HOG-AdaBoost Implementation for Human Detection Employing FPGA ALTERA DE2-115

Trio Adiono[1]
School of Electrical Engineering and Informatics (SEEI),
Institut Teknologi Bandung
Jln. Ganesha No. 10, ITB Ganesha Campus, LABTEK VIII
Building, the 3rd floor, Bandung city, Indonesia

Kevin Shidqi Prakoso[2], Christoporus Deo Putratama[3],
Bramantio Yuwono[4], Syifaul Fuada[5]
University Center of Excellence on Microelectronics,
Institut Teknologi Bandung
Jln. Tamansari No. 126, Bandung city (40132), Indonesia
{[1]Kev.prakoso95, [2]Chris.dp.41, [3]Bramantio.yuwono}

***Abstract*—Human detection system using Histogram of Oriented Gradients (HOG) feature and AdaBoost classifier (HOG-AdaBoost) in FPGA ALTERA DE2-115 are presented in this paper. This work is expanded version from our previous study. This paper discusses 1) the HOG performance in detecting human from a passive images with other point-of-views (30 deg., 40 deg., 50 deg., 60 deg. and up to 70 deg.); 2) FPS test with various image sizes (320 x 240, 640 x 480, 800 x 600, and 1280 x 1024); 3) re-measurement the FPGA's power consumption and 4) simulate the architecture in RTL. We used three databases as a parameter for test purpose, *i.e.* INRIA, MIT, and Daimler.**

Keywords*—FPGA; human detection; adaboost classifier; ALTERA DE2-115; Histogram Oriented Gradients (HOG) feature

I. Introduction

The object detection is a critical task for many implementations in computer vision research area, like entertainment and autonomous system/robotics application [1]. The Histogram of Oriented Gradients (HOG) has been widely employed as a feature for object detection [2]. The HOG feature robust to the illumination changes and has a high accuracy in detecting objects with texture variations. So far, there is none proved feature to surpass the HOG existence [3]. That is why HOG-based detectors serve as the fundamental feature for object detection systems. HOG is also commonly implemented for detecting the humans effectively [4-5].

To date, there are many papers on FPGA-based HOG that implemented in Xillix FPGA for high-speed and high-accuracy human detection systems [6-10]. In previous work [11], we have applied the FPGA-based human recognition in an image, we employed ALTERA DE2-115. The results indicate that the humans are succesfully detected from a particular image with 1280 x 1024 resolutions and frame rate with 129 FPS. In that case, we used INRIA database. In [11], we also compared the other related works with our architecture and also investigated its power consumption of FPGA using Altera's Power Analyzer when performing as human detector system.

However, the paper discussion is a lack of the results analysis when the image viewed from different angles using various databases, *e.g.* INRIA, MIT, and Daimler. Hence, we extend its study involving HOG performance for human detection in an image with other angles up to 70°; FPS test with different image resolutions; review about the power measurement of FPGA; and the simulation result of register-transfer level (RTL). To collect more data, various databases such as INRIA, MIT, and Daimler are used in this research.

This paper is sponsored by Pusat Unggulan IPTEK (PUI) that was funded by The Ministry of Research, Technology and Higher Education, Indonesia

II. Methods

A. System Description

Section II-A describes the HOG algorithm implementation on FPGA ALTERA DE2-115. The illustration of research workflow can be shown in Fig 1. The following is brief description of research methods where the details are presented in Section II-B to Section II-F.

1) Greyscaling and scaling of the image input's are conducted in the first step. Therefore, human on the image can be identified with various distance.

2) Afterwards, the process is continued by every pixel gradient calculation so the dx and dy component can be obtained for every pixel. Both of the components are used to calculate magnitude and angle that used for HOG Feature Extraction every one cell with the size of x*y.

3) HOG Feature Extraction used can minimize system throughput and logic gates utilization because there is not arctan calculation using CORDIC which use a lot of logic gates and add the system throughput.

4) After histogram of 2 x 2 cell is obtained which every cell contained 8 x 8 pixel, the normalization process is performed. In this process, the quadratic and root mathematical operation is approached by the value of the near power of two that segmented into 4 section between two near power of two values so the normalization process can be implemented with shifting operator

5) Because of the FPGA's limited size of memory, histogram binarization is performed so one histogram can be represented using 8 bit. Although the utilization of memory can be reduced to 1/64th, this histogram binarization process can affect the system performance.

