Video-Rate Stereo Depth Measurement on Programmable Hardware

Ahmad Darabiha
University of Toronto
Department of Electrical and Computer Engineering
ahmadd@eecg.utoronto.ca

Jonathan Rose
University of Toronto
Department of Electrical and Computer Engineering
jayar@eecg.utoronto.ca

W. James MacLean
University of Toronto
Department of Electrical and Computer Engineering
maclean@eecg.utoronto.ca

Abstract

This paper describes the implementation of a stereo depth measurement algorithm in hardware on Field-Programmable Gate Arrays (FPGAs). This system generates 8-bit sub-pixel disparities on 256 by 360 pixel images at video rate (30 frames/sec). The algorithm implemented is a multi-resolution, multi-orientation phase-based technique called Local Weighted Phase-Correlation [12]. Hardware implementation speeds up the performance more than 300 times that of the same algorithm running in software. In this paper, we describe the programmable hardware platform, the base stereo vision algorithm and the design of the hardware. We include various trade-offs required to make the hardware small enough to fit on our system and fast enough to work at video rate. We also show sample outputs from the functioning hardware. Although this paper is specifically focused on phase-based stereo vision FPGA realizations, most of the design issues are common to other DSP and Vision applications.

1 Introduction

High-level computer vision tasks, such as robot navigation and collision avoidance, require 3-D depth information about the surrounding environment at video rate. Current general purpose microprocessors are too slow to perform stereo vision at video rate. For example, it takes several seconds to execute a medium-sized stereo vision algorithm for a single pair of images on a 1 GHz general-purpose microprocessor. To overcome this limitation, designers in the last decade have built custom-designed hardware systems to accelerate the performance of the vision systems. Hardware implementation allows one to exploit the parallelism that usually exists in image processing and vision algorithms, and to build systems to perform specific calculations very quickly compared to software. By processing several parts of the data in parallel, we can speed up the overall functioning and achieve video-rate performance. Custom-designed hardware, however, has two major disadvantages: 1) The design cycle is slow: The time required for fabrication and test of a typical Application Specific Integrated Circuit (ASIC) is in the order of a few months; 2) The fabrication process of the chip and also circuit board is expensive, on the order of hundreds of thousands of dollars.

Over the past decade a third option between software and custom hardware has become viable: using re-programmable chips called Field-Programmable Gate Arrays (FPGAs). These devices consist of programmable logic gates and routing that can be re-configured to implement essentially any hardware function [4]. This method combines the advantage of custom-designed hardware with the advantages of software implementation, which are reprogrammability and rapid design cycle. The reprogrammability feature also allows the designers to use the same hardware system for other vision or non-vision tasks, which again can amortize the cost of the system.

This paper describes the development of a highly complex phase-based stereo vision algorithm on a reconfigurable platform. This section explains the generic architecture of an FPGA, and then introduces the programmable hardware system we used. Section 2 describes the theoretical basis of the phase-based stereo vision algorithm implemented in this work. Section 3 illustrates the implementation process of the algorithm on hardware. Finally, section 4 presents the overall performance of the hardware system and compares it with a software implementation.

1.1 Field-Programmable Gate Arrays

An FPGA is a chip that allows its user to control and reprogram the functionality of its logic circuits. All FPGAs consist of three major components [4]: 1) Logic blocks; 2) I/O blocks; and 3) programmable routing as shown in Figure 1. To implement a circuit on an FPGA, each Logic Block (LB) is programmed to perform a small part of the logic required by the circuit and each I/O block is programmed to act as an input or output, as required by the circuit. The programmable routing is also configured to make all the necessary connections between logic blocks and from logic blocks to I/O blocks.

The processing power of an FPGA is directly proportional

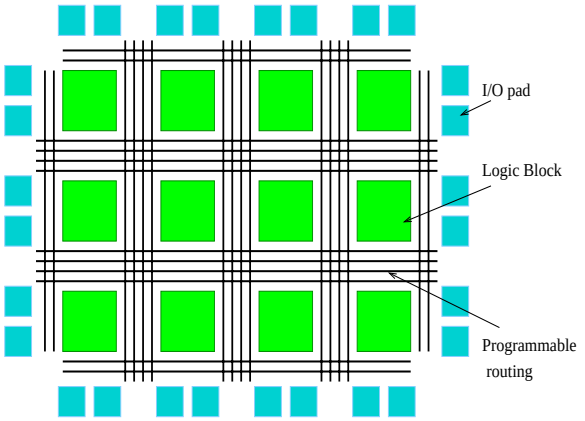


Figure 1: Architecture of a generic FPGA

to the processing capabilities of its LBs and the total number of LBs available in the array. Currently most of the commercial FPGAs use LBs that contain one or more Look-up Tables (LUTs) (typically a 4-input LUT). A 4-input LUT can implement any binary function of 4 logic inputs. Figure 2 shows the architecture of a simple LB containing one 4-input LUT and one flip-flop for storage. Modern FPGAs also contain blocks of on-chip memory as well. For example, the FPGAs used in this work contain 160 blocks of 4kbits of RAM and 38,000 LUT-flip flop pairs. Current commercial FPGAs contain 93,000 LUTs and flip flops in a single FPGA. This amount of logic circuitry along with other features of the modern FPGAs, such as on-chip memory and dedicated multipliers, makes programmable hardware a viable and efficient solution for accelerating complex image processing applications. For designs that are too large to fit on a single FPGA, a group of FPGAs connected with a programmable interconnection network can be used.

