Computing Displacement of Moving Object in a Real Time Video using EDK

Kota Solomon Raju[1], Manipati Rajesham[1] Gargi Baruah[2], Palash Phukan[2]

DSG,Council of Scientific and Industrial research (CSIR) -Central Electronics Engineering Research Institute (CEERI) CSIR-CEERI, Pilani-333031[1], Dept of Electronics & Communication Engineering, Tezpur University Tezpur-784028[2]

solomon@ceeri.ernet.in, rajesh.manipati@gmail.com, gargi369@gmail.com, palash.phukan@gmail.com

ABSTRACT

This paper concentrates on how to compute displacement of a real time moving object in EDK platform. It is implemented in Spartan 6 board. The Xilinx Spartan family has the ability for partial reconfigurability and thus can be use in real time video processing.

Keywords: Xilinx, EDK, XPS, Bhattacharya Coefficient, pixel values, tracking, real time video.

1. INTRODUCTION:

The efficient tracking of visual features in complex environment is a challenging task. Real time applications such as surveillance and monitoring [1], perceptual user interfaces [1], smart rooms [1] and video compression [1] all require the ability to track moving objects. Tracking algorithms [1] can be classified into two major groups, namely state-space approach and kernel based approach. State-space approaches are based largely on probability, stochastic processes and estimation theory, which, when combined with systems theory and combinatorial optimization, lead to a plethora of approaches, such as Kalman filter, Extended Kalman Filter (EKF) [2], Unscented Kalman Filter (UKF) [1], Particle Filter (PF) [1].

A number of attempts have been made to achieve robust, high performance target tracking [8]. But for every algorithm to work well, the calculation of a weighted histogram is very important.

Tracking objects also require complex computational processing throughput which seems challenging in terms of processing as well as cost. An FPGA can give high efficiency, flexibility, greater processing ability and can reduce costs with various verification techniques such as behavioral simulation and post route simulation. Also, Xilinx Embedded Development Kit (EDK) tools can make it possible to implement a complete digital system on a single FPGA using hardware/ software design methods.. A developer could use one chip for different tasks and switch between them during runtime. The Xilinx Spartan family provides for this partial reconfigurability.

This project aims at designing a real time video capture and computes the displacement of the target object in various frames using a Spartan 6 Industrial Video Processing Kit (Spartan 6 LX150T FPGA). This paper focuses on how we have developed the weighted histogram and used Bhattacharya Coefficient to compute displacement for the real time video involving the target and the candidate object.

This paper is divided into 2 sections. The first section describes the histogram calculation that is used for tracking and the second section describes the hardware implementation of it in Spartan 6 LX150T.

2. DISPLACEMENT CALCULATION

2.1 Target representation

A feature space is first chosen to characterize the target represented by its pdf ‘q’ in the feature space centred at a spatial location 0. In the subsequent frame, a target candidate is defined at a location ‘y’ with pdf p(y). Thus,

From the literature [6] **object model** pdf is given by

$$Qu=C\sum_{i=1}^{n} C|k(||xi||^2)\partial(b(xi)-u") \quad (1)$$

Assuming size of the model to be normalized with kernel radius h=1. Here C is the normalization constant.

$$C=[\sum_{i=1}^{n} k(||xi||)^2]^{-1} \quad (2)$$

Kernel profile k weights contribution by distance to centroid and ∂ is the Kronecker delta function

$$\partial(a) = \begin{cases} 1 & if\ a=0 \\ 0 & otherwise \end{cases} \text{ i.e. } (k||xi||)^2 = Qu \text{ only if } b(xi)=u. \quad (3)$$

Target model for target centred at y and yi=1,2..nh are the pixel locations, Ch is the normalization constant.

$$p(y)=Ch\sum_{i=1}^{nh} \left|(| \left|\frac{y-yi}{h}\right| |^2\right) \partial(b(yi)-u) \quad (4)$$

