Hindawi Publishing Corporation
EURASIP Journal on Advances in Signal Processing
Volume 2008, Article ID 187413, 13 pages
doi:10.1155/2008/187413

Research Article

An Adaptive Motion Segmentation for Automated Video Surveillance

M. Ali Akber Dewan,[1] M. Julius Hossain,[2] and Oksam Chae[1]

[1] *Image Processing Lab, Department of Computer Engineering, Kyung Hee University, Yongin-si, Gyeonggi-do 446-701, South Korea*
[2] *Centre for Image Processing and Analysis, School of Electronic Engineering, Dublin City University, Glasnevin, Dublin 9, Ireland*

Correspondence should be addressed to Oksam Chae, oschae@khu.ac.kr

Received 4 April 2008; Revised 17 August 2008; Accepted 6 November 2008

Recommended by Dimitrios Tzovaras

This paper presents an adaptive motion segmentation algorithm utilizing spatiotemporal information of three most recent frames. The algorithm initially extracts the moving edges applying a novel flexible edge matching technique which makes use of a combined distance transformation image. Then watershed-based iterative algorithm is employed to segment the moving object region from the extracted moving edges. The challenges of existing three-frame-based methods include slow movement, edge localization error, minor movement of camera, and homogeneity of background and foreground region. The proposed method represents edges as segments and uses a flexible edge matching algorithm to deal with edge localization error and minor movement of camera. The combined distance transformation image works in favor of accumulating gradient information of overlapping region which effectively improves the sensitivity to slow movement. The segmentation algorithm uses watershed, gradient information of difference image, and extracted moving edges. It helps to segment moving object region with more accurate boundary even some part of the moving edges cannot be detected due to region homogeneity or other reasons during the detection step. Experimental results using different types of video sequences are presented to demonstrate the efficiency and accuracy of the proposed method.

1. INTRODUCTION

Automatic segmentation of moving object is the fundamental technique for analyzing image sequence in video surveillance, video conferencing, multimedia, or real-time imaging applications. Successful detection of motion helps to reduce the information redundancy in all of these applications and thus motion detection has become an active research issue. A lot of research works have been afforded in this area and it is still challenging. Details of existing research works can be found in [1]. One of the typical approaches for moving object detection is background subtraction where background modeling is an unavoidable part intending to accommodate the changes in the environment. However, most of the background-modeling approaches are complex and time consuming for real-time processing [2]. Moreover, these methods show poor performance due to lack of compensation with the dynamism of real environments [3]. Some researchers use edge information instead of intensity as edge shows more robustness against illumination variation and noise [4]. Nonetheless, it is difficult to keep track of the variations in the environment because of the poor representation of edges by the existing edge pixel-based methods.

To get rid of these challenges, some methods do not utilize any background; instead they make use of temporal information of consecutive frames [5, 6]. Because of region homogeneity and high sensitivity of pixel intensity to noise, these methods show poor performance in detecting moving object region properly. With compare to region-based methods, edge-based methods are robust, as the difference of consecutive frames produces significant difference only on the boundary region of moving object [2, 7, 8]. However, these methods treat each edge point independently without carrying shape and neighborhood information and thus it is not convenient for matching, segmentation, and

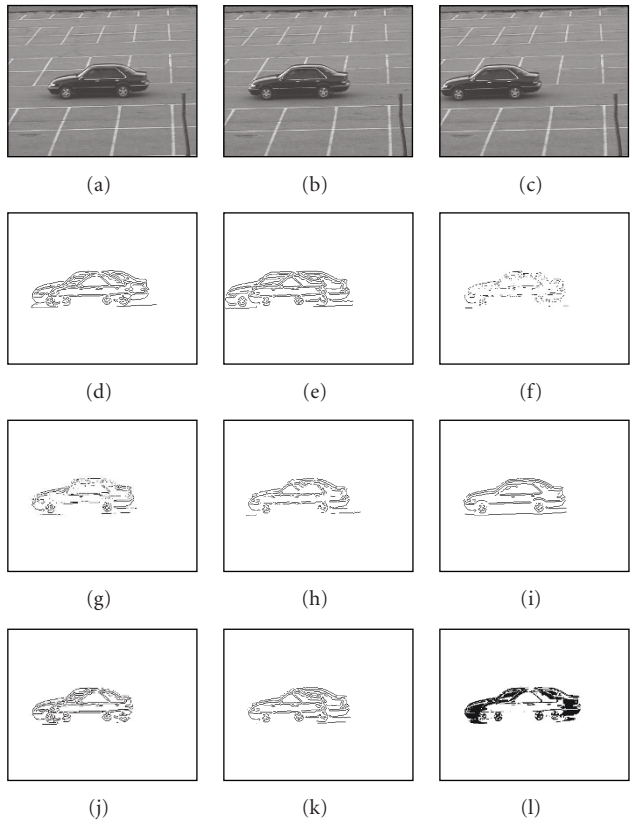

Figure 1: Illustration of problems in existing methods. (a) I_{n-1}; (b) I_n; (c) I_{n+1}; (d) left difference image edge map; (e) right difference image edge map; (f) scattered moving edges obtained from exact matching of left and right difference image edge maps; (g) moving edges obtained using more than three consecutive frames; (h) moving edges obtained allowing some distance threshold during matching; (i) edges expected to be extracted for moving object in I_n; (j) edges of left difference image edge map within the region of moving object in I_n; (k) edges of left and right difference image edge maps within the region of moving object in I_n; (l) missing of moving object region in three-frame based method due to region homogeneity.

tracking. Due to illumination variation, quantization error and speckle error, edge pixel may change its location in subsequent frames which is termed as edge-localization error. Hence, by pixel-wise matching, these methods result in scattered moving edges and show frequent failures in detecting moving object.

Figure 1 illustrates some of the limitations of existing methods. Figures 1(a)–1(c) show three successive frames: previous (I_{n-1}), current (I_n), and next (I_{n+1}). Left and right difference image edge maps computed from the difference images of these three frames are shown in Figures 1(d) and 1(e), respectively. Figure 1(f) shows the scattered moving edges obtained by exact matching among difference image edge maps. This deterioration is resulted from the positional variations of edge points in different frames. Some methods utilize more than three successive frames and accumulate moving edge information of I_n to improve the detection result [2] as shown in Figure 1(g). However, these methods require more computation and still suffer from scattered edges due to edge localization error. Figure 1(h) shows the moving edges, where matching between two pixels is considered if the distance is not greater than two. Still the detected moving edges are scattered and deviate from the ground truth of the moving object as shown in Figure 1(i). Existing three frame-based methods perform worse while detecting objects with slow movement in two successive frames. This is due to the insignificant gradient values in the respective edge location in overlapping region of the difference image. Moreover, edges in the overlapping region in difference image edge map deteriorate in size and shape than that in current image, I_n. So, it is difficult to recover the actual moving edges in I_n as shown in Figure 1(i) by matching left and right difference image edge maps. For the better visualization, the edges of left and right difference image edge maps within the region of moving object in I_n are shown in Figures 1(j) and 1(k), respectively. Figure 1(l) shows the missing of moving object region in the difference image obtained by subtracting two successive frames as shown in Figures 1(a) and 1(b), respectively. This problem occurs due to the region homogeneity which discourages region-based background-independent approaches to detect moving object.

Considering the above-mentioned problems, we extract moving edges from current image while removing the background edges by comparing with spatiotemporal edge information of three successive frames. We represent edges as segments, where all the edge pixels belonging to a segment are considered as a unit and processed together. A distance transformation-based flexible edge matching is proposed which is more robust than pixel-based matching. Segment-based representation of edges and flexible edge matching make the system efficient in terms of accuracy and time, and reduce the occurrence of scattered moving edges significantly. The proposed method is adaptive to the illumination variation as it uses most recent frames. A watershed-based algorithm is utilized which makes use of the extracted moving edges to segment moving object region with more accurate boundary. It ensures meaningful representation of moving objects, which is essential in video surveillance and many other image content-based applications in multimedia communication.

