INTERNATIONAL JOURNAL OF RECENT TRENDS IN ENGINEERING & RESEARCH
Impact Factor 4.101 ISSN -2455-1457
RECENT ADVANCES IN ELECTRONICS & COMMUNICATION ENGINEERING (RAECE-2K17)

Design of Central Processing Unit for Synchronization Using Image Sensors And FPGA

M.Varaprasad
Assistant professor, ECE Department
Visvesvaraya College of Engineering & Technology

K.Ramesh
Assistant professor, ECE Department
Visvesvaraya College of Engineering & Technology

***Abstract*— Recent developments in smart phones create an ideal platform for robotics and computer vision applications: they are small, powerful, embedded devices with low-power mobile CPUs. However, though the computational power of smartphones has increased substantially in recent years, they are still not capable of performing intense computer vision tasks in real time, at high frame rates and low latency. We present a combination of FPGA and mobile CPU to overcome the computational and latency limitations of mobile CPUs alone. With the FPGA as an additional layer between the image sensor and CPU, the system is capable of accelerating computer vision algorithms to real-time performance. Low latency calculation allows for direct usage within control loops of mobile robots. A stereo camera setup with disparity estimation based on the semi global matching algorithm is implemented as an accelerated example application. The system calculates dense disparity images with 752x480 pixels resolution at 60 frames per second. The overall latency of the disparity estimation is less than 2 milliseconds. The system is suitable for any mobile robot application due to its light weight and low power consumption.**

I. INTRODUCTION

Cameras are the ideal sensors for many applications, because they provide a large amount of information about the environment at high frame rates. Image sensors are also small, light-weight, and have low power consumption, which makes them ideal for embedded applications, where other sensors such as LIDAR are impractical to use. The main drawback of image sensors is the amount of image processing required to obtain usable information about the environment.

Even though most smartphones now feature at least one image sensor and can record and play back videos at high resolutions, they lack sufficient processing power to run sophisticated computer vision algorithms in real time. Cameras are widely used in mobile robot applications for obstacle detection and localization. However, because of the processing power required, they still use full-size CPUs for data processing, which have larger weight and power requirements. Using CPUs in the control loop of robotic systems also increases the latency between image acquisition and processed output, which is undesirable for high-speed robotics applications.

To address these problems, we present a computationally powerful system based on a combination of a Field Programmable Gate Array (FPGA) and a mobile CPU. This combination allows us to overcome the processing power

The authors are with the Computer Vision and Geometry Group, Institute for Visual Computing, Computer Science Department, ETH Zürich, 8092 Zürich, Switzerlandlimitations of mobile CPUs alone, while maintaining their low power consumption, light weight, and small package. Exploiting the parallel architecture of the FPGA allows us to parallelize the vision processing, decreasing latency and increasing frame rate and resolution. Subsequently, the mobile CPU receives original and processed image data and can be used for more advanced tasks such as mapping and path planning.

Stereo vision is important for robotics applications, since it gives dense 3D information about the environment. We implement a full stereo pipeline, including camera lens distortion correction, image rectification and stereo matching based on epipolar geometry as an intellectual property (IP) core in the FPGA. The output is a combination of the original image and processed disparity image, which appears simply as image data to the mobile CPU, allowing the user to not worry about the implementation details of the FPGA. The core can process the image data streams of two image sensors with 752x480 pixels resolution running at 60 frames per second (fps). A dense disparity map at full resolution and frame rate based on the semi global matching (SGM) algorithm [8] is calculated to show the potential of the FPGA. The overall latency of the disparity estimation pipeline is less than 2 milliseconds (ms) and the power consumption for the system in full operational mode is less than 5 Watts. The small size and weight of the system make it ideal for applications in constrained environments, such as mobile robots or even micro aerial vehicles (MAVs). The contributions of this work are as follows: we first show related work concerning computer vision on constrained platforms and existing implementations on FPGAs. Second, we introduce our general system setup. The efficient im- plementation of the SGM algorithm within the FPGA and details on all the major hardware components is shown afterwards. Lastly, we provide some possible applications for our device and show results of the different IP modules including latency and power consumption.

