

# Expanding Disparity Range in an FPGA Stereo System While Keeping Resource Utilization Low

Divyang K. Masrani & W. James MacLean  
Department of Electrical and Computer Engineering  
University of Toronto  
Toronto, ON, M5S 3G4  
{masrani|maclean}@eecg.toronto.edu

## Abstract

*Field Programmable Gate Array devices (FPGAs) are attractive for implementing computer vision algorithms due to the fact that they allow the designer to exploit the parallel nature of many vision algorithms, thus offering substantial speed improvements over fixed hardware/software systems. One major drawback, however, is that adding resources to reconfigurable devices is not as simple as adding memory or upgrading a hard disk. This becomes a problem when adapting a system to deal with larger image sizes or, in the case of stereo disparity estimation, larger disparity ranges (parameter ranges in general). This paper describes preliminary work on a design to expand the disparity range and input image size of an existing FPGA-based stereo system [8] while not significantly expanding resource requirements. Simulations on synthetic and real images are given, as well as a description of the proposed architecture changes in porting the system to a newer FPGA architecture.*

## 1. Introduction

Stereo disparity estimation is a prime application for embedded computer vision systems. Since stereo can provide depth information, it has potential uses in navigation systems, robotics, object recognition and surveillance systems, just to name a few. Due to the computational complexity of many stereo algorithms, a number of attempts have been made to implement such systems using hardware [3, 13, 19, 25], including reconfigurable hardware in the form of FPGAs [9, 14, 26, 17, 24, 8]. In related work, [1] implements Lucas & Kanade optic flow using FPGAs. Solutions based on reconfigurable hardware have the desirable property of allowing the designer to take advantage of the parallelism inherent in many computer vision problems, not the least of which is stereo disparity estimation.

While designing with FPGAs is faster than designing Application Specific ICs (ASICs), it suffers from the prob-

lem of fixed resources. In an application based on a serial CPU or DSP, one can typically add memory or disk space to allow the algorithm to handle a larger version of the same problem, for example larger image sizes or increased disparity ranges in the case of stereo. System performance may suffer, but the new system still runs. In the case of FPGA-based systems, each device has a finite amount of logic available, and when this is exhausted the only solutions are to add another device, or explore different system architectures. Adding another device can be costly not only with respect to the design of the hardware itself, but involves the additional design issue of how to partition the logic across several devices. When splitting a design across multiple devices there is overhead for the circuitry required to do the I/O operations. Since devices have a fixed number of connections, and printed circuit boards have finite space for signal traces, it may be necessary to multiplex the transfer of large data blocks, resulting in bandwidth concerns and additional overhead imposed by the multiplexor circuitry. Exploring alternative architectures is an attractive option, and is the focus of this paper.

In this paper we present preliminary work on modifying an existing FPGA-based stereo system [8] to accommodate larger images and increased disparity range, but without requiring larger resource utilization. The stereo algorithm, based on phase correlation [11], is extended to have a larger disparity range through the use of a shiftable correlation window that tracks the disparity estimate for each pixel, as well as a roving correlation window that explores the correlation surface outside the range of the tracking window in order to detect new matches when the shiftable window is centred on an incorrect match. This trades off resource utilization for time response, as the roving window may take several frames to explore the desired range. The basic assumption is that, in most cases, disparity values do not change radically between frames, thus allowing some of the computation to be spread over time.

The modified system will be implemented on a newer, state-of-the-art FPGA board which has over twice the num-

