IOSR Journal of Electrical and Electronics Engineering (IOSR-JEEE)
e-ISSN: 2278-1676,p-ISSN: 2320-3331, Volume 9, Issue 5 Ver. II (Sep – Oct. 2014), PP 69-78
www.iosrjournals.org

FPGA Implementation of Background Subtraction Algorithm for Image Processing

[1]Ms. P.D. Mahamuni, PG Student, [2]Prof. R. P. Patil Asst. Professor
[1,2] *E & TC Department,SKNCOE, Vadgaon Bk, Pune, India*

Abstract: *The Background Subtraction Algorithm is used to detect the object in the image. The subtraction of the image is pixel based subtraction. Before the subtraction of two images the pre-processing is done using the median filter. After the subtraction is done the segmentation is performed using threshold value. Once the segmentation is completed the morphological filtering is performed on that image to remove the unwanted blobs. System is designed using MATALAB/Simulink block sets. In MATLAB/Simulink the system generator token is used to convert Simulink model logic into the VHDL code which makes it compatible with FPGA. Using the system generator token VHDL code is generated. That VHDL code is synthesis in the Xilinx 14.1 and then .bit file is downloaded to the Virtex-5 board through JTAG cable and the output image is display on the VGA monitor which is interfaced with Virtex-5 board using DVI connector.*
Keywords: *Motion detection, Background subtraction algorithm, FPGA, VGA monitor.*

I. Introduction

In recent years extensive investigations and analyses have been done in the domain of moving object detection. Detection of moving objects in video processing plays a very important role in many vision applications. The vision systems that include image processing methods are widely implemented in many areas as traffic control, video surveillance of unattended outdoor environments, video surveillance of objects etc. In moving object detection object detection is first step. The motion detection algorithms implemented in these video systems provide low-level information that can be used by higher level algorithms to determine the desired information (the trajectory of an object, the control of traffic flow, etc). Methods for object detection must be accurate and robust so that complex video systems can operate successfully. Most of the existing algorithms for object detection assume that the illumination in a scene remains constant. Unfortunately, this assumption is not valid, especially in outdoor environment. The efficiency of some of existing techniques diminishes significantly if the illumination varies. There are two types of methods that realize moving object detection. One detects changes at pixel level and the other is based on feature comparison. The first method is better because of very fast detection of any kind of changes in the analyzed scene and it is implemented in the technique proposed in this paper.

II. Literature Survey

Detecting objects in image sequence is an important problem in computer vision, with applications of several fields, such as video surveillance and target tracking. Most techniques reported in the literature use background subtraction techniques to obtain foreground objects detection algorithms exploring information of the images to retrieve only valid objects present in image.

A segmentation algorithm begins with an identification phase that detects objects theoretically belonging to foreground pixel. The input data of segmentation is image sequence that can be both RGB and gray scale images. The literature reports various contributions for the initial segmentation of the foreground pixels. Some of the segmentation algorithms, named background subtraction, temporal difference, optical flow [10]. Chien et al. [9] builds a background model starting with the assumption that, if a pixel is stationary for a prefixed number of consecutive frames, the probability that it belongs to the background model.

One of the intermittently used methods for the subtraction of two images is temporal differencing where two or three adjacent frame based on time series image to subtract and gets difference images. This method is highly adaptive to dynamic scene changes; however, it generally fails in detecting whole relevant pixels of some types of moving objects.

A commonly used technique for object segmentation in static images is background subtraction. It will detect object regions by subtracting the current image pixel-by-pixel from a reference background image that is created by averaging images over time in an initialization period. This method is simple and easy to realize, and accurately extracts the characteristics of target data.

In some of the cases where the motion target of the vector characteristics which changed with time to detect motion area in image sequences then optical flow method is used. This method is complex and requires complicated computation and additional hardware support [8]. Shao-Yi Chien[9] proposed an efficient moving

segmentation algorithm. A background registration technique is used to construct reliable background information from the video sequence. Then, each incoming frame is compared with the background image. If the luminance value of a pixel differs significantly from the background image, the pixel is marked as moving object; otherwise, the pixel is regarded as background. Finally, a post-processing step is used to remove noise regions and produce a more smooth shape boundary.

