2014 IEEE Intelligent Vehicles Symposium (IV)
June 8-11, 2014. Dearborn, Michigan, USA

3DV – An Embedded, Dense Stereovision-based Depth Mapping System*

Gabriele Camellini[1], Mirko Felisa[1], Paolo Medici[1], Paolo Zani[1], Francesco Gregoretti[2], Claudio Passerone[2] and Roberto Passerone[3]

Abstract— **This paper describes the architecture and hardware implementation of an embedded, low-cost and low-power dense stereo reconstruction system, running at 30 fps at VGA resolution. The processing pipeline includes an initial image rectification stage, a cost generation unit based on the non-parametric census transform, a state-of-the-art Semi-Global cost optimization stage, and a final minimization and noise suppression step. The hardware implementation is based on a Xilinx Zynq™ System-on-Chip, which besides the FPGA provides a physical dual-core ARM CPU, which is exploited for control and to deliver output over the integrated Gigabit Ethernet connection.**

I. INTRODUCTION

A basic need in the robotics, automotive and industrial fields is real-time 3D environment reconstruction, since it allows safe navigation across multiple terrains, object detection and classification and advanced visualization. Dense stereovision is a popular solution to this problem, providing as a result a dense point cloud containing hundreds of thousands elements, each directly associated to exactly one pair of pixels of the input imagery.

Stereovision-based depth mapping is a widely studied subject, which has seen substantial research [1], [2], [3], [4], [5] and benchmarking [6], [7], [8] efforts. The various algorithms offer different trade-offs in terms of computational complexity, reconstruction quality and robustness against noise in the input images. In particular, the so-called Semi-Global Matching minimization strategy first proposed in [9], coupled with a census cost metric [10] has proven to be the solution of choice in many demanding real-world applications [11], [12], [13], since it is insensitive to illumination variations and can deal with significant untextured areas while still being able to correctly handle challenging depth discontinuities, such as those produced by small objects and poles. These capabilities stem from the fact that the SGM approach optimizes the matching costs across the whole image by adding a smoothness term to the computed values prior to minimization. The optimization step is computationally expensive and requires a lot of memory bandwidth to access the costs, thus making real-time implementations challenging; however, its massively parallel nature well adapts to a wide range of modern hardware devices. In particular, PC-based solutions can reach 20 fps at VGA resolution on a standard desktop CPU [14] by exploiting both multi-core and SIMD [1] processing capabilities. GPU implementations exist as well, reaching more than 60 fps at VGA resolution [15]; however, they require high end, expensive devices, with more than doubled power consumption.

All these solutions suffer from the fundamental flaw of not being suitable for use in embedded hardware designs; a viable alternative is the use of FPGA units, which well adapt to the workloads typical of the SGM algorithm [16], [17]. FPGAs offer low cost and power consumption, high reliability and availability, even for demanding environments (e.g. automotive) but require significantly longer development times.

This paper proposes a design based on the Xilinx Zynq™ System-on-Chip, which substantially reduces the development effort by integrating a capable FPGA, a dual-core ARM CPU and multiple I/Os in a single physical package.

Sec. II covers the basics of dense 3D reconstruction, while Sec. III provides some details about the chosen hardware platform and a description of the processing stage architecture. Finally, Sec. IV and V provide some performance figures and the future developments roadmap.

II. DENSE STEREO MAPPING BACKGROUND

The processing steps involved in dense stereo reconstruction are presented in Fig. 1. First the combined effect of lens distortion and cameras misalignment is removed from the input images I_L and I_R using a look-up table (LUT): this allows to operate on a pair of rectified images R_L and R_R, which reduces the matching phase to a 1-D search along the epipolar lines.

Each pair of integer coordinates $\mathbf{p} = (x, y)$ on the rectified image is associated to the fractional location $\mathbf{i} = (x_i + \alpha, y_i + \beta) = LUT(x, y)$ on the input image.

