Real-Time Background Subtraction using Adaptive Thresholding and Dynamic Updating for Biometric Face Detection

K. Sundaraj
University Malaysia Perlis
School of Mechatronic Engineering
02600 Jejawi - Perlis
MALAYSIA
kenneth@unimap.edu.my

Abstract: Face biometrics is an automated method of recognizing a person's face based on a physiological or behavioral characteristic. Face recognition works by first obtaining an image of a person. This process is usually known as face detection. In this paper, we describe an approach for face detection that is able to locate a human face embedded in an outdoor or indoor background. Segmentation of novel or dynamic objects in a scene, often referred to as background subtraction or foreground segmentation, is a critical early step in most computer vision applications in domains such as surveillance and human-computer interaction. All previous implementations aim to handle properly one or more problematic phenomena, such as global illumination changes, shadows, highlights, foreground-background similarity, occlusion and background clutter. Satisfactory results have been obtained but very often at the expense of real-time performance. We propose a method for modeling the background that uses per-pixel time-adaptive Gaussian mixtures in the combined input space of pixel color and pixel neighborhood. We add a safety net to this approach by splitting the luminance and chromaticity components in the background and use their density functions to detect shadows and highlights. Several criteria are then combined to discriminate foreground and background pixels. Our experiments show that the proposed method possesses robustness to problematic phenomena such as global illumination changes, shadows and highlights, without sacrificing real-time performance, making it well-suited for a live video event like face biometric that requires face detection and recognition.

Key–Words: Background Modeling, Face Detection, Biometric Identification.

1 Introduction

Face detection is the first critical processing stage in all kinds of face analysis and modeling applications. These applications have become increasingly attractive in modern life. Video surveillance, facial animation, facial expression analysis, video conferencing, etc. are some of the emerging applications that lure people of both academic and commercial interests over the past decade. Although many laboratory and commercial systems have been developed, most of the proposed methods aim at detecting faces over still images. Since still images convey only visual cues of colors, textures and shapes in the spatial domain, the difficulty and complexity in detecting faces surges as the conveyed spatial-domain cues become vague or noisy. Observing the above-mentioned applications, we find that the major source of faces to be detected is in video data. Video data carries not only the spatial visual information, but also the temporal motion information. The addition of temporal motion information removes some of the ambiguity suffered in spatial visual cues, lowering down the difficulty in face detection. Many of the currently developed systems design their algorithms by trading off detection accuracy for higher speeds, or vice versa. With the inclusion of temporal motion information, many complex situations that require lots of processing time or suffer from detection accuracy become simpler. Therefore, incorporating both spatial and temporal cues for face detection offers an inspirational solution to the problem.

Background subtraction is a method that takes advantage of both spatial and temporal cues to identify and track regions of interest. If these regions of interest, also known as foreground object, can be detected precisely and effectively, then subsequent image processing stages will be presented with a much limited processing area within an image. This reduction will lead to better efficiency, accuracy and computational cost for the complete vision system. Within the literature, various techniques that employ background subtraction can be found. Most of these techniques

use a background reference image to perform background subtraction. This reference image is obtained after the background is mathematically modeled. In the final step, the current image is *subtracted* from the reference image to produce a mask that highlights all foreground objects. The process of image acquisition, background modeling and finally subtracting the current image from the background reference image is implemented in what is known as the background subtraction algorithm. Needless to say, a proper combination of the three is required to obtain satisfactory results for a specific application.

Several background subtraction algorithms have been proposed in the recent literature. All of these methods try to effectively estimate the background model from the temporal sequence of the frames. One of the simplest algorithms is frame differencing [1]. The current frame is subtracted from the previous frame. This method was extended such that the reference frame is obtained by averaging a period of frames [2] [3] [4] also known as median filtering. A second extension applied a linear predictive filter to the period of frames [5] [6]. A disadvantage of this method is that the coefficients used (based on the sample covariance) needs to be estimated for each incoming frame, which makes this method not suitable for real-time operation. [7] and [8] proposed a solution to this using a much higher level algorithm for removing background information from a video stream. A further computationally improved technique was developed by [9] and it is reported that this method is successfully applied in a traffic monitoring system by [10]. In these types of time averaging algorithms, the choice of temporal distance between frames becomes a tricky question. It depends on the size and speed of the moving object. According to [11], background subtraction using time averaging algorithms, at best, only tell where the motion is. Though this is the simplest algorithm, it has many problems; interior pixels of a very slow-moving object are marked as background (known as the aperture problem) and pixels behind the moving object are cast as foreground (known as the ghost effect problem). Multi-model algorithms were then developed to solve this shortcoming. A parametric approach which uses a single Gaussian distribution [12] or multiple Gaussian distribution [13] [14] [15] can be found in the literature. Various improvements techniques like the Kalman filter to track the changes in illumination [16] [17], updating of Gaussian distributions [18] [19], inclusion of image gradient [20] and the modeling and detection of shadows and highlights [21] [22] [23] have been done to improve the accuracy of background subtraction. Non-parametric approaches have also been attempted in [24] and [25]. These approaches use a kernel function to estimate the density function of the background images.

Face biometric applications require the modeling of environmental lighting changes, shadows and reflections that appear on a face and the background. In the case of online surveillance, the application must operate in real-time. Although much as been done on individual techniques as can be seen in the numerous methods described above, very few have concentrated on real-time capabilities [26]. Very few of the above mentioned methods can be executed at a frequency of more that 15Hz. Like most image processing applications, a trade-off has to be made between speed and accuracy in order to obtain an advantage of one over the other. Hence, the focus of this paper is the development of a background subtraction algorithm which can be run in real-time and is accurate enough for the purpose of face detection to be used in a face biometric application. We propose solutions with low computational cost to solve problems like illumination changes, static thresholds, shadows, model updating and background clutter. Our algorithm is able to perform background subtraction on a image of size 640×480 at a frequency of about 32Hz. Indoor and outdoor experiments are presented at the end of this paper together with the discussion about the results.

