International Journal of Multidisciplinary and Current Research ISSN: 2321-3124
Research Article Available at: http://ijmcr.com

Moving Object Detection by Contrast Independent Threshold Background Subtraction

Neeru[#] and Davinder Parkash[#]

[#]Department of Electronics & Communication Engineering, HCTM Kaithal, Haryana, India

Accepted 07 March 2016, Available online 12 March 2016, **Vol.4 (March/April 2016 issue)**

Abstract

Moving object detection is the basic step for the analysis of video. It is the monitoring of the behavior, activities or other changing information and aims at background subtraction to locate the object position in video frame. A fast and accurate moving object detection technique is important to detect, recognize and track objects over a sequence of images. This research area has been studied for decades; many techniques have been reported and applied on different video surveillance applications. However, there are still some unsolved problems need to be addressed due to multiple objects present in the scene, whereby we wish to determine the position of the same object across time. These types of tasks require not only good initial object detection but reliable body part segmentation as well. Thresholding has found to be a well-known technique for background subtraction such that pixels labelled corresponds to object and 0 to background. Thresholding is further divide into the global and local thresholding techniques.

Keywords: *Video Frame, Local thresholding, Global thresholding, Otsu's thresholding, Texton Co-occurrence Matrix, Kalman Filter*

Nomenclature

I : Frame Data
FM : Frame Difference
FDM : Frame Difference Mask
BD : Background Difference
TCM : Texton Co-occurrence Matrix
BDM : Background Difference Mask
GLCM : Gray Level Co-occurrence Matrix
IOM : Initial Object Mask
BI : Background Information
SI : Stationary Index
BG : Background Indicator
IBG : Image Background Subtraction

1. Introduction

In video surveillance we try to detect, recognize and track objects over a sequence of images and it also makes an attempt to understand and describe object behavior by replacing the aging old traditional method of monitoring cameras by human operators. Object detection and tracking are important and challenging tasks in many computer vision applications such as surveillance, vehicle navigation and autonomous robot navigation. Object detection involves locating objects in the frame of a video sequence.

It handles segmentation of moving objects from stationary background objects. Commonly used techniques for object detection are background subtraction, statistical models, temporal differencing and optical flow. Every tracking method requires an object detection mechanism either in every frame or when the object first appears in the video.The final step of the smart video surveillance systems is to recognize the behaviors of objects and create high-level semantic descriptions of their actions.

1.1 Detection

The main aim of object detection is to distinguishing foreground objects from the stationary background. Almost the entire visual surveillance systems the first step is detecting foreground objects [12]. Short and long term dynamic scene changes such as repetitive motions (e.g. waiving tree leaves, light reflectance, shadows, camera noise and sudden illumination variations) make reliable and fast object detection difficult. It is a general idea that if an object is changing its position with respect to a point in the space, then it is considered to be moving. Rest scene is said to be the background. The movements of the object can be properly analyzed if object is detected accurately. Background subtraction is a general term for a process which aims to segment moving foreground objects from a relatively stationary background. As

illustrated in fig 1.1 there is an important distinction between the background modeling and background detection stages, which comprise the whole subtraction process. These two stages are often interrelated and sometimes overlapping. The modeling stage creates and maintains a model of the background scene. The detection process is responsible for segmenting the current image into moving (foreground) and stationary (background) regions based on the current background model. The resulting detection masks may then be fed back into the modeling process in order to avoid corruption of the background model by foreground object.

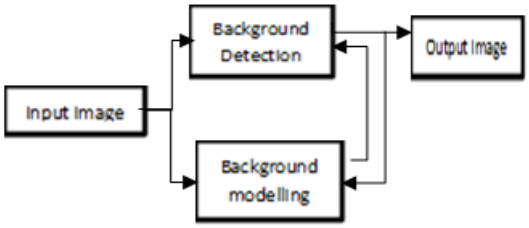


Figure 1.1: A video source produces images

The modeling stage uses previous video frames and detection results to maintain background model B. The detection stage compares the current frame to the current model to produce a detection mask E.

In order for the description to be characteristic of the true object, a reliable segmentation must be provided otherwise, errors in the detection stage will give rise to misrepresentation, which may result in misclassification.

1.2 Global Thresholding Method

Thresholding is one of the most powerful and important tools for image segmentation. The segmented image obtained from thresholding has the advantages of smaller storage space, fast processing speed and ease in manipulation compared with gray level image which usually contains 256 levels. The thresholding techniques, which can be divided into bi-level and multilevel category. In bi-level thresholding [Fig.1.2(a)], a threshold is determined to segment the image into two brightness regions which correspond to background and object. Several methods have been proposed to automatically select the threshold. Otsu et.al formulates the threshold selection problem as a discriminant analysis where the gray level histogram of image is divided into two groups and the threshold is determined when the variance between the two groups is the maximum. Even in the case of unimodal histogram images, that is, the histogram of a gray level image does not have two obvious peaks, Otsu's method can still provide satisfactory result. Therefore, it is referred to as one of the most powerful methods for bi-level thresholding. In multilevel thresholding [Fig.1.2(b)], more than one threshold will be determined to segment the image into certain brightness regions which correspond to one background and several objects.

The selection of a threshold will affect both the accuracy and the efficiency of the subsequent analysis of the segmented image. The principal assumption behind the approach is that the object and the background can be distinguished by comparing their gray level values with a suitably selected threshold value. If background lighting is arranged so as to be fairly uniform, and the object is rather flat that can be silhouetted against a contrasting background, segmentation can be achieved simply by thresholding the image at a particular intensity level.

Suppose that the gray-level histogram shown in Fig 1.2(a) corresponds to an image, f(x; y), composed of light objects on a dark background, in such a way that object and background pixels have gray levels grouped into two dominant modes. One obvious way to extract the objects from the background is to select a threshold T that separates these modes. Then any (x; y) for which f(x; y) > T is called an object point; otherwise, the point is called a background point. Fig. 1.5(b) shows a slightly more general case of this approach, where three dominant modes characterize the image histogram (for example, two types of light objects on a dark background).