Bilinear interpolation can be used to determine the rectified pixel value:

$$R(\mathbf{p}) = (1-\beta)((1-\alpha)I(x_i, y_i) + \alpha I(x_i+1, y_i)) + \beta((1-\alpha)I(x_i, y_i+1) + \alpha I(x_i+1, y_i+1)) \quad (1)$$

[1]G. Camellini, M. Felisa, P. Medici and P. Zani are with VisLab - Dipartimento di Ingegneria dell'Informazione, Università di Parma, Italy {cgabri, felisa, medici, zani}@vislab.it

[2]F. Gregoretti and C. Passerone are with Dipartimento di Elettronica e Telecomunicazioni, Politecnico di Torino, Italy {francesco.gregoretti, claudio.passerone}@polito.it

[3]Roberto Passerone is with Dipartimento di Ingegneria e Scienza dell'Informazione, Università degli Studi di Trento, Italy roberto.passerone@unitn.it

*This work has been supported by the European Research Council (ERC) within the *Sensor for 3D Vision* (3DV #297463) Proof of Concept grant.

[1]Single Instruction Multiple Data

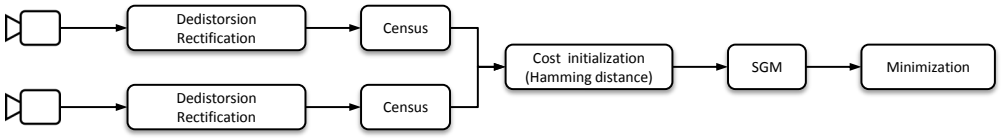

Fig. 1. Stereo reconstruction pipeline.

The $n \times n$ census transforms C_L and C_R [18] of the rectified images are then computed, and used to initialize the cost cube $C(\mathbf{p}, d)$ for each pixel $\mathbf{p}$ and disparity level d by taking the Hamming distance H of the resulting pixel pair:

$$C(\mathbf{p}, d) = H(C_L(x + d, y), C_R(x, y)) \tag{2}$$

The subsequent SGM step allows to regularize the cost cube by providing a tractable approximation of the optimal depth map that minimizes the global energy function

$$E(D) = \sum_{\mathbf{p}} C(\mathbf{p}, D_{\mathbf{p}}) + \sum_{\mathbf{q} \in N\mathbf{p}} P_1 \mathbf{T}[|D_{\mathbf{p}} - D_{\mathbf{q}}| = 1] + \sum_{\mathbf{q} \in N\mathbf{p}} P_2 \mathbf{T}[|D_{\mathbf{p}} - D_{\mathbf{q}}| > 1] \tag{3}$$

with $T[x] = 1$ if x is true, 0 otherwise.

The Semi-Global matching approach approximates the optimal solution by computing the costs arising along 1-D paths from 8 directions towards each pixel. The costs $L_{\mathbf{r}}$ along a given path $\mathbf{r}$ for each pixel $\mathbf{p}$ and disparity d are computed as

$$\begin{aligned} L_{\mathbf{r}}(\mathbf{p}, d) = C(\mathbf{p}, d) + \min(&L_{\mathbf{r}}(\mathbf{p} - \mathbf{r}, d), \\ &L_{\mathbf{r}}(\mathbf{p} - \mathbf{r}, d-1) + P_1, L_{\mathbf{r}}(\mathbf{p} - \mathbf{r}, d+1) + P_1, \\ &\min_i L_{\mathbf{r}}(\mathbf{p} - \mathbf{r}, i) + P_2) - \min_k L_{\mathbf{r}}(\mathbf{p} - \mathbf{r}, k) \end{aligned} \tag{4}$$

The final aggregated cost cube $S(\mathbf{p}, d)$ is then computed as

$$S(\mathbf{p}, d) = \sum_{\mathbf{r}} L_{\mathbf{r}}(\mathbf{p}, d) \tag{5}$$

and the winning depth map $\bar{D}$ becomes

$$\bar{D}(\mathbf{p}) = \arg\min_d (S(\mathbf{p}, d)) \tag{6}$$

In order to reduce the number of spurious reconstructions, a pixel $\mathbf{p}$ is considered valid only if the ratio between the minimum and second-minimum costs is below a predefined threshold (i.e. the minimum is strong enough).

