USING THE DISCRETE HADAMARD TRANSFORM TO DETECT MOVING OBJECTS IN SURVEILLANCE VIDEO

Chanyul Kim and Noel E.O'Connor
CLARITY: Centre for Sensor Web Technologies, Dublin City University, Glasnevin, Dublin, Ireland
Chanyul.kim,oconnorn@eeng.dcu.ie

Keywords: Discrete Hadamard Transform, Moving object detection, Edge.

Abstract: In this paper we present an approach to object detection in surveillance video based on detecting moving edges using the Hadamard transform. The proposed method is characterized by robustness to illumination changes and ghosting effects and provides high speed detection, making it particularly suitable for surveillance applications. In addition to presenting an approach to moving edge detection using the Hadamard transform, we introduce two measures to track edge history, Pixel Bit Mask Difference ($\mathcal{PBMD}$) and History Update Value ($\mathcal{HUV}$) that help reduce the false detections commonly experienced by approaches based on moving edges. Experimental results show that the proposed algorithm overcomes the traditional drawbacks of frame differencing and outperforms existing edge-based approaches in terms of both detection results and computational complexity.

1 Introduction

Moving object detection in video sequences is an important research area since it can be viewed as a lower level vision task necessary to achieve higher level event understanding. It is a critical task for applications such as video surveillance, traffic monitoring and video compression e.g. region of interest (ROI) coding. In this paper we propose a computationally efficient moving object detection algorithm based on moving edge detection using the Discrete Hadamard Transform (DHT) that is particularly suited to real-time surveillance scenarios.

Within the literature we can consider the following main approaches to object detection in surveillance scenarios: frame differencing, background subtraction, optical flow and pre-trained statistical approaches. Statistical training methods require many training samples and typically require significant computational complexity to obtain good immunity to noise (Wang and Suter, 2006). Optical flow methods allow accurate detection of moving objects that are free of ghosts, but they can have problems with illumination changes. Such approaches are often complex and not suitable for real-time systems, although there has been some work on achieving real-time operation (Jongcheol et al., 2005; Bruhn and Schnorr, 2006). The background subtraction method, sometimes referred to as background removal, is a popular approach, where each video frame is compared to a background model and significant deviations from the model are considered to be moving objects. Background models include recursive and non-recursive methods (classification terms suggested by Cheng and Kamth (Cheung and Kamath, 2004)). Background subtraction should be able to adapt to gradual or sudden illumination changes and local changes such as shadows. In order to handle non-stationary background objects, such as waving trees and image changes due to camera motion, complex algorithms are needed. Many background subtraction algorithms are surveyed in (Shireen et al., 2008). Frame differencing, sometimes referred to temporal differencing, is the most computationally efficient approach, but is prone to producing poor results. The difference frame is binarised by thresholding at some pre-determined value to obtain changed regions of the video. However, the appropriate threshold is dependent on the scene as well as illumination that may change over time. This means that the threshold value should be

calculated dynamically based on image content, so experimentally choosing a value is usually not appropriate for automatic systems.

Since the primary objective of our work is computational efficiency and ease of implementation for real-time systems, we target a frame differencing approach, but one based on image edges. Our approach does not critically depend on a particular threshold thereby avoiding some of the drawbacks of traditional frame differencing approaches. To achieve computational efficiency, we propose an edge detection algorithm based on the Discrete Hadamard Transform instead of conventional methods such as Sobel, Canny and Prewitt operations which often introduce significant complexity for background modeling (Shireen et al., 2008). Furthermore, our block-based approach lends itself to efficient memory usage and robust processing.

The remainder of the paper is organised as follows. Section 2 describes related work and the motivation for our approach. Section 3 describes the overall architecture as well as the details of the proposed method. Experimental results are presented in Section 4. Section 5 contains the conclusions.

