International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-8 Issue-4, April 2019

An FPGA based Implementation of SIFT Algorithm

T.Kavya, R .Menaka

Abstract: ***Feature extraction is a significant processing step in any machine learning application. The significant features of the image will be extracted to represent different groups for processing. Hence it must be accurately extracted and it should act as a representative for the complete dataset. Feature extraction plays a major role in facilitating the subsequent learning and generalization steps. Scale Invariant Feature Transform (SIFT) is widely used in a variety of applications like surveillance, object recognition, panoramic view generation etc. The SIFT algorithm has a main advantage of its scale and orientation invariance. The steps in SIFT algorithm incurs complex calculations and hence high power for processing. These steps can be parallelized which permits better performance in algorithm execution. This research presents an attempt for parallel implementation of SIFT algorithm in FPGA.***

Keywords: FPGA, SIFT, Xilinx, Feature extraction

I. INTRODUCTION

In the recent economy development, the traffic sign detection problems become as serious issues. As traffic accidents paid attention, traffic sign detection and recognition attracted much importance. Each traffic sign has different size and representation. This traffic sign became more and more complex as it can be easily changed on the position of its Relative location, Angle of view from the camera, Reflection problems and other surrounding conditions such as Weather, Illumination and Daytime [1]. The concept of feature extraction refers to the method which mainly aims at extracting the information from images and makes local decisions at each and every point on the image and examines every pixel to check for features at that pixel. It is a low-level processing operation. The resulted features were the subset of image domain and often in the form of isolated points [2]. Feature extraction need to be accurate to take better decisions in the case of sensitive areas like medical image processing [14-16]. As there are many ways for object detection and recognition, SIFT is the most popular way to do this. Comparing with the performance of other types of feature descriptors which also includes Harris method, Speeded-Up Robust Features (SURF), and Scale Invariant Feature Transform (SIFT), many experiment results shows that SIFT algorithm performs the best among all. Hence, it is considered as one of the best popular image processing algorithm. Due to its invariance in change of its scale and rotation and also to the change of illumination, and regardless of the outstanding performance of the SIFT feature descriptors on distinctiveness and robustness. SIFT has been applied in a wide range of applications like image stitching [11-12], medical image processing [13]. The need is to implement image processing algorithm in hardware platforms for real time implementation. Fortunately, for the real time applications, FPGA plays an important role as it includes the main and best advantage of reconfigurable logic blocks. These logic blocks support for low-power consumption and high-speed transceiver and also has a huge support for variety of standard I/O pins. Accordingly, we can utilize it to do complex parallel calculations. In the other sense, hardware acceleration and configuration can be executed in a FPGA. Researches compare the performance of parallel hardware architecture as follows: In [3], the hardware system detects keypoint features with resolution of 1920 x 1080 and produces a detection results up to 30 frames per second, and [4] produce up to 2316 descriptors inside the 33ms. In comparison with the [5], the entire design can process up to 30 outlines for every second for 320x240 pixels with 18 Gaussian channels and [6] SIFT is mainly used for picture highlight by considering strength and repeatability factors. This parallel hardware architecture of SIFT can able to detect 1275 SIFT features per frame. In this paper, SIFT algorithm is implemented in FPGA. The Input Image is given as Data format to the SIFT algorithm and the results are simulated and synthesized. The software used for converting Image to Data is Matlab 2013a and Software used for simulation purpose is Xilinx ISE Design Suite 14.7. This paper is further organized as Section II: A brief introduction to SIFT algorithm, Section III: Hardware Architecture of SIFT algorithm, Section IV: Implementation and Results, Section V: Hardware Utilization and Section VI: Conclusion and Future work.

II. SIFT ALGORITHM

The general flow of SIFT algorithm is given as

Manuscript published on 30 April 2019.
* Correspondence Author (s)
T. Kavya*, School of Electronics Engineering, VIT Chennai.
R.Menaka, Associate Professor, School of Electronics Engineering, VIT Chennai.

