Dissertation

Doctor of Philosophy

A STUDY ON HUMAN MOTION DETECTION --- TOWARD ABNORMAL MOTION IDENTIFICATION

by

Panca Mudjirahardjo

Supervised by

Associate Professor Joo Kooi Tan
Professor Seiji Ishikawa

Graduate School of Engineering
Department of Mechanical and Control Engineering
Kyushu Institute of Technology
JAPAN

2015

博士論文

人の動作の検出に関する研究
ー 異常動作の特定に向けて

Dissertation
Doctor of Philosophy

A STUDY ON HUMAN MOTION DETECTION --- TOWARD ABNORMAL MOTION IDENTIFICATION

パンチャ　ムジラハルジョ
学生番号 11584208

指導教員 タン ジュークイ 准教授
石川 聖二 教授

九州工業大学 大学院 工学府
機械知能工学専攻 知能制御工学コース
平成 27 年度

Abstract

Nowadays, there is an increasing demand for security system by continuously using CCTV systems. Most current surveillance systems need a human operator to constantly monitor them. Their effectiveness and response is largely determined not by the technological capabilities but by the vigilance of the person monitoring the display. To overcome these limitations of traditional surveillance methods, a major effort is under way in the computer vision and artificial intelligence community to develop automated systems. An Automatic Vision Surveillance (AVS) has become an important topic for the academic community, which aims to develop autonomous surveillance schemas to replace the traditional schemes.

The human motion detection can be performed in two approaches: The first one is a flow or motion based approach. The second approach is head motion based approach. These approaches are usually used to detect an abnormal motion.

The existing methods of the first approach have some inadequacies: Some of them rely on feature points extraction and tracking. Owing to the lack of feature points, it increases the false negative value; Some of them perform the observing of flow directions only; Some of them fail to detect an abnormal motion when the motion flow is not perpendicular with camera direction; Some of them rely on the training process; Some of them need a camera observation from very high position. Some of them should be applied in extremely crowded scenes.

The existing methods of the second approach have some inadequacies: Some of them rely on foreground extraction and very sensitive with a pattern which similar to a head. Some of them need a complex computation and big vector dimension.

In order to make up for the inadequacies of the existing human motion detection based on the first approach, we propose the improvement of the motion flow method. In this research, our information on motion flow is provided by the analysis of motion history image (MHI). Our method gives more accurate as well as important information for fast motion detection in a condition where the motion is not perpendicular with a camera view.

Another improvement is based on the second approach. In this thesis, we introduce a histogram of transition as a novel feature. Since this feature calculates the transition between the background and the foreground, the computation is very simple and takes a short time compared with the computation of LBP and HOG feature.

The originalities of this thesis are as follows :

In the first place, we introduce a shift histogram based on MHI representation. To the best of our knowledge, this is a new method to get shift information. Most of the existing methods use optical flow generated by Lucas-Kanade tracker or spatio-temporal gradient.

In the second place, we apply an accumulative function of the shift histogram to detect fast motion as an anomaly motion in a crowd. This approach does not necessitate motion learning. Most of the existing methods need a learning stage to learn normal and anomaly motion in a video clip.

In the third place, we introduce a function distance and a refined frame differencing as foreground extraction. Foregrounds which extracted from a static image and frame differencing images give an accurate detection of a head. They can distinguish a head and a pattern similar with a head.

In the fourth place, we introduce a histogram of transition feature as a feature to be fed into a classifier. The computation of this feature is simple and needs smaller computation time. It is suitable for real time application. The dimension of the employed feature vector is small.

Acknowledgements

It gives me great pleasure in expressing my gratitude to all those people who have supported me and had their contributions in making this thesis possible.

First and foremost, I would like to express my deepest gratitude and gratefulness to my academic supervisors, Professor Seiji Ishikawa and Associate Professor Joo Kooi Tan, who have done a great favor to my thesis. From the guiding of the research to the revision of the thesis, I have benefited greatly from their patience, encouragement and excellent guidance. What's more, I am deeply moved by their serious attitude towards academic work.

I express my sincere acknowledgements to Professor Hyoungseop Kim and Professor Seiichi Serikawa for their valuable suggestions as the co-supervisors of my thesis.

I would like to show my thankfulness to my Government, Indonesia, through DIKTI Scholarship for financial support during I study in Japan.

I would like to show my thankfulness to my current and past lab members for their kind co-operation and helpfulness in accomplishing my experiments and make my university life smooth. This three and a half years' experience of studying in Japan means a lot to me. I would like to thank all people I met here; you gave me an unforgettable memory.

Last, but by no means least, I give my special gratitude to my wife Peni Hapsari and my children Aqil, Alya and Kento for always believing me and encouraging me to follow my dreams. They help me get through the difficult times and give all the emotional support and entertainment. To Matsuoka Sensei, who always accompanied and helped me during process of pregnancy and birth of Kento. To them I dedicate this thesis.

Contents

Chapter 1
Introduction

1.1 Background

Nowadays, there is an increasing demand for security system by continuously using CCTV systems, either in a public area, particular areas, indoors or outdoors. Most current surveillance systems need a human operator to constantly monitor them. Their effectiveness and response is largely determined not by the technological capabilities but by the vigilance of the person monitoring the display [1]. To overcome these limitations of traditional surveillance methods, a major effort is under way in the computer vision and artificial intelligence community to develop automated systems. An Automatic Vision Surveillance (AVS) has become an important topic for the academic community, which aims to develop autonomous surveillance schemas to replace the traditional schemes.

Detecting human motion in a video of a surveillance system is attracting more attention due to its wide range of applications in a security system, i.e., abnormal event detection, fast motion detection, person identification, etc.; for analysis and classification, i.e., human gait characterization, gender classification, etc.; and for monitoring of an intelligent space, i.e., person counting in a dense crowd, fall detection for elderly people, etc. [2].

It should be noted that the scenes obtained from a surveillance video are usually with low resolution and most of the scenes captured by a static camera are with minimal change of the background.

1.2 Previous Work

The human motion detection can be performed in two approaches:

The first one is a flow or motion based approach. This approach analyzes human motion in a scene, especially for crowd scenes or high position monitoring. This approach is usually used to detect an abnormal motion. This approach can be divided into three methods; the first one is an optical flow based method [3,9,10,12,13,15,16,18]: This method performs feature points extraction and analyzes the optical flow generated by the Lucas-Kanade tracker. The second method is a motion vector based method [5,7,13,15]: This method divides an image into patches or blocks and calculates the similarity of each block with its neighbor, or with the same block in the previous frame. The third method is a motion pattern modeling based method [5,6,7,10,11,12,14,16]: This method divides a video into spatiotemporal volumes. It extracts a motion feature based on spatial and temporal gradients, and creates a model for motion analysis.

These existing methods have some inadequacies:

The first method relies on feature points extraction and tracking. The feature points largely rely on the texture on a human body and it is sometimes difficult to provide them due to few corners detected. Owing to the lack of feature points, it increases the false negative value.

In the second method, the partial occlusions can also cause short lived errors in the labeling [5,7], increases the computational cost [13] and performs the observing of flow directions only [15].

And in the third method, the method which based on unsupervised learning, the detected interest object or class may not have a clear semantic meaning [6]. Sometimes

it needs multiple level of measurements and affects to the computational cost [10]. The system is difficult to evaluate for individual level [11,12]. This system should be applied in extremely crowded scenes. If there is any empty space in the scene, it will be false positive [14].

The second approach for human motion detection is a head motion based approach. This approach employs a head detection method. There are two methods in detecting a head and shoulders. The first one is a shape based method [21,22,23,25,39]. This method searches a range of interest (ROI) on the foreground to fit into a template. The second one is a training of a given feature based method [24,32,40,41,42]. This method extracts the features in a ROI, and feeds them into a classifier. Some features are usually used in this method, such as local binary patterns (LBP), histograms of oriented gradients (HOG), etc.

These existing methods have some inadequacies:

The first method relies on foreground extraction and very sensitive with a pattern which is similar to a head. The computational cost of the second method depends on the number of feature dimension.

1.3 Objective of the Thesis

In order to make up for the inadequacies of the existing human motion detection based on the first approach, in this thesis, we propose a human motion detection method based on the improvement of the motion flow method. In this research, our information on motion flow is provided by the analysis of motion history image (MHI). Our method

gives more accurate as well as important information for fast motion detection in a condition where the motion is not perpendicular with a camera view.

Another improvement of the proposed human motion detection method is based on the second approach. In this thesis, we propose a human motion detection based on a training of feature based method. In this research, we introduce a histogram of transition as a novel feature. Since this feature calculates the transition between the background and the foreground, the computation is very simple and takes a short time compared with the computation of LBP and HOG feature.

1.4 Organization of the Thesis

The organization of the thesis is as follows:

In Chapter 2, we propose a method of human motion detection for detecting a fast motion by using velocity and shift histogram.

In Chapter 3, we propose a method to improve the detecting a fast motion by using a shift histogram.

In Chapter 4, we propose a method of head detection for detecting human motion by using a histogram of transition feature.

In Chapter 5, we summarize the human motion detection method which we have proposed in Chapter 2, Chapter 3 and Chapter 4. In order to prove the effectiveness of the proposed methods, we also introduce comparative fast motion detection and head motion detection methods. We carry out the comparative experiment using the same experimental videos to discuss the effectiveness of the proposed human motion detection method.

Finally, the thesis is concluded in Chapter 6.

Chapter 2

Fast Motion Detection based on a Velocity and a Shift Histogram

In this chapter, we propose a fast motion detection method based on a velocity and a shift histogram. A static camera records the environment in the coverage of a camera view, and computer begins to differentiate fast motion from normal motion. In some places, a running person or a pushing person is suspicious, which needs special attention. The proposed method requires no foreground segmentation, no motion recognition and no object detection. Its output is based on a local modeling approach.

2.1 Outline of the Proposed Method

Fast motion detection needs some velocity data. In this thesis, we provide velocity data from the trajectory of the points tracked by Lucas-Kanade tracker and a motion history image (MHI). To avoid foreground segmentation, which is a difficult task in a crowd environment, we separate an image into several blocks and evaluate them. So the detection framework is to calculate if a block (or some blocks) contains a fast motion rather than object detection.

We describe the proposed method in the following[19].

There are three main processes in the proposed method. The first one is to create a velocity histogram. A velocity histogram provides four parameters; the maximum velocity (max_V), median of velocity (med_V), mode of velocity ($mode_V$), and spreading width of the histogram (SW_V). In the second process, we define a histogram of shift numbers, which is referred to as a *shift histogram* for simplicity. A shift histogram also

provides four parameters; the maximum shift (max_S), median of shift (med_S), mode of shift ($mode_S$) and spreading width of the histogram (SW_S). Finally, we detect a fast motion in the cluttered environment by using those parameters of the histograms.

To create a velocity histogram, we do the following steps: The first step is the tracking of human motion. We extract the tracking points of moving objects in every successive frames by using Harris corner detector followed by Lucas-Kanade tracker.

The second step is creating coordinate spaces. After tracking the motion of objects, there are many tracking points belonging to a moving human and noise. To reduce the noise caused by the change of light intensity, we eliminate the static points. Then, we create coordinate spaces. Coordinate spaces are series of the tracking points from the observation frames. Each feature point in the last frame will have one coordinate space, which shows a trajectory of the point in an observation time.

We use the data in the coordinate space to calculate magnitude of velocity as [4][18]. This magnitude of velocity is used to create a velocity histogram. A velocity histogram is a set of velocity of points and the number of points at corresponding velocity.

To create a shift histogram, we do the following steps. Firstly, we create a motion history image [26]. Then we calculate shift numbers horizontally, according to the change of pixel intensity.

The quantity values to determine if a block(s) contains a fast motion are SW_V, SW_S, med_V - $mode_V$ and med_S - $mode_S$. **Figure 2.1** depicts the overview of the proposed method.

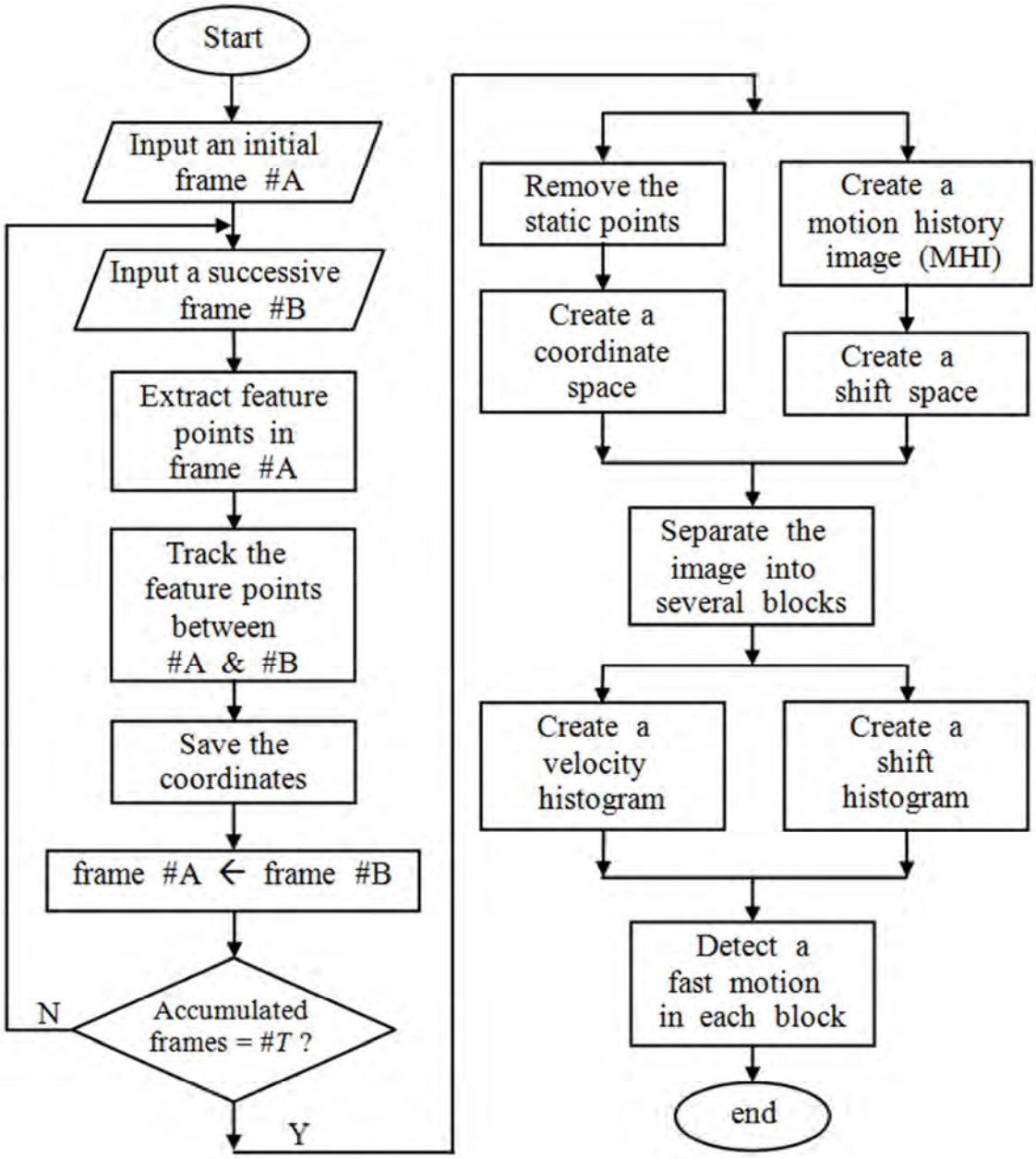


Fig. 2.1 Flowchart of the proposed fast motion detection method

2.2 Creating the Coordinate Space

For extracting and tracking feature points, we use Harris corner detector and Lucas-Kanade tracker, respectively [4][18]. Since it is difficult to detect an abnormal motion only by the motion vectors between successive frames, we need to accumulate some frames from the first frame. The accumulated frames are not overlapping with the

next accumulated frames. It means we process a set of accumulated frames to get a coordinate space and create a velocity histogram, then we proceed to the next set of accumulated frames to get another coordinate space and create another velocity histogram. The first frame is the initial frame of the accumulated frames.

The feature points are tracked over the entire frames and their location information is stored into a coordinate space [4]. Suppose that a feature point n (n=0,1,2,...,N-1) is tracked through T image frames and its position on the frame t (t=0,1,2,...,T-1) is denoted by($x_t^{(n)}$, $y_t^{(n)}$). If one or more feature points disappear during the tracking, we cannot create a coordinate space of those points. This occurs when the foreground is similar to the background. In reality, since most of human body parts are not similar to the background, enough number of feature points will remain for producing coordinate spaces. In a crowd, we may need a large number of feature points, N. In this thesis, we restrict the initial number of feature points to 1,000 feature points. Even if we reduce some static points as noise, empirically, we still have at least 90% of the initial number of feature points. These feature points are enough to build a coordinate space. Obviously, if no correspondence is found between the present frame and the next frame with a particular feature point, we cannot create a coordinate space of the point.

Since, normally, the locations of a point in the present frame and its corresponding point in the next frame are close with each other, we apply a 5×5 window to search the corresponding point in the next frame. We then define a sequence of T coordinates of a feature point by the following form [4];

$$\begin{aligned} X_0 &= \left[x_0^{(0)}, y_0^{(0)}, x_1^{(0)}, y_1^{(0)}, \ldots\ldots, x_{T-1}^{(0)}, y_{T-1}^{(0)}\right] \\ X_1 &= \left[x_0^{(1)}, y_0^{(1)}, x_1^{(1)}, y_1^{(1)}, \ldots\ldots, x_{T-1}^{(1)}, y_{T-1}^{(1)}\right] \\ &\vdots \\ X_{N-1} &= \left[x_0^{(N-1)}, y_0^{(N-1)}, x_1^{(N-1)}, y_1^{(N-1)}, \ldots\ldots, x_{T-1}^{(N-1)}, y_{T-1}^{(N-1)}\right] \end{aligned} \tag{2.1}$$

The point sequence X_n is called the coordinate space of feature point n.

