FPGA BASED OBJECT TRACKING SYSTEM

PROJECT APPROVAL

Design of Embedded System Advanced Course-EDA385
Department of Computer Science, Lund University

Submitted By

HARSHAVARDHAN KITTUR – ***aso10hki@student.lu.se***

CHUANHAI BAI-*aso10cba@student.lu.se*

ADVANCED EMBEDDED SYSTEM PROJECT APPROVAL

Abstract

In this project we propose to implement Object Tracking System which uses various video processing algorithms which have to be implemented on Digilent Nexys2 development board using Xilinx EDK. A hardware/software design approach will be used to partition this system blocks into hardware and software domains based on their criticality.

1. INTRODUCTION

Object Tracking is a process of locating the object to associate the target in successive video frame over time and it finds wide scale applications in the field of security and surveillance, video communication, augmented reality, traffic control, medical imaging etc. Object Tracking is a complex process to be implemented in hardware mainly because of the amount of data associated with the video, and hence FPGAs provide a good medium of implementation because of parallelism, low cost and low power.

In this project we investigated the best algorithm which can be implemented on FPGA and choose a optimum algorithm which is used to calculate the centre of gravity (COG) of the object to track it across the frame.

2. ALGORITHIMIC BLOCK DIAGRAM

The following modules construct the Object Tracking System :

1. Grayscale Conversion
2. Delta Frame Generation
3. Thresholding and
4. Object Identification and Tracking.

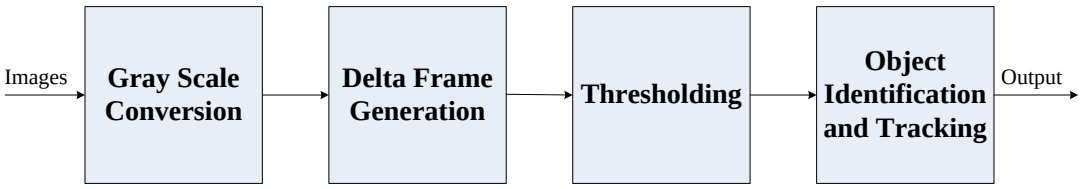

Figure 1: Object Tracking Algorithm Flow

1. Grayscale Conversion

The bitmap files are generated from each frame of demo video in Matlab and then background and object frame are selected. These files are present in RGB format at a resolution of 640x480 pixels. These frames are then converted to grayscale within a range of 0-255. This reduces the coherent effect of the environment and allows us to easily separate the object from the background.

2. Delta Frame Generation

Once the Gray Scale Conversion has been completed, the respective frames are subtracted from one another. The resulting frame is called the Delta Frame. This method of image subtraction eliminates the background and brings the object into focus, giving us information about its shape and size. The Delta frame also reduces the number of pixels that the system will have to process.

3. Thresholding

In order to further enhance the resolution of the delta frame Gray Scale Thresholding is done. Example – Figure 2. The individual pixels in the grayscale image are marked as object pixels if their value is greater than some threshold value (initially set as 40) and as background pixels otherwise. In this case the object pixel is given a value of "1" while a background pixel is given a value of "0."

Figure 2: Gray Level Thresholding

4. Object Identification and Tracking

In most applications, the center of gravity is used for tracking the target as it is a geometric property of any object. The center of gravity is the average location of the weight of an object. It can be used to describe the motion of the object through space in terms of the translation of the point of the object from one place to another

In general, determining the center of gravity is a complicated procedure because the mass may not be uniformly distributed throughout the object. In order to simplify the problem we assume the object is composed of uniform material. An operator scans the entire length of the image frame for the first white pixel. This is a clear indication of the 2D position of the object within that time frame. This is iterative process and it repeated over all the frames.

We can get the center of gravity of the project from each iterative scanning. The updated locations of the pixel points are collected to provide the approximate path taken by the object. The iterative program acquires the frames and plots the individual points to display the object path.