2 Background Modeling

The background modeling and the subtraction process is shown in Figure 1. There are four major steps in our background subtraction algorithm; preprocessing, consists of transforming the input images from the raw input video format into a format that can be processed by subsequent steps. Background modeling then learns each video frame to estimate a background model. This background model provides a statistical description of the entire background scene. Foreground detection then identifies pixels in the video frame that cannot be adequately explained by the background model, and outputs them as a binary candidate in a foreground mask. Finally, data validation examines the candidate mask, eliminates those pixels that do not correspond to actual objects, and outputs the final foreground mask. We describe the four major steps in the following subsections.

2.1 Preprocessing

In this stage, we firstly use simple temporal and spacial smoothing to reduce camera noise. Smoothing can also be used to remove transient environmental noise. Then to ensure real-time capabilities, we have to decide on the frame-size and frame-rate which are the determining factors of the data processing rate. Another key issue in preprocessing is the data format

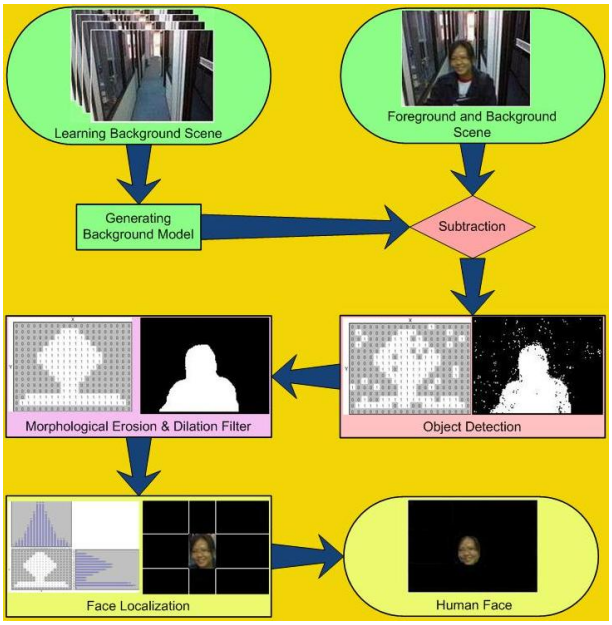


Figure 1: Block diagram of the background subtraction algorithm.

used by the particular background subtraction algorithm. Most of the algorithms handle only luminance intensity, which is one scalar value per each pixel. However, color image, in either RGB or YUV color space, is becoming more popular these days. In the case of a mismatch, some time will be spent on converting the output data from the driver of the camera to the required input data type for the algorithm.

2.1.1 Color Model

The input to our algorithm is a time series of spatially registered and time-synchronized color images obtained by a static camera in the YUV color space. This allows us to separate the luminance and chroma components which has been decoded in a 4:2:0 ratio by our camera hardware. The observation at pixel i at time t can then be written as:

$$I_C = (Y, U, V) \tag{1}$$

2.1.2 Texture Model

In our implementation, we used the texture information available in the image in our algorithm by including and considering the surrounding neighbors of the pixel. This can be obtained several ways. In our implementation, we have decided to take the image gradient of the Y component in the x and y directions. The gradient is then denoted as follows:

$$\begin{aligned} I_G = G_Y &= (G_x, G_y) \\ &= \sqrt{G_x^2 + G_y^2} \end{aligned} \tag{2}$$

This is obtained using a Sobel operator. The Sobel operator combines Gaussian smoothing and differentiation so the result is more robust to noise. We have decided to use the following Sobel filter after experimenting with several other filters.

$$\begin{bmatrix} 1 & 2 & 1 \\ 2 & 0 & 2 \\ 1 & 2 & 1 \end{bmatrix} \tag{3}$$

Figure 2 shows the behavior of the camera sensor to the neighboring pixel and the effect of the surrounding pixels for any given pixel. Notice that the edges of the red, blue and green color zones are not uniform.

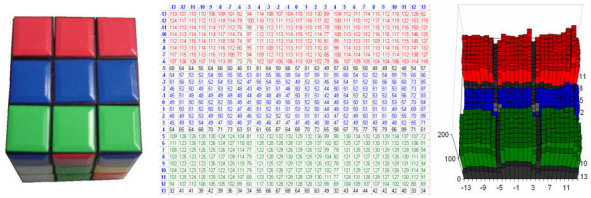


Figure 2: The neighbor pixel effect to the camera sensor.

2.1.3 Shadow Model

At this point, we still have a major inconvenience in the model. Shadows are not translated as being part of the background and we definitely do not want them to be considered as an object of interest. To remedy this, we have chosen to classify shadows as regions in the image that differ in Y but U,V rest unchanged. The shadow vector I_s is then given as follows:

$$I_S = (Y_S, U, V) \tag{4}$$

where $Y_S = Y \pm \Delta Y$. This is in fact not a disadvantage. Since the Y component is only sensible to illumination changes, it is in fact redundant for foreground or background object discrimination.