2. RELATED WORKS

A good number of research efforts have been reported in moving object detection during last few years. Background subtraction-based methods are the typical approaches for moving object detection because of its simplicity [9–11]. However, intensity of background pixel frequently changes due to object motion, illumination variation, or noise effect. To deal with these dynamisms, background subtraction-based methods need to incorporate automatic background estimation and update methods [3, 12, 13]. These methods usually utilize temporal change information or optical flow-based information to identify the appropriate pixel values in a time series for the background model, which are complex and time consuming for real-time processing. Some

optical flow-based approaches are used for moving object detection [14, 15]. For these methods, intensity changes are important cues for locating object movement in time and space. However, these methods may result false detection if the temporal changes are generated by noise or other external factors like illumination drift due to weather change. Moreover, these methods are computationally expensive. Methods that utilize temporal differencing [5, 6] use pixel-wise difference between two/three consecutive frames. Though these methods are very adaptive to the change of environment, it shows poor performance in extracting entire relevant feature pixels because of region homogeneity and excessive responsiveness of pixels to the noise.

In [2, 7, 16], authors propose edge-based methods for moving object detection utilizing double edge maps. In [16], one edge map is generated from difference image of background and I_n. Another edge map is generated from difference image of I_n and I_{n+1}. Finally, moving edge points are detected applying logical *OR* operation on these two edge maps. Due to illumination variation, random noise may vary in the background which may cause false detection of edges in the edge map. If any false edge appears in anyone of the edge maps, it is finally conveyed to the detection result because of applying logical *OR* operation on the edge maps.

In [7], one edge map is computed from the difference image of I_{n-1} and I_n, and another edge map is computed from I_n and I_{n+1}. Then the moving edges of I_n are extracted by applying logical *AND* operation on these two edge maps. However, due to random noise, edge pixel positions may be changed to some extent in consecutive frames. Moreover, edges located in the overlapping region of difference images are deteriorated due to insignificant gradient values of that region. Hence, exact matching like *AND* operation is not sufficient to extract accurate shape information of moving object. Moreover, pixel-based representation of edges is not suitable for flexible edge matching and tracking.

Some edge-based methods utilize more than three successive frames to accumulate the edge information of *n th* frame in the difference image. In [2], initially a coarse moving edge representation is computed from a given frame and two equidistant frames, and later on nondesired edges are removed by means of a filtering. Finally, iterative accumulation of detection result obtained with varying distance images is used in this method to strengthen the respective moving edge points of current image. However, this is time consuming and requires many preceding/succeeding frames in consideration which is not reasonable for real-time detection.

A pseudogradient-based moving edge extraction method is proposed in [17]. Though this method is computationally faster but its background is not efficient enough to take care of the situation when a new object arrives in the scene and stops its movement. In this stage, a stopped object is continuously detected as a moving object. As no background update is adopted in this method, it is not much robust against illumination change. Additionally, this method also suffers with scattered edge pixels of moving objects. The proposed method intends to address the drawbacks of the existing pixel-based methods by introducing an edge segment-based fast and robust solution for moving object detection.

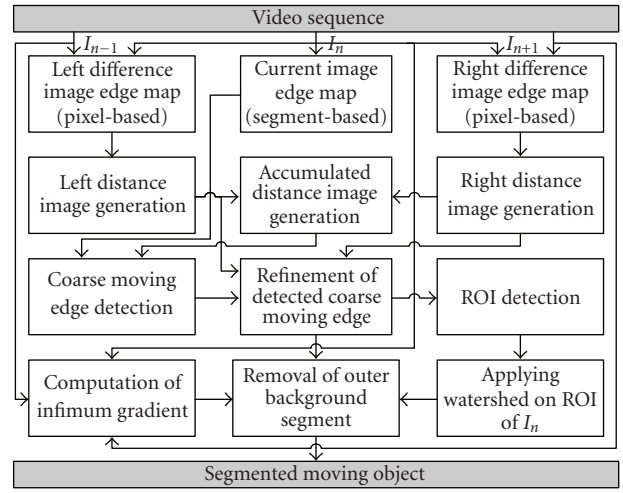

Figure 2: Flow diagram of the proposed method.

3. THE PROPOSED METHOD

3.1. Overview

Figure 2 illustrates the overall flow diagram of the proposed method. The proposed method makes use of three most recent frames I_{n-1}, I_n, and I_{n+1} for moving edge detection, and later on the detected edges are utilized for segmentation of moving object. Since this method does not require any background, it is free from complex and time-consuming background modeling technique. Moreover, it is adaptive to the change of environment because of using most recent frames.

In the first step of the proposed method, edges are extracted from current image and represent as segments to generate the current image edge map (E_n). In segment-based representation, all the edge pixels belonging to a segment are processed together instead of considering each of the edge pixels individually. It helps to take advantage of robust edge matching and shape information for moving edge detection. It also significantly reduces the occurrence of scattered edge pixels in the detection result.

To identify the moving edges from E_n, two edge maps: left difference image edge map $DE_{n-1,n}$ and right difference image edge map $DE_{n,n+1}$ are computed. $DE_{n-1,n}$ and $DE_{n,n+1}$ are utilized to generate distance transformation images $DT_{n-1,n}$ and $DT_{n,n+1}$, respectively. Distance transformation image contains the distance values from the nearest edge points of the respective edge map. It provides a linear progression of distances from edge points and is used for edge matching to detect moving edges. An accumulated distance transformation image DT_n is computed from $DT_{n-1,n}$ and $DT_{n,n+1}$. It contains the lowest distance value in the location of the difference image edge maps. This works as an accumulator of gradient values of two difference image

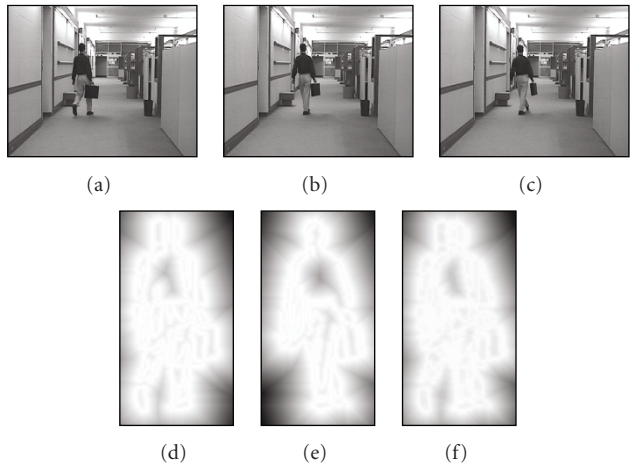

Figure 3: Illustration of advantage in using DT_n for matching. (a) I_{n-1}; (b) I_n; (c) I_{n+1}; (d) $DT_{n-1,n}$; (e) $DT_{n,n+1}$; (f) DT_n.

edge maps and reduces the loss of moving edge information on DT image due to overlapping of moving objects in the consecutive frames. It provides more information of moving edge location in current frame. It is to be noted here that minor movement of camera between successive frames is adjusted before obtaining the difference image, using the extracted edge information as described in our previous work [18, 19].

Figure 3 illustrates the advantage of using DT_n during matching. Figures 3(a)–3(c) show three consecutive frames, where Figures 3(d)–3(f) show $DT_{n-1,n}$, $DT_{n,n+1}$, and DT_n, respectively. For better visualization, we focus only on the moving object region in distance images and scale the inverted distance values in range $0 \sim 255$. Region with brighter portion of DT images represents more likelihood of containing moving edges. By accumulating $DT_{n-1,n}$ and $DT_{n,n+1}$, DT_n recovers gradient information on the overlapping region to some extent. It improves the accuracy of edge matching. It also helps to detect moving edges even in slow movement, where the existing three frame-based detection methods usually fail.

Moving edges are detected from E_n, applying edge segment-based matching by making use of DT_n. However, this process may detect some of the background edges as moving edge because of accumulating more gradient information in DT_n. These background edges are removed in the postprocessing step utilizing $DE_{n-1,n}$ and $DE_{n,n+1}$ with a variability test of matching confidence (DM). After detection, moving edges are grouped together, where each group represents a moving object [20]. Each group of moving edges is used to generate the region of interest (ROI). Watershed algorithm with reduced oversegmentation problem is applied on ROI of current image. Moving edges and gradient infimum values of two difference images are used by an iterative algorithm which removes background segments from ROI to obtain moving object regions. Since the proposed method is applied only on current image ROI, it is faster and applicable for real-time detection. The segmentation result is more accurate because of applying watershed algorithm and it is suitable for content-based applications, where motion information is important to increase the efficiency. The proposed method is described in detail in the following subsections.

