AALBORG UNIVERSITY
STUDENT REPORT

Suspicious Behavior Detection in Video Surveillance

Project Report
Group 843

Aalborg University
MSc in Vision, Graphics and Interactive Systems

School of Information and Communication Technology
Aalborg University

AALBORG UNIVERSITY
STUDENT REPORT

Title:
Suspicious Behavior Detection in Video Surveillance

Theme:
Computer Vision

Project Period:
Spring Semester 2018

Project Group:
843

Group Members:
Ana Rita Viana Nunes
Atanas Atanasov Nikolov

Supervisors:
Kamal Nasrollahi
Mohammad Naser Sabet Jahromi

Pages Number: 41

Date of Completion:
May 29, 2018

Abstract:
The detection of suspicious behaviors in surveillance videos has become a subject of interest recently. The need to protect people, especially in public areas, has driven researchers to investigate new and improved ways of detecting suspicious people in surveillance video feeds as a way of helping security officers assure public safety.

The presented work focuses on detecting people and objects in videos using as a background subtraction method the Temporal Median Filter. To aid people's accurate detection, a BLOB fusion algorithm was implemented.

Every detection in the video is tracked through time using the Kalman Filter method. To ensure the correct tracking of people, even in the occurrence of occlusions, a re-identification algorithm based on visual appearance was implemented.

The suspicious behaviors identified were fainting, loitering, running, abandoned objects and abandoned objects being picked up.

The system was tested using videos from the CAVIAR dataset and achieved good results in the performed tests.

Contents

Chapter 1

Introduction

One of the research topics in Computer Vision that has become of interest in recent years is the detection of suspicious activities in surveillance videos.

Nowadays, more than ever, public safety has become a great concern. Therefore, public spaces need to be monitored for any individual(s) that might be acting out of the ordinary with the intent of doing harm to someone else or with the intent of stealing something.

Thus, the need for automated surveillance systems has increased, with the goal of assisting security officers in performing their job more efficiently. These systems can be applied for e.g. crowd analysis, surveillance at airports or train stations, traffic monitoring, criminal activity recognition and so on.

Computer Vision algorithms can be used to detect and track moving targets in a video feed and extract features that allow a system to analyze and classify such targets as performing abnormal or suspicious behaviors and send an alert to security officers.

Initial stages of object detection and tracking in a video feed are crucial for the proper classification of behaviors. Without a correct detection of objects and the consequent accurate tracking, people in the frame cannot be correctly classified. Because of this, a lot of work must be put into finding ways of performing the first two stages accurately as basis for a good classification of behaviors.

The classification of suspicious behaviors is not linear and so research needs to be done in order to develop a system that can accurately detect different types of suspicious behaviors.

1.1 Anomalous versus Suspicious Behavior

Anomalous behavior or anomalies are patterns in data that are different from predefined normal behaviors (normal model). Figure 1.1 shows an example of anomalies in a two-dimensional dataset. There are two normal regions N_1 and N_2, all points that are lying outside those regions are considered anomalies. Those patterns might be a result of malicious activity, system failure or noise in the data. Therefore, the relevance of anomalies to real life is a key feature in anomaly detection [1].

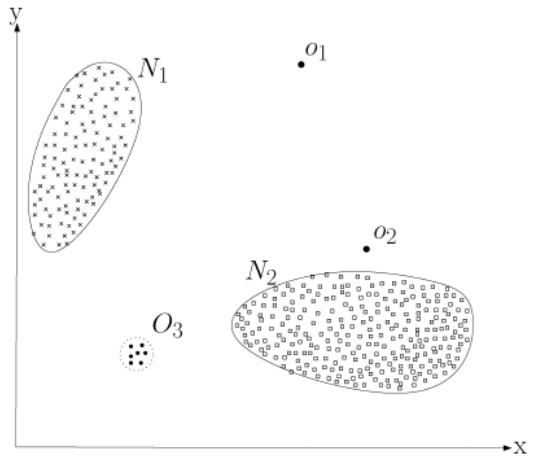


Figure 1.1: An example of anomalies in a two-dimensional dataset [4].

Novelty detection is also closely related to anomaly detection [2, 3]. Unlike anomaly detection, novelty detection is concerned with finding newly developed or previously unknown patterns that a system was not aware of, or failed to recognize during training. The main difference between the two is that novel patterns are typically later incorporated in the normal model.

While anomalous behavior can be described as behavior which differs from the expected, suspicious behavior is not that simple to model. Suspicious behavior is detected through subjective interpretation and presents a challenge even for human observers. Often, how a situation or action develops can be used as an effective tool in detecting suspicious behavior. Human observers rely on their experience often described as 'sixth sense' or 'gut feeling', to correctly detect suspicious behavior [5]. Context is another very important part of suspicious behavior detection. Behavior that is considered normal can become suspicious in a different context. Because of that, the normal behavior model needs to be updated over time as the context changes. This makes labeling data for training nearly impossible and it is also impossible to generate a dataset capturing all possible human behaviors [1].

1.2 Challenges

Since anomalies are considered deviations from the normal model, detecting them should be straightforward. One solution would be to define a normal region where all observations outside that region are abnormal. However, several issues arise in implementing this approach, as the followings:

- It is very difficult to define a normal region, that correctly represents every possible normal behavior. In addition, the separation between normal and abnormal is not always clearly defined.
- When the anomalies are caused by malicious actions, those actions are usually made to appear normal.
- The normal behavior is prone to change over time. A model which is currently a good representation of normal behavior might not be sufficient in the future.
- It is hard to find labeled data for training models.
- Detecting suspicious behavior is very subjective and depends on how actions are interpreted.
- The systems need to take into account that behavior that might be considered suspicious in one context might be normal in other contexts.

Because of these challenges, it is not easy to create a general solution for detecting suspicious behavior. Instead, most solutions are focused on very specific formulations of the problem.

1.3 Related Work

The detection of suspicious behavior by automatic systems has raised a lot of interest in recent years. To be able to accurately detect suspicious behavior a system needs to have an understanding of general human behavior but also needs to understand the context in which this behavior is presenting itself.

Wiliem et al. [6] propose a context-based system for detecting suspicious behavior. This work considers that a system like this needs to have three main components. Firstly, it needs to continuously extract and learn contextual and human behavioral information from the video stream. Hence, a context space model is introduced in which the systems designers select important information that can describe a context but also allows the system to distinguish between two different instances of contexts. Secondly, it exploits contextual information in making de-

cisions by introducing the use of a data stream clustering algorithm that enables the system to continuously update its knowledge from the data. This algorithm is capable of retrieving knowledge learned from a specific context. Lastly, the system incorporates an inference algorithm that combines contextual information and the system's knowledge to make decisions. The system also incorporates human observers' input in the decision making.

To test the system, experiments were done using the CAVIAR dataset [7] and a private dataset. The experiments showed that the system made accurate detections due to using previously acquired knowledge relevant to the context. The system was even able to detect unexpected events.

A lot of researchers resort to machine learning for detecting suspicious behavior which mostly relies on having reliable standard datasets for training and testing which, sometimes, might be hard to acquire. To overcome this issue Elhamod and Levine [8] propose a semantics-based solution which is based on human reasoning and logic. This solution elaborates a mathematical framework based on abstract descriptions shown by Fuentes and Velastin [9] for detecting suspicious behavior and also builds up on previous work by the same authors [10].

Their proposed framework tracks people and luggage in a scene. Behaviors and events happening in a scene are semantically recognized by extracting object and inter-object motion features. The context of the investigation was the detection of suspicious behavior in public transport areas.

While earlier articles focus on detecting only one type of behavior [11,12], this work analyzes different types of behavior relevant to the context such us: abandoned and stolen objects, fighting, fainting and loitering.

Detected objects in a scene are classified as being animate (e.g. people) or inanimate (e.g. luggage) which are the semantic entities associated with the events described. The extracted features are divided into single-object and inter-object. The single-object features include position, speed, direction and merged (a Boolean that specifies whether an object is merged with another or not), while distance, alignment and speed difference are the inter-object features. The method follows the concept presented by Bird et al. [12] which uses motion features to classify objects into four categories: *unknown U*, *abandoned object O*, *person P*, and *still person SP*. The motion features are calculated and recorded in historical records and behaviors are semantically defined and detected by checking the records against predefined rules and conditions.

Public datasets such as BEHAVE [13], CAVIAR (PETS 2004) [14] and PETS 2006 [15] were used to test this framework. The results showed that the framework

successfully detects all of the events discussed.

Methods for detecting suspicious behavior are usually developed for specific types of behaviors. To improve the accuracy of the system many methods require complex feature extraction algorithms that do not allow for the systems to be used in real-time. To solve this, Mu et al. [16] present a fast method for detecting suspicious behavior such as wandering, trailing, chasing and falling down. The proposed method is based on the extraction of motion vectors from a video stream to obtain the necessary features to classify behaviors in a video.