2.2 Learning Vector Model

We can now form the final vectors which we are going to observe at pixel level. They are given as follows:

$$\begin{aligned} B_1 &= (G_Y, U, V) \\ B_2 &= (Y_S) \\ B_3 &= (U, V) \end{aligned} \tag{5}$$

where B_1 is the object background model which combines the color and texture information but ignores luminance that is incorporated in B_2 which is the shadow background model and B_3 is taken as a safety

net vector which measures the chromaticity component of the background. The choice of the safety net vector is empirically chosen from the conducted experiments.

We decided to use the Mixture of Gaussian (MoG) method to maintain a density function at pixel level. This choice is made for each pixel after studying the behavior of the camera sensor. Figure 3 shows very clearly that the behavior of the camera sensor used in our experiment is Gaussian. However, to enable real-time computations, we assumed that the density function of each channel have only a single distribution which is obtained by considering only the highest peak from the graph.

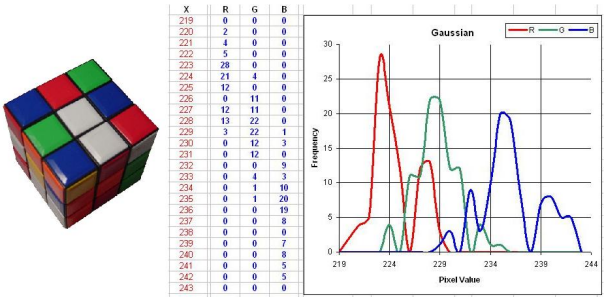


Figure 3: Camera sensor used in our experiments have a Gaussian behavior.

In the MoG method, at each pixel, the variation in the observed vectors B_1, B_2 and B_3 can be each modeled separately by a mixture of K Gaussian. The probability $P(B)$ then of belonging to the background is given by:

$$P(B_j) = \sum_{i=1}^{K} \omega_i \, f(\, B_j, \, \mu_i \, , C_i \,) \tag{6}$$

where $j = 1, 2, 3$, K denotes the number of Gaussian distributions used to model the density function, μ the mean value vector of the Gaussian, C the covariance matrix associated with this Gaussian and ω is the weight assigned to the Gaussian distribution. In our approach, B_2, B_3 and B_1 are each modeled separately as a one, two and three component density function. These density functions can be viewed as a curve, surface and ellipsoid respectively. But, since each density function of each component is assumed to only have a single peak, $K = \omega = 1 \; \forall \; j = 1, 2, 3$.

3 Background Subtraction Algorithm

In this section, we describe how the background model is constructed and used to discriminate foreground and background pixels. The background model that is constructed is stored in an image called a background reference image. During run-time, the reference image is subtracted from the current image to obtain a mask which will highlight all foreground objects. Once the mask is obtained, the background model can be updated. There a four major steps in the background subtraction algorithm. These are detailed in the following subsections.

3.1 Background Learning

In this stage, for each incoming frame, at pixel level, we store the number of samples n, the sum of the observed vector a, b, c grouped as U and the sum of the cross-product of the observed vector d, e and f grouped as V.

$$n = \sum_{i=1}^{N} 1$$

$$a = \sum_{i=1}^{N} B_1 \; , \; b = \sum_{i=1}^{N} B_2 \; , \; c = \sum_{i=1}^{N} B_3$$

$$d = \sum_{i=1}^{N} (B_1 \times B_1) \; , \; e = \sum_{i=1}^{N} (B_2 \times B_2)$$

$$f = \sum_{i=1}^{N} (B_3 \times B_3) \tag{7}$$

This stage will be defined as the learning phase and is required for initialization. From our experiments, about 100 frames is necessary to sufficiently learn the variations in the background. This corresponds to about 2 to 4 seconds of initialization using our camera.

3.2 Parameter Estimation

At the end of the learning phase, the required variables for the Gaussian models need to be calculated. They are given as follows:

$$\mu_1 = \frac{a}{n} \; , \; \mu_2 = \frac{b}{n} \; , \; \mu_3 = \frac{c}{n}$$

$$C_{B_1} = \frac{1}{n}(d) - \frac{1}{n^2}(a \times a^T)$$

$$C_{B_2} = \frac{1}{n}(e) - \frac{1}{n^2}(b \times b^T)$$

$$C_{B_3} = \frac{1}{n}(f) - \frac{1}{n^2}(c \times c^T) \tag{8}$$

These variables are calculated in a very efficient manner such that real-time compatibility is always maintained. In our implementation this stage is referred to the Gaussian distribution parameter estimation stage.

3.3 Foreground Detection

Foreground detection compares the input video frame with the background reference image and identifies candidate foreground pixels from the input frame. The most commonly used approach for foreground detection is to check whether the input pixel is significantly different from the corresponding background estimate. In the MoG method, we can do this by expanding the characterizing equation for each Gaussian distribution.

$$\begin{aligned} P(B_j) &= f(\, B_j,\, \mu_j\, , C_{B_j}\,) \\ &= (\, 2\pi^{\frac{n}{2}} \times \sqrt{|C_{B_j}|}\,)^{-1} \\ &\times \; exp\,\{\,(B_j - \mu_j)^T \\ &\times \; C_{B_j}^{-1} \times (B_j - \mu_j)\,\} \end{aligned} \tag{9}$$

where $j = 1, 2, 3$. To compute this probability, it is not entirely necessary to evaluate the whole expression. The first term is a constant and the remaining term is popularly known as the Mahalanobis Distance (MD). Hence, the decision making process is streamed down to the following three categories,

