Far East Journal of Electronics and Communications

Published Online: January 2016
http://dx.doi.org/10.17654/EC016010001
Volume 16, Number 1, 2016, Pages 1-13 **ISSN: 0973-7006**

OBJECT TRACKING ALGORITHM IMPLEMENTATION FOR SECURITY APPLICATIONS

S. Sivanantham, N. Nitin Paul and R. Suraj Iyer

School of Electronics Engineering
VIT University
Vellore-632014, Tamilnadu, India
e-mail: ssivanantham@vit.ac.in

Abstract

Object tracking is the cornerstone for all machine vision applications. The major challenge in object tracking applications has always been speed, accuracy and ability to process heavy data flow. By using field programmable gate arrays in implementing the algorithm, we can overcome the challenges primarily by using its ability of reconfigurable design and parallel architecture. The basic building blocks of this project are pre-processing, image splitting and detection followed by tracking and display. The pre-processing block uses scaling of image through bilinear interpolation so as to make it more applicable for computation, the image splitting and detection block do the splitting of image after compression, which is achieved by breaking the image into submatrices. Through consistent comparison and analysis of the stream of images with a default image, the object is tracked.

I. Introduction

Object tracking is the key for many machine vision applications which

Received: May 9, 2015; Revised: June 26, 2015; Accepted: July 31, 2015
Keywords and phrases: object tracking, scaling, parallelism, FPGA.
Communicated by S. Saravanan

uses encompassing various sectors of engineering from military to gaming and is generally the prerequisite for object recognition. It involves tracking of the object in real time video streaming from a camera so as to detect and track the object which has been introduced in the background. Object tracking is of key importance in many surveillance and security systems, traffic control, medical imaging. Object tracking plays a crucial part in development of artificial intelligence in robots, it is also an attempt to make computer vision applications as intelligent as human vision. A tracking algorithm is based on fractional differential gradient to extract three-dimensional boundary surface within biomedical image [13]. A novel strategy for modelling the motion of local patches for single object tracking that can be seamlessly applied to most part-based trackers is described in [1]. In this approach, adaptive local movement modelling (ALMM) method is able to model the local spatial distribution of the image patches defining the object to track and the reliability of each image patch.

The number of approaches has been proposed for object tracking in last few years towards increasing performance of computer vision systems. A tracker which is based on structural information captured in pixels is proposed in [5], in which a tracker distinguishes between the target and the background easily. This target-background posterior estimate is further used to track the object.

The most common type of the basic objective of object tracking is to associate target objects in successive frames of data this is done by setting the frame speed comparable to the computational speed so that no lag occurs in tracking which is highly avoidable. Many algorithms are proposed for object tracking out of which all involve heavy computation to be done the most formally used algorithm is the blob detection that involves detection of blobs or regions that differ in certain image properties and parameters. There is also a mean shift algorithm approach which employs a mathematical function of Bhattacharya coefficient which relies on calculating the minimum distance between pixels is used to detect the object. Normally, object tracking is realized with the help of Matlab, but implementation in application specific integrated circuit (ASIC) is tough because of large number of operations are operated in adder and subtractors rather than

multipliers and dividers. When implementing on field programmable gate array (FPGA), the image is considered a matrix and the object is a cofactor of it [2].

In this paper, we have used background subtraction algorithm which relies on the basic principle of subtracting the foreground from the background and then differentiate the object. Because of its robustness, background subtraction algorithm is used for object tracking in streaming videos [3]. All of these computations can be time consuming process but the real challenge lies in the fact that when object tracking is to be done in real time the processing delay cannot be long but has to be on time and should provide satisfactory results. This has been the major reason for implementation of many image processing applications in FPGA, solely because its parallel architecture it can process heavy data flow in real time quite efficiently with an added advantage of reconfigurable design and low power requirements [4]. The parallel architecture of FPGA has been exploited in this paper, which actually increases the processing speed of object detection and helps in better tracking in the overall stream of images. In this paper, we have used the Altera cyclone IVE EP4CE115F29C7 FPGA device with a Nios II processor.

