LIGHT-WEIGHT SALIENT FOREGROUND DETECTION FOR EMBEDDED SMART CAMERAS

Mauricio Casares, Senem Velipasalar

University of Nebraska-Lincoln
Dept. of Electrical Engineering, Lincoln, NE, 68588
mcasare1@bigred.unl.edu, velipasa@engr.unl.edu

ABSTRACT
Limited processing power and memory in embedded smart camera nodes necessitate the design of light-weight algorithms for computer vision tasks. Considering the memory requirements of an algorithm and its portability to an embedded processor should be an integral part of the algorithm design in addition to the accuracy requirements. This paper presents a light-weight and efficient background modeling and foreground detection algorithm that is highly robust against lighting variations and non-static backgrounds including scenes with swaying trees, water fountains, rippling water effects and rain. Contrary to many traditional methods, the memory requirement for the data saved for each pixel is very small, and the algorithm provides very reliable results with gray-level images as well. The proposed method selectively updates the background model with an automatically adaptive rate, thus can adapt to rapid changes. As opposed to traditional methods, pixels are not always treated individually, and information about neighbors is incorporated into decision making. The algorithm differentiates between salient and non-salient motion based on the reliability or unreliability of a pixel’s location, and by considering neighborhood information. The results obtained with various challenging outdoor and indoor sequences are presented, and compared with the results of different state of the art background subtraction methods. The experimental results demonstrate the success of the proposed light-weight salient foreground detection method.

Index Terms—foreground detection, background subtraction, salient motion, pixel reliability, light-weight algorithms

1. INTRODUCTION
Embedded smart camera nodes have limited processing power and limited memory. Thus, it is necessary to design light-weight algorithms for computer vision tasks. In addition to the accuracy of an algorithm, it is very important to consider its efficiency, memory requirements and portability to an embedded processor during algorithm design.

In this paper, we present a light-weight algorithm for salient foreground detection, which is the first step in most of the object tracking applications. Existing methods for foreground object detection can be broadly classified into two categories: temporal difference methods [1][2], and background subtraction methods [3][4][5][6][7][8][9]. Temporal difference methods subtract two consecutive frames and then apply a threshold to the output. These methods perform well when the background changes over time, however they cannot detect all the pixels of a moving object. Background subtraction methods build a model of the background and subtract this from the current image to detect the objects in the scene. In order to adapt to changes in the environment, the background model is usually updated over time [3][5][6][7][8]. The method proposed in this paper is a hybrid method that employs temporal difference to aid background modeling.

Oliver et al. [10] present an eigenbackground method, where images of a static background are collected, and PCA is employed to reduce the dimensionality of space. Input images are projected onto the PCA subspace, and a threshold is applied to the difference between the projected and current image to find the foreground regions.

Horprasert et al. [11] obtain expected chromaticity by the arithmetic mean of the RGB values calculated over a number of background images. By using several thresholds, pixels are classified as foreground, background, shadow and highlighted background.

Adaptive Mixture of Gaussians (MoG), introduced by Stauffer and Grimson [7], is one of the most commonly used background subtraction methods to model complex and non-static backgrounds. However, a few Gaussian distributions are usually not sufficient to accurately model backgrounds having fast variations. Zivkovic [13] proposed an improved adaptive MOG model to constantly update the parameters of a Gaussian mixture and to simultaneously select the appropriate number of components for each pixel.

Kim et al. [12] proposed an algorithm for background modeling, where sample background values at each pixel are
quantized into codebooks during training, which represent a compressed form of the background model. This algorithm performs well when background is non-static or there are lighting variations. However, its performance on different video sequences is dependent on the choice of several threshold values.

Lighting variations and non-static backgrounds, such as scenes with swaying trees and water fountains, make the foreground detection problem challenging, since we are interested only in salient motion. The necessity of handling these challenging cases increases the algorithm complexity, and thus memory requirements. For instance, Stauffer and Grimson [7] use adaptive mixture of Gaussians, with multiple Gaussian distributions per pixel, to model non-static backgrounds. Kim et al. [12] form codewords for each pixel to capture the different values at that pixel location.

In this paper, we present a method that is highly robust against lighting variations and non-static backgrounds, and is light-weight at the same time. The memory requirement of the proposed method for the data saved for each pixel is very small compared to many traditional background subtraction methods. For instance, in the case of adaptive mixture of Gaussians [7], three to five Gaussian distributions are used for each pixel. Each distribution is represented by its mean and variance, and these values are floats. In the codebook-based method [12], each codeword has nine entries. Some of these entries need to be floats, and on the average 6.5 codewords are needed for a pixel. In our method, for each pixel location, intensity value is used in the model, and only two additional counter values are needed.