3. DESIGN COMPONENTS

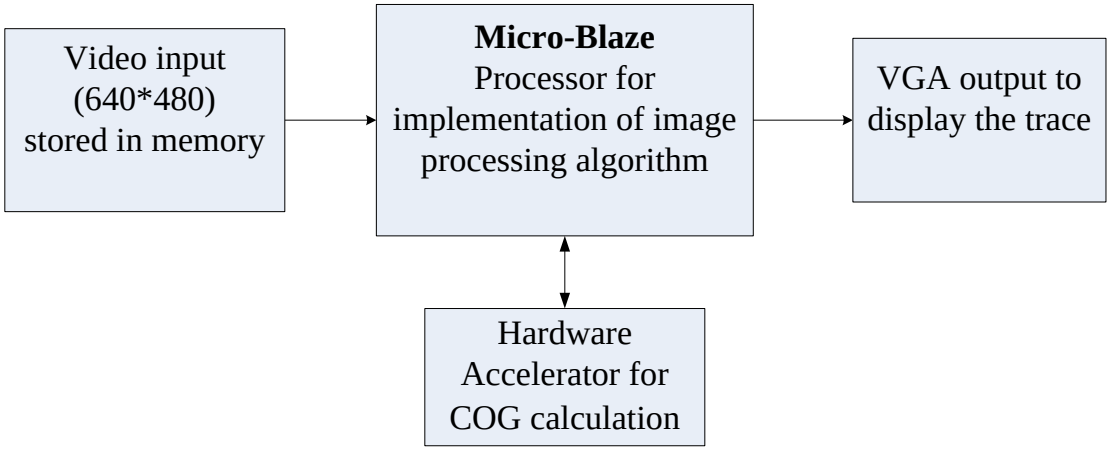

Figure 3: Components for implementation of Object Tracking

Figure 3 shows different components for our implementation of Object Tracking. We use SD card or SDRAM to store the grayscale image of each frame of demo video and utilize Micro-Blaze as processor to implement the image processing algorithms such as delta frame generation and thresholding. We put the calculation of center of gravity into hardware since it is a exhaustive process and needs to run from every row and column of every picture to get the value. Finally, we show the tracing of object in VGA.

4. CONSTRAINTS

Frame buffering requires large amounts of memory. After gray scaling of each frame of demo video in Matlab, each grayscale image is around 300 KB. We need about 50 frames, which includes effective image elements, to do the subsequent processing. That means we need at least 15 MB memory to store these images. FPGAs have very limited amounts of on-chip RAM. The logic blocks themselves can be configured to act like RAM, but this is usually an inefficient use of the logic blocks. Typically some sort of off-chip memory can be used, such as SDRAM (16 MB) or flash memory can we used, but we cannot download data on it together with the FPGA configuration but in separate way.

5. MATLAB BEHAVIOURAL MODEL

A matlab behavioural model was written in order to check the system output at every processing blocks to verify the system functionality. The following screen dumps show the matlab output at various stages.