A final equiangular interpolation step [19] is used to estimate the fractional part of the winning disparity value $\bar{d}$ for each pixel $\mathbf{p}$:

$$\bar{d}_{frac} = \frac{S(\mathbf{p}, \bar{d} - 1) - S(\mathbf{p}, \bar{d} + 1)}{\max(S(\mathbf{p}, \bar{d} - 1), S(\mathbf{p}, \bar{d} + 1)) - S(\mathbf{p}, \bar{d})} \tag{7}$$

III. SYSTEM ARCHITECTURE AND IMPLEMENTATION

A. Hardware platform

The Xilinx Zynq™ System-on-Chip family provides both an FPGA Programmable Logic (PL) and a Programmable System (PS) based on dual-core ARM® Cortex™-A9 processor in the same physical package. The PL and PS are connected by a set of AXI interfaces; in particular, four High Performance channels (HP0, HP1, HP2 and HP3) have been used to handle intermediate data transfers to and from main DDR memory. Each channel is full duplex and has a maximum theoretical bandwidth of 1.2 GB/s in each direction; burst transfers are supported in multiples of 4 or 8 bytes words. In order to store the final results (depth map and rectified images) for use by the PS processor while guaranteeing its cache coherency the available Accelerated Coherence Port (ACP) has been used; the same channel also handles the transfer of the census images C_L and C_R.

The Zynq™ SoC integrates a number of I/O peripherals, and in particular:

- GigE – used to stream the results;
- I^2C – used to control the sensors and to interface with the installed IMU unit;
- QSPI Flash – used to store the Linux OS image, rectification LUT files and the PL bitstream;
- CAN – used for configuration and high-level output;
- UART – used for debug.

The depth mapping system presented in this paper is based on the automotive-grade Z-7020 model, since it offers the best compromise between available FPGA resources and cost. Two 752×480 Aptina MT9V034 CMOS monochrome sensors have been selected for image acquisition, since they represent a cheap, easy-to-integrate and automotive grade solution.

B. System architecture

Fig. 2 illustrates how the stereo reconstruction pipeline presented in Sec. II has been mapped to the target hardware platform; the inner workings of each module are described in grater detail in Sec. III-C and Sec. III-D. It must be noted that the path aggregation stage has been split in two distinct passes, referred to as forward and backward in the following; the forward pass corresponds to the $0°$, $45°$, $90°$ and $135°$ paths, while the backward pass covers the $180°$, $225°$, $270°$ and $315°$ paths, as shown in Fig. 3. Moreover, the paths of the forward pass are summed together before being transferred to external DDR memory to reduce the bandwidth requirements. All the four HP AXI channels are used to write

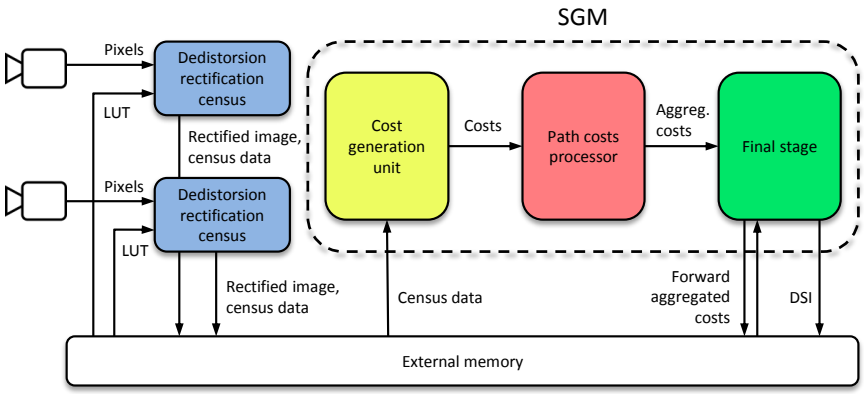

Fig. 2. Hardware architecture. For the sake of simplicity, intermediate FIFOs and BRAM buffers have been omitted.

and read back the costs generated by the forward aggregation pass, since the data produced at each clock cycle by the path aggregation stage (32 bytes) is exactly 4 times the channels word width (64 bits). The ACP channel, instead, handles the transfer of LUT data, census and rectified images and the final depth map.

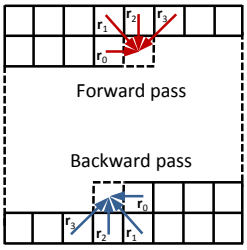

Fig. 3. Forward and backward passes to perform 8-way SGM path aggregation.