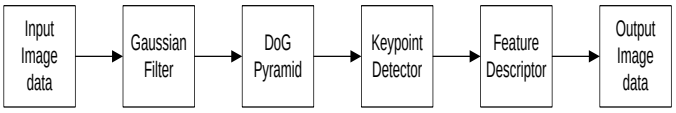

Fig 1. General Flow of SIFT Algorithm

In this section, SIFT algorithm [7] is briefly explained as follows sections

A. Gaussian Pyramid and DoG:

Given an input image for processing, SIFT features are extracted based on their scales using a scale-space representation which is implemented as an image pyramid. Gaussian filters are used to degenerate noise on image solely or make the image smaller as keeping its pixels as uniform as possible. Sometimes, Gaussian filters are applied as preprocessing for removing useless details. The main goal of using Gaussian pyramid to identify location and scale that is repeatedly assigned under various views of the same scene or object. From many experimental analyses and under a variety of assumptions, it is shown that Gaussian function is the best function as it is efficient function for computing as the smoothed images, and D function can be computed easily by the subtraction image. In order to create scale spaces for Gaussian pyramid, Gaussian blur of input image is a required task. It takes to next level of processing. Blurring an image for image processing is exact like, down-sampling it and then up sampling it. The main importance of blurring an image is, after blurring the neighboring pixels all represents the same values.

Mathematically, the blurring of an image is given as

$$L(x,y,\sigma) = G(x,y,\sigma) * I(x,y,\sigma) \quad (1)$$

$$\text{Where, } G(x,y,\sigma) = \frac{1}{2\pi\sigma 2} e^{-(x2+y2)/2\sigma 2} \quad (2)$$

Where L is the blurred image for an input image, G is the Gaussian blur image, I is the input image given, x, y are the location co-ordinates, σ is the scale parameter and * is the convolution operator used. The required amount of blur in each image is important factor. It gets changed by k*σ factor at each stage.

Difference of Gaussian (DoG) is used as an image enhancement algorithm in SIFT. Gaussian filtered images from successive scales are subtracted to build up the DoG. It is utilized to increase the visibility of edges, corners and other details present in digital image i.e., differencing simply is used to locate feature point (extract edges and corners). This function provides a nearest approximation to the Laplacian of Gaussian.

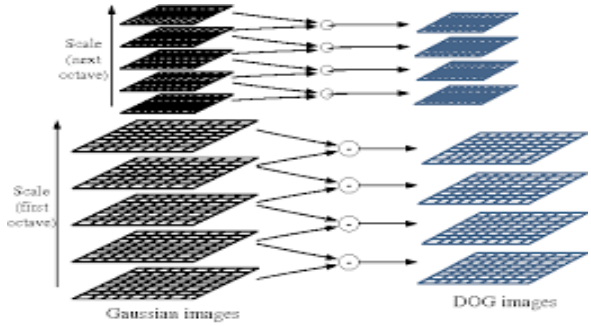

Fig 2: Gaussian pyramid and Difference of Gaussian

Mathematically, it is represented as

$$D(x,y,\sigma) = G(x,y,k\sigma)\times I(x,y) - G(x,y,\sigma)\times I(x,y)$$
$$= L(x,y,k\sigma) - L(x,y,\sigma) \quad (3)$$

Fig 2 on the left shows that input image is convolved with Gaussians to produce set of scale space images. Difference of Gaussian pyramid by the successive subtraction of scales is showed on right.

B. Stable Keypoint Detection

Scale-space extrema detection in the above step produces many keypoints, in which many are unstable. Keypoints along edges, or keypoints don't have enough contrast are not useful as features. So as to build the proficiency of SIFT, low differentiation features and features on edges and corners should be wiped out to get the high vigor.

Low contrast features can be detected after checking their intensities values. Based on the magnitude of intensity value compared to certain value, the pixels in DoG pyramid will be rejected or not. The DoG function have a strong response along edges. Therefore, in order to increase the efficiency, keypoints with poorly determined locations need to be eliminated but have high edge responses. After eliminating the low contrast features, we need to compute Hessian matrix, H, which is given as

$$H = \begin{bmatrix} Dxx & Dxy \\ Dyx & Dyy \end{bmatrix} \quad (4)$$

Consider $\boldsymbol{\alpha}$ as the eigenvalues with largest magnitude and $\boldsymbol{\beta}$ as the eigenvalues of the smallest magnitude. Mathematically, the trace of **H** and their product Determinant of **H** is calculated as

$$\mathbf{Tr(H)} = D_{xx} + D_{yy} = \alpha + \beta \quad (5)$$

$$\mathbf{Det(H)} = D_{xx}\,D_{yy} - (D_{xy})^2 = \alpha\,\beta \quad (6)$$

If γ be the ratio between magnitude of eigenvalues of largest magnitude to that of smaller one. Let $\alpha = \gamma\beta$, Then,

$$\frac{Tr(H)2}{Det(H)} = \frac{(\alpha+\beta)2}{\alpha\,\beta} = \frac{(\gamma\beta+\beta)2}{\gamma\beta 2} = \frac{(\gamma+1)2}{\gamma} \quad (7)$$

