A FPGA-Based Real-Time Hardware Accelerator for Orientation Calculation Part in SIFT

Jingbang QIU, Ying LU, Tianci HUANG, and Takeshi IKENAGA
Graduate School of IPS, WASEDA Univ.
megisgem0630@ruri.waseda.jp

Abstract

SIFT is regarded as one of the most powerful feature point detection algorithms in the world. The Orientation Calculation Part, defining major orientation of feature points, enables selected image features to be invariant to rotation changes. In this paper, we propose an FPGA-implementable hardware accelerator for this part. By introducing LUT-Based Square Root Computation and Shifting-Based Orientation Calculation with use of dual-port DDR2 memory access, we achieve to reach real-time process speed, meanwhile keeping high accuracy. By experiment, our system proves to reach Max Clock Frequency of 130.0 MHz, processing up to around 256,000 feature points including memory operations. Compared with conventional work, hardware cost is remained at the same level. Accuracy is kept at 98.9% for over 40,000 feature points from 50 images. Our proposal is suitable for a real-time SIFT system.

1. Introduction

Feature Detection has been of great interest in computer vision filed in recent years. Good features detected from images can be applied onto various circumstances. SIFT, abbreviated for Scale Invariant Feature Transform, is proposed by David Lowe [1, 2]. A significant advantage of SIFT over other algorithms is that, feature points detected are invariant to image scaling changes and rotation changes, while at the same time being robust to changes in illumination, addition of noise, cluster scene, occlusion and minor changes in viewpoint.

In cost of achieving its robustness, time consumption of SIFT is relatively huge. Hardly any real-time system exists (VGA size). GPU-based system has been proposed. However, yet accelerated, this method majorly depends on the performance of the GPU chip and the PC environment, and the results vary greatly from computer to computer. Recently, researches have been focusing on specified hardware accelerators for the SIFT algorithm based on FPGA [3, 4, 5, 6, 7], introducing some successful SIFT hardware design examples. Although these systems are far from perfect, the results showed a promising view of hardware implementation of SIFT.

As stated in [4], SIFT can be structurally divided into 4 major parts, Gaussian Pyramids & DoG Pyramids Construction, Feature Point Detection, Orientation Calculation (OC), and Descriptor Creation (Fig. 1.).

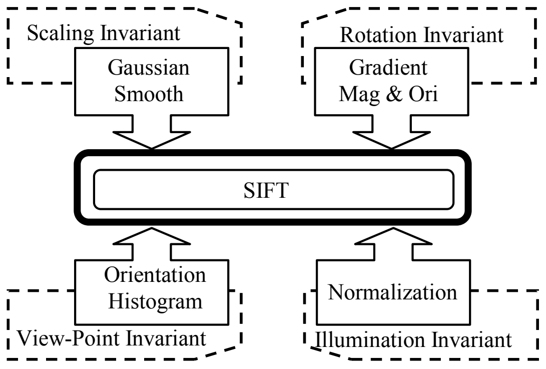

Figure 1. Components of SIFT

In this paper, we focus on hardware implementation of OC part.

1.1. Previous Work on OC

The OC Part is the third part of the SIFT algorithm. This part calculates the gradient magnitudes (GMs) and gradient orientations (GOs) of the local area of a feature point and does histogram to find out the major orientation of a feature point. Although this part consumes only 10% of the computational complexity, it enables the image features extracted to be invariant to rotation changes.

In [3], the author pre-computes gradient magnitudes and gradient orientations for all pixels in various scales and octaves. The results are then sent to a NIOS II processor for further processing. In another word, major orientation is not decided by specified hardware, but by an embedded processor. Also, a huge amount of memory is needed. Thus it's not suitable for a low power system.

In [7], a robotic system is implemented with SIFT algorithm with Stratix II FPGA board. It is claimed that the system could process one VGA image in 60ms. However, no structural information and no information on SIFT version is given.

This paper is arranged as follows. Our proposal for OC hardware accelerator will be given in SECTION II. SECTION III presents in detail experimental environment, hardware synthesis, and software simulation. SECTION IV briefly concludes our work.

2. Proposed Hardware Architecture

2.1. Overall System Structure

Our purpose is to develop a real-time processing hardware system with modest hardware cost, keeping relatively stable quality and high accuracy.

Two highlight features of our proposal are 1) LUT-based Square Root Computation (LUT-SRC), and 2) Shifting-based Orientation Calculation (SOC). Input of pixels of Gaussian Images is represented by 8 registers.

Our proposed hardware architecture is decided for the VGA image processing, but not confined to VGA size.

As few feature points have more than 2 major directions, we assume that every feature point would at the most have 2 major directions.

