A Low-Cost Embedded Real-Time 3D Stereo Matching System for Surveillance Applications

Georgia Rematska Kyprianos Papadimitriou Apostolos Dollas

Technical University of Crete
Chania, Greece

contact: dollas@ece.tuc.gr

Abstract—The implementation of low-cost 3D Stereo Vision systems requires stereo matching algorithms. Various efforts have been reported in literature, however, these systems require substantial hardware resources. The contribution of the present work is on a low-cost real-time system, which was fully implemented on a small Xilinx FPGA. The system presented in this work extends previous results of the authors through design space exploration, architecture improvements and careful problem sizing.

Keywords, 3D Stereo Matching, Real-Time, Stereo Vision, FPGA

I. Introduction and Related Work

Stereo vision includes different tasks, ranging from stereo image acquisition to image understanding. A key element to the achievement of stereo vision is stereo matching, so much so that in literature stereo vision and stereo matching are often used interchangeably. Two images are acquired simultaneously from cameras with the same orientation, yielding a reference image and a non-reference image. The basic problem of stereo matching lies in finding pairs of pixels, one in the reference image and the other in the non-reference image, that correspond to the same point in space. This is known as the correspondence problem and has been studied for many decades [1]. The difference in coordinates of the corresponding pixels (or similar features in the two stereo images) is the disparity. Based on the disparity between corresponding pixels and on stereo camera parameters such as the distance between the two cameras and their focal length, one can extract the depth of the related point in space by triangulation. This problem has been widely researched by the computer vision community and appears not only in stereo vision but in other image processing topics as well such as optical flow calculation [2]. The algorithm we study falls into the broad local category producing dense stereo maps. An extensive taxonomy of dense stereo vision algorithms is available in [3], and an online constantly renewed comparison can be found in [4], containing mainly software implementations. In general, the algorithm searches for pixel matches in an area around the reference pixel in the non-reference frame. This entails a heavy processing task as for each pixel the 2D search space should be exhaustively explored. To reduce the search space, the epipolar line constraint (i.e. the assumption that horizontal lines in the left and right images are horizontally aligned) is applied to reduce the 2D area search space to a 1D line. In the present work we assume that images are already rectified prior to processing.

The work in [5] was one of the earliest ones to combine the development of cost calculation with the Laplacian of Gaussian implemented in a DSP. More recently, several works developed different algorithms in fully functional FPGA-based systems ranging from relatively simple [6, 7] to more complex ones [9, 10, 11]. The authors of [10, 11] have designed full stereo vision systems incorporating the rectification preprocessing step. The authors in [8] provided designs of an algorithm based on census transform in three different technologies, i.e. CPU, GPU and DSP; the maximum performance was obtained with the GPU. Recent works are focusing on efficient hardware implementations targeting the embedded domain with real-time requirements. Hence, the authors in [15] provide adaptive support weight stereo correspondence algorithm, to increase the robustness of the matching process, while in [16] they suggest a configurable architecture for serving different application environments, along with a study of the impact of the various parameters on the performance and hardware/power overheads, whereas [17] is particularly tuned to stereo television applications.

In our previous work [14], we presented a new architecture suitable for real-time 3D stereo matching on a medium size FPGA (87fps at a resolution of full HD 1920X1200 images). That architecture was designed and implemented, however, testing was not performed in real-time due to the lack of suitable image acquisition subsystems. In [14] we justified the use of FPGA devices, which serve perfectly this application. The contributions of the present work vs. these in [14] are:

- An improved architecture with reduction trees rather than adder trees which were used in [14].
- A detailed analysis of hardware cost vs. the image disparity of the system.
- A full implementation on a Xilinx ATLYS FPGA system, complete with stereo cameras, real-time operation and presentation of the results on a HDMI monitor.

In the present work a minimal amount of information from [14] is repeated for readability purposes.

The paper is organized as follows: Section II discusses the census algorithm and our previous architecture. Section III describes the improvement of the reference architecture without loss of quality. Section IV has the design space exploration in order to fit the design to the small FPGA that we had as target technology and Section V has results from this work. Lastly, Section VI has conclusions from this work and future research plans.