Finally, Fig. 4 shows which units are active at any given time during the processing of one frame. Readout and rectification/costs generation are synchronous, and overlapped with the forward SGM pass and streaming of the left and right images. The return pass and depth map generation begin after the forward pass is complete, and are overlapped with DSI streaming and the exposure time of the subsequent frame.

C. Rectification and cost generation

Rectification and cost generation operate independently on the images captured by each camera.

Rectification works by inverting the deformation introduced by the lens distortion and the sensors misalignment, which results in a misplacement of the pixels in the raw cameras images. To correct this effect, a look-up table (LUT) provides for each pixel of the rectified image the coordinates of the corresponding pixel of the raw image from the camera. This information is fractional, and requires the application of a bilinear interpolation according to Eq. (1). To perform this task, the system loads the pixels from the camera and stores them in a circular buffer. To limit the internal memory usage, a rectified pixel is assumed to be located no more than 32 lines above or below the corresponding raw pixel, an amount which is sufficiently large to handle optics with a focal length greater than 3.8 mm. Therefore, the storage requirements for the raw image are limited to only 64 lines centered around the current position; a set of indices is maintained to keep track of the current memory location.

The LUT data is stored in the external DDR memory, using a compressed, incremental format. After loading each pixel, the system computes the address of the rectified pixel by using the LUT data, and fetches the corresponding four pixels from raw memory. Fixed point arithmetic with two fractional digits is being used; for this reason, the interpolation block has been custom designed to compute the result without the use of expensive multipliers.

The rectified pixels are stored in an internal memory buffer for census computation. Because the census is limited to a neighborhood of only 2 lines above and below the current line, only five lines of the image have to be stored at any time, using a sliding window approach similar to the one employed for the raw image. New pixels are loaded directly into the census computation block, while the previously loaded pixels slide through a 5×5 shift register representing the neighborhood. A simple combinational bit counting circuit is used to compute the transform, which is then stored in the external DDR memory and made available for the subsequent Semi-Global Matching step.

D. Semi-Global Matching

The Semi-Global Matching (SGM) stage starts from the census costs stored in the external DDR memory, and generates the disparity image, using a forward pass followed by a backward pass. The partial cost cube computed during the initial forward pass is progressively saved in the external DDR memory, while initial costs are re-computed (identical) for both passes, as it is less expensive than storing them. The implemented algorithm scans the images row by row, and from the leftmost pixel to the rightmost one within a row.

The SGM implementation is functionally decomposed into three separate modules, internally connected through FIFOs and communicating with the other blocks using the external

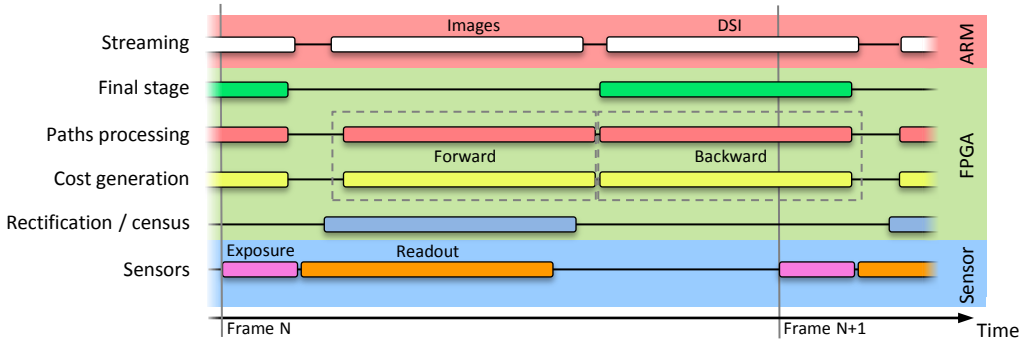

Fig. 4. System timing diagram. The colors of the various stages match those of the corresponding units in Fig. 2.

DDR memory. The modules, which are described in more detail in the following, respectively compute:

- The initial costs, i.e. the pixelwise Hamming distances $C(\mathbf{p}, d)$;
- The aggregated path costs $L_{\mathbf{r}}(\mathbf{p}, d)$ of 4 paths for forward or backward processing;
- The aggregated cost cube for all paths and the minimum disparity image.