2 Related Work

Frame differencing uses the video frame at time $t-1$ as the background model for the frame at time t. The unchanged part is eliminated in the difference image (difference map) with only the changed areas retained. Frame differencing is sensitive to noise and variations in illumination and fails to identify the interior pixels of a large, uniformly-colored moving object (Shireen et al., 2008; Radke et al., 2005). It is well known that object detection using single frame differencing, while computationally much simpler than other techniques, is more liable to generate large areas of false foregrounds known as ghosts (Archetti et al., 2006). An appropriate threshold value for localising the moving objects can be set empirically or adaptively. In the former case, the threshold value is fixed for all pixels in the frame and determined experimentally based on a large database (Durucan and Ebrahimi, 2000). In the latter case, the threshold value is adapted by pre-defined rules (Costantini et al., 2001; Otsu, 1979). To mitigate against the critical dependency on thresholds, Cavallaro proposed using a Sobel edge detector between the current and the reference image to leverage information edges that are immune to noise and are not changed by illumination variations (Cavallaro and Ebrahimi, 2001). Julius *et al.* also proposed edge based moving object detection for surveillance applications to obtain robustness to illumination changes (Julius et al., 2007). Their approach measured the difference between edge pixels to match and classify edges. Chaohui *et al.* also presented a moving object algorithm based on frame differencing and edge detection (Chaohui et al., 2007), and their approach is most similar to that proposed in this paper. In their proposed method, no threshold value is applied in the binarisation process but only used for counting non-zero pixels in the moving mask. However, their approach introduces computational complexity due to the presence of the following steps: (1) an image filter to reduce noise; (2) calculation of the amplitude and direction of the gradient of each pixel; (3) application of thinning to obtain edges one pixel in width; (4) application of the threshold (simple or complex).

3 The Proposed Approach

The proposed moving object detection method consists of three components, edge detection, moving edge detection and post-processing. An edge is detected by classifying coefficients of the DHT to avoid the computational burden of using conventional edge detection algorithms. The moving edges are extracted by frame differencing the edge map and calculating and tracking the transitions of edge patterns in the same 2×2 block. Finally, moving objects are obtained by a post-processing step that includes region marking and morphological operations.

3.1 Discrete Hadamard Transform

The two-dimensional Hadamard transform can be written in matrix and series form as;

$$F(u,v) = \frac{H_N[f(x,y)]H_N^t}{N} \quad (1)$$

$$= \frac{H_N[f(x,y)]H_N}{N} \quad (2)$$

$$= \frac{1}{N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)(-1)^{s(x,y,u,v)} \quad (3)$$

where $s(x,y,u,v) \equiv \sum_{i=0}^{n-1}[g_i(u)x_i + g_i(v)y_i]$. (x_i,y_i,u_i,v_i) is the binary representation of the (x,y,u,v), (x,y) and (u,v) are row and column position in pixel and DHT domain, and

$$\begin{aligned} g_0 &\equiv u_{n-1} \\ g_1 &\equiv u_{n-1} + u_{n-2} \\ g_2 &\equiv u_{n-2} + u_{n-3} \\ &\vdots \\ g_{n-1} &\equiv u_1 + u_0 \end{aligned}$$

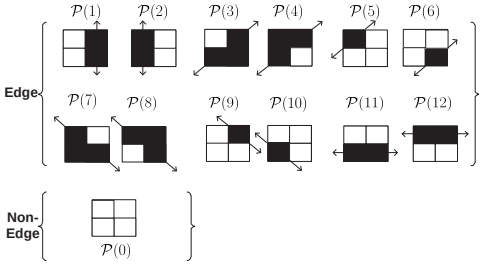

Figure 1: Edge classification patterns in a 2×2 pixel block; patterns are denoted as numbers according to edge direction (horizontal, vertical, diagonal and negative diagonal)

We use the ordered version of the DHT so that sequency is an increasing function of the number of rows (or columns). A conservation of energy property exists between the spatial domain and the Hadamard domain. It is useful as a criterion for classifying edges (as described later in Eq (5)). A 2×2 DHT is used to classify edge patterns in this paper.