From [2] the Bhattacharyya coefficient is given by

$$\rho(\vec{y}) = \rho[\vec{p}(y),\vec{q}] = \sum_{u=1}^{m}\sqrt{p_u(y),q_u} \quad (5)$$

Now the similarity function defines a distance among target model and candidates and
We define the distance between two discrete distributions as

$$d(y)=\sqrt{1-\rho[\vec{p}(y),\vec{q}]} \quad (6)$$

In our implementations we have used RGB color space. Now the choice of histogram function is also very important. If a bin is used for all possible colours for a 24 bit frame, then there will be 256*256*256= 16 million bins. So it is set up to 3 colours, 16 bin per model i.e. the feature space is quantized to 16*16*16=4096 bin values.

3. EMBEDDED DEVELOPMENT KIT

The Embedded Development Kit is the Xilinx software suite for designing complete embedded programmable systems. It enables the integration of both hardware and software components of an embedded system. In it Xilinx Platform Studio (XPS) is a graphical Integrated Design Environment (IDE) that incorporates all the Embedded System Tools for seamless creation of hardware and software components and, optionally, a verification component.

3.1. SOFTWARE FLOW COMPILATION.

The flowchart in Fig 1 is programmed in C using EDK. The Input files → *.c, *.h, libc.a, libXil.a, libm.a. The EDK goes through 4 stages.

- Pre-processor: Replaces all macros with definitions as defined in the .c or .h files
- Machine-specific and language-specific compiler: Compiles C/C++ code
- Assembler: converts machine language and generates the object file
- Linker: Links all the object files using user-defined or default linker script.

The output file is then obtained as **executable.elf.**

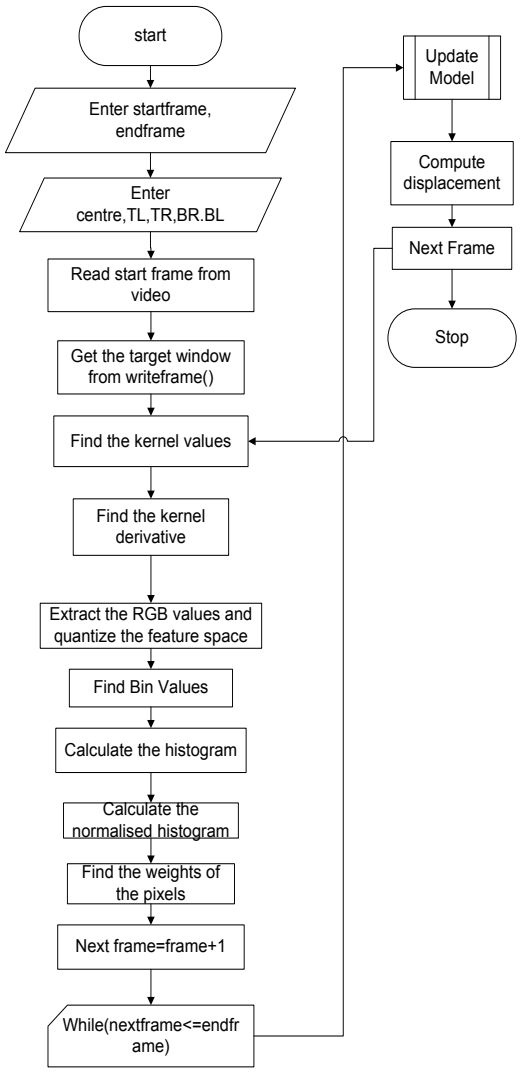

Fig 1. Flowchart for displacement calculation

4. HARDWARE DESCRIPTION

The image sensor video input source enters the Camera Input PCORE[9]. This PCORE decodes the BT656 codes to generate synchronization signals and formats the video as an XSVI bus interface. The Video Detect PCORE does not alter the video, but monitors the VSYNC and ACTIVE_VIDEO signals to determine the dimensions of the active video streaming through the FPGA. It also generates Video DMA compatible bus interface used to write video data to external memory. The Video DMA PCOREs, in collaboration with the Video Frame Buffer Controller (VFBC)[9].

