A Comparative Study of Different Background Subtraction Techniques for Dynamic Texture Scenes

Shanthi K P

***Abstract*— A common method for real-time segmentation of moving regions in image sequences involves "background subtraction," or thresholding the error between an estimate of the image without moving objects and the current image. The numerous approaches to this problem differ in the type of background model used and the procedure used to update the model. The latest being the one using fuzzy color histogram. Traditional background modelling and subtraction methods have a strong assumption that the scenes are of static structures with limited perturbation. Background subtraction is a very popular approach for detecting moving objects from a still scene. For this, most of previous methods depend on the assumption that the background is static over short time periods. However, structured motion patterns of the background (e.g., waving leaves, spouting fountain, rippling water, etc.), which are distinctive from variations due to noise, are hardly tolerated in this assumption and thus still lead to high-level false positive rates when using previous models. In this paper five different methods is being described.**

***Index Terms*— Fuzzy Histogram , Thresholding.**

I. INTRODUCTION

With increasing interest in high-level safety and security, smart video surveillance systems, which enable advanced operations such as object tracking and behaviour understanding, have been in critical demand. For the success of such systems, background subtraction, one of essential tasks in video surveillance, has been studied in various environments. Background subtraction is a very popular approach for detecting moving objects from a still scene.

The basic idea of earlier work for this task is to evaluate the difference of pixel values between the reference and current frames.To cope with their limitation of more sensitive to small variations, Stauffer and Grimson [2] formulate the distribution of each pixel value over time as a mixture of Gaussians (MoG).Inspired by their probability model and online updating scheme, numerous variants have been proposed over the last decade. These approaches perform well for the static scene even containing gradual illumination changes, however, often fail to exclude various dynamic textures (e.g., waving leaves and rippling water). This is because they assume that the state change of a pixel is derived from noise, not from structured motion patterns. Dynamic textures are sequences of images of moving scenes that exhibit certain stationarity properties in time; these include sea-waves, smoke, foliage, whirlwind etc.

Specifically, Dalley *et al.* propose a generalization scheme of the MoG model [3]..Zhang *et al.* propose the spatiotemporal local binary pattern (STLBP) [4] to model dynamic textures.On the other hand, saliency detection techniques [5] have been recently employed since those have a great ability to detect visually important regions (i.e., moving objects in video sequences) while effectively suppressing irrelevant backgrounds.The latest method being employed includes fuzzy color histogram [1].The rationale behind this model is that color variations generated by background motions are greatly attenuated in a fuzzy manner.

Manuscript received Oct. 27, 2013.

Shanthi K P received Btech in Electronics and Communication Engg from CUSAT University Kerala. Currently pursing Mtech in Signal Processing at Govt Engg college, Cherthala under Cochin University Of Science And Technology.

II. STAUFFER AND GRIMSON MODEL FOR BACKGROUND SUBTRACTION

It should be capable of dealing with movement through cluttered areas, objects overlapping in the visual field, shadows, lighting changes, effects of moving elements of the scene (e.g. swaying trees), slow-moving objects, and objects being introduced or removed from the scene.One approach, which was first introduced by Stauffer and Grimson (SG) and has since gained substantial popularity.

A. Basic approach

Each pixel is modelled as a mixture of Gaussians and using an on-line approximation to update the model. The Gaussian distributions of the adaptive mixture model are then evaluated to determine which are most likely to result from a background process. Each pixel is classified based on whether the Gaussian distribution which represents it most effectively is considered part of the background model [2]. This results in a stable, real-time outdoor tracker which reliably deals with lighting changes, repetitive motions from clutter, and long-term scene changes.

The system adapts to deal robustly with lighting changes, repetitive motions of scene elements, tracking through cluttered regions, slow-moving objects, and introducing or removing objects from the scene. Slowly moving objects take longer to be incorporated into the background, because their color has a larger variance than the background

B. The method

In practice, multiple surfaces often appear in the view frustum of a particular pixel and the lighting conditions change. Thus, multiple, adaptive Gaussians are necessary. Here a mixture of adaptive Gaussians to approximate this process is used. Each time the parameters of the Gaussians are updated, the Gaussians are evaluated using a simple heuristic to hypothesize which are most likely to be part of the

"background process." Pixel values that do not match one of the pixel's "background" Gaussians are grouped using connected components. Finally, the connected components are tracked from frame to frame using a multiple hypothesis tracker.

i. Online mixture model

Consider the values of a particular pixel over time as a "pixel process". The "pixel process" is a time series of pixel values, e.g. scalars for gray values or vectors for color images. At any time, t, what is known about a particular pixel, $\{x_0, y_0\}$, is its history

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

where I is the image sequence. Some "pixel processes" are shown by the (R,G) scatter plots in Figure 1(a)-(b) which illustrate the need for adaptive systems with automatic thresholds. Figure 1(b) also highlight a need for a multi-modal representation.

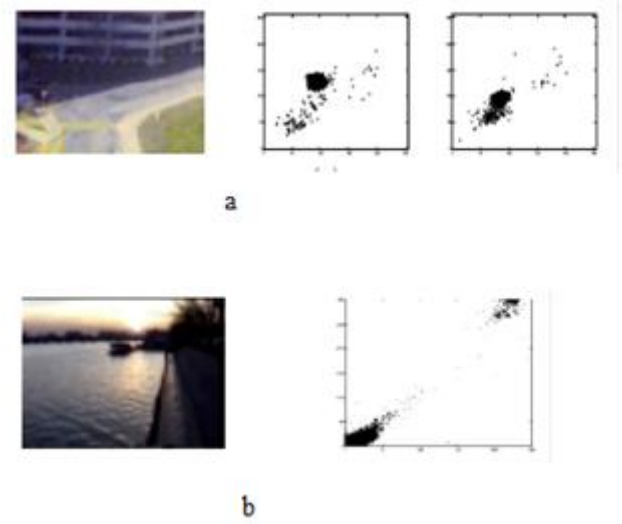

Fig 1: This figure contains images and scatter plots of the red and green values of a single pixel from the image over time. It illustrates some of the difficulties involved in real environments. (a) shows two scatter plots from the same pixel taken 2 minutes apart. This would require two thresholds. (b) shows a bi-model distribution of a pixel values resulting from specularities on the surface of water.