1) Cost generation unit: scans the left and right census images, and computes the Hamming distance for up to 128 disparities. Pixel data for the left image is stored in a 128 elements shift register, and the cost is computed by counting the number of bits set out of the 24 bits obtained by XORing together each pair of census pixels. Counting is implemented by accumulating the results of 4 6-input 1-output LUTs, each operating on a 6-bit long segment of the value.

At each clock cycle, costs for 32 disparity levels are generated and stored into the output FIFO. After 4 clock cycles, all the 128 disparity levels for a pixel are completed, and the processor moves to the next one. Forward and backward algorithms are very similar, and share most of the hardware implementation except for the initialization of the shift register.

2) Path costs processor: implements the SGM core computation of $L_r(\mathbf{p}, d)$. It concurrently computes the path costs for 32 disparity levels and 4 paths, using a 7-stage pipeline. The four paths are then aggregated and the sum is stored in the output FIFO. Similarly to the pixelwise Hamming distance processor, the throughput is one pixel every 4 clock cycles.

The implementation of the elementary block of the path cost computation is shown in Fig. 5, which follows the formulation already presented in Eq. 4. The block is replicated 32 times for each of the 4 paths, for a total of 128 blocks, to concurrently process 32 disparity levels. While most inputs depend on the disparity d, the minimum over all disparities for the previous pixel in the path does not, so it doesn't need registers in the pipeline because it is a constant, and it is updated only every 4 clock cycles. The bottom multiplexer selects the output between the initial cost for the beginning

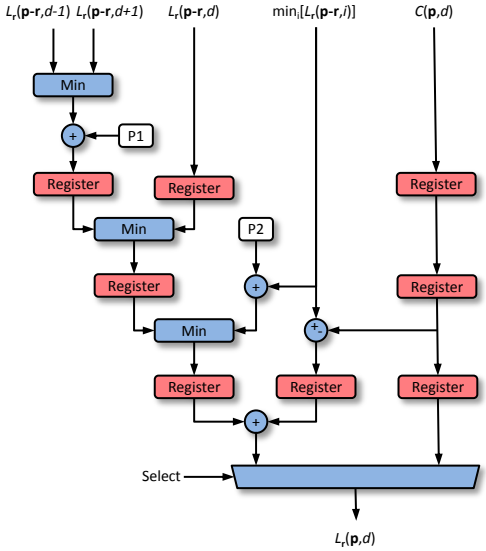

Fig. 5. Elementary block for path costs. In blue, combinational logic blocks, in red, delay blocks.

of a path, and the currently computed value along a path.

Costs computed for the $\mathbf{r}_1$, $\mathbf{r}_2$ and $\mathbf{r}_3$ paths in Fig. 3 are temporarily stored in on-chip BRAMs to be used in cost computation of pixels of the successive row. On the other hand, costs computed for the $\mathbf{r}_0$ path are directly forwarded to the proper pipeline stage for the adjacent pixel in the horizontal direction.

Besides being summed to compute the partial cost cube to be stored in the output FIFO, all costs are also fed to a tree of minimum functions, to extract the minimum over all the disparity levels (used for the subsequent pixels of each path). The tree is pipelined to increase its performance.

Forward and backward computation steps are identical, the only difference being the order of pixels. Whereas the forward pass starts at the upper-left corner of the image, the backward assumes that the lower-right pixel is processed first. However, it is not necessary to know which pass is currently executing, as pixel order is handled by the DDR interface blocks directly.