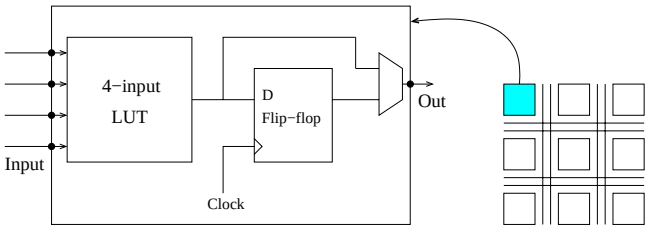


Figure 2: Simplified architecture of a logic block with one 4-LUT

1.2 Our Programmable System: Transmogrifier-3A

The Transmogrifier-3A (TM-3A) [15][25] is a reconfigurable board built at the University of Toronto containing four Xilinx Virtex2000E FPGAs [28]. Each FPGA is connected to the other three chips via a 98 bit bus. Each chip is also connected to a 256K x 64 bit synchronous SRAM memory, an I/O connector and a bus which allows communication with a housekeeping FPGA. The housekeeping chip communicates with the host computer for download and control functions. Video encoder/decoder chips are also integrated to the TM-3A board, which give the ability to receive video stream and also send output results directly to display. Circuits built on the TM-3A can operate at frequencies of up to 100 MHz.

1.3 Related Work

A variety of reconfigurable stereo machines have been introduced in recent years. The PARTS reconfigurable computer [29] consists of a 4 x 4 array of mesh connected FPGAs with a maximum total number of about 35,000 4-input LUTs. A stereo system was developed on PARTS based on the census transform, which mainly consists of bitwise comparisons and additions [30]. In [8], a 4 x 4 matrix of small FPGAs is used to perform the cross-correlation of two 256 x 256 images in 140 ms. In [17], a combination of FPGA and Digital Signal Processors (DSPs) is used to perform edge-based stereo vision. They use FPGAs to perform low level tasks like edge detection and DSPs for high level image processing tasks.

2 Local Weighted Phase-Correlation Stereo Matching Algorithm

The heart of any stereo vision system is stereo matching, the goal of which is to establish correspondence between two points arising from the same element in the scene. Stereo matching is usually complicated by several factors such as lack of texture, occlusion, discontinuity and noise. Researchers have proposed techniques to solve and improve the performance of stereo matching that can be categorized into three major groups: 1) Intensity-Based; 2) Feature-Based; and 3) Phase-Based. Intensity-Based techniques assume that image intensity corresponding to a 3-D point remains the same in binocular images. These techniques usually lead to window-based and coarse-to-fine strategies which are often thought to be unsatisfactory. Feature-Based techniques use sparse primitives such as edges [2] or straight line segments [1]. The major limitation of all feature-based techniques is that they cannot generate dense disparity maps. and hence they often need to be used in conjunction with other techniques. In phase-based techniques the disparity is defined as the shift necessary to align the phase value of band-pass filtered versions of two images. In [10] it is shown that phase-based methods are robust when there are smooth lighting variations between stereo images. It also shows that phase is predominantly linear, and hence reliable approximations to disparity can be extracted from phase displacement.

The stereo system developed in this research is based on a phase-based stereo matching technique called “Local Weighted Phase-Correlation” [12]. This algorithm combines the robustness of phase-difference methods with the simple control strategy of phase-correlation methods. The local

weighted phase-correlation algorithm has four major steps:

1. Create a Gaussian pyramid with total number of S scales for both left and right images. Then decompose each scale of the pyramid using oriented quadrature-pair filters [14]. Assuming that $K_j(x)$ is the filter impulse response of the j^{th} orientation, then we can write the complex-valued output of the convolution of $K_j(x)$ with each scale of left and right images, $I_l(x)$ and $I_r(x)$, as:

$$O_l(x) = \rho_l(x)e^{i\varphi_l(x)},\ O_r(x) = \rho_r(x)e^{i\varphi_r(x)} \qquad (1)$$

   in the polar representation, where $\rho(x) = |O(x)|$ is the amplitude and $\phi(x) = \arg[O(x)]$ is the phase of the complex response.

2. Compute voting function $C_{(j,s)}(x,\tau)$ as:

$$C_{(j,s)}(x,\tau) = \frac{W(x) \otimes [O_l(x)conj(O_r(x+\tau))]}{\sqrt{W(x) \otimes |O_l(x)|^2}\sqrt{W(x) \otimes |O_r(x)|^2}} \qquad (2)$$

   where $W(x)$ is a smoothing, small, localized window and τ is the pre-shift of the right filter output.