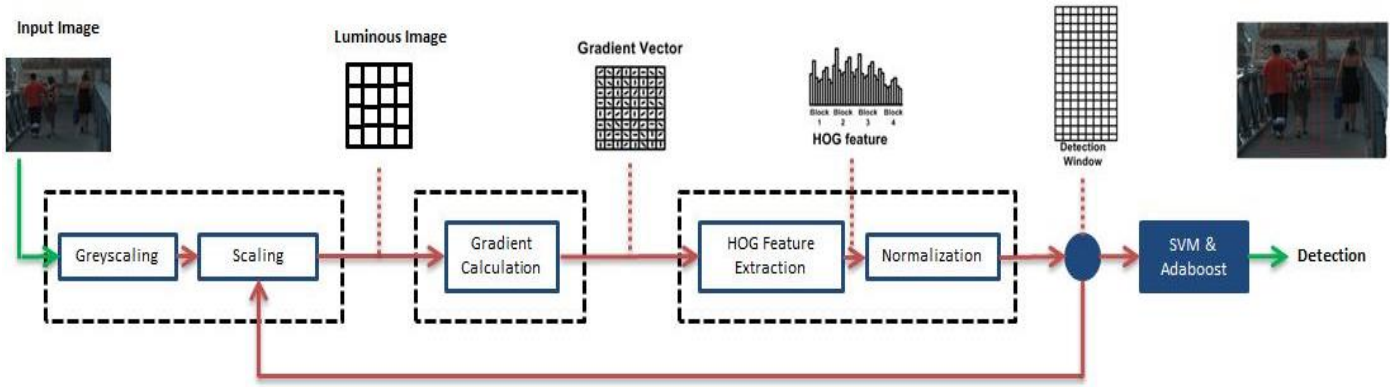


Fig. 1. Workflow of HOG-AdaBoost Implementation in an FPGA ALTERA DE2-115 for Human Detection Purpose.

6) After normalized histogram is obtained, detection process using the neural network is performed. Because this process is needed matrix multiplication, the systolic array is implemented on FPGA so the number of logic gates and system throughput can be minimized. Weight matrix and bias vector are obtained using Support Vector Machine (SVM) and AdaBoost which are done in offline.

B. Grayscaling & Scaling

As discussed before, the Grayscaling process is performed because of the HOG is an object detection method oriented on the shape of an image. The colored image has three matrices consists of R (Red), G (Green), dan B (blue) as visualized in Fig 2. Therefore, the RGB values are not needed and it can be represented by the luminance value. It can be obtained by using Eq (1),

$$Y = 0.298912 \text{ x } R + 0.586611 \times G + 0.114478 \times B \quad (1)$$

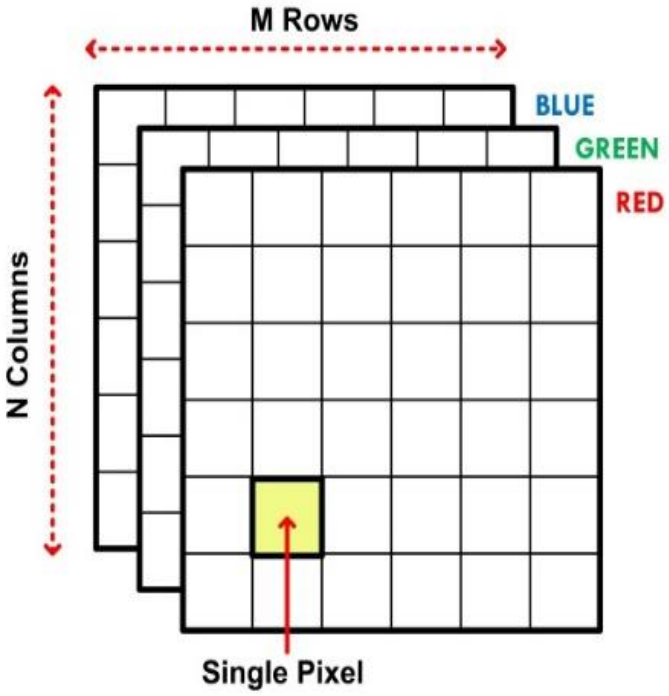


Fig. 2. Matrices of a Color Image, Reproduced from [12].