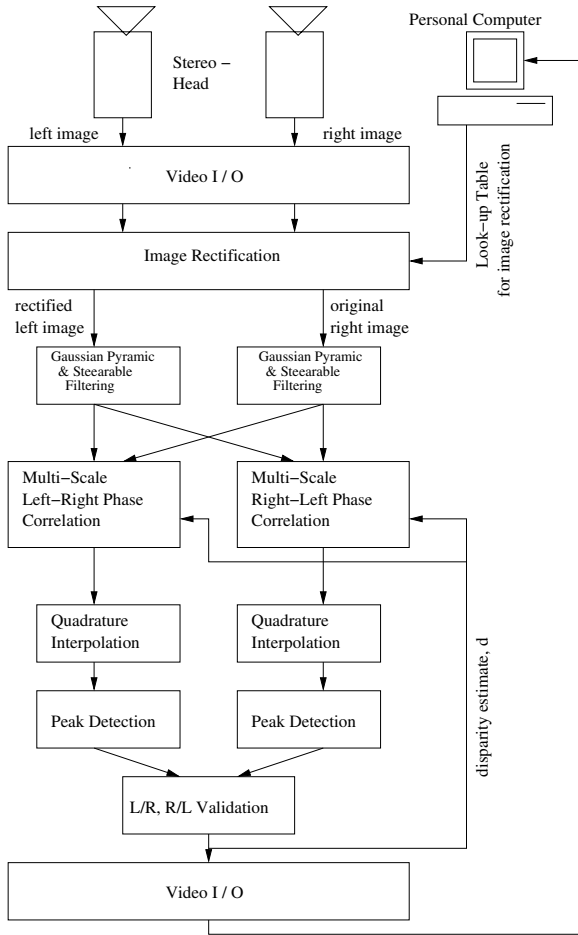


Figure 1: High-level architecture of the stereo system

ber of programmable logic blocks [4] and can be clocked up to 150 MHz to 200 MHz compared to 50 MHz for the previous system. In comparison to the 2.5Kb of on-chip and 8MB of off-chip memory on the previous system, the newer system has 29.6Mb of on-chip and 4GB of off-chip memory. The large amount of on-chip memory, which can be clocked over 250 MHz [5], can be used to buffer images and store look-up tables to avoid unnecessarily using precious logic blocks.

### 1.1. Previous Work

A variety of reconfigurable stereo machines have been reported [24, 17, 26, 9, 14]. The PARTS reconfigurable computer [24] consists of a  $4 \times 4$  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 bit-wise comparisons and additions [26]. Kanade *et al.* [17] describe a hybrid system using C40 digital signal processors

together with programmable logic devices (PLDs, similar to FPGAs) mounted on boards in a VME-bus backplane. The system, which the authors do not claim to be reconfigurable, implements a sum-of-absolute-differences along predetermined epipolar geometry to generate 5-bit disparity estimates at frame-rate. In Faugeras *et al.* [9], a  $4 \times 4$  matrix of small FPGAs is used to perform the cross-correlation of two  $256 \times 256$  images in 140 ms. In Hou *et al.* [14], a combination of FPGA and Digital Signal Processors (DSPs) is used to perform edge-based stereo vision. Their approach uses FPGAs to perform low level tasks like edge detection and uses DSPs for higher level integration tasks. In [8] a development system based on four Xilinx XCV2000E devices is used to implement a dense, multi-scale, multi-orientation, phase-correlation based stereo system that runs at frame-rates. It is worth noting that not all previous hardware approaches have been based on reconfigurable devices. In [18], a DSP-based stereo system performing rectification and area correlation, called the SRI Small Vision Module, is described. ASIC-based designs are reported in [22, 3] and in [25] commodity graphics hardware is used.

## 2. Expanding the Disparity Range

Our previous system is based on the Local-Weighted Phase Correlation (LWPC) algorithm [11], and its implementation is described in [8]. A simple and straightforward solution to expanding the range can be achieved by increasing the size of the correlation window in [8], but considering that the resource usage on the FPGAs is proportional to the size of the correlation window, it is not an optimal solution because finite resources pose a restriction. In addition, in [20] it is shown that the probability of an incorrect stereo match is proportional to the epipolar search area considered during matching. Keeping this in mind, we have modified the original LWPC algorithm to encapsulate the correlation algorithm within a temporal loop, where the disparities from the previous frame are used as the centre of the correlation window in the current frame. This correlation window has the ability to move along the epipolar line and its location is dependent on the disparity estimate at the previous frame. Since images are received at a rate of 30 frames per second, we assume that the disparity of a given pixel will not change drastically from one frame to the next. This allows us to reduce the size of the correlation window, which results in reduced hardware as well as a reduced probability of mismatch assuming we are close to the correct match. Details of the LWPC algorithm can be found in [11, 8]; our modification affects Step 2, which is revised here to include the use of a shiftable correlation window and the modified high-level system architecture is shown in Figure 1:

