A People Counting Method Based on Head Detection and Tracking

Bin Li, Jian Zhang, Zheng Zhang, Yong Xu
Bio-Computing Research Center, Shenzhen Graduate School,
Harbin Institute of Technology, Shenzhen, China
holybiner@gmail.com, zpower007@163.com,
darrenzz219@gmail.com, laterfall@hitsz.edu.cn

***Abstract*—This paper proposes a novel people counting method based on head detection and tracking to evaluate the number of people who move under an over-head camera. There are four main parts in the proposed method: foreground extraction, head detection, head tracking, and crossing-line judgment. The proposed method first utilizes an effective foreground extraction method to obtain foreground regions of moving people, and some morphological operations are employed to optimize the foreground regions. Then it exploits a LBP feature based Adaboost classifier for head detection in the optimized foreground regions. After head detection is performed, the candidate head object is tracked by a local head tracking method based on Meanshift algorithm. Based on head tracking, the method finally uses crossing-line judgment to determine whether the candidate head object will be counted or not. Experiments show that our method can obtain promising people counting accuracy about 96% and acceptable computation speed under different circumstances.**

***Keywords*—people counting; head detection; LBP; head tracking**

I. INTRODUCTION

People counting techniques have been applied in many public places with entrances, such as supermarkets, subways and bus stations. The people flow data of these scenes can supply useful information for public security, marketing decision and resource allocation. With the increasing requirements for automatic people counting systems based on digital image processing and computer vision, effective people counting methods become remarkable and meaningful.

Many studies on people counting have been done [1-5]. Charoenpong [1] introduced a head detection method by using partial head contour, and they used the eclipse model to fit the contour. Zhao *et al.* [2] utilized the hair-color and head contour features to detect human heads. It is known that human heads have an approximate circular shape, so the contour template is commonly used in head detection. But the contour template based method is often influenced by other objects similar to human heads in scenes. Mukherjee *et al.* [3] proposed an effective passenger counting method, which first uses Hough [4] circle transform to detect human heads, and then performs head tracking using the optical flow method [5]. However, the tracking method based on the optical flow requires a large amount of calculation.

In this paper, we propose a people counting method based on head detection and tracking to solve the mentioned problems. The proposed method uses an over-head camera to acquire videos of walking people. As a result, the head is a stable, visible and evident part of a moving human, and the proposed method can divide humans into individuals. We also propose a novel method for head detection using a LBP feature based Adaboost classifier, and a kind of local head tracking method based on Meanshift algorithm. We also propose an effective foreground extraction method for head detection and a crossing-line judgment method which is integrated with head tracking to perform people counting.

The remainder of this paper is structured as follows. In Section 2, the framework of our method is introduced. In Section 3, we describe the details of our method including foreground extraction, head detection, head tracking and crossing-line judgment. In Section 4, experimental results and analyses are presented. Finally, we draw our conclusions in Section 5.

II. FRAMEWORK OF OUR METHOD

The overall framework of the proposed method has been presented in Fig. 1.

