Real-Time Stereo Vision System using Semi-Global Matching Disparity Estimation: Architecture and FPGA-Implementation

Christian Banz, Sebastian Hesselbarth, Holger Flatt, Holger Blume, and Peter Pirsch
Institute of Microelectronic Systems, Leibniz Universität Hannover, Hannover (Germany)
e-mail: {banz,hesselbarth,flatt,blume,pirsch}@ims.uni-hannover.de

***Abstract*—This paper describes a new architecture and the corresponding implementation of a stereo vision system that covers the entire stereo vision process including noise reduction, rectification, disparity estimation, and visualization. Dense disparity estimation is performed using the non-parametric rank transform and semi-global matching (*SGM*), which is among the top performing stereo matching methods and outperforms locally-based methods in terms of quality of disparity maps and robustness under difficult imaging conditions. Stream-based processing of the SGM despite its non-scan-aligned, complex data dependencies is achieved by a scalable, systolic-array-based architecture. This architecture fulfills the demands of real-world applications regarding frame rate, depth resolution and low resource usage. The architecture is based on a novel two-dimensional parallelization concept for the SGM. An FPGA implementation on a Xilinx Virtex-5 generates disparity maps of VGA images (640×480 pixel) with a 128 pixel disparity range under real-time conditions (30 fps) at a clock frequency as low as 39 MHz.**

I. Introduction

Computing depth information from stereo-camera-systems (disparity information) has many advantages over other 3D sensing methods. Especially in robotics and automotive applications, passive stereo vision systems offer significant advantages compared to active systems, e.g. radar, when considering interference in environments with high sensor density. Further, depth information and original image are exactly matched, which greatly simplifies many image interpretation tasks [1].

Active research in this field has resulted in a wide range of disparity estimation algorithms. These have been extensively studied and benchmarked [2] [3]. Among the top performing algorithms is the semi-global matching (*SGM*), which is for example of special interest for the automotive sector [4]. The combination of rank transform and semi-global matching has been shown to be insensitive to noise and a number of other interferences such as lighting, exposure, and gamma differences. All of these effects occur in real-world scenarios. Further, this combination is able to deal with large untextured areas, retains edges, and does not suffer from streaking [3].

The advantageous performance of SGM is due to the fact, that SGM performs an optimization across the entire image, and not only a locally finite neighborhood. On the other hand, this results in high computational loads, extremely high memory bandwidth to store intermediate results, and irregular data access patterns. All of these are challenges for hardware implementations. Implementations of the SGM on SIMD-CPUs and GPUs achieve frame rates of 1.4 fps and 13 fps for QVGA images, respectively [5] [6]. A highly specialized VLIW-ASIP with comprehensive architectural adaptations has been shown to achieve 30 fps for VGA images [7]. Nevertheless, real-time ability of the entire image analysis task is essential when targeting automotive applications. Stereo image processing includes pre-processing (e.g. noise reduction), rectification, disparity estimation, post-processing (e.g. interpolation) and decision making. Besides real-time capability and form-factor, power consumption is an issue, which rules out GPUs. Hence, dedicated hardware solutions, based on FPGAs or ASICS, are required. With reduced innovation cycles, FPGAs offer advantages in terms of time-to-market, long-term flexibility and costs. Already specialized FPGAs for the automotive market are available.

This paper proposes a systolic-array-based, and therefore, scalable and highly parallel hardware architecture for semi-global matching disparity estimation. For this purpose a new two-dimensional parallelization concept for the SGM is introduced. Based on this, a flexible, high frame rate stereo vision system is designed which includes image capturing, noise reduction, rectification, disparity estimation and rendering for visual inspection. The complete system is implemented and evaluated on a custom-build FPGA-based platform with a stereo-camera setup.

Section II briefly reviews algorithmic background on stereo vision, rectification and semi-global matching. With this background related work is presented in Section III. The parallelization of the SGM for VLSI-implementation is presented in section IV. A detailed description of the architecture and implementation of the stereo vision system is given in section V. Experimental results are discussed and evaluated in section VI. Finally, conclusions are drawn in section VII.

II. Algorithmic Background on Stereo Vision

Disparity estimation is the task of identifying the projection point of the same 3-D real-world point in two or more images taken from distinct viewpoints. Due to the underlying epipolar geometry [2] of a stereo-camera setup, the search space for corresponding pixels is oriented along the epipolar lines. A

978-1-4244-7938-2/10/$26.00 ©2010 IEEE

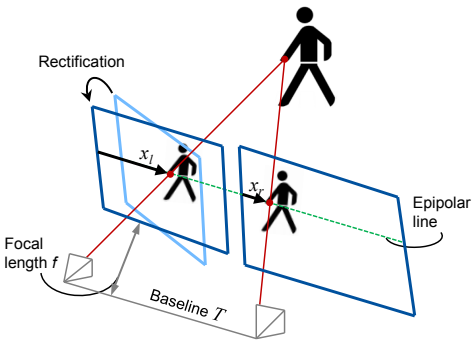


Fig. 1. Epipolar geometry of a stereo camera setup with the one-dimensional search space for corresponding pixels. The rectification to achieve a row-aligned search-space is only shown for the left projection plane.

separate image transform, the rectification, aligns the epipolar lines in such a way that they are parallel to the scanlines of the image [8]. The disparity is the location difference of the projection points in both images (see Fig. 1).

With the disparity $d = (x_l - x_r)$, the rectified focal length f, and the baseline T of the camera pair, the distance between the baseline and the 3-D point can be calculated as

$$z = fT/(x_l - x_r) = fT/d. \tag{1}$$

The image processing steps of a stereo vision system are illustrated in Fig. 2. First the stereo images are pre-processed (e.g. noise reduction, contrast enhancement, etc.) and rectified with camera individual parameters. These are the input images for the disparity estimation. A post-processing step (e.g. outlier suppression, peak removal and interpolation of small holes) can optionally be introduced. For optical presentation on a display, a rendering step creates a false color representation, in which colors code the calculated distances and the luminance values correspond to those of the rectified image. The algorithmic background on rectification and disparity estimation is discussed in detail below.

A. Rectification

During the rectification step, camera distortions (lens distortion, sensor tilting and offset from focal axis) and a non-ideal camera-pair setup (non-co-planar, non-row-aligned image planes) are compensated. Reverse mapping assigns every pixel in the rectified image a sub-pixel accurate origin in the input image. These displacement vectors are calculated using the intrinsic, extrinsic, tangential and radial distortion parameters, which are obtained by a separate calibration step. The rectified pixels are obtained using the bilinear interpolation, which exhibits a reasonable trade-off between image quality and hardware implementation costs.