7-D features $\{\theta, V, \sigma_\theta, \sigma_V, E_\theta, E_V, Inter_{Dj}\}$ are extracted from a frame to describe each target detected. θ and V represent the average direction and velocity of the target; σ_θ and σ_V represent the direction variance and the velocity variance; E_θ and E_V represent entropy of direction and velocity; $Inter_{Dj}$ is the interesting degree of inter-frame $Inter_D$. The direction, velocity and inter-frame difference proved to be the most important features. A Support Vector Machine (SVM) is used to learn and classify the input videos.

The results from the experiments are compared with those in [17–20] and this method, in the majority of aspects, shows significant improvements when compared with the other methods.

In order to increase the efficacy of a video-surveillance control center for a shopping mall, in comparison with traditional methods, Arroyo et al. [21] investigated the detection of suspicious behavior in shopping malls. The analyzed risk situations in this context were a shop entry or exit of people, loitering events that can lead to theft and unattended cash desk situations.

The proposed approach employs an innovative tracking method that manages occlusions based on SVM kernels to compute distances between appearance features such as GCH (Global Color Histogram), LBP (Local Binary Pattern) and HOG (Histogram of Oriented Gradients). With these features, color, texture and gradient information are combined to obtain a robust visual description of people in videos.

The analysis of people's entrance or exit from the shops is important in the detection of crowded situations when a lot of people enter a shop at the same time or when people run away when exiting. For this, the line of the entrance to the shop is manually placed and the directions of people passing by are analyzed. For loitering detection, risk zones are specified and loitering is detected whenever someone is in an area for a period of time longer than a defined threshold. Unattended cash desks are a risky situation because someone might try to steal money from the cash register if no one is looking. Hence, an alarm is given if a person is detected loitering around an unattended cash desk.

To evaluate the performance of their system, the publicly available CAVIAR dataset was used for testing occlusion situations and the system showed to be slightly superior to multiple others state-of-the-art methods [22–24]. The system was also tested on a private dataset and it was concluded that it could detect suspicious behaviors in the intended context.

State-of-the-art approaches need to take into consideration the context in which they want to detect suspicious behavior in order to accurately do so. There is still a lot of research that can be done to unveil new ways and approaches on how to implement even more efficient systems.

1.4 Problem Statement

A system that can accurately detect suspicious behavior in public places can be of great help for security officers to guarantee public safety. However, the detection of such behavior does not have a straightforward resolution. The approach investigated in this work is:

How to detect suspicious behavior in public areas, in real time, using a rule-based approach to classify suspicious behaviors such as fainting, loitering, running and abandoned objects.

The created system should be tested with a standard public dataset.

1.5 Organization

A system that is capable of classifying a behavior as suspicious must go through different stages before being able to classify a behavior.

Firstly, people or objects in the video need to be detected and separated from the background as explained in Chapter 2.

As a second stage, every detection must be tracked throughout time, this can be extremely complex due to people being occluded by other people passing by, as described in Chapter 3.

Chapter 4 gives details on how the designed system classifies the detected behaviors as suspicious or not. Lastly, Chapter 5 reveals the results achieved with this implementation.

Chapter 2

Object Detection

The detection of moving objects in the video stream is the first step that must be performed. For this, each frame of the video is pre-processed and, from the RGB input frame (Fig. 2.1), a monochromatic image with only gray levels of color is obtained. Also, a Gaussian Blur filter is applied to the image to reduce noise and detail (Fig. 2.2).

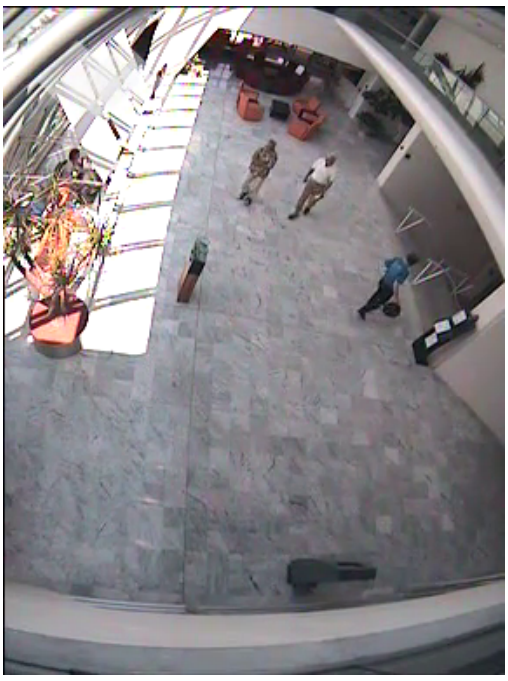


Figure 2.1: Frame from a video.

Figure 2.2: Frame in gray level and with Gaussian Blur filter applied.

The videos represented are from the CAVIAR public dataset [7].

2.1 Background Subtraction

Moving objects can be extracted from a frame by calculating the difference between the pixel values of the frame and the ones in the background. To accomplish that, the background image of the video must be constructed.

Different background subtraction methods can produce good results such as Gaussian Mixture Model [25], Temporal Median Filter [26, 27] and others. The Gaussian Mixture Model method is very good at extracting the moving foreground from the static background and is able to handle changes in the light condition of the video since it updates the background continuously. However, if the model updates too fast a person that is standing still for a while, suddenly becomes part of the background, which is not desirable, since still people or objects are a very important part of the suspicious behavior detection, to detect, for example, abandoned objects.

The Temporal Median Filter extracts the background image by removing the noise from the image. The values assigned to each pixel throughout time are stored and ordered. The median value of the list is chosen as the background, this eliminates values that are too high or too low, thus eliminating noise in the image. This method shows high accuracy and low complexity, therefore it was chosen for this application.

In this application, only very small videos are being processed, therefore it was not necessary to implement a Temporal Median Filter algorithm that updates through time. In a real-world application, such algorithm would need to be implemented in order to account for changes in e.g. the light conditions.

The algorithm implemented processes the entire video beforehand and, because consecutive frames are very similar to each other, only stores the pixel values of a frame every half second. In the end, sorts all the values in ascending order and selects the median value as the background value.

Figure 2.3 shows the background extracted from the video and Figure 2.4 displays the result of the difference between the video frame (Fig. 2.2) and the background.

Figure 2.3: Background image.

Figure 2.4: Result of background subtraction.

2.2 Pre-processing

After the background subtraction, the binarization of the image takes place by thresholding the image. This means that all the pixels with a value below a certain threshold are assigned a zero value and all pixels with an input value above the threshold are given the value one (Equation 2.1).

$$g(x,y) = \begin{cases} 1, & \text{if } f(x,y) \geqslant T. \\ 0, & \text{otherwise.} \end{cases} \tag{2.1}$$

To choose the correct threshold T value, the histogram of values from Figure 2.4 was analyzed and a zoomed image of the histogram is represented in Figure 2.5. Therefore, T was assigned a value of 30 as it demonstrates to produce good results separating objects from redundant information.

As Figure 2.6 demonstrates (for the case of a different frame), a very clear distinction between background and foreground is achieved.

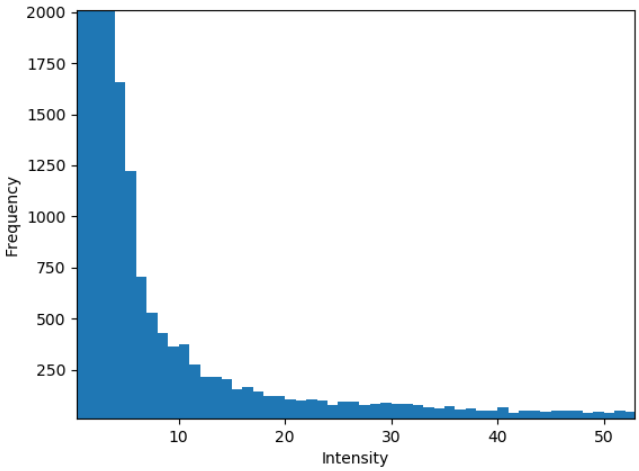


Figure 2.5: Histogram of values in frame.

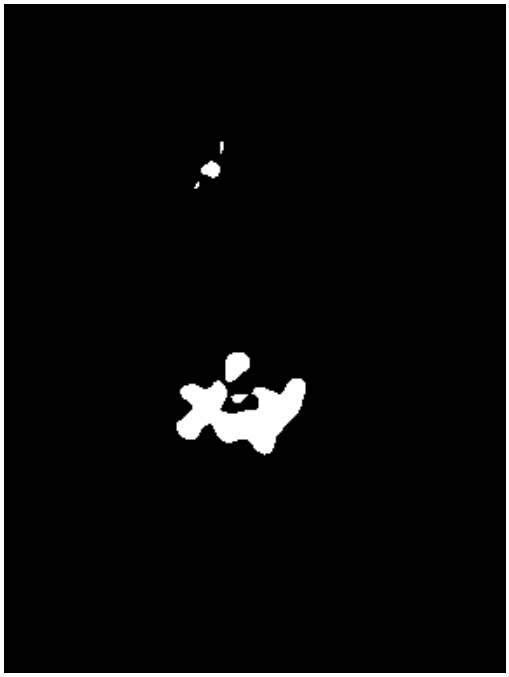


Figure 2.6: Binary frame.

