A Survey on Moving Object Detection Methods in Video Surveillance

Vishakha Sharma¹, Neeta Nain² and Tapas Badal³
Department of Computer Engineering
Malaviya National Institute of Technology
Jaipur-302017 (Rajasthan), INDIA.
Email: ¹vishakhamini1992@gmail.com, ²nnain.cse@mnit.ac.in, ³tapasbadal@gmail.com

Abstract

Moving Object detection is one of the key steps for activity analysis in video surveillance. It provides a classification of the pixels into either foreground or background. Various methods have been proposed by researchers for segmenting out foreground objects in a video sequence, each having their own merits and demerits. A good method should be robust to illumination changes, non-static background and camera noise. This paper presents a survey study of the six most widely used moving object detection methods: Frame differencing, Approximate median filter, Running Gaussian average, Gaussian mixture model (GMM), Optical flow, and ViBe (Visual Background Extractor). A comparative study is also shown to help experts choosing the method which best suits their application needs.

Keywords: Video Surveillance; Object Detection; Object Tracking; Background Subtraction

1 INTRODUCTION

Video-based surveillance systems are used in monitoring of scenes for different applications. Video streams can be exploited for extraction of information based on our interest such as security, safety, entertainment, and efficiency improvement applications. One such application of video surveillance is in Activity Recognition. Recognizing events from a field of interest can then be used for other applications, such as traffic analysis, restricted vehicle movements, multi-object interaction, etc. It helps in solving many issues which otherwise would need continuous human monitoring. The first key step to this calls for detecting the motion in video samples. The moving objects should not only be segmented from the video stream, but the method should also be free from noise.

One of the main challenges to moving object detection is sudden illumination changes (light switch). Unwanted noise will then be included in the result if the algorithm could not cope with the lighting changes and camera shaking. Non-static backgrounds would further increase the problem (moved objects). Waving trees and weather changes could result in improper result for the detection stage. A moving object may come to stop for a moment and may diffuse in the background (sleeping person, waking person). All these challenges should be met by a moving object detection method.

This paper presents a complete study of these widely used moving object detection methods. The paper is organized as follows: Section 2 includes the study of the methods, Section 3 presents the results of each method. Finally, Section 4 gives the conclusion.

2 MOVING OBJECT DETECTION METHODS

Each of the methods mentioned have been studied deeply listing their advantages and limitations. The methods surveyed in the paper includes:

- Frame differencing
- Approximate median filter
- Running Gaussian average
- Gaussian Mixture model
• Optical flow
• ViBe (Visual Background Extractor)

2.1 Frame Difference Method

This is the simplest of all the background subtraction techniques used to model a background. It is a non-recursive technique [1] where no history of the video frames are required, as buffers may be used to store the past N frame values.

In this method, moving objects are recognized by subtracting the current video frame from the previous frame. This difference is then compared to a threshold to determine the object as background and foreground [41].

Let us denote the intensity value of a pixel \( p_{x,y} \) at time \( t \) as \( I_{x,y,t} \). Then according to the method, the difference between the frame at time \( t \) and the frame at time \( t-1 \) is determined as follows:

\[
D = | I_{x,y,t} - I_{x,y,t-1} | \quad (2.1)
\]

For each pixel, the value of \( D \) is compared to a threshold \( T \) and classified as follows:

\[
p_{x,y} = \begin{cases} 
\text{foreground} & \text{if } D > T \\
\text{background} & \text{else} 
\end{cases} \quad (2.2)
\]

Here, choice of a threshold value is an important consideration. A too low value of \( T \) will add unwanted noise, while a too high value may classify a foreground pixel as background.

Variant:

A variant on this method adapts to the changes in background and illumination [39][40], in which the reference frame is updated at each iteration as follows:

\[
B_{t+1} = (1 - \alpha) B_t + \alpha I_{x,y,t} \quad (2.3)
\]

Where \( \alpha \) may range from 0 to 1.

