# FPGA Implementation of SIFT Algorithm Using Xilinx System Generator

*Dovari .Swaraj[1], Dr.G.L.Madhumati[2]*
*[1]P G Scholar VLSID, [2]Professor and Head, Department of Electronics & Communication Engineering*
*Dhanekula Institute of Engineering& Technology[1,2]*
*[1]dovariswaraj@gmail.com,[2]madhumati70@rediffmail.com*

**Abstract: This paper presents scale invariant feature transform (SIFT) implementation using system generator used for object recognition. The features are invariant to image scale and rotation, and are shown to provide robust matching across a substantial range of affine distortion, addition of noise, and change in illumination. The features are highly distinctive, in the sense that a single feature can be correctly matched with high probability against a large database of feature from many images. In this paper integral image architecture is implemented using Xilinx system generator for real time application needs. Architecture is implemented in Atlys partan6 xc6xls16-3csg324 FPGA board.**

**Keywords: SIFT, system generator, simulink, VLSI design**

## I. INTRODUCTION

Extracting robust features is an essential task in pattern recognition and image processing applications. Several feature extraction and description algorithms have been proposed to detect distinctive and repeatable points that are able to describe the content of the images. Scale invariant feature transform (SIFT) algorithm is considered as one of the most robust algorithm in computer vision. The SIFT features are invariant to change in scale, rotation, illumination and camera viewpoint. SIFT was proposed by David Lowe[1] as a robust feature for these problems and the object recognition method .This algorithm used in navigation for detection of ships and planes and used in traffic controls and detection of objects in many applications. SIFT detects and describes local feature points in images for object detection in different scenes or different angles. These feature points describe the strength and direction of the object, which resists the frame distortion such as zooming, rotation, movement, perspective change, shading and noise. To provide these stable points, SIFT computation involves the image convolution with Gaussian filters at different scales and local maximum or minimum of the difference of Gaussian (DOG) blurred images at multiple scales. However, the real time implementation of this algorithm faces the challenges of heavy computation such as large memory storage and long computational latency because of its frame level computation with iterated Gaussian blur operations on images and the frame difference operations on blurred images for feature extraction. Thus, a fast algorithm [2] and its VLSI design are demanded. Mainly SIFT algorithm consist of 4 steps they are:

1. **Scale-space extrema detection:** The first stage of computation searches over all scales and image locations. It is implemented efficiently by using Gaussian blur, [3] a difference-of-Gaussian function to identify potential interest points that are invariant to scale and orientation.
2. **Keypoint localization:** After detection of maxima at each candidate location, a detailed model is fit to determine location and scale. Keypoints are selected based on measures of their stability.
3. **Magnitude and Orientation assignment:** One or more orientations are assigned to each keypoint location based on local image gradient directions. All future operations are performed on image data that has been transformed relative to the assigned orientation, scale, and location for each feature, thereby providing invariance to these transformations.
4. **Keypoint descriptor:** The local image gradients are measured at the selected scale in the region around each keypoint. These are transformed into a representation that allows for significant levels of local shape distortion and change in illumination. Fig 1 shows the SIFT algorithm block diagram.

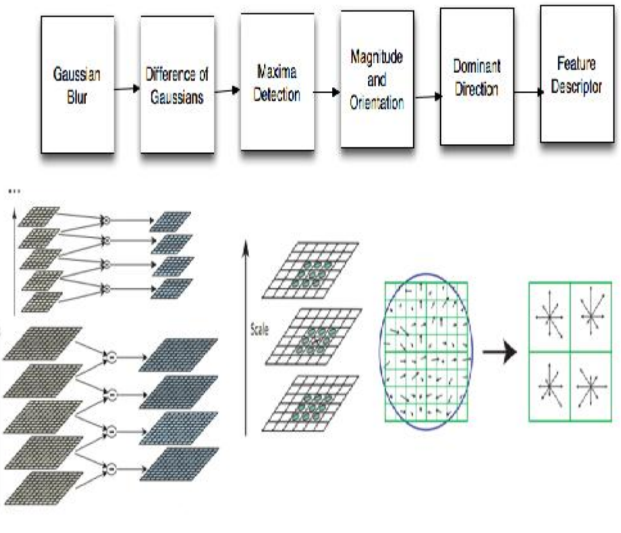


Fig1: SIFT Block Diagram

This paper performs SIFT algorithm implementation using system generator. It optimizes the Gaussian scale space generation module using integral image concept to reduce the utilization of FPGA resources. The rest of the paper is organized as follows. Section II introduces the System generator. Section III discusses SIFT algorithm using

