International Journal of Research and Development in Applied Science and Engineering (IJRDASE)
ISSN: 2454-6844

BMP/GIF/ JPEG Image Implementation Through FPGA on VGA Monitor using VHDL

Sandeep Kumar
M. Tech. (E.C.)
Integral University, Lucknow (U.P.)
sandeep.88sonu @gmail.com

Monauwer Alam
Dept of Electronics & Communication
Integral University, Lucknow (U.P.)
malam_iu@rediffmail.com

Abstract- Industrial production machines of today must be highly flexible in order to competitively account for dynamic and unforeseen changes in the product demands. Field-programmable gate arrays (FPGAs) are especially suited to fulfill these requirements; FPGAs are very powerful, relatively inexpensive, and adaptable, since their configuration is specified in an abstract hardware description language. Xilinx CORE Generator System generates and delivers parameterizable cores optimized for Xilinx FPGAs. CORE Generator is mainly used to create high density, high performance designs in Xilinx FPGAs in less time. The CORE Generator is included with the ISE WebPack and ISE Foundation software and comes with an extensive library of Xilinx LogiCORE IP. These include DSP functions, memories, storage elements, math functions and a variety of basic elements.

Keywords- Core generation, FPGA, Simulation, VGA, Xilinx.

1. Introduction:

Field Programmable Gate Array (FPGA) technology has become an alternative for the implementation of software algorithms. The unique structure of the FPGA has allowed the technology to be used in many applications from video surveillance to medical imaging applications. FPGA is a large-scale integrated circuit that can be re-programmed. The term “field programmable” refers to ability of changing the operation of the device. Gate array refers to the basic internal architecture that makes re-programming possible. Implementations of real-time image processing algorithms can be done on general purpose microprocessors. The application of FPGA in image processing has a large impact on image or video processing. This is due to the potential of the FPGA to have parallel and high computational density as compared to a general purpose microprocessor. This step is coupled together with the ability of FPGA of being re-programmable that adds flexibility in the development of image processing algorithms on FPGA. During the recent years FPGAs have become the dominant form of programmable logic.

LOW-LEVEL vision engines constitute a crucial intermediate step toward a fully symbolic interpretation of the visual environment by transforming pixel-based intensity values into a more meaningful description, such as correspondences between images. In most current vision systems, the low-level component summarizes the visual signal into a sparse set of interesting features [8] (e.g., corners and/or edges) and restricts further processing, such as correspondence finding, to this condensed representation. Such an approach ignores large parts of the visual signal (e.g., textured regions) and is often motivated by computational resource limitations. Recent advances in massively parallel hardware now make it feasible to instead establish reliable correspondences for most pixels in real time by processing the signal in its entirety. Due to the abundance of optical flow and dense stereo estimation methods, we only review a selection of recent real-time implementations. Both for optical flow and stereo, one can, broadly speaking, distinguish between local and global methods. The former only use image information in a small region surrounding the pixel whereas the latter enforce additional constraints on the estimates (such as spatial or spatiotemporal smoothness) [10]. Local methods are easier to implement efficiently in parallel architectures and real-time implementations exist on a variety of platforms (PC (CPU) [6], FPGA [7] and GPU). Although they are more accurate, global methods are more difficult to parallelize and realtime performance can only be achieved at low resolutions or through a variety of algorithmic simplifications [11]. A multiscale coarse-to-fine control scheme [12] is commonly applied in real-time implementations to efficiently extend the dynamic range of both local and global methods. This work considers local coarse-to-fine phasebased methods that exhibit an increased robustness compared to other real-time local methods (for a detailed discussion, see [9]).

In this project we will try to take an Input image from a media file in either GIF or Jpeg format and then will take the digital information of the image in the form of coefficient file with the help of MATLAB and the information would be stored into a FPGA ROM , for this Xilinx CORE Generator System generates and delivers parameterizable cores optimized for Xilinx FPGAs. CORE Generator is mainly used to create high density, high performance designs in Xilinx FPGAs in less time. The CORE Generator is included with the ISE WebPack and ISE Foundation software and comes with an extensive library of Xilinx LogiCORE IP. These include DSP functions, memories, storage elements, math functions and a variety of basic elements.