3. Combine the voting functions $C_{(j,s)}(x,\tau)$ over all orientations, $1 \le j \le F$, and scales, $1 \le s \le S$, where F is the total number of orientations and S is the total number of scales:

$$S(x,\tau) = \sum_{j,s} C_{(j,s)}(x,\tau) \qquad (3)$$

4. For each position x, find the τ value corresponding to the peak in the real part of $S(x,\tau)$ as a good estimate for the true disparity.

Two major features make this algorithm a good candidate for hardware implementation: First, it is primarily composed of linear operations which are easier to implement. Second, there is no iteration or any explicit coarse-to-fine control strategy. This property makes the real-time flow of data possible through the hardware system. In the next sections, we will describe the hardware of the system and then modifications applied to the original local weighted phase-correlation method.

3 Design of a Stereo Vision System

When implementing a complex algorithm on reprogrammable hardware, the most important issue is that there is a fixed amount of hardware available as described in section 1.2. These hardware resources include logic capacity, on-FPGA and off-FPGA available memory, memory access bandwidth and chip-to-chip communication bandwidth.

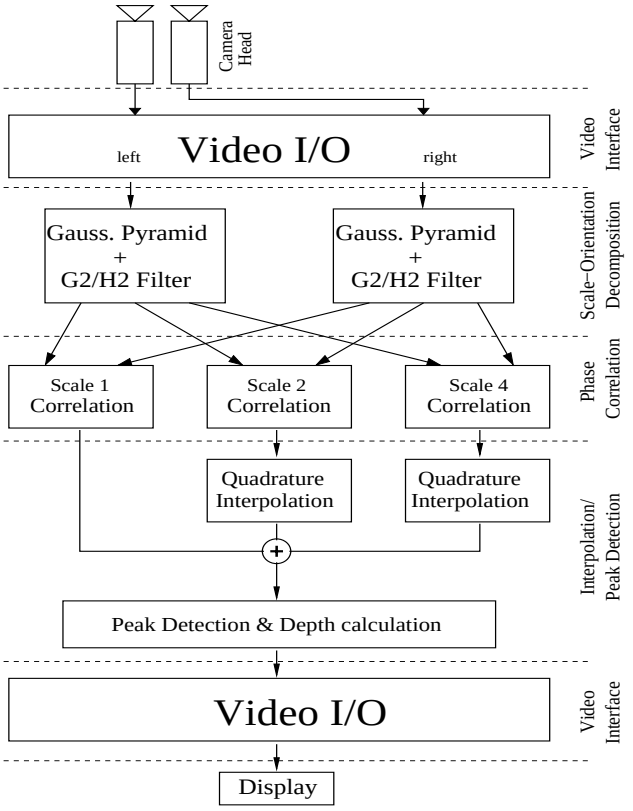


Figure 3: High Level System Architecture

Achieving the best overall performance requires efficient usage of all hardware resources.

In this work, for parallel and efficient hardware implementation of the stereo depth measurement, some modifications are introduced to the original local weighted phase-correlation algorithm. Three major modifications are: 1) Employing fixed-point data representation vs. floating-point representation; 2) Changing the location of low pass localized filter; and 3) Using L1-norm instead of L2-norm in calculating phase correlation. In this section, we first describe the major building blocks of the system and the distribution of the tasks over four FPGAs available on the TM-3A board. Then we will discuss the advantages and effects of modifications on the overall system performance.

The architecture of the stereo vision system is illustrated in Figure 3. It consists of four major units: Video Interface unit, Scale-Orientation Decomposition unit, Phase-Correlation unit and Interpolation/Peak Detection unit. Each of these units is implemented on one separate Xilinx V2000E FPGA available on Transmogrifier-3A.

The **Video Interface Unit** receives the video signal from cameras in composite NTSC format with an image size of 512 x 720 pixels. Since there is only one video input channel available at a time on the TM-3A board, we switch between camera signals after each frame such that we receive 15 frame/sec from each camera. However, the stereo vision sys-

tem itself is capable of processing 30 frame/s if there were a way to simultaneously receive two video-rate image streams from video interface. After frame buffering, the video interface unit sends two non-interlaced gray scale image streams to the Scale/Orientation decomposition unit.

In the **Scale-Orientation Decomposition Unit,** at first a 3-level Gaussian pyramid is built for both left and right images. Each of the pyramid scales is then decomposed into $45°$ and $-45°$ orientations using G2/H2 steerable filters. G2/H2 filters use a set of seven separable 7x7 FIR filters as base filters. By choosing proper coefficients for the linear combination of base filters, one can synthesize filters of arbitrary orientation. The important advantage of using G2/H2 filters for hardware implementation is that they are X-Y separable and therefore require less hardware resources than non-separable filters of the same size. At the end of this unit, filter outputs are reduced to a 16-bit representation before being sent to the phase-correlation unit. The issues related to the effects of fixed-point representation will be discussed in section 3.1

The **Phase-Correlation Unit** is the heart of the vision system. For each pixel in the left image, it computes the real part of the voting function $C_{(j,s)}(x,\tau)$ as mentioned in Eq.(2) for all $1 \le s \le S$, $1 \le j \le F$, $0 \le \tau \le D$ where s is the scale number, S is total number of scales, j is an orientation index, F is the total number of orientations, τ is a pre-shift in the right image and D is the maximum allowed disparity.