Jorge H. [6] implemented video processing modules in a Xilinx Virtex-4 FPGA (XC4VFX12) which has been designed using VHDL and synthesized using XST of Xilinx. M. J. Black [12] implemented the background subtraction algorithm on FPGA. Processor Microblaze is designed in VHDL (VHSIC hardware description language), implemented using XILINX ISE 8.1 Design suite the algorithm is written in system C Language and tested in SPARTAN-3 FPGA kit by interfacing a test circuit with the PC using the RS232 cable. The test results are seen to be satisfactory. The area taken and the speed of the algorithm are also evaluated.

III. System Design

In this paper, object detection is described using background subtraction algorithm. Motion detection methods are basically a process which detects the object in the surveillance area [1-4].

The object in the frame sequences is detected using background subtraction. For object detection, two images preferably of the same size are taken. In that, one image is initialised as the background image in which the object is not present and the second image is the current image in which the object is present. Each image has two models one is the foreground and background model. The Fig.1 shows the system block diagram.

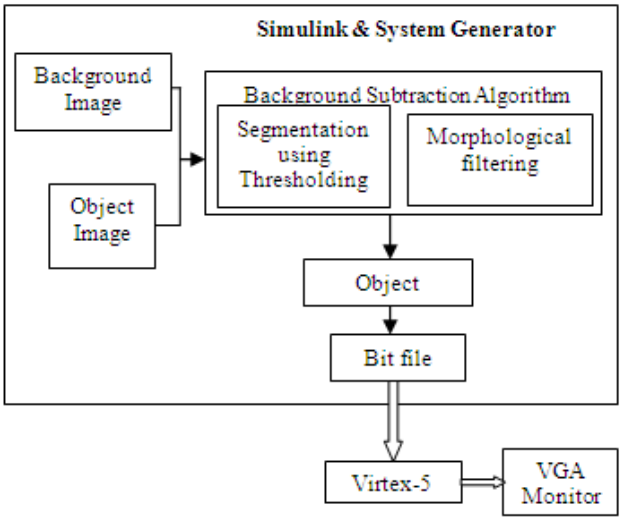


Fig. 1 System Block Diagram

The foreground model is the model in which the moving object is present and background model is the model in which the moving object is not present. The first process for motion detection is image initialisation. Image initialization is process that initializes the background image. For example, in the video the number of the frames with respect to the time, out of these frame one is initialized as the background image by taking some assumption. Hence initialization of background is essential preprocessing operation for object detection. The preprocessing is done on each frame using median filter for reducing the noise from the image. Both images are given to the background subtraction algorithm in that different operations are performed on the images such as subtraction, segmentation, morphological filtering.

1) Background and Object Image: There are many ways to obtain the initial background image. For example, the first frame as the background directly, or the average pixel brightness of the first few frames as the background or using a background image sequences without the prospect of moving objects to estimate the background model parameters and so on. Among these methods, the time average method is the most commonly used method for the establishment of an initial background. In this system background image is captured separately then for same background object image is captured.

The input images are still images. These images are converted from RGB to gray color using the MATLAB code. These images are selected in the Simulink model using the 'image from file/workspace' block. The size of the stored image is 640x280 but it will be resized to 400x400 in the MATLAB code.

2) Background Subtraction Algorithm: Background Subtraction Algorithm is implemented in four steps: pre-processing, background subtraction, segmentation and morphological filtering. These steps are explained in detailed in next section.

These all processing steps are designed in the MATLAB using Simulink block sets. The MATLAB/Simulink models are used for the compatibility of implementing the image processing system on FPGA. Because it has direct functions related to the image processing and it also provides the Xilinx block sets for the FPGA implementation. Using system generator token the VHDL code is generated and synthesis in Xilinx ISE design 14.1 and .bit file is generated. That .bit file is downloaded on the Virtex-5 board.

The Virtex-5 ML506 evaluation board is user friendly which is used to implement this system. For this system very less hardware is required such as JTAG cable, RS232 and VGA monitor. The downloading process of Virtex-5 is easy to understand. This board gives very fast results.

A. Software Design

In this paper, background subtraction algorithm is used for the detection of object in the surveillance area. The Fig. 2 shows the flow of the background subtraction algorithm based object detection process which consists of the images as the input.

1) Input Images: The input images are still images. These images are converted from RGB to gray color using the MATLAB code. These images are selected in the Simulink model using the 'image from file/workspace' block. The size of the stored image is 640 x 280 but it will be resized to 400x400 in the MATLAB code.