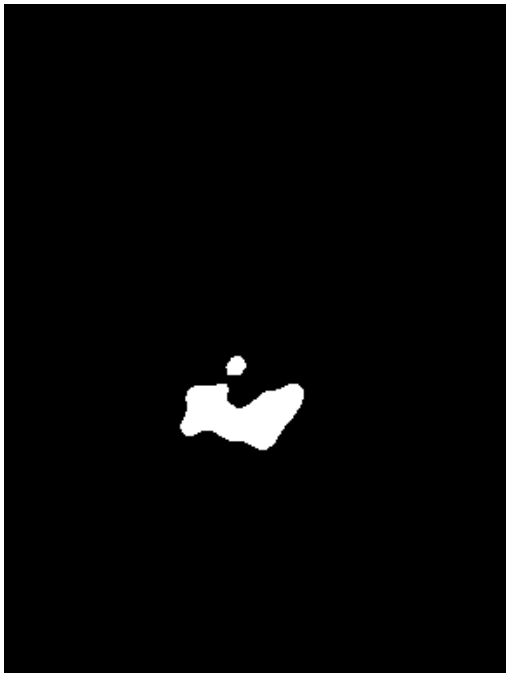


Figure 2.7: Frame with median filter applied.

It is possible to notice from Figure 2.6 some noise in the frame, that is corrected by applying a median filter to the frame (Fig. 2.7).

Lastly, the morphological operation of Closing is applied to the frame. A morphological operator takes as input a binary image and a kernel and combines them using an operation. The kernel operates over every pixel in the image. The oper-

ator used was the Closing operator that is a combination of Dilation and Erosion. The Closing operator is usually used to close small holes in the white area of the image, in the presented case of Figure 2.8 it connects the head and the body of a person, that were split on the previous steps. The Dilation increases the white region in the image while the Erosion returns the area to its original dimensions. The operation is written as

$$g(x, y) = (f(x, y) \oplus K) \ominus K \tag{2.2}$$

where K represents the kernel.

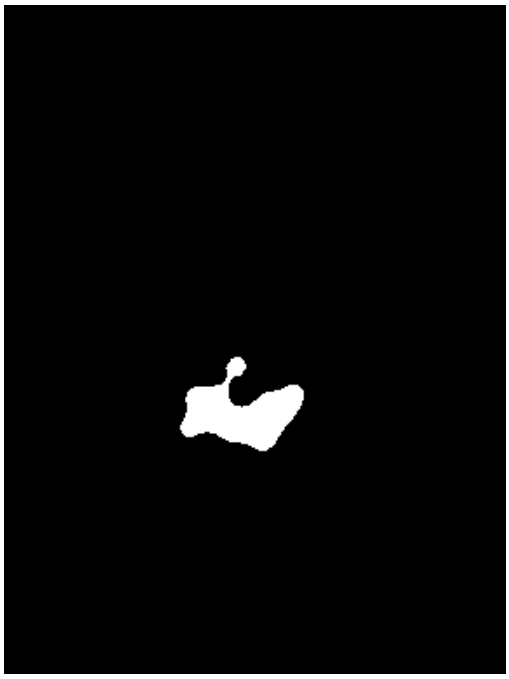


Figure 2.8: Frame with Closing morphological operation applied.

2.3 BLOB Fusion

A BLOB (Binary Large OBject) represents connected pixels in a binary image, and each one corresponds to an object detected in the frame. The BLOBs in the frame are detected by a built-in OpenCV function and the respective bounding boxes are displayed.

Figure 2.10 shows how a person in Figure 2.9 was split into two BLOBs because the previously applied methods were not effective enough on segmenting

the person, in this case. To attempt to correct this, a BLOB fusion algorithm was implemented. The algorithm was based in the one described in [21].

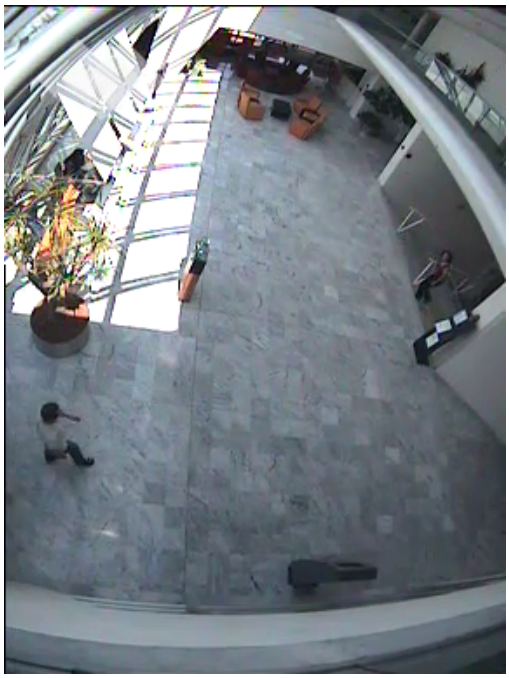


Figure 2.9: Frame of video.

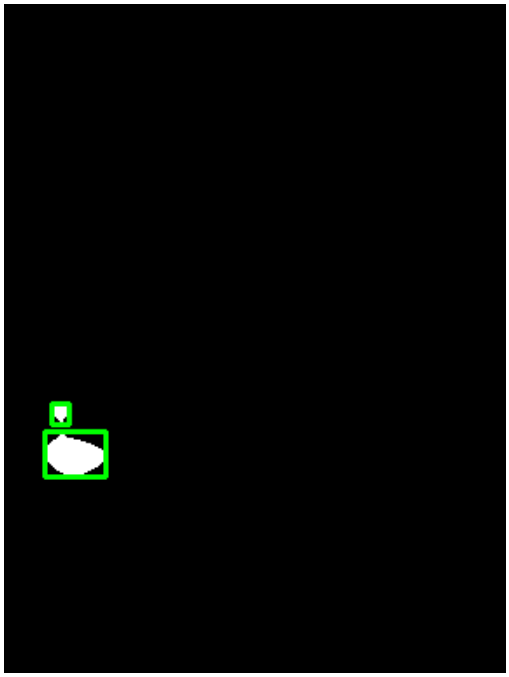


Figure 2.10: Binary image with bounding boxes surrounding detected BLOBs.

Each detected BLOB in the image is characterized by their upper left and lower right corners coordinates (x_1, y_1, x_2, y_2). Two BLOBs are fused together if they pass all of the three conditions imposed.

The first condition states that at least one of the BLOBs must have a bounding box area lower than a given limit l_1. The limit l_1 is an approximated value to the average of the area of all the BLOBs in the frame. When fusing two BLOBs, the intention is not to fuse two large BLOBs that will likely correspond to two different people, but to fuse a smaller BLOB that is most likely to relate to a bigger BLOB.

$$l_1 \approx \frac{\sum_{k=1}^{N} (x_{2_k} - x_{1_k}) \times (y_{2_k} - y_{1_k})}{N} \tag{2.3}$$

Fusion condition 1:

$$((x_{2_i} - x_{1_i}) \times (y_{2_i} - y_{1_i}) < l_1) \vee ((x_{2_j} - x_{1_j}) \times (y_{2_j} - y_{1_j}) < l_1) \tag{2.4}$$

Throughout the video, the height of the detected BLOBs is stored, so it is pos-

sible to calculate the median of these height values and thus get an approximate value of the standard height a BLOB should present. The median value of the height is used to calculate a limit l_2 to be used in the second condition. This limit is equal to half the calculated standard height.

$$l_2 = \frac{Median(\{(y_{2_0} - y_{1_0}), \cdots, (y_{2_M} - y_{1_M})\})}{2} \tag{2.5}$$

The second condition checks the vertical distance between the BLOBs. If the absolute distance between the lower corner of one of the BLOBs and the upper corner of another one is less than the set limit l_2, then the BLOBs are still candidates to being fused.

Fusion condition 2:

$$(|y_{2_i} - y_{1_j}| < l_2) \vee (|y_{2_j} - y_{1_i}| < l_2) \tag{2.6}$$

The last condition needs to make sure that the BLOBs are aligned horizontally. For that to be true, the algorithm assures that the BLOB with greater x_1 value (with the left corner more to the right) is at least slightly aligned with the other BLOB by not having its left corner with a value greater than the right corner of the first BLOB. It is allowed for the BLOB to have a small offset of value o.

Fusion condition 3:

$$(x_{1_i} > x_{1_j} \wedge (x_{1_i} + o) \leqslant x_{2_j}) \vee (x_{1_i} < x_{1_j} \wedge (x_{1_j} + o) \leqslant x_{2_i}) \tag{2.7}$$

If the pair of BLOBs passes all conditions they are merged and, therefore, are deleted from the list of candidates and a new candidate is added which is the result of the merger given by:

$$\begin{aligned} x_{1_{new}} &= \min(x_{1_i}, x_{2_i}, x_{1_j}, x_{2_j}) \\ y_{1_{new}} &= \min(y_{1_i}, y_{2_i}, y_{1_j}, y_{2_j}) \\ x_{2_{new}} &= \max(x_{1_i}, x_{2_i}, x_{1_j}, x_{2_j}) \\ y_{1_{new}} &= \max(y_{1_i}, y_{2_i}, y_{1_j}, y_{2_j}) \end{aligned} \tag{2.8}$$

The process is repeated for all the candidates until no more fusions are identified.