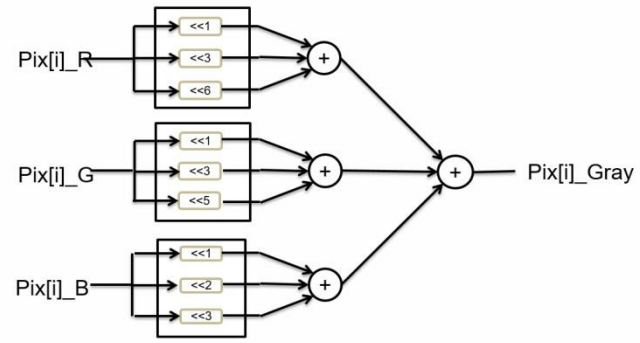


Fig. 3. Preprocessing Architecture.

We can use shifters block to approximate constants as expressed in Eq (1). To simplify the approach, we take two numbers after a comma. The following is the justification:

- 0.29 is approximated by 0.0100101, needing 3 shifters.
- 0.58 is approximated by 0.100101, needing 3 shifters.
- 0.11 is approximated by 0.000111, needing 3 shifters

The scaling process is needed so the system can detect humans from the various different distances. This process used sampling method of some specifics pixel correspond to the scaling rate. The architecture of preprocessing (grayscalling + scalling) can be illustrated in Fig 3.

C. Gradient Calculation

Gradient calculation can be done by calculating the difference of the neighboring pixel horizontally for *dx* and vertically for *dy*. The magnitude and angle value are needed to calculate histogram.

Magnitude calculation is explained on the Eq (2). The magnitude of the gradient *m* is included in histogram based on quantized difference orientation for every luminance value on a certain cell. The process can be illustrated in Fig 4. In this work, the magnitude calculation uses own algorithm to approximate roots and squares. It uses 120 modules in parallel. The architecture is simple subtraction between neighboring pixels. Afterwards, the calculation result is then saved to the SDRAM. The architecture is shown in Fig. 5.

$$m(x,y) = \sqrt{f_x(x,y)^2 + f_y(x,y)^2} \quad (2)$$

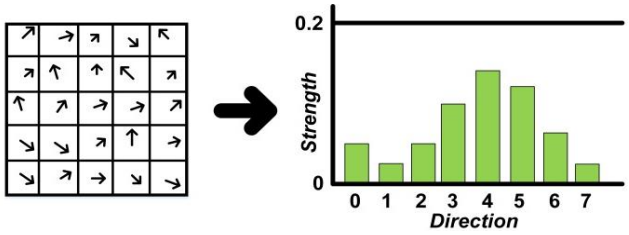


Fig. 4. The Histogram Calculation Methodology Reproduced from [13].

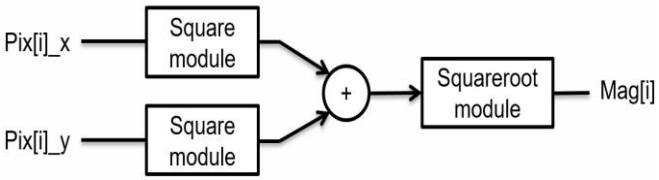


Fig. 5. Architecture for Magnitude Calculation.

D. HOG Feature Extraction

In this method, angle specific value doesn't need to be calculated, however just needed the range of angle values corresponding to the Eq (3). After the range of angle value is known, the magnitude of the pixel is added on the histogram block correspond to the angle value range. In this work, angle calculation is only compute the approximate angle, not the precise numbers so it requires two clock cycles. The architecture is shown in Fig 6.

$$\Delta L_x(x,y)\times\tan Q_{d+1} \le \Delta L_y(x,y) < \Delta L_x(x,y)\times\tan Q_d \quad (3)$$

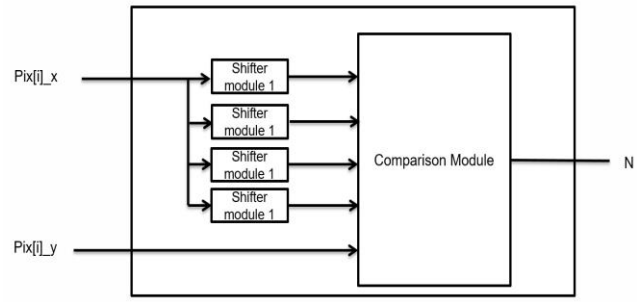


Fig. 6. Architecture for Angle Calculation.