II. Object Tracking Algorithm

The algorithm comprises of three major design blocks that is pre-processing of video received from camera, processing of the image data and then post processing. In layman terms, the basic logic for the algorithm can be termed to be the difference in the image data between a background matrix and the current frame matrix, respectively. In order to carry this out in real time and by keeping in mind the time constraints, parallelizing the computational part of algorithm helps in speeding up the process and produces favourable results. By compromising on the image quality by using scaling instead of compression, we speed up the process even more. Additional modules in the form of overlay generator and frame joiner are used in the post processing section to join back the submatrices and upscale is used to scale the image back to original size.

A video decoder is used to take the video input from the camera and read frame by frame into current random access memory (RAM) it has to be remembered that the subsampling of video frame can also be done here and the size of the image is scaled down in size so as to make it more applicable for computation and also makes the processing easier. The pre-processing phase is accelerated by using the built in operation of scaling in Altera the scaling is done through bilinear interpolation the downscaled image is fed to either the background or current frame RAM which are then split into twelve 20×20 submatrices which are then evaluated to find the object. Bilinear interpolation is good for resizing of image with a drawback of softening of details and jaggedness. The result from the submatrices is then joined by frame rebuilder as per their order in the matrix and then the image is up scaled to that of the video frame size and then passed to the overlay generator which generates the overlay and boundary for the identification of the object in the current stream of video frames by highlighting the given object based on a threshold value, all this computation is done in matrix form of image data after which the image is transferred to the video graphics array (VGA) monitor for display.

A. Image acquisition

The image acquisition is the first step in which the image from the camera is fed to a video decoder which scales down the image to 80×60 and decodes the given image data to matrix form which is used in the succeeding steps for computations. The scaling is done using bilinear interpolation even though there are other operators, bilinear interpolation results in smoother edges when compared to other methods [6]. The image can either be scaled or compressed depending on the designer but reduction in size of the image before actual computation would result in better computational speed [7].

Compression techniques like Haar or any other transform can be used to compress data. But their usage would result in introduction of extra filters in the post processing stage like erosion and dilation filters which would result in extra convolutions [8]. Hence, using inbuilt functions of scaling and subsampling of image greatly reduces the run time thus increasing speed. Also, there is an additional option of varying the type of scaling that can be

done through the Altera mega core intellectual property (IP). The downscaled image is fed into the background and current RAM based on the control input. The scaled matrix data is also fed to the buffer which is sent to the overlay generator after processing for generating the boundary over the buffered matrix so as to track the object in the stream. Thus, the object is tracked, hence the image is properly acquiesced and fed into matrix form for further processing and video buffer.

B. Background subtraction

The downscaled image from the scalar is fed into the processing stage it comprises of two random access memories. The background RAM is of type first in first out (FIFO), so as to enable for computation with current RAM which is of type static the subtractor module present in this stage calls the split 20×20 matrices each of current and background memories and send them to next stage this causes parallelism characteristic of FPGA. The background subtraction algorithm is then implemented step by step over the twelve 20×20 matrices in parallel architecture as shown in Figure 1.

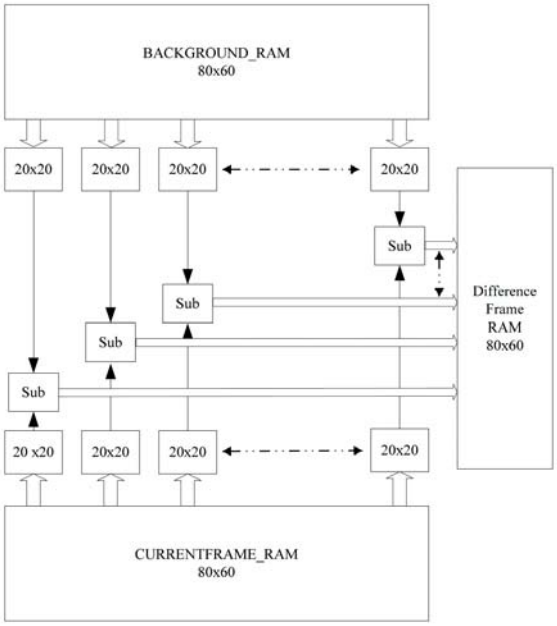

Figure 1. Parallel processing blocks.

