Cluster-based Crowd Movement Behavior Detection

Meng Yang[1] , Lida Rashidi[1], Aravinda S. Rao [2], Sutharshan Rajasegarar[3],
Mohadeseh Ganji[1], Marimuthu Palaniswami[2], Christopher Leckie[1]
[1]School of Computing and Information Systems, The University of Melbourne, Australia.
[2]Dept. of Electrical and Electronic Eng., The University of Melbourne, Australia.
[3]School of IT, Deakin University, Geelong, Australia.
Email: {myang3@student., rashidil, aravinda.rao, palani, caleckie}@unimelb.edu.au,
srajas@deakin.edu.au, mohadeseh.ganji@gmail.com

***Abstract*—Crowd behaviour monitoring and prediction is an important research topic in video surveillance that has gained increasing attention. In this paper, we propose a novel architecture for crowd event detection, which comprises methods for object detection, clustering of various groups of objects, characterizing the movement patterns of the various groups of objects, detecting group events, and finding the change point of group events. In our proposed framework, we use clusters to represent the groups of objects/people present in the scene. We then extract the movement patterns of the various groups of objects over the video sequence to detect movement patterns. We define several crowd events and propose a methodology to detect the change point of the group events over time. We evaluated our scheme using six video sequences from benchmark datasets, which include events such as walking, running, global merging, local merging, global splitting and local splitting. We compared our scheme with state of the art methods and showed the superiority of our method in accurately detecting the crowd behavioral changes.**

***Index Terms*—crowd behavior, event change detection, fixed-width clustering, video surveillance**

I. Introduction

In recent years, crowd behavior monitoring and prediction has become an important topic in video surveillance. Although people typically behave in ordinary ways, an occurance of a particular event can cause a panic. Therefore, ensuring the security and safety of the individuals in public places such as crowded streets, sports events, or shopping malls in cases of emergencies (terrorism, collapse, fire.) warrants attention. The aim of this paper is to provide an automated video analytics approach for crowd event detection.

An important objective of crowd monitoring is to provide a logical model of people movements in case of an unexpected event, i.e., path planning of evacuation. Researchers [2] have shown the importance of automatic analysis for crowd behavior and its applications, such as crowd management, which is about strategies for building evacuation in case of disaster, and alerting authorities when detecting unexpected events captured through video surveillance footage. Crowd monitoring poses many challenges due to varying environments, such as irregular illumination conditions, shadows, non-rigid pedestrian movement and occlusions [4]. There have been several existing approaches applied to video surveillance analytics, however crowd behaviour understanding is still a challenge. Many methods are based on object detection and tracking [3], but tracking multiple individuals in crowded scenarios is a very challenging task. Therefore, in this paper, we propose a scalable method that can be used for crowd movement change detection, like sudden running, merging and splitting.

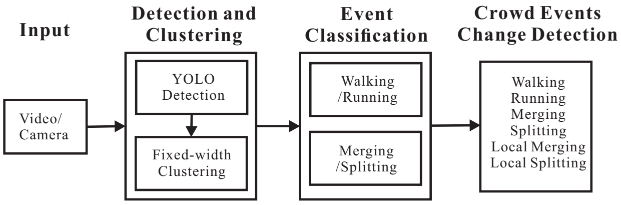


Fig. 1. Overview of our crowd monitoring system, which includes video/image input (Step1), detection and clustering (Step2), event classification (Step3) and crowd event change detection (Step4).

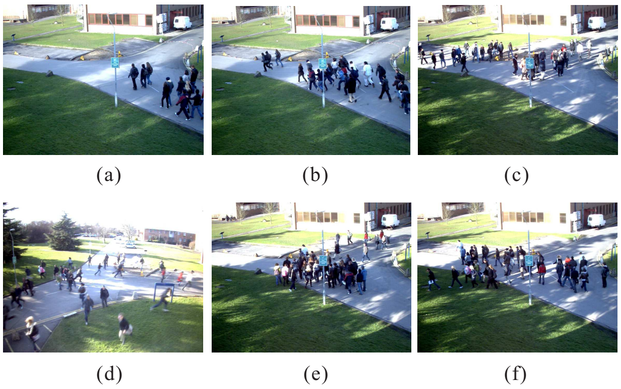


Fig. 2. Sample frames of crowd events from PETS2009 dataset. (a) Walking: people walk across the scene from right bottom to left top. (b) Running: people start running at a specific frame. (c) Merging: people from different directions merge at the center of the scene. (d) Splitting: a crowd of people split into three different small groups. (e) Local Merging: two groups of people merge at first, and then walk to the left border as one group. (g) Local Splitting: a group of people moves toward the center of screen, then splits into two smaller groups.

Our proposed architecture is based on object detection associated with fixed-width clustering. To detect multiple objects in the scene, we use a deep learning based system You only look once (YOLO), which is a real-time object detection system [8,11]. Once we detect objects in every frame, we represent the objects in a two dimensional Euclidean space, which enables the use of any clustering algorithm on the data for analysis. The clusters are then used to detect abnormal behaviour, i.e., crowd change detetcion. In this paper, we used a fixed-width clustering algorithm [13, 14] to cluster the people into several groups (clusters) who are close to each other. The cluster centroids are used as the representative of each group of people. We then use the movement behavior of the clusters, captured via features such as the number of clusters, the distances between the centroid of the clusters and the velocity of change in cluster centroid positions, to detect various crowd event changes, such as merging, splitting and running events.