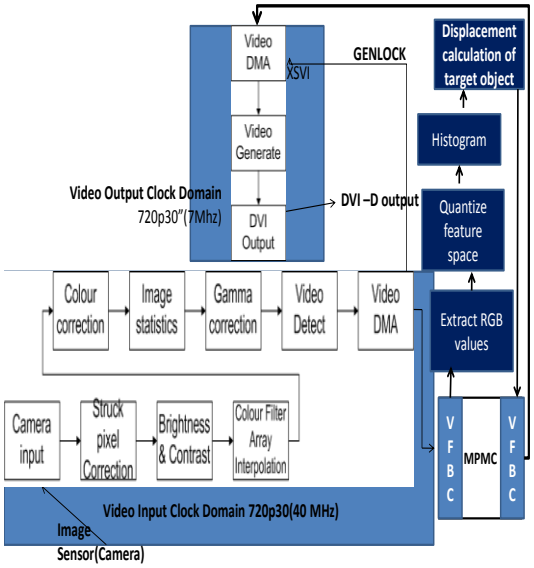

Fig: 2 Camera Frame Buffer – Video Pipeline

Interfaces on the Multi-Port Memory Controller (MPMC), perform the actual transfers to/from external memory. These cores are extremely flexible and are configured via the Micro Blaze processor. The GENLOCK port indicates where the first Video DMA has written the incoming frames. The second Video DMA reads video frames from memory based on the GENLOCK [9] information. After that the histogram calculation IP gets the pixel data and takes the RGB values, each of 8 bit. It takes the precalculated kernel values and finds out the histogram of the target and the candidate model and them computes the displacement of the target object in each frame.

Since the output frame rate is higher than the input frame rate, frames are duplicated when necessary. The Video Generate PCORE, under control of the MicroBlaze, generates video timing for the output. It also generates a Video DMA compatible bus interface used to read video data from external memory.
The DVI Output PCORE takes an XSVI bus interface as input and optionally drives the pins of the DVI output interface. These output to the FMC connector will only be driven once the FMCIMAGEOV module has properly been identified.

The video capture is at 1280x720P @ 30Hz and video playback at 1280x720P @ 60Hz. These resolutions are configured by the embedded processor (MicroBlaze) and can be modified to support other resolutions (limited by the image sensor used).

Thus the software and the hardware part is merged using EDK, and bit stream is generated which is dumped into the FPGA. Fig 3 shows it.

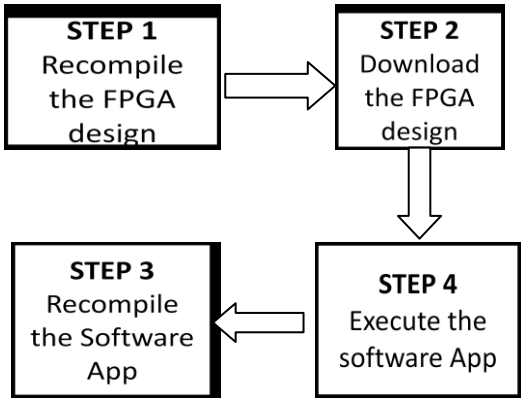

Fig 3: XPS Development Flow

5. RESULTS

The data that was obtained from the output in HyperTerminal window using the EDK tool were used in Matlab 7.8.0 (R2009a) after which the following graphs were obtained.