3.2 Edge detection based on DHT coefficients

Edge patterns are assigned identifying numbers by classifying different patterns of pixels in a 2×2 block as shown in Figure 1. Let the coefficients of the Hadamard transform in a 2×2 image block be denoted as $F(u,v)$. We first calculate the following:

$$\chi = \max(|F(1,0)|, |F(0,1)|, |F(1,1)|) \quad (4)$$

$$\mathcal{D} = \begin{cases} \left[\frac{F(1,0)}{F(0,1)} + 0.5\right] & \text{if } F(0,1) \neq 0 \\ 2 & \text{otherwise} \end{cases} \quad (5)$$

where [] is a round function.

$$\mathcal{P}(i), i = \begin{cases} 0 & \text{if } (\chi \leq \tau) \\ 1 & \text{if } (\chi > \tau \bigcap \mathcal{D} = 2 \bigcap F(1,0) > 0) \\ 2 & \text{if } (\chi > \tau \bigcap \mathcal{D} = 2 \bigcap F(1,0) < 0) \\ 3 & \text{if } (\chi > \tau \bigcap \mathcal{D} = 1 \bigcap F(1,1) > 0 \bigcap F(1,0) > 0) \\ 4 & \text{if } (\chi > \tau \bigcap \mathcal{D} = 1 \bigcap F(1,1) < 0 \bigcap F(1,0) < 0) \\ 5 & \text{if } (\chi > \tau \bigcap \mathcal{D} = 1 \bigcap F(1,1) < 0 \bigcap F(1,0) < 0) \\ 6 & \text{if } (\chi > \tau \bigcap \mathcal{D} = 1 \bigcap F(1,1) < 0 \bigcap F(1,0) > 0) \\ 7 & \text{if } (\chi > \tau \bigcap \mathcal{D} = -1 \bigcap F(1,1) < 0 \bigcap F(1,0) < 0) \\ 8 & \text{if } (\chi > \tau \bigcap \mathcal{D} = -1 \bigcap F(1,1) > 0 \bigcap F(1,0) > 0) \\ 9 & \text{if } (\chi > \tau \bigcap \mathcal{D} = -1 \bigcap F(1,1) > 0 \bigcap F(1,0) > 0) \\ 10 & \text{if } (\chi > \tau \bigcap \mathcal{D} = -1 \bigcap F(1,1) > 0 \bigcap F(1,0) < 0) \\ 11 & \text{if } (\chi > \tau \bigcap \mathcal{D} = 0 \bigcap F(0,1) > 0) \\ 12 & \text{if } (\chi > \tau \bigcap \mathcal{D} = 0 \bigcap F(0,1) < 0) \end{cases} \quad (6)$$

For a non-edge region (assigned $\mathcal{P}(0)$ as shown in Figure 1), the non zero position coefficients $(F(1,0), F(0,1), F(1,1))$ should be zero. However, the non-zero position coefficients increase according to edge strength. Therefore if χ is greater than a pre-defined threshold value (τ), the block is classified as an edge block, otherwise as a non-edge block. We set this pre-defined noise threshold to ten in this paper. This means that when the difference of pixel values is greater than ten, we consider this block as an edge. Pattern $\mathcal{P}(i)$ is obtained via Eq (5) and the properties of the Hadamard transform, as shown in Eq (6), where τ is a pre-defined threshold value and $\bigcap$ is a logical AND operation. In Eq (6), $\mathcal{D}$ indicates the direction of the edge. $\mathcal{P}(5), \mathcal{P}(6)$ and $\mathcal{P}(9), \mathcal{P}(10)$ are the one-pixel shifted edges of $\mathcal{P}(3), \mathcal{P}(4)$ and $\mathcal{P}(7), \mathcal{P}(8)$ respectively.

