



# Anomaly detection with multi-scale pyramid grid templates

Shifeng Li<sup>1</sup> · Yan Cheng<sup>1</sup> · Liuyang Zhao<sup>1</sup> · Yue Wang<sup>1</sup>

Received: 30 August 2021 / Revised: 3 May 2022 / Accepted: 21 April 2023 /

Published online: 30 June 2023

© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2023

## Abstract

In this paper, we propose a method for abnormal event detection in videos based on Multi-scale Pyramid Grid Templates (MPGT). Unlike traditional methods that usually finish anomaly detection based on a single scale feature, we propose to detect anomalies with a designed multi-scale normalized motion feature in the framework of MPGT. In our work, two scene models are proposed, including a global model and an online model, because the anomalies often occur in regions with moving objects. In addition, we propose a fast method for computing high-scale motion features using a convolution operation based on the first scale feature, and design a scheme for combining the detection results at different scales using vote and pyramid strategies. Experiments on public datasets show that our method has a balanced performance on all the testing datasets.

**Keywords** Anomaly detection · Multi-scale · Pyramid grid template

## 1 Introduction

With the increase of video in the real world, especially at airports, railway stations, and urban traffic systems, video analysis is becoming increasingly important for video surveillance. For security applications, video anomaly detection algorithms are becoming more and more important [9].

Many approaches have been proposed to improve video anomaly detection. These methods always detect anomalies in video frames using a regular grid, such as  $4 \times 4$  [14],  $4 \times 5$  [4],  $5 \times 5$  [52],  $10 \times 10$  [10, 28],  $13 \times 13$  [7],  $15 \times 15$  [4, 5],  $16 \times 12$  [46],  $16 \times 16$  [12],  $20 \times 20$  [47],  $30 \times 30$  [20],  $48 \times 48$  [30],  $80 \times 80$  [53], etc. This single-scale grid strategy may speed up anomaly detection while decreasing anomaly detection accuracy. Therefore, the multi-scale strategy has received much attention in many works [23, 24, 28]. Although the multi-scale strategy is widely used for anomaly detection, repeated computations at the same location for multi-scale features directly lead to a large increase in computations. To reduce

---

✉ Shifeng Li  
lishifeng@qyemail.bhu.edu.cn

<sup>1</sup> School of Information Science and Technology, Bohai University, Science and Technology Road, Jinzhou 121000, Liaoning, China

the computations, we first compute the low-scale features and then use the result to compute the high-scale features. This can greatly improve the efficiency of feature extraction.

In our work, scale is defined by the various combinations of adjacent basic grids. For example, suppose the basic grid is  $3 \times 3$ , then the second scale is considered to be the  $2 \times 2$  adjacent basic grid combination with size  $6 \times 6$ , the third scale is the  $3 \times 3$  basic grid combination with size  $9 \times 9$ , the fourth scale is the  $4 \times 4$  basic grid combination with size  $12 \times 12$ , and so on. These different combinations are called multiscale pyramids. We also develop a fast algorithm to compute the motion features at other scales based on the features of the base scale using a convolution operation, which is much more efficient than computing the motion feature of different scales directly. In addition, the vote strategy is introduced into the pyramid combination and two types of scene models are proposed for robust anomaly detection.

In summary, the contributions of our work are summarized as follows:

- We design a normalized histogram motion feature based on optical flow and extend it to multi-scales to capture each region's motion characteristics at a different scale. Furthermore, a fast algorithm for computing the other scale motion representation based on the basic histogram is proposed.
- Since the scene model can feedback the prior knowledge into anomaly detection, we propose to use the global and online scene models to guide anomaly detection.
- We develop a new method which integrates the vote and pyramid strategies [21] to combine the different scale anomaly detection results.

The rest of this paper is organized as follows. Section 2 provides a brief overview of previous works on anomaly detection. The detailed explanation of our method is provided in Section 3. Section 4 demonstrates the effectiveness of the proposed algorithm in the published datasets, followed by the conclusion in Section 5.

## 2 Related works

Many methods have been proposed for abnormal event detection. The trajectory-based method is usually used in the first days to detect abnormal events [2, 13, 18, 32, 34]. For example, Johnson and Hogg [18] use Active Shape Models (ASM) to track the trajectories and learn a model from the trajectories to detect events. Basharat et al. [2] propose to model the pixel-level probability density functions of object velocity and size from the traces by using a multivariate Gaussian Mixture Model (GMM) for anomaly detection. In [39], clustering with a support vector machine (SVM) is used to detect anomalous trajectories. Mo et al. [34] use joint sparse reconstruction for video anomaly detection. The main problem of the trajectory method is the detection accuracy of the trajectories. If the trajectories are detected accurately, the abnormal events can be detected easily. To ensure the accuracy of the trajectories, Hu et al. [13] propose a robust multiple object tracking algorithm based on a fast, accurate fuzzy K-means algorithm, and use the detected trajectories to detect anomalies using statistical theory. However, due to the complex scenes, it is impossible to estimate the trajectories based on the unreliable detection of moving objects, leading to the anomaly detection failures.

To take full advantage of video features, low-level features are always used in anomaly detection, such as Histogram of Gradient (HOG), Histogram of Optical Flow (HOF), and image gradient. For example, Zhao et al. [53] use sparse coding and online reconstructability for anomaly detection based on HOG and HOF. Cong et al. [4] also use a sparse coding

strategy for abnormal event detection based on the Multi-scale Histogram of Optical Flow (MHOF), an extension of HOF. The motion features HOG and HOF are also adopted by [50] based on a statistical hypothesis test, and the abnormal events are identified as those that contain abnormal event patterns and have a high abnormality detector value. In addition, gradient information is also used for abnormality detection. In [20], the 3D gradients of Spatial-temporal Volumes (STVs) are proposed to detect anomalous events based on a Hidden Markov Model (HMM). Lu et al. [28] propose a sparse combination learning algorithm for detection using these 3D STVs. Yu et al. [49] also use the 3D STVs for anomaly detection with a content-adaptive sparse reconstruction method. A similar gradient feature is also used as an appearance feature in [10, 16, 52].

With the development of deep learning, the output feature map of the neural network is generally used [11, 16, 17, 27, 36, 37, 43, 54]. Hasan et al. [11] build a fully convolutional feed-forward autoencoder to learn the local features and classifiers in an end-to-end learning system. Ionescu et al. [16] consider the activation maps of the last convolutional layer of VGG-f [3] as appearance features and combine them with 3D gradient motion features to detect abnormal events. Liu et al. [27] first use an encoder to encode the first  $t$  frames, and then feed the output into a Convolutional LSTM (ConvLSTM) [42] to obtain the final features for anomaly detection by margin learning. Motion and appearance features obtained from three convolutional autoencoders are clustered and the one-versus-rest scheme is utilized to score the anomalous events in [17]. Nguyen and Meunier [36] use a convolutional autoencoder and a U-net [41] to learn the joint appearance structures and motion patterns for anomaly detection. Most of the above methods focus on a new representation to support anomaly detection. These methods separate anomaly detection and data representation learning, resulting in a suboptimal representation for specific anomaly detection. Therefore, end-to-end methods are proposed in [11, 37, 54]. Pang et al. [37] use a few labeled anomalies and a prior probability to perform end-to-end learning of anomaly scores. Zhou et al. [54] combine feature learning, sparse representation and dictionary learning in a new neural network. Although neural networks have made a great progress in anomaly detection, some problems still need to be solved, such as expensive computational power, long training time, design of cumbersome network structure, and so on. Therefore, in this paper, we use the traditional anomaly detection method based on normal video motion.

