Fast and Low-Memory-Bandwidth Architecture of SIFT Descriptor Generation with Scalability on Speed and Accuracy for VGA Video

Kosuke Mizuno, Hiroki Noguchi, Guangji He, Yosuke Terachi, Tetsuya Kamino,
Hiroshi Kawaguchi and Masahiko Yoshimoto
Kobe University,
1-1 Rokkodai-Cho, Nada-Ku, Kobe, Hyogo 657-8501, Japan
Phone: +81-78-803-6234, Fax: +81-78-803-6234, E-mail: mi-no@cs28.cs.kobe-u.ac.jp

Abstract— **This paper describes an FPGA implementation which features a hardware-oriented Scale Invariant Feature Transform (SIFT) algorithm, a scalable architecture with high-speed mode and high-accuracy mode, and highly parallel datapath modules. The proposed FPGA implementation can generate a SIFT descriptor vector with 50 MHz for VGA resolution video (640 × 480 pixels) at 56 frames per second (fps). Our proposed implementation made the operating frequency and memory bandwidth a half or less than that of the conventional FPGA implementation and as a result, we achieved a system that can provide low power consumption.**

Keywords— low power, SIFT, image recognition, VGA

I. INTRODUCTION

In recent years, image recognition systems have been developed through evolution of circuit technologies and algorithms for image processing. A major algorithm used in image recognition systems, Scale Invariant Feature Transform (SIFT) [1], is invariant to changes of scale, rotation, and illumination. Moreover, several algorithms, PCA-SIFT [4], viewpoint invariant patches [5], Simultaneous Localization and Mapping (SLAM) [6], and SIFT flow [7], are based on the SIFT algorithm. Because of its consistency and expandability, SIFT has high adaptability to object recognition, 3D image matching, object tracking, mobile robot applications, and so on (Figure 1). However, each application requires a SIFT processor to provide different performance such as low power, high accuracy, and high speed. Therefore, the SIFT processor must have high scalability.

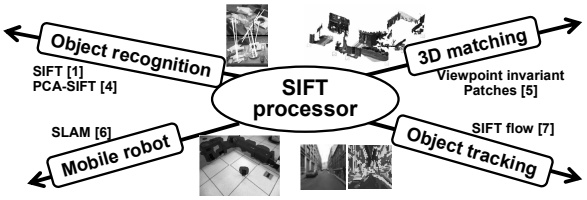

Figure 1. Various applications and algorithms based on SIFT.

Almost all SIFT applications have a common problem of power consumption because of the heavy workload required for SIFT descriptor generation. Some processors implementing SIFT have been developed [2]-[3]. V. Bonato et al. proposed the parallel feature detection architecture for QVGA resolution video (320 × 240 pixels) at 30 fps [2]. However, the descriptor vector generation part of the processor is implemented using software and does not perform in real time. For that reason, the architecture is not applicable for real-time applications using many feature points. L. Yao et al. proposed the architecture for VGA resolution video (640 × 480 pixels) at 32 fps [3]. The architecture provides high-performance operation. However, the architecture implements the optimized SIFT algorithm for image matching and consumes many hardware resources. Previous works are especially unsuitable for mobile applications under limited battery capacity because of their high operating frequency and high demand for hardware resources. The proposed processor resolves the problem of power consumption and obtains higher scalability than the previously reported processors.

II. HARDWARE-ORIENTED SIFT ALGORITHM

A. Algorithm Overview

Figure 2 portrays the workload ratio of the SIFT descriptor generation by software implementation for the original SIFT algorithm [8]. Figure 2 shows that Gaussian filtering dominates 90% of the whole process. For low-power and real-time SIFT descriptor generation, minimizing the computation of the Gaussian filtering is effective.

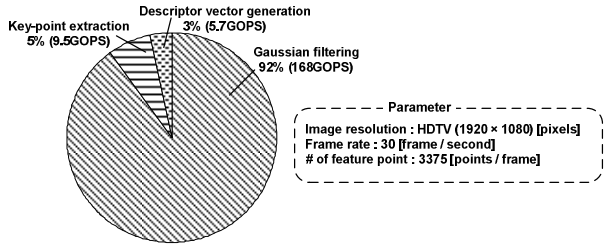

Figure 2. Workload analysis of SIFT descriptor generation.

