A statistical approach for real-time robust background subtraction and shadow detection

Article · January 1999

CITATIONS
854

READS
2,032

3 authors, including:

Thanarat H. Chalidabhongse
Chulalongkorn University
55 PUBLICATIONS **2,966** CITATIONS
SEE PROFILE

Larry S. Davis
University of Maryland, College Park
483 PUBLICATIONS **29,643** CITATIONS
SEE PROFILE

A Statistical Approach for Real-time Robust Background Subtraction and Shadow Detection

Thanarat Horprasert *thanarat@cs.umd.edu*
David Harwood *harwood@umiacs.umd.edu*
Larry S. Davis *lsd@umiacs.umd.edu*

Computer Vision Laboratory
University of Maryland
College Park, MD 20742

Abstract

This paper presents a novel algorithm for detecting moving objects from a static background scene that contains shading and shadows using color images. We develop a robust and efficiently computed background subtraction algorithm that is able to cope with local illumination changes, such as shadows and highlights, as well as global illumination changes. The algorithm is based on a proposed computational color model which separates the brightness from the chromaticity component. We have applied this method to real image sequences of both indoor and outdoor scenes. The results, which demonstrate the system's performance, and some speed up techniques we employed in our implementation are also shown.

1 Introduction

The capability of extracting moving objects from a video sequence is a fundamental and crucial problem of many vision systems that include video surveillance [1, 2], traffic monitoring [3], human detection and tracking for video teleconferencing or human-machine interface [4, 5, 6], video editing, among other applications. Typically, the usual approach for discriminating moving object from the background scene is background subtraction. The idea of background subtraction is to subtract the current image from a reference image, which is acquired from a static background during a period of time. The subtraction leaves only non-stationary or new objects, which include the objects' entire silhouette region. The technique has been used for years in many vision systems as a preprocessing step for object detection and tracking, for examples, [1, 4, 5, 7, 8, 9]. The results of the existing algorithms are fairly good; in addition, many of them run in real-time. However, many of these algorithms are susceptible to both global and local illumination changes such as shadows and highlights. These cause the consequent processes, e.g. tracking, recognition, etc., to fail. The accuracy and efficiency of the detection are clearly very crucial to those tasks. This problem is the underlying motivation of our work. We want to develop a robust and efficiently computed background subtraction algorithm that is able to cope with the local illumination change problems, such as shadows and highlights, as well as the global illumination changes. Being able to detect shadows is also very useful to many applications especially in "Shape from Shadow" problems [10, 11, 12, 13]. Our method must also address requirements of sensitivity, reliability, robustness, and speed of detection.

In this paper, we present a novel algorithm for detecting moving objects from a static background scene that contains shading and shadows using color images. We begin by introducing a new computational color model (*brightness distortion* and *chromaticity distortion*) that helps us to distinguish shading background from the ordinary background or moving foreground objects. Next, we propose an algorithm for pixel classification and threshold selection. Although we restrict our attention to indoor environments with static background, our algorithm works fairly well on real image sequences of outdoor scenes as shown in Section 6.

2 Computational Color Model

One of the fundamental abilities of human vision is *color constancy* [14]. Humans tend to assign a constant color to an object even under changing illumination over time or space. The perceived color of a point in a scene depends on many factors including physical properties of the point on the surface of the object. Important physical properties of the surface in color vision are surface spectral reflectance properties, which are invariant to changes of illumination, scene composition

or geometry. On *Lambertain*, or perfectly matte surfaces, the perceived color is the product of *illumination* and *surface spectral reflectance*.

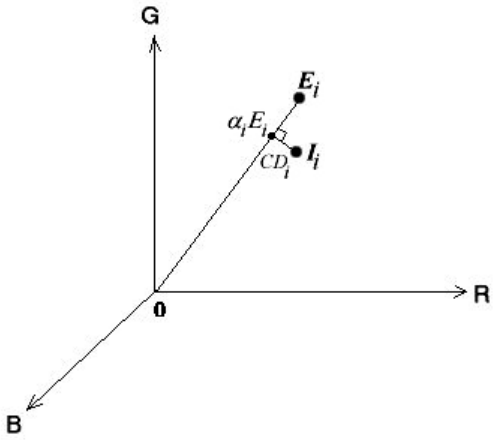

Figure 1: Our proposed color model in the three-dimensional RGB color space; the background image is statistically pixel-wise modeled. E_i represents an expected color of a given i^{th} pixel and I_i represents the color value of the pixel in a current image. The difference between I_i and E_i is decomposed into brightness (α_i) and chromaticity (CD_i) components.

This led to our idea of designing a color model that separates these two terms; in other words, that separates the *brightness* from the *chromaticity* component. Figure 1 illustrates the proposed color model in three-dimensional RGB color space. Consider a pixel, i, in the image; let $E_i = [E_R(i), E_G(i), E_B(i)]$ represent the pixel's *expected RGB color* in the reference or background image. The line OE_i passing through the origin and the point E_i is called *expected chromaticity line*. Next, let $I_i = [I_R(i), I_G(i), I_B(i)]$ denote the pixel's RGB color value in a current image that we want to subtract from the background. Basically, we want to measure the distortion of I_i from E_i. We do this by decomposing the distortion measurement into two components, *brightness distortion* and *chromaticity distortion*, defined below.

2.1 Brightness Distortion (α)

The brightness distortion (α) is a scalar value that brings the observed color close to the expected chromaticity line. It is obtained by minimizing

$$\phi(\alpha_i) = (I_i - \alpha_i E_i)^2 \tag{1}$$

α_i represents the pixel's strength of brightness with respect to the expected value. α_i is 1 if the brightness of the given pixel in the current image is the same as in the reference image. α_i is less than 1 if it is darker, and greater than 1 if it becomes brighter than the expected brightness.

2.2 Color Distortion (CD)

Color distortion is defined as the orthogonal distance between the observed color and the expected chromaticity line. The color distortion of a pixel i is given by

$$CD_i = \|I_i - \alpha_i E_i\| \tag{2}$$

3 Color Image Characteristics

Before discussing our algorithm for detecting moving objects from a static background scene based on the proposed computational color model, we need to understand some physical characteristics of real color images which are influenced by typical CCD cameras. The CCD sensors linearly transforms infinite-dimensional spectral color space to a three-dimensional RGB color space via red, green, and blue color filters; $\Pi : \varepsilon \rightarrow R^3$ defined as $\Pi(C) = [R, G, B]$. There are some characteristics of the output images, that we should account for in designing the algorithm, as follows

Color variation: In reality, we rarely observe the same RGB color value for a given pixel over a period of time due to camera noise and illumination fluctuation by light sources. Figure 2 demonstrates color variation over 100 frames of a static scene (Figure 2a). The gray-scaled value of each pixel on Figure 2b, 2c, 2d is the scaled standard deviation number of that pixel computed over 100 frames on red (Figure 2b), green (Figure 2c), and blue (Figure 2d) separately.