II. RELATED WORK

FPGAs are successfully used to accelerate computer vision algorithms. Due to its parallel characteristics, stereo matching fits well within the FPGA architecture. A real-time stereo matching implementation based on a census rank transformation by [14] is shown in [10]. The implementation in [7] calculates disparity

values for high-definition stereo video at full frame rate. The system in [9] produces disparity maps at 127 fps and 376x240 pixels resolution based on block matching. [6] and [1] show an implementation of eal-time SGM-based stereo [8] on FPGA, producing disparity maps at 640x480 resolution at 25 fps. However, these implementations are shown on high-end, large FPGA development boards, and furthermore do not address the connection of the cameras to the system. Both of these implementations use the FPGA as a co-processor, while in our system, the FPGA runs the complete pipeline.

A combination of FPGA, CPU and mobile CPU mounted on a quadrotor is shown in [12]. The system uses a FPGA and an Intel Core2Duo CPU to calculate high quality depth images with 752x480 resolution at 15 fps. The cameras are connected via USB to the CPU. The CPU performs a planar rectification and corrected image data is sent to the FPGA afterwards. Estimated disparity values are sent back to the CPU to perform post processing steps. Due to the data transfer between the devices the overall latency of their disparity estimation pipeline is quite high with 100 ms. However the system is successfully used for ego-motion computation on a quadrotor.

The vision based quadrotor shown in [5] uses an Intel Core2Duo CPU to process the data stream of a stereo camera head. Successful navigation based on a dense disparity map with limited frame-rate is shown.

In [11], the authors present a method to estimate the three dimensional motion field out of stereo sequences. A real-time implementation with a combination of FPGA and GPU is able to calculate a dense motion field at 10 frames per second. Since the system is targeted for automotive driver assistance, they do not consider the same size and weight restrictions as MAVs.

A low power and small scale FPGA system to determine optical flow within MAVs for position control is presented in [13]. 2D optical flow estimation with a single FPGA from up to 11 image sensors is proposed.

III. SYSTEM SETUP

This section shows the general system structure of the FPGA-mobile CPU combination and presents the modules of the stereo example application. An overview of the setup is shown in Figure 1. The FPGA is placed as an additional layer between image sensors and mobile CPU. It acts towards the CPU as a regular image sensor, which allows for an easy integration into the existing frame grabber infrastructure of the mobile CPU.

The image sensors are directly connected to the FPGA using low-voltage differential signaling (LVDS) interfaces. The data streams of the cameras are processed in real-time within the FPGA. Processed and raw image data streams are combined and sent to the mobile CPU using the dedicated

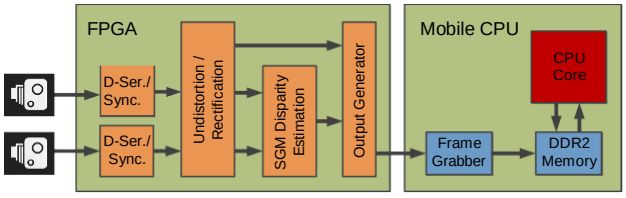


Fig. 1: System overview, the FPGA is placed between the mobile CPU and image sensors. These are directly connected to the FPGA. The video streams are deserialized, undistorted and rectified. Afterwards disparity estimation is performed. Processed and raw image data is synchronized in the output generator. Finally, data is sent to the frame grabber of the mobile CPU using its dedicated imager bus.

a horizontal resolution of three times that of the original camera. This allows to stream out left, right image, and the disparity map side by side. The synchronization is performed within the FPGA to maintain a simultaneous output of raw data and processed data.

A. De-Serialization and Synchronization Block

The high speed LVDS serial link from the image sensor is received and de-serialized within this block. The image sensors serialize pixel and control data and stream them out using a LVDS link. This allows for undisturbed data transmission using only a single pair of cables.

A synchronization of the data streams of the independent image sensors is performed if necessary. A buffer is used to delay one data stream to end up with two pixel-synchronized data streams. The image sensors stream out pixel values sequentially row by row, starting with the top left pixel of the captured image.