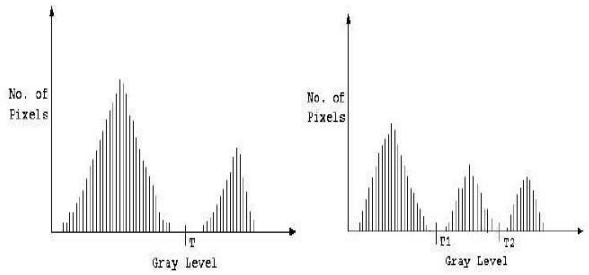


Fig.1.2 Gray-level histograms that can be partitioned by (a) a single threshold, and (b) multiple thresholds

Here, multilevel thresholding classifies a point (x; y) as belonging to one object class if T1 < f(x; y) >T2, to the other object class if f(x; y) > T2, and to the background if f(x; y)< T1. Based on the preceding discussion, thresholding may be viewed as an operation that involves tests against a function T of the form *T = T[x; y; p(x; y); f(x; y)]* where f(x; y) is the gray level of point (x; y) and p(x; y) denotes some local property of this point-for example, the average gray level of a neighborhood centered on (x; y). A threshold image g(x; y) is defined as

$$G(x,y) = \begin{cases} 0 - if - f(x,y), \leq T \\ 1 - if - f(x,y), > T \end{cases}$$

Thus, pixels labeled 1 (or any other convenient gray level) correspond to objects, whereas pixels labeled 0 (or any other gray level not assigned to objects) correspond to the background. The technique that is most frequently employed for determining thresholds involves analyzing the histogram of intensity levels in the gray image. This method is subject to some major difficulties like determining relevant thresholding level in multimodal.

Otsu's global thresholding method

This method is a nonparametric and unsupervised method of automatic threshold selection for image segmentation. An optimal threshold is calculated by the discriminate criterion, namely, so as to maximize the between-class variance or to minimize the within-class variance. The method is very simple, utilizing only the zeroth and first order cumulative moments of the gray level histogram.

Let the pixels of a given image represented in *L* gray levels [1; 2; :::; L]. The number of pixels at level *i* is denoted by *ni* and the total number of pixels by *N = n1+n2+:::+nL.* For simplification, the gray-level histogram is normalized and regarded as a probability distribution.

$$P_i = \frac{ni}{N}, \quad P_i \geq 0, \qquad \sum_{i=1}^{L} Pi = 1 \tag{1}$$

To emphasize the partitioned windows technique, only Otsu's thresholding method is considered among many other techniques. This method can be stated as follows: For a given image *f(x, y)* with m gray levels *0, 1, . . . ,m-1*, let the threshold be j, where *0< j< m-1*. Then, all pixels in image *f(x, y)* can be divided into two groups: group A with gray level values of pixels less than or equal to *j*; and group B with values greater than *j*. Also, let *(w1(j), M1(j)) (Eqn.2,3) (w2(j), M2(j)) (Eqn.4,5)* be the number of pixels and the average gray level value in group A and group B, respectively. Then

$$w_1(j) = \sum_{i=0}^{j} ni, 0 \leq j \leq m-1 \tag{2}$$

$$M_1(j) = \frac{\sum_{i=0}^{j}(i,ni)}{w1(j)}, 0 \leq j \leq m\text{-}1 \tag{3}$$

$$W_2(j) = \sum_{t=j+1}^{m-1} ni, 0 \leq j \leq m-1 \tag{4}$$

$$M_2(j) = \frac{\sum_{t=j+1}^{m-1}(i,ni)}{w2(j)}, 0 \leq j \leq m\text{-}1 \tag{5}$$

where n_i is the number of pixels with gray level value i. Expressing the average gray level value M_t (Eqn.6) of all the pixels in image f(x, y) as

$$M_{t=} \frac{w1(j)M1(j)+w2(j)\,M2(j)}{w1(j)+w2(j)} \qquad 0 \leq j \leq m-1 \tag{6}$$

the variance between the two groups, denoted as $\sigma_B^2(j)$, is

$$\sigma^2_{B(j)} = w1(j)(M1(j) - MT)2 + w2(j)(M2(j) - MT)2 \tag{7}$$

$$= \frac{w1(j)w2(j)(M1(j) - M2(j))2}{w1(j)+w2(j)} \tag{8}$$

For j ranging from 0 to m-1, calculate each $\sigma^2_{B(j)}$ Using above Eqn(7,8), and the value j corresponding to the greatest $\sigma^2_{B(j)}$ is the resulting threshold T.

1.3 Kalman Filter

The Kalman filter, known as linear quadratic estimation (LQE), is an algorithm that uses a series of measurements observed over time, containing noise (random variations) and other inaccuracies, and produces estimates of unknown variables that tend to be more correct than those based on a single measurement alone.

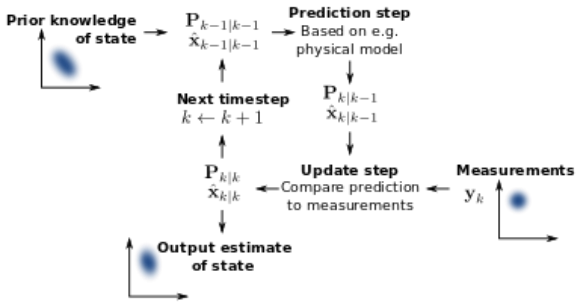


Fig.1.3: Basic Kalman filtering

The Kalman filter keeps track of the estimated state of the system and the variance or uncertainty of the estimate. The estimate is updated using a state transition model and measurements. $\hat{X}_{(k|k-1)}$ denotes the estimate of the system's state at time step *k* before the *k*-th measurement y_k has been taken into account; $\widehat{P_{k|k-1}}$ is the corresponding uncertainty. © Wikipedia

The algorithm works in a two-step process. In the prediction step, the Kalman filter produces estimates of the current state variables along with their uncertainties. Once the outcome of the next measurement (necessarily corrupted with some amount of error including random noise) is observed, these estimates are updated using a weighted average with more weight being given to estimates with higher certainty. Because of the algorithm's recursive nature, it can run in real time using only the present input measurements and the previously calculated state and its uncertainty matrix; no additional past information is required.

