Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 3, Issue. 9, September 2014, pg.675 – 683

RESEARCH ARTICLE

New Approach for Moving and Static Vehicle Detection Using Motion Energy

Josna George

M-Tech Student, Sree Narayana Gurukulam College of Engineering, Kerala

josnageorge91@gmail.com

Abstract - Motion detection and segmentation of traffic vehicles in an outdoor environment, particularly under non ideal weather conditions, in the presence of camera noise and with variable or unfavourable luminance conditions is still an area of active research. Gaussian based background modelling is commonly used to detect moving objects in computer vision systems. However, it has some limitations: It cannot effectively deal with sudden change in illumination, snowfall, fog, and repetitive motions such as swaying leaves. These nonideal outdoor conditions result in false motion detection. An alternative technique to detect and segment the moving vehicles by making use of dynamically adaptive threshold using the full-search sum of absolute difference (FSSAD) algorithm is proposed. Because FSSAD is computationally expensive, a modification using the adaptive-motion threshold is proposed, which not only reduces the false motion but improves the computational efficiency as well. In order to detect static vehicles from video sequences with complex background, we propose an algorithm which is based on running average background modelling and temporal difference method. Performance evaluation of our proposed framework achieves better segmentation and is suitable for real-time implementation.

Keywords – Block Matching Algorithm (BMA), adaptive threshold, running average background modelling, traffic monitoring, temporal difference

I. INTRODUCTION

Video Surveillance allows us to remotely monitor a live or recorded video feed which often includes people. There has been a tremendous proliferation of video surveillance cameras in public locations such as stores, ATMs, highways, traffic signals, schools, buses, subway stations, and airports in order to detect and track the scenario. As a technology for monitoring people using computer vision, video surveillance has received increased attention. Automated systems which help security personals to detect and track people are gaining importance.

Detection and tracking of moving objects is very important to monitor public transportation, and critical assets. The difficulty of moving object detection is mainly caused by the changing scenes, the moving objects may become a part of the scene when they come to a stop, meanwhile, the scene maybe affected by the

illumination changing, camera shaking, leaves swaying, etc. The existing detection and tracking based approaches for video object tracking are unreliable in complex surveillance videos due to problems like occlusions, lighting changes, and other factors. The segmentation of moving vehicles in outdoor traffic sequences is quite challenging due to various aspects such as unfavourable weather conditions, noisy low-quality videos, and dynamic background. . To provide efficient results, our robust segmentation algorithm not only localizes the object of interest but suppresses the noise that is introduced due to poor-quality low-resolution videos and small camera movements as well. In addition, it reduces the false motion that overshadows the actual objects of interest due to nonconductive weather conditions such as change in illumination and snowy or foggy weather, where the visibility is quite low.

The motivations for our proposed approach are given as follows: 1) to develop an algorithm that dynamically determines the threshold value based on the scene change to avoid extensive calibrations that are required to adapt to constantly changing weather conditions; 2) to minimize the rate of false motion while retaining the system sensitivity to detect the motion that is caused by the object of interest; and 3) to propose the solution for real-time applications that can be implemented on cost-effective platform for feasible practical deployment. We propose an approach using the full-search (FS) block- matching algorithm, which robustly detects the motion of interest and can suppress the false motion using the adaptive thresholding technique in a challenging outdoor environment. The sum of absolute difference (SAD) is commonly used measure in BMA and has proved to be one of the effective means of determining motion/change in video sequences. The motion vectors that are obtained using BMA provide a vital clue about the location of moving regions, which can efficiently be used to localize the objects. However, the segmentation using BMA is challenging, because it is essential to suppress the motion vectors that result from changing background.

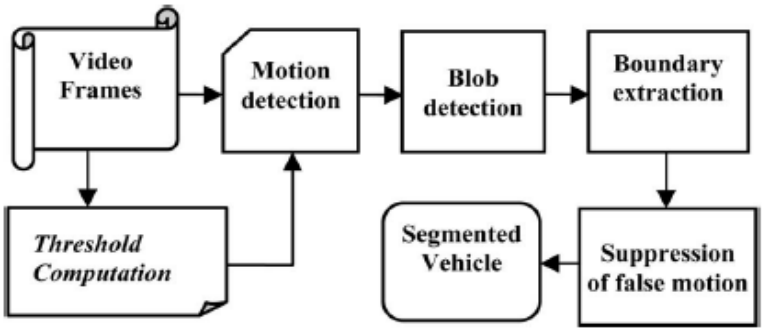

Fig 1. Framework of moving vehicle segmentation

In recent years, many static object detection algorithms have been proposed, they involve background subtraction, temporal difference, optical flow, and so on. Between these algorithms, the most widely used one is the background subtraction. Running average background model dynamically update the background image to adapt to the scene changing by using the weighed sum of the current image and background image. Because the running average background just needs to compute the weighted sum of two images, so it has low computational complexity and space complexity. Dynamically updating the background makes this model can adapt to very complex scene.

