Design and Implementation of Extracting Haar-like Feature IP in the Image of Front-vehicle

Meihua Xu[1, a], Gaopan Chen[1, b], and Aiyin Guo[1,2, c]

[1]Microelectronic R&D Center, Shanghai University, Shanghai, 200072, China

[2]Department of Electrical Engineering, Shanxi Light Industry Vocational and Technical College, Taiyuan, 030013, China

[a]mhxu@shu.edu.cn, [b]shu_cgp@shu.edu.cn, [c]gayshh@shu.edu.cn

Keywords: Haar-like Feature; Pipeline; Data multiplex; IP design

Abstract. This paper presents an integrate circuits IP for extracting Haar-like features in the image of front vehicle. We improve the performance of the IP by selecting the architecture of integral image, pipeline, ping-pong operations, data multiplexing etc. The verification of the IP is based on the VCS, and the result shows that the processing time of whole image is twice as much as transporting time, and the storage space of integral image requires only 59kbits.

Introduction

In the methods of detecting front vehicle, research based on vision detect vehicle according to intuitive visual information [1], such as symmetry of front-vehicle [2], cast shadow [3] and so on in early days. With the development of research, the hotspots have turned to Histograms of Oriented Gradients (HOG) [4] and Haar-like [5] [11] which are more robust and accurate. Beleznai [6] who detected pedestrian using GPU having achieved relatively good results, but this hardware is hardly used in embed system on vehicles. In paper [7], FPGA is used to detect front-vehicle, but the robustness and exactness is not very well as using intuitive visual information. In this paper we present an IP design of quickly extracting Haar-like features in the image of front vehicle and it can be used in FPGA and ASIC design.

Haar-like Feature

Haar-like Feature is named by Haar wavelet， and the value of Haar-like is the sum of pixels value in white block minus the sum of pixels value in black block. For a 24x24 pixels sub-window, its total number of Haar-like [9] [10] feature is more than 160, 000, so the speed of training and detection will decrease. In order to calculating the Haar-like feature faster, Viola presents the integral image. The equation of integral image is shown as follow:

$$ii(x, y) = \sum_{x' \le x, y' \le y} i(x', y') \quad (1)$$

In the eq.1, ii(x, y) represents the value at the point of (x, y), i(x’, y’) represents the gray value at point (x’, y’). We can understand it from the following picture.

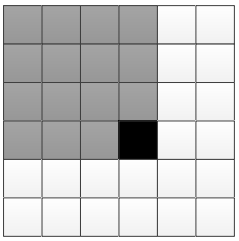

Fig.1 integral image

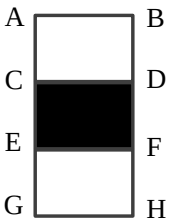

Fig.2 template of Haar-like feature

In Fig.1, the integral value at black point equals to the sum of all gray pixels and black point. The Fig.2 is a template of Haar-like feature and the value is Y in eq.

$$Y = A - B - 2C + 2D + 2E - 2F - G + H \quad (2)$$

We can calculate the value of Haar-like very quickly in this way.

Design of extracting Haar-like Feature

In consideration of the concept about integral image proposed by Violap, we can realize the full integral image naturally. In this way, the full integral image should be calculated firstly, next step is for Haar-like feature in the every sliding window. The method of full integral image is very simple, but it needs lots of Storage Resources, so it's hardly used in FPGA or ASIC with on chip memory.

Lai took the structure of partial integral image [8], which calculates the integral image only for horizontal part of the full picture, and the Structure diagram shows in fig.3

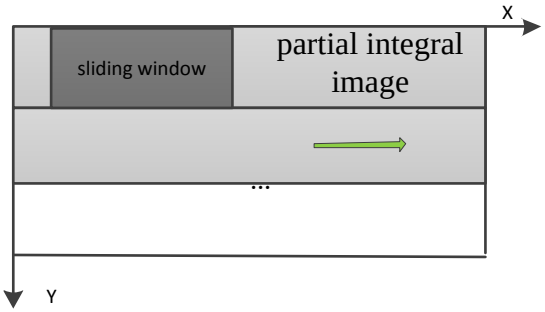

Fig.3 integral image structure

In paper [8], the operation mode divides the whole image into m lines firstly, which have the same width with sliding window, and then calculates integral image for every part(the grey part). The second step copies data from partial integral image to sliding window which contains many registers, and then calculates Haar-like feature for every sliding window. This method takes less memory, but not less enough to only take on chip memory to store integral image. When the integral image is being generated, the process of calculating Haar-like feature is stopped.