Simulink in MATLAB. Section IV illustrate about image-pre processing and post processing and Section V deals with Xilinx models for the operation of image processing and Section VI presents the proposed hardware architecture and its simulation results and Section VII furnish details about FPGA implementation Finally, section VIII concludes this paper.

## II. XILINX SYSTEM GENERATOR

System Generator is a DSP design tool from Xilinx that enables the use of the MathWorks model-based Simulink design environment for FPGA design. The design tools facilitate the design process by obscuring the technical knowledge necessary for FPGA a Register Transfer Level (RTL) design. Instead, a design is modeled using the intuitive visual environment within Simulink that uses several specific block sets accelerate the development. Additionally, System Generator can perform the FPGA implementation steps: synthesis, mapping, and place and route to generate the FPGA executable file. System Generator is part of the ISE® Design Suite and provides Xilinx DSP Block set such as adders, multipliers, registers, filters and memories for application specific design. These blocks leverage the Xilinx IP core generators to deliver optimized results for the selected device. Designs are captured in the DSP friendly Simulink modeling environment using a Xilinx specific Block set. All of the downstream FPGA implementation steps including synthesis and place and route are automatically performed to generate an FPGA programming file. Advantage of using Xilinx system generator for hardware implementation is that Xilinx Block set provides close integration with MATLAB Simulink that helps in co-simulating the FPGA module with pixel vector provided by MATLAB Simulink Block. For accomplishing Image processing task using Xilinx System Generator [4] needs two Software tools to be installed. One is MATLAB Version R2013b or higher and Xilinx ISE 14.5. The System Generator token available along with Xilinx has to be configured to MATLAB. Fig 2 shows the Xilinx system generator flow.

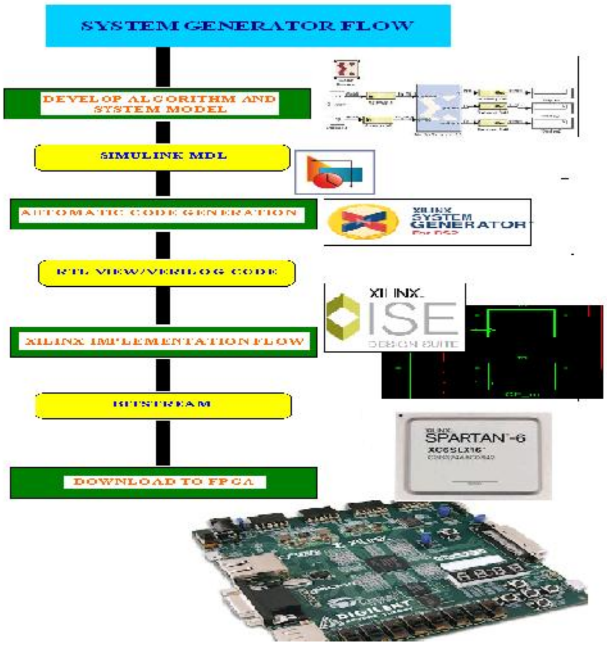


Fig 2: Xilinx System Generator Flow

Once the expected results are obtained System Generator is configured for suitable FPGA board. Xilinx ATLYS board with Spartan6 xc6xls16-3csg324 FPGA is used. Clock planning is done and the model is implemented for JTAG hardware co-simulation. The System generator parameters are set and generated. On compilation the net list is generated and a draft for the model and programming file in Verilog HDL is created which can be accessed using Xilinx ISE. The module is checked for behavioral syntax check, synthesized and implemented on FPGA. The Xilinx System Generator itself has the feature of generating User constraints file (UCF), Test bench and Test vectors for testing architecture. Xilinx System Generator has created primarily to deal with complex Digital signal processing (DSP) applications, but it has other application of this theme such as image processing also work with it. Bit stream compilation is done which is necessary to create an FPGA bit file which is suitable for FPGA input.

