCG is a crowded real-world dataset, where standing or loitering are considered as anomalous movement patterns. Consequently, we use the direction-magnitude dominant feature $s_{\phi m}$ as the similarity measure. We assume the importance of the direction and magnitude features is the same, so we set the parameter α in Equation (7) to be 0.5. The comparison result with the existing state of the art algorithms is shown in Fig. 9

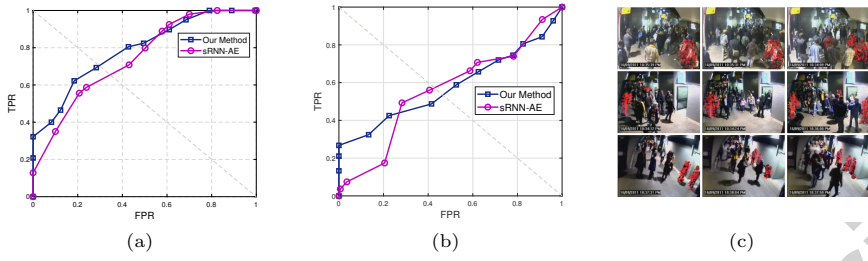


Fig. 9: ROC Curve for MCG Dataset, (a) Detection at frame-level; (b) Detection at pixel-level; (c) Example of detection results.

We conduct an experimental evaluation on the 16-Sep-2011-C2, 16-Sep-2011-C5 and 16-Sep-2011-C6 videos of the MCG dataset. Each video is a five-hour video, we only take 30 minutes for each of the experiments. The results are summarized and shown as ROC curves. The experimental steps are the same for the UCSD dataset, where the ROC curve is generated by changing the threshold δ_a in Algorithm 2. We illustrate the results and the sample detection results in Fig. 9. The AUC is 79.8% and 58.2% at the frame-level and pixel-level, where the best-performance method [36] in UCSD dataset achieves 74.0% and 55.8%, respectively. The evaluation result shows that our method can perform better in such a crowded scenario.

9 Conclusion

In this paper, the crowd movement behavior is modeled as a time-varying graph sequence. At first, each moving individual in the video is denoted as a node in the graph, and then we proposed different types of similarity measures according to the type of anomalies or behaviors of interest. The proposed framework uses a max-flow/min-cut surveillance (MFMCPoS) scheme to detect graph component(s) that corresponds to bottlenecks in the graph network, where depicts the individuals or groups with anomalous movement patterns. In order to evaluate the proposed method, we did experiments on three benchmark datasets and one private dataset. The experimental results demonstrate that the our framework achieves superior detection performance compared to other recently published state-of-the-art methods, where it achieves 91.6% and 63.6% AUC at frame-level and pixel-level in UCSD Ped1 dataset, 88.3% in UCSD Ped2 dataset, 85.6% in Avenue dataset. In some cases, our proposed framework delivers highest crowd anomaly detection rates. Considering that our proposed approach has relatively low computational complexity and can be used in real-time environments, it is crucial for present day video surveillance.

10 Acknowledgements

The authors are very grateful to Editor and the anonymous reviewers for their valuable comments and suggestions that improved the presentation and quality of this paper highly. This work was supported by the Natural Science Foundation of China under Grants 12201523, and also supported by the Fundamental Research Funds for the Central Universities under Grants No. 2682021CX078.

11 Declarations

Conflict of interest The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability statement The datasets and code generated during and analysed during the current study are available from the corresponding author on reasonable request.