In this system, we decompose the input images to three scales and two orientations (S=3, F=2). The maximum value for disparity (D) for the finest scale (s=1) is set to 20 pixels.

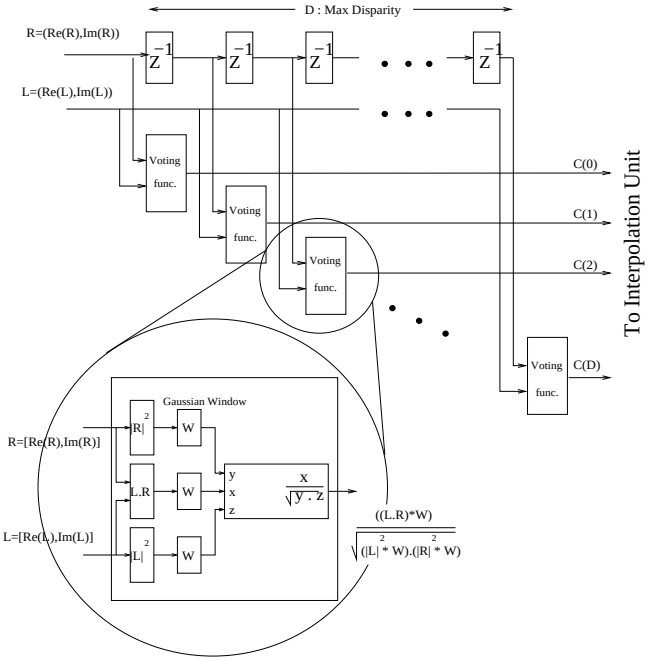


Figure 4: Original correlation block architecture

Hence, for the next coarser scales, D is set to 10 and 5 respectively.

Figure 4 shows the architecture of the phase correlation block. There are two major differences between the original method of computing phase correlation and the version implemented here in hardware. We will discuss these differences and their effects in the next section. The voting functions $C_{(j,s)}(x,\tau)$ are reduced to 8-bit representation before being sent to interpolation unit.

The **Interpolation/Peak-Detection Unit** interpolates two coarser scale voting functions, $C_{(j,2)}(x,\tau)$ and $C_{(j,3)}(x,\tau)$, in both x and τ domain such that they can be combined with the finest scale voting function $C_{(j,1)}(x,\tau)$. It performs quadrature interpolation in the τ domain and constant interpolation in the x domain. The interpolated voting functions are then added together to produce the overall voting function $S(x,\tau)$.

The final step in this unit is peak detection. For each pixel x in the image, it detects the value of τ for which $S(x,\tau)$ is maximum. By performing sub-pixel interpolation of the disparities, this unit produces disparity values with 8-bit resolution from 20-pixel disparity range. The final disparity results are sent back to the **video interface unit** to be written in the video output buffer and displayed on a monitor. Table 1 lists the hardware resources used in each unit in terms of number of Look Up Tables (LUTs), flip-flops and the number of on-chip fast memory banks. In the design of the hardware, we have assumed that the two cameras have identical focal length and are vertically aligned such that no rectification [26] is required. Also, there is no post-processing stage such as left-to-right/right-to-left validation or smoothing/gap filling implemented in hardware.

Table 1: Hardware resources for each unit

Unit Name	# 4-input LUTs	# of flip-flops	On-chip Memory (bits)
Video Interface	169	71	-
Scale/Orient. Decomp.	23,151	18,020	614,400
Phase-correlation	16,709	30,961	-
Interpolation	26,615	33,974	172,032

3.1 Fixed-Point Representation

Floating-point mathematical operations require extensive resources to implement in hardware. Since we know the required scale and precision in each stage of the stereo algorithm, we can implement it in a far more efficient fixed-point representation. However, there is always a trade-off between the accuracy of fixed-point representation and the hardware efficiency: minimum quantization error requires using wide fixed-point representations, but wider signals require larger

mathematical operators (dividers, multipliers, adders, etc.). In addition, more hardware resources are needed for data path circuits. As an example, Figure 5(a) shows the relation between the required number of LUTs to create parallel (one clock cycle) multipliers or dividers versus the input width of the multiplier or divider. The number of LUTs increases with the square of the input width.

So, we need to make good decisions for the precision of the variables and operations in each stage of the algorithm. This analysis requires both knowledge of the target hardware and the algorithm itself. Few tools have been developed to solve this problem. For example, [6] is a framework for automatically determining fixed-point precision of floating-point calculations. In our work, before moving to hardware, we have conducted emulations in order to find the most efficient precisions for each stage of the system. To find the optimum precision, we have simulated our hardware system for a range of precisions and calculated the quantization error.