II. The AD Census Algorithm and the Reference Architecture

The algorithm implemented in our previous work [14] consists of four parts: the cost computation step implemented by the Absolute Difference (AD) census combination matching cost, a simple fixed window aggregation scheme, a left/right consistency check and a scan-line belief propagation solution as a post processing step. The AD measure is defined as the absolute difference between two pixels, $C_{AD} = |p_1 - p_2|$, while census [12] is a window based cost that assigns a bit-string to a pixel and is defined as the sum of the Hamming distance between the bit-strings of two pixels. Let W_c be the size of the census window. A pixel's bit-string is of size ($W_c^2 - 1$ and is constructed by assigning 1 if $p_i > p_c$ or 0 otherwise, for $p_i \in Window$, and p_c the central pixel of the window. The two costs are fused by a truncated normalized sum:

$$C(pR; pT) = \max\{ [C_{AD}(pR; pT) / CMax_{AD}) + C_{Census}(pR; pT)/CMax_{Census})], \lambda_{trunc}\}$$

where λ_{trunc} is the truncation value given as parameter. This matching cost encompasses image local light structure (census) as well as information about the light itself (AD), and produces better results than its parts alone, as was shown in [13]. At object borders, the aggregation window necessarily includes costs belonging to two or more objects in the scene, whereas ideally we would like to aggregate only costs of one object. For this reason, truncating the costs to a maximum value helps to limit the effect of any outliers in each aggregation window [3]. We have settled on a $W_c = 9 \times 9$ sized census window, a $W_a = 5 \times 5$ sized aggregation window.

In the reference architecture of [14] a disparity search range of $D_{max} = 64$ was chosen; these values offer a good trade-off between overall quality and computational requirements but they severely affect the required hardware resources. We followed a similar procedure to determine all the other secondary parameters as well, such as the confident neighborhood queue size and the neighborhood queue size of the scanline belief propagation module, and the LR check threshold of the LR consistency check module [8]. The resulting architecture for the cost computation and the aggregation steps is shown in Fig. 1, whereas in Fig. 2, we show the architecture for the left/right consistency check, as well as the scan line belief propagation on the right side:

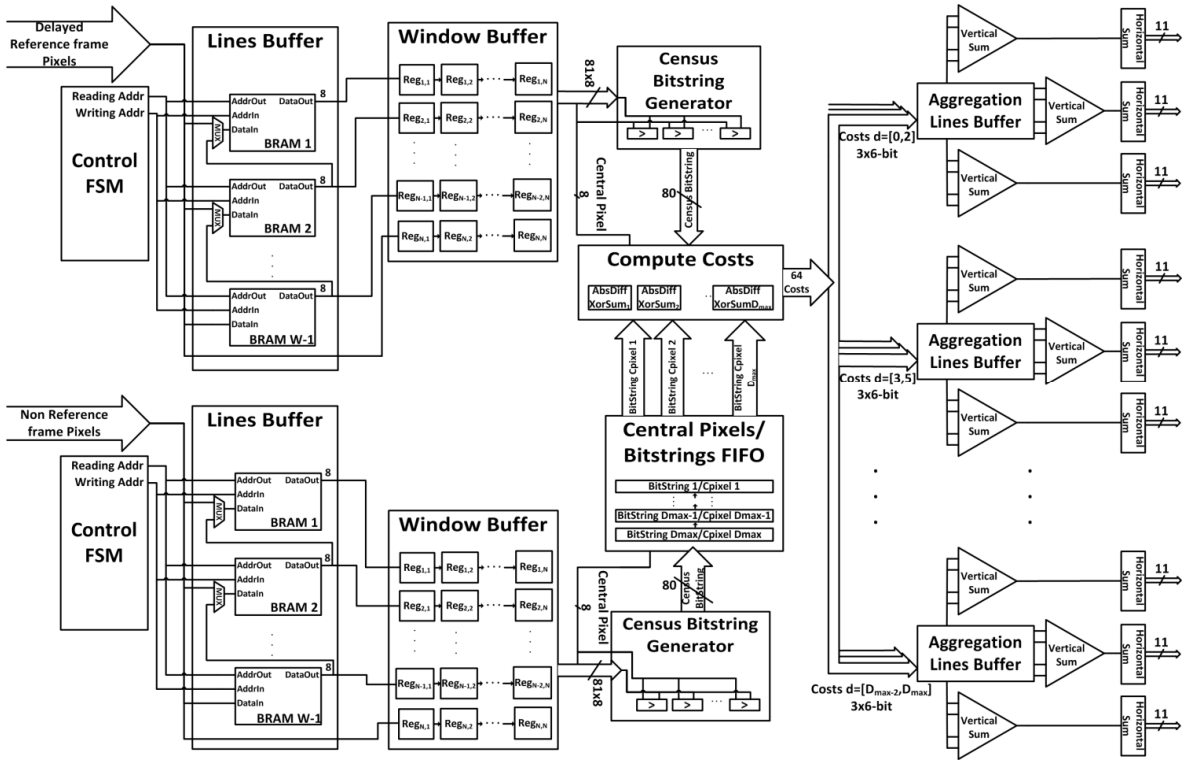

Fig 1. Datapath of the cost computation (left side) and aggregation (right side)

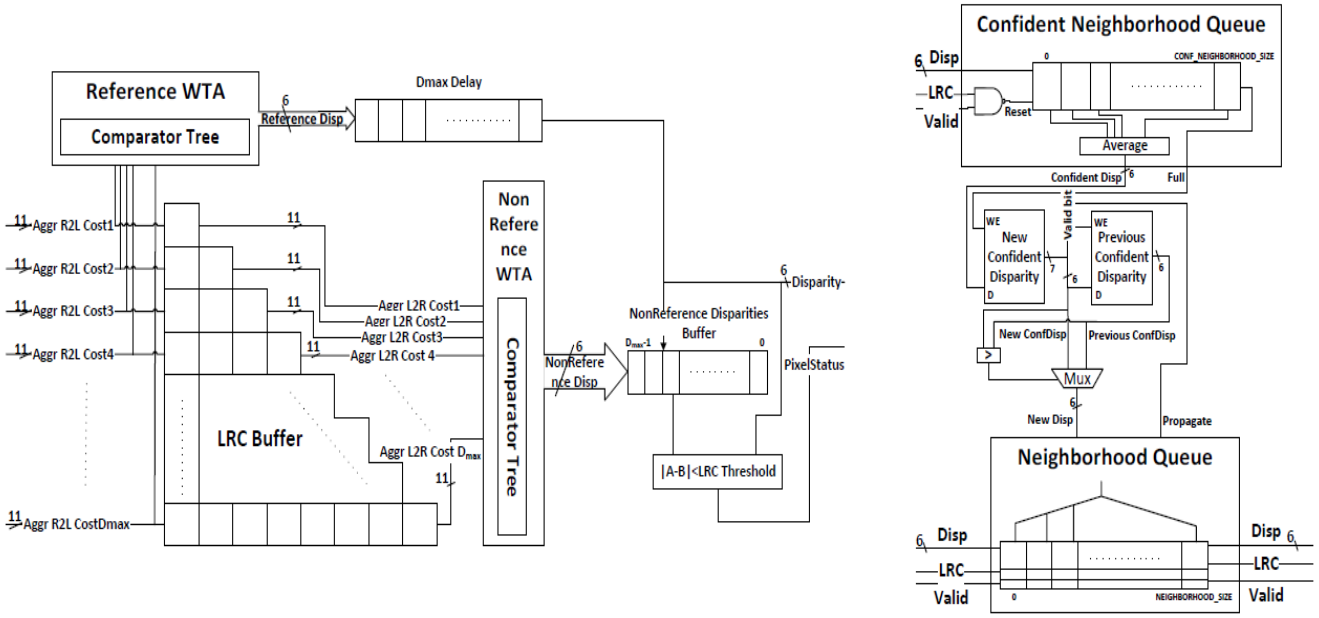

Fig 2. Datapaths fo the left/right consistency check (left side) and scan-line belief propagation (right side)

III. ARCHITECTURE DATAPATH IMPROVEMENT

In order to achieve our goal of real-time low-cost embedded operation we studied the hardware cost of several choices. The resource utilization in the reference architecture is shown in Table 1, below:

Table 1. Resource utilization in the Reference Architecture - Virtex XC5VLX110T FPGA for Dmax = 64, Wc = 9 Wa = 5.