## III. SIFT ALGORITHM USING SIMULINK IN MATLAB

First the concept of SIFT algorithm is realized by building a simulink model using various blocks. Simulink has many predefined blocks out of them few blocks like “image from file” block is utilized for taking image from desktop and “Gaussian pyramid” block is for Gaussian filtering and for displaying the output image or videos in the simulink “video display” block is used. In the implementation of SIFT algorithm Gaussian Pyramid block computes Gaussian pyramid reduction or expansion to resize an image. The image reduction process involves low pass filtering and down sampling the image pixels. The image expansion process involves up sampling the image pixels and low passes filtering. By connecting those blocks the implemented SIFT algorithm is shown in Fig 3.

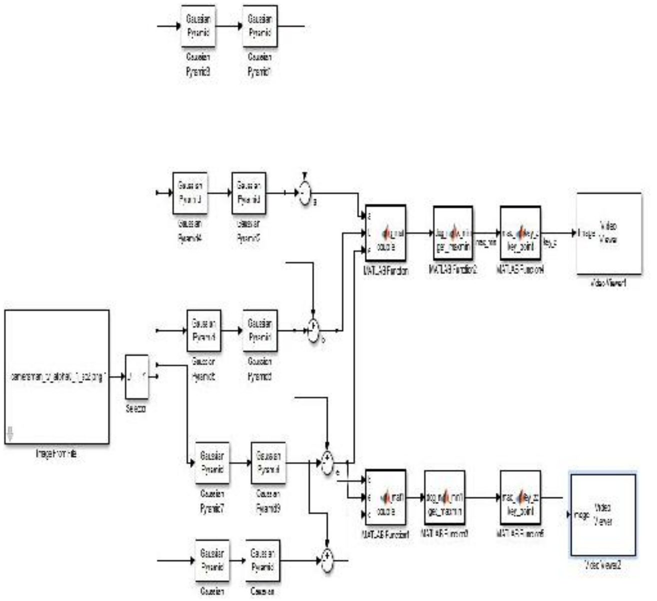


Fig 3: SIFT Algorithm Implementation in Simulink

The implementation of SIFT first reads the image and then apply Gaussian pyramid block for smoothing the image and then subtract the image from above image this step is called Difference of Gaussian(DOG). After DOG find the maxima by comparing the 3 images above and below. Write the code for maxima detection after detection describes the keypoints. Results of Gaussian pyramid is shown in Fig 4.

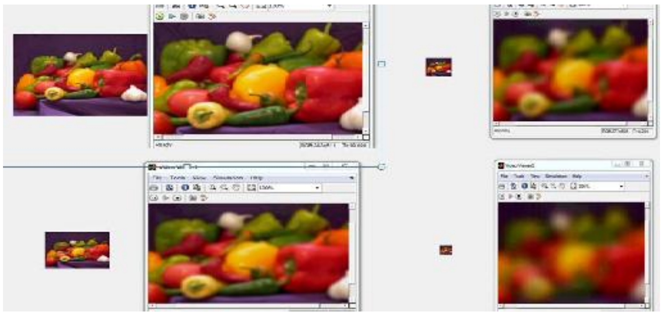


Fig 4: Results of Gaussian Pyramid

The output results are shown in Fig 5.

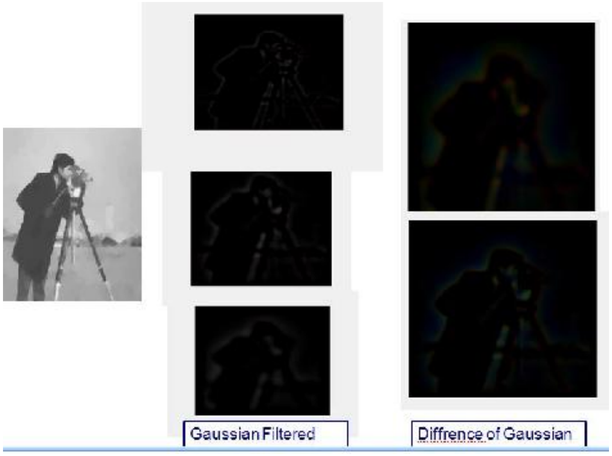


Fig 5: Results of Gaussian pyramid and Difference of Gaussian (DOG)

## IV. IMAGE PRE-PROCESSING AND POST-PROCESSING

In System generator tool there is need for image pre-processing and post-processing. For doing operations on images pre-processing is required. In pre processing first that image is converted into array of pixels. Image pre-processing [5] in MATLAB helps in providing input to FPGA as specific test vector array which is suitable for FPGA Bit stream compilation using system generator.