TABLE I

STEREO RECONSTRUCTION IMPLEMENTATIONS COMPARISON

Implementation	Hardware platform	Algorithm	Image size $[px]$	Time $[ms]$	Disparity rate $[10^6/s]$
Gehrig ECVW10 [20]	Intel® Core™ i7 975 EX @ 3.3 GHz	CT + SGM (8) + MF + L/R	640×320 @ 128	224	117
Broggi IROS11 [14]	Intel® Core™ i7 920 @ 3.20 GHz	CT + SGM (8) + L/R	640×320 @ 128	27	970
Hirschmüller ISVC10 [21]	NVIDIA® GeForce™ 8800 Ultra	HMI +SGM (8) +MF +L/R	640×480 @ 128	238	165
Nedevschi IV10 [19]	NVIDIA® GeForce™ GTX 280	CT + SGM (8) + L/R + MF	512×383 @ 56	19	578
Banz ICCV11 [15]	NVIDIA® Tesla C2050	RT + SGM (8) +MF	640×480 @ 128	16	2457
Gehrig ICVS09 [16]	Xilinx® Virtex-4 FX140	ZSAD + SGM (8) + L/R	$2 \times 340 \times 200$ @ 64	40	218
Banz SAMOS10 [17]	Xilinx® Virtex-5 LX 220T-1	RT + SGM (4) + L/R + MF	640×480 @ 128	9.7	4053
this paper	Xilinx® Zynq™ 7020	CT + SGM (8) + 2ndmin	640×480 @ 128	33	1192

Overview of current SGM implementations; in parentheses, the number of aggregation paths. Different cost functions are used, namely the census transform (CT), rank transform (RT), hierarchical mutual information (HMI), and zero-mean sum of absolute differences (ZSAD). L/R denotes the left-right consistency check, MF median filtering and 2ndmin the minimum vs 2nd minimum ratio check.

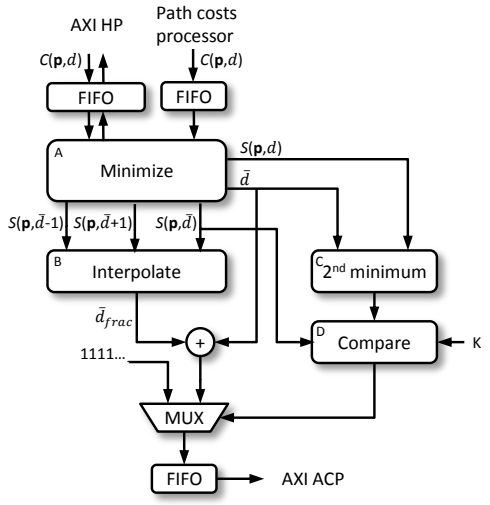

Fig. 6. Final stage block diagram

3) Final stage: computes the optimal depth map with subpixel accuracy, and invalidates pixels with weak minima, as illustrated in Fig. 6.

During the forward pass it receives data from the path processor and forwards them to the DDR memory for storage. During the backward pass it adds data from the path processor (32 costs per clock cycle) to those fetched from the memory into a unique aggregate cost vector $S(\mathbf{p}, d)$ for each pixel $\mathbf{p}$, and computes the disparity $\bar{d}$ as the index of the value of the vector with the minimum value (see Eq. 6). These two tasks are performed by block A, which also provides at its output the value of the minimum cost and the value of the costs for disparities $\bar{d} - 1$ and $\bar{d} + 1$. This data is then processed by unit B, which increases the resolution of the pixel disparity by performing an equiangular interpolation between the values of the supplied costs, as described in Eq. 7.

In parallel, unit C removes from $S(\mathbf{p}, d)$ the minimum cost and the costs for disparities $\bar{d} - 1$ and $\bar{d} + 1$, then computes a new minimum value among the remaining costs. A comparison is then performed between the first minimum multiplied by 128 and the second minimum multiplied by a constant value K and upon this comparison a multiplexer outputs as a final value either the disparity in fixed point format or an invalid disparity code corresponding to all "1"s.

The final disparity value is stored into an output FIFO, which manages the synchronization between the final stage and the external DDR memory into which the final disparity image is stored.

IV. RESULTS

A performance comparison between the proposed system and other publicly available SGM implementations is presented in Tab. I. While two solutions exhibit a higher disparity rate [15], [16], and one has an essentially identical performance [14], all of them run on hardware devices of considerably higher cost and power consumption.

Memory traffic is detailed in Tab. II. Most of the bandwidth is consumed to store and read back the partial summed costs produced during the forward aggregation pass: while the system can be configured to run without performing the backward pass – thus eliminating the need for intermediate data storage, much like it is done in [17] – having a two-pass approach is essential to produce consistent results.