Band unbalancing: Figure 2 also demonstrate unequal variation among color bands (the average standard deviation of R, G, B of the scene in Figure 2a is 1.860, 1.857, and 1.971 respectively). Cameras typically have different sensitivities to different colors. Thus, in order to make the balance weights on the three color bands (R,G,B), the pixel values need to be rescaled or normalized by weight values. In this work, we normalized the pixel color by its standard deviation (s_i) which is given by

$$s_i = [\sigma_R(i), \sigma_G(i), \sigma_B(i)] \tag{3}$$

where $\sigma_R(i)$, $\sigma_G(i)$, and $\sigma_B(i)$ are the standard deviation of the i^{th} pixel's red, green, blue values computed over N frame of the background frames.

Clipping: Since the sensors have limited dynamic range of responsiveness, this restricts the varieties of color into a RGB color cube which is formed by red, green, and blue primary colors

as orthogonal axes. On 24-bit images, the gamut of color distribution resides within the cube range from [0, 0, 0] to [255, 255, 255]. Color outside the cube (negative or greater than 255 color) can not be represented. As the result, the pixel value is clipped in order to lie entirely inside the cube. This causes an unusual shape of the color distribution. The distribution of the saturated pixels (pixels that have negative or greater than 255 value on any band) is very peaked, with the variance being or almost zero (See Figure 2; the dark areas in the scene tend to have very small standard deviations). We should take this fact into account and set a default minimal value for s_i if a pixel's s_i is zero.

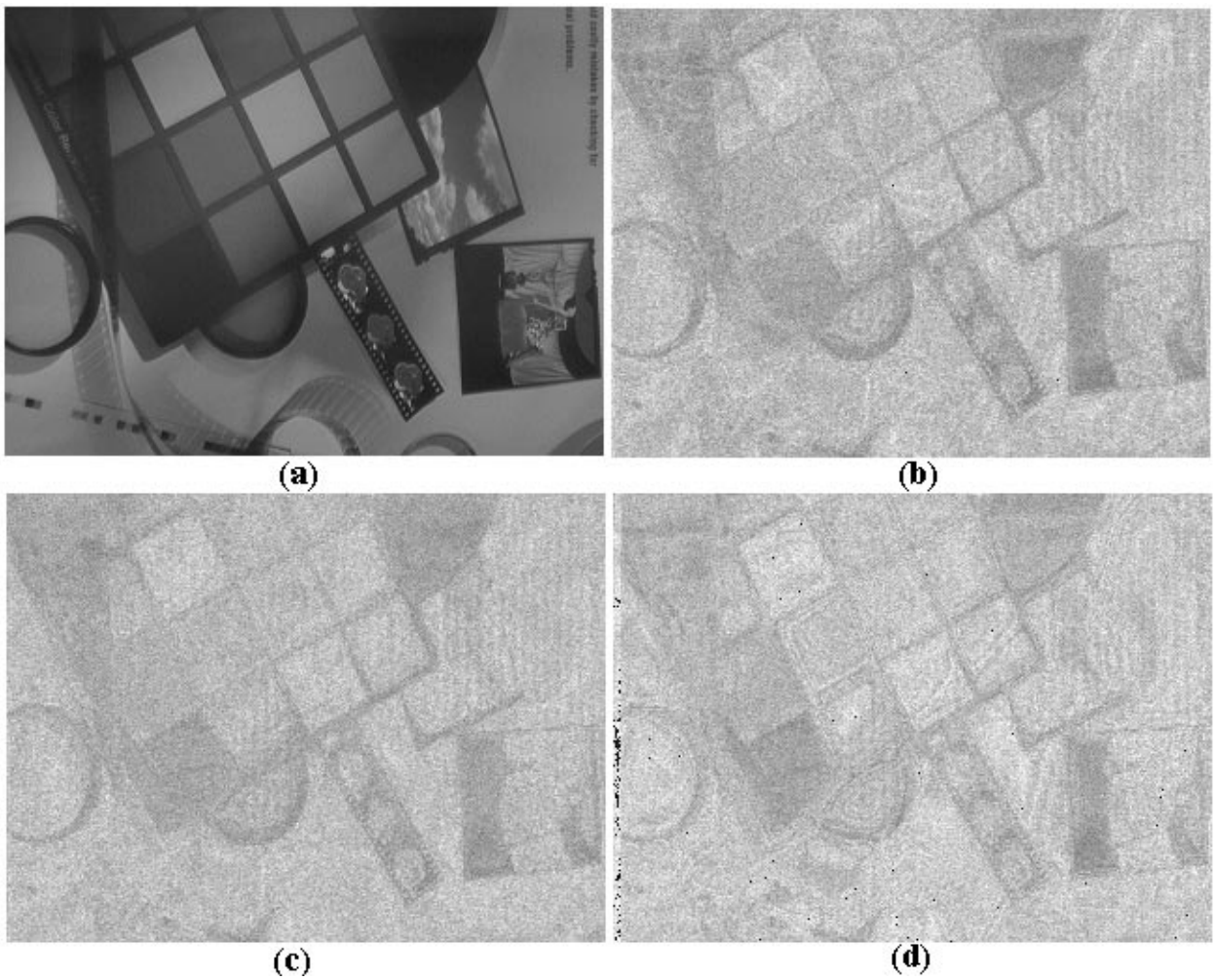

Figure 2: An illustration shows variation of color values on each color band over 100 frames of the scene shown in (a). (b), (c), and (d) depict standard deviations, s, (scaled by 100) images of red, green, and blue. The average standard deviations of red, green, and blue color bands are 1.860, 1.857, and 1.971 respectively.

4 Background Subtraction

The basic scheme of background subtraction is to subtract the image from a reference image that models the background scene. Typically, the basic steps of the algorithm are as follows:

- *Background modeling* constructs a reference image representing the background.
- *Threshold selection* determines appropriate threshold values used in the subtraction operation to obtain a desired detection rate.
- *Subtraction operation or pixel classification* classifies the type of a given pixel, i.e., the pixel is the part of background (including ordinary background and shaded background), or it is a moving object.

4.1 Background Modeling

In the background training process, the reference background image and some parameters associated with normalization are computed over a number of static background frames. The background is modeled statistically on a pixel by pixel basis. A pixel is modeled by a 4-tuple $< E_i, s_i, a_i, b_i >$ where E_i is the expected color value, s_i is the standard deviation of color value which is defined in Equation 3, a_i is the variation of the brightness distortion, and b_i is the variation of the chromaticity distortion of the i^{th} pixel. E_i, a_i and b_i are defined explicitly later in this section.

The expected color value of pixel i is given by

$$E_i = [\mu_R(i), \mu_G(i), \mu_B(i)] \tag{4}$$

where $\mu_R(i)$, $\mu_G(i)$, and $\mu_B(i)$ are the arithmetic means of the i^{th} pixel's red, green, blue values computed over N background frames.

