Al-Mustansiriyah Journal of Science
ISSN: 1814-635X (print), ISSN: 2521-3520 (online)
Volume 28, Issue 3, 2017
DOI: http: //doi.org/10.23851/mjs.v28i3.35

Research Article

Anomalous Behavior Detection Based on Geometrical Complex Moments in Crowd Scenes

Abdulamir A. Karim[1], Narjis M. Shati[2]
[1]Department of computer science, University of Technology, IRAQ
[2]Department of computer science, College of science, Mustansiriyah University, IRAQ
*Correspondent Author Email: nareen_manar@yahoo.com

ArticleInfo

Received
8 Mar. 2017

Accepted
17 Oct. 2017

Abstract

In this research work a data stream clustering method done by extracting regions of interest from the frames of video clips (UCSD pedestrian dataset (ped1 and ped2 datasets) video clips, and VIRAT VIDEO dataset video clips). In extraction process, the HARRIS or FAST detector applied on the frames of video clips to extract list of pairs of interest points. From these pairs a list of features such as: distance, direction, x-coordinate, y-coordinate obtained to use as an input to the clustering method based on seed based region-growing technique. From these clusters, a regions of interest extracted according the pairs coordinates of each cluster. Finally, from these regions a set of geometrical complex moments obtained and then used in anomaly detection system. The results indicated that using HARRIS detector achieved detection rates are 7.88%, 51.30%, and 56.67% with false alarms are 19.39%, 32.61%, and 60 % by using Ped1, Ped2, and VIRAT datasets respectively. For the case of using FAST detector, the best detection rates are 6.67%, 44.78%, 53.33% with false alarm rates are 33.33%, 41.74%, 70% by using the datasets respectively.

Keywords: Smart Surveillance System, Anomaly Events, Abnormal Behavior, Anomaly Detection, Abnormal Detection.

الخلاصة

في هذا البحث يتم انجاز طريقه لعنقدة دفق البيانات وذلك باستخلاص مناطق الاهتمام من الـ (frames) الخاصة بمقاطع فديو الـ (datasets) والتي هي (Ped1، Ped2، VIRAT). في عملية الاستقطاع تم تطبيق الـ (HARRIS detector) أو الـ (FAST detector) على الـ (frames) للحصول على أزواج نقاط الاهتمام، ومن هذه الأزواج نستخلص مجموعة من الخصائص (المسافة، الاتجاة، احداثيات المحورين السيني والصادي) وذلك لاستخدامها كمدخل لطريقة العنقدة المقترحة على أساس تقنية الـ (seed filling). ومن هذه الـ (cluster) التي نحصل عليها يتم اقتطاع منطقة الاهتمام على أساس احداثيات كل (cluster). وآخرا يتم حساب مجموعه من الـ (geometrical complex moments) من كل منطقة اهتمام لاستخدامها في نظام تحديد الحالات الشاذة المقترح.

اشارت نتائج استخدام الـ (HARRIS detector) الحصول على نسبة تمييز 7,88%، 51,30%، 56,67% مع نسبة خطأ في التمييز هي 19,39%، 32,61%، 60% باستخدام مجموعة فديوات Ped1، Ped2، VIRAT بالتتابع. اما في حالة الـ (FAST detector) فأن أفضل نسبة تمييز كانت 6,67%، 44,78%، 53,33% مع نسبة خطأ بالتمييز هي 33,33%، 41,74%، 70% وذلك باستخدام مجموعة الفديوات سابقة الذكر.

Introduction

Intelligent Surveillance System (ISS) helps people to transform video surveillance into a real-time, proactive, event-driven process by reducing the vast amount of information contained in video and extracting the interesting information from it. ISS includes many intelligent analysis modules, e.g., video motion detection, camera tampering detection, people counting, face recognition, vehicle license plate recognition, and so on. Also intelligent video systems not only provide real-time abnormal event detection but also acquire continuous video sequences from adjacent cameras using panning, tilting, and zooming (Pan-Tilt-Zoom (PTZ)) with multichannel, multi-area, and an immediate acquisition of the monitoring area video with complex and vast environment, which needs lots of complex

computation and is almost impossible for people to finish in real time [1].
Some related researches that investigating the detection of anomaly behavior in crowed scene are presented in [2] [3] [4] [5] [6] [7].

Suspicious Activities Behavior Detection

Approaches in detecting suspicious activities behaviors are based on detecting an intrusion. When there are patterns not matched for intrusion with the previously created attacks that were created for the known attacks an anomaly detection model is detected newly created attacks. The anomaly detection model creates a long-term usage profile. This profile represents the common users' activities. The short-term profile (the current user patterns) is compared with the long-term profile. An attack is detected when the short-term profile deviates too far from the long-term profile. This latter approach is more popular as it is able to handle unseen patterns [8].
Measuring the deviation of a behavior pattern from the others could be done in different ways. One may describe the deviation in terms of whether a behavior can be constructed from the normal behavior patterns database or not [9] [10]. The other determines the deviation by simply considering a behavior that cannot be classified into one of the known normal behavior categories that deviate from normal [11]. These approaches have to have a complete set of normal behavior. This becomes problematic in real-life scenarios since it is difficult to define all possible normal behaviors. The number of suspicious behavior types is less than the normal ones but also defining all possible suspicious behavior is a difficult problem [10].
The solution to address this problem is to design a system which is able to update its normal behavior patterns database adaptively by grouping similar behavior patterns and defining their representation which will be used for classifying a new incoming behavior pattern into the existing groups [12].
There are three categories of approaches used for understanding suspicious behavior patterns (which are subset of human behavior patterns) such as scene interpretation, holistic approach, and a rule based approach.