2.3 Creating a Velocity Histogram

However, the movement cannot be known only by the position information of points in the coordinate space. We use the data in the coordinate space to calculate the magnitude of velocity by the following equation as in [4][18][19]. This magnitude of velocity is used to create a velocity histogram.

$$\begin{aligned} V_0 &= \sqrt{\left(x_{T-1}^{(0)} - x_0^{(0)}\right)^2 + \left(y_{T-1}^{(0)} - y_0^{(0)}\right)^2} \\ V_1 &= \sqrt{\left(x_{T-1}^{(1)} - x_0^{(1)}\right)^2 + \left(y_{T-1}^{(1)} - y_0^{(1)}\right)^2} \\ &\vdots \\ V_{N-1} &= \sqrt{\left(x_{T-1}^{(N-1)} - x_0^{(N-1)}\right)^2 + \left(y_{T-1}^{(N-1)} - y_0^{(N-1)}\right)^2} \end{aligned} \tag{2.2}$$

It is noted that ’velocity’ needs division by a time factor in Eq.(2.2), but, since the number of frames T is a constant, V_n is referred to as velocity.

A velocity histogram is a set of velocity of points and the number of points at corresponding velocity [19]. Based on Eq. (2.2), we get small real numbers of velocity. However, a velocity histogram needs a discrete abscissa. To get a discrete abscissa, the small real number is converted to the nearest integer, i.e. 0-0.4 are grouped into0, and 0.5-1.4 are grouped into 1, etc. A velocity histogram and its property are shown in **Figure 2.2**. A velocity histogram provides four parameters; the maximum velocity (max_V), median of velocity (med_V), mode of velocity ($mode_V$), and spreading width of the histogram(SW_V), where, med_V is (max_V - min_V)/2; $mode_V$ is the value of velocity which has the maximum number of points; and SW_V is max_V - min_V.

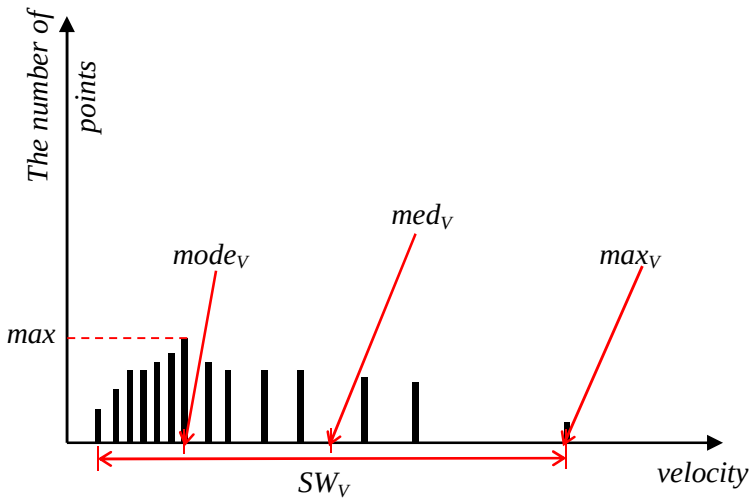


Fig. 2.2 A velocity histogram and its property

2.4 Creating a Motion History Image (MHI)

In the proposed method, an idea of object shift is employed. To calculate the object shift, we use a Motion History Image (MHI) [26] which is an established motion representation strategy in the motion analysis field.

Motion History Image is a view-specific representation of movement, where movement is defined as the motion over time. MHI, as the name implies, keeps track of a motion history, i.e., representing how an object interested is moving along a certain period of time. The MHI is a frame-based temporal template for human motion. In generating a MHI, temporal information is specified by the pixel intensity. Let $H_\tau(x,y,t)$ be the pixel intensity function of the temporal history of motion at a particular point (x,y) and at time t on an image. Then the function $H_\tau(x,y,t)$ is defined as given in Eq.(2.3).

$$H_\tau(x,y,t) = \begin{cases} \tau & \text{if } D(x,y,t)=1 \\ \max(0, H(x,y,t-1)-1) & \text{otherwise} \end{cases} \tag{2.3}$$

Here, $D(x,y,t)$ is a difference image in a binarized form constructed from successive frame difference. In this research, the threshold value to get $D(x,y,t) = 1$ is 30, which is experimentally chosen.

The function $H_\tau(x,y,t)$ returns a scalar value. According to the function, the more recently moving pixels are brighter than the past moving pixels in the generated MHI. In Eq.(2.1), τ is taken as the temporal extent which is critical to define. But for the flexibility of the value of τ, it can be taken as the maximum gray level 255 or the maximum number of frames defining a motion. An example of a MHI is shown in **Figure 2.3**.

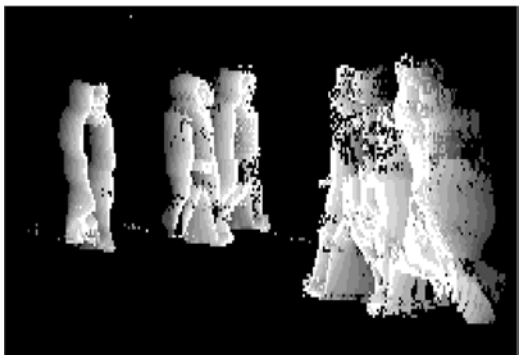


Fig. 2.3 An example of a MHI. A MHI is a gray scale image that describes a motion history.

2.5 Creating a Shift Space and a Shift Histogram

A shift space does not represent motion direction. It only represents horizontal motion of a point. To get the information of points motion from a MHI, we need to convert it into a shift space. A shift space is a way of numerical representation of points motion from MHI information: When a point moves p pixels, its numerical representation is p. Here p is a positive integer. Since we assume two motion directions in an image frame, to the left and to the right, we have to scan MHI twice in each direction to get intermediate grayscale images, $I_{L(x,y)}$ and $I_{R(x,y)}$, with the initial pixel

values of these images 0. Here $I_{L(x,y)}$ and $I_{R(x,y)}$ are the left scan image and the right scan image, respectively. The first scan is to one direction: We put an increasing digit, if the gray scale value at the neighborhood is more than the current pixel. The second scan is in opposite direction: We put the maximum digit between the current pixel and the neighborhood. In this way, an intermediate grayscale image is produced. Finally, the obtained two intermediate grayscale images are merged into a single image, a shift space $S_{x,y}$, by maximization operation. These procedures are given by Eqs.(2.4)-(2.8).

In case that the initial scan proceeds from the left to the right, the first scan is defined as follows [19];

$$I_{L(x,y)} = \begin{cases} I_{L(x-1,y)} + 1 & \text{if } H_{\tau(x,y)} > H_{\tau(x-1,y)} \\ I_{L(x-1,y)} & \text{if } H_{\tau(x,y)} = H_{\tau(x-1,y)} \\ 0 & \text{otherwise} \end{cases} \tag{2.4}$$

The second scan from the right to the left is defined by the following;

$$I_{L(x,y)} = \begin{cases} \max(I_{L(x,y)}, I_{L(x+1,y)}) & \text{if } I_{L(x,y)} < I_{L(x+1,y)} \text{ and } I_{L(x,y)} \neq 0 \\ I_{L(x,y)} & \text{if } I_{L(x,y)} > I_{L(x+1,y)} \\ 0 & \text{otherwise} \end{cases} \tag{2.5}$$

On the other hand, in case that the initial scan proceeds from the right to the left, the first scan is defined as follows;

$$I_{R(x,y)} = \begin{cases} I_{R(x+1,y)} + 1 & \text{if } H_{\tau(x,y)} > H_{\tau(x+1,y)} \\ I_{R(x+1,y)} & \text{if } H_{\tau(x,y)} = H_{\tau(x+1,y)} \\ 0 & \text{otherwise} \end{cases} \tag{2.6}$$

The second scan from the left to the right is defined by the following;

$$I_{R(x,y)} = \begin{cases} \max(I_{R(x,y)}, I_{R(x-1,y)}) & \text{if } I_{R(x,y)} < I_{R(x-1,y)} \text{ and } I_{R(x,y)} \neq 0 \\ I_{R(x,y)} & \text{if } I_{R(x,y)} > I_{R(x-1,y)} \\ 0 & \text{otherwise} \end{cases} \tag{2.7}$$

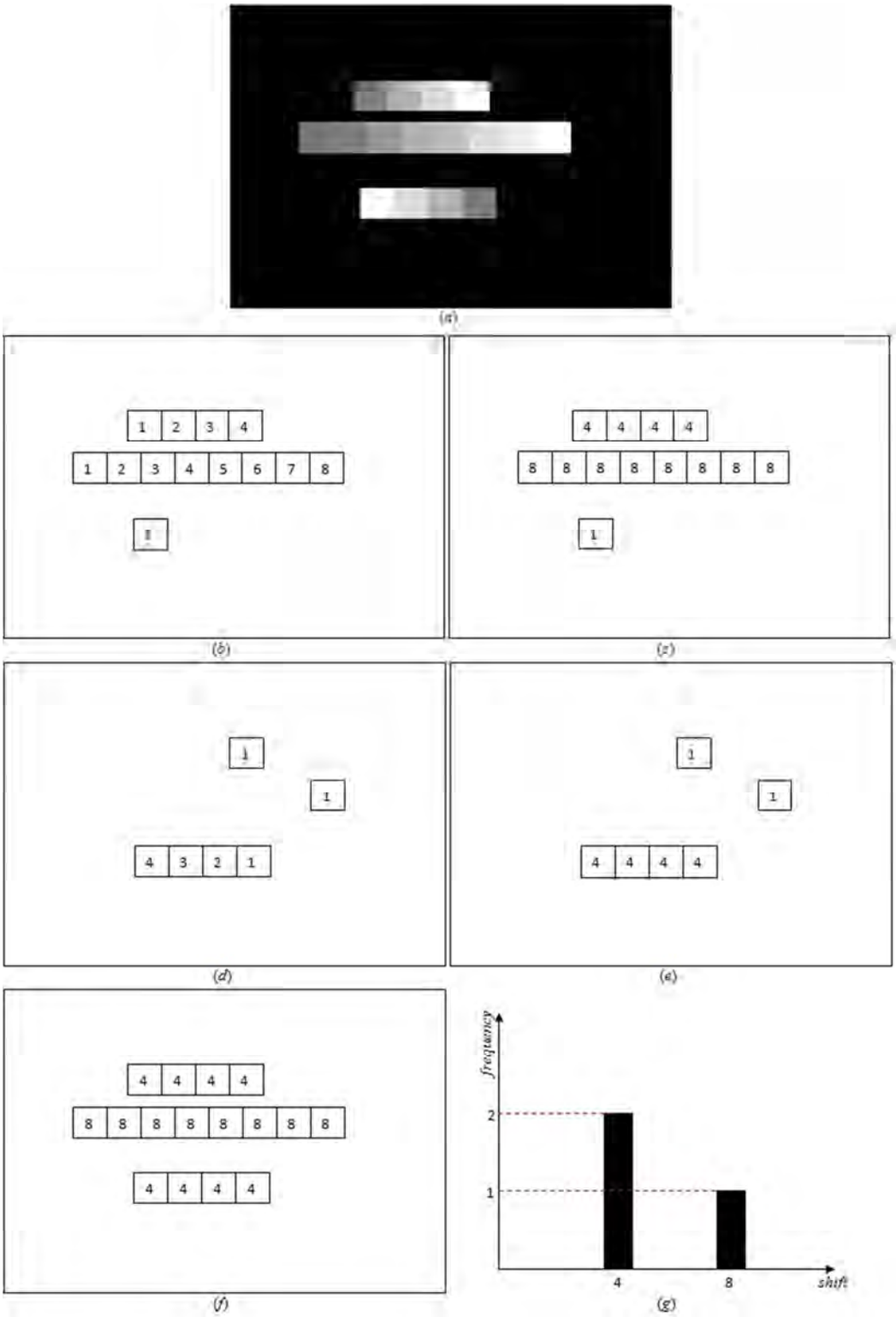


Fig.2.4 MHI, a corresponding shift space and a shift histogram: (*a*) A MHI, (*b*) $I_{L(x,y)}$ obtained after the first scan, (*c*) $I_{L(x,y)}$ after the second scan, (*d*) $I_{R(x,y)}$ after the first scan, (*e*) $I_{R(x,y)}$ after the second scan, (*f*) a shift space, $S_{x,y}$, (*g*) a corresponding shift histogram of (*f*)

Finally, from (2.5) and (2.7), we combine these images to yield a shift space, $S_{x,y}$.

$$S_{x,y} = \max(I_{R(x,y)}, I_{L(x,y)}) \tag{2.8}$$

Figure 2.4 depicts the calculation of a shift space of a given MHI.

Based on the data of the shift space, $S_{x,y}$, we calculate the frequency of occurrence of shift numbers. Different from a velocity histogram, a shift histogram doesn't need the conversion from real number to integer number for its abscissa, because a shift space uses integer numbers. Then we create a shift histogram and get max_S, med_S, $mode_S$ and SW_S as in **Figure 2.5**, where, med_S is $(max_S - min_S)/2$; $mode_S$ is the value of shift which has the maximum number of points; and SW_S is $max_S - min_S$.

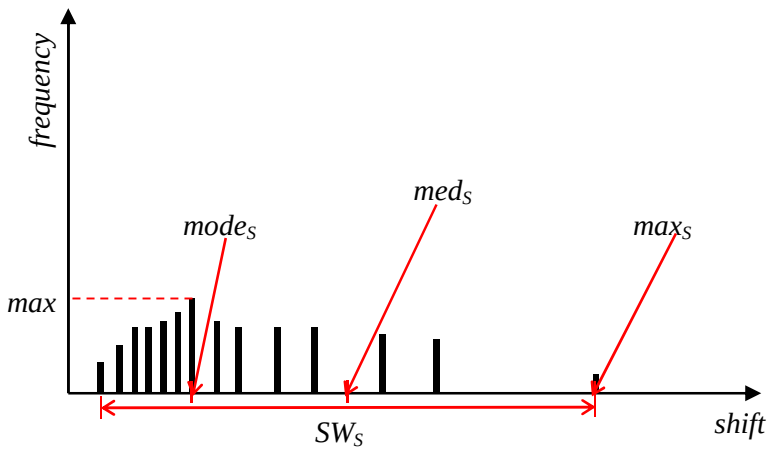


Fig. 2.5 A shift histogram and its property

2.6 Detecting a Fast Motion

Instead of object detection directly, we separate an image into several blocks, as shown in **Figure 2.6**. Then, based on the parameters of a velocity and a shift histogram on each block, we determine whether or not a block contains a fast motion.

Let us denote, for block B, the judgment index of $SW_{V,B}$ by $p_{1,B}$, the judgment index of $SW_{S,B}$ by $p_{2,B}$,the judgment index of $med_{V,B}$ - $mode_{V,B}$ by $p_{3,B}$, and the judgment index

of $med_{S,B}$ - $mode_{S,B}$ by $p_{4,B}$, where $B \in \{0, ..., 15\}$ in the performed experiment. Then we have,

$$p_{1,B} = \begin{cases} 1 & if\ \ SW_{V,B} > th_1 \\ 0 & otherwise \end{cases} \tag{2.9}$$

$$p_{2,B} = \begin{cases} 1 & if\ \ SW_{S,B} > th_2 \\ 0 & otherwise \end{cases} \tag{2.10}$$

$$p_{3,B} = \begin{cases} 1 & if\ \ med_{V,B} - mode_{V,B} > th_3 \\ 0 & otherwise \end{cases} \tag{2.11}$$

$$p_{4,B} = \begin{cases} 1 & if\ \ med_{S,B} - mode_{S,B} > th_4 \\ 0 & otherwise \end{cases} \tag{2.12}$$

The threshold values, th_1, th_2, th_3, and th_4, are determined by the observation of a velocity and a shift histogram for normal and abnormal conditions for a few minutes in a video. Change of these thresholds will cause a running motion detected as a normal motion (false negative).We also denote the total of the above parameters by $p_{T,B}$ defined by

$$p_{T,B} = \sum_{i=1}^{4} p_{i,B} \tag{2.13}$$

Then block B is judged as containing fast motion, if $p_{T,B} \geq 3$ is satisfied.

Fig. 2.6 An image frame and its division into blocks

2.7 Experimental Results

2.7.1 Experimental Environment

In this section, we examine the proposed fast motion detection method using three videos. Video 1 is captured in an artificial scene inside the campus: In this scene, a person is running in the same direction as walking people. Video 2 is the same location as video 1; but a person is running in an opposite direction against walking people. Video 3 is a similar scene with Video 1, but the camera position is higher than the case of Video 1.

In the proposed fast motion detection method, an object having fast motion is defined as a running person who has a speed greater than that of walking people. According to this definition, a correct object in Video 1, 2 and 3 is a running person.

The configurations of the PC used in the experiments are shown in **Table 2.1**.

Table 2.1 Configurations of the PC used in the experiments.

OS	Microsoft Windows 7 ultimate
CPU	Intel® Core™ CPU 870 @2.93 GHz.
Memory	8.0 GB
Software Tool	Microsoft Visual Studio 2010.

2.7.2 Detection of Fast Motion

The parameters used in this experiment are shown in **Table 2.2**.

Table 2.2 Parameters used in the experiment.

Parameters	**Values**
Accumulated frames, T	10
The number of blocks, B	16

Table 2.2 Continued.

