ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 1, January 2014

Parallel Architecture for Optical Flow Detection Based on FPGA

Mr. Abraham C. G[1], Amala Ann Augustine

Assistant professor, Department of ECE, SJCET, Palai, Kerala, India [1]

M.Tech Student, Department of ECE, SJCET, Palai, Kerala, India [2]

ABSTRACT: Optical flow means the movement of an object, it creates a motion effect between observer and seen. It mainly deals with motion analysis; in motion analysis moving body detection is the important part. This analysis combines with modern mechanics like computer vision and has been employed in intelligent control, human computer interaction, surveillance application, and virtual reality. The detection of moving objects from the background image in a video sequence has an effective role to play in follow up treatment such as object classification, target tracking and behaviour savvy. This article focuses to find the moving object by background subtraction and then median filtering is provided. To figure out a complete object, brightness based optimization threshold method also is practiced. The Parallel architecture goal is achieved by using an FPGA for processing over a DSP processor.

Keywords: Edge detection, Delta frame, Background subtraction, Frame separation, Optical flow

I. INTRODUCTION

Optical flow estimation is very relevant; it has much application in today's market, academic communities and in real time video processing requirements. In current scenario, optical flow estimation of a video sequence remains as a challenge. There is no accurate or specific method for this optical flow estimation. The existing method for this computation will not provide correct segmentation, so the obtained output will not give an exact image of the object. The most sensing applications require some form of digital signal processing so, these are implemented primarily on serial processors. Here a MATLAB based segmentation is done; then the further processing is to obtain an accurate image of the object. While the required output is achievable, low cost and low power consumption are achieved by the FPGA. Field programmable gate array contains a number of programmable logical components. So FPGA can be programmed according to our needs, quick development and prototyping is also possible. The FPGA is programmed to perform complex mathematical functions, making them suitable for any type of matrix applications. The main attraction FPGA is parallel architecture.

Currently background subtraction is utilized for the movement detection in a video sequence. For this subtraction method, segmentation of video sequence into frames is performed through the serial processor, and fix the first frame obtained as the background. Each frame comes in the orderliness of the video sequence; One subtracted from the first frame which set as the background frame. This subtraction between two consecutive will determine the presence of the moving object. This calculation is very simple and implemented simply; it has a strong adaptability for a variety of dynamic environments. A different method is used to detect moving object from the current image and background image.

II. PROPOSED SYSTEM

This block diagram shows the proposed model of the system, to be executed. First a video is fed into a MATLAB program through the GUI, which will read .avi file, and convert it into frames. Each of the file is stored in an individual bitmap file, these bitmap files are stored in the order of occurrence of their arrival in the video.

ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 1, January 2014

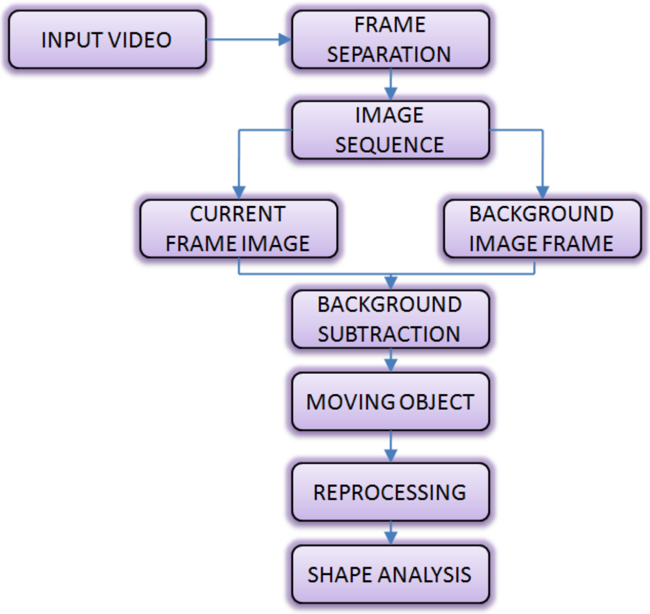


Figure 1: Proposed system

The first frame is selected as background and the remaining frames are subtracted from the beginning, the resultant images used for target tracking, reorganization. Then this resultant image will determine whether any movement occurs between two frames, by this motion detection can be identified. When the motion is detected we have to reprocess the resultant image for getting an accurate image of the object. In reprocessing, the image goes through median filtering for reducing the noise in the image, before that a thresholding is applied for differentiating background and foreground. After reprocessing, it goes through shape analysis segment. For the analysis of shape, the image is subjected to edge detection procedure. By this the boundaries of the image detected and shape of the object is also clearly visible.