Features of Human Behavior

Human behavior features are divided into two major parts: Human global motion and Human local motion features.
Human Global Motion features utilize the information on a subject location at various times. It is useful when human limb movements are not observable. The systems using this features considers only the tracking information of each subject's locations at a time. There is a lot of information extracted from human trajectories, such as (Commonly occupied region, Human walking path shape) [13].
Human Local Motion features are useful when human limb movements can be discerned in video feeds. Based on the degree of not-rigidity of the objects, human motion is classified as rigid or not rigid motions. Also there is articulated motion in which limb motions are rigid but overall motion is not rigid [14].
There are two categories of human motion approaches (Model based approaches, Appearance based approaches). The first using prior knowledge about the shape of objects [15], but the second building body representation in a bottom-up fashion by first detecting appropriate features in a sequence of frames. Silhouettes and interest points are some examples of these features [16] [17]. Appearance based approaches are classified into three types: Flow based approaches, Spatial-temporal shape template base approach, and Interest point based approaches
The interested points extracted by interest point detectors. It is extension of key point's concept for object detection in images. One of the advantage of using interest points is that they can be used to search an action contained in the short query video over a large resolution without using background subtraction and tracking of the object [17] [18].
In this paper interest point features are used by utilizing HARRIS or FAST detector, for more information about these detectors see [19].

Pixel Statistics Features

In general, moments are a set of parameters which describe the distribution of material from a reference point or an axis. In image processing material is replaced by brightness. The idea of using moments to construct the image feature vectors is one of the commonly utilized methods today. Each moment order reflects different information about the same image [20].

Some sets of moments based on complex numbers were suggested by [21]. It was shown that these moments have more robustness against 2D-geometical transformation (like, rotation, scaling). The simplest form of complex moments is given as [22]:

$$M(n)=\sum_{y=0}^{H-1}\sum_{x=0}^{W-1}\left\{\left(\frac{x-x_c}{L}\right)+j\left(\frac{y-y_c}{L}\right)\right\}^n f(x,y) \qquad (1)$$

Where

M(n): the nth order complex moment.

F(x, y): the pixel value of position (x, y).

W: the image width.

H: the image height.

$j=\sqrt{-1}$: is the complex number.

and:

$$x_c=\frac{W-1}{2} \qquad (2)$$

$$y_c=\frac{H-1}{2} \qquad (3)$$

$$L=\max(x_c,y_c) \qquad (4)$$

The modulus value of moment, M(n), is given as:

$$|M(n)|=\sqrt{M(n)M^*(n)} \qquad (5)$$

The detection methods based on moment's features are utilized in this work.

Means Clustering and Seed filling Technique

K-Means clustering is a partitional clustering approach. Each cluster is represented by one prototype object, and a new data sample is assigned to the nearest prototype and therefore to that cluster, for more information sees [23].

Seed Based Region Growing method is Segmentation technique. The basic idea in this method is to group a collection of pixels to form a region based on some similar properties; these properties may include intensity, texture or color, as shown in Figure [24] [25].

Data Stream Clustering

A data stream model is defined as an ordered sequence of points $x_1, \ldots, x_n$ where ($n \approx \infty$). The sequence has to be read in order and once or a small number of times [26].

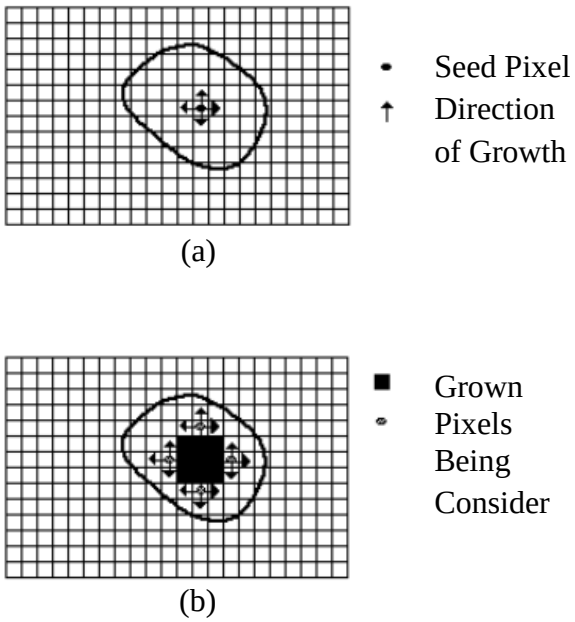

Figure 1: Example of region growing. (a) Start of Growing a Region. (b) Growing Process after a Few Iterations.

The data stream model requires decisions to be made before all the data becomes available. This model is similar to online models. So these models need Algorithm (data stream Algorithm). These Algorithms are allowed to take action after a group of points arrives [26].

Data stream and online clustering approaches are similar in that both of them require decisions made before all data are available. But these models are not identical because online Algorithm can access the first i data

point (with its previous i decisions) when reacting to the $(i+1)^{th}$ point, the amount of memory available to data stream Algorithm is bounded by a function of the input size (sublinear function used). In addition, a data stream approach is not required to take action after the arrival of each point (after a group of points) [8].

Data stream clustering approaches store and processes large-scale data efficiently because it provides summarizations of the past data, see [27] [28] [29] [30] [31] [32] for more information. In this work seed filling and k-mean, clustering Algorithms is utilized to form new data stream Algorithm.

Design of Data Stream Clustering Algorithm utilizing K-means and Seed Filling Techniques

Anomaly behavior detection systems are created and the same datasets used for training and testing. Their performance were measured using (Detection Rate, False Alarm Rate, Recall, Precision, The Coverage Test), for more detail about these measurements see [22] [28].

The Proposed Systems Layouts

The diagram for the proposed anomaly detection system is shown in Figure .