B. Undistortion and Rectification Block

Lens distortion correction and rectification are combined into a single operation since they are both warp operations. Incoming pixel values are stored in a buffer. The coordinate of the correct pixel location with respect to lens distortion and rectification is calculated and used to find the corresponding pixel value in the buffer. Radial and tangential distortion parameters as described in [4] are taken into account. Distorted coordinates x_d, normalized with respect to focal length f_c and principal point c_c, are calculated in real-time according to

$$x_d = (1 + \kappa_1 r^2 + \kappa_2 r^4 + \kappa_3 r^6)x + x_t. \quad (1)$$

Where x denotes the original normalized coordinate and r the distance to the principal point

imager bus. A frame grabber module captures the data and stores it in system memory of the mobile CPU using direct memory access. This allows for guaranteed transmission time of image data between sensor and mobile CPU.
Resolution and frame rate of the received image at the mobile

CPU are fully flexible. For example, in a stereo camera setup with disparity computation, the resulting image could have

$$r^2 = x_1^2 + x_2^2 \quad (2)$$

and

$$\lfloor \xi_1 (r^4 + 2x_2^2) + 2\xi_2 x_1 x_2 \rfloor$$

represents the shift caused by the tangential distortion.
The radial distortion parameters $[\kappa_1, \kappa_2, \kappa_3]$, tangential

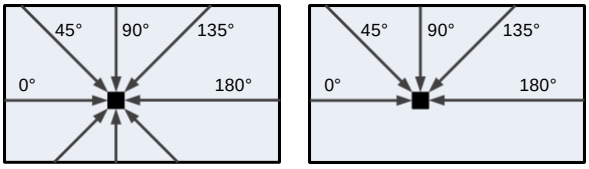


Fig. 2: Eight possible directions for the cost paths in the SGM algorithm are shown on the left. Paths from bottom to top increase the latency of the disparity estimation. The five used directions without increasing the latency are shown on the right.

distortion parameters $[\xi_1, \xi_2]$, camera intrinsic parameters $[f_{cx}, f_{cy}, c_{cx}, c_{cy}]$ as well as the homography for rectification are estimated using camera calibration methods as described in [3]. The outputs of this block are undistorted and rectified pixel values where the epipolar lines are matched with the horizontal direction of the image.

C. SGM Stereo Block

Stereo matching with epipolar geometry is performed on the two data streams after the undistortion and rectification block. The best disparity matching candidate is selected depending on a local cost function and global consistency constraints based on the SGM stereo algorithm.

The constraint costs are aggregated along independent one-dimensional paths across the image, starting at the image border. Since images are sent out line by line starting at the top left pixel, only top to bottom optimization path directions are used in order to not increase the latency. The drawback of using only five paths is a slightly worse matching performance. In [1] a comparison between eight and four paths is made using the Middlebury set. Figure 2 shows the possible directions of the paths. The aggregated costs of a pixel p along a path a for a disparity candidate d are given by he independent data streams of the different IP blocks are combined to a single stream and sent to the mobile CPU in

where $C(p, d)$ denotes the local cost function at pixel d for a disparity candidate d. The minimal path cost from the previous pixel in the direction of the path is added to the local cost function. P_1 and P_{2a} are penalty values for changes or discontinuities in the disparity values. Subtracting the minimal path costs from the previous pixels among all disparity candidates prevents the path costs from increasing monotonically. A detailed description of the parameters is given in [8].

The local costs and global constraint costs for all disparity candidates and path directions are calculated in real-time on the pixel data stream.

Afterwards, a left-right consistency check to detect occluded regions is performed on the estimated disparity value. Finally, a median filter removes spikes in the left-right checked disparity values. The output of this block is a disparity value stream.

D. Output Generator

this block. Control signals to the image sensors are generated according to the selected output resolution and frame rate. Buffers are used to maintain the synchronization of the different data streams.

E. FPGA Infrastructure

Besides the Computer Vision IP cores there are other components included in the FPGA. We instance a Microb-laze soft-core CPU to perform maintenance tasks, such as receiving parameters and configuring the image sensors over an I2C interface.

This small CPU is connected to the IP modules, but is not involved in processing the image data stream. It is tightly coupled with registers of the IP modules to set the parameters in the undistortion and rectification module and adjust the penalty values of the SGM module.

A phase-locked loop is instanced to generate the different clocks needed by the IP cores as well as by the image sensors. Both image sensors are connected to the same clock domain, and after a synchronous reset their internal state ma-chines are in the same state. This leads to a synchronization on a per-pixel basis which supersedes any synchronization buffers and lowers the latency.