III. WORKING OF THE SYSTEM

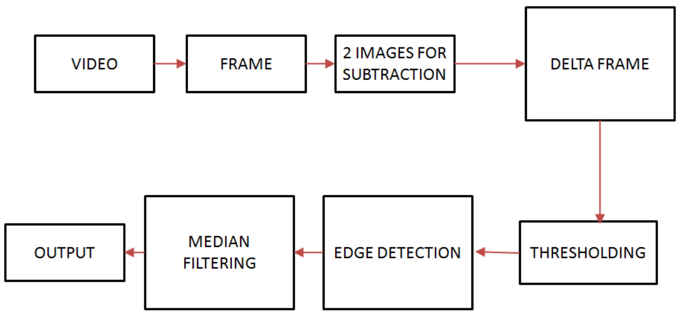


Figure 2: Working of the system

The working of the system can be explained using the block diagram given here. The video is given to the proposed system, and then the video segmentation has taken place. 31 frames per second is obtained, the first frame obtained is taken as the background image, Each of the other frames taken as the foreground image for the subtraction purpose. The resultant image after the subtraction forms a delta shape. The delta frame image is gone through a brightness based threshold value for differentiating the foreground and background. Then median filtering is used for getting a noise free image. The median filtering value is obtained by examining the neighborhood pixel values. To obtain the precise

ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 1, January 2014

shape of the image edge detection is employed. The sobel edge detection method is used, that will preserve all relevant details of the image. After going through this series of process the output will show in the visual basic display screen. The overall working of the system is explained with the help of figure 3. In this the hardware portion is also included. The FPGA has not enough memory to store a video clip. So the video clips converted into several frames. Frame separated bitmap files are in RGB format at a resolution of 640x480 pixels, in order to store this image in FPGA, resolution has to reduce; that is the grey scale conversion of the colour image. By this the pixel value lies between 0 and 255. Our system is designed using system C language, because it is a universally accepted language and also hardware interaction has is becoming easy. But the problem is the grey scale image is not detected by this system, and also size has to reduce. So as shown in figure 3 we convert the grey scale image into a header file format, the size is also reduced. When both the foreground and background image are converted into header file it will be stored in the SRAM of FPGA for further processing. Then the other system C based code is also loaded into Spartan 3 FPGA. The header file conversion of each image is shown in the figure 4.A MATLAB coding is used for this conversion. After this transition, it will reduce coherent effect and easily separate object from background.

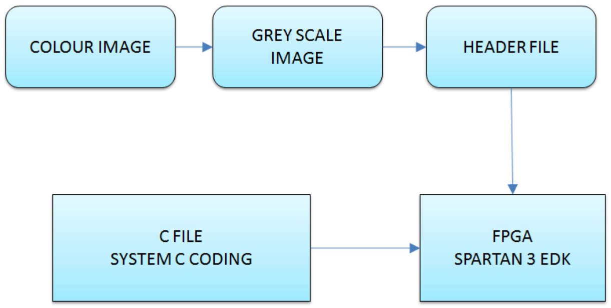


Figure 3: Over all process of the system

The background subtraction is done between these images and a resultant image is formed .The resultant image is known as delta frame. This will eliminate the background and brings the object into focus, giving as information about shape and size. This also reduces the pixels that the arrangement has to process. After delta frame formation, in order to further enhance this, a threshold is applied. Here we fix a threshold value; above that value an object is detected; otherwise it will be the background. The object pixel will give a value 1 and background pixel give a value of zero after applying threshold value. The Initial value of the threshold is set by median or mean value. This is justified when an object pixel is brighter than the background.

The edge detection highlighting the edges of the image and the other parts becomes black. The points at which image brightness changes abruptly are typically termed as edges. Edge detection reduces the amount of data processed, it also filters out non relevant data in order to preserve structural properties of the data. The main application of this is to find out the information about shape and reflectance or transmission of an image. The last step of the process is applying a median filtering method; it to reduce the noise in the image; It will preserve useful details in the image. In this we are considering each pixel in the image and decide whether to represent it as its surroundings, and then replace the pixel value of the median value of surrounding pixel value. The median value is computed by sorting all pixel value from the neighborhood in numerical ascending order and replacing the pixel with middle pixel value. The main advantage of median filtering is more robust than mean, So single unrepresentative pixel in a neighborhood will not affect median value significantly, also better in preserving sharp edges.