E. The Normalization and Binarization

After histogram is obtained, normalization of the histogram collection is performed correspond with the Eq (4). Afterwards, the approached of the divisor in the Eq (4) is conducted to the power of two value that divided into 4 intervals with every near power of two values.

$$\sqrt{\left\|V_{i,j}\right\|^2+e^2} = \left\|V_{i,j}\right\| \quad (4)$$

When normalized histogram is obtained, histogram binarization is performed with classified the histogram value based on a threshold value. When the histogram value is lower than the threshold and it will be represented by "1" binary value and vice versa as illustrated in Fig 7. The normalization block requires a maximum of 19 shifters, 4 clock cycles, and using 61 stage RAM Buffer as shown in Fig 8.

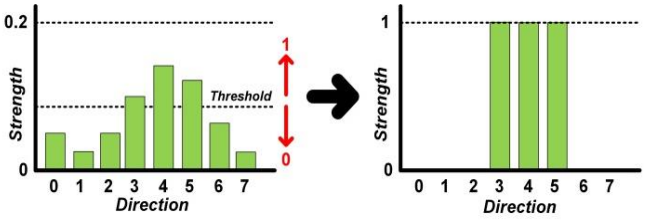


Fig. 7. Normalization Technique, Reproduced from [13].

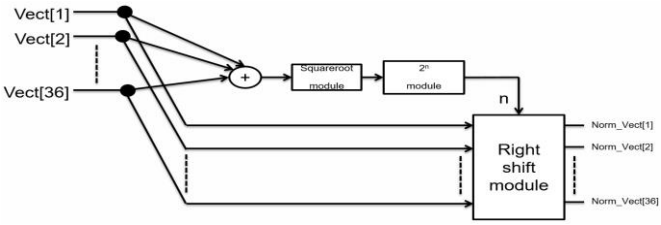


Fig. 8. Architecture for Normalization.

TABLE I. HOG-ADABOOST IMPLEMENTATION IN PSEUDOCODE

```
% Real-time Human Detection with HOG-AdaBoost Pseudocode
%Read an RGB value of each pixel from the image
read_image;
%Greyscaled an image based on the RGB value of each pixel
image_greyscaling;
%Saving the luminance value of each pixel to RAM
save_greyscaled_image;
for every_scaled_image
%Scaled down an image size with the ratio scale 1.25 every iteration
        image_scaling;
        %image window iteration with 8 pixel shifting every iteration
        for every_image_window:8:image_size
                %save image window luminance map into RAM
                temporary_save_image_window;
                for every_cells
                        %calculate gradient using imfilter
                        imfilter_image_window;
                        %converts gradient vector to polar
coordinates for each pixel
                        gradient_vector_to_polar;
                        for every_blocks
                                for every_pixel_inblock

        locate_histogram_angle;
                                        %add magnitude to
histogram based on the angle location

        add_magnitude_to_histogram(angle_location);
                                end
                        end
                        histogram_normalization;
                        saved_temporary_histogram_matrix;
                        %Humn detection using SVM with known
weight 'w' and bias 'b'
                        SVM_classifier(histogram_matrix);
                        if(SVM_score>0)
                                %detection matrix : scale_ratio
and cells information
                                save_detection_matrix;
                end
        end
        %draw a rectangle around human image
draw_rectangle(detection_matrix);
    end
```

F. SVM and AdaBoost

Detection process using HOG used SVM that can classified histogram data into two classifiers, the human is detected or not. Training is done offline using two datasets, *MIT pedestrian database*, and *INRIA database*. Then boosting is performed to increase the accuracy of detection process with combined some classifier with a different algorithm.

G. Implemented Algorithm in Pseudocode

Table I shows the pseudocode for HOG-AdaBoost Implementation in FPGA.

III. Results and Analysis

A. RTL Simulation

FPGA implementation of human detection result is shown in Table II. Every steps of RTL simulation result (as depicted in Fig 1) are shown in Fig 9 to Fig 13. Simulation result in Fig 9 shows the quadratic value process. It can be seen that the quadratic value is obtained in 2 clock cycle.