Implementation of subtraction part in parallel form speeds up the process and can also be termed as the block mode analysis in which the whole computation is broken down to respective blocks and works in parallel mode since all the values of the matrices can be accessed at the same time thus starting all the computations at the same time [9]. Hence, speeding up the process when compared to subtraction of entire matrix at a time RAM depth is 4800 bytes for each of background and current memories and both of the memories are addressed through dual port.

C. Post processing

The post processing stage uses three modules of the form frame joiner overlay generator and displaying the output on VGA monitor the frame joiner does the basic operation of joining the twelve matrices in a predefined order so as to form the entire frame together the joined frame would be of size 80×60 this frame is then merged with that of overlay generator with input frame of the video. The up scaled frame of 640×460 consists of the output from the background subtraction module which consists of the cofactor matrix of the object from the entire matrix of the background this cofactor matrix would consist the data pixels of the object and the rest pixels as they are similar to that of background would be zero and will be black by providing an optional threshold value the edges of the object can be detected [10].

D. Flowchart

The flowchart for the given algorithm is given in Figure 2, the video frames are obtained from the camera module at a size of 640×480 and are subsampled to 80×60 depending upon the global input RST when RST is enabled the first frame of the video is passed onto the background RAM of type FIFO which would latch onto the RAM for further computation until the RST input is changed once the input is disabled the current frame RAM would read the frames of the video one by one comparing them with the background RAM frame, In the next stage, twelve subtractors are instantiated in parallel each with a capability of subtracting a 400 byte RAM and storing it in a RAM of the same size this is the area where parallel architecture was

used and the advantage of FPGA was exploited [11]. Doing the subtraction in parallel architecture helps load the entire pixel values at once in a single cycle rather than a row by row or a column by column subtraction by taking the entire matrix at a time.

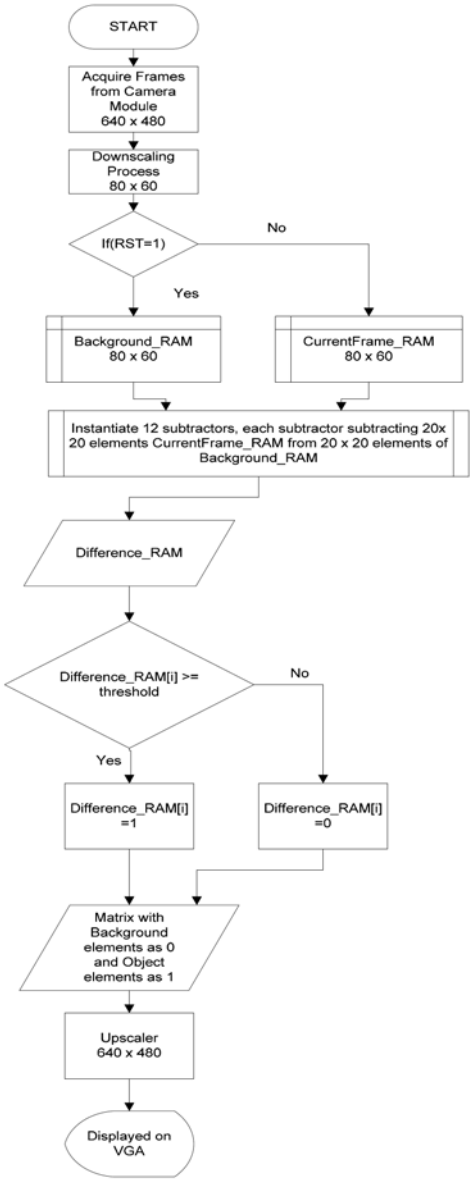

Figure 2. Flowchart of algorithm.

