A Low-Power AdaBoost-Based Object Detection Processor Using Haar-Like Features

Motoki Kimura, Janarbek Matai, Matthew Jacobsen and Ryan Kastner
Computer Science and Engineering
University of California, San Diego
La Jolla, California 92093
Email: {mokimura, jmatai, mdjacobs, kastner}@cs.ucsd.edu

***Abstract*—This paper presents an architecture of a low-power real-time object detection processor using Adaboost with Haar-Like features. We employ a register array based architecture, and introduce two architectural-level power optimization techniques; signal gating domain for integral image extraction, and low-power integral image update. The power efficiency of our proposed architecture including nine classifiers is estimated to be 0.64mW/fps when handling VGA(640×480) 70fps video.**

***Keywords*—Object detection, Haar-Like features, VLSI**

I. Introduction

Object detection, which finds particular objects from an image frame plays an important role in a wide range of embedded applications such as wireless sensors, video surveillance, and advanced driver assistance. To develop embedded systems for these applications on battery-operated devices, power consumption is considered as one of the most important factor.

Many different object detection algorithms have been proposed in recent decades. Among them, AdaBoost with Haar-like features [1] is commonly used because of it's low computational cost. In this algorithm shown in Fig. 1, a trained feature data set of weak classifiers are connected to form a cascaded classifier which can detect particular objects. A detecting window which scans original and scaled images (layers) is verified by this classifier. Each weak classifier has a simple feature composed of two or three rectangles. Thus, feature value for each classifier is simply calculated from integral image values $I(x, y)$ at all corners of the rectangles, after all integral image values in a detecting window are generated. This simple calculation method helps reduce computational cost and hardware size.

For that reason, many hardware architectures employing AdaBoost with Haar-like features have been proposed [2]–[5]. [2] is a chip implementation which stores integral image values into memories to reduce chip size. But the performance of this implementation suffers from limited bandwidth between these memories and a classifier module. In order to obtain high performance characteristics, [3]–[5] employs register arrays to store integral image values, and have multiple classifier modules operated in parallel. However, these architectures do not focus as much on power consumption which is critical for embedded applications.

Therefore, we develop a register array based architecture, aiming at sufficient performance for real-time object detection.

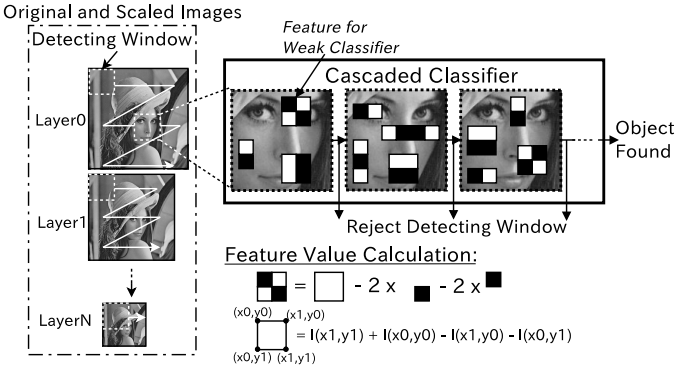

Fig. 1. Object detection using Adaboost with Haar-Like features.

Then, we introduce two power optimization techniques to these arrays; signal gating domain for integral image extraction and low-power integral image update, which are our main focus in this paper.

By applying these two techniques, 41% of dynamic power can successfully be reduced, and simulation results for frontal face detection application show that the power efficiency of our proposed object detection processor is estimated to be 0.64mW/fps using 45nm CMOS technology. Moreover, parallel processing by nine classifiers provides performance of VGA 70fps with 93% detection rate.

The rest of this paper is organized as follows. In Section II, an overview of our object detection processor is described. Our proposed power optimization techniques are introduced in Section III. Section IV shows our implementation results and comparison with previous works. Section V concludes this paper and gives future works.

II. Architecture Overview

Fig. 2 shows an overview of our proposed architecture. The size of a detecting window is 20×20 pixels, and the maximum supported size of input image is 640×480.