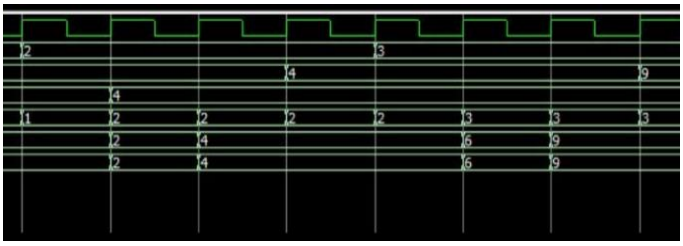


Fig. 9. The Square Root Computation Simulation Result.

TABLE II. The FPGA Implementation Result, obtained from [11]

FPGA Resources	Used	Available	%
Number of Logic Elements	83,497	114,480	72.9
Number of Registers	17,383	28,800	60.4
Number of fully used LUT-FF pairs	2070	23,109	9
Number of Memory bits	2,800,000	3,900,000	71.8
Maximum Frequency	50 MHz		

Therefore, simulation result on Fig 10 shows the magnitude calculation process which adds the two squares value and root process output if given the 16 and 9 as the input is 5.

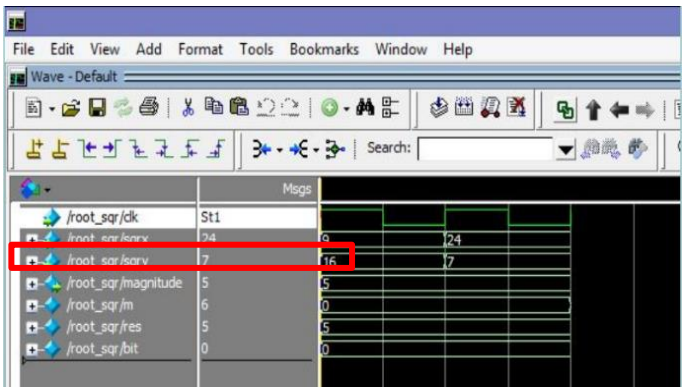


Fig. 10. The Magnitude Computation Simulation Result.

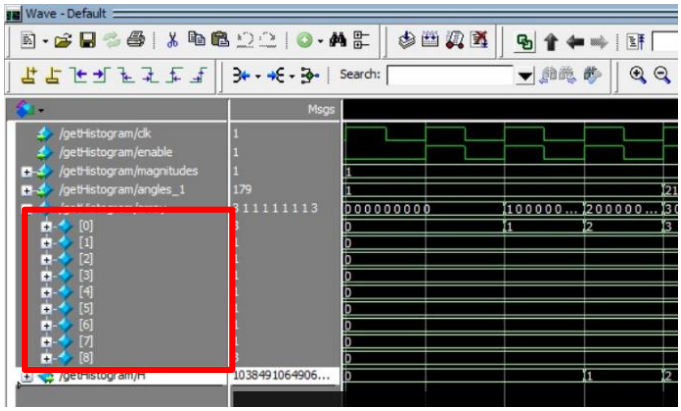


Fig. 11. The Histogram Computation Simulation Result.

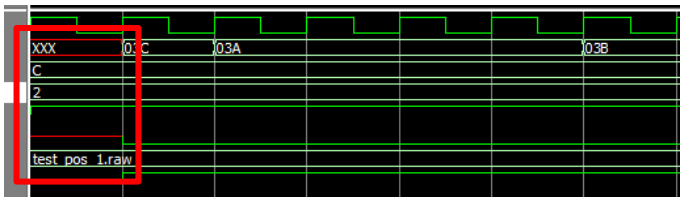


Fig. 12. The Grayscaling Simulation Result.

Fig 11 simulation result shows the histogram calculation added input value to the array with 9 elements for every clock cycle. Fig 12 shows the simulation result of the grayscaling process performed the calculation based on an equation as explained before (Eq 1) and produce output every one clock cycle.

Simulation process of histogram normalization can be seen in Fig 13. The normalization calculation result for every inputs are 300 that approached. Therefore, the divisor value is 256 and the division process can be implemented using shifter.