3.2. Edge detection and representation as segment

Three edge maps: E_n, DE_{n-1}, and DE_{n+1} are utilized in the proposed method for moving edge detection. E_n is computed from I_n by using Canny edge detection algorithm. Two difference image edge maps, $DE_{n-1,n}$ and $DE_{n,n+1}$, are obtained utilizing I_{n-1}, I_n, and I_{n+1} as follows:

$$DE_{x,x+1} = \varphi(\nabla G^* \left| I_x - I_{x+1} \right|), \quad x \in \{n-1, n\}, \tag{1}$$

where φ, ∇, and G represent Canny edge detector, gradient operator, and Gaussian mask for noise filtering, respectively. Though we use fixed camera for moving object detection, minor displacement of camera frequently occurs in real application and it is adjusted using distance transformation-based translation with the help of edge segments. Camera adjustment procedure is described in detail in our previous work [18, 19]. Extracted edges from I_n are represented as segments to form E_n utilizing an efficiently designed edge class. In this representation, an edge segment consists of a number of neighboring consecutive edge pixels, where edge operations are performed on whole segment instead of individual edge pixel. Detail description of the edge class can be found in [18, 21]. This representation provides the shape information of an edge and allows local geometric operation. Segment-based representation of edges helps to incorporate an efficient and flexible edge-matching algorithm with higher accuracy and moderate computation time. Since we extract edges from I_n and apply segment-based flexible edge matching, detected moving edges preserve the shape information and missing of edge pixels is reduced significantly.

Figure 4 illustrates the robustness and suitability of using edge segment-based approach over edge pixel-based approach during matching. Figures 4(a) and 4(b) show two edge images of an object taken at different times. Due to edge localization error, there are some displacements of edge pixel position in these two different frames. As a result, pixel-based matching is not suitable in this situation and produces scattered edge pixels in the detection result which is shown in Figure 4(c). Value of disparity threshold (matching flexibility) was set to 1.5 for this illustration. However, in the case of segment-based representation, no edge pixels are missed as all the pixels belonging to a segment are processed together. Result of segment-based matching is shown in Figure 4(d). It is to be noted that about 17% of the edge pixels are missed in the case of pixel-based matching as compared to segment-based matching.

3.3. Moving edge detection

Moving edges are detected from E_n by eliminating background edges using $DE_{n-1,n}$ and $DE_{n,n+1}$. Equation (1) is used to compute $DE_{n-1,n}$ and $DE_{n,n+1}$ from the difference

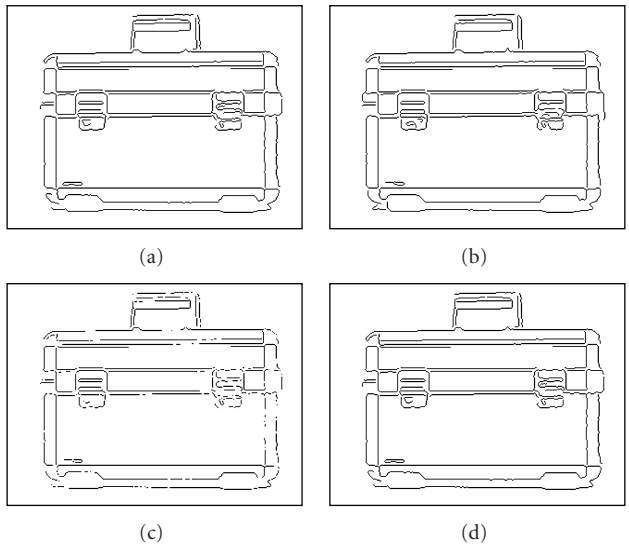

Figure 4: Advantages of segment-based matching. (a) Edge image of an object; (b) edge image of the same object at different time; (c) result obtained through pixel-based matching; (d) result obtained through segment-based matching.

images of consecutive frames. These two edge maps can also be computed by differencing edge points of E_{n-1}, E_n, and E_{n+1}. However, edge differencing approach is more noise prone as random noise in one frame is different from that of the successive frame [16]. Hence, (1) is utilized to generate difference image edge maps instead of utilizing edge differencing approach. Still, shape of edges on the overlapping regions in $DE_{n-1,n}$ and $DE_{n,n+1}$ changes to some extent than that in E_n. Thus, existing three frame-based methods that match two difference image edge maps fail to extract accurate shape information of moving object. To solve this problem, we detect moving edges from E_n by making use of distance transformation images of $DE_{n-1,n}$ and $DE_{n,n+1}$, instead of comparing these two difference image edge maps directly. Since distance transformation image provides a linear progression of distances from the edge points, the combined distance transformation image provides better information to extract moving edges using E_n. Moreover, segment-based representation of edges and flexible edge matching increase the robustness in terms of accuracy and computational speed. Figure 5 illustrates some of the intermediate steps of moving edge detection process. Figures 5(a)–5(c) show I_{n-1}, I_n, and I_{n+1}, where E_n, $DE_{n-1,n}$, and $DE_{n,n+1}$ are shown in Figures 5(d)–5(f), respectively. Distance transformation and edge matching procedure utilizing E_n, $DE_{n-1,n}$, and $DE_{n,n+1}$ are described as follows.

3.3.1. Distance transformation

During edge matching, one edge map is converted to distance transformation image, DT and another edge map is overlaid on it to compute disparity in matching, DM for each of the edge segments. In DT, each pixel contains the distance to the nearest edge pixels. Since the true Euclidean is resource demanding in terms of time and memory, therefore an integer approximation (3/4 Chamfer [22]) is used. Thus, distance image can be generated in linear time without considering any floating point operation.

The basic idea behind DT image generation is that global distances in the image are approximated by propagating local distances. This transformation is performed in three steps. In the first step, all the edge pixels are initialized with zero and other pixels are initialized with high value. Second stage is accomplished with a forward pass that scans the image from left to right and top to bottom and update the distances as follows:

$$\begin{aligned} DT[i,j] = \min\big(&DT[i-1][j-1]+4,\ DT[i-1][j]+3,\\ &DT[i-1][j+1]+4, DT[i][j-1]+3,\\ &DT[i][j]\big). \end{aligned} \tag{2}$$

Finally, a backward pass scans the image from right to left and bottom to top and modifies the DT image as follows:

$$\begin{aligned} DT[i,j] = \min\big(&DT[i+1][j+1]+4,\ DT[i+1][j]+3,\\ &DT[i+1][j-1]+4, DT[i][j+1]+3,\\ &DT[i][j]\big). \end{aligned} \tag{3}$$

Since, this algorithm uses only two passes to generate DT image, it is faster and can be computed in linear time. Using this algorithm, $DT_{n-1,n}$ and $DT_{n,n+1}$ are computed from $DE_{n-1,n}$ and $DE_{n,n+1}$, respectively. However, insignificant gradient value in the overlapping region may result failure of detecting moving edges in $DE_{n-1,n}$ and $DE_{n,n+1}$. Hence, to have more information in the distance image, we compute an accumulated distance image DT_n as depicted in the following equation:

$$DT_n = \min\left(DT_{n-1,n}, DT_{n,n+1}\right). \tag{4}$$

3.3.2. Computation of disparity in matching

$DT_{n-1,n}$ contains moving edge information of I_{n-1} and I_n, as it is computed from the difference image of these two frames. Similarly, $DT_{n,n+1}$ contains moving edge information of I_n and I_{n+1}. Therefore, DT_n contains moving edge information of I_{n-1}, I_n, and I_{n+1}. Moving edges in I_n are detected by making use of these three DT images, where DT_n is used first to detect the coarse moving edge list, and later on $DT_{n-1,n}$ and $DT_{n,n+1}$ are used for noise filtering.