Along these lines, in order to wipe out strong edge response points,

$$\frac{Tr(H)2}{Det(H)} \leq \frac{(\gamma+1)2}{\gamma} \quad (8)$$

C. Orientation Assignment

Once the keypoints are distinguished, each keypoint is assigned for at least one orientation dependent on picture slope course. As keypoint descriptor is spoken to in respect to its orientation, this orientation task is a key advance for accomplishing invariance to rotation. To assign orientation for each keypoint, orientation histogram is utilized so most unmistakable slope orientation(s) are accomplished. Another important usage in orientation task is, it can split one keypoint to multiple based on their highest peak values. Once the highest peak in histogram is detected, if it is one

peak, then detected as keypoint else if it is multiple, then peaks with more than 80 percent of detected highest peak is converted and assigned as new keypoint. This newly formed keypoints has equal location and scale as the real input image but with different directions. Mathematically, the equations for gradient magnitude m(x,y) and gradient direction $\theta(x,y)$ calculations are given as

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication (BEIESP)

$$m(x,y)=\sqrt{(L(x+1,y)-L(x-1,y))2+(L(x,y+1)-L(x,y-1))2} \quad (9)$$

$$\theta(x,y)=\alpha\tan2(L(x+1,y)-L(x-1,y), L(x,y+1) - L(x,y-1)) \quad (10)$$

D. Build Feature Descriptor

After keypoints were detected and orientation was assigned to it, building the feature descriptor is the next task. From the above step, we can conclude that SIFT is invariant to scale and rotation. Now descriptor vector for each keypoint is computed such that descriptor is highly distinctive and partially invariant to illumination, 3D view point and other minor affine changes. This step can be performed when image scale is closest to keypoint's scale.

In this histogram, for example, a 16x16 neighborhood around the keypoint is taken. It is divided into 16 sub-blocks that contain samples of 4x4 size of the original neighborhood region. For each sub-block, 8 bin orientation histogram is created. So totally, 128 bin values are available. It is representated as a vector with unity magnitude to form keypoint descriptor. It contributes significantly to the stability of matching as matching exactly between two different images is not an easy task.

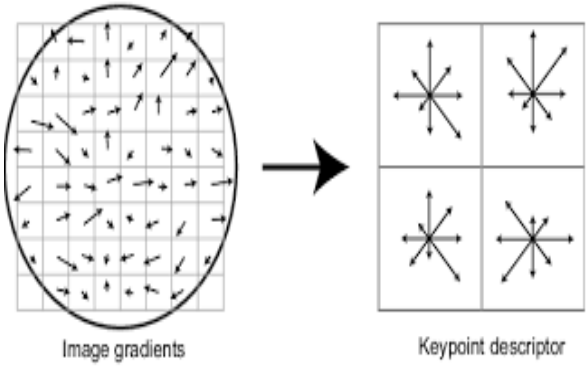

Fig 3: Feature Descriptor

III. FPGA IMPLEMENTATION

A. Introduction

An hardware architecture of SIFT algorithm is given below

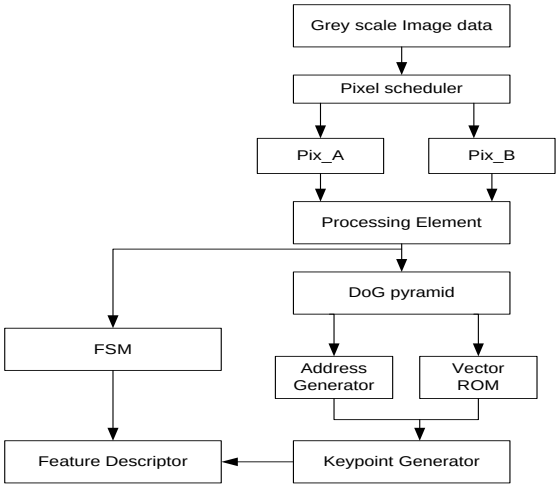

Fig 4: Hardware architecture of SIFT

In this chapter, the hardware architecture of FPGA-Based complete SIFT algorithm and hardware architecture of each block used in algorithm were explained below. The input image for SIFT feature detection is detected by 256 x 256 image with 8 bit each. In real time applications the input section of SIFT detector is given by camera.