Parameters	**Values**
Frame size	Video 1 : 320 × 160 pixels
	Video 2 : 320 × 160 pixels
	Video 3 : 320 × 160 pixels
Threshold 1, th_1	Video 1 = 100
	Video 2 = 100
	Video 3 = 90
Threshold2, th_2	Video 1 = 30
	Video 2 = 30
	Video 3 = 10
Threshold3, th_3	Video 1 = 13
	Video 2 = 13
	Video 3 = 13
Threshold4, th_4	Video 1 = 6
	Video 2 = 6
	Video 3 = 6

2.7.3 Comparative Experiments

We perform comparative experiments by using a velocity histogram and then clustering the direction of motion [18], a velocity histogram based method and a shift histogram based method. In [18] they extracted feature points by using Harris corner detector and tracked them by using Lucas-Kanade tracker. They created a coordinate space, then calculated magnitude and orientation of motion velocities. A velocity

histogram is built from these magnitudes of motion velocities. Fast motion detection is determined by clustering the velocity in the histogram.

In a velocity histogram based method, we build a velocity histogram as described in Section 2.3. For detecting of fast motion, we use parameters of velocity, i.e. Eqs. (2.9) and (2.11).

In a shift histogram based method, we build a shift histogram as described in Section 2.5, and detect fast motion by using Eqs. (2.10) and (2.12).

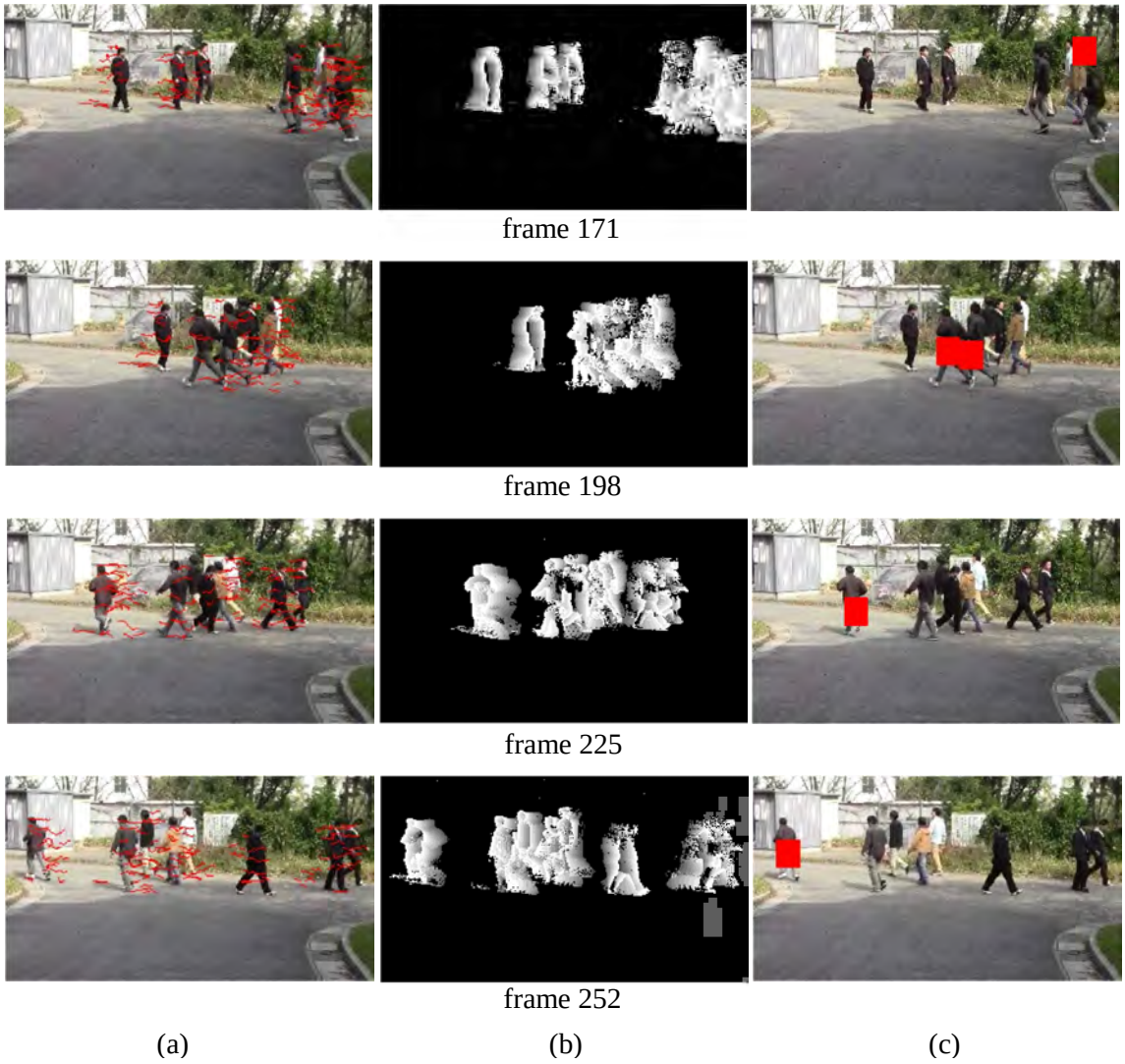


Fig. 2.7 The result of fast motion detection (Video 1): (a) Optical flow image, (b) MHI, (c) the result of fast motion detection

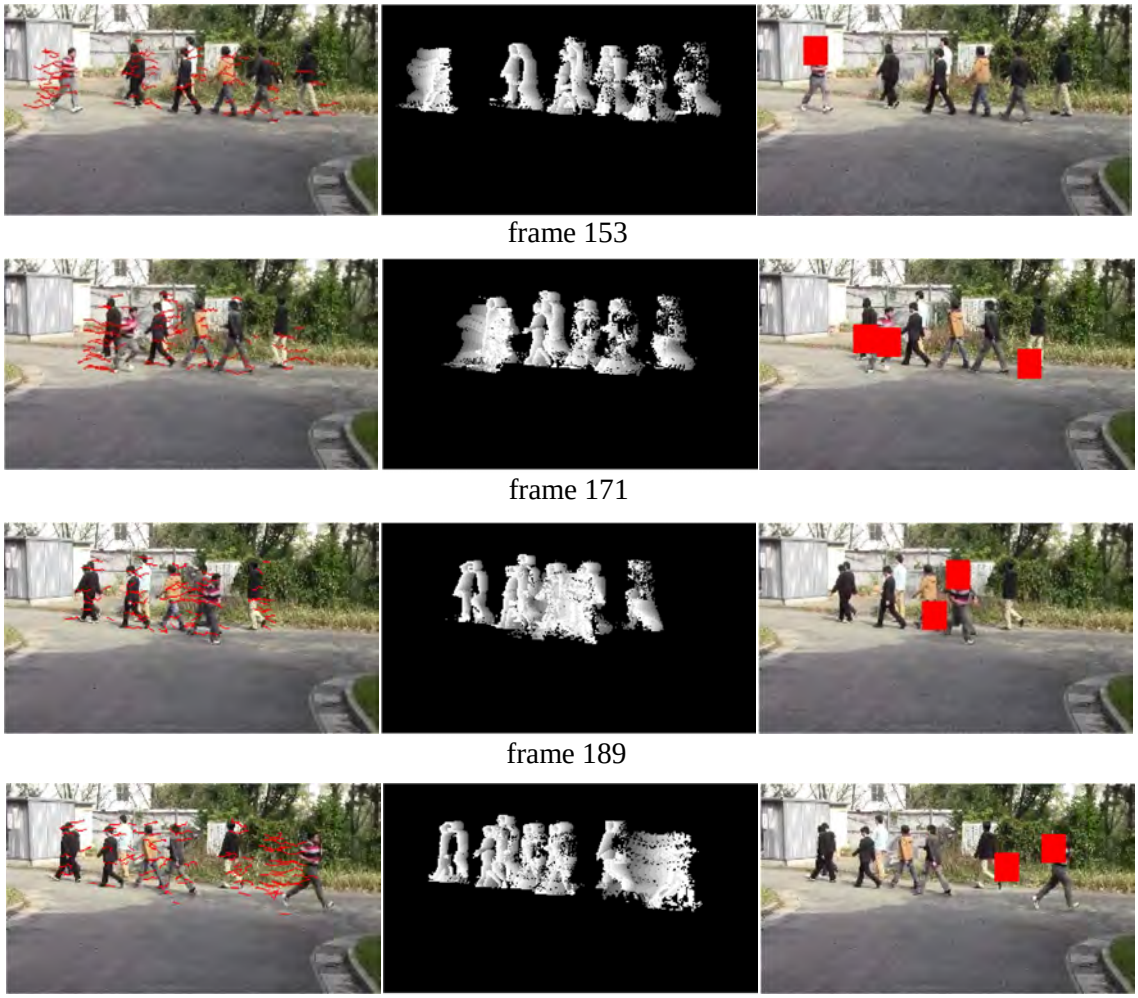


Fig. 2.8 The result of fast motion detection (Video 2): (a) Optical flow image, (b) MHI, (c) the result of fast motion detection

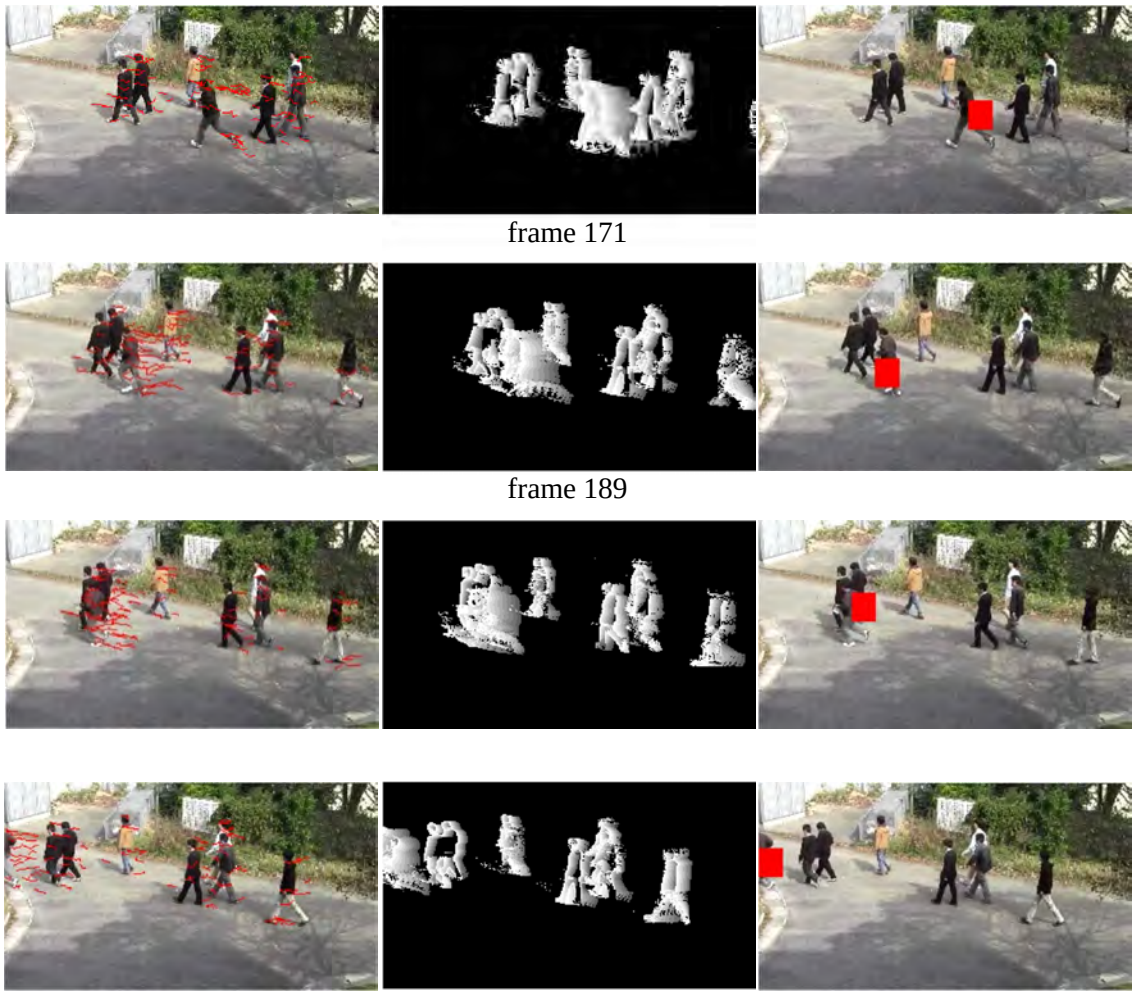


Fig. 2.9 The result of fast motion detection (Video 3): (a) Optical flow image, (b) MHI, (c) the result of fast motion detection

2.8 Evaluation

In order to evaluate the effectiveness of the proposed fast motion detection method, let us define *recall*, *precision*, *FPR* and *F* measure by

$$recall = \frac{TP}{TP + FN} \times 100\% \tag{2.14}$$

$$precision = \frac{TP}{TP + FP} \times 100\% \tag{2.15}$$

$$FPR = \frac{FP}{FP + TN} \times 100\% \tag{2.16}$$

$$F = \frac{2 \times recall \times precision}{recall + precision} \tag{2.17}$$

Here

TP : fast motion is detected as fast motion,

FN : fast motion is detected as normal motion,

FP : normal motion is detected as fast motion,

TN : normal motion is detected as normal motion.

The performance of the proposed method is given numerically in **Table 2.3** and **Table 2.4**. Table 2.3 shows the evaluation results of different scenes, whereas Table 2.4 shows the evaluation results when we use different methods. When using the method [18], the FPR is high, because some normal motions close to a camera are detected as abnormal motion, and some running motions away from a camera are detected as normal motion. By using the proposed method, the system reduces the FPR, rather than using one histogram only.

Table 2.3 Evaluation of the performance on the three videos

Video	**recall (%)**	**precision (%)**	**FPR (%)**	**F (%)**
1	83.7	80.5	2.3	82.1
2	83.4	81.3	1.7	82.3
3	83.3	81.2	1.8	82.2

Table 2.4 Evaluation of the comparative experiment for video 1

Method	recall (%)	precision (%)	FPR (%)	F (%)
velocity histogram then clustering the direction of motion [18]	78.5	33.3	6.2	46.7
velocity histogram	60.3	50.1	2.5	54.7
shift histogram	88.4	34.6	7.9	49.7
velocity histogram and shift histogram **(the proposed method)**	**83.7**	**80.5**	**2.3**	**82.1**

2.9 Discussion and Conclusion

In this chapter, we proposed a method to detect a fast (abnormal) motion in a crowd environment using a velocity histogram and a shift histogram. The method assumes that a camera view is not perpendicular to motion directions.

A velocity histogram is easy to obtain, as it calculates the magnitude of velocity defined by Eq. (2.2). However its drawback is that it relies on feature point extraction and tracking. The feature points largely rely on the texture on a human body and it is sometimes difficult to provide them due to few corners detected. Owing to the lack of feature points, it may lose high velocity data and increase False Negative values. On the other hand, a shift histogram has many shift data extracted from all the contour pixels on moving objects in an image. Unlike the velocity histogram, it won’t lose high velocity data. The drawback of the shift histogram is that, since some shifts remain behind the interest objects in a shift space, it causes increase of False Positive values. By using these two histograms, however, the system can reduce False Negative and False Positive values and increase Recall and Precision values.

Unlike existent methods, in the proposed method, the camera location is not very high from the ground. Instead of object detection and to avoid foreground segmentation, the proposed method calculates if a block(s) of image contains a fast motion. An evaluation indices, given by Eqs.(2.9) - (2.13), to determine if a block(s) contains a fast motion is based on a velocity histogram and a shift histogram.

There are some parts of a body which move faster than other parts, such as hands and legs. These partial motions may affect negatively in evaluating False Positives.

The proposed method has some advantages over the existing fast motion detection methods. In the first place, the proposed method can detect arbitrary fast motion. This is helpful because fast motion in a normal environment needs special attention for a security system. In the second place, the proposed method can detect fast motion in a crowd with condition that a camera location is lower than former methods, and a camera direction and human motion flow don’t have to be perpendicular with each other. This will produce more general and realistic cases with respect to abnormal motion detection. In the third place, the proposed method doesn’t need a learning algorithm for normal motion and a special calibration for the coverage of a camera view. It only needs an observation of normal motion in a few minutes of video running.

Chapter 3

Fast Motion Detection based on a Shift Histogram

In this chapter, we propose fast motion detection based on a shift histogram. The static camera records the environment in the coverage of camera view, and the computer begins to differentiate fast motion from normal motion. In some places, a running person or a pushing person is suspicious, which needs special attention. The proposed method requires no foreground segmentation, no motion recognition and no object detection. Its output is based on local modeling approach.

3.1 Outline of the Proposed Method

Fast motion detection needs some velocity data. In this thesis, we provide velocity data from a motion history image (MHI). To avoid foreground segmentation, which is a difficult task in a crowd environment, we separate an image into several blocks and evaluate them. So the detection framework is to calculate if a block (or some blocks) contains a fast motion rather than object detection.

We define a histogram of shift numbers, which is referred to as a shift histogram for simplicity. To create a shift histogram, we do the following steps. Firstly, we create a motion history image [26]. Then we calculate shift numbers horizontally, according to the change of pixel intensity.

Then we calculate an accumulation of histogram to detect a fast motion in the cluttered environment.

Figure 3.1 shows the flowchart of the proposed fast motion detection.