The algorithm presented selectively updates the background model with an automatically adaptive rate, thus can adapt to rapid changes as well. For instance, if a location is deduced to be reliable, a higher update rate is used, i.e. this location is incorporated to the background faster. The reliability concept and how the background model is updated will be explained in detail below. Unlike many traditional methods treating each pixel individually, in the proposed method, information is obtained from neighboring pixels and incorporated into decision making, which increases accuracy and robustness. Most background subtraction methods assume that the empty scene without any moving objects is available for the training period during which the background model is learned. In contrast, the proposed algorithm is capable of handling moving objects in the scene during the building of the background model. The algorithm gives very reliable results with gray level images. All the experimental results presented in Section 3 were obtained with gray level images. The experiments were performed on three different video sequences, with non-static backgrounds and varying levels of difficulty, and the same threshold values were used for all of them. Thus, the dependency on the threshold values is low. The experimental results also demonstrate the success of the proposed light-weight salient foreground detection method in challenging situations such as scenes with water fountains, swaying trees, and strong wind and rain. We ran our algorithm and five other state of the art background subtraction methods on the same sequences, and compared the results.

The rest of the paper is organized as follows: The details of the proposed method is explained in Section 2. Experimental results are presented in Section 3, and the paper is concluded with a summary and discussion of future work in Section 4.

2. THE PROPOSED METHOD

The proposed algorithm employs a temporal difference method until a complete background model is built. It differentiates between salient and non-salient motion based on the reliability or unreliability of a pixel’s location, and by incorporating neighborhood information. The reliability of a pixel is defined as follows: At each frame, each pixel is classified either as a background or a foreground pixel, and its state is set to be 0 or 1, respectively. For a pixel at location \((i, j)\), a counter \(h(i, j)\) holds the number of changes in the state of this pixel, i.e. the counter \(h(i, j)\) keeps the number of times a pixel’s state changes from 0 to 1 or vice versa. The reliability of a pixel at location \((i, j)\) is determined by this counter \(h(i, j)\). The motivation is that the lower the value of \(h(i, j)\), the more stable and reliable that location is, or vice versa. Until a complete background model is built, the state of a pixel is determined by using temporal difference.

The algorithm also has an adaptive background model update rate. If the counter \(h(i, j)\) for a pixel has not changed, i.e. the state of that pixel has remained the same for the last \(n\) frames, then the value of this pixel is incorporated to the background model with a higher weight.

Instead of treating each pixel independently, information from neighboring pixels is used to differentiate between salient and non-salient motion, and in turn to classify a pixel as a foreground or background pixel.

The details of the proposed algorithm will be explained by referring to the pseudo-code provided in Table 1. In Table 1, \(M\) and \(M_{prev}\) denote the current and previous background models, respectively. \(s(i, j)\) denotes the status of a pixel at location \((i, j)\), and \(h(i, j)\) holds the number of times that a pixel changes states in the last 100 frames. The background model \(M\) is built by using a temporal difference method. In order to detect slow motions or stopping objects, a weighted accumulation, \(I_{t}^{ac}\), is used for temporal difference. At pixel location \((i, j)\), \(I_{t}^{ac}\) is defined as:

\[
I_{t}^{ac}(i, j) = (1 - w_{ac}) I_{t-1}^{ac}(i, j) + w_{ac} [I_{t}(i, j) - I_{t-1}(i, j)]
\]

where \(t\) is the current frame number, \(I_{t}\) is the current image frame, \(w_{ac} = 0.5\), and \(I_{t}^{ac}\) is set to be an empty image.

The background model \(M\) is formed as follows: At the beginning \(M(i, j) = -1\) for every pixel location \((i, j)\). \(M(i, j)\) is set to be \(I_{t}(i, j)\), if \(I_{t}^{ac}(i, j) < T\), where \(T\) is
a difference threshold. Thus, as moving objects that exist in the scene change their location, the $M$ will gradually be filled. If $M$ is not complete, the difference image ($I_{diff}$) is set to be $I_t^{ac}$. If the $M$ is complete, then $I_{diff} = I_t^{md}$, where $I_t^{md} = |I_t - M|$.