A brief overview of our proposed real-time event detection framework is shown in Fig. 1, where we first use YOLOv2 [11] for object detection in each frame, and the obtained objects are represented as the feature vectors at a spatial level. We then apply our proposed fixed-width clustering algorithm to find the group of objects in the frame, and then use the adjacent frames of the video sequence for event detection. The events that we focus on can be classified into two main categories, walking/running and merging/splitting. The merging and splitting event is further classified into local and global events. Each video sequence is then converted into a time series for crowd event change detection. Our framework is capable of detecting events such as walking, running, crowd formation (merging), scattering/splitting and local merging detection, as shown in Fig. 2. The main contributions of this paper are as follows:

1) Our proposed method is real-time since it can detect the events as soon as they occur. The framework makes use of only the detected objects coordinates for event detection, which is often in contrast to other existing algorithms that use shape, appearance and even audio information.
2) We use a novel cluster based method to represent groups of objects and define events by analysing the cluster movement profiles in a video sequence.
3) We conduct experiments on six video sequences of PETS2009, which contain global events: walking, running, merging, splitting and local events: partly merging and splitting. The evaluation reveals that such events can be detected, and the result is competitive when compared with other existing state-of-the-art algorithms. The quantitative accuracy for all six sequences are all more than 80%.

The rest of the paper is organized as follows. Section II reviews previous algorithms for video-based crowd event detection, and Section III presents the problem statement. Further, our proposed methodology is introduced in Section IV, and Section V shows the experimental results on six sequences from PETS2009 at a global level and local level. Finally, the conclusion is given in Section VI.

II. RELATED WORK

There are several existing methods for crowd behavior analysis. In this section, we review the approaches for anomalous event change detection, which are more related to the aim of this paper. In order to address this task, spatio-temporal information based methods are quite popular, and these approaches can be grouped into two types: trajectory analysis and motion representation. Trajectory analysis based algorithms use tracking to obtain tracks and then distinguish abnormal ones from normal tracks in the scenario [5]. Motion representation methods use flow analysis to monitor events, for instance, the architecture in [18] uses Lagrangian particle dynamics with chaotic modeling to monitor and localize the abnormal objects and activities in crowded scenes. Further, representative tracks are identified and used as the compact modeling features. These trajectories produce temporal data that can be utilized for chaotic modeling in a simple way. The approach in [1] builds an optical flow related framework based on the stability of a dynamical model that can detect pre-defined events in the crowd. Moreover, Su et al. [6] used the spatio-temporal volume for variation matrix calculation, whose corresponding eigenvalues are obtained to describe local fluctuations.

Another class of approaches investigate theoretical models of crowd dynamics for event analysis. In [17], a social force model (SFM) is explored for anomalous activity detection and localization. The authors used the interaction of particles from optical flow information, which is estimated using the SFM, and these features are used for event analysis. Chen et al. [19] used optical flow based clustering for human crowd grouping in an unsupervised manner, and this method is called adjacency-matrix-based clustering (AMC). Zhou et al. [7] provided a novel mixture model of dynamic human agents (MDAs) that could learn the collective behaviour of people. The detailed approach is based on a bag of trajectories for MDA learning, then measuring the likelihoods of the tracks with the MDAs for anomalous behaviour detection. In [20], a Bayesian model is proposed for detecting escape events in crowded scenarios. First, foreground pixels and optical flows are extracted and used for crowd behavior characterisation, which can model escape and non-escape crowd behaviour using a Bayesian model.

As discussed above, most crowd event detection strategies in this domain rely on either tracking algorithms or complex modeling techniques. For instance, the authors of [5] used a tracking method to follow individuals in a crowded scene. However, it suffers from high computational complexity, and the accuracy of tracking based algorithms is low, because of overlapping and occlusion. In contrast, modelling-based strategies have been proposed, such as SFM [17]. Although such methods can achieve accurate crowd event detection, the number of parameters to be tuned is very large. Tuning such parameters relies on domain knowledge or pre-processing techniques.

III. PROBLEM STATEMENT

In order to achieve real-time crowd event detection, our proposed scheme utilises an unsupervised clustering method to characterize human behaviour. The groups of people in a scene are clustered, and the evolution of the clusters are used

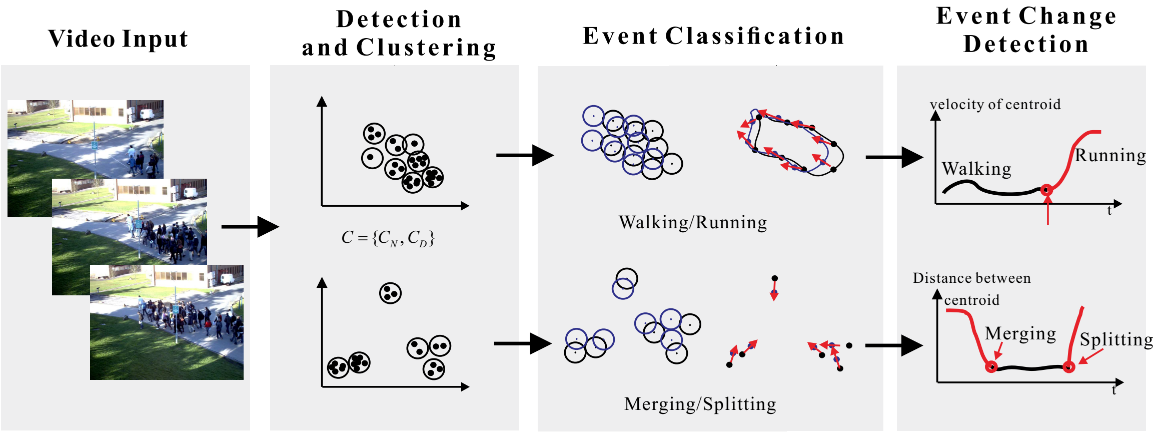


Fig. 3. The steps involved in the proposed crowd event detection approach. As the video sequence is presented (step1), the pedestrians are detected and clustered (step2), and then the numbers and direction of the center of each cluster is recorded and used for event classification. The blue nodes denote the nearest cluster of each cluster in the adjacent frame (step3). Finally, cluster features are used to produce the time series event curve and detects any change in the curve (step4).

to detect the change in crowd behavior. A brief overview of our proposed monitoring system is shown in Fig. 1. Step1: the video/image sequences are provided as input. Step2: The objects/people present in the image are detected and the groups of objects are found using a clustering method. Step3: Various events of crowd activity are detected, such as walking/running and merging/splitting; and Step4: changes in the crowd events are detected using a novel change detection method.