So far, we have defined E_i and s_i. We also discussed about balancing color bands by rescaling the color values by the pixel variation factors (s_i). Thus the brightness distortion in Equation 1 and the chromaticity distortion in Equation 2 become

$$\begin{aligned} \alpha_i &= \min \left[\left(\frac{I_R(i) - \alpha_i \mu_R(i)}{\sigma_R(i)} \right)^2 + \left(\frac{I_G(i) - \alpha_i \mu_G(i)}{\sigma_G(i)} \right)^2 + \left(\frac{I_B(i) - \alpha_i \mu_B(i)}{\sigma_B(i)} \right)^2 \right] \\ &= \frac{\left(\frac{I_R(i)\mu_R(i)}{\sigma_R^2(i)} + \frac{I_G(i)\mu_G(i)}{\sigma_G^2(i)} + \frac{I_B(i)\mu_B(i)}{\sigma_B^2(i)} \right)}{\left(\left[\frac{\mu_R(i)}{\sigma_R(i)} \right]^2 + \left[\frac{\mu_G(i)}{\sigma_G(i)} \right]^2 + \left[\frac{\mu_B(i)}{\sigma_B(i)} \right]^2 \right)} \end{aligned} \tag{5}$$

$$CD_i = \sqrt{\left(\frac{I_R(i) - \alpha_i \mu_R(i)}{\sigma_R(i)} \right)^2 \left(\frac{I_G(i) - \alpha_i \mu_G(i)}{\sigma_G(i)} \right)^2 \left(\frac{I_B(i) - \alpha_i \mu_B(i)}{\sigma_B(i)} \right)^2} \tag{6}$$

Next, we consider the variation of the brightness and chromaticity distortions over space and time of the training background images. We found that different pixels yield different distributions of α and CD, shown in Figure 3a, b. These variations are embedded in the background model as a_i and b_i in the 4-tuple background model for each pixel, and are used as normalization factors. For data analysis, we want to compare the variations between brightness and chromaticity components; hence we compute the distance between αE_i and E_i and its RMS which is shown in Figure 3c. The figure shows the brightness variation over the image and have the same unit as the chromaticity variation b_i which is shown in Figure 3b, so they are comparable. We found the variation in brightness (distant between αE_i and E_i) is much greater than the variation in chromaticity (distant between αE_i and I_i). This confirms that our computational color model is similar to human vision which is more sensitive to changes in luminosity than to changes in color.

a_i represents the variation of the brightness distortion of i^{th} pixel, which is given by

$$a_i = RMS(\alpha_i) = \sqrt{\frac{\sum_{i=0}^{N}(\alpha_i - 1)^2}{N}} \tag{7}$$

b_i represents the variation of the chromaticity distortion of the i^{th} pixel, which is given by

$$b_i = RMS(CD_i) = \sqrt{\frac{\sum_{i=0}^{N}(CD_i)^2}{N}} \tag{8}$$

4.2 Pixel Classification or Subtraction Operation

In this step, the difference between the background image and the current image is evaluated. The difference is decomposed into brightness and chromaticity components. Applying the suitable thresholds on the brightness distortion (α) and the chromaticity distortion (CD) of a pixel i yields an object mask $M(i)$ which indicates the type of the pixel. Our method classifies a given pixel into four categories. A pixel in the current image is

- *Original background (B)* if it has both brightness and chromaticity similar to those of the same pixel in the background image.
- *Shaded background or shadow (S)* if it has similar chromaticity but lower brightness than those of the same pixel in the background image. This is based on the notion of the shadow as a semi-transparent region in the image, which retains a representation of the underlying surface pattern, texture or color value [15].
- *Highlighted background (H)*, if it has similar chromaticity but higher brightness than the background image.
- *Moving foreground object (F)* if the pixel has chromaticity different from the expected values in the background image.

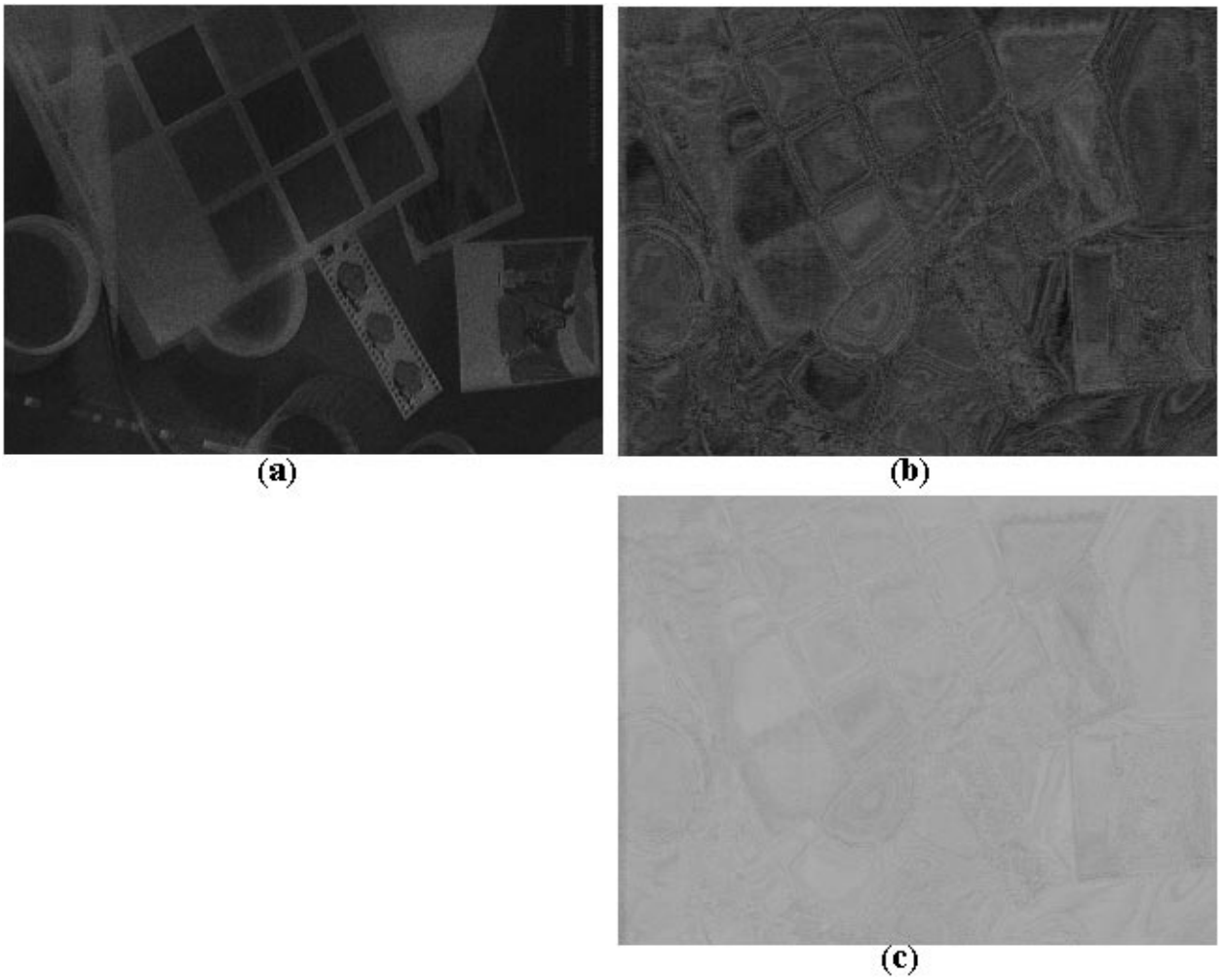

Figure 3: An illustration demonstrates the variations of brightness distortion and chromaticity distortion of different pixel colors over 100 images of the static scene. (a) is an image of a_i scaled by 2000 and (b) is an image of b_i scaled by 100, (c) is an image of RMS of distance between αE_i and E_i scaled by 100.

