Adaptive Parametric Statistical Background Subtraction for Video Segmentation

P. Amnuaykanchanasin, T. Thongkamwitoon, N. Srisawaiwilai, and S. Aramvith  
Department of Electrical Engineering, Chulalongkorn University  
Pathumwan, Bangkok 10330, Thailand  
Tel. (662) 218-6909  
Supavadee.A@chula.ac.th

T. H. Chalidabhongse  
Faculty of Information Technology, King Mongkut's Institute of Technology Lardkrabang  
Bangkok 10520, Thailand  
Tel. (662) 727-2551 ext. 526  
thanarat@it.kmitl.ac.th

ABSTRACT
The Background Subtraction Algorithm has been proven to be a very effective technique for automated video surveillance applications. In statistical approach, background model is usually estimated using Gaussian model and is adaptively updated to deal with changes in dynamic scene environment. However, most algorithms update background parameters linearly. As a result, the classification results are erroneous when performing background convergence process. In this paper, we present a novel learning factor control for adaptive background subtraction algorithm. The method adaptively adjusts the rate of adaptation in background model corresponding to events in video sequence. Experimental results show the algorithm improves classification accuracy compared to other known methods.

Categories and Subject Descriptors
I.4.6 [Image Processing and Computer Vision]: Segmentation – pixel classification.

General Terms
Algorithms

Keywords
Unimodal distribution, adaptive background subtraction, pixel classification, non-linear parameters update.

1. INTRODUCTION
Computer vision systems for surveillance application mostly rely on the process of object detection and tracking. In vision based systems, such detection is usually carried out by using background subtraction methods. In early ages of researching, unimodal distribution approaches [1,2,3] play a significant role in background modeling scheme and give satisfactory classification rate as shown in [4]. However, most of them only work in static background scenario, but not in dynamic background scenario which causes many types of error in classification process. As a result, most researchers proposed several works on adaptive background modeling approach. Sophisticated adaptation methods are required to solve major two problems in dynamic scene: changes of illumination and changes of background content. Among several approaches proposed, e.g. [5] bases on Mixture of Gaussian (MOG) model which is originally presented by [6]. MOG approach can solve several problems occurred in dynamic background, especially in case of repetitive background motion such as waving tree. Nevertheless, MOG and other methods proposed update background models using linear model which is not adequately adapted according to the changes in the background scene.

In this paper, we present a novel learning factor control for adaptive background modeling proposed in [7] with capability to intelligently adjust background model parameters according to activity in video scene. The proposed algorithm can be applied to both single Gaussian and MOG. Nevertheless, to simplify the complexity incurred and based on the fact that a few Gaussians in MOG dose not give distinctive classification rate with respect to single Gaussian [4,8], our proposed algorithm in this paper perform adaptive background modeling based on unimodal distribution.

The paper is organized as follows. Section 2 presents background modeling method. Proposed learning factor control algorithm is introduced in Section 3. Section 4 presents conclusion.

2. BACKGROUND MODELING
Our proposed algorithm consists of background modeling and non-linear learning factor control. The system block diagram is shown in Fig. 1.
2.1 Background Model Initialization

First, considering YCbCr color pixels from video sequence. Let \( X_{i,j} = \{X_{i,j}[1],X_{i,j}[2],\ldots,X_{i,j}[N]\} \) be a training sequence of single pixel consisting of N frame. A color vector at the pixel of the \( n^{th} \) frame is depicted as in eq. (1),

\[
X_{i,j}[n] = (X_{i,j}^Y[n], X_{i,j}^{Cb}[n], X_{i,j}^{Cr}[n])
\]  

where \( X_{i,j}^Y[n], X_{i,j}^{Cb}[n], X_{i,j}^{Cr}[n] \) are Y, Cb, Cr components at pixel \((i,j)\) of the \( n^{th} \) frame. Assuming Gaussian noise is incurred in the sampling process. The recent history of each pixel, \( X_{i,j} \), is modeled by Gaussian distribution centered at the mean pixel value. This process is a stationary background modeling process in which we collect \( N \) color vectors for each pixel. Naturally, we obtain two significant parameters automatically. The first one is “Expected Color Vector,” as in eq. (2),

\[
E_{i,j} = E\{X_{i,j}[n]\} ; 1 \leq n \leq N
\]  

where \( E\{\cdot\} \) is expectation operation. So, \( E_{i,j}(E_{i,j}^Y,E_{i,j}^{Cb},E_{i,j}^{Cr}) \) represents mean of color vectors at pixel \((i,j)\) over \( N \) frames. The latter is “Color Covariance Matrix”. The covariance matrix, \( C_{i,j} \), is assumed to be diagonal to reduce computational cost, and can be written as in eq. (3).

\[
C_{i,j} = \Pi[(\sigma_{i,j}^Y)^2 (\sigma_{i,j}^{Cb})^2 (\sigma_{i,j}^{Cr})^2]
\]  

Next, we compute the distortion of \( X_{i,j}[n] \) from its mean, \( E_{i,j} \), by considering two orthogonal distortion parameters, “Brightness Distortion” \( (\alpha_{i,j}[n]) \) and “Color Distortion” \( (\lambda_{i,j}[n]) \).