1.3.1 Underlying dynamic system model

The Kalman filters are based on linear dynamic systems discretized in the time domain. They are modeled on a Markov chain built on linear operators perturbed by errors that may include Gaussian noise. The state of the system is represented as a vector of real numbers. At each discrete time increment, a linear operator is applied to the state to generate the new state, with some noise mixed in and optionally some information from the controls on the system if they are known. Then another linear operator mixed with more noise generates the observed outputs from the true ("hidden") state. The Kalman filter may be regarded as analogous to the hidden Markov model, with the key difference that the hidden state variables take values in a continuous space (as opposed to a discrete

state space as in the hidden Markov model). There is a strong duality between the equations of the Kalman Filter and those of the hidden Markov model.

In order to use the Kalman filter to estimate the internal state of a process given only a sequence of noisy observations, one must model the process in accordance with the framework of the Kalman filter. This means specifying the following matrices: F_k, the state-transition model; H_k, the observation model; Q_k, the covariance of the process noise; R_k, the covariance of the observation noise; and sometimes B_k, the control-input model, for each time-step, *k*, as described below.

The Kalman filter model assumes the true state at time *k* is evolved from the state at (*k*−1) according to

$$x_k = F_k x_{k-1} + B_k u_k + w_k \quad (1)$$

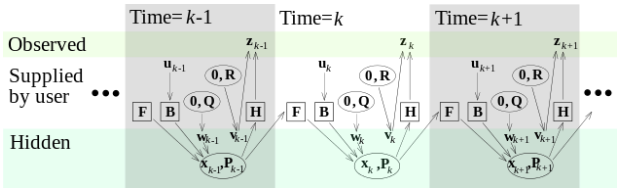


Fig.1.4: Kalman filter model

In model underlying the Kalman filter. Squares represent matrices. Ellipses represent multivariate normal distributions (with the mean and covariance matrix enclosed). Unenclosed values are vectors. In the simple case, the various matrices are constant with time, and thus the subscripts are dropped, but the Kalman filter allows any of them to change each time step.

Where

- F_k is the state transition model which is applied to the previous state x_{k-1};
- B_k is the control-input model which is applied to the control vector u_k;
- w_k is the process noise which is assumed to be drawn from a zero mean multivariate normal distribution with covariance Q_k.

$w_k \sim N(0, Q_k)$

At time *k* an observation (or measurement) z_k of the true state x_k is made according to

$$z_k = H_k x_k + v_k \quad (2)$$

where

- H_k is the observation model which maps the true state space into the observed space
- v_k is the observation noise which is assumed to be zero mean Gaussian white noise with covariance R_k.

$\boldsymbol{v_k} \sim N(0, R_k)$

The initial state, and the noise vectors at each step $\{x_0, w_1, \ldots, w_k, v_1 \ldots v_k\}$ are all assumed to be mutually independent.

1.3.2 Details

The Kalman filter is a recursive estimator. This means that only the estimated state from the previous time step and the current measurement are needed to compute the estimate for the current state. In contrast to batch estimation techniques, no history of observations and/or estimates is required. In what follows, the notation $\hat{X}\, n/m$ represents the estimate of x at time *n* given observations up to and including at time $m \leq n$.

The state of the filter is represented by two variables:

- $\widehat{x_{k|k-1}}$, the a posteriori state estimate at time *k* given observations up to and including at time *k*;
- $\widehat{P_{k|k-1}}$, the a posteriori error covariance matrix (a measure of the estimated accuracy of the state estimate).

The Kalman filter can be written as a single equation; however, it is most often conceptualized as two distinct phases: "Predict" and "Update". The predict phase uses the state estimate from the previous time step to produce an estimate of the state at the current time step. In the update phase, the current a priori prediction is combined with current observation information to refine the state estimate. This improved estimate is termed the a posteriori state estimate.

1.3.3 Predict

Predicted (*a priori*) state estimate

$$\widehat{x_{k|k-1}} = F_k \widehat{x_{k-1|k-1}} + B_k u_k \quad (3)$$

Predicted (*a priori*) estimate covariance

$$\widehat{P_{k|k-1}} = F_k \widehat{P_{k-1|k-1}} F_k^T + Q_k \quad (4)$$

1.3.4 Update

Innovation or measurement residual

$$\widetilde{y_k} = z_k - \widehat{x_{k|k-1}} H_k \quad (5)$$

Innovation (or residual) covariance

$$S_k = H_k P_{k|k-1} H_k^T + R_k \quad (6)$$

Optimal Kalman gain

$$K_k = P_{k|k-1} H_k^T S_k^{-1} \quad (7)$$

Updated (*a posteriori*) state estimate

$$\widehat{x_{k|k}} = \widetilde{y_k} K_k + \widehat{x_{k|k-1}} \quad (8)$$

Updated (*a posteriori*) estimate covariance

$$P_{k|k} = (1 - K_k H_k) P_{k|k-1} \quad (9)$$

The formula for the updated estimate and covariance above is only valid for the optimal Kalman gain.

1.3.5 Invariants

If the model is accurate, and the values for $\hat{X}(0|O)$ and P$(0|0)$ accurately reflect the distribution of the initial state values, then the following invariants are preserved: (all estimates have a mean error of zero)

$$E\left[X_k - \widehat{x_{k|k}}\right] = E\left[X_k - \widehat{x_{k|k-1}}\right] = 0 \quad (10)$$

$$E[Y_k] = 0 \quad (11)$$

where $E[\xi]$ is the expected value of ξ, and covariance matrices accurately reflect the covariance of estimates

$$P_{k|k} = cov\left(X_k - \widehat{x_{k|k}}\right) \quad (12)$$

$$P_{k|k-1} = cov\left(X_k - \widehat{x_{k|k-1}}\right) \quad (13)$$

$$S_k = cov(\widetilde{y_k}) \quad (14)$$

1.3.6 Optimization and Performance

It is known from the theory that the Kalman filter is optimal in case that a) the model perfectly matches the real system, b) the entering noise is white and c) the covariance of the noise are exactly known. Several methods for the noise covariance estimation have been proposed during past decades. After the covariance's are identified, it is useful to evaluate the performance of the filter, i.e. whether it is possible to improve the state estimation quality. It is well known that, if the Kalman filter works optimally, the innovation sequence (the output prediction error) is a white noise. The whiteness property reflects the state estimation quality. For evaluation of the filter performance it is necessary to inspect the whiteness property of the innovations. Several different methods can be used for this purpose.
The Kalman filter is a recursive two-stage filter. At each iteration, it performs a predict step and an update step. The predict step predicts the current location of the moving object based on previous observations. For instance, if an object is moving with constant acceleration, we can predict its current location,$\hat{x}_t$based on its previous location, $\hat{x}_{t-1}$, using the equations of motion. The update step takes the measurement of the object's current location (if available), z_t and combines this with the predicted current location, $\hat{x}_t$, to obtain an *a* posteriori estimated current location of the object, x_t. Here after initializing the state equation values, background subtraction is done. For background subtraction adaptive background subtraction is used. In this current frame is stored and subtracted from next frame. Assumption behind this is background is constant and object is moving. When object moves its pixels change and detecting that change in pixels object can be identified [17]. But it is rare in chance that background remains stationary. So a threshold value is decided so that pixel having value greater than that can be considered foreground. For threshold value, mean value of pixels is considered. Pixel having value greater than mean will be considered object in foreground. After detecting the object bounding box coordinates are found out. The value of these coordinates will be outcome prediction step of Kalman filter.