During matching, edge segments in E_n are overlaid on DT_n and respective distance values are accumulated. DM for an edge segment is computed by taking a normalized average of distance values in DT_n that are hit by the edge segments of E_n, shown as follows:

$$DM[l] = \frac{1}{3}\sqrt{\frac{1}{k}\sum_{i=1}^{k}\{\mathrm{dist}(l_i)\}^2}, \tag{5}$$

where k is the number of edge points in the lth edge segment of E_n and $\mathrm{dist}(l_i)$ is the distance value at ith edge

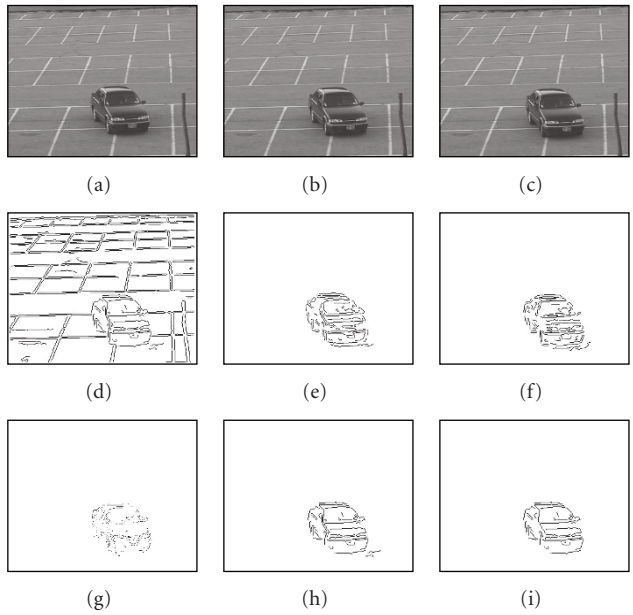

Figure 5: Illustration of the proposed moving edge detection method. (a) I_{n-1}; (b) I_n; (c) I_{n+1}; (d) E_n; (e) $DE_{n-1,n}$; (f) $DE_{n,n+1}$; (g) moving edges obtained from pixel-based matching of $DE_{n-1,n}$ and $DE_{n,n+1}$; (h) coarse moving edge list; (i) moving edges after noise filtering.

point of lth edge segment. A segment is aligned with the distance transformation image by translation in a small search window. As we represent edges as segments, this translation is performed easily and very fast by just adding the necessary displacement [18] with each of the edge points. The translation which results in the lowest *DM* value is finally selected.

During matching, two similar edge segments produce a lower disparity value *DM*. An edge segment l in E_n with distance image DT_n having $DM[l] \leq \tau$ is considered as moving edge and enlisted in the coarse moving edge list (*CMEL*). Here, τ is the disparity threshold and we use it to allow some flexibility during matching. Flexibility is allowed as edges change their location in different frames due to noise, illumination variation, and quantization error. Edges in the overlapping region of difference image also experience size, shape, and positional variations than that in current frame (as shown in Figure 1). Moreover, if the object movement is not very significant, low gradient values in the overlapping region cause missing of some moving object pixels in the difference image edge maps. We make use of DT_n along with τ to handle this variation of edges during matching. However, selection of a very high threshold value for τ might allow an edge segment E_n to be matched with a different edge in the difference image edge maps and thereby increases false positive in the detection result. On the other hand, selection of a very low threshold value might miss some of the moving edge segments to be matched with and thereby increases false negative. In our implementation, we set $\tau = 1.5$ empirically for all datasets as it gives comparatively better result in most of the cases.

3.3.3. Noise filtering

CMEL enlists the edge segments of E_n that have higher possibility of being moving edges in I_n. However, some background edges of I_n may erroneously enlisted in *CMEL* due to excessive incorporation of moving edge information in DT_n. Hence, further filtering procedure is applied to make the detection result more accurate. We perform a variability test for each of the edge segments enlisted in *CMEL* for the final classification as moving edge. Steps of variability test for noise filtering are as follows.

(i) Select an unclassified edge segment from *CMEL*.

(ii) Use (5) to compute disparity of matching $DM_{n-1,n}$ and $DM_{n,n+1}$ utilizing $DT_{n-1,n}$ and $DT_{n,n+1}$, separately.

(iii) If $(|DM_{n-1,n} - DM_{n,n+1}|) > \sigma$, then it is discarded from *CMEL*. Here, σ is a threshold to allow some flexibility.

(iv) Repeat steps (i) to (iii) until all the edge segments of *CMEL* are considered.

In noise filtering, our intension is to observe the variation of $DM_{n-1,n}$ and $DM_{n,n+1}$ for each of the edge segments in *CMEL*. Since moving edges of current image also exist in both $DT_{n-1,n}$ and $DT_{n,n+1}$, absolute difference of their matching confidence value is expected to be zero in the ideal situation. However, due to noise and edge localization error, some flexibility is needed and hence we use σ. But selection of high value for σ might causes false matching of edges whereas very low threshold might cause missing of true moving edges in the final detection result. Considering the above issues, we set $\sigma = 1.5$ empirically in our experiment. Figure 5(h) shows the edges initially enlisted in *CMEL*, where the final detection result after noise filtering is shown in Figure 5(i). To show the advantages of segment-based matching, we also present the result of pixel-based matching in Figure 5(g), where most of the moving edge pixels are missed due to edge-localization error.

3.4. Segmentation of moving object

In segmentation, moving regions are extracted from moving edges by using a watershed-based iterative background removal technique. Detected moving edges do not provide the complete boundary of moving object. Thus, a separate algorithm is required for segmentation. The segmentation algorithm is applied on *ROI* of I_n and it makes use of moving edge segments and gradient infimum values of difference image. Moving object segmentation procedure is described in the following subsections.

3.4.1. ROI detection and segmentation

Rectangular bounding box of moving edges is used to determine the *ROI* of moving object for segmentation. Figure 6(a) shows the detected moving edges, where the defined *ROI* using these edges is shown Figure 6(b). Use

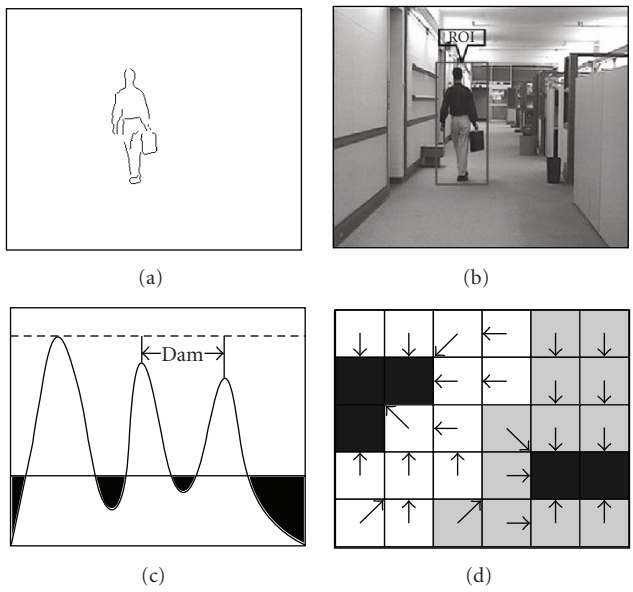

Figure 6: *ROI* detection and segmentation. (a) Moving edges; (b) *ROI*; (c) gray-level topographic surface; (d) flooding process and dam building.

of watershed only on *ROI* helps to reduce processing time significantly. Since watershed has been proven to be very useful and efficient for image segmentation [23], it extracts more accurate moving object region during segmentation.

In watershed algorithm, image is split into areas known as catchment basins based on the topology of the image. Catchment basin is defined as the region over which all points flow "downhill" to a common point as shown in Figure 6(c). The local minima (black regions) and maxima (dotted line) of the gray-level data yield catchment basins and watershed lines (dam), respectively. An efficient watershed transformation is flooding process, where only the dams emerge, defining the watershed of the image [23] as shown in Figure 6(d). However, watershed segmentation frequently results oversegmented image with hundreds or thousands of catchment basins; each corresponds to a minimum of the gradient, some of which may be due to small variations caused by noise. Considering the accuracy and efficiency, we have adopted Vincent-Soilly watershed algorithm [23] in our proposed method for segmentation. To solve the oversegmentation problem, we replace the gradient value by zero, where it is less than a particular threshold, T_1. T_1 is determined by mean of the gradient image minus one fourth of its standard deviation. Thus, around fifty percent [24] of the gradient values are replaced with zeros, which reduces the oversegmentation problem significantly.