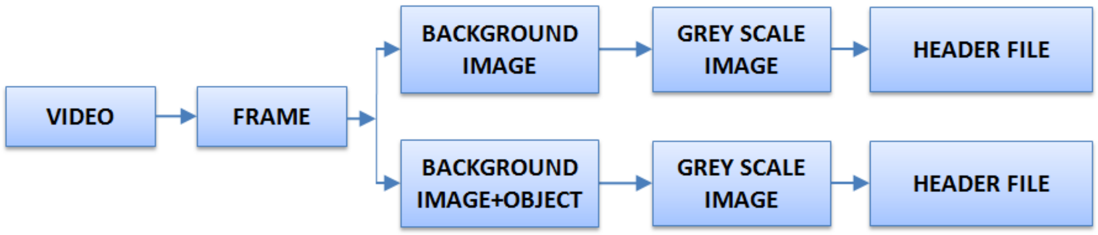
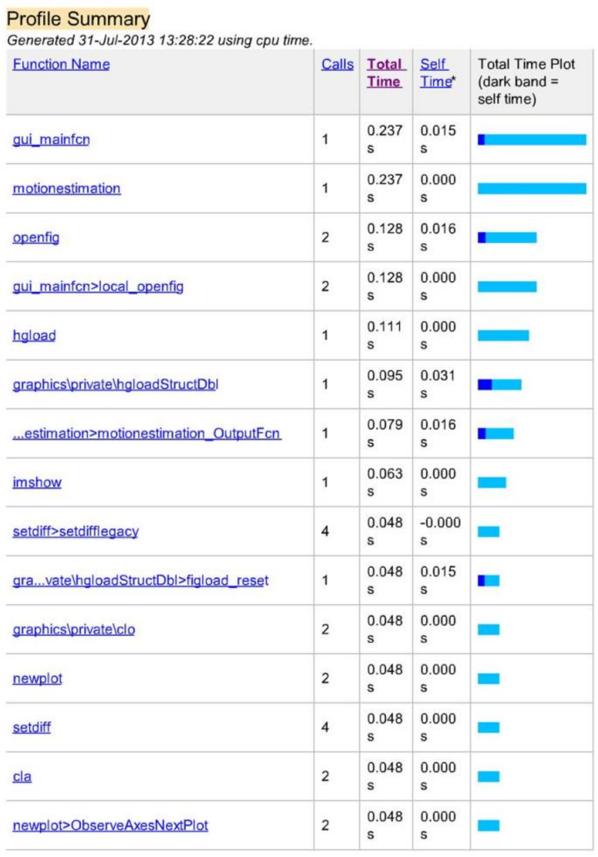


Figure 4: Conversion of images to header file

IV. RESULT AND DISCUSSION

This work is implemented on MATLABR2010a and the Parallel processor Xilinx FPGA Spartan III. The processing time obtained when implemented on MATLAB is 0.237 sec while the processing time obtained from Xilinx FPGA Spartan III is 13.79ns(72.495MHz). So this work optimize the processing time of optical flow detection in a video sequence.

A. *Matlab Simulation Results*



Profile Summary

Generated 31-Jul-2013 13:28:22 using cpu time.

Function Name	Calls	Total Time	Self Time*	Total Time Plot (dark band = self time)
gui_mainfcn	1	0.237 s	0.015 s	
motionestimation	1	0.237 s	0.000 s	
openfig	2	0.128 s	0.016 s	
gui_mainfcn>local_openfig	2	0.128 s	0.000 s	
hgload	1	0.111 s	0.000 s	
graphics\private\hgloadStructDbl	1	0.095 s	0.031 s	
...estimation>motionestimation_OutputFcn	1	0.079 s	0.016 s	
imshow	1	0.063 s	0.000 s	
setdiff>setdifflegacy	4	0.048 s	-0.000 s	
gra...vate\hgloadStructDbl>figload_reset	1	0.048 s	0.015 s	
graphics\private\clo	2	0.048 s	0.000 s	
newplot	2	0.048 s	0.000 s	
setdiff	4	0.048 s	0.000 s	
cla	2	0.048 s	0.000 s	
newplot>ObserveAxesNextPlot	2	0.048 s	0.000 s	