The minimum or smallest bounding or enclosing box is a term used in geometry. For a point set (*S*) in *N* dimensions, it refers to the box with the smallest measure (area, volume, or hyper volume in higher dimensions) within which all the points lie. When other kinds of measure are used, the minimum box is usually called accordingly e.g. "minimum-perimeter bounding box".

1.4 Proposed Work

In this section proposed thresholding technique for image background subtraction is explained. For a degraded image adaptive contrast map is constructed and then threshold is calculated from that which converts the image into binary image. Here is the explanation.

The human visual system is more sensitive to contrast than absolute luminance, we can perceive the world similarly regardless of the huge changes in illumination over the day or from place to place. Contrast is the difference in luminance or color that makes an object (or its representation in an image or display) distinguishable. In visual perception of the real world, contrast is determined by the difference in the color and brightness of the object and other objects within the same field of view. The contrast of image can be categorized as global contrast and local contrast. Global contrast measures the brightness difference between the darkest and brightest element in the entire image. Tools like Curves and Levels only change global contrast as they treat all pixels with the same brightness levels identical.

The global contrast has three main regions

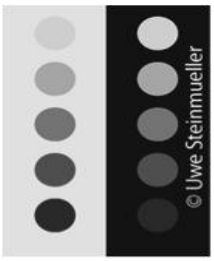
- Mid-tones
- Highlights
- Shadows

The sum of the contrast amounts of these three regions defines the global contrast. This means if you spend more global contrast on the mid-tones (very commonly needed) you can spend less global contrast on highlights and shadows at any given global contrast level.

The mid-tones normally show the main subject. If the mid-tones show low contrast the image lacks "snap".

Adding more global contrast to the mid-tones ("snap") often results in compressed shadows and highlights. Adding some local contrast can help to improve the overall image presentation.

The local contrast is based on the retinex theory according to which our eyes sees the difference in respect to surroundings, a color map below can prove this point.



The circles in each row have exactly the identical brightness levels. Yet the top right circle looks a lot brighter than the one on the left. This is because our eyes see the difference to the local surrounding. The right circle looks much brighter with the dark gray background compared to a brighter background on the left. Just the opposite is true for the two circles on the bottom. For our eyes the absolute brightness is of less importance than the relative relation to other close areas. So, local contrast is very important for processing or enhancement of any image.

In our work because of this human visual system local contrast map is extracted from an image and then on the basis of that a local thresholding approach will be used to convert the image onto binary format. Previously image gradient and normalize image gradient were used to extract local contrast of image, these methods are quite good, although the variation of bright to weak contrast can be compensated by these methods yet these don't perform well in case of document which have bright text. This is because a weak contrast will be calculated for stroke edges of the bright text. Calculation of local contrast and then global thresholding algorithm like otsu is used and then local image edge detection is used in paper published by Bolan Su (2013). We have followed the same line of action but rather than using global thresholding, we use local thresholding, it removes the need of using again local edge detection algorithm like canny edge detection. Gray level co –occurrence matrix (GLCM) also called texton co- occurrence matrix (TCM) fulfills our purpose. It is a local contrast mapping method. Here basically TCM serves two purposes: make image's local contrast map, unaffected by the illumination variation of image and local edge detection. Further, the GLCM functions characterize the texture of an image by calculating how often pairs of pixel with specific values and in a specified spatial relationship occur in an image, creating a GLCM, and then extracting statistical measures from this matrix. A gray-level co-occurrence matrix (GLCM) is generated by calculating how often a pixel with the intensity (gray-level) value *i* occurs in a specific spatial relationship to a pixel with the value *j*. By default, the spatial relationship is defined as the pixel of interest and the pixel to its immediate right (horizontally adjacent), but you can specify other spatial relationships between the two pixels. Each element (*i, j*) in the resultant glcm is simply the sum of the number of times that the pixel with value *i* occurred in the specified spatial relationship to a pixel with value *j* in the input image. The number of gray levels in the image determines the size of the GLCM. GLCM of an image is computed using a displacement vector d, defined by its radius δ and orientation θ. To illustrate, the following figure shows how gray comatrix calculates the first three values in a GLCM. In the output GLCM, element (1,1) contains the value 1 because there is only one instance in the input image where two horizontally adjacent pixels have the values 1 and 1, respectively. glcm(1,2) contains the value 2 because there are two instances where two horizontally adjacent pixels have the values 1 and 2. Element (1,3) in the GLCM has the value 0 because there are no instances of two horizontally adjacent pixels with the values 1 and 3. Gray comatrix continues processing the input image, scanning the image for other pixel pairs (*i, j*) and recording the sums in the corresponding elements of the GLCM. Figure 1.5 shows this concept.

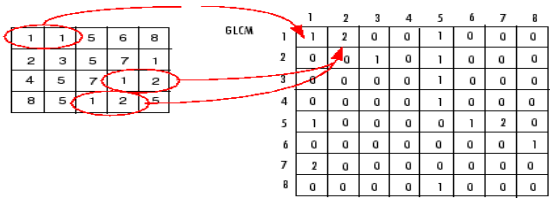


Fig. 1.5: GLCM output of a test matrix

A single GLCM matrix might not able to define all texture features of image, so multiple GLCM at different orientations are calculated. Above given example was with 0^{o} orientation i.e. horizontally matching pairs are checked. Further it can be done at angle 45^{o}, 90^{o}, 135^{o} as shown in figure 1.6.