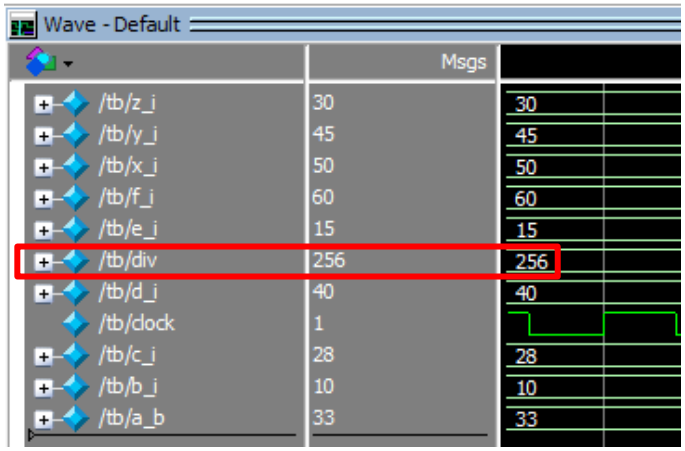


Fig. 13. The Histogram Normalization Simulation Result.

The systolic array implementation module simulation result can be seen in Fig 14. The output can be obtained after 6 clock cycles from the moment data is given, this result is 6 times faster than the usual matrix multiplication in which it is needed 3*3*3 = 27 multiplication iterations in 3 x 3 matrix multiplication.

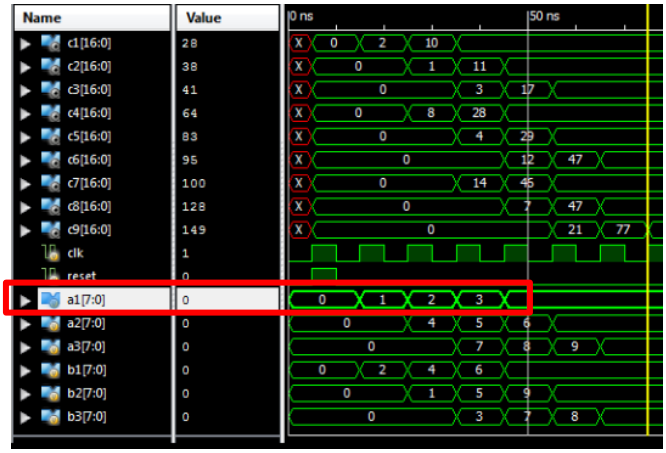


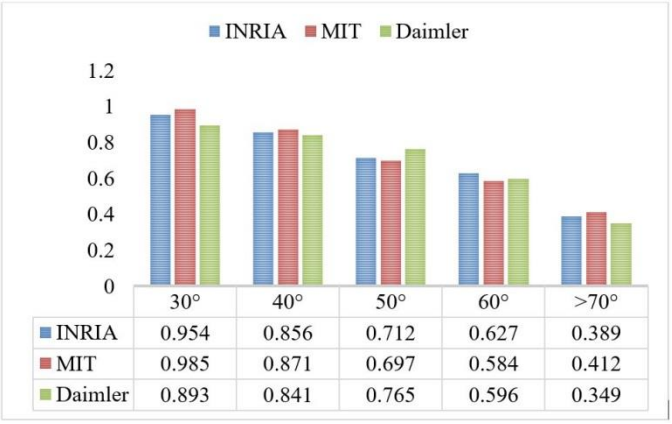
Fig. 14. The Matrix Multiplication Simulation Result.

B. Power Consumption

To measure the power consumption of FPGA DE2-115 implementation, we used The ALTERA's Power Analyzer tool and obtain 3.6 Watt total power.

C. Test with Other Angles

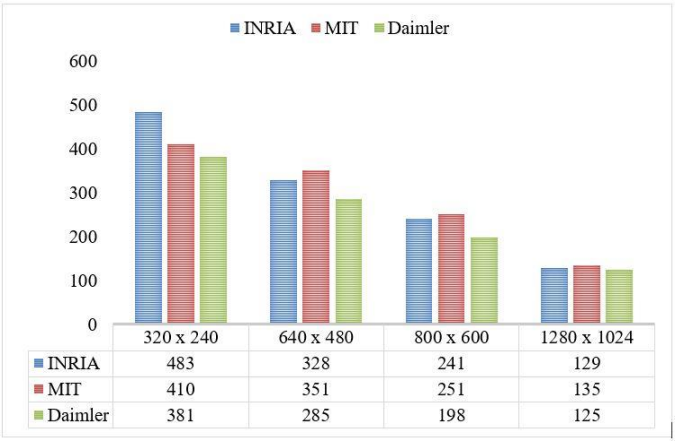
In this testing, human image from three datasets, INRIA, MIT [14], and Daimler with quite a high angle are applied to see the effect of those angle against the detection accuracy. In this testing, 5 angle categories are used as shown in Fig 15. The result is accuracy detection compared to the dataset that used for training. From this data, can be seen for the angle 50^o and below, detection process still quite reliable. However, for high angle, the advance adjustment is needed.