II. MOTION DETECTION

Block-based motion estimation is an efficient approach for identifying moving regions between video frames. To determine motion, we match the blocks of the reference frame (previous frame) with the current frame. If the block being matched is not in the same location, then it has moved. The location of the movement is obtained by the motion vector, and such blocks are considered moving regions. Different measures have been proposed in the literature as matching criterions such as the mean of SAD, SAD, and the sum of squared difference. SAD is considered an effective metric for motion detection and has been a widely used measure in BMAs. To obtain distinct objects during tracking, the frame rate should be coordinated with the motion available between frames. The elevation of the camera also plays a role in deciding the frame rate. As

the height of the elevation of the camera grows, so does the area that is covered, resulting in lower motion observed in subsequent frames. Similarly, when the camera elevation is lowered, the coverage area would be limited, thus providing larger motion between subsequent frames. Then, if the camera is covering a longer horizon, vehicles that approach or leave the horizon will cause relatively less motion in subsequent frames.

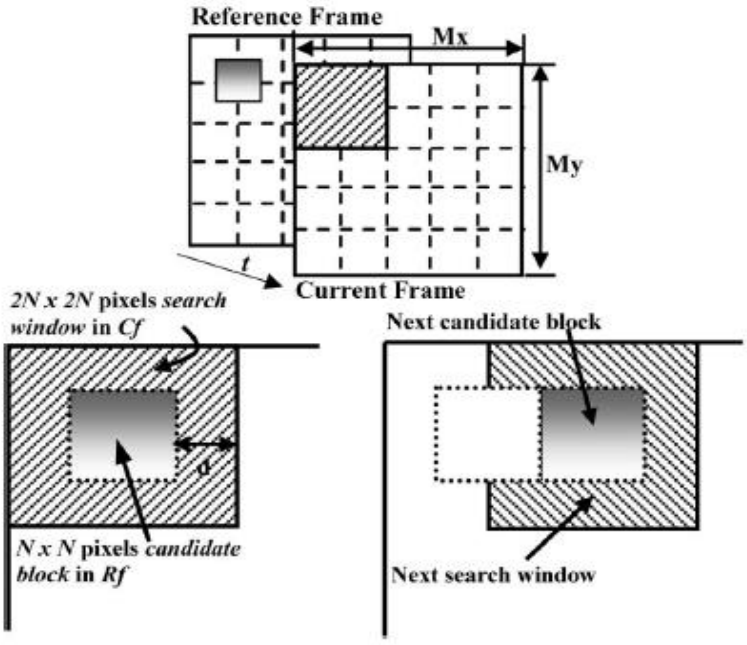

Fig 2. Concept of motion detection using a candidate blocks and search window

To compute the motion between frames, we divide the reference frame *Rf* into nonoverlapping candidate blocks *Cb*, which are identified as *Cb*(*p*,*q*), i.e., *Rf* ={*Cb*(*p*,*q*),(*p*, *q*) $\in$ ϕ}, where ϕ represents the total number of blocks in the *x*- and *y*-axes, given as $\phi = \{(p, q)|0 \leq p \leq Wx - 1, 0 \leq q \leq Wy - 1|\}$, and *Wx* and *Wy* are the numbers of blocks in the *x*- and *y*-axes, respectively. We compute the motion for a candidate block of size *N* × *N* using

$$\mathrm{SAD}(m,n) = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} |Rf(i,j,t_1) - Cf(i+m, j+n, t_2)|$$

