The Visual Computer (2023) 39:2127–2147
https://doi.org/10.1007/s00371-022-02469-3

ORIGINAL ARTICLE

Motion-compensated online object tracking for activity detection and crowd behavior analysis

Ashish Singh Patel[1] · Ranjana Vyas[2] · O. P. Vyas[2] · Muneendra Ojha[1] · Vivek Tiwari[1]

Accepted: 15 March 2022 / Published online: 13 April 2022

Abstract

It is a nontrivial task to manage crowds in public places and recognize unacceptable behavior (such as violating social distancing norms during the COVID-19 pandemic). In such situations, people should avoid loitering (unnecessary moving out in public places without apparent purpose) and maintain a sufficient physical distance. In this study, a multi-object tracking algorithm has been introduced to improve short-term object occlusion, detection errors, and identity switches. The objects are tracked through bounding box detection and with linear velocity estimation of the object using the Kalman filter frame by frame. The predicted tracks are kept alive for some time, handling the missing detections and short-term object occlusion. ID switches (mainly due to crossing trajectories) are managed by explicitly considering the motion direction of the objects in real time. Furthermore, a novel approach to detect unusual behavior of loitering with a severity level is proposed based on the tracking information. An adaptive algorithm is also proposed to detect physical distance violation based on the object dimensions for the entire length of the track. At last, a mathematical approach to calculate actual physical distance is proposed by using the height of a human as a reference object which adheres more specific distancing norms. The proposed approach is evaluated in traffic and pedestrian movement scenarios. The experimental results demonstrate a significant improvement in the results.

Keywords Object tracking · Pedestrian movement · Activity recognition · Loitering · Physical distancing

1 Introduction

Video surveillance systems are becoming an indispensable part of Smart City infrastructure for ensuring the security and safety of public places. But they are left with human intervention to detect and identify potentially dangerous situations, which is very error-prone and costly [39]. Thus, an automatic surveillance system [63] is required, which can process a massive amount of incoming data and generate meaningful information by isolating the suspicious events for ensuring overall safety. Furthermore, detecting certain activities like loitering and physical distancing violation becomes undoubtedly crucial in the time of natural pandemic like COVID-19.[1] Thus, an intelligent system must identify such behavior in public places to ensure safety and security with law enforcement.

However, detecting such activities requires spatial and temporal information, which consists of the interaction of numerous objects in the same frame and the subsequent temporal frames requiring accurate tracking of the multiple objects.

Multi-object tracking becomes complex in a dense environment where many objects with similar appearances are

✉ Ashish Singh Patel
ashish@iiitnr.edu.in

Ranjana Vyas
ranajana@iiita.ac.in

O. P. Vyas
dropvyas@gmail.com

Muneendra Ojha
muneendra@iiitnr.edu.in

Vivek Tiwari
vivek@iiitnr.edu.in

[1] Department of Computer Science and Engineering, International Institute of Information Technology Naya Raipur, Atal Nagar, India

[2] Department of Information Technology, Indian Institute of Information Technology Allahabad, Prayagraj, India

[1] https://www.who.int/emergencies/diseases/novel-coronavirus-2019.

present, leading to object occlusion (due to self or other multiple nearby objects) and frequent crossing trajectories. In such an environment, most of the tracking algorithm fails to perform accurately in achieving robust multi-object tracking [37]. Moreover, due to the emergence of Deep Neural Network (DNN) models, various automatic object detection approaches like Yolov3 [52], single-shot object detector [36], etc., showed their applicability with high accuracy in real time [64,67]. However, occasionally missed detection, false detection, and multiple detections still occur due to short-term object occlusion, which causes tracks misleading and hampering the activity detection. Most of the work in literature is for pedestrian detectors, using frame by frame high-quality object detectors [13,18], and then associating with online and offline trackers [5,67,69]. The majority of the approaches are used either appearance-based methods [1,7,9,11,66] or Kalman filter [30], and particle filter [29]. These methods fail where objects targets are too close, or object movement is nonlinear. Few approaches use tracking by comparing the trajectories (dynamics similarity) but are too complex.

The presented study is focused on improving the object detection method and wrong association of the track, which normally occurs due to crossing trajectories of multiple objects. Furthermore, we proposed a method for loitering detection and physical distancing detection. The proposed method for finding physical distancing is often sufficient to comply with norms of the COVID-19 pandemic, where the norms are relative to the dimensions of the objects. However, certain scenarios/use-cases may require more specific distancing norms requiring identification of actual physical distance. Thus, we proposed an algorithm to estimate the actual physical distance between adults by using the height of the human as a reference. The major contributions are summarized as follows:

– An algorithm for better localization of an object in the video through a data association approach to estimate object tracks that alleviate ID Sw. Furthermore, the proposed approach handles short-term object occlusion by keeping the estimated track alive for a certain number of frames or until the object is detected again.
– The tracking algorithm to identify suspicious activity ("loitering") in a video and assigned a severity level (high/medium/low) using an online approach.
– A method to identify the physical distancing in the crowd by using reference thresholding.
– A method to estimate actual physical distance using reference objects.

The proposed approach performs well to improve detection errors. It is validated by recognizing the activity of loitering and achieved state-of-the-art results.

The remaining of the paper is organized as follows. The existing literature work is presented in Sect. 2, the multi-object tracking methodology, along with a novel algorithm for identifying loitering and violation of physical distancing in Sect. 3. Experimental results on various datasets and their discussion are presented in Sect. 4. Section 5 concludes the paper by highlighting the possible future directions.

2 Related work

The major focus of the study is online object tracking for crowd behavior analysis. In this regard, literature discussion has been categorized into two segments, first with the tracking algorithms including object detection and, later, crowd behavior analysis (loitering and physical distancing).

2.1 Brief survey on object detection approaches

Plenty of Deep Neural Network (DNN) [12,25,28] architectures were proposed to extract the bounding box and further classification. The object detection methods can be viewed in two ways. The first set of methods requires the generation of region proposals where the probability of locating an object is high. These region proposals of the objects are passed to a neural network that further classifies the object in a predefined class. The R-CNN [25] is the first and state-of-the-art method. Furthermore, many other methods are proposed, which are incremental development over the R-CNN which includes spatial pyramid pooling (SPP-net) [28], region-based fully convolutional network (R-FCN) [12], feature pyramid networks (FPN) [34], Fast R-CNN [24], Faster R-CNN [54], and Mask R-CNN [27]. The second type of detection method extracts the bounding box location of the object and class of the object. The major approached are Multi-Box [17], Deconvolutional Single Shot Detector (DSSD) [23], Attention Net [68], G-CNN [43], Single Shot Multi-Box Detector (SSD) [36], Deeply Supervised Object Detectors (DSOD) [59], Yolo [50], and yolov2 [51]. The YOLOv3 [52] has been used in the proposed work as it is a robust and accurate method for object detection with bounding box extraction.

2.2 Recent works on object tracking

The track of the object and the temporal relations between the objects spanning over multiple frames is widely used for behavior analysis of the crowd at public places ([2,49]). Tu et al. [62] highlighted the inability of CNNs for modeling temporal information, and they proposed an action stage that emphasized spatiotemporal vector created by aggregating local feature descriptors. Basly et al. [4] proposed a Residual Convolutional Network-based deep temporal residual system for recognition day to day activities. They have used LSTM for extracting temporal features from the video. However, detecting unusual and abnormal behavior is still challenging as it requires tremendous training data. Dawn et al. [14] surveyed the various spatiotemporal interest points (STIP)-based techniques for human action recognition which uses spatiotemporal information. Mabrouk et al. [38] proposed a method for violence detection by utilizing global and local features (optical flow and STIP). Ramfrez et al. [20] proposed recognition of group social behavior based on Individual Profiles (IP) to classify as Equally Interested (EI), Balanced Interest (BI), Imbalance Interests (UI), and Chatting (CH). The IP is categorized as Exploring/Interested/Distracted or Disoriented by ranking the features (trajectory, distance, speed, and gaze) using centered kernel alignment.

Feng et al. [19] proposed a unified MOT algorithm by utilizing long-term and short-term cues. It classifies potential user and causer using a Switch Aware Classifier (employ features of the target and switcher), sub-net-based single object tracking by capturing short-term cues, and a re-identification based subnet for extracting long-term cues. Furthermore, many approaches for multi-object tracking are proposed by generating detections and associating them to establish the track hypothesis. However, two approaches are followed for associating detections with the track. First, associate detections frame by frame locally, while others follow a global approach. Wu et al. [65] followed the approach of associating detections with object hypotheses locally by defining an affinity measure based on position, color, and size. Jeany et al. [60] proposed a tracking algorithm by associating detections; using Quad-CNN, which performs the association of objects across subsequent frames by learning quadruplet losses and the appearances of the target object (including their temporal adjacencies). The quadruplet loss adds an extra constraint that forces proximate detections to locate near compared to the target, which has a significantly sizable temporal gap. Bewley et al. [8] proposed a real-time MOT approach by using a robust detector for object detection and then assigning the track based on the Hungarian method and Kalman filter by considering the velocity of the objects. Wojke et al. [64] improved their previous approach [8] by integrating appearance information to handle the object occlusion and reduce the number of identity switches.

Dicle et al. [16] proposed an algorithm that employs the motion dynamics to distinguish targets having a similar appearance, minimize target identification error, use Generalized Linear Assignment (GLA) for identifying dynamics. The approach is independent of track length and powerful to capture the motion dynamics of the target. However, it is computationally costly. Bae et al. [3] discussed an online algorithm for multi-target tracking using discriminative appearance learning. Firstly, the confidence of the track is calculated using continuity and detectability. The track grows incrementally by associating with detections, and linear discriminant analysis selects the target where appearance is similar. It shows track confidence estimation is computationally costly and also highly dependent on the quality of detectors, fails to handle object occlusion and crossing trajectory. The frame-by-frame association suffers from the drift when multiple objects are too close. This problem is addressed by optimizing multiple trajectories in [53], and joint probabilistic data association filters [22] by generating a single state hypothesis for association likelihood. It is further improved in [31,55] tracking and demonstrated substantial results. Leibe et al. [32] used quadratic Boolean programming to perform association globally (processed the entire video as a batch). Zhang et al. [69] introduced network flow formulations to perform multi-target data association. However, these algorithms are not applicable in online real-time scenarios.

To the best of our knowledge, no work has been sighted that uses the direction of the objects to handle crossing trajectories. It is also worth exploring frame interleaving to improve the detections errors. The major focus of the work is the estimation of accurate tracking information for activity recognition to analyze crowd behavior.

2.3 Crowd behavior analysis survey

Anomalous crowd behavior analysis is challenging in intelligent surveillance systems due to the lack of a standard definition of anomaly and limited generalized examples. Chang et al. [10] proposed a technique employing deep autoencoder and k-means clustering to detect anomalies in video. The spatiotemporal information is divided into two sub-processes; one autoencoder for spatial information and the other for temporal information. The consecutive frames are processed on temporal auto-encoder while spatial auto-encoder processes the last individual frame. The optical flow is constructed by utilizing the RGB difference by taking the last frame while the temporal part process the consecutive frames. As a result, abnormal events tend to have distinctive spatial and temporal characteristics that produces significant reconstruction error. Furthermore, deep k-means clusters are designed to extract the common factors of variation on the dataset trained on general events. The anomaly is detected by computing the deviation of the representation with the cluster along with reconstruction error.