2. Related Work:

Michael Schaeferling et. al. (2015), [1] according to them image processing on embedded platforms is still a challenging task, especially when implementing extensive computer vision applications. Field-programmable gate arrays (FPGAs) offer a suitable technology to accelerate

image processing by customized hardware. Most available image processing frameworks for FPGAs concentrate on pixel-based modules for simple preprocessing tasks. This work presents a framework which also aims to cover the integration of higher-level algorithms. Therefore, it offers modules and interfaces to perform window-oriented filter operations and incorporate software defined operations. Several complex, higher-level algorithms, such as undistortion and rectification, natural feature description, edge detection and Hough transform have been adopted and integrated into the frameworks image processing flow. This work describes the framework, its interfaces and some of the existing modules. Finally, some applications which were already implemented using this framework are presented.

Mariangela Genovese and Ettore Napoli (2014), [2] according to them background identification is a common feature in many video processing systems. This work proposed two hardware implementations of the OpenCV version of the Gaussian mixture model (GMM), a background identification algorithm. The implemented version of the algorithm allows a fast initialization of the background model while an innovative, hardware-oriented, formulation of the GMMequations makes the proposed circuits able to perform real-time background identification on high definition (HD) video sequences with frame size 1920 × 1080. The first of the two circuits is designed with commercial field-programmable gate-array (FPGA) devices as target. When implemented on Virtex6 vlx75t, the proposed circuit process 91 HD fps (frames per second) and uses 3% of FPGA logic resources. The second circuit is oriented to the implementation in UMC-90 nm CMOS standard cell technology, and is proposed in two versions. Both versions can process at a frame rate higher than 60 HD fps. The first version uses the constant voltage scaling technique to provide a low power implementation. It provides silicon area occupation of 28847 µm2 and energy dissipation per pixel of 15.3 pJ/pixel. The second version is designed to reduce silicon area utilization and occupies 21847 µm2 with an energy dissipation of 49.4 pJ/pixel.

Carlos Gonzalez et. al. (2013), [3] they worked on hyper spectral imaging that is a growing area in remote sensing in which an imaging spectrometer collects hundreds of images (at different wave length channels) for the same area on the surface of the Earth. Hyper spectral images are extremely high-dimensional, and require advanced on-board processing algorithms able to satisfy near real time constraints in applications such as wild land fire monitoring, mapping of oil spill sand chemical contamination, etc. One of the most widely used techniques for analyzing hyper spectral images is spectral un mixing, which allows for sub-pixel data characterization. This is particularly important since the available spatial resolution in hyper spectral images is typically of several meters, and therefore it is reasonable to assume that several spectrally pure substances (called end members in hyper spectral imaging terminology) can be found within each imaged pixel. In this work we explore the role of hardware accelerators in hyper spectral remote sensing missions and further inter-compare two types of solutions: field programmable gate arrays (FPGAs) and graphics processing units (GPUs). A full spectral un mixing chain is implemented and tested in this work, using both types of accelerators, in the context of areal hyper spectral mapping application using hyper spectral data collected by NASA's Airborne Visible Infra-Red Imaging Spectrometer (AVIRIS). The work provides a thoughtful perspective on the potential and emerging challenges of applying these types of accelerators in hyper spectral remote sensing missions, indicating that the reconfigurability of FPGA systems (on the one hand) and the low cost of GPU systems (on the other) open many innovative perspectives toward fast on-board and on-the-ground processing of remotely sensed hyper spectral images.

Different image processing approaches are presented by **Horace Josh et. al., (2013),** [4] that are candidates for the Monash Vision Group prosthetic vision device. As described in a companion work, the Monash Vision Group is developing a bionic eye based on the implantation of 7-11 stimulation tiles on the primary visual cortex of the brain. In the lead up to an expected 2014 first in-human trial of this device, potential image processing techniques and intuitive user interfaces need to be developed and evaluated. An FPGA Hatpack has been
developed to give normally sighted people a similar binary limited resolution visual experience expected of a Bionic Eye recipient. The Hatpack has been used to generate new psychophysics results to evaluate and improve the performance and practicality of three different methods of luminance threshold selection. The results show an interesting difference in terms of performance and highlight areas of possible improvement for the algorithms used.
In this work we have outlined different luminance threshold selection techniques and presented results of a preliminary round of psychophysics testing to determine which technique performs best. The Manual and Automatic modes were found to provide significantly better response accuracy than the Static mode. However, there was little difference when comparing the two modes against each other which will need to be addressed in future testing involving a performance task.