As mentioned above the different pixels yield different distributions of α_i and CD_i. In order to use a single threshold for all of the pixels, we need to rescale the α_i and CD_i. Let

$$\widehat{\alpha_i} = \frac{\alpha_i - 1}{a_i} \tag{9}$$

$$\widehat{CD_i} = \frac{CD_i}{b_i} \tag{10}$$

be *normalized brightness distortion* and *normalized chromaticity distortion* respectively.

Based on these definitions, a pixel is classified into one of the four categories B, S, H, F by the following decision procedure.

$$M(i) = \begin{cases} F & : \quad \widehat{CD_i} > \tau_{CD}, \quad else \\ B & : \quad \widehat{\alpha_i} < \tau_{\alpha 1} \quad and \quad \widehat{\alpha_i} > \tau_{\alpha 2}, \quad else \\ S & : \quad \widehat{\alpha_i} < 0, \quad else \\ H & : \quad otherwise \end{cases} \tag{11}$$

where τ_{CD}, $\tau_{\alpha 1}$, and $\tau_{\alpha 2}$ are selected threshold values used to determine the similarities of the chromaticity and brightness between the background image and the current observed image. In next subsection, we will discuss the method to select suitable threshold values.

However, there might be a case where a pixel from a moving object in current image contains very low RGB values. This dark pixel will always be misclassified as a shadow. Because the color point of the dark pixel is close to the origin in RGB space and the fact that all chromaticity lines in RGB space meet at the origin, thus the color point is considered to be close or similar to any chromaticity line. To avoid this problem, we introduce a lower bound for the normalized brightness distortion ($\tau_{\alpha lo}$). Then, the decision procedure Equation 11 becomes

$$M(i) = \begin{cases} F & : \quad \widehat{CD_i} > \tau_{CD} \quad or \quad \widehat{\alpha_i} < \tau_{\alpha lo}, \quad else \\ B & : \quad \widehat{\alpha_i} < \tau_{\alpha 1} \quad and \quad \widehat{\alpha_i} > \tau_{\alpha 2}, \quad else \\ S & : \quad \widehat{\alpha_i} < 0, \quad else \\ H & : \quad otherwise \end{cases} \tag{12}$$

4.3 Automatic Threshold Selection

Typically, if the distortion distribution is assumed to be a Gaussian distribution, then to achieve a desired detection rate,r, we can threshold the distortion by $K\sigma$ where K is a constant determined by r and σ is the standard deviation of the distribution. However, we found from experiments that the distribution of $\widehat{\alpha_i}$ and $\widehat{CD_i}$ are not Gaussian (see Figure 4). Thus, our method determines the appropriate thresholds by a statistical learning procedure. First, a histogram of the normalized brightness distortion, $\widehat{\alpha_i}$, and a histogram of the normalized chromaticity distortion, $\widehat{CD_i}$, are constructed as shown in Figure 4. The histograms are built from combined data through a long sequence captured during background learning period. The total sample would be NXY values

for a histogram. (The image is $X \times Y$ and the number of trained background frames is N.) After constructing the histogram, the thresholds are now automatically selected according to the desired detection rate r. A threshold for chromaticity distortion, τ_{CD}, is the normalized chromaticity distortion value at the detection rate of r. In brightness distortion, two thresholds ($\tau_{\alpha 1}$ and $\tau_{\alpha 2}$) are needed to define the brightness range. $\tau_{\alpha 1}$ is the $\widehat{\alpha_i}$ value at that detection rate, and $\tau_{\alpha 2}$ is the $\widehat{\alpha_i}$ value at the $(1-r)$ detection rate.

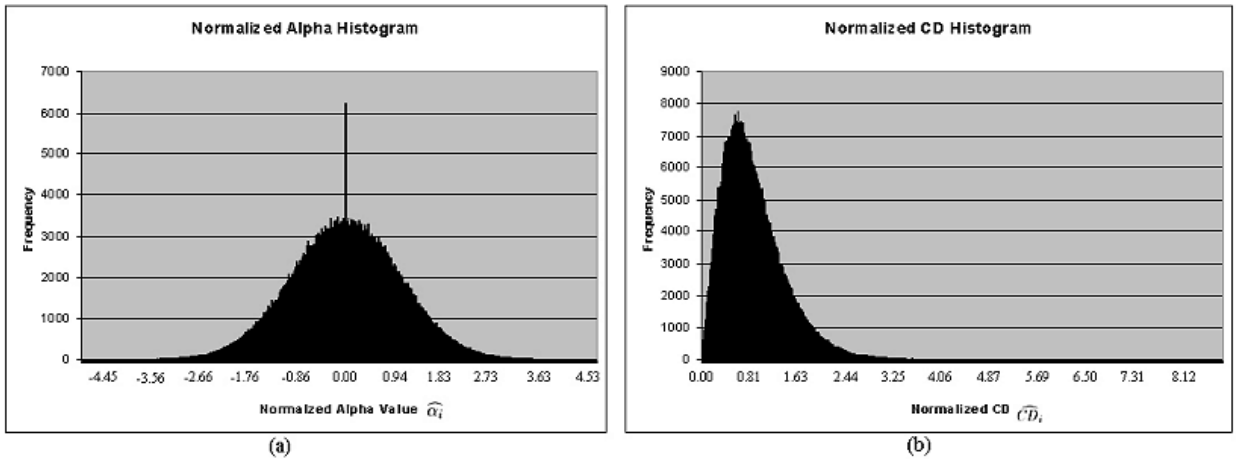

Figure 4: **(a)** is the normalized brightness distortion ($\widehat{\alpha_i}$) histogram, and **(b)** is the normalized chromaticity distortion ($\widehat{CD_i}$) histogram.

5 Clustering Detection Elimination

In our experiments and analysis, we found that the false detection tends to be clustering in some spots as shown in Figure 5a,c. We tested the detection performance by subtracting 100 background frames from the background models using the methods explained above. The error rate $(1-r)$ was set at 0.01%. Yellow pixel shows that false positive detection occurs once on that pixel over the 100 testing images. Red pixel depicts the false positive detection occurs more than once on that particular pixel. From observation, we found that those pixels are those that have very small variation in chromaticity distortion, in other words, very small b_i. When such small b_i are used in normalization, it makes $\widehat{CD_i}$ too big, and likely to exceed the threshold. We, hence, propose a method to limit the value of b_i to be a certain number called *default minimum* b_i. To systematically obtain the default minimum b_i, an optimization process is performed. The process starts by assigning a default minimum b_i value to all b_i that are smaller than the default; then performing the detection on all frames. Next, we compare the error rate of those pixels that have b_i bigger than the default value against the error rates of those pixels that have the default b_i value. A search is performed to find the default minimum b_i that yields a balanced error rates (the error

rates of both groups of pixels are equalized). Figure 6 shows plots of error rates versus the default minimum b_i value of the sequence shown in Figure 5a. The default minimum b_i obtained from the optimization is 0.55. Figures 5b,d show the result of false detections over the same sequences after applying the computed default minimum b_i values. The false detection spatial distributions are sparser than the originals in both sequences.