Loitering is one of the most common unusual behaviors from the perspective of surveillance systems [39]. Thus, in-time identification of such loitering (anomalous behavior) can mitigate many potential dangerous scenarios. Nam [44] proposed an algorithm for loitering detection by associating tracks in the crowded scene using a histogram of oriented gradients. Consequently, versatile loitering [57] is proposed by Arivazhagan et al. by using wavelet transformation for blob detection and further classifying them as pedestrian tracks using an SVM classifier. However, such methods suffer in the presence of multiple pedestrians in the vicinity. Loitering detection through frame-by-frame tracking is efficient, but it predominantly depends on the tracking accuracy [42]. The misses can occur due to object occlusions and detection errors, especially in crowded scenes. The proposed tracking algorithm can recover the missed tracks and handle object occlusion. Recent works on loitering detection rely on the trajectory and duration of stay of the person, but it is insufficient, as the stay duration depends on various parameters like the environment of the place, number of people in the place, or a visual salience place. In such cases, a person can spend significantly more time than just crossing the place. Thus, the proposed work has given the major focus on loitering detection and further assigned a severity level as low, medium, or high based on the time spent. The initial threshold is assigned and updated online after the end of each track.

2.3.1 Brief survey on physical distance estimation approaches

It is worth to note physical distancing in public places is a must to protect against the various infectious diseases that spread through the air. However, physical distancing violation is common in public places like malls, railway stations, airports, etc. Mercaldo et al. [40] proposed a simple method for physical distancing estimation using object detection with YOLO and Euclidean distance between the centroid of the detected object. However, a predefined threshold is not effective as the dimensions of the object and threshold depend on the position of the camera. Sugianto et al. [61] proposed to identify the physical distancing by performing a homographic transformation. However, the homographic transformation has certain limitations and may not work accurately in crowd surveillance scenarios. Saponara et al. [58] introduced a system for real-time crowd behavior analysis for tackling from COVID-19 pandemic. The method estimates physical distancing using the Euclidean distance between the centroid of the detected bounding box of the objects. However, Euclidean is only effective in very limited planar scenarios. Zuo et al. [70] proposed a method to calculate the real-time distance by finding a “box” with four hyper-planes; two pedestrians belong to two of the hyperplanes, and the other two hyperplanes are perpendicular and intersect with the pedestrians. The distance is estimated by calculating the rate at which the height of the object in pixel changes. However, the different locations usually have different views of perspective and maybe re-positioned at different places. So, it is not always possible to find a box with such conditions. Furthermore, the method is computationally costly to perform in real time. Gupta et al. [26] discussed a novel method to calculate the actual physical distance of each object to the camera by considering the focal length and distance with the bounding box of the objects. Furthermore, the distance between the objects is estimated by the absolute horizontal pixel distance. However, such a method is only effective in planar scenarios where the object size remains the same at all positions, often infeasible in a real-time environment.

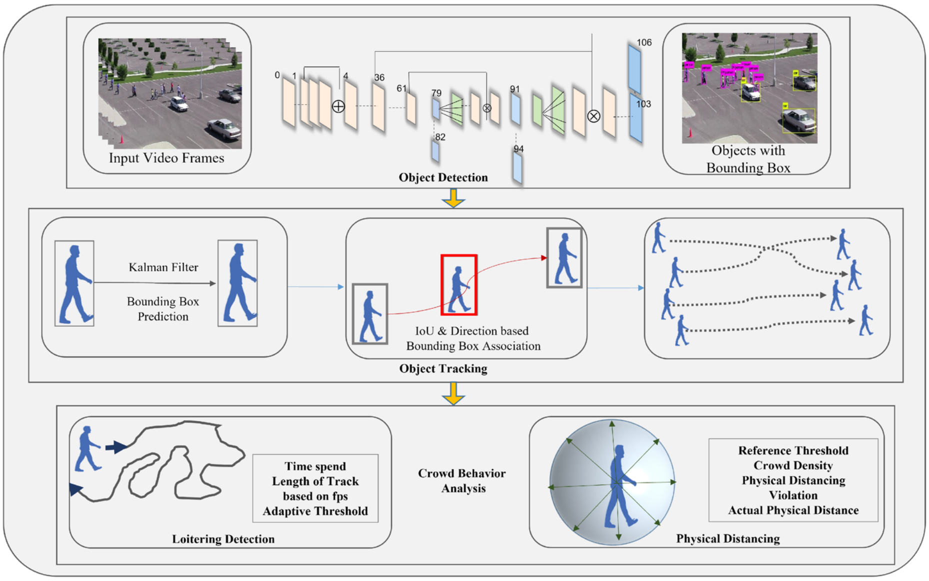
Thus, we propose an algorithm for analyzing crowd behavior by identifying anomalous behavior of loitering and detecting violation of physical distancing by object detection and utilizing the tracking information. The threshold for physical distancing is set adaptively, independent of the viewpoint, and proportionate to the size of the object.

3 Proposed work

The major contribution and methodology have been categorized into four problems: object detection/tracking, loitering classification, adaptive threshold-based, and actual physical distance estimation. Moreover, an adaptive threshold-based method is well suitable where norms are relative to the dimensions of the objects. However, certain scenarios/use-cases may require more specific distancing norms (actual physical distance). Thus, an algorithm to measure the actual physical distance is also proposed. The framework of the proposed tracking-based activity detection and behavior analysis is subdivided into three parts: object detection, object tracking, and crowd behavior analysis as depicted in Fig. 1. In the first part of the figure, the object detection algorithm (YOLOv3 [52]) details are presented by including the layered model. Further, the information on the tracking algorithm is presented, which involves the prediction step followed by data association. The tracking information is additionally utilized for crowd behavior analysis to identify the activity of loitering and physical distancing with actual physical distance estimation presented in the third part. The key features and parameters employed in crowd behavior analysis are also given under each step. The key feature contains the adaptive threshold techniques for assigning a severity level and adaptive threshold for physical distancing based on the size of the bounding box with the actual physical distance by taking the height of the human as a reference in the case of adults.

Figure 2 demonstrates a better understanding of the proposed object tracking approach. Firstly, the objects in each frame of the videos are identified using a detection algorithm. The tracking information of the objects is generated by combining the predictions (motion compensation) and a data association algorithm. The proposed motion estimation approach can handle short-term object occlusion, including detection error (missing, double, and false detections). The identity switches (due to crossing trajectories) are handled by including the direction of the motion. The proposed tracking

Fig. 1 A proposed framework for activity detection and crowd behavior analysis



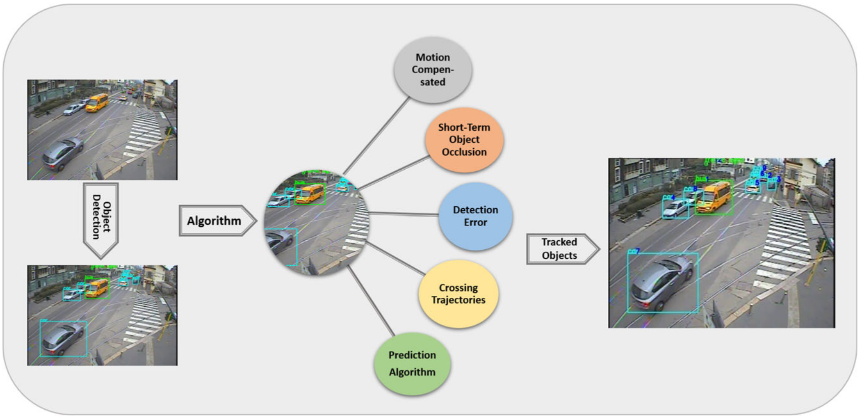
approach consists mainly of object detection, track prediction, and association of detections with tracks. However, the physical distance estimation approach has been discussed separately at last.

3.1 Object detection

An object is first identified and then extracted along with its bounding box using the DNN-based object detection method. YOLOv3 [52] was employed for object detection, which is a Fast RCNN [24] based framework, pre-trained to detect 80 classes of Microsoft Coco dataset [35]. It is worth mentioning that various similar methods were tested in this scenario and found YOLOv3 is outperformed. However, there may have a chance to see performance variation with respect to other datasets and scenarios. The model worked well to detect the car, person, and bus but was found less effective for the truck. In this regard, a separate training has been carried out explicitly with 5000 images of the truck (learning rate was set to 0.02). The batch size was set to 100 with ten epochs and 50 iterations per epoch (500 iterations in total). Furthermore, few MOT benchmark datasets have public detections, consisting of pre-existing detections for benchmarking the results. However, object detection methods have some limitations and may not perform accurately due to detection errors, which are summarized as below:

- *Missed Detection*—When an object in an image goes undetected. It usually happens due to lack of training or object occlusion.
- *Small Detection*—A small portion of the object (small bounding box) is detected along with the full part of the object due to object occlusion, lack of clear visibility, or poor image resolution.
- *Large Detection*—A bigger bounding box is also generated when two objects are close by along with individual detections.

Fig. 2 Object tracking approach workflow



- *Double Detection*—Two bounding box of the object is generated; one of each type may occur due to misclassification.

The proposed tracking algorithm handles such errors by employing frame interleaving.

3.2 Object tracking

The tracking algorithm is based on the Kalman filter [30] which can handle the continuous variable for estimating state variables from noisy observations over time. For example, looking for a bird movement so that we do not lose its sight. Kalman filter uses linear Gaussian distributions, such that the subsequence state is a linear function of the present state with some Gaussian noise. The algorithms predict the state variables comprising the width, height, and location of the object of the next state based on the previous states. The new state of the object is used in the data association algorithm. It consists of three steps: predict, data association, and update. The new trajectory is initialized in a case where detection of the next frame does not match with existing trajectories. The old trajectories are terminated only when associations do not occur with new objects for a certain number of frames (frame interleaving). The pixel coordinate of the center of the object in the current frame is calculated using an extracted bounding box, denoted by (x_t and y_t). The predicted and actual coordinate of the object in the subsequent frame is denoted by $\hat{x}$, $\hat{y}$,, x_{t+1}, and y_{t+1}. The track of the object is assigned with an identifier denoted by track_id (referred to as tracks). Prediction of the object in the next frame is performed using Kalman filter-based estimator. It has a horizontal and a vertical component denoted by x_δ and y_δ . The bounding box of the object is represented with left, top, width and height of the object (denoted by o_l, o_t, o_w and o_h). The horizontal and vertical coordinates of the center of the object are given by o_x and o_y. The state equation of Kalman filter is given by Eq. (1) of m components. Each state has four components, width, height, horizontal, and vertical pixel positions. After each iteration, we get a new set of q number of observations denoted by y containing quantities denoted by $h(x)$. The $h(x)$ depend non-linearly on the state, with observation errors ý having a square covariance matrix (C^o) of size qxq. The initial estimate of the model state X, matrix of error, and covariances C between the estimated state variables are measured by multiple iterations of the algorithm. It will have blended in enough observations to make it reasonably insensitive to have precise initialization details.