2) Pre-processing: This is the first step in background subtraction algorithm and is performed on both input images. The aim of pre-processing is an improvement of the image data that suppresses unwanted distortions or enhances some image features important for further processing. The pre-processing can be done using median filtering, mean filtering and convolution. It will remove the noise which is present in the input images. Due to impurities in the background, the discrepancy image obtained contains the motion region as well as noise. This noise might be included in the image due to environmental factors, illumination changes, during transmission of video from the camera to further processing. Therefore, noise needs to be removed.

For this system median filter is used in Simulink model. In Simulink library the 'median filter' block is directly available for pre-processing. This in turn blurs the image frames which help in shadow removal.

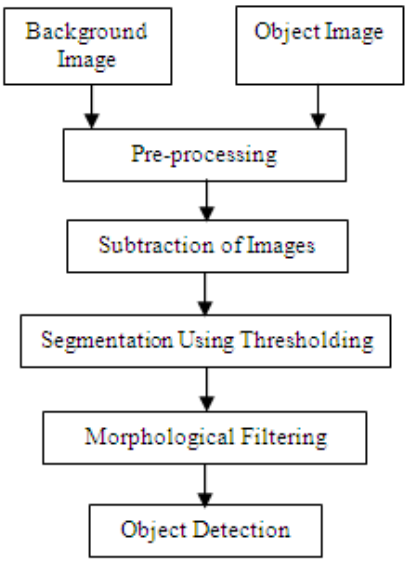


Fig. 2 Flow of background subtraction algorithm

3) Background Subtraction: After the pre-processing is done on the input images the subtraction of the background image and object image is done. It removes the background from the image to detect the object in that image. This happens because backgrounds of the both input images are same so the pixel value of the background in both input images are same.

This subtraction code is written in MATLAB using subtracts function and that code is used in 'm-function' block of the Simulink model. M-function block is a block in which the MATLAB code is written for the functions. In this the subtraction is done pixel by pixel.

4) Segmentation using Thresholding: Thresholding is a procedure that eliminates an unwanted range of pixels in the scene with respect to certain threshold values. Data validation is involved with the collection of techniques to reduce the misclassification of pixels. The threshold value is very important for the clarity of the image. Deciding the threshold value gives the pixel information, when a pixel is supposed to be considered as either a background pixel, or a foreground pixel:

Pixel (x , y) is foreground pixel, if

$$| f(x,y) | > T_d \quad (1)$$

Pixel (x , y) is background pixel, if

$$| f(x,y) | < T_d \quad (2)$$

Where, T_d is the Threshold value.

After the subtraction of two images, the pixel value is compared with the threshold value and then tagged to '1' or '0'. After the segmentation the object detected image is in binary form.

$$| f(x,y) - B(x,y) | < T_d \ ; \quad (3)$$

Pixel tagged to 0= background (black)

$$| f(x,y) - B(x,y) | > T_d \ ; \quad (4)$$

Pixel tagged to 1= moving object (white)

Where, $f(x,y)$ is the pixel value of background image

$B(x,y)$ is the pixel value of object image

The image segmentation used in this system is threshold segmentation. To select the threshold values for the segmentation of an image observed the pixel values of the subtracted image. According to the threshold value the corresponding pixel is divided into two categories, the foreground and background. In the simplest case the image after the single-threshold segmentation can be defined as

$$g(x,y) > T \quad (5)$$

Where, $g(x,y)$ is the subtracted image,

T is threshold value.

Threshold segmentation has two main steps:

1) Determine the threshold T

2) Pixel value will be compared with the threshold value T

The above steps are used for segmentation. For determining the threshold value of system the subtracted image is read in MATLAB using 'imread' function. It gives all pixel values of the subtracted image. By observing the pixel value the threshold value is selected.