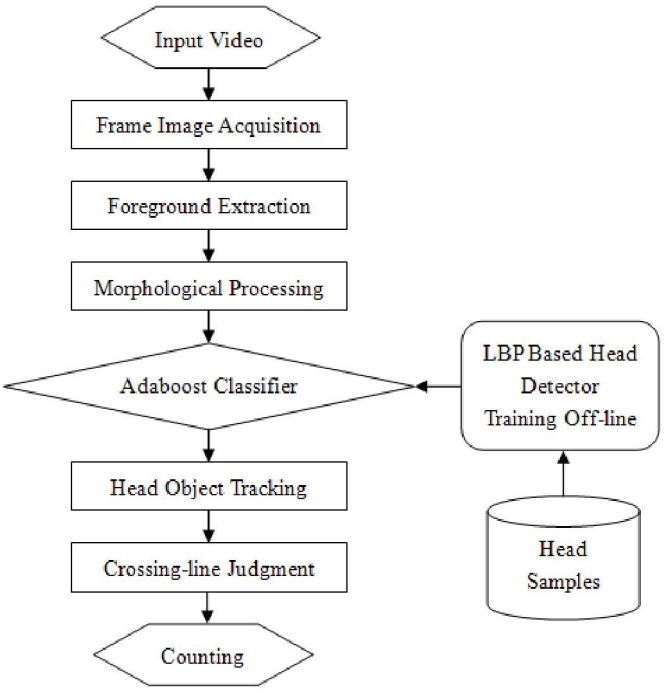


Fig. 1 Framework of our method

978-1-4799-5711-8/14/$31.00 ©2014 IEEE

There are four main parts in our method: foreground extraction, head detection, head tracking, and crossing-line judgment. The method starts with frame image acquisition from input videos. And then the foreground extraction is performed to obtain regions of moving objects in frame images. Here VIBE proposed by Olivier Barnich *et al.* [6][7] is used as the foreground extraction technique. And some morphological operations are also employed to optimize the foreground regions. After the foreground extraction is implemented, a trained LBP feature based Adaboost classifier is applied to detect human heads in the extracted foreground regions. Next, a local tracking method based on Meanshift algorithm is used to track the detected human heads. Finally, the tracking results combined with positions of the human heads are applied to count the heads which are crossing a counting line. The counting line is a straight line drawn by user and should be vertical or horizontal across the frame image. It will be introduced in details later in this paper. It is clear that the number of heads and people are the same.

III. PROPOSED METHODS

A. Foreground Extraction

In order to obtain regions of moving objects in frame images, foreground extraction techniques should be performed. Many techniques, such as background subtraction, frame difference, and GMM have been proposed for foreground extraction [8]. The background subtraction method is quite sensitive to light changes and shadows and frame difference method often produces ghosts and holes. The GMM method [8] is very time-consuming in real-time applications. In contrast, the VIBE algorithm [6][7] is efficient, anti-noise and it takes up less memory. Therefore, we adopt it to perform foreground extraction. The preliminary results are shown in Fig. 2.

(a) (b)

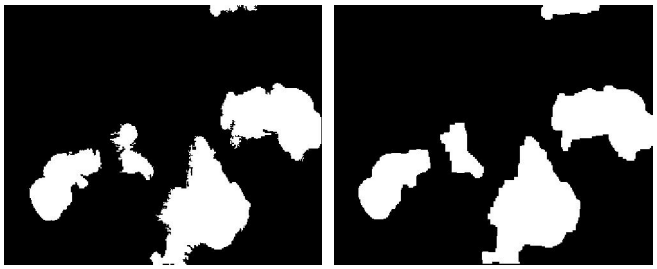
Fig. 2. The preliminary results of foreground extraction: (a) current frame image; (b) result of VIBE algorithm.

The foreground regions obtained using the above methods usually contain much noise and a lot of holes, and some blobs which are not components of moving human bodies. In order to remove the blobs, we utilize a thresholding formula to judge whether blobs should be reserved or not as follows:

$$B(i,j)=\begin{cases}0, & Area(B)<T_{area}\\ 1, & Area(B)\geq T_{area}\end{cases} \quad (1)$$

Where $Area(B)$ is contour area of a blob, and T_{area} is an area threshold determined by prior knowledge of head size. In our experiment, we set its value as 200. The value is suitable for that the camera is installed about 2~3 metres high from the ground. This height is commonly used in practical usage.

Furthermore, morphological operations including image erosion and dilation are used to eliminate the noise and fill the holes. In this paper, we firstly use closing operation with 2*2 template to process foreground regions, and then exploit opening operation with 2*2 template to process the regions. The optimized foreground regions after above operations are shown in Fig. 3. From Fig. 3(a), we can see that irrelevant blobs in Fig. 2(b) are removed. Moreover, from Fig. 3(b) we can see that noise in Fig. 2(b) is disposed and holes in Fig. 2(b) are filled.