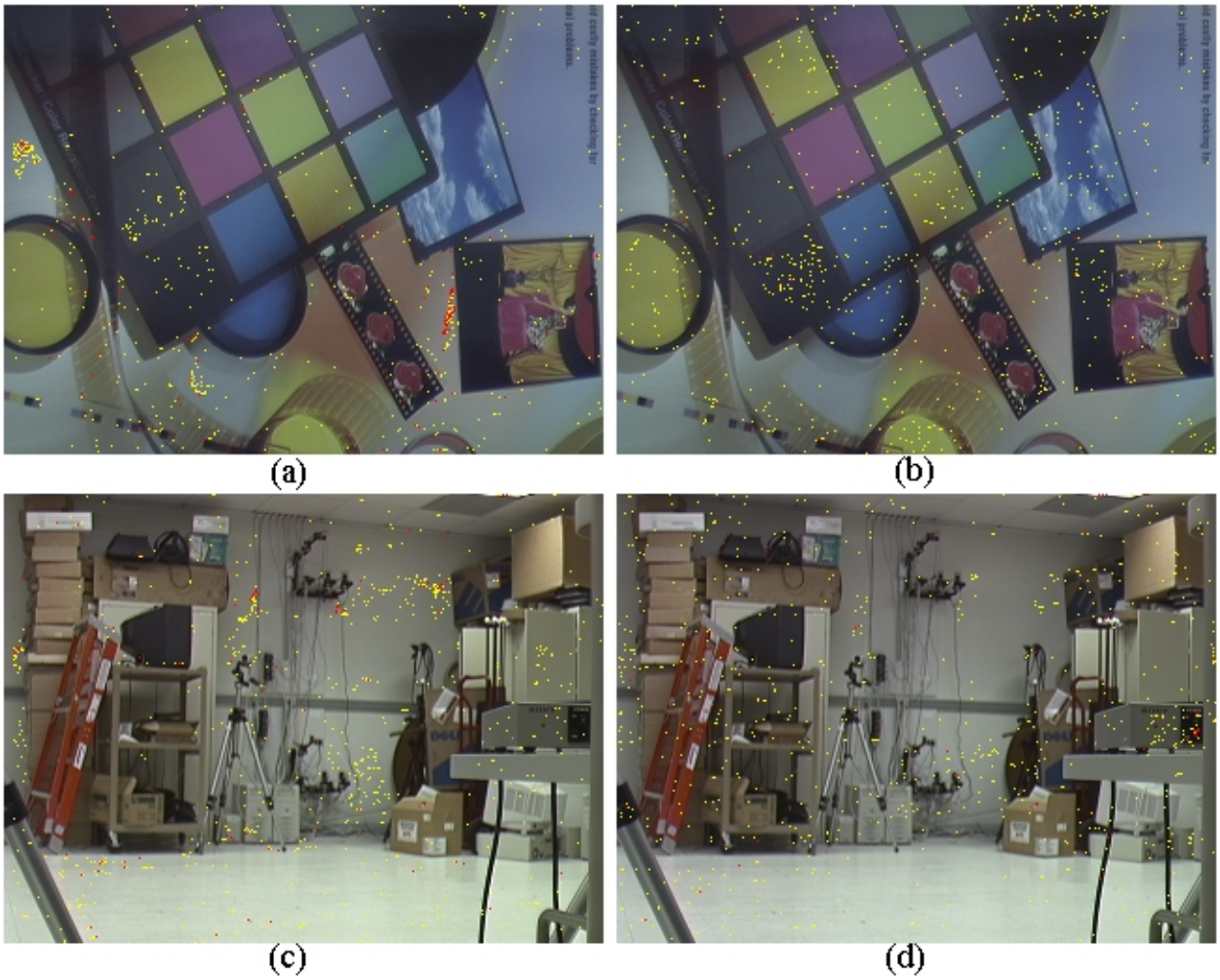

Figure 5: **(a)** and **(c)** demonstrate the clustering of the false positive detection over 100 frames of color picture sequence and indoor scene sequence. Yellow pixels represent that the false positive occurs once on that pixel over the 100 testing images; red pixels depicts the false positive occurs more than once on that pixel. **(b)** and **(d)** show the result of the proposed optimization method to de-clustering the false positive detections by limiting the minimum b_i value.

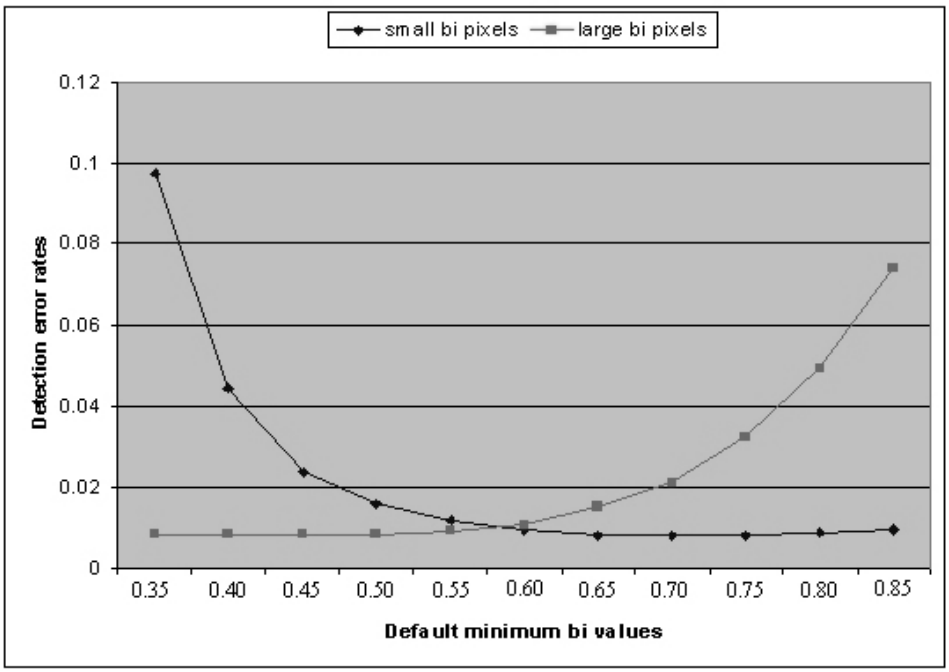

Figure 6: An illustration shows the error rates of those pixels which have small b_i and those pixels that have larger b_i. The crossing point which depicts the desired default minimum b_i value that balances the error rates between those two groups of pixels.