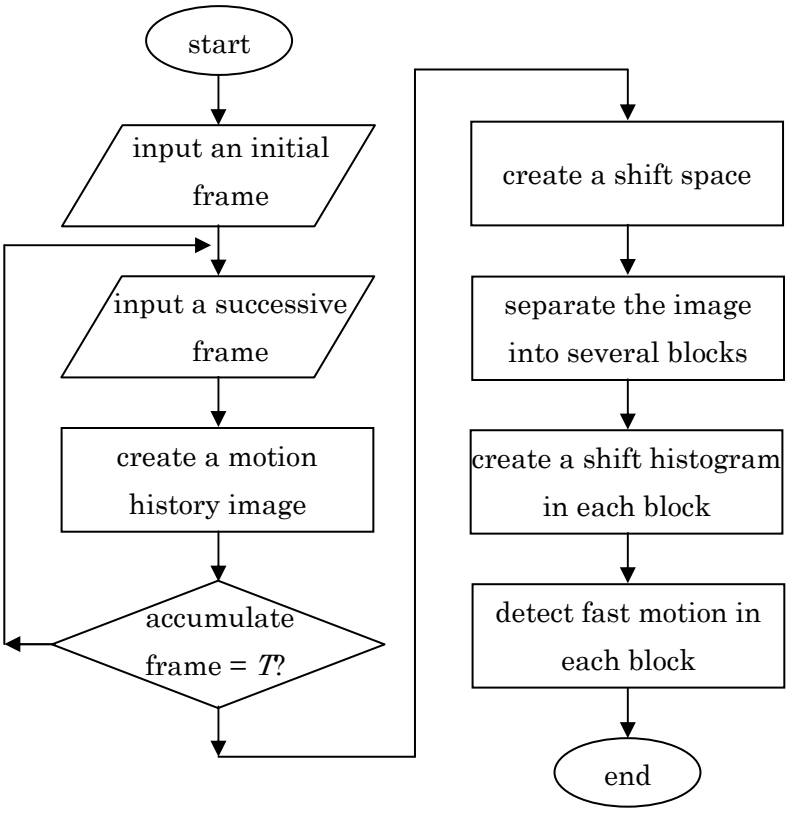


Fig. 3.1 Flowchart of the proposed fast motion detection method

3.2 Creating a Motion History Image (MHI)

In the proposed method, object shift, is employed. To calculate the object shift, we use a Motion History Image (MHI) [26] which is an established motion representation strategy in the motion analysis field.

Creating a motion history image (MHI) is explained in Section 2.4.

3.3 Creating a Shift Space and a Shift Histogram

A shift space does not represent motion direction. It only represents horizontal motion of a point. To get the information of points motion from a MHI, we need to convert it into a shift space. Creating a shift space and a shift histogram are explained in Section 2.5.

3.4 Detecting a Fast Motion

With the shift histogram which was introduced in **Figure 2.5**, the amplitude of the shift of a fast motion is higher than the normal one. From a shift histogram belonging to B^{th} block, let us define an accumulation function in that block, Σ_B, as follows;

$$\Sigma_B = \sum_{i \in shift} shift_{B,i} \times freq_{B,i} \tag{3.1}$$

Then, we define a motion as,

$$\text{motion} = \begin{cases} \text{fast} & \text{if } \Sigma_B > th \\ \text{normal} & \text{otherwise} \end{cases} \tag{3.2}$$

where a threshold, *th*, is determined by the observation of motions in a normal condition for a few minutes in a video.

3.5 Experimental Results

3.5.1 Experimental Environment

In this section, we examine the proposed fast motion detection method using three videos. Video 1 is captured in the artificial scene inside the campus:In this scene,a running person is the same direction as walking people. Video 2 is the same location as video 1; but a running person is in an opposite direction against walking people. Video 3 is an indoor scene in a bar captured by CCTV [27].

In the proposed fast motion detection method, fast motions are defined as arbitrary objects which have speed greater than walking motions. According to this definition, correct object in video 1 and 2 is a running person; correct objects in video 3 are pushing or fighting people and a thrownchair.

The configurations of the PC used in the experiments are shown in **Table 3.1**.

Table 3.1 Configurations of the PC used in the experiments.

OS	Microsoft Windows 7 ultimate
CPU	Intel® Core™ CPU 870 @2.93 GHz.
Memory	8.0 GB
Software Tool	Microsoft Visual Studio 2010.

3.5.2 Detection of Fast Motion

The parameters of this experiment are shown in **Table 3.2**.

Table 3.2 Parameters of this experiment.

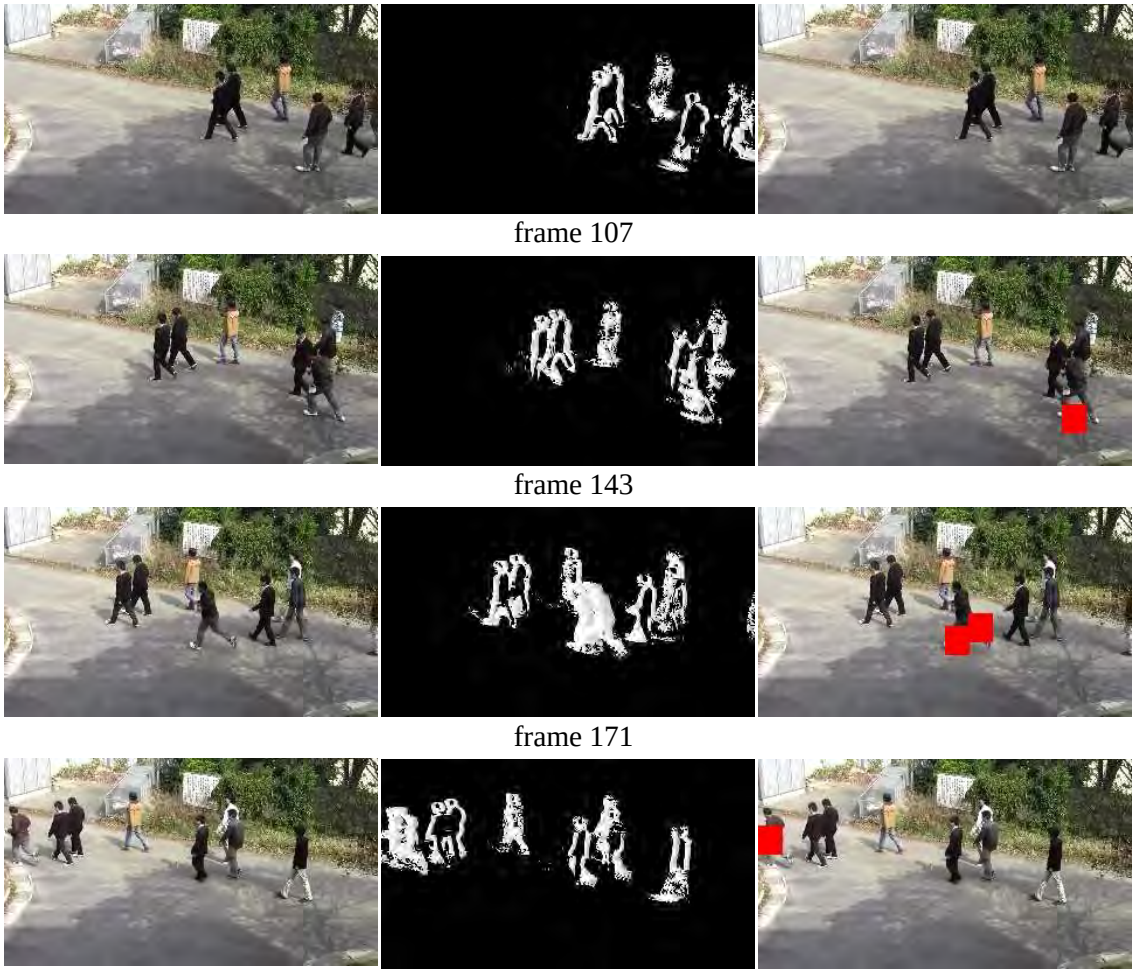
Parameters	**Values**
Accumulated frames, *T*	4
The number of blocks, *B*	16
Frame size	Video 1 : 320 × 160 pixels
	Video 2 : 320 × 160 pixels
	Video 3 : 426 × 240 pixels
Motion threshold, *th*	Video 1 : 4600
	Video 2 : 4600
	Video 3 : 24000

3.5.3 Comparative Experiments

We perform comparative experiments by using velocity histogram method [4][18],combination of the velocity and the shift histogram method [19][20] and the accumulation function based on velocity histogram. In [4] and [18], they extracted feature points by using Harris corner detector and tracked them by using

Lucas-Kanadetracker. They created a coordinate space, then calculated magnitude and orientation of motion velocities. A velocity histogram is built from these magnitudes of motion velocities. Fast motion detection is determined by clustering the velocity in thehistogram.

In [19] and [20], they built a velocity histogram as in [18] and a shift histogram as was explained in Section 2.3. In these histograms they defined parameters of velocity and shift, such as *spreading width*, *maximum*, *mode*, and *median*. They used these parameters to determine whether or not the motion in an image frame is fast.



(a) (b) (c)

Fig 3.2 The results of fast motion detection (Video 1): (a) Input images, (b) MHI images, (c) the results of fast motion detection.

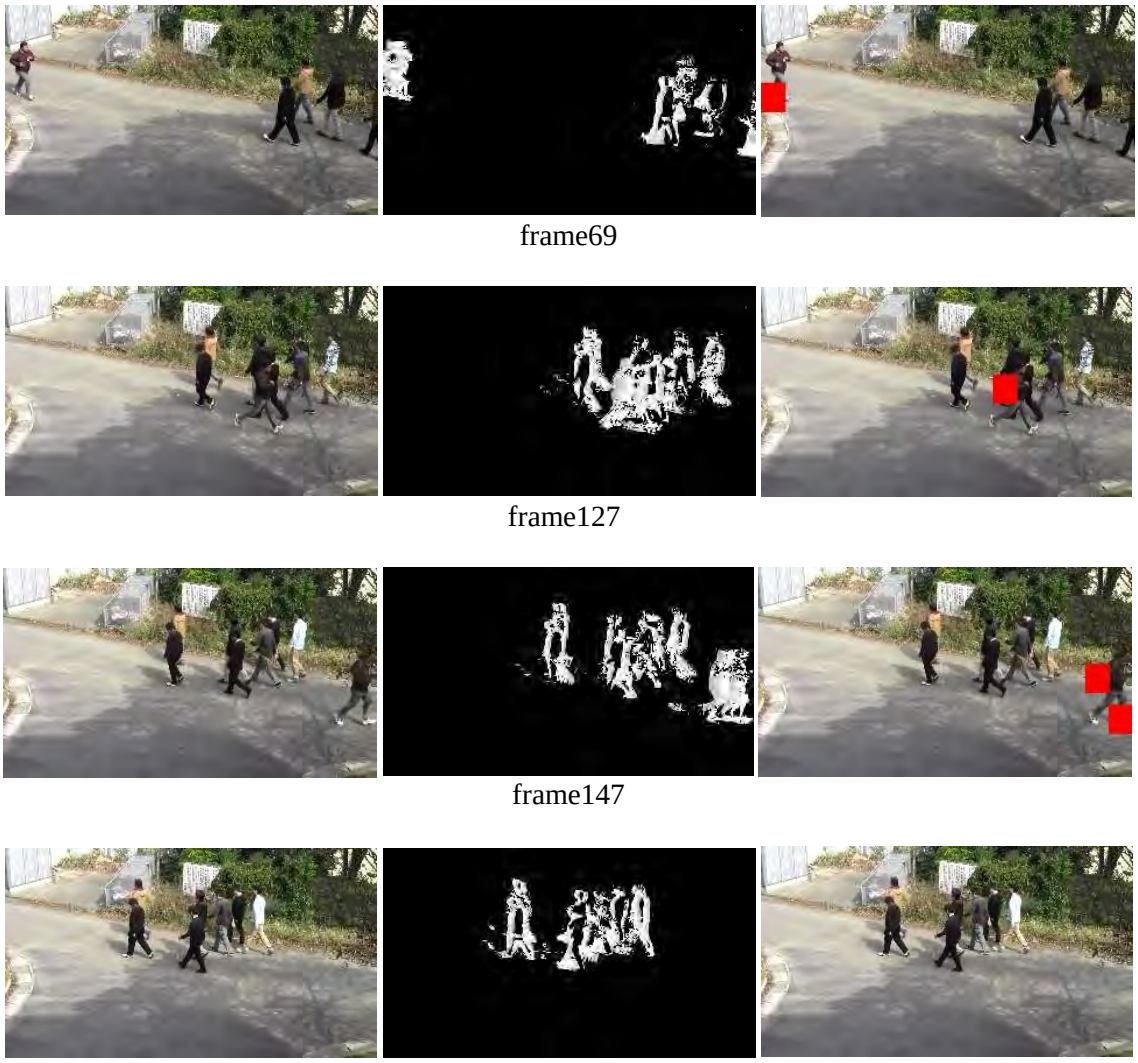


Fig 3.3 The results of fast motion detection (Video 2): (a) Input images, (b) MHI images, (c) the results of fast motion detection.

(a) (b) (c)

Fig 3.4 The results of fast motion detection (Video 3): (a) Input images, (b) MHI images, (c) the results of fast motion detection.

3.6 Evaluation

In order to evaluate the effectiveness of the proposed fast motion detection method, let us define *recall*, *precision* and *FPR* by

$$recall = \frac{TP}{TP + FN} \times 100\% \tag{3.3}$$

$$precision = \frac{TP}{TP + FP} \times 100\% \tag{3.4}$$

$$FPR = \frac{FP}{FP + TN} \times 100\% \tag{3.5}$$

$$F = \frac{2 \times recall \times precision}{recall + precision} \tag{3.6}$$

Here

TP : fast motion is detected as fast motion,

FN : fast motion is detected as normal motion,

FP : normal motion is detected as fast motion,

TN : normal motion is detected as normal motion.

Then the performance of the proposed method is given numerically in **Table 3.3** and **Table 3.4**. Table 3.3 shows the evaluation results of different scenes,whereas Table 3.4 shows the evaluation results when we use different methods. When using the method [18], the FPR is high, because some normal motions close to a camera are detected as abnormal motion, and some running motions away from a camera are detected as normal motion. By using the combination of a velocity histogram and a shift histogram [19][[20], the system reduces the FPR, rather than using one histogram only. However, they need four threshold values which are sensitive in their determination.

Table 3.3 Evaluation of the performance on the three videos

video	**recall (%)**	**precision (%)**	**FPR (%)**	***F* (%)**	**execution time (ms)**
1	93.3	96.5	0.3	94.8	78.51
2	93.3	96.5	0.3	94.8	78.51
3	82.1	85.7	0.4	83.8	367.28

Table 3.4 Evaluation of the comparative experiment for video 1

method	**recall (%)**	**precision (%)**	**FPR (%)**	**F (%)**	**execution time (ms)**
velocity histogram then clustering the direction of motion [18]	78.5	33.3	6.2	46.7	145.33
velocity histogram and shift histogram [19][20]	83.3	81.2	1.8	82.2	384.25
accumulation function based on velocity histogram	83.3	76.9	1.89	79.9	142.91
accumulation function based on shift histogram (**the proposed method**).	**93.3**	**96.5**	**0.3**	**94.8**	**78.51**

3.7 Discussion and Conclusion

In this chapter, we proposed a method to detect a fast (abnormal) motion in a crowd environment using a shift histogram. The method assumes that a camera view is not perpendicular to motion directions.

A velocity histogram is easy to obtain, however its drawback is that it relies on feature point extraction and tracking [18][19][20]. The feature points largely rely on the texture on a human body and it is sometimes difficult to provide them due to few corners detected. Owing to the lack of feature points, it loses high velocity data and increases False Negative values. On the other hand, a shift histogram has many shift data extracted from all the contour pixels on moving objects in an image. Unlike the velocity histogram, it will not lose high velocity data. The drawback of the shift

histogram is that, since shift numbers remain behind interested objects in a shift space, it causes increase of False Positive values.

Instead of object detection and to avoid foreground segmentation, the proposed method calculates if a block(s) of image contains a fast motion. This method effectively detects a fast motion among normal motion in a crowd.

The proposed method has some advantages over the existing fast motion detection methods. In the first place, the proposed method uses an accumulation function of a shift histogram. This realizes a smaller computation time. It is also advantageous for achieving real-time processing. In the second place, the proposed method can detect arbitrary fast motion. This is helpful because fast motion in a normal environment needs special attention for a security system. In the third place, the proposed method can detect fast motion in a crowd with condition that a camera direction and human motion flow don’t have to be perpendicular with each other. This will produce more general and realistic cases with respect to abnormal motion detection. In the fourth place, the proposed method doesn’t need a learning algorithm for normal motion and a special calibration for coverage of camera view. It only needs an observation of normal motion in a few minutes of video running.

Chapter 4

Head Motion Detection based on a Histogram of Transition

In this chapter, we propose a method of head motion detection based on a histogram of transition. A static camera records the environment in the coverage of a camera view, and the computer begins to detect a head and shoulders. In some places, a person or some people move in the coverage of a camera view and the motion sometimes needs a special attention. Especially in an intelligent room, his/her motion represents his/her activity, and occasionally we may have to judge if the activity is normal or abnormal.

4.1 Outline of the Proposed Method

We describe the proposed method in the following. There are two main processes in the proposed method. The first process is the detection of a human head candidate. After we perform frame differencing as in [21,28], we scan the area which contains a motion by a block and calculate motion pixels in the block. If the number of motion pixels within the block is greater than a threshold, we perform the next process.

The second process is the recognition of a head. We perform foreground refinement by hole filling [29] and foreground extraction by using variable reference coordinates of foreground labels. The resultant images are fed into the calculation of transition feature. We modify the calculation of the transition feature in [30,31]. Finally we recognize a human head by using a SVM classifier.