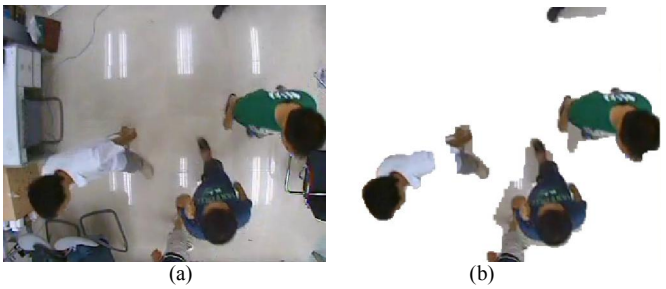


(a) (b)

Fig. 3. The optimized results of foreground regions: (a) result by using an area threshold; (b) result by using morphological operations.

B. Head Detection

In order to detect human heads in frame images accurately, we use the binary image of foreground regions as mask to extract region of interest (ROI) from initial video frame images. Generally, in the binary image resulting from foreground extraction, foreground regions are with gray value 1 and background regions are with gray value 0. Thus, we firstly invert the binary image, convert it from binary to RGB mode, and then perform bitwise-or operation with the initial frame image. Finally, we can obtain useful foreground regions including moving human objects of initial frame image. The results are shown in Fig. 4.



(a) (b)

Fig. 4. The results of ROI extraction: (a) the initial frame image; (b) the ROI image including moving human objects.

After the operations stated above, the retained regions are defined as the candidate regions for head detection. Then, an off-line trained LBP based Adaboost classifier is used to detect human heads. The LBP feature has been proved to be a very good texture descriptor for objects [11]. We evaluate some other feature descriptors for head detection during experiments, and finally find that LBP feature can give the

best detection accuracy and achieve a quite high speed.

In our experiment, we resize each sample image of human head to 24*24 pixels. And for simplicity's sake, we regard each resized sample image as a feature window. Each feature window is split into 3*3 blocks with the size of 8*8 pixels. The non-uniform 8-neighborhood LBP feature is adopted. Then we can figure out the LBP descriptor in each block using integral image method [9][10], and further get the LBP descriptor for the whole feature window.

After preliminary head detection, we use two thresholds of head size to filter some false detected head objects. The minimum head size $S_{\min}$ is set to 20*20 because some objects could be smaller than training samples, and the maximum head size $S_{\max}$ is set to 100*100 according to the video resolution. The final head detection results in candidate regions are shown in Fig. 5.

Fig. 5. Head detection results in candidate regions

C. Head Tracking

After head detection, we take detected human heads as candidate head objects to be tracked, and propose a local head tracking method based on Meanshift algorithm. A counting line should be specified vertically or horizontally across the frame image. Due to the time elapsed when people crossing line is very short, so head objects just move in a local area around counting line during this period. According to this, we add a counting region centred on counting line to assist in tracking as shown in Fig. 6.

(a) (b)

Fig. 6. Counting line and counting region: (a) horizontal; (b) vertical.

The proposed local head tracking algorithm is stated as follows:

1) Initialization: In the beginning, the algorithm produces a tracking list of head objects and initializes the tracking list to empty.

2) Update: If a head object is detected in a frame image, it will be determined whether to be added to the tracking list or not according to the following rules:

a) Out of counting region or not: If the centre of the head object is out of the counting region, it will not be added to the tracking list. Here the centre means the centre of bounding rectangle of the head object. The decision rule can be stated as follows:

$$H = \begin{cases} 0, & (x_h < x_L \,\|\, x_h > x_R) \\ & \|\, (y_h < y_U \,\|\, y_h > y_D) \\ candidate, & otherwise. \end{cases} \tag{2}$$

Where H denotes the head object, x_h and y_h respectively represent x-coordinate and y-coordinate of centre of the head object H, x_L and x_R respectively represent x-coordinate of left and right border of the counting region (if vertical), and y_U and y_D respectively represent y-coordinate of up and down border of the counting region (if horizontal).

b) Already in the tracking list or not: If the head object is in the counting region, it will be determined whether it is already in the tracking list or it is a new head object according to the decision rule as follows:

$$H = \begin{cases} 0, & Area_{overlap}(H, H_{list}) > T \\ newobj, & otherwise. \end{cases} \tag{3}$$

Where H_{list} is each head object already in the tracking list, $Area_{overlap}(H, H_{list})$ means the area of overlap between H and H_{list}, and T is a threshold to measure degree of overlap. In our experiment, the value of T is set as 1.