References

- [1] Rao, A.S., Gubbi, J., Marusic, S., Palaniswami, M.: Estimation of crowd density by clustering motion cues. *The Visual Computer* **31**, 1533–1552 (2015)
- [2] Li, L., Huang, W., Gu, I.Y.-H., Luo, R., Tian, Q.: An efficient sequential approach to tracking multiple objects through crowds for real-time intelligent cctv systems. *IEEE Trans. on Systems, Man, and Cybernetics, Part B* **38**, 1254–1269 (2008)
- [3] Yang, M., Rashidi, L., Rajasegarar, S., Leckie, C.: Graph stream mining based anomalous event analysis. In: *Pacific Rim International Conference on Artificial Intelligence*, pp. 891–903 (2018). Springer
- [4] Yang, M., Rashidi, L., Rajasegarar, S., Leckie, C., Rao, A.S., Palaniswami, M.: Crowd activity change point detection in videos via graph stream mining. In: *IEEE Conference on Computer Vision and Pattern Recognition Workshop*, pp. 215–223 (2018)
- [5] Moshtaghi, M., Rajasegarar, S., Leckie, C., Karunasekera, S.: An efficient hyperellipsoidal clustering algorithm for resource-constrained environments. *Pattern Recognition* **44**(9), 2197–2209 (2011)
- [6] Bouguet, J.-Y.: Pyramidal implementation of the affine Lucas Kanade feature tracker description of the algorithm. *Intel Corporation* **5**, 4 (2001)
- [7] Mo, X., Monga, V., Bala, R., Fan, Z.: Adaptive sparse representations for video anomaly detection. *IEEE Trans. on Circuits and Systems for Video Technology* **24**, 631–645 (2014)

- [8] Mehran, R., Oyama, A., Shah, M.: Abnormal crowd behavior detection using social force model. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 935–942 (2009). IEEE
- [9] Mahadevan, V., Li, W., Bhalodia, V., Vasconcelos, N.: Anomaly detection in crowded scenes. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 1975–1981 (2010). IEEE
- [10] Kim, J., Grauman, K.: Observe locally, infer globally: a space-time mrf for detecting abnormal activities with incremental updates. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 2921–2928 (2009). IEEE
- [11] Bansod, S.D., Nandedkar, A.V.: Crowd anomaly detection and localization using histogram of magnitude and momentum. *The Visual Computer* **36**(3), 609–620 (2020)
- [12] Farooq, M.U., Saad, M.N.M., Khan, S.D.: Motion-shape-based deep learning approach for divergence behavior detection in high-density crowd. *The Visual Computer* **38**(5), 1553–1577 (2022)
- [13] Sabih, M., Vishwakarma, D.K.: Crowd anomaly detection with LSTMs using optical features and domain knowledge for improved inferring. *The Visual Computer* **38**(5), 1719–1730 (2022)
- [14] Patel, A.S., Vyas, R., Vyas, O., Ojha, M., Tiwari, V.: Motion-compensated online object tracking for activity detection and crowd behavior analysis. *The Visual Computer*, 1–21 (2022)
- [15] Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: Towards real-time object detection with region proposal networks. In: Advances in Neural Information Processing Systems, pp. 91–99 (2015)
- [16] Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, pp. 1097–1105 (2012)
- [17] Xu, D., Ricci, E., Yan, Y., Song, J., Sebe, N.: Learning deep representations of appearance and motion for anomalous event detection. In: Proceedings of the British Machine Vision Conference 2015, BMVC, pp. 801–812 (2015)
- [18] Sabokrou, M., Fayyaz, M., Fathy, M., Moayed, Z., Klette, R.: Deep-anomaly: Fully convolutional neural network for fast anomaly detection in crowded scenes. *Computer Vision and Image Understanding* (2018)