$$X = [o_x, o_y, o_w, o_h]^T \tag{1}$$

3.2.1 Prediction

The state variables of the object in the subsequent frame are predicted by the estimator function as shown in Eq. (2) and (3). X_n^p and C_n^p denote the estimated state and covariance matrix at the nth frame, while $\hat{X}_{n-1}$ and $\hat{C}_{n-1}$ represent the state variable and covariance matrix of the previous frame. F_{n-1} matrix is of dimension mxm, calculated as Jacobian matrix of the non-linear function f by linearizing the model which is used to predict the state n_{th} using the previously estimated state ($n-1$) state. Thus, the bounding box of the object is estimated using the predicted state variables.

$$X_n^p = f\left(\hat{X}_{n-1}\right) \tag{2}$$

$$C_n^p = F_{n-1}\hat{C}_{n-1}F_{n-1}^T \tag{3}$$

3.2.2 Direction and IoU based data association

The existing tracks are associated with the detections in the next frame by calculating intersection over union (IoU) with each object and matching the direction of the track. Each track is assigned with a direction as shown in Table 1. D_x and D_y represent the difference between horizontal pixel and vertical pixel of the object in the current frame and previous frame. The direction is calculated by obtaining the difference (D_x and D_y) followed by the assignment of direction based on Table 1. The direction of the track is only updated when there exists a unique match based on the IoU matrix. In the case of multiple matches, the direction is calculated by associating the track with both the detections. Finally, a track where direction remains the same is selected for the object. Consequently, a new *track id* is assigned to the detected object If an association is not successful with any detections. The threshold value for IoU is denoted by IoU_{th}. The associations are successful if Eq. (4) is satisfied. The association algorithm is demonstrated in Fig. 3 which works based on the following cases:

1. **Unique match** - In case of only one match on IoU greater than IoU_{th}, then the object is included in the track, and the direction of the track is updated as in Table 1.
2. **Multiple matches with no direction update** - In the case when there exists more than one match on IoU greater than IoU_{th}, then the direction is calculated by associating the object with each track in from the list of tracks in which IoU is greater than IoU_{th}. The direction of the track is updated based on Table 1. In this case, one or more tracks exist when the updated directions remain the same after associating the object with the tracks. If there is only one track where the direction is not changed, that track is selected; if multiple such tracks are found, then the track that resulted in maximum IoU is selected. This

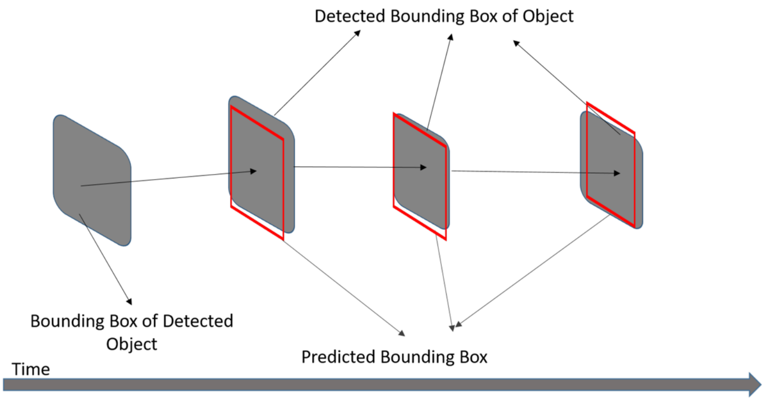

Detected Bounding Box of Object
Bounding Box of Detected Object
Predicted Bounding Box
Time

Fig. 3 Association of objects based on IoU threshold

scenario corresponds to the possibility of crossing trajectories. However, in the case of multiple tracks with no change in direction, it may occur when tracks are moving very close with almost in a similar direction.

3. **Multiple matches with direction update** - When there exists more than one match on IoU greater than IoU_{th}, then the direction is calculated by associating all the matched tracks, and the direction of the track is updated as in Table 1. However, for all associations, the direction is updated, which leads to a change in the direction of the track. In such cases, the track which results in maximum IoU is selected.
4. **No Match** - A new track is created with a track_id and direction id is set to 0.

$$IoU_n < IoU_{th} \tag{4}$$

IOU_{th} and threshold for substantial distance is set to 0.6 and 5 (obtained experimentally for all scenarios).

The inclusion of direction while assigning tracks to detections reduces the number of ID Sw due to more than one object being too close, dominant in case of crossing trajectories, shown in Fig. 4.

The direction of each track is calculated based on the rules below also listed in Table 1.

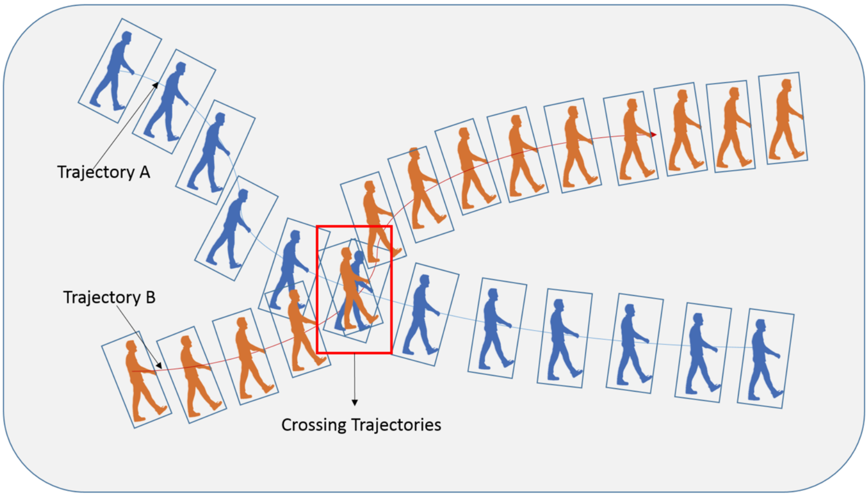
1. ***North East***: If D_x and D_y are positive and magnitude is substantially high then this direction is assigned with id 1.
2. ***East***: If D_x is positive and only D_x have substantial magnitude, then this direction is assigned with id 2.
3. ***South East***: If D_x is positive while D_y is negative, and both have substantially high magnitude then this direction is assigned with id 3.
4. ***North***: If D_y is positive and only D_y have substantial magnitude, then this direction is assigned with id 4.

Table 1 Direction assignment of the track

D_x		D_y		Direction	
Sign	Substantial	Sign	Substantial	Name	Id
+	Yes	+	Yes	NorthEast	1
+	Yes	+	No	East	2
+	Yes	−	Yes	SouthEast	3
+	Yes	−	No	East	2
+	No	+	Yes	North	4
+	No	+	No	No change	previous
+	No	−	Yes	South	5
+	No	−	No	No change	previous
−	Yes	+	Yes	North West	6
−	Yes	+	No	West	7
−	Yes	−	Yes	SouthWest	8
−	Yes	−	No	West	7
−	No	+	Yes	North	4
−	No	+	No	No Change	previous
−	No	−	Yes	South	5
−	No	−	No	No Change	previous

5. ***South***: If D_y is negative and only D_y have substantial magnitude, then this direction is assigned with id 5.
6. ***North Wes***t: If D_x is negative while D_y is positive, and their magnitude is substantially high then this direction is assigned with id 6.
7. ***West***:If D_x is negative and only D_x have substantial magnitude, then this direction is assigned with id 7.
8. ***South West***: If D_x and D_y are negative and their magnitude is substantially high, then this direction is assigned with id 8.
9. ***No Change***: If both D_x and D_y have substantially low magnitude, no change in direction is performed.

Fig. 4 Handling crossing trajectories by including direction of motion



3.2.3 Frame interleaving for handling detection errors and object occlusion

It has been observed that the majority of the detection algorithms miss the object in a few frames due to changes in intensity and luminosity. Thus, frame interleaving is proposed to keep the tracks active even direct association is not possible. This is predominant and useful in case of missing detection for a certain number of frames. Furthermore, the missing detections may also occur due to short-term object occlusion. In this regard, the existing track will be kept active till those many frames where the detection error probability is high. The optimal value of the hyper-parameter (γ) is obtained experimentally by repeating the experiment for a certain number of trials. The value of hyperparameter depends on the environment, speed of the vehicle, etc. The proposed optimization of keeping tracks active (frame interleaving) handles short-term object occlusion and missing detections, giving the object more precise tracking.

3.2.4 Estimator update

The velocity estimators are updated in case of successful association with the pre-existing tracks by adjusting the variation between actual and predicted detection using Eqs. (5), (6), and (7). The predictions is combined with actual observations at instant n to update the estimated state X^n and covariance matrix C^n. Here $C^p[mxm]$ is the covariance matrix of the prediction at time instant n, $\hat{C}$ is the covariance matrix of the update, $C^o[qxq]$ is the covariance matrix of the observations, and $K[mxq]$ is the Kalman gain matrix. The measurement matrix $H_n[qxm]$ relates the prediction uncertainty $\acute{y}_p$ in the measurements to the prediction uncertainty $\acute{x}_p$ in the state.

$$\hat{X} = X^p + K(yo - h\left(X^p\right) \tag{5}$$

$$K = C^p H_n^T \left(H_n C^p H_n^T + C^o\right)^{-1} \tag{6}$$

$$\hat{C}_n = (I - K H_n)\, C^p \tag{7}$$

3.3 Evaluation metric

An ideal tracking algorithm should correctly detect all the objects and estimate their position precisely with their tracking information over time by assigning a unique identifier for each tracked object sequence. Furthermore, it should generate only one trajectory per object. Thus, the following parameters are considered to evaluate its effectiveness.

- *False Positive* - The number of tracks which do not associate with the real object.
- *False Negative* - The number of target detections which are not associated with any tracks, but in actual have a valid association. It also referred as missed targets.
- *ID Sw* [33] - The number of identity switches for all the objects by counting the total number of association of the same object with different tracks.
- *IDF1* [56] - The ratio of correctly identified detections with the average of ground truth and computed detections.
- *MOTA* [44] - Multiple Object Tracking Accuracy (MOTA) is the measure combining false-positives, missed targets, and the number of identity switches given by (8), m_t, fp_t, g_t, and mme_t are the number of misses, of false positives, ground truth, and mismatches, respectively, with time t.

$$MOTA = 1 - \frac{\sum_t (m_t + fp_t + mme_t)}{\sum_t g_t} \tag{8}$$

- *MT* - Mostly tracked targets. It is the ratio of ground truth trajectories covering more than 80%.
- *MOTP* [6] - Multiple Object Tracking Precision (MOTP) specified in (9), ct is the number of matches over the time

period t, and d_t^i represents the distance between the object and its corresponding hypothesis.

$$MOTP = \frac{\sum_{i,t} d_t^i}{\sum_t c_t} \tag{9}$$

3.4 Enhanced loitering classification through dynamic thresholding

A time-stamp-based approach has been introduced to identify loitering. The time spent by a person is calculated using the function of the length of track and frames per second of the captured frame, given by Eq. (10). T_o refers to the time spent, and the number of frames per second is denoted by fps. The threshold parameters are calculated for initial n sample tracks only (training phase of threshold parameters). The value of n controls the parameter for detecting the activity of loitering.

$$T_o = \frac{L_t}{fps} \tag{10}$$

The initial value of the threshold is set to 10, 20, and 30 seconds for low, medium, and high severity levels, respectively. The threshold for severity is updated after each iteration by calculating the deviation from the time of stay of a current person with the average time of stay. Mean is given by μ and calculated using (11) and updated after every iteration.

$$\mu_{n+1} = \mu_n + \frac{T_o}{n} \tag{11}$$

Thus, threshold values are denoted by Th_L, Th_M, and Th_H, given by (12), (13), and (14) for low, medium, and high level of severity, respectively.

$$Th_L = max\ (2\mu_{n+1}, 10) \tag{12}$$

$$Th_M = max\ (3\mu_{n+1}, 20) \tag{13}$$

$$Th_H = max\ (4\mu_{n+1}, 30) \tag{14}$$

The thresholds are measured experimentally and very reasonable for a given environment as it depends on the basic principle of unusual behavior.

3.5 Adaptive threshold based physical distancing detection

Monitoring public places to ensure physical distancing in a real-time environment is considered one of the most challenging tasks. It comes with an enormous challenge as estimating physical distance is almost infeasible without a reference object. Furthermore, the size of the object varies with pixel location as the viewpoint of the camera changes. However, the dimension of the object is correlated in close proximity (a small region of interest) having similar dimensions. The region of interest is defined as a small area where the dimension of a type of object is relatively similar. Thus, our proposed algorithm exploits this property and utilizes the dimensions of the target object to identify if physical distancing. We propose an algorithm to detect the physical distancing violation both spatially and temporally. The algorithm identifies physical distance by calculating the physical threshold as a function of the object's size, as demonstrated in Fig. 5. The visualization and idea of the algorithm are shown in Fig. 6. Two possible scenarios are demonstrated with different viewpoints. It is evident that the number of pixels required by a reference object varies according to the angle, which is captured by the variation in objects' size. The description of the algorithm 1 is as follows:

Algorithm 1: Physical Distancing Detection

```
Input: Bounding box of detected person along with the frame id
       and track id
Output: Set the Physical Distancing parameter of objects and
        tracks
for $f \leftarrow 1$ to $F$ do
    for $i \leftarrow 1$ to $n$ do
        $o_{ix} = o_{ileft} + \frac{o_{iright} - o_{ileft}}{2}$;
        $o_{iy} = o_{itop} + \frac{o_{ibottom} - o_{itop}}{2}$;
        $o_{iwidth} = o_{iright} - o_{ileft}$;
        $o_{iheight} = o_{ibottom} - o_{itop}$;
        $Th_{pd} = o_{iwidth} + o_{iheight}$;
        for $j \leftarrow 1$ to $n$ do
            if $(i! = j) \&\& (o_{ipd} == 0)$ then
                $o_{jx} = o_{jleft} + \frac{o_{jright} - o_{jleft}}{2}$;
                $o_{jy} = o_{jtop} + \frac{o_{jbottom} - o_{jtop}}{2}$;
                $ED_{ij} = \sqrt{(o_{jx} - o_{ix})^2 + (o_{jy} - o_{iy})^2}$;
                if $ED_{ij} <= Th_{pd}$ then
                    $o_{ipd} = 1$;
                    $break$;
                else
                    $o_{ipd} = 0$;
for $t \leftarrow 1$ to $T$ do
    for $i \leftarrow 1$ to $n$ do
        if $o_{ipd} == 1$ then
            $track_{pd} = 1$ ;
            $break$ ;
```

- The set of frames of a video scene with tracked objects along with its bounding box are given as input.
- The flag is set for the objects which do not follow the social distancing as per the algorithm 1.
- The threshold for physical distancing is adaptive based on the width and height of the object. The objects which are near from camera have a larger size as compared

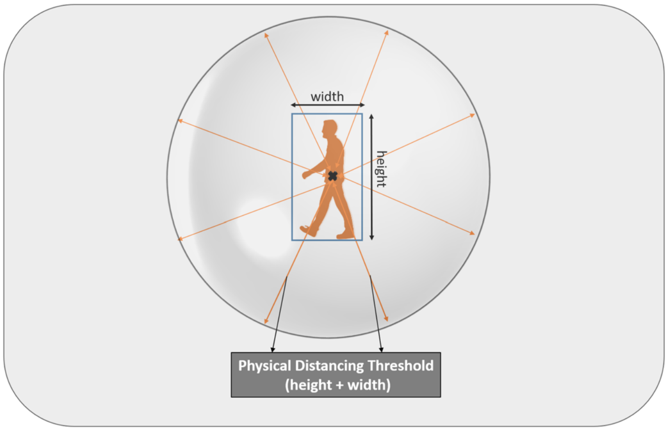


Fig. 5 Demonstration of adaptive physical distancing threshold based on the dimension of the object

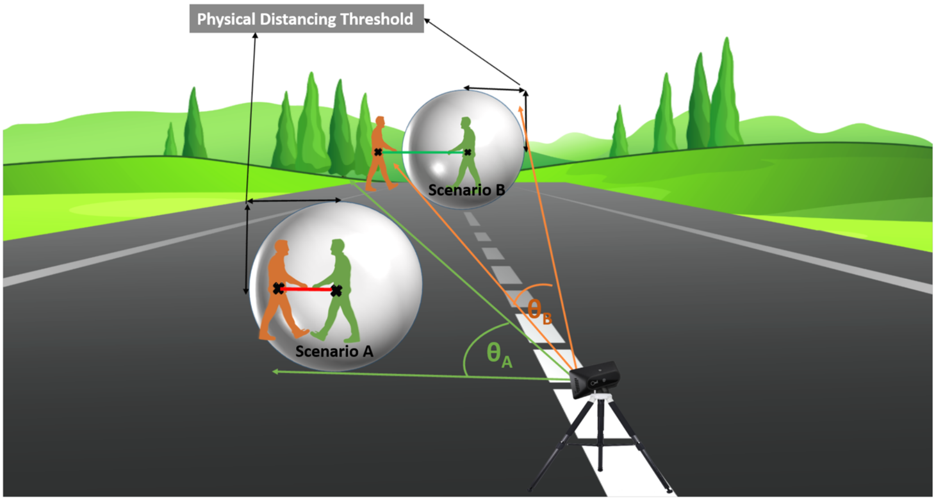


Fig. 6 Demonstration of adaptive physical distancing algorithm in two possible scenario

to farther objects due to changes in camera viewpoint. Thus, a common threshold cannot fit all the objects in a frame. However, the size of the person is proportional to the width and height of the object, irrespective of the location in the frame.

- The violation in physical distancing is calculated by finding the Euclidean distance with all other objects denoted by ED_{ij} and comparing it with the adaptive threshold based on the size of the object.
- The threshold is kept as the sum of height and width of the person as the height of the person is around 5–6 feet, and width is 3 feet, sufficient and adaptive based on the location of the objects in the image.
- Violation of physical distancing at temporal level is identified by identifying the physical distancing flag of all the objects included in the track. If even a single object in a track has a flag as set, the track is classified as violated physical distancing.
- Total number of objects present in the frame is also calculated to identify the overall level of crowding of the place.
- The objects that violate the physical distancing have the value of flag as one, while for others that follow physical distancing, the flag value is zero.

The tracking id of an object is represented by t and goes up to T. The total number of frames is F, and a frame is represented using f. The number of objects in a frame is n and given by O. The pixel value of the center of the object is given by o_x and o_y. The width and height of the object are given by o_{width} and o_{height}. The bounding box of the object is represented by o_{left}, o_{top}, o_{right}, and o_{bottom}. The flag representing the physical distancing of an object and track is denoted by o_{pd} and $track_{pd}$.

3.6 Actual physical distance estimation

The proposed method for finding physical distancing can identify the activity to safeguard from COVID-19 pandemic where the norms are relative to the dimensions of the objects. However, certain scenarios/use-cases may require more specific distancing norms requiring identification of actual physical distance. Thus, we propose to calculate the actual physical distance between person i and j is calculated using Eq. (15) denoted by PD_{ij} is a function of the average height of the human being, the height of the person in the frame, and the Euclidean distance between them, calculated in meters.

At each pixel, the number of pixels contained by the reference object varies depending on the viewpoint angle. As the object goes farther from the viewpoint, the viewpoint angle and number of pixels contained by the object also decreases; thus, the size of the reference object is directly proportional to the viewing angle. This is evident in Fig. 6, in scenario B, the viewing angle is less than the viewpoint angle in scenario A. Thus, the proposed algorithm is adaptive and calculates the actual physical distance at each pixel change based on Eq. (15). The average height of the person (adults) acting as a reference from the real world is a constant and kept as 1.65 m [45] for this experiment which is consistent with our dataset as well. However, the bounding box height is a little larger than the actual object. Thus we have kept the height of the bounding box to 1.8 m as a reference for our calculation.

In the case of another type of object, such as a car that has similar dimensions, the actual physical distance can be estimated using its dimension as a reference. Furthermore, in the case of the objects whose real-world dimensions are not known, then a normalized physical distance can be calculated by using the height of the object in one location of the image to the other. This distance can be converted to real-world metrics like (meters, inches) when height in the real world is established. One such scenario is a school environment where the height of the children is less and has variation but has a common height in a group (a particular class/age group). In such cases, the normalized physical distance is estimated as the children belonging to the same group will possess similar height.

The proposed algorithm is adaptive and calculates the actual physical distance between two persons, given by Eq. (15). The Euclidean distance between the center of the objects is calculated at each pixel value, by varying the value of y by 1 after each instant keeping Δy to 1 and calculating the corresponding value of x at that position using Eq. (16), where Δx refers to a change in the values of x after changing the y by 1. This is repeated from vertical coordinate of object i to vertical coordinate of object j (o_{iy} to o_{jy}) to get the distance from object i to object j. The distance after each iteration is normalized with the ratio of half of the human height and the number of pixels denoting half of the height at that instant as the upper half of the objects is only considered from the center.

Equation (16) is derived from the equation of the line between two points, joining the center of the i_{th} and j_{th} object, m is the slope between two points o_{ix}, o_{iy}, and o_{jx}, o_{jy} given by Eq. (17), and c is a constant. The constant c in Eq. (16) is calculated by assigning x with o_{ix}, y with o_{iy} and m from Eq. (17) in Eq. (16). The vertical point of the bounding box of the person after each update is denoted by y_t, which is calculated by putting the value of x from Eq. (16) in line joining the coordinated of top of the i_{th} object and j_{th} object. The algorithm is demonstrated in Fig. 7. The ratio of human height and number of pixels spanned by the height at that instant normalizes the distance after each iteration, multiplied by two as only one-half of the object is considered.

$$PD_{ij} = \sum_{y=o_{iy}}^{o_{jy}} \frac{\sqrt{\Delta x^2 + \Delta y^2} * 1.8}{(y - y_t) * 2} \tag{15}$$

$$x = \frac{y - c}{m} \tag{16}$$

$$m = \frac{o_{iy} - o_{jy}}{o_{ix} - o_{jx}} \tag{17}$$

Thus, our approach finds the actual physical distance between the people using the height of the person as a reference.