For the one-dimensional case with an origin of $x_0 + \alpha$, α being the fractional component between the integer-accurate positions x_0 and $x_0 + 1$, the new intensity value of a pixel $I_{rec}(x_r)$ is

$$I_{rec}(x_r) = (1 - \alpha) I_{in}(x_0) + \alpha I_{in}(x_0 + 1). \tag{2}$$

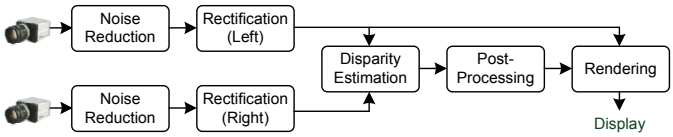


Fig. 2. Image processing tasks and data flow within the stereo vision system.

Eq. (2) is a one-dimensional linear interpolation. As bilinear interpolations are separable they can be realized using linear interpolations. For a detailed discussion of (stereo-)camera models and the rectification process see [8].

B. Disparity Estimation

In most stereo matching methods, a similarity measure between two pixels in the base and match image (or left and right image, respectively) is calculated and those with highest accordance or correlation are assigned as corresponding. In this work, the matching costs $C(\mathbf{p}, d)$ are calculated as

$$C(\mathbf{p}, d) = |R_b(p_x, y) - R_m(p_x - d, y)| \tag{3}$$

where $\mathbf{p} = [p_x, p_y]^T$ is the pixel location in the left image and R is the area-based non-parametric rank-transform [9]. It is defined as the number of pixels $\mathbf{p}'$ in a square $M \times M$ neighborhood $A(\mathbf{p})$ of the center pixel $\mathbf{p}$ with an intensity I less than $I(\mathbf{p})$.

$$R(\mathbf{p}) = \|\{\mathbf{p}' \in A(\mathbf{p}) \mid I(\mathbf{p}') < I(\mathbf{p})\}\|. \tag{4}$$

It is noteworthy that C exhibits a word width of $\lceil log_2(M^2) \rceil$ which is independent from the word width of the input data.

In many cases, pixel-wise calculated matching costs (i.e. locally calculated) yield non-unique or wrong correspondences due to low texture and ambiguity. Therefore, semi-global matching [10] introduces global consistency constraints by aggregating matching costs along several independent, one-dimensional paths across the image. These are formulated recursively by the definition of the path costs $L_r(\mathbf{p}, d)$ along a path $\mathbf{r}$.

$$\begin{aligned} L_r(\mathbf{p}, d) = C(\mathbf{p}, d) + \min[&L_r(\mathbf{p} - \mathbf{r}, d), \\ &L_r(\mathbf{p} - \mathbf{r}, d - 1) + P_1, \\ &L_r(\mathbf{p} - \mathbf{r}, d + 1) + P_1, \\ &\min_i L_r(\mathbf{p} - \mathbf{r}, i) + P_2] - \\ &\min_l L_r(\mathbf{p} - \mathbf{r}, l) \end{aligned} \tag{5}$$

The first term describes the primary matching costs. The second term adds the minimal path costs of the previous pixel $\mathbf{p} - \mathbf{r}$ including a penalty P_1 for disparity changes and P_2 for disparity discontinuities, respectively. Discrimination of small changes $|\Delta d| = 1$ pixel *(px)* and discontinuities $|\Delta d| > 1$ px allows for slanted and curved surfaces on the one hand and preserves disparity discontinuities on the other hand. P_1 is an empirically determined constant. P_2 is adapted to the image content with $P_2 \propto |I(\mathbf{p}) - I(\mathbf{p} - \mathbf{r})|^{-1}$. The last term prevents constantly increasing path costs. For a detailed discussion refer to [10].

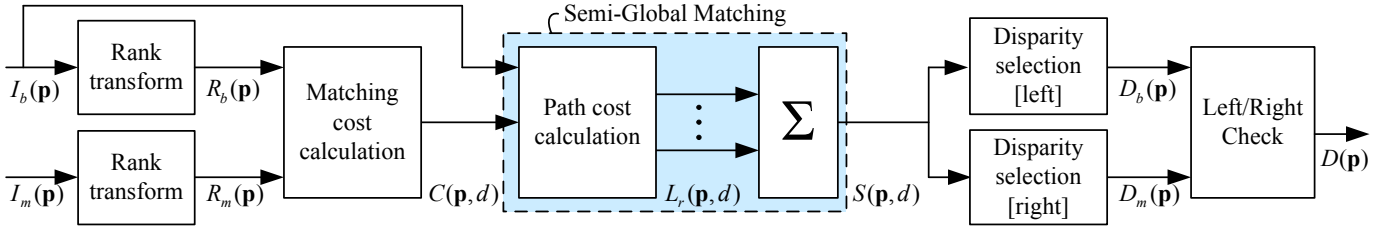


Fig. 3. System architecture for disparity estimation using rank-transform and semi-global matching

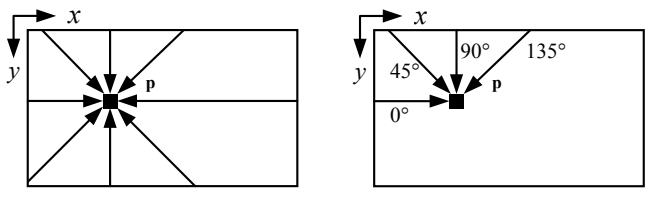
Quasi-global optimization across the entire image is achieved by calculating path costs from multiple directions to a pixel, as shown in Fig. 4. The aggregated costs S are the sum of the path costs

$$S(\mathbf{p},d) = \sum_{\mathbf{r}} L_r(\mathbf{p},d). \tag{6}$$

The disparity map $D_b(p_x,p_y)$ from the perspective of the base camera is calculated by selecting the disparity with the minimal aggregated costs $\min_d S(\mathbf{p},d)$ for each pixel. For calculating $D_m(q_x,q_y)$, the minimal aggregated costs along the corresponding epipolar lines, i.e. $\min_d S(q_x+d,q_y,d)$, are selected. Both, uniqueness-check and left/right check are performed to ensure that only valid disparities with high confidence level are used for further processing steps. The uniqueness check sets disparities invalid if the minimum $\min_d S(\mathbf{p},d)$ is not unique. The left/right-check sets disparities invalid if the disparity $D_b(\mathbf{p})$ and its corresponding disparity of D_m differ by more than 1 px. As a basic post-processing step, a 3×3 median filter is applied to suppress outliers. An overview of the processing steps of the disparity estimation algorithm used in this work is given in Fig. 3.