- [19] Morais, R., Le, V., Tran, T., Saha, B., Mansour, M., Venkatesh, S.: Learning regularity in skeleton trajectories for anomaly detection in videos. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 11996–12004 (2019)
- [20] Dubey, S., Boragule, A., Gwak, J., Jeon, M.: Anomalous event recognition in videos based on joint learning of motion and appearance with multiple ranking measures. *Applied Sciences* **11**(3), 1344 (2021)
- [21] Ravanbakhsh, M., Nabi, M., Mousavi, H., Sangineto, E., Sebe, N.: Plug-and-play CNN for crowd motion analysis: An application in abnormal event detection. In: Winter Conf. on Applications of Computer Vision (WACV), pp. 1689–1698 (2018). IEEE
- [22] Xu, K., Sun, T., Jiang, X.: Video anomaly detection and localization based on an adaptive intra-frame classification network. *IEEE Transactions on Multimedia* **22**(2), 394–406 (2019)
- [23] Lucas, B.: An iterative image registration technique with an application to stereo vision. In: DARPA Image Understanding Workshop, pp. 121–130 (1981)
- [24] Zhou, B., Wang, X., Tang, X.: Random field topic model for semantic region analysis in crowded scenes from tracklets. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 3441–3448 (2011). IEEE
- [25] Tomasi, C., Kanade, T.: Detection and tracking of point features. CMU Technical Report (1991)
- [26] Bay, H., Tuytelaars, T., Van Gool, L.: SURF: Speeded up robust features. In: 9th European Conference on Computer Vision, pp. 404–417 (2006). Springer
- [27] Zeppelzauer, M., Zaharieva, M., Mitrovic, D., Breiteneder, C.: A novel trajectory clustering approach for motion segmentation. In: Intern Conf. on Multimedia Modeling, pp. 433–443 (2010). Springer
- [28] Chaker, R., Al Aghbari, Z., Junejo, I.N.: Social network model for crowd anomaly detection and localization. *Pattern Recognition* **61**, 266–281 (2017)
- [29] Dantzig, G., Fulkerson, D.R.: On the max flow min cut theorem of networks. *Linear Inequalities and Related Systems* **38**, 225–231 (2003)
- [30] Schroeder, J., Guedes, A., Duarte Jr, E.P.: Computing the minimum cut and maximum flow of undirected graphs. *Relatório Técnico RT-DINF 003/2004* (2004)

- [31] Li, W., Mahadevan, V., Vasconcelos, N.: Anomaly detection and localization in crowded scenes. *IEEE Trans. on Pattern Analysis and Machine Intelligence* **36**, 18–32 (2014)
- [32] Lu, C., Shi, J., Jia, J.: Abnormal event detection at 150 FPS in Matlab. In: *IEEE International Conference on Computer Vision*, pp. 2720–2727 (2013)
- [33] Reddy, V., Sanderson, C., Lovell, B.C.: Improved anomaly detection in crowded scenes via cell-based analysis of foreground speed, size and texture. In: *IEEE Conference on Computer Vision and Pattern Recognition Workshop*, pp. 55–61 (2011). IEEE
- [34] Cong, Y., Yuan, J., Liu, J.: Sparse reconstruction cost for abnormal event detection. In: *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3449–3456 (2011). IEEE
- [35] Liu, W., Luo, W., Lian, D., Gao, S.: Future frame prediction for anomaly detection—a new baseline. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6536–6545 (2018)
- [36] Luo, W., Liu, W., Lian, D., Tang, J., Duan, L., Peng, X., Gao, S.: Video anomaly detection with sparse coding inspired deep neural networks. *IEEE transactions on pattern analysis and machine intelligence* **43**(3), 1070–1084 (2019)
- [37] Hasan, M., Choi, J., Neumann, J., Roy-Chowdhury, A.K., Davis, L.S.: Learning temporal regularity in video sequences. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 733–742 (2016)
- [38] Smeureanu, S., Ionescu, R.T., Popescu, M., Alexe, B.: Deep appearance features for abnormal behavior detection in video. In: *International Conference on Image Analysis and Processing*, pp. 779–789 (2017). Springer
- [39] Luo, W., Liu, W., Gao, S.: Remembering history with convolutional lstm for anomaly detection. In: *2017 IEEE International Conference on Multimedia and Expo (ICME)*, pp. 439–444 (2017). IEEE
- [40] Ionescu, R.T., Smeureanu, S., Popescu, M., Alexe, B.: Detecting abnormal events in video using narrowed normality clusters. In: *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 1951–1960 (2019)
- [41] Luo, W., Liu, W., Gao, S.: A revisit of sparse coding based anomaly detection in stacked RNN framework. In: *Proceedings of the IEEE International*

Conference on Computer Vision, pp. 341–349 (2017)

Accepted manuscript