FPGA Implementation Of Efficient Algorithm Of Image Splitting For Video Streaming Data

Swamy.TN*, Rashmi. KM, Dr.P.Cyril Prasanna Raj ***, Dr.S.L.Pinjare ******

*(Assistant Prof, Rajiv Gandhi Institute of Technology, Bangalore)
** (Assistant Prof, SDMIT, Ujire, Mangalore)
***(Director, NXG semiconductor Technologies, Bangalore)
****(PG coordinator, Nitte Meenakshi Institute of Technology, Bangalore)

ABSTRACT

Video splitting is the process of dividing the video into non overlapping parts. Then row mean and column mean or each part is obtained. After applying transform on these, features sets can be obtained to be used in image retrieval. By using splitting higher precision and recall can be obtained. Streaming refers to transferring video data such that it can be processed as a steady and continuous stream over the network. With streaming the client browser or plug in can start displaying the multimedia data before the entire file has been transmitted. In this paper original image of particular size is split into blocks. Each block is enlarged to the original dimension without blurring. Enlarged image is displayed on the big screen. This can be done on video streaming data. Designs are implemented using Xilinx and Mat lab tools and Xilinx Vertex-2p FPGA board.

Keywords– Croma, interpolation, luma, simulink, system generator.

I. INTRODUCTION

1.1 Video splitting

Video splitting is the process of dividing the video into non overlapping parts. Then row mean and column mean of each part is obtained. After applying transform on these, feature sets can be obtained to be used in image retrieval. By using splitting higher precision and recall can be obtained. Natural images are captured using image sensors in the form of voltage intensities that are digitized and stored in memory banks. Large storage space is required to store and process these digital samples [1]. For example, a color image of size 256*256 represented using 24-bit requires a storage space of 47 Mega bits. Similarly the storage space for a three hour movie requires 92000 Giga bits.

Processing and transmission of huge image data is time consuming and very cumbersome. Hence the project has a major application in all areas of image processing. Fig1 shows original and spitted non overlapping parts respectively. Streaming refers to transferring video data such that it can be processed as a steady and continuous stream over the network. With streaming the client browser or plug in can start displaying the multimedia data before the entire file has been transmitted.

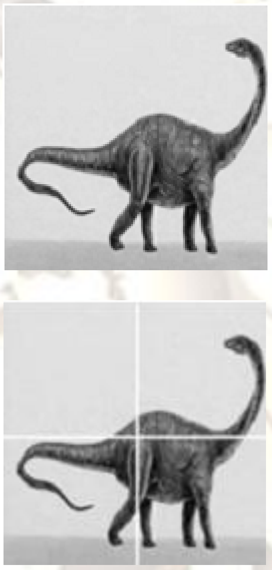
Fig1. Original image and splitted image

1.2 Interpolation

The goal of interpolation is to produce the acceptable images at different resolution from a single low resolution image. The actual resolution of the image is defined as the number of pixels. Image interpolation is a basic tool used extensively in zooming, shrinking, rotating and geometric corrections. Interpolation is the process of using known data to estimate values of unknown locations, suppose that an image of size 500*500 pixels has to be enlarged 1.5 times to 750*750 pixels. A simple way to visualize zooming is to create an imaginary 750*750 grid with the same pixel spacing as the original, and then shrink it so that if fits exactly over the original image. Obviously, the pixel spacing in the shrunken 750*750 grid will be less than the pixel spacing in the original image [2]. To perform intensity-level assignment for any point in the overlay, we look for its closest pixel in the original image and assign the intensity of that pixel to the new pixel in the 750*750 grid. When we are finished assigning intensities to all the points in the overlay grid, we expand it to the original size to obtain the zoomed image. This method is called nearest neighbor interpolation because it assigns to each new location the intensity of its nearest neighbor. The basic criteria for a good interpolation method are geometric invariance, contrast invariance, noise, edge preservation, aliasing, texture preservation, over smoothing, application awareness, and sensitivity to parameters [3].

II. DESIGN

The block diagram of the design is shown in Fig2. Video from a web camera at 30 frames per second is applied to the video input block. Resize block enlarge or shrinks image size. Captured video will be in RGB format. Input video is converted into croma and luma components.

Luma represents the brightness in an image and it represents the achromatic image without any color while the croma component represents the color information. Image split block splits the image into number of blocks. Each spitted block is resized using bicubic interpolation technique. The split image is displayed using the video display output block.

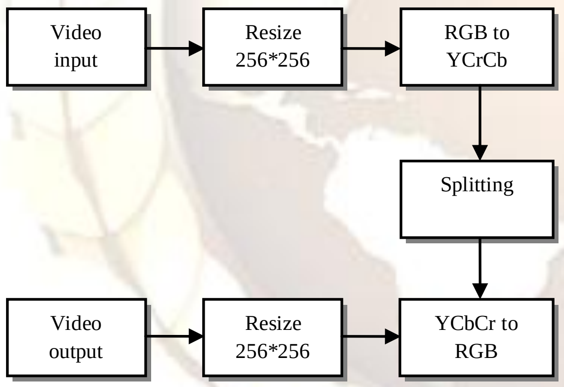

Fig2. Block diagram of image splitting

The design is implemented using simulink and system generator [3][4] environment.