Consider an input as a video sequence $Vid = \{f_i, i = 1,..,t\}$, where f_i is the frame number. The aim is to find the time when an event change happens $P(f_{t'+1})$, $t' = 1...t$. In each frame f_i, we denote the detected objects as $S = \{s_j, j = 1....n\}$, where n is the number of detected objects. For each pedestrian, we take the centroid of the detected bounding box as the coordinates of this object, which is described using $o_i = (x_i, y_i)$, followed by being represented as a spatial feature $M_i = \{m^i_k, k = 1...n_b\}$, where n_b is the block size. We used block sizes of 64 (i.e., the frame is divided into 8 by 8 blocks), 256 (16 by 16) and 1024 (32 by 32). The feature matrix of all the incoming frames $M = \bigcup_{i=1,...,n} M_i$, is used for clustering, using a fixed-width clustering algorithm.

The PETS2009 dataset [9] is used for evaluation in this paper. The six different types of crowd events that we analyze are as defined in [10]. The process of our proposed architecture is summarized in Fig. 3. After obtaining the clusters, the events are detected. They are categorized into two main types: *Walking/Running* and *Merging/Splitting*. Then, in the later stage, these events are further classified into four sub-types: *Global Merging*, *Global Splitting*, *Local Merging*, and *Local Splitting*. Finally, each video sequence is converted to a time series for crowd event change detection. The final output provides the time instance of a particular $P(f_{t'+1})$.

IV. EVENT DEFINITION

A. Walking and Running Event

(1) Walking: This is a human behavior where a group of pedestrians move at a normal (average) velocity, which is much slower than the velocity of running events.

(2) Running: This event is associated with the above walking event, but with a faster group movement.

Definition1: In order to detect a (group) event, we consider two adjacent video frames in the video sequence. In order to track the movement of a particular cluster (a group of objects/people) from the first frame to the second frame, we need to find the association of the same cluster in both frames. We find this associated cluster in the second frame using the nearest cluster principle, i.e., we find the nearest cluster in the second frame, and treat that as the associated cluster of the first frame's corresponding cluster. We then record the change in direction, C_D, of the center of each cluster, and use this to define events based on the following criteria: If the angle between more than 80% of the pairs of clusters in the adjacent frames movement direction is smaller than 90°, then this event is defined as Walking/Running (W/R).

B. Merging and Splitting Events

(1) Global Merging: Pedestrians from different directions converging to the same point in the scene. This is denoted as $M = \{M_G\}$, where M_G is for global crowd merging.

(2) Global Splitting: This is the opposite event compared with merging, where people from the same crowd move to different points of the scene, and is denoted as S_G.

(3) Local Merging: This event is a more complicated activity compared with global group merging. In some sequences of PETS2009, it is observed that some pedestrians merge at first, and walk as one group, then others come to merge. In this case, we use M_L to represent this event, so the merging event can be denoted as $M = \{M_G, M_L\}$.

(4) Local Splitting: Similar with local merging, a local splitting event is shown as a group of people who move toward the center of the screen, then a small part of this group split into another group, the rest of the original group walk until they split at another point. We denote this event as S_L, so the whole splitting event is denoted as $S = \{S_G, S_L\}$.

Algorithm 1 Event Change Detection System

```
Require: Video Sequence $Vid = \{f_i, i = 1, .., t\}$
while $i \leq 2$ do
    YOLO Detection : For each detected pedestrian $S = \{s_j, j = 1....n\}$, the recorded coordinate is $\mathrm{o_j} = (\mathrm{x_j}, \mathrm{y_j})$.
    Fixed − width Clustering : Representing vectors $v_j$ and clusters $\mathrm{C} = \{\mathrm{c_r} : \mathrm{r} = 1...\mathrm{N_c}\}$.
    Event Classification :
        Walking/Running $\leftarrow Definition1$
        Merging/Splitting $\leftarrow Definition2$
end while
Ensure: $W/R$ or $M = \{M_G, M_L\}/S = \{S_G, S_L\}$
while $2 < i < t$ do
    YOLO Detection.
    Fixed − width Clustering.
    Event Change Detection :
    if $W/R$ then
        y − axis value $\leftarrow$ calculate the velocity of $C$
        Event change point $\leftarrow$ sharply rise/decline
    else
        if $M_G/S_G$ then
            y − axis value $\leftarrow$ calculate the distance of $C$
            Event change point $\leftarrow$ sharply rise/decline
        else
            Locally merge $\leftarrow C_N$ of $G_i$ decreases to 0
            Locally split $\leftarrow C_N$ of $G_i$ increases
        end if
    end if
    $i = i + 1$
end while
Ensure: Event detected $\mathrm{P(f_{t'+1})}$
```

Definition2: Similar to Definition1, we consider two adjacent video frames in the video sequence. We find the associated cluster in the second frame using the nearest cluster principle, i.e., we find the nearest cluster in the second frame, and treat that as the associated cluster of the first frame's corresponding cluster. We then record the change in direction, C_D, of the center of each cluster, and use this to define events based on the following criteria: we randomly take one cluster's direction C_D and calculate the angles between the more distant cluster's direction with itself. If there is more than one angle that is larger than 90°, this event is defined as a M/S event.