3) Meanshift iteration: The Meanshift algorithm is used to calculate new coordinates of each head object in the tracking list. During this process, the algorithm compares the new coordinates with the previous coordinates of the head object. If the Euclidean distance between them is less than a threshold T_d or number of iterations is larger than another threshold T_n, the algorithm stops the iteration and updates the head object. Finally the new position of the head object is added to its route list. In our experiment, T_d is set to 0.2 and T_n is set to 10.

4) Termination: If the tracking list is not empty, the algorithm uses crossing-line judgment to determine whether each head object in tracking list is crossing counting line or not. If the crossing-line condition is satisfied, the algorithm adds count number by 1; otherwise it repeats the process from 2) to 4).

D. Crossing-line Judgment

After the counting line and region are specified, relative locations between a head object and counting line and region

can be determined. Thus, we can use the information to determine the initial moving direction of the head object as follows:

- If the counting line and region is vertical:

$$D_H = \begin{cases} left2right, & x_{h\_ini} < x_L \\ right2left, & x_{h\_ini} > x_R \end{cases} \quad (4)$$

- If the counting line and region is horizontal:

$$D_H = \begin{cases} up2down, & y_{h\_ini} < y_U \\ down2up, & y_{h\_ini} > y_D \end{cases} \quad (5)$$

Where D_H is the initial moving direction of the head object, and x_{h_ini} and y_{h_ini} respectively represent initial x-coordinate and y-coordinate of centre of the head object.

After the initial moving direction of the head object is determined, we can perform crossing-line judgment using current position of the head object as rules below:

$$D_C = \begin{cases} left2right, & D_H = left2right \quad and \quad x_h > x_R \\ right2left, & D_H = right2left \quad and \quad x_h < x_L \\ up2down, & D_H = up2down \quad and \quad y_h > y_D \\ down2up, & D_H = down2up \quad and \quad y_h < y_U \end{cases} \quad (6)$$

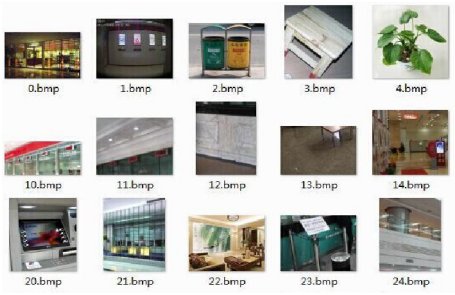
Where D_C is the direction for counting people, and x_h, y_h represent the current x-coordinate and y-coordinate of centre of the head object H, respectively. If the head object satisfies one of cases listed above, we add count number corresponding to D_C by 1; otherwise the head object will be discarded.

IV. EXPERIMENTAL RESULTS

A. Training

Because of lack of famous open-access datasets for head detection, so we collected some head images from camera, pedestrian datasets and internet. There are 2800 positive samples with the size of 24*24 pixels, and 3000 negative samples with different kind of sizes but larger than that of positive samples. Some positive and negative samples are shown in Fig. 7.

(a)



(b)

Fig. 7. Some training samples: (a) positive samples; (b) negative samples.

Here we use LBP feature based Adaboost classifier to train samples to get head detector. The typical process of training can refer to [9]. In our training process, the number of stages of classifier is set to 20, the minimum hit rate of each stage is 0.995, and the maximum false alarm rate of each stage is 0.5.