This paper presents a new structure to calculate Haar-like feature faster. Firstly, the method calculates Haar-like feature with on-chip RAM, but can't takes full integral image as there is not enough RAM. To solve it, the partial integral image can be further divided: the integral image stores data in one sliding window only. The storage resources will be reduced by using presented structure, but the calculating speed is very slow because we need to recalculate the next integral image every time after extracting Haar-like feature. So we present the calculating and storage structure as fig.4

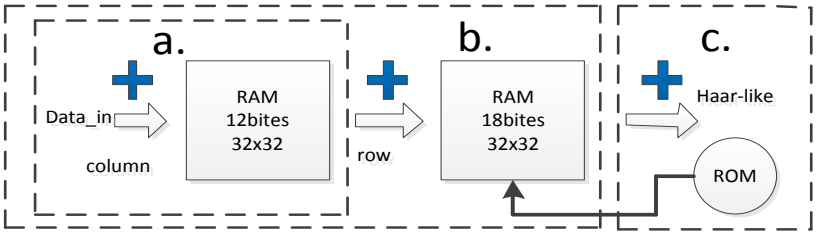

Fig.4 structure of storage and calculating

As shown in fig.4(a), the data coming from data_in port in the module of extracting Haar-like feature will be stored by registers and then calculated, stored to RAM. This way avoids traditional several cycle operations (reading, calculating, and storing) through pipeline structure. But we can't get the Haar-like feature if it calculates every feature template only once, so we need a structure which separates column-integral and row-integral, as fig.4(b) shown. The next step is calculating Haar-like feature by feature template and integral image as in fig.4(c). The ROM is used to store the address of feature template.

Although the pipeline is used, we will find that this structure can't support consecutive image data sending to Data_in Port for multiple sliding windows as when data flows to first RAM, the data flows from first RAM to second RAM is stopped. To improve the utilization efficiency of IP interface, the ping-pong operation is added as in fig.5

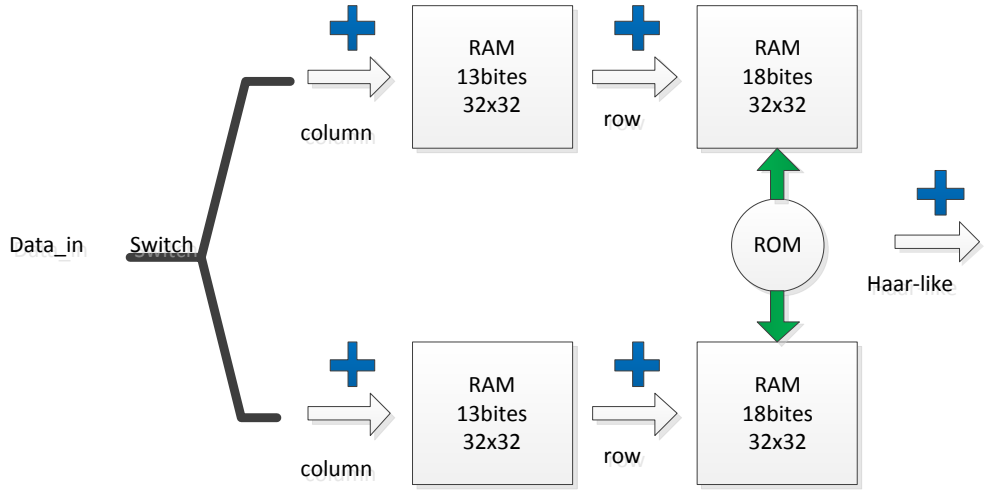

Fig.5 ping-pong operation

The structure in fig.6 consumes more storage resource and it also need a Multiplexer, but the utilization of IP interface is increased to 100%. For further analysis of the scanning process, we will find that the integral data in adjacent sliding windows are same in part. To improve the calculating speed, we present the structure of data reusing as in fig.6. The sliding window will move right after the data of integral image stored in B1 RAM which contains B11 and B12 has been used. If we need to use the original data in sliding window, the data in B11 can be updated only; when the sliding window should be updated next time, the new data will cover B12, so Bl1 and B12 will update in turn. The calculating speed of IP is twice as much as before, the needed RAM decreases to 59k bits, and the processing time of full image is twice as much as translating it.

