2012 International Conference on Image, Vision and Computing (ICIVC 2012)
IPCSIT vol. 50 (2012) © (2012) IACSIT Press, Singapore
DOI: 10.7763/IPCSIT.2012.V50.3

High Speed Object Recognition Based on SIFT Algorithm

Hossein Borhanifar [1] [+] and Vahid Mirza Naeim [2]

[1] Electrical and Computer Department, Islamic Azad University, NazarAbad Center, Iran
[2] Electrical and Computer Department, Zanjan University, Iran

Abstract. In the paper, a parallel hardware architecture based on scale invariant feature transform (SIFT) is presented to recognize image features. The hardware is independent completely and doesn't need another system like a computer. In fact, the system read image data directly and gives the results. We assume that the input image is 320×240 pixels and 30 frames per second. High density and speed of FPGA chips make that we can implement accuracy requirements of the design. This work is an operational implementation of the reconfigurable architecture. It is targeted to an Altera Cyclone III FPGA to take advantages of computational specialization and parallelism. In addition to logic cells in FPGAs Some tools such as shift registers, memories, multipliers and PLLs make our design achieve real time image processing.

Keywords: filter, image, algorithm, SIFT, FPGA.

1. Introduction

Digital image processing is an ever expanding and dynamic area with applications reaching out into our everyday life such as medicine, space exploration, surveillance, authentication, automated industry inspection and many more areas.

For any object in an image, interesting points on the object can be extracted to provide a "feature description" of the object. This description, extracted from a training image, can then be used to identify the object when attempting to locate the object in a test image containing many other objects. To perform reliable recognition, it is important that the features extracted from the training contrast regions of the image, such as object edges. Recognizing an image by SIFT algorithm is invariant to uniform scaling, orientation, and partially invariant to affine distortion and illumination changes.

Field Programmable Gate Arrays are reconfigurable devices. Hardware design techniques such as parallelism and pipelining can be developed on a FPGA, which is not possible in dedicated DSP designs. Implementing image processing algorithms on reconfigurable hardware minimizes the time-to-market cost, enables rapid prototyping of complex algorithms and simplifies debugging and verification. Therefore, FPGA is an ideal choice for implementation of real time image processing algorithms [4].

2. SIFT Algorithm

The SIFT algorithm was initially presented in a paper by David Lowe in 1999 (Lowe 1999) and then he summarized his algorithm in 2004(Lowe 2004). Lowe's method for image feature generation transforms an image into a large collection of feature vectors, each of which is invariant to image translation, scaling, and rotation, partially invariant to illumination changes and robust to local geometric distortion. Key locations are defined as maxima and minima of the result of difference of Gaussians function applied in scale-space to a series of smoothed and resampled images. Low contrast candidate points and edge response points along an edge are discarded. Dominant orientations are assigned to localized keypoints. These steps ensure that the

[+] Corresponding author. Tel.: + 98 2166013308; fax: +98 2188672290.
E-mail address: hborhanifar@yahoo.com.

keypoints are more stable for matching and recognition. SIFT descriptors robust to local affine distortion are then obtained by considering pixels around a radius of the key location, blurring and resampling of local image orientation planes [1].

3. Hardware Implementation

The FPGA implementation of the SIFT algorithm for image recognition is partitioned into four different stages as shown in Figure 1. This design has been implemented completely within a single Altera Cyclone III and all processing is calculated in different stages.

Data of an image is entered in the circuit and in the first block, Gaussian and differential of Gaussian images are computed in 3 octaves and 6 intervals. The results of DoG block are prepared for two next blocks. OriMag calculates orientation and gradient magnitude for every pixel. Finding and checking the stability of keypoints are done in KP block. Finally, the descriptor block computes a 128 bytes definition for every stable keypoint. For synchronizing the circuit with other systems there is a FIFO memory in output to save the results of the plan.

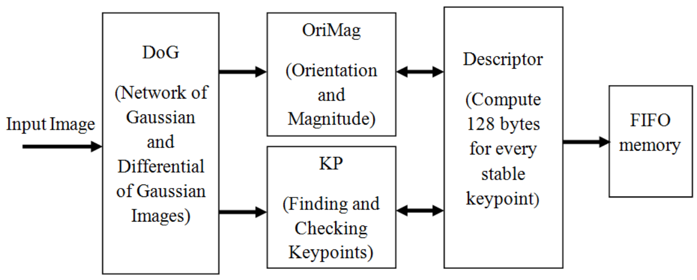

Input Image
DoG
(Network of Gaussian and Differential of Gaussian Images)
OriMag
(Orientation and Magnitude)
KP
(Finding and Checking Keypoints)
Descriptor
(Compute 128 bytes for every stable keypoint)
FIFO memory