Overall system structure is shown is Fig. 2.

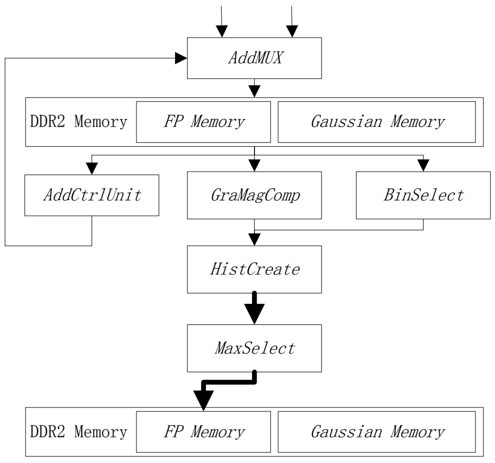

Figure 2. Overall system structure

2.2. LUT-Based Square Root Computation

This part is implemented in *GraMagComp* module.
GM in this process is computed by Formula 1.

$$\mathrm{GM}(x,y) = [dx^2+dy^2]^{1/2} \tag{1}$$

where $\mathrm{GM}(*)$ denotes Gradient Magnitude; dx denotes $L(x+1,y) - L(x-1,y)$; dy denotes $L(x,y+1) - L(x,y-1)$; $L(*)$ denotes pixel values of Gaussian Images.

Although square is easy to compute, square root computation is generally regarded as the hardware-consuming part. That's because, to generate the square root of a value, one needs to consider not only the integer part but also the numerical part, which concerns complex floating point calculations.

However, in this work, we use pixel input of 8 registers, which represent integers ranging from -127 to 128. Thus, square roots computed would be within the range from 0 to 128. Furthermore, as the results after square root computations are used for creating histograms, it is reasonable that we may ignore the numerical part which is just a small part under huge amount of data.

As a result of the above reasons, we propose to build up a LUT for only integer part of square root computation instead of designing complex hardware. As page is limited, detailed LUT construction is not given here, but can still be abstracted as in Formula 2.

By this method, altogether 128 thresholds are needed to define integer square roots from 0 to 128. By hardware optimization, fewer thresholds are needed.

$$S_{\mathrm{quare}}\, R_{\mathrm{oot}} = \begin{cases} 0, & x^2+y^2=0 \\ 1, & 1 \le x^2+y^2 \le 2 \\ & \vdots \\ 127, & 16000 \le x^2+y^2 \le 16256 \\ 128, & x^2+y^2 > 16256 \end{cases} \tag{2}$$

2.3. Shifting-Based Orientation Calculation

This part is implemented in *BinSelect* module.
GO in this process is computed by Formula 3.

$$\mathrm{GO}(x,y) = \tan^{-1}\left[\frac{L(x,y+1)-L(x,y-1)}{L(x+1,y)-L(x-1,y)}\right] \tag{3}$$

where $\mathrm{GO}(*)$ denotes Gradient Orientation; $L(*)$ denotes pixel values of Gaussian Images.

Like square root computation, arctan computation is a hardware-consuming component. Also, division in Formula 3 is not welcome in hardware design. As a result, we re-design this part based on multiply operations.

In respect with that computed GOs are then used to indicate to which bin a pixel belong, where altogether 36 bins exists, it is reasonable that we don't compute the exact values of GOs, but directly computes the exact bins that pixels belong to.

To define the bin, we may again use the LUT method. Although there are 36 bins within 360 degrees, dividing 10 degrees for each, the values of arctan function are anti-symmetric, as a result of which we may only consider values from 0 degree to 90 degrees. For other values from -180 degrees to 0 degree, and those from 90 degrees to 180 degrees, we may use plus/minus information of dx and dy to define.

To decide a pixel is in a certain bin within degrees from $T_{hd\,(l)}$ to $T_{hd\,(u)}$, we may use Formula 4.

$$\tan\frac{T_{hd\,(l)}}{180/\pi} \le \frac{dy}{dx} < \tan\frac{T_{hd\,(u)}}{180/\pi} \tag{4}$$

where both dx and dy are larger than 0; both $T_{hd\,(l)}$ and $T_{hd\,(u)}$ are within 0 to 90 degrees.

By multiplying dx to both sides we get,

$$dx \cdot \tan\frac{T_{hd\,(l)}}{180/\pi} \le dy < dx \cdot \tan\frac{T_{hd\,(u)}}{180/\pi} \tag{5}$$

As indicated in Formula 5, we only need to pre-decide the tangent value of the thresholds and then by multiplication and comparisons we can know the bin that a pixel belongs to.