Figure 5(b) shows the Mean Square Error between using full-precision and a fixed-point multiplier in the correlation unit for a range of precisions (assuming that all other stages are in full precision). In this case. for example, based on Figure 5(a) and Figure 5(b), an 8-bit multiplier was chosen for the phase-correlation unit. Since the proper resolution is not fixed across all stages of the system, we performed the analysis similar to the above for each of the units and we have chosen 16 and 8 bit widths for Orientation Decomposition Unit and Interpolation Unit respectively.

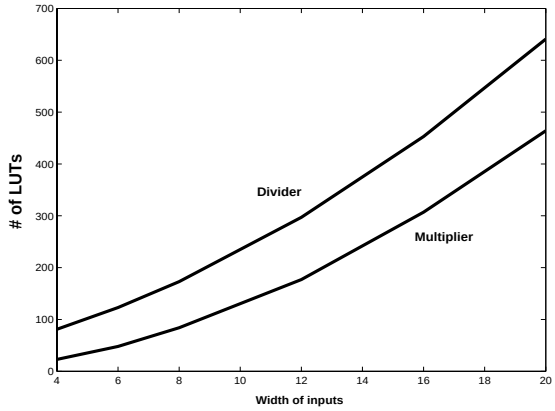


Figure 5(a): Divider and multiplier hardware cost

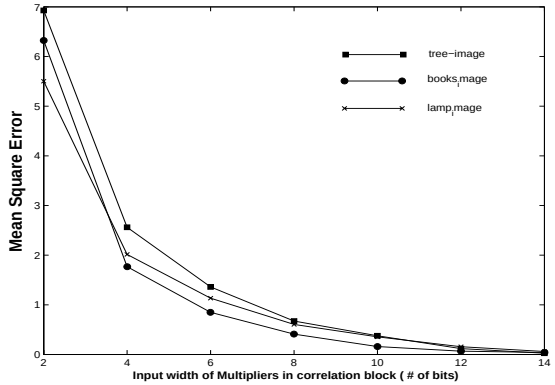


Figure 5(b): Error in final disparity resulted from fixed-point multipliers

3.2 Location of the Gaussian Window

The implementation of the correlation block based on the architecture of Figure 4 requires hardware resources beyond the capacity of a Virtex 2000E FPGA. To shrink the size of the circuit, we have modified the correlation block as shown in Figure 6. In this revised architecture we have placed the Gaussian window, *w*, at the end of voting function blocks. This change allows us to extract the common portions of the computations out of voting function units as much as possible. Extracting the two normalization blocks in Figure 6 leaves a simple inner product calculation and a single Gaussian window inside each voting function block. Each normalization block receives a complex-valued input and divides it by its magnitude such that the output has the same phase but with unity magnitude. Changing the location of Gaussian window along with sharing the normalization block reduces the total number of multipliers, dividers and square roots in the correlation unit by more than 65%.

The drawback of this architecture is that Gaussian window moved to the end of voting function block is not an exact analytical equivalent of the original method. But one can show that for FIR filters close to an impulse function, this is a reasonable approximation [7]. Since in our stereo vision system a 3-tap Gaussian LP filter is used as window, *w*, this trade off seems to be reasonable.

3.3 Normalization

For further reduction in the size of the phase-correlation

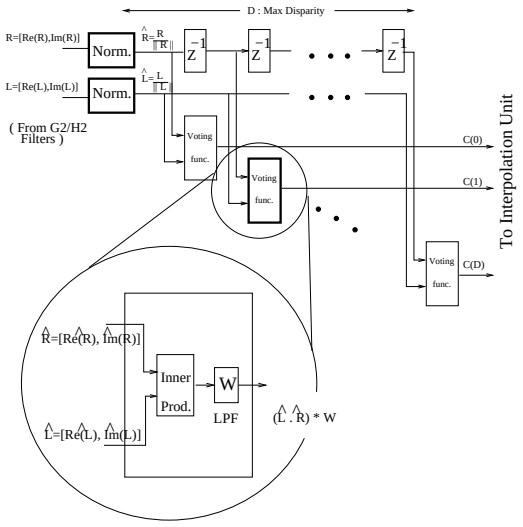


Figure 6: Revised correlation architecture with normalization blocks

unit, we have modified the architecture of normalization block. The major portion of this block is for computing the magnitude of complex-valued inputs. The magnitude of a complex number A is defined as the L2-norm of the 2D vector, A, with $Re(A)$ and $Im(A)$ as its elements:

$$\text{L2-norm:} \quad \|A\|_2 = \sqrt{Re(A)^2 + Im(A)^2} \tag{4}$$

The hardware implementation of $\|A\|_2$ is expensive because it requires two multipliers, one square root and one adder. Instead, we replace $\|A\|_2$ with L1-norm of vector A, $\|A\|_1$, defined as:

$$\text{L1-norm:} \quad \|A\|_1 = |Re(A)| + |Im(A)| \tag{5}$$

where $|x|$ is absolute value of x. Figure 7 shows the effect of replacing $\|A\|_2$ with $\|A\|_1$ on the normalized output. In L2 method, all the normalized vectors are located on the unit circle in the Real-Imaginary plane but in L1, they are projected on a square as shown in Figure 7. This is because the sum of absolute real and imaginary parts of normalized vectors are always unity and therefore they all become projected to straight lines which form a square instead of a unit circle. This technique provides enough accuracy for our application and can also be used in similar applications. To improve the accuracy and still avoid implementation of $\|A\|_2$, there is one possible solution: since $Re(A_{u1})$ and $Im(A_{u1})$ in Figure 7 are always limited between -1 and 1, one can use a memory block as a look up table with appropriate size to replace current A_{u1} vectors with those closer to unit circle. The important point is that these look up tables can be built using on-chip memory which is available in most of the current FPGA devices without the need to use logic elements of the device.

