A Real-Time Large Disparity Range Stereo-System Using FPGAs

Divyang K. Masrani and W. James MacLean

Department of Electrical and Computer Engineering, University of Toronto,
Toronto, Ontario, Canada
{masrani, maclean}@eecg.toronto.edu

Abstract. In this paper, we discuss the design and implementation of a Field-Programmable Gate Array (FPGA) based stereo depth measurement system that is capable of handling a very large disparity range. The system performs rectification of the input video stream and a left-right consistency check to improve the accuracy of the results and generates subpixel disparities at 30 frames/second on 480×640 images. The system is based on the Local Weighted Phase-Correlation algorithm [9] which estimates disparity using a multi-scale and multi-orientation approach. Though FPGAs are ideal devices to exploit the inherent parallelism in many computer vision algorithms, they have a finite resource capacity which poses a challenge when adapting a system to deal with large image sizes or disparity ranges. In this work, we take advantage of the temporal information available in a video sequence to design a novel architecture for the correlation unit to achieve correlation over a large range while keeping the resource utilisation very low as compared to a naive approach of designing a correlation unit in hardware.

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 [2, 10, 14, 19], including reconfigurable hardware in the form of FPGAs [6, 11, 20, 12, 18, 5]. 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 problem 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, there is a finite amount of logic available, and when this is exhausted the only solution is to add another device

P.J. Narayanan et al. (Eds.): ACCV 2006, LNCS 3852, pp. 42–51, 2006.

or modify the algorithm. Not only is this costly from the design point of view, but may also involve the additional design issue of how to partition the logic across several devices.

In this paper we present the development of a versatile real-time stereo-vision platform. The system is an improvement of an earlier one [5] and addresses specific limitations of the previous system; capability to handle very large disparities, improving the accuracy of the system by pre-processing (input image rectification) and post-processing (consistency check), and finally the ability to handle larger images. The highlight of the work is the development of a novel architecture for the *Phase Correlation Unit* that can handle the correspondence task for scenes with very large disparities, but without increased resource usage on the FPGA, as compared to [5] which is capable of handling a disparity of only 20 pixels. The key to achieving large disparity correspondence matches is the use of a shiftable correlation window that tracks the disparity estimate for each pixel over time, 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. 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.

In Section 2, we briefly outline the technology used in this work and the platform used for the system development. In Section 3, we cover the theoretical basis of the phase-based stereo algorithm and then describe the architecture and implementation of the system. Section 4 discusses the results and the use of the *correlation unit* in alternate situations.

1.1 Previous Work

A variety of reconfigurable stereo machines have been reported [18, 12, 20, 6, 11]. The PARTS reconfigurable computer [18] consists of a 4×4 array of mesh-connected FPGAs with a maximum total number of about 35,000 4-input LUTs. A stereo system was developed on PARTS hardware using the census transform, which mainly consists of bit-wise comparisons and additions [20]. Kanade *et al.*[12] 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.*[6], a 4×4 matrix of small FPGAs is used to perform the cross-correlation of two 256×256 images in 140 ms. In Hou *et al.*[11], 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 [5] 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 30 frames/second (fps). It is worth noting that not all previous hardware approaches have been based on reconfigurable devices. In [13], a DSP-based stereo system performing rectification and area correlation, called the SRI Small Vision Module, is described. ASIC-based designs are reported in [16, 2] and in [19] commodity graphics hardware is used.