The FPGA resources usage of the system is reported in Tab. III; it is worth to note that 43 % of the available BRAMs is occupied by the SGM path costs processor buffers, 14 % are used by the rectification stage, 9 % for memory communication and just 1 % to compute the census transform. SGM BRAMs are mostly employed to store the previous cost values along each path, and their number depends both on the image width and the supported disparity levels. Also, each additional 18 kbit BRAM devoted to rectification allows to increase the corresponding buffer size by about 6 image rows.

The implementation phase produced the following frequencies for the three main clock sources in use:

- 26.7 MHz – dedistorsion, rectification and census stage, synchronous with the input camera pixel clock;
- 130 MHz – SGM processing pipeline;
- 150 MHz – memory communication logic.

these values are compatible with the underlying hardware, and allow the system to smoothly run at 30 fps.

TABLE II

TRAFFIC TO AND FROM EXTERNAL DDR MEMORY.

	Forward pass			Backward pass		
	Data	Read	Write	Data	Read	Write
HP0	$S_{fwd}(\mathbf{p}, [0-31])$	-	0.58 GB/s (41%)	$S_{fwd}(\mathbf{p}, [0-31])$	0.58 GB/s (47%)	-
HP1	$S_{fwd}(\mathbf{p}, [32-63])$	-	0.58 GB/s (41%)	$S_{fwd}(\mathbf{p}, [32-63])$	0.58 GB/s (47%)	-
HP2	$S_{fwd}(\mathbf{p}, [64-95])$	-	0.58 GB/s (41%)	$S_{fwd}(\mathbf{p}, [64-95])$	0.58 GB/s (47%)	-
HP3	$S_{fwd}(\mathbf{p}, [96-127])$	-	0.58 GB/s (41%)	$S_{fwd}(\mathbf{p}, [96-127])$	0.58 GB/s (47%)	-
ACP	LUTs, R_L, R_R, C_L, C_R	0.27 GB/s (22%)	0.27 GB/s (22%)	C_L, C_R, D	0.21 GB/s (18%)	0.053 GB/s (4%)
DDR	All	2.6 GB/s (60%)	2.6 GB/s (60%)	All	2.5 GB/s (59%)	0.053 GB/s (1%)

Maximum bandwidths for each AXI channel and the DDR controller are respectively 1.2 GB/s and 4,3 GB/s in both directions.

TABLE III

FPGA RESOURCES USAGE.

Resource	Total	Percentage
LUT	23600	44 %
BRAM (18 kbit)	189	68 %
DSP	48	21 %

V. CONCLUSIONS AND FUTURE WORK

The proposed system (depicted in Fig. 7) can produce dense depth maps with a resolution of 640×480 pixels at 30 frames per second on a low-power, low-cost and automotive-grade SoC. Eight aggregation paths have been

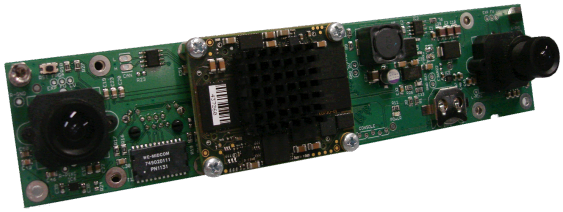

Fig. 7. Prototype board for the proposed system, with a baseline of 15 cm.

used to improve the reconstruction quality in challenging situations (e.g. poor illumination conditions, lack of texture), at the cost of a significant traffic to and from the external DDR memory.

In order to increase the system robustness against occlusions and mismatches a left-right consistency check will be introduced in future system revisions, together with adaptive mean and a gap-filling filters [4], which have proven suitable to further enhance the depth map quality.

REFERENCES

[1] O. Faugeras, B. Hotz, H. Mathieu, T. Viville, Z. Zhang, P. Fua, E. Thron, and P. Robotvis, "Real time correlation-based stereo: Algorithm, implementations and applications," 1996.

[2] H. Hirschmuller, "Stereo processing by semiglobal matching and mutual information," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 2, pp. 328–341, Feb. 2008. [Online]. Available: http://dx.doi.org/10.1109/TPAMI.2007.1166

[3] M. Felisa and P. Zani, "Incremental Disparity Space Image computation for automotive applications," in *Procs. IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, St.Louis, Missouri, USA, Oct. 2009.