III. Related Work

Stereo matching methods can be classified into local and global approaches [2] with direct implications for hardware design complexity and robustness. Local (or area-based) stereo matching methods are well suited for hardware implementation due to the implicit parallelism and little data dependencies. These methods perform matching solely on the information based in a local search window and are challenged by areas with low or repetitive textures due to a high level of ambiguity [3]. Global methods, to which the SGM belongs, perform an optimization across parts of the entire image and can cope with these problems, but have higher computational loads and often irregular data access patterns, which is a challenge for hardware implementations.



(a) 8 paths (b) 4 paths

Fig. 4. Possible path orientations for eight and four paths

A large percentage of architectures for stereo vision are based on local stereo matching methods. FPGA-implementations of local SAD-based methods (sum of absolute differences) with a number of additional improvements have been implemented by various researchers [11][12][13][14]. Depending on the emphasis of the referenced work, the results vary in throughput and resolution up to 640×480 pixel (*px)* for 64 px disparity range at 31 frames per second (*fps*). In [15] the local census transform is used at a resolution of 320×240 px and 20 px disparity range at 150 fps. SAD, rank transform, and census transform are extended to adaptive block sizes in [16] in order to deal with changing scenarios (e.g. texture-less or cluttered environments). In [1] the Tyzx ASIC is presented for color-image census-based stereo-matching achieving 200 fps at 512×480 px and 52 px disparity range. Another ASIC (EyeQ2) which targets the automotive sector is offered by the company MobilEye utilizing a local correlation variant [17]. Recently, Jin *et al.* [18] presented a stereo vision system which is entirely driven by the pixel input clock frequency and achieves up to 230 fps at 64 px disparity range for VGA images. It also utilizes a local census-based stereo matching method.

In [19] a local, phase-based method is extended to large disparity ranges without significant additional hardware cost by adapting an offset of the smaller disparity search window across multiple frames. After large disparity changes, a latency of several frames occurs before correct disparity information can be regained. A bio-inspired method based on gabor filters is introduced in [20]. Due to the low-pass filtering nature of the gabor filters, high disparity changes are lost when searching through large disparity ranges.

Among the global methods are dynamic programming (*DP*) approaches, which typically suffer from streaking (inconsistency between scanlines). In [21] a trellis-based DP implementation, using a single interline consistency constraint, is presented. A DP approach based on a maximum-likelihood method is implemented in [22] achieving 64 fps at 640×480 px with 128 px disparity range. A SGM implementation has been introduced by Gehrig *et al.* [4]. It achieves 27 fps at 320×200 px and 64 px disparity range, but has limited scalability to achieve higher data throughput.

However, many of todays real-world applications not only require a frame rate of 30 fps at a minimum resolution of

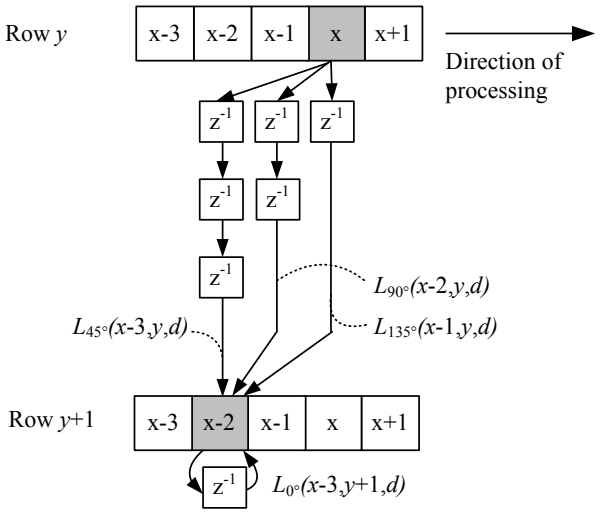


Fig. 5. Synchronized and parallel calculation of four paths for pixels in multiple image rows using delay elements

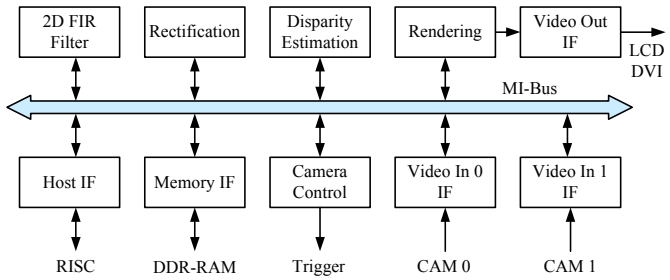


Fig. 6. Top level hardware architecture of the stereo vision system. The dedicated processing elements are connected via the MI-Bus.

640×480 with 128 px disparity range but also the extremely high algorithmic performance offered by the SGM. Despite the considerable progress on real-time stereo vision, fulfilling both demands is still an open issue. This contribution proposes a solution to this apparent discrepancy by introducing a novel architectural concept for the SGM. A detailed comparison is provided in Section VI.

IV. Parallelization of SGM

A crucial point for VLSI-implementation is the mapping of the algorithm into a parallel-processable and stream-based flow that only requires a single-pass across the input images. Challenges are imposed by the semi-global matching due to the recursively defined paths and their orientations within the images, which are not aligned to a stream-based flow.

Furthermore, all path costs $L_r(\mathbf{p},d)$ for a given pixel $\mathbf{p}$ *and* the minimum $\min_i L_r(\mathbf{p},i)$ must be known in order to calculate $L_r(\mathbf{p}+\mathbf{r},d)$. This remains independent from the approach of parallelization. Therefore, a buffer with $d_{\text{max}}+1$ entries is required for each path that is to be processed in parallel. Let a processing step be defined as the calculation of all path costs over d for a pixel $\mathbf{p}$ and path $\mathbf{r}$ with an arbitrary number of cycles. At the end of a processing step, all path costs $L_r(\mathbf{p},d)$ for a pixel $\mathbf{p}$ are available in the corresponding path cost buffer.