IV. IMPLEMENTATION

In the following section, we describe the architecture of the developed computer vision IP cores performing synchro-nization, lens distortion correction and real-time SGM stereo matching in detail.

A. De-Serialization and Synchronization

The image sensor outputs a start bit, an 8-bit pixel value,

using a single LVDS line with 12x pixel clock speed. An instanced 6:1 de-serializer generates an internal 6-bit signal at 2x pixel clock rate and a successive 2:1 de-serializer reconstructs the original 12 bit signal at pixel clock speed. The synchronization of the serial frames is performed using the included start and stop bits. The image sensors output pixels at 25 MHz clock speed, which results in a 300 MHz transmission of the serialized data over the LVDS line.

B. Undistortion and Rectification

The synchronous data streams of both cameras are then connected to the undistortion and rectification module. Both data streams are stored in an internal buffer. In parallel, the distorted and unrectified address is computed with respect to the camera calibration parameters as described in Eq. (1).

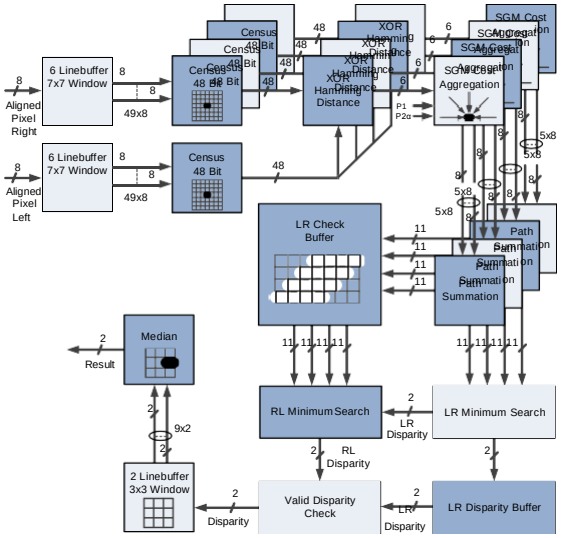


Fig. 3: Disparity estimation data path for a simplified disparity search range of four pixels. The census cost is calculated based on a 7x7 pixel window in parallel for the four candidates. SGM costs are aggregated for five independent paths in parallel for every disparity candidate. The candidate with the minimal sum of the five aggregated path costs is taken as the valid disparity output. A left-right consistency check is performed to detect occluded regions, and finally a 3x3 pixel sized median filter removes spikes.

The generated address points to the buffer location where the pixel values are stored to create an undistorted and rectified data stream. As the addresses point to fractions of buffer positions, bilinear interpolation is performed with the four closest neighbours to create an accurate output. The precision of the fixed point algorithm is chosen to fit in the available 18-bit multiplication units. A delay of several lines is included with this module, since the pixel displacement caused by the lens distortion and the rectification forces a buffer of adequate size.

The address generator unit is time-shared between both cameras to save multiplication units. Therefore, the clock speed is doubled within the address calculation of the module.

C. SGM Stereo

A local cost computation, based on the census rank transformation [14], is performed in the SGM stereo block. A 7x7 pixel window is used to create the census mask. The hamming distance, which is the number of bits different in the two census masks, measures the dissimilarity between the two masks. Since the census mask is less sensitive to illumination changes than other algorithms (such as sum of absolute differences) the internal automatic gain and expo-

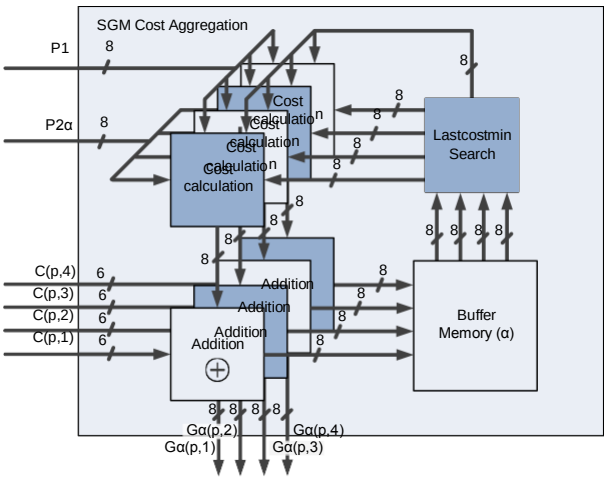


Fig. 4: Cost aggregation data path, simplified example for

sure functions of the image sensors can act independently from each other.