The value of each pixel represents a measurement of the radiance in the direction of the sensor of the first object intersected by the pixel's optical ray. With a static background and static lighting, that value would be relatively constant. If lighting changes occurred in a static scene, it would be necessary for the Gaussian to track those changes. If a static object was added to the scene and was not incorporated into the background until it had been there longer than the previous object, the corresponding pixels could be considered foreground for arbitrarily long periods. This would lead to accumulated errors in the foreground estimation, resulting in poor tracking behaviour. These factors suggest that more recent observations may be more important in determining the Gaussian parameter estimates.

These are the guiding factors in their choice of model and update procedure. The recent history of each pixel, $\{X_1, ...,X_t\}$, is modeled by a mixture of K Gaussian distributions. The probability of observing the current pixel value is

$$P(X_t) = \sum_{i=1}^{K} \omega_{i,t} * \eta(X_t, \mu_{i,t}, \Sigma_{i,t}) \quad (2)$$

where K is the number of distributions, $\omega_{i,t}$ is an estimate of the weight (what portion of the data is accounted for by this Gaussian) of the i^{th} Gaussian in the mixture at time t, $\mu_{i,t}$ is the mean value of the i^{th} Gaussian in the mixture at time t, $\Sigma_{i,t}$ is the covariance matrix of the i^{th} Gaussian in the mixture at time t, and where η is a Gaussian probability density function

$$\eta(X_t, \mu, \Sigma) = \frac{1}{(2\pi)^{\frac{n}{2}} |\Sigma|^{\frac{1}{2}}} e^{-\frac{1}{2}(X_t-\mu_t)^T \Sigma^{-1}(X_t-\mu_t)} \quad (3)$$

K is determined by the available memory and computational power. Currently, from 3 to 5 are used. Also, for computational reasons, the covariance matrix is assumed to be of the form:

$$\Sigma_{k,t} = \sigma_k^2 \mathbf{I} \quad (4)$$

This assumes that the red, green, and blue pixel values are independent and have the same variances. While this is certainly not the case, the assumption allows us to avoid a costly matrix inversion at the expense of some accuracy. Thus, the distribution of recently observed values of each pixel in the scene is characterized by a mixture of Gaussians. A new pixel value will, in general, be represented by one of the major components of the mixture model and used to update the model.

Because there is a mixture model for every pixel in the image, implementing an exact EM algorithm on a window of recent data would be costly. Instead, an on-line K-means approximation is implemented. Every new pixel value, X_t, is checked against the existing K Gaussian distributions, until a match is found. A match is defined as a pixel value within 2.5 standard deviations of a distribution1.

If none of the K distributions match the current pixel value, the least probable distribution is replaced with a distribution with the current value as its mean value, an initially high variance, and low prior weight. The prior weights of the K distributions at time t, $\omega_{k,t}$, are adjusted as follows

$$\omega_{k,t} = (1 - \alpha)\omega_{k,t-1} + \alpha(M_{k,t}) \quad (5)$$

where α is the learning rate2 and $M_{k,t}$ is 1 for the model which matched and 0 for the remaining models. After this approximation, the weights are renormalized. $1/\alpha$ defines the time constant which determines the speed at which the distribution's parameters change. $\omega_{k,t}$ is effectively a causal low-pass filtered average of the (thresholded) posterior probability that pixel values have matched model k given observations time 1 through t. This is equivalent to the expectation of this value with an exponential window on the past values.

The μ and σ parameters for unmatched distributions remain the same. The parameters of the distribution which matches the new observation are updated as follows

$$\mu_t = (1-\rho)\mu_{t-1} + \rho X_t \quad (6)$$

$$\sigma_t^2 = (1-\rho)\sigma_{t-1}^2 + \rho(X_t-\mu_t)^T(X_t-\mu_t) \quad (7)$$

where

$$\rho = \alpha\eta(X_t|\mu_k,\sigma_k) \quad (8)$$

which is effectively the same type of causal low-pass filter as mentioned above, except that only the data which matches the model is included in the estimation. One of the significant advantages of this method is that when something is allowed to become part of the background, it doesn't destroy the existing model of the background. The original background color remains in the mixture until it becomes the K^{th} most probable and a new color is observed. Therefore, if an object is stationary just long enough to become part of the background and then it moves, the distribution describing the previous background still exists with the same μ and $\sigma 2$, but a lower ω and will be quickly re-incorporated into the background.

i. Background Model Estimation

As the parameters of the mixture model of each pixel change, which of the Gaussians of the mixture are most likely produced by background processes is to be determined. Heuristically, the Gaussian distributions which have the most supporting evidence and the least variance is considered.

First, the Gaussians are ordered by the value of ω/σ. This value increases both as a distribution gains more evidence and as the variance decreases. After re-estimating the parameters of the mixture, it is sufficient to sort from the matched distribution towards the most probable background distribution.This ordering of the model is effectively an ordered, open-ended list, where the most likely background distributions remain on top and the less probable transient background distributions gravitate towards the bottom and are eventually replaced by new distributions. Then the first B distributions are chosen as the background model, where

$$B = argmin_b\left(\sum_{k=1}^{b}\omega_k > T\right) \quad (9)$$

where T is a measure of the minimum portion of the data that should be accounted for by the background. This takes the "best" distributions until a certain portion, T, of the recent data has been accounted for. If a small value for T is chosen, the background model is usually unimodal. If this is the case, using only the most probable distribution will save processing. If T is higher, a multi-modal distribution caused by a repetitive background motion (e.g. leaves on a tree, a flag in the wind, a construction flasher, etc.) could result in more than one color being included in the background model. This results in a transparency effect which allows the background to accept two or more separate colors.

i. Connected components

The method described above allows to identify foreground pixels in each new frame while updating the description of each pixel's process. These labelled foreground pixels can then be segmented into regions by a two-pass, connected components algorithm. Because this procedure is effective in determining the whole moving object, moving regions can be characterized not only by their position, but size, moments, and other shape information. Not only can these characteristics be useful for later processing and classification, but they can aid in the tracking process.