As stated above, $T$ is a difference threshold. When the model $M$ is not complete, and $I_{diff}$ is based on temporal difference method $T = T_d = 15$ is used (Table 1). When the model $M$ is complete, and $I_{diff}$ is set to be $I_t^{md}$, then $T = T_m = 25$ is employed. The same values have been used in the experiments for all the video sequences. Since temporal difference is based on consecutive frames, and tends to give smaller differences, $T_d$ has a smaller value than $T_m$.

In parallel to building the model, counters $h(i,j)$ are updated. These counters determine the reliability or unreliability of a pixel, which is used to decide if that pixel can be classified as salient motion or can be incorporated to the background.

The update of the background model $M$ is performed in a selective way, and with an automatically adaptive rate. The motivation is that when a pixel’s location is deduced to be consistently reliable, then the value at that location is incorporated into the background model with a higher weight. As opposed to traditional model-based background subtraction approaches, in the proposed scheme satisfying $I_t^{md}(i,j) > T$ is not enough for the pixel location $(i,j)$ to be classified as foreground. Instead, reliability constraints are employed to differentiate between salient and non-salient motion. A pixel location satisfying $I_t^{md}(i,j) > T$ is classified as foreground only if its counter $h(i,j)$ satisfies $h(i,j) < T_p$, where $T_p = 15$ is the percentage threshold. The reasoning is that if $h(i,j) < 15$, then it means that the state of the pixel at this location changed less than 15% of the time during the last 100 frames making this location a reliable one. In other words, this location is not likely to be in a non-salient motion region. Thus the intensity difference greater than $T$ is caused by a salient motion with high probability. If $I_t^{md}(i,j) > T$ and $h(i,j) < T_p$, then we do not classify this location as background right away. We take a $(2w+1) \times (2w+1)$-window neighborhood, where $w = 1$, around location $(i,j)$ and check the $h$ counter for all the neighbors. In Table 1, $N$ is the number of neighbors whose counter $h$ is less than $T_p$. If the majority of the neighbors have a low counter, i.e., $h < T_p$, then location $(i,j)$ is set to be a foreground pixel or vice versa. This way, we take into account the fact that neighboring pixels are not independent from each other. Unlike many traditional methods treating each pixel individually, we obtain information from neighbors, which increases accuracy and robustness. If above condition is not satisfied, and this location is set as background, then the model is updated by giving only 5% weight to the current frame, and 95% weight to the previous model, as can be seen in Table 1.

If $I_t^{md}(i,j) \leq T$, then we can conclude that at this location it is safe to update the background model. However, by looking at the summary of the recent past of a pixel, we can give a higher weight to the current pixel value, and better adapt to faster changes in the background. In other words, we have an automatically adaptive background update rate. The very compact summary of a pixel’s history is formed as follows: Rather than saving many values for each pixel location, such as averages for three color values, multiple Gaussian dis-

<table>
<thead>
<tr>
<th>Table 1. Salient foreground detection algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set $M(i,j) = -1$ for all $i,j$.</td>
</tr>
<tr>
<td>Set status variables $s(i,j) = 0$ for all $i,j$.</td>
</tr>
<tr>
<td>Set $I_t = t$th frame, $I_{ac}^{md}(i,j) = -1$ for all $i,j$. for every frame $t &gt; 1$.</td>
</tr>
<tr>
<td>If $M(i,j) = -1$, compute $I_{ac}^{md}$, $I_{diff} = I_{ac}^{md}$, $T = T_d$ else</td>
</tr>
<tr>
<td>Set model_complete = true, compute $I_t^{md}$, $I_{diff} = I_t^{md}$, $T = T_m$ for all $i,j$.</td>
</tr>
<tr>
<td>If $I_{diff} \geq T$</td>
</tr>
<tr>
<td>If $s(i,j) = 1$</td>
</tr>
<tr>
<td>Update counters and set $s(i,j) = 1$ else if $s(i,j) = 1$</td>
</tr>
<tr>
<td>Update counters and set $s(i,j) = 0$ if $M(i,j) = -1</td>
</tr>
<tr>
<td>$M(i,j) = I_t(i,j)$</td>
</tr>
<tr>
<td>Update $h(i,j)$ if $t$ is a multiple of 50</td>
</tr>
<tr>
<td>If $h(i,j) \leq h_{prev}(i,j)$</td>
</tr>
<tr>
<td>Set $R_{flag}(i,j) = 1$</td>
</tr>
<tr>
<td>Set $h(i,j) = 0$, set $h_{prev} = h$ if model_complete = true</td>
</tr>
<tr>
<td>If $I_t^{md}(i,j) &gt; T$</td>
</tr>
<tr>
<td>If $h(i,j) &lt; T_p$</td>
</tr>
<tr>
<td>$I_{out}(i,j) = 1$; else</td>
</tr>
<tr>
<td>Set $R_{flag}(i,j) = 1$</td>
</tr>
<tr>
<td>$I_{out}(i,j) = 1$; else if $R_{flag}(i,j) = 1$</td>
</tr>
<tr>
<td>$M(i,j) = 0.5 \times I_t(i,j) + 0.5 \times M_{prev}(i,j)$ else</td>
</tr>
<tr>
<td>$M(i,j) = 0.5 \times I_t(i,j) + 0.5 \times M_{prev}(i,j)$ Set $R_{flag}(i,j) = 0$ for all $i,j$. Set $I_{prev} = I_{ac}^{md}$, $M_{prev} = M$ return $I_{out}$</td>
</tr>
</tbody>
</table>