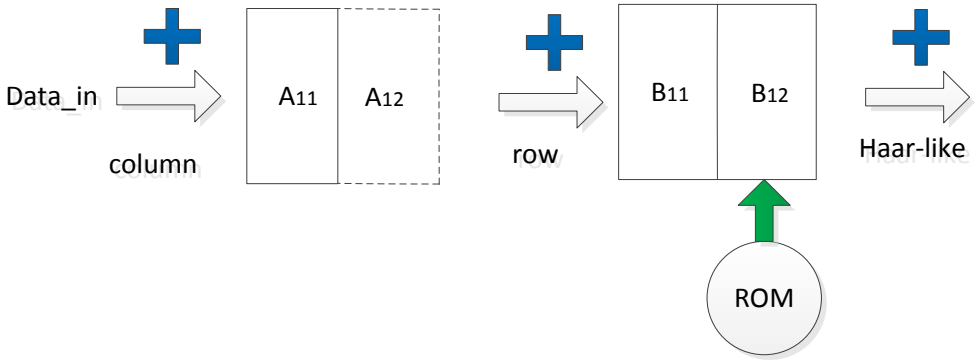

Fig.6 data reusing structure

Above is the basic IP design, we provide the enhanced IP where more Haar-like feature is needed and the structure of it is shown in fig.7A1-An are pipeline structure and “Switch” will send data to different input ports. The enhanced IP can generate more Haar-like feature once and the utilization of IP interface is 100%. The choice of n is based on the need number of Haar-like feature.

Experiment and analysis of result

We describe the specific digital circuit in HDL language for basic IP and the ports are Data_in, Data_out, and Haar_ready. Data_in is the input port which receives pixel data. Data_out is the output port which generates the value of Haar-like feature. When Haar-like features have been calculated , the haar-ready will be set to 1.

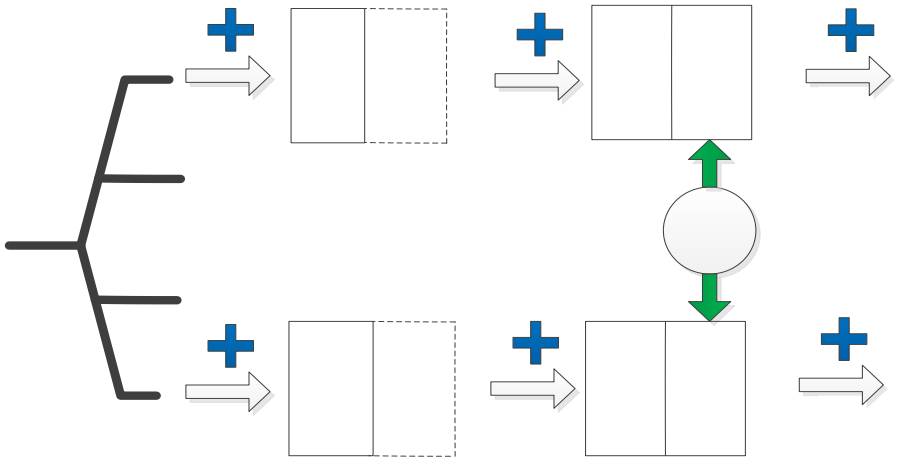

Fig.7 enhanced IP

We programed test bench for this IP with VCS of Synopsys, and one of the test case takes feature template in Fig.2. The coordinate of every point is: A(0 0), B(0 10), C(1 0), D(1 10), E(2 0), F(2 10), G(3 0), H(3 10), and the input pixel data is always 1, so the value of Haar-like feature should be 10. The waveform of simulation shows in fig.8, and we can find that the data of data_in is always 1; data sent to ROM at lut_in are 0x800, 0x80a, 0x820, 0x82a, 0x840, 0x84a, 0xc60, and 0xc6a; the data at data_out is 0x000a when the Haar_ready change to 1; the port of Haar_ready set every four clock cycles and it will continue. The first 2 bits of lut_in are flags and the last 10 bits are used to store addresses of templates. These are in line with our expectations, so the IP function can meet the demand of design.