Definition1 and Definition2 are used for classifying W/R and M/S events, which are based on the first two adjacent frames of a video sequence. Definition3 and Definition4 that we present below are used to find the event occurrence time of global merging and local merging, which are based on the whole video sequence. The number of clusters C_N should be considered along with the cluster direction change C_D, which are described below in Definition3 and Definition4.

Definition3: We randomly take one cluster, and record the number C_N and the direction C_D of the clusters that are more distant from its surrounding ones. These more distant clusters could be identified as different groups $G = \{G_i, i = 1...n_g\}$, where n_g represents the number of groups. If all of the distant clusters in G decrease to 0, it means all groups of people have merged together. The time/frame is identified as the event occurrence point for global merging.

Definition4: We randomly take one cluster, and record the number and the direction of the clusters that are more distant from its surrounding ones. These more distant clusters can be considered as different groups $G = \{G_{i}, _{j}, _{k} ..., \ i, j, k = 1...n_g\}$, where n_g is the number of groups. If the number of clusters C_N for any one group in G, like G_k decreases to 0, it indicates that two groups of people have merged together. The time/frame is identified as the event occurrence point for local merging.

V. Methodology

In this section, we describe the various components of our proposed framework in detail. A brief overview of our proposed monitoring system is shown in Fig. 1, where we first use YOLOv2 [11] for object detection in each frame, and the detected objects are clustered using our proposed fixed-width clustering approach. We then use each pair of adjacent frames in the video sequence and Definition1 for the event definition. In order to detect the W/R event, we consider the time series curve of velocity of the centroid of each cluster, and detect any sharp rise or fall, i.e., change of the curve. An example curve is shown in Fig 3 that shows a W/R event detected using change detection of the velocity curve. For the M/S event type, based on Definition3, when the video sequence is identified as M_G/S_G, we calculate the Euclidean distance of each cluster and take the average value as the y-axis value of the curve. Like the sample curve shown in Fig. 3, our system can do M_G/S_G event change detection and event monitoring. About the M_L/S_L type, which is automatically classified using Definition4, if C_N for one G_i decreases to 0, it means two groups of people merged at first. The whole process is summarized in **Algorithm 1**.

A. YOLO Object Detection

YOLO is a state-of-the-art, real-time object detection system. In this paper, we use YOLOv2 as our object detection tool [11]. In terms of many existing detection systems, they repurpose classifiers for detection, and apply the architecture to each frame at multiple scales and locations. Then the high scoring regions are recognized and used for detection.

In the YOLO scheme, it divides the image into an $S \times S$ grid. For each grid cell, it predicts B bounding boxes and confidence scores. In each grid cell it also predicts C conditional class probabilities that could be described as $\Pr(\mathrm{Class}_i \,|\mathrm{Object})$. The C are conditional on the grid cell that contains the object. During the testing phase, the class-specific confidence score for every box is calculated using equation (1), based on the conditional class probabilities and the box confidence predictions.

$$\begin{aligned}&\mathrm{Pr(Class}_i\,|\mathrm{Object)} \times \mathrm{Pr(Object)} \times \mathrm{IOU}_{\mathrm{pred}}^{\mathrm{truth}}\\&= \mathrm{Pr(Class}_i) \times \mathrm{IOU}_{\mathrm{pred}}^{\mathrm{truth}}\end{aligned} \tag{1}$$

where $\mathrm{Pr(Object)} \times \mathrm{IOU}_{\mathrm{pred}}^{\mathrm{truth}}$ is defined as the confidence predictions. If no object exists in this grid cell, the confidence score becomes 0. In contrast, the confidence score is equal to the intersection over the union (IOU) between the predicted box and the ground truth, which is denoted as $\mathrm{IOU}_{\mathrm{pred}}^{\mathrm{truth}}$.

B. Fixed-width Clustering

In terms of the feature representation, we use an 8 by 8 block size. For example, if the detected object is at block 26, the fearture is represented as a vector $[0, 0, ..., 1, ..., 0]$, where the value 1 occurs at the 26th position in the vector. Then, the obtained feature matrix $M = \bigcup_{i=1...n} M_i$ is considered as the input for fixed-width clustering.

The purpose of the clustering [13, 14] is to group nearby people at the spatial level. The steps involved in this process are as follows: At first, a fixed-width clustering is used. The initial cluster c_1 is centred on the first vector $M_1 = \{m_k^1, k = 1...n_b\}$ with a fixed cluster radius ω. Then in general for each c_i the Euclidean distance d_i between the nearest vector c_r and itself is computed.

$$d_i = \mathrm{Distance}(c_i,c_r) \tag{2}$$

If the distance d_i is less than the radius ω of the nearest cluster c_r, the vector will be added to that cluster, and the centroid m_r will be updated using the new vector. Otherwise, if it is larger than ω, this vector is used to form a new cluster c_f. We repeat this process until all vectors are considered.

$$\mathrm{update}(c_r, m_r) = \begin{cases} c_{r+1}, & \text{if } d_i \leq \omega \\ C \bigcup c_f, & \text{if } d_i > \omega \end{cases} \tag{3}$$

After obtaining the clusters from the last step, for each pair of clusters c_r and c_j we compute the distance D_r between their cluster centroids. If D_r is smaller than the threshold τ, these two clusters c_r and c_j are merged to form a new cluster c_m.

$$c_m = \mathrm{MergeCluster}(c_r, c_j), \quad \text{if } D_r \leq \tau \tag{4}$$

Finally, the whole cluster set $C = \{c_i : i = 1 \ldots C_N\}$ is obtained, and it is used for crowd event change detection.

C. Event Classification

Considering adjacent frames of the video sequence, for each cluster $C_i^{t_1}$ in Frame 1, the associated cluster in Frame 2 is selected as the nearest cluster $C_i^{t_2}$. The change in direction of each cluster is denoted as $C_{D_i} = \mathrm{Direction}(C_i^{t_1}, C_i^{t_2})$, where $i \in 1...n$, n is the number of clusters.