2. For each scale and orientation, compute local voting

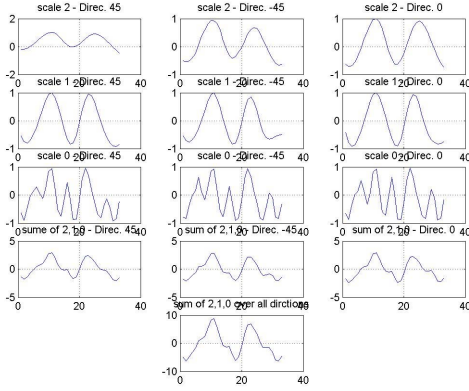


Figure 2: LWPC tends to cancel erroneous matches across scales, leaving only the true match after voting.

functions  $C_{j,s}(x, \tau)$  in a window centred at  $\tau_c$  as

$$C_{j,s}(x, \tau) = \frac{W(x) \otimes [O_l(x)O_r^*(x + \tau)]}{\sqrt{W(x) \otimes |O_l(x)|^2} \sqrt{W(x) \otimes |O_r(x)|^2}}, \quad (1)$$

where  $W(x)$  is a smoothing, localized window and  $\tau$  is the pre-shift of the right filter output centred at the disparity of the pixel from the previous frame.

When propagating disparity estimates between frames, it is necessary to consider that such algorithms suffer from the risk of getting stuck in a local minima (wrong matches) [7], especially during the initial few frames. We have employed an initialization stage to obtain an accurate disparity map. In [6] it is claimed that a coarse-to-fine strategy is preferred over an initialization stage that uses a window to incrementally search over a wider range, but from our experiments on real image sequences we have found that a secondary correlation window that performs disparity calculations at regularly spaced intervals in successive frames, similar to the initialization stage employed in [7], gives good results. Disparity calculations are performed at three scales and in three orientations, the results of which are summed together to strengthen the value of the confidence measure function at the correct peak and cancel any false peaks. This is a built-in error-canceling feature of the algorithm to avoid false matches, and is shown in Figure 2. This feature can be thought of as a coarse-to-fine scheme that considers multiple paths through the scale hierarchy, and selects the best overall path.

An initialization window is preferred over a static coarse-to-fine strategy because the hardware of this correlation window is an exact replica of the regular correlation window used in the original system. Another advantage of this correlation window is its' ability to aide in recovery

from a mismatch *after* the initialization stage. Currently both windows are 9 pixels wide, and the roving secondary correlation window centres at a disparity of 10 for the first frame, and in increments of 10 for the successive frames until it reaches a disparity of 70 pixels. At the next frame, the window centres at 0 disparity, after which it circles back to being centred at 10 and the cycle continues (thereby having a worst-case recovery time of 233 milliseconds). This means that the effective range of disparity that our system can handle is 70 pixels, but this can easily be increased to 100 pixels or even more. The drawback of increasing the disparity range would be an increase in the recovery time should a pixel's disparity estimate get stuck in a local minima. The system is designed such that the maximum range and increment value of this secondary window are user configurable at run-time without the need for reconfiguring the hardware. A decrease in increment value would result in searching overlapping regions from one frame to the next. This allows a user to decide on the trade-off between range capability and recovery time.

In summary, we have made two changes to the original local weighted phase-correlation algorithm, which enable us to achieve a greater range of disparity calculation while keeping the hardware resource usage to slightly less than in the original implementation. The first is the use of a tracking correlation window that uses temporal information from the previous time frame. The second is the use of a secondary correlation window that searches at regularly spaced intervals as previously discussed. The total correlation window size is 18 pixels at the finest scale and can handle a disparity of 70 pixels (or more) in comparison to the original implementation that uses a correlation window of 20 pixels at the finest scale and thus only can handle a disparity of 20 pixels.