Fig. 1: The diagram of Hardware Implementation

3.1. Differential of gaussian images

The Gaussian filter is implemented considering that its kernel is separable, since this approach is computationally more efficient than the traditional one, which directly convolves a 2 D kernel with the image. For a 7 × 7 kernel, to achieve the minimal required parallelism, we propose an optimized hardware module, as presented in Figure 2. The incoming pixels are shifted through the line buffers that create a delay line. The buffer depth depends on the number of pixels in each line of the frame that in the design we assume a filter with 7 × 7 dimension. These delay lines feed the filter array simultaneously with pixels from all the relevant video lines. The 2-D filter is separated into two 1-D filters implemented first in the horizontal direction, followed by the vertical direction as shown in Figure 2. This technique can be applied to both the smoothing stage with the Gaussian filter, and the identification of the gradient with the derivative of the Gaussian.

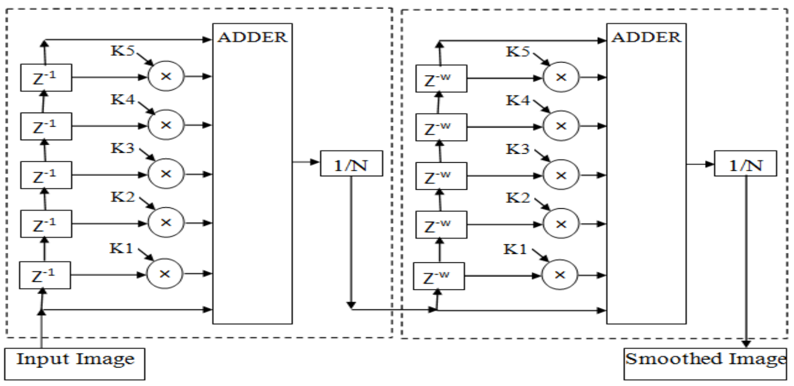


Fig. 2: Gaussian filter architecture

For row convolution, after getting 7 pixels, the first result can be calculated by a 6-bit shift register. While for column convolution, the distance between two pixels is equal as much as column number. W parameter is the number of column and Z^{-W} is a shift register with W bytes. This module takes the advantage of the Gaussian kernel symmetry and save two by assuming that the kernel vector has always the values one or zero at position 1 and 7 and consequently avoid the multiplications at those points. It is possible to keep this pattern for kernels generated with any values by simply multiplying the kernel coefficients by an appropriate constant. Since this multiplication only alters the kernel values, the filtered pixel is normalized simply by dividing its value by the new kernel sum (N).

According to figure 3 that is repeated for every octave it is obvious that the output of every Gaussian filter is the input of the next one. In the output of every smoothed image there is a buffer which has 3W length to synchronize two sequence smoothed image. Differential of Gaussian images are calculated and after a delay by a buffer is an input for the next block.