Video Dataset

The selected video datasets are two publicly available datasets. The first is the UCSD dataset; which contains two pedestrians' dataset (ped1 and ped2); and the second is VARAT dataset. The selected video dataset is divided into two sets of videos: (1) a training video set is used for the training phase of the system, and (2) a test video set contains videos used to measure the anomaly behavior detection performance of the system.

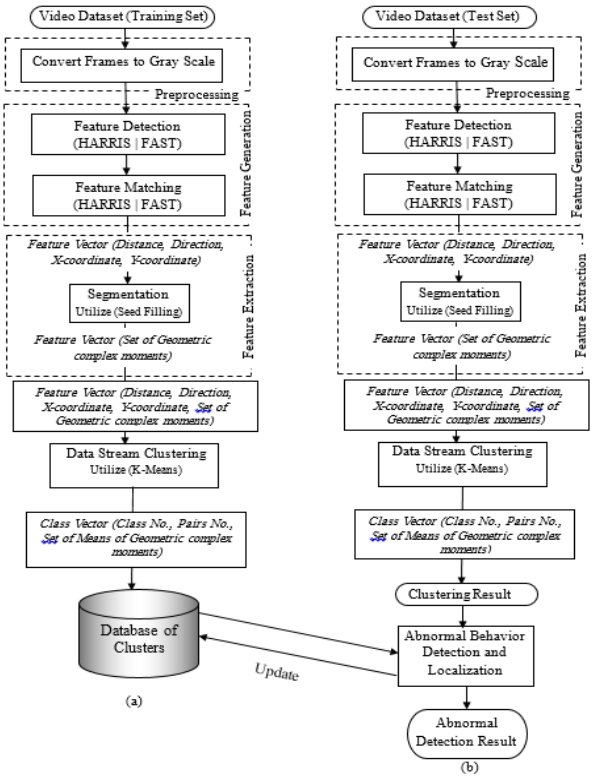

Figure 2: Diagram of the proposed systems: (a) Enrolment phase (b) Anomaly detection and localization phase

Preprocessing

To make the data of both training and test sets more suitable and easier to analyses the preprocessing stage is necessary. It is implemented on both training and test datasets by converting each pixel in extracted color sub-image has three components of color (red, green, blue) to gray by using the average method. This will lead to convert the color frame to gray frame.

Feature Generation

A context information (such as: interest point pairs) and from these pairs feathers like distance, direction, x-coordinate, and y-coordinate are extracted from the training and test dataset. This done by utilizing interest point detectors HARRIS or FAST to extract interest point from the current frame and previous frame (according to predefined threshold PFT) and then matching these interest point from both frames to obtains list of pairs of interest points. The process illustrated in Algorithms (1, and 2). After that, an extraction process will be done to estimate list of features (distance, direction, x-

coordinate, y-coordinate) from these pairs, see Figure . This illustrated in Algorithm (3).

Algorithm (1): Detection and matching interest points using HARRIS

Input: Frames_List, PFT (Previous Frame Threshold), No. of Frames
Output: LOP (List of interest point pairs)

```
For n = 0 to Frame_List.count
    Detect interest point using HARRIS corner Detector (Frames_List[n], Interest_Points_List [n])
    If n > PFT then
        Matching (Frames_List[n], Frames_List [n -PFT], Interest_Points_List[n], Interest_Points_List [n-PFT]), LOP)
        For i = 0 to LOP.count
            Compute (LOP[i].distance) using Euclidean distance between the two points of the pair (LOP[i])
            If LOP[i].distance < Low_Threshold or LOP[i].distance > Max_Threshold
            Then Remove (LOP[i])
            End If
        End For
    End If
End For
End.
```

Algorithm (2): Detection and matching interest points using FAST

Input: Frames_List, PFT (Previous Frame Threshold), No. of Frames, Low_Threshold (Low threshold of removing error of interest points matching, its value equal 1), Max_Threshold (Max threshold of removing error of interest points matching, its value equal 500).
Output: LOP (List of interest point pairs)

```
For n = 0 to Frame_List.count
    Detect interest point using FAST Detector (Frames_List[n], Interest_Points_List[n])
    If n > PFT then
        Matching (Frames_List[n], Frames_List[n-PFT], Interest_Points_List[n], Interest_Points_List[n-PFT]), LOP)
        For i = 0 to LOP.count
            Compute (LOP[i].distance) using Euclidean distance between the two points of the pair (LOP[i])
            If LOP[i].distance < Low_Threshold or LOP[i].distance > Max_Threshold
            Then Remove (LOP[i])
            End If
        End For
    End If
End For
End.
```

Euclidean distance used to compute the distance between the two points (P1(x1,y1) and P2(x2,y2)) of each pair in list of pairs. Then calculate direction by finding the angle of inclination, as shown in the following equation and illustrated in Algorithm (4).

$$Dir = \begin{cases} 0 & if\ (x_1 - x_2 > 0)\ and\ (y_1 - y_2 = 0) \\ 90 & if\ (x_1 - x_2 = 0)\ and\ (y_1 - y_2 > 0) \\ 180 & if\ (x_1 - x_2 < 0)\ and\ (y_1 - y_2 = 0) \\ 270 & if\ (x_1 - x_2 = 0)\ and\ (y_1 - y_2 < 0) \\ \tan^{-1}(y_1 - y_2)/(x_1 - x_2) & Otherwize \end{cases} \quad (6)$$