### 3 Our work

In this work, anomaly detection is considered as a template matching problem, where the feature of the testing motion histogram in each grid is compared with the trained template to evaluate the anomalous events at each scale. A scene model can incorporate the knowledge of the scene into the anomaly detection. Therefore, we use a trained scene model as a reference to limit the range of anomaly detection, and employ an online scene model to control anomaly detection outside this range. The framework of our method is shown in Fig. 1. The top and bottom subgraphs describe the training process and the middle one describes the testing process.

#### 3.1 Scene model

Inspired by the fact that the abnormal events usually occur on the moving objects and cannot occur on the background, we use a scene model to describe this property. We have two kinds

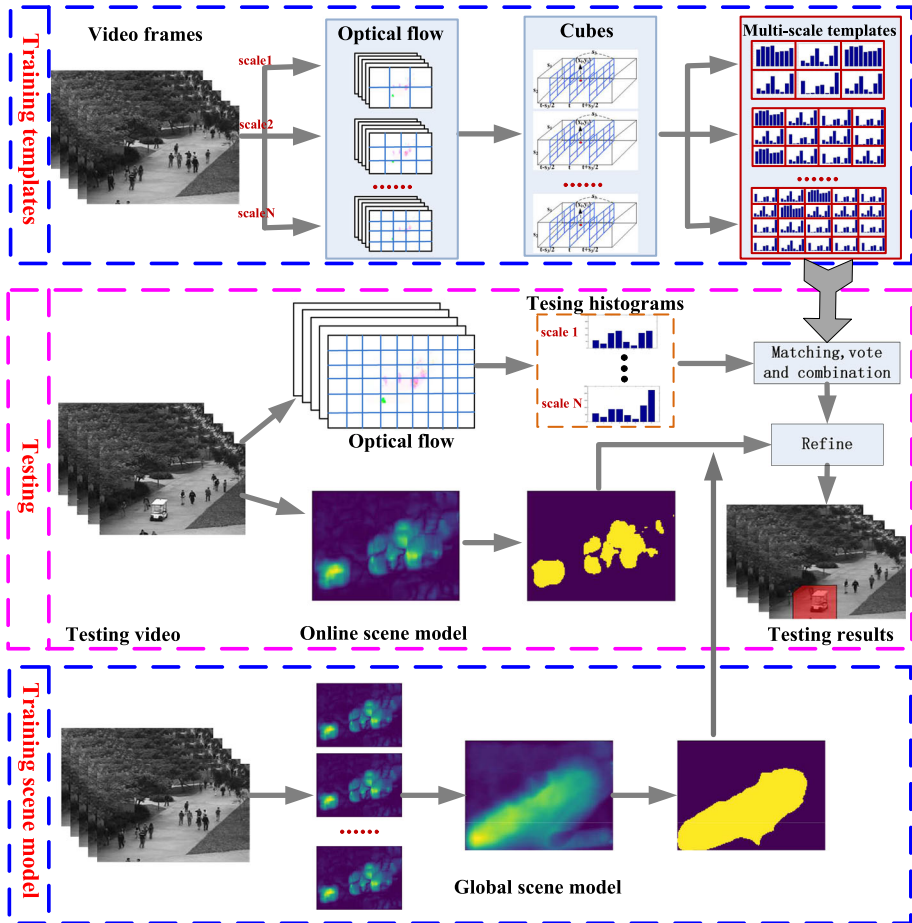


Fig. 1 Framework of our method

of scene models: a global reference scene model and an online model. The scene model is modeled at the pixel level and is based on the optical flow obtained from [8]. Assume that the optical flow at the point  $(x, y)$  is  $(v_x, v_y)$ , where  $v_x$  and  $v_y$  are the horizontal and vertical components of the optical flow. The magnitude  $M$  at pixel  $(x, y)$  is calculated as follows:

$$M_{(x,y)} = \sqrt{v_x^2 + v_y^2}. \tag{1}$$

The magnitude of motion represents the intensity of object motion. If the magnitudes in a certain region are smaller than a threshold, it means that this region could be more of a background. So the region of moving objects can be determined as follows:

$$B_{(x,y)} = \begin{cases} 1, & M_{(x,y)} \geq \theta \\ 0, & M_{(x,y)} < \theta \end{cases} \tag{2}$$

Where  $\theta$  is a threshold, obtained by an adaptive method [35]. For the global scene model, we first compute magnitudes of each frame in the training samples, and use (2) to binarize each frame. Then, we calculate the average of the whole binarization frames in the training

dataset to obtain the scene model. Finally, we estimate one threshold from the global scene model (more details can refer to Section 4.1 Parameters Setting) and use it to obtain the corresponding binarization of global scene model. In the testing phase, we consider the magnitude obtained from (1) as the online scene model. Figure 2 shows the scene model on the UCSD Ped1 dataset. With the global scene model, we can determine the area that the object usually passes through. While with the online scene model, we can determine the area that the object is currently passing through.

In general, the moving region (foreground) in the global scene model should contain the region under the online scene model. If the objects appear in the background region of the global scene model, it is more likely that anomalies will occur in that region. Figure 2(e) shows an example. The regions of the online scene model (yellow regions in Fig. 2(d)) are mostly in the foreground of the global scene model. The regions outside the global scene model (red regions in Fig. 2(e)) indicate that few objects in the training set occur in that region. Therefore, we process two different criteria inside and outside the global scene model.

### 3.2 Normalized histogram and fast computation

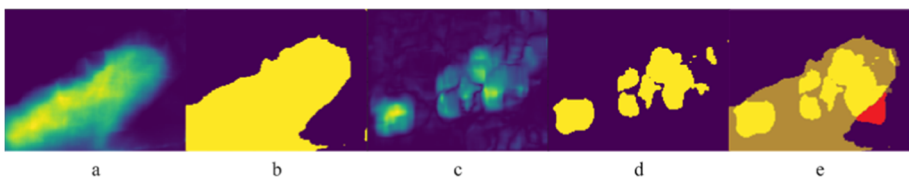
First, we propose a normalized histogram for representing object motion at one scale. Second, we extend this normalized motion feature into multi-scales and design a fast algorithm for high-scale histogram computation based on the base scale histogram. Third, we introduce how to obtain the MPGT from calculated motion features. Last, we introduce the method to score the testing frames using an anomaly detection score function, as well as the scheme to combine multi-scale detections using vote and pyramid strategies.

#### 3.2.1 Normalized histogram feature

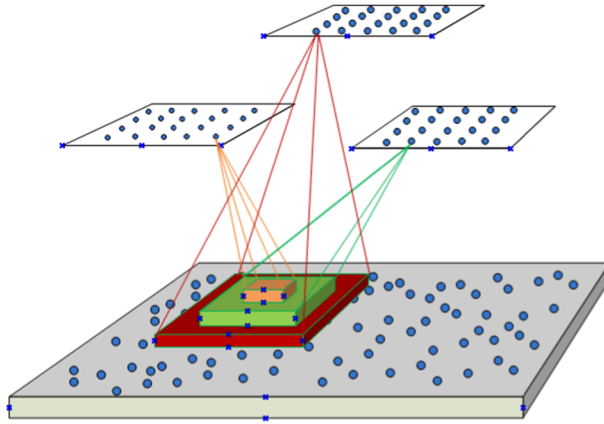
Many works use histograms, such as HOG [6], HOF [53], and MHOF [4]. Our motion feature is also based on histograms. We also extend the optical flow based histogram to include normalization for each bin. Suppose each frame is divided into  $s \times s$  grids, and the grids at the same location with continuous  $t$  frames form a cube. A histogram is computed for each cube with the size  $s \times s \times t$ . Each pixel  $(x, y)$  in a cube is quantified into one of  $d$  directions based on its angle:

$$\theta_{(x,y)} = \tan^{-1}(v_y/v_x). \quad (3)$$

We calculate a histogram for each direction with magnitude as the weight. Let the weighted histogram at the location  $(x_{cube}, y_{cube})$  be  $H_{(x_{cube}, y_{cube})} = [h_1, h_2, \dots, h_d]$ ,