	LUTs (%)	Flip-Flops (%)	BRAMs (%)
Available	69,120	69,120	148
Total consumed	37,986 out of 69,120 (55%)	41,792 out of 69,120 (60%)	59 out of 148 (40%)
AD Census	25,135 out of 37,986 (66%)	29,167 out of 41,792 (70%)	8 out of 59(14%)
Aggregation	6,547 out of 37,986 (17%)	7,312 out of 41,792 (17%)	51 out of 59(86%)
Left/Right Check	4,638 out of 37,986 (12%)	4,734 out of 41,792 (11%)	0 out of 59 (0%)
Scanline Belief Propagation	543 out of 37,986 (1.5%)	634 out of 41,792 (1.5%)	0 out of 59 (0%)

The purpose of the present work is to produce a low-cost, embedded version of the architecture. Thus, we studied the characteristics of the algorithm *vis a vis* a low cost, small Xilinx Spartan 6 FPGA, and we derived the results shown in Table 2. No percentages are shown in this Table, because it is very clear that the architecture does not fit in the Spartan 6 FPGA, even if we do not account for the resources needed for video capture and output display. The critical resource is the LUTs, of which 36,034 would be needed (plus those needed for video capture and output), with only 27,288 available.

Table 2. Resource utilization in Spartan 6 XC6SLX45 FPGA for Dmax = 64, Wc = 9, Wa= 5.

	LUTs	Flip-Flops	BRAMs
Available	27,288	54,576	116
Req'd AD Census	22,943	30,219	8
Req'd Aggregation	8,303	6,449	66
Req'd Left/Right Check	4,345	3,459	0
Req'd Scanline Belief Prop.	443	449	0
Required Total	36,034	40,576	74

As shown in Table 1 and Table 2, the AD Census part of the algorithm dominates the resource utilization, and hence this is the subsystem which ought to be optimized. Hence, we analyzed the resources of the AD Census, as shown in Table 3.

Table 3. Resource utilization in Spartan 6 XC6SLX45 FPGA of Compute Cost for Dmax = 64, Wc = 9, Wa = 5. (Unoptimized)

	LUTs (%)	Flip-Flops(%)	BRAMs (%)
Available	27,288	54,576	116
AD Census	22,943 (84 %)	30,219 (55%)	8 (6 %)
Compute Costs Total	21,481 out of 22,943 (94 %)	22,529 out of 30,219 (76 %)	0 out of 8 (0%)
Compute Cost simple Unit	336	353	0
XOR	80 out of 336 (23 %)	0 out of 353 (0%)	0 (0%)
Summing	228 out 336 (68 %)	328 out 353 (93 %)	0 (0%)
AD-normalized	36 out of 336 (10 %)	9 out of 353 (3%)	0 (0%)

We note that the Summing Unit dominates the cost, together with the XOR gates needed to compute the Hamming distance. The summing unit was implemented as an adder tree,

using the adder libraries of Xilinx which are very efficient. However, we have 80-bit strings of which we want to add the "1" values. By grouping them in bundles of 6 equal-weight inputs, we could implement the core of this component as a reduction tree rather than an adder tree. Fig.3 shows the original design of the reference architecture, whereas Fig. 4 shows the reduction tree.