This architecture connected to buses by two interfaces is composed of pixel line memory(PM), three classifier memories(CM), integral image generator(IG)/extractor(IE), squared integral image generator(SIG), three variance calculation unit(VC), and nine object classifier pipeline(OC)s in three triple classifiers to verify detecting windows. A 64-bit bus interface is implemented to fetch pixel data of image layers

978-1-4799-1412-8/13/$31.00 ©2013 IEEE

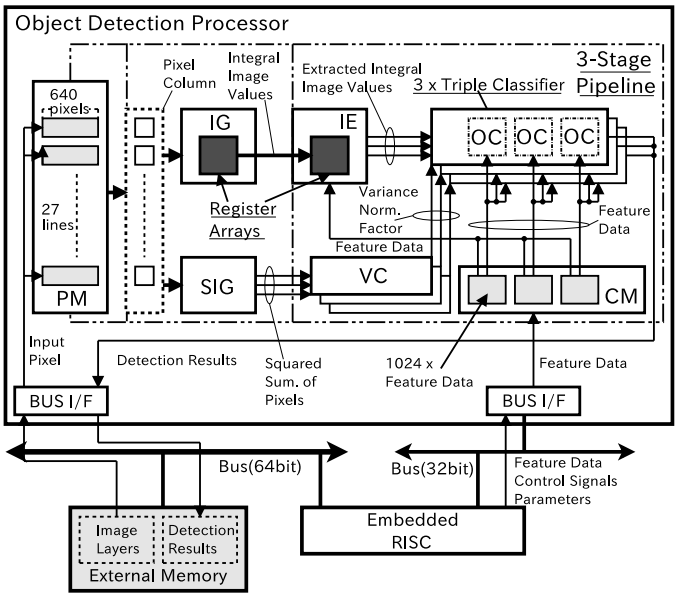

Fig. 2. Architecture overview.

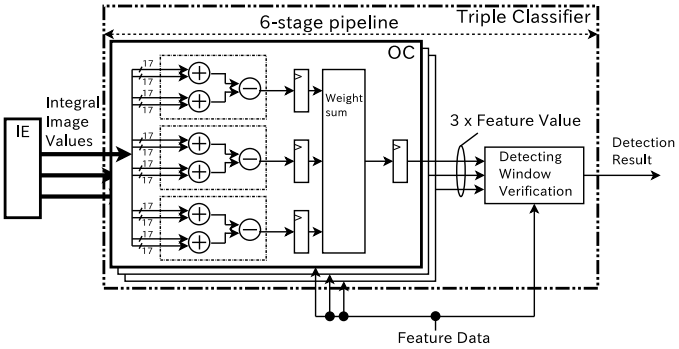

Fig. 3. Triple classifier.

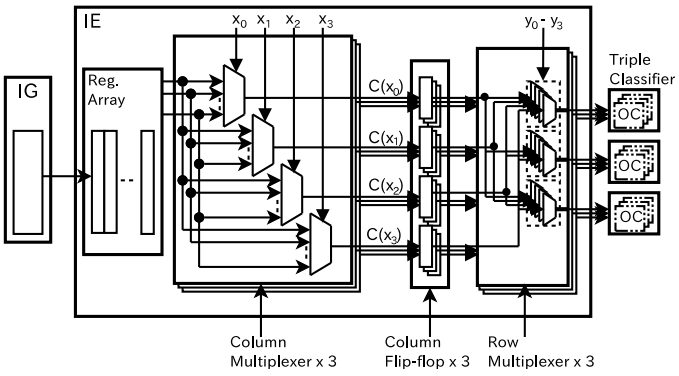

Fig. 4. IE architecture.

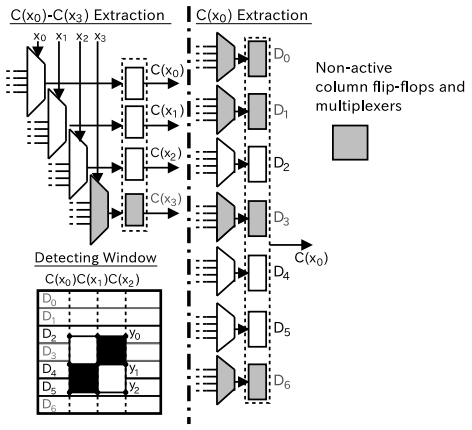

Fig. 5. Signal Gating Domain in IE.

from the external memory, and to write the coordinates of detected objects. Integration of PM connected to the 64-bit bus interface allows the architecture efficient transfer of pixel data from the external memory. The memory size of PM including a prefetch buffer is 640 pixels $\times$27 lines to support a 20×20 detecting window.

