Unmanned object detection using computer vision

Pavithran Ravichandiran
University of Wisconsin-Madison
pravichandir@wisc.edu

Lincolnspartacus James
University of Wisconsin-Madison
spartacus@cs.wisc.edu

ABSTRACT

Use of computer vision for security purposes has been gaining traction in recent years. One very important application of computer vision in the security systems is video surveillance. A video surveillance system observes the scene in the image area and tries to identify the abnormal activities and alerts the concerned authorities by triggering notifications for any situation that requires manual intervention. This research area is of notable significance as timely detection of illegal activities can save countless lives across the globe.

Detecting an object that has been carried by a human into the scene and suddenly left unattended is an important problem in visual surveillance research. Since the spectrum of suspicious objects is broad, we can use general purpose object detection algorithms to identify these objects in the scene. We aim to develop a system that initially detects the static foreground objects and then analyzes the back-traced trajectories of object owners to decide whether the object is left unattended or not.

PVLDB Reference Format:
Pavithran Ravichandiran and Lincolnspartacus James. Unmanned object detection using computer vision. PVLDB, 14(1): XXX-XXX, 2021.
doi:XX.XX/XXX.XX

1 INTRODUCTION

In public places like railway stations, airports there may be scenarios where a person enters a scene with an object, places the object that may be suspicious and leaves the scene after placing the object. There are incidents where a vehicle is parked in a no parking zone. These kinds of objects are difficult to identify in a video scenario as they change from a moving position to a static position. There are many algorithms to find the moving foreground objects and also the objects that are static from the beginning of the scene. The algorithms that detect the objects that change from moving to stationary either do not detect the objects accurately or add more overhead in time and memory. Also, there may occur a scenario where a person places an object accidentally and then returns back to take the object. These kinds of scenarios should not result in an alarm, so it requires a tracking of the owner of the object. After the verification of the owner and if the owner does not return to the object only then the alarm should be raised.

Proceedings of the VLDB Endowment, Vol. 14, No. 1 ISSN 2150-8097.
doi:XX.XX/XXX.XX

The objectives of the proposed work include:

- Find the static foreground object that changes its state from moving to static with less overhead of time and accurately when compared to the current state-of-the-art work.
- Track the owner of the candidate static object identified to verify that owner does not return to the object and raise an alarm.

2 PREVIOUS WORK

Stationary foreground objects [1] are detected by using a finite state machine(FSM) approach where three detector results are passed on to a FSM to classify the pixels between background and foreground objects. Short, medium and long term detectors are used to update the background at different rates. Based on the update status on objects in each detector, the objects are classified as either foreground or background objects. The three detectors which are used for detecting the different types of objects make a disadvantage as they have processing and time overhead.

The survey [2] gives a good insight on various stationary foreground object detection techniques. The methods discussed in the survey are used in objects that become stationary completely or only for some amount of time. The survey mentions background subtraction as the best technique for stationary object detection as it compares the previous frames with the current frame and identifies granular differences in the pixels. The survey acts as a premise for implementing background models with different absorption rates to identify the foreground objects.

Suspicious objects [3] are detected by different morphological and thresholding techniques. The video frames are taken one at a time and thresholding techniques are applied to separate the background and foreground objects. Some techniques involve multi-level thresholding and histogram. To get a clear picture of the detected regions different morphological techniques like erosion / dilations are used. The main advantage is the ability to distinguish foreground and background objects in the video. However, this method suffers due to its performance as it takes longer time to process as the video quality diminishes and the method does not perform verification of the owner.

3 CONTRIBUTION AND METHODOLOGY

The time taken in the work [1] by three detector models - long term, medium term and short term is directly proportional to the number of models used and it results in 8 different states in a finite state machine. The proposed work reduces the model to two detectors, short and long term detectors, thus improving the performance and reducing the number of states in FSM to identify static foreground objects without losing much of its accuracy.

The unmanned object detection system starts by identifying a candidate static foreground region and then to analyse the region

for unmanned object. Identifying a static object is difficult as there may be many objects surrounding an object that has changes its state from a moving to static object which is the main area of interest. The steps involved in identifying the static foreground object are shown in Figure 1. After selecting the region of interest the input video is fed into two background models namely Short and Long Learning models with different learning rates which learn the background at different updation rates. The result of these models are subsequently passed through a Finite State Machine the result of which shows whether there is any candidate static object or not. The states of the Finite State Machine represent the different states of the object at each stage.