2 Reconfigurable Computing Platform

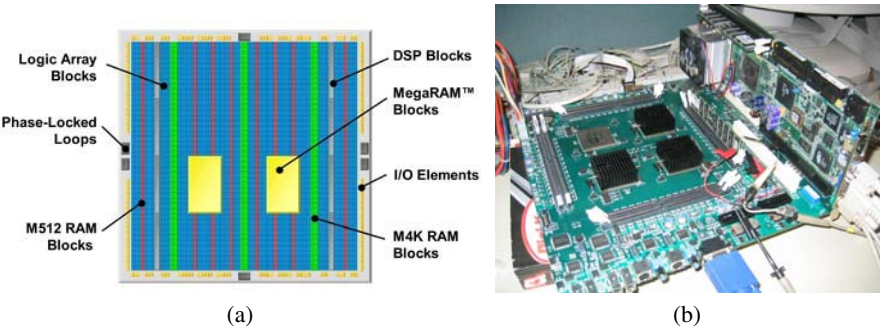
2.1 Field-Programmable Gate Arrays

An FPGA is an array of logic gates whose behaviour can be programmed by the end-user to perform a wide variety of logical functions, and which can be reconfigured as requirements change. FPGAs generally consist of four major components: 1) Logic blocks/elements (LB/LE); 2) I/O blocks; 3) Logic interconnect; and 4) dedicated hardware circuitry. The logic blocks of an FPGA can be configured to implement basic combinatorial logic (AND, OR, NOR, etc gates) or more complex sequential logic functions such as as microprocessor. The logic interconnect in an FPGA consists of wire segments of varying lengths which can be interconnected via electrically programmable switches. The density of logic blocks used in an FPGA depends on the length and number of wire segments used for routing.

Most modern FPGAs also have various dedicated circuitry in addition to the programmable logic. These come in the form of high-speed and high-bandwidth embedded memory, dedicated DSP blocks, Phase-Locked Loops (PLLs) for generating multiple clocks, and even general purpose processors. The FPGA we are using in our system, the Altera Stratix S80, comes with three different memory block sizes; 512 bits, 4 Kbits, and 512 Kbits for a maximum of 7 Mbits of embedded memory and 22 DSP blocks consisting of multipliers, adders, subtractors, accumulators, and pipeline registers. Figure 1 (a) shows an overview of the Altera Stratix S80 chip [3].

2.2 Transmogrifier-4Reconfigurable Platform

The TransmogrifierFour [7] (b) is a general purpose reconfigurable prototyping system containing four Altera Stratix S80 FPGAs. The board has specific features to support image processing and computational vision algorithms; these include dual-channel NTSC and FireWire camera interfaces, video encoder/decoder chip, and 2GB of DDR RAM connected to each FPGA. Each FPGA is also connected to the other three FPGAs and a PCI interface is provided to communicate with the board over a network. This can be used to send control signals or for debugging. The board with its major components is shown in Figure 1 (b).



(a) (b)

Fig. 1. (a)Typical features of a modern FPGA [3]. (b) Transmogrifier-4 reconfigurable computing board [7].

3 Large Disparity Stereo-System Development

The system implemented in this work is based on the "Local Weighted Phase Correlation" (LWPC) algorithm [9], which estimates disparity at a set of pre-shifts using a multi-scale, multi-orientation approach. A version of this algorithm was implemented in [5] but the system is limited to handling a maximum disparity of 20 pixels due to resource limitations on the FPGA. In the current implementation, we use two shiftable windows in the correlation unit to increase the disparity range of the system to 128 pixels (theoretically, the system can be implemented to handle a disparity range as large as the image width) without an increase in resource usage. There is a trade-off between the maximum disparity the system can handle and the time to initialise the system or recover from a mismatch, typically in the range of few tens of milliseconds.

3.1 Temporal Local-Weighted Phase Correlation

Based on the assumption that at video-rate (30 fps) the disparity of a given pixel will not change drastically from one frame the next, we use temporal information by performing localised correlation using a window centred on the disparity a pixel is expected to have at the current frame. This is discussed further below where we describe the architecture of the *Phase Correlation Unit*. Disparity calculations are performed at three scales$(1, 2, 4)$ and in three orientations (-45^o, 0^o, $+45^o$), the results of which are summed across scale and orientation. The expected interval between false peaks is approximately the wavelength of the filters applied at each scale. Thus the false peaks at different scales occur at different disparities and summation over the scales yields a prominent peak only at the true disparity [9]. The details of the LWPC algorithm can be found in [8]. Step 2 of the algorithm reflecting the incorporation of the temporal information is shown below:

2. For each scale and orientation, compute local voting functions $C_{j,s}(x, \tau)$ in a window centred at τ_c as

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

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

In addition, pre-processing (image rectification) and post-processing (left-right / right-left validation check) stages are also implemented to increase the accuracy of the system.

3.2 System Architecture

The high level architecture of the complete system is shown in Figure 2. It consists of six major units: Video Interface unit, Image Rectification unit, Scale-Orientation Decomposition unit, Phase-Correlation unit, Interpolation and Peak Detection unit, and Consistency Check unit.