At first, we aim to classify these events into two main types: W/R and M/S. If it satisfies the following equation, this event will be identified as W/R.

$$\mathrm{Event} = W/R, \text{ if } \{\mathrm{Ang}(C_{D_m}, C_{D_n}) \leq 90°\} \tag{5}$$

where $m, n \in 1...n_p$, $n_p \geq 80\% \times n$, $\mathrm{Ang}(\cdot)$ is the angle between two cluster directions.

Otherwise, i.e., if it does not satisfy equation (5), we then randomly take one cluster C_{D_m} and calculate the angles between the clusters C_{D_f} that are far away from its surrounding ones and itself. If there is more than one angle that is larger than 90°, i.e., $\mathrm{Ang}(C_{D_m}, C_{D_f}) \geq 90°$, then this event will be classified as M/S.

$$\mathrm{Event} = M/S, \text{ if } \{\mathrm{Ang}(C_{D_m}, C_{D_n}) > 90°\} \tag{6}$$

where $m, n \in 1...n_p$, $n_p \geq 2$. After the above step, we still need to further classify if it is a global or local M/S event. For detecting global/local merging, the number of clusters C_N is considered along with the cluster direction change C_D.

To do that, we randomly take one cluster C_{D_m}, and record the number C_{N_f} and the direction of the clusters C_{D_f} that are far away from its surrounding ones. Based on $\mathrm{Ang}(C_{D_m}, C_{D_f})$, these far away clusters can be identified as belonging to different groups $G = \{G_i, i = 1...n_g\}$, where n_g is the number of groups. If C_{N_f} of all G decreases to 0, it reveals that all the groups of people have merged together. This event could be identified as a global merging event M_G. On the other hand, if C_N of one G_i decreases to 0, it means that two groups of people are merged at first. This event could be identified as a local merging event M_L.

We can detect both M/S based on the curve of the centroid distance, which we discuss in the following section.

D. Crowd Event Change Detection

In order to detect a Walking/Running event, we take the average coordinates $\mathrm{AverC} = \{\mathrm{AverC}_i, i = 1...n\}$ of all clusters, where n is the total number of frames. We monitor the velocity change of AverC, and use velocity $V_{W/R} = \{\mathrm{Velo(AverC}_i)_i, i = 1...n\}$ as the y-axis value. The video is converted to a time series with the frame number as the x-axis and Velo(AverC) as the y-axis. Our aim is to detect the specific frame number as the crowd event change point. When the Velo(AverC) changes smoothly, that event is identified as a walking event. If the Velo(AverC) changes sharply, then that event is recognized as a running event. The frame number at which the sudden change in Velo(AverC) occurs is taken as the change point.

In terms of a M_G/S_G event, we use the average distance $V_{M/S} = \{\mathrm{AverDis}(C_m, C_k)_i, i = 1...n\}$ of each pair of clusters as the y-axis value, where C_m and C_k are the clusters. The video will be converted to a time series with $V_{M/S}$ as the y-axis and the frame number as the x-axis. When $V_{M/S}$ is large, the event is identified as splitting, and after a sudden sharply decline, $V_{M/S}$ changes to be a small value, which is recognized as merging. Otherwise, the y-axis value changes from small to large, and the change point is identified as the time when the change occurred.

For a M_L/S_L event, as with M_G/S_G, we take the average distance $V_{M/S} = \{\mathrm{AverDis}(C_m, C_k)_i, i = 1...n\}$ as the y-axis value. We randomly take one cluster C, if the C_N of one G_i decreases to 0 at time t, where $i = 1, ..., l$, and G_i are the

TABLE I

DETAILS OF THE PETS2009 DATA USED. THE FIRST COLUMN OF THE TABLE SHOWS THE ALTERNATIVE NAME OF THESE VIDEO SEQUENCES.

Sequence	Video	Event	Total Frame
Seq1	S3-HL, 14-16, View-001-p1	Walking/Running	107
Seq2	S3-HL, 14-16, View-001-p2	Walking/Running	114
Seq3	S3-HL, 14-33, View-001	Merging/Splitting	377
Seq4	S3-HL, 14-33, View-002	Merging/Splitting	377
Seq5	S3-HL, 14-31, View-001	Local Splitting	130
Seq6	S3-MF, 14-37, View-001	Local Merging	108

groups that are not close to the group C, where C_N is the size of C. This means only two groups merged, which is identified as local merging, and we note the time of this merge. On the other hand, if C_N of one G_i increases sharply, it means a local split has occurred.

VI. RESULT AND DISCUSSION

A. Experimental Setup

Our proposed crowd event detection monitoring system is implemented in Java. Object detection is based on YOLOv2, and we use the pre-trained darnet model directly, which used ordinary public datasets for training. Our system combines the YOLOv2 based detection with the object representation and fixed-width clustering. Finally, the resulting time series curve is obtained and analyzed. We use Windows 7 (64bit) with a 4 GB NVIDIA graphics card (NVIDIA NVS 315 HD), a multi-core Intel? i7-4790 3.6GHz CPU and 16GB RAM.

B. Datasets

The PETS2009 S3-High-Level (S3-HL) and S3-Multiple-Flow (S3-MF) datasets [9] are used for evaluating our proposed approach (refer to Table I).

The crowd events analysis are categorized from the dataset S3-HL and S3-MF dataset. The sequences that we choose are those where there is a change in the motion pattern (speed and/or orientation), such as people suddenly starting to run (change in speed), or merging/splitting from a point (change in direction). The ground truth values are the frame number when the change starts [15, 16].

We conduct experiments on the above video sequences and compare our results with existing state-of-the-art approaches that also perform event detection in such sequences. In the following section, we describe S3-HL, 14-16, View-001, part1 as Seq1, S3-HL, 14-16, View-001, part2 as Seq2, and S3-MF, 14-37, View-001 as Seq6.

