Real-Time Human Detection Circuit by Template Matching using B-HOG Feature Amount

Naotaka Hasegawa, Yuya Kimura, and Yoji Shibuya
Graduate School of Engineering, Chiba University
Chiba, Japan
E-mail: afka3451@chiba-u.jp

***Abstract*—We have designed and implemented a human detection circuit by template matching using B-HOG (Binarized-Histograms of Oriented Gradients) feature amount. Our circuit permits real-time processing of 30fps video at 76% of precision.**

***Keywords*—Human detection, HOG, Real-time processing**

I. Introduction

We have implemented the circuit to develop a compact and high-precision surveillance camera system. To detect humans, we have used template matching method using the HOG[1] feature amount which is strong to change in the lighting environment and easy to capture shapes of the object.

We have realized the circuit that does not require Block RAM and external memories by the raster-scan using shift registers. Moreover, by various techniques such as a down-sampling and approximation, we have succeeded in real-time processing of 640×480 sized / 30fps video.

Additionally, we have verified precision of the circuit by inputting 50 images in which people appear, and confirmed that the precision of detection is about 76%.

II. System

Figure 1 shows the schema of devised system. First, host-PC receives the frame captured by web camera (Logicool HD Pro Webcam C920t). Second, host-PC executes down-sampling to captured frame as pre-processing. Third, host-PC sends down-sampled frame to the evaluation board (Xilinx Artix-7 XC7A100T). The evaluation board executes template matching using B-HOG feature amount and calculates the similarity between the frame and template image while scanning. Fourth, getting higher similarity than threshold, the evaluation board sends the coordinates to host-PC. Finally, host-PC surrounds each detected person in a red frame using the coordinates.

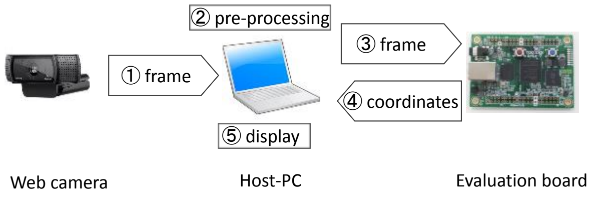


Fig. 1 Devised system

III. Circuit Architecture

Figure 2 shows the flow of the implemented system.

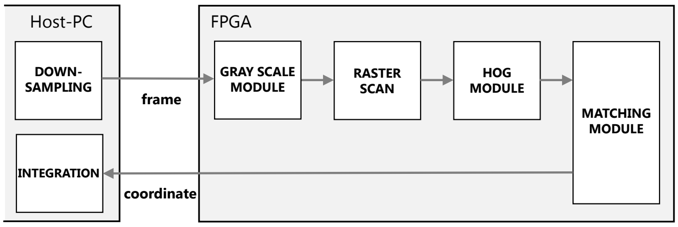


Fig. 2 Human detection system

DOWN-SAMPLING

The frame size (640×480) captured by web camera is too large for processing capacity of the evaluation board to process in real time. Therefore, host-PC executes down-sampling at first. TABLE 1 shows the resolution of down-sampled frame and template image.

Next, as shown in Figure 3, host-PC sends sequentially pixels of the down-sampled frame from the upper left to the evaluation board.

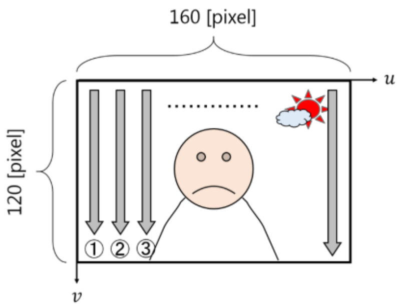


Fig. 3 Transfer order of the input frame

TABLE 1 Down-sampled and template resolution

Name	Resolution [pixels]
Down-sampled frame	160×120
Template	12×30

GRAY SCALE MODULE

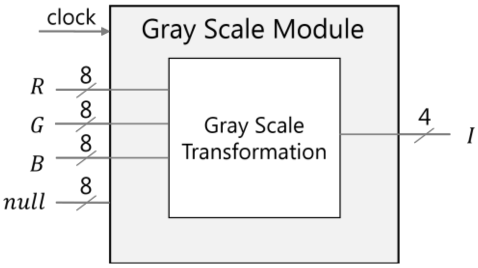


Fig. 4 Block diagram of gray scale module

This module transforms the 24bits color image into a 4 bits gray scale image as shown Eq. (1). Figure 4 shows the block diagram of gray scale module. $R(u,v)$, $G(u,v)$ and $B(u,v)$ are the red, green, and blue intensity value, and $null$ is non-value. $R(u,v)$, $G(u,v)$, $B(u,v)$ and $null$ are 8 bits, and $I(u,v)$ is 4 bits.