The entire system performance can be accelerated by using the parallel architecture. The detection module provides with a constant throughput defined by the number of octaves and scales in the DoG section. In this design, we are using 12 Gaussian filters for the two image octaves.

The detailed description of each block is discussed in the following sections

B. Gaussian module:

Generally, from [8] the Gaussian filters for given input images can be obtained by convolution process of Gaussian kernel image and the input image.

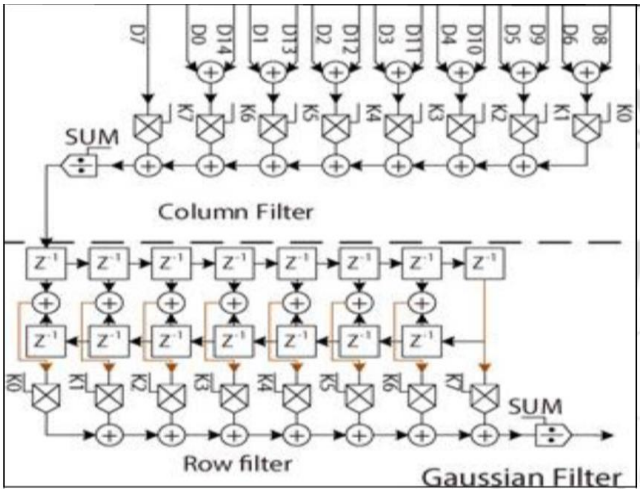

Fig 5: Gaussian Filter

The Convolution is more effective by first convolving the Gaussian kernel weight in vertical direction and then second convolution is done in horizontal direction as shown in above fig 5, which is the traditional filtering approach. In this particular implementation, row Gaussian filter is executed in x direction and column Gaussian filter is executed in y direction.

C. Architecture of DoG module:

The difference of Gaussian module incorporates the three operations such as image buffering, 2-D Gaussian filtering and image subtraction as in Fig 7; the image buffer module used here is a line buffer section providing a series of delay elements that works in parallel. These line buffers provide sharing in the same octave as in Fig 6, The filtering section is separable and provides row and column operation for every pixels, which provides better smoothening of the image.

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication (BEIESP)

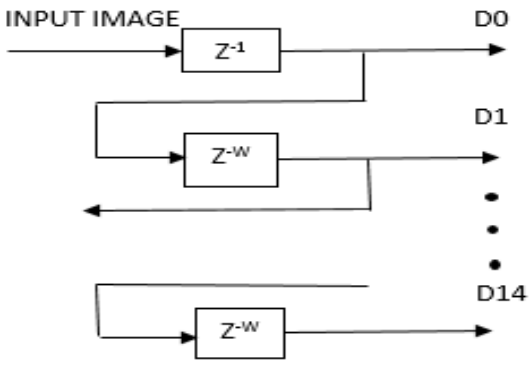

Fig 6: Architecture of image buffer

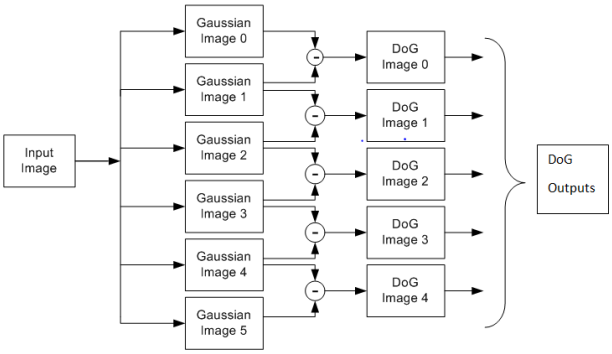

Fig 7: Architecture of DoG module

D. Architecture of Keypoint Detection Module:

The feature detection module is a fully parallelized section with four separate modules [9]. They are window generator, extremum detection, edge response rejection and the low contrast rejection modules. The hardware based parallel architecture for key-point detection is shown in Fig 8. Here the window generator module is to generate 3×3 pixel windows from the DoG outputs. The extremum detection module is used for detecting the local maxima by comparing each pixel value with its neighbors. The edge response rejection module is used to eliminate edge features of the image which makes its key-points unstable. This was done by some edge threshold value. The low contrast rejection module is also used to eliminate or reject the less contrast features to make the key-point detection more accurate. Finally all these modules are combined to implement the stable key-point detector section using SIFT. Due to its parallelized design we can detect a large number of features in a defined clock frequency.