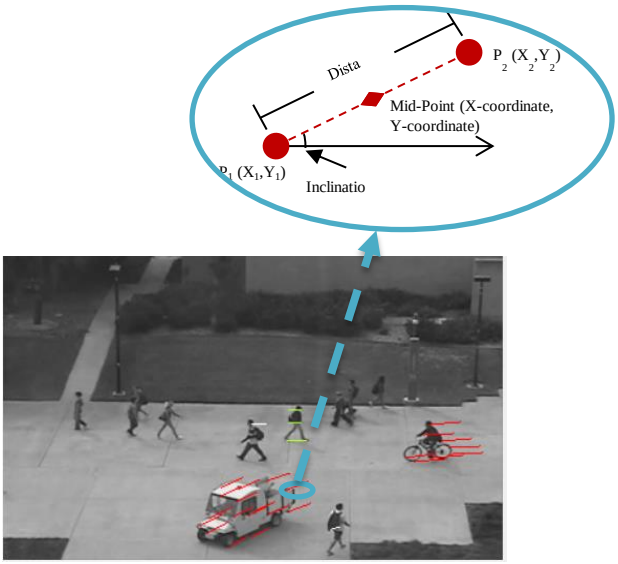

Figure 3: Illustration of distance, inclination angle, x-coordinate, and y-coordinate for two points (P1 and P2).

Algorithm (3): Compute distance, direction coordinate, y-coordinate from interest point pairs

Input: List of interest point pairs (LOP)
Output: LOF (List of feature vector (distance, direction, x-coordinate, y-coordinate))

```
For i=0 to LOP.count
    Compute (LOF[i].dis) using Euclidean distance between P1(x1, y1) and P2 (x2, y2) in one pair
```

Compute (Direction) using Algorithm (4),
Compute (x-coordinate, y-coordinate),
Set LOF[i].x ← (LOP[i,1].x + LOP[i,2].x)/2
Set LOF[i].y ← (LOP[i,1].y + LOP[i,2].y)/2
End For
End.

Algorithm (4): Compute direction

Input: two interest point P_1 (x_1, y_1) & P_2 (x_2, y_2)
Output: D (Direction)

If $(x_1 - x_2) = 0$ **then**
  If $(y_1 - y_2) > 0$ **then**
    Set D ←90
  Else If $(y_1 - y_2) < 0$ **then**
    Set D ←270
Else If $(y_1 - y_2) = 0$ **then**
  If $(x_1 - x_2) > 0$ **then**
    Set D ← 0
  Else
    Set D ←180
Else
  Set D ← $\tan^{-1}(y_1 - y_2)/(x_1 - x_2)$
End If
End.

After that, compute the x-coordinate and y-coordinate for the pair by finding the mid-point between the first and the second point of each pair, as show in the following equations.

$$x = (x_1 + x_2) / 2 \quad (7)$$
$$y = (y_1 + y_2) / 2 \quad (8)$$

Proposed Data Stream Clustering

In this paper, a data stream model is defined as four features of each pairs of interest point which are obtained from the previous section with a set of geometric complex moments. The proposed Algorithm is used to make decision after a group of these features arrived and then these decisions (past data of clusters) are summarized and organized in a database; that gives a summarization of all clusters in limited memory size. In this system before the data stream clustering is done, the extraction of the interest regions needs to be generated in order to obtain a set of geometrical complex moment and use it as a part of the feature vector. This is done by utilizes seed based region growing technique to cutting out a number of sub-images from the frames of dataset video clips. The first step in the phase of cutting out regions of interest is the generation of the interest regions coordinates and then extracted these regions. The second step is obtaining set of geometrical complex moment from these extracted regions of interest. After that list, the interest point features (the set of geometrical complex moment) as a feature vector to the proposed data stream clustering Algorithm (5) illustrates the implemented steps for the third propose data stream clustering.

Algorithm (5): Third Proposed Data Stream Clustering System

Input: Video data set (Train or Test)
Output: Database_Clusters (database of clusters)

Step$_1$: Read Video
Step$_2$: Convert color frame to gray frame
Step$_3$: Detection and matching interest points using (HARRIS | FAST) by using Algorithms (1, 2) respectively
Step$_4$: Compute distance, direction, x-coordinate, y-coordinate from interest point pairs using Algorithm (3)
Step$_5$: Clustering list of features by Algorithm (6) using Seed filling
Step$_6$: Segment each cluster according to its position (Min and Max coordinate of pairs for each cluster)
Step$_7$: Compute geometric complex moments for each segment, using Algorithm (9).
Step$_8$: Clustering list of features (set of geometrical complex moment) by Algorithm (10) based on K-means.
Step$_9$: If training phase **then**
  Create Database_Clusters using Algorithm (13)
Else
  Detect and localize anomaly behavior using Algorithm (14)
End If
End.

Starting with reading video clips from datasets (the training phase or testing phase) to obtain list of video frames and number of frames, and then converting these frames to gray frames. After applying interest point detector (HARRIS or FAST) using Algorithm (1 or 2) to obtain list of interest pairs. From these pairs, a list of features (such as: distance, direction, x-coordinate, y-coordinate.) extracted, as shown in Algorithm (3). Using these features a clusters obtained by a data stream clustering Algorithm based on seed filling technique, as illustrated in Algorithm (6).