Each element of the difference RAM is then compared with the specified threshold value which is passed as a parameter and differentiated between foreground and background. The value of threshold is arrived at after many iterations and is set to a optimized value so as to detect valid edges of the object in the difference RAM matrix, so the resultant matrix from this stage would be a cofactor of the real image frame and would consist of black region engulfing the entire object with the unchanged pixel values of the object which would form the boundary for the object based on the threshold value, the pixels are classified as part of background or foreground depending upon the threshold value if the value is greater than threshold the pixel is assigned a value of one equivalent to 255 or white and is a part of the foreground else it is assigned a value of zero equivalent to zero or black.

Sample code

```
input video_stream;
input reset;
output video_out;
parameter threshold;
interface camera to board
                    acquire frames( );

if(reset)
        Background_ram<=Current_Frame;
else
     Currentframe_ram<=Current_Frame;

always@(Current_Frame)
     Difference_ram<=Background_ram-
currentframe_ram;

if(Difference_ram[i] ≥threshold_value)
   Difference_ram<=foreground;
else
Difference_ram<=background;

Object_Boundary( )
object_boundary=create boundarybox(foreground)

video_out=overlay(Current_Frame,Boundary);

display(video_out)
```

Figure 3. Sample code for algorithm.

III. Hardware Implementation

The implementation is proposed on Altera DE2 board with cyclone IVE EP4CE115F29C7 FPGA device the RST input is used as a button to activate and load the current RAM to background RAM once the button is pressed back the device loads the current frames into current RAM. The background RAM is designated of type FIFO and the current RAM is designed to be a static RAM with dual port the difference is stored in difference RAM which is of type FIFO so the difference is stored on a first in first out basis difference from the background and current RAM and stored in the difference RAM all the memories are initiated on the board with the camera on a daughter board with GPIO interface connected to Altera, the corresponding frame output onto a VGA monitor on another daughter board.

A. System architectural overview

The image capture is performed using a video camera and a video decoder which decodes the video stream to matrix form of size 640×480 elements for further computations. The effective image size is therefore reduced to 80 by 60 pixels. The received image is changed from red green blue (RGB) to luminance blue difference chromaticity red difference chromaticity (YCbCr). Out of which the first frame which considers the grayscale data is extracted to matrix normally the RGB to YCbCr involves inverse matrix multiplication which is tedious [12], but usage of this from the same module of the decoder by using mega core IP function in the processing stage two types of parallel architectures can be used that is either temporal or spatial in this implementation we use spatial in which an 80×60 has been broken down to twelve 20×20 are formed and the background subtraction for all the twelve matrices is done concurrently in temporal architecture, pipelined architecture for feeding the data is used. The image matrix is then joined by a frame joiner to form a matrix of size 80×60 this matrix is then subsequently scaled back to its original size by up scaling it using the upscale module in the overlay module a box gives the position of the detected object from the matrix that has been obtained from the difference RAM.

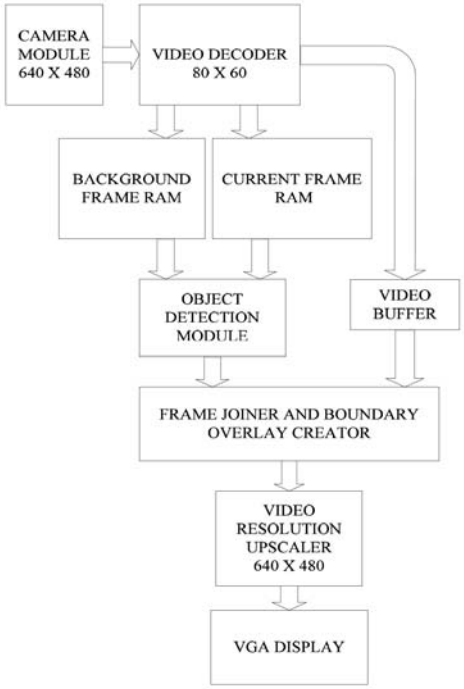

Figure 4. Architectural overview.