$$I(u,v) = \left\{ \begin{array}{l} 0.298912 * R(u,v) \\ + 0.586611 * G(u,v) \\ + 0.114478 * B(u,v) \end{array} \right. \quad (1)$$

RASTER SCAN

By using the shift register, we have implemented raster scan of the input frame without deterioration of throughput [2]. We explain about raster scan with $W_T \times H_T$ (=12×30) sized scanning window in $W_i \times H_i$ (=160×120) sized input frame. In this case, we use the 1,350 steps shift register shown in Figure 5 and Eq. (2). In every clock, a pixel intensity value $I(u,v)$ is thrown into the shift register in order of Figure 3.

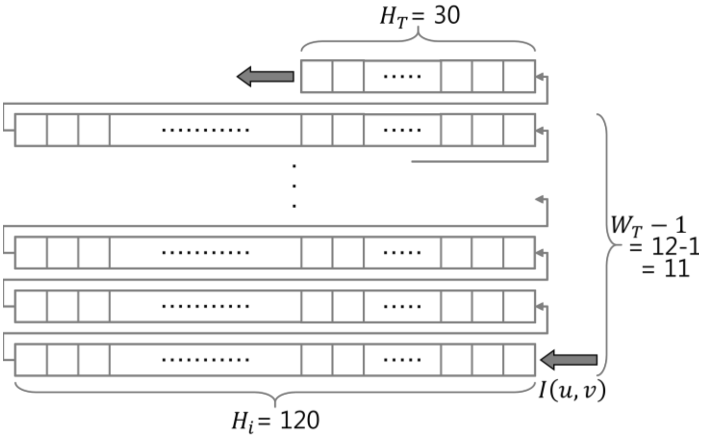


Fig. 5 Shift register of 1,350 steps

$$\begin{aligned} \text{steps} &= H_i \times (W_T - 1) + H_T \\ &= 1{,}350 \end{aligned} \quad (2)$$

When $I(u,v)$ is thrown pixel by pixel and the shift register is filled up, the relationship of gray area of the shift register and scanning window in input image becomes as Figure 6. In next clock, $I(11{,}30)$ is thrown into the shift register and $I(0{,}0)$ is thrown out of the shift register (Fig. 7). Therefore, scanning window moves one pixel in v direction. Repeating this flow every clock, we have enabled raster scan using shift register in order Figure 3. Thereby, it is possible to calculate B-HOG and the similarity in the scanning window area by extracting gray area into selector every clock.