Figure 2 illustrates a comparison of the result of edge detection using Canny edge detection (Canny, 1986) and the proposed method. The Canny operator has the feature of high precision localisation of a single edge, but when the background of the scene is complicated, too many edges will be detected. An accurate result requires that well-matched thresholding values be calculated, and this causes additional computational complexity. The proposed edge detection algorithm suffers sensitivity to noise and some edge lines are disconnected due to noise and block-based processing. Despite this, the proposed edge detection algorithm has two advantages over Canny edge detection: (1) it does not need to set critical threshold values and to perform a filtering operation (instead of using a filter, noise effects are eliminated via the moving object detection algorithm described in Section 3.3); (2) the block based approach provides for memory efficiency and low complexity.

3.3 Moving edge detection

After edge detection, the blocks corresponding to moving edges are determined. As an object moves in the scene it covers and uncovers background around its borders and possibly deforms. These phenomena result in a change of the edge characteristics within blocks on the object's boundary. This can be used to detect moving edge blocks from frame to frame. There are 3 possibilities to consider: (1) edge to non-edge (2) non-edge to edge (3) edge direction changing.

Within a frame differencing framework, the first possibility above will result in ghost edges that should be removed if they can be detected. In the ideal case, the other possibilities above will result in real moving edges but in practice there will be a lot of noise. To

Figure 2: (a) The 1500^{th} frame of the dataset3 camera1 sequence of PETS2001; (b) edge detection using the Canny detector with a high threshold of 190, a low threshold of 110 and a 3×3 Sobel mask; (c) our proposed edge detection approach.

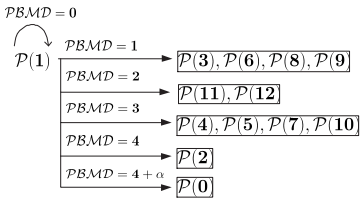

Figure 3: The example of $\mathcal{PBMD}$ between $\mathcal{P}(1)$ and other patterns

.

reduce the effect of noise, the history of the edge is examined and processed in our proposed method. We introduce the Pixel Bit Mask Difference ($\mathcal{PBMD}$) as an observation factor. $\mathcal{PBMD}$ is the number of different bits between assigned edge patterns. It is increased based on the strength of the noise that potentially results in a bit change. $\mathcal{PBMD}$ can be calculated as follows (where black and white pixels in Figure 1 are set to 0 and 1 respectively):

$$\mathcal{PBMD} = \sum_{(u,v)\in\{0,1\}} \mathcal{P}(i)^{(u,v)} \oplus \mathcal{P}(j)^{(u,v)} \quad (7)$$

where $\oplus$ is a XOR bit operation, $\mathcal{P}(i)^{(u,v)}, \mathcal{P}(j)^{(u,v)}$ are mask bits of $\mathcal{P}(i)$ and $\mathcal{P}(j)$. For example, $\mathcal{PBMD}$ is calculated as '1' (meaning there is one bit difference) between $\mathcal{P}(1)$ and $\mathcal{P}(3), \mathcal{P}(6), \mathcal{P}(8), \mathcal{P}(9)$ by comparing the pixel bit masks as shown in Figure 3. The $\mathcal{PBMD}$ between $\mathcal{P}(1)$ and $\mathcal{P}(0)$ (non-edge block) should be controlled in a different way by setting a maximum value of $\mathcal{PBMD}$ (i.e. by setting α to '1' as shown in Figure 3.

The differencing between edges in temporal sequences sometimes generates false alarms if edges vary in the background. This is an inevitable consequence of frame differencing methods. We introduce a compensation method for false alarms using the History Update Value ($\mathcal{HUV}$) based on $\mathcal{PBMD}$. $\mathcal{HUV}$ is observed at each frame and compared to the $\mathcal{PBMD}$ to classify moving edges from all possible

