Trinocular Disparity Processor using a Hierarchic Classification Structure

Andy Motten, Luc Claesen
Expertise Centre for Digital Media
Hasselt University – tUL – IBBT
Wetenschapspark 2, 3590 Diepenbeek, Belgium
{firstname.lastname}@uhasselt.be

Yun Pan
Institute of VLSI Design
Zhejiang University
Hangzhou, China
panyun@vlsi.zju.edu.cn

Abstract—**This paper presents a real-time trinocular disparity processor. The core module performs a pairwise segmented window matching for both the center-right and center-left image pair as their scaled down image pairs. The resulting cost functions are combined which results into nine different curves. A hierarchical classifier is presented which selects the most promising disparity value using information provided by the calculated cost curves and the pixels spatial neighborhood using a two level classification architecture. The disparity processor has been evaluated with an indoor dataset and with a real-time implementation using an FPGA and three cameras. Special care has been taken to reduce the memory footprint so that the processor doesn't need external memory.**

Keywords-component; trinocular camera; real-time matching; confidence metric; computer vision; system-on-chip; FPGA;

I. Introduction

Trinocular vision makes use of three cameras to calculate a disparity search image (DSI). The DSI is generated by pairwise matching the images from the different cameras which is based on a local window based stereo matching architecture.

An improvement of occlusion handling in trinocular vision compared to stereo vision is achieved by Mozerov [1]. The main idea is based on the assumption that any occluded region in a matched stereo pair (center-left images) in general is not occluded in the opposite matched pair (center-right images). They use a global optimization technique to derive the composite DSI. Bidirectional matching using trinocular stereo is used by Ueshiba [2] to detect half-occlusions and to discard false matches. It uses a cumulative cost function derived from a summation of both cost curves.

This paper will likewise calculate several DSI's. However, instead of combining them, a hierarchical classifier is used to select the most likely disparity for each pixel in the final DSI. The matching algorithm is based on the adaptive-weight algorithm proposed by Yoon [3], which adjusts the support weight of each pixel in a fixed sized window. The support weights are depending on the color and spatial difference between each pixel in the window and the center pixel. Dissimilarities are computed based on the support weights and the plain similarity scores. Their experiment indicates that a local based stereo matching algorithm can produce depth maps similar to global algorithms. A hardware implementation using the same ideas is published by Motten [4].

For each matching result, a confidence metric is calculated. A good comparison between different confidence metrics can be found in the evaluation paper of Hu [5]. Confidence metrics suitable for hardware implementation can be found in [6]. They conclude that neighboring pixels contain valuable information to distinguish good matches from bad ones.

Recently many stereo implementations have been proposed for hardware implementations. A real-time FPGA-based stereo vision system is presented by Jin [7] that makes use of the census transform. Their system includes all the pre- and post-processing functions such as; rectification, LR-check and uniqueness test in a single FPGA. Another extensive implementation can be found in [8]. They divide the problem into two parts: first a rough depth map is constructed using a segmentation based sum of absolute differences (SAD) window comparison, second a disparity refinement module identifies false matches and replaces them with new estimates. Hardware implementations of a trinocular disparity processor are limited. A sum of SAD's with a fixed windows implementation can be found in [9].

This paper combines the strengths of an advanced stereo vision system with a two-scale adaptive window SAD incorporated in a trinocular setup.

II. System Overview

A. General Architecture

The trinocular disparity processor takes three images that have been taken by three cameras that have a vertical alignment and a horizontal offset. Objects will appear on the same horizontal line (The epipolar line) on all images. The horizontal distance between the same objects on the center image and the left (or right) image is called the disparity. If calibrated correctly, the disparity of an object between the center-left and the center-right image pair should be the same. This characteristic can be used to discard false matches using bidirectional matching [2] or to improve the quality of the DSI especially in occluded regions [1].

The architecture consists of three main blocks. The first block captures the pixel streams, generates the scaled images and places them in multiple on-chip parallel memories. The

This research has been sponsored in part by the BOF (Bijzonder Onderzoeks Fonds uHasselt), Flanders FWO (Fonds voor Wetenschappelijk Onderzoek) and Chinese MOST (Ministry of Science and Technology) project number G.A.063.10

second block performs a pair wise window matching of the different streams (see Fig. 1) using a binary adaptable SAD cost aggregation [8]. The third block calculates its confidence for each data stream and selects the final disparity value.

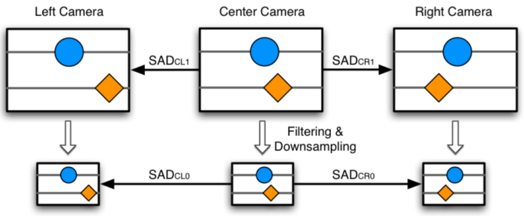

Figure 1. Different window matching.