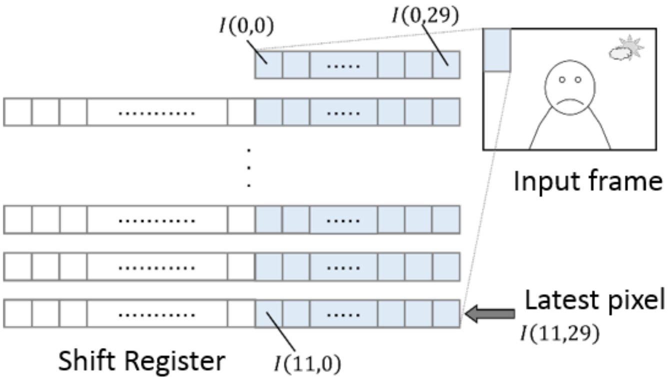


Fig. 6 When shift register is filled up

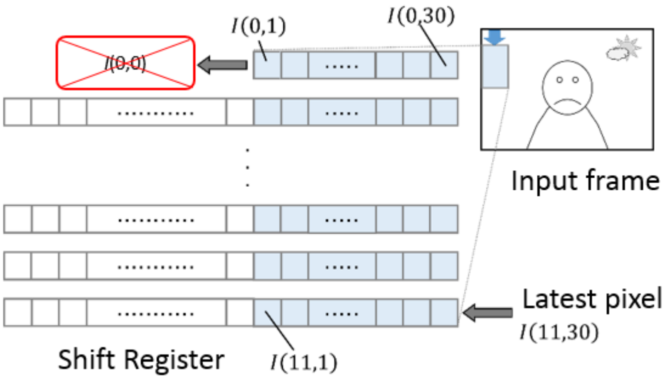


Fig. 7 Relationship between shift register and scanning window

HOG MODULE

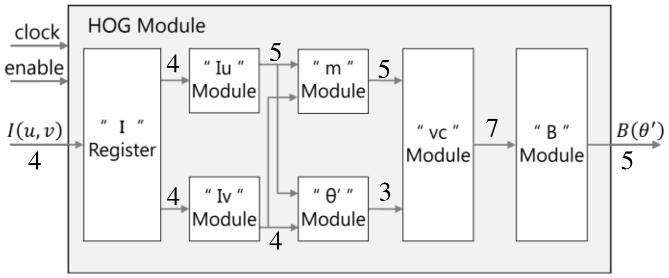


Fig. 8 Block diagram of HOG module

HOG (Histograms of Oriented Gradients) is a feature amount based on histograms of intensity gradients in local area. It is able to capture the shape of objects. However, since it has a high computational cost, we calculate HOG feature amount using some approximate calculations.

Figure 8 shows the block diagram of HOG module. First, an intensity gradient $I_u(u,v)$ of horizontal direction in scanning window area is calculated with intensity value $I(u,v)$ by Eq. (3). Similarly, an intensity gradient $I_v(u,v)$ of vertical direction is calculated by Eq. (4).

$$\begin{cases} I_u(u,v) = I(u+1,v) - I(u-1,v) & (3) \\ I_v(u,v) = I(u,v+1) - I(u,v-1) & (4) \end{cases}$$

Next, a gradient strength $m(u,v)$ is calculated. $m(u,v)$ is usually calculated by Eq.(5). However, Eq. (5) which is Euclidean distance has a high computational cost. Instead, we have adopted Manhattan distance (Eq. (6)) which had a low cost. A gradient direction $\theta(u,v)$ is calculated by Eq. (7).

$$m(u,v) = \sqrt{I_u(u,v)^2 + I_v(u,v)^2} \quad (5)$$

$$m(u,v) = |I_u(u,v)| + |I_v(u,v)| \quad (6)$$

$$\theta(u,v) = \tan^{-1}\frac{I_v(u,v)}{I_u(u,v)} \quad (7)$$

$$(0 \le \theta(u,v) < \pi)$$

Then, as shown in Figure 9, a gradient direction $\theta(u,v)$ is quantized by $\alpha(u,v)$ whose quantization width is $\pi/8$. When Eq. (9) is satisfied, a quantized gradient direction $\theta'(u,v)$ is generated using Eq. (8).

$$\theta'(u,v) = \frac{\alpha(u,v)}{\pi/8} \quad (8)$$

$$\begin{pmatrix} \alpha(u,v) \le \theta(u,v) < \alpha(u,v) + \dfrac{\pi}{8} \\ \alpha(u,v) = \dfrac{n\pi}{8} \quad (n = 0,1,2,\ldots,7) \end{pmatrix} \quad (9)$$