Hyper spectral remote sensing attempts to identify features in the surface of the Earth using sensors that generally provide large amounts of data. The data are usually collected by a satellite or an airborne instrument and sent to a ground station that processes it. The main bottleneck of this approach is the (often reduced) bandwidth connection between the satellite and the station, which drastically limits the information that can be sent and processed by **Carlos González et. al., (2012),** [5] in real time. A possible way to overcome this problem is to include onboard computing resources able to preprocess the data, reducing its size by orders of magnitude. Reconfigurable field-programmable gate arrays (FPGAs) are a promising platform that allows hardware/software codesign and the potential to provide powerful onboard computing capability and flexibility at the same time. Since FPGAs can implement custom hardware solutions, they can reach very high performance levels.

Moreover, using run-time reconfiguration, the functionality of the FPGA can be updated at run time as many times as needed to perform different computations. Hence, the FPGA can be reused for several applications reducing the number of computing resources needed. One of the most popular and widely used techniques for analyzing hyperspectral data is linear spectral unmixing, which relies on the identification of pure spectral signatures via a so-called endmember extraction algorithm. In this work, we present the first FPGA design for N-FINDR, a widely used endmember extraction algorithm in the literature. Our system includes a direct memory access module and implements a prefetching technique to hide the latency of the input/output communications. The proposed method has been implemented on a Virtex-4 XC4VFX60 FPGA (a model that is similar to radiation-hardened FPGAs certified for space operation) and tested using real hyperspectral data collected by NASA's Earth Observing-1 Hyperion (a satellite instrument) and the Airborne Visible Infra-Red Imaging Spectrometer over the Cupritemining district in Nevada and the Jasper Ridge Biological Preserve in California. Experimental results demonstrate that our hardware version of the N-FINDR algorithm can significantly outperform an equivalent software version and is able to provide accurate results in near real time, which makes our reconfigurable system appealing for onboard hyperspectral data processing.

3. Methodology:

Algorithms for image processing are normally classified into one of three levels: low, intermediate or high. Low-level algorithms operate on individual pixels or neighbourhoods. Intermediate-level algorithms either convert pixel data into a different representation, such as a histogram, coordinate or chain code, or operate on one of these higher representations. High-level algorithms aim to extract meaning from the image using information from the other levels. This could be in the form of rejecting a component or identifying where an object is within an image. When moving from low to the high-level representations there is a corresponding decrease in exploitable parallelism due to the change from pixel data to more descriptive representations. However there is also a reduction in the amount of data that must be processed, allowing more time to do the processing. Due to their structure, FPGAs are most appropriate for use with computationally intensive tasks which form the vast majority of low and intermediate-level operations. The large data sets and regular repetitive nature of the operations can be exploited. For this reason it has been traditional in many systems for the

FPGA to handle the low-level operations and then pass the processed data to a microprocessor which then executes the high-level operations. With increasing FPGA size, it is now possible to implement processor cores on the reconfigurable fabric, which means the FPGA can form the core of the system.

The application of FPGAs to image processing is a rapidly growing research area given recent increases in the power and size of FPGAs. The potential speed gains make it an attractive topic, although there are many challenges to implementing working algorithms on FPGAs. Most newcomers consider simply porting an existing software algorithm to an FPGA implementation. Unfortunately, this generally gives disappointing results. This Project aims to help those wishing to use FPGAs to accelerate image processing algorithms through some of the pitfalls, and provide a range of techniques that result in an efficient implementation, both computationally and in terms of resource requirements.

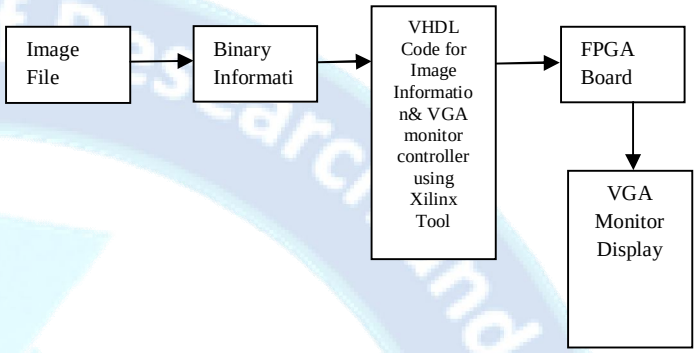

Fig 1: Block Diagram of Proposed Work.