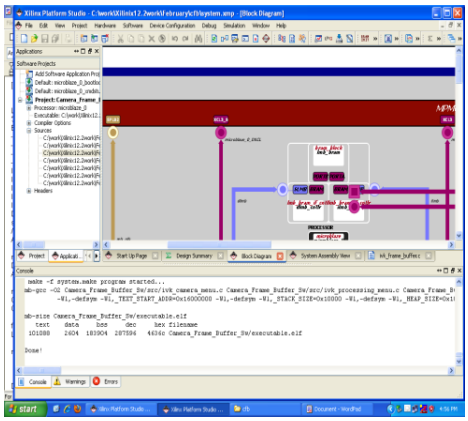

Fig 3: The elf file (Camera_Frame_Buffer_Sw/executable.elf)

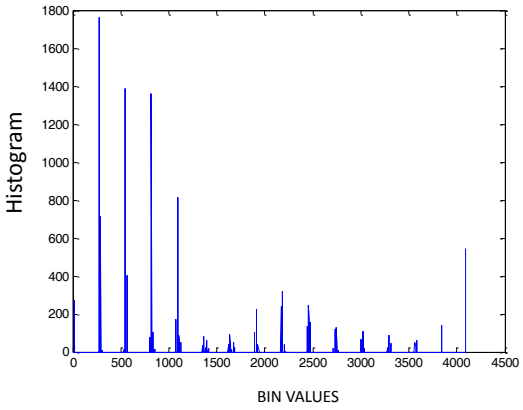

Fig 5: Histogram of a 160 X 80 target window of 1280X720 frame(1st frame)

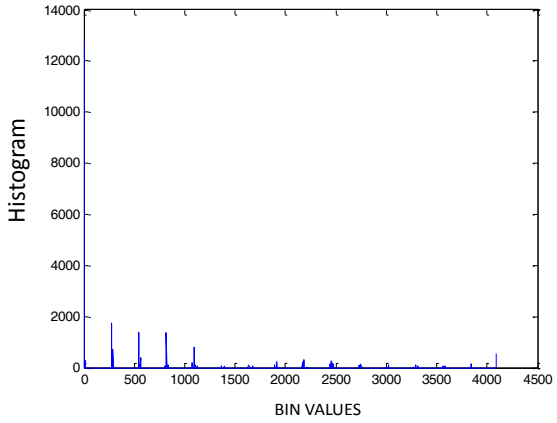

Fig 6: Histogram of a 160 X 80 target window of 1280X720 frame(30th frame)

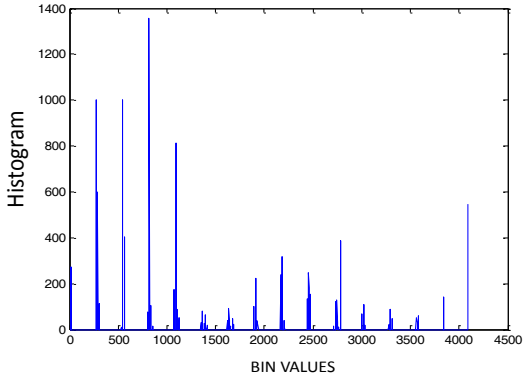

Fig 7: Histogram of a 160 X 80 target window of 1280X720 frame(50th frame)