**Fig. 2** Our scene model on the UCSD Ped1 dataset. (a) is the global scene model. (b) is the binarization of the (a). (c) is an online scene model of one frame. (d) is the corresponding binarization of (c). (e) the trace of moving object is in the background and is colored with red. The yellow region in (b) or (d) represents the foreground and the black stands for the background



**Fig. 3** Multi-scale pyramid histogram architecture. The video is divided into different scale cubes which are colored with orange, green and red, to extract the feature at different scales. The all scale motion features compose the multi-scale pyramid motion features, which can be used to train MPGT

where  $h_i$  represents the sum of weight in  $i$ th direction. Let the standard histogram be  $B_{(x_{cube}, y_{cube})} = [b_1, b_2, \dots, b_d]$ , where  $b_i$  is the count of the pixels located in the  $i$ th direction. The final histogram  $H_{(x_{cube}, y_{cube})}^f$  is normalized with the  $B_{(x_{cube}, y_{cube})}$ :

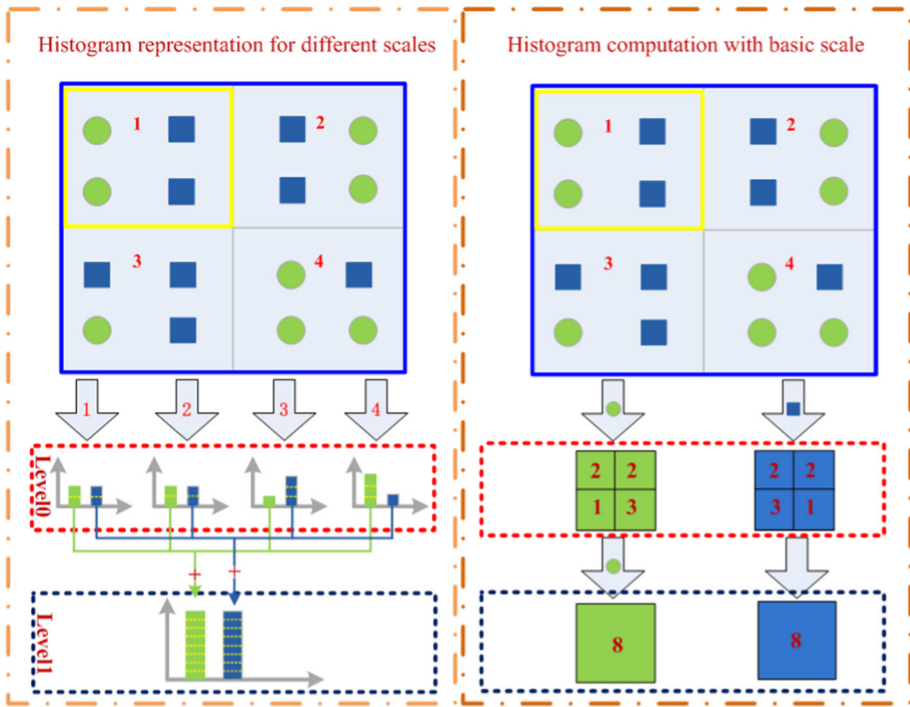
$$\begin{aligned}
 H_{(x_{cube}, y_{cube})}^f &= H_{(x_{cube}, y_{cube})} ./ B_{(x_{cube}, y_{cube})} \\
 &= [h_1/b_1, h_2/b_2, \dots, h_d/b_d],
 \end{aligned}
 \tag{4}$$

where the operator  $./$  is the element-wise division, and here suppose that  $b_i \neq 0$ . If  $b_i$  is zero, the corresponding histogram is 0 too.

### 3.2.2 Fast algorithm for multi-scale histogram

We define the concept of scale differently from the previous works [21, 48], which defined scale as the image resolution. We define the scale with reference to the receptive field. In Fig. 3, for example, different grid sizes represent different scales. The histogram of each scale can be calculated from (4) if the  $i$ th scale is  $s_i \times s_i$  (we remove  $t$  from  $s_i \times s_i \times t$  for simplicity). The computation is extremely large and wasteful if the histograms are computed directly based on optical flow. Therefore, we propose a rapid algorithm to complete this task using the convolution operation.

According to the histogram definition, the value of each bin for each scale is defined as the summary of pixels in that direction. Figure 4 provides an example of this computation. In the left subfigure, the circle and square stand for two different bins, and the yellow square represents a base scale (level 0), whose corresponding histograms are shown in the red dotted rectangle. The blue square represents a high scale (level 1) and its histogram is represented in the dark dotted rectangle. As shown in this example, the high scale region of level 1 contains four base scale regions of level 0. The histogram of each bin on the level 1 scale is the sum of the four corresponding bins on the level 0. It is efficient to calculate the histogram of level 1 from level 0 using a simple addition operation. The level 1 histogram calculation is displayed in the right subfigure. The histogram of the first bin on level 0 is [2, 2, 1, 3] (see the green



**Fig. 4** Toy example of our fast multi-scale histogram computation. The yellow square on the top subgraph stands for the base scale (level 0) and the blue square represents the second scale (level 1)

square in the left subgraph). The histogram value on level 1 is 8 and is also the convolution of level 0 with a  $2 \times 2$  kernel of value 1.

For one video, let the base scale histogram without normalization at location  $(x, y)$  be  $H_{(x,y)} = [h_1, h_2, \dots, h_d]$ , and the standard histogram be  $B_{(x,y)} = [b_1, b_2, \dots, b_d]$ . Based on the base scale histogram, we can calculate the histogram for any scale as long as the size of the other scales is an integer multiple of the base scale. That is, suppose that the base scale is represented as  $s_1 \times s_1$ , we can obtain the histogram of other scale, whose spatial size is  $s_2 = I \times s_1$ , where  $I$  is one positive integer, using the convolution operation. The algorithm is shown in Alg. 1, where  $w$  is a multiple of the base scale size. Suppose the size of the base scale is  $3 \times 3$ . If we want to obtain the histogram of  $6 \times 6$ , the kernel size should be  $2 \times 2$  with  $w = 2$ . Similarly, we can obtain the other scale motion histograms using Alg. 1, and then calculate the normalized motion feature employing (4).

### 3.3 Multi-scale pyramid grid templates

Based on the normalized histogram, we can calculate the MPGT by combining each maximum grid template at every scale. Suppose there are  $n_h$  normalized histograms at location  $(x_i, y_i)$  in the whole training data, i.e.  $H^1_{(x_i,y_i)}, H^2_{(x_i,y_i)}, \dots, H^{n_h}_{(x_i,y_i)}$ . We model the maximum grid template at location  $(x_i, y_i)$  as follows:

$$g_{(x_i,y_i)} = \max(H^1_{(x_i,y_i)}, H^2_{(x_i,y_i)}, \dots, H^{n_h}_{(x_i,y_i)}), \tag{5}$$

**Algorithm 1** Histograms computation using convolution.

**Input**

- $H$ : the basic weighted histogram.
- $B$ : the basic standard histogram with the same shape as  $H$ .
- $w$ : the kernel width.