3.4.2. Computation of gradient infimum value

Gradient infimum value is computed from the gradient values of difference images of consecutive frames. It is utilized for making decision on watershed segments for classification as foreground or background. Left gradient image ($\nabla_{n-1,n}$) and right gradient image ($\nabla_{n,n+1}$) are computed using the following equation:

$$\nabla_{x,x+1} = \nabla G^* \left| I_x - I_{x+1} \right|, \quad x \in \{n-1, n\}. \tag{6}$$

Due to the above formulation, high gradient values in $\nabla_{n-1,n}$ exist only in the region, where moving object boundaries exist in I_{n-1} and I_n. Similarly, high gradient values in $\nabla_{n,n+1}$ exist on the boundaries of moving object of I_n and I_{n+1}. Hence, to obtain high gradient values only on moving object boundary region in I_n, gradient infimum values, ∇_{inf} is computed from $\nabla_{n-1,n}$ and $\nabla_{n,n+1}$. To achieve more robustness, pixels including their eight neighbors are considered while computing ∇_{inf} as shown in the following equation:

$$\nabla_{inf}[i][j] = \max_{i-1 \Leftarrow k \Leftarrow i+1,\ j-1 \Leftarrow l \Leftarrow j+1} \min\left(\nabla_{n-1,n}[k][l], \nabla_{n+1,n}[k][l]\right). \tag{7}$$

3.4.3. Iterative procedure for background removal

In background removal technique, it is tried to remove the segments adjacent to the outer boundary of selected region in every step if it is identified as background. At first iteration, segments adjacent to the outer boundary of *ROI* are selected for consideration. If the common boundary portion of the selected segment and the outer boundary belongs to a moving edge, the segment is marked as foreground. Otherwise, gradient values in the position of boundary pixels of selected segments are checked from ∇_{inf}. If high gradient values (greater than or equal to threshold, T_2) exist in more than $NB * \mu$ pixels, the segment is marked as moving object segment. Here, NB is the number of boundary pixels in a segment and μ is an adjusting parameter. In ideal situation, high gradient values are expected to exist in the regions of ∇_{inf}, where boundary of moving object region exists. Due to noise, illumination variation and existence of low contrast between foreground and background regions, high gradient values do not exist in many boundary pixel positions of moving object region. Moreover, insignificant interframe displacement of moving object in the consecutive frames also reduces high gradient information on the boundary region. Hence, to allow some flexibility during segment boundary matching, we set $\mu = 0.75$ empirically.

If the boundary pixels of the selected segment do not satisfy the above condition, it is considered as background segment. At the end of the first iteration, all the background segments neighboring to *ROI* are removed and the outer boundary is updated as well. In the following iterations, adjacent segments of updated outer boundary are selected and classified as foreground or background similarly. However, the segments classified as foreground in the previous step are not taken into consideration. This iterative process continues until no further outer regions are classified as foreground. In this stage, the remaining segments represent regions of moving object. In the case of computing the value of T_2, we utilize the same procedure as like T_1. The convergence of the algorithm depends on the amount of background segments

presented inside bounding box after applying watershed segmentation. The background segment removal procedure is done as follows.

(i) All the segments are initialized as unmarked and outer boundary pixels of *ROI* are enlisted as outer boundary list, L_{OB}.

(ii) Segments neighboring to outer boundary and not marked yet are enlisted in current segment list, L_{CS}.

(iii) A segment is selected from L_{CS} for marking. If the common boundary portion of the selected segment and L_{OB} belongs to a moving edge, the segment is marked as foreground. Else all its boundary positions are checked in ∇_{inf}. If more than $NB * \mu$ pixels in ∇_{inf} contain high gradient values, the segment is marked as foreground. Otherwise, the segment is marked as background.

(iv) All the segments marked as background are removed. L_{OB} is updated by removing the portion common to the boundary of the removed background segment and including the rest of the boundary of removed segment. L_{CS} is updated accordingly.

(v) Stop the process and constitute moving object from remaining segments if L_{OB} is not updated any more in step (iv). Repeat step (ii) to step (iv) for all the segments in L_{CS}.

Figure 7 illustrates the steps of the proposed segmentation method. Figure 7(a) shows the segmented *ROI* of current image, where Figure 7(b) shows only watershed lines. Figure 7(c) shows ∇_{inf}, where high gradient values exist on the boundary region of moving object. Initially, L_{ob} contains the pixels of bounding box of *ROI* and is shown in Figure 7(d). The shaded segments neighboring to L_{ob} in Figure 7(e) are selected for L_{CS} in the first iteration. The segments belonging to white region are marked as background and thus removed at the end of first iteration, depicted in Figure 7(f). Updated L_{ob} is shown in Figure 7(g) and thereby, its neighboring segments are enlisted in L_{cs} for consideration in second iteration. Figures 7(h) and 7(i) show the regions enlisted in L_{cs} and the segmentation result in second iteration. Figures 7(j) and 7(k) show the updated outer boundary and selected segments neighboring to L_{ob} of third iteration, respectively. Figure 7(l) shows the segmentation result obtained in the final iteration. From the result, it can be noticed that watershed algorithm is effective to extract the complete and more accurate boundary of moving object.

4. RESULTS AND ANALYSIS

Experiments were carried out with several video sequences captured from indoor as well as outdoor environment to verify the effectiveness of the proposed method. We utilized a system with Intel Pentium IV 1.5 GHz processor and 512 MB of RAM. Visual C++ 6.0 and Multimedia Technology for Educational System (MTES [25]) were used as of our working environment tools. The above system can process

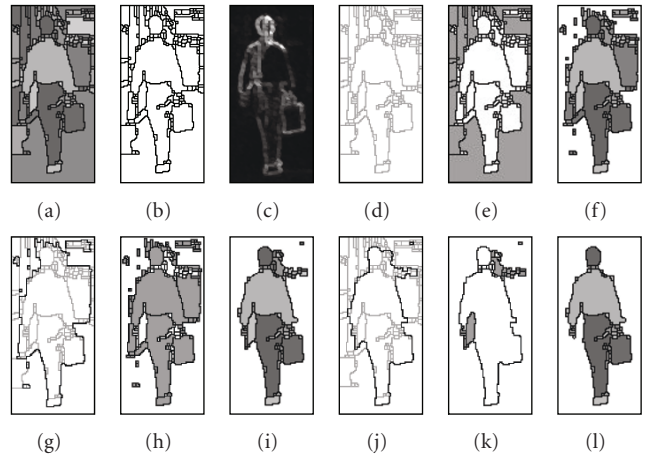

Figure 7: Segmentation of moving object from moving edges.

7 frames/second if the frame size is 320×260. Various image sequences are utilized to investigate the performance of the proposed method in different situations. Obtained results are evaluated in two ways: subjective evaluation and quantitative evaluation. In the case of subjective evaluation, detected moving edges and segmented moving object are visualized and compared with the results obtained by other related edge-based methods. As for the quantitative evaluation, we analyze the accuracy of the detected moving edge points and these data are also compared with other methods to investigate the robustness of the proposed method.

4.1. Subjective evaluation

Figure 8 shows the detection results obtained by the proposed method in "Hall Monitor" sequence and a comparison with two standard reference-independent moving edge detection methods. Figures 8(a)–8(c) show three consecutive frames, I_{46}, I_{47}, and I_{48}, respectively. Figure 8(d) shows the detection result using the method proposed by Dailey et al. [7] (*DC*), which uses frame differences of three consecutive frames followed by an *AND* operation for moving edge detection. However, in the difference image edge maps, edges change their shape (deteriorated) to some extent due to extracting edges from the difference image. Moreover, illumination variation and noise also cause edge localization error. Hence, exact matching like *AND* operation fails to produce better result in real scenario.