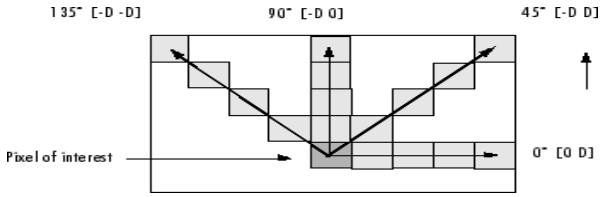


Fig. 1.6: Multiple orientations to calculate GLCM

In actual every pixel has eight neighboring pixels allowing eight choices for θ, which are 0°, 45°, 90°, 135°, 180°, 225°, 270° or 315°. However, taking into consideration the definition of GLCM, the co-occurring pairs obtained by choosing θ equal to 0° would be similar to those obtained by choosing θ equal to 180°. This concept extends to 45°, 90° and 135° as well. Hence, one has four choices to select the value of θ.

Above we have mentioned a term radius about GLCM. In the last example matching pairs have been taken up to one distance, this constitutes the radius of GLCM. Various research studies show δ values ranging from 1, 2 to 10. Applying large displacement value to a fine texture would yield a GLCM that does not capture detailed textural information. From the previous studies, it has been concluded that overall classification accuracies with δ = 1, 2, 4, 8 are acceptable with the best results for δ = 1 and 2. This conclusion is justified, as a pixel is more likely to be correlated to other closely located pixel than the one located far away. Also, displacement value equal to the size of the texture element improves classification.

The dimension of a GLCM is determined by the maximum gray value of the pixel. Number of gray levels is an important factor in GLCM computation. More levels would mean more accurate extracted textural information, with increased computational costs. The computational complexity of GLCM method is highly sensitive to the number of gray levels. As in above example in figure 1.5, the size of GLCM is 8 by 8 matrix as 8 gray levels have been considered. Thus for a predetermined value of G, a GLCM is required for each unique pair of δ and θ. GLCM is a second-order texture measure. The GLCM's lower left triangular matrix is always a reflection of the upper right triangular matrix and the diagonal always contains even numbers.

0	0	1	1
0	0	1	1
0	2	2	2
2	2	3	3

4	2	1	0
2	4	0	0
1	0	6	1
0	0	1	2

Fig.1.7 Test matrix **Fig.1.8** test matrix with θ 0

Various GLCM parameters are related to specific first-order statistical concepts. For instance, contrast would mean pixel pair repetition rate, variance would mean spatial frequency detection etc. Association of a textural meaning to each of these parameters is very critical. Traditionally, GLCM is dimensioned to the number of gray levels G and stores the co-occurrence probabilities g_{ij}. To determine the texture features, selected statistics are applied to each GLCM by iterating through the entire matrix. The textural features are based on statistics which summarize the relative frequency distribution which describes how often one gray tone will appear in a specified spatial relationship to another gray tone on the image.

Following notations are used to explain the various textural features:

g_{ij} = $(i, j)^{th}$ entry in GLCM
$g_x(i)$ = i^{th} entry in marginal probability matrix obtained by summing rows of gij
N_g = Number of distinct gray levels in the image

$g_y(i) = \sum_{i=1}^{N_g} g(i,j)$
Contrast (con) = $\sum_i \sum_j (i-j)^2 g_{i,j}$

This statistic measures the spatial frequency of an image and is difference moment of GLCM. It is the difference between the highest and the lowest values of a contiguous set of pixels. It measures the amount of local variations present in the image. A low contrast image presents GLCM concentration term around the principal diagonal and features low spatial frequencies.

From this GLCM process local contrast of image is obtained. From this we have developed the equation to calculate the threshold value. Since it will be a local threshold value so, the size of threshold matrix will be same as test image. For this formula we were inspired by Abdenour Sehad's work in 2013. We have done changes in that and final thresholding formula includes local mean of image and a gain factor which will act as a bias factor. This factor has the range (0-1) always. Its value will be determined experimentally. This formula is used in a window size of image and later on combined. Mathematical expression is shown below.

$$Threshold(i,j) = k(I_{mean}(i,j) + \sqrt{contrast})$$

Since the above method discussed calculates the local threshold, the operations are done in image blocks like if outcome of the algorithm is block size =55, then image matrix will be segmented into 55 by 55 blocks and thresholded.

1.4.1 Background subtraction

Section 1.2 discussed the threshold method used for the background subtraction using segmentation. Video can be considered as multiple frames. Each frame is different from other in pixel values. But those which are equal are treated as background pixels as background don't moves but there is problem when background consists of slow moving objects like swaying of trees which must be considered as background, but due to difference of each frame to next frame these also appear into foreground as pixel's value changes for these too. To avoid this problem, we have used multi background registration concept. In this frame difference mask along with background difference mask is generated and both are used to decide which is pixel constitutes the foreground. Here is the work given in detail with the help of flow chart.

1.4.1.1 Flow Chart of IBG

A flow chart for image background subtraction by proposed work is shown below, whole algorithm is divided into four steps:

Step1: Frame Difference

In Frame Difference, the frame difference between current frame and previous frame, which is stored in

Frame Buffer, is calculated and thresholded. It can be presented as

$$FD(x,y,t) = I(x,y,t) - I(x,y,t-1)$$
$$FDM(x,y,t) = \begin{cases} 1 \; if FD \geq Th \\ 0 \; if \; FD < Th \end{cases}$$

where I is frame data, FD is frame difference, and is FDM Frame Difference Mask, 't' represents the time of coming frame, 't-1' is for previous frame. Note that there is a parameter Th needed to be set in advance. The method to decide the optimal is discussed in section 3.1. Pixels belonging to FDM are viewed as "moving pixels."

Step 2: Background Registration

Background Registration can extract background information from video sequences. According to FDM, pixels not moving for a long time are considered as reliable background pixels. The procedure of Background Registration can be shown as

$$SI(x,y,t) = \begin{cases} SI(x,y,t-1)+1 & if FDM = 0 \\ 0 & if \; FDM = 1 \end{cases}$$
$$BG(x,y,t) = \begin{cases} I(x,y,t) & if FDM = 0 \\ BG(x,y,t-1) & else \end{cases}$$
$$BI(x,y,t) = \begin{cases} 1 & if SI(x,y,t) = Fth \\ BI(x,y,t-1) & else \end{cases}$$

Where SI is Stationary Index, BG is Background Indicator, and BI is the background information. The initial values all are set to "0." Stationary Index records the possibility if a pixel is in background region. If SI is high, the possibility is high; otherwise, it is low. If a pixel is "not moving" for many consecutive frames, the possibility should be high, which is the main concept of SI equation. When the possibility is high enough, the current pixel information of the position is registered into the background buffer, which is shown as BG. Besides, Background Indicator is used to indicate whether the background information of current position exists or not, which is shown as BI.

