Efficient ASIC Implementation of a Real-Time Depth Mapping Stereo Vision System

Michael Kuhn[1] (Student), Stephan Moser[1] (Student), Oliver Isler[1] (Student),
Frank K. Gürkaynak[2], Andreas Burg[2], Norbert Felber[2], Hubert Kaeslin[3] and Wolfgang Fichtner[2]

[1] Student at the Department of Information Technology and Electrical Engineering, ETH Zurich
[2] Integrated Systems Laboratory, ETH Zurich
[3] Microelectronics Design Center, ETH Zurich

***Abstract*— This paper presents a fast and area-efficient implementation of a real-time stereo vision algorithm for spatial depth mapping. The design combines two well-known area-based approaches to stereo matching and includes an occlusion detection method. Hardware efficiency is achieved by storing only partial images on-chip, avoiding full-sized frame buffers. A low-latency dataflow-oriented structure makes it possible to process 256×192 pixel input streams with a rate in excess of 50 frames per second, amounting to more than 54 million pixel × disparity measurements per second (PDS) (for a 25-pixel disparity range), or roughly 18 GOPS. The design has been integrated in a 0.25 μm standard CMOS technology and occupies an area of less than 3 mm^2.**

I. Introduction

Depth mapping by passive stereo vision is a method to extract spatial depth information for a scene from a parallel pair of horizontally displaced stereo images. From the relative perspective shift of objects or image features a distance value is calculated. The advantage over prevalent distance measurement devices like ultrasonic or laser equipment is that a complete scene is captured at once, yielding a contact-free acquisition of the spatial impression. Possible applications include collision detection for intelligent transportation systems [3]–[5] and autonomous vehicles [6] as well as industrial automated production [7]. Current stereo vision algorithms are demanding with respect to calculation time and require high data throughput. FPGA-based approaches have been presented by [8], [9], a multi-component solution including DSPs can be found in [10]. These implementations are made up of several processing devices, yielding costly and power-intensive hardware. A low power solution has been presented in [11]: The ”Small Vision System” operates at 8 frames per second (fps) on 160×120 images while consuming 600 mW of power.

In this paper a hardware-efficient architecture of a stereo vision module for fast dynamic applications is presented and a complexity analysis is provided. The design simultaneously applies two stereo matching methods and combines them. As the algorithm works on partial images there is no need to store entire frames. The output is made up of a disparity map for visualization or further digital processing. A prototype for 256×192 grayscale images capable of operation at a frame rate in excess of 50 fps has been designed, fabricated and measured.

II. Algorithm

A. Current Approaches

Algorithms for stereo vision can generally be classified into feature-based and area-based approaches. Feature-based stereo matching methods first extract strong image features by e.g. edge detection before finding the corresponding feature in the other frame. Sparse depth maps with very few erroneous matches result from them. As large parts of the images must be available on-chip, this type of algorithm has not further been considered.

In contrast, area-based algorithms lead to dense depth maps that include more uncertainties. Correlation is determined by pattern matching, thereby neglecting the actual image content. This class of methods is specially suited for hardware integration since it allows for a homogenous, content-independant dataflow. However, it is more vulnerable to some of the inherent problems mentioned later.

A simple but efficient area-based technique is *block searching*. A block of pixels of the right stereo image is horizontally scanned for in the left image, starting at the same image coordinates (*displacement 0*), continuing over the entire search range to the *maximum displacement*. The best matching block displacement is determined by a correlation function and is considered as the local *disparity value*.

Commonly used correlation functions include SAD (sum of absolute differences), SSD (sum of squared differences), non-parametric transforms (Census, Rank transform) [1] and NCC (normalized cross correlation). An evaluation of these functions within our setup showed that the SSD approach produced higher quality results than the SAD method. The NCC function is not very efficient with respect to hardware [2]. The same holds for the Rank transform, the Census transform on the other hand is attractive for integration as it only requires addition operations.

B. Inherent Problems of Stereo Vision

As a scene is viewed from two different viewpoints there are regions that are visible to only one camera. This is due to foreground objects hiding objects in the background. This effect ist called *occlusion* and leads to uncorrelated information on the image pair.

Reflective surfaces and transparent objects falsify the appearent spatial geometry of a scene and are difficult to detect. Horizontally periodic structures can lead to incorrect matchings. Furthermore, low-textured areas (e.g. a white wall) yield weak correlations in stereo image pairs and must possibly be neglected to avoid incorrect results.