Brightness distortion implies the brightness intensity of input color vector, \( X_{i,j}[n] \), with respect to the expected color vector, \( E_{i,j} \), and can be obtained as in eq. (4), where \( u_y \) is unit-vector in Y-axis.

\[
\alpha_{i,j}[n] = (X_{i,j}[n] - E_{i,j}) \cdot u_y
\]  

Now, simplifying and normalized, we get

\[
\alpha_{i,j}[n] = \frac{X_{i,j}^Y[n] - E_{i,j}^Y}{\sigma_{i,j}^Y}
\]  

On the other hand, color distortion is defined as the orthogonal distance between input color vector and the reference expected color vector, and is given in eqs. (6)-(7).

\[
\lambda_{i,j}[n] = \frac{\|X_{i,j}[n] - E_{i,j} - \alpha_{i,j}[n]u_Y\|}{\gamma}
\]

\[
\lambda_{i,j}[n] = \sqrt{\frac{X_{i,j}^{Cb}[n] - E_{i,j}^{Cb}\sigma_{i,j}^{Cb}}{\sigma_{i,j}^{Cb}}^2 + \frac{X_{i,j}^{Cr}[n] - E_{i,j}^{Cr}\sigma_{i,j}^{Cr}}{\sigma_{i,j}^{Cr}}^2}
\]

As shown in [3], there are variations of \( \alpha_{i,j}[n] \) and \( \lambda_{i,j}[n] \); and their values are different for different pixels. Thus, to optimize the detection process, we compute two variation parameters: one represents the variation of brightness distortion \( (\alpha_{i,j}) \) and another one represents the variation of color distortion \( (\lambda_{i,j}) \), as defined respectively in eqs. (8) and (9).

\[
a_{i,j} = RMS(\alpha_{i,j}[n]) = \sqrt{\sum_{n=1}^{N} (\alpha_{i,j}[n])^2} \quad (8)
\]

\[
b_{i,j} = RMS(\lambda_{i,j}[n]) = \sqrt{\sum_{n=1}^{N} (\lambda_{i,j}[n])^2} \quad (9)
\]

Then, the initial background model is represented by a “four-tuple” statistical parameters \( \Phi_{i,j} = \{E_{i,j}[n],C_{i,j}[n],a_{i,j}[n],b_{i,j}[n]\} \) for each pixel \((i,j)\).

This background model will be used as an initial model for subtraction. To make the algorithm be able to cope with changes in dynamic scene, adaptive background model update is needed.

2.2 Adaptive Background Model

After initialization, system starts online processing by using set of static background parameters as seed of adaptation \((n=1)\). To deal with changes in the dynamic scene, we update the background model continuously while performing the subtraction. The 4-tuple dynamic model \( \Phi_{i,j}[n] = \{E_{i,j}[n],C_{i,j}[n],a_{i,j}[n],b_{i,j}[n]\} \) are constructed and linearly updated as in eqs. (10)-(13),

\[
E_{i,j}[n] = (1 - \gamma)E_{i,j}[n-1] + \gamma X_{i,j}[n] \quad (10)
\]

\[
C_{i,j}[n] = (1 - \gamma)C_{i,j}[n-1] + \gamma (X_{i,j}[n]-E_{i,j}) (X_{i,j}[n]-E_{i,j})^T \quad (11)
\]

\[
a_{i,j}[n] = \lambda_{i,j}[n] = \sqrt{(1 - \gamma)(a_{i,j}[n-1])^2 + \gamma(a_{i,j}[n])^2} \quad (12)
\]

\[
b_{i,j}[n] = \lambda_{i,j}[n] = \sqrt{(1 - \gamma)(b_{i,j}[n-1])^2 + \gamma(b_{i,j}[n])^2} \quad (13)
\]

where parameter \( \gamma \) can be interpreted as a “Learning factor”. Thus, \( 1 / \gamma \) effectively defines the time constant that implies speed of the model change or update.