Figure 3 shows a flow diagram of the hardware-oriented SIFT algorithm. The SIFT descriptor generation consists of Gaussian filtering, key-point extraction and descriptor vector generation. The input image is smoothed in Gaussian filtering for key-point extraction. Adjacent Gaussian filtered images are subtracted for difference-of-Gaussian (DOG) generation. Key-point is detected by searching on DOG. Finally, SIFT descriptor vectors are outputted by calculating a gradient histogram of luminance around the key-point.

1946-1488/10 $26.00 © 2010 IEEE
DOI 10.1109/FPL.2010.119

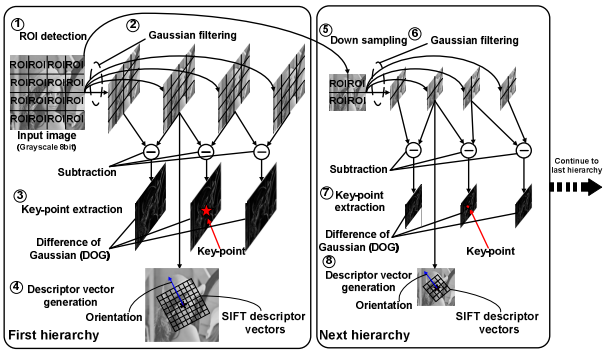

Figure 3. Hardware-oriented SIFT descriptor generation.

The original scheme handles a whole image all at once. It can provide high-accuracy results, but it increases the use of hardware resources. In contrast, the proposed method processes an image divided into regions of interest (ROIs). The introduction of ROIs provides a structure that is appropriate for clock gating and operating frequency control with little accuracy degradation. Additionally, it reduces the capacity in an on-chip memory. In an original algorithm, a Gaussian filtered image in any hierarchy is generated from an input image or Gaussian filtered image in the previous hierarchy. This method raises the accuracy of key-point extraction, but it increases latency. The proposed scheme generates a 2D Gaussian filtered image solely from an input image. It can accelerate the operation of Gaussian pyramid generation with the shortcoming of slight accuracy degradation.

B. ROI

Figure 4 depicts a conceptual diagram introducing ROI into the original SIFT algorithm: (a) is the original algorithm, and (b) is a ROI-based algorithm dividing input image into four ROIs. The ROI-based algorithm drastically reduces the capacity for working memory with low accuracy degradation. In the proposed algorithm, ROI sizes are 40 × 30 in high-speed mode and 80 × 60 in high-accuracy mode. In the case of VGA resolution, the proposed algorithm requires the working memory for 80 × 60 pixels, while original SIFT does 640 × 480 pixels. Consequently, the proposed method reduces working memory by 98% compared with the original scheme.

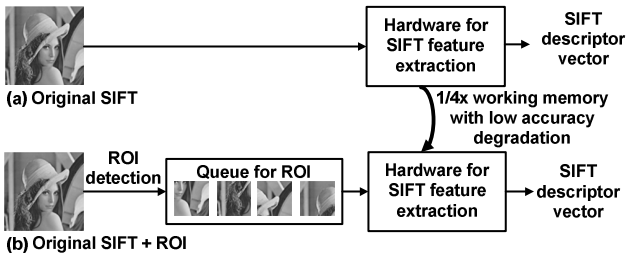

Figure 4. Conceptual diagram of introducing ROI into original SIFT.

C. Two-Dimensional Gaussian Filtering

In this subsection, we describe two-dimensional Gaussian filtering for a hardware implementation. Figure 5 shows two-dimensional Gaussian filtering at a pixel level, where *Y* is the input image, *C* is the coefficient of Gaussian function, and *G* is the result of one Gaussian filtering. A result of two-dimensional Gaussian filtering is the sum of a result by multiplication of one pixel *Y* and one coefficient *C*. The reuse of an intermediate result is difficult using this scheme because the same combination of one pixel and one coefficient do not exist in adjacent Gaussian filtering. Figure 6 depicts the combination of two one-dimensional Gaussian filterings where *CV* is the vertical coefficient, *CH* is the horizontal coefficient of Gaussian function, and *G'* is the intermediate result of one-dimensional Gaussian filtering. This scheme can drastically reduce the number of memory accesses and output the same result as two-dimensional Gaussian filtering because this method provides the reuse of an intermediate result *G'*.

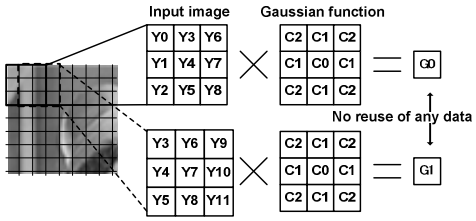

Figure 5. Disadvantage of using two-dimensional Gaussian filtering.