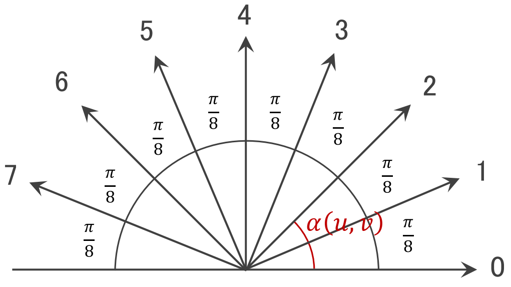
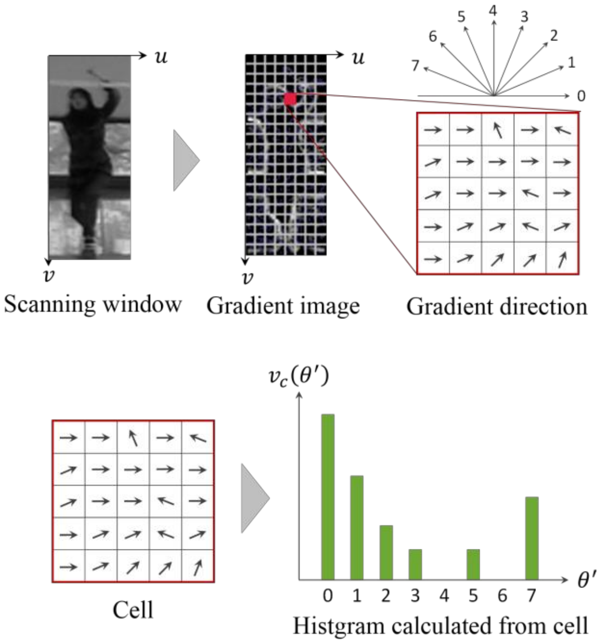


Fig. 9 Labeling

As shown in Figure 10, some pixels are combined and we call this bunch of pixels "cell". A gradient direction histogram $v_c(\theta')$ is calculated every cell by Eq. (10). δ is a delta function of Kronecker. δ becomes 1 when θ' is the same histogram element as $\theta(u,v)$. Otherwise, δ becomes 0.

$$v_c(\theta') = \sum_u \sum_v m(u,v)\delta[\theta', \theta(u,v)] \quad (10)$$



Scanning window Gradient image Gradient direction

Cell Histgram calculated from cell

Fig. 10 HOG feature amount

Subsequently, some cells ($q \times q$ cells) are combined and we call this "block". B-HOG (Binarized-HOG) [3] is calculated by performing a threshold processing as shown Eq. (11). In this design, we have set th =0.03.

$$b_c(\theta') = \begin{cases} 1 & if \;\; v_c(\theta') \ge th \times \sum_{k=0}^{7q^2} v_c(k)^2 \\ 0 & otherwise \end{cases} \quad (11)$$

Last, as shown Eq. (12), the binarized histogram $B(\theta')$ is generated using $b_c(u,v,\theta')$ every block.

$$B(\theta') = \sum_u \sum_v b_c(u,v,\theta')\delta(\theta') \qquad (12)$$

In this way, B-HOG of the image in the scanning window area is sequentially calculated by pipeline processing. TABLE 2 shows the size of a cell and a block in this design.

TABLE 2 Size of cell and block

Name	Size
Cell	6[pixel]×6[pixel]
Block	2[cell]×2[cell]

MATCHING_MODULE

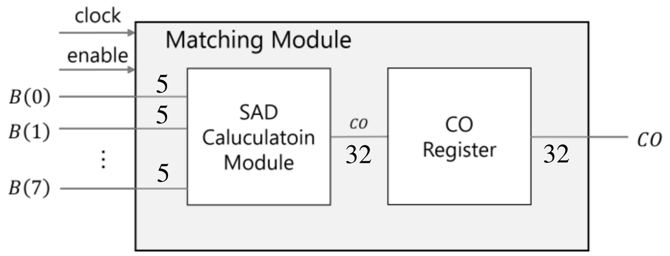


Fig. 11 Block diagram of matching module

Figure 11 shows the block diagram of matching module. Template matching is executed by using the calculated $B(\theta')$. In this design, we have adopted SAD (Sum of Absolute Difference), which has a low computational cost.

We regard $B(\theta')$ of the template as $B_T(\theta')$ and that of the image in the scanning area as $B_I(\theta')$. Then, SAD is expressed by Eq. (13). When we get smaller SAD, the similarity is high. When SAD is less than 10, the left upper coordinates CO of the scanning window are stored away at the registers.

When a calculation of the template matching for one frame is finished, the detection coordinates are transmitted to host-PC. In matching calculation, we have adopted the table reference method for a calculation of $B_T(\theta')$, and reduced calculation cost. TABLE 3 shows $B_T(\theta')$ values.