$$\text{where } (m,n) = \{-d \leq m,n \leq d\}, \text{ and}$$

$$\mathrm{MV} = (m,n)|_{\min \mathrm{SAD(m,n)}}$$

where SAD(*m*, *n*) is the absolute difference at pixel location, (*m*, *n*), [−*d*, *d*] is the search region, and MV refers to the motion vector at a minimum value of SAD. Every candidate block carries out the search within a specified area of the current frame, called a search window, as shown in Fig. 2. We divide *Cf* into overlapped search windows of size 2*N* × 2*N*. The overlapping of the search windows ensures that the motion observed on the boundary of two search regions can effectively be identified. Let search window *Sw* be identified as *Sw*(*p*,*q*), i.e., *Cf* = {*Sw*(*p*,*q*), (*p*, *q*) $\in$ ϕ}. Candidate blocks generate the motion vector, i.e., *MV* = {*MV*(*p*,*q*), (*p*, *q*) $\in$ ϕ}, representing the motion observed between the frames separated at an interval of time *t*1 and *t*2. Deciding the size of the candidate block and the search windows is not a trivial task. Large candidate blocks are usually less sensitive to the small motion and may produce inaccurate results that span beyond the object boundaries. Although smaller block sizes are sensitive to noise, they can provide good approximation to object boundaries and provide better contours. Similarly, as the size of the search region grows, so does the computational complexity. We have chosen the candidate block of size 4 × 4 pixels and the search window as 8 × 8 pixels. It was observed that the chosen block size provides better results compared to a larger block size.

III. COMPUTATION OF THE DYNAMIC THRESHOLD AND BLOB DETECTION

Many researchers have used a frame-difference-based approach for differentiating between foreground and background. However, these approaches either make use of a threshold value that is predetermined or fixed or the proposed adaptive strategy is suitable for stationary background only. To obtain appropriate threshold, a lot of experimentation must be conducted on each video sequence. However, because the outdoor sequences have dynamic background that varies from frame to frame, even an empirically determined single value proves to be ineffective. Therefore, an adaptive threshold that is derived from every frame being processed that can effectively differentiate foreground from background is necessary. In this section, we propose a novel strategy to obtain a frame-based adaptive threshold that can be used to retain only the motion of the object of interest. To differentiate between the properties of moving vehicles and the background, we need to identify the distinct properties of vehicle and that of the background. The motion energy of a frame obtained by the summation of absolute difference of pixels provides a vital clue for discrimination. We convert the frames from color to grayscale and take two frames each from sequence with swaying leaves, fog, snowfall, and heavy snowfall to compute the motion energy as follows. Consider the frames to be of size $Mx \times My$. Let $I(x, y)$ denote the pixel intensity at location (x, y), $t1$ and $t2$ denote time; then, the motion energy at every pixel location is computed as

$$MEx,y = |I(x, y, t1) - I(x, y, t2)|$$

To compute the motion energy per block, we divide the frames into blocks of size $N \times N$. Let the number of blocks along the x- and y-axes be Wx and Wy; then, the motion energy computed for each block is given as

$$ME_{w,h} = \sum_{p=0}^{N-1} \sum_{q=0}^{N-1} ME_{p,q}$$

where $(w, h)|0 \le w \le Wx - 1, 0 \le h \le Wy - 1|$, and p, q are the location of pixels relative to each block.

Here the energy observed at the positions of moving vehicle is more than *twice* compared to the background such as swaying trees, fog, or snowfall, and because vehicles have a solid cohesive shape, these values cluster together to form a homogeneous shape. Consider frames of size $Mx \times My$ and let $I(\cdot)$ be the pixel intensity; then, the average motion energy (AME) per pixel along the x- and y-axes can be computed as

$$AME = \sum_{x=0}^{Mx-1} \sum_{y=0}^{My-1} |I(x, y, t_1) - I(x, y, t_2)| / (Mx \times My).$$

we can compute the AME per candidate block of size $N \times N$ as

$$Cb_{ME} = N^2 \times AME.$$

However, this is the motion energy caused due to moving object and dynamic background. With probability that moving vehicles cause at least twice the motion energy than the AME, we can compute the *adaptive-motion threshold Th* to retain only the vehicle motion as

$$Th = 2CbME + \beta CbME$$

where β is a precision factor. Whenever the background changes rapidly, the motion energy is higher, and when the background is stationary, the motion energy is lower. We propose the following method to automatically compute the value of β. Let $Wx \times Wy$ be the total number of candidate blocks along the x- and y axes. As aforementioned based on the analysis, the motion energy of the moving vehicles is higher than nearly twice the AME. Let *Thi* be the initial threshold, where

$$Thi = 2CbME.$$

Each candidate block based on the motion energy computed is categorized as either foreground or background block as under

$$Decision = \begin{cases} Foreground, & \text{if } ME_{w,h} > Th_i \\ Background, & \text{otherwise} \end{cases}$$

where $(w, h)|0 \le w \le Wx - 1, 0 \le h \le Wy - 1|$. Let *Bblocks* be the total number of background blocks computed, then the AME experienced by background blocks can be computed as

$$Avg_{bg} = \frac{1}{B_{blocks}} \times \sum_{i=0}^{B_{blocks}-1} ME_i$$

The relationship of *CbME* and *Avgbg* can then be established as

$$Ratio_{bavg} = \frac{Avg_{bg}}{Cb_{ME}}$$

When there are no moving vehicles in the scene, the resulting motion energy is primarily due to background blocks, making *Ratiobavg* approach 1. Similarly, when the majority of the resulting motion energy is due to foreground vehicles, *Ratiobavg* approaches 0. Thus, $\{Ratiobavg \in \gamma r, 0 \leq \gamma r \leq 1\}$.Whenever the ratio is higher, the value of β should be positive, increasing the threshold value to suppress the interference of background regions; otherwise, it should be zero or lower. The mean of variation range is used to decide the value of β as follows:

$$mean_{\gamma_r} = (\max(\gamma_r) + \min(\gamma_r))/2$$
$$\beta = Ratio_{bavg} - mean_{\gamma_r}.$$

This value of β is used to compute the threshold.

IV. MODIFICATION TO FSSAD

We propose an alternative strategy that relies on the dynamic threshold for discarding the motion caused other than the moving vehicles. Conventional BMAs calculate the absolute difference within the search window in a raster scan fashion to determine the motion vector. However, when the search window remains motionless, the matching block is found right at the center of the search window. As a result, even for the motionless window, a lot of time is spent in arriving at the conclusion. Similarly, when the search window experiences a motion, the overlapping blocks exhibit higher motion energy. These two facts have been exploited to modify FSSAD.

ALGORITHM FOR THE MODIFIED FSSAD

```
Divide Rf into candidate blocks of size N × N
Divide Cf into overlapping search windows of size 2N × 2N
for i:=0 to Wx − 1 do //All search windows on frame width
    for j:=0 to Wy − 1 do //All search windows on frame height
        compute SAD_OB save location as MV (zero-motion)
        if SAD_OB > Th then //Is motion due to moving vehicle?
            //Begin search within search window
            for all blocks within a search window do
                compute the SAD value;
                if SAD < Th then
                    continue;
                else
                    compare SAD_min and SAD and update MV
                end if
            end for all
        else
            save MV as zero-motion ;//Do not search
        end if
    end for j
end for i
```


We compute the absolute difference *SADOB*, indicating the motion energy of *overlapping* candidate blocks of *Rf* and *Cf*, and use it to decide the searching decision within the search window. *SADOB* is computed as

$$SAD_{OB} = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} |Rf(i, j, t_1) - Cf(i, j, t_2)|$$

where *i* and *j* refer to the same pixel location in *Rf* and *Cf*. For every search window, we first compute *SADOB* and compare it with the adaptive-motion threshold. If the value is found to be less than the threshold, then the search region is considered to be either motionless or background; otherwise, we initiate the search for the best match within the search window. Next, for every possible candidate block, within the search window, we compare the computed SAD value with the threshold by moving pixel by pixel in a raster scan fashion, and if it is found to be greater than the threshold, only then would the motion vector be updated. This procedure is applied to every search window of the given frame. The proposed approach not only reduces the false motion but also lowers the computational time.

V. BLOB DETECTION

In the blob detection step, we group the motion blocks based on their relationship between neighbouring blocks. We consider only bocks that have been retained after applying the threshold. The notation of block connectivity describes a relationship between two or more blocks.

For a block *b* with coordinates (*x, y*), a set of blocks as given below are called 4-neighbors

$$N4(b) \{(x + 1, y), (x, y + 1), (x - 1, y), (x, y - 1)\}$$

and a set of blocks as given below are called 8-neighbors

$$N8(b) = N4(b) \cup \{(x + 1, y + 1), (x + 1, y - 1), (x - 1, y - 1), (x - 1, y + 1)\}$$

We make use of 8 neighbours to identify the blocks that can be grouped together to form one blob. We make use of the DFS-based connected-component labelling to group the blocks together. A set of connected blocks are identified as one component, and a set of such components amount to total components as an output of this step.

VI. BOUNDARY EXTRACTION AND SUPPRESSION OF FALSE MOTION

The application of threshold reduces unwanted motion blocks that pertain to the dynamic background, and the blob detection step provides the connected motion blocks. When the vehicles have uniform color, motion blocks usually occupy the vehicle edges, because edges reflect higher temporal change than the body. Thus, if we regroup blocks that are on the edges of the vehicle, then we can get the boundary of the vehicles. BMAs base the very idea of motion detection on the aggregated motion energy experienced by all the pixels of the candidate block. Thus, the mean of motion experienced by all the pixels within the candidate block can safely be approximated to converge on the center of the block. Thus, the center of each block represents one point. We make use of convex hull to connect the blocks that fall on the object boundary. After the application of the dynamic threshold, most of the false motion that was caused due to dynamic background is eliminated. A chosen candidate block of size 4 × 4 can detect a fine motion exhibited by the moving vehicle and occupies multiple blocks that cover the vehicle area, whereas in most of the cases, the background motion blocks exhibit one of the following characteristics:

- a single block;
- a connected chain of blocks in a straight line;
- spanning few blocks across the left and right axes of the center of the convex hull.

When building the convex hull, we check if the connected component exhibits any one of the aforementioned properties, and if the answer is positive, we discard the respective motion blocks. this step further minimizes the rate of false motion. After the boundary extraction, we reset the background to black and retain only pixels that fall within the boundary of an object.

Now we can discuss the steps necessary for the static object detection using running average background subtraction method as follows.

VII. RUNNING AVERAGE BACKGOUND

Running average background model dynamically update the background image to adapt to the scene changing by using the weighed sum of the current image and background image. The updating formula is:

$$B_{t+1}(x,y) = (1-\alpha)B_t(x,y) + \alpha F_t(x,y)$$

Where α is the updating rate, t B is the background image at the time t , t F is the current image at time t . The updating rate α represents the speed of new changes in the scene updated to the background frame. However, α cannot be too large because it may cause artificial "tails" to be formed behind the moving objects. Because the running average background just needs to compute the weighted sum of two images, so it has low computational complexity and space complexity. Dynamically updating the background makes this model can adapt to very complex scene.

VIII. TEMPORAL DIFFERENCE METHOD

Temporal difference method computes the difference image between two consecutive frames, then, after thresholding the difference image, we can get the static vehicles. This method has low computational complexity, however, it is very sensitive to the threshold, too small threshold will cause a lot of noise in the detected results, but if the threshold is too large, much information of the objects will be missing.

The traditional running average background using the weighted sum to update every pixel in the background image, however it is not reasonable to do so, because it will introduce some pixels in the current image into the background, then the background will be polluted by pixel logically not belonging to it. So, we use the selectivity running average background model to dynamically update the background image. We introduce the temporal difference method into the selectivity running average background model to improve the accuracy of the detected results .

The updating formula is as follows:

$$\begin{cases} B_{t+1}(x,y) = \alpha F_t(x,y) + (1-\alpha)B_t(x,y) \text{ if } F_t(x,y) \text{ is background} \\ B_{t+1}(x,y) = B_t(x,y) \text{ if } F_t(x,y) \text{ is foreground pixel} \end{cases}$$

Figure 3 shows the processing steps of our proposed method. We select a small updating rate to update the background, and use the current image to subtract the background image to get the foreground image.

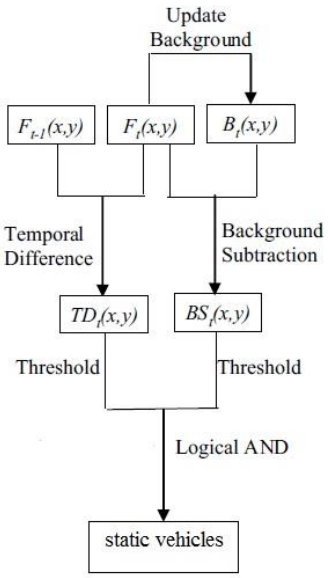

Fig 3. Concept of running average and temporal difference method

We threshold the foreground image by using a large threshold. The vehicles in the foreground image may have drop shadow because the background updates too slow. However, there is only little noise, and the

most information of the vehicles is preserved. We use the temporal difference method to get the difference image, and thresholding it by using a very small threshold. The difference image may contain a lot of noise, because of the small threshold, but the contours of the static vehicles have been preserved very well. Then, we combine the foreground image with the difference image through the "Logical AND" operation between the two binary images. By doing the "Logical AND" operation, we eliminate a lot of noise in the difference image, and also eliminate the drop shadow in the foreground image.

IX. CONCLUSION

Compared to the previous methods for moving vehicle detection, this method is robust and efficient in segmenting moving vehicles under complex outdoor conditions. The motivation of this proposed framework was to exploit the motion energy of moving vehicles to segregate them from the changing background. The framework was tested using traffic videos with challenging outdoor conditions such as heavy snow showers, poor visibility due to fog, vibrating camera, and swaying leaves. The strategy for the computation of the adaptive motion threshold not only distinguished the moving vehicles from the background but also led to considerable reduction in the computation time of FSSAD.

Certain improvements can be made to the proposed method. When the two vehicles come very close to each other, they tend to form a single blob. By using some post processing techniques, this condition can be overcome. Although it was successful in considerably reducing false motion that was caused due to different weather conditions, the effect of heavy snowfall still retains a moderate effect, particularly of the longer tail heavy showers. By preprocessing these frames or by the knowledge of shape of these showers, it could be possible to further reduce their effect and obtain a much clearer background.

This paper also proposes a static object detection algorithm which combines the running average background method and temporal difference method. Through taking the advantage of each method's strong points, our method can get good detect result and keeps low computational complexity. This method can deal with slow slighting changes by slowly updating the background. It also deals with swaying branches, and can be applied to real time surveillance systems. In the future research work, designing a more robust moving object detection algorithm, and integrate it into an embedded surveillance application system can be done.

REFERENCES

[1]. J. M. Ferryman, S. J. Maybank, and A. D. Worrall, "Visual surveillance for moving vehicles," Int. J. Comput. Vis., vol. 37, no. 2, pp. 187–197, Jun. 2000.

[2]. W. Niu, L. Jiao, D. Han, and Y. Wang, "Real-time multiperson tracking in video surveillance," in Proc. Pacific Rim Conf. Multimedia, Singapore, Dec. 15–18, 2003, vol. 2, pp. 1144–1148.

[3]. J. Lipton, H. Fujiyoshi, and R. S. Patil, "Moving target classification and tracking from real-time video," in Proc. 4th IEEE Workshop Appl. Comput. Vis., Princeton, NJ, Oct. 19–21, 1998, pp. 8–14.

[4]. L. D. Stefano and E. Viarani, "Vehicle detection and tracking using the block-matching algorithm," in Proc. 3rd IMACS/IEEE Multiconf. Cir- cuits, Syst. Commun. Comput., Athens, Greece, Jul. 4–8, 1999, vol. 1, pp. 4491–4496.

[5]. M. Vargas, J. M. Milla, S. L. Toral, and F. Barrero, "An enhanced back- ground estimation algorithm for vehicle detection in urban traffic scenes," IEEE Trans. Veh. Technol., vol. 59, no. 8, pp. 3694–3709, Oct. 2010.

[6]. P. Kaewtrakulpong and R. Bowden, "An improved adaptive background mixture model for real-time tracking with shadow detection," in Proc. 2nd Eur. Workshop Adv. Video Based Surv. Syst., Kingston, U.K., Sep. 2001.

[7]. W. Hu, T. Tan, L. Wang, and S. Maybank, "A survey on visual surveillance of object motion and behaviors," IEEE Trans. Syst., Man, Cybern. C: Appl. Rev., vol. 34, no. 3, pp. 334–350, Aug. 2004.

[8]. Shu-Te Su and Yung-Yaw Chen Department of Electrical Engineering, "Moving Object Segmentation Using Improved Running Gaussian Average Background Model ".

[9]. Zheng Yi ,Fan Liangzhong "Moving Object Detection Based on Running Average Background and Temporal Difference "

[10]. W. Niu, L. Jiao, D. Han, and Y. Wang, "Real-time multiperson tracking in video surveillance," in *Proc. Pacific Rim Conf. Multimedia*, Singapore, Dec. 15–18, 2003, vol. 2, pp. 1144–11