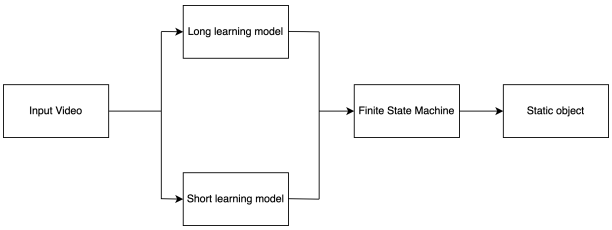

Figure 1: Architecture Overview

Figure 2: ROI Set Window

4 IMPLEMENTATION DETAILS

4.1 Region of Interest Selection

In a typical surveillance video without pre-processing there may be lots of unwanted spaces which also comes under the processing area of the algorithm. The algorithms generally capture every pixel in the video and do analysis even if there is no useful information available. In order to avoid this overhead the video is analysed only for selective regions as per the interest. For example, a surveillance camera placed in a no parking area may also cover the roads nearby which may cause unwanted overhead in the algorithm identifying the objects. This can be avoided by selecting only the no parking area and monitoring it. Given the input video four points are selected in the order of the shape required for the ROI. The region is formed by joining the points selected by the user in the direction specified. For example, in order to make a rectangle as the ROI the top left and right corners followed by bottom right and bottom left are made in order to from the ROI. Selecting the ROI reduces the overhead and also the memory and time costs.

Figure 3: ROI Mask

The PETS 2006 dataset is taken as the input and fed into the algorithm. The first step is selecting the region of interest for efficient usage of algorithm. The first step consists of a ROI setting window shown in Figure 2. The corresponding shape of the ROI mask region chosen and the points selected are shown in Figure 3. These are the points that are joined together to form the ROI.

4.2 Background Subtraction

A widely deployed object identification technique is the background subtraction algorithm of Gaussian Mixture Model. The Gaussian Mixture model is the primary algorithm for detecting moving objects in a video because of its ability to detect various scenarios in a video. Each pixel in this method is made by a separate Gaussian mixture that is learnt continuously as the video proceeds. This method is used the most because of its ability to handle the changes in lightning etc. Moving objects can be identified using this method but in order to identify objects that come from a moving to a static state an extension is needed. To use the extended algorithm the objects must attain a static condition from a moving condition. First the generic Gaussian mixture model is used to detect the moving objects then the extension is added.

Foreground Detection or background subtraction is a technique used to identify objects in the foreground of a video. This technique makes use of a background model based on pixels that is learnt sequentially from the previous images of the input video. Using the learnt model the pixels of the incoming images can be classified as either background or foreground pixels. If the pixel is found as a background pixel, the features of the pixel such as colour can be used to update the model so that the model is very recent. A general algorithm used for detection works as follows given a sequence of images of size x * y.

Pixel Values P_i	Pixel Type
00	Background pixel
01	Occluded that is exposed in a recent image
10	Likely to be Static Object
11	Moving Objects pixel

Table 1: PIXEL TYPES

(1) For all the pixels of the incoming input image, a Background model B is created.
(2) If the pixel (m,n) of the image I ϵ B (m,n), then the pixel is a background pixel else it is a foreground pixel.
(3) The background model is updated for all the background pixels identified.
(4) The next image is iterated and Step 2 is followed

Every pixel in the Gaussian model is made as a mixture of m Gaussian distributions. Each pixel has the following value observed in them.

$$V(Y_t) = \sum_{j=1}^{m} w_{j,t} * P(Y_t, u_{j,t}, T_{j,t}) \quad (1)$$

Where Y_t represents the pixel value in gray scale, m specifies the number of distributions of Gaussian used. The weight of the ith distribution at t time in the Gaussian is denoted by $w_{j,t}$, $u_{j,t}$ is the mean value of the Gaussian distributions and P denotes the density function of Gaussian. The matrix of co-variance is denoted by $T_{j,t}$. Initially all M distributions are considered as pixels that form the background. At t time, if the current pixel is not matched by any of M distributions, it will replace one of above M distributions; a weight with lowest value will be replaced by the above one and every other weight is changed. The pixel's weight will increase if any of the distributions matches the pixel's distribution. Initially M distributions are classified as either foreground or background by their weight. B denotes the number of background models at time t. A pixel is foreground if none of them matches to the first b distributions else it becomes a pixel which is background. The dynamic changes in the video updated as per this rule.

$$w_{m,t} = w_{m,t-1} + \mu(K_{m,t} - w_{m,t-1}) \quad (2)$$

where the learning rate and $K_{m,t}$ is 0 for the non matching pixel's and 1 for the remaining models. This rule is the key factor in detecting the static object. In detecting moving objects the rule is useful and it makes even the objects that move from a moving state to a static object to be attached into the background. To detect the static object an extended Gaussian Mixture Model is used.

In order to update the model as specified in Step 3 of the generic background model a learning rate μ is used. This learning rate provides the difference between the various models learnt. So the update of the model depends on the learning rate μ.

Figure 4 shows the initial updates on the background model that takes place for a period of 500 frames. After each frame is processed the background model is updated.