$$MD_j = (B_j - \mu_j)^T \times C_{B_j}^{-1} \times (B_j - \mu_j) \tag{10}$$

for $j = 1, 2, 3$. This expression evaluates the difference between the reference image and the current image using the three density functions. The difference is decomposed into there MD's. Applying suitable thresholds yields an object category which indicates the type of the pixel. As mentioned before, static thresholds are unsuitable for dynamic environments. In order to make the threshold adaptive, their values are derived form the magnitude of the covariance matrices of each Gaussian distribution which indicates the extent of variation in the observed values. We note here that we do not ignore inter-channel dependency in the computation of our covariance matrices and their inverses as some authors do. Our method classifies a given pixel into the following three categories,

$$\begin{aligned} OB \text{ if } &\begin{cases} MD_1 > \epsilon_1 |C_{B_1}| \\ MD_2 > \epsilon_2 |C_{B_2}| \\ MD_3 > \epsilon_3 |C_{B_3}| \end{cases} \\ SH \text{ if } &\begin{cases} MD_1 > \epsilon_1 |C_{B_1}| \\ MD_2 > \epsilon_2 |C_{B_2}| \\ MD_3 < \epsilon_3 |C_{B_3}| \end{cases} \\ BG \text{ if } &\begin{cases} MD_1 < \epsilon_1 |C_{B_1}| \\ MD_2 < \epsilon_2 |C_{B_2}| \\ MD_3 < \epsilon_3 |C_{B_3}| \end{cases} \end{aligned} \tag{11}$$

where ϵ_1, ϵ_2 and ϵ_3 are constants which depend on the camera sensor and the output from the camera driver. They are determined empirically during the experiments.

3.4 Model Update

In order to fully capture the changing dynamic environment, the background model has to be updated. In the proposed algorithm, we need to update the mean vector, the covariance matrix and its inverse. In order to avoid recalculating all the coefficients altogether, a recursive update is preferred. The will allow us to obtain the new values of the model parameters from the old ones. From the general expression for the mean vector and the correlated Gaussian distributions,

$$\begin{aligned} \mu_t &= \frac{U_t}{n_t} \\ C_t &= \frac{1}{n_{t-1}+1} \\ &\times \; \Big\{ \Big[V_{t-1} + (B_t \times B_t^T) \Big] \\ &- \; \Big[\frac{(U_{t-1} + B_t) \times (U_{t-1}^T + B_t^T)}{n_{t-1}+1} \Big] \Big\} \end{aligned} \tag{12}$$

we can then obtain a simplified expression for the updated mean vector and the updated inverse covariance matrix,

$$\begin{aligned} \mu_t &= \frac{\mu_{t-1} n_{t-1} + B_t}{n_{t-1}+1} \\ C_t^{-1} &= (C_{t-1} + B_t B_t^T)^{-1} \\ &= C_{t-1}^{-1} - \frac{C_{t-1}^{-1} B_t B_t^T C_{t-1}^{-1}}{1 + B_t^T C_{t-1}^{-1} B_t} \end{aligned} \tag{13}$$

The updating of covariances matrices and the mean values are only done to pixels which were assigned as background (BG). Once the covariances matrices are updated, the respective thresholds are also updated. The updated inverse covariance matrices are used in subsequent decision making process.

4 Data Validation

We define data validation as the process of improving the candidate foreground mask based on information obtained from outside the background model. Inaccuracies in threshold levels, signal noise and uncertainty in the background model can sometimes lead to pixels easily mistaken as true foreground objects and typically results in small false-positive or false-negative regions distributed randomly across the candidate mask. The most common approach is to combine morphological filtering and connected component grouping to eliminate these regions. Applying

morphological filtering on foreground masks eliminates isolated foreground pixels and merges nearby disconnected foreground regions. Opening and clos-

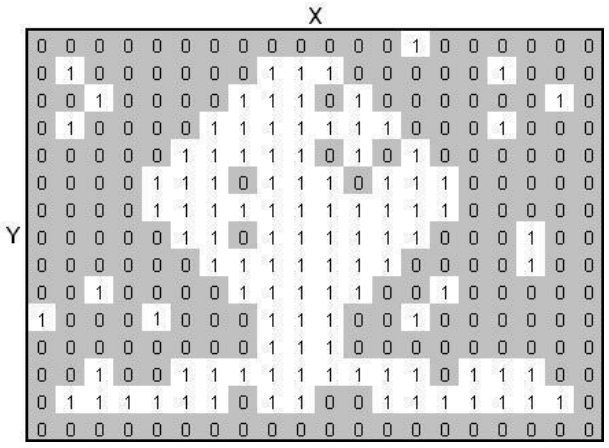


Figure 4: Image before applying morphological filter.

ing are two important operators that are both derived from the fundamental operations of erosion and dilation. These operators are normally applied to binary images. The basic effect of an opening is somewhat like erosion in that it tends to remove some of the foreground (bright) pixels from the edges of regions of foreground pixels. However, it is less destructive than erosion in general. Figure 5 shows the result af-

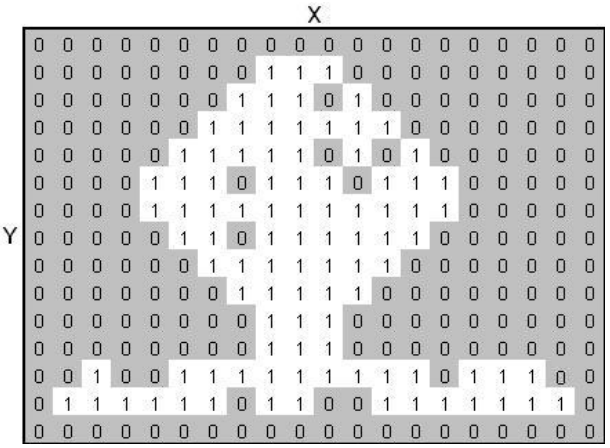


Figure 5: Image after applying the erosion filter.