C. Parameter Setting

The threshold setting for YOLO is set to 0. For fixed-width clustering, the radius of each cluster is set to be $\omega = 0.5$ for all six video sequences in the PETS2009 dataset. The merging threshold is $\tau = \omega/2$.

In order to show that our results are not sensitive to these parameter settings, we conduct an experiment on Seq1 with varing values of cluster radius ω. Fig. 4 shows how the detection accuracy varies with the chosen cluster width setting ω. When $\omega \in [0.2, 3]$, the accuracy does not change, so we could pick any value within this range. For other cluster radius ω, the accuracy is still quite competitive. We discuss how we obtain these accuracy values in the following subsections.

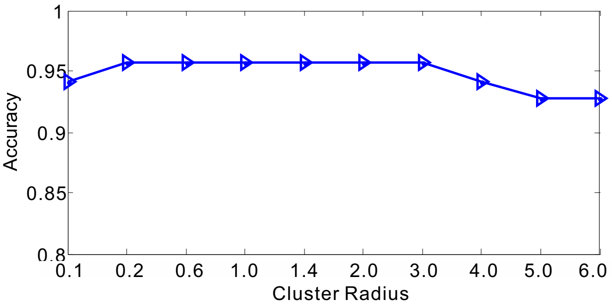


Fig. 4. Parameter setting experiment on Seq1: x-axis is the cluster radius ω, y-axis is the associated detection accuracy.

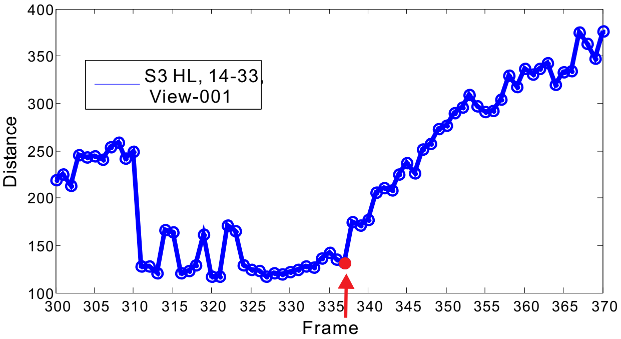


Fig. 5. PETS2009 S3-HL, 14-33, View-001 (Seq3): Sample output time series obtained for merging/splitting event, x-axis is the frame number, y-axis is the average value of Euclidean distance between each pair of clusters. Red node denotes the detected event change point.

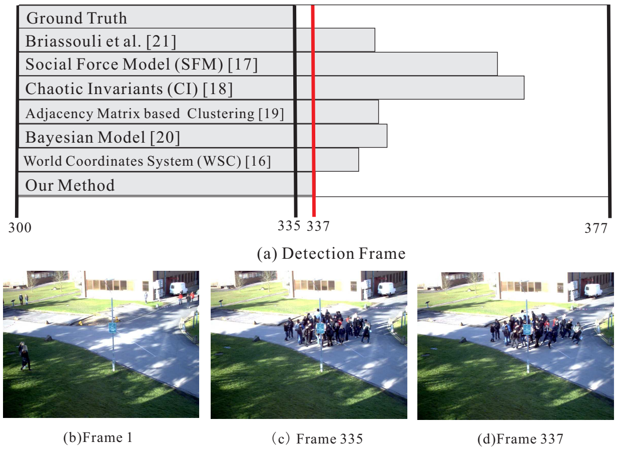


Fig. 6. Seq3, merging/splitting event: (a) Frame detection, baseline with the ground truth, other state-of-the-art methods, and our detected frame marked in red. (b) Frame 1 of the video sequence. (c) Ground truth frame when change starts. (d) The frame that we detected.

D. *Evaluation*

We use the first four sequences for global detection and the latter two sequences for local detection. Fig. 5-11 shows the results of the proposed approach. The evaluation results show that they are all competitive when compared with the other methods.

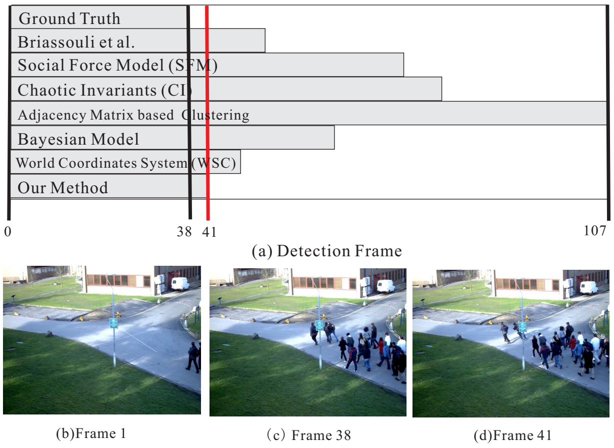


Fig. 7. Seq1, walking/running event: (a) Frame detection, baseline with the ground truth, other state-of-the-art methods, and our detected frame marked in red. (b) Frame 1 of the video sequence. (c) Ground truth frame when change starts. (d) The frame that we detected.

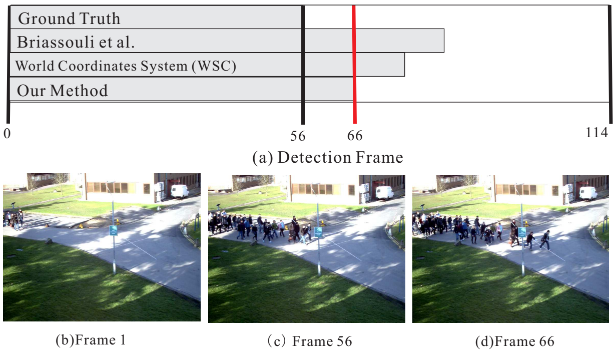


Fig. 8. Seq2, walking/running event: (a) Frame detection, baseline with the ground truth, other state-of-the-art methods, and our detected frame marked in red. (b) Frame 1 of the video sequence. (c) Ground truth frame when change starts. (d) The frame that we detected.