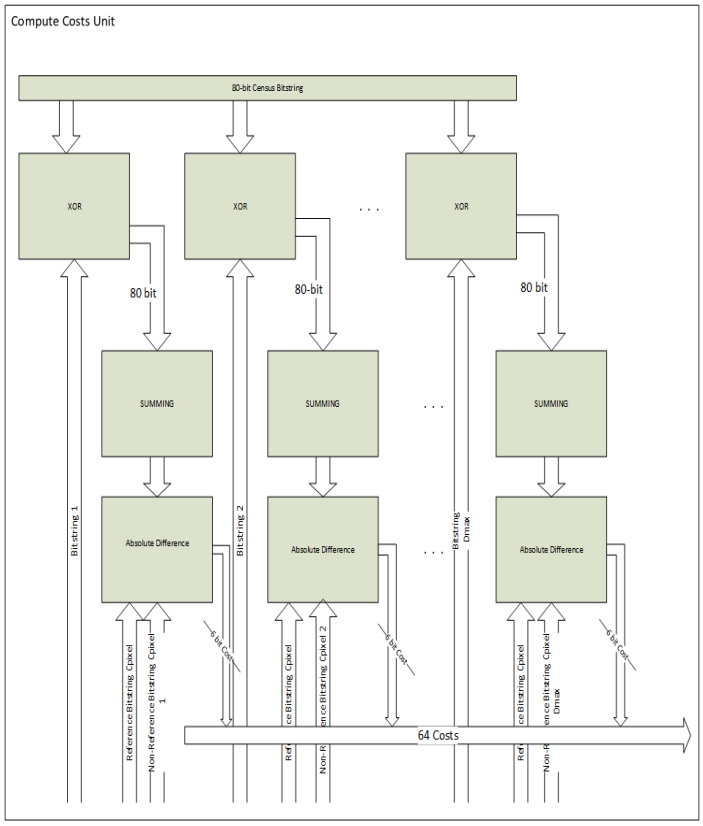

Fig. 3. Compute Costs Unit Datapath

As shown on the diagrams, in Fig. 3 we have usage of adders throughout the design, whereas in Fig. 4 we use adders only in the last stage. All remaining units are reduction trees, which can be thought of as adders of unary quantities, taking (up to) six inputs *of the same weight* and outputting the 3-bit binary number corresponding to the number of "1" inputs. The outputs of these units are then separated to signals of the same order of magnitude, which are put on similar reduction units. This improvement reduced the required LUTs (the critical resource) from 22,943 to 15,526, a 33% improvement. As shown in Table 4, however, even this improvement still would require 101% of the Spartan 6 LUT resources, without accounting for the video capture/output subsystems.

The improvement presented here results in a system with identical quality vs. the reference architecture, and the solution is sufficiently general to be used in any kind of implementation.

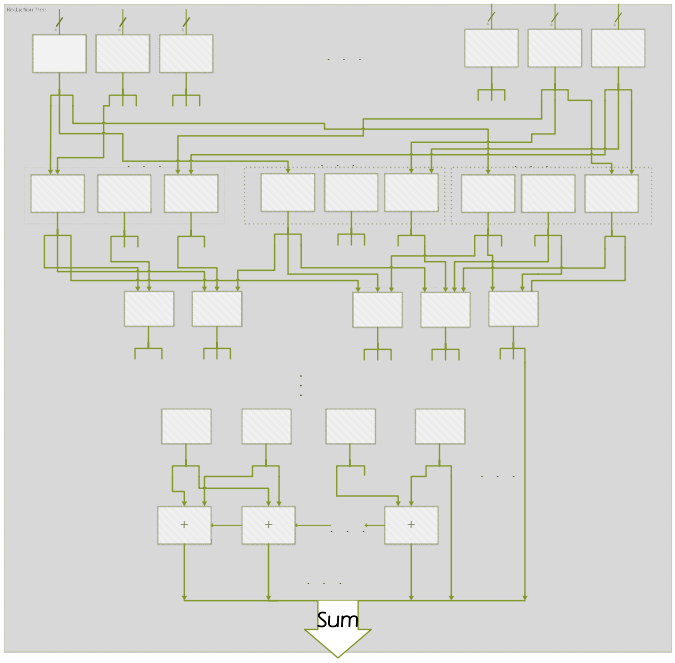

Fig. 4. Reduction Tree

Table 4. Resource utilization in Spartan 6 XC6SLX45 FPGA Dmax = 64, Wc = 9, Wa = 5 (Optimized)

	LUTs(%)	Flip-Flops(%)	BRAMs(%)
Available	27,288	54,576	116
Total Used	27,674**(101%)**	29,840 (54%)	74 (63 %)
AD Census	15,526	19,698	8
Aggregation	8,303	6,449	66
Left/Right Check	4,345	3,459	0
Scanline Belief Propagation	443	449	0

The next step in developing the new architecture was to evaluate how a reduction in the image resolution and/or maximum disparity would affect the quality of the system. Hence we performed design space exploration to evaluate various tradeoffs.

IV. Design Space Exploration

In order to develop a fully functional system we took into account the resources needed for image capture as well as for display. We also considered the tradeoff of lower resolution and maximum disparity, as for many surveillance applications no HD 1920 ×1080 resolution is needed. The target technology was a Xilinx ATLYS system with a small Spartan 6 FPGA, and a daughterboard with VmodCam board having two Aptina MT9D112 2-megapixel CMOS image sensors. The cameras provide 30fps and the data are stored in a DDR2 SDRAM memory. The output of our system is also stored in the DDR2 memory, and from there it is read and displayed in the HDMI monitor. In order to have a realistic system for surveillance applications, we chose a 640×480 resolution.