Figure 4.1 depicts the overview of the proposed system. “A” is a connection node in the figure.

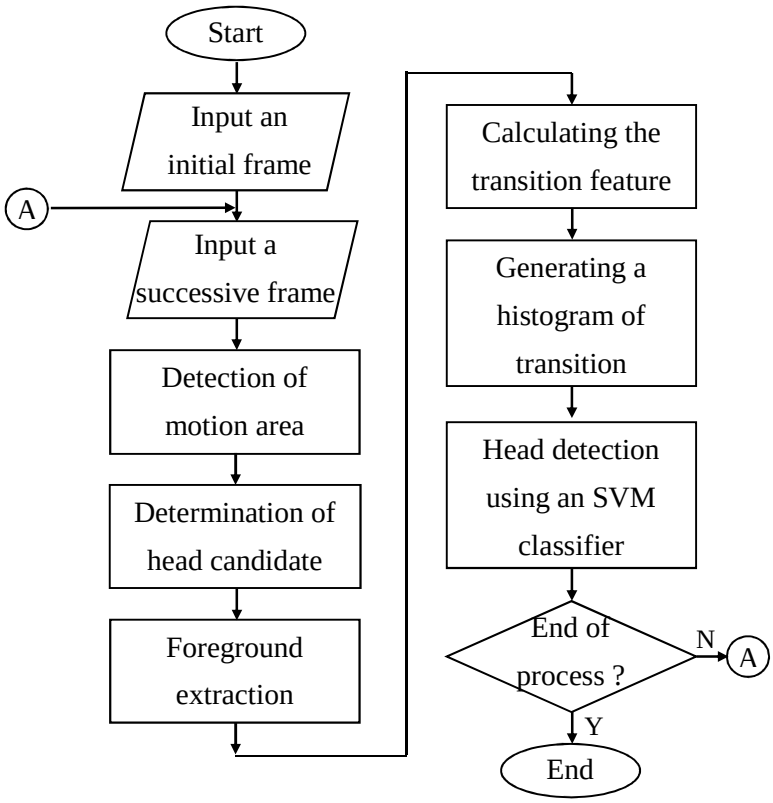


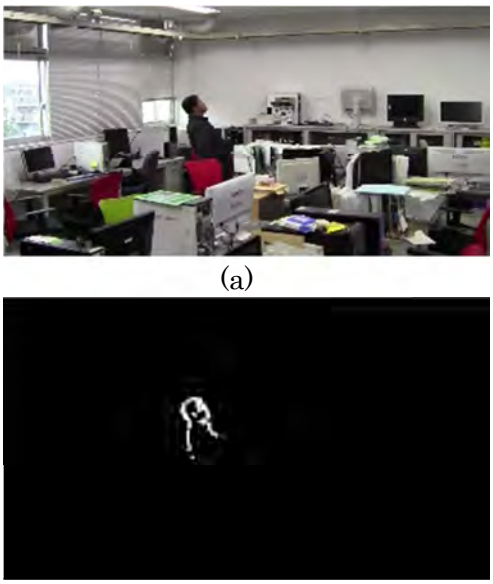
Fig. 4.1 Overview of the proposed head motion detection method

4.2 Motion Detection

Occasionally the most naive approach is found to yield adequate results. This is the case with motion detection. Simple frame differencing is used to find pixels corresponding to moving objects. If a pixel's intensity, $I(x,y,t)$, changes significantly from one frame to the next, it is considered moving [21,28].

$$I_t(x,y,t) = \frac{\partial}{\partial t} I(x,y,t) = I(x,y,t) - I(x,y,t-1) \tag{4.1}$$

This approach requires little computation and has minimal latency. First, we change the color image into gray image for both image frame $I(x,y,t\text{-}1)$ and $I(x,y,t)$. Then we apply Eq. (4.1). The result is still a gray image, then, with a gray level threshold, we change the gray image into a binary image. The results of this procedure can be seen in **Figure 4.2.**We call the white pixels as a *motion pattern*.



(a)

(b)

Fig. 4.2 Motion detection using frame differencing: (a) A raw frame of video, (b) the result of the motion detection.

4.3 Determination of Head Candidate

To determine a head candidate, we scan the area of motion by using a block having $m \times n$ pixel size. Area of motion is a range of interest (ROI) in a frame which contains a motion pattern only. An $m \times n$ pixel size is an image size for head detection. A head candidate is determined by the following equation;

$$\text{head\_candidate} = \begin{cases} \textit{true} & \text{if } \textit{motion pattern} > \textit{th} \\ \textit{false} & \text{otherwise} \end{cases} \tag{4.2}$$

If a head candidate is true, then we keep this block's location in a binary image as ROI_{HBin} and in an original image as ROI_{HCol}. Then we perform the next process, i.e., foreground extraction for ROI_{HCol} and foreground refinement by hole filling for ROI_{HBin}.

4.4 Foreground Extraction

Our transition feature relies on foreground extraction. As the foreground extraction is not our main topic, however, we explain it not in detail.

A simple algorithm to extract foreground is that we determine some reference pixel coordinates as foreground [32]. Then we compare another pixel's intensity (I_x) to the reference pixel's intensity (I_R). Since we extract foreground in an RGB image, we have pixel's intensity in red, green and blue. We determine a pixel at coordinate (x,y) as foreground or background by equation (4.3),

$$I(x,y) = \begin{cases} \text{foreground} & \delta(I_x, I_R) < th \\ \text{background} & \text{otherwise} \end{cases} \tag{4.3}$$

where $\delta(.)$ is a distance function.

Instead of using the fixed reference pixel's coordinates of foreground as in [32] we use variable coordinates in every $\mathrm{ROI_{HCol}}$ based on motion pattern. Since a head motion in a $\mathrm{ROI_{HCol}}$ is different from other $\mathrm{ROI_{HCol}}$, then reference pixel's coordinates are also different from each other.

Figure 4.3 shows the determination of reference pixel's coordinates. First, we determine the initial coordinates a,b,c,d,e,f,g at a motion pattern in a $\mathrm{ROI_{HBin}}$. Then, we determine reference pixel's coordinates $r_1,\ldots,r_5$ in a $\mathrm{ROI_{HCol}}$. These reference pixel's coordinates are used in Eq. (4.3).

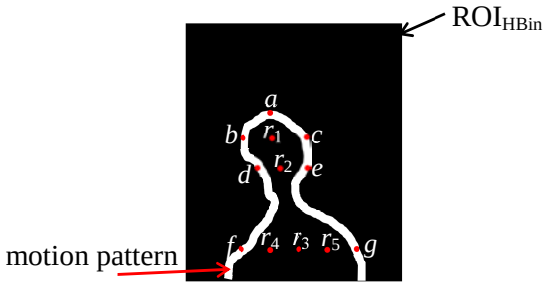


Fig. 4.3 The initial coordinates a,b,c,d,e,f,g in a ROI$_{HBin}$ to determine the reference pixel's coordinates $r_1,\ldots,r_5$ in a ROI$_{HCol}$.

The preprocessing of ROI$_{HBin}$ is explained in the following. Once the foreground pixels are obtained, the foreground is further refined by expanding the blob in three directions and then taking their intersection. This helps in filling the voids and empty spaces in blobs [29][32] (See **Fig. 4.4**).

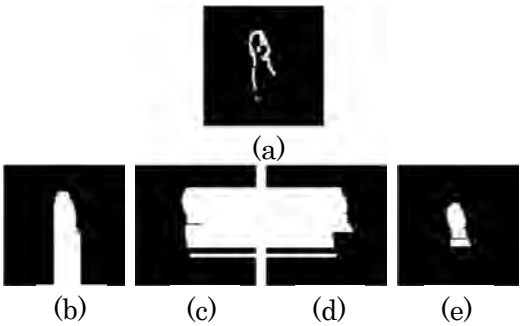


Fig. 4.4. Foreground refinement by hole filling:
(a) An original image, (b) top to bottom, (c) left to right,
(d) right to left, (e) refined foreground

The overview of the proposed method for foreground extraction can be summarized as in **Figure 4.5**. There are two kinds of foreground, foreground #1 and foreground #2. Further, these foregrounds are extracted by histogram of transition as their feature, as in section 4.4.

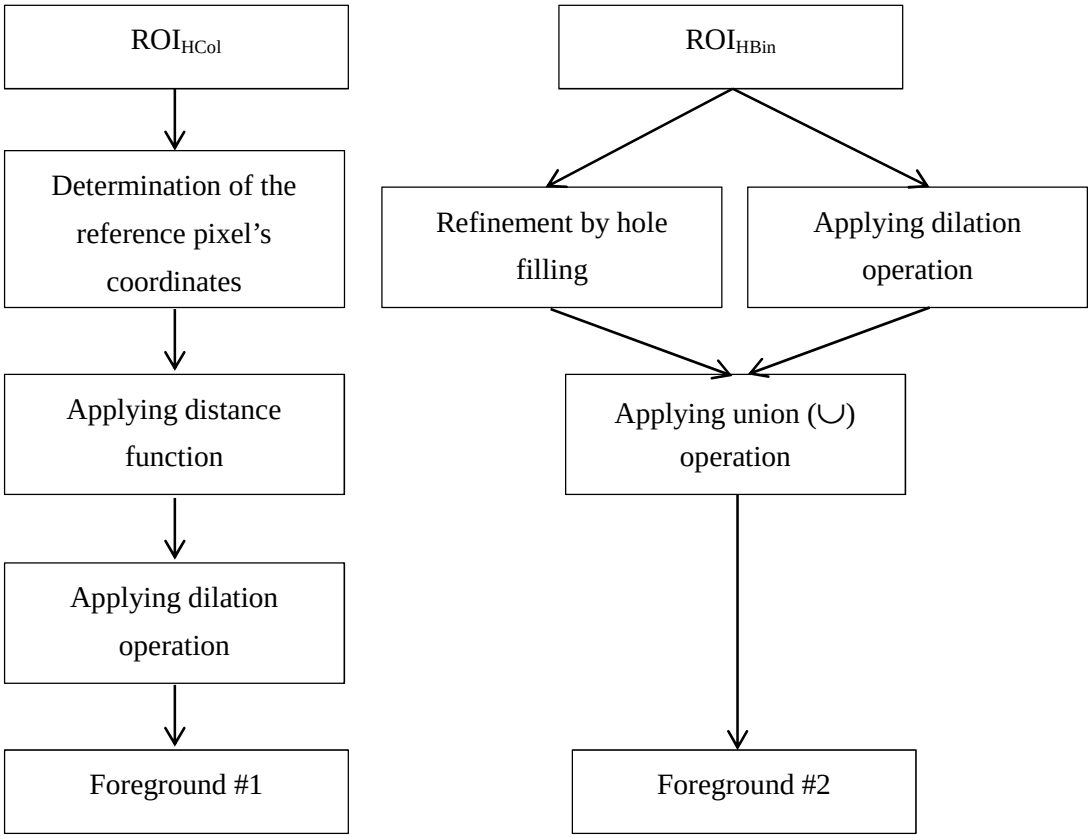


Fig. 4.5 The proposed method to extract the foreground of the head candidate

4.5 Transition Feature and a Histogram of Transition

After we get the foregrounds, then we extract the feature of these foregrounds. Our feature refers to [30,31]. Transition feature has been used successfully in handwritten recognition, but it hasn't been used in head detection yet. Due to a simple calculation to create a feature vector, we apply the idea to head detection. We do some modifications on it to be able to be used for head detection.

The idea is to compute the location and number of transitions from background to foreground along horizontal and vertical lines. This transition calculation is performed from right to left, left to right, top to bottom, and bottom to top. Since a constant dimension feature is required as input to the SVM classifier, an encoding scheme is

developed.

In the first stage of feature extraction, the transition in each direction is calculated. Each transition is represented as a fraction of the distance across the image in the direction under consideration. These fractions are computed in the increasing order, differed from [31] in decreasing order. For example, when calculating the location of transitions from left-to-right, a transition close to the left edge would have a low value and a transition far from the left edge would have a high value as illustrated in **Figure 4.6**.

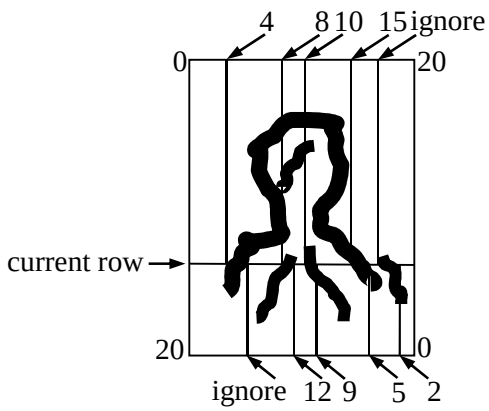


Fig. 4.6 The first stage of transition feature extraction shown for transitions from the left and from the right on one row of the image, with M = 4.

The maximum number of transitions, M, are counted on each line. If there are more than M transitions in a line, then only the first M are counted, the rest are ignored. M is set to 4. If there are less than M transitions on a line, then the "nonexistent" transitions are assigned as a value of 0.

More precisely, by a line we mean a row or a column of the head image. Let h be the height of the image and w be the width of the image. We assign exactly M values to each line, say $t_1,t_2,\ldots,t_M$. We assume that there are n transitions on a line located at (x_i,y_i) for i = 1,2, … n. The algorithm for calculating the transition feature can be represented

as follows: It doesn't require normalization as in [31]:

```
For i = 1 to min(n,M)
        If the line is row then
                t_i = x_i;
        Else
                t_i = y_i;
        end if;
End for;
If n<M then
        For i = n+1 to M
                t_i = 0;
        end for;
end if;
```

The transitions are resampled to a 4-point sequence for each direction and assembled into a feature vector. The four transitions for each row (column) are represented as two-dimensional (2-D) array, $t = [t_{ij}]$ for $i = 1,\ldots,h(w)$ and $j = 1,\ldots,4$.

The second stage is generating a histogram of transition. It is different from [31] where they calculated local averaging on the columns of t. Histogram of transition shows how often the location of the transition occurs at each transition. An example of generating a histogram of transition for transition left-to-right is shown in **Figure 4.7**.

This histogram of transition creates a feature vector to be fed into the input of a SVM classifier [33].

The HOG feature contains gradient information of a pixel among its neighbor. Thus they give a high magnitude at the edge. On the other hand, the LBP feature gives a binary pattern with a pixel among its neighbor. The histogram of transition feature looks like the HOG feature: It gives the edge position from right, left, top and bottom side. In

contrast to the HOG feature, the calculation of the histogram of transition feature is simpler.

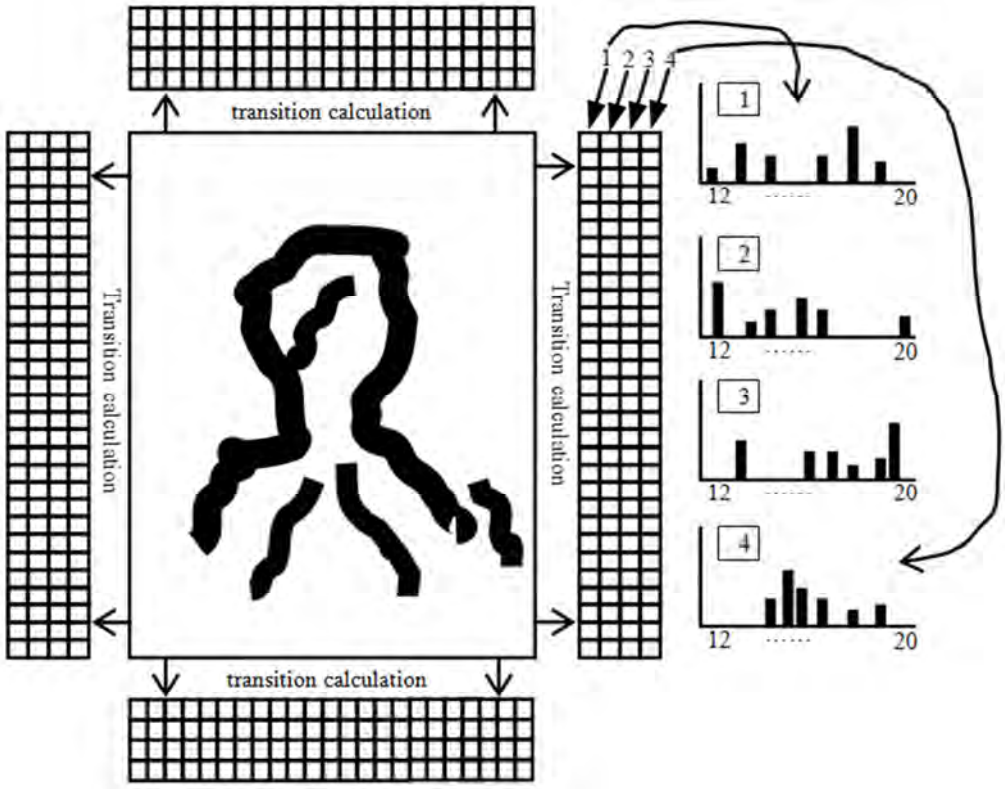


Fig. 4.7 The second stage of transition feature calculation consisting of generating a histogram of transition

4.6 Head Detection using an SVM Classifier

In the following sections, we propose a head detection method using Support Vector Machine (SVM). This method includes two steps: training and test.

The original Support Vector Machine (SVM) algorithm was invented by Boser, Guyon and Vapnik in COLT-92. On the other hand, the current standard SVM was proposed by Cortes and Vapnik in 1995 [34].

The SVM is a useful classification tool that uses a machine learning theory to maximize predictive accuracy while automatically avoiding over-fit to the data. A

classification task usually involves with training and test data which consist of some data instances. Each instance in the training data contains one target values and several attributes. The goal of SVM is to produce a model which predicts a target value of data instances in the test data which give only attributes.