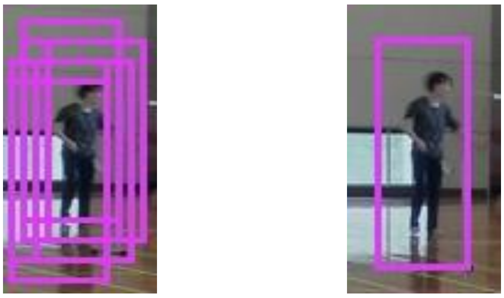
$$SAD = \sum_{\theta'} |B_T(\theta') - B_I(\theta')| \qquad (13)$$

TABLE 3 $B_T(\theta')$ values

$B_T(\theta')$	Value
$B_T(0)$	8
$B_T(1)$	9
$B_T(2)$	6
$B_T(3)$	8
$B_T(4)$	4
$B_T(5)$	4
$B_T(6)$	9
$B_T(7)$	6

INTEGRATION

The coordinates transferred to host-PC are necessary to integrate as shown in Figure 12. We have adopted Mean Shift[4] and integrated coordinates.



(a) before (b) after

Fig. 12 Coordinate integration

IV. CIRCUIT EVALUATION

A. *Circuit size*

TABLE 4 shows the performance of the designed circuit. The maximum frequency is 124.097 MHz. Therefore, we have set the operating frequency to 100 MHz. As this table shows, this circuit does not use Block RAM at all.

TABLE 4 Performance of designed circuit

Maximum Frequency [MHz]	**124.097**
Maximum combinational Path delay [ns]	0.001
Minimum period [ns]	8.058
Number of Slice Register	12,708
Number of Slice LUTs	44,516
Number of LUT Flip Flop pairs used	45,746
Number of Block RAM/FIFO	0
Number of BUFG/BUFGVTRLs	2
Levels of Logic	19

B. Computational time

We have evaluated the computational time of this designed circuit which operate at 100 MHz.

- ***Frame transfer***
  The transfer time for down-sampled frame is 638 µs.
- ***Gray scale transformation***
  This computational time is contained in frame transfer time. Therefore, we do not consider about this time.
- ***Raster scan***
  The delay until raster scan is started is 1,350 clocks which are the steps of the shift register. However, its computational time is contained in frame transfer time.
- ***B-HOG and Similarity***
  B-HOG calculation and similarity calculation are performed by pipeline processing of 16 stages. Therefore, these time is 0.16 µs (=16×10 ns).
- ***Coordinate transfer***
  The time to transfer a detected coordinate to host-PC is 265 µs . Therefore, when n detected coordinates are transferred, the time become 265 n µs.
- ***Total***
  TABLE 5 shows the detail of the computational time. The coordinate transfer time depends on the number of coordinates. The web camera operates at 30 fps, so that the total computational time in 1 frame must be smaller than 33 ms to realize real-time processing. The real-time processing is possible when the number of detected coordinates are less than 100 ($n \leq 100$).

TABLE 5 Detail of computational time

	Function	Computational time [µs]
Hardware	Frame transfer	638
	Gray scale transformation	—
	Raster scan	—
	B-HOG and Similarity	0.16
	Coordinate transfer	265n
	Total	**638.16+265n**
Host-PC	Down-sampling	86.0
	Coordinate integration	9.69
	Total	95.69
	Grand total	**734+265n**

C. Accuracy

We have verified the accuracy of this circuit using 50 images (Fig. 13). The number of people in these images is 118.

We have calculated detected rate R_d, undetected rate R_u and false detected rate R_f. Equations (14)-(16) show each evaluation equations[5]. “Number of detections” is the number of people detected correctly. “Number of false detections” is the number of detected objects which it is not human. “Total number of detections” is a grand total of detected coordinates.

$$R_d = \frac{\text{"Number of detections"}}{\text{"Number of people for detection"}} \quad (14)$$

$$R_u = 1 - R_d \quad (15)$$

$$R_f = \frac{\text{"Number of false detections"}}{\text{"Total number of detections"}} \quad (16)$$

Fig. 13 Images used verification

Since we have aimed the reduction of the circuit size and the improvement of the computational speed, the various approximate calculations are performed at the designed circuit. We have examined whether these influence the accuracy. Then, we have compared the results of calculation including no approximation by software with that of calculation including approximation by hardware.