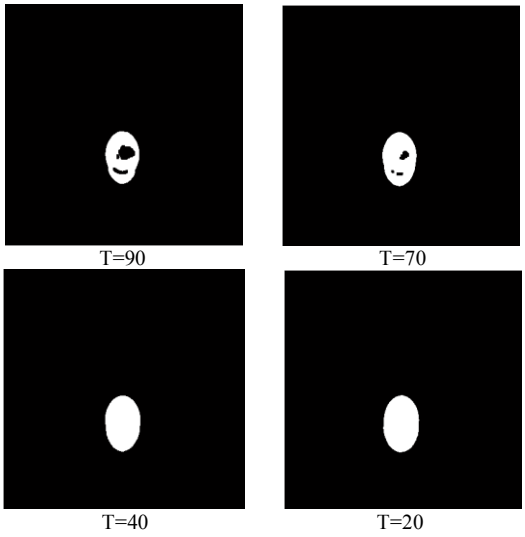


Fig. 3 Results at Different Threshold Values

Then the each pixel value is compared with that threshold value and tagged to '1' or '0'. If the pixel value is greater than threshold value it is tagged to '1' i.e. object, and if less than threshold then tagged to '0' i.e. background. This is done in MATLAB coding.

The above Fig. 3 shows the output images at different threshold values to the results of the same images. In the algorithm, after the subtraction of the image segmentation is done using thresholding. The threshold values are decided by observing the subtracted image pixel values. The threshold value should be as small as possible. The below Fig. 3(a) shows the result at threshold value 90 and Fig. 3(b), (c) and (d) shows the result at 70, 40, 20 threshold values respectively. By observing the above result images it is concluded that for good result threshold value should be less, because at threshold values 40, 20 it gives good result as compared to result at thresholds 90, 70.

5) Morphological filtering: The segmented frame is now given to the morphological filtering for reducing the noise. The function of the morphological filtering is to remove the small regions probably created by noise; fill up unnecessary cavities, smoothing boundaries, extracting edges. It will give pixel level operations. In Simulink model the 'erosion' and 'dilation' function blocks are available for the morphological filtering.
6) Motion detection: After the segmentation and morphological filtering the moving object is clearly seen in the frame and that will be the output of our system which is display on the VGA monitor.

C. Simulink model for Object Detection

The Simulink model shown in Fig. 4 is prepared in MTALAB with the help of Simulink library browser which contains the Image and Video processing block set. The software model for object detection consists of Image from file, Video Viewer, Embedded function block, parameter blocks. Both the input images are displayed using the video viewer block. Each time run the Simulink model, the different image dataset has to be selected.
1) Image from File Block: This block is basically used to access the two input images which are to be subtracted. Double-clicking on this block, to select the path