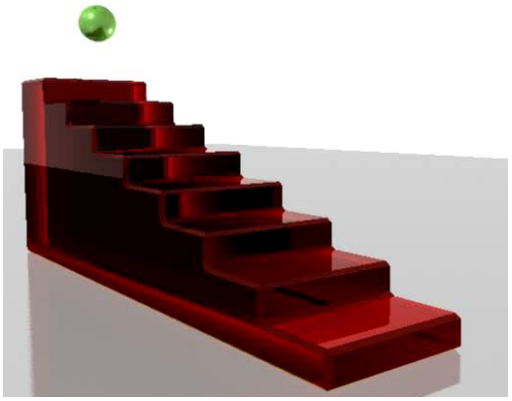

Figure 3: Bitmap of frame captured

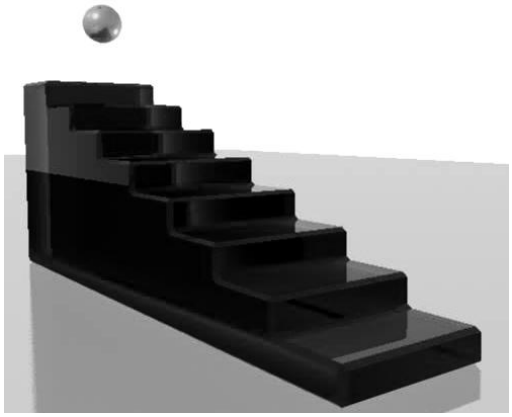

Figure 4: Gray scale of the Bitmap

Figure 5: Delta frame to isolate the object

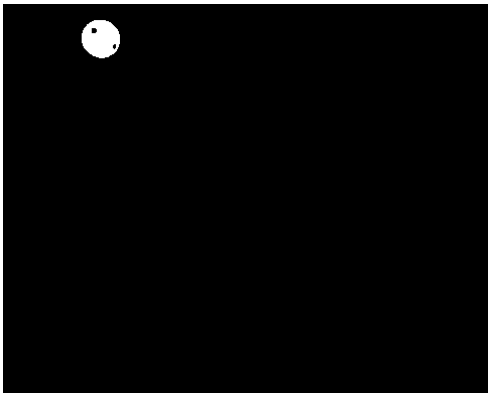

Figure 7: Result after thresholding

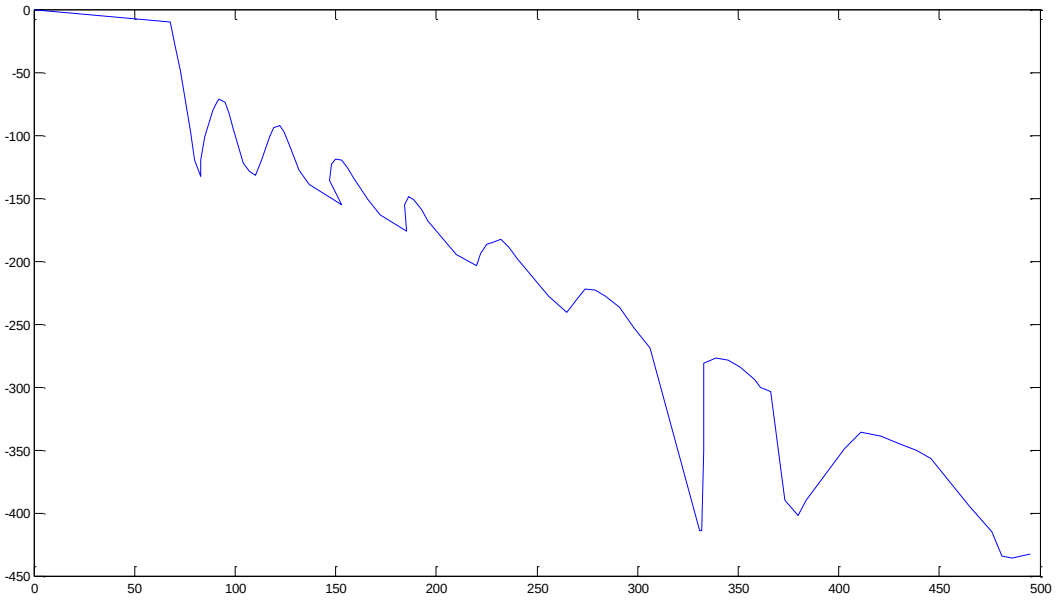

Figure 8: Plot of the centre of gravity of the objects for different frames

6. TIMELINE AND WORKBREAK UP

Week	Harshavardhan	Chuanhai
Week 1	Matlab model	Matlab model
Week 2	Design	Design
Week 3	Software Blocks	Software Blocks
Week 4	Hardware Blocks	Hardware Blocks
Week 5	Testing	Testing
Week 6	Improvements	Improvements
Week 7	Report	Report