ter applying the opening (erosion) filter to Figure 4 which eliminates the foreground pixels in the scene that is misjudged by the background subtraction algorithm or due to improper thresholding. We use a 5×5 mask to eliminate this foreground noise that is not part of the scene or the object of interest. Another added advantage of using this filter is to remove any unconnected pixel that does not belong to the object. The size of the mask is determined empirically from our experiments. Closing is similar in some ways to dilation in that it tends to enlarge the boundaries of foreground (bright) regions in an image (and shrink background color holes in such regions), but it is less destructive to the original boundary shape. Figure 6

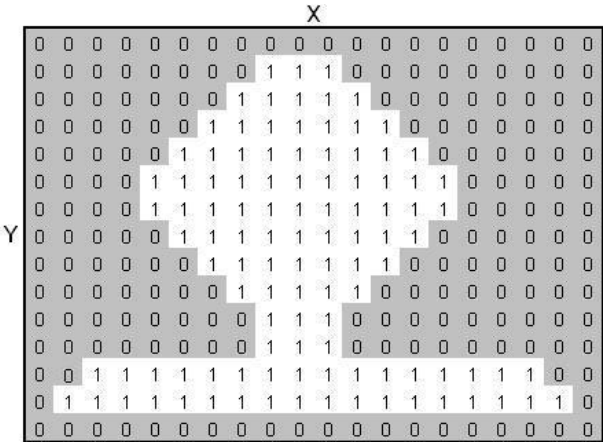


Figure 6: Image after applying erosion and dilation filter.

shows the result of applying an opening and closing filter to Figure 4. We apply a 5×5 closing mask after applying the opening filter. The effect of applying both operators is to preserve background regions that have a similar shape to the structuring element, or that can completely contain the structuring element, while eliminating all other regions of background pixels. The advantage of this is that the foreground pixels which are misjudged as background pixels make the object to look as if it is not connected. Opening and closing are themselves often used in combination to achieve more subtle results. It is very clear that applying morphological opening and closing filters has a positive effect on the process of extracting object from the scene by removing the noise from the subtracted foreground.

5 Face Localization

The target of this research is to use background subtraction technique to detect an object in the scene. The object of interest to us is the human face which is required for biometric identification. Every object or target has its own methods for uniquely recognizing themselves based upon one or more inherent physical or behavioral character. Our implementation is to detect the face of a human when the subject is standing in front of a camera and then by initiating a capture sequence. The hardware for this has been developed and attached to the system through an RS-232 communication port interface. The space between the camera and the subject should ideally be around 1m to 1.5m.

This would normally give us an image that shows the head of the subject and part of its shoulder. Figure 7 shows the analysis of an image (one frame) of size $M \times N$ after the background is subtracted and data validation completed. The only object that appears in the scene is the face and part of the shoulder. We then proceed to calculate the vertical (V) and horizontal (H) density of the image (binary image) and obtain the density graphs as shown in Figure 7. The height of

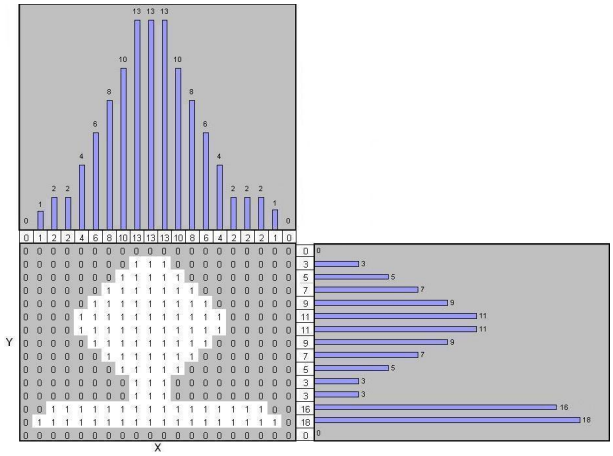


Figure 7: Face localization using vertical and horizontal densities.

the graph bars will be considered from the top of the image. With a suitable algorithm, we can compute the vertical and horizontal density of the mask as follows:

$$V(x) = \sum_{x=1}^{M} I(x,y) \ \forall \ x = \{1, N\}$$

$$H(y) = \sum_{y=1}^{N} I(x,y) \ \forall \ y = \{1, M\} \qquad (14)$$

From these values, we can then localize the center of the face and the face size. Next, we proceed to obtain a rectangular mask and apply it to our image. This will give us the first results for face detection. These results can be easily saved in a database or compared with an existing image in a database for biometric purposes or for personnel identification. After the rectangle mask is obtained, we can then refine the result to obtain an ellipse mask around the face based on the side lengths of the rectangle. A simple approach can be used to determine the center of the ellipse, the major axis and the minor axis.

6 Experimental Results

The background subtraction algorithm that is proposed in this paper was tested in various outdoor and indoor scenes in various environmental conditions. In our experiments, we used a Marlin FO33C 1/2" progressive scan color CCD firewire camera connected to a Pentium IV 2.8 GHz PC operating under Windows XP environment to capture and process the incoming frames in Visual C++ 6.0 programming language. Five examples scenes are presented in this paper. The first is a closed environment inside the office as shown in Figure 8 and the second is outside the main office in an outdoor environment as shown in Figure 9. The third test shown in Figure 10 focuses on background clutter while the fourth test shown in Figure 11 considers highlights. In Figure 12, the final test aims at shadow detection and removal. The first image (a) is the scene that the algorithm will learn. The second image (b) is the subject in front of the camera and when capture is initialized manually. The results after applying the background subtraction algorithm is shown in the third image (c). This is a binary image. Morphological filters cleans the obtained results as shown in the fourth image (d). The fifth image (e) is the face located in the scene and the sixth image (f) is the face ellipse. The obtained results are fairly good and shows that the proposed algorithm is accurate.