	30°	40°	50°	60°	>70°
INRIA	0.954	0.856	0.712	0.627	0.389
MIT	0.985	0.871	0.697	0.584	0.412
Daimler	0.893	0.841	0.765	0.596	0.349

Fig. 15. Test with High Angles (For Interpretation of the Color in this Figure Legend, the Reader is Referred to the Web Version of this Article).

Fig 16 shows table and graph of testing results with three datasets, and the results obtained as detection speed in Frame per Second (FPS). This test is done by inserting test images from the three datasets as fast as possible. Next image will be inserted after the previous image detection process is done. Detection accuracy is maintained still at more than 0.85, if less than 0.85, the pause will be given before the next image is inserted.



	320 x 240	640 x 480	800 x 600	1280 x 1024
INRIA	483	328	241	129
MIT	410	351	251	135
Daimler	381	285	198	125

Fig. 16. The FPS Tests (For Interpretation of the Color in this Figure Legend, the Reader is Referred to the Web Version of this Article).

From that results, can be seen that Daimler consistently has the lowest FPS value. This is because Daimler dataset is designed to have a high level of complexity. However, this result can't be compared with the testing using video, because in video file there is a certain thing, like encoding that not taken into account.

IV. Conclusion

In this paper, we present the human detection system using HOG features and its classifiers using AdaBoost implemented in FPGA ALTERA DE2-115. The RTL simulation shows that the calculation of HOG-AdaBoost implementation in FPGA is approaching the theory. Furthermore, we also perform the power consumption measurement for HOG-AdaBoost in FPGA. Later is the high angle test and FPS test. Based on testing, it can be concluded that for the angle<50 degrees, the detection process still reliable. However, with angle>50, needed further system optimation. One of the methods can be done by using own specific dataset for human images taken from high-angle. With new dataset, the SVM is expected to recognize human with an image taken from a high-angle.

Acknowledgment

This paper is written based on part of technical report for 20th LSI Design Contest, Okinawa, Japan, in March 2017 (International competition). In this contest, we got Analog Devices awards. We would like to thanks to everyone who help us in preparing the manuscript, especially Miss Yulian Aska (lecture assistant of VLSI course in School of Electrical Engineering and Informatics, Institut Teknologi Bandung) and our friend, another LSI Design & Contest finalist from ITB: Hans Ega, Hans Kasan, and Revie Marthensa.

References

[1] T. Adiono, H. Ega, H. Kasan, R. Marthensa, and S. Fuada, "A high frame-rate cell-based HOG human detector architecture and its FPGA implementation," *Unpublished.*

[2] J. Rettkowski, A. Boutros, D. Göhringer, "HW/SW Co-Design of the HOG algorithm on a Xilinx Zynq SoC," *J. Parallel Distrib. Comput.*, Vol.109, pp. 50–62, May 2017.

[3] K. Takagi, et al., "A Real-time Scalable Object Detection System using Low-power HOG Accelerator VLSI," *J. Sign. Process. Syst.*, January 2014. DOI: 10.1007/s11265-014-0870-7.

[4] A. Suleiman and V. Sze, "An Energy-Efficient Hardware Implementation of HOG-Based Object Detection at 1080 HD 60 fps with Multi-Scale Support" *J. Sign. Process. Syst.*, December 2015. DOI: 10.1007/s11265-015-1080-7.

[5] Y. Li, et al., "Adaptive human detection approach using FPGA-based parallel architecture in reconfigurable hardware," *Concurrency Computat.: Pract. Exper.*, June 2016. DOI: 10.1002/CPE.3923.

[6] M.D.T. Kryjak, "FPGA implementation of multi-scale face detection using HOG features and SVM classifier," *Image processing & communications*, Vol. 21(3), pp. 27-44, 2016.

[7] T.S. Saidani and Y.F. Said, "Design of embedded architecture for pedestrian detection in image and video," *IJCSNS*, Vol. 17(12), pp. 120-129, December 2017.