**Output**

- $H_c$ : the output weighted histogram,
- $B_c$ : the output standard histogram.
- for** each direction  $i$  **do**
  - step 1:** Obtain the weighted and standard histogram at the  $i^{th}$  direction.  
 $h_i = H[... , i], b_i = B[... , i]$ .
  - step 2:** Produce a 2D convolution kernel  $k$ , with the kernel width as  $w$  and values as 1.  
 $k = Conv2D(w, w)$
  - step 3:** Compute 2D convolution between  $k$  and  $h_i$ .  
 $h_i^c = k \odot h_i$ , where  $\odot$  is convolution operation.
  - step 4:** Save the output weighted histogram.  
 $H_c[... , i] = h_i^c$ .
  - step 5:** Compute 2D convolution between  $k$  and  $b_i$ .  
 $b_i^c = k \odot b_i$ .
  - step 6:** Save the output standard histogram.  
 $B_c[... , i] = b_i^c$ .

**end for**

where the max operation selects the maximum histogram in every bin for d directions among all training histograms. Therefore, our maximum grid template can capture the movement distribution in various locations. For the  $i$ th scale, suppose one frame can be divided into non-overlap  $m \times n$  grids. With each local maximum grid template, we can obtain the maximum grid template for the  $i$ th scale

$$G = \begin{bmatrix} g(x_1, y_1) & g(x_1, y_2) & \cdots & g(x_1, y_n) \\ g(x_2, y_1) & g(x_2, y_2) & \cdots & g(x_2, y_n) \\ \cdots & \cdots & \cdots & \cdots \\ g(x_m, y_1) & g(x_m, y_2) & \cdots & g(x_m, y_n) \end{bmatrix}. \tag{6}$$

We also extend the template into the multi-scale cases. Intuitively, the cube with small size can detect the details of the motion. The cube with the large size can capture the high-level object knowledge. Therefore, we describe the motion using different scales at the same time to improve the detection. Suppose the first scale maximum template is  $G_0$ , it can be obtained from the basic histograms of  $s_0 \times s_0 \times t$  cubes. Then, we obtain the second scale histograms using Alg. 1 with the basic histograms, as well as the second scale maximum template  $G_1$ . The third and other scale maximum template  $G_i$  can be also achieved using the same strategy. All the different scale templates compose the MPGTs.

**3.4 Anomaly detection**

With our proposed MPGT, we can finish the anomaly detection task. In this section, we first introduce how to score the testing frames, and then describe the combination of multi-scale detection results.



### 3.4.1 Anomaly detection score function

We use the logistic function to score the final anomaly detection score. Let the histogram  $H_{(x,y)}^{max} = (h_1^{max}, h_2^{max}, \dots, h_d^{max})$  be one scale maximum template at location  $(x, y)$ , and  $H_{(x,y)} = (h_1, h_2, \dots, h_d)$  be the corresponding testing histogram also located at  $(x, y)$ . The score function for the location  $(x, y)$  grid is expressed by:

$$s_{(x,y)} = \frac{1}{1 + \exp(-\beta * \sum_{\substack{k=1, \dots, d \\ \wedge h_k > h_k^{max}}} (h_k - h_k^{max}))}, \quad (7)$$

where  $\beta$  is the logistic function parameter, and  $\wedge$  is the AND operation. The bins whose value are larger than the corresponding bins of maximum template in (7) are considered. The template  $g_{(x,y)}$  at location  $(x, y)$  stands for the maximum motion intensity. If one bin of a new observation is larger than the corresponding bin of the template, it is abnormal for this new observation at this location. The larger the new observation bin value exceeds the grid template, the more likely the abnormal events will appear.

We can divide each frame into the foreground and background regions with the global and online scene models. Then we detect anomaly in the foreground using (7) and background with a weighted MPGT:

$$G = \begin{bmatrix} g_{(x_1, y_1)} & g_{(x_1, y_2)} & \dots & g_{(x_1, y_n)} \\ g_{(x_2, y_1)} & g_{(x_2, y_2)} & \dots & g_{(x_2, y_n)} \\ \dots & \dots & \dots & \dots \\ g_{(x_m, y_1)} & g_{(x_m, y_2)} & \dots & g_{(x_m, y_n)} \end{bmatrix} * c, \quad (8)$$

where  $c$  is a softening factor, which can reduce the template effectiveness in the background regions. If  $c = 1$ , the whole frame has the same template for anomaly detection. In fact, if some object moves in the background, the anomaly has more probability to occur. Therefore, we set  $c < 1$  to decrease the anomaly detection condition and improve the accuracy of detecting abnormal events in the background. For all testings, we set  $c = 0.8$  in all experiments.

### 3.4.2 Strategy for combination of multi-scale detection

Many applications use the pyramid approach, such as scene category recognition [21], image classification [48], object detection [25] and event detection [51]. In this work, we first calculate the probability of abnormalities on each grid location at each scale using (7). Then, we use a pyramid strategy to combine multi-scale detection results and incorporate the voting strategy in the pyramid scheme:

$$p(x, y) = \sum_{i=0}^{n_s} 1/2^{n_s-i} s_{(x,y)}, \quad (9)$$

$$i=0 \wedge \text{vote}(x, y) > v$$

where  $n_s$  is the total scale levels,  $\text{vote}(x, y)$  stands for vote strategy at location  $(x, y)$ ,  $\wedge$  is the AND operation. For example,  $\text{vote}(x, y) > 2$  indicates that if there are more than two abnormal events, this position  $(x, y)$  would be considered as abnormal candidates. Obviously, if  $\text{vote}(x, y) > 0$ , Eq. (9) is similar to that in [28]. In our work, we put more emphasize on the high scale for that the high level can capture more significant holistic feature.

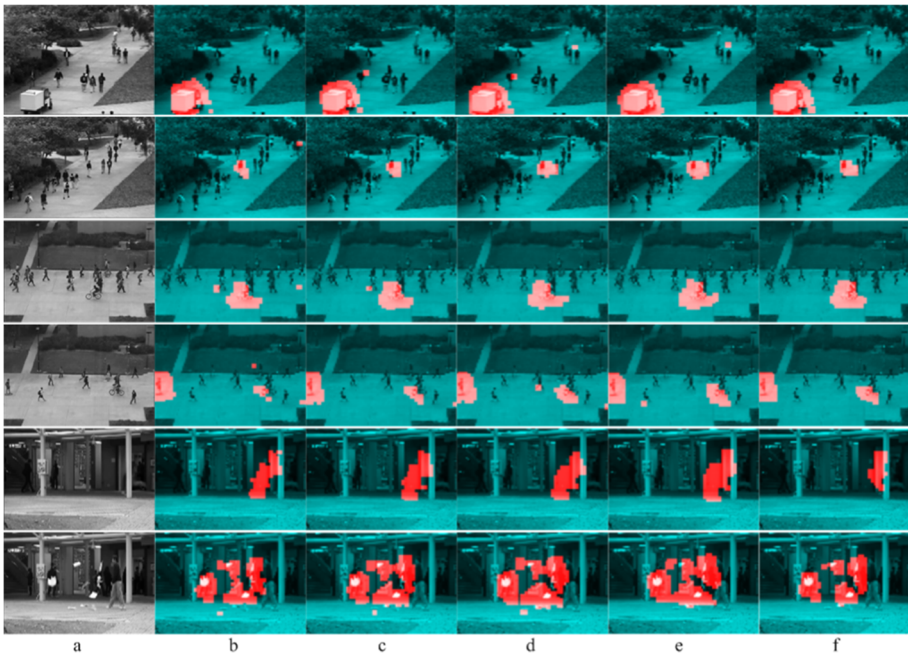
## 4 Experiments

In this section, we demonstrate the experimental performance of the proposed algorithm using the published datasets UCSD [31] and Avenue [28]. The video frames are first downscaled to a resolution of  $120 \times 160$  and four scales of  $3 \times 3$ ,  $6 \times 6$ ,  $9 \times 9$  and  $12 \times 12$  are used for the motion features in all experiments. The base scale is set to  $3 \times 3$  and the motion feature of the other scales is calculated based on the first scale using the fast algorithm proposed in Section 3.2.2. Figure 5 shows some detection results of our method. As can be seen in the figure, our approach can effectively detect various anomalies, such as bicyclists, runners, skaters, and cars.