6 Result

This section demonstrates the performance of the proposed algorithm on several image sequences of both indoor and outdoor scenes. Sequences shown here are 320x240 images. The detection rate, r, was set at 0.9999, and the lower bound of the normalized brightness distortion ($\tau_{\alpha lo}$) is set at 0.4. Figure 7 shows the result of applying the algorithm to several frames of an indoor scene containing a person walking around the room. As the person moves, he both obscures the background and casts shadows on the floor and wall. Red pixels depict the shadow, and we can easily see how the shape of the shadow changes as the person moves. Although it is difficult to see, there are green pixels which depict the highlighted background pixels, appearing along the edge of the person's sweater. Figure 8 shows a sequence of an outdoor scene containing a person walking across a street. Although there are small motions of background objects, such as the small motions of leaves and water surface, the result shows the robustness and reliability of the algorithm. Figure 9 illustrates our algorithm being able to cope with the problem of global illumination change. It shows another indoor sequence of a person moving in a room; at the middle of the sequence, the global illumination is changed by turning half of the fluorescence lamps off. The system is still able to detect the target successfully.

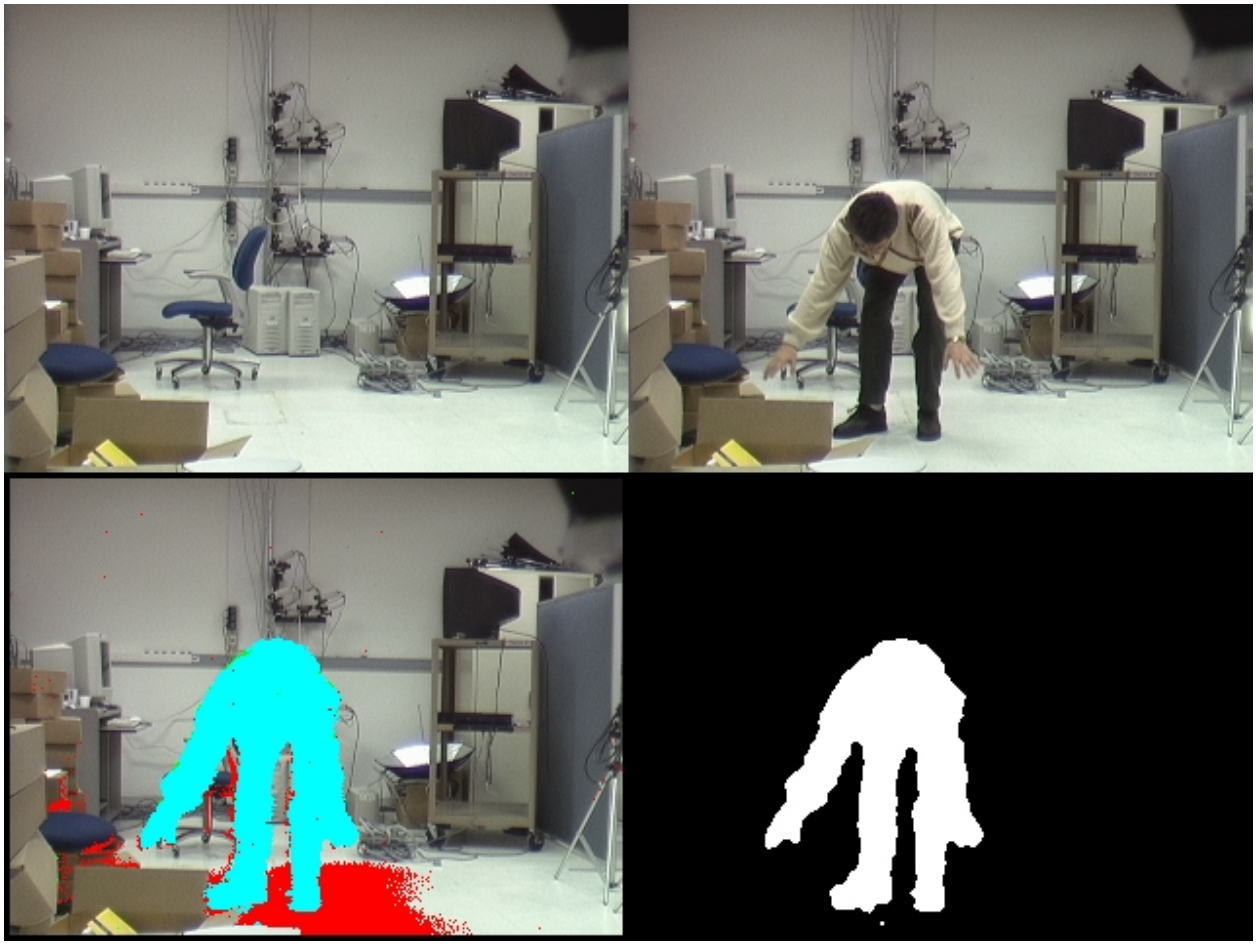

Figure 7: An example shows the result of our algorithm applying on a sequence of a person moving in an indoor scene. The upper left image is the background scene, the upper right image is the input sequence, and the lower left image shows the output from our background subtraction (the foreground pixels are overlaid by blue, the shadows are overlaid by red, the highlights are overlaid by green, and the ordinary background pixels are kept as the original color.) The lower right image shows only foreground region after noise cleaning is performed.