In this project we will try to take an Input image from a media file in either GIF/ Jpeg format. And then will take the digital information of the image in the form of coeffiecient file with the help of Matlab and the information would be stored into a FPGA ROM , for this Xilinx CORE Generator System generates and delivers parameterizable cores optimized for Xilinx FPGAs. CORE Generator is mainly used to create high density, high performance designs in Xilinx FPGAs in less time. The CORE Generator is included with the ISE WebPack and ISE Foundation software and comes with an extensive library of Xilinx LogiCORE IP. These include DSP functions, memories, storage elements, math functions and a variety of basic elements.

Xilinx provides a flexible Block Memory Generator core to create compact, high-performance memories running at up to 450 MHz. Block Memory Generator provides single port and dual port block memory. These memory types differ in selection of operating modes. Matlab tool is used to convert the image that is being processed to .coe file format. Xilinx Core Generator is used to store the coefficient file(.coe) in single port Block ROM by defining the width and depth of the image and image is displayed on VGA monitor using Spartan3E FPGA Board.

Builtin knowledge about Xilinx device architectures allow it to leverage specialized FPGA architectural features to create the most compact, high performance solution. The Block Memory Generator [13] core uses embedded Block Memory primitives in Xilinx FPGAs to extend the functionality and capability of a single primitive to memories of arbitrary widths and depths. Sophisticated algorithms within the Block Memory Generator core produce optimized solutions to provide convenient access to memories for a wide range of configurations.

4. Result and Discussion:

In this work we have taken many signal as a Input (CLK , Reset, User input (3 bits)) and we took for Output (Hsync

(Horizontal Synchronous), VSync (Vertical Synchronous), VGA_out_green, VGA_out_red, VGA_out_blue). All these output signals will change the value as per the given clock frequency and corresponding to that various data would be call from for different locations so that complete image would appear on the Screen. The user can change the background color with user input pins, appered as a switch , on the board for physically and manually during the simulation. The Simulation graph is shown below for various information in fig 2.

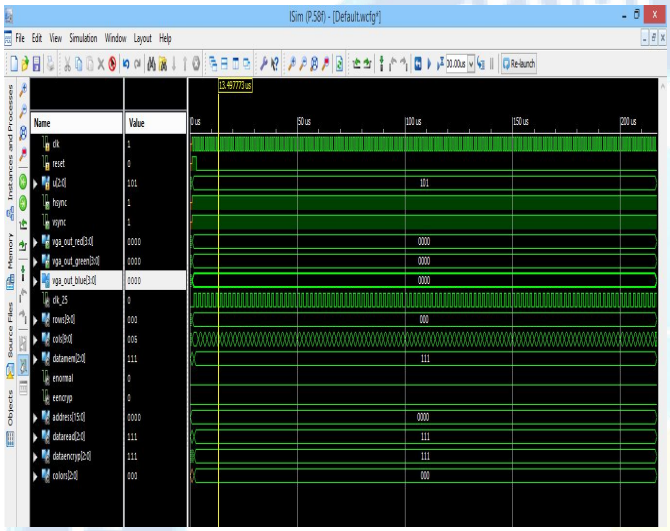

Fig 2: Simulation Result

After simulation we have worked on synthesis. Fig 3 shows the main RTL, Fig 4 shows the internal RTL of our design.