### 3. Simulation Results

Simulation results from two real, pre-rectified sequences are presented here and compared with the original local weighted phase-correlation algorithm with a fixed upper disparity limit. The first, MDR-1, is a scene with a static camera and a moving person, and has a maximum disparity of around 16 pixels. The second, MDR-2, is a more complex scene with a moving person and a moving camera, and has a maximum disparity of approximately 30 pixels.

The simulation is performed in MATLAB on  $320 \times 240$  images using double-precision floating-point arithmetic. Results from our modified algorithm which incorporates temporal information are compared against the previous algorithm. The hardware is being designed to handle an image size of  $640 \times 480$  pixels with fixed-point precision. An analysis of the optimal fixed-point widths was done in [8] and will be retained in the hardware implementation of our

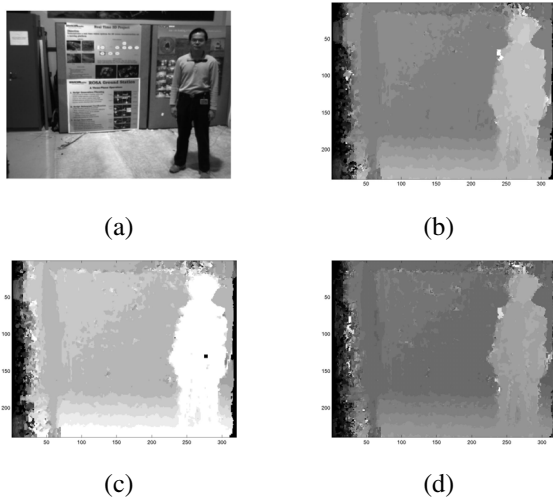


Figure 3: In sequence MDR-1, we see that the proposed range-expansion algorithm (d) matches the original algorithm (b) by frame 2. The first frame from the range-expansion algorithm is shown in (c).

system.

Frame 2 of the MDR-1 sequence is shown in Figure 3, and the disparity map from the fixed-range LWPC algorithm (for a range of 20 pixels) is shown in Figure 3 (b). The corresponding disparity map from the temporal algorithm is shown in Figure 3 (d). It can be seen that our temporal algorithm gives the same results as the original algorithm once the system has settled into the global minimum. The settling process is shown in Figure 3 (c) for frame 1. For this particular sequence the algorithm settles into the global minimum by the second frame.

In Figure 4 we show the difference in recovery time for the cases when the secondary correlation window is shifted up to a disparity of: i) 70 pixels and ii) 30 pixels. Figure 4 (a) shows frame 11 for case (i); the results start to deteriorate but are completely recovered by frame 15, Figure 4 (b). For case (ii), the results deteriorate at frame 12, Figure 4 (c), and are already recovered by frame 13, Figure 4 (d). In the MDR-1 sequence, we know that the maximum disparity is around 16 pixels and in such cases where we have prior knowledge of the scene, the ability of the end-user of the system to select the maximum disparity parameter can yield better results.

The disparity maps from the MDR-2 sequence for frame 4 (Figure 5 (a)) are shown in Figure 5 (b) for the original implementation and Figure 5 (c) for the temporal algorithm. The original implementation, when limited to a maximum disparity of 20 pixels, is clearly inadequate in this case, while the modified system shows promising results. With the camera and the person both in motion, our as-

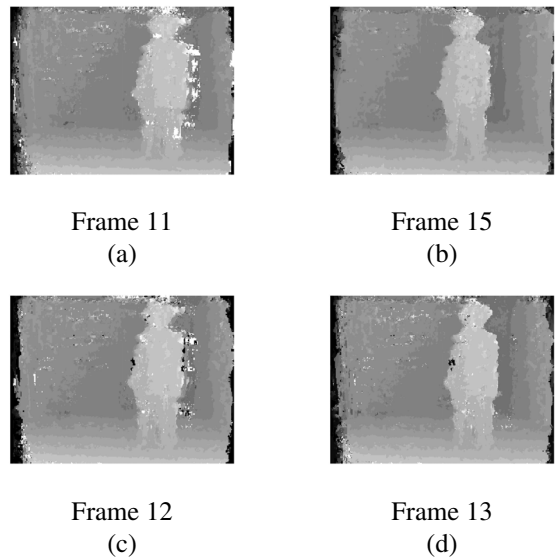


Figure 4: The recovery time for the system with a maximum secondary shift of 70 pixels is shown in (a) and (b). This can be reduced by using a smaller maximum shift, *e.g.* 30 pixels as shown in (c) and (d). In the latter case, recovery occurs in one frame as opposed to four.