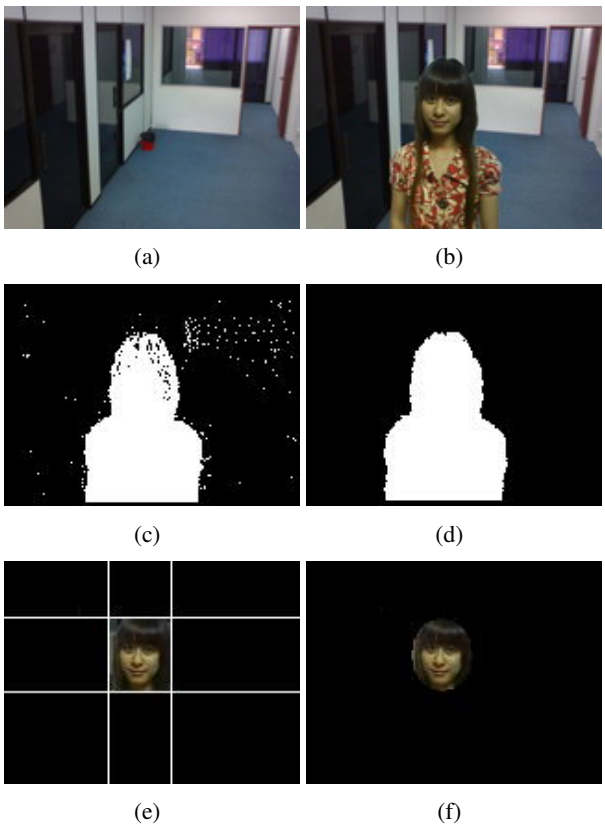
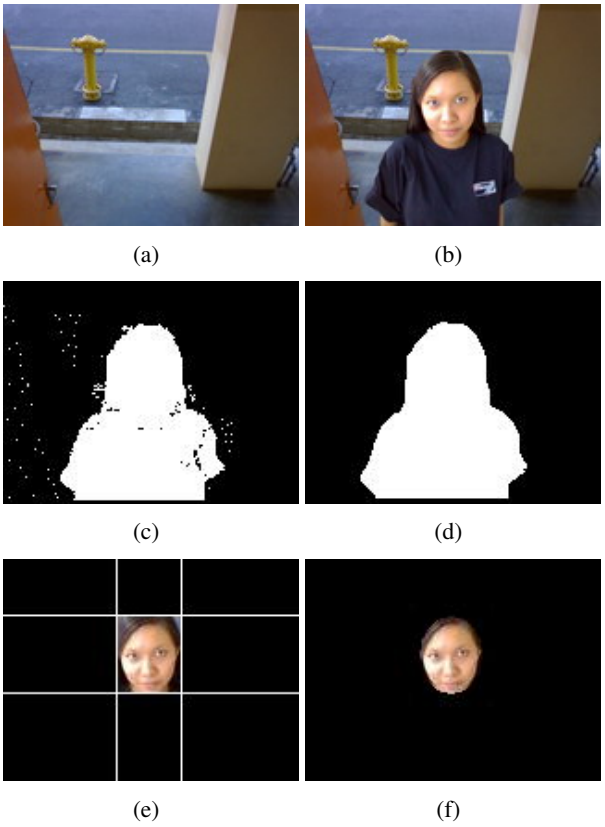


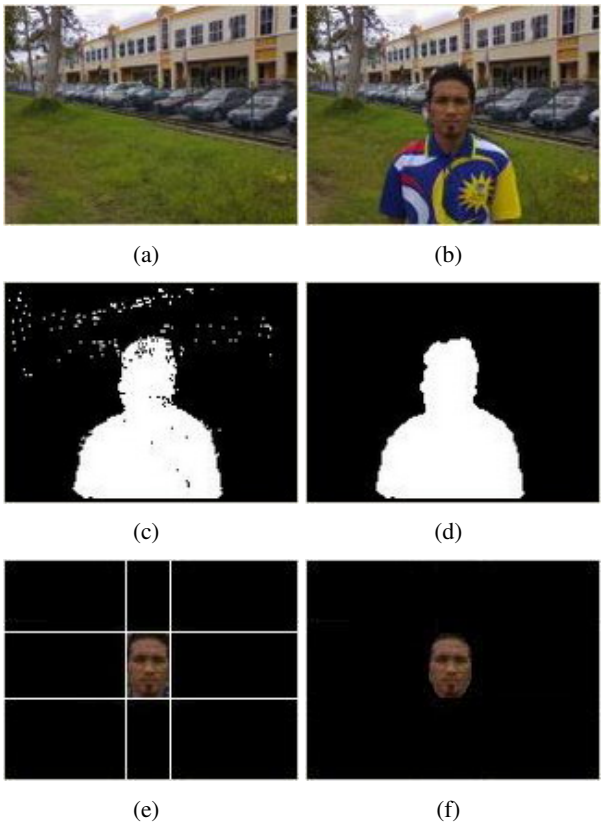
Figure 8: Experimental results of indoor scene test.

From the results, we find that the proposed algorithm is tolerant to illumination changes and back-



(a) (b) (c) (d) (e) (f)

Figure 9: Experimental results of outdoor scene test.