Figure 5 :Profile Summary of MATLAB Simulation

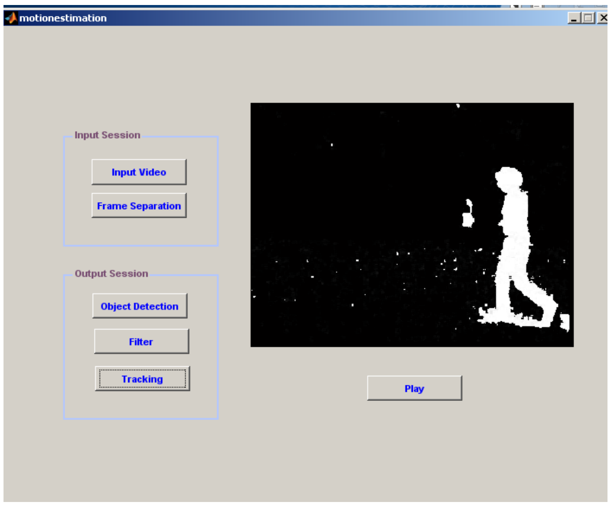
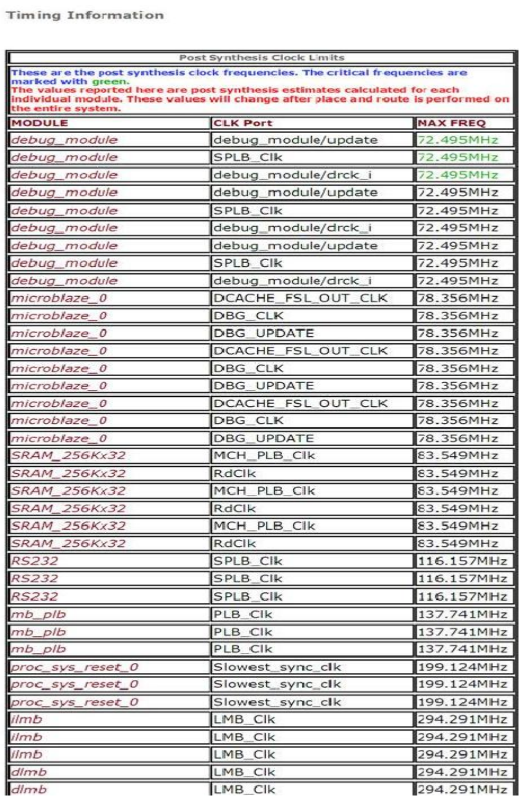


Figure 6:Object tracking and filtering

B. FPGA Implementation Results

Motion analysis is an important technology which modem bio-mechanics combines with computer vision and has been widely used in intelligent control, human computer interaction, motion analysis and virtual reality and other fields. The moving body detection is the most important part of the human body motion analysis, the purpose is to detect the moving human body from the background image in video sequences, and for the follow-up treatment such as the target classification, the human body tracking and behavior understanding, its effective detection plays a very important role. This design implementation required Xilinx Platform Studio (XPS) EDK 10.1 software platform along with MATLAB R2010a & Visual Basic Studio 6 to display images on computer screen. The conversion of true color image into gray scale image as well as resizing of image into (128 * 128) format was carried out using MATLAB R2010a Image Processing Toolbox. While coding of our design which include Background subtraction, Delta frame creation, thresholding, edge detection, filtering was carried out using Impulse C Language in XPS EDK 10.1. For a comparison between a parallel processor and serial processor the work is implemented on MATLABR2010a and the Parallel processor Xilinx FPGA Spartan III. The processing time obtained when implemented the work on MATLAB, is 0.237 sec while the processing time obtained from Xilinx FPGA Spartan III is 13.79ns. So this work optimize the processing time for optical flow motion detection. From the device utilization summery, utilization of 4 input LUTs is 77.7%.



Timing Information

Post Synthesis Clock Limits

These are the post synthesis clock frequencies. The critical frequencies are marked with green.
The values reported here are post synthesis estimates calculated for each individual module. These values will change after place and route is performed on the entire system.

MODULE	CLK Port	MAX FREQ
debug_module	debug_module/update	72.495MHz
debug_module	SPLB_Clk	72.495MHz
debug_module	debug_module/drck_i	72.495MHz
debug_module	debug_module/update	72.495MHz
debug_module	SPLB_Clk	72.495MHz
debug_module	debug_module/drck_i	72.495MHz
debug_module	debug_module/update	72.495MHz
debug_module	SPLB_Clk	72.495MHz
debug_module	debug_module/drck_i	72.495MHz
microblaze_0	DCACHE_FSL_OUT_CLK	78.356MHz
microblaze_0	DBG_CLK	78.356MHz
microblaze_0	DBG_UPDATE	78.356MHz
microblaze_0	DCACHE_FSL_OUT_CLK	78.356MHz
microblaze_0	DBG_CLK	78.356MHz
microblaze_0	DBG_UPDATE	78.356MHz
microblaze_0	DCACHE_FSL_OUT_CLK	78.356MHz
microblaze_0	DBG_CLK	78.356MHz
microblaze_0	DBG_UPDATE	78.356MHz
SRAM_256Kx32	MCH_PLB_Clk	83.549MHz
SRAM_256Kx32	RdClk	83.549MHz
SRAM_256Kx32	MCH_PLB_Clk	83.549MHz
SRAM_256Kx32	RdClk	83.549MHz
SRAM_256Kx32	MCH_PLB_Clk	83.549MHz
SRAM_256Kx32	RdClk	83.549MHz
RS232	SPLB_Clk	116.157MHz
RS232	SPLB_Clk	116.157MHz
RS232	SPLB_Clk	116.157MHz
mb_plb	PLB_Clk	137.741MHz
mb_plb	PLB_Clk	137.741MHz
mb_plb	PLB_Clk	137.741MHz
proc_sys_reset_0	Slowest_sync_clk	199.124MHz
proc_sys_reset_0	Slowest_sync_clk	199.124MHz
proc_sys_reset_0	Slowest_sync_clk	199.124MHz
ilmb	LMB_Clk	294.291MHz
ilmb	LMB_Clk	294.291MHz
ilmb	LMB_Clk	294.291MHz
dlmb	LMB_Clk	294.291MHz
dlmb	LMB_Clk	294.291MHz

