Hardware Implementation of Adaboost-based Sign Recognition for Miniature Robot

Yutaka Usui*† and Katsuya Kondo†
*Raytron Inc, 1-4-19 Minamimorimachi, Kita-ku, Osaka, 530-0054, Japan
E-mail: usui@raytron.co.jp Tel: +81-6-6366-0661 Fax:+81-6-6366-5072
†Dept. Information and Electronics, Graduate School of Eng., Tottori University
Minami 4-101, Koyama-Minami, Tottori, 680-8552, Japan
E-mail: kondo@ele.tottori-u.ac.jp Tel/Fax: +81-857-31-5699

Abstract— **In this paper, we propose a hardware implementation of a sign recognition system for miniature robot based on Adaboost algorithm. We describe the hardware design techniques including preprocessing, pipelined feature generator and classifier processing. The proposed pipelined architecture reduces working memory significantly compared to straightforward implementation that requires large frame memory. The architecture has been designed by using SystemVerilog and Verilog-HDL. Implementation result shows that FPGA lcell usage of Altera Cyclone-III(EP3C55F) is 44%. Evaluation result shows that the error rate is 6% or less, and processing time is within video frame rate.**

I. Introduction

When robots are moved out of factory and introduced into our daily lives, they have to face many challenges such as moving in uncertain environments. Routing miniature robot in a room is also challenging task. One approach for this task is monitoring for environmental information and building a map. Many works have done about this approach. Therefore this approach needs computational complexity. Another approach is using visible sign as landmarks for robot. Concretely, visible sign for robot are placed in advance. Routing robot have vision sensor and image recognition function can find the sign and recognize it. The visible sign functions as a landmark for the robot. This approach is cost effective because the sign need not to be intelligent data carriers using infrared or wireless wave. To this end, we implemented sign recognition algorithm to an FPGA.

Moving miniature robots are small size. It means computational resource is extremely limited, and the sign recognition algorithm should be implemented into the limited resources.

Proposed pipeline architecture reduces working memory. This can reduce logic and memory implemented to an FPGA.

Many papers have focused on a design and implementation for hardware real time object detection. Nair *et al.* implemented an embedded system for human detection on an FPGA[1]. The system can detect people at speed of 2.5 frames per second. However, It requires large size SDRAM memory. Cho *et al.* designed and implemented a face detection system[2] in Xilinx Vertex-5 FPGA using scalable architecture. However, It uses 41 BRAMs for 320x240 (QVGA) resolution images. The 41 BRAMs on Virtex-5 device corresponds 1.4Mbit memory.

This paper organized as follows: In Section II, overview of the algorithm is described. In Section III, we propose hardware implementation. Section IV describes evaluation of the proposed architecture. Section V concludes this paper.

II. Algorithm

This section describes an algorithm for sign recognition. It is based on Viola-Jones face detector [3][4].

Firstly, images from an camera is preprocessed. Secondly, it computes haar-like feature value. Thirdly, classification by using adaboost has done. Note that we do not deal here with a region detection method in this paper.

A. *Preprocessing*

Image data from camera module is YUV color space represented. YUV data had been converted to L*a*b* color space. The reason why using L*a*b* is possibility for colored sign in the future, and L*a*b* was better result than YUV in preliminary experiment.

B. *Haar-like feature generator*

Haar-like features have values that represent differences in average intensities between adjacent rectangular regions. They can extract texture information without depending on absolute intensities or color value. They are not too much affected by noise because it is based on integral value. Figure 1 shows 2 types of haar-like features. We used horizontal and vertical haar-like features.

C. *Classification using adaboost*

AdaBoost is an effective method to form a highly accurate predictor combined by many weak learners. Fig.2 shows general adaboost procedure[5]. In proposed system, training data have generated on the PC software in advance. The trained data is programmed parameter ROM area in the FPGA.