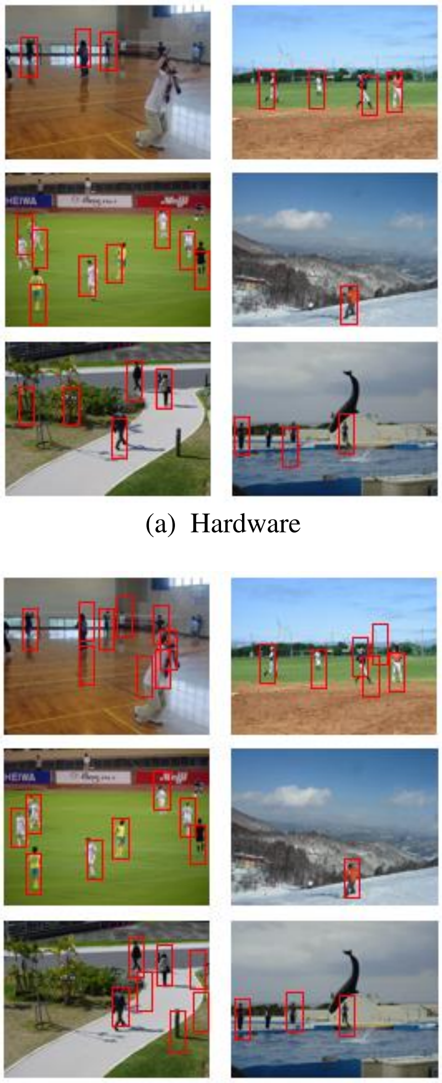
TABLE 6 shows both accuracies, and Figure 14 shows the output results. About detected rate R_d and undetected rate R_u, software had slightly better accuracy than hardware. However, about false detected rate, hardware had better accuracy than software. Most of the detection windows surround the people in Figure 14(a), which is the result of hardware. Thereby, we are able to confirm that the false detected rate $R_f$is low. On the other hand, in Figure 14(b), there are few un-detections, but there are many false detections, which is the result of software.

When we consider the balance of undetected rate R_u and false detected rate R_f, the both accuracies are nearly equal.

Therefore, we have succeeded in maintaining the accuracy, and reducing the cost of B-HOG calculation.

TABLE 6 Comparison of detection results

	Detected rate R_d	Undetected rate R_u	False detected rate R_f
Hard-ware	76.3%	23.7%	48.3%
Soft-ware	79.7%	20.3%	59.8%



(a) Hardware

(b) Software

Fig. 14 Output results

V. Conclusion

We have implemented the human detection system which uses template matching based on B-HOG feature amount and detects humans in pictures captured by web camera. We have resized captured images from 640×480 to 160 × 120 in host-PC. Thus, we have reduced whole calculation cost. Additionally, when we send captured frame, we have implemented fast raster scan without external memory by storing the pixel data of captured frame into shift register. Furthermore, we have implemented B-HOG calculation using approximation whose calculation cost is small, so that we have shortened calculation time more.

As a result of these techniques, we have succeeded in real-time human detection of 30fps video captured by web camera. Therefore, we have achieved development of a compact and high-precision surveillance camera system we desired.

References

[1] Yuji Yamauchi, Hironobu Fujiyoshi, Bon-Woo Hwang and Takeo Kanade, “People Detection Based on Co-occurrence of Appearance and Spatiotemporal Features”, Pattern Recognition, 2008. ICPR 2008. 19th International Conference on, 2008.

[2] 柴田裕一郎, “メモリ・レスの画像検出回路を実現する”, Digital Design Technology, CQ 出版, 15, 88-105, 2012.

[3] 松島千佳, 山内悠嗣, 山下隆義, 藤吉弘亘, “物体検出のための Relational HOG 特徴量とワイルドカードを用いたバイナリーのマスキング”, 電子情報通信学会論文誌 D, Vol.J94-D, 8, 1172-1182, 2011.

[4] D. Comanic and P. Meer, “Mean Shift Analysis Applications”, Computer Vision, 1999, The Proceeding of the Seventh IEEE Conference, 2,1197-1203, 1999.

[5] 前渕啓材, 原田祥吾, 呉海元, 和田俊和, “ハウスドルフ距離による近赤外線画像からの夜間歩行者検出”, 画像の認識・理解シンポジウム(MIRU2011), IS2-30, 703-709, 2011.