C. Results

On an SGI O2 with a R10000 processor, this method can process 11 to 13 frames a second (frame size 160x120pixels). The variation in the frame rate is due to variation in the amount of foreground present. The tracking system has been effectively storing tracking information for five scenes for over 16 months.

D. Drawbacks

One of its main drawbacks is the assumption that the background is static over short time scales. This is a strong limitation for scenes with spatiotemporal dynamics. Although the model allows each pixel to switch state, and tolerates some variability within the state, the Gaussian mixture assumes that the variability derives from noise, not the structured motion patterns that characterize moving water, burning fire, swaying trees, etc.Due to lack of consistency between the state of adjacent pixels, which sometimes leads to noisy foreground-background segmentations.

III. GENERALIZED STAUFFER GRIMSON BACKGROUND SUBTRACTION FOR DYNAMIC SCENES

A number of extensions to the background subtraction method of SG have been appeared. One of its main drawbacks is the assumption that the background is static over short time scales. This is a strong limitation for scenes with spatiotemporal dynamics.One approach that has shown promise for modeling these spatiotemporal dynamic processes is the dynamic texture representation. Dynamic textures model a spatiotemporal volume as a sample from a linear dynamical system (LDS),and have shown surprising robustness for video synthesis, classification , segmentation , and image registration.

While able to capture background dynamics, these approaches lack the two most compelling (and dynamic) aspects of the SG method: (1) the ability to account for transitory events, due to motion of foreground objects; and (2) simple model management.

Fig. 2 A scene with dynamic background. The background consists of water waves, which are changed by the turbulent wake of a boat

Consider, for example, the aquatic scene of Fig. 2. As the jet-skier traverses a video patch, the video goes through the following state sequence: normal waves, occluded by jet-ski, turbulent waves that trail the jet-ski, return to normal waves. In the absence of a hidden discrete state variable, the dynamic texture will slowly interpolate through all these states. Both the transition from occluded to turbulent, and turbulent to normal waves, will generate outliers which are incorrectly

marked as foreground. If the jet-ski cyclically passes through the same location, these errors will repeat with the same periodicity.

In summary, background subtraction requires both a state based representation (as in SG) and the ability to capture scene dynamics within the state (as in the dynamic texture methods). This suggests a very natural extension of the two lines of work: to represent spatiotemporal video cubes as samples from a mixture of dynamic textures . However, as in the static case, exact learning of the mixture parameters is computationally infeasible in an online setting. To address this problem,here two observations are combined. The first is the main insight of SG: that parameter updates of a Gaussian mixture model only require a small set of sufficient statistics. The second is that, because the dynamic texture is a member of the exponential family , it exhibits the same property.

The generalized SG (GSG) algorithm [3] inherits the advantages : (1) it adapts to long-term variations via online estimation; (2) it can quickly embrace new background motions through the addition of mixture components; and (3) it easily discards outdated information by dropping mixture components with small priors.

A. *Basic approach*

This model is used to extend the background subtraction algorithm of SG to dynamic scenes. An overview of the proposed algorithm is shown in Fig. 2. Each video location is represented by a spatiotemporal neighbourhood, centered at that location (in this work we use a 7 × 7 × 5 volume). The background scene is modeled as a mixture of K dynamic textures, from which spatiotemporal volumes are drawn. The j^{th} dynamic texture is denoted (both its parameters and image statistics) by Θj , and a prior weight ω_j , s.t. $\sum_{j=1}^{K} \omega j = 1$, is associated with each dynamic texture. Given a spatiotemporal observation $Y_{1:\tau} \in R^{m\times\tau}$ (m = 49 and τ = 5 in all experiments reported), the location is marked as background if the log-likelihood of the observation under an "active" background component is above a threshold. The background model is then updated, using an online approximation to EM. As in SG, this consists of updating the mixture component with the largest log-likelihood of generating the observation $Y_{1:\tau}$, if the log-likelihood is above a second threshold. If not, a new dynamic texture component learned from $Y_{1:\tau}$ replaces the mixture component with lowest prior weight.

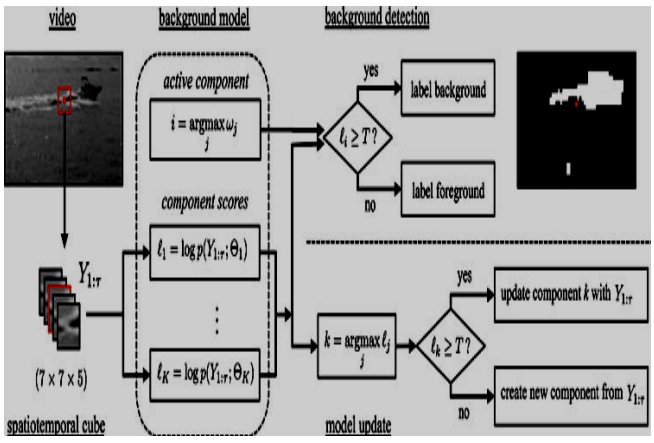

Fig. 3 Overview of generalized Stauffer–Grimson background modelling for dynamic textures. A video location is represented by a neighbouring spatiotemporal volume $Y1{:}\tau$. The location is marked as background if the log-likelihood of $Y1{:}\tau$ under the active background component is above a threshold. Next, the mixture component with the largest log likelihood of generating $Y1{:}\tau$ is updated, if the log-likelihood is above threshold. Otherwise, a new component is learned, replacing the component of lowest prior probability

B. *Background detection*

The determination of whether a location belongs to the background requires the assignment of mixture components to background and foreground. Support for multiple background components is crucial for SG because background colors can switch quickly (e.g. a flashing light). Under the dynamic texture mixture model, rapid changes in color and texture are modeled by the dynamic texture components themselves. It follows that multiple active background components are not necessary, and we simply select the component of largest prior as the "active" background component.

$$i = \underset{j}{\text{argmax}}\, w_j \quad \text{———}$$

A video location is marked as belonging to the background if the log-likelihood of the corresponding spatiotemporal volume $Y_{1:\tau}$ under this mixture component is greater than a threshold T ,

$$\log p(Y_{1:\tau}|\Theta_i) \geq T. \quad \text{———}$$

Note that the log-likelihood can be rewritten in "innovation" form

$$\log p(Y_{1:\tau};\Theta_i) = \log p(y_1;\Theta_i) + \sum_{t=2}^{\tau} \log p(y_t|Y_{1:t-1};\Theta_i), \quad \text{—}$$

which can be efficiently computed with recourse to the Kalman filter.