- Resize: Set Input dimensions for an image and interpolation i.e. bicubic it helps in preserving fine detail in an image.
- Convert 2-D to 1-D: Converts the image into single array of pixels.
- Frame conversion and buffer: It helps in setting sampling mode and buffering of data.

The model based design used for image pre-processing is shown in Fig 6 and the blocks utilized here are discussed. Input images which could be color or grayscale are provided as input to the File block.

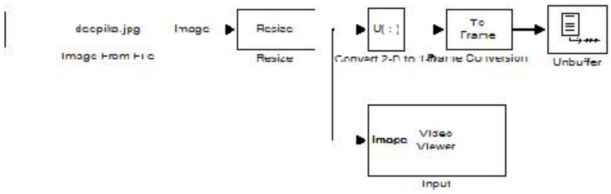


Fig 6: Image pre- processing

In case of 5X5 Filter generation the system is clocked 1/5 times with normal clocking. The internal delay is occurred in this block which causes certain rows at the bottom of the image to be shifted to the top. To avoid this error the image is translated at the bottom for same number of rows. This helps in retrieving the entire image after filtering.

Image post- processing helps recreating image from 1D array shown in Fig 7. Post-processing uses-

->Data type conversion: It converts image signal to unsigned integer format.

-> Buffer: Converts scalar samples to frame output at lower sampling rate.

-> Convert 1D to 2D: Convert 1D image signal to 2D image matrix.

-> Sink: It is used to display the output image back on the monitor.

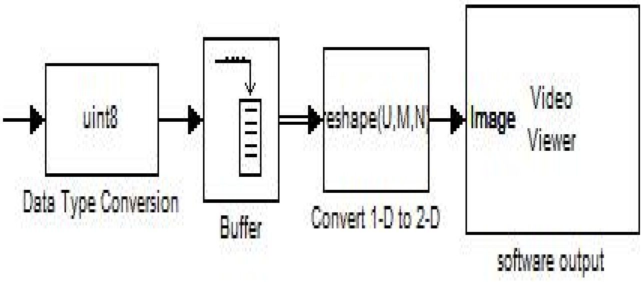


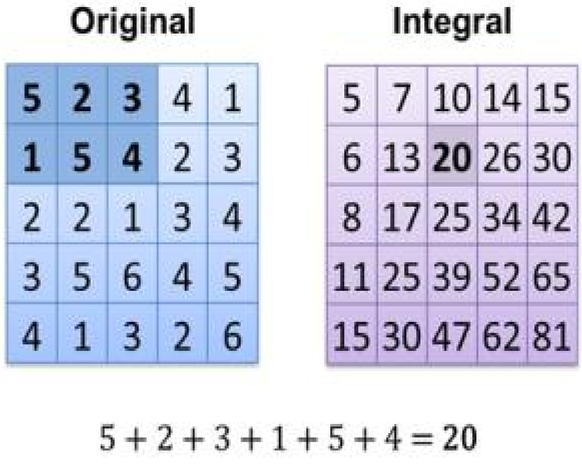
Fig 7: Image Post - processing

## V. XILINX MODELS FOR OPERATIONS OF IMAGE PROCESSING

For implementation of SIFT algorithm integral image approach is used to create a box kernel with the response similar to that of the Gaussian filter so as to keep the feature matching performance as close as that of SIFT but without complex Gaussian filters. By adopting the integral image approach for multiple box kernels in each scale, reduce computational complexity significantly and enable parallel computation for all scales.

### Integral Image Concept:

The extraction of key points in an image is a process that requires image filtering. Integral images [6] are very easily computed by adding the right pixel values. In an integral image every pixel is the sum of all pixels located in a rectangular window formed by that pixel and the origin, with the origin being the most top-left pixel. Fig 8 below shows the integral image concept.