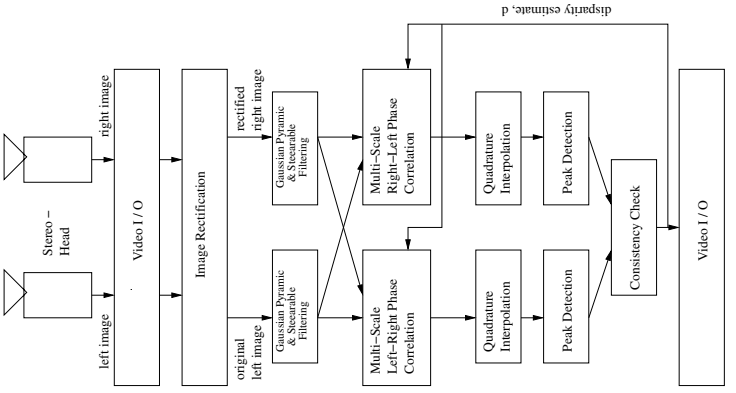


Fig. 2. High-level architecture of the stereo system

The **Video Interface Unit** is capable of receiving video signals from either NTSC or FireWire cameras at 30 fps and an image size of 480×640. In addition to the pixel values, the *Video Interface Unit* output "new line" and "new frame"signals. The data is sent to the *Image Rectification Unit* as it arrives without any buffering. This unit runs on the *camera clock*.

The **Image Rectification Unit** (Figure 3) treats the left input as the reference image and rectifies the right input using bilinear interpolation [17]. A stereo-setup with a worst-case vertical misalignment of 32 *scanlines* between the left and right image is assumed, which requires buffering of 64 *scanlines* of both the left and right image. This unit, as the rest of the system except the *Video I/O Unit*, run on the *system clock*. A *synchroniser circuit* is designed to handle glitch-free transfer of data between the two asynchronous clocks.

The warping operation for image rectification is approximated using the following bicubic polynomial:

$$\begin{aligned} x^{'} &= a_0 + a_1 x + a_2 y + a_3 x^2 + a_4 xy + a_5 y^2 \\ &\quad + a_6 x^3 + a_7 x^2 y + a_8 x y^2 + a_9 y^3 \\ y^{'} &= b_0 + b_1 x + b_2 y + b_3 x^2 + b_4 xy + b_5 y^2 \\ &\quad + b_6 x^3 + b_7 x^2 y + b_8 x y^2 + b_9 y^3 \ , \end{aligned} \tag{2}$$

where the a_i and b_i coefficients are computed by offline calibration.

The **Scale-Orientation Decomposition Unit** first builds a three-level Gaussian Pyramid by passing the the incoming right and left images through low-pass filters and sub-sampling. The pyramids are then decomposed into three orientations (-45^o, 0^o, +45^o) using G2/H2 steerable filters. G2/H2 filtering is implemented using a set of seven basis filters. By choosing a set of proper coefficients for the linear combination of the basis filters, filters of any arbitrary orientation can be synthesised. Since G2/H2 filters are X-Y separable, they require considerably less hardware resources than non-separable filters. The filter output is reduced to a 16-bit representation which is then sent to the Phase-Correlation unit.

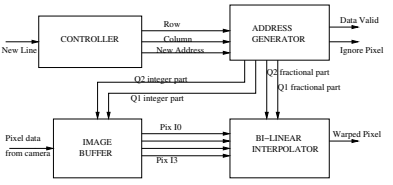


Fig. 3. Architecture of Image Rectification Unit

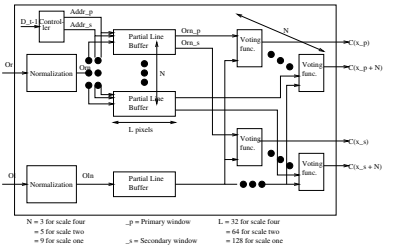


Fig. 4. Modified correlation unit with two shiftable windows

The **Phase-Correlation Unit** computes the real part of the voting function $C_{j,s}(x,\tau)$ as mentioned in Eq. 1 for all $1 \leq s \leq S, 1 \leq j \leq F, 0 \leq \tau \leq D$, where S is the total number of scales, F is the total number of orientations, and D is the maximum allowed disparity.