Figure 4.8 shows the proposed method for head detection using an SVM classifier.

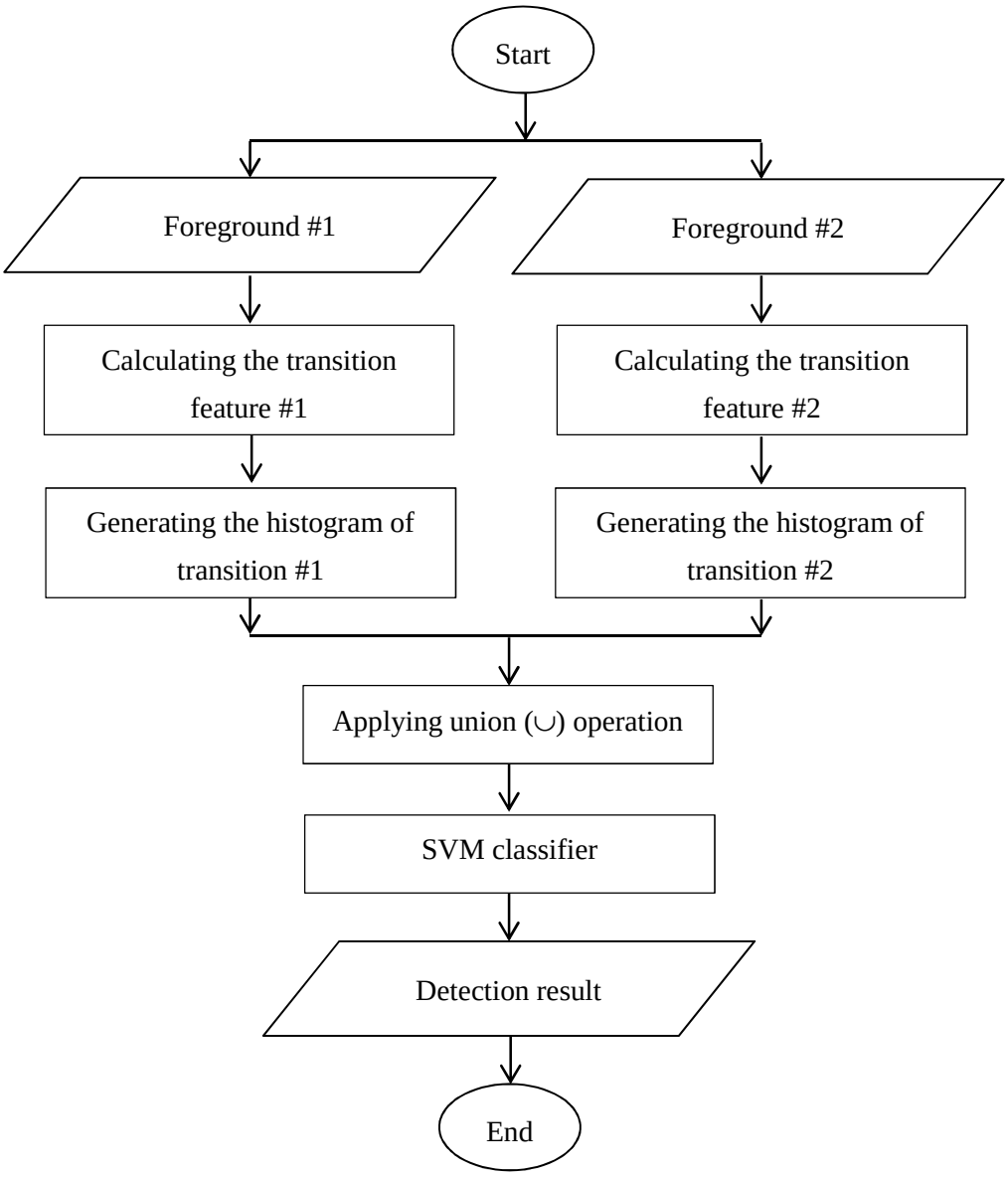


Fig. 4.8 The outline of proposed method for head detection using an SVM classifier

4.7 Experimental Results

4.7.1 Experimental Environment

In this section, we examine head detection in a static image and in a video. In a static image, the training and testing data are color images. The image size is ROI size. Training and testing data both include positive and negative samples. Positive samples are head and shoulders images, while negative samples are non-head images.

Examination using static images is aimed at examining the proposed foreground extraction method, i.e. foreground #1. On the other hand, examination using video scenes is aimed at examining the proposed histogram of transition as feature to be fed into the input of SVM classifier, as described in Section 4.5.

The configurations of the PC used in the experiments are shown in **Table 4.1**.

Table 4.1 Configurations of the PC used in the experiments.

OS	Microsoft Windows 7 ultimate
CPU	Intel® Core™ CPU 870 @2.93 GHz.
Memory	8.0 GB
Software Tool	Microsoft Visual Studio 2010.

4.7.2 Foreground Extraction Method

We extract a foreground from a color static image using a distance function as described in Section 4.3. The parameters of extraction are shown in **Table 4.2**. We use INRIA dataset[35][36]. **Figure 4.9** shows the result of the foreground extraction method.

Table 4.2 Parameters used in the foreground extraction method.

Parameters	**Values**
ROI size	20×30 pixels
Reference pixel's coordinate (x,y)	r_1 : (10,8)
	r_2 : (10,15)
	r_3 : (10,22)
	r_4 : (5,22)
	r_5 : (15,22)
Distance function	Euclidean distance
Threshold	10

4.7.3 Head Detection in a Static Image

In the first place, we create foreground #1 from a color static image using the distance function as described in Section 4.3. Then we extract the transition feature and create a histogram of transition as a feature vector, as described in Section 4.4. Finally, we examine the feature vector using an SVM classifier.

The parameters for the head detection are shown in **Table 4.3**. We use INRIA data for training and testing images [35-36].

Table 4.3 Parameters for the head detection in a static image

Parameters	**Values**
ROI size	20×30 pixels
Training data (INRIA data)	Positive : 2,000 images
	Negative : 4,500 images
Testing data (INRIA data)	Positive : 100 images

Table 4.3 Continued

Parameters	**Values**
	Negative : 300 images
Reference pixel's coordinate (x,y)	r_1 : (10,8)
	r_2 : (10,15)
	r_3 : (10,22)
	r_4 : (5,22)
	r_5 : (15,22)
Distance function	Euclidean distance
Threshold	10
The maximum number of transition, M	4
Dimension of feature vector	400

4.7.4 Head Detection in a Video Scene

This section implements the outline of the proposed method for head detection using an SVM classifier, as described in Section 4.5. In the first place, we create foreground #1 from a color static image using a distance function and foreground #2 using frame differencing, as described in Section 4.3. Then we extract the transition feature and create a histogram of transition as a feature vector for both foreground #1 and foreground #2, as described in Section 4.4. Finally, we combine the feature vector of foreground #1 and the feature vector of foreground #2 using a union operation, then examine the combination of the feature vector using the SVM classifier.

The parameters of head detection are shown in **Table 4.4**. The reference pixel's coordinates are variable, as determined in **Figure 4.3**. We use our data for training and another video scene for test images.

For this experiment, we examine the proposed head motion detection method using three video. Video 1 and video 2 are captured in a room with many furniture. The scenes in video 1 and video 2 include a shape similar to a head pattern, i.e. a tennis racket and a ball, respectively. Video 3 is captured outdoor; In this scene, there are people who are walking and running. **Figure 4.12** and **Figure 4.13** show the performance of the head motion detection in a video.

Table 4.4 Parameters used in the head detection in a video scene

Parameters	**Values**
ROI size	20×30 pixels
Training data	Positive : 500 images
	Negative : 1,500 images
Reference pixel's coordinate (x,y)	r_1 : variable
	r_2 : variable
	r_3 : variable
	r_4 : variable
	r_5 : variable
Distance function	Euclidean distance
Threshold	10
The maximum number of transition, M	4
Dimension of feature vector	800

4.7.5 Comparative Experiments

To evaluate the performance of the proposed method, we conduct some comparative experiments. In the first place, we compare the performance of the foreground extraction method between the distance function (proposed method) and Linear Neighborhood Propagation (LNP) method [37].

As in LNP, we initialize to label some pixels manually as foreground and background and let some unlabeled pixels labeled automatically. For this requirement, we label 20 and 8 pixels as foreground and background, respectively. These labeled pixels are shown in **Figure 4.10 (b)**. These label coordinates are fixed for all the training and the test data. The result of foreground extraction is shown in **Figure 4.11**.

In the second place, we compare the performance of feature extraction for head detection in static images, between the histogram of transition (proposed method), histogram of oriented gradient (HOG) [35][43] and linear binary pattern (LBP) feature [38]. **Table 4.5** shows a comparison result of performance of feature extraction.

In the third place, we compare the performance of the foreground extraction for head detection using the histogram of transition feature, between a distance function, LNP, frame differencing then dilation and frame differencing with refined foreground. This experimental result is shown in **Table 4.6**.

Finally, we compare the performance of the head detection in video scenes, between the histogram of transition feature, HOG and LBP. The comparison of the performance is summarized in **Table 4.7**.

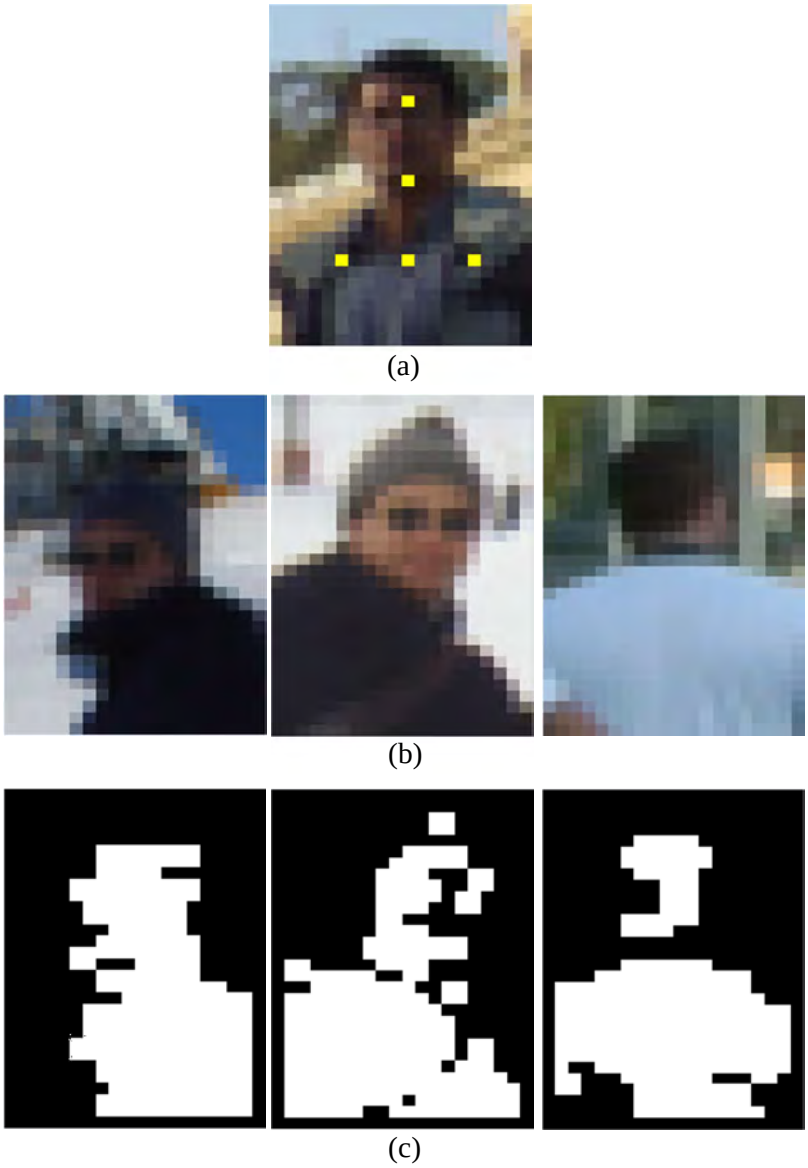
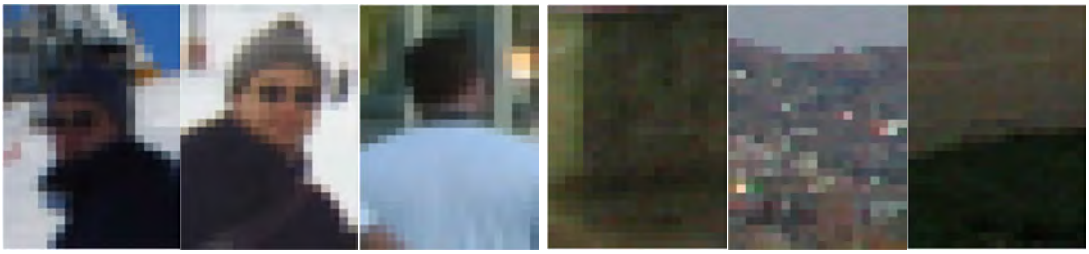


Fig. 4.9 The result of foreground extraction.

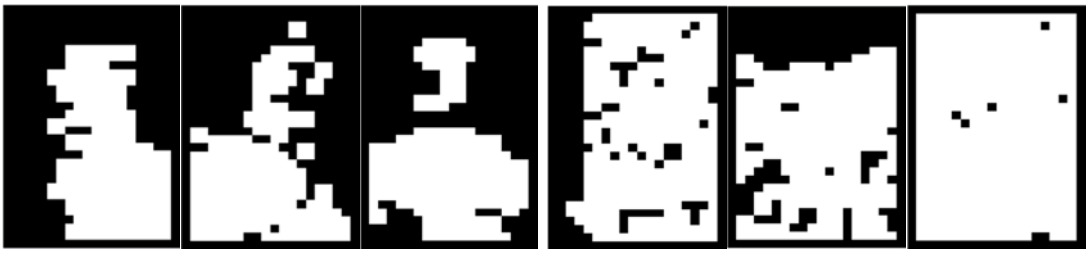
(a) Coordinate of reference pixel (b) original image (c) foreground extraction.

(a) (b)

Fig. 4.10 The coordinate of reference pixel: (a) The distance function method, (b) LNP method: yellow are foreground, magenta are background.



(a)



(b)

(c)

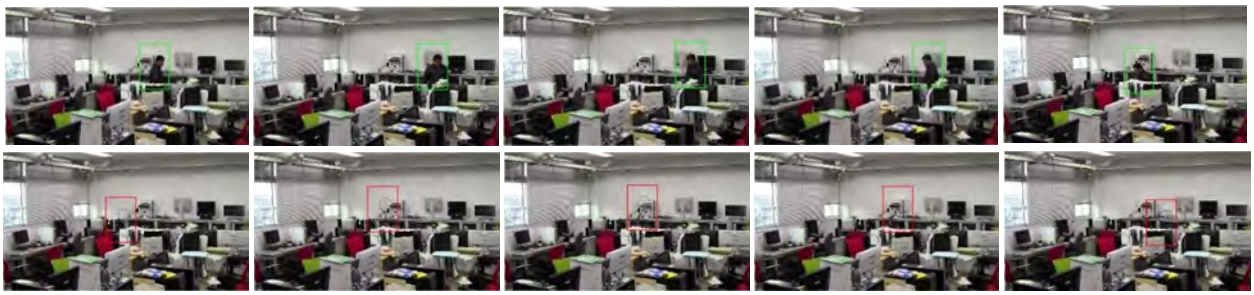
Fig. 4.11 The result of foreground extraction: (a) The original image, positive and negative samples, (b) the result of distance function method, (c) the result of LNP method.

Table 4.5. Evaluation of the feature extraction method

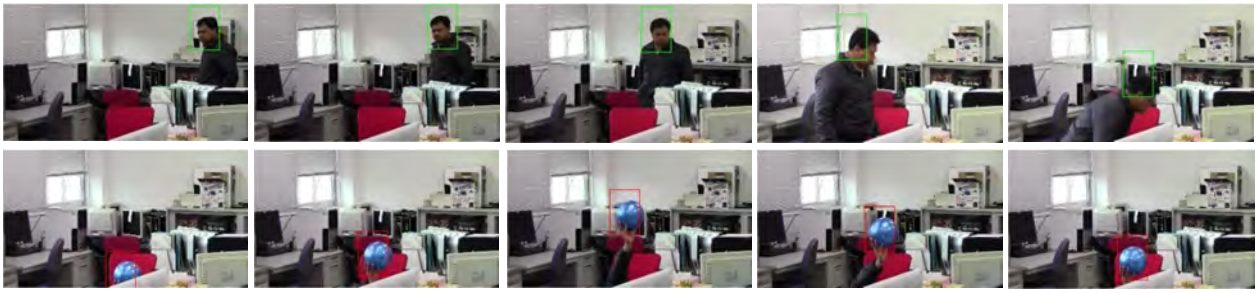
Feature	**The number of array**	**Detection rate (%)**		**Execution time (ms)**
		Positive	**Negative**	
HOG [35][43]	648	84	98.3	0. 353
LBP [38]	1020	**93**	80	0. 261
Histogram of transition	**400**	91	**99.7**	**0.077**

Table 4.6. Evaluation of the foreground extraction method

Foreground extraction method	**Detection rate (%)**		**Execution time (ms)**
	Pos.	**Neg.**	
LNP [37]	52	93	32.931
Distance function (Eq. 4.3)	89	99.7	0.077
Frame differencing then dilation	92	99.7	0.077
Frame differencing with refined foreground	80	93	0.077



(a)



(b)

Fig. 4.12. Performance of head detection. The green box shows that the proposed system detects a motion and a head, the red box shows that the system detects a motion but not a head . (a) Video 1: The system can distinguish a head and a racket. (b) Video 2: The system can distinguish a head and a ball

Fig. 4.13 Performance of multiple head detection.