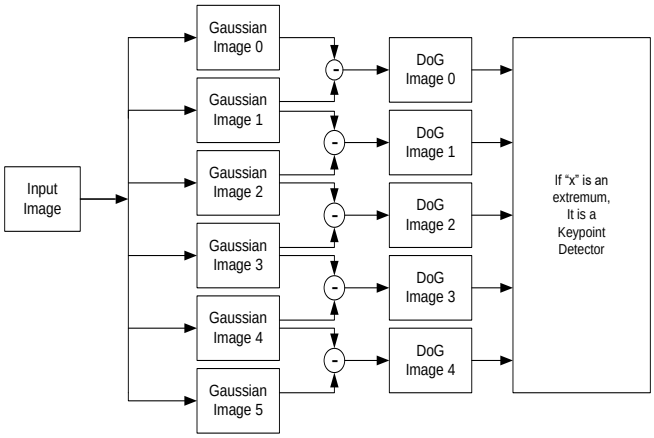

Fig 8: Architecture of Keypoint Detection

E. Feature Descriptor Module:

In this module, for feature descriptor and matching purpose, two images is taken as input i.e., one image is the correct input image and the second image is used for reference [10].

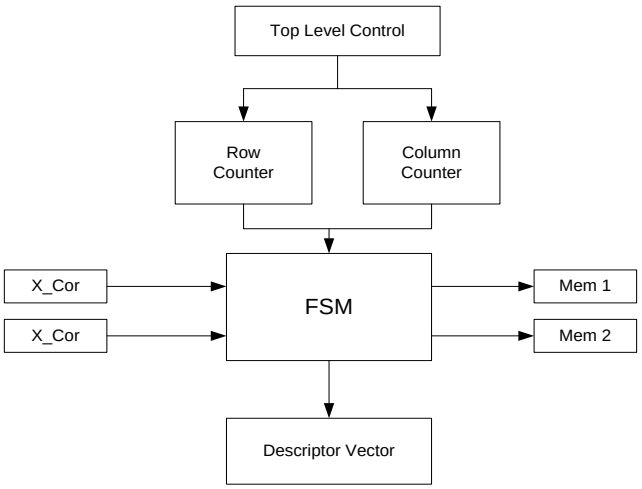

Fig 9: Architecture for feature Descriptor

Each image size used in this design is 256x256 pixels. In respect to the X and Y co-ordinates, by comparing correct input image and the reference image, we can say that Feature was matched are not. In this design, for feature descriptor module and matching, the concept of state machine is used. Among the 9 points, 5th point is taken as middle point and the points 2,4,6,8 are taken as matching for edge responses and the points 1,3,5,7 are taken as matching for corner responses.

IV. IMPLEMENTATION AND RESULTS

The design is implemented in Modelsim PE Student Edition v11.0 with Verilog coding. The input image is given as an input in the form of data. In order to convert image to data, matlab 2013a with matlab coding is used. In this design, RAM module is used to divide the image to sub- blocks for easy calculation and the result after the calculating DoG pyramid is sent as input to RAM. Two different sizes are used as i) RAM_15x15 and ii) RAM_23x23.

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication (BEIESP)

Here, Gaussian filter is separately calculated as module and given directly the values at the top module.

In this design, 2 octaves at the top module for the construction of DoG module were designed. Finally, it calculates each pixel at a time and the final result is stored as image data.