The result is demonstrated in Figure 2.11 where it is possible to notice the algorithm now assumes both BLOBs as being part of the same person.

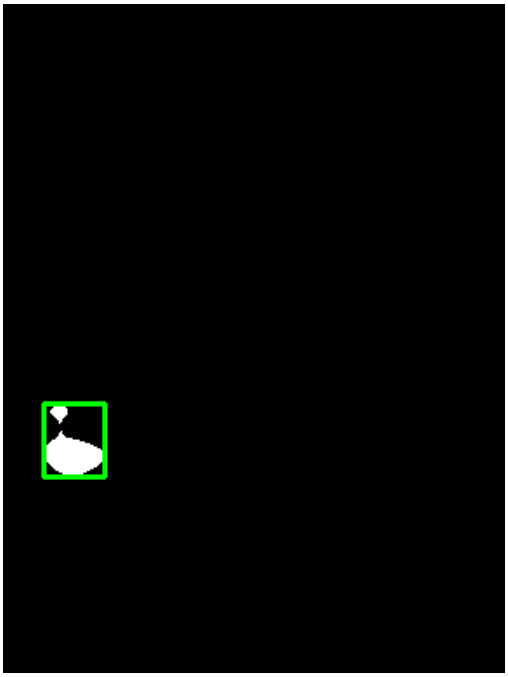


Figure 2.11: Binary image with BLOB fusion algorithm applied.

Chapter 3

Tracking

3.1 Kalman Filter

The Kalman Filter introduced by Rudolf E. Kálmán in [28] is now a widely used algorithm for a range of different applications. One of its applications is the tracking of detected BLOBs throughout a video feed.

The Kalman Filter is an optimal estimation algorithm that, given a series of measurements obtained over time, filters their possible noise and predicts the next state and minimizes the estimated error covariance.

The algorithm has a prediction step, where it estimates the next state of a system based on its previous state, and a correction step where it updates the state based on a new observed measurement. The algorithm iterates recursively across these steps (Fig. 3.1).

The state x of a discrete-time process is characterized by the linear stochastic difference equation:

$$x_k = Ax_{k-1} + Bu_k + w_{k-1} \tag{3.1}$$

where A is the transition matrix that relates the state x at the previous time step $k-1$ to the state at the current step k. B is the matrix that relates the optional control input u to the state x (in this case it is discarded). The process noise is represented by w.

The measurement z is portrayed by the equation:

$$z_k = Hx_k + v_k \tag{3.2}$$

where the matrix H relates the state to the measurement z_k and v_k represents the measurement noise.

The Kalman Filter estimates the state x_k with the measurement z_k.

The variables w_k and v_k have normal probability distributions with covariance matrices of Q and R, respectively. These matrices must be tuned. A low R value tells the system to rely more on the measurements and vice-versa.

$$p(w) \sim \mathcal{N}(0, Q) \tag{3.3}$$

$$p(v) \sim \mathcal{N}(0, R) \tag{3.4}$$

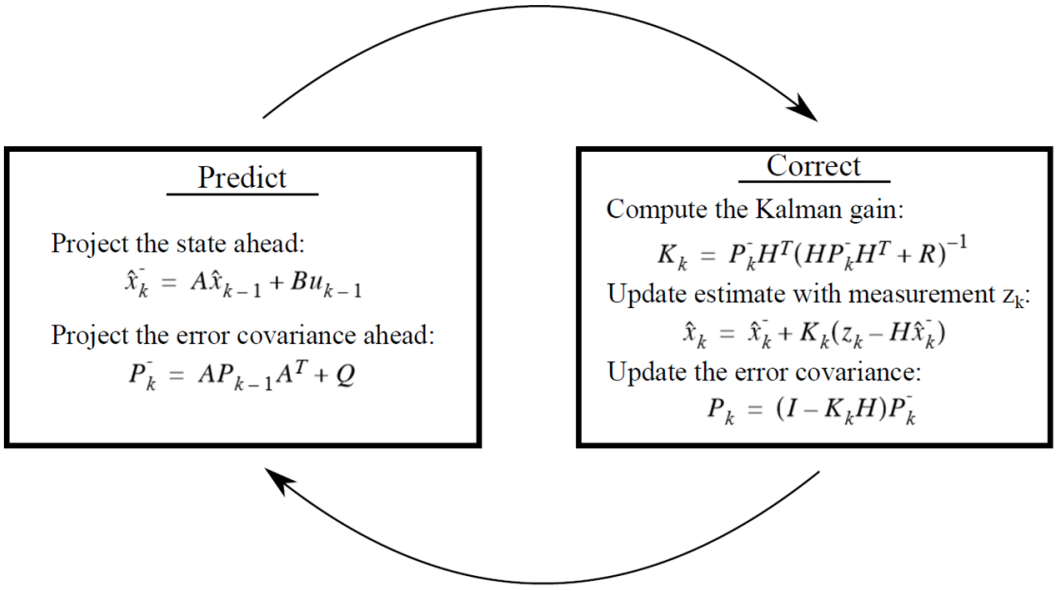


Figure 3.1: Procedure of the Kalman Filter [29].

In the implemented system the state x has four dynamic variables, those being the x- and y-positions, and the x- and y-velocities. The measurement z represents the observed x- and y-positions.

This algorithm was implemented with OpenCV functions that allow for an easy implementation.

3.2 Data Association

For each detected person in a video, a new Kalman Filter is created and the predicted locations are stored. Each existing Kalman Filter is assigned to the nearest detection if, in that frame, no detection has already been assigned to that Filter (to prevent different detections close to each other from being assigned to the same Kalman Filter) and if the distance between the detection and the last predicted position of the Filter is smaller than a defined threshold.

Kalman Filters without assigned detections are still continued based on predictions for a maximum of five frames without assigned detections and they are deleted from memory after three seconds without new detections.

For each detection that is not assigned to any Kalman Filter, a new one is created, thus beginning a new track.

Figure 3.2 shows the predicted tracks of each detected person in the video.

Figure 3.2: Result of the Kalman Filter tracking.

3.3 Occlusion Handling

If a person is not occluded by another person or if its BLOB is not merged with the BLOB of another person in the segmentation stage, then it is easy to predict the person's trajectory based on the algorithms explained previously.

However, if any of the presented cases occurs, the people affected cannot be properly re-identified once the occlusion ends. Figure 3.3 demonstrates a case where, because two people are standing close to each other, their BLOBs were merged together in the segmentation stage. This causes the tracking of the person on the right to be lost and both people are associated with the track of the person on the left. After the occlusion ends, in Figure 3.4, the person on the right cannot be re-identified and so a new track is created.

To minimize the occurrence of this issue, an algorithm based on visual appearance was implemented.

Figure 3.3: Poor tracking due to occlusion.

Figure 3.4: After the end of the occlusion a person cannot be re-identified.

Throughout the video, a list with the information of all the tracks detected is kept. These tracks ideally correspond to one person or object detected but, in cases of poor segmentation or tracking, a track can correspond to multiple people or a person can have multiple tracks assigned to them.

Each track contains a lot of information that helps characterize it, examples of that are: all the positions predicted by the Kalman Filter; a track identification number; the number of frames on which the track's detection has been lost; a list with the dimensions of the bounding box correspondent to the detection; along with other features.

Whenever the number of frames on which a track has been lost is greater than zero the algorithm will try to identify whether an occlusion has begun.

For each of the other tracks that are currently being detected the algorithm will calculate the area of intersection between the lost track's bounding box and the candidate track's bounding box. If the area of intersection is significant and close to the area of the lost track's bounding box, then an occlusion has officially begun and the candidate track has now merged with the lost one.

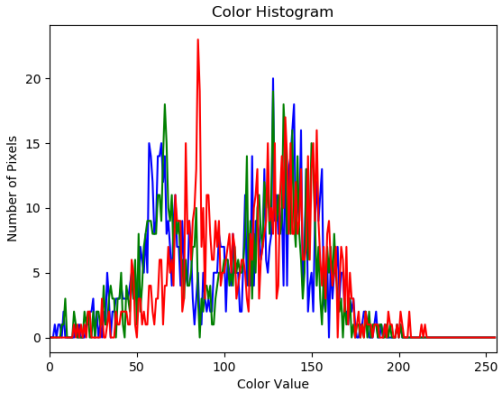
Figure 3.5 shows a frame right before an occlusion begins. The track identification number is displayed above each bounding box. In Figure 3.6 it is possible to notice that the bounding box that now corresponds to both people is now where the other two bounding boxes used to be. The area of intersection between the bounding box number two in Figure 3.5 and the new bounding box in Figure 3.6 is equal to the area of the lost track's bounding box.

Figure 3.5: Frame right before occlusion.

Figure 3.6: Frame where occlusion begins.

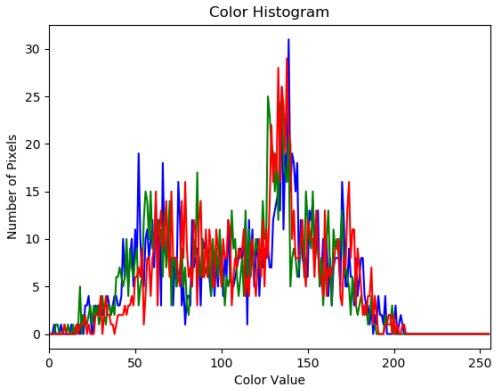
The identification number of the lost track is added to the one that merged with it and the color histograms, of both detections in the frame before the occlusion, are calculated and stored.