C. Algorithm Description

The proposed design implements a block matching scheme that uses a combination of two correlation functions. The SSD as well as the Census approach can efficiently be integrated and their characteristics complement each other to a certain degree (see Table I).

TABLE I
COMPARISON BETWEEN SSD AND CENSUS TRANSFORMATIONS

Criterion	SSD	Census
bias-independent	no	yes
homogenous areas	weak	strong
feature-rich areas	strong	weak
hardware complexity	moderate	low

The *SSD* function calculates a match quality value based on a least-square comparison of corresponding pixel intensities. It is strong on feature-rich image regions whereas homogenous areas lead to noisy results.

The *Census* function first performs a non-parametric transform [1] on pixel blocks (see Fig. 1): The basic operation works on square pixel blocks of odd size W_c. The center pixel is characterized by the surrounding pixels by an intensity comparison. If the neighbouring pixel is brighter, a “1” is set, otherwise a “0”. These $(W_c^2 - 1)$ bits form a bias-independent signature. In the example of Fig. 1, eight strings of eight bits each make up a signature string for one 10×3 pixel block. Two blocks are compared by the Hamming distance (the number of unequal bits) of their signature strings. The Census function performs well on lightly structured image regions but smears edges to a certain degree. The transformation has been shown in [8] to provide good results up to a block size of 11×11 pixels.

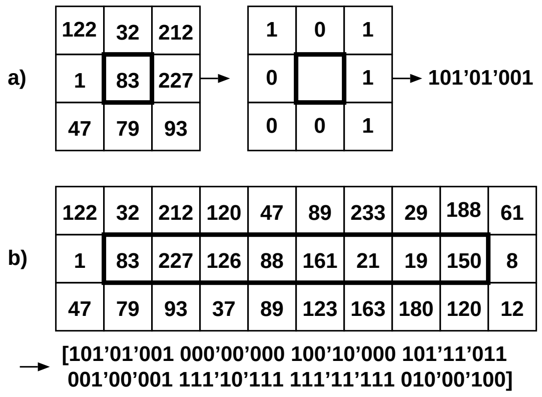

Fig. 1. Census transform example; a) transform of a single 3×3 pixel block. b) transform applied to a 10×3 pixel block with complete signature string for Hamming distance comparison.

Occlusion effects are effectively circumvented by an *LR-RL* consistency check [12], in which block matching is performed twice, from right to left (RL) and from left to right (LR). Occluded areas yield uncorrelated results. So, a superposition of RL and LR results allows to drop inconsistent areas of the scene. Furthermore, most erroneous matches from homogenous image areas are eliminated as well. As both correlation functions are applied RL and LR, a selection function picks results based on a priority scheme. Despite the occlusion detection mechanism there is always a certain possibility that unwanted results slip through, so the output image is post-filtered using a median function.

III. ARCHITECTURE

A. Dataflow

Six functional blocks make up the design as depicted in Fig. 2. The *input buffer* stores several image lines and maintains two shift register banks on which block searching is performed. It is implemented using a RAM and organized as an extended ringbuffer that provides the succeeding modules with data. Two purely combinational modules calculate the SSD and Census correlations (Fig. 3, C and D) and pass their results on to the *displacement module*. There, data is collected and perspectively mapped to the virtual viewpoint of the output map. Four intermediate disparity maps are thus created; an LR and an RL map for each correspondence function. These maps are merged into one single map by the *merge module* (Fig. 3, E). Several parameters configure the merge function according to the application needs. The *output filter* eliminates remaining erroneous pixels and smoothes the image by means of a median filter (Fig. 3, F).

TABLE II
COMPLEXITY ESTIMATIONS OF FOUR SAMPLE CONFIGURATIONS, COMPARED TO ACTUAL IMPLEMENTATION.