3. NON-LINEAR LEARNING FACTOR CONTROL

In previous section, the learning factor of adaptation \( (\gamma) \) defines speed of background model adaptation. If \( \gamma \) is large, the effect of the relocation of background objects, such as moving chair in the office scene, will be updated quickly. At the same time, the true background model might be rapidly lost in the area that has high frequency of moving foreground objects appearance as well as in the
case of moving foreground objects become stationary for a period of time. Thus, an appropriate choice of learning factor, $\gamma$, is required and necessary.

In this research, we propose algorithm for learning factor control which is performed in two distinct levels, i.e., frame level and pixel level. We define learning factor of the $n$th frame, $\gamma_{i,j}[n]$, as a combination of Frame-level learning control factor, $\rho[n]$, for coarse adjustment at frame level and Pixel-level learning control factor, $\varphi_{i,j}[n]$, for fine adjustment at pixel level, as shown in eq. (14),

$$\gamma_{i,j}[n] = K\rho[n]\varphi_{i,j}[n]$$  \hspace{1cm} (14)

where $K$ is user defined gain of control. From now on, we replace learning factor, $\gamma$, in eqs. (10) - (13), with adaptive learning factor $\gamma_{i,j}[n]$ introduced in eq. (14).

### 3.1 Frame Based Learning Factor Control

For the frame level, learning control factor for each frame is selected on each frame based on frame-based activity. We define frame-based activity, $\Pi[n]$, as average frame based change detection of $W \times H$ pixels image, as in eq. (15).

$$\Pi[n] = \left[ \sum_{(i,j)\in W \times H} \left\| X_{i,j}[n] - X_{i,j}[n-1] \right\|^2 \right] / (3 \cdot 255^2 \cdot W \cdot H)$$  \hspace{1cm} (15)

where $\| \Delta \|$ is Euclidean Distance of vector $\Delta$. If the value $\Pi[n]$ is large, i.e., frame-based activity is high, the updated speed should be slow. Thus, $\rho[n]$ is set according to eq. (16).

$$\rho[n] = 1 - \Pi[n]$$  \hspace{1cm} (16)

### 3.2 Pixel Based Learning Factor Control

For pixel level, parameter settings of each pixel are updated based on activity level at each pixel region. The update process is taken into account a spatio-temporal factor. Parameters must be updated as slow as possible if continuous movement occurred in designated pixel. On the other hand, high speed updating is required if designated pixel stays still. In this work, there are three methods to set pixel based learning control factor, $\varphi_{i,j}[n]$, as follows.

#### 3.2.1 Deterministic Control Factor

In this method, we define pixel based control factor by determining High-rate factor, $\delta_H$, and Low-rate Factor, $\delta_L$, by using the following hypothesis: “Active pixel must be updated slower than inactive pixel.” The definition of “active” and “inactive” can be stated by using temporal change detection of pixel $(i,j)$ of two successive frames, as shown in eq. (17).

$$\Delta_{i,j}[n] = X_{i,j}[n] - X_{i,j}[n-1]$$  \hspace{1cm} (17)

Then, change notification mask, $M^{CHN}_{i,j}[n]$, is defined as shown in eq. (18),

$$M^{CHN}_{i,j}[n] = \begin{cases} 0 : \left\| \Delta_{i,j}[n] \right\| < \tau_{CHN} \\ 1 : \left\| \Delta_{i,j}[n] \right\| \geq \tau_{CHN} \end{cases}$$  \hspace{1cm} (18)

where $\tau_{CHN}$ is threshold for changed notification and the value of 18 is empirically chosen. A pixel, $P(i,j)$, is considered as active when $M^{CHN}_{i,j}[n] = 1$. Therefore, pixel based control factor is set, as shown in eq. (19).

$$\varphi_{i,j}[n] = \begin{cases} \delta_H : M^{CHN}_{i,j}[n] = 0 \\ \delta_L : M^{CHN}_{i,j}[n] = 1 \end{cases}$$  \hspace{1cm} (19)

Note that $\delta_H$ and $\delta_L$ are set manually and $0 < \delta_L < \delta_H < 1$.

#### 3.2.2 Temporal Classification Data

For object detection and segmentation, binary classification mask, $M^{C}_{i,j}[n]$, as shown in eq. (20), is used to differentiate pixel as if it belongs to the background or foreground regions.

$$M^{C}_{i,j}[n] = \begin{cases} 0 : \text{Pixel} (i,j) \text{ is Background} \\ 1 : \text{Pixel} (i,j) \text{ is Foreground} \end{cases}$$  \hspace{1cm} (20)

We then define change significance, $\Gamma_{i,j}[n]$, as a difference of binary classification mask in consecutive frames, as shown in eq. (21).

$$\Gamma_{i,j}[n] = M^{C}_{i,j}[n] - M^{C}_{i,j}[n-1]$$  \hspace{1cm} (21)

When $\Gamma_{i,j}[n] = 1$ i.e., there are changes in binary classification mask, it implies that there are new appearance of foreground object or relocation of background content.