| Note: $I_t^{md}$ is the difference image between the current frame and the previous model. $I_{ac}^{md}$ is an average of multiple Gaussian distributions. $I_{diff}$ is the difference image between the current frame and the previous model. $I_{out}$ is the output image. $h(i,j)$ is the reliability counter for pixel $(i,j)$. $M$ is the model of background. $T_d$ and $T_m$ are the threshold for temporal difference. $T_p$ is the percentage threshold. $R_{flag}$ is a flag for the current pixel. $N$ is the number of neighbors whose counter $h$ is less than $T_p$. $I_{prev}$ is the previous frame. $M_{prev}$ is the previous model.
Fig. 1. Illustration of how $h(i, j)$ is computed.

for the pixel at location $(i, j)$.

3. EXPERIMENTAL RESULTS

The experiments were performed with several video sequences with varying levels of difficulty. The proposed method was tested and compared with five other background subtraction methods on these sequences. All the displayed outputs are the images obtained without applying any morphological or post-processing operations. All the results of our algorithm were obtained by using the same threshold values for all videos. Specifically, $T_d = 15$, $T_m = 25$, $T_p = 15$, and $\alpha = 0.05$. Figure 2 shows the outputs obtained with the proposed method and different state of the art background subtraction algorithms. This is a video of a scene with a fountain, where the water level goes up and down. Moreover, during the video, the lighting changes due to moving clouds. First and second columns of Fig. 2 correspond to frames 983 and 1800, respectively. The difference in the lighting can be seen from Figures 2(a) and 2(b). As the figure illustrates, the original MoG method is not fast enough to adapt to the lighting change. The improved MoG method cannot detect most of the foreground pixels. The proposed method performs better compared to others by having the least amount of noisy pixels and good detection at the same time. We also tried the eigenbackgrounds [10] and the method by Horprasert et al. [11] on all the videos, but only displayed the outputs of the algorithms giving the better results.

Figure 3 shows the outputs obtained on an indoor video sequence. Although the video was captured indoors, the flickering of the overhead lights affects the performance of the algorithms. The proposed method performs reliably, and like in the above example, can differentiate the salient motion from non-salient one.

Figure 4 is another example of an outdoor video, where there is strong wind and rain. The proposed algorithm per-
forms better in terms of eliminating non-salient motions caused by wind and rain.

As stated above, while the other algorithms were run with color images, the results of our method were obtained with gray-level images. Thus, overall, the algorithm provides better background update and better elimination of non-salient motions while requiring less memory for data storage at the same time.

4. CONCLUSIONS AND FUTURE WORK

We presented a background modeling and salient foreground detection algorithm that is light-weight, and can handle challenging non-static backgrounds including scenes with swaying trees, water fountains, rippling water effects, rain and flickering lights. Salient motion is differentiated from non-salient background motion based on the reliability of a pixel’s location, and by incorporating information from neighboring pixel locations into decision making. The algorithm performs well with gray-level images, and does not need to save large amounts of data for each pixel location, which is very important for portability to an embedded controller. It selectively updates the background model with an automatically adaptive rate, thus can adapt to rapid changes. The results obtained with various challenging outdoor and indoor sequences have been presented, and compared with the results of different state of the art background subtraction methods. The experimental results demonstrate the success of the proposed light-weight salient foreground detection method.

As future work, we would like to incorporate shadow detection and removal into our algorithm.

5. REFERENCES


Fig. 2. Comparison of foreground detection results of several different algorithms on a video of a fountain.
Fig. 3. Foreground detection results of several different algorithms on a challenging video with flickering lights.

Fig. 4. Foreground detection results of several different algorithms on a challenging video with strong wind and rain.