Step 3: Background Difference

The procedure of Background Difference is similar to that of Frame difference. What is different is that the previous frame is substituted by background frame. After Background Difference, another change detection mask named Background Difference Mask is generated. The operations of Background Difference can be shown by

$$BD(x,y,t) = |I(x,y,t) - BG(x,y,t-1)|$$
$$BDM(x,y,t) = \begin{cases} 1 \; if BD \geq Th \\ 0 \; if \; BD < Th \end{cases}$$

where BD is background difference, is background frame, and is BDM Background Difference Mask, respectively.

Step 4: Object Detection

Both of FDM and BDM are input into Object Detection to produce Initial Object Mask (IOM). The procedure of Object Detection can be presented as the following equation.

$$IOM(x,y,t) = \begin{cases} BDM(x,y,t) & if \; BI(x,y,t) = 1 \\ FDM(x,y,t) & else \end{cases}$$

In IOM every frame is passed through morphological imclose operation which will fill the pixels in 3*3 neighborhood.

In post processing work done till now is used conditionally to extract background and foreground separately. These conditions are shown in table 1.1.

Table 1.1: Conditions to separate background and foreground

	IOM	FDM	BDM	BI
Foreground Object	1	-	1	-
Background	0	0	0	1

Result & Discussion

The tracking algorithm has been successfully applied to video surveillance. The result shows that the algorithm has been able to track any single moving object. The algorithm has been implemented and tested on MATLAB with operating system window 10. Comparison of results is shown with otsu's thresholding algorithm which is a global processing thresholding method and our proposed thresholding scheme is based on local processing. Time consumed in both processes is compared. In Kalman filter it is required to segment the moving object from background and background may consists small moving objects like tree leaves or moving fans in buildings behind etc. or lamination variation also. These changes pose threats to the background segmentation task. Global processing thresholding doesn't deal with or other method in collaboration has to use to compensate the variations or small motions. But in our work our thresholding algorithm itself is capable to remove these issues.

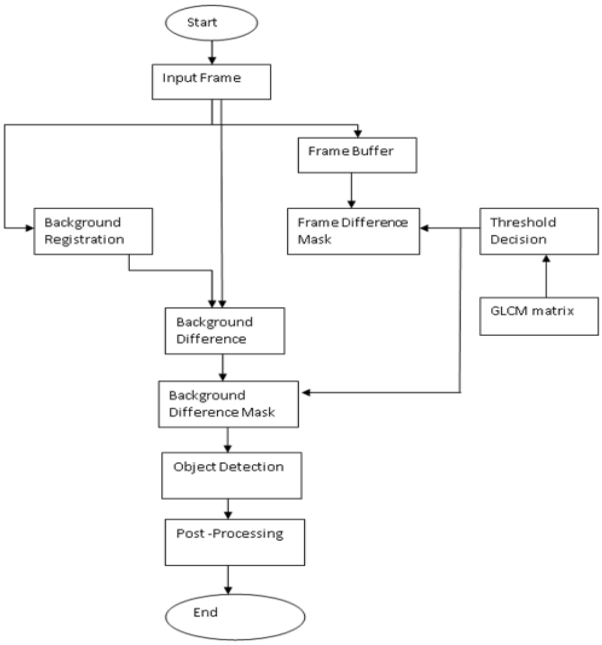
Fig.1.9: Gray scale frames

Initially we will discuss about the results in which small leaves movement is in the background. These frames are further converted to gray scale frames to save memory requirements for the processing of three color bands of colored image. Gray scale conversion reduces the memory requirements keeping all information in the image, although very small losses are still but negligible. The gray scale frames are shown in figure 1.9.

In our work, initially background variable is initialized by making every pixel element value zero. Then all coming frames are subtracted from the previous frame based on the fact that the pixel's stationary in both frames are considered as background and will become zero after subtracting next frame from previous frame. This will constitute a frame difference matrix. A threshold value using proposed work is generated corresponding to every pixel of the frame difference matrix and compared with each pixel to get a binary frame difference mask (FDM) which will have only moving object in each frame. Background pixel considered in that particular frame will be black as FDM is binary image. But a lot of noise in the background is found by Otsu's algorithm.

It is the strong point of kalman filter that even with lot of noise in the background it is able to locate the object in the image. In our case no morphological opening and closing and nor pre and post processing operations of image processing are done. All results are true outcome of algorithm processing.

Further the tracking of object is done by placing a rectangle over moving part in the frame. Figures 1.10 and 1.11 show the moving object tracking by proposed and Otsu's method. It is noticed that in the video frame at a single frame the single body part is in motion so our method tracks only that part. This is a very precise tracking of motion. Leaves waving in background bushes are ignored by the algorithm. While in figure 1.11 in some frames it includes the bushes along with full body movement. Otsu's thresholding gave many false tracking results. It also included stationary pixels in the bounding box used for tracking movement. Even time consumed by Otsu's algorithm along with kalman filter is more than our proposed. Otsu's algorithm took 48.450924 seconds in full process while ours did this in 14.075072 seconds. Clearly it is visible that proposed threshold algorithm which uses gray level co-occurrence matrix and local mean is giving very accurate results than Otsu's thresholding algorithm. Results are checked on another input frames in which object is moving in front of a factory and fan's motion in the background even though very less visible, is present, shown in figure 1.12-1.13. This video also suffers from the luminance variation problem as it is captured during evening time, so a frame is at some places is suffering from brightness variation. Proposed work also compensates that results are not affected by this luminance variation and fan's movement in the background. It is noticed that in some frames fans motion is also tracked which shouldn't be.