The simulation results of Keypoint Detection Module is given below as

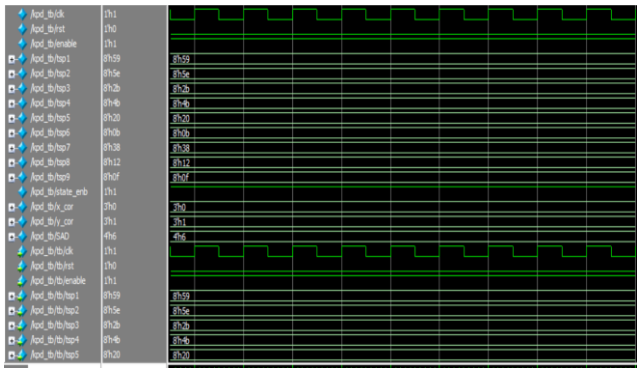

Fig 10: Simulation of Keypoint Detection

The simulation results of Feature Descriptor module is given below as

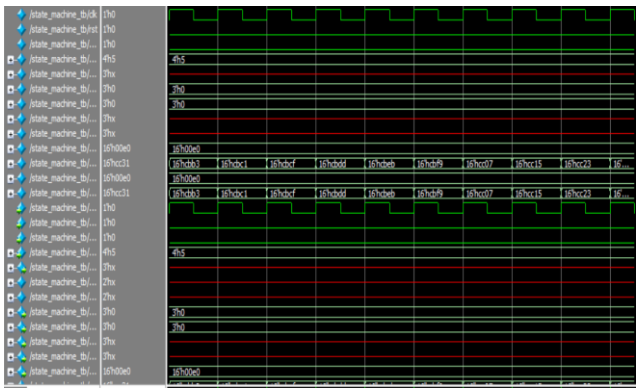

Fig 11: Simulation of Feature Descriptor

Simulation results of overall architecture of SIFT algorithm is given below as

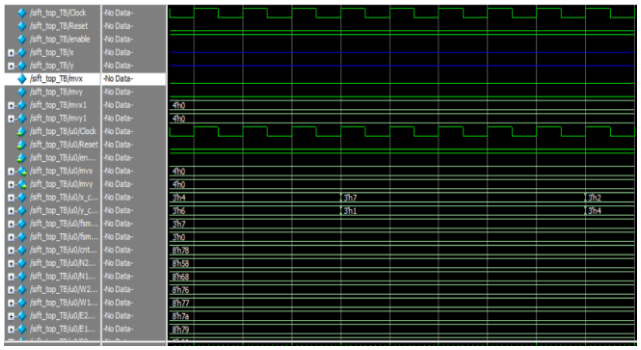

Fig 12: Simulation of SIFT

V. HARDWARE UTILIZATION

In this section, the hardware utilization of each module used in SIFT algorithm is presented individually. This gives the summary of both design and device utilization. Design Summary is given in terms of Flip-flops or Latches used, Clock buffers used and I/O buffers used. The device utilization summary is given in terms of its slice registers, slice LUTs, bonded I/Os and Gate delay. These results are obtained from synthesis report generated from Xilinx Design Suite 14.7 ISE environment. It is individually presented for RAM, Keypoint Detection module and Feature Descriptor module.

This synthesis report is synthesized on Xilinx Spartan 6 (XC6SLX4) target.

Design summary:

Design modules	RAM	KPD	FDM
Flipflops/Latches	7	74	60
Clock Buffers	1	1	1
I/O buffers	79	84	82

Table 1: Design summary for different modules

Device Utilization Summary:

RAM:

Slice Logic Utilization	Used	Available	Utilization
Slice Registers	7	4800	1%
Slice LUTs	53	2400	1%
Bonded I/Os	8	102	7%
Gate delay	6.997ns		

Table 2: Device Utilization summary for RAM

Keypoint Detection:

Slice Logic Utilization	Used	Available	Utilization
Slice Registers	74	4800	1%
Slice LUTs	156	2400	6%
Bonded I/Os	85	102	83%
Gate delay	3.634ns		

Table 3: Device Utilization summary for Keypoint Detection

Feature Descriptor:

Slice Logic Utilization	Used	Available	Utilization
Slice Registers	48	4800	1%
Slice LUTs	102	2400	4%
Bonded I/Os	83	102	81%
Gate delay	3.701ns		

Table 4: Device Utilization summary for Feature Descriptor

Execution time:

The maximum operation frequencies for each module: RAM, Keypoint Detection and Feature Descriptor is given below

Frequency	RAM	KPD	FDM
Operational Frequency (MHz)	379.983	204.182	192.004

VI. CONCLUSION

Real time image processing algorithms can be implemented in hardware using FPGA as they have temporal and spatial parallelism. Feature detection in images is a very frequently used process in image processing for various biometric and video tracking applications. Among many algorithms, SIFT algorithm is one of the best for local features extraction even under varying illumination, scale and angles.