3.4 Optics

The camera head of the stereo system consists of two digital cameras with CCD size of 7.55mm x 6.45mm and using 12mm lenses. There is a separation of 70mm between the two cameras. In the design process of the stereo vision system, the size of the required hardware is directly proportional to maximum disparity between two images. We have set the maximum possible disparity to 20 pixels. This maximum disparity becomes translated to a minimum allowed distance of about 2m from the objects to the camera head. Other parameters such as separation between the cameras, CCD size and resolution can reduce the minimum distance, but they are usually restricted by optical and physical constraints. For example, to reduce the minimum distance, wider lenses can be used but lenses with focal length of less than 8 mm introduce fisheye distortion in which straight lines in the scene are no longer straight in the image. It should be noted that the dynamic range of the depth measurements can be expanded by image pre-warping. This pre-warping can be done either statically for all the frames or dynamically with varying pre-shift values depending on the current distance.

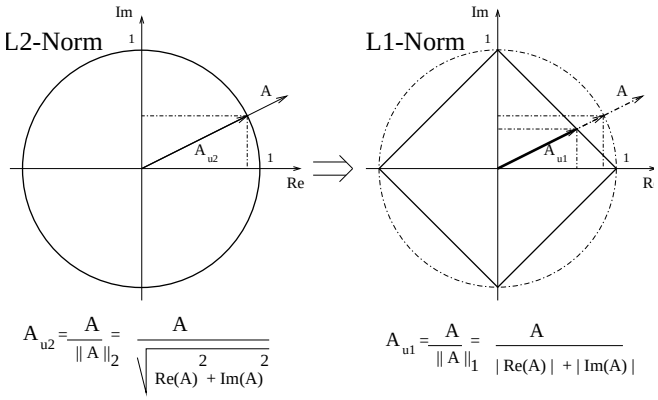
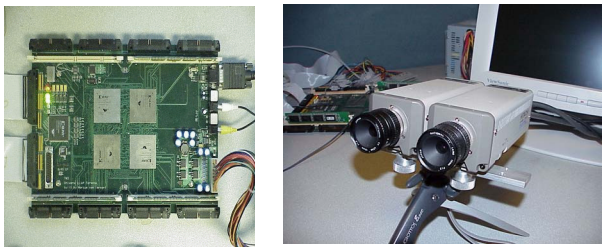


Figure 7: Effect of using L1-norm instead of L2-norm

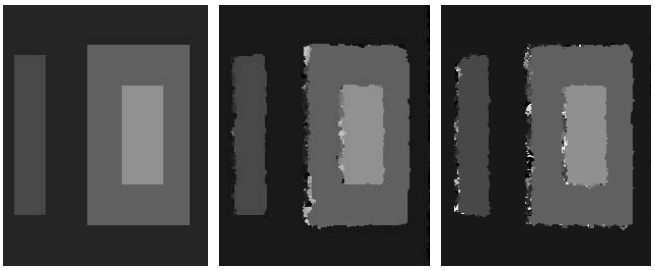


(a) (b)

Figure 8:(a) TM-3A board, (b) Camera head

4 Performance

The FPGA stereo system developed in this research performs multi-resolution, multi-orientation depth extraction based on local weighted phase-correlation. It can produce a dense disparity map of size 256 x 360 pixels with 8-bit subpixel accuracy disparity results at the rate of 30 frames/sec. In the metric of Points x Disparity measures per second (PDS), this system achieves a performance of 60 million PDS, which is among the highest rates reported [29]. While the PDS metric reflects the density and the speed of the system, it does not reflect its complexity and accuracy of the implemented algorithm. The important feature of this system

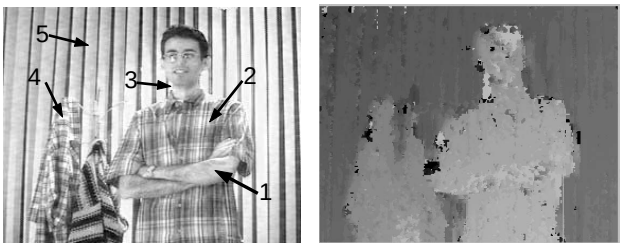


(a) (b) (c)