Algorithm (6): Clustering list of features using Seed Filling technique based on coordinate

```
Input: LOF
Output: LOF, Centroids_list
    For i = 0 to LOF.count
        If LOF[i]  is not classified then
            Set LOF[i] as a new cluster
            Add LOF[i] to Cluster_List
            do
                For j = 0 to Cluster_List.count
                    Compute closest neighbor for each
                    Cluster_List [j] in (LOF) using
                    Euclidean distance by implemented
                    by Algorithm (7).
                    If neighbor of Cluster_List [j] is
                    not classified then
                      Set neighbor (Cluster_List [j].
                      cluster) ← List_class [j]. cluster
                      Add neighbor (Cluster_List [j]) to
                      Cluster_List
                    End If
                End for
            End while (listing all Cluster_List piars)
            If (Cluster_List.length > 1) then
                Calculate new centroids
                (dis+dir+cords+start+end) using
                Algorithm (8)
                Add this centroid to centerids_list
            Else
                Set LOF[i] as not classified
                Empty Cluster_List
            End if
        End if
    End for
End.
```

In Algorithm (6) test all pairs in list of pairs will be done, if the pair is not classified under any cluster, then it will be classified as new cluster and added to the list of classified neighbor pairs to be tested through computing the closest neighbor to it in list of pairs by using Euclidian distance, as shown in Algorithm (7), where the coordinate threshold is used to determine the neighbor when distance is less than it. And then, test all neighbor pairs if it is not classified to any cluster then it will be classified as the cluster of the tested pair and added to the classified neighbor pairs list to examine the neighbor pairs for it. After completing the test of the neighbor pairs list and there is more than one pair in the set a computation of new centroids for these neighbor pairs will be done and these new centroids are added to the list of centroids otherwise will be discarded and removed from the neighbor pairs list. The computation of new centroids will be done by allocating minx, miny, maxx, maxy for each group of neighbor pairs in the same cluster to determine the coordinates of the new cluster, also the average of distance and direction will be computed, as shown in Algorithm (8).

Algorithm (7): Compute closest neighbor

```
Input: LOF
Output: Neighbor_List
    For i = 0 to LOF.count
        For j=0 to LOF.count
            If (LOF[i].dis - LOF[j].dis) <
               dis_thr and (LOF[i].dir -
               LOF[j].dir) < dir_thr and
               (Euclidean distance between
               (LOF[i], LOF[j])) < dis_thr
            Then Set neighbor_List [i,j]
               ← true
            End if
        End for
    End for
End.
```

Algorithm (8): Calculate new centroids

```
Input: Cluster_List
Output: One centroid
    Set min_x ← Cluster_List[1].start.x
    Set min_y ← Cluster_List[1].start.y
    Set max_x ← Cluster_List[1].end.x
    Set max_y ← Cluster_List[1].end.y
    For each pair in Cluster_List
        If (min_x > pair.start.x) than  Set min_x
        ← pair.start.x
        If (min_y > pair.start.y) than  Set min_y
        ← pair.start.y
        If (max_x < pair.end.x) than  Set min_x
        ← pair.end.x
        If (max_y < pair.end.y) than  Set min_y
        ← pair.end.y
        Set Dis ← dis + pair.dis
        Set Dir ← dir + pair.dir
    End for
    Set Centroids.dis ← dis/centroids.count
    Set Centroids.dir ← dir/centroids.count
    Set Centroids.coords.x ← (min_x+max_x)/2
    Set Centroids.coords.y ← (min_y+max_y)/2
    Set Centroids.start.x ← min_x
    Set Centroids.start.y ← min_y
    Set Centroids.end.x ← max_x
    Set Centroids.end.y ← max_y
End.
```

The next step is using these clusters to segment regions of interest in order to obtain a set of geometrical complex moments as shown in Algorithm (9), the process illustrated in Figure 4. After that the feature vector constricted having the set of geometrical complex moments, in order to apply Data stream clustering based on K-means, as seen in Algorithm (10).

Algorithm (9): Geometrical complex moments

Input: Sub_Images_List, SubH,SubW, MomNo
Output: Complex_Moments_list
Set Sum ← 0
For i = 1 **to** MomNo
Set MomX(i) ← 0
Set MomY(i) ← 0
End for
Set Hm ← SubH−1
Set Wm ← SubW−1
Set Hh ← Hm / 2
Set Wh ← Wm / 2
For Y = 0 **to** Hm
Set Yn ← (Y−Hh) / Hh
Set Yp(1) ← Yn
For X = 0 **to** Wm
Set Xn ← (X−Wh) / Wh
Set Xp(1) ← Xn
Set Im ← Sub_Image_List (Y,X)
Set Sum ← Sum + Im
Set MomX(1) ← MomX(1) + Im × Xn
Set MomY(1) ← MomY(1) + Im × Yn
For i = 2 **to** MomNo
Set j ← i −1
Set Xp(i) ← (Xp(j) × Xn) – (Yp(j) × Yn)
Set Yp(i) ← (Xp(j) × Yn) + (Yp(j) × Xn)
Set MomX(i) ← MomX(i) + Im × Xp(i)
Set MomY(i) ← MomY(i) + Im × Yp(i)
End for
End for
End for
For i = 1 **to** MomNo
Set Complex_Moments_List(i) ← (MomX(i) × MomX(i) + MomY(i) × MomY(i))^2 / Sum
End for
End.

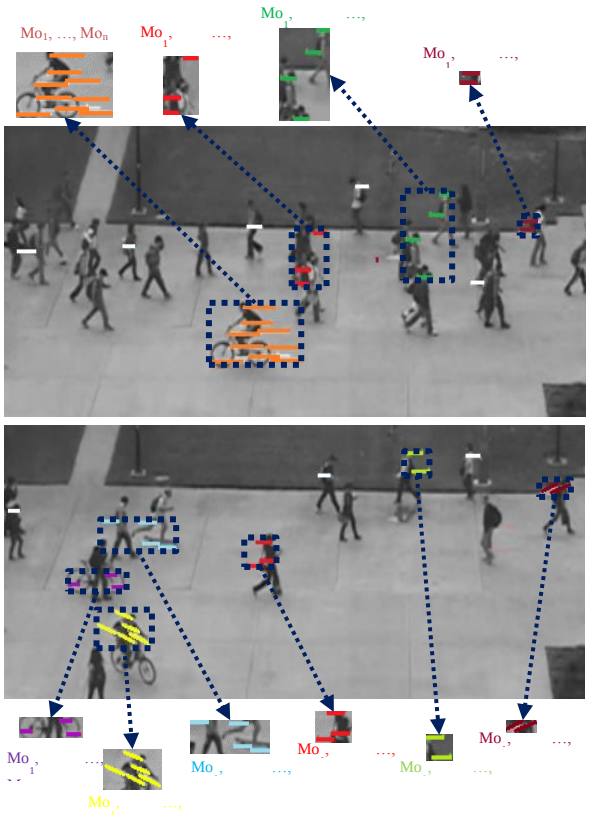

Figure 4: Illustrates computing geometrical complex moments from extracted sub-images.