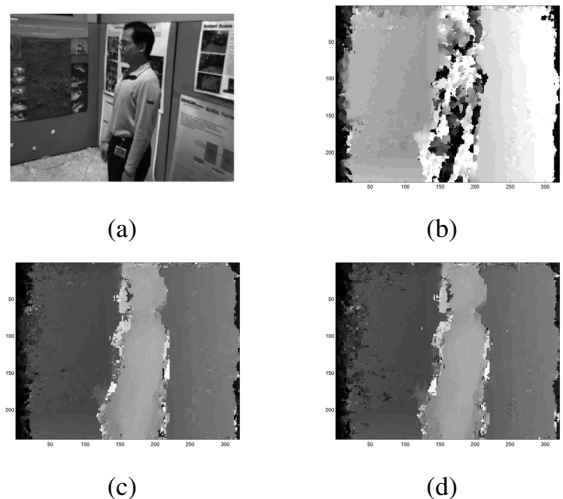


Figure 5: In sequence MDR-2, we see that the proposed range-expansion algorithm (c) performs significantly better than the original algorithm (b). The disparity map using a larger primary correlation window of 13 pixels (d) is a slight improvement over (c).

sumption of a pixel not changing in disparity by more than 4 pixels (one-half the tracking window width) in a single frame does not hold for every pixel. The disparity map near the depth discontinuities, namely the edge of the person, is inaccurate due to significant occlusion between the left and right images as the person and camera move. The occlusion affected pixels can easily be identified by left/right - right/left validation that is planned in the current redevelopment of the system (see Section 4). In Figure 5 (d), the disparity map obtained with an increased primary correlation window of size 13 (i.e. ability to handle motion of 6 pixels on either side of the centre position) is shown. The disparity map is slightly improved over that of the smaller correlation window, while keeping the hardware resource usage almost the same. To achieve improved results when tracking is lost without a significant increase in hardware, we are currently working on an improved secondary correlation window as explained in Section 4.

## 4. Further Improvements

As this is an on-going work, we are considering further improvements to the algorithm to increase the scenarios in which the system will perform robustly and return accurate results. One simple and straightforward solution is to increase the size of the tracking correlation window. This increases resource requirements, but provides the capability to handle larger pixels shifts between frames. The new size of the tracking window will depend on the expected maximum pixel shift per frame.

It is possible to improve the secondary correlation window so that it is centred intelligently along the epipolar line instead of at pre-determined locations. The approaches mentioned in [21], [16], and [2] require the implementation of a Kalman filter or particle filter (respectively) for each pixel in order to acquire a dense disparity map. The hardware requirement can be reduced to one for each column in the input image because the processing is done one scan line at a time on pre-rectified images, but state information must still be stored for all pixels. This still proves very costly in terms of hardware resources. A simplified approach can be considered where the roving correlation window is first centred around the disparity of neighbouring pixels using a simple constant-velocity motion model. A disparity velocity estimate can be obtained by taking the difference between disparities in the previous two frames,  $v_t = d_{t-2} - d_{t-1}$  where,  $v_t$  is the predicted disparity velocity for the current frame. This only requires the storage of an extra set of disparity values, and subtraction units equal to the number of columns in the image.

Another possibility is computing the secondary correlation in a coarse-to-fine strategy as in [6]. We can compute the correlation at the two coarser scales (25% and 50%

of the image) and retain the voting scheme in order to retain the error canceling properties of the algorithm. The resource savings from eliminating correlation at the finest scale can be utilized to increase the width of the correlation window at the coarser scales. This can be used in conjunction with a right-left/left-right validation scheme [12] to select the disparity values from either the tracking or secondary correlation window. If the disparity value for a particular pixel is validated, then the value from the tracking correlation window is selected, else the value from the secondary correlation window is selected. This value is then used as an input to the temporal loop for shifting the primary correlation window. This, in effect, performs an on-the-fly re-initialization.