[4] A. Geiger, M. Roser, and R. Urtasun, "Efficient large-scale stereo matching," in *Proceedings of the 10th Asian conference on Computer vision - Volume Part I*, ser. ACCV'10. Berlin, Heidelberg: Springer-Verlag, 2011, pp. 25–38. [Online]. Available: http://dl.acm.org/citation.cfm?id=1964320.1964325

[5] R. Ranftl, S. Gehrig, T. Pock, and H. Bischof, "Pushing the Limits of Stereo Using Variational Stereo Estimation," in *IV*, 2012, to appear.

[6] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *International Journal of Computer Vision*, vol. 47, pp. 7–42, 2001.

[7] P. Steingrube, S. K. Gehrig, and U. Franke, "Performance evaluation of stereo algorithms for automotive applications," in *Proceedings of the 7th International Conference on Computer Vision Systems: Computer Vision Systems*, ser. ICVS '09. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 285–294.

[8] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Computer Vision and Pattern Recognition (CVPR)*, Providence, USA, June 2012.

[9] H. Hirschmuller, "Accurate and efficient stereo processing by semi-global matching and mutual information," in *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 2 - Volume 02*, ser. CVPR '05. Washington, DC, USA: IEEE Computer Society, 2005, pp. 807–814.

[10] H. Hirschmuller and D. Scharstein, "Evaluation of stereo matching costs on images with radiometric differences," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 9, pp. 1582–1599, Sept. 2009.

[11] http://www.ensenso.de.

[12] M. Bertozzi, L. Bombini, A. Broggi, M. Buzzoni, E. Cardarelli, S. Cattani, P. Cerri, A. Coati, S. Debattisti, A. Falzoni, R. I. Fedriga, M. Felisa, L. Gatti, A. Giacomazzo, P. Grisleri, M. C. Laghi, L. Mazzei, P. Medici, M. Panciroli, P. P. Porta, P. Zani, and P. Versari, "VIAC: an Out of Ordinary Experiment," in *Procs. IEEE Intelligent Vehicles Symposium 2011*, Baden Baden, Germany, June 2011, pp. 175–180, iSSN: 1931-0587.

[13] U. Franke, D. Pfeiffer, C. Rabe, C. Knoeppel, M. Enzweiler, F. Stein, and R. G. Herrtwich, "Making bertha see."

[14] A. Broggi, M. Buzzoni, M. Felisa, and P. Zani, "Stereo obstacle detection in challenging environments: the VIAC experience," in *Procs. IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, San Francisco, California, USA, Sept. 2011, pp. 1599–1604.

[15] C. Banz, H. Blume, and P. Pirsch, "Real-time semi-global matching disparity estimation on the gpu," in *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, 2011, pp. 514–521.

[16] S. K. Gehrig, F. Eberli, and T. Meyer, "A real-time low-power stereo vision engine using semi-global matching," in *Proceedings of the 7th International Conference on Computer Vision Systems: Computer Vision Systems*, ser. ICVimages/Capitolo4S '09. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 134–143.

[17] C. Banz, S. Hesselbarth, H. Flatt, H. Blume, and P. Pirsch, "Real-time stereo vision system using semi-global matching disparity estimation: Architecture and fpga-implementation," in *Embedded Computer Systems (SAMOS), 2010 International Conference on*, 2010, pp. 93–101.

[18] R. Zabih and J. Woodfill, "Non-parametric local transforms for computing visual correspondence," in *Computer Vision ECCV '94*, ser. Lecture Notes in Computer Science, J.-O. Eklundh, Ed. Springer Berlin Heidelberg, 1994, vol. 801, pp. 151–158. [Online]. Available: http://dx.doi.org/10.1007/BFb0028345

[19] I. Haller, C. Pantilie, F. Oniga, and S. Nedevschi, "Real-time semi-global dense stereo solution with improved sub-pixel accuracy," in *Intelligent Vehicles Symposium (IV), 2010 IEEE*, 2010, pp. 369–376.

[20] S. Gehrig and C. Rabe, "Real-time semi-global matching on the cpu," in *ECVW10*, 2010, pp. 85–92.

[21] H. Hirschmüller and I. Ernst, "Mutual information based semi-global stereo matching on the gpu." in *ISVC (1) 08*, 2008, pp. 228–239.