The color histogram is a good feature that distinguishes people from each other based on the color of their clothes [30]. Figures 3.7 and 3.8 demonstrate the color histograms for each of the people involved in the occlusion. This information differentiates them quite well and will be used to distinguish the people once the occlusion ends.



(a) Section of frame with person. **(b)** Color histogram.

Figure 3.7: The color histogram information for person with identification number two.



(a) Section of frame with person. **(b)** Color histogram.

Figure 3.8: The color histogram information for person with identification number three.

When a new detection appears in the video with no assigned track it is verified if this detection might correspond to the end of an occlusion. This is done by calculating the intersection between the bounding box of this detection and the

bounding boxes of all other detections in the video in the prior frame, that are marked as having more than one person merged in them. If there is significant intersection this means that an occlusion has ended.

Figure 3.9 shows the same people after they have moved a bit compared to Figure 3.6. Figure 3.10 displays the frame right after, when the occlusion has ended.

Figure 3.9: Frame right before occlusion ends.

Figure 3.10: Frame after occlusion ends.

When the end of an occlusion is verified the color histogram of the new detection (Fig. 3.11) is compared to all color histograms stored in the track that contained occluded people (Fig. 3.7 and 3.8).

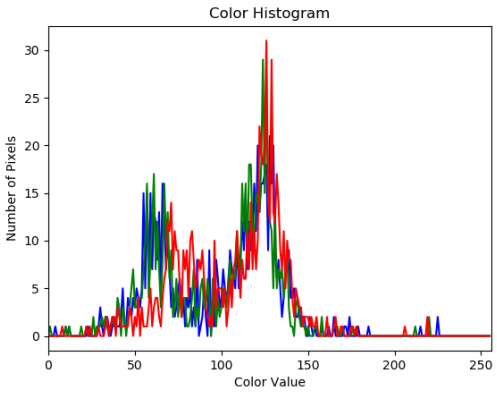
The method used for histogram comparison is the Correlation method characterized by the equations:

$$d(H_1, H_2) = \frac{\sum_I (H_1(I) - \bar{H_1})(H_2(I) - \bar{H_2})}{\sqrt{\sum_I (H_1(I) - \bar{H_1})^2 \sum_I (H_2(I) - \bar{H_2})^2}} \tag{3.5}$$

where

$$\bar{H_k} = \frac{1}{N} \sum_J H_k(J) \tag{3.6}$$

and N is the total number of histogram bins. In this method, a high score represents a better match than a low score.



(a) Section of frame with person. **(b)** Color histogram.

Figure 3.11: The color histogram information for the new detection.

For the presented case, the results of the histograms comparison between Figure 3.11b and Figure 3.7b is approximately 0.8416 and the comparison with the histogram in Figure 3.8b has the approximate result of 0.7751. Because of this, the new detection is assigned to the track with the identification number two.

As soon as the corresponding track is found, the track identification is deleted from the occluded detection and the new detection is assigned to the corresponding track. As demonstrated in Figure 3.10 the track identification is correctly assigned.

Chapter 4

Classification

As addressed in Section 1.3, different types of behaviors can be classified as suspicious. This work focuses on only four types which are fainting or falling down, loitering, running, abandoned objects and abandoned objects being picked up.

Based on the work done in [8], all the detections are divided into four categories: *unknown*, *object*, *person*, and *still person*.

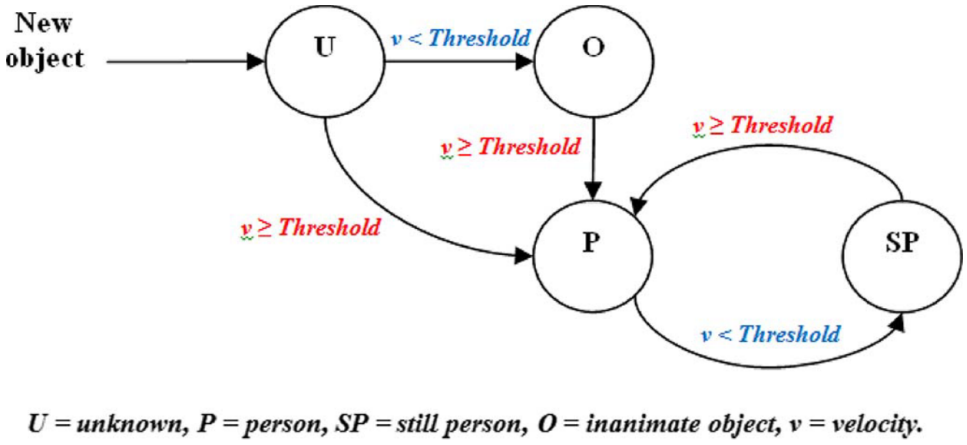


Figure 4.1: Object categorization state diagram [8].

Figure 4.1 shows the state diagram for the implemented algorithm presented by Elhamod and Levine in [8]. As the diagram shows, when a new object is detected in the video, it is initially categorized as *unknown*. After evaluating its velocity, this *unknown* detection is categorized either as an *object* or a *person*, depending on whether its velocity is lower or greater than a specified threshold, respectively. After being categorized as a *person*, a detection can be then categorized as a *still*

person, in case its velocity decreases to below the defined velocity threshold. This model prevents a *still person* from being miscategorized as an *object*.

4.1 Fainting

The aspect ratio of a person's bounding box is a widely used feature to detect fainting. It is defined as the ratio between the height and the width of the person's bounding box. When a person is standing up the ratio will be greater than one and if a person is laying down it will be less than one.

$$\text{aspect ratio} = \frac{\text{height of bounding box}}{\text{width of bounding box}} \tag{4.1}$$

Figure 4.2: Frame of video with no suspicious behaviors detected.

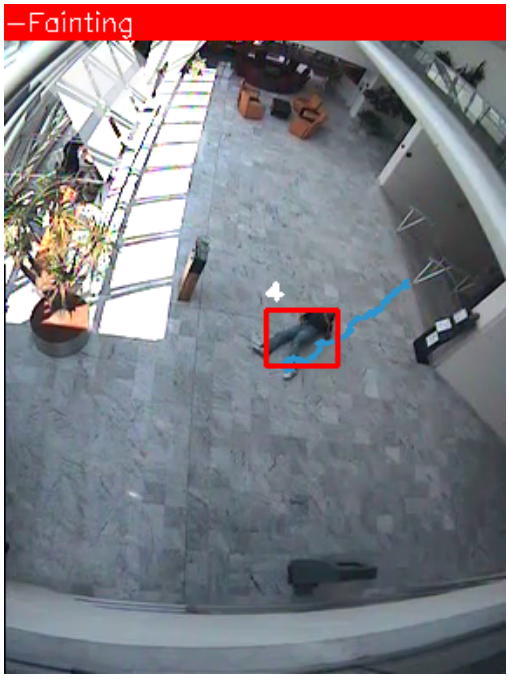


Figure 4.3: Frame of video with fainting detected.

Figure 4.2 shows a frame of a video with no suspicious behaviors detected. A line on top of the video is green, which means that no suspicious behaviors are being detected in the video, at the time. The person in the video has also a bounding box colored green to demonstrate a normal behavior. On the other hand, on Figure 4.3 the same person has a bounding box colored red, this means that a suspicious behavior is being detected. The line on top of the video is now

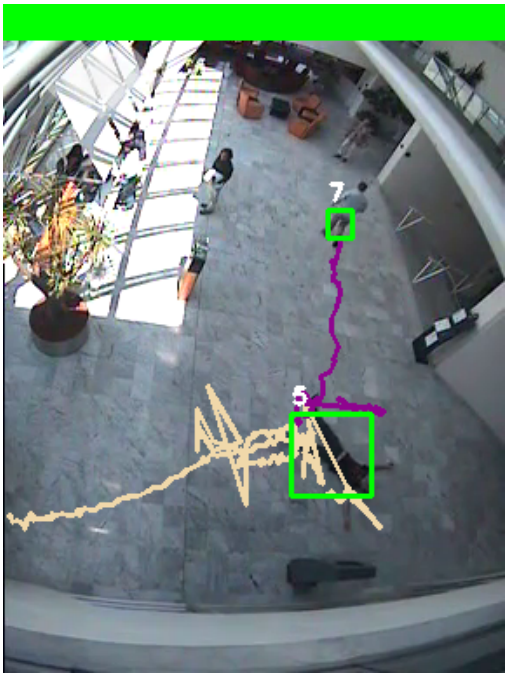


Figure 4.4: Poor detection of the fainting behavior.