For every window that needs to be matched, a SAD calculation is performed. The larger the disparity search width, the more SAD calculations are needed. The result is an array that contains a SAD score for each disparity value (usually starting from 0), this array is also known as the cost curve. In this architecture nine different cost curves are calculated for each pixel in the DSI (1).

$$\begin{cases} SAD_{CL1}, SAD_{CR1}, SAD_{CL0}, SAD_{CR0} \\ SAD_{CLR1} = SAD_{CL1} + SAD_{CR1} \\ SAD_{CLR0} = SAD_{CL0} + SAD_{CR0} \\ SAD_{CL01} = SAD_{CL1} + SAD_{CL0} \\ SAD_{CR01} = SAD_{CR1} + SAD_{CR0} \\ SAD_{CLR01} = SAD_{CLR1} + SAD_{CLR0} \end{cases} \quad (1)$$

In this paper, C_1 stands for the lowest SAD score (the minima of the Cost Curve). C_2 stands for the second lowest SAD score, and so on. Their corresponding depths are indicated by D_1 and D_2. Most matching algorithms calculate the disparity from the cost curve using a “Winner Takes All” (WTA) approach. Doing so, the minima of the cost curve (C_1) will become the calculated disparity D_1.

III. Hierarchical Classification

In the previous section it is explained that nine disparity values are generated for each pixel (1). In order to select one of them for generating the DSI, a two level hierarchical classifier is constructed (Fig. 2). In the first level of the hierarchy, the disparity values are investigated independently of each other. For each disparity value a binary confidence classifier is constructed using the methods presented in [6]. These confidences are passed on to the second level classifier which selects the disparity to use, or indicates that no disparity has been found.

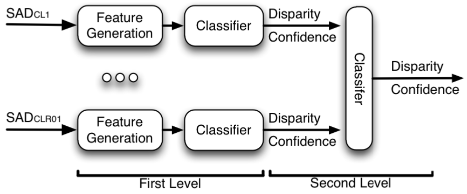

Figure 2. Hiearchical classification.

For each level of the hierarchy, a different set of features is needed for classification. The first level of classifiers uses information obtained from the pixel neighborhood and from its corresponding cost curve. According to [6], the most important features on this level are the *Segmentation Size (SEG)* and the *Sum of Neighboring Depths Differences Binary Window (SNDDBW)*. At this level, a decision tree (DT) is chosen as classifier for each disparity stream individually.

The second level classifier uses the generated binary confidence values together with the agreement between the different disparity values. This new feature is called the *Sum of Streaming Depths Differences (SSDD)*, it calculates the depth difference between the different disparity streams taking the confidence value into account (2).

$$SSDD = \sum_{i=1}^{streams} confidence_i * |D_1 - D_1(i)| \quad (2)$$

The goal of this classification level is to select the most promising disparity value (3). The input of this classification level is the SSDD for each disparity stream. The output of this classification level is a disparity selection. A confidence value is afterwards generated using the same method as for each individually stream.

$$Disparity = \min_{\mathrm{i:1 \rightarrow streams}} \mathrm{SSDD\ (i)} \quad (3)$$

An exhaustive search is performed in order to know which combination of streams provides the highest disparity improvement. This process will be elaborated in future writings. From Fig. 3 we can see that the addition of extra streams improves the quality of the DSI. The trinocular setup improves the DSI most noticeably at occluded regions. The scaled images improves the disparity map at parts with little texture.

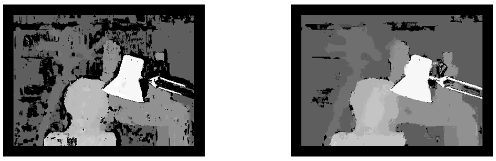

Figure 3. Depth map quality of the Tsukuba dataset [10]. Comparison of DSI generated from CL0 data stream (Left) and DSI generated from the combination of CL0, CL1, CR0 and CR1 data streams (Right).

IV. System Design

The hardware architecture consists of three main modules. First a filter and subsampling module has been added to the pre-processing module [8] so that a scaled image is generated with one-fourth the size of the original image. Second the window matching module is modified from [8] to allow for multiple data stream matching. Third a hierarchic classification module is constructed to select the most promising disparity from the different disparity results.

A. Pre-Processing Module

The pre-processing module consists of four different entities for each pixel stream: first a demosaicing algorithm is used to reconstruct the color image, next a rectification module is used to remove lens distortion and perform trinocular calibration and lastly the image is filtered and downsampled to generate a scaled image.

Pixels generated by the camera are formatted in a Bayer pattern consisting of the four colors: Red (R), Green1 (G1), Blue (B) and Green2 (G2), representing the three color filters.