Image will be in matrix format. It has to be converted into serial data before applying into the system generator block. The design for converting matrix to serial format is as shown in Fig3.

Transpose block is used to convert M*N matrix into N*M matrix. The block named convert 2D to 1D block reshapes an M*N matrix into a 1D vector with length M*N. Frame conversion block specifies the frame status of the output signal. The unbuffer block unbuffers frame based input into sample based output.

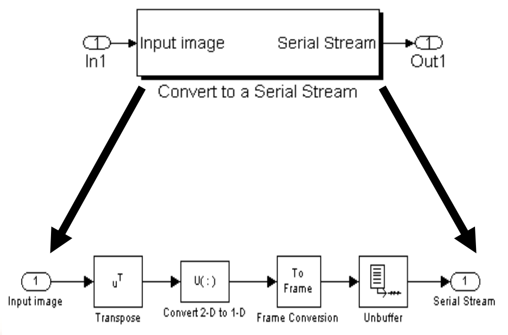

Fig3. Image matrix to serial format

The conversion of serial data into matrix format is shown in Fig4. The hardware design using the system generator is shown in Fig5.

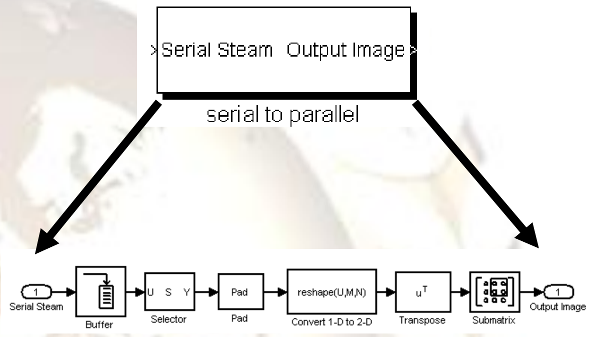

Fig4. Image serial to parallel conversion

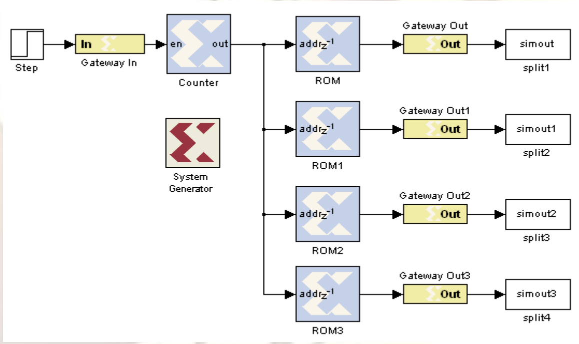

Fig5. System generator model

III. PARAMETERS

3.1 Video input block

3.1.1 Device

Device describes the image acquisition device to which we want to connect. The items in the drop down list of device vary depending on which devices we have connected to the system. All video capture devices supported by image acquisition toolbox are supported by the block.

3.1.2 Video format.

Video format shows the video formats supported by the selected device. The drop down list of video format varies with each device. If the device supports the use of camera files, from camera file will be one of the choices in the list.

3.1.3 Block sample time.

It specify the sample time of the block during the simulation. This is the rate at which the block is executed during simulation. The default is 1/30.

3.1.4 Ports model.

It is used to specify either a single output port for all color spaces, or one part for each band (for example R, G, B). When you select one multidimensional signal, the output signal will be combined into one line consisting of signal information for all color signals. Select separated color signals if you want to use three ports corresponding to the uncompressed red, green, and blue color bands.

3.1.5 Data type

Image data type when the block outputs frames. This data type indicated how image frames are output from the block to simulink. It supports all Matlab data types and single is the default.

3.2 Color space conversion

The color space conversion block converts color information between color spaces. Conversion parameter is used to specify the color spaces we are converting. Conversion between RGB and YCbCr and vice-versa are defined by the following Equation (1).

$$\begin{bmatrix} Y' \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 16 \\ 128 \\ 128 \end{bmatrix} + A \times \begin{bmatrix} R' \\ G' \\ B' \end{bmatrix}$$

$$\begin{bmatrix} R' \\ G' \\ B' \end{bmatrix} = B \times \left(\begin{bmatrix} Y' \\ Cb \\ Cr \end{bmatrix} - \begin{bmatrix} 16 \\ 128 \\ 128 \end{bmatrix} \right) \quad (1)$$

IV. RESULTS

4.1 Simulink output

The input image is in RGB format and of size 256*256*3 shown in Fig6.

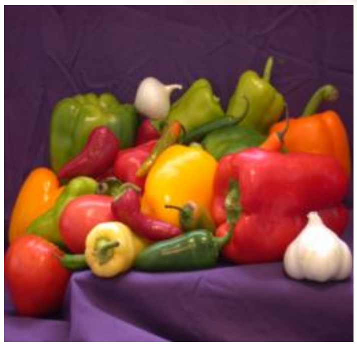
Fig6. Original image 256*256*3

The input image/video is splitted into four blocks of dimension 128*128 each as shown in Fig7.