Start by choosing initial k centroids from set of geometrical complex moments of region of interest, and then calculate the closest centroids for all moments of each region by using Algorithm (11), where computing by the average of difference between the region set of moments and all centroids set of moments. Then the centroids that own the closest differences will be owned these moments.

Algorithm (10): Clustering list of moments using K-means

Input: LOMom
Output: LOMom, Centroids_list
Step$_1$: Choose initial K centroids randomly from LOMom named Centroids_list
Step$_2$: For each moments determine which centroid (Centroids_list) closest to it using Algorithm (11).
Step$_3$: Compute new centroids using Algorithm (12)
Step$_4$: Compare new centroids with previous centroids; if it changed go to step$_2$.
End.

Algorithm (11): Determine closest centroid

```
Input: LOMom, Centroids_list
Output: LOMom
    For i = 0 to LOMom.count
        For j = 0 to Centroids_list.count
            Compute (Dif_c[j] using Euclidean
                    difference between
                    (LOMom [i],
                    Centroids_list
                    [j].moment)
        End For
        Put in (LOMom [i].cluster) the index of
        minimum value in (Dif_c)
    End For
End.
```

A new centroids will be computed which represents the average for all moments in this cluster, as shown in Algorithm (12). Finally compare the new centroids with the previous ones, if there is any change, go again to compute closest centroids for each pair and so on until no change happened.

Algorithm (12): Compute new centroid

```
Input: LOMom
Output: Centroids_list (list of K centroid)
  For i = 0 to LOF.count
    Set Cluster ← Centroids_list [i].cluster
    For j=0 to LOMom.Count
      If LOMom[i].cluster = Cluster then
        For L=1 to MomNo
          Set Sum[L] ←
          Sum[L] +
          LOMom[i].moment[L]
        End For
        Set Co ← Co + 1
      End If
    End For
    For L=1 to MomNo
        Set Centroids_list[j]. moment[L] ←
        Sum[L]/Co
    End For
  End For
End.
```

In enrolment phase, the feature vector (from training video clips of datasets) used to create normal cluster database, see Algorithm (13). While in the anomaly detection and localization phase, the anomaly event and its location be detected with the help of previously database created.

Algorithm (13): Create database of normal clusters

```
Input: Centroids_list
Output: Database_Clusters (database of clusters)
  For i = 0 to Centroids_list.count
    For each record in Database_Clusters
      If record is equal to Centroids_list [i]
      Then
        Set Record. Frequnce ← Record. Frequnce+1
      Else
        Database_Clusters.add (Centroids_list [i])
      End If
    End For
  End For
End.
```

Anomaly Behavior Detection and Localization

After completing the construction of cluster database from passing the training video dataset, it will be used in anomaly behavior detection in order to detect anomaly behavior in test video dataset and localize its position; Algorithm (14) describes the anomaly behavior detection and localization. In this Algorithm, the class obtained from the proposed data stream clustering Algorithm in test phase and then search the database of clusters; that is obtained from the training phase; to find if this class is in the database, so it will be detected as anomaly event and its location in the frame is marked by using the coordinate from the clustering data stream process.

Algorithm (14): Anomaly behavior detection and localization

```
Input: Test video dataset, Database_Clusters (database
of clusters)
Output: Anomaly_behavior_location
  For i=0 to centroids_list.count
    If record in Database_Clusters where record equal
    to centroids_list[i] then
      Set Record.Frequnce ← Record. Frequnce +1
    Else
      Mark the position of anomaly behavior
    End If
  End For
End.
```

Results and Discussions

The proposed approach is tested on two datasets. The UCSD pedestrian dataset it contains video sequences from two pedestrian

(Ped1 and Ped2) walkways where abnormal events occur.

The dataset contains different crowd densities, and the anomalous patterns are the presence of non-pedestrians on a walkway (bicyclists, skaters, small carts, and people in wheelchairs). The second dataset is VIRAT Video Dataset was designed by Kitware Company to be more realistic, natural and challenging for video surveillance domains than existing action recognition datasets in terms of its resolution, background clutter, diversity in scenes, and human activity/event categories.

Data are collected in natural scenes showing people performing normal actions in standard contexts, with uncontrolled, cluttered backgrounds. The key characteristics of these datasets are summarized in Table 1.

Table 2 presents the values of the used data stream clustering parameters. The experiment was repeated 15 times, and the anomaly detection test results of UCSD (Ped1), UCSD (ped2), and VIRAT video clips sets using features extracted from the interest point pairs as an input to the anomaly behavior detection system in both method of feature extraction using HARRIS or FAST detectors are presented in Tables 3, 4, and 5.

Table 1: Key characteristics of training and testing video clips in datasets.

			UCSD (Ped1)	UCSD (Ped2)	VIRAT
Entire dataset	training	Number of video clips	34	16	6
		Frame Dimension	238×158	360×240	426×240
		Frame rate	15 frames/ second	15 frames/ second	15 frames/ second
	test	Number of video clips	36	12	8
		Frame Dimension	238×158	360×240	426×240
		Frame rate	15 frames/ second	15 frames/ second	15 frames/ second
Anomaly Behavior Event	test	Number of anomaly events	1460	987	335

Table 2: The proposed data stream clustering parameters.