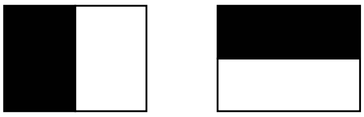


Fig. 1 Example of haar-like features. The sum of pixels which lie within the white rectangles are subtracted from the sum of pixels in the black rectangles.

```
1. Prepare a set of N labeled samples as
   $(x_1, y_1), \cdots, (x_N, y_N).y \in \{+1, -1\}$ is the class
   label associated with the sample image $x_i$ .
2. Initialize weights $D_1(i) = \frac{1}{N}$ .
3. For t=1,….,T:
   (A) For each feature, calculate a feature value.
   (B) Train a weak classifier.
   (C) Choose $h_t(x)$ with the lowest error $\varepsilon_t$ .
       The error is evaluated with respect to the
       sample weight $D_t(i)$.,
       $\varepsilon_t = \sum_{i:y_i \neq h_t(x_i)} D_t(i)\cdot$
   (D) Update the weight
       $D_{t+1}(i) = \frac{D_t(i)\exp(-y_i \alpha_t h_t(x_i))}{\sum_i D_t(i)\exp(-y_i \alpha_t h_t(x_i))}$
       Where $\alpha_t = \frac{1}{2}\log\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)$
4. Output the final classifier:
   $H(x) = sign(\sum_{t=1}^{T} \alpha_t h_t(x))$
```

Fig. 2 General adaboost learning procedure

III. IMPLEMENTATION TO FPGA

This section describes an architecture and implementation to FPGA. We have implemented the algorithm on the Altera Cyclone III FPGA. Hardware description is written in SystemVerilog and Verilog-HDL. The fixed point hardware implementation was developed based on the floating point software version. Since the software uses floating point, testing was required to convert the software implementation to work with fixed point operations.

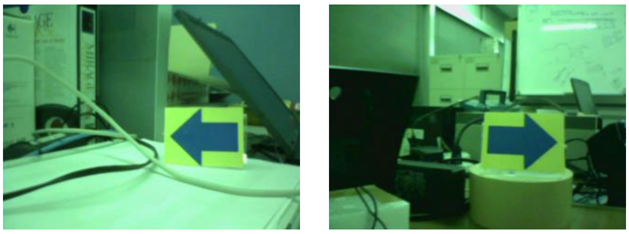
TABLE I
CAMERA MODULE SPECIFICATION

Pixel Resolution	640x480x8bit
Pixel Clock	24MHz
Frame rate	0-30fps

MC5VC camera module provided by Konica Minolta for mobile phone applications was used for this work. Table1 shows the specification.

Fig.4 shows a block diagram of the hardware design. An image data from camera module leads to camera I/F module, which converts camera data serial to parallel. Image data from camera I/F leads to matrix and converted to L*a*b* color space. Converted data had sent to haar-like feature generator and adaboost classifier. Parameters of haar-like features are programmed in the ROM.

Interface using USB2.0 port is implemented for test purpose. This interface is for capturing raw image data to PC, testing and debugging. Fig.3 shows images from onboard camera captured via the interface.



(a)Left Arrow 'LT' (b)Right Arrow 'RT'

Fig. 3 Images from Onboard Camera

A. *Pipelined Haar-Like Feature Generator*

This section describes a pipelined haar-like feature generator. The most popular implementation of haar-like feature generation is based on *integral image* [3], defined as the summation of the pixel values of the original image. However, the method requires large memory since all summation pixel data should be reserved in working memory.

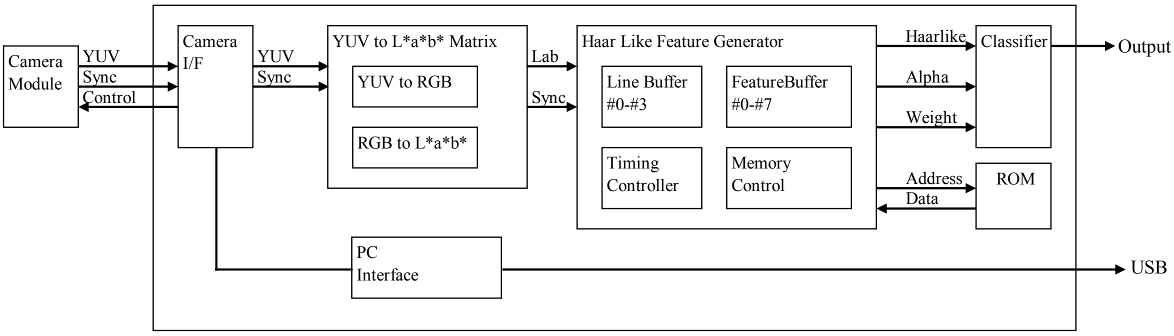


Fig. 4 Hardware architecture block diagram

Now, Fig.5 shows the pipelined haar-like feature generation timing. This architecture does not use integral image, use multiple line and feature buffer. One weak learner corresponds one haar-like feature. The position, size and type (vertical/horizontal) of each feature have determined in training stage. That is, position of each haar-like feature is fixed at training time. The original image is row-column scanned. Since, if all haar-like feature position had sorted in advance, haar-like position can be predictable.