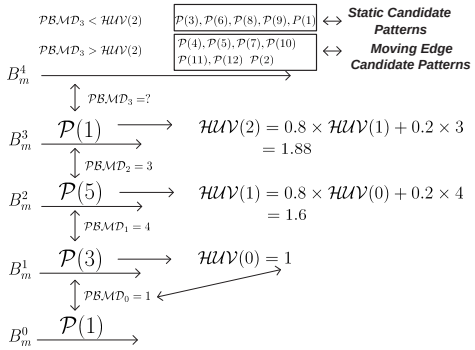

Figure 4: An example of detecting moving edges by observing the relationship between $\mathcal{PBMD}$ and $\mathcal{HUV}$, B_m^i is a 2×2 block m in the i^{th} frame. $\beta = 0.8$

candidate edges. When the current block has the same edge pattern from the past frame, the $\mathcal{HUV}$ is zero. This means that any edge pattern appearing in this block except the same edge pattern from the previous frame is considered as a moving edge. On the contrary, the higher number of $\mathcal{HUV}$ represents much noise existence from the past frames, only restricted edge patterns are decided as a moving edge. $\mathcal{HUV}$ can be obtained by using Eq (8). Figure 4 shows the process to detect moving edges using $\mathcal{PBMD}$ and $\mathcal{HUV}$ at β of 0.8. The 2x2 block is decided that has a moving edge by comparing $\mathcal{HUV}$ in the previous frame and the current $\mathcal{PBMD}$.

$$\mathcal{HUV}(i) = \beta \times \mathcal{HUV}(i-1) + (1-\beta) \times \mathcal{PBMD}(i)$$

$$\begin{cases} \mathcal{PBMD}(i+1) > \mathcal{HUV}(i) & \text{moving block} \\ \mathcal{PBMD}(i+1) \leq \mathcal{HUV}(i) & \text{non moving block} \end{cases} \quad (8)$$

where $\mathcal{PBMD}(i), \mathcal{HUV}(i)$ are the $\mathcal{PBMD}$ and $\mathcal{HUV}$ values of a block in the i^{th} frame and β is a weighted constant satisfying the condition $\beta \in [0,1]$.

In summary, the overall moving edge detection algorithm can be explained as follows.

1. Obtain average pixel differences between frames t and $t-1$ ($ZS^t = (F(0,0)^t - F(0,0)^{t-1})$). If ZS^t

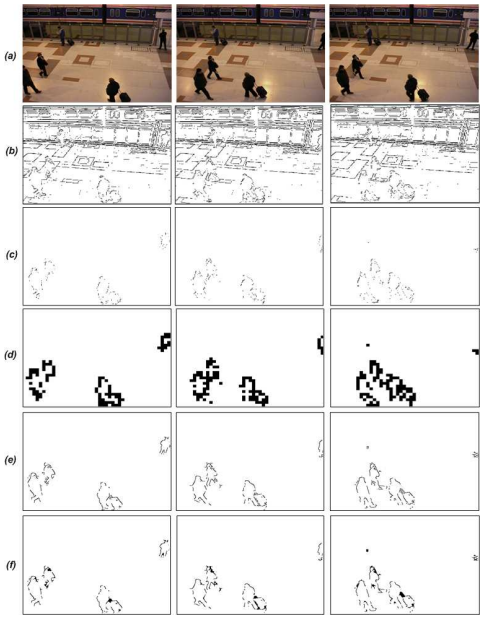

Figure 5: (a) 520^{th},530^{th},540^{th} frames of PETS2006 dataset; (b) edge map detected by the proposed method; (c) moving edge extraction result; (d) region marking based on moving edges; (e) AND operation between marked regions and edges using Canny operator; (f) after morphological closing.

is larger than a pre-defined value ($\tau^{\star}$), a block is considered a candidate moving block, otherwise non-moving block.

2. If $ZS^t > \tau^{\star}$, $\mathcal{PBMD}$ and $\mathcal{HUV}$ are calculated. If $\mathcal{PBMD}(t) > \mathcal{HUV}(t-1)$, this block is considered as a moving block, the value of the $\mathcal{HUV}$ is updated with Eq (8).
3. When moving edge blocks are obtained, moving edge pixels are marked as the whole 2×2. This has the effect of generating a sub-sampled moving edge binary image.

The entire process is illustrated in Figure 5(a)–(c).