(a) (b) (c) (d) (e) (f)

Figure 10: Experimental results of background clutter scene test.

Table 1: Average error rate (%) of 25 frames for various experimented scenes.

Scene / Method	[20]	[21]	Ours
Illumination Change	13.4	7.2	6.5
Background Clutter	15.6	8.9	5.4
Shadows	23.4	8.1	5.9

Table 2: Average frame rate (Hz) of experimented methods.

Method	[20]	[21]	Ours
Frame rate	9.6	12.3	32.5

ground clutter. The indoor and outdoor scenes were repeated numerous times under different lighting conditions and background texture. We also experimented with various outdoor scenes with moving leaves, waving branches and moving clouds. In all these experiments, the proposed algorithm successfully executed in real-time ($\approx$ 32Hz), which makes it well suited for systems that require online monitoring. During these experiments, we also found that with a higher number of learning frames, accuracy can be improved. Figure 12 shows the shadow detection results of our proposed method. Shadows are detected and removed from the image, thus preventing undesired corruption to the final result which may lead to problems such as the detected object's shape being distorted or the object's shape being merged. This will introduce inaccuracies in the estimation of face location. In order to test the robustness of the proposed algorithm, we benchmarked our algorithm with that of [20] and [21]. In the benchmark test, 25 frames from the results of the background subtraction algorithm of various experiments of indoor and outdoor scenes were compared to obtain the recognition rate of correctly assigned background pixels. Subjects were told to move around in all experiments. The error rate is based on manually obtained predetermined ratio of the correct number of background pixels to the correct number of foreground pixels. The results of this benchmark is tabulated in Table 1 and 2.

7 Conclusion

This paper presents the implementation details for a fast background subtraction algorithm to detect an localize a human face from a dynamic background scene that contains shading and shadows using color im-

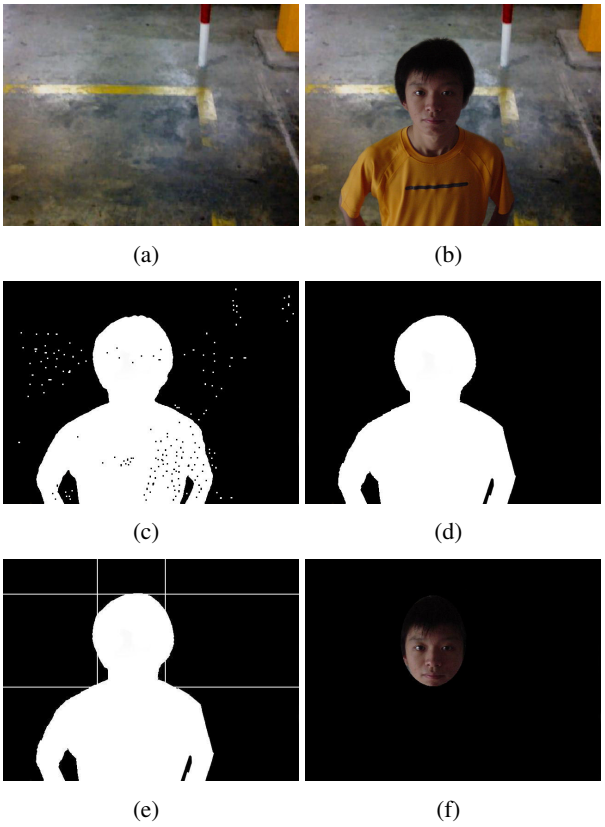


Figure 11: Experimental results of highlight scene test.

ages. The experimental results in real-time applications shows that the propose method is accurate, robust, reliable and computed efficiently. This method was designed under an assumption that the background scene although dynamic, can be effectively modeled by allowing the system to update adaptively during runtime. The background reference model which is obtained from the fast update contributes to the efficiency of this algorithm. We have implemented these routines as efficient as possible. Like all learning algorithms, caution must be observed because algorithms can be fed with information that might lead to *wrong* learning or too little information leading to *insufficient* learning. Several benchmarks concerning accuracy and speed have been presented. In this paper, we have implemented a system that uses an image size of about 640×480 pixels. Using this option, we have found that the system runs at about 30 to 35 frames per second which is generally sufficient for real-time applications.

References:

[1] N. Friedman and S. Russel, "Image segmentation in video sequences: A probabilistic approach," in *Proceedings of IEEE International Conference on Uncertainty in Artificial Intelligence*, San Francisco, USA, 1997, pp. 175–181.

[2] D. Hong and W. Woo, "A background subtraction for a vision-based user interface," in *Proceedings of IEEE International Conference on Information, Communications and Signal Processing*, vol. 1, 2003, pp. 263– 267.

[3] M. Harville, G. Gordon, and J. Woodfill, "Foreground segmentation using adaptive mixture models in color and depth," in *Proceedings of IEEE Workshop on Detection and Recognition of Events in Video*, Los Angeles, USA, 2001, pp. 3–11.

[4] R. Cucchiara, C. Grana, M. Piccardi, and A. Prati, "Detecting moving objects, ghosts and shadows in video streams," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 10, pp. 1337–1342, 2003.

[5] I. Haritaoglu, R. Cutler, D. Harwood, and L. Davis, "Backpack - Detection of people carrying objects using silhouettes," in *Proceedings of Computer Vision and Image Understanding*, vol. 81, no. 3, New York, USA, 2001, pp. 385–397.