Sappa and Dornaika [2] (*SD*) tries to solve the problem by considering a combination of m frame pairs equidistant from current frame. Figure 8(e) depicts the result obtained by this method, where value of m is 2. This iterative solution improves the result in some extent but increases the processing time significantly due to the usage of more future and preceding frames. This method still results in scattered moving edges because of pixel-based processing. The proposed method does not suffer with the problem as edges are extracted as segments from current image and flexible matching is used to obtain the moving edges.

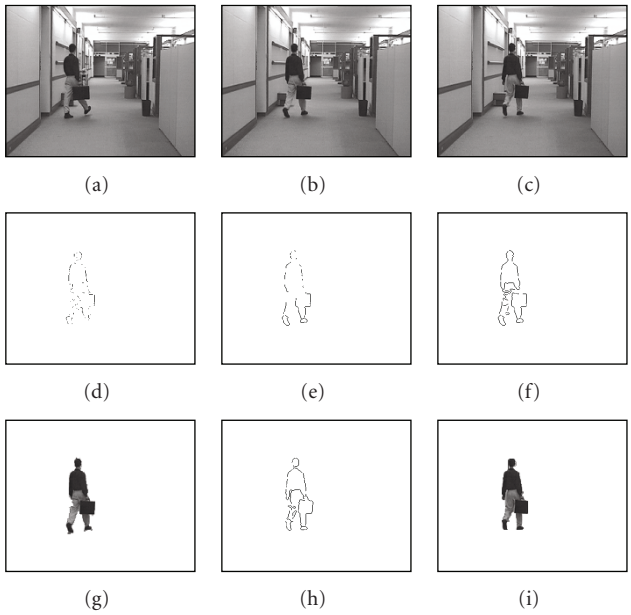

Figure 8: Detection and segmentation of moving object in "Hall Monitor" sequence. (a) I_{46}; (b) I_{47}; (c) I_{48}; (d) detected moving edges of I_{47} by *DC*; (e) detected moving edges of I_{47} by *SD*; (f) detected moving edges of I_{47} by the proposed method; (g) segmented moving edges by of I_{47} the proposed method; (h) moving edges of I_{48} by the proposed method; (i) segmented moving object of I_{48} by the proposed method.

Figure 8(f) shows moving edge detection result using proposed method whereas Figure 8(g) shows the segmentation result. Figures 8(h) and 8(i) show the moving edge detection and segmentation result of I_{48} by the proposed method.

Figure 9 illustrates the performance of the proposed method with the change of illumination. Figure 9(a) shows the background frame. Figures 9(b) and 9(c) show I_{52} and I_{53} in different illumination conditions (more bright). Figure 9(d) shows the result obtained by the method proposed by Kim and Hwang [16] (*KW*), where many background edge pixels are also detected as moving edges. This is due to the inefficiency in updating background to adapt with the illumination changes in *KW*. However, the proposed method works with the most recent frames. Thus, it is capable of adapting with the illumination change without any requirement of background update. Figures 9(e) and 9(f) show the moving edge detection and segmentation result, respectively, by the proposed method.

Figure 10 shows that the proposed method is robust against slight movement of camera. Figures 10(a)–10(c) show three consecutive frames: I_{255}, I_{256}, and I_{257}, respectively. Frames I_{255}, I_{256}, and I_{257} have movement of 2, 3, and 4 pixels with respect to the background along the upper left direction. Thus, each pair of consecutive frames has movement of 1 pixel. Figure 10(d) is the frame I_{257} having similar movement of frame I_{255}. These displacements were manually adapted to illustrate the robustness of the proposed method. Figure 10(e) shows the result obtained by *DC*. It is noticeable that many background edge pixels

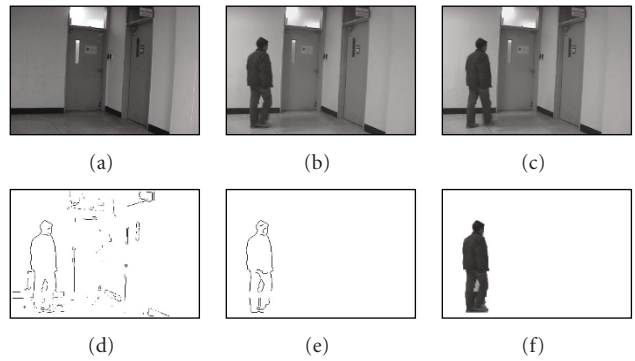

Figure 9: Performance with the change of illumination. (a) Background; (b) I_{52}; (c) I_{53}; (d) detected moving edges of I_{53} by *KW*; (e) detected moving edges of I_{53} by the proposed method; (f) segmented moving object by the proposed method.

are detected as foreground. Due to camera movement, background edge pixels of one frame cannot cancel out that of other frame during difference image edge map generation. The result is even worse when previous (Figure 10(a)) and next (Figure 10(d)) frames have similar movement with respect to the current frame. In this case, *AND* operation induces most of the background pixels in the detection result. This result is shown in Figure 10(f). The result obtained by *KW* is shown in Figure 10(g). Due to camera movement, background edge pixels cannot cancel out the background edge pixels in current frame. Thus, difference image edge map contains some of the background edge pixels which cause false detection in the final detection result. However, our method overcomes this problem as we align different successive frames before obtaining difference image and apply a flexible matching for each of the edge segments of current image which can tolerate the minor movement of camera in video sequence. The result obtained by the proposed method is shown in Figure 10(h), where Figure 10(i) shows the segmentation result of moving object.

Figure 11 presents experimental results for moving edge detection and segmentation of moving objects from "Hall Monitor" and "Highway" sequences to illustrate the comprehensiveness of the proposed method in dynamic environment. Figures 11(a)–11(f) illustrate the result obtained from "Hall Monitor" sequence, where background is much cluttered and high level of noise is present. Figures 11(a) and 11(d) show I_{133} and I_{210}, respectively, of "Hall Monitor" sequence. Figures 11(b) and 11(e) show the corresponding moving edge detection results, where segmentation results are shown in Figures 11(c) and 11(f), respectively. Figures 11(g)–11(l) show the detection results for "Highway" sequence. Figures 11(g) and 11(j) show I_{36} and I_{61} whereas Figures 11(h) and 11(k) show the corresponding moving edge detection result. Figures 11(i) and 11(l) show the final segmentation result, respectively. It is to be noted that in Figure 11(j) top right and top left vehicles are missed to be detected. "Highway" sequence is challenging with background movement and cluttered scene. Moreover, interframe displacement of some cars on top of the images is

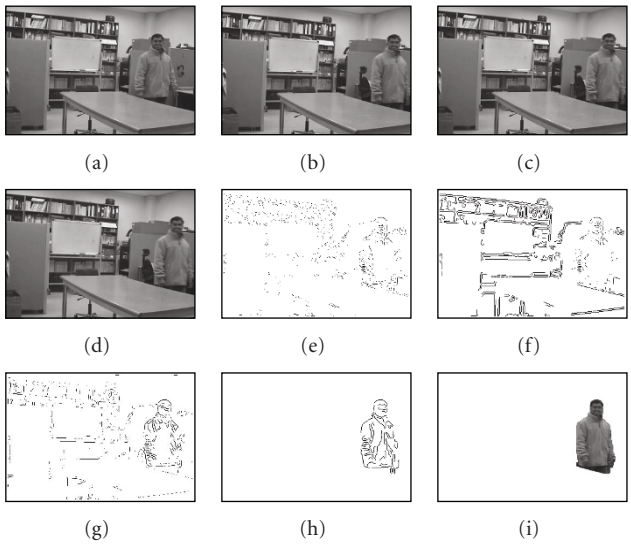

Figure 10: Results illustrating the performance in camera movement. (a)–(c) Frames I_{255}, I_{256} and I_{257} with different camera movement, respectively; (d) frame I_{257} having similar movement with frame I_{255}; (e) result obtained by *DC* using frames in figures (a), (b), and (c); (f) result obtained by *DC* using frames in figures (a), (b), and (d); (g) result obtained by *KW*; (h) result obtained by the proposed method; (i) segmentation result.

very insignificant, which results in less gradient information in the difference image edge maps to detect moving edges. As a result, some of the moving edges are missed to be detected, which eventually results in the missing of moving objects.