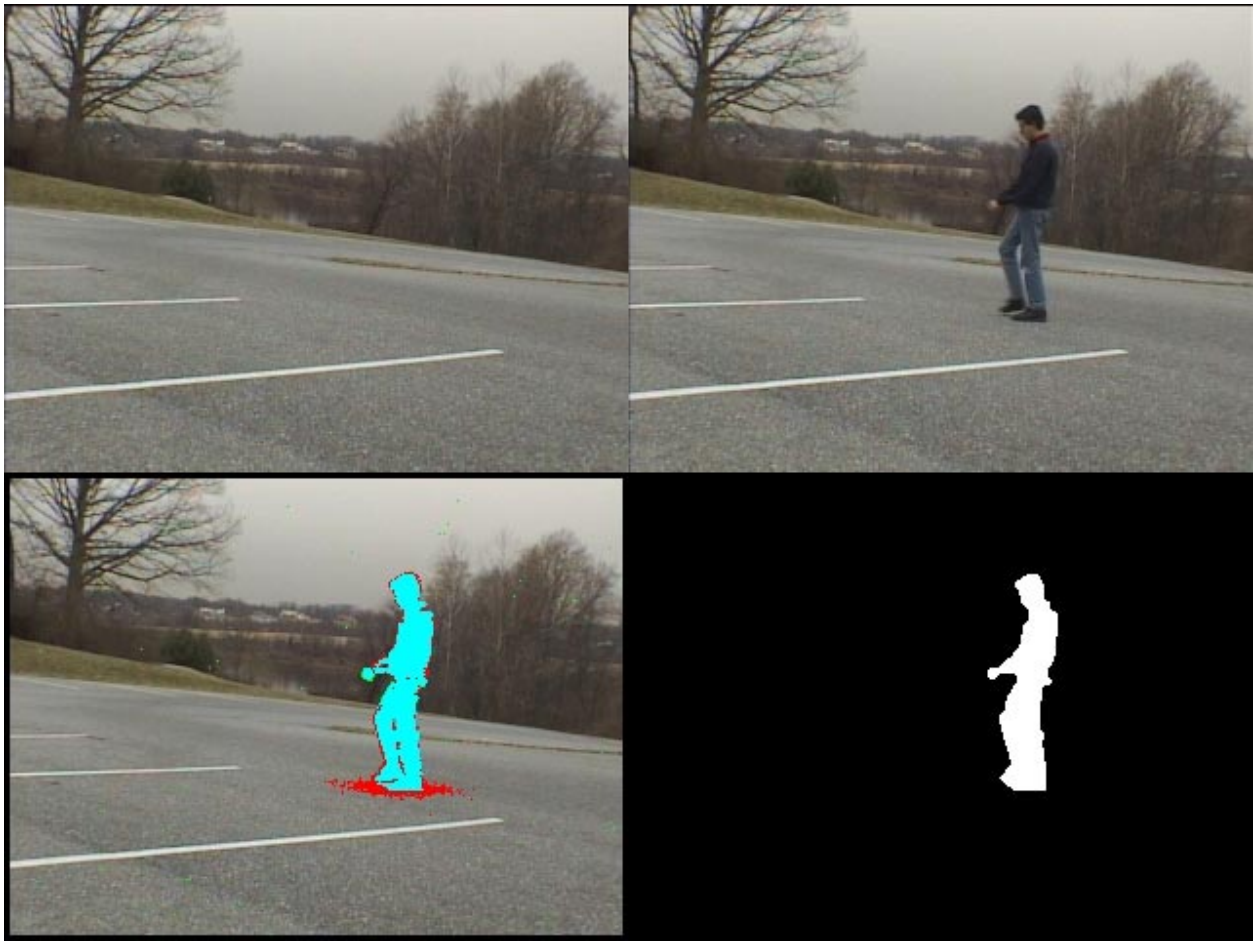

Figure 8: Another example shows the results of applying our algorithm on a sequence of an outdoor scene containing a person walking across the street.

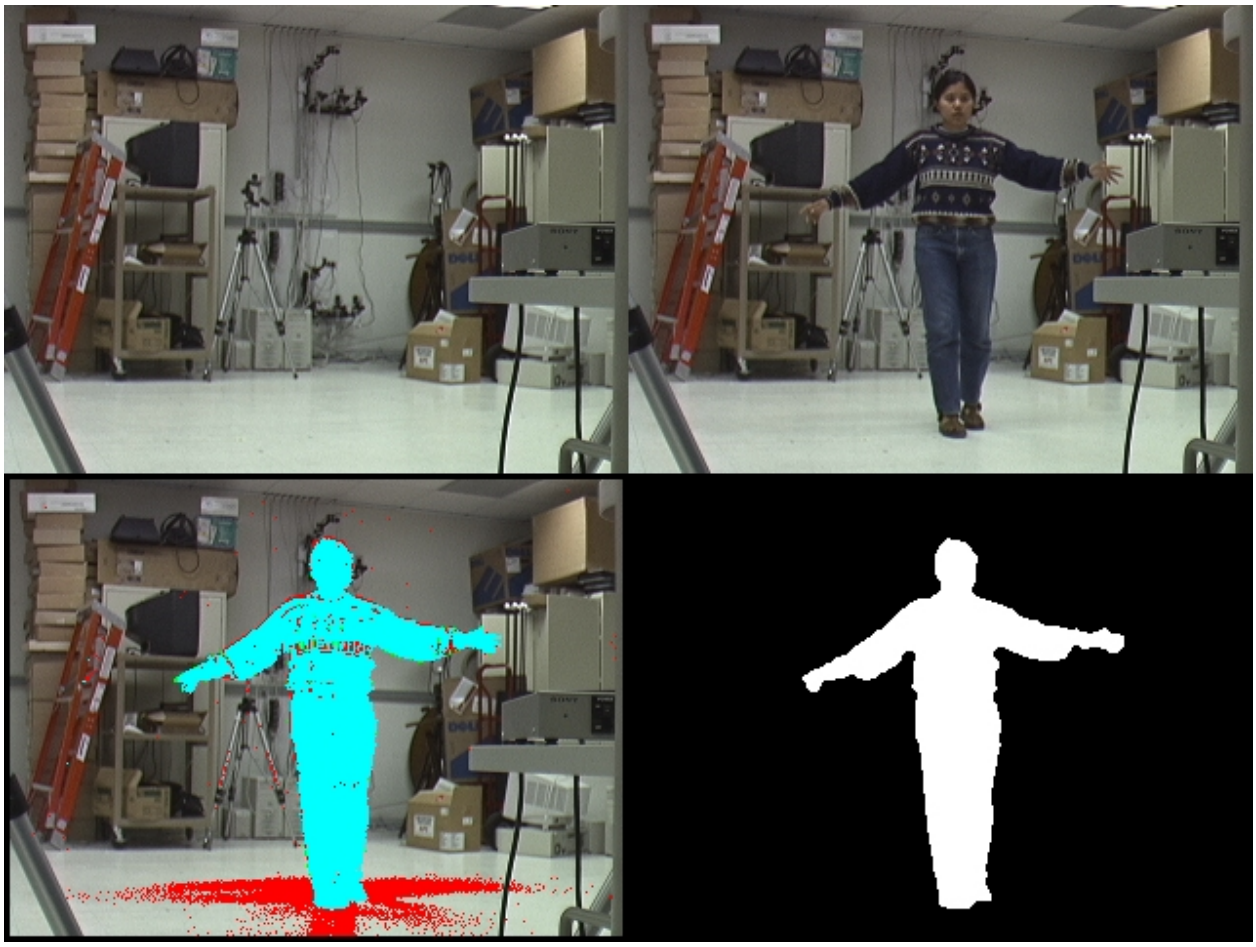

Figure 9: An illustration shows our algorithm can cope with the global illumination change. At the middle of the sequence, half of the fluorescence lamps are turned off. The result shows that the system still detects the moving object successfully.

7 Speed-Up Technique Used in Our Implementation

In designing the algorithm, while we emphasized accuracy and robustness, we also took an issue of speed into account. To speed up the system, we employed the following techniques.

Reduce number of operations at run-time: If memory is not a hard constraint, we can pre-compute some constant parameters and store them during the background learning period. This reduces the number of arithmetic operations that are computed during the detection time. For example, some of the parameters used in Equation 5 can be pre-computed as follows:

$$\begin{aligned} A_i &= \left(\left[\frac{\mu_R(i)}{\sigma_R(i)}\right]^2 + \left[\frac{\mu_G(i)}{\sigma_G(i)}\right]^2 + \left[\frac{\mu_B(i)}{\sigma_B(i)}\right]^2 \right) \\ B_i &= \frac{\mu_R(i)}{A_i \sigma_R^2(i)} \\ C_i &= \frac{\mu_G(i)}{A_i \sigma_G^2(i)} \\ D_i &= \frac{\mu_B(i)}{A_i \sigma_B^2(i)} \end{aligned}$$

At run-time, the computations become

$$\alpha_i = B_i I_R(i) + C_i I_G(i) + D_i I_B(i)$$

In addition, because a multiplication operation is computed faster than a division operation, we store $(s_i)^{-1}$, $(a_i)^{-1}$, and $(b_i)^{-1}$ in stead of s_i, a_i, and b_i for normalizing processes. These techniques sped up our system significantly.