Parameter	Value
Previous frame threshold (PFT)	3
Distance threshold	2
Direction threshold	1
Coordinate threshold	25

Table 3: Results of the anomaly behavior detection system by using UCSD (Ped1) dataset.

	Expt.	Training Time (ms/ frame)	Testing Time (ms/ frame)	DR (%)	FAR (%)	R	P	CT
HARRIS	Min	2.18	2.92	7.27	21.21	0.07	0.26	0.11
	Max	2.18	2.82	8.48	20.00	0.08	0.30	0.13
	Average	2.18	2.77	7.88	19.39	0.08	0.29	0.12
FAST	Min	3.04	3.84	6.06	30.30	0.06	0.17	0.09
	Max	3.04	3.21	7.88	34.55	0.08	0.19	0.11
	Average	3.04	3.40	6.67	33.34	0.07	0.17	0.10

Table 4: Results of the anomaly behavior detection system by using UCSD (Ped2) dataset.

	Expt.	Training Time (ms/ frame)	Testing Time (ms/ frame)	DR (%)	FAR (%)	R	P	CT
HARRIS	Min	2.85	3.01	47.83	34.78	0.48	0.58	0.52
	Max	2.85	2.97	52.17	33.48	0.52	0.61	0.56
	Average	2.85	3.10	51.30	32.61	0.51	0.61	0.56
FAST	Min	3.24	3.30	43.91	42.61	0.44	0.51	0.47
	Max	3.24	3.47	45.65	39.13	0.46	0.54	0.49
	Average	3.24	3.35	44.78	41.74	0.45	0.52	0.48

Table 5: Results of the anomaly behavior detection system by using VIRAT dataset.

	Expt.	Training Time (ms/ frame)	Testing Time (ms/ frame)	DR (%)	FAR (%)	R	P	CT
HARRIS	Min	2.78	3.02	55.00	61.67	0.55	0.47	0.51
	Max	2.78	3.30	63.33	65.00	0.63	0.49	0.55
	Average	2.78	3.28	56.67	60.00	0.57	0.49	0.52
FAST	Min	3.27	3.57	51.67	66.67	0.52	0.44	0.47
	Max	3.27	3.05	58.33	76.67	0.58	0.43	0.50
	Average	3.27	3.45	53.33	70.00	0.53	0.43	0.48

The test result indicates that the proposed anomaly detection system with HARRIS detector have average detection rate (7.88%) for the test video clips from ped1 dataset, and average false alarm rate (19.39%). Also have average DR (51.30%) for the test video clips from ped2 dataset, and average FAR (32.61%). Finally in VIRAT dataset the DR= 56.67% with FAR= (60%). But the test result of proposed system with FAST detector gives (6.67%, 44.78%, and 53.33%) as a detection rates and (33.34%, 41.74%, 70.00%) as a false alarm rates. The obtained detection

performance measures R, P, CT are shown in Figure 5 and Figure 6.

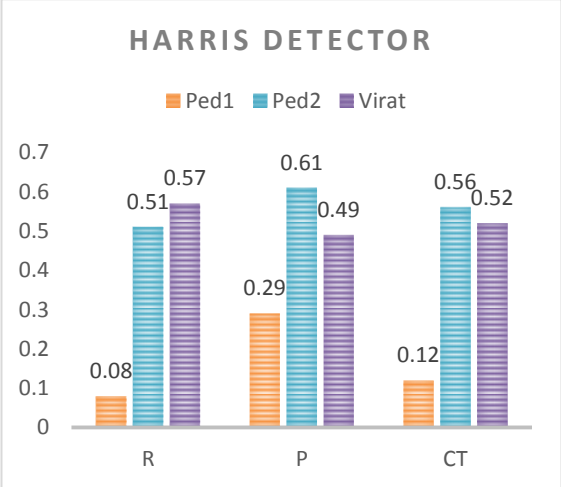

Figure 5: The bar chart result of proposed anomaly detection system using HARRIS detector.

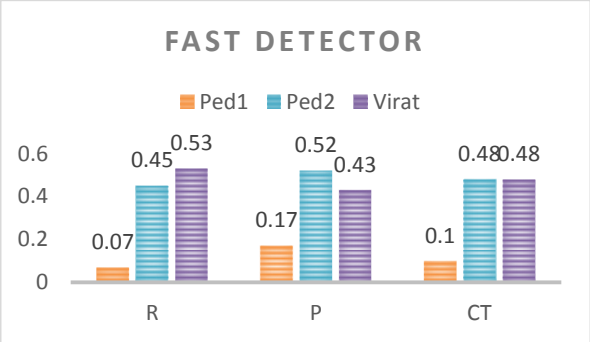

Figure 6: The bar chart result of proposed anomaly detection system using FAST detector.

Conclusions

This paper proposed a novel method utilizing K-mean and seed filling techniques in data stream clustering Algorithm for automatic detection of anomaly events system. This approach has lower computational complexity and fewer demands of dynamic parameter adjustment. The performance of this approach is better when using HARRS detector than FAST detector to obtain interest point pairs in order to extracted significant features used as an input feature to the system. Also the result of this method suffers from the stability problem of the result from dataset to other. In ped1 dataset the view of camera affects the result that indicate poor detection rate. In ped2 dataset the detection rate improved but with amount of false alarm rates. Finally in VIRAT dataset give good detection rate but also with high false alarms. Further work is needed to increase the detection rate and decrease the false alarm rate.