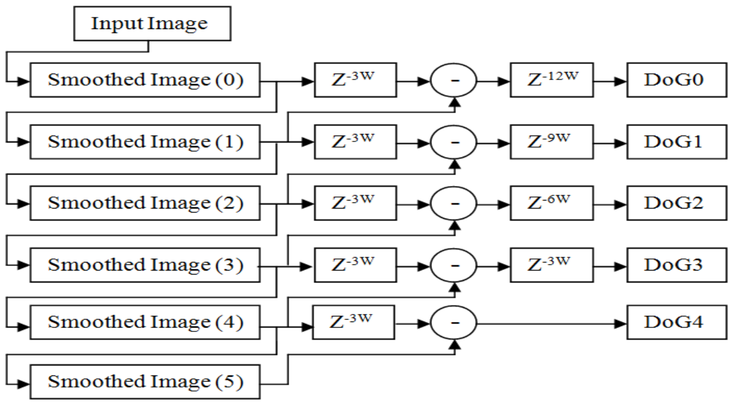


Fig. 3: Hardware implementation of DoG

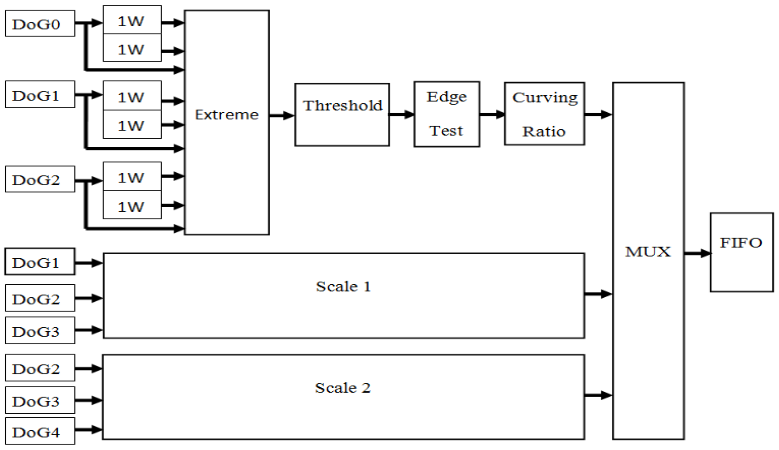


Fig. 4: The diagram of finding and checking keypoints

3.2. Finding and checking keypoints

The proposed hardware for this block is the same as Figure 4. At first, input data from DoGs is shifted in registers which has 2W length and there is an output per 1W. Therefore 3 inputs from every DoG and in total 9 inputs receive to extreme block to detect maximum value of the candidate pixel depending on its 26 neighbourhood pixels. In the next stages we check the stability of the keypoint by testing threshold, edge location and curving ratio. If the candid pixel passes all conditions, it will be selected as a stable keypoint. The keypoint is written as a 24-bits stream in a FIFO memory which includes location, scale and the value of keypoints.

3.3. Orientation and magnitude

We implement this block by using of Coordinate Rotation Digital Computer (CORDIC) algorithm. For calculating the algorithm, fixed-point format is used to reduce the amount of operations in compared with floating-point. Since the maximum internal value is 584, which is given by the maximum value from multiplied by the CORDIC gain 1.618, and with 8 bits for the fraction part, the final results are presented in 10 bits for magnitude and 6 bits for orientation. Data is received for CORDIC circuit from the smoothed image, and then magnitude (m) and gradient (θ) are computed. We use a DPRAM to increase the speed of processing. In this case, OriMag block doesn't depend on KP block, and it performs its duty separately.

3.4. Descriptor

This block can be implemented in hardware or in software by a processor. While there is complex algorithm in the part, software method is easier than hardware method. But for achieving to high speed and real time processing, it is better that we use hardware.

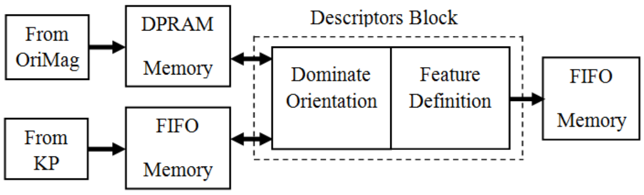


Fig. 5: The diagram description block and connection with other blocks

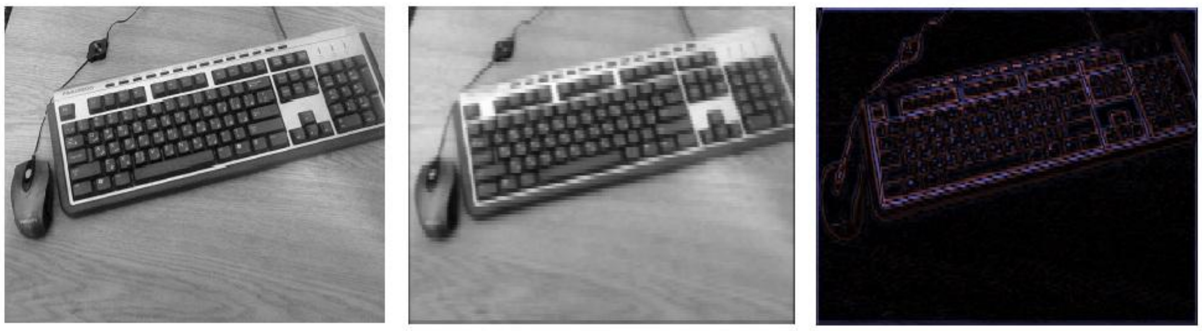
Figure 5 shows a diagram of the block. Keypoints are read from the FIFO memory and other information such as magnitude, orientation and its neighbourhood pixels. In the first section, a histogram which includes 36 bins is made for the keypoint. The histogram is based on the magnitude and orientation of the neighbourhood of the keypoint. Some peaks are found in the histogram and the highest one is selected as a feature. Then a description, which is discussed in section 2.4, is defined in 128 bytes and is pushed in the FIFO to use other systems. Therefore all process that we need to implement the SIFT algorithm has done and features of the image are obtained.

4. Result

Quartus II software is used for compiling of the design; and we use EP3C120F484C7 FPGA of Cyclone III series which is proper for this application. The results of compile are shown in table 1. Figure 6(a) is the main image and Figures 6(b) and 6(c) are examples of smoothed and DoG Images. And figure 7 shows some keypoints on the image that descriptors are defined for the keypoints.