Figure 3 shows an overview of the disparity estimation module. The SGM costs are dependent on the actual census cost and the minimum of the previous SGM costs along the

a disparity search range of four pixels. Local cost functions $C(p, d)$ and path costs are added to the final cost $tt_a(p, d)$ along the path in direction a. Costs are also stored in a buffer and reused to calculate the next cost depending on the minimal costs and the penalty values P_1 and P_{2a}. The size of the buffer is dependent on the direction a of the path.

according direction. To find the minimum SGM cost from the previous step among all disparity candidates and calculate the new SGM cost, the module internal clock is a multiple of the pixel clock speed. This allows for a calculation of SGM path costs in real-time.

In order to not increase the overall latency, only cost path directions from top to bottom are considered . Additional the direction from right to left, reverse the pixel stream, can be added as well, increasing the latency by only two lines. Figure 2 shows the five possible cost paths directions without increasing the latency of the disparity estimation.

The architecture of the cost path calculation is identical for all directions. Different directions are achieved by buffering the calculated costs in memory modules. The size of the buffer defines the path direction. Figure 4 shows the archi- tecture of a single path constraint cost calculation module based on Eq. (4).

The smoothing penalty factors P_{2a} are inversely proportional to the difference of the pixel values of neighbouring pixels in the according direction of path a. This allows for smooth areas within the disparity image by not diffusing gaps caused by a sudden depth change in the scene. P_1 is constant and empirically determined.

The costs for every direction and every disparity candidate are accumulated and stored in independent internal memory blocks of the FPGA. Disparity values are estimated with pixel accuracy.

D. Android Kernel

We developed a custom camera driver for the Android system running on the mobile CPU. The Android camera

stack handles the connected FPGA as a regular camera. Image data is transfered using direct memory access without stressing the mobile CPU. It is possible to do image capturing and even video recording at full frame rate with real-time H.264 encoding. By exploiting the dedicated encoding hardware present in the SoC module, the mobile CPU is not stressed, leaving more resources for higher-level processing and other applications.

E. Hardware Components

We use MT9V034 CMOS image sensors from Aptina. They have a global shutter architecture and provide images with 752x480 pixels resolution at up to 60 frames per second. A small sized FPGA board from Enclustra[1] with the shape of a SO-DIMM memory module is used. This board is equipped with a Xilinx Artix7 XC7A100T FPGA. The Artix FPGA family offers lowest cost and lowest power consumption among the different available families.

As mobile CPU we use a Samsung Exynos 4412 System on Chip (SoC) Module with a built in Cortex-A9 Quad Core CPU. There is a dedicated camera interface available to receive image data from the FPGA. 2 GByte DDR2 memory and a separate 3D accelerator make the module a powerful platform. The SoC module offers also a lot of common interfaces as HDMI, USART, USB and a SD card slot.

V. RESULTS

In this section, we show the implementation results and resource usage of the IP cores instanced on the FPGA. We then show the output of the different IP cores. Finally, we present the specifications of the system.

A. Implementation

The used resources of the SGM stereo core with 32 disparity candidates, the lens distortion correction and epipolar line rectification core are shown in Table I. Additional the resources occupied by the implemented soft-core CPU performing maintenance tasks are shown. The SGM stereo block consumes most of the available slices as every disparity candidate and according path costs are computed in parallel. 50 lines per camera are buffered in the rectification and undistortion module, which is sufficient in most situations and even for poorly aligned cameras.

The IP blocks are pipelined and run at 25 MHz pixel clock speed, with a few exceptions. The multiplication units within the undistortion rectification block are time shared between both image streams and run at 50 MHz. The SGM cost aggregation modules run at 250 MHz to support cost calculation between successive pixels. The overall system supports a pixel clock speed up to 75 MHz and is therefore able to process resolutions and frame rates up to 720p60. Image data of the right camera processed by the undistortion and rectification module is shown in Figure 5b. The output of the SGM stereo module is presented in Figure 5c.