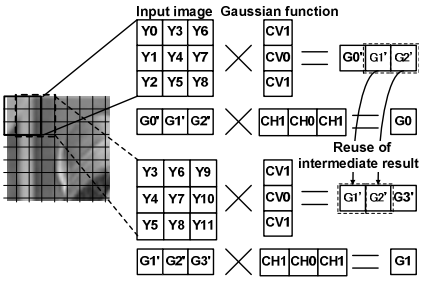

Figure 6. Advantage of using one-dimensional Gaussian filtering.

D. Gaussian Pyramid Generation

In the original algorithm, a Gaussian filtered image in any hierarchy is computed recursively from an input image. The method raises the key-point extraction accuracy, but it causes an increase of latency disturbing high-speed operation. Therefore, some changes are necessary for a real-time application. The proposed scheme generates any Gaussian filtered image solely from an input image. The method has scale parallelism and hierarchical parallelism. Therefore, the method can accelerate the operation of Gaussian pyramid generation with the shortcoming of slight accuracy degradation.

E. Simulation Results

This subsection describes effects of the proposed algorithm in memory size, the number of memory accesses, and performance. The proposed method reduces the size of working memory by 98% compared with the original scheme. The proposed Gaussian filtering reduces the number of memory accesses by 93%, compared with the general two-dimensional Gaussian filtering. We conducted a simulation for accuracy degradation using software for object recognition to confirm the performance of the proposed algorithm in high-

accuracy mode. The respective numbers of objects and test images are 79 and 100. Results of the experiment show that the recognition rate degradation is 2.1% (Figure 7). The quality loss is allowable for general-purpose video applications. High-speed mode degrades the feature-point accuracy to 29% of that in high-accuracy mode.

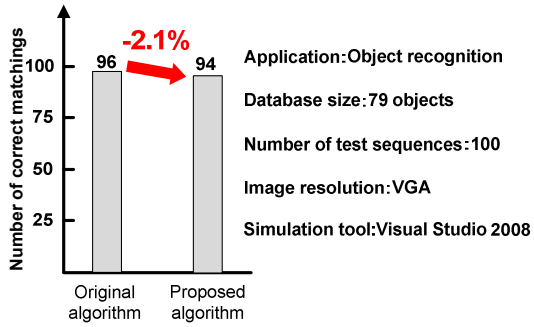

Figure 7. Accuracy degradation by the proposed algorithm.

III. ARCHITECTURE

A. Architecture Overview

Figure 8 presents the whole diagram of the proposed architecture. The architecture comprises a main SIFT vector generation module and controllers for external devices such as an SDRAM, a CPU, and a camera. The proposed architecture has two operation modes: high-speed mode and high-accuracy mode. The former mode supports high-speed operation through a pipeline using small ROI. Consequently, the mode is suitable for real-time application. The latter mode provides more accurate operation using a large ROI. The sections below describe the two operation modes in greater detail.

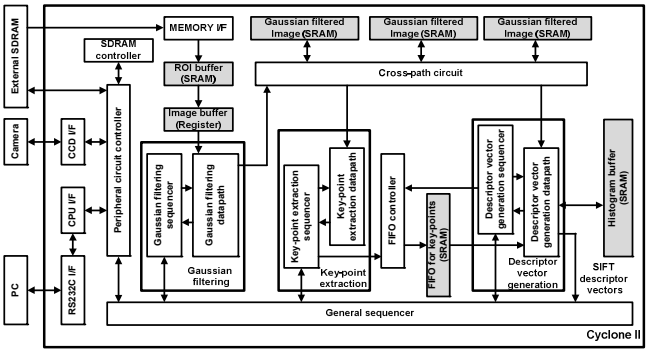

Figure 8. Proposed FPGA architecture.

B. Architecture in High-Speed Mode

In the high-speed mode, the architecture performs pipelining using an ROI of 40 × 30 pixels. Figure 9 presents the pipeline flow in the high-speed mode. The pipeline is divided into four stages: ROI reading, Gaussian filtering, key-point extraction, and descriptor vector generation. The high-speed mode provides SIFT descriptor vector generation at 56 fps.

