Forefront and Shadow Detection for Video Surveillance

SUWOO PARK*, JOOSEOP YUN*, SEHYUN PARK**, YONGTAE DO**†

*GIVET, **Daegu University
Naeri-Ri 15, Jinryang-Eup, Gyeongsan-City, Gyeongbuk, 712-714
SOUTH KOREA
†ytdo@daegu.ac.kr http://cms.daegu.ac.kr/ytdo

Abstract: - In automatic visual surveillance, moving target extraction and shadow depression are prerequisites for higher level image processing steps such as target tracking and scene understanding. In this paper, we present effective techniques to detect pixels of moving targets in a scene and pixels of their shadows. The foreground is extracted by adaptive background subtraction. Incomplete detection problem, which often occurs due to color or occlusion, is solved by blob merging. Shadow pixels are detected by comparing the color values of foreground pixels with those of corresponding background pixels using either a color transition model or a neural network. Techniques proposed showed high shadow detection rates in experiments with real images.

Key-Words: - Video surveillance, Foreground image, Shadow, Moving targets, Neural network.

1 Introduction
For recent years, video surveillance (VS) has been attracting increasing attention in computer vision community. Human and vehicle are two important targets in most VS applications. The former in video image sequence is often monitored for security purpose [1] while the latter is monitored mainly for traffic control [2].

Moving targets such as human and vehicle in images are extracted first as foreground by various methods, and only foreground regions are processed further in later steps. Detecting foreground pixels effectively is of great importance but blobs of foreground pixels detected on an image are often incomplete. A target, for example, can be divided into multiple blobs. Causes include similar color values between the target and background, or obstacles in front of the target.

When detecting targets, their shadows are also detected because shadows move together with the targets. As shadows can cause errors in target classification and analysis, removing shadow pixels is necessary.

In this paper, we propose a technique to merge blobs of a target image due to incomplete foreground detection. Techniques to detect the shadows of targets in foreground are also proposed by modeling color valued transition due to shadow and by employing a feedforward neural network.

This paper is organized as follows. In Section 2, foreground detection problem is discussed. In Section 3, techniques to detect shadow pixels are proposed. In Section 4, experimental results are provided, and finally, in Section 5, concluding remarks are given.

2 Foreground Detection and Merging
Moving target detection is the most fundamental procedure in VS. There are three existing methods, temporal differencing [3], using optical flow [4], and background subtraction [5]. The background subtraction is popularly used because it keeps the shapes of moving objects unlike temporal differencing, and its processing is simpler comparatively to optical flow technique.

An image pixel is classified as foreground if it is quite different from the corresponding pixel in the background image. In [5], the background image is updated adaptively according to surrounding change. A pixel \( p_n \) on the \( n \)’th frame is regarded as foreground if it stays as a foreground candidate for longer than some fixed time duration. A foreground candidate pixel is found by Eq. (1) for properly set parameters \( k \) and \( \alpha \).

\[ p_D = |p_n - \overline{p_n}| \geq k\overline{\sigma}_{n-1} \]
where \( \overline{p_n} = \alpha p_n + (1-\alpha)\overline{p}_{n-1} \)
\( \overline{\sigma}_n = \alpha |p_n - \overline{p}_n| + (1-\alpha)\overline{\sigma}_{n-1} \)
Eq. (1) proposed in [5] is adaptive for varying light condition and practically effective. However, often a target detected is separated into two or more parts due mainly to color and occlusion by an obstacle. It thus needs a procedure to merge blobs detected separately but originally belong to the same target. Fig. 1 shows steps for foreground detection and blob merging.

To determine if two foreground blobs can be merged into one, the distance between them is checked. In Fig. 2, when two blobs have radiiuses $r_1$ and $r_2$, they are merged into one if $r_1 + r_2 < \text{threshold}$. By this way, the two blobs of a walking person in Fig. 1 are merged. A car detected in two blobs due to a street lamp is represented in a single bounding box in Fig. 3.

3 Shadow Pixel Detection

When a target moves in a camera’s view, not only the target but also its shadow is detected as it is attached to the target. In [6], Cucchiara and colleagues detected shadow pixels using the fact that shadow does not change H and S values much but lowers V by significant amount in [H,S,V] color space. Specifically, Eq.(2) was used as the conditions to detect shadow pixels.

\begin{align}
\text{(i)} & \quad \alpha \leq \frac{I_v(x,y)}{B_v(x,y)} \leq \beta \\
\text{(ii)} & \quad dS(x,y) = I_s(x,y) - B_s(x,y) \leq \tau \\
\text{(iii)} & \quad |I_h(x,y) - B_h(x,y)| \leq \lambda
\end{align}

where subscripts mean V, S, and H color values. $I(x,y)$ and $B(x,y)$ are the pixel values at coordinate $(x,y)$ on an image being processed and corresponding background image respectively, and $\alpha$, $\beta$, $\tau$, $\lambda$ are decision thresholds. A problem of this method is that these thresholds are set on entire image and local characteristics of a scene cannot be considered. In Fig. 4(b), in which black dots represent shadow pixels detected by the Cucchiara’s method [6] for Fig. 4(a), shadows on the grass field are not detected well.