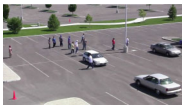
4 Results and discussion

The proposed approach is evaluated on different datasets in various scenarios of the public domain along with the specifically tailored PETS2006 and PETS2016 datasets for performance evaluation for tracking and surveillance. The effectiveness of the object tracking approach is demonstrated by detecting the activity of loitering, which uses the tracking information as a baseline. Thus, the proposed approach is evaluated in three basic steps from the perspective of activity detection; object detection, object tracking, and loitering detection. A comparison with current methods is performed on the benchmark dataset, demonstrating the effectiveness of the proposed approach. Furthermore, the performance of the physical distancing algorithm is demonstrated on various benchmark datasets.

4.1 Dataset

In this work, various datasets are used as listed in Table 2 consisting of traffic environment, warehouse environment, crowd scene, and parking environment for demonstrating the applicability of the proposed approach in broad scenarios. The PETS dataset is also used, a benchmark dataset for performance evaluation of object tracking, and consists of various activities such as loitering. Furthermore, the proposed tracking and physical distancing approach including the estimation of actual physical distance is validated on MOT benchmark datasets [41] (PETS09-S2L2, ETH Crossing, AVG-TownCentre, TUD-Stadmitte, and Venice-2)

Table 2 Detail description of various datasets used in this paper

Dataset	Description	Snapshot
IIIT-NR Parking Lot (D1) [46]	Various scenarios of crowd movement nearby parked car containing trimmed activities	
Warehouse Dataset [47] (D2)	Contains the CCTV surveillance data recorded from a warehouse of Chhattisgarh state storage center, comprising the movement of transport vehicles.	
PNNL2 [68] (D3)	Small sequence of parking lot scenarios published by University of Florida consisting of movement of crowd around parked cars for benchmarking multi object tracking.	
Square Dataset (D4)	It contains the traffic movement of the cars in a traffic environment of a city captured in near a square.	
PETS-2006 [21] (D5) (sequence S1-T1-C(S1), S2-T3-C (S2) and S3-T7-A (S3))	The dataset consists of a left luggage scenario with varied complexity consisting of the multiple moving person in public places. Each sequence contains a calibrated scenario of person(s) with abandoned luggage and loitering.	
PETS-2016 [48] (D6) (sequence 03_06 and 14_05)	A multi-camera dataset contains various scenarios around a parked vehicle. In total 22 different scenarios of various abnormal activities were recorded to identify various potential threats. However, in this work we have used only a part of dataset with sequences containing the activity of loitering.	

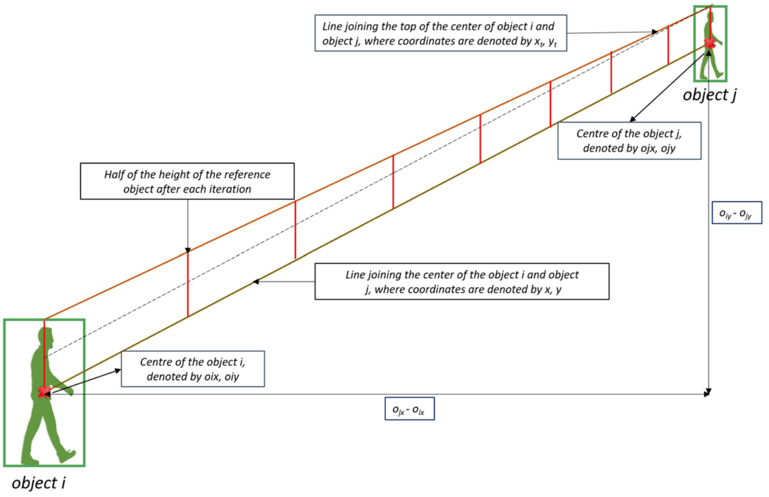

Line joining the top of the center of object i and object j, where coordinates are denoted by x_t, y_t
object j
Centre of the object j, denoted by ojx, ojy
Half of the height of the reference object after each iteration
o_{iy} - o_{jy}
Line joining the center of the object i and object j, where coordinates are denoted by x, y
Centre of the object i, denoted by oix, oiy
o_{jx} - o_{ix}
object i

Fig. 7 Demonstration of estimation of actual physical distance by using reference object (person) between two persons

demonstrating the effectiveness of the approach. The MOT benchmark datasets already have object detections, thus isolating the bias on object detection method for measuring tracking performance.

4.2 Object detection

The objects are detected on each image of the video frame using Yolov3 [52], which detects the objects along with its bounding box. The proposed model can detect and extract bounding box of vehicles (car, bus, truck including rear, front, and side) and person. The effectiveness of the detection algorithm is demonstrated by calculating precision and recall as shown in Table 3.

- Dataset D1 does not contain truck and bus.
- Dataset D2 does not contain any car and bus.
- Dataset D3 does not contain any truck and bus
- Dataset D4 does not contain any truck.
- Dataset D5 and D6 contain person only.

Trained model performs well over different dataset covering multiple scenarios and domain. The trained model is capable of detecting objects to generate tracking information.

4.3 Tracking

The detections of each video sequence frame are passed as the input to the object tracking framework. The detections are associated with a track; subsequently, the algorithm adds the tracking id. The benchmark parameter is calculated using motmetrics[2] which is a python library.

Table 3 Precision and Recall of detections on various datasets

Dataset	Metric (%)	Truck	Car	Person	Bus
D1	Precision	–	98.5	97.3	–
	Recall	–	97.6	96.1	–
D2	Precision	96.7	–	97.7	–
	Recall	97.6	–	97.7	–
D3	Precision	–	98.5	98.1	–
	Recall	–	97.9	97.4	–
D4	Precision	–	98.2	97.7	98.4
	Recall	–	98.3	97.5	92.4
D5	Precision	–	–	100	–
	Recall	–	–	100	–
D6	Precision	–	–	100	–
	Recall	–	–	100	–

Table 4 Evaluation of the approach on benchmark parameters

Dataset	Evaluation Metric			
	MOTA	MOTP	MT	IDSw
D1	80.7	87.2	93.4	6
D2	98.3	95.6	94.1	11
D3	91.4	61	92.5	4
D4	94.6	92.3	93.7	12

The proposed object tracking approach achieves high accuracy on benchmark parameters on different datasets. The tracking results on sequences of PETS2006 and PETS2016 dataset is entirely accurate, laying the strong foundation for activity detection. The results of the proposed tracking approach are demonstrated in Table 4.

A comparison is also performed with the recent object tracking approaches on the dataset D3 (PNNL2) in Table 6. Our proposed method gives better accuracy and also reduces the total number of ID Sw significantly.

[2] https://github.com/cheind/py-motmetrics.

Table 5 Effect of γ on MOT benchmark parameter over Dataset D4

Hyper parameter (γ)	MOTA	MOTP	MT	IDSw
0	81.4	84.4.2	71.6.4	58
5	98.3	95.6	94.1	11
10	91.4	61	92.5	4
15	94.6	92.3	93.7	12
20	94.6	92.3	93.7	12
25	92.5	91.2	90.3	17

Table 6 Performance of the proposed algorithm on dataset D3 (PNNL2)

Approach	Evaluation Metric			
	MOTA	MOTP	MT	ID Sw
CMOT [3]	80.7	58	85.9	6
IHTLS [16]	78.8	57	87.5	5
GMMCP [15]	87.6	58	90.6	7
SORT [8]	84.6	59	88.3	6
Proposed Work	91.4	61	92.5	3

Table 7 Analysis of the proposed approach on benchmark dataset with public detections

Dataset	Approach	MOTA	IDF1	MT	IDSw
PETS 09 S2L2	SORT [8]	49.6	33.4	38.5	289
	Proposed Work	57.7	44.6	59.2	123
ETH Crossing	SORT [8]	54.2	59.7	30.7	12
	Proposed Work	61.6	63.4	72.3	3

The number of ID Sw is significantly reduced as the proposed approach employs direction and IoU based data association along with the frame interleaving, which is very useful in case of crossing trajectories and detection errors. The hyper-parameter (γ) further enhances the tracking by configuring the frame interleaving. (γ) is set experimentally, keeping the tracks active even in case of no association is possible, primarily due to short-term occlusion and detection errors. The effect of maintaining the tracks active is shown in Table 5 on dataset D4. The D4 dataset has 15 frames per second. The duration is kept in terms of a number of frames with a variation of 5, ranging from 0 to 25. It is evident that the best results are obtained at $\gamma = 20$, found after experimenting through multiple possible values and iterations. Furthermore, the hyper-parameter value depends on the target environment, datasets, and detection method, so it needs to be set experimentally.

Moreover, to demonstrate the effectiveness of the proposed approach on benchmark datasets that have public detections is shown in Table 7. The proposed approach outperforms method [8] producing less number of ID Sw and tracking the objects more accurately. Both the methods are evaluated on the dataset having public detections (detections already provided with the dataset). This way, the dependency of the detections is eliminated as certain detection algorithms may be better than the other, thus avoiding ambiguity in tracking accuracy making difficult to analyze the performance of the tracking algorithm.

4.4 Loitering detection

We have demonstrated the loitering detection with various sequences of PETS 2006 and PETS 2016 dataset as shown in Table 8. The value *Yes* corresponds to *Loiter* while *No* is marked when it not classified as *Loiter*. For each sequence, first of all, objects are detected using YOLOv3 [52]. Subsequently, the track is assigned to detections using our proposed tracking algorithm. The time spent by the person is calculated using Eq. (10), which is further classified as a loiter based on the threshold values, calculated adaptively.

The existing approach is better than the versatile loitering because it worked on GAIT feature. A person can still be a loiter if he walks fast and moves around the same place. In this case, it would not be classified as loiter because detection is based on GAIT feature corresponding to slow walk. Furthermore, the area under trajectory can be less when he still moves to and from along a straight line. Thus, the proposed approach of loitering detection based on the time of stay is superior to other methods of classifying by calculating the shape of the trajectory and GAIT features.