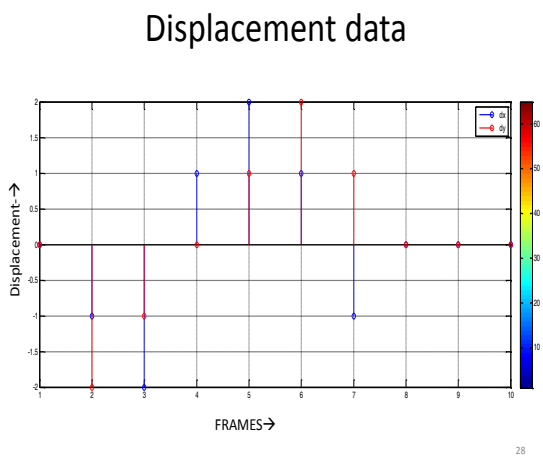

Fig 8: Displacement of the target object in 50 consecutive frames

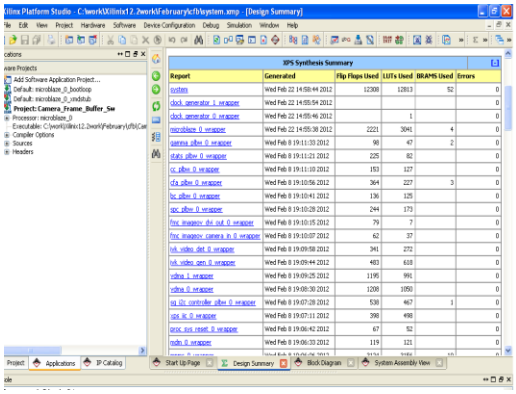

Fig 9: XPS Synthesis summary

1. developing new image processing algorithms

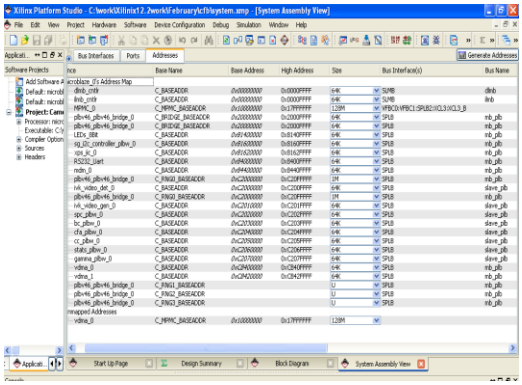

Fig 10: IP Addresses

6. CONCLUSION AND FURTHER WORK

In this paper we have explored the use of variable kernels to enhance a weighted histogram and then compute the displacement of an object in the video frames which can be used for various tracking and other video processing algorithms. The main advantage is that a system has been developed which is not only accurate but its computation is very high compared to other software platforms since in EDK 100% bit stream is generated.

Future work is currently underway to extend our testbed platform for tracking of objects in real time b

2. complementing our motion-tracking algorithm by adding further improved calculations

ACKNOWLEDGMENT

The work reported in this paper is a part funded by CSIR-Supra Institutional Project.
We would also like to thank Mr. Manoj Pandey and Mr. Vaibhawa Mishra for their help and support.

REFERENCES

[1] Alper Yilmaz, Omar Javed, Mubarak Shah, "**Object Tracking- A Survey",** ACM Computing Surveys, vol. 38, No. 4, Article 13, pp.145-190, December 2006

[2] Welch & Bishop**, "An Introduction to the Kalman Filter"** UNC-Chapel Hill, TR 95-041,vol. 40, No. 7,pp 234-246, July 24, 2006

[3] M. Sanjeev Arulampalam, Simon Maskell, Neil Gordon, and Tim,"**A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian**

Tracking" IEEE transactions on signal processing, vol. 50, No. 2, pp 55-70, February 2002.

[4] K. Fukunaga, L.D. Hostetler, "**The Estimation of the Gradient of a Density Function, with applications in Pattern Recognition**", IEEE Transactions on Information Theory, vol. 21, pp. 32-40. January 1975.

[5] D. Comaniciu, P. Meer, "**Mean shift analysis and applications**," In IEEE Int. Conf. on Computer Vision, vol. 2, pp. 1197-1203, March 1999

[6]D. Comaniciu, V. Ramesh, P. Meer, "**Kernel-based object tracking**," IEEE Trans. On Pattern Analysis and Machine Intelligence, pp. 564-575, Dec 2003.

[7]D. Comaniciu and P. Meer, "**Mean shift: A robust approach toward feature space analysis**," IEEE Trans. Pattern Anal. Machine Intel., vol. 24, no. 5, pp. 603–619,Dec 2002.**.**

[8] Benjamin Gorry, Zezhi Chen, Kevin Hammond, Andy Wallace, and Greg Michaelson, "**Using Mean-Shift Tracking Algorithms for Real-Time Tracking of Moving Images on an Autonomous Vehicle Test bed Platform**" ,International Conference on Intelligent Robotics and Manufacturing Automation, Venice, Italy, 2007 , World Academy of Science, Engineering and Technology (PWASET) , pp-45-48, November 23-25, 2007

[9] Spartan-6 Industrial Video Processing Kit – EDK Reference Design Tutorial