3.4 Post-Processing

After separating moving edges, we need to locate the object so as to get the position and the shape of moving objects. We propose a simple three step procedure to extract location and shape. All post-processing is applied to the down sampled image obtained automatically, leading to very efficient processing.

1. A 4×4 window scans the whole image in raster scan order and finds a region containing moving edges. Figure 5(d) shows how the marked region area can contain missed moving edges.

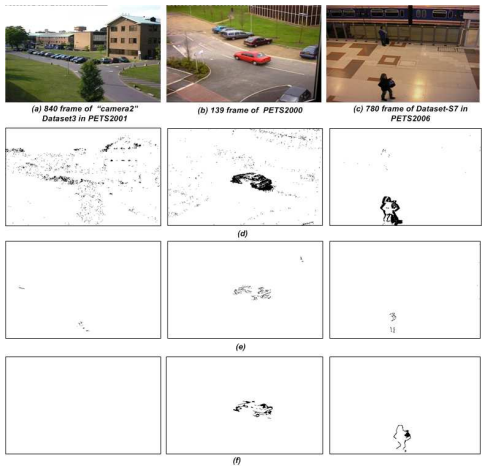

Figure 6: (a)(b) & (c) Sample test sequences used to illustrate performance for illumination changes, ghosting and fast moving objects; (d) results based on frame differencing, threshold ; (e) Chaohui's approach ; (f) the proposed method

2. There are many algorithms to fill a contour boundary such as connected components algorithms and neighbor searching. The proposed method uses an AND operation performed between a marked region and the edges in the original image obtained using a Canny operator. This does not generate significant computational complexity since it acts only on the marked region. The result is depicted in Figure 5(e).
3. The morphological closing operation is applied on the moving edge binary mask to fill small missing edges. Closing is able to eliminate salt and pepper noise, narrow cracks, small holes and fill the gaps in the contour. The closing of A by B, $A \bullet B = (A \oplus B) \ominus B$, is simple dilation of A by B followed by erosion of the result by B. Figure 5(f) shows the output image after morphological closing.

4 Experimental Results

Our experiments aimed at evaluating the moving object detection algorithm's ability to deal with illumination changes, in avoiding ghost effects and assessing its complexity. All processing is performed on the Y component of the YUV signal. All tests are performed on an Intel Core(TM)2 Duo 3.0GHz with 2GB RAM using Window XP version 2002 with service pack 2 written in ANSI C++. The popular and commonly available surveillance datasets,

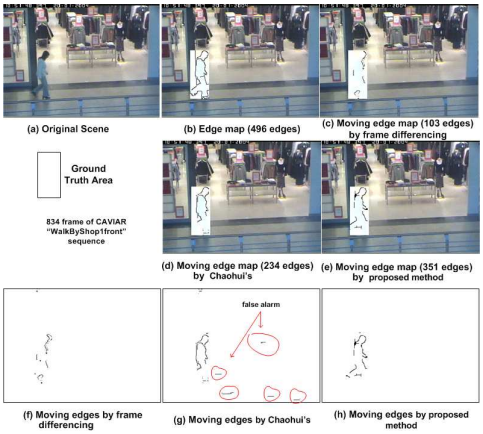

Figure 7: (a) Original scene (b) Edge map detected by the Canny detector in the ground truth; (c), (d) & (e) Moving edges detected by (Otsu, 1979), (Chaohui et al., 2007) and our method : recall = $\frac{103}{496}$, $\frac{234}{496}$, $\frac{351}{496}$ (f), (g) & (h)Detected moving edges in the entire frame for each approach.