In Table 5 we have shown how the maximum disparity Dmax affects the resource utilization for the 640×480 images, always assuming a real-time 30 fps operation, and accounting for the resources needed for image I/O.

Table 5 Resource Utilization depending on Dmax for the whole system,when Wc=9 and Wa=5

	LUTs (%)	Flip-Flops (%)	BRAMs (%)
Available	27.288	54.576	116
Dmax value			
20	11,026 (40 %)	11,363 (20%)	58 (50 %)
30	15,241 (55 %)	15,777 (28 %)	59 (51 %)
35	17,339 (63 %)	18,212 (33 %)	68 (59 %)
40	19,421 (71 %)	20,574 (37 %)	76 (67 %)
45	21,525 (78 %)	22,684 (41 %)	84(73 %)
50	23,529 (86 %)	24,847 (45 %)	68 (59 %)
54	25,752 (94 %)	27,123 (49 %)	68 (59 %)

We notice that the maximum disparity for the system to fit in is Dmax=54 (the reference architecture had Dmax=64). This difference affects the depth of field of the system as follows: The camera baseline is b=63 mm and the focal length is f = 3.79 mm, hence we can calculate the minimum and maximum depth from the following equations:

$$Depth_{\min} = \frac{b * f}{D\max * DP_{hor}}$$

$$Depth_{\max} = \frac{b * f}{D\min * DP_{hor}}$$

where DP_{hor} is the pixel horizontal dot pitch, which depends on the image resolution. Thus, for a 640×480 resolution we have: DP_{hor} = 0,0055mm and thus $Depth_{min}$= 0.79m and $Depth_{max}$ = 42.64m. For the original D_{max}=64 we would have $Depth_{min}$= 0.68m and $Depth_{max}$ = 42.64m, i.e. we only lose roughly 11cm (roughly 14%) in the near field depth.

Thus, we proceeded with the design and implementation of the architecture with Dmax=54, Wc=9, Wa=5. The following Section has results from the system operation.

V. RESULTS

The system was fully implemented; the design is fully pipelined and operates in real time. Its clock frequency is 25MHz, which could be substantially increased but there was no reason to do so. The characteristics of the final design are shown in Table 6. In this Table the resources needed for the AD Census algorithm are shown, however, with the image acquisition and output the FPGA utilization approaches 100%.

The hardware of the system is shown in Fig. 5, it is an off-the-shelf Xilinx ATLYS system made by Digilent. The cameras can be seen at the lower portion, near the Digilent logo. Figure 6 has the left camera image and the right camera image. Figure 7 shows the disparity map as an actual photo-shoot of the monitor screen connected to the FPGA board, and for the images in Fig. 6. As expected, the flat one-color surfaces (such as the doors in the background are not identified well as many pixels have the same value and their disparity cannot be correctly identified, whereas the subject, box and chair are well identified.

Table 6 . Recourse Utilization for the StereoVision Module when Dmax =54 ,Wc =9 and Wa=5

	LUTs (%)	Flip-Flops (%)	BRAMs (%)
Available	27,288	54,576	116
Total Consumed	22,942 (84 %)	25,102 (45 %)	89
AD Census	13,041/22.942 (57 %)	16,704 /25,102 (67 %)	35/89 (39%)
Aggregation	6,255 /22,942 (27%)	4,845 /25,102 (19%)	54/89 (61%)
Left/Right Check	3,617/22,942 (16%)	3,039/25,102 (12%)	0/89 (0%)
Scanline Belief Propagation	344/22,942 (2 %)	351 /25,102 (1 %)	0/89 (0%)

Fig. 5. System Hardware

VI. CONCLUSIONS AND FUTURE WORK

In this paper we have presented how a previous architecture for 3D stereo matching was improved to take 33% fewer resources without loss in the quality of results, and then with thorough design space exploration it was reduced both in terms of image resolution and maximum disparity in order to fit in a small FPGA. The resulting image resolution of 640×480 pixels is quite acceptable for many surveillance applications, whereas the final disparity of 54 resulted in a system with 14% lower depth of field in the near field.

In the future we will investigate whether the compute power of the FPGA which allows for much higher clock frequencies, in conjunction with the available DDR2 memory can allow for higher disparities by offloading portions of the computation to external memory. As such operations need to be interleaved with image acquisition and output the timing will be quite sensitive, and thus require separate analysis.