Multiplying tangent values can be substituted by combinations of shifting different digits of the input dx, shown in table 1. Multiplying integer parts of Binary numbers can be implemented by right shifting dx; Multiplying numerical parts can be implemented by left shifting dx. Degree differences are within ∓ 0.03.

Table 1. Tangent values by binary numbers in *ThdCreator*

Degree	Tangent	Binary Tangent	Difference (degree)
10	0.17633	000.00101101	-0.03
20	0.36397	000.01011101	-0.03
30	0.57735	000.10010100	+0.03
40	0.83910	000.11010111	+0.03
50	1.19175	001.00110001	+0.01
60	1.73205	001.10111100	+0.03
70	2.74748	010.10111111	-0.01
80	5.67128	101.10101100	+0.00

Table 2. Bin definition by threshold signals in *AngleCmp*

a1	a2	a3	a4	a5	a6	a7	a8	Bin	Range
0	X	X	X	X	X	X	X	0	0~10
1	0	X	X	X	X	X	X	1	10~20
1	1	0	X	X	X	X	X	2	20~30
1	1	1	0	X	X	X	X	3	30~40
1	1	1	1	0	X	X	X	4	40~50
1	1	1	1	1	0	X	X	5	50~60
1	1	1	1	1	1	0	X	6	60~70
1	1	1	1	1	1	1	0	7	70~80
1	1	1	1	1	1	1	1	8	80~90

Defining threshold signal indicating dy larger than dx multiplying tangent value of 10 degrees as *a1*, we have similarly *a2* to *a8*. By table 2, we may define the bins.

Combining bin information obtained from table 2, we may use plus/minus information of dx and dy to fully define bins from -180 to 180 degrees.

A *BinSelect* unit in Fig.1 is constructed as in Fig. 3.

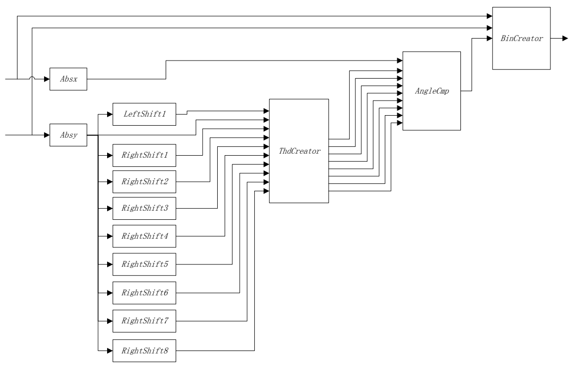

Figure 3. Construction of a *BinSelect* unit

2.4. Dual-Port DDR2 Memory operations

Dual-port DDR2 memory is embedded on Virtex®-V FPGA board. Two independent ports of 36-bit read/write width are provided. Read or Write memory operations can be finished in one clock, and the two ports can independently be Read or Write. This provides a variety of memory operations. In one clock, at the most 72 bits can be loaded or written; or 36-bit Read can be done in one port, while 36-bit Write done in the other port. In our implementation, we assign dual-port DDR2 memory operations as in Table 3.

Table 3. Dual-Port DDR2 Memory Port Assignment

Stage	Port A	Port B	Contents	Clocks
Initiation	36-bit Read	36-bit Read	4 pixels and position	1
Loading	36-bit Read	No use	4 pixels	1
Writing (worst case)	36-bit Write	No use	Pixel value, position, and orientation	1

As a result, to compute gradient magnitude and orientation for one pixel, it takes altogether 3 clocks. Worst case is defined when every feature point has 2 major directions.

3. Experimental Results

Hardware Synthesis is done with Xilinx ISE WebPACK 10.1 with Virtex®-5 FPGA board. Software Simulation is done with Microsoft® Visual Studio® 2008 on a PC of Intel® Core™ 2 CPU 6700 @ 2.66GHz 2.67GHz, 2.00GB RAM. Hardware synthesis detail and comparisons are shown in table 4. Evaluation Dataset of 50 images are used, including static objects, surveillance camera shots, landscapes, office shots and humans.

From table 4, we may find out that although hardware cost is a bit increased, the process speed is greatly improved to 250 fps, giving much space for system overhead. More hardware consumption is brought by extra hardware for deciding bins, creating histogram and max selection.

Accuracy compared with original software implementation [8] is given in table 5 and Fig. 4. Averaged accuracy for 50 examined images is 98.9%.