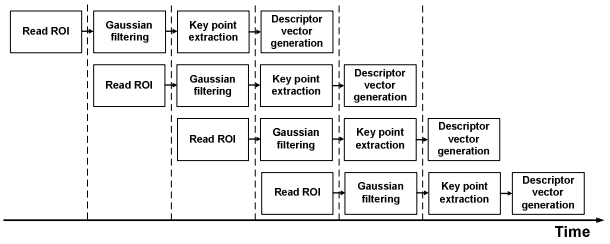

Figure 9. Pipeline flow in high-speed mode.

C. Architecture in High-Accuracy Mode

In the high-accuracy mode, the architecture performs a sequential operation using ROI of 80 × 60 pixels. Figure 10 presents the operation flow in the high-accuracy mode. In the operation, ROI reading and descriptor vector generation are overlapped for efficient processing. The high-accuracy mode enables SIFT descriptor vector generation at 32 fps. In addition to the real-time operation, the mode provides a more precise key-point than the high-speed mode.

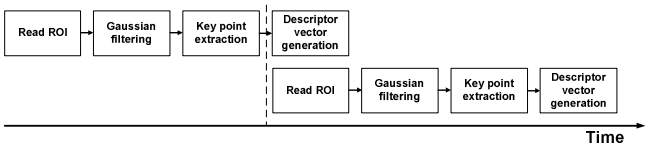

Figure 10. Operation flow in high-accuracy mode.

D. Elemental Architecture in the Datapath Modules

This subsection introduces important modules for Gaussian filtering and key-point extraction.

First, a Gaussian filtering module is described. Gaussian filtering accounts for most of the workload in the SIFT algorithm. Therefore, this part is the most important circuit for a low-power and high-speed system. We implemented a 60-parallel systolic-array (SA) architecture to resolve the workload problem. Figure 11 shows the block diagram of Gaussian filtering module comprising 60 SAs. In fact, an SA has ring-connected registers, two multiplier accumulators (MAC) and one multiplexer. Each SA reads one pixel per cycle and operates with a pipeline of pixel level. Results show that the Gaussian filtering circuit operates with 50 MHz and VGA resolution at 62 fps.

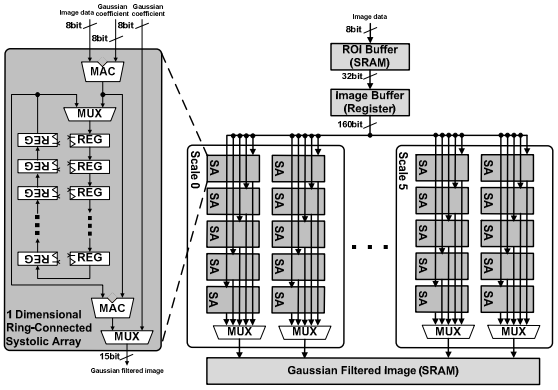

Figure 11. Block diagram of the Gaussian Filtering Datapath.

Next, a key-point extraction module is described. The key-point extraction has the second heaviest workload in the SIFT algorithm, as shown in Figure 2. Therefore, the key-point extraction module requires parallelization for real-time SIFT descriptor generation. The key-point extraction datapath consists of three key-point extraction cores. The key-point extraction requires 27 pixels in three adjacent difference of Gaussian (DoG in Figure 3). For that reason, the key-point extraction core comprises 27 processing elements (PE), as shown in Figure 12. The input data for PE flow from top to bottom. Each PE reuses a pixel received from the upper PE. The reuse of pixels reduces the number of memory accesses to 34% of the former level.

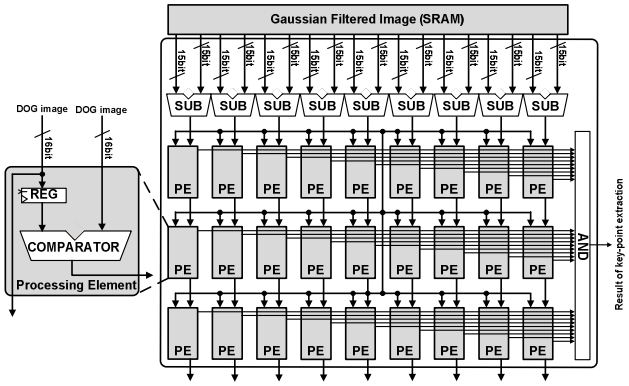

Figure 12. Block diagram of key-point extraction core.