The parallelization concept is shown in Fig. 5 and will be introduced for the path directions of $0°$, $45°$, $90°$, $135°$. Pixels are processed from left to right along the image row ($0°$ path). The positions and directions of these paths in the base image are depicted in Fig. 4b. After processing pixel $\mathbf{p}_{-1}=[x-1,y]$ of the upper row, all path costs over d of all directions are available in the path costs buffers $\left(\mathrm{z}^{-1}\right)$. Path costs are delayed, according to their path directions of $90°$ and $45°$ by one and two *additional* processing steps, respectively. Afterwards, path costs of $L_{45°}(x-3,y,d)$, $L_{90°}(x-2,y,d)$, $L_{135°}(x-1,y,d)$ are available at the output of the path cost buffers. These are exactly those path costs needed for parallel and synchronous calculation of all path costs of all orientations for pixel $\mathbf{p}_2=[x-2,y+1]$. Synchronous calculation allows direct summation of path costs in a pipeline that returns the aggregated costs S.

Therefore, all paths to the pixels $\mathbf{p}_1=[x,y]$ *and* $\mathbf{p}_2=[x-2,y+1]$ are calculated in parallel in a single processing step. This concept is extendable to an arbitrary number of rows. An additional delay by two pixels is introduced for each new row, as illustrated in Fig. 5. Images are separated into image slices of N parallel rows in order to process whole images. Path costs of the last row of an image slice need to be stored and made available to the first row of the next slice according to the method described above. Exceptions need to be considered for boundary pixels with missing stereo overlap in the original images.

Independently of the parallelization concept, the lower limit for the size of the path cost buffers can be given as $d_{\text{max}}+1$ entries per path and pixel that are to be processed in parallel, as the minimum $\min_i L_r(\mathbf{p},i)$ and $L_r(\mathbf{p},d)$ themselves have to be known. The introduced parallelization concept requires additional buffers with $d_{\text{max}}+1$ and $2\cdot(d_{\text{max}}+1)$ entries for the path directions $90°$ and $45°$, respectively. Hence, in total, 7 instead of 4 buffers are employed per row. This additional memory requirement is a small drawback compared to the expected benefits through parallelization.

Generalization of this concept is only limited by the fact that the maximum angle range is $[0,180°)$. This means that no paths in opposite directions can be directly supported without additional hardware. It will be shown in Sec. VI that performance decrease by this specific path reduction is negligible for real-time applications.

This two-dimensional parallelization produces regular data access of the input images and all intermediate values.

V. Hardware Architecture and Implementation

A. System Design

The high computational load requires a system design that allows parallel execution of the image processing tasks as well as massively parallel computation on a data level within each processing task. This is given by the modular coprocessor architecture which has been developed at our research group [23] and is well suited for high-throughput

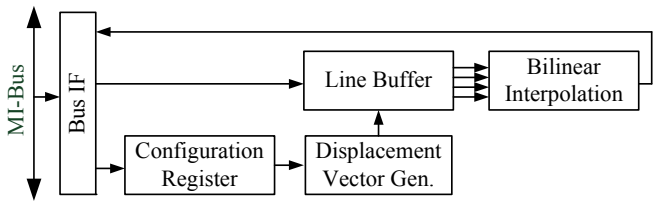


Fig. 7. Top-level block diagram of hardware architecture of the rectification unit.

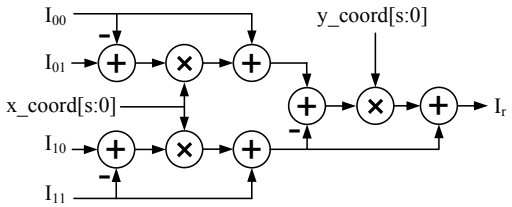


Fig. 8. Implementation of bilinear interpolation to calculate rectified pixels from sub-pixel positions with $s+1$ bit sub-pixel accuracy. Input pixels I_{nm} are ordered correctly by the output logic of the line buffer unit.

image processing as shown in [24]. Further, highly customized data paths are required for the parallelization concept of the SGM, which is fulfilled by utilizing an FPGA or ASIC.

The modular coprocessor architecture comprises a performance-optimized 128-bit multi-layer module interconnect bus (*MIB*) that connects several dedicated processing elements (*PEs*), which are entirely mapped onto an FPGA. A PE is either an interface (*IF*) to an externally connected device (external memory IF, host IF, etc.) or performs a specific image processing task. A RISC processor is connected to the bus via the host IF and can set configuration registers, start and stop PEs, which otherwise run independently. Through the high-level programmability of the RISC processor, a high degree of flexibility for system control and synchronization functions is achieved.

The stereo vision system setup is shown in Fig. 6. In order to achieve real-time throughput, the image processing steps identified in Section II have been realized as highly parallel dedicated processing elements. The cameras are synchronously triggered and the video-in interfaces run with the camera pixel input clock. The video-in interfaces automatically synchronize to the connected cameras and conduct color conversion and image cropping, if needed. Noise reduction can be applied by means of the 2D-FIR filter unit with a kernel size of up to 5×5. Processing of stereo images with the filter unit or rectification unit is done by using the same hardware unit consecutively twice with appropriate parameter setups. In order to reduce latency, these units have been designed with a throughput of one pixel per clock cycle, so that noise reduction and rectification can be executed on both images within the execution time of the disparity estimation unit. The two main modules, rectification and disparity estimation, are discussed in detail below. A basic post-processing step has been included in the SGM unit, but a more sophisticated PE can easily be added due to the modular nature of the system architecture. The video-out interface, running at pixel output clock, generates appropriate synchronization signals and converts the data format according to the sink (DVI, LCD, etc.). The video-out interface and the rendering unit are implemented as one combined PE, which enables direct data exchange between both units without requiring bus access. For data exchange between the PEs, multi-buffering, controlled by the RISC, is used, which allows task-parallel, macro-pipelined processing and avoids tearing (i.e. writing to a frame buffer while another PE still reads from it).

B. Rectification Architecture

The rectification unit consists of four dedicated functional blocks as shown in Fig. 7. The displacement vector for the reverse mapping is calculated on the fly from the stereo camera parameters stored in the bus-accessible configuration registers. Camera parameters are static and are calculated offline once using the OpenCV library [8]. The configuration registers are duplicated to allow fast switching between the two rectification setups for the left and right camera. Calculation on the fly requires additional hardware resources compared to the alternative of storing two-dimensional displacement maps for both images in an external memory, but requires no external bandwidth. Incoming pixels are distributed into four separate buffers (within the line buffer unit) according to odd and even coordinates in x- and y-direction. This allows parallel access to all four neighboring pixels for a given sub-pixel position and, therefore, a throughput of one rectified pixel per cycle, without increasing the total amount of line memory required. Maximum vertical displacement is dictated by the amount of line memory invested, and it has been limited to 64 lines. The line buffer is implemented as a circular buffer to allow gap-free, stream-based processing. An integrated addressing and control unit translates absolute pixel positions and displacement vectors to memory locations in the line buffer. The fully pipelined bilinear interpolation unit has been implemented similar to [25] with a minimum amount of multipliers, as shown in Fig. 8.