Flow chart of proposed background subtraction method

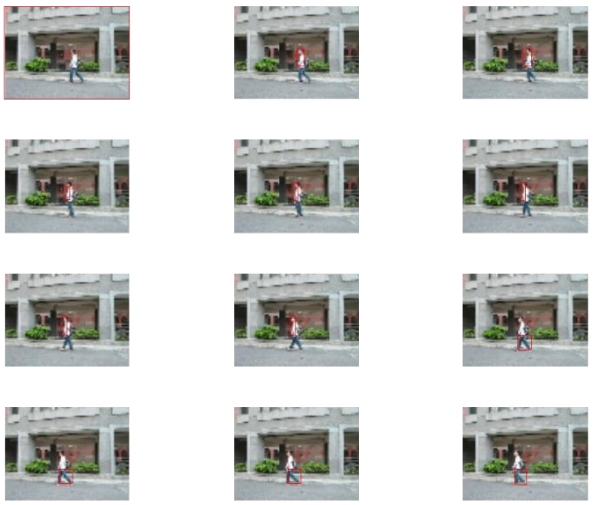


Fig.1.10: Motion detection by Proposed Method

Fig.1.11: Motion detection by Otsu' Method

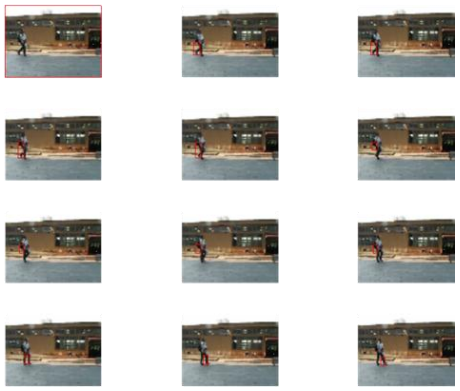


Fig.1.12: Another input with fan's revolving in the background by proposed scheme

Fig.1.13: Input with fan's revolving in the background by Otsu's scheme

Table 1.2: Time elapsed in each input video frames

	Proposed Scheme (in sec)	Otsu's scheme (in sec)
Input video 1 (16 frames)	14.075072	48.450924
Input video 2 (16 frames)	18.031187	57.678786

Table 1.3: Average Threshold value for background subtraction for input video frames

	Proposed Scheme	Otsu's scheme
Input video 1	5.3236	0.4980
Input video 2	6.5004	0.4980

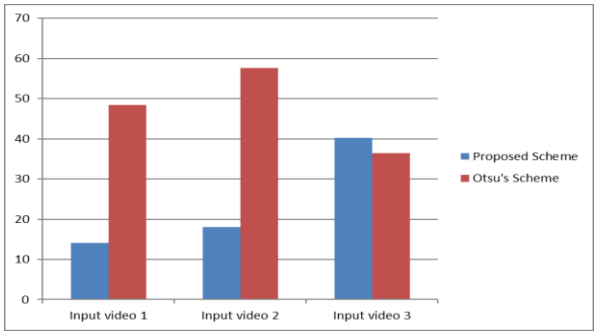


Fig.1.14: Time elapsed Comparison in both cases

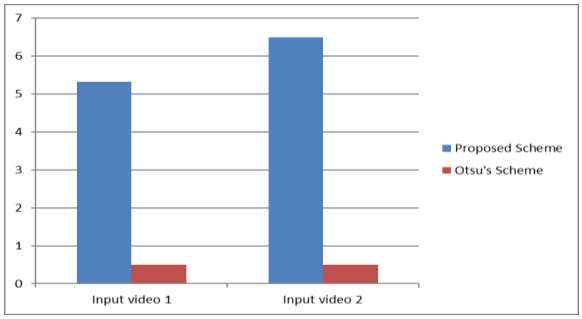


Fig.1.15: Threshold value Comparison in both cases

Conclusion

For surveillance purpose the tacking of human or any object form the camera at road side or in any building, increases the efficiency. If tracking is implemented successfully then the imagination of automatic cars and vehicles will convert into reality. Our work is a step ahead to fulfill that imagination. In object tracking various problem come in the way, one of them is the light intensity variation which results in false tracking of object. In this work illumination variation problem in object detection is tackled. Kalman filter is used in combination to track the object. Basically the problem lies with the foreground subtraction and in Kalman filter also the smallest rectangle coordinates are predicted and updated. If there is illumination variance, then false object detection may take place. That's why we develop an algorithm which is resist to illumination variation and gives rise to correct tracking of object. For this we developed a thresholding algorithm which can compensate the illumination variation. This also reduces the time consumed in operation. The difference between Otsu's global thresholding and ours thresholding in kalman filter are clearly shown in chapter 5.

A background registration technique is used to construct reliable background information from the video sequence. Then, each incoming frame is compared with the background image. If the luminance value of a pixel differs significantly from the background image, the pixel is marked as moving object; otherwise, the pixel is regarded as background. The adaptive background thresholding algorithm is used which uses gray level co-occurrence matrix and local mean to calculate the threshold value corresponding to each pixel. This is called local processing and results are compared with global processing like Otsu's threshold method. Shadow effect is a problem in many change detection based segmentation algorithms. In the proposed algorithm, a morphological gradient operation is used to filter out the shadow area while preserving the object shape. In order to achieve the real-time requirement for many multimedia communication systems, our algorithm avoids the use of computation intensive operations.

References

[1] Weilong Chen, Meng JooEr, Member, IEEE, and Shiqian Wu, Member, IEEE, "Illumination Compensation and Normalization for Robust Face Recognition Using Discrete Cosine Transform in Logarithm Domain" IEEE Transactions On Systems, Man, And Cybernetics—Part B: Cybernetics, Vol. 36, No. 2, April 2006

[2] Harsha Varwani Heena Choithwani, "Understanding various Techniques for Background Subtraction and Implementation of Shadow Detection" IJCTA Vol 4 (5),822-827,2006

[3] Virendra P. Vishwakarma, Sujata Pandey Member IEEE, and M. N. Gupta, "A Novel Approach for Face Recognition Using DCT Coefficients Re-scaling for Illumination Normalization" 15th International Conference on Advanced Computing and Communications © 2007 IEEE

[4] D. Forsyth, P. Torr, and A. Zisserman (Eds.): ECCV 2008, Part III, LNCS 5304, pp. 276–289, 2008 ©Springer-Verlag Berlin Heidelberg 2008