In this paper, every pixel on the background image is assumed to have its own position-dependent features. So, their pixel values vary differently when
shadows are on them. In our experimental study, shadow decreases the V value of a pixel less at dark background than it does at bright parts. This fact can be modeled by a cubic curve like Eq. (3).

\[ D_c = k_1B^1_v + k_2B^2_v + k_3B^3_v + k_4 \]  

(3)

A least-square method is employed to determine the coefficients, \( k_1, \ldots, k_4 \), using a number of sample shadow pixels, and \( \pm 3\sigma \) from the curve is used as thresholds in detecting shadow pixels, where \( \sigma \) indicates the standard deviation of V drops for the same intensity of V component in the background image. A condition to determine a shadow pixel is then

\[ |D_v - D_c| \leq \alpha \]  

(4)

where \( D_v = B_v(x,y) - I_v(x,y) \) and \( \alpha \) is a threshold. When our method is applied to the Fig. 4(a), the result is as shown in Fig. 4(c), and it is effective on most parts of the image unlike Fig. 4(b).

![Fig. 4. Shadow pixel detection: (a) Original image, (b) By the method of Cucchiara [6], (c) By the color transition model proposed.](image)

An artificial neural network (ANN) is also employed for determining shadow pixels. A two-layer feedforward network is constructed and trained by back-propagation (BP) algorithm [7]. An ANN has learning capability and practically advantageous. We use [R,G,B] values of foreground and background image pixels for network input as shown in Fig. 5.

![Fig 5. Neural network constructed for detecting shadow pixels.](image)

### 4 Result

The shadow detection techniques proposed were tested with real video images. The performance was measured by detection rate \( \eta \) and discrimination rate \( \xi \), which were defined by Prati [8] as Eq.(5),

\[
\eta = \frac{TP_S}{TP_S + FN_S} \quad (5.a) \\
\xi = \frac{TP_F}{TP_F + FN_F} \quad (5.b)
\]

where the subscripts \( S \) and \( F \) indicate shadow and foreground respectively. The \( TP \) stands for the number of pixels correctly classified while \( TP_F \) is the number of foreground points minus that of points detected as shadow but actually belong to foreground.

An experimental result for the image of Fig. 4 is summarized in Table 1. Method of [6] is compared with the modeling method proposed in Eqs. (3) and (4). The method of this paper showed better performance for both cement and grass regions of the scene.

<table>
<thead>
<tr>
<th>Techniques</th>
<th>Regions</th>
<th>( \eta )</th>
<th>( \xi )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cucchiara method</td>
<td>Cement</td>
<td>0.75</td>
<td>0.71</td>
</tr>
<tr>
<td></td>
<td>Grass</td>
<td>0.39</td>
<td>0.74</td>
</tr>
<tr>
<td>Color transition model</td>
<td>Cement</td>
<td>0.80</td>
<td>0.77</td>
</tr>
<tr>
<td></td>
<td>Grass</td>
<td>0.85</td>
<td>0.75</td>
</tr>
</tbody>
</table>
When an ANN was used with different numbers of hidden nodes for the image of Fig. 6, we could get results like Table 2. The detection rate $\eta$ is significantly better with the ANN method while the discrimination rate $\xi$ is similar compared to the method of [6].

Table 2. Shadow detection performance for the image of Fig.6(a).

<table>
<thead>
<tr>
<th>Techniques</th>
<th># (Hidden nodes)</th>
<th>$\eta$</th>
<th>$\xi$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cucchiara method</td>
<td>0.76 0.85</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ANN</td>
<td>4 0.92 0.84</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8 0.89 0.86</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Fig. 6. Shadow pixel detection: (a) Original image, (b) Shadow pixels in black dots detected by a neural network of four hidden nodes.

5 Conclusion

Detecting moving targets in an image sequence is a fundamental task for video surveillance. In this paper we proposed efficient techniques to extract moving targets in color video image. Using background subtraction, foreground image blobs are first obtained. If any two blobs among them are close enough, they are considered as those separated due to incomplete detection and merged into a single blob. Shadows distort the shape of a target detected as foreground, and we proposed to use a color transition model or a neural network. Both techniques showed better results in our experiments compared to a widely used technique proposed in [6]. Specifically, color transition model showed consistent performance for different areas of the scene monitored. The performance of neural network was good also and did not vary much for different numbers of hidden nodes.

References: