Chapter 3

Application to Surveillance Camera

3.1 Moving Object Extraction

Moving object extraction is a fundamental technique for applications such as object tracking and segmentation, traffic monitoring and surveillance systems. Extracting moving vehicles from image sequences of traffic scenes recorded by stationary camera become one of the basic technologies which are used to acquire traffic information such as vehicle type, vehicle velocity and traffic flow. Finding moving objects from background is the most important task in extracting technologies for image sequences. In application such as surveillance systems, background image in which there is no object can be used to detect moving objects in the scene. However, background can change according to changes in illuminations or changes in background itself time to time. Therefore, it is important to maintain the background frame. In our experiment, we compare two object extraction methods which are accumulative difference images and background differencing.

3.1.1 Accumulative Difference Images (ADIs)

Image motion is one of the most important information that can be used for image segmentation. In imaging applications, motion arises from a relative displacement between the sensing system and the scene being viewed. One of the simplest approaches for detecting changes between two image frames $f(x,y,t_i)$ and $f(x,y,t_j)$ taken at time t_i and t_j , respectively, is to compare the two images pixel by pixel. One procedure for doing this is to form a difference image.

Suppose that we have a reference image containing only stationary components. Comparing this image against a subsequent image of the same scene, but including a moving object, results in the difference of the two images cancelling the stationary elements, leaving only nonzero entries that correspond to the non-stationary image components.

A difference image between two images taken at times t_i and t_j can be defined as

$$d_{ij}(x,y) = \begin{cases} 1 & if \left| f(x,y,t_i) - f(x,y,t_j) \right| > T \\ 0 & Otherwise \end{cases} \tag{3.1}$$

where T is a specified threshold. $d_{ij}(x,y)$ has a value of 1 at spatial coordinates (x,y) only if the gray-level difference between the two images is appreciably different at those coordinates, as determined by the specified threshold T.

In dynamic image processing, all pixels in $d_{ij}(x, y)$ with value 1 are considered the result of object motion. This approach is applicable only if the two images are registered spatially and if the illumination is relatively constant within the bounds established by *T*. In practice, 1-valued entries in $d_{ij}(x, y)$ often arise as a result of noise. Typically, these entries are isolated points in the difference image and a simple approach to remove by a thresholded connectivity analysis. However, this filtering process can also remove small or slow-moving objects. To solve this problem, we consider changes at a pixel location over several frames. The idea is to ignore changes that occur only sporadically over a frame sequence and can therefore be attributed to random noise.

Consider a sequence of image frames $f(x,y,t_1)$, $f(x,y,t_2)$,…, $f(x,y,t_n)$ and let $f(x,y,t_1)$ be the reference image. An *accumulative difference image* (ADI) is formed by comparing this reference image with every subsequent image in the sequence. A counter for each pixel location in the accumulative image is incremented every time a difference occurs at that pixel location between the reference and a image in the sequence. Thus when the k^{th} frame is being compared with the reference, the entry in a given pixel of the accumulative image gives the number of times the gray level at that position was different from that corresponding pixel value in the reference image.

There are three types of accumulative difference images: *absolute, positive* and *negative* ADIs. Assuming that gray-level values of the moving objects are larger than the background, these three types of ADIs are defined as follows. Let $R(x, y)$ denote the reference image and let k denote tk, so that $f(x, y, k) = f(x, y, t_k)$. We assume that $R(x, y) = f(x, y, 1)$.

Then, for any $k > 1$ and keeping in mind that the values of ADIs are counts, we define the following for all relevant values of (x, y):

$$A_k(x,y) = \begin{cases} A_{k-1}(x,y)+1 & \text{if } \left|R(x,y) - f(x,y,k)\right| > T \\ A_{k-1}(x,y) & \text{Otherwise} \end{cases}$$

$$P_k(x,y) = \begin{cases} P_{k-1}(x,y)+1 & \text{if } \left[R(x,y) - f(x,y,k)\right] > T \\ P_{k-1}(x,y) & \text{Otherwise} \end{cases}$$

$$N_k(x,y) = \begin{cases} N_{k-1}(x,y)+1 & \text{if } \left[R(x,y) - f(x,y,k)\right] < -T \\ N_{k-1}(x,y) & \text{Otherwise} \end{cases} \tag{3.2}$$

where $A_k(x,y)$, $P_k(x,y)$ and $N_k(x,y)$ are the absolute, positive and negative ADIs, respectively, after the k^{th} image in the sequence is encountered.

We test the three types on ADIs on fifty frames of synthetic image sequence with size 256x256. The sequence includes one bright object (intensity = 255) and one dark object (intensity = 10) with stationary background (intensity = 50). Figure 3.1(a) is the synthetic test image sequence. Figure 3.1(b-d) are the absolute, positive and negative ADIs. It can be seen that positive ADI can detect the bright object and the moving path of dark object while the negative ADI can detect the initial position of dark object and moving path of bright object. Absolute ADI shows the initial position of moving objects and also their moving paths which is the combination of positive ADI and negative ADI.

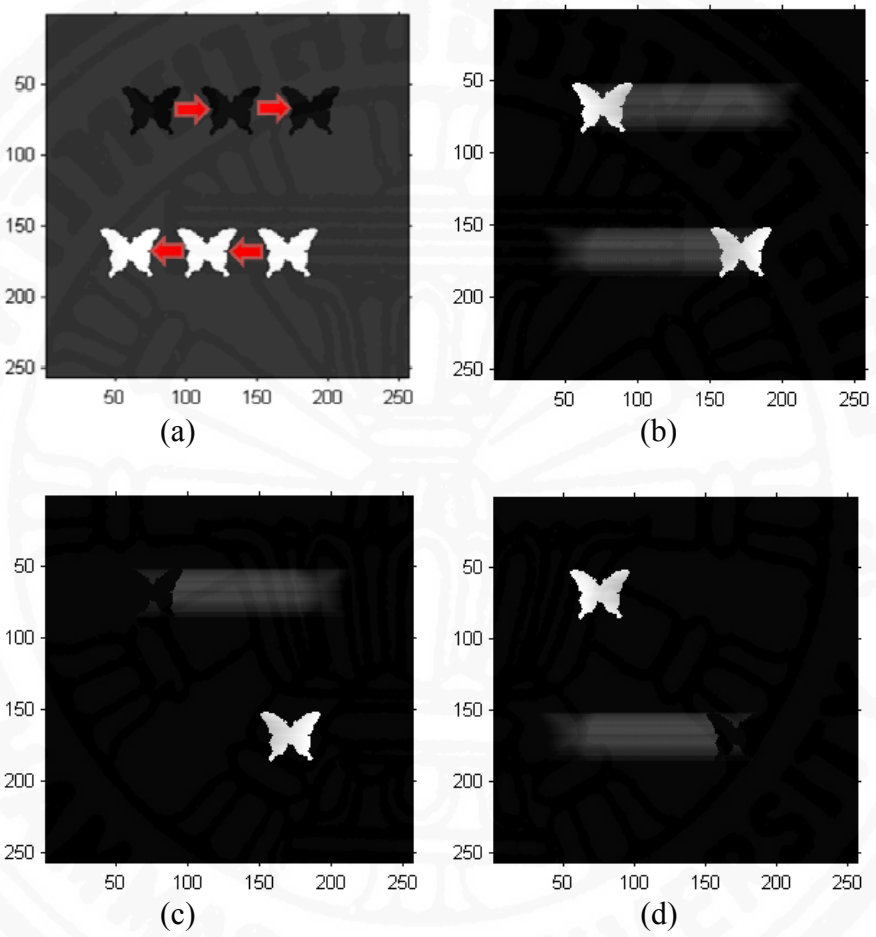


Figure 3.1 (a) Motion direction in original image sequences, (b) Absolute ADI, (c) Positive ADI and (d) Negative ADI of Butterfly Synthetic Sequence

Absolute ADI is tested for segmenting moving objects. Thresholding is applied to remove those shadows. After thresholding, morphological operations such as opening and closing are applied to smoothen the result. Figure 3.2(a) shows the binary image after thresholding and 3.2 (b) is the result after morphological operations. We tested the method on several image sequences.

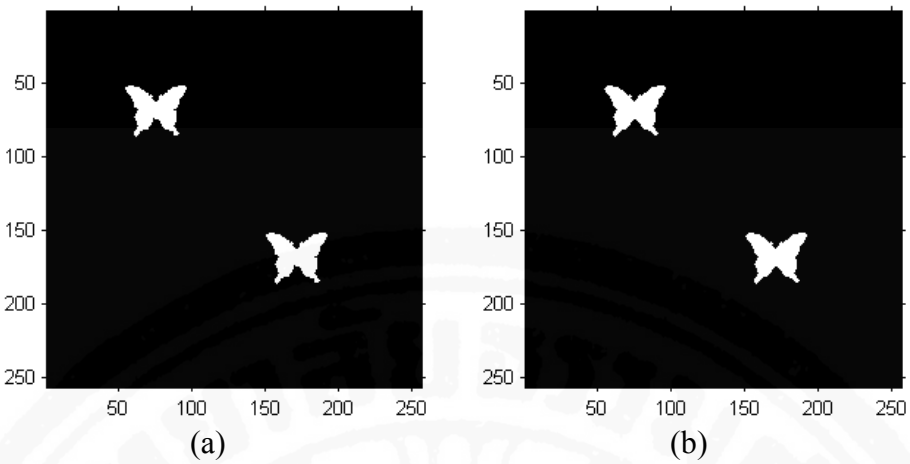


Figure 3.2 (a) Binary Image after Thresholding and (b) Image after Morphological Operations of Butterfly Synthetic Sequence

AADI is simple and easy to implement however, there is a limitation of this method because it is sensitive to changes in scenes. To make AADI effective, background of a sequence needs to be stationary. For the previous synthetic butterfly image sequence, the background is stationary so we can apply AADI to extract moving objects perfectly. However, in real time applications, there cannot always achieve to get the stationary background image sequences.

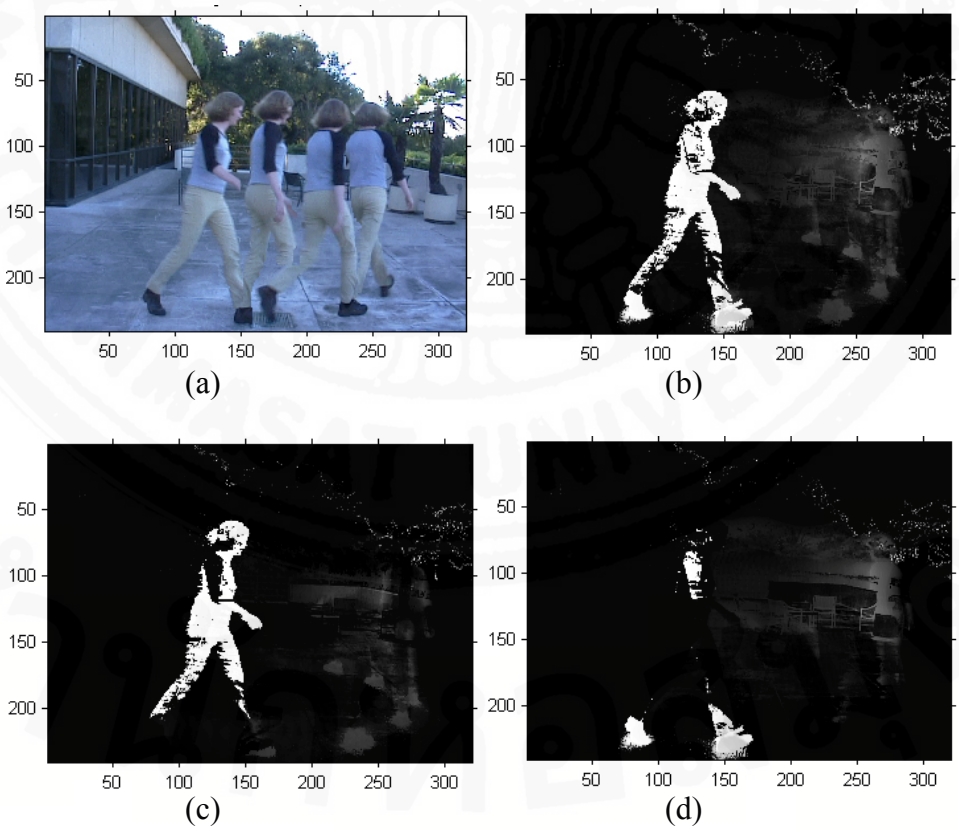


Figure 3.3 (a) Original image sequences, (b) Absolute ADI, (c) Positive ADI and (d) Negative ADI of Walking Circle Sequence

Figure 3.3 shows applying ADI on the real time image sequence [taken from http://www.nada.kth.se/~hedvig/data.html]. Figure 3.3 (a) is the moving direction of object in the sequence by showing the selected frames. Total 50 frames of images are used for this experiment. Figure 3.3 (b) is AADI result by using the specified threshold value T = 20. This threshold value can be varied according to the illumination of the images. From Figure 3.3 (c) and (d), it can be seen that positive ADI can detect the parts that are brighter than background while negative ADI can detect objects that are darker than background such as sleeve and shoes of walking person.

Figure 3.4 is the Binary image after thresholding and morphological operations. It can be seen that some part of moving object is missing and it is also hard to find appropriate threshold value. In this case, AADI has failed to obtain a perfect segmentation.

Therefore, to deal with the camera movements or non-stationary background, we are going to generate by using median filter and use that background instead of the real background to extract moving objects.

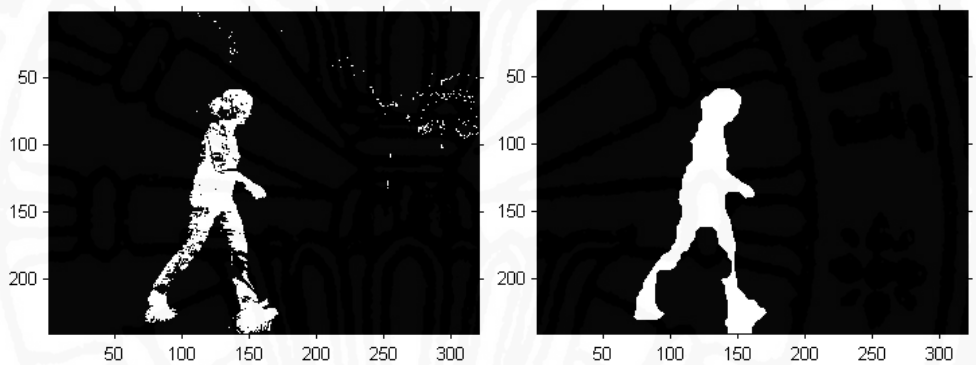


Figure 3.4 (a) Binary Image after Thresholding and (b) image after Morphological Operations of Walking Circle Sequence

3.1.2 Background Subtraction using Median Filter

Background generation is widely used in video-based tracking and surveillance systems as a preprocessing method. Reliable background generation is important in robust video object tracking in surveillance systems.

Temporal median filtering is applied to an image sequence in time direction. Figure 3.5 shows the steps how to apply temporal median filtering to a sequence.

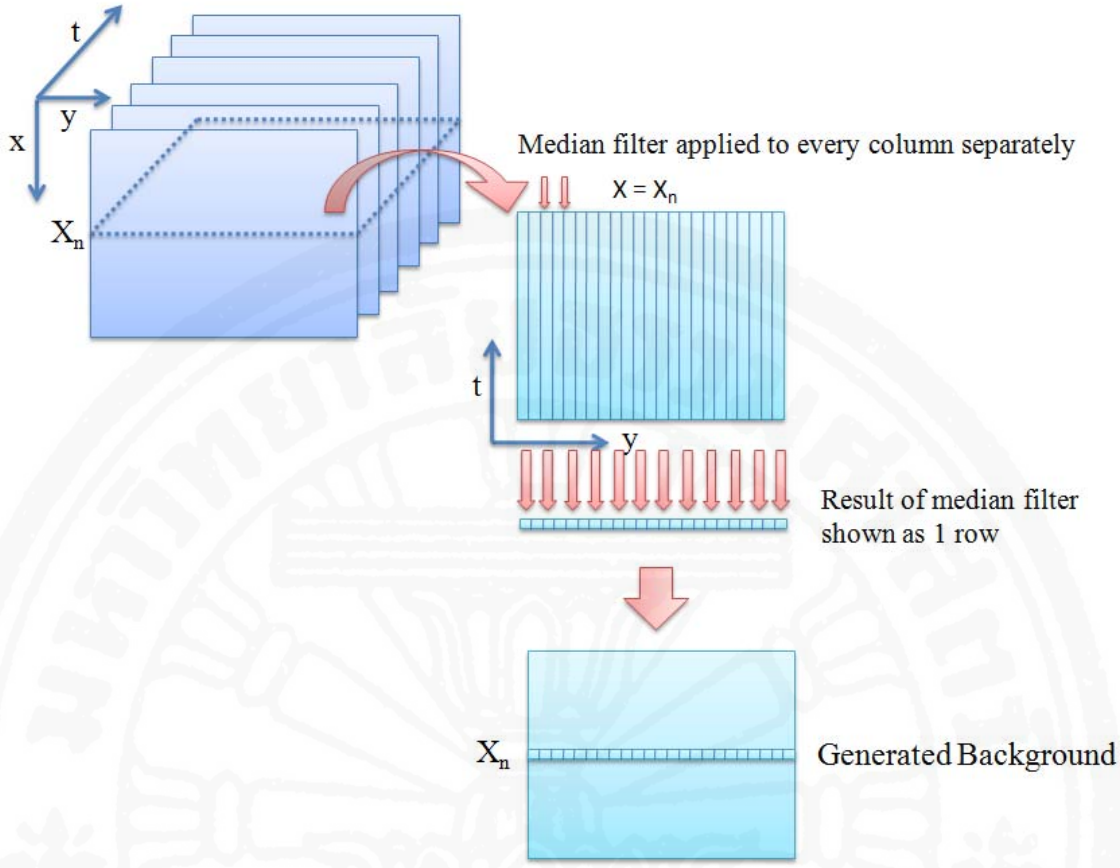


Figure 3.5 Steps to perform median filtering to obtain new background

Temporal median filtering can be done by considering the every row of the sequence in temporal direction and find the median values of every column in y-direction. Moving objects which are different intensities from background can be discarded and only background with stationary components is generated. In this way, we can generate the reliable background of the sequence. Background is not always available in every test sequences and intensity can be changed according to time and weather. Therefore, creating robust and reliable background is necessary for effective background subtracting in moving object extraction techniques.

We apply the temporal median filtering onto walking circle image sequence and background is generated by using 50 frames images. Figure 3.7 (a) shows the original background and Figure 3.7 (b) is the generated background by using temporal median filtering.

Moving objects can be extracted by subtracting the frame of interest in the sequence with the generated background. Therefore, all the stationary components are deleted and left with only non-stationary components which represent moving objects. Figure 3.7 (c) and (d) show the reference frame and image after subtracting the reference frame with generated background. Thresholding and morphological operations are applied to image after subtracting. Figure 3.7(a) and (b) show binary image after thresholding and morphological operations, respectively.

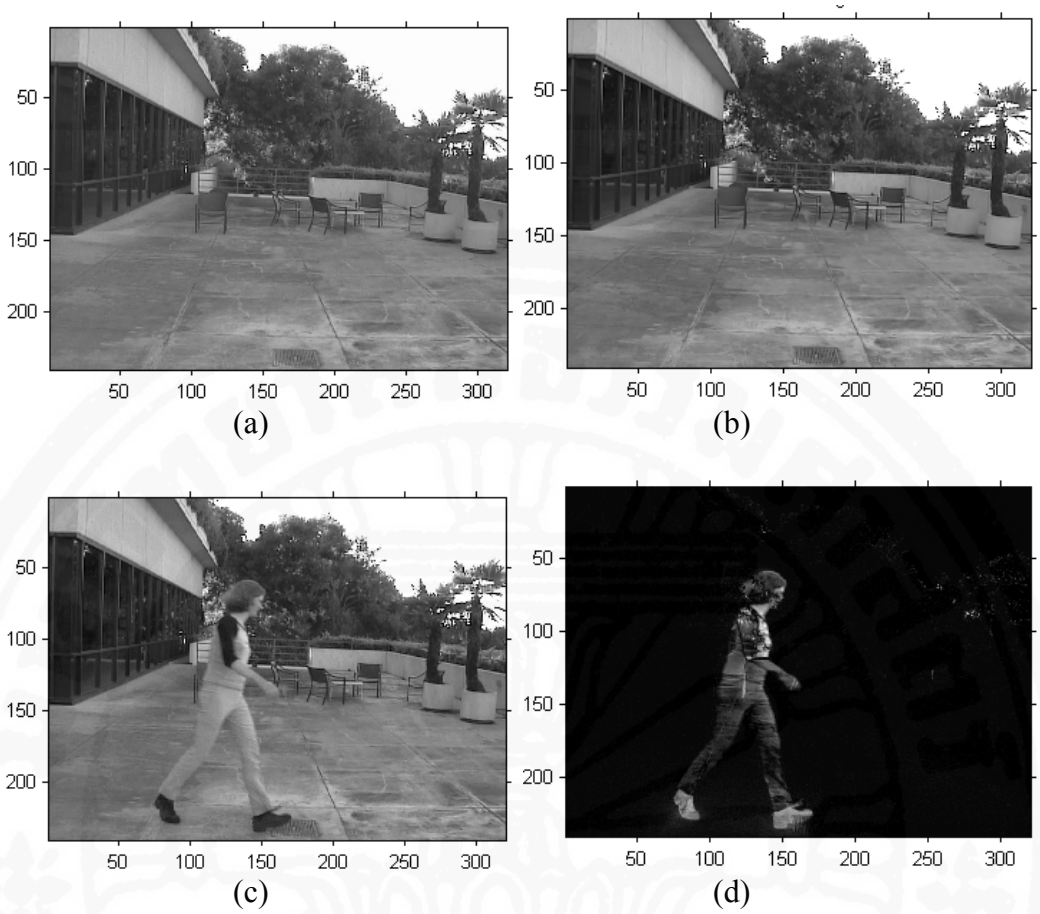


Figure 3.6 (a) Original background, (b) Generated background, (c) Reference frame and (d) Difference image between reference frame and generated background

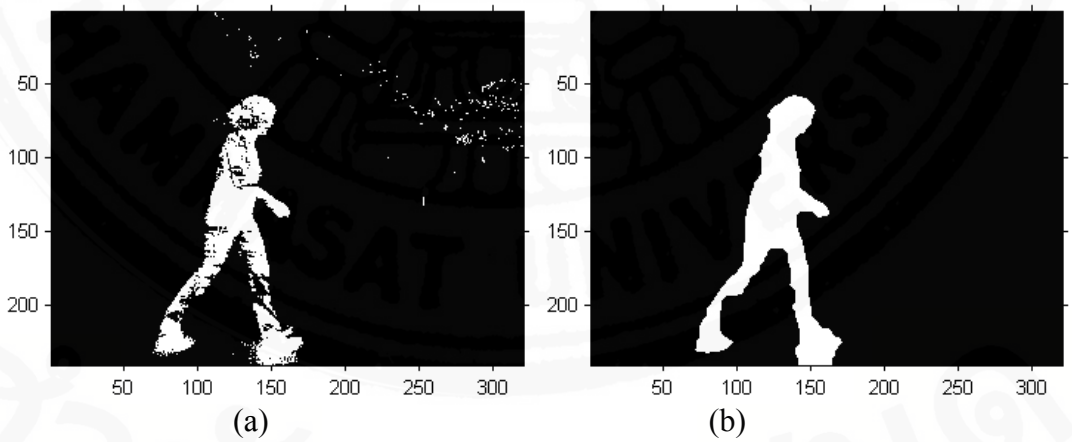


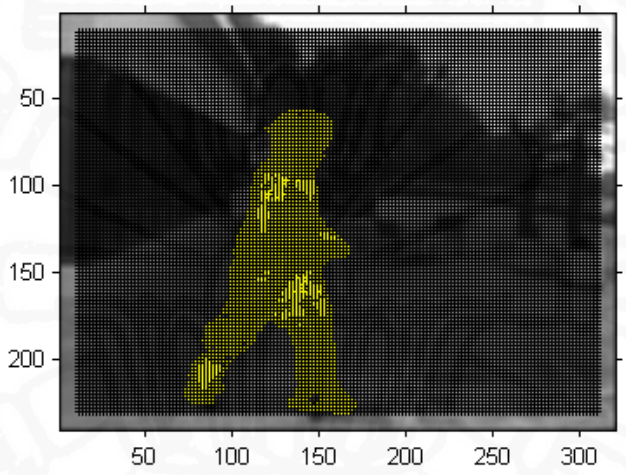
Figure 3.7 (a) Binary image after thresholding and (b) final segmented result after morphological operations

We can see that using the generated background, we can extract out the moving objects clearly. From the above sequence, it shows that background subtraction method using temporal median filtering can recover the part of moving object that is failed by using AADI method. Therefore, temporal median filtering method is chosen to use for detecting moving objects with stationary background. Results for more image sequences will be shown in Chapter 4.

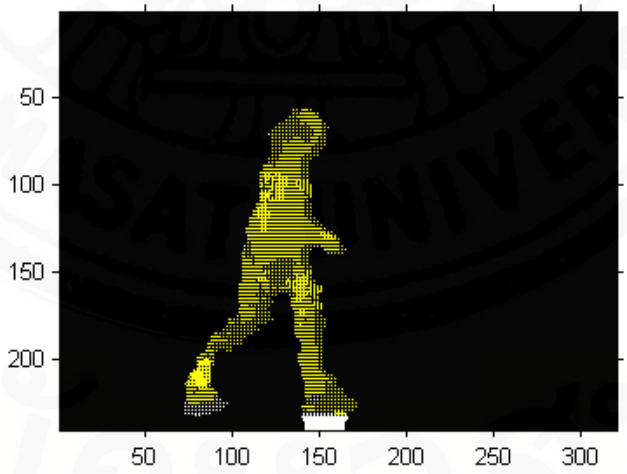
3.2 Motion Visualization within Moving Object

After we obtain the binary result of temporal median filtering background subtraction, we use it as a mask of moving object. Dense motion vectors are calculated by GSTM on the reference frame. The size of the integration area is 2×2×3 voxels and margin is left by 10 pixels. Therefore, we have 110×150 motion vectors in the image sequence of size 320×240 pixels.

Median filter of size 7×7 is applied to the image to remove large motion errors which may occur at the boundaries of moving objects. Binary mask is then applied onto the motion vectors obtained from GSTM. In this case, we obtain the motion vectors only within the moving objects. Figure 3.8 (a,b) show the Walking Circle sequence with motion vectors estimated by GSTM and after masking it by using the binary result from temporal median filtering.



(a)



(b)

Figure 3.8 (a) Motion vectors estimated by GSTM for entire image and (b) motion vectors only within moving object after masking

Classification is done by sending motion vectors within moving object to SOM neural network. For real image sequences, we use SOM network with two input neurons (u, v) with three output neurons. Final feature map after 5,000 iterations is shown in Figure 3.9 (a) and values of motions represented by each color in Figure 3.9 (b). Different colors indicate different directions and/or speeds of motion of respective regions. Using this final map, motion vectors within moving object are segmented into 9 different colors as shown in Figure 3.10.

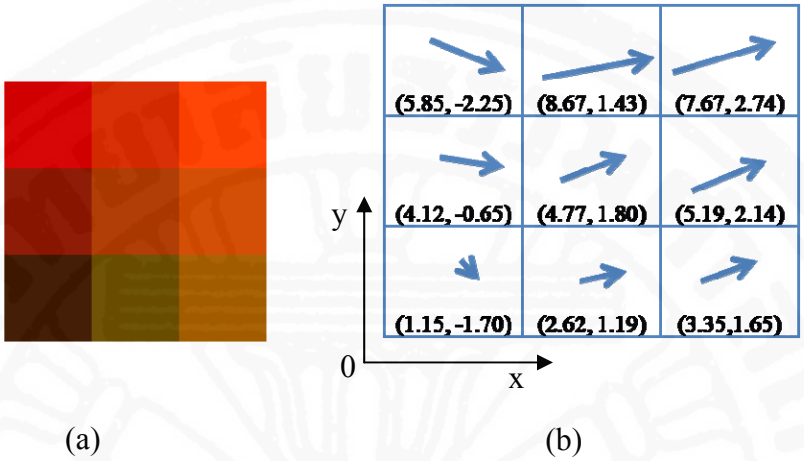


Figure 3.9 (a) Final feature map and (b) corresponding motion vectors (u, v) for each color for walking circle image sequence.

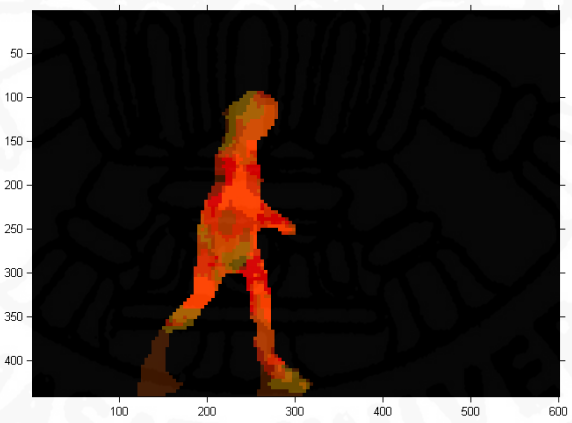


Figure 3.10 Classification result after 5,000 iterations (walking circle sequence)