Fig. 6. Left and Right Image

Fig.7 Output of the Stereo Matching algorithm implemented in the FPGA

ACKNOWLEDGMENT

This work was co-funded by the European Regional Development Fund (ERDF) and national resources of Greece and Cyprus through the project "EVAGORAS" of the INTERREG funding program, and the Cyprus Research Promotion Foundation program [ΠΕΝΕΚ/0311/32].

REFERENCES

[1] D. H. Ballard and C. M. Brown, *Computer Vision*, Englewood Cliffs, NJ, US: Prentice-Hall, 1982.

[2] D. Marr, *Vision*, San Francisco, CA, US: Freeman, 1982

[3] D. Scharstein and R. Szeliski, "A Taxonomy and Evaluationof Dense Two-Frame Stereo Correspondence Algorithms*," International Journal of Computer Vision,* vol. 47, no. 1-3, pp. 7–42, April-June 2002.

[4] http://vision.middlebury.edu/stereo/eval/.

[5] K. Konolige, "Small Vision Systems: Hardware and Implementation," in *Proceedings of the International Symposium on Robotics Research*, pp. 111–116, 1997.

[6] C. Murphy, D. Lindquist, A. M. Rynning, T. Cecil, S. Leavitt, and M. L. Chang, "Low-Cost Stereo Vision on an FPGA," in *Proceedings of the IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM)*, pp. 333– 334, April 2007.

[7] S. Hadjitheophanous, C. Ttofis, A. S. Georghiades, and T. Theocharides, "Towards Hardware Stereoscopic 3D Reconstruction, A Real-Time FPGA Computation of the Disparity Map," in *Proceedings of the Design, Automation and Test in Europe Conference and Exhibition (DATE),* pp. 1743–1748, March 2010.

[8] M. Humenberger, C. Zinner, M. Weber, W. Kubinger, and M. Vincze, "A Fast Stereo Matching Algorithm Suitable for Embedded Real-Time Systems," *Computer Vision and Image Understanding*, vol. 114, no. 11, pp. 1180–1202, November 2010.

[9] D. K. Masrani and W. J. MacLean, "A Real-Time Large Disparity Range Stereo-System using FPGAs," in *Proceedings of the IEEE International Conference on Computer Vision Systems*, pp. 42–51, 2006

[10] S. Jin, J. U. Cho, X. D. Pham, K. M. Lee, S.-K. Park, and J. W. J. Munsang Kim, "FPGA Design and Implementation of a Real-Time Stereo Vision System," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 20, no. 1, pp. 15–26, January 2010.

[11] http://danstrother.com/2011/01/24/fpga-stereo-visionproject/.

[12] R. Zabih and J.Woodfill, "Non-parametric Local Transforms for Computing Visual Correspondence," in *Proceedings of the European Conference on Computer Vision (ECCV),* pp. 151–158, 1994.

[13] X. Mei, X. Sun, M. Zhou, S. Jiao, H. Wang, and X. Zhang, "On Building an Accurate Stereo Matching System on Graphics Hardware," in *Proceedings of the IEEE International Conference on Computer Vision Workshops (ICCV),* pp. 467–474, November 2011.

[14] S. Thomas, K. Papadimitriou, A. Dollas, "Architecture and Impleementation of 3D Stereo Vision on a Xilinx FPGA", in Proceedings, *21st IFIP/IEEE International Conference on Very Large Scale Integration (VLSI-SoC)*, pp. 190-195, Istanbul, October 2013.

[15] C. Ttofis, T. Theocharides, "Towards accurate hardware stereo correspondence: A real-time FPGA implementation of a segmentation-based adaptive support weight algorithm", *in Proc. Design, Automation & Test in Europe (DATE) Conference and Exhibition,* pp. 703-708, March 2012.

[16] C. Ttofis, S. Hadjitheophanous, A. S. Georghiades, T. Theocharides, "Edge-Directed Hardware Architecture for Real-Time Disparity Map Computation", *IEEE Transactions on Computers*, vol. 62, no. 4, pp. 690-704, April 2013.

[17] A. Aysu, M. Sayinta, C.Cigla, "Low Cost FPGA Design and Implementation of a Stereo Matching System for 3D-TV Applications", in Proceedings*, 21st IFIP/IEEE International Conference on Very Large Scale Integration (VLSI-SoC),* pp. 208-213, Istanbul, October 2013.