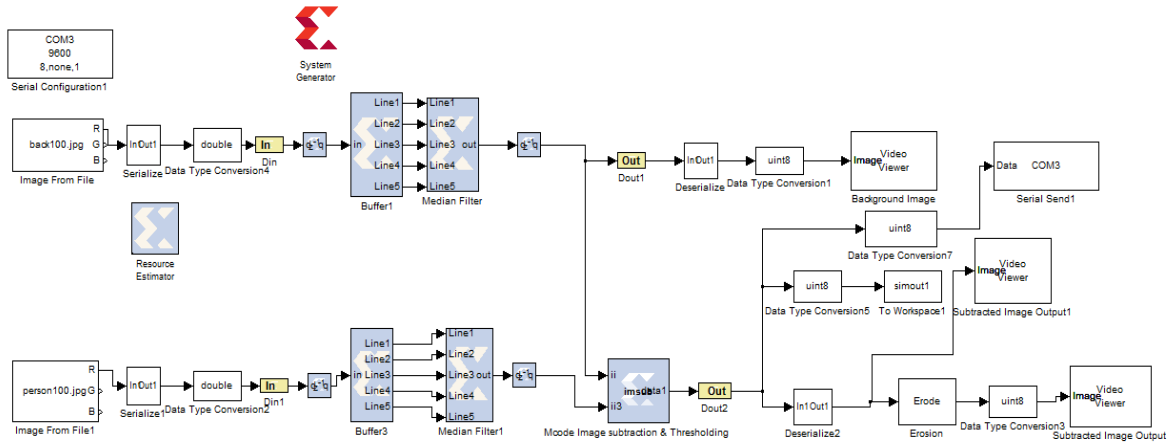


Fig. 4 Simulink Software for Motion Detection

Where, the two input images are stored with size 256 × 256. Each time select the images from file to run the different datasets. The use of Image from File block is to import an image from a supported image file also to read the image from file. If the image is a M-by-N array, the block outputs a binary or intensity image, where M and N are the number of rows and columns in the image. If the image is a M-by-N-by-P array, the block outputs a color image, where M and N are the number of rows and columns in each color plane. It can provide the image signal selection for this model. The image signal is set to separate the color but for the system, gray scaled images are required. So color signals are terminated.
2) Video viewer: The Video Viewer block enables to view a binary, intensity, or RGB image or a video stream. The block provides simulation controls for play, pause, and step while running the model. The block also provides pixel region analysis tools. During code generation, Simulink Coder software does not generate code for this block. This block is used to display the clock images in MATLAB. The input images namely, input image 1 and input image 2 are displayed using this block.
3) Gateway In: This Xilinx block is the basic element of Xilinx Block set. This block is used to convert the Simulink input of type integer, single or double format to Xilinx data type. This Xilinx block is the basic element of Xilinx Block set. This block is listed in the following Xilinx block set libraries: Basic Elements, Data Types, Floating-Point and Index. The Xilinx Gateway In blocks are the inputs into the Xilinx portion of your Simulink design. These blocks convert Simulink integer, double and fixed-point data types into the System Generator fixed-point type. Each block defines a top-level input port in the HDL design generated by System Generator. This block is used to convert the Simulink input of type integer, single or double format to Xilinx data type.
4) Gateway Out: This block is listed in the following Xilinx block set libraries: Basic Elements, Data Types, Floating-Point and Index. Xilinx Gateway Out blocks are the outputs from the Xilinx portion of your Simulink design. This block converts the System Generator fixed-point or floating-point data type into a Simulink integer, single, double or fixed-point data type. According to its configuration, the Gateway Out block can either define an output port for the top level of the HDL design generated by System Generator, or be used simply as a test

point that is trimmed from the hardware representation. This Xilinx block is used to convert the Xilinx data type of fixed-point or floating point into Simulink data type.

5) Serialize: This block is basically a sub-system which is used to convert the data into serial form, since the Xilinx system generator supports serial data transfer.

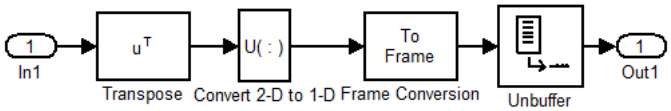


Fig. 5 Simulink Model for Serialization

Fig. 5 shows the different steps of serialize block, are their like transpose of frame and then conversion of 2D to 1D for serialization and finally un-buffering of the data to serially transfer it to FPGA. Transpose gives the data in form of [Mx1] vector. Then that vector is converted into 1-Dimension, and then un-buffers the data to serially transmit it.

6) Register: This block is listed in the following Xilinx Block set libraries: Basic Elements, Control Logic, Memory, Floating-Point and Index. The Xilinx Register block models a D flip-flop-based register, having latency of one sample period.

7) Data Type Conservation: Convert the input to the data type and scaling of the output. The conversion has two possible goals. One goal is to have the Real World Values of the input and the output be equal. The other goal is to have the Stored Integer Values of the input and the output be equal. Overflows and quantization errors can prevent the goal from being fully achieved. For this model it will set to RWV.

8) Virtex2-5 Line Buffer: The Xilinx Virtex2-5 Line Buffer reference block buffers a sequential stream of pixels to construct 5 lines of output. Each line is delayed by N samples, where N is the length of the line. Line 1 is delayed 4*N samples, each of the following lines are delay by N fewer samples, and line 5 is a copy of the input. This block uses Virtex2 Line Buffer block which is located in the Imaging library of the Xilinx Reference Block set.

9) Median Filter: The Xilinx 5x5 Filter reference block is implemented using 5 n-taps MAC FIR Filters. The filters can be found in the DSP library of the Xilinx Reference Block set. Nine different 2-D filters have been provided to filter gray scale images. The filter can be selected by changing the mask parameter on 5x5 Filter block The 2-D filter coefficients are stored in a block RAM, and the model makes no specific optimizations for these coefficients.

10) Resource Estimator: This block is listed in the following Xilinx Block set libraries: Tools and Index. The Xilinx Resource Estimator block provides fast estimates of FPGA resources required to implement a System Generator subsystem or model. These estimates are computed by invoking block-specific estimators for Xilinx blocks, and summing these values to obtain aggregated estimates of lookup tables (LUTs), flip-flops (FFs), block memories (BRAM), 18x18 multipliers, tri-state buffers, and I/Os.