Figure 9: Overall system performance on random stereograms: (a) true depth map (b) depth from original software (c) depth from hardware

in comparison with other hardware stereo machines is its high performance complex phase-based algorithm. To realize a phase-based algorithm in video rate, the system performs the equivalent of more than 10 billion 16 x 16 bit multiplications per second and the four Virtex devices communicate in a data rate of up to 200 Mbytes/sec. Figure 9 shows the depth map from a pair of random stereograms extracted by the original phase based algorithm and also by the hardware system. Figure 10 shows a sample image from the camera head and the depth map generated by the hardware. The depth maps in these figures are the output of the Peak-Detection unit without any post-processing such as left-to-right/right-to-left validation or smoothing/gap filling. We believe that rectification and left-to-right/right-to-left validation can be integrated with the current system by approximately half of the size of one Xilinx Virtex2000E FPGA. As an other solution, the rectification can be avoided as pre-rectified images can be directly generated using special optical setups [15]. Also one efficient technique to integrate the left-right, right-left validation feature to the current system is to alternate between left and right images after each subsequent frame. Since left-right and right-left matching is performed in different time slices, they can share the same blocks and hence there is no need for extra hardware. Even if we want to perform both matchings at the same time slice we can still share the filtering blocks and just need to re-do the correlation and peak detection separately.

In Table 2, the depth measurements obtained by our system are compared with the actual depths. As in most of the other stereo matching algorithms, the phase-based stereo matching algorithm is sensitive to depth discontinuity and lack of texture. In the regions with enough texture, most of the depth measurement errors are less than 5%, but in the regions with fine texture that is not sensed by the cameras or at the depth discontinuities, the depth values are less reliable (for example, point 3 in Figure 10(a)).



(a) (b)

Figure 10: (a) Input from right camera (b) Output disparity map

Table 2: Depth measurements for five points of figure 10(a)

Point #	1	2	3	4	5
Ground truth depth (cm)	300	315	320	354	396
Depth from stereo system (cm)	309	320	276	355	402

To compare with other stereo systems, a number of fast algorithms that do not use reconfigurable hardware also exist in the literature. Some of these take advantage of special hardware, specifically SIMD (single instruction, multiple data) instructions in Intel MMX processors [17][22]. A number of intensity-based cross-correlation techniques are reported [17][22][23][24][27]. Approaches to speed up algorithms include the use of image pyramids [24][21] or simplified algorithms [3].

The two fastest algorithms both use MMX processors. Hirschmuller et al. [17] achieve 4.7 frames/second on 320 x 240 images using intensity correlation in 7 x 7 windows with Gaussian pre-filtering. Their method includes both rectification and left-right consistency checking (without these the frame rate is estimated at 5.7 frames/second). Their hardware is a 450MHz Pentium running Linux. Muhlmann [22], using an 800MHz MMX processor, achieves less than 5 frames/second on 348 x 288 colour images, again using intensity correlation. A trinocular disparity system is implemented by Mulligan et al. [23] on a 4-processor system utilizing Intel IPL libraries. This system takes 456 ms to compute disparity for three 320 x 240 images up to a maximum disparity of 64 pixels. Sun [24] computes disparity (up to 10 pixels) in 450ms on a 500MHz Pentium processor based on fast intensity correlation in an image pyramid.

Birchfield & Tomasi [3] use a simplified algorithm that minimizes a 1-D cost function based on pixel intensity differences. They report speeds of 4 seconds/frame on a 333 MHz Pentium for 640 x 480 images. Given a faster processor their algorithm would doubtlessly perform better than one frame/second. It has the advantage of explicitly accounting for occlusion as well as propagating disparity information between scan lines.

In summary it should be seen that our system compares favorably with the systems described. While some of the systems are starting to achieve frame rates upwards of 5 frames/second, none of them are doing the amount of computation that our system is. For example, while others are using correlation methods, none are computing complex filter responses at multiple scales and orientations, and then recovering and correlating the phase responses. While others are using image pyramid approaches, they use the pyramid to simplify computation at subsequent scales, as opposed to combining the information across scales. Finally, while we have attempted to compare running speeds of other algorithms based on the same size of image that our system currently processes, it must be mentioned that we are able to handle

larger images at 30 frames/second if the algorithm is moved to larger FPGAs. Our speed is thus dependent on the processing capacity of the FPGA, and not its clock speed.

5 Conclusion

This paper has shown the feasibility of implementing challenging vision applications on FPGAs to achieve real-time performance. After describing programmable hardware and also the theory of the Local Weighted Phase-Correlation algorithm, it illustrates the design techniques and the trade-offs for fast and efficient hardware implementation. Although this paper is specifically focused on phase-based stereo vision FPGA realizations, most of the design issues are common among other DSP and video applications. In fact, most of the components of this system can be re-used in other vision algorithms. Prefiltering and building image pyramids are basic operations used by many algorithms doing tasks such as stereo, motion analysis and object recognition & localization. In particular, our implementation of steerable filters has great potential for re-use. Examples of algorithms based on these filters include optic flow computation [11][9], object tracking [19] and recovery of rotation and illumination invariant features [5], which can be used for feature tracking or perhaps object recognition. The implementation of the correlation units can be adapted for use in other systems that require correlation, such as template matching.

6 Acknowledgment

The authors would like to thank Micronet and NSERC Canada for their financial support.