Table. 1: The result of synthesis

Blocks	Logic Cell Combinational	Logic Cell Registers	Memory Bits	DSP 18x18
DoG	675	705	809364	0
KP	1029	1005	50436	0
OriMag	1041	720	1612800	0
Descriptor	40818	12300	344064	45
Total	43563	14730	2816664	45



(a) (b) (c)

Fig. 6: (a) Main Image (b) Sample Smoothed Image (c) Sample DoG

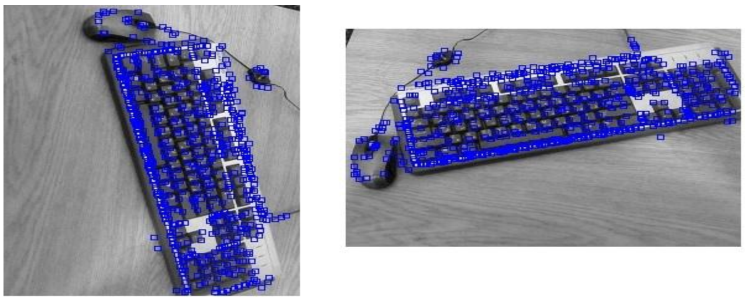


Fig. 7: Some selected keypoints

5. Conclusion

In the paper, we present a fully embedded system for SIFT algorithm to recognize an image. We tried to achieve high performance in the hardware. However, it does not have flexibility the same as software. In comparison to other works in the area, we achieve maximum speed to analyze SIFT algorithm. The recent works have found real time pattern recognition an image with 320x240 resolution and 30 frames per second. But the speed was limited because for the last part, description, they use an embedded processor (NIOS II) [12]. In our design, all sections are implemented in hardware. Preliminary results show the FPGA design running at 100MHz targeting Cyclone III. By this throughput and clock rate, it is possible to process over 50 frames of images at the 320x240 resolution. The design can be improved for high resolution and more frames by doing more parallelism and increase clock frequency.

6. Acknowledgments

Funding for this research was provided by the Islamic Azad University - NazarAbad Branch through its contract research program project on “high speed object recognition based on SIFT algorithm” (Grant No. 23-2706).

7. References

[1] D. G. Lowe: *Distinctive image features from scale-invariant keypoints,*International Journal of Computer Vision, vol. 60, no. 2, 2004.

[2] S.V.Gemignani, M. Demi, M Paterni, M Giannoni and A Benassi: *DSP implementation of real time edge detectors*, In the Proceedings of speech and image processing pp 1721-1725,2001.

[3] Nelson: *Implementation of Image Processing Algorithms on FPGA Hardware,* Masters Thesis, Graduate School of Vanderbilt University, 2000.

[4] Shinichi Hirai, Masakazu Zakouji, Tatsuhiko Tsuboi: *Implementing Image Processing algorithms on FPGA-based Realtime Vision System*, Proc. 11th Synthesis and System Integration of Mixed Information Technologies(SASIMI 2003), pp.378-385, Hiroshima, April, 2003.

[5] Peter Baran, Ralph Bodenner and Joe Hanson: *Reduce Build Costs by Offloading DSP Functions to an FPGA,* FPGA and Structured ASIC Journal.

[6] J. Maddocks & R. Williams: *VHDL Image Processing Source Modules REFERENCE MANUAL*. Nov 2003.

[7] Altera: *Edge Detection Using SOPC Builder & DSP Builder Tool Flow*, ver1.0, Application Note 377, May 2005.

[8] Hong Shan Neoh and Asher Hazanchuk: *Adaptive Edge Detection for Real-Time Video Processing using FPGAs,* GSPx 2004 Conference, 2004.

[9] B. Cope: *Implementation of 2d convolution on fpga, gpu and cpu,* Technical report, Department of Electrical and Electronic Engineering, Imperial College, London, UK, 2006.

[10] N. Dalal and B. Triggs: *Histograms of oriented gradients for human detection,* In Proc. of Computer Vision and Pattern Recognition, 2005.

[11] C. Farabet, C. Poulet, J. Y. Han, and Y. LeCun: *An fpga-based processor for convolutional networks,* In International Conference on Field Programmable logic and Applications, Prague, September 2009. IEEE.

[12] Vanderlei Bonato, Eduardo Marques, and George A. Constantinides: *A Parallel Hardware Architecture for Scale and Rotation Invariant Feature Detection,* IEEE transaction on circuits and systems for video technology, vol.18, no.12, Dec 2008.

[13] Uwe Meyer-Baese: *Digital Signal Processing with Field Programmable Gate Arrays,* Springer, Verlag Berlin Heidelberg, 2007.