C. *On-line updating of the background model*

The background mixture model is learned with an online K-means algorithm. During training, an initial dynamic texture Θ_1 is learned, and mixture weights are set to ω = [1, 0, . . . , 0]. Given a new spatiotemporal observation $Y_{1:\tau}$, the mixture parameters are updated as follows. First, the mixture component with the largest loglikelihood of having generated the observation.

$$k = \underset{j}{\text{argmax}} \log p(Y_{1:\tau};\Theta_j) \quad \text{———} \quad 4$$

is selected. If this log-likelihood is above the threshold T

$$\log p(Y_{1:\tau};\Theta_k) \geq T, \quad \text{———} \quad 5$$

the sufficient statistics of the kth component are combined with the sufficient statistics $\{\Phi', \phi', \varphi', \psi', \eta', \xi'\}$ derived from $Y_{1:\tau}$

$$\begin{aligned} \Phi &\leftarrow (1-\alpha)\Phi + \alpha\Phi', & \eta &\leftarrow (1-\alpha)\eta + \alpha\eta', \\ \phi &\leftarrow (1-\alpha)\phi + \alpha\phi', & \xi &\leftarrow (1-\alpha)\xi + \alpha\xi', \\ \psi &\leftarrow (1-\alpha)\psi + \alpha\psi', & \bar{y} &\leftarrow (1-\alpha)\bar{y} + \alpha\bar{y}', \\ \varphi &\leftarrow (1-\alpha)\varphi + \alpha\varphi'. \end{aligned} \quad 6$$

As before, α is a learning rate which weighs the contribution of the new observation. Finally, the parameters of the mixture component are re-estimated , and prior weights are adjusted according to

$$w_j \leftarrow (1-\beta)w_j + \beta\mathbb{I}(j=k),\ \forall j, \quad 7$$

(and normalized to sum to one). It can be shown that the successive application of the online estimation algorithm is equivalent to batch least-squares estimation with exponentially decaying weights on the observations. This allows the dynamic texture to adapt to slow background changes (e.g. lighting and shadows). If (5) does not hold, the component with smallest prior ($i = \text{argmin}_j w_j$) is replaced by a new dynamic texture learned from $Y_{1:\tau}$. A regularization term σI is added to the sufficient statistics $\{\Phi, \phi, \varphi, \eta\}$, to guarantee a large initial variance (in $\hat{}Q$, $\hat{}S$, and $\hat{}R$). As the component is updated with more observations, the influence of this regularization term vanishes. Finally, prior weights are adjusted according to:

$$\omega_j \leftarrow (1-\beta)\omega_j\mathbb{I}(j \neq i) + \beta\mathbb{I}(j=i),\ \forall j$$

and normalized to sum to one. The learning rate β adjusts the speed at which prior weights change, controlling how quickly a mixture component can become the "active" background component. The online background update algorithm is summarized in Algorithm 1.

Algorithm 1 Online updating of the background model

1: **Input:** Spatiotemporal observation $Y_{1:\tau}$, dynamic texture components $\{\Theta_j\}_{j=1}^{K}$, prior weights ω, learning rates α and β, threshold T.
2: Find closest component: $k = \text{argmax}_j \log p(Y_{1:\tau}; \Theta_j)$
3: **if** $\log p(Y_{1:\tau}; \Theta_k) \geq T$ **then**
4: {Update component k}
5: Update the sufficient statistics of Θ_k with $Y_{1:\tau}$ using (6).
6: Estimate the dynamic texture parameters Θ_k
7: Adjust priors (7) and normalize.
8: **else**
9: {Create new component}
10: Find smallest prior: $k = \text{argmin}_j \omega_j$
11: Compute new sufficient statistics Θ_k from $Y_{1:\tau}$.
12: Estimate the parameters Θ_k
13: Adjust priors (8) and normalize.
14: **end if**

D. Conclusion

While the original algorithm restricts the background model to a Gaussian mixture, which is only suitable for static scenes, the generalization supports models with arbitrary component densities. The only restriction is that these densities can be summarized by sufficient statistics.They have applied the generalized SG model to the case where the component densities are dynamic textures, producing an adaptive background subtraction algorithm based on the mixture of dynamic textures, which is suitable for dynamic scenes. But when the dimension of the spatiotemporal volume, or the number of dynamic texture components, is large it may be impractical to compute and store the required sufficient statistics.

IV. SPATIO-TEMPORAL LOCAL BINARY PATTERNS

In order to model the dynamic scenes using both spatial texture and temporal motion information together, we extend ordinary local binary patterns from spatial domain to spatiotemporal domain, and propose a new online dynamic texture extraction operator, named spatio-temporal local binary patterns (STLBP) [4] . STLBP features have three advantages: 1) it is robust to monotonic gray-scale changes; 2) it is online and very fast to compute; 3) it can extract spatial texture and temporal motion information of a pixel. These three advantages are all very important for modelling the dynamic natural scenes.

A. Spatio-temporal Local Binary Pattern Histogram

Let f_t be the current frame at time t and f_{t-1} be the previous frame at time $t-1$. In the frame f_t, the central pixel is $(x_{t,c}, y_{t,c})$ with grey value gt,c. P equally spaced neighboring pixels $(xt,0, yt,0), \ldots, (xt,P-1\ yt,P-1)$ with grey values $gt,0, \ldots, gt,P-1$ on a circle of radius $RLBP$ in ft are defined to be the spatial neighboring pixels of $(xt,c,\ yt,c)$. In the $ft-1$, the corresponding position pixels of P spatial neighboring pixels is $(xt-1,0,\ yt-1,0), \ldots, (xt-1,P-1,\ yt-1,P-1)$ which are defined to be the P temporal neighboring pixels of $(xt,c,\ yt,c)$ with grey values $gt-1,0, \ldots, gt-1,P-1$.
Local Binary pattern (LBP) is a gray-scale invariant texture description.