Original

| | | | | |
|---|---|---|---|---|
| 5 | 2 | 3 | 4 | 1 |
| 1 | 5 | 4 | 2 | 3 |
| 2 | 2 | 1 | 3 | 4 |
| 3 | 5 | 6 | 4 | 5 |
| 4 | 1 | 3 | 2 | 6 |

Integral

| | | | | |
|---|---|---|---|---|
| 5 | 7 | 10 | 14 | 15 |
| 6 | 13 | 20 | 26 | 30 |
| 8 | 17 | 25 | 34 | 42 |
| 11 | 25 | 39 | 52 | 65 |
| 15 | 30 | 47 | 62 | 81 |

$$5+2+3+1+5+4=20$$

Fig 8: Integral Image implementation Concept

## VI. INTEGRAL IMAGE ARCHITECTURE USING SYSTEM GENERATOR

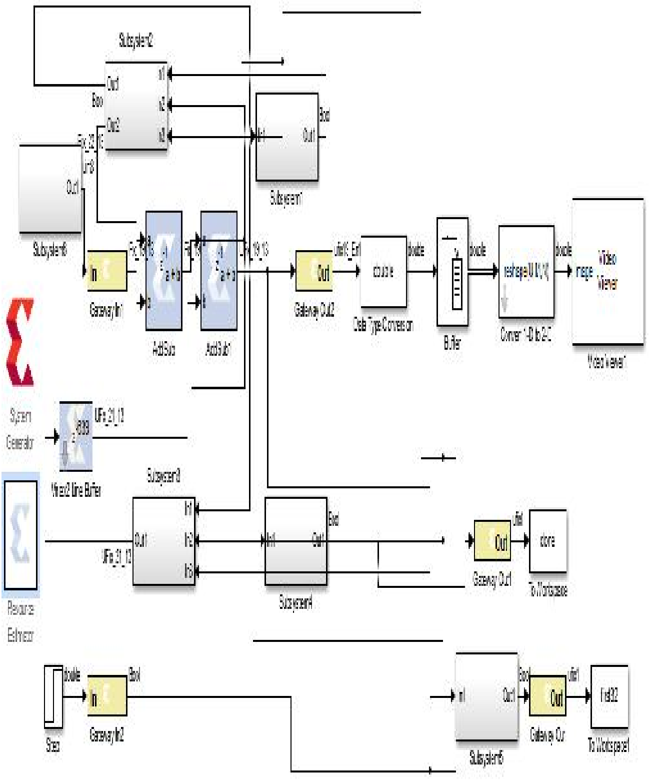


Fig 9: Integral Image Architecture

For the implementation of SIFT algorithm using integral image there is no need of Gaussian filters. Connect the blocks according to steps followed in SIFT algorithm. Fig 9 shows the architecture for integral image concept of SIFT algorithm implemented using system generator and results are shown in Fig 10.