The design is parameterized as to the number of bits invested for the fractional precision of the displacement vector calculation and the interpolation. Multiplications and divisions are performed to full word width of the available DSP-slices on the target device to avoid accumulation of rounding errors. However, exploration of the required fixed-point fractional precision has shown that quality improvement stagnates starting with 6 bit fractional precision.

C. Semi-Global Matching Architecture

A block diagram of the hardware architecture for calculation of the disparity maps is given in Fig. 9. Computation of the rank transform of both images and calculation of the data dependent penalty term P_2 is done in parallel and synchronously utilizing the same data path. Adjusting the penalty term P_2 to the characteristics of the rank-transform based matching cost calculation allows for hardware-friendly, division-less calculation.

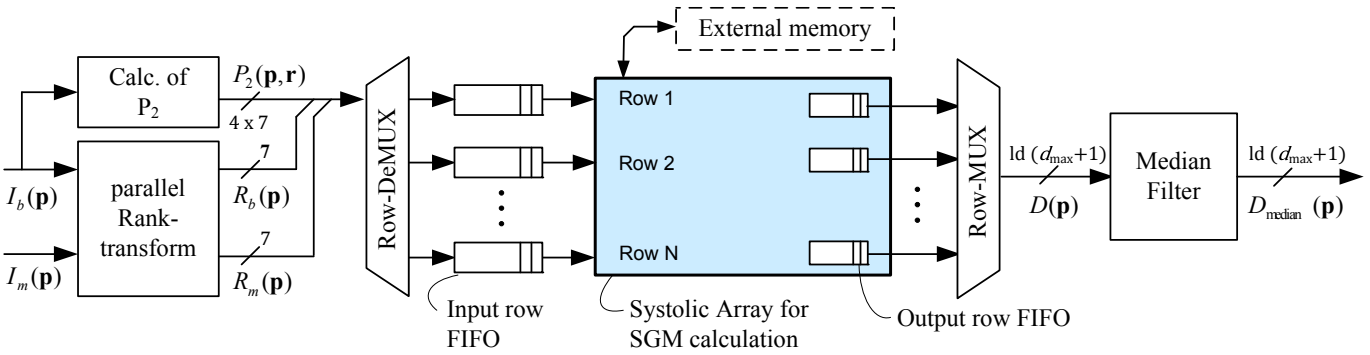


Fig. 9. Hardware architecture for calculation of disparity maps using rank-transform and semi-global matching. The bus interface is not shown.

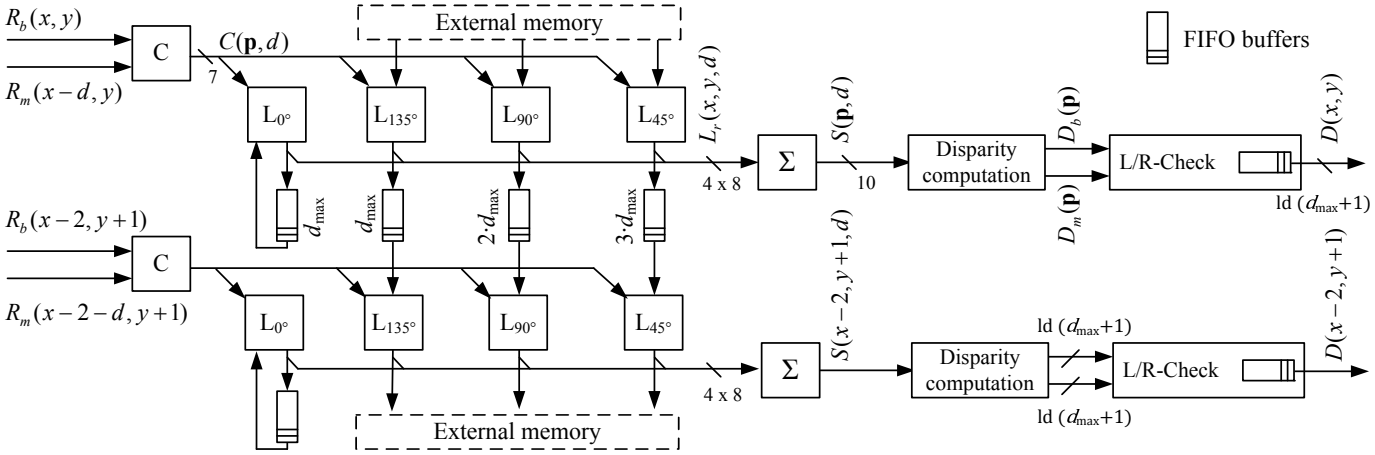


Fig. 10. Hardware architecture of the systolic array for parallel path cost calculation of the semi-global matching for two parallel rows.

An N-row buffer provides this data to the systolic array, which calculates the disparities of all N rows in parallel according to the parallelization concept of section IV. A median filter realizes the outlier suppression.

A heterogeneous, completely synchronized systolic array realizes the parallelization concept for the semi-global matching utilizing path directions from $0°$, $45°$, $90°$, $135°$. Fig. 10 shows the corresponding block diagram. Processing of a pixel $\mathbf{p}$ is carried out sequentially over all disparities of this pixel. The first processing elements (*C-PEs*) calculate the matching costs $C(\mathbf{p}, d)$. Each of the following PEs (*L-PEs*) calculates the path costs L_r along a path $\mathbf{r}$ according to Eq. (5). The results are buffered in the appropriate path cost buffers. All L-PEs are completely identical and the path orientations are solely defined by the delays introduced by the path cost buffers. Path costs are summed to S and then processed by disparity computation PEs (*D-PEs*). D-PEs locate the minimum, i.e. the correct disparity, for the disparity maps D_b and D_m. A final L/R-Check-PE projects the disparity map D_m to the perspective of the base camera, executes the left/right check including occlusion detection, and marks pixels accordingly. A local single row buffer is needed for the projection. It functions simultaneously as an output buffer.