Limitations:

It has low complexity and high computational efficiency [20]. All the pixel values are modelled on a single threshold value, but some pixels may exhibit more variations than the other pixels. For those moving objects having some large uniform color area, the pixel values are determined as background showing some missing areas of object. This is called the aperture problem. Also, its accuracy is dependent on the frame rate of the video and speed of the moving object. [42]

2.2 Approximate Median Filter

Non-Recursive technique:

One of the commonly used background modelling techniques is median filtering [29][30]. Lo and Velastin [2] proposed this method. Here, the background model is taken as the median of the last \( n \) frames. It is based on the assumption that a pixel belongs to background for at least half of the frames in the buffer.

A buffer is maintained of size \( n \) to store the recent \( n \) pixel values of the past \( n \) frames. Median is then calculated from values present in the buffer. This median value acts as a reference value for the next input frame. Foreground pixels are determined by calculating the difference of the current pixel value to the median value as follows:

\[
p_{x,y,t} = \begin{cases} 
\text{foreground} & \text{if } | I_{x,y,t} - Med_{x,y,t} | > Th \\
\text{background} & \text{else} 
\end{cases} \quad (2.4)
\]

Where \( Med_{x,y,t} \) is the median value of the buffer at time \( t \), \( I_{x,y,t} \) is the intensity value of \( p_{x,y,t} \) at time \( t \), \( Th \) is the threshold value.

The median value is updated for each new pixel frame.

It has the disadvantage of maintaining a buffer in memory of recent pixel values.

Recursive technique:

McFarlane and Schofield [3] proposed the recursive technique for median filtering which does not maintain a buffer. Instead, the background model is updated recursively for each new input frame [31][32].
In this method, if the input pixel is larger than the background model, the median is incremented by 1. If it is less than the estimate, the median is decremented by 1, otherwise it remains same. It uses the following update equation:

\[ B_{t+1} = \begin{cases} 
B_t + 1 & \text{if } I_{x,y,t} > B_t \\
B_t - 1 & \text{if } I_{x,y,t} < B_t \\
B_t & \text{if } I_{x,y,t} = B_t 
\end{cases} \]  

This will give a value which is larger than half of the pixels and smaller than half of the rest pixels. The value thus obtained is the median of all the pixel values.

**Advantages:**
This approach is simple, computationally efficient, and robust to noise. The recursive technique requires less space as compared to non-recursive technique.

**Limitations:**
As seen by Cheung and Kamath, it has the drawback of adapting to a background slowly when there is a large change in the background. Thus, if a long time stationary object starts moving suddenly, it may show as faded into background before it gets many frames to learn the new background region.

Also, the variance of a pixel is not modelled.

### 2.3 Running Gaussian Average

This method was proposed by Wren. The running gaussian average model consists of fitting a gaussian probability density function (pdf) on the last \( n \) pixel’s intensity values. This would require to fit the pdf from scratch on each pixel at the time of each new input frame. In order to avoid this, a running average method is used.

The pdf of each pixel has two parameters: mean(\( \mu \)) and variance(\( \sigma^2 \)). Initially, mean and variance for the first input frame is set as follows:

\[ \mu = I_0 \]  
\[ \sigma^2 = V \]

Where, \( I_0 \) is the intensity value of first initial frame, \( V \) is any default value generally taken as 36.

At each new frame, the mean is updated as:

\[ \mu_t = \alpha I_t + (1 - \alpha)\mu_{t-1} \]

Where \( I_t \) is the current pixel intensity value at time \( t \), \( \mu_t \) is the previous average, \( \alpha \) is the empirical weight which ranges from 0 to 1.

Similarly the variance is updated as:

\[ \sigma^2_t = d^2\alpha + (1 - \alpha)\sigma^2_{t-1} \]

\[ d = |I_t - \mu_t| \]

Where \( d \) is the Euclidean distance.

The foreground pixels are then determined for each frame if the following condition holds true:

\[ p_{x,y,t} = \begin{cases} 
\text{foreground} & \text{if } |I_t - u_t| > k\sigma \\
\text{background} & \text{if } |I_t - u_t| \leq k\sigma 
\end{cases} \]

where \( k \) is generally taken as 2.5

**Variant:**

Koller et al. introduced a variant of the method which updates the mean only when the corresponding pixel is characterized as background. This prevents newly introduced moving objects from fading into background. The model is updated as:

\[ \mu_t = M\mu_{t-1} + (1 - M)(\alpha I_t + (1 - \alpha)\mu_{t-1}) \]

Where \( M = 1 \) if \( I_t \) is classified as foreground, \( M = 0 \) if \( I_t \) is classified as background.
Advantages:

It has computationally good speed and low memory requirement.