B. Testing

Testing videos are obtained by using an indoor over-head camera. Each video has the same resolution of 352*288(CIF). All the experiments run on a computer with 2GB memory and AMD 2.49 GHz CPU.

In order to measure the performance of our method, we could calculate detection rate, recall, and accuracy through the following formulas:

$$DetectionRate = \frac{TP}{TP + FP} \quad (7)$$

$$Recall = \frac{TP}{TP + FN} \quad (8)$$

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \quad (9)$$

Where TP denotes true positives, FP denotes false positives, TN denotes true negatives, and FN denotes false negatives.

In the first experiment, we compared our method with HOG based Adaboost classifier [12] and HOG based SVM method [13] on testing videos. The comparison results are shown in Table I.

Table I. The comparison with other methods

	SVM-HOG	Adaboost-HOG	Our method
Detection rate	88.4%	94.1%	98.9%
Recall	97.1%	86.5%	99.1%
Accuracy	86.1%	82.1%	98.1%
Average time per frame	67.8ms	40.7ms	36.4ms

Compared with other relevant methods, our method can achieve the highest detection rate, recall and accuracy. Meanwhile, our method costs the minimum time and thus has the best real-time performance. We demonstrate some detection results of mentioned methods in Fig. 8.

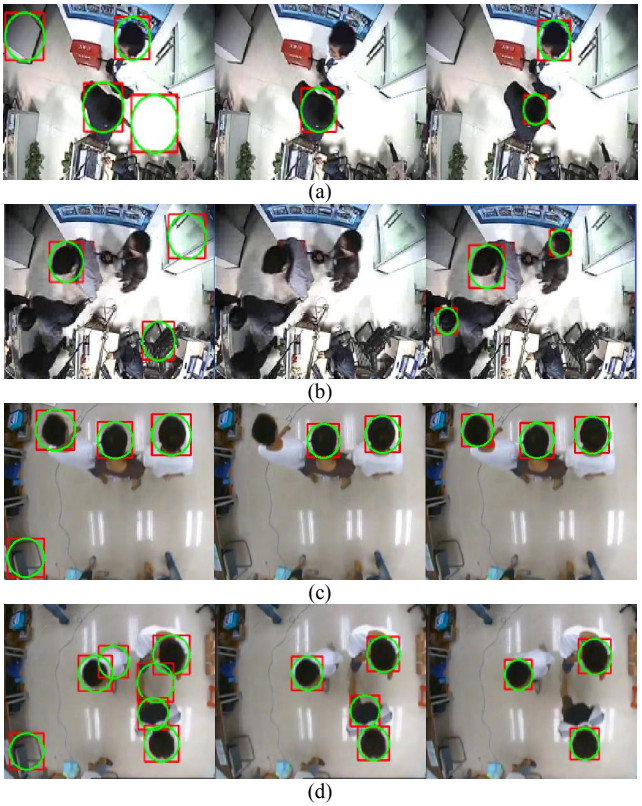


Fig. 8. Experimental results of mentioned methods, and from left to right they are: SVM-HOG, Adaboost-HOG, and our method: (a) detection results when people crossing; (b) detection results of one-way walking people; (c) (d) detection results when crowd density is high.

From the detection results in Fig. 8, we can see that HOG based SVM method often gives high false positive rate, and HOG based Adaboost method often gives high false negative rate. But our method can often provide accurate detection results.

In the second experiment, we tested our method on videos of different scenes. Here we choose 4 kinds of scenes. The results are shown in Table II.

Table II. Experiment results in different scenes

	Bi-directional	High density	Fuzzy scene	Carrying things
Real number	158	124	12	76
True positives	156	122	12	76
False positives	0	2	1	2
False negatives	2	2	0	0
Accuracy	98.7%	96.8%	92.3%	97.4%

We also provide some people counting results in mentioned scenes in Fig. 9.

The first scene is that people walk bi-directionally, the second one is that the crowd has high density, the third one is that light changes and the scene is fuzzy sometimes, and the last one is that people are carrying something else. From Table II and Fig. 9 we can see that our method fit well in different scenes, which proves that our method is quite robust and effective.