Boundary treatment for pixels with missing stereo overlap (i.e. $x < d_{\max}$) significantly reduces the number of entries of the cost spaces $C(\mathbf{p}, d)$, $L_r(\mathbf{p}, d)$, $S(\mathbf{p}, d)$, and, consequently, leads to a computing time reduction. For VGA images and a disparity range of 128 px the reduction is 9.9 %.

An additional latency of two pixels per parallel processed row is introduced due to the systolic array. However, performance decrease is less than 1 %, because the latency is defined by the pixels on the left image border with high levels of missing stereo overlap. But these non-existing potential disparities are not computed due to the boundary treatment.

An external interim memory is required for storing the path costs of the three non-horizontal paths of the last row of an image slice and providing them to the first row of the consecutive image slice. With respect to the boundary treatment the required memory size is

$$m = 3 \cdot e \cdot \lceil \log_2 (L_{\max} + 1) \rceil \quad [bit] \tag{7}$$

where e is the number of elements per path over all pixels of a row. Let w denote the image width, then it is

$$e = w \cdot (d_{\max} + 1) - \frac{d_{\max}(d_{\max} + 1)}{2}. \tag{8}$$

The required memory size is independent from the number of parallely processed rows. For an exemplary 8-bit VGA-image with 128 px disparity, m is approximately 220 kByte. The data

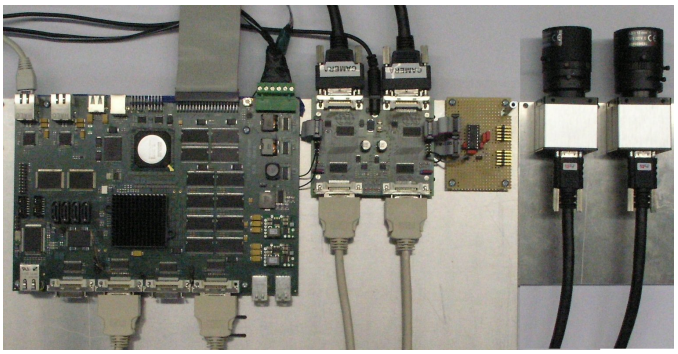


Fig. 11. Hardware setup of the stereo vision system with the internally developed main system board, the converter board, and the stereo camera rig.

rate per second dps can be calculated with the frame rate per second fps and number of image slices as

$$dps = m \cdot (slices - 1) \cdot fps. \tag{9}$$

Due to the extremely regular data transfer, obeying the FIFO-principle, and the low transfer rates, external SSRAM and SDRAM-memories can be used. Alternatively, on-chip memory can be considered due to the quite low absolute memory requirements.

D. Hardware Setup

Fig. 11 shows the hardware setup of the stereo vision system, which was designed at our institute and comprises a Xilinx Virtex-5 LX220T-1 and an Intel IXP 460 ARM XScale RISC. The entire hardware design, including the modular coprocessor architecture and the image processing units, is mapped on the FPGA. The RISC is running a Linux 2.6 kernel and allows configuration, debugging and controlling of the tasks executed on the FPGA. Two synchronously triggered Leutron Vision P83B-RTF cameras with 1033×779px at 30fps are connected via a converter board, which supports Power over CameraLink (PoCL). The results are displayed on a standard LCD and can be sent via a ChannelLink interface to be analyzed on a standard PC.

VI. Experimental Results and Evaluation

The stereo vision system has been implemented on the custom-build, Virtex-5 FPGA-based hardware platform described in the previous section. The kernel size of the rank-transform and median filter is 9×9 and 3×3, respectively.

Maximum performance of the complete system and scalability of the SGM unit are analyzed for the target maximum clock frequency of 133 MHz. The results are given in Table I. The SGM unit achieves, depending on the level of parallelism, frame rates from 37 up to 103 fps for VGA images with 128 px disparity range. This is also the maximum throughput of the system, since multi-buffering is used. The filter and rectification units are fast enough to always support processing of both stereo images consecutively within the execution time of the SGM unit.

TABLE I
Maximum frame rates of the dedicated hardware units for images with 640×480 px at a clock frequency of 133 MHz on a Virtex-5. System throughput is identical to that of the SGM unit.

	d_{Range}	Frame rate
SGM (10 parallel rows)	128 px	37 fps
SGM (20 parallel rows)	128 px	72 fps
SGM (30 parallel rows)	128 px	103 fps
SGM (10 parallel rows)	64 px	66 fps
SGM (20 parallel rows)	64 px	122 fps
SGM (30 parallel rows)	64 px	167 fps
2D-FIR Filter		429 fps
Rectification		369 fps

TABLE II
Minimum clock frequencies and SGM memory bandwidth for a fixed resolution of 640×480 px with 128 px and 64 px disparity range at 30 fps and maximum latency of 2 frames

	d_{Range}	Ext. memory data rate	Clock frequency
SGM (10 rows)	128 px	297.6 MB/s	108 MHz
SGM (20 rows)	128 px	143.8 MB/s	55 MHz
SGM (30 rows)	128 px	93.9 MB/s	39 MHz
SGM (10 rows)	64 px	155.5 MB/s	60 MHz
SGM (20 rows)	64 px	75.5 MB/s	33 MHz
SGM (30 rows)	64 px	49.3 MB/s	24 MHz
2D-FIR Filter and Rectification			31 MHz

However, in real-world applications the required throughput is usually specified by external circumstances (e.g. by the cameras, required depth resolution, etc.). The goal of the designer is then to find the pareto optimal point for this specification in terms of power, latency, throughput and resource usage. Table II provides the results for a typical parameter set of 640×480 px at 30 fps under the condition that both, filtering and rectification, are executed within the time of one SGM pass to keep a low latency. This results in a clock frequency of 31 MHz for the filtering and rectification unit, which are clocked with the same frequency to reduce the amount of different clock domains. With increasing parallelism, the SGM unit can be clocked with lower frequencies, while reducing the required external bandwidth at the price of higher area requirements.

FPGA resource usage is summarized in Table III. Resource usage for the systolic array of the SGM unit increases linearly with the number of parallel processed image rows. Path cost buffers have been instantiated with Distributed RAM, which utilizes LUTs as RAM (*LUTRAM*). The path cost buffers account for about 10 % of the LUTs utilized by the SGM unit. Due to the small size of required external memory (here: 220 kByte), it has been emulated using BlockRAMs (*BRAM*). These are not included in the resource usage report, as the use of external memory is assumed, which has no influence on the performance.