The *Phase Correlation Unit* is implemented using two shiftable correlation windows (see Figure 4) instead of a fixed window as is the traditional approach. One window, the *Primary Tracking Window* (*PTW*) uses temporal information to perform correlation in a localised region for each pixel. The tracking algorithm is currently a very simple one; the window is centred at the disparity estimate from the previous frame for a given pixel. More complex algorithms can be used as discussed in Section 4. 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) [4], especially during the initial frames. We have employed an initialisation stage to obtain an accurate disparity map. A second window, the *Secondary Roving Window* (*SRW*) (see Figure 5) does an incremental search up to a user-specifiable maximum disparity value. The increments are set equal to length of the correlation window, L, but these can be modified by a user at *run-time*. The *SRW* also aides in recovery from a mismatch after the initialisation stage. In situations where a new object enters the scene or a region becomes dis-occluded, the *SRW* will pick up this new information and provide a disparity estimate with a higher confidence value than the *PTW*, which can then latch on to this new estimate. There is a tradeoff between the time to recovery from a mismatch and the maximum disparity that the system can handle. For a maximum disparity of 128 pixels with increments of 10 pixels per frame for the *SRW*, the worst-case time to recovery is 233 milliseconds.

The **Interpolation/Peak-Detection Unit** interpolates the voting function results, $C_{j,2}(x,\tau)$ and $C_{j,4}(x,\tau)$, from the two coarser scales, in both x and τ domains such that they can be combined with the results from the finest scale, $C_{j,1}(x,\tau)$. Quadrature interpolation is performed in the τ domain and constant interpolation in the x domain. The interpolated voting functions are then combined across the scales and orientations to produce overall voting function $C(x,\tau)$. The peak in the voting function is then detected for each pixel as the maximum value of $C(x,\tau)$.

The **Consistency Check Unit** receives the estimated disparity results from both left-right and right-left correlations and performs a validity check on the results. The disparity value is accepted as valid if the results from the two correlation windows do not

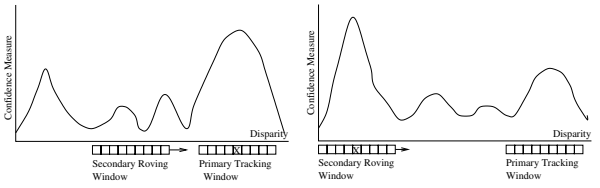


Fig. 5. PTW is correctly tracking the peak (denoted by an **X**) in the confidence measure in (a). In (b), PTW has lost track of the peak, but SRW has picked it up. PTW will latch on to this estimate at the next frame.

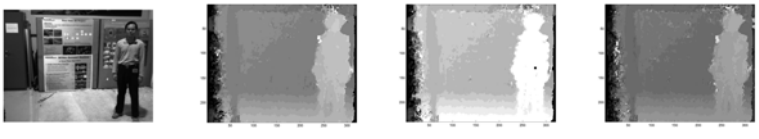
differ by more than one pixel. The checked disparity values are then sent back to the video interface unit to be displayed on a monitor. The invalid values are assigned special flag for display purposes.

4 Performance and Suggestions

The stereo system presented in this paper performs multi-scale, multi-orientation depth extraction for disparities up to 128 pixels using roughly the same amount of hardware resource as the previous system that is capable of handling disparities of only 20 pixels [5]. A dense disparity map is produced at the rate of 30 frames / second for an image size of 480 x 640 pixels. In terms of the Points x Disparity per second metric measure, the system is theoretically capable of achieving a performance of over 330 million PDS, which is considerably greater than the any of the others listed [18, 5].

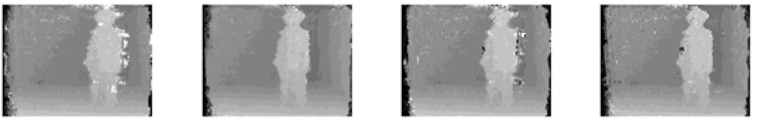
To better understand the workings of the modified correlation unit, we look at results from two real image sequences. The first, MDR-1, is a scene with a static camera and a moving person, and has a maximum disparity of around 16 pixels. The second, MDR-2, is a more complex scene with a moving person and a moving camera, and has a maximum disparity of approximately 30 pixels.

Frame 2 of the MDR-1 sequence is shown in Figure 6 (a). The disparity map during the initialisation stage is shown in (Figure 6 (c)) and the disparity map once the system has settled into the global minimum is shown in Figure 6 (d). For this particular sequence the algorithm settles into the global minimum by the second frame. The disparity map from the fixed correlation window of [5] is shown in Figure 6 (b) for comparison.



(a) (b) (c) (d)

Fig. 6. 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).