Figure 12 illustrates the segmentation result of the proposed method to comprehend its robustness even in the absence of some moving edge pixels in the detection result. Segmentation result of the proposed method is compared with the result obtained from *VOP* extraction method proposed by *KW*. In *KW*, moving object regions are segmented out by horizontal and vertical scanning followed by morphological operation. Figure 12(a) shows the edge detection result by *KW*. Since moving edges form almost complete boundary of the moving object, *VOP* extraction method with the help of morphological closing (with 9×9 structuring element) extracts moving object region effectively as shown in Figure 12(b). In such situation, the proposed method also works well as shown in Figures 12(c) and 12(d), respectively.

However, due to the presence of low contrast between foreground and background in the scene or in presence of illumination variation or noise, moving edge detection result may be deteriorated which may eventually degrade the segmentation result as well in *KW*. Figures 12(e) and 12(i) show moving edge detection result for two different experiments by *KW*, where moving edges do not form complete boundary. As a result, horizontal and vertical scanning-based*VOP* extraction method fails to extract moving object region properly as shown in Figures 12(f) and 12(j). Segmentation result of this method is largely dependent on extracted moving edges and the size of the morphological operator. Figures 12(h) and 12(l) illustrate the moving

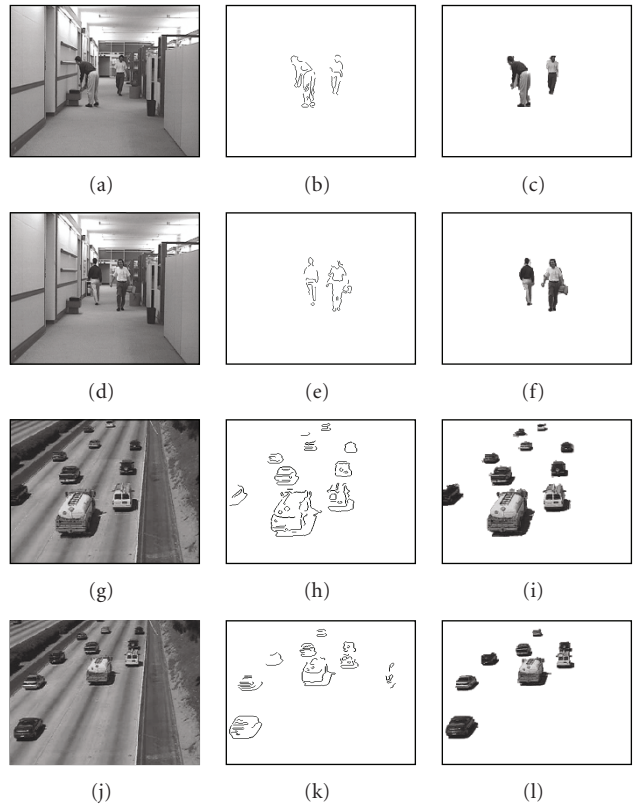

Figure 11: Results obtained from "Hall Monitor" and "Highway" video sequences by the proposed method. (a) I_{133} "Hall Monitor"; (b) edge image of I_{133}; (c) segmented moving object of I_{133}; (d) I_{210} "Hall Monitor"; (e) edge image of I_{210}; (f) segmented moving objects of I_{210}; (g) I_{36} of "Highway"; (h) edge image of I_{36}; (i) segmented moving object of I_{36}; (j) I_{61} of "Highway"; (k) edge image of I_{61}; (l) segmented moving objects of I_{61}.

object segmentation result by the proposed method whereas respective moving edges are shown in Figures 12(g) and 12(k). In case of moving edge detection result, some of the edges were missed as well. Since, we utilize watershed segments of current image *ROI* with the gradient infimum value instead of relying only on moving edges, the proposed method segment out the moving object region properly even in such challenging environment.

4.2. Quantitative evaluation

As for the quantitative evaluation, the accuracy of detected moving edges is determined, where ground truth is obtained by extracting moving edges manually. This evaluation is done using two criteria: precision and recall, defined in the following equations:

$$\text{Precision} = \frac{\text{Actual moving edge pixels extracted}}{\text{Total number of edge pixels extracted}} \tag{8}$$

$$\text{Recall} = \frac{\text{Actual moving edge pixels extracted}}{\text{Total number of acutual moving edge pixels}} \tag{9}$$

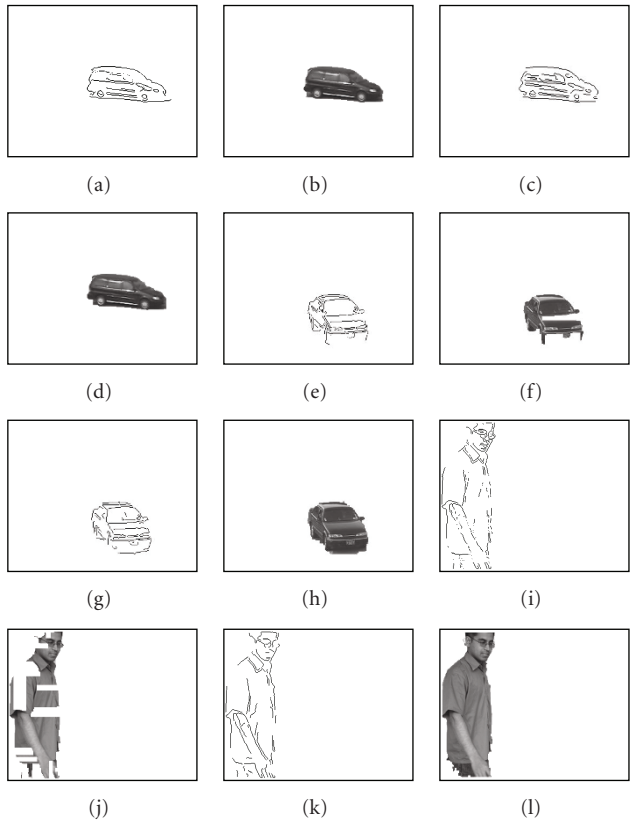

Figure 12: Segmentation of moving object. (a), (e), (i) Moving edge detection results by *KW*; (b), (f), (j) segmented moving objects of corresponding edge images by *KW*; (c), (g), (k) edge image of detected moving object by the proposed method; (d), (h), (l) segmented moving object of corresponding detected edge image by the proposed method.

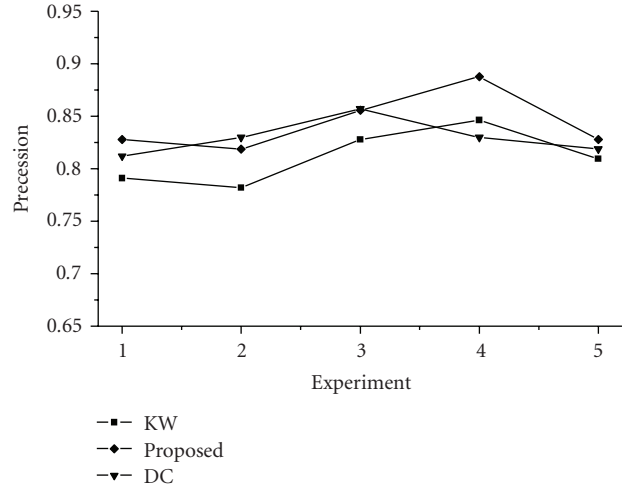

Figure 13: Precision of detected moving edges by different methods.

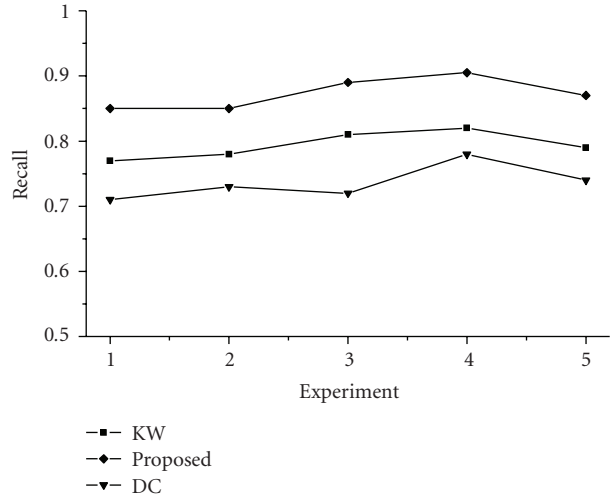

Figure 14: Recall of detected moving edges by different methods.