Algorithmic performance is often measured by percentages of erroneous disparities on the Middlebury test images [26]. These are given in Table IV. The error rates are

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

Fig. 12. Results computed with the proposed stereo vision system: (a) Left input image of a real-world scenario, (b) disparity map; (c) left camera image of a scene from the lab, (d) raw disparity map and (e) false-color result rendered for optical viewing including the distance measure in real-world coordinates.

TABLE III
RESOURCE USAGE ON AN VIRTEX-5 XC5LX220T WHICH IS INDEPENDENT OF THE DISPARITY RANGE. RESOURCES OF ALL BUS INTERFACES ARE INCLUDED IN THE SYSTEM INFRASTRUCTURE.

Unit	LUTs	LUTRAM	18kb-BRAMs	DSP-Slices
SGM (10 rows)	17743	2336	40	0
SGM (20 rows)	31606	4896	80	0
SGM (30 rows)	45614	7456	120	0
Parallel Rank-T.	3231	0	9	0
Median filter	961	0	3	0
Rectification	3537	0	36	15
2D-Filter (5×5)	2052	0	5	25
Video IF (×3)	6100	0	4	9
Rendering	1422	10	0	1
System infrastruc.	16989	0	78	0
Total (10 rows)	68427	2346	183	50
Perc. of available	51,2 %		43,2 %	39,1 %

TABLE IV
PERCENTAGES OF ERRONEOUS DISPARITIES WITH A THRESHOLD OF 1 px IN NON-OCCLUDED AREAS WITHOUT POST-PROCESSING (E.G. INTERPOLATION) FOR THE MIDDLEBURY TEST SETS .

No. of paths	*Cones*	*Teddy*	*Tsukuba*	*Venus*
4 paths	9.5 %	13.3 %	6.8 %	4.1 %
8 paths	8.4 %	11.4 %	4.1 %	2.7 %

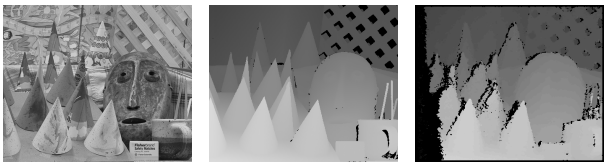


Fig. 13. Results for the *Cones* image set: (a) left image, (b) ground truth and (c) disparity map calculated on the proposed architecture.

typical for current hardware implementations due to the near-ideal imaging conditions of the test images. Fig. 13 shows the results for the *Cones* dataset. However, real-world scenarios do have difficult and uncontrollable lighting conditions. A typical result from the *.enpeda.* data set [27] is shown in Fig. 12a-b. Furthermore, in Fig. 12c-e the results for a scene which has been recoded at our laboratory are shown. The disparity map in Fig. 12e has been processed by the rendering module, representing distances as colors in the original image. Real-world distances are calculated from the disparities and the camera calibration parameters. For these results no post-processing such as peak-removal or interpolation has been conducted, which would further improve the results.

The path reduction necessary for the parallelization concept from 8 to 4 paths leads to a minimal increase of the error rate by an average of 1.7 percentage points on the Middlebury test images. In real-world sequences a minor anisotropic effect is observed, since the paths directions are not evenly distributed over the image.

A detailed comparison of real-time disparity estimation systems is complex due to the number of deployed algorithms, architectures and evaluation platforms. In order to perform a fair benchmark a platform-independent metric, the required minimum clock frequency of a given implementation to fulfill a specific (real-time) constraint, is used. This metric is independent of the utilized evaluation platform because it is independent of the maximum achievable clock frequency [28]. This does not, however, consider how the design scales for higher disparity ranges. An equally important aspect is the algorithmic performance of the deployed algorithm, where SGM outperforms local stereo matching methods.

The most recent implementation of a local stereo matching method has been presented by Jin *et al.* [18] and is based on a local census-transform/correlation approach. The minimum clock frequency for 640×480 px images with 64 px disparity range at 30 fps is 12 MHz. Although the proposed architecture for the SGM requires 60 MHz, it requires slightly less logic resources and a significantly lower amount of BRAMs. Sabihuddin *et. al.* [22] implemented the DPML-algorithm (*dynamic programming maximum likelihood*) on a Virtex-II Pro. The minimum clock frequency for 128 px disparity range is 38 MHz compared to 108 MHz here. Even though the utilized dynamic programming approach is a global method, it suffers from inter-scanline inconsistencies (*streaking*). This is reflected in the error rate for the Middlebury test image *Venus* of 13,5 % compared to the error rate of 4,1 % achieved by the systolic array. Although these two implementations have a higher throughput by factor 3 to 5 both implementations are based on algorithms that do not reach the algorithmic performance of the SGM in terms of accuracy and robustness.

The only implementation of the SGM known to the authors has been presented by Gehrig *et al.* [4]. It utilizes 8 paths by instantiating two SGM units, one for 4 path each. Due to the introduced sub-sampling, the minimum clock frequency for the normalized performance conditions of 30 fps for 640×480 px with 64 px disparity range is 664 MHz. For the same performance demands, the minimum frequency of the architecture

proposed here is 60 MHz. This is a performance increase of factor 11 while logic resources are comparable and the amount of BlockRAMs is halved. This comparison demonstrates the efficiency of the proposed parallelization concept.

VII. Conclusions

The modular stereo vision system integrates all necessary image processing tasks as separate processing elements, allowing task- and pixel-parallel processing. Key feature is the use of the semi-global matching disparity estimation algorithm, which is among the top-performing stereo matching methods and vastly outperforms locally-based methods usually employed in hardware implementations. This is a mandatory step towards the robustness required for camera-based driver assistance systems.

Main architectural feature to obtain real-time capability is the novel two-way parallelization concept for semi-global matching, which ensures efficient, massively parallel computation. Moreover, the systolic-array-based architecture for the SGM is highly scalable and allows adjustment of the pareto optimal point for a given application, in terms of resources, throughput and latency, as the number of parallely processed rows can be freely chosen. The required clock frequency for generating disparity maps including left/right check under real-time conditions (30 fps) for VGA images and a disparity range of 128 px is, depending on the degree of parallelism, between 39 MHz and 108 MHz. These clock frequencies pose no challenges on todays FPGAs. Compared to [4], which is the only other hardware-implementation of the SGM known to the authors, this is a performance increase of factor 11.

Even though the architecture is scalable and larger image sizes can processed, future work includes extending the systolic array by another layer of parallelization to increase throughput without an increase in local buffer requirements. The SGM architecture will also be extended to compute 8 paths to remove the anisotropic effect. Additionally sub-pixel estimation will be integrated to increase the precision of the distance measurement.