The increased capacity of our newer FPGA board also allows us to improve accuracy over the current system by increasing the fixed point precision and using an L2 norm instead of an L1 norm [8]. We have also included an image rectification module at the front-end of the system. But the core focus of this work remains that of achieving a robust stereo algorithm with a greater disparity range without significantly increasing hardware resources by improving the correlation module.

## 5. Proposed Architecture

The system will be implemented on the Transmogripher-4 (TM-4) board whose architecture and layout is very similar to the TM-3 board on which the original algorithm was implemented [15], [10]. The general architecture of the original implementation will be retained where Video I/O, Scale/Orientation Decomposition, Phase-Correlation, and Interpolation/Peak Detection will each be implemented on separate FPGAs. The TM-4 board utilizes four Altera Stratix S80 FPGAs, each of which provides just over twice the number of logic cells as that of the Xilinx Virtex XVC2000E on the TM-3 [4]. This will allow us to replicate the phase-correlation unit for right-left/left-right stereo depth validation and also allow us to utilize three orientations instead of two as on the original system.

A rectification block is also included in our design just after the video acquisition block. The incoming stereo pairs are rectified (using bilinear interpolation [23]) which often causes missing pixels at the edges in one image. A 1-bit flag is created for each pixel to indicate whether a pixel is valid or not. This will be used to avoid correlating with invalid pixels during the phase-correlation stage. A stereo-setup with a worst-case vertical misalignment of 32 rows between the left and right image is assumed. In terms of hardware resources this will require buffering of 64 rows of both the left and right image, which will consume approximately 2 of 9 M-RAM blocks on FPGA #0. Since four pixels are needed for the computation of each rectified pixel,

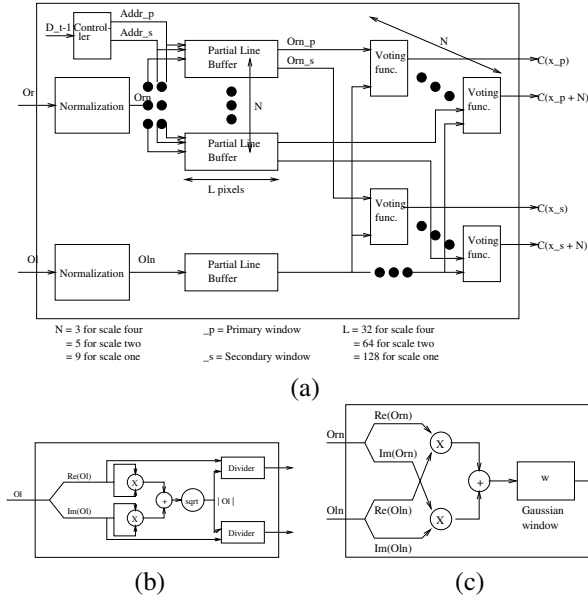


Figure 6: Modified Phase Correlation unit with partial line buffer (a), Normalization unit shown for left image data (b), and Voting Function unit (c)

a valid output results only every fourth cycle effectively reducing the output rate by a factor of 4. This should not pose a problem because preliminary analysis performed on individual modules (Image Rectification module, Scale Decomposition module, and Orientation Decomposition module) shows that each of these is capable of running over 160 MHz, which corresponds to an output rate of 40 MHz. This is well above the required 10 MHz which corresponds to the input rate of a 640x480 image with 8 bits per pixel. With the various modules interconnected and the limitations on chip-to-chip communication, a conservative estimate yields the system running at 80 MHz, an effective output rate of 20 MHz which is still twice the input rate.