10) De-serialize: This Xilinx block is again used to convert the data in parallel i.e. de-serialize form, since it is given as input to Simulink block. Fig. 6 shows de-serialize block which is the exactly inverse process of the serialize process. This will first buffer the data and then convert it into 2-Dimention form and then transpose is taken of the data.

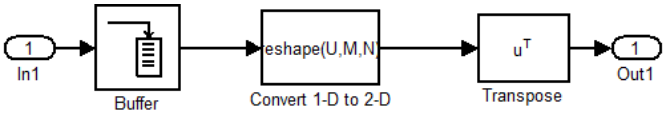


Fig. 6 Simulink Model for De-serialization

11) Embedded MATLAB function block: This block is also called as S-function block; in this block code for the designed algorithm is embedded. This block is programmed using Background Subtraction algorithm for subtraction of the two input images and the software model is prepared as shown in Fig. 4. The subtracted image of size 100×100 is then displayed using Video Viewer block. .

12) Simout: This block is used to write input to specified timeseries, array, or structure in a workspace. For menu-based simulation, data is written in the MATLAB base workspace. Data is not available until the simulation is stopped or paused. For command-line simulation using the sim command, the workspace is specified using DstWorkspace field in the option structure. This block is used to log a bus signal, and uses "Timeseries" save format.

13) Erosion: It is one of the function of the morphological filtering. The Erosion block slides the neighborhood or structuring element over an image, finds the local minima, and creates the output matrix from these minimum values. If the neighborhood or structuring element has a center element, the block places the minima there. If the neighborhood or structuring it does not have an exact center, the block has a bias toward the upper-left corner

and places the minima there. Use the Neighborhood or structuring element parameter to define it that the block applies to the image. Specify a neighborhood by entering a matrix or vector of ones and zeros. Specify a structuring element using the 'strel' function.

14) Xilinx System Generator Token: Xilinx System Generator token available in Xilinx block set is basically used for FPGA compatibility i.e. HDL code generation of the designed Simulink model. Using both Simulink and Xilinx blocks in the model helps for Xilinx System Generator simulation, code generation, and synthesis. Double clicking on the Xilinx system generator will ask for the path to store the generated code, for which device to generate, the code and in which language like HDL, VHDL. For this system the Virtex-5 board is used and VHDL code is generated.

IV. Implementation And Results

Background subtraction algorithm is designed in MATLAB/Simulink and it implemented on Virtex-5 FPGA as shown in the following Fig. 7.

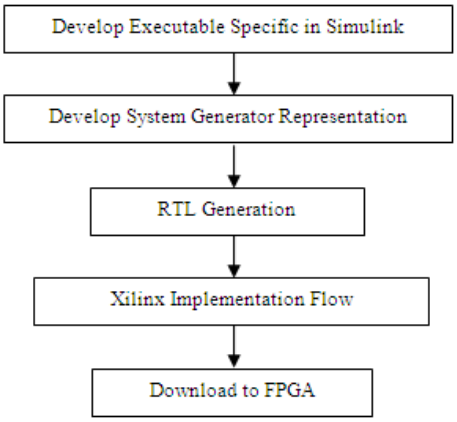


Fig. 7 Design and Implementation Flow

Develop Executable Specific in Simulink: In this step generate the Simulink model for the system using Simulink block sets in MATLAB.

Develop System Generator Representation: Taking the reference of Simulink model generate another system generator model for the same system using Xilinx block sets for the compatibility of FPGA implementation. Then the VHDL code is generated using System generator token. The VHDL code which is generated using the system generator is open in the Xilinx ISE 14.1. Then synthesis of the code will give the information about the errors, warnings, RTL schematic of that code, device utilization summary etc.

RTL schematic of the Simulink model: Synthesis is a process by which an abstract form of designed circuit behavior or register transfer level (RTL) has been converted into a design implementation i.e., in terms of logic gates. The synthesis of VHDL code has been carried out by Xilinx Synthesis Technology (XST) tool, which is a part of Xilinx ISE 14.1 software. The Top-level RTL schematic for the Background Subtraction Algorithm developed and implemented on FPGA is shown in Fig. 8. This is a schematic representation of the pre-optimized design shown at the Register Transfer Level (RTL). This representation is in terms of generic symbols, such as adders, multipliers, counters, AND gates, OR gates and is generated after the HDL synthesis phase of the synthesis process. The two default clock drivers are available for the system. This system blocks are designed for the Virtex-5 ML506 board.