Limitations:

It cannot cope with multi-modal backgrounds. Backgrounds with waving trees and wall behind are the multi-modal backgrounds. Such scenes and similar will be incorrectly classified as foreground regions [22].

2.4 Gaussian Mixture Model

This method is used when there are multi-modal backgrounds. Multiple surfaces form part of a background, and thus multiple gaussians are necessary [25]. Stauffer and Grimson [9] proposed this method by modelling each pixel as a mixture of multiple gaussians.

A “pixel process” is considered which contains the pixel’s history as the intensity values till time $t$:

$$X_1 \ldots X_t = \{ I(x_0, y_0, i) : 1 \leq i \leq t \}$$

(2.13)

where $I$ is the frame sequence.

This history of a pixel at time $t$ is modelled by a mixture of $K$ Gaussian distributions. The probability of occurrence of the current pixel value is:

$$P(X_t) = \sum_{i=1}^{K} \omega_{i,t} \times \eta \left( X_t, \mu_{i,t}, \Sigma_{i,t} \right)$$

(2.14)

where $K$ is the number of gaussian distributions, $\omega_{i,t}$ is the weight estimate of the $i^{th}$ Gaussian in the model at time $t$, $\mu_{i,t}$ is the mean value of the $i^{th}$ Gaussian in the model at time $t$, $\Sigma_{i,t}$ is the covariance matrix of the $i^{th}$ Gaussian in the model at time $t$, and $\eta$ is a Gaussian probability density defined as follows:

$$\eta(X_t, \mu, \Sigma) = \frac{1}{(2\pi)^{n/2} |\Sigma|^{1/2}} e^{-\frac{1}{2} (X_t - \mu)^T \Sigma^{-1} (X_t - \mu)}$$

(2.15)

$K$ is taken as 3 to 5 depending on available memory. Assuming the red, green and blue pixels as independent of each other for computational reasons, the covariance matrix reduces to diagonal:

$$\sum_{k,t} = \sigma^2 I$$

(2.16)

An on-line $K$-means approximation is used where every new pixel value, $X_t$, is checked against the existing $K$ Gaussian distributions, until a match is found. A pixel is said to be matched if its value lies within 2.5 standard deviation of a distribution. If there is no match for all of the $K$ distributions, the least probable distribution having the lowest weight is replaced with the current value as its mean value, an initially high variance, and low prior weight. The weights for $K$ distributions are updated as follows:

$$\omega_{k,t} = (1 - \alpha) \omega_{k,t-1} + \alpha(M_{k,t})$$

(2.17)

Where $\alpha$ is the learning rate between 0 to 1.

The parameters for the matched component is updated as follows:

$$\mu_t = (1 - \rho) \mu_{t-1} + \rho X_t$$

(2.18)

$$\sigma^2 = (1 - \rho)\sigma^2_{t-1} + \rho(X_t - \mu_{t})^T(X_t - \mu_{t})$$

(2.19)

Where $\rho$ is another learning parameter approximated as:

$$\sigma / \omega_{k,t}$$

(2.20)

The Gaussians are ordered by the value of $\omega_{k,t}/\sigma$. Then simply the first $B$ distributions are chosen as the background model:

$$B = \arg\min_b \left( \sum_{k=1}^{b} w_k \geq T \right)$$

(2.21)
where $T$ is a threshold. Then, those pixels whose color $I_{x,t}$ is located at more than 2.5 standard deviations away from every $B$ distributions are labelled foreground.

**Advantages:**
A different threshold is selected for every pixel. This threshold is not global and adapts with time for each pixel. It has high accuracy\(^{26}\).

**Limitations:**
Its parameters require careful tuning and should be selected intelligently. It is computationally intensive. It cannot deal with sudden drastic changes in illumination\(^{28}\).

### 2.5 ViBe-Visual Background Extractor

O. Barnich and M. Van Droogenbroeck\(^{10}\)[11] proposed the foreground object segmentation algorithm ViBe which stands for Visual Background Extractor. ViBe has been well explored in the literature\(^{33}\)[34][35].