The demosaicing algorithm is used to estimate the color components for each pixel. Using linear interpolation, the missing RGB colors are reconstructed from the adjacent pixels [8]. The proposed architecture makes use of the YCrCb color space. The Luminance (Y) values are used to compare the two input streams. While the chrominance values (Cr, Cb) are used to construct the binary mask window. Hence, the reconstructed RGB color space needs to be transformed into the YCrCb color space (8).

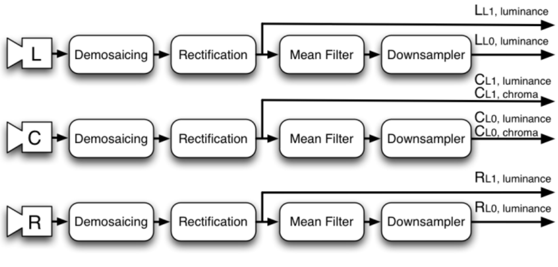

Figure 4. Pre-processing module

Two different kinds of distortions are present in a trinocular camera setup. The first ones are the lens distortions, the second one is the misalignment of the three cameras. Since the search space is only located on the epipolar line, both distortions should be resolved before matching can be performed. The intrinsic and extrinsic parameters of the cameras individually and the transformation matrix of the trinocular setup are determined offline using images of checkerboard patterns [11]. These parameters are hence used to construct the x and y mapping coordinates for each pixel in the image. The rectification module uses those coordinates to rectify the images in real time [6]. The rectified pixel stream is passed through a 3x3 mean filter and downsampled by a factor of two. The original pixel stream is annotated with level 1 (L1) while the scaled pixel stream is annotated with level 0 (L0).

B. Window Matching Module

The pixel streams originated from the right and left camera are compared with the center camera using a segmentation based SAD calculation (Fig. 5). During every clock cycle a window of the center camera is compared with four windows of the left or right camera. Since four successive pixels are stored in one memory location, one memory read accesses four pixels, hence four comparison modules are running in parallel.

On every clock cycle, the stream selection unit (SSU) determines where each data stream is written to and which windows are compared.

The frequency of the window matching module directly controls the possible disparity search width of the trinocular matching architecture and can be adapted to the available resources. The higher the frequency difference between the pixel stream and the window matching module, the more comparisons can be executed. In the example on Fig. 6 the window matching module is clocked twenty-four times higher than the pixel streams.

On each clock cycle (CC), the comparison module compares the reference window with four consecutive windows (Fig. 5). The lowest SAD score and its corresponding index are saved in a register, so that on the next clock cycle this lowest SAD score can be compared against the SAD scores of the next four windows. When the end of a search window is reached, the index indicates the disparity result and a new search window is initiated. In our example, in the first eight clock cycles, the center image is compared with the right image. In the next eight CC's the center image is compared with the left image.

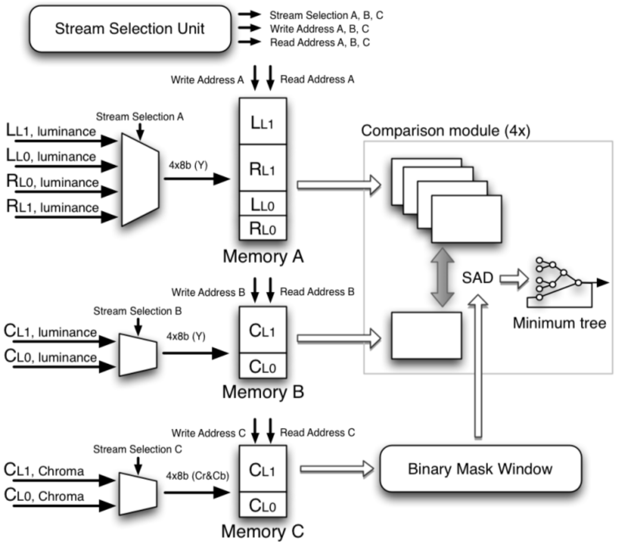

Figure 5. Window matching module

In the following four CC's the scaled center image is compared with the scaled right image and in the last four CC's the scaled center image is compared with the scaled left image. This leads to a combined disparity search width of thirty-two.

This architecture makes it possible to easily change the disparity search width and comparison data streams for each pixel in the DSI. By adapting the SSU it is possible to switch between a trinocular disparity search width of thirty-two to a stereo disparity search width of one hundred twenty-four without changing the architecture.