$$LBP(x_c, y_c) = \sum_{p=0}^{7} s(g_p - g_c)2^p \quad (1)$$

where gc corresponds to the grey value of the central pixel (xc, yc) and gp to the grey values of the eight neighboring pixels. The function $s(x)$ is defined as follows:

$$s(x) = \begin{cases} 1 & x \geq 0 \\ 0 & x < 0 \end{cases} \quad (2)$$

Two P-bit LBP codes for central pixel (xt,c, yt,c) as follows:

$$LBP^{t}_{P,R}(x_{t,c}, y_{t,c}) = \sum_{p=0}^{P-1} s(g_{t,p} - g_{t,c})2^p \quad (3)$$

$$LBP^{t-1}_{P,R}(x_{t,c}, y_{t,c}) = \sum_{p=0}^{P-1} s(g_{t-1,p} - g_{t,c})2^p \quad (4)$$

$LBP^{t}_{P,R}(x_{t,c}, y_{t,c})$ and $LBP^{t-1}_{P,R}(x_{t,c}, y_{t,c})$ are called spatial and temporal local binary patterns of pixel $(x_{t,c}, y_{t,c})$, respectively. The former extracts the spatial texture features and the latter extracts the motion information of neighboring two frames.

$$H_{t,i} = \sum_{(x,y)\in R} I\{LBP^{t}_{P,R}(x,y) = i\}, i = 0,\ldots,2^{p}-1 \quad (5)$$

$$H_{t-1,i} = \sum_{(x,y)\in R} I\{LBP^{t-1}_{P,R}(x,y) = i\}, i = 0,\ldots,2^{p}-1 \quad (6)$$

where Ht,i and $Ht-1,i$ are the histogram values at i^{th} bin of Ht and$Ht-1$, respectively.Then these two histograms can be sum up to form a spatiotemporal local binary pattern (STLBP) histogram H as follows:

$$H_t = \omega H_{t-1,i} + (1-\omega)H_{t,i}, \qquad i = 0,\ldots,2^{p}-1 \quad (7)$$

where Hi is the histogram value at i^{th} bin of H. Parameter ω is the spatio temporal rate

B. *Background Subtraction Using STLBP*

Based on STLBP histograms, a new method of dynamic background modeling and subtraction is proposed. In proposed method, the dynamic background model of a pixel is built using a group of STLBP histograms. When a new frame is arriving, a STLBP histogram of the pixel can be computed using the current frame and previous one which was stored in memory at last time. Then labeling the pixel and updating its background model is the same as that in texture based method . Notice that the previous frame of the video sequence is just stored in memory to compute STLBP histograms for pixels in the current frame.

Parameter ω in Eq. 7 should be set according to changes degree of the dynamic background. A small value is sufficient for scenes which have small changes, whereas a larger value is required in the scenes which have strong changes.

C. *Drawbacks*

However, such methods using filter responses require many parameters to be estimated for detecting saliency in dynamic scenes.

V. SPATIOTEMPORAL SALIENCY DETECTION MODEL –PQFT

Although previous approaches for detecting salient regions are very diverse, most of them fail to minimize false positives which occur in highly textured background areas. Guo and Zhang proposed a spatiotemporal saliency model called phase spectrum of quaternion Fourier transform (PQFT) [5].

Furthermore, PFT can be easily extended from a two-dimensional Fourier transform to a Quaternion Fourier Transform (QFT) if the value of each pixel is represented as a quaternion composed of intensity, color and motion feature. The added motion dimension allows the phase spectrum to represent spatio-temporal saliency in order to engage in attention selection for videos as well as images. The Phase spectrum of QFT (PQFT) is used to obtain the spatio-temporal saliency map, which considers not only salient spatial features like color, orientation and etc. in a single frame but also temporal feature between frames like motion.

This method can be divided into two stages. First, the image should be represented as a quaternion image which consists of four features. Second, PQFT needs to be calculated in order to obtain the spatio-temporal saliency map. The spatio-temporal saliency map using PQFT considers the features such as motion, color, intensity and orientation mentioned in literature. These features are represented as a quaternion image, which means that they are processed in a parallel way. Thus, it saves a lot of computational costs and is fast enough to meet real-time requirements.

VI. FUZZY COLOR HISTOGRAM

Compared to previous methods using local kernels, the proposed method does not require estimation of any parameters (i.e., nonparametric). This is fairly desirable for achieving the robust background subtraction in a wide range of scenes with spatiotemporal dynamics. Specifically, here the local features from the fuzzy color histogram(FCH) is been proposed [1] . Then, the background model is reliably constructed by computing the similarity between local FCH features with an online update procedure.

A. *Color histogram*

Each histogram bin represents a local color range in the given color space, color histogram represents the coarse distribution of the colors in an image. Two similar colors will be treated as identical provided that they are allocated into the same histogram bin. On the other hand, two colors will be considered totally different if they fall into two different bins even though they might be very similar to each other. This makes color histograms sensitive to noisy interference such as illumination changes and quantization errors.

Color histograms are easy to compute, and they are invariant to the rotation and translation of image content. However, color histograms have several inherent problems for the task of image indexing and retrieval. The first concern is their sensitivity to noisy interference such as lighting intensity changes and quantization errors. The second problem is their high dimensionality on representation. Even with coarse quantization over a chosen color space, color histogram feature spaces often occupy more than one hundred dimensions (i.e., histogram bins) which significantly increases the computation of distance measurement on the retrieval stage.

B. *CCH Versus FCH*

A conventional color histogram (CCH) considers neither the color similarity across different bins nor the color dissimilarity in the same bin. Therefore, it is sensitive to noisy interference such as illumination changes and quantization errors. Furthermore, CCHs large dimension or histogram bins requires large computation on histogram comparison. To address these concerns, a new color histogram representation, called *fuzzy color histogram* (FCH) [6] is been presented, by considering the color similarity of each pixel's color associated to all the histogram bins through fuzzy-set membership function.