Figures 8 and 9 demonstrate the loitering behavior in the sequence 03–06 and 12–05 with high severity as the time

Table 8 Analysis of loitering detection with versatile loitering [57]

Dataset	Sequence	Actual	Approach [57]	Proposed	Severity
D5	S1	Yes	Yes	Yes	Low
	S2	Yes	Yes	Yes	High
	S3	Yes	Yes	Yes	High
D6	0306 TRK1	Yes	No	Yes	Low
	0306 TRK2	Yes	No	Yes	High
	1405 TRK3	Yes	No	Yes	High

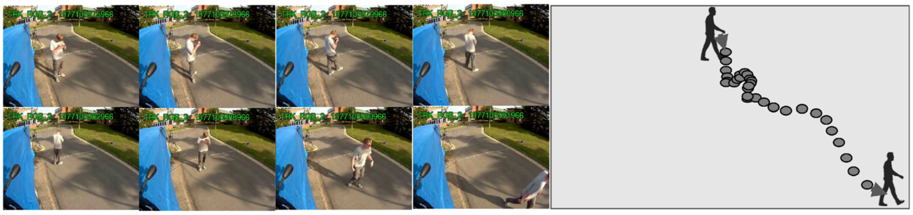


Fig. 8 Demonstration of loitering detection based on the length of the track of the person on PETS 2016 0306 TRK2 dataset

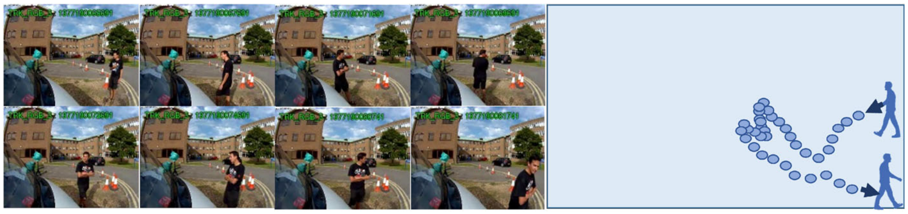


Fig. 9 Demonstration of loitering detection based on the length of the track of the person on PETS 2016 1405 TRK2 dataset

Table 9 Evaluation of the physical distancing on various dataset including MOT 16 benchmark datasets [41]

Dataset	Evaluation Metric			
	Precision	Recall	Track (%)	Avg. Person Count
AVG-TownCentre	92.3	94.6	56.1	11
PETS09-S2L2	71.4	61.4	95.5	30
TUD-Stadtmitte	80.7	87.2	93.4	6
Venice-2	87.6	82.3	93.7	16
D3	97.6	92.3	73.4	12
D5	100	100	66.6	2

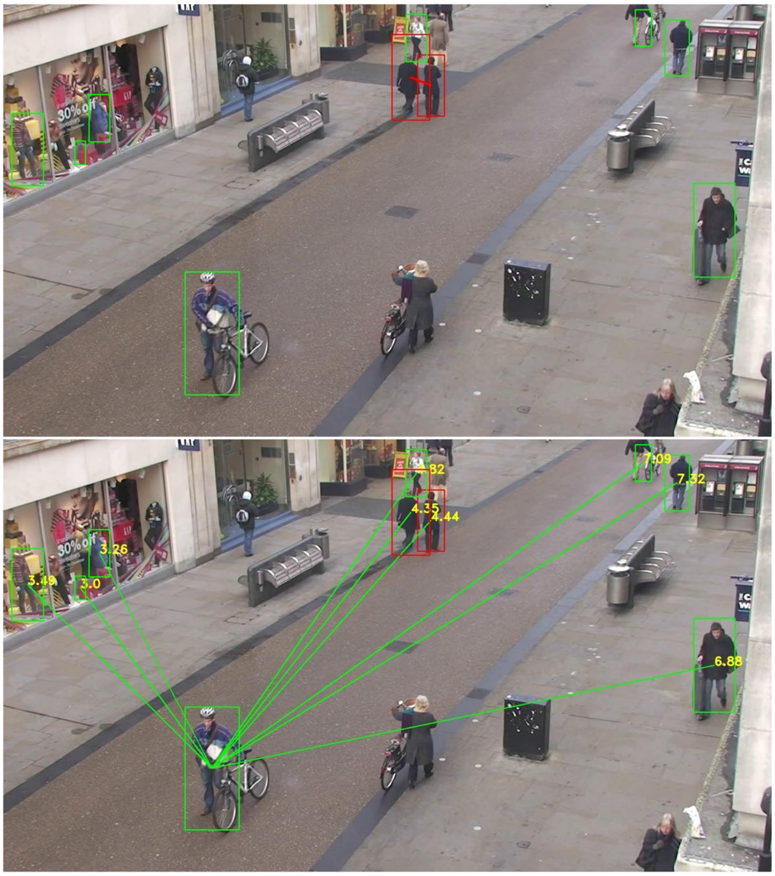
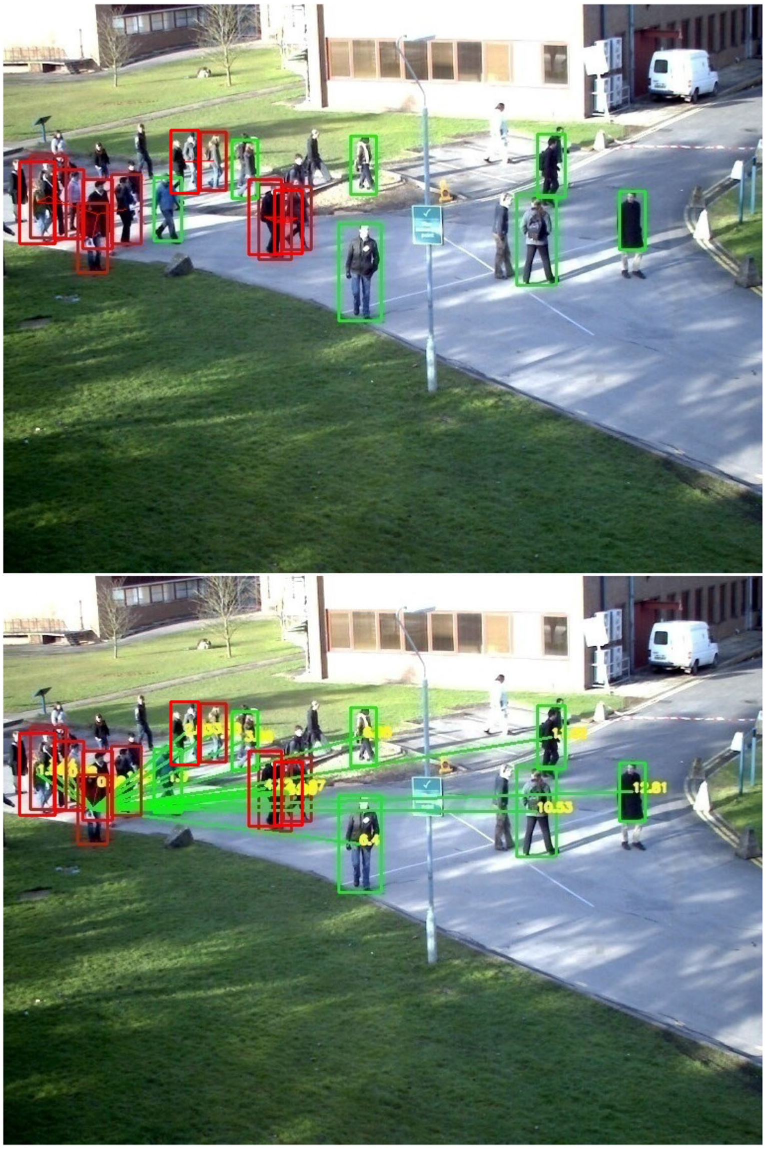


Fig. 10 Demonstration of physical distancing and actual distance on AVG-TownCenter Dataset

Fig. 11 Demonstration of physical distancing and actual distance on PETS09 Dataset



spent by the person is more than the threshold which was set adaptive. In the first part of the figures, the key-frames of the sequence are shown at the interval of two seconds involving loitering. In the second part, the track of the person is plotted. The approximate center position of the person in half a second is shown with a dot.

4.5 Physical distancing

The proposed algorithm for violation of physical distancing detection is evaluated on four benchmark datasets as shown in Table 9. The table presents precision and recall in the spatial direction, i.e., in a single video frame. However, the results significantly depend on the detection accuracy. The percentage of track that has violated physical distancing presents the overall idea about following the norms in a temporal direction. The average count of the person present in a video frame is also listed, giving information about the occupancy of the crowd in a public place.

The dataset D5 only has one sequence with multiple persons, while dataset D3 (PNNL2) has multiple persons.

The physical distancing and actual distance of the various MOT 16 benchmark dataset also listed in Table 9 are shown in Figs. 10, 11, 12, and 13. The first image in the figure shows the status of the person. If the distance is maintained larger than the threshold, the bounding box is labeled green, otherwise red. Furthermore, a physical distancing violation is shown with a red line drawn between the objects.

The second set of the image shows the actual physical distance in meters from one specific object with others. It is assumed that the height of the objects (person in this case) has similar dimensions. Furthermore, while estimating the

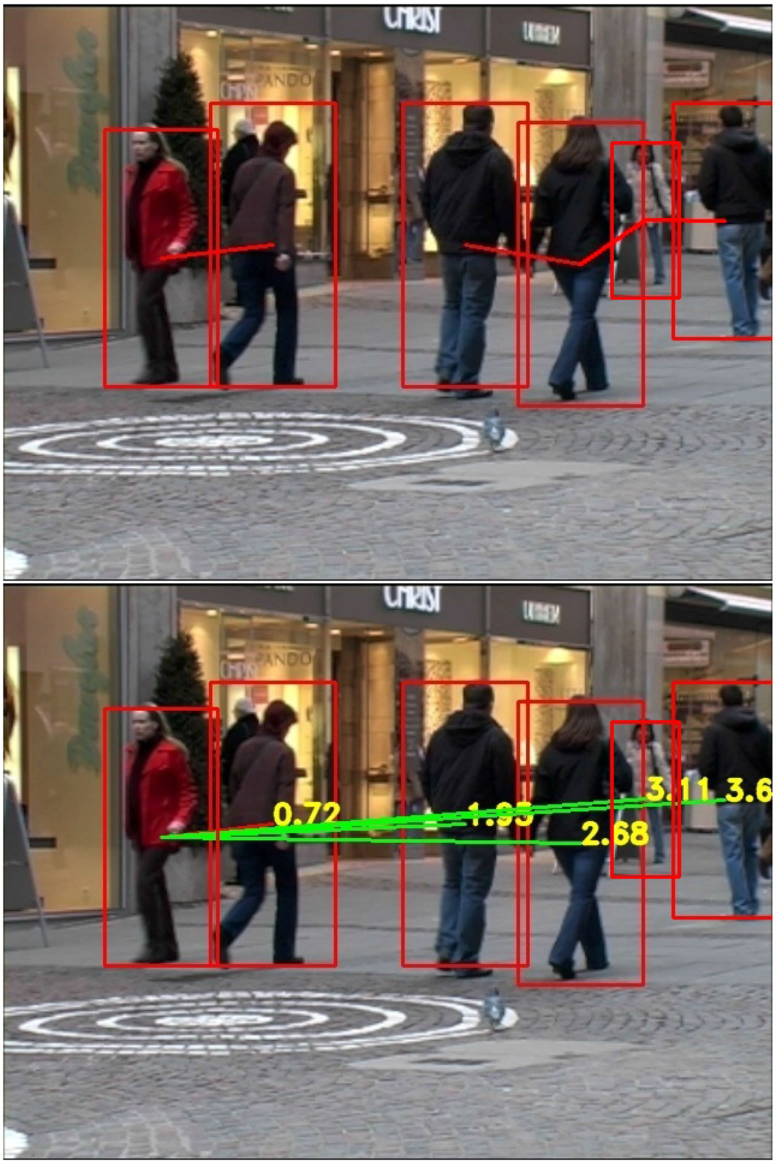


Fig. 12 Demonstration of physical distancing and actual distance on TUD-Stadmitte Dataset

actual physical distance between the objects, the real-world size of the object needs to be known.

In some cases, for identifying the physical distancing when both the target objects and the camera lie on the same line of sight, the distance between the objects will not give accurate results. This type of scenario cannot be solved using a single camera. Thus, multiple cameras need to be installed to get more precise results.