Module	Slice Registers	occupied Slices	Embedded Memory (Kbits)	DSP 48E1 Units
Undist.& Rect.	6431	1225	1000	41
SGM Stereo	30633	8429	2377	0
Microblaze	12633	4827	76	3
Others	440	82	19	0
Total	50137(39%)	14563(91%)	3472(71%)	44(18%)

TABLE I: Used resources of the SGM stereo design including distortion correction, rectification and Microblaze Soft-Core CPU, all implemented on an Artix7 FPGA.

B. Specifications

We present the specifications of the FPGA-mobile CPU combination with disparity estimation as an accelerated example in Table II. Our system produces dense disparity maps based on the SGM algorithm in real-time. The FPGA implementation is pipelined at pixel clock speed of 25 MHz. Resolution of 752x480 pixels and frame-rate of 60 fps is limited by the used image sensors, not the computational power of the system. The overall latency of the disparity estimation pipeline is 2 ms, corresponding to 60 lines at pixel clock speed, and is mostly caused by the buffer in the lens distortion correction module.

The total power consumption is less than 5 Watts for the disparity estimation including cameras, FPGA, mobile CPU and power converters. The system is small in size at 76 mm by 46 mm and light weight at 50 grams. Figure 5a shows the baseboard.

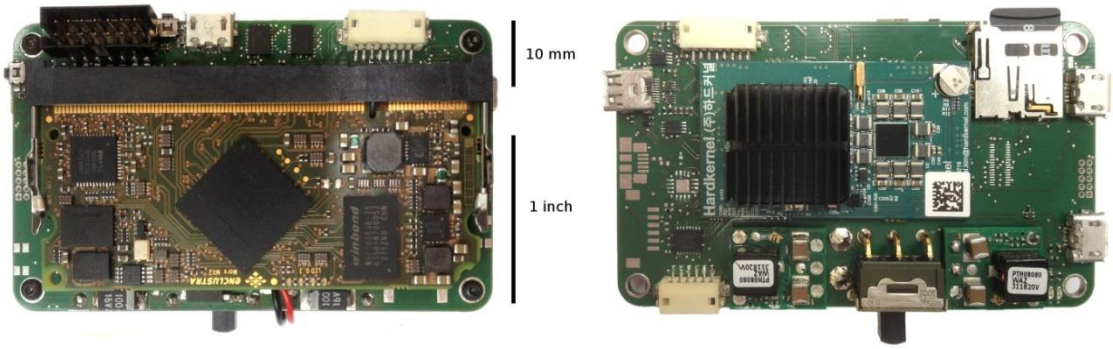
VI. CONCLUSION

In this work, we have presented a small size, light weight, and low power consumption system for doing vision processing on FPGA and mobile CPU. The implemented example of SGM-based stereo matching, running at 60 Hz at 752x480 resolution, is a substantial performance improvement over any other system available in the form factor and power budget. This makes the system perfect for mobile robotics applications, especially for MAVs. Additionally, because the dynamics of MAVs are very fast, our system's low latency and high update rate are essential for obstacle detection and state estimation. For example, high speed maneuvers with fixed-wing airplanes such as described in [2] require on-board processing in real-time in order to work outside of a motion capture environment.

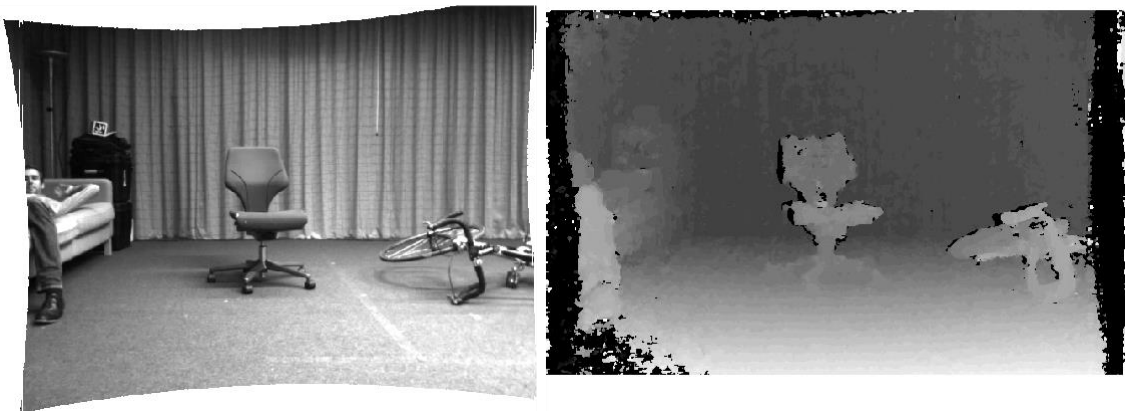
Since the configuration of the FPGA is interchangeable, the stereo core can be replaced to accelerate any other algorithm. Using pre-compiled IP blocks even inexperienced