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication (BEIESP)

FPGA-Based hardware architecture for SIFT algorithm is presented in this work. The various steps in SIFT implementation as keypoint detection, feature description and bin segregation are implemented and the utilization summary is presented in this work. The future work includes as, the hardware resource utilization can be reduced at the stage of building Gaussian pyramid. The multipliers and adders used in Gaussian filter can be replaced by array multipliers and array adders for better utilization.

REFERENCES

1. Xiaoguang HU, Xinyan ZHU, Deren LI and Hui LI, "Traffic Sign Recognition Using Scale Invariant Feature Transform and SVM", ISPRS Technical Commission IV & AutoCarto Fall Speciality Conference, 2010.
2. Feixiang ren, jinsheng Huang, ruyi Jiang, Reinhard klette, "General Traffic Sign Recognition by Feature Matching", 24th International Conference on Image and Vision Computing NewZealand, 2009.
3. J. Q. Peng, Y. H. Liu, Fellow, IEEE , C.Y. Lyu, Y. H. Li , W. G. Zhou and K. Fan, "FPGA-Based Parallel Hardware Architecture For SIFT Algorithm", IEEE International Conference on Real-time Computing and Robotics, 2016
4. Murad Qasaimeh, Assim Sagahyroon and Tamer Shanableh, Department of Computer Science and Engineering, American University of Sharjah, "A Parallel Hardware Architecture for Scale Invariant Feature Transform (SIFT)", IEEE, 2014
5. Vanderlei Bonato, Eduardo Marques, and George A. Constantinides, "A Parallel Hardware Architecture for Image Feature Detection", 2014.
6. J. Wang, S. Zhong, L. Yan, and Z. Cao "An Embedded System-on-chip Architecture for Real-time Visual Detection and Matching", IEEE Transactions on circuits and systems for video technology, vol. 24, no. 3, pp. 525-538, March 2014.
7. https://opencv-pythontutroals.readthedocs.io/en/latest/ py_tutorials /py_feature2d/py_sift_intro/py_sift_intro.html
8. Syama K Nair, Ragimol, "Modified SIFT Algorithm for Image Feature Detection", International Journal of Science and Research (IJSR), Vol. 5, Issue 7, 2016.
9. S. Zhong, J. Wang, L. Yan, L. Kang, and Z. Cao, "A real-time embedded architecture for SIFT," J. Syst. Arch., vol. 59, no. 1, pp. 16-29, Jan 2013.
10. V. Bonato, E. Marques, G. A. Constantinides, "A Parallel Hardware Architecture for Scale and Rotation Invariant Feature Detection", IEEE Transaction on Circuits and System for Video Technology, vol. 18, no. 12, pp. 1703-1712, Dec. 2008.
11. Menaka, R., & Karthik, R. (2016). A novel feature extraction scheme for visualisation of 3D anatomical structures. International Journal of Biomedical Engineering and Technology, 21(1), 49.
12. R. Karthik, A. AnnisFathima and V. Vaidehi, "Panoramic view creation using invariant momentsand SURF features," *2013 International Conference on Recent Trends in Information Technology (ICRTIT)*, Chennai, 2013, pp. 376-382.
13. Fathima, A. A., Karthik, R., & Vaidehi, V. (2013). Image Stitching with Combined Moment Invariants and Sift Features. Procedia Computer Science, 19, 420–427.
14. Menaka, R, Chellamuthu, C, Karthik, R Efficient feature point detection in CT images using Discrete Curvelet Transform. Journal of Scientific and Industrial Research, 312-315, 2013.
15. Karthik. R and Menaka. R. A critical appraisal on wavelet based features from brain MR images for efficient characterization of ischemic stroke injuries. Electronic Letters on Computer Vision and Image Analysis 15(3):1-16; 2016.
16. Karthik, R. and Menaka, R. "Statistical characterization of ischemic stroke lesions from MRI using discrete wavelet transformation", Transactions on Electrical Engineering, Electronics, and Communications, Vol. 14, No. 2, pp. 57-64, 2016.