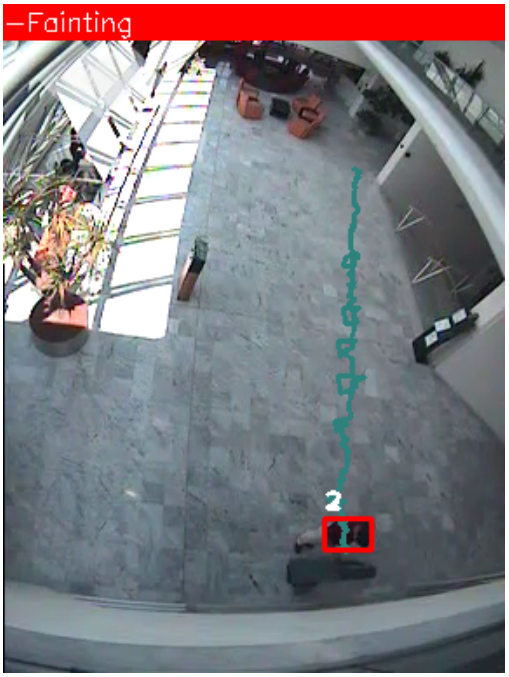


Figure 4.5: False detection of the fainting behavior..

red and displays the class of the suspicious behavior that is being detected which, in this case, is fainting.

For a person to be flagged as having fainted or fallen down it is required that its detection is classified as being a *still person*, the ratio of the bounding box must be less than a specified number, it must not have merged with another detection and, lastly, the width of the bounding box must have an approximated value to the standard height registered on the video.

$$category = \text{SP} \wedge ratio < threshold \wedge merged = \text{False} \wedge width \approx standard\_height \quad (4.2)$$

In the presented data set, the video is filmed from above which poses a problem for the detection of this behavior. In Figure 4.4 a person has fallen on the floor, however, its bounding box does not have the characteristics to classify for the fainting behavior. Due to the video being filmed from above, the person appears to be standing up to the system, even though a human observer can easily classify this person as having fallen down.

On the other hand, Figure 4.5 displays a person that is standing up, however, because the camera is filming this person from above, the bounding box exhibits characteristics that are classified has fainting behavior even though it is clearly not.

4.2 Abandoned Object

An important occurrence to detect in videos is abandoned objects or luggage, in case someone tries to leave e.g. a bomb in a public area.

If a detection is categorized as an *object* and it has been detected for longer than a specified period of time, the system will try to find the person that left the object unattended.

Every detection in the video categorized as a *person* or a *still person* is a candidate for having left the object. The system tries to find the location and the bounding box data of the people, around the time of when the object appeared in the video. If the bounding box of the object intersects almost completely with the bounding box of a person, around that time, this means that it was that person that left the object there, and both, the object and the person, are flagged as suspicious.

The alert is only given if the person is at a certain distance from the object, the system should not flag as suspicious someone that is still close to the object, thus not leaving it unattended. The person must also have been tracked for longer than the object, otherwise, it could not belong to that person.

Figure 4.6: Person leaving object on floor.

Figure 4.7: Person stepping away from object.

$$\begin{aligned} \text{category}(det1) = \text{O} \wedge \text{lifetime}(det1) > t_{abandoned} \wedge \ \text{category}(det2) \in \{\text{P, SP}\} \\ \wedge \ \text{lifetime}(det2) > \text{lifetime}(det1) \wedge \text{distance}(det1, det2) > d_{abandoned} \\ \wedge \ \text{intersection\_area}(det1, det2) \approx det1\_area \end{aligned} \tag{4.3}$$

Figure 4.6 shows a person as they are a placing an object on the ground. Figure 4.7 shows a person that stepped away from the object but is still close to it. The bounding box of the object intersects entirely with the bounding box of the person just a few frames before, in Figure 4.6.

After moving away from the object, both the person and the object are flagged as being suspicious (Fig. 4.8).

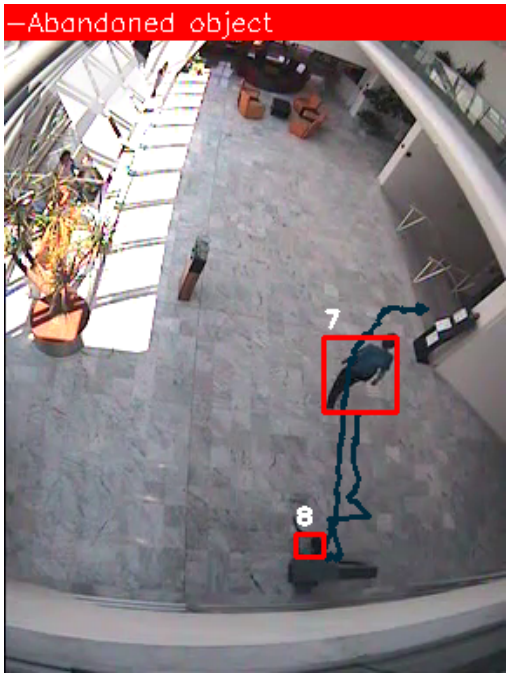


Figure 4.8: Person leaving object unattended, rendering both person and object as suspicious.

4.3 Abandoned Object Picked Up

A distinction is made between the act of leaving an object unattended and picking up an object left by someone. This type of behavior is important to identify in cases of illicit exchanges between people or stealing.

Later in the video displayed previously, another person is detected (Fig. 4.9) and picks-up the object (Fig. 4.10).

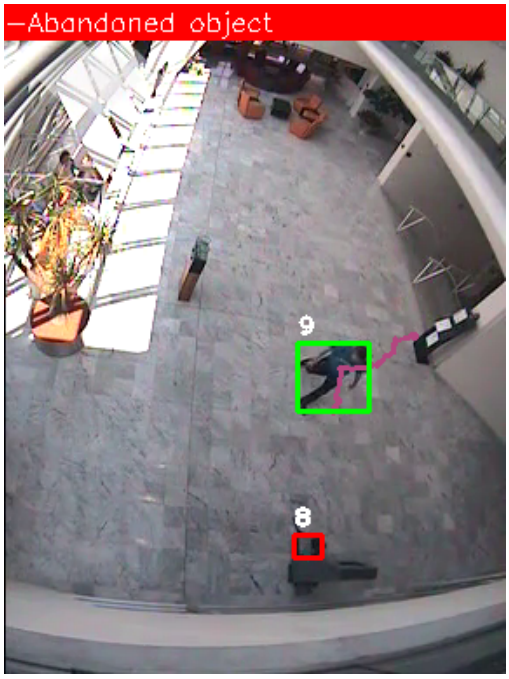


Figure 4.9: Another person is approaching the suspicious object.

Figure 4.10: Person and object merge together.

As soon as a suspicious object stops being detected the system will look for someone that might have picked up the object. If the bounding box of where the object used to be, now intersects almost entirely with the bounding box of a person, then this person is flagged as possibly being suspicious. It is not considered to be automatically a suspicious behavior because the person might be just passing by and have merged with the object momentarily. However, if the person leaves the location of the object and the object is not detected again, it means that the person picked-up the object indeed and it is flagged as suspicious (Fig. 4.11).

$$\begin{aligned} \text{category}(det1) = \text{O} \wedge \text{classification}(det1) = suspicious \\ \wedge\ \text{lost\_track}(det1) = \text{True} \wedge \text{intersection\_area}(det1, det2) \approx det1\_area \end{aligned} \tag{4.4}$$

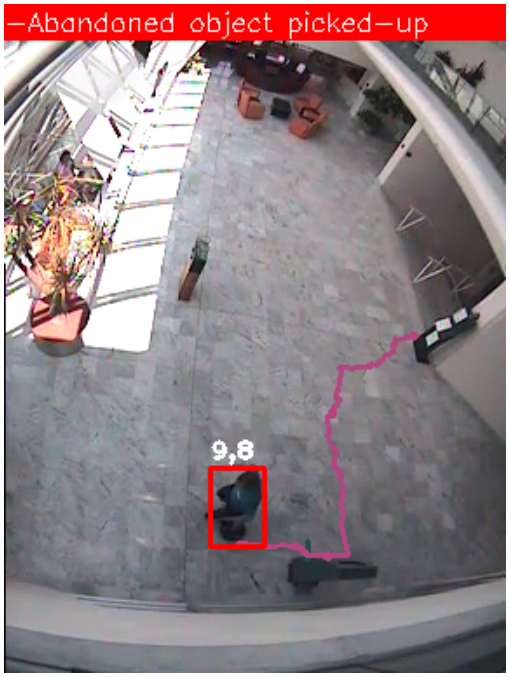


Figure 4.11: Person picks up suspicious object.

4.4 Loitering

Loitering is defined as the presence of someone in an area for longer than a defined period of time. This behavior might be suspicious if a person is waiting or observing with the intent of doing something illicit. The defined time period threshold will depend on where the system is applied, there might be places where it is normal for people to stand for a while, such as outside cafes or bars where people might just be smoking or having a drink. And there are other places such as besides an ATM machine, where, if a person stands there for a couple of minutes, they might not have good intentions.