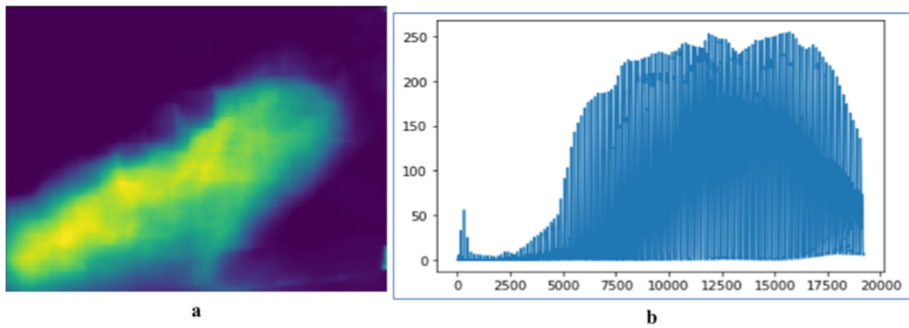
### 4.1 Parameters setting

There are three types of parameters in our method. The first is the threshold in the scene model, the second is  $\beta$  in (7), and the last is the bin number  $d$  of the histogram in the low-level feature.

Intuitively, the magnitudes in (1) represent motion intensity, so they should be low for the background and high for the moving objects. To determine the threshold in the scene model, we first flatten the scene model and plot it with the number of pixels as the y-axis and each pixel position as the x-axis. Then we examine the threshold on the low point in the graph. In



**Fig. 5** Detection samples on the published datasets, with the red mask for the anomaly region. (a) is original images. (b) is the detection at the first scale. (c) is the detection at the second scale. (d) is the detection at the third scale. (e) is the detection at last scale. (f) is the detection using our combined strategy



**Fig. 6** Threshold decision on our scene model. (a) is the scene model on UCSD Ped1 dataset. (b) is the plot for the scene model with each pixel as x-axis and the average of pixel appearing times as y-axis

our experiments, we set the threshold value to 25, 20, and 35 for the UCSD Ped1, Ped2, and Avenue dataset respectively. Figure 6 shows the procedure for estimating the threshold for the UCSD Ped1 dataset.

For the other two parameters, we find that both parameters have limited effectiveness in detecting abnormalities. To test the influence of the parameters, we first fix one parameter and change the other within a certain range. We then compute Area Under the Curve (AUC) for the evaluation on UCSD Ped1 dataset. The comparison with different bins using  $\beta = 0.01$  is shown in Table 1. As shown in this table, when the number of bins increases, the AUC decreases a little, which implies that the small bins can perform well, and when the bins are 2, the performance is the best. Intuitively, when the number of bins is small, there are more pixels in a bin, and more data makes the histogram more robust against noise.

Table 2 shows the AUC comparison for different  $\beta = 0.0001, 0.001, 0.01, 0.05, 0.1, 1, 5$  on the UCSD Ped1 dataset, where the movement histogram is fixed at 2 bins. As can be seen from the comparison, when  $\beta < 0.001$ , the larger  $\beta$ , the lower the AUC. When  $\beta > 0.001$ , the smaller  $\beta$  is, the smaller the AUC is. The average AUC hovers around 0.88, indicating that our method is robust to the parameter changes. In the following experiments with the UCSD dataset, we set  $\beta = 0.001$  and  $d = 2$ .

## 4.2 Experiments on UCSD dataset

In the UCSD dataset, there are two pedestrian scenes on campus, Ped1 and Ped2. Each dataset contains a training set and a testing set. For UCSD Ped1, the training set contains 34 short clips for learning normal patterns and the testing set contains 36 short clips. Each clip in Ped1 has 200 frames with a resolution of  $158 \times 238$ . The anomalies occur in multiple frames in each clip in the testing set, and there is a subset of 10 clips with pixel-level binary masks that can be used to identify regions that contain anomalies. The UCSD Ped2 dataset contains

**Table 1** Comparison with different histogram bins on UCSD Ped1 dataset using frame-level ground truth

<i>bins</i>	1	2	4	6	8
AUC	0.8719	0.8757	0.8621	0.8480	0.8380

**Table 2** Comparison with different  $\beta$  on UCSD Ped1 dataset using frame-level ground truth

$\beta$	0.000001	0.00001	0.0001	0.001	0.005	0.01	0.05	0.1
AUC	0.8726	0.8763	0.8797	0.8804	0.8781	0.8757	0.8652	0.8594

scenes of pedestrians moving parallel to the camera at a resolution of  $240 \times 360$ , with 16 clips for training and 12 clips for testing. In the following experiments, we evaluate our approach based on frame and pixel ground truth.

#### 4.2.1 Evaluation on UCSD Ped1 dataset

For the frame-level comparison, we compare our approach with the state-of-the-art methods, including Scan Statistic (SS13) [15], Sparse Combination (SC13) [28], Spatio-Temporal Motion Context (STMC) [5], Convolutional Auto Encoder (Conv-AE) [11], Combining Motion and Appearance (CMA16) [52], Winner-Take-All SVM (WTA-SVM) [44], our former work MAP [24], MultiLevel Anomaly Detector (MLAD) [45], Multivariate Gaussian fully Convolution Adversarial Autoencoder (MGCAA) [22] and Self-trained Deep Ordinal Regression (SDOR) [38]. These compared methods not only contain the traditional machine learning methods, such as SS13 [15], SC13 [28], STMC [5], MAP [24] and CMA16 [52], but also contain the deep learning methods, e.g. Conv-AE [11], WTA-SVM [44], MLAD [45], MGCAA [22] and SDOR [38]. The comparison based on the frame-level ground truth is shown in the second column in Table 3<sup>1</sup>. On the UCSD Ped1 dataset, our method wins the second place in the compared methods, is comparable to SC13 [28], and is better than the other compared methods. Although our method is a little worse compared with SC13 [28], the performance is better than SC13 on the Avenue dataset.

To evaluate our method at the pixel level, we use an adaptive threshold to obtain the binarization of each frame. Following [5, 28, 31, 40], the corresponding frame is considered correctly detected if more than 40% of the truly anomalous pixels are detected. The parameters used for pixel-level comparison are the same as those used for frame-level evaluation. We compare our method with Adam [1], Mixture of Probabilistic Principal Component Analyzers (MPPCA) [19], Social Force (SF) [33], MPPCA+SF [31], Mixture of Dynamic Textures (MDT) [31], Spatio-Temporal Motion Context (STMC) [5], Sparse [4], Stacked RNN (SRNN) [30], Future Frame Prediction (FFP) [26], and Anomaly Net (ANet) [54]. The comparison is shown in the second column in Table 4. It is obvious that our method has the best performance of all compared methods, which shows the effectiveness of our approach.

#### 4.2.2 Evaluation on UCSD Ped2 dataset

In the UCSD Ped2 dataset, the parameter settings are the same as in the UCSD Ped1 dataset. For frame-level comparison, in addition to the above methods, we also compare our method with Temporally-coherent Sparse Coding (TSC) [30], Unmask [16], and ConvLSTM- AE [29]. The comparison is reported in the third column in Table 3. The performance of our proposed method is similar to SS13 [15], slightly worse than WTA-SVM [44] and MLAD [45], but better than the other methods. In fact, the methods of WTA-SVM [44] and MLAD

<sup>1</sup> The results have been retained with two significant digits, for that we obtain the performance from different references, where some of the detection values only have two significant digits.