C. *Mathematical approach*

The color histogram is viewed as a color distribution from the probability viewpoint. Given a color space containing n

color bins, the color histogram of image I containing N pixels is represented as , $H(I) = [h_1, h_2, \ldots, h_n]$, where $h_i=N_i/N$ is the probability of a pixel in the image belonging to the ith color bin, and N_i is the total number of pixels in the i^{th} color bin. According to the total probability theory, h_i can be defined as follows:

$$h_i = \sum_{j=1}^{N} P_{i|j} P_j = \frac{1}{N} \sum_{j=1}^{N} P_{i|j}$$

where P_j is the probability of a pixel selected from image I being the j^{th} pixel, which is 1/N, and $P_{i/j}$ is the conditional probability of the selected j^{th} pixel belonging to the i^{th} color bin.

In the context of CCH, is defined as

$$P_{i|j} = \begin{cases} 1, & \text{if the } j\text{th pixel is quantized into the } i\text{th color bin} \\ 0, & \text{otherwise.} \end{cases}$$

This definition leads to the boundary issue of CCH such that the histogram may undergo abrupt changes even though color variations are actually small. This reveals the reason why the CCH is sensitive to noisy interference such as illumination changes and quantization errors.

Instead of using the probability $P_{i/j}$, FCH consider each of the N pixels in image I being related to all the n color bins via fuzzy-set membership function such that the degree of "belongingness" or "association" of the j^{th} pixel to the i^{th} color bin is determined by distributing the membership value of the j^{th} pixel,μ_{ij} , to the i^{th} color bin.

Definition (Fuzzy Color Histogram): The fuzzy color histogram (FCH) of image I can be expressed as $F(I)=[f_1,f_2,\ldots,f_n]$, where

$$f_i = \sum_{j=1}^{N} \mu_{ij} P_j = \frac{1}{N} \sum_{j=1}^{N} \mu_{ij}.$$

P_j has been defined as before, and μ_{ij} is the membership value of the j^{th} pixel in the ith color bin.

D. *Fuzzy Membership Based Local Histogram Features*

First of all, in a probability view, the conventional color histogram (CCH) can be regarded as the probability density function. Thus, the probability for pixels in the image to belong to the i^{th} color bin w_i can be defined as follows:

$$h_i = \sum_{j=1}^{N} P(\omega_i \mid \mathbf{x}_j) P(\mathbf{x}_j) = \frac{1}{N} \sum_{j=1}^{N} P(\omega_i \mid \mathbf{x}_j) \qquad 1$$

where N denotes the total number of pixels.$P(x_j)$ is the probability of color features selected from a given image being those of the j^{th} pixel, which is determined as 1/N. The conditional probability $P(w_i/x_j)$ is 1 if the color feature of the selected j^{th} pixel is quantized into the i^{th} color bin and 0 otherwise.In contrast to that, FCH utilizes the fuzzy membership to relax such a strict condition.

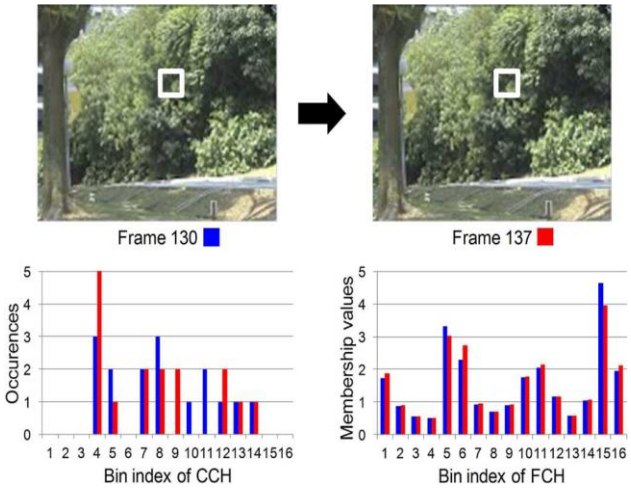

Fig. 4. Comparison of CCH and FCH between the 130th frame and the 137th frame in the "campus" sequence [14] containing strongly waving leaves. Note that CCH and FCH are obtained from the local region of 4 4 pixels illustrated as the white square.

Fig. 4 shows the robustness of local FCH to dynamic textures compared to CCH. As can be seen, local CCHs obtained from the same pixel position of two video frames are quite different due to strongly waving leaves. In contrast to that, we confirm that FCH provides relatively consistent results even though dynamic textures are widely distributed in the background. Therefore, it is thought that our local FCH features are very useful for modeling the background in dynamic texture scenes.

Now, a rather critical issue is to efficiently compute such membership values. Here we employ a novel color quantization scheme based on the fuzzy –c means (FCM) clustering technique: First, the RGB color space is uniformly and finely quantized into m histogram bins (e.g., 4096) and subsequently convert them into the CIELab color space. Finally, we classify these m colors in the CIELab color space to clusters (each cluster represents an individual FCH bin) using the FCM clustering technique (m>>c). More specifically, the FCM algorithm finds a minimum of a heuristic global cost function defined as follows

$$J = \sum_{i=1}^{c} \sum_{j=1}^{m} [P(\omega_i \mid \mathbf{x}_j)]^b \|\mathbf{x}_j - \mathbf{v}_i\|^2 \qquad —$$

where x and v denote the feature vector (e.g., values of each color channel) and the cluster center, respectively. b is a constant to control the degree of blending of the different clusters and is generally set to 2. Then we have following equations, i.e., $\partial J/\partial vi = 0$ and $\partial J/\partial Pj = 0$ where Pj denotes the prior probability of $P(w_j)$, at the minimum of the cost function. These lead to the solution given as

$$\mathbf{v}_i = \frac{\sum_{j=1}^{m} [P(\omega_i \mid \mathbf{x}_j)]^b \mathbf{x}_j}{\sum_{j=1}^{m} [P(\omega_i \mid \mathbf{x}_j)]^b}, \qquad —$$

$$P(\omega_i \mid \mathbf{x}_j) = u_{ij} = \frac{\left(\frac{1}{d_{ij}}\right)^{1/(b-1)}}{\sum_{r=1}^{c} \left(\frac{1}{d_{rj}}\right)^{1/(b-1)}} \qquad —$$

Where $d_{ij}=\|xj - vi\|^2$ Since (3) and (4) rarely have analytic solutions, these (i.e., cluster center and membership

value) are estimated iteratively. It is worth noting that these membership values derived from (4) only need to be computed once and stored as a membership matrix in advance. Therefore, we can easily build FCH for the incoming video frame by directly referring to the stored matrix without computing membership values for each pixel. For the robust background subtraction in dynamic texture scenes, we finally define the local FCH feature vector at the j[th] pixel position of the k[th] video frame as follows:

$$\mathbf{F}_j(k) = (f^k_{j,1}, f^k_{j,2}, \ldots, f^k_{j,c}), \quad f^k_{j,i} = \sum_{q \in W^k_j} u_{iq} \qquad 5$$

where W^k_j denotes the set of neighboring pixels centered at the position j. u_{iq} denotes the membership value obtained from (4), indicating the belongingness of the color feature computed at the pixel position to the color bin as mentioned. By using the difference of local features defined in (5) between consecutive frames, can build the reliable background model easily.