[6] K. M. Cheung, T. Kanade, J. Bouguet, and M. Holler, "A real-time system for robust 3D voxel reconstruction of human motions," in *Proceedings of Computer Vision and Pattern Recognition*, vol. 2, 2000, pp. 714 – 720.

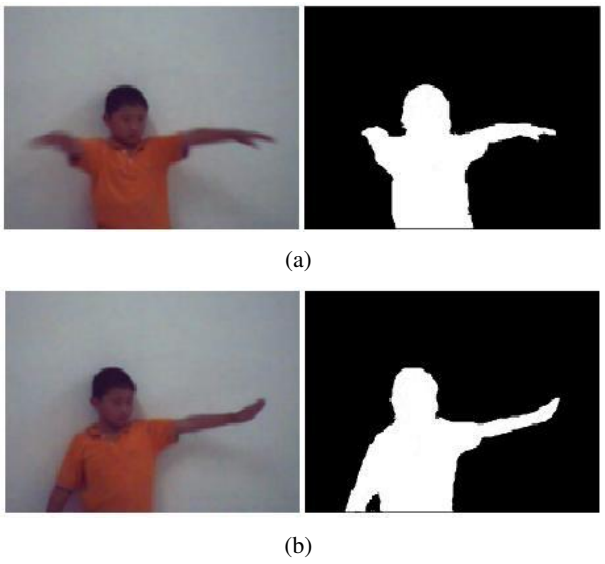


Figure 12: Experimental results of shadow detection.

[7] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers, "WallFlower - Principles and practice of background maintenance," in *Proceedings of International Conference on Computer Vision*, California, USA, 1999, pp. 255–261.

[8] C. Ianasi, V. Gui, F. Alexa, and C. I. Toma, "Noncausal adaptive mode tracking estimation for background subtraction in video surveillance," *WSEAS Transactions on Signal Processing*, vol. 2, no. 1, pp. 52–59, 2006.

[9] N. J. B. McFarlane and C. P. Schofield, "Segmentation and tracking of piglets in images," *Machine Vision and Applications*, vol. 8, no. 3, pp. 187–193, 1995.

[10] S. C. S. Cheung and C. Kamath, "Robust background subtraction with foreground validation for urban traffic video," *EURASIP Journal of Applied Signal Processing*, vol. 2005, no. 1, pp. 2330–2340, 2005.

[11] Y. Chen and T. S. Huang, "Hierarchical MRF model for model-based multi-object tracking," in *Proceedings of IEEE International Conference on Image Processing*, vol. 1, Thessaloniki, Greece, 2001, pp. 385–388.

[12] C. R. Wren, A. Azarbayejani, T. Darrell, and A. Pentland, "Pfinder: Real-time tracking of the human body," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 780–785, 1997.

[13] J. Heikkila and O. Silven, "A real-time system for monitoring of cyclists and pedestrians," in *Proceedings of IEEE Workshop on Visual Surveillance*, vol. 22, no. 7, Washington, USA, 2004, pp. 563–570.

[14] K. Kim, T. H. Chalidabhongse, D. Harwood, and L. Davis, "Background modeling and subtraction by codebook construction," in *Proceedings of International Conference on Image Processing*, vol. 5, 2004, pp. 3061– 3064.

[15] D. S. Lee, J. J. Hull, and B. Erol, "A bayesian framework for gaussian mixture background modeling," in *Proceedings of International Conference on Image Processing*, 2003, pp. 973–976.

[16] D. Koller, J. Weber, and T. Huang, "Towards robust automatic traffic scene analysis in real-time," in *IEEE International Conference on Decision and Control*, 1994, pp. 3776–3782.

[17] P. Roth, H. Bischof, D. Skovcaj, and A. Leonardis, "Object detection with bootstrapped learning," in *Proceedings 10th Computer Vision Winter Workshop*, 2005, pp. 33–42.

[18] C. Stauffer and W. E. L. Grimson, "Adaptive background mixture models for real-time tracking," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, 1999, pp. 246–252.

[19] N. Li, D. Xu, and B. Li, "A novel background updating algorithm based on the logical relationship," in *WSEAS International Conference on Signal, Speech and Image Processing*, 2007, pp. 154–158.

[20] O. Javed, K. Shafique, and M. Shah, "A hierarchical approach to robust background subtraction using color and gradient information," in *Proceedings of IEEE Workshop on Motion and Video Computing*, Washington, USA, 2002, pp. 22–27.

[21] T. Horprasert, D. Harwood, and L. Davis, "A statistical approach for real-time robust background subtraction and shadow detection," in *International Conference on Computer Vision*, Kerkyra, Greece, 1999, pp. 1–19.

[22] P. L. Rosin and T. Elis, "Image difference threshold strategies and shadow detection," in *Proceedings of the British Machine Vision Conference*, 1995.

[23] T. M. Su and J. S. Hu, "Robust background subtraction with shadow and highlight removal for indoor surveillance," *EURASIP Journal on Advances in Signal Processing*, vol. 2007, p. 14, 2007.

[24] A. Elgammal, D. Harwood, and L. Davis, "Non-parametric model for background subtraction," in *Proceedings of the European Conference on Computer Vision*, 2000.

[25] A. Elgammal, R. Duraiswami, D. Harwood, and L. Davis, "Background and foreground modeling using non-parametric kernel density estimation for video surveillance," *Proceedings of the IEEE*, vol. 90, no. 7, pp. 1151–1163, 2002.

[26] K. Appiah, A. Hunter, and T. Kluge, "GW4: a real-time background subtraction and maintenance algorithm for FPGA implementation," *WSEAS Transactions on Systems*, vol. 4, no. 10, pp. 1741–1751, 2005.