**Table 3** Comparison with the state-of-the-art methods on UCSD and Avenue datasets at frame-level

Methods	Ped1	Ped2	Avenue	Ped12	Ped1A	Ped2A	AllID
SS13 [15]	0.87	0.94	-	0.905	-	-	-
SC13 [28]	0.91	-	0.84	-	0.875	0.88	-
STMC [5]	0.85	0.86	-	0.855	-	-	-
Conv-AE [11]	0.81	0.90	0.70	0.855	0.755	0.8	0.803
CMA16 [52]	0.85	0.90	-	0.875	-	-	-
WTA-SVM [44]	0.81	0.97	-	0.89	-	-	-
MLAD [45]	0.82	0.98	0.72	0.90	0.77	0.82	0.84
Unmask [16]	-	0.82	0.81	-	-	0.815	-
MAP [24]	0.86	0.92	0.87	0.89	0.865	0.895	0.883
ConvLSTM-AE [29]	-	0.77	0.88	-	-	0.825	-
TSC [30]	-	0.81	0.91	-	-	0.86	-
MGCAA [22]	0.85	0.916	0.842	0.883	0.846	0.879	0.869
SDOR [38]	0.717	0.832	-	0.775	-	-	-
Our method	0.88	0.93	0.88	0.905	0.88	0.905	0.897

[45] have better performance than our method in the Ped2 dataset, but their performance in the UCSD Ped1 dataset is worse than our method.

The comparison with state-of-the-art methods based on pixel-level ground truth at UCSD Ped2 dataset is shown in the third column in Table 4. The compared methods include [26], SRNN [30], and ANet [54]. As can be seen from the comparison, our method wins the first place among all the compared methods.

### 4.3 Experiments on Avenue dataset

The videos in the Avenue dataset published by [28] are from an avenue on the CUHK campus with a total of 30652 frames, including 16 videos in the training set and 21 videos in the testing set, and the pixel-level ground truth is also provided by the authors. This dataset

**Table 4** Comparison with the state-of-the-art methods on UCSD datasets at pixel-level ground truth

Methods	Ped1	Ped2
Sparse [4]	0.461	-
Adam [1]	0.197	-
MPPCA [19]	0.133	-
MPPCA+SF [31]	0.205	-
SF [33]	0.213	-
MDT [31]	0.441	-
STMC [5]	0.471	-
FFP [26]	0.334	0.406
SRNN [30]	-	0.448
ANet [54]	0.452	0.528
Our method	0.493	0.535

presents a greater challenge in three respects: (1) The camera vibrates in some videos. (2) There are a few outliers in some testing videos. (3) Some normal patterns rarely occur in the training data. Similar to the experiments on the UCSD dataset, we downscale the sequences in this dataset to a resolution of  $120 \times 160$ , set the quantized histogram directions to 8, and set  $\beta$  to 20.

In addition to the methods of SC13 [28], Conv-AE [11], and MLAD [45], we also compare our method with TSC [30], Unmask [16], and ConvLSTM-AE [29]. The comparison is shown in the fourth column in Table 3. As shown in this comparison, the AUC of our method is 0.88, which is better than SC13, Conv-AE [11], Unmask [16], and MLAD [45], on par with ConvLSTM-AE [29], and a little worse than TSC [30]. The TSC [30] method has the best performance on the Avenue dataset among the compared methods, and our method ranks second. But the performance of our method is better than that of TSC [30] on the UCSD Ped2 dataset. Also, the method ConvLSTM-AE [29] has the same performance as our method on the Avenue dataset, but the performance of ConvLSTM-AE [29] on the UCSD Ped2 dataset is worse than our method.

#### 4.4 Balanced performance comparison

To further demonstrate the superiority of our method, we calculate the AUC average for different combinations, e.g., the AUC average for the UCSD subsets Ped1 and Ped2 (we call it Ped12), the AUC average for the Ped1 and Avenue datasets (Ped1A), the AUC average for Ped2 and Avenue (Ped2A), and the AUC average for all testing datasets (AllD). The results are shown in the last columns in Table 3, i.e., the fifth column represents the AUC average on Ped12, the sixth column represents the AUC average on Ped1A, the seventh column Ped2A, and the last column represents the AUC average on all datasets. We can see that our method and SS13 have the same performance on Ped12 and rank first on Ped12. In addition, our method has the best performance on all combined datasets, including Ped1A, Ped12, Ped2A, and AllD, indicating that our method has balanced performance on all test datasets.

#### 4.5 Validation of our method

In recent years, deep learning-based methods have been applied to anomaly detection with great success [11, 29, 30, 38, 54]. However, the main weakness is the dependence on a large amount of hardware resources. Unlike deep learning methods, our proposed method requires little GPU hardware, and you can train the template model quickly. Moreover, our method has low dataset size requirements and needs only one template in the testing phase, which requires little memory. Moreover, the experiments on public datasets show that the performance of our method is also comparable to deep learning-based methods.

##### 4.5.1 Effectiveness of our scene model

The scene model proposed in this paper is simple but effective in detecting anomalies. To further demonstrate that the scene model can improve the accuracy of anomaly detection, we compare the performance between the methods with and without the scene model on the UCSD and Avenue datasets using the AUC. We also use  $\beta = 0.001$  and 2 bins for all evaluations on the UCSD dataset and  $\beta = 20$  and 8 bins on the Avenue dataset. As reported in Table 5 (second and third columns), the performance with the scene model is similar to

**Table 5** Comparison between the method with scene model (SM) and the one without (the second and third columns), and the comparison between the method with normalization and the one without (the second and last columns)

Dataset	with SM	No SM	No normalization
UCSD Ped1	0.8804	0.8509	0.7823
UCSD Ped2	0.9303	0.9152	0.7926
Avenue	0.8790	0.8424	0.8092

that without the scene model in the UCSD Ped1 dataset, but better than the performance in the UCSD Ped2 and Avenue datasets, indicating that the proposed scene model actually increases the anomaly detection performance.

#### 4.5.2 Necessity of histogram normalization

Normalization of the histogram is necessary for robust anomaly detection. It can increase the robustness to noise and improve the accuracy of anomaly detection. To test the effectiveness of histogram normalization, we also calculate the AUC for the UCSD and Avenue datasets using the same parameters as the previous experiments, i.e., we set  $\beta = 0.001$  and 2 bins for the UCSD dataset and use  $\beta = 20$  and 8 bins for the Avenue dataset. Table 5 shows the comparison between the histogram with and without normalization (the second and last columns). As can be seen in this table, the AUC for all test datasets is much better than that without normalization, which means that normalization to the motion feature is essential for anomaly detection.

#### 4.5.3 Validity of multi-scale pyramid histograms

Multi-scale histogram features can benefit the accuracy of anomaly detection. To show the effectiveness of multi-scale strategy, we compare the method using four scales histogram feature and the ones employing only one single scale. In this experiment, the parameter settings are the same as in the above experiments with the UCSD and Avenue datasets.

Similar to the above experiments, we employ four scales for the multi-scale experiment in testing, and use  $3 \times 3$ ,  $6 \times 6$ ,  $9 \times 9$  and  $12 \times 12$  for one scale respectively. The comparison is shown in Table 6. As can be seen in this table, although the performance in the UCSD Ped2 dataset with the  $12 \times 12$  scale is slightly better than the multi-scale method, the AUC in the UCSD Ped1 and Avenue datasets with multi-scale features is significantly higher than all detections with only one scale, indicating that the multi-scale strategy does improve detection accuracy.