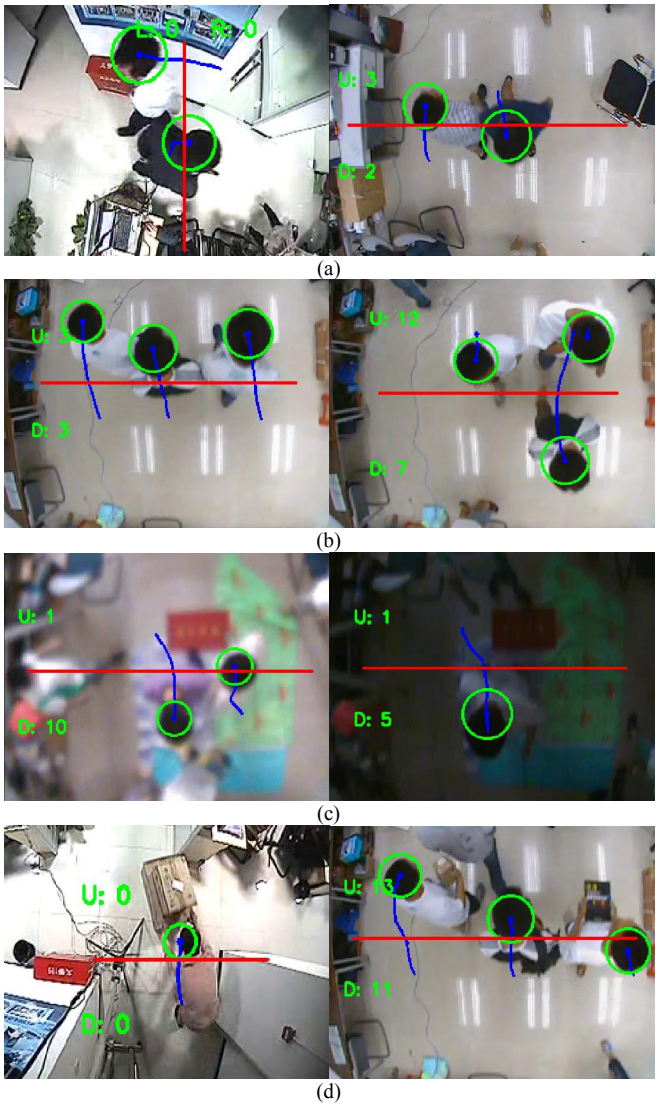


Fig. 9. People counting results in different scenes: (a) people walk bi-directionally; (b) the crowd has high density; (c) light changes and the scene is fuzzy; (d) people are carrying things.

C. Analysis

Compared with other reported approaches, our method can achieve better performance. The comparison results are shown in Table III.

Table III. The comparison with other reported methods

Method	Recall	Accuracy
BFR+MEA [14]	94.3%	94.5%
Hough+Optical flow [3]	97.4%	93.5%
Face detection [15]	82.6%	87.8%
Our method	98.9%	96.3%

Firstly, our method uses an over-head camera to acquire videos of walking people in order to divide humans into individuals. This approach can avoid overlap of moving people which frequently occurs in people counting based on face detection, head-shoulder detection and human detection. Also, we reduce false detection rate through the foreground extraction and the LBP feature based Adaboost classifier,

which can eliminate other static and irrelevant objects in frame images. Furthermore, our method produces a tracking list of head objects and updates it all the time, to store track of each head object and ensure accurate people counting of the crowd.

Compared with method proposed in [14], our method solves overlap problem caused by head-shoulder detection of moving people, and works well when crowd density is high. Compared with method proposed in [3], our method has faster head detection and tracking speed, and solves problem of inaccuracy in detection caused by Hough circle transform. Compared with method proposed in [15], our method mainly uses head features to represent human characteristics, because head feature is simple and effective for head tracking.