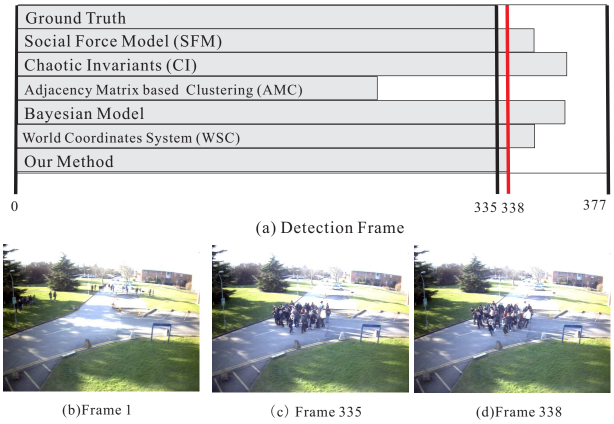


Fig. 9. Seq4, merging/splitting event: (a) Frame detection, baseline with the ground truth, other state-of-the-art methods, and our detected frame marked in red. (b) Frame 1 of the video sequence. (c) Ground truth frame when change starts. (d) The frame that we detected.

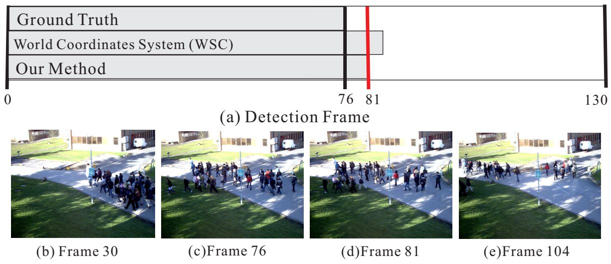


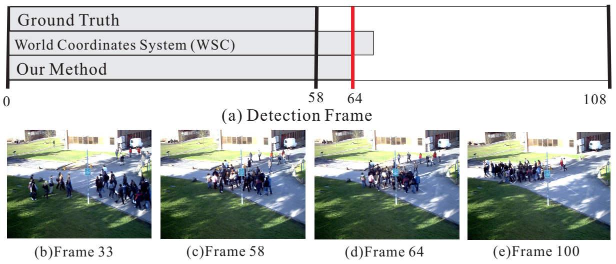
Fig. 10. Seq5: (a) Frame detection, baseline with the ground truth, other state-of-the-art methods, and our detected frame. (b) A frame illustrating the whole scenario. (c) Ground truth frame when local change first starts. (d) The frame that we detected. (e) Another frame describing the whole scenario.

Fig. 5 shows the sample curve output of Seq3, where the y-axis shows the average distance of each pair of clusters and the x-axis is the frame number. The red point of the curve is identified as the event change occurrence point. Fig. 6 shows our detected result, ground truth and the other competitive results on Seq3. Fig. 7-9 demonstrate the evaluation results of the rest of the global event video sequences in the same way as Fig. 6, as (a) the baseline with the ground truth, other state-of-the-art methods, and our detected frame; (b) Frame 1 of the video sequences; (c) Ground truth frame when the change starts; (d) The frame that we detected.

As shown in Fig. 6, the ground truth of event change is Frame 335, and the change that we detected is Frame 337. For PETS2009, it is 7 frames per second, so our detected result is a 0.29s delay after the ground truth. For Fig. 7, the ground truth is Frame 38, and our detected result is Frame 41, so our detected result is a 0.43s delay after the ground truth. The same is observed with Fig. 8-9, it is 1.43s and 0.43 delay respectively. Furthermore, for these four videos sequences, our result is closest to the ground truth when compared with the other existing approaches.

In order to evaluate the result, we define the accuracy to be $1 - \frac{\text{Detected}-\text{Ground Truth}}{\text{Total Frame}-\text{Ground Truth}}$. If the detected frame is closer to the ground truth, the performance is better. The quantitative evaluation is summarized in Table II.

In terms of local event detection, Seq5 shows a crowd splitting event where a group of people moves toward the center of the screen, then splits into two smaller groups (refer to Fig. 10). (a) is the ground truth when the local event first occurs, and the detection frame based on other existing methods and our method; (b) is a sample frame to show the whole event; (c) is the ground truth frame; and (d) is our detected frame; (e) is another sample frame to show the whole event. Our system detected the first splitting activity at frame 81, where the ground truth is at Frame 76, and Frame 83 is the result of the WSC method. We have a has 0.7s delay,



(a) Detection Frame

(b)Frame 33 (c)Frame 58 (d)Frame 64 (e)Frame 100

Fig. 11. Seq6: (a) Frame detection, baseline with the ground truth, other state-of-the-art methods, and our detected frame. (b) A frame illustrating the whole scenario. (c) Ground truth frame when local change first starts. (d) The frame that we detected. (e) Another frame describing the whole scenario.

TABLE II
QUANTITATIVE EVALUATION ON SIX VIDEO SEQUENCES. THE LAST ROW OF THE TABLE INDICATES THE ACCURACY OF OUR PROPOSED APPROACH.

	Seq1	**Seq2**	**Seq3**	**Seq4**	**Seq5**	**Seq6**
Chaotic	21.7%	-	40.5%	52.4%	-	-
SFM	27.5%	-	52.4%	69.0%	-	-
BRIASS.	79.7%	50.0%	69.0%	-	-	-
Bayesian	53.6%	-	64.3%	45.2%	-	-
WSC	85.5%	63.8%	81.0%	71.4%	87.0%	80.0%
Ours	**95.7%**	**82.8%**	**95.2%**	**92.9%**	**90.7%**	**88.0%**

which is better than WSC's 1s delay. Seq6 is a local merging event where two groups of people merged at first, and then walk to the left border as one group (refer to Fig. 11). Our system detects that the first local merging occurs at Frame 64, compared with the existing method that detected it in Frame 68. Ours is also the closet one to the ground truth. The quantitative evaluation of local change detection is also shown in Table II.