4.6 Runtime

The proposed framework uses detected objects along with a bounding box for generating tracking information and crowd behavior analysis. The objects are detected on an input video frame which is passed to the online tracking algorithm in the pipeline, which assigns a $track\text{-}id$ in real time. The objects with the bounding box and $track\text{-}id$ are further passed to the crowd behavior analysis. The object tracking algorithm works in three parts where prediction thorough Kalman filter have linear bound, data association have polynomial time complexity on $O(t^2)$ and update step requires linear time proportional to the number of active tracks. The loitering detection involves linear computational complexity proportional to the number of active tracks (t) followed by updating threshold for severity level. The physical distancing detection has time complexity with $O(n^2)$ as for each bounding box, it calculates distance with all other existing bounding boxes in a frame. The actual physical distance estimation also has the time complexity of $O(n^2)$ is the distance between two objects are estimated where n is the number of objects.

The whole framework is tested on systems with varied configurations and performs in real time on an HD stream

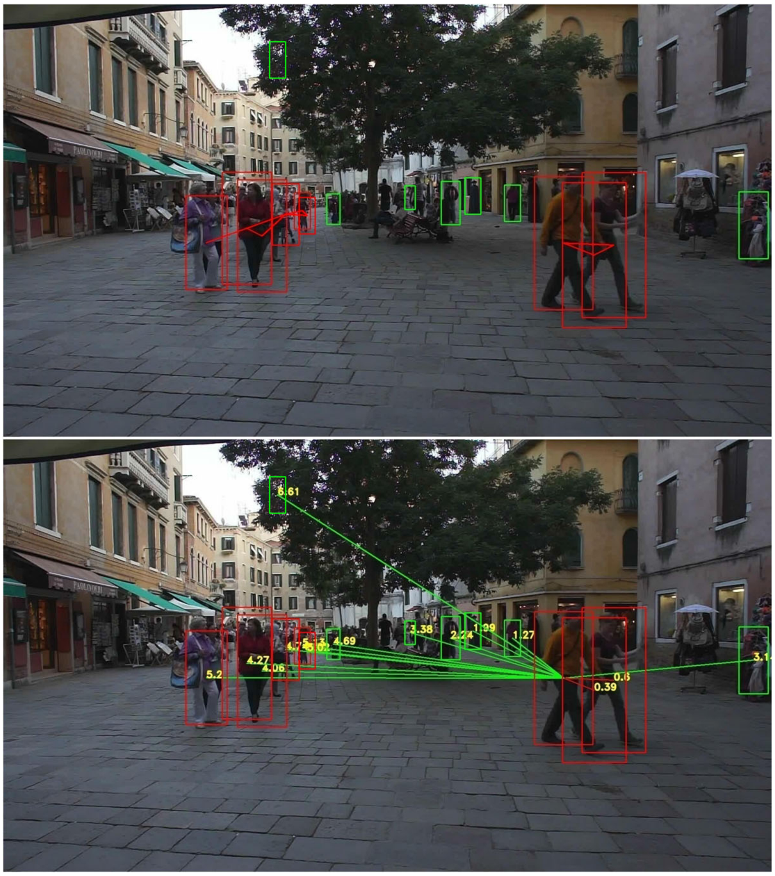


Fig. 13 Demonstration of physical distancing and actual distance on Venice-2 dataset

Table 10 Average runtime of the framework in different configurations for one input frame

CPU	Memory (GB)		Processing time (milliseconds)				
	System	GPU	Object Detection	Tracking	Loitering	Physical Distancing	Actual physical distance
Intel i5 (2.4 Ghz)	8	2	955	3.2	≈1	<1	<1
Intel i7 (2.5 Ghz)	16	4	340	2.4	≈1	<1	<1
Intel Xeon (1.7 Ghz)	32	11	23	≤1	<1	<1	<1

(1280 × 720 resolution) with 30 fps on a system with 3rd generation Intel Xeon processor and NVIDIA Titan 1080 Ti GPU (11GB). Table 10 demonstrates the computation time taken for tracking and crowd behavior analysis along with the object detection. It is to be noted the time taken by object detection significantly depends on the GPU and outperforms with NVIDIA Titan 1080 Ti GPU achieving the performance required for real-time scenarios for HD video. The runtime for tracking and crowd behavior analysis is independent of GPU. It only depends on the CPU and performs reasonably well with 8GB of RAM and 10th generation Intel i5 and i7 processors.

5 Conclusion and future work

In this paper, we have explored the approach for multi-object tracking to handle short-term object occlusion, detection errors, and ID Sw to generate accurate tracks of the objects for analyzing crowd behavior in public places. Firstly, the object bounding box is extracted using a detection algorithm. The bounding box of the objects in the subsequent frame is predicted using a linear velocity estimate of the object based on Kalman filter, which is later on compared and replaced with actual detections in case an association is found. Whenever data association is not feasible, the existing predicted tracks are kept alive for some time, handling the missing detections and short-term object occlusion. ID Sw are handled by explicitly considering the motion direction of the objects at the time of association with the detections, which mainly occurs due to crossing trajectories. The proposed approach is evaluated in traffic environment and pedestrian movement scenarios and achieves high accuracy, which can be explicitly used for event detection. A trajectory-based method to detect the activity of loitering is also proposed capable of classifying the loitering at three levels of suspicion (low, medium, and high). Furthermore, the accuracy of the proposed loitering detection approach demonstrates the fruitfulness of the proposed work. Moreover, an adaptive algorithm for identifying physical distancing is proposed, which utilizes the object detection and tracking information to calculate the actual distance between the persons and the statics about the place if it is too much crowded by counting the average number of persons present at the moment.

The proposed work opens up a plethora of use-cases for detecting usual and unusual activities/events by using the tracking information of the objects in a surveillance environment. In future work, the work can be extended to track multiple types of objects simultaneously and handle long-term object occlusion. The use of context and tracking information can be vital in detecting activities in real-time and complex scenarios. Furthermore, due to the lack of a standard benchmarking dataset with the labeled actual distance between the objects, complete validation will be taken up as future work.

Funding This study does not receive any funding.

Declarations

Conflict of interest All authors have participated in the design and approval of the final manuscript version. This manuscript has not been submitted to, nor is under review at, another journal or other publishing venue. The contents of this manuscript are not copyrighted, submitted, or published elsewhere, while acceptance by the Journal is under consideration. There are no directly related manuscripts or abstracts, published or unpublished, by any authors of this paper. All authors declares that he/she has no conflict of interest.

References

1. Andriyenko, A., Schindler, K., Roth, S.: Discrete-continuous optimization for multi-target tracking. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition, pp. 1926–1933. IEEE (2012)
2. Arroyo, R., Yebes, J.J., Bergasa, L.M., Daza, I.G., Almazán, J.: Expert video-surveillance system for real-time detection of suspicious behaviors in shopping malls. Expert Syst. Appl. **42**(21), 7991–8005 (2015)
3. Bae, S.H., Yoon, K.J.: Robust online multi-object tracking based on tracklet confidence and online discriminative appearance learning. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1218–1225 (2014)
4. Basly, H., Ouarda, W., Sayadi, F.E., Ouni, B., Alimi, A.M.: Dtr-har: deep temporal residual representation for human activity recognition. Vis. Comput. 1–21 (2021)
5. Benfold, B., Reid, I.: Stable multi-target tracking in real-time surveillance video. In: CVPR 2011, pp. 3457–3464. IEEE (2011)
6. Bernardin, K., Stiefelhagen, R.: Evaluating multiple object tracking performance: the clear mot metrics. EURASIP J. Image Video Process. **2008**, 1–10 (2008)
7. Betke, M., Hirsh, D.E., Bagchi, A., Hristov, N.I., Makris, N.C., Kunz, T.H.: Tracking large variable numbers of objects in clutter. In: 2007 IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–8. IEEE (2007)
8. Bewley, A., Ge, Z., Ott, L., Ramos, F., Upcroft, B.: Simple online and realtime tracking. In: 2016 IEEE international conference on image processing (ICIP), pp. 3464–3468. IEEE (2016)
9. Brau, E., Dunatunga, D., Barnard, K., Tsukamoto, T., Palanivelu, R., Lee, P.: A generative statistical model for tracking multiple smooth trajectories. In: CVPR 2011, pp. 1137–1144. IEEE (2011)
10. Chang, Y., Tu, Z., Xie, W., Yuan, J.: Clustering driven deep autoencoder for video anomaly detection. In: European Conference on Computer Vision, pp. 329–345. Springer (2020)
11. Collins, R.T.: Multitarget data association with higher-order motion models. In: 2012 IEEE conference on computer vision and pattern recognition, pp. 1744–1751. IEEE (2012)
12. Dai, J., Li, Y., He, K., Sun, J.: R-fcn: Object detection via region-based fully convolutional networks. arXiv preprint arXiv:1605.06409 (2016)
13. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: 2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05), vol. 1, pp. 886–893. Ieee (2005)
14. Dawn, D.D., Shaikh, S.H.: A comprehensive survey of human action recognition with spatio-temporal interest point (stip) detector. Vis. Comput. **32**(3), 289–306 (2016)
15. Dehghan, A., Assari, S.M., Shah, M.: Gmmcp tracker: Globally optimal generalized maximum multi clique problem for multiple object tracking. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4091–4099 (2015). https://doi.org/10.1109/CVPR.2015.7299036
16. Dicle, C., Camps, O.I., Sznaier, M.: The way they move: Tracking multiple targets with similar appearance. In: Proceedings of the IEEE international conference on computer vision, pp. 2304–2311 (2013)
17. Erhan, D., Szegedy, C., Toshev, A., Anguelov, D.: Scalable object detection using deep neural networks. In: 2014 IEEE Conference on Computer Vision and Pattern Recognition, pp. 2155–2162 (2014). https://doi.org/10.1109/CVPR.2014.276
18. Felzenszwalb, P.F., Girshick, R.B., McAllester, D., Ramanan, D.: Object detection with discriminatively trained part-based models. IEEE Trans. Pattern Anal. Mach. Intell. **32**(9), 1627–1645 (2009)
19. Feng, W., Hu, Z., Wu, W., Yan, J., Ouyang, W.: Multi-object tracking with multiple cues and switcher-aware classification. arXiv preprint arXiv:1901.06129 (2019)
20. Fernández-Ramírez, J., Álvarez-Meza, A., Pereira, E., Orozco-Gutiérrez, A., Castellanos-Dominguez, G.: Video-based social behavior recognition based on kernel relevance analysis. Vis. Comput. **36**(8), 1535–1547 (2020)
21. Ferryman, J.: Pets 2006 benchmark data. http://www.cvg.reading.ac.uk/PETS2006/data.html
22. Fortmann, T., Bar-Shalom, Y., Scheffe, M.: Sonar tracking of multiple targets using joint probabilistic data association. IEEE J. Ocean. Eng. **8**(3), 173–184 (1983)
23. Fu, C.Y., Liu, W., Ranga, A., Tyagi, A., Berg, A.: Dssd : Deconvolutional single shot detector. ArXiv:1701.06659 (2017)
24. Girshick, R.: Fast r-cnn. In: 2015 IEEE International Conference on Computer Vision (ICCV), pp. 1440–1448 (2015). https://doi.org/10.1109/ICCV.2015.169
25. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation.