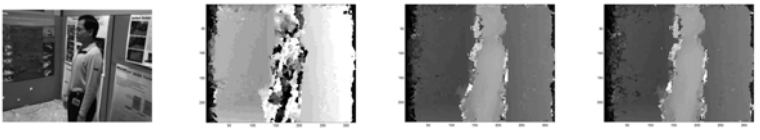


Frame 11 (a) Frame 15 (b) Frame 12 (c) Frame 13 (d)

Fig. 7. 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.

In Figure 7 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 7 (a) shows frame 11 for case (i); the results start to deteriorate but are completely recovered by frame 15, Figure 7 (b). For case (ii), the results deteriorate at frame 12, Figure 7 (c), and are already recovered by frame 13, Figure 7 (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 to select the maximum disparity parameter can yield better results. The disparity maps from the MDR-2 sequence for frame 4 (Figure 8 (a)) are shown in Figure 8 (b) for the implementation in [5] and Figure 8 (c) for our implementation. In [5], where the maximum disparity is limited to 20 pixels, the system cannot handle this sequence whereas our system shows good results.

A number of variations of the design can be implemented to achieve better results without having to make any changes to the correlation unit. Instead of the simple tracker that we are currently using for the *PTW*, a tracker based on a constant-velocity motion model can be used to achieve better tracking. The 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. Similarly, the location of the secondary window can be computed using a probabilistic likelihood estimate instead of the pre-determined roving locations. Other options include the possibility of concatenating the two correlation windows after the initialisation stage so as to support greater movement of objects from one frame to the next. The decision of when to concatenate the windows and when to use them individually in parallel can be made by



(a) (b) (c) (d)

Fig. 8. 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).

a simple count of the number of invalid disparity estimates after the validation check phase. This can be done for the whole image, region by region, or even for individual pixels. The issue of boundary overreach in correlation based algorithms [15] can also be solved by simply shifting the correlation windows by $\pm L/2$, where L is the length of the correlation window, so that the window does not cross over an object boundary. All of these modifications require the implementation of a post-processing stage that generates the appropriate input parameters for the correlation unit without having to make internal changes to the correlation unit itself.

The use of the correlation unit is not limited to a stereo-system. It can also be used in other systems such as object recognition using template matching, for *e.g.*, appearance models for object recognition. The two correlation windows can be used independently to search different regions of an image thereby speeding up the search process or they can be combined to support a larger template.

5 Summary

We have presented an FPGA-based real-time stereo system that is capable of handling very large disparities using limited hardware resources. We achieve this by designing a novel architecture for the correlation unit and also suggest possible uses of the correlation unit in variations of the stereo algorithm and even uses in different algorithms.

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. Peter J. Burt. A pyramid-based front-end processor for dynamic vision applications. *Proceedings of the IEEE*, 90(7):1188–1200, July 2002.
3. Altera Corporation. Stratix devices. http://www.altera.com/products/devices/stratix/stx-index.jsp, 2003.
4. 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.
5. 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.
6. 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.
7. Josh Fender. Transmogrifier 4 preliminary information. http://www.eecg.toronto.edu/~fender/tm4/sointroduction.shtml, August 2003.
8. David J. Fleet. *Measurement of Image Velocity*. Kluwer Academic Publishers, Boston, Massachusetts, 1992.
9. David J. Fleet. Disparity from local weighted phase correlation. In *International Conference on Systems, Man and Cybernetics*, volume 1, pages 48–54, 1994.
10. 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. stereo,intensity correlation,MMX, fast.

11. 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.
12. 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.
13. Kurt Konolige. Small vision systems: Hardware and implmentation. In *Proceedings of the Eighth International Symposium on Robotics Research (Robotics Research 8)*, pages 203–212, Hayama, Japan, October 1997.
14. 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. stereo,intensity correlation,MMX,fast.
15. M. Okutomi and Y. Katayama. A simple stereo algorithm to recover precise object boundaries and smooth surfaces. In *Proceedings of the IEEE Workshop on Stereo and Multi-Baseline Vision—SMBV'01*, 2001.
16. G. van der Wal and P. Burt. A VLSI pyramid chip for multiresolution image analysis. *Int. Journal of Computer Vision*, 8:177–190, 1992.
17. George Wolberg. *Digital Image Warping*. IEEE Computer Society Press, 1994.
18. 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.
19. 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.
20. 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.