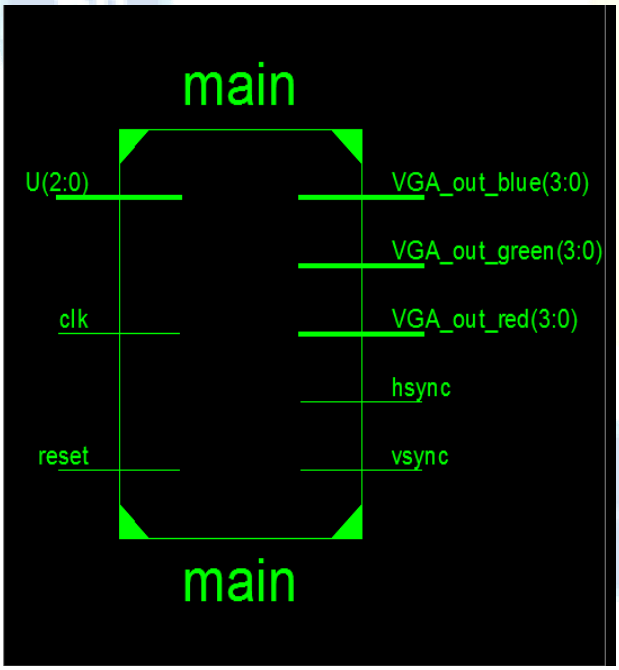

Fig 3: Main RTL

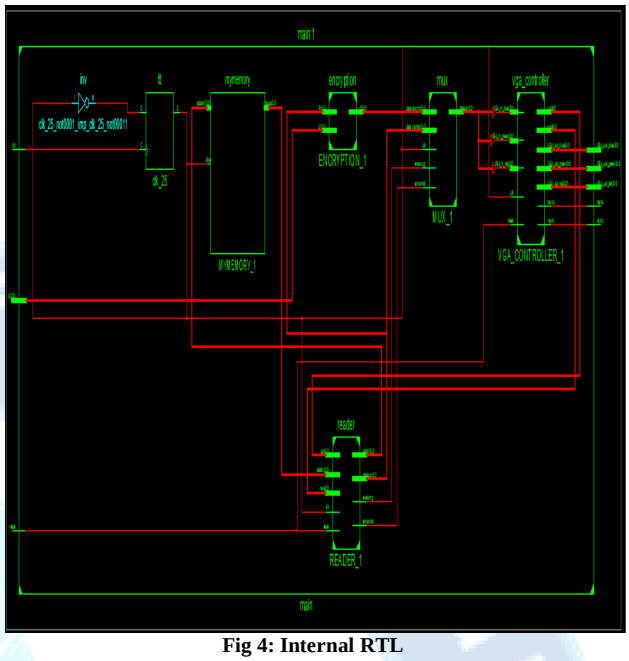

Fig 4: Internal RTL

In this project we had successfully implemented an Image onto Spartan 3E FPGA using VHDL and Xilinx ISE tool. We had created one BROM for storing of the data of an image into digital format , called '.COE' or Coefficient file. For this we used the IP core feature of Xilinx ISE tool, i.e. called Xilinx IP Core Generator . First we generated an IP symbol named Block Memory Generator. In this we had provide the information of Rom Size i.e. Address Bits and Data Bits shows in fig 5 and 6.

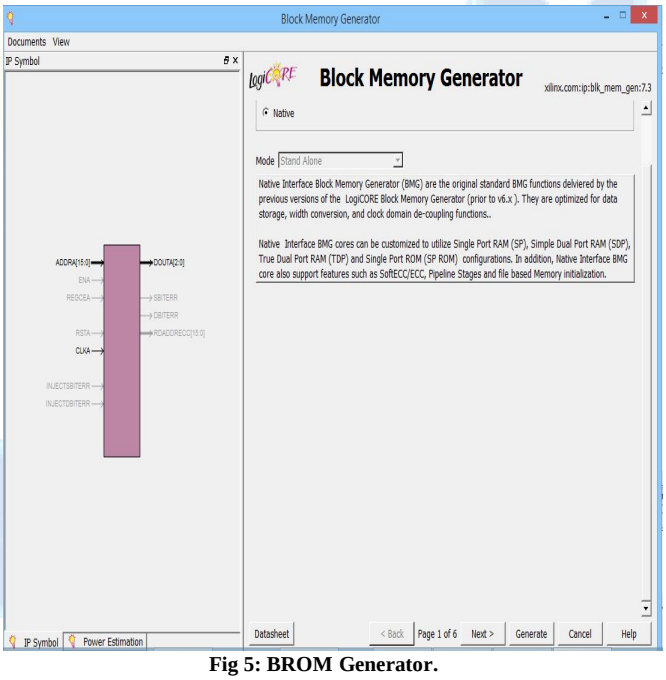

Fig 5: BROM Generator.

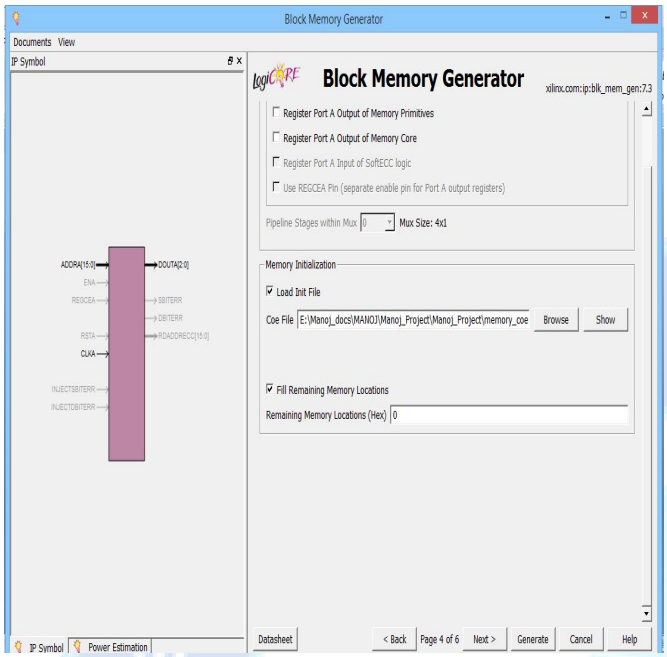

Fig 6: BROM Generator.