Figure 5 shows some results using different scales. From left to right, the first column is the original image, the second column is the detection of anomalies at the base scale of  $3 \times 3$ ,

**Table 6** Comparison between the method with multi-scale scheme and the one with only one scale

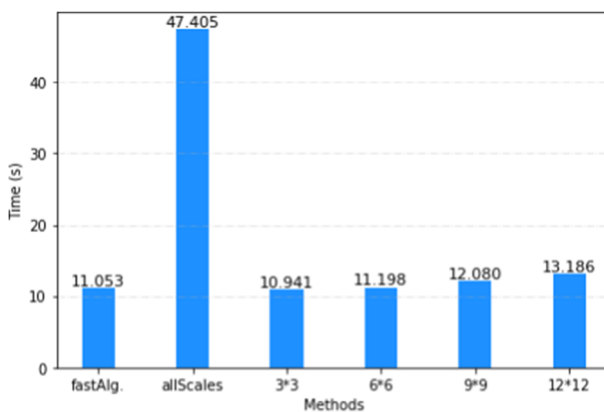
Dataset	multi-scale	$3 \times 3$	$6 \times 6$	$9 \times 9$	$12 \times 12$
UCSD Ped1	0.8804	0.8239	0.8315	0.8338	0.8335
UCSD Ped2	0.9303	0.8522	0.8935	0.9254	0.9382
Avenue	0.8790	0.8187	0.8239	0.8222	0.8159

the third column is the detection at the scale of  $6 \times 6$ , the fourth column is the detection at the scale of  $9 \times 9$ , the fifth column is the detection at the scale of  $12 \times 12$ , and the last column is the combined detection of the four scales. As these examples show, false alarms may occur in the small scale, and the detection result may be rough in the large scale. However, the combined detection can effectively eliminate the false alarms to improve the detection accuracy.

#### 4.6 Comparison for running time

In our method, we use the basic motion histogram of the grid to compute the motion histogram of the high scale grid, which can greatly improve the speed of feature computation. In this section, we compare the running time of the method with convolution and the method that computes the histogram directly. The experiments are performed on a laptop with Intel(R) Core i7-10750H CPU and 8G memory. The code was written using Python and is available at <https://github.com/limax2008/Anomaly-DetectionpyramidGridTemplates>.

Similar to the above experiments, the basic grid is set to  $3 \times 3$ , and we compute  $6 \times 6$ ,  $9 \times 9$ , and  $12 \times 12$  directly or using convolution, respectively. The video is first resized to  $120 \times 160$  and the motion histogram bins are set to 2. The experiment is finished by randomly selecting a video from the UCSD Ped1 dataset and repeated 10 times. The final running time is considered as the average of the 10 repetitions. The comparison is shown in Fig. 7. From the comparison, we can see that: (1) For direct computation, the time increases as the grid size increases. (2) Our fast algorithm for calculating all scale histograms is significantly faster than the direct calculation (more than 4 times faster). Moreover, it is worth noting that the running time of our fast algorithm is the total time including  $3 \times 3$ ,  $6 \times 6$ ,  $9 \times 9$ , and  $12 \times 12$  (the first bar in Fig. 7). If we directly subtract the time for computing the  $3 \times 3$  grid histogram, the time for computing the  $3 \times 3$ ,  $6 \times 6$ ,  $9 \times 9$ , and  $12 \times 12$  histograms is about  $(11.053 - 10.941) / 3 = 0.0373$  seconds for 200 frames at one scale, indicating that our proposed algorithm is fast enough for computing the motion histogram. Table 7 shows the comparison



**Fig. 7** Comparison of running time between the method using convolution and the method computing histogram directly



**Table 7** Comparison of running time standard deviation between the method using convolution and the method computing histogram directly

Methods	fastAlg	allScales	3 × 3	6 × 6	9 × 9	12 × 12
Times(s)	0.138	0.506	0.229	0.089	0.148	0.172

of the time standard deviation. We can see that the time standard deviation of our proposed method is small, which indicates that our method is stable for all videos.

## 5 Conclusion

In this work, we first improve the histogram by normalization to increase the robustness of the motion feature. Then, we propose a simple but powerful scene model for anomaly detection, where we combine the global scene model and the online scene model to detect anomalies based on their intersection. Next, we propose a fast motion feature computation algorithm based on the basic histogram feature using a convolution operation. Finally, we use our proposed MPGT for anomaly detection based on vote and pyramid strategies. Experimental results on public datasets show the effectiveness of the proposed method. Currently, anomaly detection considers only one type of motion features. There is a possibility to further improve the accuracy by considering appearance in our future work.

**Acknowledgements** This work was jointly supported by the National Natural Science Foundation of China (61402049), Science and Technology Research Project of the Department of Education of Liaoning Province (LJKZ1019) and Social Science Planning Fund of Liaoning Province (L21BGL002).

## Declarations

**Competing interest** The authors declare no conflict of interest.

## References

1. Adam A, Rivlin E, Shimshoni I, Reinitz D (2008) Robust real-time unusual event detection using multiple fixed-location monitors. *IEEE Trans Pattern Anal Mach Intell* 30(3):555–560
2. Basharat A, Gritai A, Shah M (2008) Learning object motion patterns for anomaly detection and improved object detection. In: 2008 IEEE Conference on Computer Vision and Pattern Recognition, 1–8
3. Chatfield K, Simonyan K, Vedaldi A, Zisserman A, Valstar MF, French AP, Pridmore TP (2014) Return of the devil in the details: delving deep into convolutional nets. In: Valstar MF, French AP, Pridmore TP (eds) British Machine Vision Conference, BMVC 2014, Nottingham, UK, September 1–5, 2014. BMVA Press
4. Cong Y, Yuan J, Liu J (2011) Sparse reconstruction cost for abnormal event detection, 3449–3456
5. Cong Y, Yuan J, Tang Y (2013) Video anomaly search in crowded scenes via spatio-temporal motion context. *IEEE Trans Inf Forensics Secur* 8(10):1590–1599
6. Dalal N, Triggs B (2005) Histograms of oriented gradients for human detection. *IEEE Computer Society*, 886–893
7. Dutta JK, Banerjee B, Bonet B, Koenig S (2015) Online detection of abnormal events using incremental coding length. In: Bonet B, Koenig S (eds) Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25–30, 2015, Austin, Texas, USA. AAAI Press, pp 3755–3761
8. Farneback G, Bigün J, Gustavsson T (2003) Two-frame motion estimation based on polynomial expansion. In: Bigün J, Gustavsson T (eds) Image Analysis, 13th Scandinavian Conference, SCIA 2003, Halmstad, Sweden, June 29 – July 2, 2003, Proceedings, vol 2749. Lecture Notes in Computer Science. Springer, pp 363–370