**Pixel Model and Classification Process:**
Contrary to the background models which are based on probability distribution function, the background model in ViBe consists of a set of observed pixel values.

Let the value of a pixel located at $x$ in a given color space be denoted as $v(x)$, and $v(i)$ as the $i^{th}$ sample of the background model. Then for each pixel $x$, the background model is defined as the collection of $N$ background sample values:

$$M(x) = \{v_1, v_2, ..., v_N\}$$  \hspace{1cm} (2.22)

Classification of a pixel is done by defining a sphere $S_R(v(x))$ of radius $R$ having centre at $v(x)$ point as shown in figure 2.1. The pixel is compared with the closest value (computed as Euclidean distance) from this sphere samples. A minimum cardinality is priory set denoted as $\#_{\text{min}}$. A pixel is considered as background if at least $\#_{\text{min}}$ sphere samples matches to the model $M(x)$:

$$\{S_R(v(x)) \cap \{v_1, v_2, ..., v_N\}\} > \#_{\text{min}}$$  \hspace{1cm} (2.23)

where $\#_{\text{min}}$ denotes the cardinality of the set intersection. This involves the computation of $N$ distances between $v(x)$ and model samples, and $N$ comparisons with a thresholded Euclidean distance.

**Update Policy:**
In updating a pixel model, the sample to be replaced is chosen randomly. The new value then replaces the chosen random value as shown in figure 2.2. The expected remaining lifespan of any sample value of the model decays exponentially as:

$$P(t_0, t_1) = e^{-\ln(\frac{N}{\#_{\text{min}}})} t_{1} - t_{0}$$  \hspace{1cm} (2.24)

Where $P(t_0, t_1)$ is the probability of a sample at time $t_0$ to be still present at time $t_1$.

**Model Initialization:**
The first frame is used to initialize the model. Values from spatial neighbourhood of each pixel are used to populate the model. These neighbours are chosen randomly. Segmentation of video sequences then starts from the second frame.

**Advantages:**
*ViBe* shows accurate results in various environments without requiring any fine tuning of parameters[11]. It has three fixed parameter values (matching threshold between a sample and a pixel value, number of samples stored for each pixel model, and the cardinality for the matches).

It is stable for changes in illumination and camera shake.

**Limitations:**
A moving object in the first frame will not be detected and introduces a ghost, that fades over time.

*ViBe* cannot generate a background image for each frame. It is not deterministic, as the results always differ if the algorithm is applied on same video multiple times[10].

### 2.6 Optical Flow

Optical flow methods uses the flow vectors of moving objects over time for detection of moving regions in an image [36][37][38]. Lucas and Kanade [12] used optical flow for motion detection. Even when the camera is moving, optical flow method is able to detect moving objects in video sequences.

It is based on the assumption used by most of the optical flow method that intensity $I$ of moving pixel is constant in subsequent frames. It is computed by taking two images at time $t$ and $t + \delta t$.

$$I(x, y, t) = I(x + \delta x, y + \delta y, t + \delta t)$$

(2.25)

Using Taylor series, above equation is expanded to:

$$I(x + \delta x, y + \delta y, t + \delta t) = I(x, y, t) + \frac{dI}{dx}\delta x + \frac{dI}{dy}\delta y + \frac{dI}{dt}\delta t + H.O.T$$

(2.26)

Avoiding higher order terms(H.O.T.), the equation reduces to:

$$\frac{dI}{dx}\delta x + \frac{dI}{dy}\delta y + \frac{dI}{dt}\delta t = 0$$

(2.27)

$$\frac{dI}{dx}(\delta x/\delta t) + \frac{dI}{dy}(\delta y/\delta t) + \frac{dI}{dt} = 0$$

(2.28)

$$\frac{dI}{dx}V_x + \frac{dI}{dy}V_y + \frac{dI}{dt} = 0$$

(2.29)

$$I_x.V_x + I_y.V_y = -I_t$$

(2.30)

where $V_x, V_y$ represents optical flow vectors and $I_x, I_y$ represent derivatives of the image intensities at coordinate $(x, y, t)$. The values $V_x, V_y$ are used to get the motion vector for the object detection by applying thresholding technique. Magnitude of motion vector is found as:

$$Th = \sqrt{V_x^2 + V_y^2}$$

(2.31)
Table 1: Comparison table for the detection methods [14]-[19]