Fig 7: Experimental Setup.

Fig 8: Result of FPGA Kit.

5. Conclusion:

To increase software development productivity, efficient code reuse is important. With field-programmable gate array (FPGA) hardware, reusable code blocks often called IP blocks are created. Xilinx Core Generator provides such flexibility to create IP cores for high performance memories. Block Memory Generator is one of the IP core that is provided by Xilinx Core Generator which allows to store larger images (Depends on The FPGA Family and Board Supported ROM/RAM capacity). Matlab tool is used to convert image of any type to .coe file format and is stored in Single Port BLOCK ROM. The data in .coe file is read and the IP core is successfully created in Xilinx ISE environment. The top level design in ISE is synthesized and download.bit file is loaded into Digilent Spartan3E FPGA Starter Board and is displayed on VGA monitor.

References:

[1] Michael Schaeferling et. al., "ASTERICS - An Open Toolbox for Sophisticated FPGA-Based Image Processing" embedded world Conference 2015.

[2] Mariangela Genovese and Ettore Napoli, "ASIC and FPGA Implementation of the Gaussian Mixture Model Algorithm for Real-Time Segmentation of High Definition video" IEEE transactions on very large scale integration (VLSI) systems, vol. 22, no. 3, march 2014

[3] Carlos Gonzalez et. al., " Use of FPGA or GPU-based architectures for remotely sensed hyper spectral image processing" INTEGRATION, theVLSIjournal46(2013)89–103.

[4] Horace Josh et. al., " Psychophysics Testing Of Bionic Vision Image Processing Algorithms Using An FPGA Hatpack" 978-1-4799-2341-0/13/$31.00 ©2013 IEEE.

[5] Carlos González et. al., "FPGA Implementation of the N-FINDR Algorithm for Remotely Sensed Hyperspectral Image Analysis" IEEE transactions on geoscience and remote sensing, vol. 50, no. 2, February 2012.

[6] G. Bradski, “The OpenCV Library,” Dr. Dobb’s J. Software Tools, vol. 25, pp. 120-126, 2000.

[7] J. Dı´az, E. Ros, F. Pelayo, E. Ortigosa, and S. Mota, “FPGA-Based Real-Time Optical-Flow System,” IEEE Trans. Circuits and Systems for Video Technology, vol. 16, no. 2, pp. 274-279, Feb. 2006.

[8] D. Lowe, “Distinctive Image Features from Scale-Invariant Keypoints,” Int’l J. Computer Vision, vol. 60, no. 2, pp. 91-110, 2004.

[9] S. Sabatini, G. Gastaldi, F. Solari, K. Pauwels, M. Van Hulle, J. Dı´az, E. Ros, N. Pugeault, and N. Kru¨ ger, “A Compact Harmonic Code for Early Vision Based on Anisotropic Frequency Channels,” Computer Vision and Image Understanding, vol. 114, no. 6, pp. 681- 699, 2010.

[10] A. Bruhn, J. Weickert, and C. Schnorr, “Lucas/Kanade Meets Horn/Schunck: Combining Local and Global Optic Flow Methods,” Int’l J. Computer Vision, vol. 61, no. 3, pp. 211-231, 2005.

[11] A. Bruhn, J. Weickert, T. Kohlberger, and C. Schnoerr, “A Multigrid Platform for Real-Time Motion Computation with Discontinuity-Preserving Variational Methods,” Int’l J. Computer Vision, vol. 70, no. 3, pp. 257-277, 2006.

[12] J. Bergen, P. Anandan, K. Hanna, and R. Hingorani, “Hierarchical Model-Based Motion Estimation,” Proc. European Conf. Computer Vision (ECCV), pp. 237-252, 1992.

[13] Hutchings, B. and Villasenor, J., The Flexibility of Configurable Computing IEEE Signal Processing Magazine, vol. 15, pp. 67-84, Sep, 1998.