4.8 Evaluation

In order to evaluate the effectiveness of the proposed head detection method, let us define *recall*, *precision FPR* and *F* by

$$recall = \frac{TP}{TP + FN} \times 100\% \tag{4.4}$$

$$precision = \frac{TP}{TP + FP} \times 100\% \tag{4.5}$$

$$FPR = \frac{FP}{FP + TN} \times 100\% \tag{4.6}$$

$$F = \frac{2 \times recall \times precision}{recall + precision} \tag{4.7}$$

Here

TP : head is detected as a head,

FN : head is detected as non-head,

FP : non-head is detected as a head,

TN : non-head is detected as non-head.

Then the performance of the proposed method is given numerically in **Table 4.7** and **Table 4.8**.

Table 4.7. Evaluation of the performance

video	Recall (%)			Precision (%)			FPR (%)		
	HOG	LBP	Hist. of Transition	HOG	LBP	Hist. of Transition	HOG	LBP	Hist. of Transition
1	87.2	85.0	**89.8**	88.0	85.2	**90.2**	8.1	8.8	**7.5**
2	77.1	**84.4**	83.7	80.5	78.1	**82.5**	10.4	11.5	**10.2**
3	**91.7**	68.5	90.1	78.4	65.1	**82.3**	16.6	**9.7**	12.3

Table 4.7. Continued

video	F (%)		
	HOG	LBP	Hist. of Transition
1	87.6	85.1	**89.9**
2	78.7	81.1	**83.1**
3	84.5	66.7	**86.0**

Table 4.8. Execution time

Method of feature	Execution time (ms)
Histogram of transition (800 dimensions)	**168.87**
LBP (1020 dimensions)	678.29
HOG (648 dimensions)	1041.78

4.9 Discussion and Conclusion

In this chapter, we proposed a method to extract a foreground by using a distance function and frame differencing. Another novel method is head motion detection by using the histogram of transition feature.

Image segmentation by using LNP method [37] needs accurate labeling. This method requires labeling pixels manually, then proceeds to automatic segmentation. This method force unlabeled pixels to be labeled. In case of an image without foreground, the resulting image should have foreground and background.

On the other hand, foreground extraction by using a distance function needs the accurate coordinate of reference pixels. After the initial result of extraction, there are some area of foreground detected as background. To refine this, we perform morphology operation, i.e. dilation operation. The advantage of the proposed method over the LNP method is that the proposed method needs reference pixels for foreground label only. So, if an image contains foreground and background, our method can create two patterns, i.e. foreground and background. If an image contains background only, our method can create one pattern, i.e. foreground. This case is no problem for the classification result, because one pattern will be detected as negative.

A histogram of transition feature as the novel method has good performance among HOG and LBP feature for head recognition. The advantage of this histogram is simple in computation. It calculates the location of transition between foreground and background. Due to its simplicity, the dimension of the feature vector is small. The simplicity and the small dimensional nature achieve a low computational cost and it is suitable for real time application. On the other hand, the drawback of the proposed method is that its feature relies on foreground extraction.

After all, the proposed methods for foreground extraction and feature extraction for head motion detection are satisfied. They achieve more than 80% both in recall and precision evaluation.

Chapter 5

Final Experimental Results and Evaluation

In this chapter, we use the results of fast motion detection and head motion detection to optimize the results of human motion detection. We also carry out a comparative experiment using the same experimental videos to discuss the effectiveness of the proposed human detection method.

5.1 Proposed Human Motion Detection Methods

In this section, we summarize two human motion detection methods which we have proposed in Chapter 2, Chapter 3 and Chapter 4.

Proposed Method 1:

STEP1: Motion representation (described in Section 2.2).

We use two kinds of motion representation. The first one is optical flow generated by Lucas-Kanade tracker as described in Section 2.2, and the second one is MHI, as described in Section 2.4.

STEP2: Motion feature extraction (described in Section 2.3).

We use a histogram of velocity and a shift histogram (described in Section 2.3 and Section 2.5).

STEP3: Fast motion detection (described in Section 2.6).

The results of fast motion detection using the proposed method 1 are shown in Figure 2.7 –2.9.

Proposed Method 2:

STEP1: Motion representation (described in Section 2.4).

STEP2: Motion feature extraction (described in Section 2.5).

STEP3: Fast motion detection (described in Section 3.4).

Our method based on accumulation function.

The results of fast motion detection using the proposed method 2 are shown in Figure 3.2 –3.4.

Proposed Method 3:

STEP1: Motion detection (described in Section 4.2).

We use frame differencing for motion detection. The result of this step is shown in Figure 4.2.

STEP2: Head candidate detection (described in Section 4.3).

STEP3: Foreground extraction (described in Section 4.4).

We use a distance function and refined frame differencing as foreground extraction. The coordinates of the reference pixels for the distance function is shown in Figure 4.3. The process of refined frame differencing is shown in Figure 4.4. The results of foreground extraction are shown in Figure 4.9.

STEP4: Feature extraction (described in Section 4.5).

We use a histogram of transition as the feature. Calculation of transition feature and generation of a histogram are shown in Figure 4.6 and Figure 4.7, respectively.

STEP5: Head detection (described in Section 4.6).

We use a SVM classifier to detect a head. We train this classifier using positive samples (head images) and negative samples (non-head images).

The results of head motion detection using the proposed method 3 are shown in Figure 4.12 and Figure 4.13.

5.2 The Methods in the Comparative Experiment

In this section, we introduce some comparative methods. There are two kinds of comparative experiments; the comparative experiments for fast motion detection and comparative experiments for head motion detection.

Comparative Method 1 [18] :

STEP1: Motion representation.

We use optical flow generated by Lucas-Kanade tracker for motion representation.

STEP2: Motion feature extraction.

We use a velocity histogram and direction of motion.

STEP3: Fast motion detection.

We use clustering parameters of the velocity histogram, i.e. *spreading width*, *mode of velocity* and *average of velocity*; and direction of motion.

The results of fast motion detection using the comparative method 1 are shown in Figure 5.1.

Comparative Method 2:

STEP1: Motion representation.

Motion representation is optical flow generated by Lucas-Kanade tracker.

STEP2: Motion feature extraction.

The motion feature is a histogram of velocity [18][19][20].

STEP3: Fast motion detection.

Fast motion detection based on an accumulation function.

The results of fast motion detection using the comparative method 2 are shown in Figure 5.2.

Comparative Method 3-1:

STEP1: Motion detection (described in Section 4.2).

STEP2: Head candidate detection (described in Section 4.3).

STEP3: Foreground extraction (described in Section 4.4).

STEP4: Feature extraction.

We use Histogram of Oriented Gradients (HOG) [35].

STEP5: Head detection (described in Section 4.6).

The results of head motion detection using the comparative method 3-1 are shown in Figure 5.3.

Comparative Method 3-2:

STEP1: Motion detection (described in Section 4.2).

STEP2: Head candidate detection (described in Section 4.3).

STEP3: Foreground extraction (described in Section 4.4).

STEP4: Feature extraction.

We use Linear Binary Pattern feature (LBP) [38].

STEP5: Head detection (described in Section 3.6).

The results of head motion detection using the comparative method 3-2 are shown in Figure 5.4.

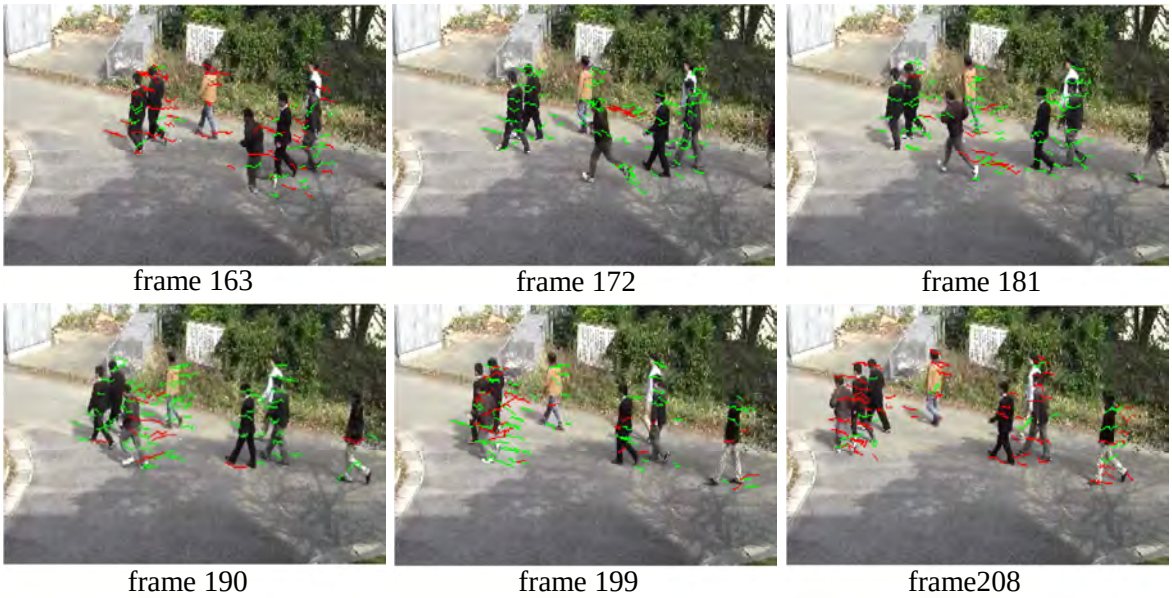


Fig. 5.1 The results of fast motion detection using the comparative method 1.

Fig. 5.2 The results of fast motion detection using the comparative method 2

Fig. 5.3 The results of head motion detection using the comparative method 3-1.

Fig. 5.4 The results of head motion detection using the comparative method 3-2.

5.3 Evaluation

In this section, we evaluate the effectiveness of the proposed method 1, the

proposed method 2 and the comparative method 1 and 2 using the evaluation method described in Section 3.6. And we evaluate the effectiveness of the proposed method 3 and the comparative method 3-1 and 3-2 using the evaluation method described in Section 4.8. The result of evaluation is composed of three values: *Precision*, *Recall*, *FPR* and *F*-measure. *Precision* is a measure of exactness or fidelity; *Recall* is a measure of completeness; *FPR* is a measure of inaccuracy.

Table 5.1 shows the results of evaluation using three different detection methods (the proposed method 1, the proposed method 2 and the comparative method 1 and 2).

Table 5.2 shows the results of evaluation using two different detection methods (the proposed method 3 and the comparative method 3-1 and 3-2).

Table 5.1 The result of evaluation on fast motion detection using three different detection methods.

method	**recall (%)**	**precision (%)**	**FPR (%)**	**F (%)**
Proposed Method 1	**83.3**	**81.2**	**1.8**	**82.2**
Proposed Method 2	**93.3**	**96.5**	**0.3**	**94.8**
Comparative Method 1	78.5	33.3	6.2	46.7
Comparative Method 2	83.3	76.9	1.89	79.9

Table 5.2 The result of evaluation on head motion detection using two different detection methods.

method	**recall (%)**	**precision (%)**	**FPR (%)**	**F (%)**
Proposed Method 3	**83.7**	**82.5**	**10.2**	**83.1**
Comparative Method 3-1	77.1	80.5	10.4	78.8
Comparative Method 3-2	84.4	78.1	11.5	81.1

5.4 Discussion

In this chapter, we summarized two proposed human motion detection methods which have been described in Chapter 2, Chapter 3 and Chapter 4. We also introduced comparative human motion detection methods, and then compared these human motion detection methods and evaluated the effectiveness of them.

In Table 5.1, the proposed method 1 and the proposed method 2 work well than the comparative methods. It is because, compared with motion representation based on optical flow generated by Lucas-Kanade tracker, MHI based representation gives much more information of motion. The optical flow relies on feature point extraction and tracking. The feature points largely rely on the texture on a human body and it is sometimes difficult to provide them due to few corners detected. Owing to the lack of feature points, it loses high velocity data and increases False Negative values. On the other hand, a MHI has many shift data extracted from all the contour pixels on moving objects in an image.

The proposed method 2 gives a good result among the comparative methods, because the accumulative function gives the exact discrimination between fast and normal motion in a block observation. While the discrimination based on the spreading width of the motion histogram and the velocity or shift average are very sensitive measures, depending on the location of motion to the camera position. If the motion is far away from camera position, the spreading width is almost similar between the motion which contains both fast and normal and the motion which contains normal only.

In Table 5.2, the proposed method 3 gives a good result among the comparative methods. It processes an image which contains two separated patterns, foreground and

background, in a binary image. Then a transition feature represents a position of foreground's edge. While the comparative methods process an image which contains many texture in the foreground and also in the background, it represents a pattern of foreground's edge. But sometimes the comparative methods fail to extract a pattern of foreground's edge.

Chapter 6
Conclusion

6.1 Conclusion

In this thesis, we proposed human motion detection using fast motion detection and head motion detection approach. These approaches are necessary for an automated security and monitoring system.

In Chapter 2, a method of automatic fast motion detection is proposed for detecting an anomaly motion using optical flow generated from Lucas-Kanade tracker and motion history image (MHI) as motion representation. MHI representation is often used in a motion recognition. In the proposed method, we apply it to represent a horizontal shift of an object.

In the part of fast motion detection, we carried out four experiments. These experiments are grouped into two approaches, optical flow based on the Lucas-Kanade tracker approach, shift based on the MHI approach and combination of them. To avoid foreground segmentation, motion recognition and object detection, we separate an image into several blocks and detect a fast motion in every block.

In the performed experiments, the proposed method is better than the comparative method. It can detect fast motion in a crowd with the condition that a camera direction and human motion flow don’t have to be perpendicular with each other and the camera location is not very high from the ground. It also does not need a special calibration.

In Chapter 3, different from Chapter 2, the proposed method uses the motion history image (MHI) only for motion representation. For the scenario of fast motion detection, this method is totally different with the scenario in Chapter 2.

In the part of fast motion detection, we carried out four experiments. These experiments are grouped into two approaches, optical flow based on the Lucas-Kanade tracker approach and shift based on the MHI approach.

In the performed experiments, the proposed method works better than the comparative method. It can detect fast motion in a crowd with the condition that a camera direction and human motion flow don't have to be perpendicular with each other. Also, it does not need special calibration. The execution time of the proposed method is faster than the comparative method. Therefore it is suitable for real time application.

In Chapter 4, a method of head motion detection is proposed for a monitoring system in a room using a histogram of transition feature. In this Chapter, we introduce two core methods: The first one is foreground extraction and the second one is feature extraction.

In the foreground extraction method, we propose a distance function and refined frame differencing as a foreground. In the feature extraction method, we propose a histogram of transition as a feature. A transition feature is often used in handwritten character recognition. In the proposed method, we apply it for head pattern representation by extracting the position of foreground's edge.

In the part of foreground extraction, we carried out two experiments. The first experiment is the proposed method, and the second experiment is image segmentation using linear neighborhood propagation (LNP). These experiments are preprocessing for

head detection. The effectiveness of the introduced foreground extraction is shown in the detection rate.

In the performed experiments, the performance of the proposed method is better than the comparative method. It gives a high recognition rate for both positive and negative images.

In the part of feature extraction, we carried out three experiments. We compare the proposed feature with the existing feature, i.e. histogram of oriented gradients (HOG) and linear binary pattern (LBP) feature. The effectiveness of feature extraction is shown in the detection rate.

In the performed experiments, our proposed method is better than the comparative method. It gives a high detection rate for both positive and negative images.

Finally, we conduct head motion detection in three videos. We carried out three experiment based on feature extraction. The evaluations of this experiment are *recall*, *precision*, *FPR* and *F* measure. In the performed experiment, the proposed method gives better evaluation in average than the comparative method. It gives high values for *recall* and *precision* and low value for *FPR*. The execution time of the proposed method is faster than the comparative method. Hence it is suitable for real time application.

In Chapter 5, we summarized the human motion detection methods proposed in Chapter 2, Chapter 3 and Chapter 4. In order to prove the effectiveness of the proposed methods, we also introduced some comparative methods. In the fast motion detection part of the comparative method, they detected a fast motion using another method (fast motion detection method using parameters of velocity histogram, shift histogram and combination of them). This fast motion detection method using the parameters of histograms need some threshold values which must be predefined carefully.

In the head motion detection part of the comparative methods, they used feature extraction based on the pattern of foreground's edge. If there is not enough texture on the foreground and the background, they may fail to recognize a head. The proposed method 1, 2 and 3 work better than the comparative method (shown in Table 5.1 and Table 5.2).

The proposed fast motion detection method has some advantages over the existing fast motion detection methods:

In the first place, the proposed method applies an accumulative function of a shift histogram, which is simple calculation. It is also advantageous for achieving real-time processing.

In the second place, the proposed method can be applied to detect fast motion in a crowd with the condition that a camera direction and human motion flow don't have to be perpendicular with each other.

In the third place, the proposed method requires no foreground segmentation, no motion recognition and no object detection.

And the fourth place, the proposed method requires no special calibration.

The proposed head motion detection method has some advantageous over the existing head detection methods:

In the first place, the calculation of feature extraction is simple and needs smaller computation time. It is suitable for real time application.

In the second place, since the dimension of the employed feature vector is small, it needs small memory usage.

And the third place, the proposed future is able to distinguish a head and a shape similar to a head.

After all, the originalities of this thesis are as follows:

In the first place, we introduce a shift histogram based on MHI representation. To the best of our knowledge, this is a new method to get shift information. Most of the existing methods use optical flow generated by Lucas-Kanade tracker or spatio-temporal gradient.