Embedded RISC can send a feature data set and parameters to this object detection processor through a 32-bit data bus. Therefore, by changing the feature data set depending on a particular object type such as a frontal face, a human body, and a traffic sign, the processor can be utilized to detect various types of objects. CM is integrated as a 3-bank on-chip memory and has capacity of storing 1024×3 feature data. Each bank of CM is connected to one OC in each triple classifier, so that the triple classifier can read three feature data in one cycle. In this work, we do not adopt any classifier data compression techniques described in [2], [4] to support OpenCV non-tilted feature types [6]. By applying these techniques, we can reduce the size of CM.

In order to obtain sufficient performance, three detecting windows are processed in parallel by triple classifiers. To derive the processing capability of nine OCs, efficient integral image generation and extraction are also required. We adopt 3-stage pipeline structure and implement two register arrays into IG/IE to satisfy this requirement. In this 3-stage pipeline, the register array in IE behaves as a large pipeline register, while the register array in IG behaves a barrel shifter for integral image generation. Detail of IE and IG is described in Section III-A and Section III-B.

The triple classifier shown in Fig. 3 is based on a 6-stage pipeline to be operated at a high clock frequency. This module is mainly composed of three OCs so that it can calculate three feature values in one cycle. The integral image update process described in [3] limits input bit width of operators in OC to 17bit and reduces the area of OC. This approach also reduces required bit width for representing each integral image value in a detecting window, which results in eliminating unnecessary flip-flops or multiplexers in the register arrays.

III. Power Optimization Techniques

Our proposed architecture described in Section II has two large register arrays driven by a high-frequency clock signal. Therefore, dynamic power consumption forms a dominant part in total power consumption of this processor. In this work, we focus on dynamic power of register arrays constituted from a large number of flip-flops and multiplexers and introduce two architectural-level power optimization techniques.

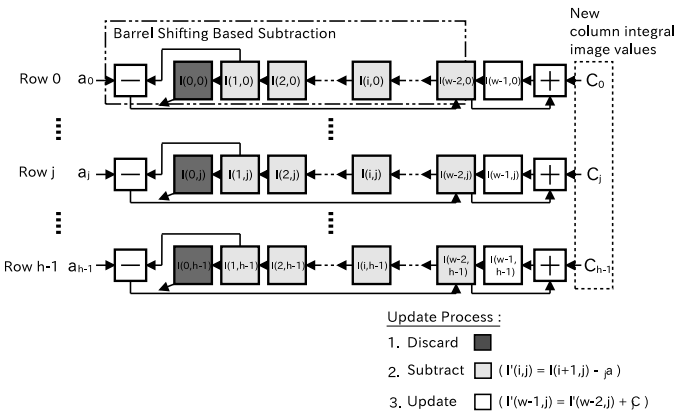

Fig. 6. Integral image update process in IG.

A. Signal Gating Domain

Fig. 4 shows IE architecture. IE is mainly composed of the register array, the column multiplexer, the column flip-flop, and the row multiplexer. First, IE reads $x_0 \sim x_3$ and $y_0 \sim y_3$ which represent coordinates of all rectangle corners in a feature from CM. Then, the column multiplexer selects 4 columns $C(x_0) \sim C(x_3)$ including integral image values at all corners from the register array, and stores them in the column flip-flop. After that, the row multiplexer extracts integral image values at all corners from stored columns using $y_0 \sim y_3$, and sends them to OCs. The maximum number of integral image values extracted in one cycle is 12 corners/feature $\times$ 3 features $\times$ 3 windows $= 108$.

It is clear that most of integral image values in selected columns are not extracted to OCs, and we divided a detecting window into 7 domains($D_0 \sim D_6$) shown in Fig. 5. If all of $y_0 \sim y_3$ of a feature are outside of a domain, any integral image value in the domain is not used for calculating the feature values in OCs. In the example shown in Fig. 5, some integral image values in D_2, D_4 and D_5 are used for calculating the feature value of F. On the other hand, any integral image value in D_0, D_1, D_3 and D_6 is not used. Thus, the column flip-flops and multiplexers for these unused domains are not active. In addition, when any integral image value in each column $C(x_0) \sim C(x_3)$ is not used, the column flip-flops and multiplexers for the column are not active.

IED module which consumes only 3Kgates automatically detects unused domains and columns from each feature data loaded from CM. Then, it disables clock supply to the column flip-flops and stop signal toggles of column multiplexers for the domains and columns. By using this technique, we can reduce dynamic power by 30%.