The main architectural change will be in the phase-correlation block to include the temporal information. This means that the pixels cannot follow a regular shifting pattern as in [8], but rather need to be stored in high-bandwidth M4K on-chip RAM blocks and accessible by pixel location. The architecture of the phase-correlation block is shown in Figure 6 (a). A 16-bit signed resolution is retained from the Scale/Orientation decomposition block which is then fed into the normalization module. An 8-bit wide signed output is retained after normalization and this data will be buffered for correlation. For left-right correlation (where the correlation is performed with reference to the left image), multiple copies, three for scale 4, five for scale 2, and nine for scale 1, of the rows in the right image need to be buffered and a single copy of the rows in the left image. Similar buffering

Unit	# of Blocks	Altera LEs	On-chip M4K RAM	9-bit DSP Elements
Normalization	18	29,070	-	72
Two row buffers	36	-	300	-
Voting function	204	43,452	-	-
S80 Available Resources		79,040	364	172

Table 1: Estimate of resource consumption of modified Phase Correlation module on an Altera S80 FPGA.

is needed for right-left correlation. The normalization unit (Figure 6 (b)) consists of two 16-bit multipliers, one adder, one square root function, and two dividers. Two normalization units are required per phase-correlation block for a total of 18 units across three scales and three orientations. The voting function unit (Figure 6 (c)) consists of two 8-bit multipliers, an adder, and a low-pass filter. Seventeen voting function units are required per orientation for a total of 204 units across three orientations, two correlation windows (primary and secondary), and two correlation modes (left-right and right-left correlation). An estimate of the resource consumption shows that 72,522 LEs out of 79,040 LEs available, 300 M4K RAM blocks out of 364 blocks available, and 72 9-bit DSP Elements out of 176 available on the Stratix S80 FPGA are used. (Note : The LE count for the voting function block does not include the low-pass filter as this will be done on a separate FPGA) A breakdown of the estimate for the phase-correlation module is given in Table 1.

The TM-4 board also provides 1 GB of off-chip DDR SDRAM connected to each FPGA for a total of 4GB which can be accessed at 133 MHz [10]. Thus larger image sizes can easily be accommodated by storing the images and look up tables used in the design in the off-chip memory rather than the on-chip memory. The on-chip memory, which can be clocked up to 250 MHz, can be used to cache the blocks of the images and look-up tables that are currently being processed.

## 6. Summary and Conclusions

We have presented preliminary work that takes an existing FPGA-based stereo system, and increases its disparity range without increased resource utilization. Although the re-design effort is still in progress, simulations on synthetic and real data demonstrate the potential of the new system.

## Acknowledgments

The author would like to thank the following for valuable discussions related to this work: Jonathan Rose, Ahmed Darabiha, and Allan Jepson.