Criterion	Configuration				
	Implemented Architecture	(1) $\frac{1}{4}$PAL small	(2) $\frac{1}{4}$PAL large	(3) VGA	(4) VGA pipelined
Image Width	256	384	384	640	640
Image Height	192	288	288	480	480
Image Mode	8-bit grayscale				
Displacement	24	24	36	60	60
Block Width	10	10	15	25	25
Block Height	3	3	3	5	5
Area (mm^2)	3	3.1	3.84	7.8	25
Ram (KBytes)	1.34	2.02	2.02	5.9	5.9
Clock (MHz)	75	100	90	70	120
Frames/s	54.5	32.3	20.2	3.6	25

B. Complexity considerations

Some applications might require different image dimensions; e.g. a collision detection system with an increased viewing angle would allow faster response to approaching obstacles. Further, using higher resolution cameras permits to extract more detailed results with increased depth resolution.

A complexity analysis for four sample configurations with commonly used image dimensions is listed in Table II and

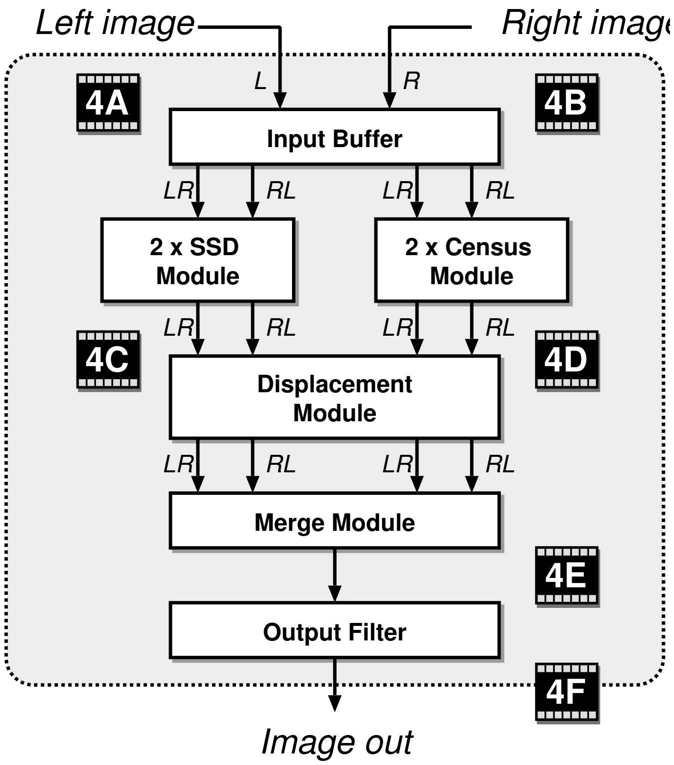

Fig. 2. Functional block diagram

compared to the actually implemented configuration. The two "$\frac{1}{4}$ PAL" configurations demonstrate the influence of the application-dependent parameters *block size* and *maximum disparity*. Linear scaling of the architecture leads to a low frame rate in the case of VGA resolution (see example #3, VGA). A reasonable frame rate can be achieved by replicating the correspondence function units and by pipelining the circuit (#4, VGA). The comparison indicates that the proposed architecture is suitable for a variety of medium- and low-resolution video formats.

IV. VLSI Implementation

The VLSI implementation accepts two 256×192 8 bit grayscale image streams that are fed in using a generic protocol. The correspondence functions work on 10×3 pixel blocks over a displacement range of 25 pixels. For two standard 35 mm cameras with a baseline distance of 8 cm this yields a depth range from 82 cm to 20 m in 24 steps, plus one value for infinity. A minimum of 28 clock periods make up one pixel cycle, resulting in an overall latency of 15,204 clock periods or 548 pixels.

The design has been implemented and fabricated using a 0.25 μm 5 Metal process and occupies a total core area of less than 3 mm^2. The fabricated samples were verified to be functional at a clock frequency of 75 MHz and thus achieve a frame rate of more than 50 fps. The necessary on-chip RAM amounts to a mere 1.35 KBytes.