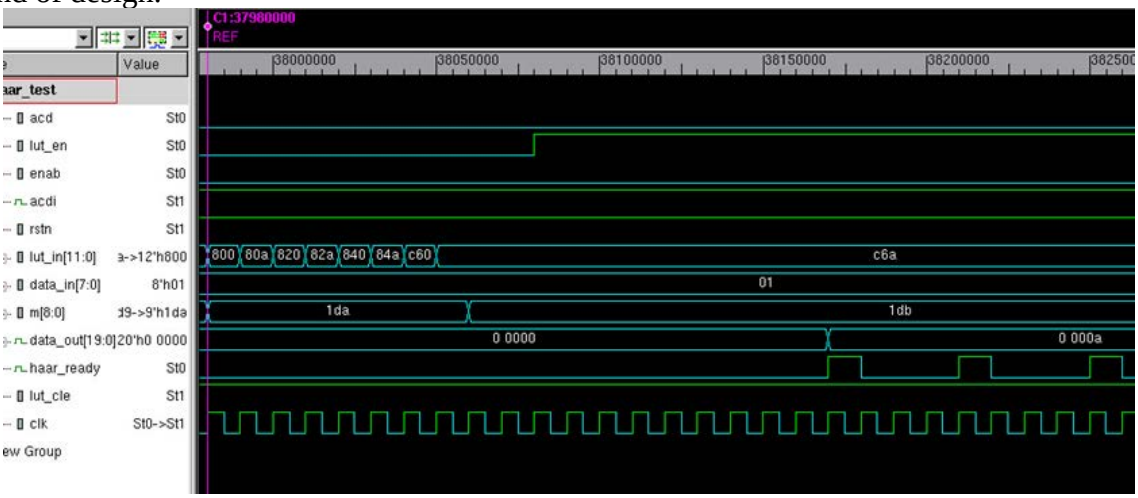

Fig.8 simulation of VCS

Conclusion

This paper presents an IP design of quickly extracting Haar-like features in the image of front vehicle. We improve the performance of the IP by selecting architecture of integral image, pipeline, ping-pong operations, data multiplexing etc. we verified the IP Design according to VCS, and the result shows that the processing time of whole image is twice as much as transporting time, and the storage space of integral image requires only 59kbits. The IP can be applied to FPGA and the design of ASIC, and it is helpful to accelerate the process of the practical application of Driver Assistance System.

Acknowledgement

This work is supported by the National Natural Science Foundation of China (Grant: 61376028), and the Key Laboratory of Advanced Display and System Application (Shanghai

University)(P200803).

References

[1] Sivaraman S, Trivedi M M. Looking at vehicles on the road: A survey of vision-based vehicle detection, tracking, and behavior analysis[J]. Intelligent Transportation Systems, IEEE Transactions on, 2013, 14(4): 1773-1795.

[2] Hilario C, Collado J M, Armingol J M, et al. Pyramidal image analysis for vehicle detection[C]//Intelligent Vehicles Symposium, 2005. Proceedings. IEEE. IEEE, 2005: 88-93.

[3] Nuevo J, Parra I, Sjöberg J, et al. Estimating surrounding vehicles' pose using computer vision[C]//Intelligent Transportation Systems (ITSC), 2010 13th International IEEE Conference on. IEEE, 2010: 1863-1868.

[4] Dalal N, Triggs B. Histograms of oriented gradients for human detection[C]//Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on. IEEE, 2005, 1: 886-893.

[5] Viola P, Jones M. Rapid object detection using a boosted cascade of simple features[C]//Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on. IEEE, 2001, 1: I-511-I-518 vol. 1.

[6] Beleznai C, Schreiber D, Rauter M. Pedestrian detection using GPU-accelerated multiple cue computation[C]//Computer Vision and Pattern Recognition Workshops (CVPRW), 2011 IEEE Computer Society Conference on. IEEE, 2011: 58-65.

[7] Jin J, Nguyen V D, Lee S J, et al. FPGA design and implementation of a real-time vehicle detection system[C]//Control, Automation and Systems (ICCAS), 2012 12th International Conference on. IEEE, 2012: 204-207.

[8] Lai H C, Savvides M, Chen T. Proposed FPGA hardware architecture for high frame rate (≫ 100 fps) face detection using feature cascade classifiers[C]//Biometrics: Theory, Applications, and Systems, 2007. BTAS 2007. First IEEE International Conference on. IEEE, 2007: 1-6.

[9] Hiromoto M, Sugano H, Miyamoto R. Partially parallel architecture for adaboost-based detection with haar-like features[J]. Circuits and Systems for Video Technology, IEEE Transactions on, 2009, 19(1): 41-52.

[10] Mita T, Kaneko T, Hori O. Joint haar-like features for face detection[C]//Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on. IEEE, 2005, 2: 1619-1626.

[11] Barreto J, Menezes P, Dias J. Human-robot interaction based on haar-like features and eigenfaces[C]//Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on. IEEE, 2004, 2: 1888-1893.