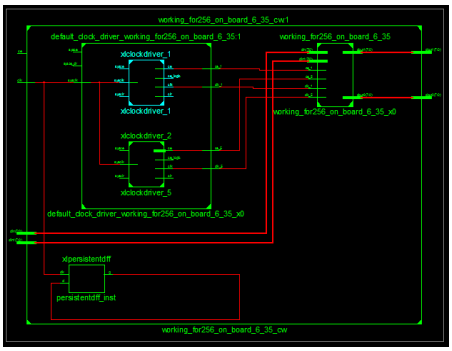


Fig. 8 Top-level RTL Schematic

Device Utilization Summary: Object detected in image of size 100 X 100 is done using Xilinx 14.1, on Virtex-5 evaluation b

Table I. Device Utilization Summary

Device Utilization Summary(Estimated values)			
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slice Registers	961	32640	2%
Number of slice LUTs	361	32640	1%
Number of used LUT-FF pairs	339	18123	34%
Number of bonded IOBs	33	480	6%
Number of BUFG	1	32	3%

Device utilization summary shows information related to device utilization analysis. Detailed reports provide access to reports that are generated as the design is processed. Secondary reports - provides access to the secondary reports of users choice. Table I shows the device utilization summary for the system. Slice registers available are 32640, out of which 961 are used for implementation which is equal to 2% of available resources. Number of slice LUTs used is 1% which gives 361 usages out of 32640. Usage of LUT-FF pairs is 339 out of 18123 which is equal to 34%. Utilization of Bonded IOBs is 33 out of 480 which 6% of available IOBs. Numbers of BUFG used are 1out of 32 which is 3% of available. These summaries tell that the use of the device or hardware for system is very less.

The system is implemented using Xilinx 14.1 and evaluation board of Virtex-5 ML506. Connection is made using JTAG cable for downloading the program to FPGA, DVI connector for VGA interface.

Fig. 9 Experimental Setup for System

Fig. 9 shows the experimental setup for Virtex-5 board. On laptop the .bit file and Simulink models in MATLAB are generated. And using Xilinx 14.1 version Xilinx platform studio the .bit file is synthesis and downloaded to Virtex-5 board. Output is displayed on VGA monitor for this DVI connector is used to interface the VGA monitor to Virtex-5 board. RS232 serial cable is used to transmit the image serially to the board. After the synthesis of the VHDL code the synthesis report is generated in which the device utilization summary is given.

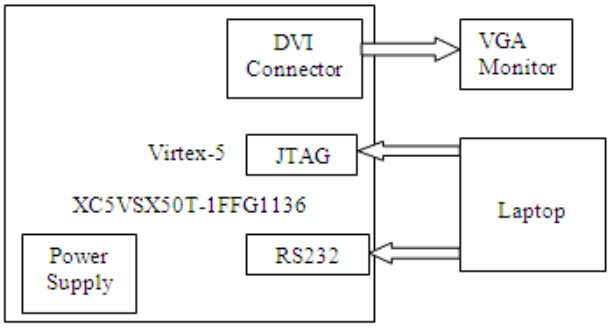


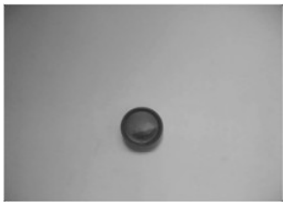
Fig. 10 Interfacing of Virtex-5 with VGA

The above Fig. 10 showes interfacing of Virtex-5 with VGA monitor for display the subtracted object image. VGA monitor is interfaced with Virtex-5 using DVI connector which is avilable on the board. The JTAG cable is connected to laptop and board is used to download the program to FPGA. The RS232 cable is used to transfer the images from laptop to board for the processing, it transfer the data serially.

The Fig. 11(a) shows the backgrounds image which the object is not present and this image of size 400 x 400 Fig. 11(b) shows the object image in which the background is same as in background image and the object is ball. So when the subtraction of the background image and object image is done the object is detected in the final image.