E. Performance Evaluation

This subsection describes effects by the proposed architecture. The cycle times of Gaussian filtering module, key-point extraction module are reduced by 98.4% and 98%, respectively, compared with the non-parallelized architecture. The number of memory accesses of the Gaussian filtering module and the key-point extraction module are reduced by 88% and 66% respectively, compared with the results obtained using the non-parallelized architecture. Pixel reuse by the systolic-array architectures in the Gaussian filtering datapath and the key-point extraction datapath contributes to the memory access reduction.

IV. FPGA IMPLEMENTATION

We designed the proposed architecture of SIFT descriptor generation implementing a hardware-oriented algorithm and a scalable architecture with a high-speed mode and high-accuracy mode. The development board (DE2-70; Altera Corp.) was used for the implementation. Figure 13 shows a comparison to conventional FPGA implementation. The SRAM capacity and the number of LUTs are reduced by about 75% and 9% of that described in prior report [3] respectively. But the number of registers and DSP blocks are increased by 19% and 165%. That is because our proposed architecture adopts highly parallel circuits taking advantage of pixel reuse for acceleration. The capacity for working memory of the two proposed scheme, high-accuracy mode, and high-speed mode are reduced by 73% and 79% compared with those of the conventional architecture [3]. The proposed architecture in high-accuracy mode provides a SIFT descriptor vector with 50 MHz for VGA resolution video (640 × 480 pixels) at 32 fps, and the proposed architecture in high-speed mode does with 28.5 MHz. In addition to real-time operation, the required frequencies of the two proposed schemes—high-accuracy mode and high-speed mode—are reduced by 50% and 71.5% compared with the conventional architecture.

	[2]	[3]	Proposed architecture	
			High-accuracy mode	High-speed mode
FPGA	Stratix II	Virtex-5	Cyclone II	Cyclone II
# of LUTs	43,366	35,889	32,592	32,592
# of registers	19,100	19,529	23,247	23,247
# of DSP blocks	64	97	258	258
Working memory (Mbits)	1.35	3.24	0.87	0.67
Resolution	QVGA (320x480)	VGA (640x480)	VGA (640x480)	VGA (640x480)
Frame rate (fps)	30	32	32	56
Operating frequency	50	100	50	50

Figure 13. Comparison to conventional FPGA implementation.

V. CONCLUSION

We proposed a novel FPGA implementation of the SIFT descriptor generation. This implementation has a hardware-oriented algorithm, scalable architecture with high-speed mode and high-accuracy mode, and parallel datapath modules for Gaussian filtering and key-point extraction. Highly parallel circuits in the two modules accelerate the SIFT descriptor generation. Pixel reuse by the systolic-array architectures in the two modules contributes to the low memory bandwidth. Evaluation of the proposed design demonstrates 50% required-frequency reduction of the real-time SIFT descriptor generation compared with that of the conventional processor [3]. Results show that this proposed architecture enables low-power SIFT descriptor generation. In addition to power savings, the proposed architecture provides scalability on speed, and accuracy for several applications.

ACKNOWLEDGMENTS

This work was supported by KAKENHI(18200003).

REFERENCES

[1] D. G. Lowe, “Distinctive image features from scale invariant keypoints,” International Journal of Computer Vision, vol. 60, no. 2, pp.91-110, 2004.

[2] V. Bonato, E. Marques, and G. A. Constantinides, "A Parallel Hardware Architecture for Scale and Rotation Invariant Feature Detection," IEEE Trans. Circuits Syst., vol. 18, no. 12, pp. 1703-1712, Dec. 2008.

[3] L. Yao, H. Feng, Y. Zhu, Z. Jiang, D. Zhao, and W. Feng, "An architecture of optimised SIFT feature detection for an FPGA implementation of an image matcher," ICFPT 2009.

[4] Y. Ke and R. Sukthankar, “PCA-SIFT: A More Distinctive Representation for Local Image Descriptors,” CVPR 2004.

[5] C. Wu, B. Clipp, X. Li, J.-M. Frahm and M. Pollefeys, “3D model matching with Viewpoint-Invariant Patches (VIP),” CVPR 2008.

[6] S. Se, D. Lowe, and J. Little, "Mobile Robot Localization and Mapping with Uncertainty using Scale-Invariant Visual Landmarks," The International Journal of Robotics Research vol. 21, no. 8, August 2002, pp. 735-758.

[7] C. Liu, J. Yuen, A. Torralba, J. Sivic, and W. T. Freeman, “SIFT Flow: Dense Correspondence across Different Scenes,” ECCV 2008.

[8] R. Hess, “SIFT feature detector (source code),” 2007. Available: http://web.engr.oregonstate.edu/~hess/