9. Gandhi T, Trivedi MM (2007) Pedestrian protection systems: issues, survey, and challenges. *IEEE Trans Intell Transp Syst* 8(3):413–430
10. Giorno AD, Bagnell JA, Hebert M (2016) A discriminative framework for anomaly detection in large videos. In: Leibe B, Matas J, Sebe N et al (eds) *Computer Vision - ECCV 2016–14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part V*, vol 9909. *Lecture Notes in Computer Science*. Springer, pp 334–349
11. Hasan M, Choi J, Neumann J, Roy-Chowdhury AK, Davis LS (2016) Learning temporal regularity in video sequences. *IEEE Computer Society*, 733–742
12. He C, Shao J, Sun J (2018) An anomaly-introduced learning method for abnormal event detection. *Multim Tools Appl* 77(22):29573–29588
13. Hu W et al (2006) A system for learning statistical motion patterns. *IEEE Trans Pattern Anal Mach Intell* 28(9):1450–1464
14. Hu X, Huang Y, Gao X, Luo L, Duan Q (2019) Squirrel-cage local binary pattern and its application in video anomaly detection. *IEEE Trans Inf Forensics Secur* 14(4):1007–1022
15. Hu Y, Zhang Y, Davis LS (2013) Unsupervised abnormal crowd activity detection using semiparametric scan statistic. In: *IEEE Conference on Computer Vision and Pattern Recognition, CVPR Workshops 2013, Portland, OR, USA, June 23–28, 2013*, pp 767–774
16. Ionescu RT, Smeureanu S, Alexe B, Popescu M (2017) Unmasking the abnormal events in video. 2914–2922
17. Ionescu RT, Khan FS, Georgescu M, Shao L (2019) Object-centric auto-encoders and dummy anomalies for abnormal event detection in video. *Computer Vision Foundation / IEEE*, 7842–7851
18. Johnson N, Hogg DC (1996) Learning the distribution of object trajectories for event recognition. *Image Vis Comput* 14(8):609–615
19. Kim J, Grauman K (2009) Observe locally, infer globally: a space-time MRF for detecting abnormal activities with incremental updates. 2921–2928
20. Kratz L, Nishino K (2009) Anomaly detection in extremely crowded scenes using spatio-temporal motion pattern models. 1446–1453
21. Lazebnik S, Schmid C, Ponce J (2006) Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. *IEEE Computer Society*, 2169–2178
22. Li N, Chang F (2019) Video anomaly detection and localization via multivariate gaussian fully convolution adversarial autoencoder. *Neurocomputing* 369:92–105
23. Li N, Guo H, Xu D, Wu X (2014) Multi-scale analysis of contextual information within spatio-temporal video volumes for anomaly detection. *IEEE*, 2363–2367
24. Li S, Liu C, Yang Y (2018) Anomaly detection based on maximum a posteriori. *Pattern Recognit Lett* 107:91–97
25. Lin T et al (2017) Feature pyramid networks for object detection. *IEEE Computer Society*, 936–944
26. Liu W, Luo W, Lian D, Gao S (2018) Future frame prediction for anomaly detection - a new baseline. *IEEE Computer Society*, 6536–6545
27. Liu W, Luo W, Li Z, Zhao P, Gao S, Kraus S (2019) Margin learning embedded prediction for video anomaly detection with a few anomalies. In: Kraus S (ed) *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10–16, 2019*. ijcai.org, pp 3023–3030
28. Lu C, Shi J, Jia J (2013) Abnormal event detection at 150 FPS in MATLAB. 2720–2727
29. Luo W, Liu W, Gao S (2017a) Remembering history with convolutional LSTM for anomaly detection. *IEEE Computer Society*, 439–444
30. Luo W, Liu W, Gao S (2017b) A revisit of sparse coding based anomaly detection in stacked RNN framework. *IEEE Computer Society*, 341–349
31. Mahadevan V, Li W, Bhalodia V, Vasconcelos N (2010) Anomaly detection in crowded scenes. 1975–1981
32. Medioni GG, Cohen I, Brémond F, Hongeng S, Nevatia R (2001) Event detection and analysis from video streams. *IEEE Trans Pattern Anal Mach Intell* 23(8):873–889
33. Mehran R, Oyama A, Shah M (2009) Abnormal crowd behavior detection using social force model. 935–942
34. Mo X, Monga V, Bala R, Fan Z (2014) Adaptive sparse representations for video anomaly detection. *IEEE Trans Circuits Syst Video Techn* 24(4):631–645
35. Otsu N (1979) A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man and Cybernetics* 1:62–66
36. Nguyen T, Meunier J (2019) Anomaly detection in video sequence with appearance-motion correspondence. *IEEE*, 1273–1283
37. Pang G, Shen C, van den Hengel A, Teredesai A et al (2019) Deep anomaly detection with deviation networks. In: Teredesai A, Kumar V, Li Y, et al (eds) *Proceedings of the 25th ACM SIGKDD International*

- Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4–8, 2019. ACM, pp 353–362
38. Pang G, Yan C, Shen C, van den Hengel A, Bai X (2020) Self-trained deep ordinal regression for end-to-end video anomaly detection. *IEEE*, 12170–12179
  39. Piciarelli C, Micheloni C, Foresti GL (2008) Trajectory-based anomalous event detection. *IEEE Trans Circuits Syst Video Techn* 18(11):1544–1554
  40. Reddy V, Sanderson C, Lovell BC (2011) Improved anomaly detection in crowded scenes via cell-based analysis of foreground speed, size and texture. *IEEE Computer Society*, 55–61
  41. Ronneberger O, Fischer P, Brox T, Navab N, Hornegger J, III WMW, Frangi AF (2015) U-net: convolutional networks for biomedical image segmentation. In: Navab N, Hornegger J, III WMW, Frangi AF (eds) *Medical Image Computing and Computer-Assisted Intervention - MICCAI 2015 - 18th International Conference Munich, Germany, October 5 - 9, 2015, Proceedings, Part III, Lecture Notes in Computer Science*, vol 9351. Springer, pp 234–241
  42. Shi X, et al (2015) Convolutional LSTM network: a machine learning approach for precipitation nowcasting. In: Cortes C, Lawrence ND, Lee DD, Sugiyama M, Garnett R (eds) *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7–12, 2015, Montreal, Quebec, Canada*, pp 802–810
  43. Sultani W, Chen C, Shah M (2018) Real-world anomaly detection in surveillance videos. *IEEE Computer Society*, pp 6479–6488
  44. Tran H, Hogg DC (2017) Anomaly detection using a convolutional winner-take-all autoencoder. In: *British Machine Vision Conference 2017, BMVC 2017, London, UK, September 4–7, 2017*. BMVA Press
  45. Vu H, Nguyen TD, Le T, et al (2019) Robust anomaly detection in videos using multilevel representations. In: *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):5216–5223
  46. Wang T et al (2019) Generative neural networks for anomaly detection in crowded scenes. *IEEE Trans Inf Forensics Secur* 14(5):1390–1399
  47. Xu J, Denman S, Sridharan S, Fookes C, Rana R (2011) Dynamic texture reconstruction from sparse codes for unusual event detection in crowded scenes. *J-MRE '11*, pp 25–30
  48. Yang J, Yu K, Gong Y, Huang TS (2009) Linear spatial pyramid matching using sparse coding for image classification. *IEEE Computer Society*, 1794–1801
  49. Yu B, Liu Y, Sun Q (2017) A content-adaptively sparse reconstruction method for abnormal events detection with low-rank property. *IEEE Trans Syst Man Cybern Syst* 47(4):704–716
  50. Yuan Y, Feng Y, Lu X (2017) Statistical hypothesis detector for abnormal event detection in crowded scenes. *IEEE Trans Cybern* 47(11):3597–3608
  51. Zhang H, Ngo C (2019) A fine granularity object-level representation for event detection and recounting. *IEEE Trans Multim* 21(6):1450–1463
  52. Zhang Y, Lu H, Zhang L, Ruan X (2016) Combining motion and appearance cues for anomaly detection. *Pattern Recogn* 51:443–452
  53. Zhao B, Li F, Xing EP (2011) Online detection of unusual events in videos via dynamic sparse coding. *CVPR 2011, Colorado Springs, CO, USA, 2011*, pp. 3313–3320. <https://doi.org/10.1109/CVPR.2011.5995524>
  54. Zhou JT, Du J, Zhu H et al (2019) AnomalyNet: An anomaly detection network for video surveillance. *IEEE Trans Inf Forensics Secur* 14(10):2537–2550

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.