Concretely, when scanning image data comes to haar-like feature #1 area, its pixel data is summed and stored to line buffer #0. Likewise, when image data comes to haar-like feature #2 area, the data is stored to line buffer #1. When scanning image comes to end of horizontal right edge, the data is summed to feature buffer. Each line buffer and feature buffer should be controlled adequately.

So, fully pipelined processing synchronized with video timing is available, and no large working memory required. This architecture contributes small circuit size and low power consumption.

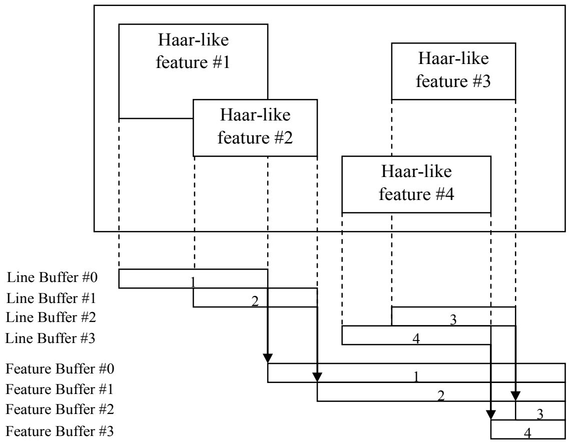


Fig.5 Feature Buffer timing

B. Classifier Implementation

Fig.6 shows the structure of adaboost classifier procedure.

The threshold value and weight had read from ROM sequentially, according to video timing.

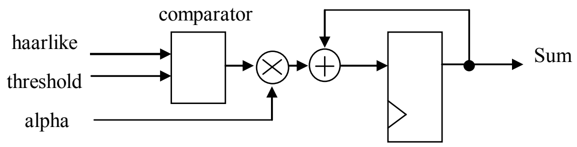


Fig. 6 Adaboost classifier architecture

IV. EXPERIMENTAL RESULTS

We tested proposed system by using sample image data from onboard camera. Signs for recognition targets are the "RT" and the "LT" as shown in Fig.8 . 200 images of the "RT" and 200 images of the "LT" were used as positive samples. 500 negative images were used. Fig. 8 shows some examples of training image data. This images are come from onboard camera. 325 negative, 275 of the "RT" and 275 of the "LT" images were used for testing. Figure 9 shows examples of testing data.

Table II shows error rate using test samples (1). Error rates are 5% or less. It shows efficient rate for the purpose.

Furthermore, we tested another shape of signs. 500 negative, 300 of the "RT" and 300 of the "LT" images were used for training. Fig. 10 shows examples of testing data. 250 negative, 300 of the "RT" and 300 of the "LT" images were used for testing. Fig. 11 shows examples of testing data. Table III shows experimental results. The error rates are 6% or less unless another shape of signs mixed. This result shows that

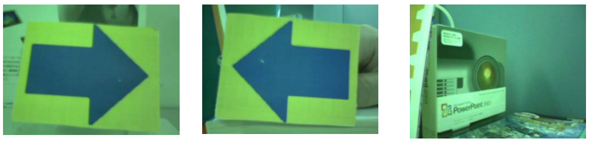

(a) RT (b) LT (c)NEG

Fig. 8 Training Samples(1)