[5] CHI Jian-nan, ZHANG Chuang, ZHANG Han, LIU Yang, YAN Yan-Tao, "Approach of Moving Objects Detection in Active Video Surveillance" Joint 48th IEEE Conference on Decision and Control and 28th Chinese Control Conference Shanghai, P.R. China, December 16-18, 2009

[6] Caius SULIMAN, Cristina CRUCERU, Florin MOLDOVEANU, "Kalman Filter Based Tracking in a Video Surveillance System" 10th International Conference on Development and Application Systems, Suceava, Romania, May 27-29, 2010

[7] Parisa Darvish Zadeh Varcheie, Michael Sills-Lavoie and Guillaume-Alexandre Bilodeau, "A Multiscale Region-Based Motion Detection and Background Subtraction Algorithm" Sensors 2010, ISSN 1424-8220, 1041-1061

[8] Prashant P. Baveja, Drew N. Maywar, Member, IEEE, Aaron M. Kaplan, and Govind P. Agrawal, Fellow, IEEE "Self-Phase Modulation in Semiconductor Optical Amplifiers: Impact of Amplified Spontaneous Emission" IEEE journal of quantum electronics, VOL. 46, NO. 9, SEPTEMBER 2010

[9] Virendra P. Vishwakarma, Sujata Pandey and M. N. Gupta, "An Illumination Invariant Accurate Face Recognition with Down Scaling of DCT Coefficients" Journal of Computing and Information Technology - CIT 18, 2010, 1, 53–67

[10] Vinayak G Ukinkar, Makrand Samvatsar, "Object detection in dynamic background using image segmentation: A review" IJERA, ISSN: 2248-9622 Vol. 2, Issue 3, May-Jun 2012, pp.232-236

[11] Kalyan Kumar Hati, Pankaj Kumar Sa, and Banshidhar Majhi, "Intensity Range Based Background Subtraction for Effective Object Detection" IEEE Signal Processing Letters, Vol. 20, No. 8, August 2013,Pp759-762

[12] Hemavathy R, Dr. Shobha G, "Object Detection and Tracking under Static and Dynamic environment: A Review" International Journal of Advanced Research in Computer and Communication Engineering Vol. 2, Issue 10, October 2013pp4095-4100

[13] Deepak Kumar Rout, Sharmistha Puhan, "Video Object Detection in Dynamic Scene using Inter-Frame Correlation based Histogram Approach" International Journal of Computer Applications (0975 – 8887) Volume 82 – No 17, November 2013, pp19-24

[14] Farah Yasmin Abdul Rahman, AiniHussain, WanMimiDiyanaWanZaki, HalimahBadiozeZaman, and NooritawatiMdTahir, "Enhancement of Background Subtraction Techniques Using a Second Derivative in Gradient Direction Filter" Hindawi Publishing Corporation Journal of Electrical and Computer Engineering Volume 2013

[15] Olga Zoidi, Anastasios Tefas, Member, IEEE, and Ioannis Pitas, Fellow, IEEE "Visual Object Tracking Based on Local Steering Kernels and Color Histograms" IEEE transaction on circuits and system for video technology VOL:25 NO:3 YEAR 2013.

[16] Bo Liu, Yan Lin, Guan Guan, "A Method of Multi-scale Edge Detection for Underwater Image" Journal of Information & Computational Science 10: 2 (2013) 345–354

[17] Pranam Janney and Glenn Geers, "A Robust Framework for Moving-Object Detection and Vehicular Traffic Density Estimation" Arxiv:1402.0289v1 [Cs.CV] 3 Feb 2014

[18] Saranya M*, Padmavathi S** "Face Tracking in Video by Using Kalman Filter" Saranya M Int. Journal of Engineering Research and Applications ISSN: 2248-9622, Vol. 4, Issue 6(Version 3), June 2014, pp.54-58.

[19] Barga Deoriand Dalton Meitei Thounaojam "A Survey on object tracking in video" International Journal on Information Theory (IJIT), Vol.3, No.3, July 2014

[20] Malik M. Khan, Tayyab W. Awan, Intaek Kim, and YoungsungSoh, "Tracking Occluded Objects Using Kalman Filter and Color Information" International Journal of Computer Theory and Engineering, Vol. 6, No. 5, October 2014

[21] Y.-P. Guan, "Motion Objects Segmentation and Shadow Suppressing without Background Learning" Hindawi Publishing Corporation Journal of Engineering Volume 2014

[22] Amr M. Nagy1, Ali Ahmed2 and Hala H. Zayed3 "A Robust approach of object tracking based on particle filter and optimised likelihood" International Journal of Emerging Technologies in Computational and Applied Sciences (IJETCAS)

[23] Horst Eidenberger "Illumination-invariant Face Recognition by Kalman Filtering" Vienna University of Technology, Favoritenstrasse 9-11, 1040Vienna, Austria.

[24] Brendan Klare, SudeepSarkar "Background Subtraction in Varying Illuminations Using an Ensemble Based on an Enlarged Feature Set" Dept of Computer Science and Engineering University of South Florida.

[25] Hua Yang Greg Welch Marc Pollefeys "Illumination Insensitive Model-Based 3D Object Trackingand Texture Refinement" Computer Science DepartmentUniversity of North Carolina at Chapel Hill.

[26] Dwarikanath Mahapatra1, Mukesh Kumar Saini2 and Ying Sun1 "Illumination Invariant tracking in office environment using neirobiology-saliency based particle filter" National University of Singapore.

[27] Christian Ku¨ blbeck*, Andreas Ernst "Face detection and tracking in video sequences using the modified census transformation" Department of Electronic Imaging, Fraunhofer Institute for Integrated Circuits, Am Wolfs mantel 33, 91058 Erlangen, Germany

[28] Stephen Se, David Lowe, Jim Little "Vision-based Mobile Robot Localization And Mapping using Scale-Invariant Features" Department of Computer Science University of British Columbia Vancouver, B.C. V6T 1Z4, Canada.

[29] Teresa Ko, Stefano Soatto, and Deborah Estrin "Background Subtraction on Distributions Vision" Lab Computer Science Department University of California, Los Angeles405 Hilgard Avenue, Los Angeles – CA 90095

[30] Wei-Kai Chan and Shao-Yi Chien "Real-Time Memory-Efficient Video Object Segmentation in Dynamic Background with Multi-Background Registration Technique" Graduate Institute of Electronics Engineering and Department of Electrical Engineering National Taiwan University