Table 4. Hardware synthesis detail and comparisons. Assuming averagely every VGA image has 1000 feature points

Item	Proposed	Convention[3]	Convention[7]
Image Size	640x480	320x240	640x480
#FP/Image	~1000	~150	~1000
Max Clock Frequency	130.0 MHz	184 MHz	---
Process Speed	256 fps	30 fps	15 fps
Slice Registers	1220	670	---
Slice LUTs	2548	1863	---

Table 5. Accuracy comparisons with original software implementation. TP denotes correct major orientation detected; FP denotes false major orientation which does not exist in original software implementation; FN denotes hardware solution does not extract a major orientation where there should be one

Image	#FP	TP	FP	FN
Builing06	588	576	10	12
Man01	586	580	13	6
Office05	550	542	12	8
Land05	452	445	15	7

4. Conclusion

In this paper, we proposed a real-time hardware accelerator for Orientation Calculation part in SIFT algorithm. Altogether 1220 Slice Registers and 2548 Slice LUTs are used, which is under affordable hardware range of a Virtex®-V FPGA board.

Based on dual-port DDR2 memory embedded with Virtex®-V FPGA, we propose LUT-Based Square Root Computation and Shifting-Based Orientation Calculation. By introducing the two schemes, our proposed system reaches Max Clock Frequency of 130.0 MHz, with worst-case process speed for up to 256 fps (around 256,000 feature points), which is faster than real-time. Averaged accuracy for 50 examined images is 98.9%. Our system proves to be a suitable architecture for real-time SIFT system.

5. Acknowledgement

This work was supported by fund from MEXT via Kitakyushu innovative cluster projects and CREST.

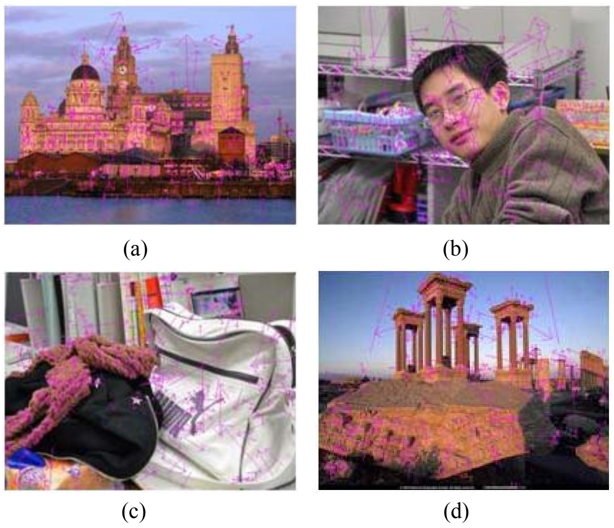

Figure 4. Feature points extracted by proposed hardware implementation. (a) Building06; (b) Man01; (c) Office05; (d) Land05

6. References

[1] David G. Lowe, "Object recognition from local scale-invariant features" International Conference on Computer Vision, Corfu, Greece, pp. 1150-1157, Sep. 1999

[2] David G. Lowe, "Distinctive image features from scale-invariant keypoints" International Journal of Computer Vision, 60, 2, pp. 91-110, 2004

[3] Vanderlei Bonato, Eduardo Marques, and George A., "A Parallel Hardware Architecture for Scale and Rotation Invariant Feature Detection" IEEE Transaction on Circuits and Systems for Video Technology, Vol. 18, No. 12, pp. 1703 – 1712, Dec. 2008

[4] Jingbang QIU, Tianci HUANG, Takeshi IKENAGA, "Hardware Accelerator for Feature Point Detection Part in SIFT Algorithm & Corresponding Hardware-Friendly Modification" Workshop on Synthesis And System Integration of Mixed Information Technologies, Okinawa, Japan, pp. 213 – 218, Mar. 2009

[5] Jingbang QIU, Tianci HUANG, Yiqing HUANG, Takeshi IKENAGA, "A Hardware Accelerator with Variable Pixel Representation & Skip Mode Prediction for Feature Point Detection Part of SIFT Algorithm" IAPR Conference on Machine Vision Applications, Tokyo, Japan, May. 2009

[6] Jingbang QIU, Tianci HUANG, Takeshi IKENAGA, "1D-based 2D Gaussian Convolution Unit Based Hardware Accelerator for Gaussian & DoG Pyramid Construction in SIFT" The Institute of Electronics, Information and Communication Engineers general conference, Matsuyama, Japan, Mar. 2009

[7] Stephen Se, ho-Kong Ng, Pitor Jasiobedzki, Tai-Jing Moyung, "VISION BASED MODELING AND LOCALIZATION FOR PLANETARY EXPLORATION ROVERS" 55th International Astronautical Congress 2004, Vancouver, Canada, pp. 11-11 Oct. 2004

[8] Rob Hess – School of EECS @ Oregon State University http://web.engr.oregonstate.edu/~hess/