B. Low-Power Integral Image Update

Fig. 6 shows the integral image update process in IG when a detecting window moves to its next position. As described in Section II, we implement as many flip-flops as required in the register array, by calculating optimal bit width for each row. In this process, a_j is subtracted from $I'(0,j) \sim I'(w-2,j)$ in each row j by equation (1), before calculating new integral image values $I'(w-1,j)$, where w is the width of a detecting window.

$$I'(i,j) = I(i+1,j) - a_j, \text{ where } 0 \le i \le w-2 \quad (1)$$

This subtraction is executed using a barrel shifting operation in IG, which is shown in Fig. 6. However, all bits of all elements in the array have to be shifted to their next position continuously during the subtraction. Thus, all flip-flops in the array require clock supply in the subtraction, which causes large power consumption.

Meanwhile, value R for a rectangle in a non-tilted Haar-Like feature is obtained from equation (2), where $I'(x_0,y_0)$, $I'(x_0,y_1)$, $I'(x_1,y_0)$, and $I'(x_1,y_1)$ are integral image values at corners of the rectangle.

$$\begin{aligned} R &= I'(x_1,y_1) + I'(x_0,y_0) - I'(x_0,y_1) - I'(x_1,y_0), \\ &\quad \text{where } x_0 < x_1, \;\; y_0 < y_1 \end{aligned} \quad (2)$$

By applying equation (1), equation (2) can be written as equation (3).

$$\begin{aligned} R &= (I(x_1+1,y_1) - a_{y_1}) + (I(x_0+1,y_0) - a_{y_0}) - \\ &\quad (I(x_0+1,y_1) - a_{y_1}) - (I(x_1+1,y_0) - a_{y_0}) \\ &= I(x_1+1,y_1) + I(x_0+1,y_0) - \\ &\quad I(x_0+1,y_1) - I(x_1+1,y_0) \end{aligned} \quad (3)$$

From equation (3), R is independent from a_{y_0} and a_{y_1}, and any value can be used for a_j in the subtraction. Therefore, in order to reduce the power consumption in the subtraction, optimal a_j for each row j of the register array is calculated from equation (4), where s_j is defined in Table I.

$$a_j = (I(1,j) \gg s_j) \ll s_j \quad (4)$$

TABLE I. DEFINITION OF s_j VALUE.

j	s_j	j	s_j
0	8	$3 \sim 6$	11
1	9	$7 \sim 14$	12
2	10	$15 \sim$	13

Equation (4) generates optimal a_j by masking LSB s_j bit of $I(1,j)$ to 0, which means LSB s_j bit of $I'(i,j)$ and $I(i+1,j)$ in equation (1) are always equal. Also, no additional circuit to generate optimal a_j is required. Fig. 7 shows our proposed subtraction with optimal a_j, where LSB s_j bit of integral image value is not required to be shifted. After the subtraction, $I'(w-1,j)$ is calculated using new column integral image value C_j in one cycle, when LSB s_j bit is shifted. By disabling clock supply to flip-flops for these non-shifted bits which account for 70% of flip-flops in the register array during the subtraction, an additional 11% of dynamic power reduction is achieved. It should be noted that this approach can be applied only to Haar-Like non-tilted feature, because it depends so much on the calculation of Haar-Like feature value.

IV. IMPLEMENTATION RESULTS

Our proposed object detection processor is synthesized using TSMC 45nm LVCMOS library. The number of gates and SRAM size is 475Kgate and 45KB at the target clock frequency of 400MHz, respectively. The maximum performance

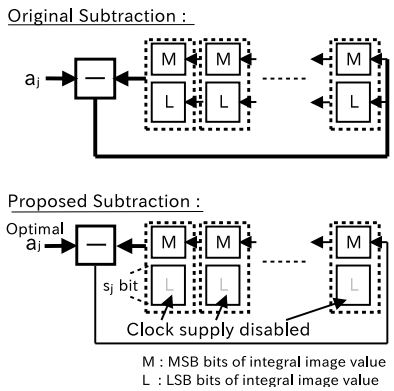

Fig. 7. Proposed subtraction.

is VGA 70fps when the number of image layers is 18. Gate-level simulation results under worst condition show that power consumption and efficiency is estimated to be 45.2mW and 0.64mW/fps for frontal face detection application, respectively. Implementation results are summarized in Table II.