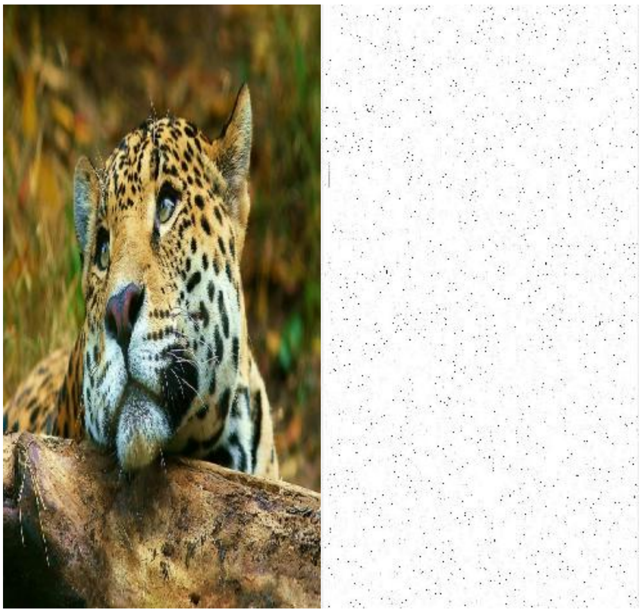


Fig 10: Results of Integral Image

## VII. FPGA IMPLEMENTATION

System generator output generates the Xilinx files and HDL code. Now open ISE [7] and open the Xilinx file which was generated by the system generator block. After opening the Xilinx file, simulate, synthesize and then implement the design. For implementing in FPGA kit interface HDMI/DVI ports for displaying output. Using user constraints file (.ucf) assign the pins correctly. After integrating the HDMI port and system generator output click implement on Xilinx and then generate the bitstream file. After generating the net list or bit stream file is dumped into the FPGA with the help of Digilent Adept software. After dumping the bit stream file, the information is processed in FPGA and then output will be displayed on the Monitor.

Below Table 1 shows the comparison between SURF algorithm [6] and SIFT algorithm using integral image. By comparing these two algorithms the components used by SURF algorithm is more. Over all the SIFT algorithm reduces 40% of slices and 20% of flip-flop's and 30% of LUT's by using this phenomenon. SIFT use less number of components so the complexity is less and the latency is also less in SIFT algorithm using integral image concept.

Table 1: Comparison between SURF and implemented SIFT Algorithm

| Components | SURF Algorithm | Implemented SIFT Algorithm |
|---|---|---|
| Slices | 400 | 146 |
| FF's | 100 | 87 |
| BRAM'S | 90 | 4 |
| LUT'S | 680 | 241 |
| IOB'S | 179 | 41 |
| DSP48'S | 0 | 0 |

## VIII. CONCLUSION

This paper presents SIFT algorithm for object recognition using Xilinx System Generator tool, a new application in image processing, which offers a friendly environment design for the processing. This tool supports software simulation but the most important is that it can synthesize in FPGA hardware with the parallelism, robust and speed. These features are essentials in image processing. By using integral image, the complexity of SIFT algorithm is reduced.

.

## REFERENCES

[1] D. G. Lowe, "Distinctive image features from scale-invariant key points," *International Journal of Computer Vision,* Vol. 60, No. 2, pp. 91–110, 2004.

.

[2] Liang-Chi Chiu, Tian-Sheuan Chang, *Senior Member, IEEE*, Jinn-Yen Chen, and Nelson Yen-Chung Chang "fast SIFT design for real time visual feature extraction".

[3] Wennie Tabib School of Computer Science "FPGA-Based Feature Detection".

[4] Neha. P. Raut1, Prof.A.V.Gokhale2 "FPGA Implementation for Image Processing Algorithms Using Xilinx System Generator".

[5] Alareqi Mohammed, Elgouri Rachid1,and Hlou Laamari "High Level FPGA Modeling for Image Processing Algorithms Using Xilinx System Generator".

[6] H. Bay, T. Tuytelaars, and L. J. V. Gool, "SURF: Speeded up robust Features," in *Proceedings of European Conference on Computer Vision*, 2008, pp. 404–417.

[7] V. Bonato, E. Marques, and G. A. Constantinides, "A parallel hardware Architecture for Scale and Rotation Invariant Feature Detection," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 18, No. 12, pp. 1703–1712, Dec. 2008.

**First Author:**

**D. Swaraj** was born in Kuchipudi, Andhra Pradesh, India on 17th June 1991.He received his B.Tech degree in Electronics and Communication Engineering from Vasavi Engineering College, Tadepalligudem in 2012 and currently pursuing M.Tech in VLSI Design from Dhanekula Institute of Engineering & Technology, Ganguru. His research interests includes Image processing and VLSI.

**Second Author:**

**Dr.G.L.Madhumati** is working as Professor and Head of the Department of Electronics and Communication Engineering, Dhanekula Institute of Engineering and Technology, Ganguru, Vijayawada. She completed her Doctoral degree in Mixed Signal VLSI Design in the

Year 2011 from Jawaharlal Nehru Technological University Hyderabad, Hyderabad. Her research interests include design of low power and mixed signal circuits, Digital Design using PLD's, Mixed signal Design, Data Converters Design, Hardware/Software Co-design, Memory Design, Embedded System design and Digital Image Processing architectures and applications. She has published 16 publications in various journals and conferences at National and International level. Presently she is guiding three students for Ph.D in VLSI area.