Figure 4.13 displays a person that has been loitering in the lobby for a minute, this time threshold is not ideal but was chosen for testing purposes.

$$category \in \{P, SP\} \wedge lifetime > t_{loitering} \tag{4.5}$$

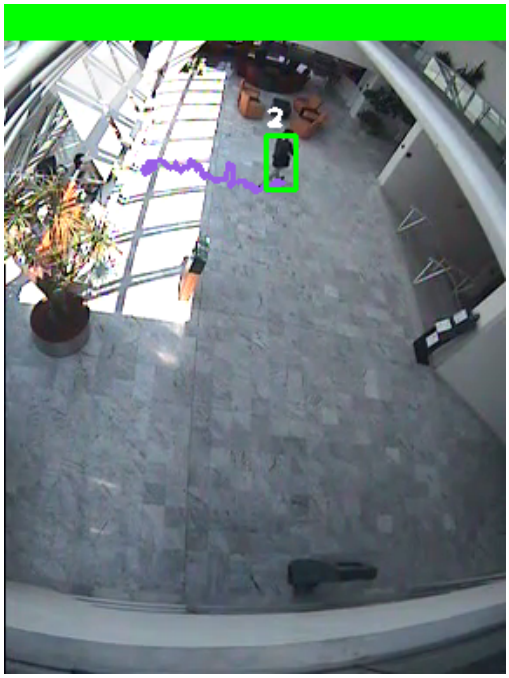


Figure 4.12: Person standing in lobby.

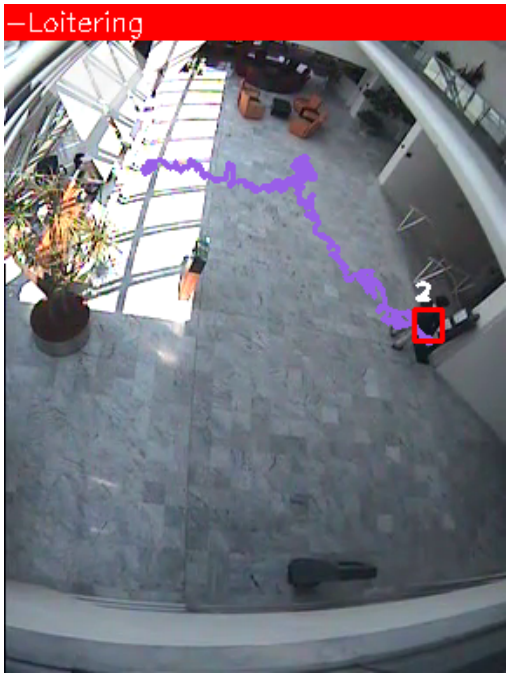


Figure 4.13: Suspicious behavior flagged as loitering.

4.5 Running

Running behavior can be classified in two categories, normal running and abnormal running. Normal running is usually observed when people are in a hurry or exercising. On the other hand, abnormal running is observed in robbery cases and other criminal activity [31]. The following two definitions explain the difference.

Definition 1 - Normal Running - A walking or stationary object maintains a constant acceleration over a long period of time before reaching and exceeding a set normal running velocity (Fig. 4.14). This behavior is referred to as Normal Running Behavior.

Definition 2 - Abnormal Running - A walking or stationary object suddenly accelerates before reaching and exceeding normal running velocity (Figure 4.15). This behavior often reaches greater velocity than the Normal Running Behavior and is referred to as Abnormal Running Behavior.

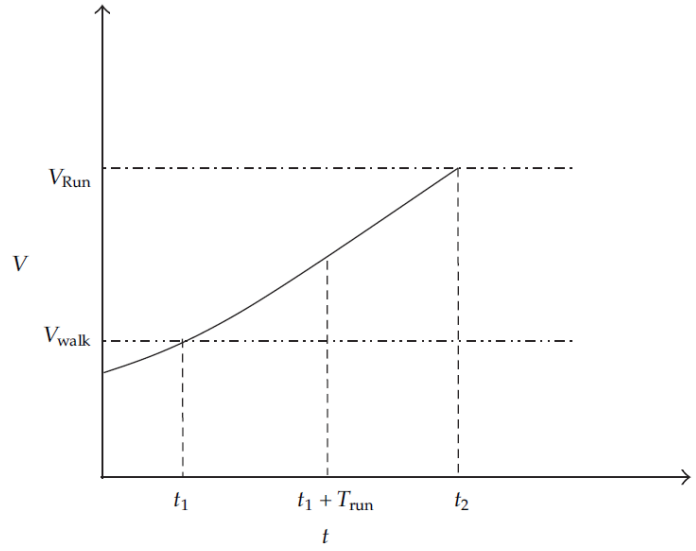


Figure 4.14: Normal Running [31].

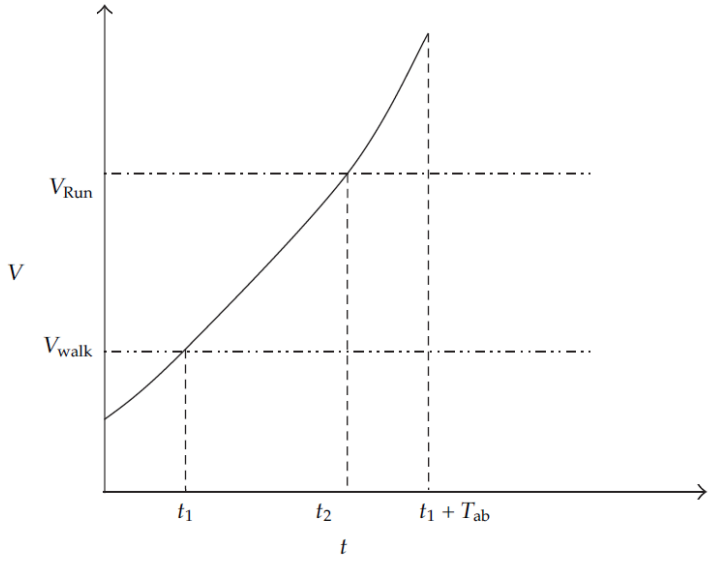


Figure 4.15: Abnormal Running [31].

Determining whether an observed running behavior is normal or abnormal by the above mentioned definitions is done by comparing the average velocity over the past five frames to a normal velocity threshold, if the average velocity is greater than the threshold the observed behavior is classified as running. In accordance

with *Definition 2*, to classify a running behavior as abnormal the acceleration leading to the running state needs to be rapid. To determine that the average acceleration over the past five frames is compared to a normal acceleration threshold. If the current acceleration exceeds the threshold the detected running behavior is marked as abnormal and hence suspicious.

$$velocity > v_{threshold} \wedge acceleration > a_{threshold} \tag{4.6}$$

Figure 4.16 displays a person, flagged in red, running away after hitting the person lying on the ground.

Figure 4.16: Person running after fight.

Chapter 5

Results

All of the behaviors were tested using the CAVIAR dataset [7]. This dataset contains videos obtained in the lobby of the INRIA Labs at Grenoble, France. The dataset contains behaviors such as browsing, fainting, abandoned bags, groups of people walking together and people fighting. It contains all the necessary behaviors to test the implementation, but in a very limited number. Because of this, the results exhibited might not be accurate representations.

The number of True Positives (TP), False Positives (FP), False Negatives (FN) and True Negatives (TN) were annotated for each behavior on each video. A True Positive means a suspicious behavior that was correctly identified, on the other hand, a False Positive represents a false alarm, an incorrect classification of a suspicious behavior, in contrast, a False Negative signifies a suspicious behavior that was wrongly classified as non-suspicious and a True Negative symbolizes a behavior correctly identified as non-suspicious.

Recall or True Positive Rate demonstrates the ability of the system to find all the suspicious behaviors within the dataset.

$$\text{recall} = \frac{TP}{TP + FN} \tag{5.1}$$

Precision expresses the proportion of behaviors that the system identified as suspicious that were actually suspicious behaviors.

$$\text{precision} = \frac{TP}{TP + FP} \tag{5.2}$$

False Alarm Rate or False Positive Rate shows the probability of a false alarm,

it is the ratio between the number of behaviors wrongly identified as suspicious and the total number of non-suspicious behaviors.

$$\text{false alarm rate} = \frac{FP}{FP + TN} \tag{5.3}$$

The accuracy demonstrates how close the number of detected suspicious behaviors is to the actual true number.

$$\text{accuracy} = \frac{TP + TN}{TP + FN + TN + FP} \tag{5.4}$$

5.1 Fainting

The system recognizes 66,7% of all the fainting occurrences but only 22,2% of the behaviors flagged as fainting are actual fainting occurrences. The reasons for such a low number are explained in Section 4.1.

- Recall = 66,7%
- Precision = 22,2%
- False Alarm Rate = 3,1%
- Accuracy = 96,5%

5.2 Abandoned Object

The True Positive Rate of 66,7% demonstrates that this percentage of abandoned objects are identified as such. A precision of 100% assures that all the objects identified as being abandoned are truly abandoned which relates to a False Alarm Rate of 0% and a very hight Accuracy.

- Recall = 66,7%
- Precision = 100%
- False Alarm Rate = 0%
- Accuracy = 99,5%

5.3 Abandoned Object Pickep Up

Only half of the times an abandoned object was picked up were detected by the system, however every detection was precise.

- Recall = 50%
- Precision = 100%
- False Alarm Rate = 0%
- Accuracy = 99,5%

5.4 Loitering

The system classified correctly and precisely every loitering episode.