7 References

[1] N. Ayache and B. Faverjon, Efficient registration of stereo images by matching graph descriptions of edge segments, *IJCV, 1(2)*:107:131, 1987.

[2] H.H. Baker and T.O. Binford, Depth from edges and intensity based stereo, *Proc. 7th Intern. Joint Conf. Artif. Intell.*, Vancouver, pp. 631-636, August 1981.

[3] S. Birchfield and C. Tomasi, Depth discontinuities by pixel-to-pixel stereo. *IJCV*, 35(3):269-293, 1999.

[4] S. Brown, R. Francis, J. Rose, Z. Vranesic, *Field-Programmable Gate Arrays*, Kluwer Academic Publishers, May 1992.

[5] G. Carneiro and A. D. Jepson, Local phase-based features. *In Proceedings of the 2002 European Conference on Computer Vision*, v. 1, pp. 282-296, Copenhagen, Denmark, 2002.

[6] M.L. Chang and S. Hauck, Precis: A design-time precision analysis tool, *IEEE FCCM*, April 2002.

[7] A. Darabiha, Video-Rate Stereo Vision on Reconfigurable Hardware, *Master's thesis, University of Toronto*, 2003.

[8] O. Faugeras, et al., Real time correlation based stereo: algorithm, implementations and applications, Research Report 2013, INRIA Sophia-Antipolis, 1993.

[9] D.J. Fleet, M. J. Black, and A. D. Jepson, Motion feature detection using steerable flow fields, *CVPR*, pp. 274-281, June 1998.

[10] D.J. Fleet, A.D. Jepson and M. Jepson, Phase-based disparity measurement, *CVGIP: Image Understanding*, 53(2):198-210, 1991.

[11] D.J. Fleet and A. D. Jepson. Computation of component image velocity from local phase information. *IJCV*, 5(1):77-104, 1990.

[12] D.J. Fleet, Disparity from local weighted phase-correlation, *Int. Conf. on Systems, Man, and Cybernetics*, pp. 48-54 v.1, 1994.

[13] D.J. Fleet and A.D. Jepson, Stability of phase information, *IEEE Trans. PAMI*, 15(12):1253-1268, 1993.

[14] W.T. Freeman and E.H. Adelson, The design and use of steerable filters, *PAMI, Trans. on*, Volume: 13, Issue: 9, Page(s): 891 - 906, Sept. 1991.

[15] J. Gluckman and S.K. Nayar, Rectified catadioptric stereo sensors. *IEEE Transactions on PAMI*, 24(2):224-236, February 2002.

[16] M. Van Ierssel, D. Galloway, P. Chow, J. Rose, The Transmogrifier-3a: Hardware and Software for a 3 Million Gate Rapid Prototyping System, *Micronet Annual Workshop*, 2001.

[17] H. Hirschmuller, et al., Real-time correlation-based stereo vision with reduced border errors. *IJCV*, 47(1/2/3):229-246, 2002.

[18] K.M. Hou, et al., A reconfigurable and flexible parallel 3D vision system for a mobile robot, *Proc. Computer Architectures for Machine Perception*, pp. 215 - 221, Dec 1993.

[19] A.D. Jepson, et al., Robust on-line appearance models for visual tracking, *CVPR*, vol. 1, pp. 415-422, Kauai, Dec 2001.

[20] F. Jutand, et al, ENSTA Single chip VLSI architecture for a real time stereo vision processor, *ICASSP-88.*, pp.1965-1968, vol. 4, 1988.

[21] G. Van Meerbergen, M. Vergauwen, M. Pollefeys, and L. Van Gool. A hierarchical symmetric stereo algorithm using dynamic programming, *IJCV*, 47(1/2/3):275-285, 2002.

[22] K. Muhlmann, D. Maier, Jurgen Hesser, and R. M. Anner, Calculating dense disparity maps from color stereo images, an efficient implementation. *IJCV*, 47(1/2/3):79-88, 2002.

[23] J. Mulligan, et al, Trinocular stereo: A real-time algorithm and its evaluation, *IJCV*, 47(1/2/3):51-61, 2002.

[24] C. Sun. Fast stereo matching using rectangular subregioning and 3d maximum-surface techniques, *IJCV*, 47(1/2/3):99-117, 2002.

[25] TM-3 documentation, *http://www.eecg.utoronto.ca/~tm3/*

[26] E. Trucco, A. Verri, *Introductory Techniques for 3-D Computer Vision*, Prentice Hall, 1998.

[27] O. Veksler. Stereo correspondence with compact windows via minimum ratio cycle, *IEEE Trans. on PAMI*, 24(12):1654-1660, Dec 2002.

[28] Xilinx data sheets, *http://direct.xilinx.com/bvdocs/publications/ds022-1.pdf*

[29] J. Woodfill and Von Herzen, Real-time stereo vision on the PARTS reconfigurable computer, *The 5th Symposium on FCCM Proceedings*, Pages:201-210, 1997.

[30] R. Zabih and J. Woodfill, Non-parametric Local Transforms for Computing Visual Correspondence, *Proceedings of 3rd European Conf. on Computer Vision*, pp. 150-158, May 1994.