In the second place, we apply an accumulative function of the shift histogram to detect fast motion as an anomaly motion in a crowd. This approach does not necessitate motion learning. Most of the existing methods need a learning stage to learn normal and anomaly motion in a video clip.

In the third place, we introduce a function distance and a refined frame differencing as foreground extraction. Foregrounds which extracted from a static image and frame differencing images give an accurate detection of a head. They can distinguish a head and a pattern similar with a head.

In the fourth place, we introduce a histogram of transition feature as a feature to be fed into a classifier. Most of the existing head recognition methods apply HOG and/or LBP as their feature. But, as was experimentally shown, the proposed feature gives better performance than them.

6.2 Future Work

The proposed human motion detection method has some disadvantages.

The fast motion detection method has the following disadvantages: In the first place, it is weak when the motion spreads out of a camera view. In the second place, its output is not the shape of a human or an object.

On the other hand, the head motion detection method has the following disadvantage. It relies on foreground extraction. Sometimes a foreground fails to be extracted which contains many texture patterns.

These disadvantages remain to be solved as future work.

References

[1] M. Shah, O. Javed, K. Shafique, “Automated visual surveillance in realistic scenarios”, Multimedia Magazine, pp. 30-39, 2007.

[2] M. Paul, S.M.E. Haque, S. Chakraborty, “Human detection in surveillance videos and its applications – a review”, EURASIP Journal on Advances in Signal Processing, pp. 1-16, 2013.

[3] R. Mehran, A. Oyama, M. Shah. “Abnormal crowd behavior detection using social force model”, Proc. of Conf. on Computer Vision and Pattern Recognition (CVPR 2009), pp. 1-8, 2009.

[4] M. Park, J. K. Tan, Y. Nakashima, H. Kim, S. Ishikawa. “Detecting human flows on a road different from main flows”, Proc. of Artificial Life and Robotics 2011 (AROB 16th 2011), pp. 1-4, 2011.

[5] I. Tziakos, A. Cavallaro, L. Q. Xu. “Event monitoring via local motion abnormality detection in non-linear subspace”, Journal of Neurocomputing, pp. 1-12, 2009.

[6] T. Xiang, S. Gong. “Video behavior profiling for anomaly detection”, IEEE, Trans. on Pattern Analysis and Machine Intelligence, Vol. 30, No. 5, pp. 893-908, 2008.

[7] Y. Shi, Y. Gao, R. Wang. “Real time abnormal event detection in complicated scenes”, Proc. of Int. Conf. on Pattern Recognition, pp. 1-4, 2010.

[8] D. Tran, J. Yuan, D. Forsyth. “Video event detection : from subvolume localization to spatio temporal patch search”. IEEE, Trans. on Pattern Analysis and Machine Intelligence, Vol. 36, No. 2, pp. 404-416, 2014.

[9] B. Antic, B. Ommer. “Video parsing for anomaly detection”, Proc. of the 13th Int. Conf. on Computer Vision (ICCV’11), pp. 1-8, 2011.

[10] Y. Cong, J. Yuan, J. Liu. “Sparse reconstruction cost for abnormal event detection”, Proc. of Computer Vision and Pattern Recognition, pp. 3449-3461, 2011.

[11] X. Cui, Q. Liu, M. Gao, D.N. Metaxas. “Abnormal detection using interaction energy potentials”, Proc. of Computer Vision and Pattern Recognition (CVPR), pp. 3161-3167, 2011.

[12] S. Wu, B.E. Moore, M. Shah. “Chaotic invariants of lagrangian particle trajectories for anomaly detection in crowded scenes”, Proc. of Computer Vision and Pattern Recognition (CVPR), pp. 1-7, 2010.

[13] E. Ricci, G. Zen, N. Sebe, S. Messelodi. “A prototype learning framework using EMD application to complex scenes analysis”, IEEE Trans. on Pattern Analysis

and Machine Intelligence, Vol. 35, No. 3, pp. 513-526, 2013.

[14] L. Kratz, K. Nishino. “Anomaly detection in extremely crowded scenes using spatio-temporal motion pattern models”, Proc. of Computer Vision and Pattern Recognition (CVPR), pp. 1-8, 2009.

[15] A. Adam, E. Rivlin, I. Shimshoni, D. Reinitz. “Robust real-time unusual event detection using multiple fixed-location monitors” IEEE, Trans. on Pattern Analysis and Machine Intelligence, Vol. 30, No. 3, pp. 555-560, 2008.

[16] S. Saxena, F. Bremond, M. Thonnat, R. Ma. “Crowd behavior recognition for video surveillance”, Advanced Concepts for Intelligent Vision Systems, Lecture Notes in Computer Science Volume 5259, pp 970-981, 2008.

[17] J. Kim, K. Grauman. “Observe locally, infer globally: a space-time MRF for detecting abnormal activities with incremental updates”. Proc. of the IEEE conf. on Computer Vision and Pattern Recognition, pp. 1-8, 2009.

[18] P. Mudjirahardjo, J.K. Tan, H. Kim, S. Ishikawa, “Abnormal motion detection in an occlusive environment”. Proc. of SICE Annual Conference 2013, pp. 1398-1402. 2013.

[19] P. Mudjirahardjo, J.K. Tan, H. Kim, S. Ishikawa, “Temporal analysis for fast motion detection in a crowd”, Journal Artificial Life and Robotics, Vol. 20. 2015.

[20] P. Mudjirahardjo, J.K. Tan, H. Kim, S. Ishikawa, “Fast motion detection in a dynamic background”. Proc. of Int. Symp. on Artificial Life and Robotics 2014, ISAROB 19th2014, pp. 896-900, 2014.

[21] J.S. Mickelson. “Design and application of a head detection and tracking system”. Master Thesis – Massachusetts Institute of Technology, pp. 1-58, 2000.

[22] X. Huang, L. Li, T. Sim., “Stereo-based human head detection from crowd scenes”, Proc. of Int. Conf. on Image Processing, Vol.2, pp. 1353-1356, 2004.

[23] K.E. Aziz, D. Merad, B. Fertil, N. Thome.“Pedestrian head detection and tracking using skeleton graph for people counting in crowded environments”, Proc. of Conf. on Machine Vision Applications MVA2011, pp.516-519, 2011.

[24] C. Zeng, H. Ma. “Robust head-shoulder detection by PCA-based multilevel HOG-LBP detector for people counting”, Proc. of Int. Conf. on Pattern Recognition, pp. 2069-2072, 2010.

[25] M.P. Da Silva, V. Courboulay, A. Prigent, P. Estraillier, “Fast, low resource, head detection and tracking for interactive applications”, PsychNology Journal, Vol. 7, No. 3, 243 – 264, 2009.

[26]A. Bobick, J. Davis, The recognition of human movement using temporal templates, IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 23,No. 3, pp.

257-267, 2001.
[27] https://www.youtube.com/watch?v=z2bs3QoIZV4&spfreload=10
[28] P. Viola, M. Jones, D. Snow. “Detecting pedestrians using patterns of motion and appearance”. Technical Report on Mitsubishi Electric Research Laboratories, pp. 1-11, 2003.
[29] M. Baranwal, M.T. Khan, C.W.D. Silva. “Abnormal motion detection in real time using video surveillance and body sensors”. International Journal of Information Acquisition.Vol. 8, No. 2, pp. 103-116, 2011.
[30] P. Mudjirahardjo. “Penerapan jaringan perambatan balik untuk pengenalan kode pos tulisan tangan” (The implementation of back-propagation network for recognition of handwritten zip code), Master Thesis, Universitas Gadjah Mada, pp. 1- 100, 2001.
[31] P.D. Gader, M. Mohamed, J.H. Chiang. “Handwritten word recognition with character and inter-character neural networks”, IEEE Trans. On Systems, Man, and Cybernetics – Part B: Cybernetics, Vol. 27, No. 1, pp. 158-164, 1997.
[32] P. Mudjirahardjo, J.K. Tan, H. Kim, S. Ishikawa. “Head detection and tracking for an intelligent room”, Proc. of SICE Annual Conference 2014, pp. 353-358, 2014.
[33]C.W. Hsu, C.C. Chang, C.J. Lin. “A practical guide to support vector classification”, Proc. of IEEE Intelligent Vehicles Symposium, Vol. 1, pp. 1-6, 2010.
[34] C. Cortes and V. Vapnik: “Support-vector networks”, Machine Learning, Vol. 20, Iss.3, pp. 273-297, 1995.
[35] N. Dalal, B. Triggs. “Histograms of oriented gradients for human detection”, Proc. of Computer Vision and Pattern Recognition, Vol. 1, pp. 886-893, 2005.
[36] http://pascal.inrialpes.fr/data/human/
[37] F. Wang, J. Wang, C. Zhang, H.C. Shen, , “Semi-supervised classification using linear neighborhood propagation”, Proc. of Computer Vision and Pattern Recognition (CVPR2006), Vol. 1, pp. 17-22, 2006.
[38] G. Heusch, Y. Rodriguez, S. Marcel. “Local binary patterns as an image preprocessing for face authentication”, Proceeding of Automatic face and Gesture Recognition (FGR 2006), pp. 9-14, 2006.
[39] T.V. Oosterhout, S. Bakkes, B. Krose. “Head detection in stereo data for people counting and segmentation”, Proc. of Int. Conf. on Computer Vision Theory and Applications (VISAPP), pp. 620–625, 2011.
[40]M. Chen, G. Ma, S. Kee, “Multi-view human head detection in static images”. Proc. of Machine Vision Applications, pp. 100-103, 2005.
[41] V.B. Subburaman, A. Descamps, C. Carincotte. “Counting people in the crowd

using a generic head counter", Proc. of IEEE 9th Int. Conf. Advanced Video and Signal-Based Surveillance (AVSS), pp. 470–475, http://dx.doi.org/10.1109/AVSS.2012.87, 2012.

[42] M. Weber, W. Einhäuser, M. Welling, and, P. Perona. "Viewpoint-invariant learning and detection of human heads", Proc. of Fourth IEEE International Conference on Automatic Face and Gesture Recognition, pp. 1-8, 2000.

[43] M. Perdersoli, J. Gonzalez, B. Chakraborty, J. Villanueva. "Boosting histograms of oriented gradients for human detection". Proc. 2nd Computer Vision: Advances in Research and Development (CVCRD), pp. 1–6, 2007.

Appendix

This section consists of the experimental result of fast motion detection based on a velocity and a shift histogram and the experimental result of head motion detection based on a histogram of transition.

A. Fast Motion Detection based on a Velocity and a Shift Histogram

In this section, we describe two experimental results. **Figure A.1** depicts that the proposed method can detect a person who rides a bicycle and runs through walking people. **Figure A.2** depicts that the proposed method can detect a car, a bicycle and a running person at the same time. Both of the motion direction in **Figure A.1** and **Figure A.2** are not in perpendicular with the camera direction.

The evaluation of the performance on the two videos is summarized in **Table A.1** using the evaluation method described in Section 2.8.

Table A.1 Evaluation of the performance on the two videos

Video	**recall (%)**	**precision (%)**	**FPR (%)**	***F* (%)**
4	84.1	80.5	0.7	82.3
5	85.0	83.6	0.8	84.3

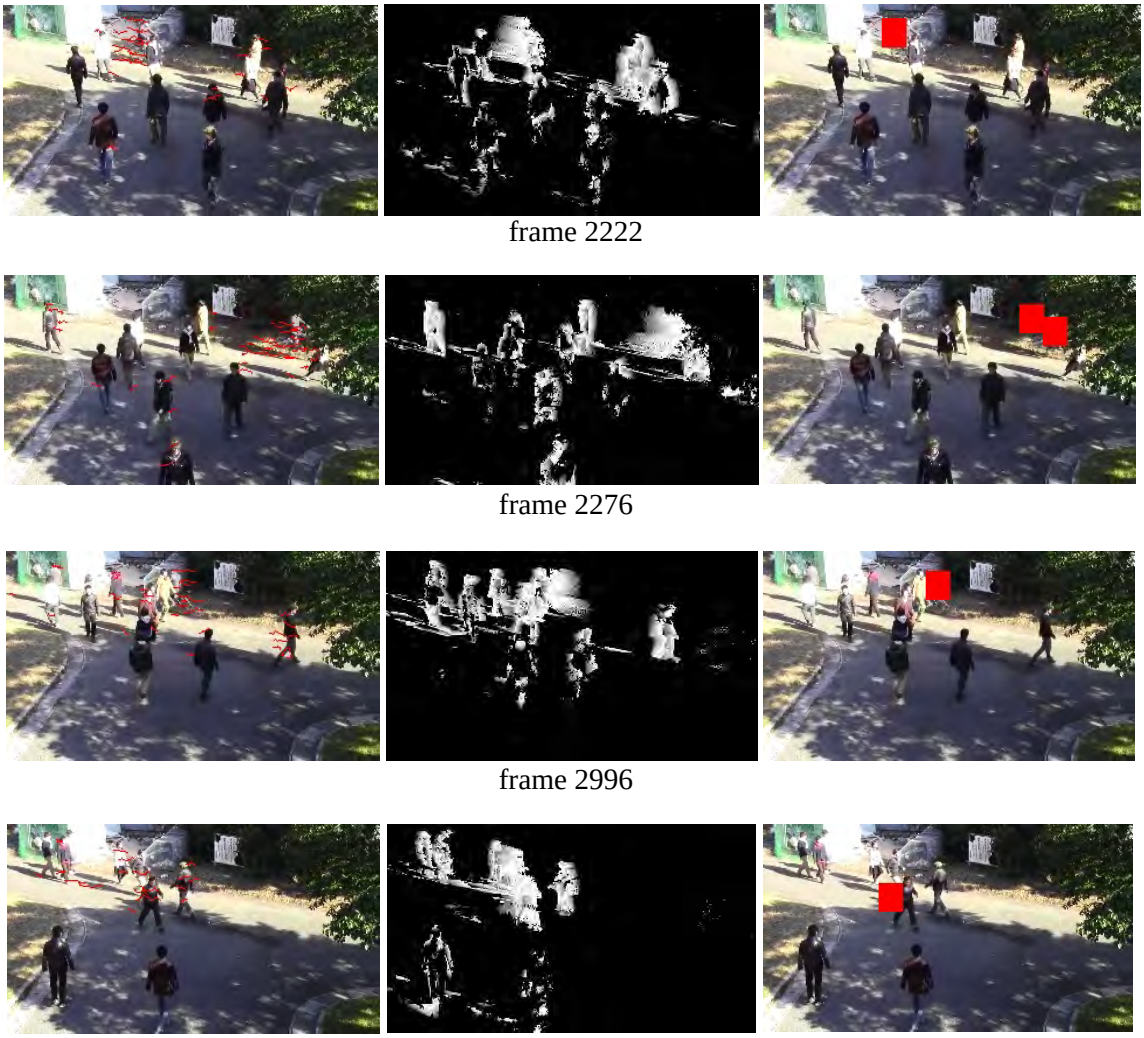


Fig. A.1 The result of fast motion detection (Video 4): (a) Optical flow image, (b) MHI, (c) the result of fast motion detection

frame 267

frame 447

frame 467

frame 483

(a) (b) (c)

Fig. A.2 The result of fast motion detection (Video 5): (a) Optical flow image, (b) MHI, (c) the result of fast motion detection

B. Head Motion Detection based on a Histogram of Transition

In this section, further experimental result is given with respect to head motion detection. As shown in **Figure B.1**, the proposed method can detect a head and ignore a balloon which has the shape similar to a head. The evaluation of the performance is summarized in **Table B.1** using the evaluation method described in Section 4.8.

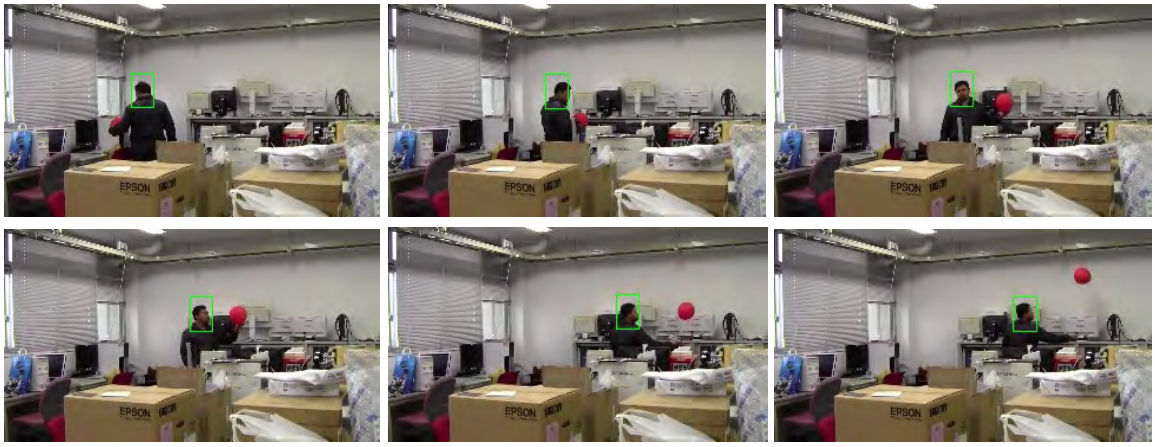


Fig. B.1 The performance of the proposed method on head motion detection. The system can distinguish a head and a balloon.

Table B.1 Evaluation of the performance

Feature method	**recall (%)**	**precision (%)**	**FPR (%)**	***F* (%)**
HOG [35][43]	87.9	88.9	7.9	88.4
LBP [38]	85.4	85.4	8.0	85.4
Histogram of transition	**89.8**	**90.3**	**7.1**	**90.0**