References

[1] Gao, W.; and Ma, S., Advance video coding systems, Springer, 2014.

[2] Li, W.; Mahadevan, V.; and Vasconcelos, N., "Anomaly detection and localization in crowded scenes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 1, 2014.

[3] M. J. Roshtkhari, "Visual Event Description in Videos," 2014.

[4] Zin, T. T.; and Et al., "An integrated framework for detecting suspicious behaviors in video surveillance," *Video Surveillance and Transportation Imaging Applications*, 2014.

[5] Guoa, H.; and Et al., "Quaternion discrete cosine transformation signature analysis in crowd scenes for abnormal event Detection," *Neuro computing*, 2015.

[6] Lee, D.G.; and Et al., "Motion influence map for unusual human activity detection and localization in crowded scenes," *IEEE*, pp. 1051-8215, 2015.

[7] Biswas, S.; and Gupta, V., " Abnormality detection in crowd videos by tracking sparse components," *Machine Vision and Applications*, 2016.

[8] A. Wiliem, Robust suspicious behaviour detection for smart surveillance systems, Australia: Queensland University of Technology; Faculty of Built Environment and Engineering; School of Engineering System, 2010.

[9] Boiman, O.; and Irani, M.., "Detecting Irregularities in Images and in Video," *International Journal of Computer Vision; Springer*, vol. 74, no. 1, p. 17–31, 2007.

[10] Hua, Z.; Jianbo, S.; and Visontai, M., "Detection unusual activity in video," in *In*

Computer Vision and Pattern Recognition. CVPR. IEEE Conference on; Vol. 2; Pp. 819-826, 2004.

[11] Yue, Z.; and Et al., "Detection anomaly in videos from trajectory similarity analysis," in *In Multimedia and Expo, IEEE International Conference on; Pp. 1087-1090*, 2007.

[12] Kratz, L.; and Nishino, K., "Anomaly detection in extremely crowded scenes using spatio-temporal motion pattern models," in *Computer Vision and Pattern Recognition, IEEE Computer Society Conference*, 2009.

[13] Morris, B. T., and Trivedi, M.M., "A survey of vision based trajectory learning and analysis for surveillance," 2008.

[14] Aggarwal, J. K.; and Sangho, P., "Human motion: modeling and recognition of actions and interactions," *3D Data Processing, Visualization and Transmission. 3DPVT. Proceedings. 2nd International Symposium*, pp. 640-647, 2004.

[15] Niyogi, S. A.; and Adelson, E. H., "Analyzing and recognizing walking figures in xyt," technical report, M.I.T. Media Lab Vision and Modeling Group Technical Report; No. 223, 1994.

[16] Bobick, A.; and Davis J., "The recognition of human movement using temporal templates," *Pattern Analysis and Machine Intelligence, IEEE Transactions*, vol. 23, no. 3, pp. 257-267, 2001.

[17] Dollar, P.; and Et al., "Behavior recognition via sparse spatio-temporal features," *Visual Surveillance and Performance Evaluation of Tracking and Surveillance, IEEE International Workshop*, pp. 65-72, 2005.

[18] Shechtman, E.; and Irani, M., "Space-time behavior-based correlation or how to tell if two underlying motion fields are similar without computing them," *Pattern Analysis and Machine Intelligence, IEEE Transactions*, vol. 29, no. 11, pp. 2045-2056, 2007.

[19] R. Szeliski, Computer Vision: Algorithms and Applications, Springer, 2010.

[20] A. M. Mohammed, Hand Identification Using Fuzzy-Neural, Al-Nahrain University; College of Science, 2005.

[21] J. Flusser, On the Independence of Rotation Moment Invariants, Institute of Information Theory and Automation, Academy of Sciences of the Czech Republic, 1999.

[22] M. N. Shati, 2D-Obiect Detection Using Back Propagation Neural Networks, Al-Mustansiriah University; College of Science, 2006.

[23] G. Dougherty, Pattern recognition and classification: An introduction, Springer, 2013.

[24] Matta, S., "Review: Various image segmentation techniques," *International Journal of Computer Science and Information Technologies (IJCSIT)*, vol. 5, no. 6, pp. 7536-7539, 2014.

[25] Langote, V. B.; and Chaudhari, D. S., "Segmentation techniques for image analysis," *International Journal of Advanced Engineering Research and Studies (IJAERS)*, vol. 1, no. 11, pp. 252-255, 2012.

[26] Guha, S.; and Et al., "Clustering data streams: Theory and practice," *Knowledge and Data Engineering, IEEE Transactions*, vol. 15, no. 3, pp. 515-528, 2003.

[27] O'Callaghan, L.; and Et al., "Streaming-data Algorithms for high-quality clustering," in *Data Engineering, Proceedings. 18th International Conference; pp. 685-694*, 2002.

[28] Aggarwal, C. C.; and Et al., "A framework for clustering evolving data streams," in *the 29th VLDB Conference*, Germany, 2003.

[29] Aggarwal, C. C.; and Yu, P. S., "A framework for clustering uncertain data streams," *Data Engineering. ICDE. IEEE 24th International Conference*, pp. 150-159, 2008.

[30] Cheng, L.; and Et al., "An online discriminative approach to background subtraction," in *IEEE International*

Conference on Advanced Video and Signal based Surveillance, 2006.

[31] Lühr, S.; and Lazarescu, M., "Incremental clustering of dynamic data streams using connectivity based representative points," *Data and Knowledge Engineering*, vol. 68, no. 1, pp. 1-27, 2009.

[32] Wiliem, A.; and Et al., "Adaptive unsupervised learning of human actions," in *The 3rd International Conference on Imaging for Crime Detection and Prevention; Kingston University*, London, 2009.