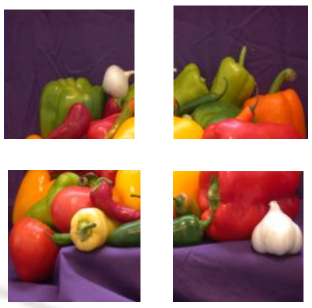
Fig7. Splitted image(128*128*3)

These splitted images are interpolated to the original dimension of the input, i.e. 256*256. Bicubic interpolation is used to resize the splitted image into the original dimension. The resized image is of good quality because it resembles almost the resolution of the original image. The interpolated images are shown in Fig8.

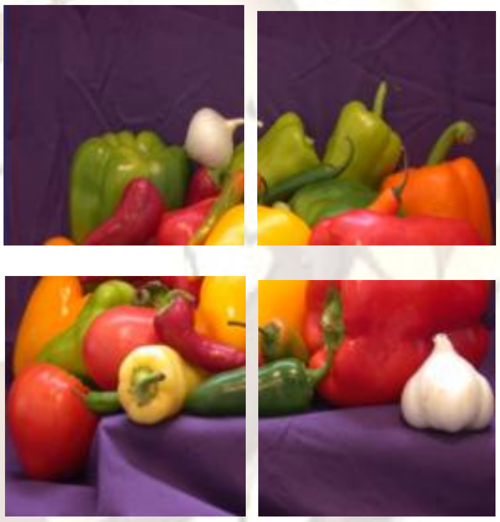
Fig8. Interpolated image(256*256*3)

4.2 Hardware output

The RGB image of dimension 128*128*3 is considered as input image as shown in Fig9.

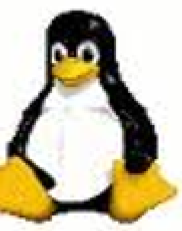
Fig9. Input image for hardware(128*128*3)

The image is resized to 128*128 and converted into gray image as shown in Fig10.

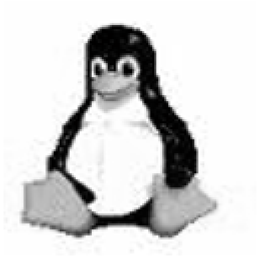

Fig10. Gray image(128*128)

The output on the FPGA hardware is as shown in Fig11.

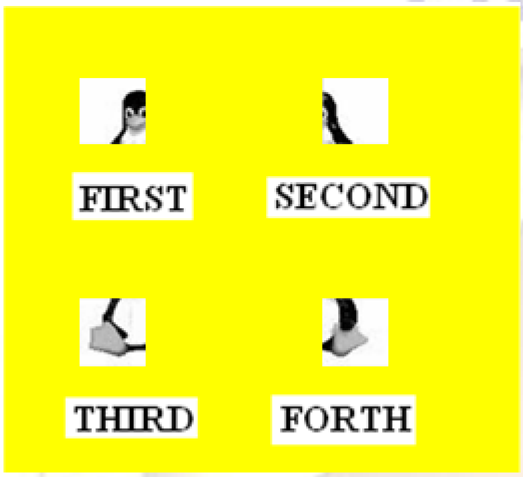

Fig11. Hardware output image(128*128)

V. CONCLUSION

This paper is very useful in displaying wider images for commercial purposes in Aerodromes and railway stations. And can be widely used in medical fields. In medical field doctor can easily indentify the affected area by zooming that part without zooming entire image.

REFERENCES

[1] Ian kuon and Jonathan Rose, Measuring the gap between FPGAs and ASICS, IEEE transcation on Computer-Aided of Integrated Circuits and systems, 26(2): 203-215,2007.

[2] Marco Aurelio,Nunu maganda, Real time FPGA based architecture for bicubic interpolation: an application for digital image scaling. Proc of the International conference on reconfigurable computing and FPGAs, 2005

[3] T.Saidani, D. Dia, W. Elhamzi, M. Atri, and R. Tourki, Hardware Cosimulation for Video Processing using Xilinx System Generator, Proc of the World Congress on Engineering, Vol 1, page 1-3, 2009

[4] Matthew Own by, Dr Wagdy.H. mhmoud, A design methodology for implementing DSP with xilinx system generator for Matlab, processdings of 35th south eastern symposium, Vol 15, page 2226-2238, 2006

[1] [5]Akhtar. P and Azhar. F, A single image interpolation scheme for enchanced super resolution in Bio-Medical Imaging, 4th International Conference on Bio-informatics and Bio medical Engineering, pages 1-5, June 2010.

[6] Fatma T. Arslan and Artyom M. Grigoryan, Method of image enhancement by splitting signal, IEEE international Conference on Acoustics, speech and signal processing, vol 4, page iv/177 - iv/180, March 2005

[7] Rafael C. Gonzalez and Richard E. Woods, Digital Image Processing, Pearson Prentice Hall, 3 Edition, 2008.

[8] Samir Palnitkar, Verilog HDL a Guide to Digital Design and Synthesis, Pearson Education, 2 Edition, 2009.