Characteristic	Value	Unit	Characteristic	Value	Unit
Size	76x46	mm	Resolution	752x480	pixel
Weight	50	g	Frame rate	60	s^{-1}
Power Consumption	5	Watts	Latency	2	ms
			Pixel Clock	25	MHz

TABLE II: Overview of the technical specifications.



(a) Baseboard bottom side with FPGA on the left, and top side with mobile CPU on the right



(b) Right corrected image (c) Disparity map

Fig. 5: Photo of baseboard with FPGA and mobile CPU in (a), right image after lens distortion correction and rectification in (b), and the disparity map calculated in the SGM stereo module is shown in (c).

FPGA users can adapt the configuration to suit their application. The output of the FPGA appears as a regular camera video stream to the mobile CPU.

Future work will include demonstrating the robotics applications of this system by mounting it on a MAV. Additionally, we plan to investigate an implementation of visual odometry estimation using the combined depth and image data directly on the FPGA.

REFERENCES

[1] 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*, pages 93 –101, july 2010.

[2] A. Barry. Flying between obstacles with an autonomous knife-edge maneuver. Master's thesis, Massachusetts Institute of Technology, 2012.

[3] J. Bouguet. Camera calibration toolbox for matlab. 2004.

[4] DC Brown. Decentering distortion of lenses. *Photogrammetric Engineering*, (32):444–462, 1966.

[5] F. Fraundorfer, L. Heng, D. Honegger, G. Lee, L. Meier, P. Tanskanen, and M. Pollefeys. Vision-Based Autonomous Mapping and Exploration Using a Quadrotor MAV. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, oct 2012.

[6] S. 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*, Lecture Notes in Computer Science.

[7] P. Greisen, S. Heinzle, M. Gross, and A. Burg. An FPGA-based processing pipeline for high-definition stereo video. In *EURASIP Journal on Image and Video Processing*. Springer, September 2011.

[8] H. Hirschmuller. Accurate and efficient stereo processing by semi-global matching and mutual information. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 2, pages 807–814 vol. 2, 2005.

[9] D. Honegger, P. Greisen, L. Meier, P. Tanskanen, and M. Pollefeys. Real-time velocity estimation based on optical flow and disparity matching. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, oct 2012.

[10] S. Jin, J. Cho, X. Pham, K. Lee, S. Park, M. Kim, and J. Jeon. FPGA design and implementation of a real-time stereo vision system. *Circuits and Systems for Video Technology, IEEE Transactions on*, 20(1):15 – 26, jan. 2010.

[11] C. Rabe, T. Mueller, A. Wedel, and U. Franke. Dense, Robust, and Accurate Motion Field Estimation from Stereo Image Sequences in Real-Time. In Kostas Daniilidis, Petros Maragos, and Nikos Paragios, editors, *Proceedings of the 11th European Conference on Computer Vision*, volume 6314 of *Lecture Notes in Computer Science*, pages 582–595. Springer, September 2010.

[12] K. Schmid and H. Hirschmuller. Stereo vision and imu based real-time ego-motion and depth image computation on a handheld device. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 4671–4678. IEEE, 2013.

[13] D. Watman and H. Murayama. Design of a miniature, multi-directional optical flow sensor for micro aerial vehicles. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 2986 –2991, may 2011.

[14] R. Zabih and J. Woodfill. Non-parametric local transforms for computing visual correspondence. pages 151–158. Springer-Verlag, 1994.