V. CONCLUSIONS

This paper presents a novel method for people counting based on head detection and tracking. The proposed method generally consists of four parts: foreground extraction, head detection, head tracking, and crossing-line judgment. To solve the problems of light variations, background interference and irrelevant objects, we firstly perform foreground extraction and morphological operations to get candidate regions. Then we apply LBP feature based Adaboost classifier to detect human heads in the candidate regions. The detected heads are tracked by a local head tracking method based on Meanshift algorithm. Finally we use crossing-line judgment based on tracking results and positions of head objects to determine whether the head object should be counted or not. The proposed method allows us to count moving people from different directions.

Experimental results show that our method can achieve promising people counting accuracy and acceptable computation speed, and is suitable for the real-time applications. The proposed method works well in different scenes and various crowd densities.

ACKNOWLEDGMENT

This paper is partly supported by NSFC under grants No. 61300032 and 61332011，as well as the Shenzhen Municipal Science and Technology Innovation Council (Nos. JCYJ20120613153352732 and JCYJ20130329151843309).

REFERENCES

[1] Theekapun Charoenpong, “Human head detection by using partial head contour,” Proceedings of the Third International Conference on Knowledge and Smart Technologies, pp. 29-32, 2011.

[2] Min Zhao, Di-hua Sun, and Wan-mei Fan, “Hair-color model and adaptive contour templates based head detection,” Proceedings of the 8th World Congress on Intelligent Control and Automation, pp. 6104-6108, July 2010.

[3] Satarupa Mukherjee, BaidyaNathSaha, Iqbal Jamal, Richard Leclerc, and Nilanjan Ray, “A novel framework for automatic passenger counting,” IEEE International Conference on Image Processing, pp. 2969-2972, 2011.

[4] R. C. Gonzalez and R. E. Woods, Digital Image Processing, 3rd ed., Prentice Hall, 2008.

[5] B. K. P. Horn and B. G. Schunck, “Determining optical flow”, Artif. Intell., vol. 17, pp. 185-203, 1981.

[6] Olivier Barnich and Marc Van Droogenbroeck, “ViBe: a universal background subtraction algorithm for video sequences,” IEEE Transactions on Image Processing, vol. 20, no. 6, pp. 1709-1724, June 2011.

[7] Olivier Barnich and Marc Van Droogenbroeck, “ViBe: a powerful technique for background detection and subtraction in video sequences,” IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 945-948, April 2009.

[8] Chris Stauffer and W.E.L Grimson, “Adaptive background mixture models for real-time tracking,” Computer Vision and Pattern Recognition, vol. 2, pp. 252-258, 1999.

[9] Paul Viola and Michael Jones, “Rapid object detection using a boosted cascade of simple features,” Computer Vision and Pattern Recognition, vol. 1, pp. 511-518, 2001.

[10] Paul Viola and Michael Jones, “Robust real-time face detection,” International Journal of Computer Vision, vol. 2, pp. 137-154, 2004.

[11] T. Ojala, M. Pietikainen, and D. Harwood, “A comparative study of texture measures with classification based on feature distributions,” Pattern Recognition, vol. 29, pp. 51-59, 1996.

[12] Qiang Zhu, Shai Avidan, Mei-Chen Yeh, and Kwang-Ting Cheng, “Fast human detection using a cascade of histograms of oriented gradients,” Computer Vision and Pattern Recognition, vol. 2, pp. 1491-1498, 2006.

[13] Navneet Dalal and Bill Triggs, “Histograms of oriented gradients for human detection,” Computer Vision and Pattern Recognition, vol. 1, pp. 886-893, 2005.

[14] Yaowu Hu, Ping Zhou, Hao Zhou, “A new fast and robust method based on head detection for people-flow counting system,” International Journal of Information Engineering, vol. 1, pp. 33-43, 2011.

[15] Tsongyi Chen, Chaoho Chen, Dajinn Wang, and Yili Kuo, “A people counting system based on face-detection,” International Conference on Genetic and Evolutionary Computing, pp. 699-702, 2010.