Furthermore, we define Last Classification Change as an indicator to set appropriate value of pixel based learning factor control.

#### 3.2.3 Last Classification Change

$L$ is defined as the last classification change when $\Gamma_{i,j}[n-L] = 1$ and $\Gamma_{i,j}[n-\ell] = 0$ while $\ell = 1,2,3,\ldots,L-1$.

Physically, last classification change represents the duration in which pixel classification stay unchanged since the last change occurred. If $L$ is long enough, that pixel is considered to be inactive. Pixel based learning factor control can be set as a condition in eq. (22),

$$\varphi_{i,j}[n] = \begin{cases} \delta_H : L > L_E \\ \delta_L : L \leq L_E \end{cases}$$  \hspace{1cm} (22)

where $L_E$, an acceptable equi-static duration, implies the minimum duration since the last change that pixel stays unchanged.

In practical sense, the efficiency of each pixel based learning control factor depends highly on the nature of each scene. Thus, we heuristically propose a condition for pixel based learning control factor based on deterministic control factor and temporal classification, as shown in eq. (23). In the experiments, the parameters are empirically chosen as follows: $\delta_H = 0.007$, $\delta_L = 0.001$, $L_E = 60$. 

---

65
\[ \varphi_{i,j}[n] = \begin{cases} \delta_{H} : M_{i,j}^{\text{off}}[n] = 0 \text{ and } L > L_{k} \\ \delta_{L} : M_{i,j}^{\text{off}}[n] = 1 \end{cases} \] (23)

### 3.3 Online Subtraction and Classification Process

This section describes real-time subtraction process and pixel classification. We start by initializing the online background model by the background model (its seed). For each input nth frame, we compute \( \alpha_{i,j}[n] \) and \( \lambda_{i,j}[n] \) using Eqs. (5) and (7), and normalize them by on-line background parameters as in eqs. (24) and (25).

\[ \hat{\alpha}_{i,j}[n] = \frac{\alpha_{i,j}[n]}{a_{i,j}[n]} \] (24)

\[ \hat{\lambda}_{i,j}[n] = \frac{\lambda_{i,j}[n]}{b_{i,j}[n]} \] (25)

Then, pixel mask \( M_{i,j}[n] \) can be classified into 4 classes: \( B \): Background, \( F \): Foreground, \( S \): Shadow, \( H \): Highlight by these conditions, as in eq. (26).

\[ M_{i,j}[n] = \begin{cases} F : \hat{\lambda}_{i,j}[n] > \tau_{\lambda} \text{ or } \hat{\alpha}_{i,j}[n] < \tau_{\alpha_{0}}, \text{else} \\ B : \hat{\alpha}_{i,j}[n] < \tau_{\alpha_{0}} \text{ and } \hat{\alpha}_{i,j}[n] > \tau_{\alpha_{2}}, \text{else} \\ S : \alpha_{i,j}[n] < 0, \text{else} \\ H : \text{otherwise} \end{cases} \] (26)

where

\( \tau_{\lambda} \) is background chrominance threshold,

\( \tau_{\alpha_{0}} \) is upper background luminance threshold, and

\( \tau_{\alpha_{2}} \) is lower background luminance threshold

These thresholds are calculated based on a given detection error-rate \( (r) \) provided by the user. \( \tau_{\alpha_{0}} \) is user-defined threshold to limit degree of shadow in case of too dark objects. In the experiments, these thresholds are empirically chosen to be as follows: \( \tau_{\lambda} = 18 \), \( \tau_{\alpha_{0}} = -10 \), \( \tau_{\alpha_{2}} = 15 \).

### 4. CONCLUSIONS

The novel learning factor control for adaptive background subtraction has been introduced in this paper. The algorithm gives solution for adaptively adjusted adaptation rate of background model. Learning factor control is performed in pixel level and frame level. To further simplify the computational complexity required, the adaptive background model could be adequately represented by the expected color vector, \( E_{i,j}[n] \), where \( C_{i,j} = a_{i,j}[n] = b_{i,j}[n] = 1 \) can be set with negligible tradeoff in the classification results.

The experimental results of algorithm shown without simplification in this paper could be found in [9,10] where the improvement in system performance and the reduction of false positive are shown compared to other known algorithms.

### 5. ACKNOWLEDGMENTS

This work is supported in part by the Cooperation Project between Department of Electrical Engineering and Private Sector for Research and Development, Chulalongkorn University.

### 6. REFERENCES