<table>
<thead>
<tr>
<th>METHOD</th>
<th>ADVANTAGES</th>
<th>DISADVANTAGES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frame Difference</td>
<td>Simple</td>
<td>Suffers from aperture problem</td>
</tr>
<tr>
<td></td>
<td>Low complexity</td>
<td>Uses single global threshold for all pixels</td>
</tr>
<tr>
<td></td>
<td>High computational efficiency</td>
<td></td>
</tr>
<tr>
<td>Approximate Median Filter</td>
<td>Robust to noise</td>
<td>Adapts slowly to a large change in background</td>
</tr>
<tr>
<td></td>
<td>Simple and efficient</td>
<td>Variance of a pixel is not modelled</td>
</tr>
<tr>
<td></td>
<td>Requires less space</td>
<td></td>
</tr>
<tr>
<td>Running Gaussian Average</td>
<td>Good computational speed</td>
<td>Cannot cope with multi-modal backgrounds</td>
</tr>
<tr>
<td></td>
<td>Low memory requirement</td>
<td></td>
</tr>
<tr>
<td>Gaussian Mixture Model</td>
<td>Deals with multi-modal backgrounds</td>
<td>Requires careful tuning of parameters</td>
</tr>
<tr>
<td></td>
<td>Uses different threshold for each pixel</td>
<td>Cannot deal with sudden drastic illumination changes</td>
</tr>
<tr>
<td></td>
<td>High accuracy</td>
<td>Computationally intensive</td>
</tr>
<tr>
<td>ViBe</td>
<td>Accurate in various environments</td>
<td>Introduces ghost in initial frames</td>
</tr>
<tr>
<td></td>
<td>Uses fixed parameters value(no tuning)</td>
<td>Not-deterministic( different result on each run of same output)</td>
</tr>
<tr>
<td></td>
<td>Stable for illumination changes &amp; camera shake</td>
<td></td>
</tr>
<tr>
<td>Optical Flow</td>
<td>Can cope even when camera is shaking</td>
<td>Computationally complex</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Need specialized hardware for real-time use</td>
</tr>
</tbody>
</table>

Thresholding is applied on this $T_h$ value. Finally, the moving object is detected with the help of morphological operations.

**Limitations:**
Optical flow methods are computationally complex and need specialized hardware for use in real-time [13].

3 RESULTS

We have evaluated the results of the detection methods on a video sequence from i-Lids dataset for (AVSS 2007) and KTH. Figure 2.3 shows the original image frame with their respective masked images and the resultant foreground objects detected after applying the detection methods.

4 CONCLUSIONS

In this paper, we have surveyed six most widely used moving object detection methods in detail. Each of the methods have their own advantages and limitations. Comparisons of the techniques are also done as shown in Table 1 which states merits and demerits of each method briefly. This would be helpful to the experts for which method to choose for their purpose.

References


Vishakha Sharma, Neeta Nain, Tapas Badal


Vishakha Sharma, is currently pursuing M.tech. at the department of Computer Science & Engineering, Malaviya National Institute of Technology, Jaipur. She has received her B.Tech. degree in Computer Science & Engineering from Rajasthan Technical University, Kota in 2014. Her research area is Object Detection, Object Tracking, and Video Surveillance.

Dr. Neeta Nain, is an assistant professor at the department of Computer Science & Engineering, Malaviya National Institute of Technology, Jaipur. She was awarded her Ph.D. in Computer Science & Engineering from the Malaviya National Institute of Technology, Jaipur. She also received her MCA from Banasthali University. Her research area is Image processing, Pattern Recognition, Multimedia Techniques and Computer Graphics. Presently she is guiding research in Handwritten Text Recognition and Image vectorization.

Tapas Badal, is currently pursuing Ph.D. at the department of Computer Science & Engineering, Malaviya National Institute of Technology, Jaipur. He received the B.Tech. degree in Computer Engineering from IES-IPS Academy Indore in 2006. He was awarded his M.Tech. in Computer Engineering from ABV - Indian Institute of Information Technology & Management, Gwalior in 2009, which involved research into Advance Networks. His research area is Pattern Recognition, Activity Analysis, and Content Based Video Processing.