Figure 7: Timing information and EDK synthesis report

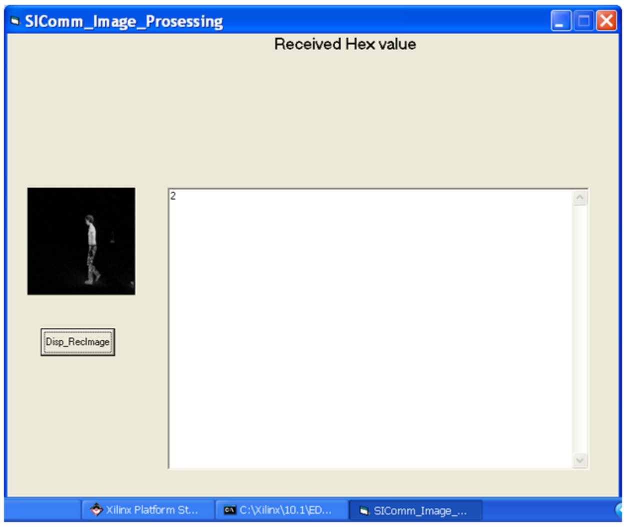


Figure 8: Simulation result

ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 1, January 2014

TABLE I. DEVICE UTILIZATION SUMMARY

Elements in FPGA	**Amount Utilized**	**Amount Available**	**% Utilization**
Number of Slices	1850	1920	96%
Number of Slice Flip Flop	2109	3840	54.92%
Number of 4 input LUTs	2985	3840	77.7%
Number of GCLK	3	8	37.5%
Number of BRAMs	4	12	33%
Number of MULTI 18x18s	3	12	25%
Number of DCMS	1	4	25%

V. CONCLUSION

In this research, a real-time and accurate method for detecting moving object is proposed, based on background subtraction. It uses a pixel network as the core of the architecture to implement segmentation with some degree of parallelism. An efficient colour segmentation algorithm for colour images are implemented using background subtraction. Experimental results show that our algorithm can produce good results. The traditional approach of segmentation using region growing method yields fruitful results only when the complexity is less. At last combine contour projection analysis with shape analysis to remove the shadow effect. Experiments show that the algorithm is fast and simple, able to detect moving object better and it has a broad applicability.

REFERENCES

[1] Weiqiang Wang, Jie Yang, Wen Gao ,"Background and Segmenting Moving Objects from Compressed Video",IEEE transactions on circuits and systems for video technology,vol. 18, NO. 5, MAY 2008.

[2] Du-Ming Tsai , Shia-Chih Lai ,"Independent Component Analysis-Based Background Subtraction for Indoor Surveillance", IEEE Transactions On Image Processing, Vol. 18,No. 1, January 2009

[3] Javier Diaz, Eduardo Ros, Francisco Pelayo, Eva M. Ortigosa, Sonia Mota ,FPGA-Based Real-Time Optical-Flow System", IEEE Transactions On Circuits And Systems For Video Technology, Vol. 16, No. 2, February 2006

[4] Yasuyuki Matsushi,Eyal Ofek, Weina Ge, Xiaoou Tang, Heung-Yeung Shum ,"Full- Frame Video Stabilization with Motion Inpainting", IEEE Transactions On Pattern Analysis And Machine Intelligence, Vol. 28, No. 7, July 2006

[5] Francisco Barranco, Javier Diaz, Eduardo Ros, and Begona del Pino, "Visual System Based on Arti_cial Retina for Motion Detection ", IEEE Transactions On Systems, Man,And Cybernetics|Part B: Cybernetics, Vol. 39, No. 3, June 2009