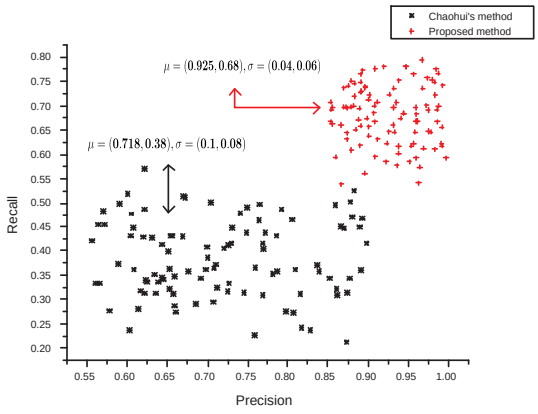

Figure 8: Recall and Precision for 100 frames of the CAVIAR sequence; μ is a average value and σ is a standard deviation

namely CAVIAR[1] and PETS[2], were used to evaluate the performance of the proposed method. The proposed method is compared with two frame differencing low complexity methods. The first uses adaptive thresholding based on observing histograms as described in (Otsu, 1979). The second is the edge-based frame differencing and thresholding approach of Chaohui (Chaohui et al., 2007). We qualitatively illustrate our approach in Figure 6 using 3 different scenarios. Figure 6(a) has no moving objects, and only features luminance changes. Figure 6(b) shows a moving object that has uniform luminance in its internal and boundary area, so that some edges do not appear. Figure 6(c) represents a fast moving object that generates the ghost effect. The simple frame differencing approach suffers under luminance changes and also introduces the ghost effect as seen in Figure 6(d). The edge based approach of Chaohui has good quality compared to frame differencing, but it also suffers false alarms due to variation of edges in successive frames, as shown in Figure 6(e). The proposed method gives good results with no false alarms since our approach uses not only frame differencing but also tracks the history of edges.

In order to have a quantitative evaluation of the performance, we use recall and precision, to quantify how well each algorithm matches the ground truth suggested by Cheung and Kamath (Cheung and Kamath, 2004). To adapt the quantitative method to the proposed method, we only compared moving edge pixels and not all object pixels. Our modified metric can be defined by Eq (9)(10).

$$\text{Recall} = \frac{\text{\# of moving edges detected in the ground truth}}{\text{\# of edge pixels in the ground truth}} \quad (9)$$

$$\text{Precision} = \frac{\text{\# of moving edges detected in the ground truth}}{\text{\# of moving edges detected in a whole frame}} \quad (10)$$

For clarity, the evaluation process is illustrated in Figure 7 for the different moving object detection algorithms considered. The ground truth region is highlighted Figure 7(b) and shows the edges generated by the Canny edge detector within this region. The frame differencing and thresholding method generates ghosts that produce multiple edges as shown in Figure 7(f). This points to a potential problem with our proposed Recall metric, whereby approaches prone to ghost effects will results in artificially high Recall values (due to the presence of more moving edges). For this reason, we exclude the simple frame differencing approach from the quantitative evaluation and focus only on our method and that of Chaohui. The calculated Recall is depicted in Figure 7(c)(d)(e). Figure 8 shows Precision and Recall graphs for both approaches for 100 frames of the CAVIAR sequence. The proposed method shows the high performance(large μ) and independence of image contents(small σ). The improvement of our approach over Chaohui's is mainly due to the lack of false alarms it produces – as is clearly illustrated in Figure 7(g) & (h).

Finally, we compare computational complexity with Chaohui's method. We use from the 830^{th} to the 1229^{th} frame (500 frames in total) of the "WalkByShop1front" sequence of CAVIAR in 384×288 resolution. The proposed method shows higher de-

[1] available at http://homepages.inf.ed.ac.uk/rbf/CAVIAR
[2] available at http://www.cvg.rdg.ac.uk/slides/pets.html

	Time (*sec*)
Frame differencing (Otsu, 1979)	*5.4*
Chaohui's method (Chaohui et al., 2007)	*2.4*
The proposed method	*1.1*

Table 1: Performance analysis of the computational complexity of different approaches. T is the total time(seconds) for 500 frames

tection speed than other fast detection algorithms as illustrated in Table 1. The improved detection speed is mainly as a result of the fact that the proposed method uses an efficient edge detection algorithm rather than other conventional approaches and the fact that post-processing is performed on the down-scaled image.