- Recall = 100%
- Precision = 100%
- False Alarm Rate = 0%
- Accuracy = 100%

5.5 Running

The implemented system is capable of recognizing all the running occurrences, however, only approximately a quarter of the flagged running occurrences correspond to the truth, which relates to a relatively high false alarm rate.

- Recall = 100%
- Precision = 26%
- False Alarm Rate = 13%
- Accuracy = 88%

5.6 Existing Similar System

In the work developed by Elhamod and Levine in [8] the authors investigated behaviors of fainting, abandoned luggage, theft of luggage and loitering, among others. It is possible to relate abandoned luggage to the abandoned object behavior and the theft of luggage to the abandoned object picked up behavior investigated in the presented work.

Elhamod and Levine also calculate the recall and precision of each behavior in the same dataset as the one exhibited. However, they present the results for specific videos, which means these results relate to singular videos and not to the entire dataset as the results of this work do.

Table 5.1 exhibits the results from the similar system for each behavior specified for individual videos from the same dataset as the used in this work. The results from the table are mostly superior to the ones obtained from the discussed implementation. However, these results are only indicative as they were obtained in different ways, which might not have a straightforward comparison.

Suspicious Behavior	**Recall**	**Precision**
Fainting	100%	80%
	80%	69%
	100%	67%
Abandoned luggage	89%	77%
	93%	82%
	81%	100%
Theft of luggage	100%	100%
Loitering	98%	100%

Table 5.1: Results from [8].

Chapter 6

Conclusion

Overall, the developed system has a good performance and is able to identify correctly most of the suspicious behaviors in the videos. Still, some aspects could be further improved in order to achieve better results.

The occlusion handling algorithm, though working acceptably and being able to re-identify most people after an occlusion situation, still shows cases where the re-identification does not work optimally. This algorithm could benefit from using another visual descriptor, besides the color histogram, to identify the people affected by the occlusion. This descriptor could be, for example, LBP (Local Binary Pattern) which is a texture descriptor, or HOG (Histogram of Oriented Gradients) which is a descriptor based on shape. By combining more descriptors it is possible that the re-identification would improve in the cases where it fails.

Another suboptimal aspect of the presented implementation is the classification of the fainting suspicious behavior. The results show that most of the behaviors classified as fainting are not actual fainting behaviors. This has greatly to do with the camera set up which, because it is a video recorded from above the people, their BLOBs shown from above are sometimes not actual representations of the person that they correspond to. Nevertheless, it should be expected that the algorithm can detect suspicious behaviors regardless of the camera set up and the environment. More research should be done on the most optimal ways of identifying this type of behavior.

The suspicious running is another behavior which could benefit from more research into better ways of classifying this type of behavior. The implemented algorithm is very sensitive to the applied thresholds which were tested based on trial and error. A most optimal way of defining the thresholds should be investigated.

The other behaviors' classification show good results, however, because the dataset had a very limited number of behaviors to test, the obtained results in Chapter 5 might not be accurate representations of the truth.

Bibliography

[1] Wiliem, Arnold, Madasu, Vamsi K., Boles, Wageeh W., Yarlagadda, Prasad K. *A suspicious behaviour detection using a context space model for smart surveillance systems.* Computer Vision and Image Understanding, 116(2), pp. 194-209, 2012.

[2] M. Markou and S. Singh. *Novelty detection: A review-part 1: Statistical approaches*. Sig. Proc. 83, 12, 2481–2497, 2003.

[3] M. Markou and S. Singh. *Novelty detection: A review-part 2: Neural network based approaches*. Sig. Proc. 83, 12, 2499–2521, 2003.

[4] C. Varun, B. Arindam, K. Vipin. *Anomaly Detection: A survey*. ACM Comput. Surv. 41 (3), 2009.

[5] Wells, Helene. A., Allard, Troy, Wilson, Paul. *Crime and CCTV in Australia: Understanding the Relationship*. Centre for Applied Psychology and Criminology: Bond University, Australia, 2006.

[6] A. Wiliem, V. Madasu, W. Boles and P. Yarlagadda. *A suspicious behaviour detection using a context space model for smart surveillance systems*. Computer Vision and Image Understanding, Elsevier, October 2011.

[7] EC Funded CAVIAR project/IST 2001 37540.
`http://homepages.inf.ed.ac.uk/rbf/CAVIAR/`

[8] M. Elhamod and M. D. Levine. *Automated Real-Time Detection of Potentially Suspicious Behavior in Public Transport Areas*. IEEE Transactions on Intelligent Transportation Systems, vol. 14, no. 2, June 2013.

[9] L. M. Fuentes and S. A. Velastin. *Tracking-based event detection for CCTV systems*. Pattern Anal. Appl., vol. 7, no. 4, pp. 356–364, December 2004.

[10] M. Elhamod and M. D. Levine. *A real time semantics-based detection of suspicious activities in public scenes*. Proc. 9th Conf. CRV, Toronto, ON, Canada, pp. 268–275, 2012.

[11] N. D. Bird, O.Masoud, N. P. Papanikolopoulos, and A. Isaacs. *Detection of loitering individuals in public transportation areas*. IEEE Trans. Intell. Transp. Syst., vol. 6, no. 2, pp. 167–177, June 2005.

[12] N. Bird, S. Atev, N. Caramelli, R. Martin, O. Masoud, and N. Papanikolopoulos. *Real time, online detection of abandoned objects in public areas*. Proc. IEEE ICRA, pp. 3775–3780, 2006.

[13] S. J. Blunsden, R. B. Fisher. *The BEHAVE video dataset: ground truthed video for multi-person behavior classification*. Annals of the BMVA, Vol 2010(4), pp 1-12, 2010.

[14] PETS-ECCV. 2004.
`http://www-prima.imag.fr/PETS04/caviar_data.html`

[15] PETS 2006 Benchmark Data. 2006.
`http://www.cvg.rdg.ac.uk/PETS2006/data.html`

[16] C. Mu, J. Xie, W. Yan, T. Liu and P. Li. *A fast recognition algorithm for suspicious behavior in high definition videos. Multimedia Systems*. Vol. 22, Iss. 3, 2016-6, p. 275–285, June 2016.

[17] K. Schindler and L. v Gool. *Action snippets: How many frames does human action recognition require?* IEEE Conference on Computer Vision and Pattern Recognition, 2008. CVPR 2008.

[18] A. Gilbert, J. Illingworth and R. Bowden. *Fast realistic multi-action recognition using mined dense spatio-temporal features*. Conference on Computer Vision, 2009 IEEE 12th International, 2009.

[19] A. Yao, J. Gall and L. V Gool. *A Hough Transform-Based Voting Framework for Action Recognition*. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2010.

[20] S. Sadanand and J. J. Corso. *Action Bank: A High-Level Representation of Activity in Video*. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), July 2012.

[21] R. Arroyo, J. J. Yebes, L. M. Bergasa, I. G. Daza and J. Almazán. *Expert video-surveillance system for real-time detection of suspicious behaviors in shopping malls*. Expert Systems with Applications, Elsevier, June 2015.

[22] T. Zhao and R. Nevatia. *Tracking multiple humans in crowded environment*. Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004., Vol. 2, 2004, p. II–406-II-413 Vol.2, July 2004.

[23] B. Wu and R. Nevatia. *Tracking of Multiple, Partially Occluded Humans based on Static Body Part Detection*. IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 1 (CVPR'06), Vol. 1, 2006, p. 951–958, July, 2006.

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

[25] P. KaewTraKulPong and R. Bowden. *An Improved Adaptive Background Mixture Model for Realtime Tracking with Shadow Detection*. Remagnino P., Jones G.A., Paragios N., Regazzoni C.S. (eds) Video-Based Surveillance Systems. Springer, Boston, MA, 2002.

[26] M. H. Hung, J. S. Pan and C. H. Hsieh. *Speed Up Temporal Median Filter for Background Subtraction*. 2010 First International Conference on Pervasive Computing, Signal Processing and Applications, Harbin, pp. 297-300, 2010.

[27] ObedAppiah and J. B. Hayfron-Acquah. *A Robust Median-based Background Updating Algorithm*. I.J. Image, Graphics and Signal Processing, 2, 1-8, 2017.

[28] R. E. Kalman. *A new approach to linear filtering and prediction problems*. Transactions of the ASME–Journal of Basic Engineering, 82, 35–45. 1960.

[29] R. Gade and T. B. Moeslund. *Thermal Tracking of Sports Players*. Sensors, 14(8), 13679-13691. DOI: 10.3390/s140813679, 2014.

[30] W. Lu and Y. Tan. *A color histogram based people tracking system*. The 2001 IEEE International Symposium on Circuits and Systems (Cat. No.01CH37196), Sydney, NSW, 2001, pp. 137-140 vol. 2. doi: 10.1109/ISCAS.2001.921025. ISCAS 2001.

[31] Ying-Ying Zhu, Yan-Yan Zhu, Wen Zhen-Kun, Wen-Sheng Chen, and Qiang Huang. *Detection and Recognition of Abnormal Running Behavior in Surveillance Video*. Mathematical Problems in Engineering, vol. 2012, Article ID 296407, 14 pages, 2012.