E. *Background Subtraction With Local FCH Features*

To classify a given pixel into either background or moving objects in the current frame, first compare the observed FCH vector with the model FCH vector renewed by the online update as expressed in (6)

$$B_j(k) = \begin{cases} 1, & \text{if } S(\mathbf{F}_j(k), \hat{\mathbf{F}}_j(k)) > \tau, \\ 0, & \text{otherwise,} \end{cases} \qquad 6$$

where $B_j(k) = 1$ denotes that the j[th] pixel in the k[th] video frame is determined as the background whereas the corresponding pixel belongs to moving objects if $B_j(k) = 0$. T is a thresholding value ranging from 0 to 1. The similarity measure S(.,.) used in (6), which adopts normalized histogram intersection for simple computation, is defined as follows:

$$S(\mathbf{F}_j(k), \hat{\mathbf{F}}_j(k)) = \frac{\sum_{i=1}^{c} \min\left[f^k_{j,i}, \hat{f}^k_{j,i}\right]}{\max\left[\sum_{i=1}^{c} f^k_{j,i}, \sum_{i=1}^{c} \hat{f}^k_{j,i}\right]} \qquad (7)$$

where $\hat{\mathbf{F}}_j(k)$ denotes the background model of the j[th] pixel position in the k[th] video frame, defined in (8). Note that any other metric (e.g., cosine similarity, Chi-square, etc.) can be employed for this similarity measure without significant performance drop.In order to maintain the reliable background model in dynamic texture scenes, need to update it at each pixel position in an online manner as follows:

$$\hat{\mathbf{F}}_j(k) = (1-\alpha) \cdot \hat{\mathbf{F}}_j(k-1) + \alpha \cdot \mathbf{F}_j(k), \quad k \geq 1 \qquad (8)$$

where $\hat{\mathbf{F}}_j(0) = \mathbf{F}_j(0)$. $\alpha \in [0, 1]$. is the learning rate. Note that the larger α denotes that local FCH features currently observed strongly affect to build the background model. By doing this, the background model is adaptively updated.

For the sake of completeness, the main steps of the proposed method is summarized in Algorithm 1.

Algorithm 1 Background subtraction using local FCH features

Step 1. Construct a membership matrix using fuzzy c-means clustering based on (3) and (4) (conducted offline only once).
Step 2. Quantize RGB colors of each pixel at the kth video frame into one of m histogram bins (e.g., rth bin where $r = 1, 2, \ldots, m$).
Step 3. Find the membership value u_{ir} at each pixel position $(i = 1, 2, \ldots, c)$.
Step 4. Compute local FCH features using (5) at each pixel position of the kth video frame.
Step 5. Classify each pixel into background or not based on (6).
Step 6. Update the background model using (8).
Step 7. Go back to Step 2 until the input is terminated $(k = k + 1)$.

F. *Conclusion*

For computing FCHs, here an efficient method based on fuzzy c-means clustering algorithm performed on the color components recorded in the perceptually uniform CIELAB color space is proposed.

From the observation of the interplay between FCH and quadratic histogram distance, proposed FCH not only addresses the noise sensitivity issue of CCH but also avoids intensive online computation encountered in computing the quadratic histogram distance. Finally, exploiting FCH into other image processing frameworks and even extending similar soft clustering approach to other low-level visual features (e.g., shape, texture, etc.) are also recommended.

VII. COMPARISONS

To compare different algorithms discussed earlier three video sequences i.e., campus, fountain (both are 160*128 pixels), and boat sequences ,(180*100 pixels) are obtained. They are taken in outdoor environments with dynamic textures. We set the learning rate α=0.01 and used the local window W^k_j of 5 * 5 pixels to extract local FCH features.m and c are set to 4096 and 16, respectively

Fig. 5 shows several results of background subtraction.Traditional MoG (t-MoG) method, generalized MoG (g-MoG) method,STLBP-based method, PQFT-based method,CCH-based method and FCH based are compared here. To make fair comparisons,experimental results at a true positive (TP) rate of 0.8, which is good enough to correctly extract moving objects for further applications are reported. As can be seen in Fig. 5, previous methods fail to reliably build the background model due to variability from temporally dynamic textures. In particular, waving leaves widely distributed in the background lead to high-level false positive rates. Although some approaches (e.g., STLBP-based, PQFT-based, and CCH-based methods) perform rather stably, those still suffer from abrupt changes in the background such as turbulent water by a boat (see the last row of Fig. 5). In contrast to that, FCH approach provides

reliable background models even in the presence of various dynamic textures.

For the quantitative analysis, we evaluated the false positive (FP) rate at the TP rate of 0.8 based on 20 video frames randomly selected from each test video and corresponding results are shown in Table I. Moreover, we also plot the ROC curve using campus sequences in Fig. 6. From Table I and Fig. 6, we confirm that the FCH method is capable of correctly extracting moving objects with low false positives in dynamic texture scenes.The comparison of the processing time is shown in Table II. We can see that our method achieves about 0.027 s/frame (i.e., 37 fps) on average, which can be applied for various real-time applications, while providing much better background subtraction performance compared to previous methods.