Acknowledgment

This work has been supported in part by the Hans L. Merkle-Stiftung (Stifterverband für die deutsche Wissenschaft). The authors are grateful for the financial support.

References

[1] J. Woodfill, G. Gordon, and R. Buck, "Tyzx DeepSea High Speed Stereo Vision System," in *Computer Vision and Pattern Recognition Workshop. Conference on*, June 2004, pp. 41–41.

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

[3] H. Hirschmüller and D. Scharstein, "Evaluation of Cost Functions for Stereo Matching," *Computer Vision and Pattern Recognition. IEEE Conference on*, pp. 1–8, 2007.

[4] S. Gehrig, F. Eberli, and T. Meyer, "A real-time low-power stereo vision engine using semi-global matching," in *Computer Vision Systems: International Conference on,*. Springer, 2009, pp. 134–143.

[5] H. Hirschmüller, "Accurate and Efficient Stereo Processing by Semi-Global Matching and Mutual Information," in *Computer Vision and Pattern Recognition, IEEE Conference on*, vol. 2, 2005, pp. 807–814.

[6] J. Gibson and O. Marques, "Stereo Depth with a Unified Architecture GPU," in *Computer Vision and Pattern Recognition Workshops, IEEE Computer Society Conference on*, 2008, pp. 1–6.

[7] G. Payá-Vayá, J. Martín-Langerwerf, C. Banz, F. Giesemann, P. Pirsch, and H. Blume, "VLIW Architecture Optimization for an Efficient Computation of Stereoscopic Video Applications," in *IEEE International Conf. on Green Circuits and Systems. Accepted for publication.*, 2010.

[8] G. Bradski and A. Kaehler, *Learning OpenCV: Computer Vision with the OpenCV library*. O'Reilly Media, Inc., 2008.

[9] R. Zabih and J. Woodfill, "Non-parametric Local Transforms for Computing Visual Correspondence," in *European Conference on Computer Vision*, 1994, pp. 151–158.

[10] H. Hirschmüller, "Stereo Processing by Semiglobal Matching and Mutual Information," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 30, no. 2, pp. 328–341, 2008.

[11] M. Hariyama, N. Yokoyama, M. Kameyama, and Y. Kobayashi, "FPGA Implementation of a Stereo Matching Processor Based on Window-parallel-and-pixel-parallel Architecture," in *Circuits and Systems, 48th Midwest Symposium on*, Aug. 2005, pp. 1219–1222 Vol. 2.

[12] Y. Jia, X. Zhang, M. Li, and L. An, "A Miniature Stereo Vision Machine (MSVM-III) for Dense Disparity Mapping," *Pattern Recognition. 17th International Conference on*, vol. 1, pp. 728–731 Vol.1, 2004.

[13] S. Lee, J. Yi, and J. Kim, "Real-time Stereo Vision on a Reconfigurable System," *LNCS*, vol. 3553, p. 299, Springer 2005.

[14] S. Perri, D. Colonna, P. Zicari, and P. Corsonello, "SAD-Based Stereo Matching Circuit for FPGAs," in *Electronics, Circuits and Systems. 13th IEEE International Conference on*, 2006, pp. 846–849.

[15] C. Murphy, D. Lindquist, A. M. Rynning, T. Cecil, S. Leavitt, and M. L. Chang, "Low-Cost Stereo Vision on an FPGA," *Field-Programmable Custom Computing Machines. 15th IEEE Symp. on*, pp. 333–334, 2007.

[16] K. Ambrosch, W. Kubinger, M. Humenberger, and A. Steininger, "Flexible Hardware-based Stereo Matching," *EURASIP Journal on Embedded Systems*, 2008.

[17] *MobilEye: EyeQ2 Vision-system-on-a-chip*. [Online]. Available: `www.mobileye.com/manufacturer-products/brochures`

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

[19] D. Masrani and W. MacLean, "A Real-Time Large Disparity Range Stereo-System using FPGAs," *Computer Vision Systems. IEEE International Conference on*, pp. 13–13, Jan. 2006.

[20] J. Diaz, E. Ros, R. Carrillo, and A. Prieto, "Real-Time System for High-Image Resolution Disparity Estimation," *Image Processing, IEEE Transactions on*, vol. 16, no. 1, pp. 280–285, 2007.

[21] S. Park and H. Jeong, "Real-time Stereo Vision FPGA Chip with Low Error Rate," in *Multimedia and Ubiquitous Engineering. International Conference on*, 2007, pp. 751–756.

[22] S. Sabihuddin, J. Islam, and W. J. MacLean, "Dynamic Programming Approach to High Frame-Rate Stereo Correspondence: A Pipelined Architecture Implemented on a Field Programmable Gate Array," *Electrical and Computer Engineering. IEEE Can. Conf. on*, pp. 1461–1466, 2008.

[23] H. Flatt, S. Hesselbarth, S. Flügel, and P. Pirsch, "A Modular Coprocessor Architecture for Embedded Real-Time Image and Video Signal Processing," in *Embedded Comp. Systems: Architectures, Modeling, and Simulation*, vol. 4599. Springer, 2007, pp. 241–250.

[24] H. Flatt, I. Schmädecke, M. Kärgel, H. Blume, and P. Pirsch, "Hardware-Based Synchronization Framework for Heterogeneous RISC/Coprocessor Architectures," in *Embedded Comp. Systems: Architectures, Modeling and Simulation; IEEE Conf. on*, 2009, pp. 125–132.

[25] S. Fahmy, "Generalised Parallel Bilinear Interpolation Architecture for Vision Systems," in *Reconfigurable Computing and FPGAs, International Conference on*, 2008, pp. 331–336.

[26] D. Scharstein and R. Szeliski, "High-Accuracy Stereo Depth Maps using Structured Light," *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, vol. 1, pp. I–195–I–202 vol.1, 2003.

[27] T. Vaudrey, C. Rabe, R. Klette, and J. Milburn, "Differences between Stereo and Motion Behaviour on Synthetic and Real-world Stereo Sequences," in *Image and Vision Computing New Zealand. International Conference*, 2008, pp. 1–6, `http://www.mi.auckland.ac.nz/EISATS`.

[28] G. Payá-Vayá, J. Martín-Langerwerf, and P. Pirsch, "A Multi-Shared Register File Structure for VLIW Processors," *Journal of Signal Processing Systems*, vol. 58, no. 2, pp. 215–231, 2010.