5 Conclusions

In this paper, we present edge based moving object detection using the Hadamard transform. The Hadamard transform is a computationally efficient tool because it consists of only subtractions and adders. We propose a moving object detection algorithm based on this new edge detection approach. The edge pattern of a 2×2 block is classified by observing the coefficients of the Hadamard transform. $\mathcal{PBMD}$ and $\mathcal{HUV}$ are defined in order to detect moving blocks by observing the history of edges. The block-based approach provides potential to save on memory usage and data processing.

The proposed method targets reducing complexity while addressing the main problems encountered by frame differencing such as illumination changes and ghost effects. We have followed this approach since we believe that edge features are potentially superior to other types of features – they are robust to sudden illumination changes and computationally efficient because edge information is stored in binary form. However, we acknowledge that the proposed edge detection algorithm suffers discontinuity of edge shape that mainly arises due to slow moving objects. This could potentially be recovered by using the U and V components based on a uniform colour assumption for objects, and we plan to investigate this in future work.

ACKNOWLEDGEMENTS

Authors would like to acknowledge the support of Samsung Electronics and Science Foundation Ireland under grant 07/CE/I1147.

REFERENCES

Archetti, F., Manfredotti, C. E., Messina, V., and Sorrenti, D. G. (2006). *Foreground-to-Ghost Discrimination in Single-Difference Pre-processing*. Advanced Concepts for Intelligent Vision Systems.

Bruhn, A. Weickert, J. K. T. and Schnorr, C. (2006). A multigrid platform for real-time motion computation with discontinuity-preserving variational methods. *International Journal of Computer Vision*, 70:255–277.

Canny, J. (1986). A computational approach to edge detection. *IEEE Trans Patten Analysis and Machine Intelligence*, 8:679–698.

Cavallaro, A. and Ebrahimi, T. (2001). Change detection based on color edges. In *Circuits and Systems, 2001. ISCAS 2001. The 2001 IEEE International Symposium on*, volume 2, pages 141–144, Sydney, NSW, Australia.

Chaohui, Z., Xiaohui, D., Shuoyu, X., Zheng, S., and Min, L. (2007). An improved moving object detection algorithm based on frame difference and edge detection. In *the Fourth International Conference on Image and Graphics*, pages 519–523.

Cheung, S. and Kamath, C. (2004). Robust techniques for background subtraction in urban traffic. In *Proc Elect Imaging : Visual Comm Image Proc*.

Costantini, R., Ramponi, G.and Bracamonte, J., Piller, B., Ansorge, M., and Pellandini, F. (2001). Countering illumination variations in a video surveillance environment. *SPIE proceedings*, 4304:85–97.

Durucan, E. and Ebrahimi, T. (2000). Robust and illumination invariant change detection based on linear dependence for surveillance applications. In *European signal processing conference*, pages 1041–1044, Tampere, Finland.

Jongcheol, K., Takumi, T., and Yasuo, S. (2005). Moving object detection using optical flow in mobile robot with an omnidirectional camera. *Nippon Robotto Gakkai Gakujutsu Koenkai Yokoshu*, 23:1B17.

Julius, H. M., Dewan, M., and Oksam, C. (2007). Moving object detection for real time video surveillance: An edge based approach. *IEICE Transactions on Communications*, E90-B:3654–3664.

Otsu, N. (1979). A threshold selection method from gray level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9:62–66.

Radke, R. J., Andra, S., Al-Kofahi, O., and Roysam, B. (2005). Image change detection algorithms: a systematic survey. *IEEE Transactions on Image Processing*, 14(3):294–307.

Shireen, Y., Elhabian Khaled, M., El-Sayed, and Sumaya, H. A. (2008). Moving object detection in spatial domain using background removal techniques - state-of-art. *Recent Patents on Computer Science*, 1:32–54.

Wang, H. and Suter, D. (2006). A novel robust statistical method for background initialization and visual surveillance. In *Asian conference on Computer Vision*, volume 3851/2006.