The Fig. 11(c) shows the output image in which the object is detected. The subtraction of the image is done pixel by pixel so the background of the current image and the reference image has the same pixel value due to this subtraction of the background pixel is zero so it will be represented in the black colour and the object is represented by white colour.

Fig. 11(a) Background image



(b) Object image

(c) Output image

For checking the accuracy of system some parameter values are calculated for the image. The accuracy of the ball object detection is 99.8431, recall is 0.06132, precision is 0.22804, and MCC is 0.1177. Recall parameter gives the measure of completeness. Defined as number of true positives pixels divided by the total number of elements that actually belong to the foreground objects. Precision gives the exactness of the algorithm. Segmentation is properly done or not is checkout using the Matthew Correlation Coefficient.

Fig.12(a) Object Image

(b) Subtracted Image

Above Fig. 12(a) shows another object. The object is tree. The Fig.12(b) shows the subtracted image. It gives good object shape. But the shape of some of the part of the tree is not detected due to light effect the color of the background and leaves of the tree are matched due to this it will not detect the shape of the leaves. It gives accuracy 97.998, recall 0.1192, pricision 0.0280 and MCC is 0.5578. So at the end the output image is the binary image. After the binary image getting the morphological filtering is applied to that image and it will perform the operation like opening, closing, sharpening the edges and it will also remove the noise from that frame.

V. Conclusion

The implementation of Background Subtraction Algorithm on FPGA is described, and it is carried out successfully. The hardware and software implementation of system is found to be working properly. The MATLAB/Simulink models are used for the compatibility of implementing the image processing system on FPGA. Because it has direct functions related to the image processing and it also provides the Xilinx block sets for the FPGA implementation. The Virtex-5 ML506 evaluation board is user friendly which is used to implement this system. For this system very less hardware is required such as JTAG cable, RS232 and VGA monitor. The result of background subtraction algorithm is a new image that retains the most desirable information and characteristics of each input image. The subtracted image contains greater information about the object only which is in foreground model. It provides an effective way of reducing the increasing volume of information while at the same time extracting all the useful information from the source images.

References

[1]. M. Kalpana Chowdary , S. Suparshya Babu , S. Susrutha Babu , Dr. Habibulla Khan"FPGA Implementation of Moving Object Detection in Frames by Using Background Subtraction Algorithm" International conference on Communication and Signal Processing, April 3-5, 2013.

[2]. Can Yang, Weichuan Yu "Moving Object Detection by Detection Contiguous Outliers in the Low Rank Representation" IEEE transaction on pattern analysis and intelligence, vol. 35, no.3, 2013.

[3]. C. S´anchez-Ferreira, J. Y. Mori, C. H. Llanos "Background Subtraction Algorithm for Moving Object Detection on FPGA " Department Mechanical Engineering University of Brasilia 2012.

[4]. "Motion and Feature Based Person Tracing In Surveillance Videos" transactions on computer vision 2011.

[5]. Jorge Hiraiwa, Enrique Vargas "An FPGA based Embedded Vision System for Real-Time Motion Segmentation" IWSSIP 2010 - 17th International Conference on Systems, Signals and Image Processing

[6]. Ying-Hao Yu, Q. P. Ha, and N. M. Kwok, "Chip-based Design for Real-time Moving Object Detection using a Digital Camera Module", The International Congress on Image and Signal Processing, CISP, 2009

[7]. Bahadır Karasulu "Review and Evaluation of Well-known Methods for Moving Object Detection and Tracking in Videos" Journal of aeronautics and space technologies july 2010 volume 4 number 4 (11-22)

[8]. Thomas B. Moeslund, Adrian Hiton, "A survey of advances in vision-based human motion capture and analysis", Computer Vision and Image Understanding 104 (2006) 90–126

[9]. Shao-Yi Chien "Efficient Moving Object Segmentation Algorithm Using Background Registration Technique" IEEE transactions on circuits and systems for video technology, vol. 12, no. 7, july 2002

[10]. Christopher Wren, Ali Azarbayejani "Pfinder: Real-Time Tracking of The Human Body", published in IEEE Transaction on Pattern Analysis and Machine Intelligence July 1997 vol.19

[11]. Michael J. Black "The Robust Estimation of Multiple Motions: Parametric and Piecewise-Smooth Flow Fields" Computer vision and image understanding Vol. 63, No. 1, January, pp. 75–104, 1996