Precision tends to evaluate the accuracy of the detected moving edges, while recall is used to measure how much of the actual moving edges are extracted by a particular method. Precision also helps to determine the number of nonmoving edge pixels detected as moving edges whereas the second parameter, recall, provides the quantity of moving edges missed during the detection process.

Figure 13 shows the precision of detected moving edges by the proposed method. For comparison, precision values for *KW* and *DC* are also included. Results of five different experiments each having 12 frames are included here, where first two experiments are done for the image sequence obtained in indoor environment. As indoor environment is more challenging than outdoor environment, the results are comparatively a bit worse than that in outdoor sequence. Precision of the proposed method is better than other approaches. Flexible edge matching based on distance transformation and segment-based representation of edges of current image contributes to this improvement. Moreover, we apply a further refinement process on the detected coarse moving edges to get rid of background noisy edge segments. Results of experiment 5 are comparatively worse than that in two other outdoor experiments due to frequent variations of scene constituent in the busy road scene and noise.

Figure 14 shows the values of recall by different methods. Benefit of segment-based representation is clearly visible here as recall is higher for the proposed method in comparison to others. Due to the individual participation of edge pixel in matching, many edge pixels are missed to be detected. These results in scattered moving edges which eventually lead lower recall value for *KW* and *DC*. Degradation of result by *DC* occurred due to the loss of moving edges through matching by *AND* operator. Because of false matching, some of the moving edge pixels are classified as background and removed in detection process, which results in degradation of recall value in some extent in the proposed method. Some of the moving edges are also removed during postprocessing to filter out noisy edge segments. However, overall recall value is satisfactory considering the dynamism of environment and the obtained precision value together.

5. CONCLUSIONS

The proposed method presents a novel solution for moving object segmentation which is computationally efficient and suitable for real-time automated video surveillance system. This method overcomes some major limitations of existing background-independent methods by utilizing segment-based representation of edges and combining gradient information of moving edges in accumulated distance image. It also shows robustness against sensor noise, quantization error, and edge-localization error. Since the method utilizes most recent frames, it automatically adapts to the change of environment and it does not require any reinitialization step. The proposed edge matching method is performed in linear time and it is effective considering both accuracy and speed together. Segment-based representation of edges can be easily extended to moving object tracking, recognition, and classification. Extracted boundary of the segmented moving object by the proposed method is more precise as we apply watershed algorithm. Experimental results and comparative studies justify the effectiveness of the proposed method for moving object segmentation. However, the effectiveness of the proposed method can be further improved by determining the application-specific suitable values for the threshold parameters. Our future works focus on tracking of moving object using edge segment. Tracking information may also assist to adjust some of the threshold values dynamically to achieve better performance.

REFERENCES

[1] R. J. Radke, S. Andra, O. Al-Kofahi, and B. Roysam, "Image change detection algorithms: a systematic survey," *IEEE Transactions on Image Processing*, vol. 14, no. 3, pp. 294–307, 2005.

[2] A. D. Sappa and F. Dornaika, "An edge-based approach to motion detection," in *Proceedings of the 6th International Conference on Computational Science (ICCS '06)*, vol. 3991 of *Lecture Notes in Computer Science*, pp. 563–570, Reading, Mass, USA, May 2006.

[3] D. Gutchess, M. Trajković, E. Cohen-Solal, D. Lyons, and A. K. Jain, "A background model initialization algorithm for video surveillance," in *Proceedings of the 8th IEEE International Conference on Computer Vision (ICCV '01)*, vol. 1, pp. 733–740, Vancouver, Canada, July 2001.

[4] J. R. Bergen, P. J. Burt, R. Hingorani, and S. Peleg, "A three-frame algorithm for estimating two-component image motion," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 9, pp. 886–896, 1992.

[5] Y. Kameda and M. Minoh, "A human motion estimation method using 3-successive video frames," in *Proceedings of the 2nd International Conference on Virtual Systems and Multimedia (VSMM '96)*, pp. 135–140, Gifu, Japan, September 1996.

[6] M. Yokoyama and T. Poggio, "A contour-based moving object detection and tracking," in *Proceedings of the 2nd Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance (VS-PETS '05)*, pp. 271–276, Beijing, China, October 2005.

[7] D. J. Dailey, F. W. Cathey, and S. Pumrin, "An algorithm to estimate mean traffic speed using uncalibrated cameras," *IEEE Transactions on Intelligent Transportation Systems*, vol. 1, no. 2, pp. 98–107, 2000.

[8] C. Vieren, F. Cabestaing, and J.-G. Postaire, "Catching moving objects with snakes for motion tracking," *Pattern Recognition Letters*, vol. 16, no. 7, pp. 679–685, 1995.

[9] J. B. Kim and H. J. Kim, "Efficient region-based motion segmentation for a video monitoring system," *Pattern Recognition Letters*, vol. 24, no. 1–3, pp. 113–128, 2003.

[10] Q. Cai and J. K. Aggarwal, "Tracking human motion in structured environments using a distributed-camera system," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 11, pp. 1241–1247, 1999.

[11] D. Murray and A. Basu, "Motion tracking with an active camera," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, no. 5, pp. 449–459, 1994.

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

[13] D.-S. Lee, "Effective Gaussian mixture learning for video background subtraction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 5, pp. 827–832, 2005.

[14] J. H. Duncan and T.-C. Chou, "On the detection of motion and the computation of optical flow," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 3, pp. 346–352, 1992.

[15] S. M. Smith and J. M. Brady, "ASSET-2: real-time motion segmentation and shape tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 8, pp. 814–820, 1995.

[16] C. Kim and J.-N. Hwang, "Fast and automatic video object segmentation and tracking for content-based applications," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 12, no. 2, pp. 122–129, 2002.

[17] A. Makarov, J.-M. Vesin, and M. Kunt, "Intrusion detection using extraction of moving edges," in *Proceedings of the 12th IAPR International Conference on Pattern Recognition (ICPR '94)*, vol. 1, pp. 804–807, Jerusalem, Israel, October 1994.

[18] M. J. Hossain, M. A. A. Dewan, and O. Chae, "Edge segment-based automatic video surveillance," *EURASIP Journal on Advances in Signal Processing*, vol. 2008, Article ID 743202, 14 pages, 2008.

[19] M. J. Hossain, *An edge segment based moving object detection for automated video surveillance*, Ph.D. dissertation, Computer Engineering Department, Kyung Hee University, Yongin, South Korea, February, 2008.

[20] M. A. A. Dewan, M. J. Hossain, and O. Chae, "Moving object detection and classification using neural network," in *Proceedings of the 2nd KES International Symposium on Agent and Multi-Agent Systems: Technologies and Applications (KES-AMSTA '08)*, vol. 4953 of *Lecture Notes in Computer Science*, pp. 152–161, Incheon, Korea, March 2008.

[21] M. J. Hossain, M. A. A. Dewan, and O. Chae, "Moving object detection for real time video surveillance: an edge segment based approach," *IEICE Transactions on Communications*, vol. E90-B, no. 12, pp. 3654–3664, 2007.

[22] G. Borgefors, "Hierarchical chamfer matching: a parametric edge matching algorithm," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, no. 6, pp. 849–865, 1988.

[23] L. Vincent and P. Soille, "Watersheds in digital spaces: an efficient algorithm based on immersion simulations," *IEEE*

Transactions on Pattern Analysis and Machine Intelligence, vol. 13, no. 6, pp. 583–598, 1991.

[24] R. D. Yates and D. J. Goodman, *Probability and Stochastic Processes*, John Wiley & Sons, New York, NY, USA, 2nd edition, 2005.

[25] J. Lee, Y. Cho, H. Heo, and O. Chae, "MTES: visual programming environment for teaching and research in image processing," in *Proceedings of the 5th International Conference on Computational Science (ICCS '05)*, vol. 3514 of *Lecture Notes in Computer Science*, pp. 1035–1042, Atlanta, Ga, USA, May 2005.