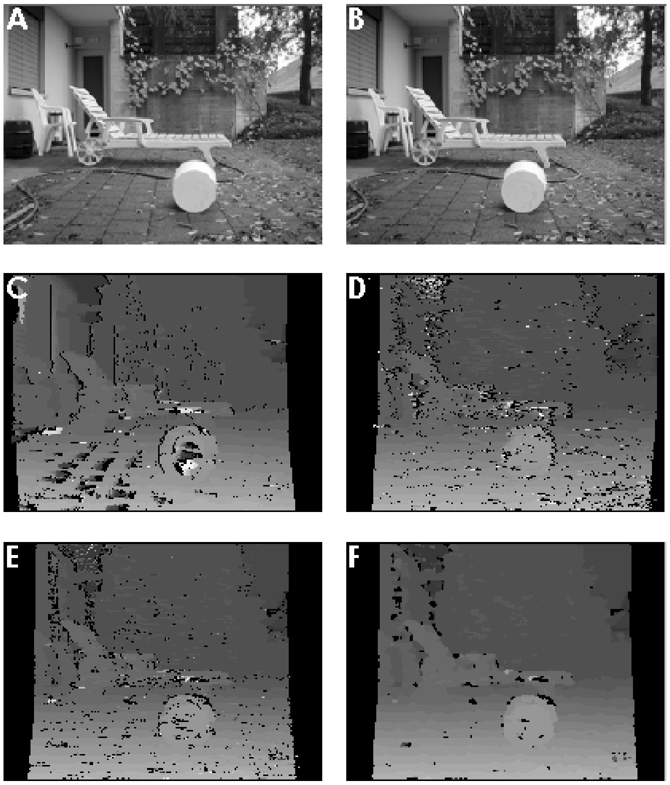

Fig. 3. *A typical output disparity map and how it evolves as it passes through the design.* A,B: Two 8 bit grayscale input images of size 256×192. C: An SSD result image with block searching performed from left to right (LR). D: A Census result image (RL case). E: Priority-based combination after the merge module which still contains scattered erroneous pixels. F: Median-filtered output image.

V. Conclusion

A hardware-efficient stereo vision ASIC has been presented. The approach of combining three well-known methods (SSD, census transformation and occlusion detection) in stereo vision has been proven to deliver high-quality results with low hardware complexity. The presented architecture is scalabale to accomodate various resolutions and frame rates. Furthermore, the continuous dataflow does not rely on large on-chip RAM blocks to store entire frames, resulting in an area-efficient design. The implementation works at more than 50 fps which enables real-time applications in very dynamic environments.

Acknowledgment

Special thanks go to Dr. Tomas Svoboda from the Computer Vision Laboratory of ETH Zurich for the review of the algorithm.

References

[1] R. Zabih and J. Woodfill, *Non-parametric Local Transforms for Computing Visual Correspondence*, Third European Conference on Computer Vision, (Stockholm, Sweden) 1994.

[2] R. B. Porter and N. W. Bergmann, *A Generic Implementation Framework for FPGA Based Stereo Matching*, IEEE TENCON, 1997.

[3] I. Masaki, *Machine-Vision Systems for Intelligent Transportation Systems*, IEEE Intelligent Systems, Massachusetts Institute of Technology, 1997.

[4] M. Bertozzi and A. Broggi, *GOLD: a parallel real-time stereo vision system for generic obstacle and lane detection*, IEEE Transactions on Image Processing, Volume: 7 Issue: 1 , Jan 1998 Pages: 62-81.

[5] Zheng-Tie Sun, Li-Chen Fu and Shin-Shinh Huang, *On-road computer vision based obstacle detection*, IEEE/RSJ International Conference on Intelligent Robots and Systems 2002.

[6] S. B. Goldberg, M. W. Maimone and L. Matthies, *Stereo vision and rover navigation software for planetary exploration*, IEEE Aerospace Conference Proceedings, Volume: 5, 2002.

[7] Y. Kimura, T. Naito, M. Nakano, H. Moribe, T. Kuno, *Stereo vision system for car assembly*, IEEE International Conference on Robotics and Automation, Proceedings, Volume: 2, 1995.

[8] J. Woodfill and B. Von Herzen, *Real-time stereo vision on the PARTS reconfigurable computer*, The 5th Annual IEEE Symposium on FPGAs for Custom Computing Machines, 1997.

[9] P. Corke and P. Dunn, *Frame-rate stereopsis using non-parametric transforms and programmable logic*, IEEE International Conference on Robotics and Automation, 1999.

[10] T. Kanade, A. Yoshida, K. Oda, H. Kano, M. Tanaka, *A stereo machine for video-rate dense depth mapping and its new applications*, IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1996.

[11] K. Konolige, *Small Vision Systems: Hardware and Implementation*, Eighth International Symposium on Robotics, 1997.

[12] G. Egnal, R. P. Wildes, *Detecting binocular half-occlusions: empirical comparisons of five approaches*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume: 24 Issue: 8 , Aug 2002