In general, for all six video sequences, comparing our proposed method with other state-of-the-art (SOTA) methods and the ground truth, our approach has a superior performance.

VII. CONCLUSIONS

In this paper, we propose a crowd event change detection framework that can achieve good performance at both a global level and local level. The architecture of our framework is based on the YOLOv2 model for object detection and our proposed coordinates representation and the fixed-width clustering is used to produce the clustering result, like cluster coordinates, the numbers of clusters, and changes in cluster direction. Further, these features are used for event classification and event change detection. The events are classified into two main types, walking/running and merging (global and local)/splitting (global and local). Six video sequences of the PETS2009 dataset are used for evaluation. In comparison with other existing methods, our experimental results show that the accuracy is between 80%-95.7%. On all the video sequences our method achieves very competitive performance.

REFERENCES

[1] B. Solmaz, B. E. Moore, and M. Shah, Identifying behaviors in crowd scenes using stability analysis for dynamical systems, in IEEE Transactions on Pattern Analysis and Machine Intelligence, 2012, 34(10), pp. 2064-2070.

[2] J. C. S. J. Junior, S. R. Musse, and C. R. Jung, Crowd analysis using computer vision techniques. In IEEE Signal Processing Magazine, 2010, 27(5), pp. 66-77.

[3] W. Hu, T. Tan, L. Wang, and S. Maybank, A survey on visual surveillance of object motion and behaviors, in IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 2004, 34(3), pp. 334-352.

[4] L. Li, W. Huang, I. Y. H. Gu, R. Luo, and Q. Tian, An efficient sequential approach to tracking multiple objects through crowds for real-time intelligent CCTV systems, in IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 2008, 38(5), pp. 1254-1269.

[5] C. Stauffer, and W. E. L. Grimson, Learning patterns of activity using real-time tracking, in IEEE Transactions on Pattern Analysis and Machine Intelligence, 2000, 22(8), pp. 747-757.

[6] Su, H., Yang, H., Zheng, S., Fan, Y., and Wei, S. (2013). The large-scale crowd behavior perception based on spatio-temporal viscous fluid field. IEEE Transactions on Information Forensics and Security, 8(10), 1575-1589.

[7] B. Zhou, X. Wang, and X. Tang, Understanding collective crowd behaviors: Learning a mixture model of dynamic pedestrian-agents, in IEEE Conference on CVPR, 2012, pp. 2871-2878.

[8] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, You only look once: Unified, real-time object detection, in IEEE Conference on CVPR, 2016, pp. 779-788.

[9] J. Ferryman. (2009). PETS 2009 Benchmark Data. [Online]. Available: http://www.cvg.reading.ac.uk/PETS2009/a.html, accessed, Jun. 28, 2015.

[10] A. S. Rao, J. Gubbi, S. Marusic, and M. Palaniswami, Crowd event detection on optical flow manifolds, in IEEE Transactions on Cybernetics, 2016, 46(7), pp. 1524-1537.

[11] J. Redmon, and A. Farhadi, YOLO9000: better, faster, stronger, arXiv preprint, 2017.

[12] M. Yang, L. Rashidi, S. Rajasegarar, C. Leckie, A. S. Rao, and M. Palaniswami, Crowd Activity Change Point Detection in Videos via Graph Stream Mining. in Proceedings of the IEEE Conference on CVPRW, 201, pp. 215-223.

[13] S. Rajasegarar, C. Leckie, and M. Palaniswami, Hyperspherical cluster based distributed anomaly detection in wireless sensor networks, Journal of Parallel and Distributed Computing, 2014, 74(1), pp. 1833-1847.

[14] M. Yang, S. Rajasegarar, A. S. Rao, C. Leckie, and M. Palaniswami, Anomalous Behavior Detection in Crowded Scenes Using Clustering and Spatio-Temporal Features, in International Conference on Intelligent Information Processing, 2016, pp. 132-141.

[15] I. R. de Almeida, and C. R. Jung, Change detection in human crowds, in SIBGRAPI-Conference on Graphics, Patterns and Images (SIBGRAPI), 2013, pp. 63-69.

[16] de Almeida, I. R., Cassol, V. J., Badler, N. I., Musse, S. R., and Jung, C. R. (2017). Detection of global and local motion changes in human crowds. IEEE Transactions on Circuits and Systems for Video Technology, 27(3), 603-612.

[17] R. Mehran, A. Oyama, and M. Shah, Abnormal crowd behavior detection using social force model, in IEEE Conference on CVPR, 2009, pp. 935-942.

[18] S. Wu, B. E. Moore, and M. Shah, Chaotic invariants of Lagrangian particle trajectories for anomaly detection in crowded scenes, in IEEE Conf. Comput. Vis. Pattern Recognit., Jun. 2010, pp. 2054C2060.

[19] D. Y. Chen, and P. C. Huang, Motion-based unusual event detection in human crowds, Journal of Visual Communication and Image Representation, 2011, 22(2), pp. 178-186.

[20] S. Wu, H. S. Wong, and Z. Yu, A Bayesian model for crowd escape behavior detection, IEEE Trans. Circuits Syst. Video Technol., vol. 24, no. 1, pp. 85C98, Jan. 2014.

[21] A. Briassouli, and I. Kompatsiaris, Spatiotemporally localized new event detection in crowds, in IEEE International Conference on Computer Vision Workshops (ICCV Workshops), 2011, pp. 928-933.