In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 580–587 (2014)

26. Gupta, S., Kapil, R., Kanahasabai, G., Joshi, S.S., Joshi, A.S.: Sd-measure: A social distancing detector. In: 2020 12th International Conference on Computational Intelligence and Communication Networks (CICN), pp. 306–311. IEEE (2020)
27. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. IEEE Trans. Pattern Anal. Mach. Intell. **42**(2), 386–397 (2020). https://doi.org/10.1109/TPAMI.2018.2844175
28. He, K., Zhang, X., Ren, S., Sun, J.: Spatial pyramid pooling in deep convolutional networks for visual recognition. IEEE Trans. Pattern Anal. Mach. Intell. **37**(9), 1904–1916 (2015)
29. Isard, M., Blake, A.: Condensation-conditional density propagation for visual tracking. Int. J. Comput. Vis. **29**(1), 5–28 (1998)
30. Kalman, R.E., Bucy, R.S.: New results in linear filtering and prediction theory (1961)
31. Kim, C., Li, F., Ciptadi, A., Rehg, J.M.: Multiple hypothesis tracking revisited. In: Proceedings of the IEEE international conference on computer vision, pp. 4696–4704 (2015)
32. Leibe, B., Schindler, K., Van Gool, L.: Coupled detection and trajectory estimation for multi-object tracking. In: 2007 IEEE 11th International Conference on Computer Vision, pp. 1–8. IEEE (2007)
33. Li, Y., Huang, C., Nevatia, R.: Learning to associate: Hybrid-boosted multi-target tracker for crowded scene. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition, pp. 2953–2960 (2009). https://doi.org/10.1109/CVPR.2009.5206735
34. Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2117–2125 (2017)
35. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: common objects in context. In: European Conference on Computer Vision, pp. 740–755. Springer (2014)
36. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: Ssd: Single shot multibox detector. In: European Conference on Computer Vision, pp. 21–37. Springer (2016)
37. Luo, W., Xing, J., Milan, A., Zhang, X., Liu, W., Kim, T.K.: Multiple object tracking: a literature review. Artificial Intelligence p. 103448 (2020)
38. Mabrouk, A.B., Zagrouba, E.: Spatio-temporal feature using optical flow based distribution for violence detection. Pattern Recognit. Lett. **92**, 62–67 (2017)
39. Mabrouk, A.B., Zagrouba, E.: Abnormal behavior recognition for intelligent video surveillance systems: a review. Expert Syst. Appl. **91**, 480–491 (2018)
40. Mercaldo, F., Martinelli, F., Santone, A.: A proposal to ensure social distancing with deep learning-based object detection. In: 2021 International Joint Conference on Neural Networks (IJCNN), pp. 1–5 (2021). https://doi.org/10.1109/IJCNN52387.2021.9534231
41. Milan, A., Leal-Taixe, L., Reid, I., Roth, S., Schindler, K.: Mot16: A benchmark for multi-object tracking (2016)
42. Morris, B.T., Trivedi, M.M.: A survey of vision-based trajectory learning and analysis for surveillance. IEEE Trans. Circuits Syst. Video Technol. **18**(8), 1114–1127 (2008). https://doi.org/10.1109/TCSVT.2008.927109
43. Najibi, M., Rastegari, M., Davis, L.S.: G-cnn: An iterative grid based object detector. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2369–2377 (2016). https://doi.org/10.1109/CVPR.2016.260
44. Nam, Y.: Loitering detection using an associating pedestrian tracker in crowded scenes. Multimed. Tools Appl. **74**(9), 2939–2961 (2013). https://doi.org/10.1007/s11042-013-1763-7
45. (NCD-RisC), N.R.F.C.: A century of trends in adult human height. eLife **5**, e13410 (2016). https://doi.org/10.7554/eLife.13410
46. Patel, A.S., Merlino, G., Bruneo, D., Puliafito, A., Vyas, O., Ojha, M.: Video representation and suspicious event detection using semantic technologies. Semantic Web **12**(3), 467–491 (2021). https://doi.org/10.3233/sw-200393
47. Patel, A.S., Vyas, O.P., Ojha, M.: Vehicle tracking and monitoring in surveillance video. In: 2019 IEEE Conference on Information and Communication Technology, pp. 1–6 (2019). https://doi.org/10.1109/CICT48419.2019.9066256
48. Patino, L., Cane, T., Vallee, A., Ferryman, J.: Pets 2016: Dataset and challenge. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 1240–1247 (2016). https://doi.org/10.1109/CVPRW.2016.157
49. Rai, H., Kolekar, M.H., Keshav, N., Mukherjee, J.: Trajectory based unusual human movement identification for video surveillance system. In: Progress in Systems Engineering, pp. 789–794. Springer (2015)
50. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 779–788 (2016). https://doi.org/10.1109/CVPR.2016.91
51. Redmon, J., Farhadi, A.: Yolo9000: Better, faster, stronger. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 6517–6525 (2017). https://doi.org/10.1109/CVPR.2017.690
52. Redmon, J., Farhadi, A.: Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767 (2018)
53. Reid, D.: An algorithm for tracking multiple targets. IEEE Trans. Autom. Control **24**(6), 843–854 (1979)
54. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: towards real-time object detection with region proposal networks. IEEE Trans. Pattern Anal. Mach. Intell. **39**(6), 1137–1149 (2017). https://doi.org/10.1109/TPAMI.2016.2577031
55. Rezatofighi, S.H., Milan, A., Zhang, Z., Shi, Q., Dick, A., Reid, I.: Joint probabilistic data association revisited. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 3047–3055 (2015)
56. Ristani, E., Solera, F., Zou, R., Cucchiara, R., Tomasi, C.: Performance measures and a data set for multi-target, multi-camera tracking. In: Hua, G., Jégou, H. (eds.) Computer Vision—ECCV 2016 Workshops, pp. 17–35. Springer, Cham (2016)
57. S, A., R, N.S.: Versatile loitering detection based on non-verbal cues using dense trajectory descriptors. Multimedia Tools and Applications **78**(8), 10933-10963 (2018). https://doi.org/10.1007/s11042-018-6618-9
58. Saponara, S., Elhanashi, A., Gagliardi, A.: Implementing a real-time, ai-based, people detection and social distancing measuring system for Covid-19. J. Real-Time Image Process. 1–11 (2021)
59. Shen, Z., Liu, Z., Li, J., Jiang, Y., Chen, Y., Xue, X.: Dsod: Learning deeply supervised object detectors from scratch. In: 2017 IEEE International Conference on Computer Vision (ICCV), pp. 1937–1945 (2017). https://doi.org/10.1109/ICCV.2017.212
60. Son, J., Baek, M., Cho, M., Han, B.: Multi-object tracking with quadruplet convolutional neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 5620–5629 (2017)
61. Sugianto, N., Tjondronegoro, D., Stockdale, R., Yuwono, E.I.: Privacy-preserving ai-enabled video surveillance for social distancing: responsible design and deployment for public spaces. Information Technology & People (2021)
62. Tu, Z., Li, H., Zhang, D., Dauwels, J., Li, B., Yuan, J.: Action-stage emphasized spatiotemporal vlad for video action recognition. IEEE Trans. Image Process. **28**(6), 2799–2812 (2019). https://doi.org/10.1109/TIP.2018.2890749
63. Vishwakarma, S., Agrawal, A.: A survey on activity recognition and behavior understanding in video surveillance. Vis. Comput. **29**(10), 983–1009 (2013)

64. Wojke, N., Bewley, A., Paulus, D.: Simple online and realtime tracking with a deep association metric. In: 2017 IEEE international conference on image processing (ICIP), pp. 3645–3649. IEEE (2017)
65. Wu, B., Nevatia, R.: Detection and tracking of multiple, partially occluded humans by bayesian combination of edgelet based part detectors. Int. J. Comput. Vision **75**(2), 247–266 (2007)
66. Wu, Z., Kunz, T.H., Betke, M.: Efficient track linking methods for track graphs using network-flow and set-cover techniques. In: CVPR 2011, pp. 1185–1192. IEEE (2011)
67. Xing, J., Ai, H., Lao, S.: Multi-object tracking through occlusions by local tracklets filtering and global tracklets association with detection responses. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition, pp. 1200–1207. IEEE (2009)
68. Yoo, D., Park, S., Lee, J., Paek, A.S., Kweon, I.S.: Attentionnet: Aggregating weak directions for accurate object detection. In: 2015 IEEE International Conference on Computer Vision (ICCV), pp. 2659–2667 (2015). https://doi.org/10.1109/ICCV.2015.305
69. Zhang, L., Li, Y., Nevatia, R.: Global data association for multi-object tracking using network flows. In: 2008 IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–8. IEEE (2008)
70. Zuo, F., Gao, J., Kurkcu, A., Yang, H., Ozbay, K., Ma, Q.: Reference-free video-to-real distance approximation-based urban social distancing analytics amid covid-19 pandemic. J. Transp. Health **21**, 101032 (2021)

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

Mr. Ashish Singh Patel has completed his B.Tech. from NIT Raipur in Information Technology and M.Tech. in Computer Science and Engineering from Indian Institute of Technology Guwahati. After completing PhD work from DSPM IIIT Naya Raipur, currently he is working as Principal Project Scientist at Indian Institute of Technology, Delhi. His area of interest includes Data Science, IOT, Video Analysis, and Machine Learning.

Dr. Ranjana Vyas is currently working at Govt. of India's Indian Institute of Information Tech nology-Allahabad (IIITA) as Assistant Professor (Business Informatics) since July 2013. Her research interests include Business Intelligence, Data Mining, Information Systems Strategy, Semantic Web.

Prof. O. P. Vyas is currently Professor (Information Technology), Indian Institute of Information technology, Allahabad (UP) India, Coordinator Indo- Norway ICT Project (DST-India, RCN-Norway). He has done M.Tech. in "Computer Science & Data Processing" from IIT Kharagpur and Ph.D. under joint collaboration with Technical University of Kaiserslautern (Germany) and IIT Kharagpur. He has authored over 100 research articles in International journals and conferences. His research interests include Data Mining and Business Intelligence, Mobile Adhoc Networks and Wireless Sensor networks, Future Internet, and Software Engineering.

Dr. Muneendra Ojha is Assistant Professor in the department of Computer Science and Engineering. Dr. Ojha earned his PhD from Indian Institute of Information Technology-Allahabad and MS(Research) from University of Missouri-Columbia, USA. He started his career as Assistant Systems Engineer at Tata Consultancy Services. Prior to joining IIIT-NR, Dr. Ojha has been associated as faculty with several renowned academic institutions such as IIITAllahabad and Shiv Nadar University. His research interest include Multiobjective Optimization, Evolutionary Algorithms, Machine Learning, Artificial Intelligence, and Multi-agent systems.

Dr. Vivek Tiwari is an Assistant Professor in the Department of Computer Science and Engineering at DSPM IIIT-Naya Raipur, C. G. India. He received the B.Eng. Degree (Comp. Sci. & Engg.) from the Rajiv Gandhi Technical University, Bhopal, in 2004, and the M.Tech. degree (Comp. Sci. & Engg.) from SATI, Vidisha (MP) in 2008. He has obtained PhD degree from the National Institute of Technology, Bhopal (MANIT), India, in 2015 in the area of Data mining and warehousing. Earlier, he worked with the Modi Institute of Technology and Science (MITS), Deemed University, Sikar, Rajasthan, and Caresoft Incorporation (based at Middlesex, NJ, USA) as Software Engineer.