IV. Simulation Results

The simulation has been done modulewise on Matlab with background subtraction algorithm as the basis for calculating the difference of frames. The streaming video input has been taken and was processed by capturing the first frame as the background and the succeeding frames are compared successively. The test video used is a 30 second quarter video graphics array (QVGA) of codec cinepa. The width and height of the frame are 320×240 with a data rate of 2800kbps and frame rate of 15 frames per second. The streaming video frame is indicated in Figure 5, the background frame is the first frame of the video which is the frame that is shown in Figure 6, the background subtraction algorithm is carried out from which the current streaming frame is continuously subtracted from background frame to generate the difference frame which is shown in Figure 7 the difference frame shows all the pixel values that are different frame from the background

frame based on the luminance value which form the objects or foreign objects that have been introduced into the frames. By setting a specified threshold value which was 25 in this case, we could cut down on a significant quantity of noise.

Figure 5. Video frame.

Figure 6. Background frame.

Figure 7. Output frame with objects.

V. Conclusions

In this work, we have proposed an algorithm for object tracking for security applications. The algorithm has background subtraction as its basis followed by continuous difference of the current frame with background frame. Through simulation we have observed that by altering with the threshold value the level to which we can differentiate pixels that are different from the background frame. At lower thresholds, we are able to observe a lot of noise interference so in order to set threshold to optimum value certain number of iterations have to be performed and threshold determined.

Acknowledgement

The authors thank the anonymous referees for their valuable suggestions which let to the improvement of the manuscript.

References

[1] Baochang Zhang, Li Zhigang, A. Perina, A. Del Bue and V. Murino, Adaptive local movement modelling for object tracking, IEEE Winter Conference on Applications of Computer Vision (WACV), 2015, pp. 25-32.

[2] Rahul V. Shah, Amit Jain, Rutul B. Bhatt, Pinal Engineer and Ekata Mehul, Mean shift algorithm in verilog HDL approach, Int. J. Adv. Res. Comput. Comm. Engineer. 1(2) (2012), 181-194.

[3] Su Liu, Alexandros Papakon Stantinous, Hongjun Wang and Deming Chen, Real time object tracking system on FPGAs, Symposium on Application Accelerators in High - Performance Computing (SAAHPC), 2011, pp. 1-7.

[4] Scott Hauck, The roles of FPGAs in reprogrammable systems, Proceedings of the IEEE 86(4) (1998), 615-639.

[5] K. Phalke and R. Hegadi, Pixel based object tracking, 2nd International Conference on Signal Processing and Integrated Networks (SPIN), 2015, pp. 575-578.

[6] John Canny, A computational approach to edge detection, pattern analysis and machine intelligence, IEEE Transactions, 1986, pp. 679-698.

[7] C. T. Johnston, D. G. Bailey and K. T. Gribbon, Optimisation of a colour segmentation and tracking algorithm for real-time FPGA implementation, Proceedings of Image and Vision Computing Conference New Zealand (IVCNZ'05), Dunedin, New Zealand, November 2005, pp. 422-427.

[8] Fatih Porikili, Achieving real time object detection and extreme conditions, Journal of Real-Time Image Processing 1(1) (2006), 33-40,

[9] K. Yamaoka, T. Morimoto, H. Adachi and T. Koide, Image segmentation and pattern matching based FPGA/ASIC implementation architecture of real time object tracking, Design Automation Asia and South Pacific Conference, Jan. 2006, pp. 24-27.

[10] Joan S. Weszka, A survey of threshold selection techniques, Computer Graphics and Image Processing 7 (1978), 259-265.

[11] J. Batlle, J. Martı, P. Ridao and J. Ama, A new FPGA/DSP-based parallel architecture for real-time image processing, Real-Time Imaging 8 (2002), 345-356.

[12] D. Crookes, T. J. Brown, Y. Dong, G. McAleese, P. J. Morrow, D. K. Roantree and I. T. A. Spence, A self-optimising coprocessor model for portable parallel image processing, Second International Euro-Par Conference, Springer, Berlin, Heidelberg, 1996, pp. 213-216.

[13] Ma Yu, Zhang Yanning, Liang Huiling, Xu Shuang and Yu Ting, A tracking algorithm based on fractional differential gradient within 3D image, International Conference on Biomedical Engineering and Biotechnology (ICBEB), 2012, pp. 696-699.