Table III shows comparison with previous Adaboost-based object detection processors using Haar-Like features. For fair comparison, our processor supporting a 24×24 detecting window is also implemented, and evaluated using the same OpenCV face detection data set [6] used in [4]. Our frame rate is 83fps and is less than [4], but the performance is competitive by taking the number of layers into account. Also we verified that our architecture has a sufficient detection rate using 1000 test images which are mainly obtained from the face database in [7] and contain several frontal faces of different sizes. Fig. 8 shows sample detection results from the verification.

The power efficiency of our processor supporting a 24×24 detecting window is 0.61mW/fps, which is smaller than previous works. [2], [5] measured power consumption from a test chip fabricated by using several power implementation techniques such as multi-threshold voltage. These techniques will help reduce the power consumption when our processor is implemented into a chip.

V. Conclusion

In this paper, we introduce two architecture-level power optimization techniques, so as to reduce power consumption of an AdaBoost-based object detection processor using Haar-Like features. Simulation results using 45nm CMOS technology show that our proposed architecture including nine classifier pipelines consumes 45.2mW when handling VGA 70fps video for frontal face application. These results show that out proposed object detection processor is suitable to be integrated in an embedded SoC which has low-power object detection requirements. We plan to implement our architecture into a test chip and measure the power consumption.

TABLE II. Implementation Results

Technology	TSMC 45nm
Supply Voltage	0.72V
Resolution	640x480
Gate Count (2NAND) / SRAM	475K / 45KB
Operating Frequency	400MHz
The Number of Image Layers	18
Maximum Frame Rate	70fps
Power Consumption	45.2mW

TABLE III. Comparison with previous works.

	MICRO [2]	VLSI [4]	SOVC [5]	This Work	
Window Size	24x24	24x24	32x32	20x20	24x24
Technology	90nm	65nm	40nm	45nm	
Supply Voltage	0.9V	1.0V	0.9V	0.72V	
Gate Count	180K$*1$	21M$*1$	900K$*2$	475K	586K
SRAM	28KB	53KB	24KB	45KB	49KB
Operating Frequency	54MHz	800MHz	220MHz	400MHz	
# Image Layers	14$*3$	12$*4$	9	18	17
Maximum Frame Rate (VGA)	2fps$*3$	118fps	72fps	70fps	83fps
Detection Rate	81%	95%	90%	93%	90%
Power Efficiency (mW/fps)	1.86$*3$	2.76$*3$	2.15$*5$	0.64	0.61

$*1$: Gate count is estimated from the number of transistors.
$*2$: Gate count is estimated from chip micrograph.
$*3$: Resolution is up-scaled to VGA for comparison.
$*4$: The number of layers is estimated from the scaling factor.
$*5$: Power efficiency is calculated from fps/W of DT mode.

Fig. 8. Sample Frontal Face Detection Results.

References

[1] P. Viola and M. Jones, "Rapid Object Detection Using a Boosted Cascade of Simple Features," in *Proc. of IEEE Computer Vision and Pattern Recognition*, pp.511–518, Dec. 2001.

[2] Y. Hori, Y. Hanai, J. Nishimura, and T. Kuroda, "Architecture Design of Versatile Recognition Professor for Sensornet Applications," *IEEE Micro*, vol. 29, no. 6, pp.44–57, Nov./Dec. 2009.

[3] J. Cho, S. Mirzaei, J. Oberg, and R. Kastner, "FPGA-Based Face Detection System Using Haar Classifiers," in *Proc. of International Symposium of Field Programmable Gate Arrays(FPGA)*, pp.103–112, Feb. 2009.

[4] K. Christos and T. Theocharides, "A Flexible Parallel Hardware Architecture for AdaBoost-Based Real-Time Object Detection," *IEEE Trans. on Very Large Scale Integration Systems*, vol. 19, no. 6, pp.1034–1047, June 2011.

[5] Y. Tsai, T. Yang, C. Tsai, K. Huang, and L. Chen, "A 69mW 140-meter/60fps and 60-meter/300fps Intelligent Vision SoC for Versatile Automotive Applications," in *Proc. of IEEE Symposium on VLSI Circuit*, pp.152–153, June 2012.

[6] Open Computer Vision Library, Mar. 2009. DOI:http://sourceforge.net/projects/opencvlibrary/.

[7] V. Jain and E. L. Miller, "FDDB: A Benchmark for Face Detection in Unconstrained Settings," Tech. Rep. UM-CS-2010-009, Dept. of Computer Science, University of Massachusetts, Amherst, 2010. DOI:http://vis-www.cs.umass.edu/fddb/.