## References

- [1] Javier Díaz Alonso. Real-time optical flow computation using FPGAs. In *Proceedings of the Early Cognitive Vision Workshop*, Isle of Skye, Scotland, June 2004.
- [2] Sam P. Bromley, John S. Zelek, and Robert D. Dony. Commodity real-time stereo-vision for navigation. In *Proceedings of the First Canadian Conference on Computer and Robot Vision*, pages 424–431. IEEE Computer Society, May 2004.
- [3] Peter J. Burt. A pyramid-based front-end processor for dynamic vision applications. *Proceedings of the IEEE*, 90(7):1188–1200, July 2002.
- [4] Altera Corporation. Device compare. [http://www.altera.com/cgi-bin/device\\_compare.pl](http://www.altera.com/cgi-bin/device_compare.pl), 2003.
- [5] Altera Corporation. Stratix devices. <http://www.altera.com/products/devices/stratix/stx-index.jsp>, 2003.
- [6] S. Crossley, A. J. Lacey, N. A. Thacker, and N. L. Seed. Benchmarking of bootstrap temporal stereo using statistical and physical scene modelling. In *Proceedings of the British Machine Vision Conference*, pages 346–355, 1998.
- [7] S. Crossley, N. A. Thacker, and N. L. Seed. Robust stereo via temporal consistency. In *Proceedings of the British Machine Vision Conference*, pages 659–668, 1997.
- [8] Ahmad Darabiha, Jonathan Rose, and W. James MacLean. Video-rate stereo depth measurement on programmable hardware. In *Proceedings of the 2003 IEEE Computer Society Conference on Computer Vision & Pattern Recognition*, volume 1, pages 203–210, Madison, WI, June 2003.
- [9] Olivier Faugeras, Bernard Hotz, Hervé Mathieu, Thierry Viéville, Zhengyou Zhang, Pascal Fua, Eric Théron, Laurent Moll, Gérard Berry, Jean Vuillemin, Patrice Bertin, and Catherine Proy. Real time correlation-based stereo: Algorithm, implementations and applications. Technical Report Research Report 2013, INRIA Sophia Antipolis, August 1993.
- [10] Josh Fender. Transmogripher 4 preliminary information. <http://www.eecg.toronto.edu/~fender/tm4/sointroduction.shtml>, August 2003.
- [11] David J. Fleet. Disparity from local weighted phase correlation. In *International Conference on Systems, Man and Cybernetics*, volume 1, pages 48–54, 1994.
- [12] Pascal Fua. A parallel stereo algorithm that produces dense depth maps and preserves image features. *Machine Vision & Applications*, 6(1):35–49, 1993. Inria Research Report 1369.
- [13] Heiko Hirschmüller, Peter R. Innocent, and Jon Garibaldi. Real-time correlation-based stereo vision with reduced border errors. *International Journal of Computer Vision*, 47(1/2/3):229–246, 2002.
- [14] K. M. Hou and A. Belloum. A reconfigurable and flexible parallel 3d vision system for a mobile robot. In *IEEE Workshop on Computer Architecture for Machine Perception*, New Orleans, Louisiana, December 1993.
- [15] Marcus Van Ierssel, David Galloway, Paul Chow, and Jonathan Rose. The Transmogripher 3-a: Hardware and software for a 3 million gate rapid prototyping system. In *Micronet Annual Workshop*, 2001.
- [16] M. Isard and A. Blake. Condensation – conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29(1):5–28, 1998.
- [17] Takeo Kanade, Atsushi Yoshida, Kazuo Oda, Hiroshi Kano, and Masaya Tanaka. A stereo machine for video-rate dense depth mapping and its new applications. In *Proceedings of the 15th IEEE Computer Vision & Pattern Recognition Conference*, pages 196–202, San Francisco, June 1996.
- [18] Kurt Konolige. Small vision systems: Hardware and implementation. In *Proceedings of the Eighth International Symposium on Robotics Research (Robotics Research 8)*, pages 203–212, Hayama, Japan, October 1997.
- [19] Karsten Mühlmann, Dennis Maier, Jürgen Hesser, and Reinhard M. Anner. Calculating dense disparity maps from color stereo images, an efficient implementation. *International Journal of Computer Vision*, 47(1/2/3):79–88, 2002.
- [20] N. A. Thacker and P. Courtney. Statistical analysis of a stereo matching algorithm. In *Proceedings of the British Machine Vision Conference*, pages 316–326, 1992.
- [21] Arun P. Tirumalai, Brian Schunck, and Ramesh Jain. Dynamic stereo with self calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(12):1184–1189, 1992.
- [22] G. van der Wal and P. Burt. A VLSI pyramid chip for multiresolution image analysis. *Int. Journal of Computer Vision*, 8:177–190, 1992.
- [23] George Wolberg. *Digital Image Warping*. IEEE Computer Society Press, 1994.
- [24] J. Woodfill and B. Von Herzen. Real time stereo vision on the parts reconfigurable computer. In *5th Annual IEEE Symposium on Field-Programmable Custom Computing Machines*, pages 201–210, 1997.
- [25] R. Yang and M. Pollefeys. Multi-resolution real-time stereo on commodity graphics hardware. In *Proceedings of the 2003 IEEE Conference on Computer Vision and Pattern Recognition*, pages 211–218, Madison, Wisconsin, June 2003.
- [26] R. Zabih and J. Woodfill. Non-parametric local transforms for computing visual correspondence. In *Proceedings of the 3rd European Conference on Computer Vision*, pages 150–158, May 1994. <http://www.cs.cornell.edu/rdz/Papers/Archive/neccv.ps>, <http://www.cs.cornell.edu/rdz/Papers/Archive/nplt-journal.ps.gz>.