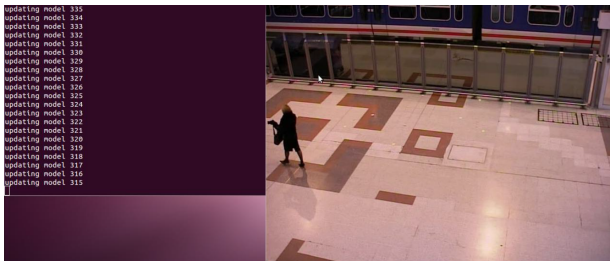

Figure 4: Model Updation

5 FUTURE WORK

The extension proposed to identify the static foreground objects proceeds with Gaussian Mixture model by building two models that are generated at different learning rates. A model that learns and updates quickly is called a short learning model and the model that learns and updates slowly is a long learning model. The usage of both the models can be used to detect the stationary foreground object as the long learning model would make the stationary object as a foreground object as it updates at a slower speed while the short learning model considers it as a background object.

5.1 Long Term and Short Term Detectors

$$P_i = M_L(i) * M_S(i) \quad (3)$$

Let M_L and M_S be the models built using longer and short rates. A pixel i is represented as the combination of two models as represented in equation 3.

The values of M_L and M_S either 0 or 1 depending on background or foreground pixel. We can classify the pixels based on the value of P_i as mentioned in Table 1.

(1) When both long and short term models are 0 i.e P_i = 00 it shows a pixel that is a background one.
(2) When both long and short term models are 1 i.e P_i = 11 it shows a pixel that is a moving foreground object.
(3) When the long term is 0 and short term model is 1 i.e P_i = 01 it shows a pixel that is occluded by an object temporarily and which is shown in a recent frame.
(4) When the long term is 1and short term model is 1 i.e P_i = 10 it shows a pixel that is likely to be a static object.

5.2 Finite State Machine

As videos suffer from noises the codes can be temporary so this is why detection based on single images fail. Rather than using the pixel status of each image, sequential information of all the images is used to identify the stationary foreground object. Therefore the static object detection algorithm involves combining the short and long rate learning models and then sending them via a finite state machine that identifies the type of object eventually finding the static object. An image pixel can be of only one type at a time t. The state of the pixel i can be changed from time t to time t+1 based on the two models that are short and long learning rate models. Therefore the finite state machine's result depends on the pixel's combined long and short term value. The static object is detection based on a particular pattern that appears in the video.

Phases	Actions and Goals	Deadline
Phase 1	Region of interest selection	Completed
Phase 2	Implementing Gaussian mixture model	Completed
Phase 3	Short and Long term detector model	Apr 7
Phase 4	Defining and implementing finite state machine to detect static foreground objects	Apr 15
Phase 5	Running performance evaluation and surveillance dataset	Apr 25
Phase 6	Webpage completion	May 4

The FSM consists of a start state and the machine is started only when a moving object pixel is identified i.e. $P_i = 11$ occurs. This is because the main aim is to detect unmanned objects. An object becomes unmanned only when it moves from a moving to a static state. Therefore the machine should start in this state. The machine remains in start state for all the other pixel types like moving object, background pixel and temporarily occluded object. Next when an object is left unmanned the short rate model updates the object into the background model quickly as it learns quickly and the other model does not update as it learns slowly This leads to a change in pixel state as $P_i = 10$. Therefore when this pixel state arrives the FSM moves to the next state. When this state remains for a particular amount of time then the pixel can be considered as being part of the static region. This is because only when an object is static the FSM stays in the same state. Else the FSM moves back to the previous state when any other pixel type comes. This scenario occurs when the static object becomes a moving object again. When the final state is reached, only those pixels that are part of the transition are considered as static.

The Finite State Machine states the following rules when a pixel that is represented by a two bit code is given. If there is a large sequence starting with 11 and continued by a further long sequence of 10 the associated pixels form the static foreground. These pixels are collected for further verification. If none of the pixels reach the final state of the machine there is no static foreground and therefore no verification is required. The figure 5 represents the Finite State Machine. By using this FSM the candidate static object is identified.

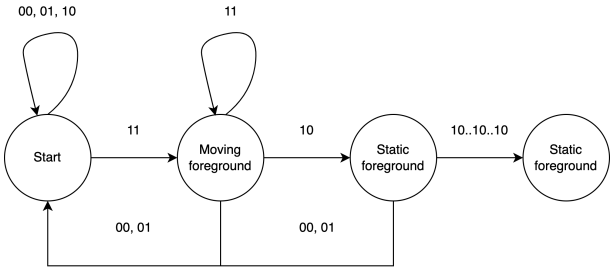

Figure 5: Finite State Machine

6 PROPOSED TIMELINE

The proposed timeline of the project is shown in the above table.

REFERENCES

[1] C. Cuevas, R. Martínez, D. Berjón, and N. García. Detection of stationary foreground objects using multiple nonparametric background-foreground models on a finite state machine. *IEEE Transactions on image processing*, 26(3):1127–1142, 2016.

[2] C. Cuevas, R. Martinez, and N. Garcia. Detection of stationary foreground objects: A survey. *Computer Vision and Image Understanding*, 152:41–57, 2016.

[3] T. M. Pandit, P. Jadhav, and A. Phadke. Suspicious object detection in surveillance videos for security applications. In *2016 International Conference on Inventive Computation Technologies (ICICT)*, volume 1, pages 1–5. IEEE, 2016.