[8] J. Rettkowski, et al., "Real-time pedestrian detection on a xilinx zynq using the HOG algorithm," *Proc. of ReConFig*, pp. 1-8, December 2015.

[9] H. Ameur, A. Helali, and A. Youssef, "Hardwardware implementation of an improved HOG descriptor for pedestrian detection," *Proc. Of ICCAD*, pp. 406-410, January 2017.

[10] M. Hemmati, B-A, Morteza, S. Berber, and S. Niar, "HOG feature extractor accelerator for real-time pedestrian detection," *Proc. of 17th Euromicro conf. On DSD*, pp. 443-550, August 2014.

[11] T. Adiono, K.S. Prakoso, C.D. Putratama, B. Yuwono, and S. Fuada, "Practical Implementation of Real-time Human Detection with HOG-AdaBoost in FPGA," *Proc. of The IEEE Region 10 Conf.* (TENCON), October 2018.

[12] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. of the IEEE Conf. Computer Vision Pattern Recognition*, Vol. 1, pp. 886–893, June 2005.

[13] K. Negi, K. Dohi, Y. Shibata and K. Oguri, "Deep pipelined one-chip FPGA implementation of a real-time image-based human detection algorithm," *Proc. of the Int. Conf. on Field-Programmable Technology* (FPT), pp. 1-8, Decembers 2011.

[14] The MIT pedestrian database, [Online], available at http://cbcl.mit.edu/software-datasets/PedestrianData.html, accessed in August 2017.

AUTHOR'S PROFILE

Assoc. Prof. Trio Adiono received the B.Eng. degree in electrical engineering and M. Eng. degree in microelectronics from Institut Teknologi Bandung, Indonesia, in 1994 and 1996, respectively. He obtained his Ph.D. degree in VLSI Design from Tokyo Institute of Technology, Japan, in 2002. He holds a Japanese Patent on "High Quality Video Compression System". He is now a lecturer at the School of Electrical Engineering and Informatics and also serves as the Head of the University Center of Excellence on Microelectronics, Institut Teknologi Bandung. His research interests include VLSI design, signal and image processing, visible light communication, smart card, electronics solution design and integration

Christoporus Deo Putratama was born in Bandung, Indonesia, in 1995. He received the B.E. degree in electrical engineering from Institut Teknologi Bandung, Bandung, Indonesia, in 2017. During the college years, his study focused in microelectronics, and integrated circuit. In the final year at Institut Teknologi Bandung, he received 1st runner-up position and Analog Devices awards in the 20th LSI Design Contest, Okinawa, Japan. He joined PT. Riset Kecerdasan Buatan, Bandung, Indonesia, in 2017 as a researcher. His current research interests are artificial intelligence, computer vision, machine learning, and neural networks.

Bramantio Yuwono was born in Jakarta, Indonesia, 26th July 1995. He received the B.E. degree in electrical engineering from ITB, Bandung, Indonesia, in 2017. He received 1st runner-up position and Analog Devices awards in the 20th LSI Design Contest, Okinawa, Japan, in March 2017. During the college years, his study focused in embedded systems and control.

Kevin Shidqi received a bachelor's degree in Electrical Engineering, ITB in 2017. He received 1st runner-up position and Analog Devices awards in the 20th LSI Design Contest, Okinawa, Japan, in March 2017. His current field placement is Large Scale Integration and Computer Architecture

Syifaul Fuada received a B.A. in Electrical Engineering Education from Universitas Negeri Malang (UM), Indonesia, in 2014/2015 and an M.Sc. in Microelectronics Engineering from the School of Electrical Engineering and Informatics, Institut Teknologi Bandung (ITB), Indonesia, in 2016/2017. Now, he is with the University Center of Excellence at Microelectronics Institut Teknologi Bandung. He has several achievements, such as receiving one of the 106 Indonesia Innovation by BIC-RISTEK DIKTI awards in 2014, a top 10–student travel grant to the IEEE Asia Pacific Conference and Systems (APCCAS) 2016 that was held in Jeju, South Korea, receiving one of 108 Indonesia Innovation by BIC-LIPI awards in 2016, receiving best paper in IEEE IGBSG 2018 and IEEE ICTRuDev 2018. He is a member of IAENG and Associate Editor of INFOTEL Journal. His research interests include analog circuit design, circuit simulation, VLSI design, DSP, engineering education, multimedia learning development and VLC.