Global S vs local s_i: In most cases, we can use a global square root of average variance (S) in stead of a per-pixel s_i to be used in rescaling the color values. S is given by

$$S = [S_R, S_G, S_B]$$

where

$$S_C = \sqrt{\frac{1}{NXY} \sum_{n=1}^{N} \sum_{i=1}^{XY} (I_C(i,n) - \mu_C(i,n))^2} \quad ; \quad C = R, G, B$$

By using this global S for all pixels in the subtraction process (Equation 5, 6), S always resides in cache. As a result, the processing time is sped up approximately by a factor of 2 as shown in Table 1 below.

	Using a global $\mathbf{S}$ for color normalization	Using local s_i for color normalization
Single 400MHz CPU	27.6 ms/frame	48.9 ms/frame
Dual 400MHz CPUs	13.8 ms/frame	30.6 ms/frame

Table 1: Background subtraction timing result shows a comparison between using global vs local normalization factors, and spped up gained from utilizing parallelism.

Screening Test: Another technique that we employ to speed up our detection time is utilization of a screening test. Because the $\widehat{\alpha_i}$ and $\widehat{CD}_i$ computations require many arithmetic operations, we want to avoid computing these variables of every pixel in the image. We add a low computational cost screening test on color disparity between the current image and the background image. This screening test is simply done by computing an absolute different between the observed pixel and the corresponding background pixel as follows:

$$|I_i - E_i| < T$$

The result from this screening test yields only a subset of pixels that has too much disparity from the background image. Hence, with this screening test, we can reduce the number of pixels to which we need to apply the refining test on $\widehat{\alpha_i}$ and $\widehat{CD}_i$ as in Equation 5, 6, 9, 10, and 12. One question arises: what is the appropriate T value to preserve the desired detection, r. We solve this by an empirical method. We construct a histogram of $|I_i - E_i|$ and find the maximum T that yields a probability that the screening test passes, given that the pixel is classified as an original background, equal to 100%:

$$P(|I_i - E_i| < T | M(i) = B) = 1$$

Our experimental results show this screening test can speed up the system approximately by a factor of 1.5 - 2 on average.

Parallel Processing: Since our background modeling and background subtraction are pixel by pixel, the algorithm can be parallelized with little effort by dividing the images into segments and performing the operations independently on each segment.

8 Conclusion and Discussion

In this paper, we presented a novel background subtraction algorithm for detecting moving objects from a static background scene that contains shading and shadows using color images. The method

is shown to be very accurate, robust, reliable and efficiently computed. Experimental results and speed up techniques were also shown.

This method may suffer from dynamic scene change such as an extraneous event in which there are new objects deposited into the scene and become part of the background scene. However, this problem can be coped with by allowing the system to adaptively update the background model on-the-fly, as done in [16] and [17]. Another limitation of the system is the problem of reflection on highly specular surfaces (such as mirror, steel, or water surface) when the color of a point on such surfaces can change non-deterministically. This problem is a part of our on-going research.

Streaming SIMD technology, introduced in Intel PentiumIII processor, defines a new SIMD architecture for floating-point operations. These instructions can also significantly improve the speed of our method which is using floating-point intensive computations.

References

[1] Ismail Haritaoglu, David Harwood, and Larry S. Davis, "W4: Who? When? Where? What? a Real-time System for Detecting and Tracking People," *Proc. the third IEEE International Conference on Automatic Face and Gesture Recognition (Nara, Japan)*, IEEE Computer Society Press, Los Alamitos, Calif., 1998, pp.222-227.

[2] P.L.Rosin, "Thesholding for Change Detection," *Proceedings of International Conference on Computer Vision*, 1998.

[3] N. Friedman, and S. Russell, "Image Segmentation in Video Sequences: A Probabilistic Approach", Proceedings of the 13th Conference on Uncertainty in Artificial intelligence, Morgan Kaufmann, 1997.

[4] C.R. Wren, A. Azarbayejani, T. Darrell, and A. Pentland, "Pfinder: Real-time Tracking of the Human Body," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol.19, No.7, IEEE Computer Society Press, Los Alamitos, Calif., July 1997, pp.780-785.

[5] J. Ohya, et al., "Virtual Metamorphosis", IEEE Multimedia, April-June 1999.

[6] J. Davis, and A. Bobick, "The representation and Recognition of Action using Temporal Templates", Proceedings of Conference on Computer Vision and Pattern Recognition, 1997.

[7] A. Utsumi, H. Mori, J. Ohya, and M. Yachida, "Multiple-Human Tracking using Multiple Cameras", Proceedings of the Third International Conference on Automatic Face and Gesture Recognition (FG'98), April 1998.

[8] M. Yamada, K. Ebihara, and J. Ohya, "A New Robust Real-time Method for Extracting Human Silhouettes from Color Images", Proceedings of the Third International Conference on Automatic Face and Gesture Recognition (FG'98), April 1998.

[9] T. Horprasert, I. Haritaoglu, C. Wren, D. Harwood, L.S.Davis, and A. Pentland, "Real-time 3D Motion Capture", Proceedings of 1998 Workshop on Perceptual User Interfaces (PUI'98), November 1998.

[10] S.A. Shafer, and T. Kanade, "Using Shadows in Finding Surface Orientations", CVGIP 22:145-176, 1983.

[11] C. Lin and R. Nevatia, "Building Detection and Description from a Single Intensity Image", CVIU 72:101-121, 1998.

[12] J. Segen, and S. Kumar, "Shadow Gestures: 3D Hand Pose Estimation using a Single Camera", Proceedings of Conference on Computer Vision and Pattern Recognition, 1999.

[13] J-Y. Bouguet, M. Weber, and P. Perona, "What do planar shadows tell about scene geometry?", Proceedings of Conference on Computer Vision and Pattern Recognition, 1999.

[14] A.C. Hurlbert, "The computation of Color", MIT Artificial Intelligence Laboratory Technical Report 1154.

[15] P.L. Rosin, and T. Ellis, "Image difference threshold strategies and shadow detection", Proceedings of the sixth British Machine Vision Conference, 1994.

[16] C. Ridder, O. Munkelt, and H. Kirchner, "Adaptive Background Estimation and Foreground Detection using Kalman-Filtering", ICRAM, 1995.

[17] A. Elgammal, D. Harwood, and L.S. Davis, "Non-parametric Model for Background Subtraction", ICCV Frame Rate Workshop, 1999.

[18] G.J. Klinker, S.A. Shafer, and T. Kanade, "A Physical Approach to Color Image Understanding", Int

[19] S.A. Shafer,"Using Color to Separate Reflection Components", COLOR Research and application, 10(4):210-218, 1985