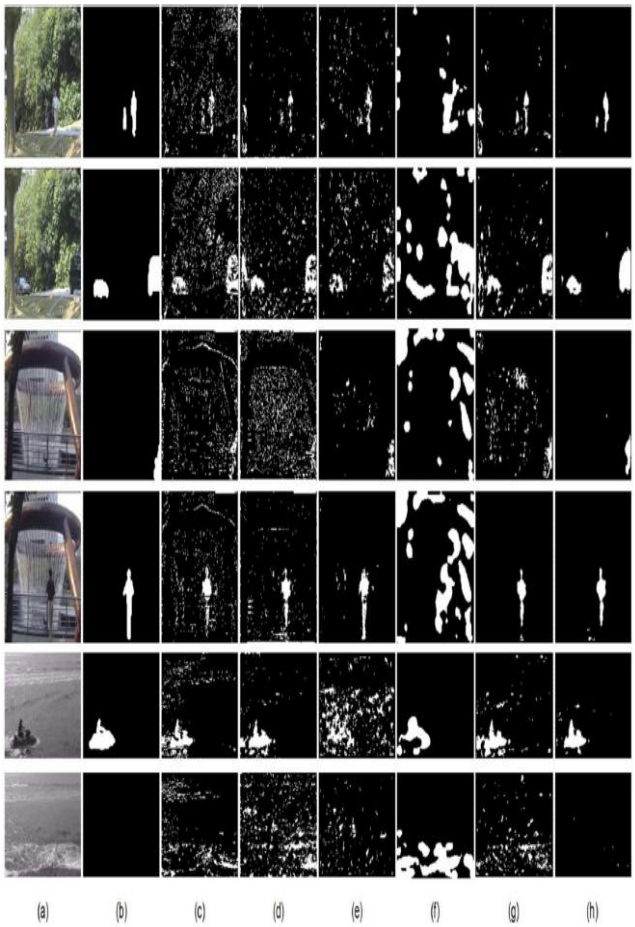

Fig. 5 Examples of background subtraction results obtained from campus (1st and 2nd rows), fountain (3rd and 4th rows), and boat (5th and 6th rows) sequences. (a) Input video frames. (b) ground truth. (c) traditional MoG (t-MoG) method [1]. (d) generalized MoG (g-MoG) method [6]. (e) STLBP-based method [7]. (f) PQFT-based method [9]. (g) CCH-based method. and (h) proposed method (LFCH). Note that background subtraction results of each method are taken with the same level of detection rate (i.e., true positive rate = 0.8).

TABLE I
PERFORMANCE COMPARISON BY FP RATES MEASURED AT THE TP RATE OF 0.8

Methods	t-MoG [1]	g-MoG [6]	STLBP-based [7]	PQFT-based [9]	CCH-based	LFCH-based (proposed)
Campus	0.55	0.51	0.25	0.26	0.14	0.07
Fountain	0.57	0.56	0.07	0.28	0.11	0.08
Boat	0.62	0.22	0.49	0.21	0.61	0.19

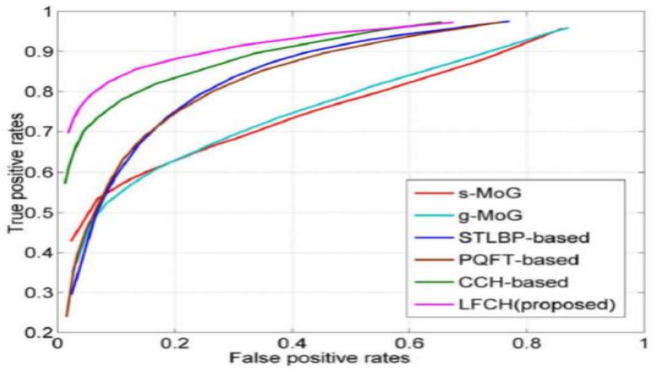

Fig. 6 ROC curve for background subtraction on campus sequences.

TABLE II
PERFORMANCE COMPARISON BY THE AVERAGE PROCESSING TIME PER FRAME

Methods	t-MoG [1]	g-MoG [6]	STLBP-based [7]	PQFT-based [9]	CCH-based	LFCH-based (proposed)
sec	0.011	0.022	0.071	0.018	0.017	0.027

VIII. CONCLUSION

Traditional background modeling and subtraction methods have a strong assumption that the scenes are of static structures with limited perturbation. These methods will perform poorly in dynamic scenes. A simple and robust method for background subtraction for dynamic texture scenes has been proved to be FCH. The basic idea is to adopt FCH in a local manner to minimize color variations generated by background motions. Background subtraction is conducted by computing the similarity between the observed and the model FCH features, renewed by online update procedures. Based on extensive experimental results, it is confirmed that the algorithmusing FCH provides the reliable background model in dynamic texture scenes.

REFERENCES

[1] Background Subtraction for Dynamic Texture Scenes Using Fuzzy Color Histograms -Wonjun Kim, *Student Member, IEEE*, and ChangickKim, *Senior Member*,IEEE SIGNAL PROCESSING LETTERS, VOL. 19, NO. 3, MARCH 2012

[2] C. Stauffer and W. E. L. Grimson, "Adaptive background mixture models for real-time tracking," in *Proc. IEEE Computer Vision and Pattern Recognition (CVPR)*, Jun. 1999, vol. 2, pp. 246–252.

[3] A. B. Chan, V. Mahadevan, and N. Vasconcelos, "Generalized Stauffer Grimson background subtraction for dynamic scenes," *Mach. Vis. Applicat.*, vol. 22, no. 5, pp. 751–766, 2011.

[4] S. Zhang, H. Yao, and S. Liu, "Dynamic background modeling and subtraction using spatio-temporal local binary patterns," in *Proc. IEEE Int. Conf. Image Processing (ICIP)*, Oct. 2008, pp. 1556–1559.

[5] C.Guo and L. Zhang, "Anovelmultiresolution spatiotemporal saliency detection model and its applications in image and video compression," *IEEE Trans. Image Process.*, vol. 19, no. 1, pp. 185–198, Jan. 2010.

[6] J. Han and K.-K.Ma, "Fuzzy color histogram and its use in color image retrieval," *IEEE Trans. Image Process.*, vol. 11, no. 8, pp. 944–952, Nov. 2002.

Shanthi K P received Btech in Electronics and Communication Engg from CUSAT University Kerala. Currently pursing Mtech in Signal Processing at Govt Engg college, Cherthala under Cochin University Of Science And Technology. Areas of interest includes Signal Processing, Image Processing, Artificial Neural Networks.