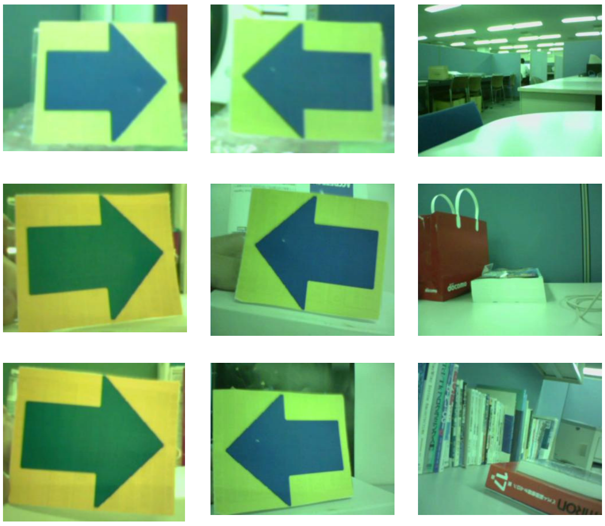
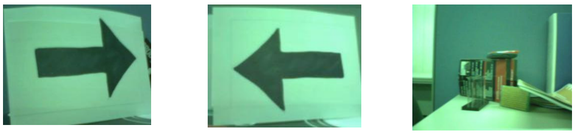

(a) RT (b) LT (c)NEG

Fig.9 Test Samples(1)

TABLE II
TEST RESULT

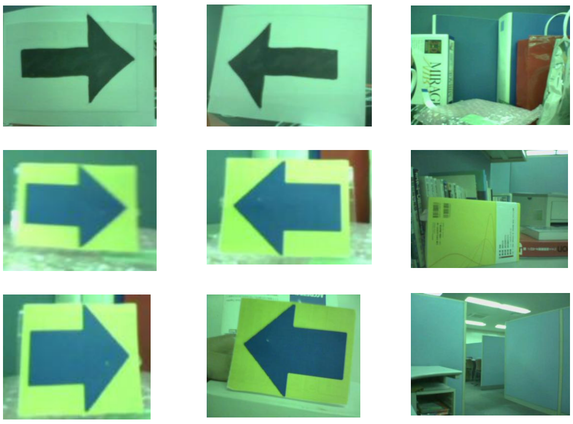
Class	Error Rate (%)
NEG	0.364
RT	4.692
LT	4.769

the algorithm have efficient accuracy for the application.
Table IV shows the statistics for the implementation of proposed system on CycloneIII device. The circuit size is quite small through proposed implementation, and maximum clock frequency have sufficient margin for 24MHz camera pixel clock. The total memory requirements of the system are less than 1.6Kbyte.



(a)RT (b)LT (c)NEG

Fig.10 Training Samples(2)



(a)RT (b)LT (c)NEG

Fig.11 Test Samples(2)

TABLE III
TEST RESULT

Class	Error Rate (%)
NEG	5.46
RT	1.91
LT	3.81

TABLE IV
FPGA DESIGN SUMMARY

Device	Altera CycloneIII(EP3C55F)
Total Logic elements	24573/55856 (44%)
Total Memory bits	12800/2396160 (<1%)
Clock Fmax	27.5MHz

V. CONCLUSIONS

We presented an FPGA implementation of sign recognition for miniature moving robot. The pipelined architecture reduces working memory significantly. And the size of the proposed FPGA circuit revealed its low cost in terms of logic and memory (less than 1.6Kbyte). Evaluation result shows that the error rate is 6% or less. Processing time is within video frame rate.

Recognition performance could be further improved using video frame rate filter. Future work includes implementing region detection method.

REFERENCES

[1] Vinod Nair, Pierre-Olivier Laprise, and James J. Clark, “An FPGA-Based People Detection System,” EURASIP Journal on Applied Signal Processing, vol. 2005, no. 7, pp.1047-1061, 2005

[2] Junguk Cho , Shahnam Mirzaei , Jason Oberg , Ryan Kastner, “Fpga-based Face Detection System using Haar Classifiers,” Proc. of the ACM/SIGDA Int'l symposium on Field Programmable Gate Arrays, pp. 103-112, 2009.

[3] Paul Viola, Michael J.Jones, “Rapid Object Detection using a Boosted Cascade of Simple Features,” Proc. of IEEE Computer Vision and Pattern Recognition (CVPR), pp.511-518, 2001.

[4] Rainer Lienhart and Jochen Maydt, “An Extended Set of Haar-Like Features for Rapid Object Detection,” Proc. Int'l, Conf. on Image Processing, Vol.1, pp. 900-903, 2002.

[5] Takafumi Kanamori, Kouhei Hatano and Osamu Watanabe, Boosting. :Morikita Publishing ,2006. (In Japanese)