C. Hierarchical Classification Module

The hierarchical classification module consists of the generation of the features used during the classification phase and the two classification steps. The first level classifier calculates the confidence of each stream in the selection (4, 5). For each stream, different thresholds are selected. However, the main structure of the classifier remains the same. The second level classifier selects the most promising disparity stream for the final DSI (6, 7, 8).

$$\text{streams} = \{\text{CL0,CL1,CR0,CR1}\} \tag{4}$$

$$\begin{cases} Conf_x = \ if\big(SNDDBW_{21,10} < a\big)then \\ \ [if(TEX > b)then\ 1\ else\ 0\ ]\ else\ 0 \\ \qquad\qquad x \in streams \end{cases} \tag{5}$$

$$\text{SSDD}_\text{x} = \textstyle\sum_{\text{y} \in \text{streams}} \text{Conf}_\text{y} {*} \big|\text{D}_\text{x}\text{-D}_\text{y}\big|\ ,\ \text{x} \in \text{streams} \tag{6}$$

$$Disparity\_selection = \min_{\text{y} \in \text{streams}} SSDD_y \tag{7}$$

$$Disparity = \text{streams}\ (Disparity\_selection) \tag{8}$$

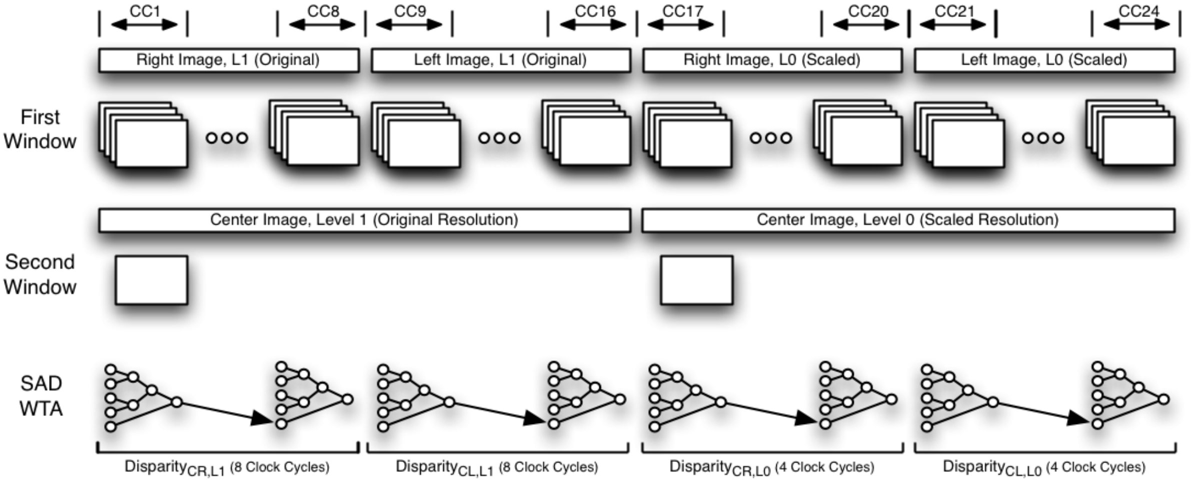
CC1
CC8
CC9
CC16
CC17
CC20
CC21
CC24
Right Image, L1 (Original)
Left Image, L1 (Original)
Right Image, L0 (Scaled)
Left Image, L0 (Scaled)
First Window
Center Image, Level 1 (Original Resolution)
Center Image, Level 0 (Scaled Resolution)
Second Window
SAD WTA
Disparity$_{CR,L1}$ (8 Clock Cycles)
Disparity$_{CL,L1}$ (8 Clock Cycles)
Disparity$_{CR,L0}$ (4 Clock Cycles)
Disparity$_{CL,L0}$ (4 Clock Cycles)

Figure 6. Window matching of different data streams.

V. Implementation

The architecture and methods presented in this paper have been implemented on an FPGA system, based on an Altera Cyclone IV with 114,480 logic elements and 432 memory blocks. The sources of the input streams are three cameras with a resolution of 640x480 and a pixel clock of 16 MHz resulting in a refresh rate of 52 Hz. The current implementation consists of the proposed design using a 7x7 binary adaptive window SAD with a window matching clock of 96 MHz.

The architecture has been constructed to reduce memory usage. Hence there is no need for external memories. The reduction of external memory usage has the additional advantage that the latency between input frame and output frame becomes minimal. This makes this system suitable to be incorporated in real-time control loops. In addition to the evaluation presented in section II, the system has also been tested in real life environments.

VI. Conclusions

A trinocular disparity processor has been proposed. We investigated nine cost curves resulting from pairwise comparison of three cameras. Each data stream has been investigated independently from one another and ultimately a hierarchic classification algorithm chooses the most promising disparity value.

For each of the nine cost curves, a classification algorithm is trained in order to provide a confidence indication for their disparity values. These confidences are passed on to the second level classifier which selects the disparity to use, or indicates that no disparity has been found.

The selection of classification algorithms has been used as guideline for the implementation in an FPGA. From the results we can conclude that the quality of the disparity space image increases by using more cost curves from a trinocular camera.

Due to the adaptability of the window matching module and the hierarchic classification structure, the system can easily be expanded with more data streams to further improve the disparity space image.

References

[1] M. Mozerov, J. Gonzalez, X. Roca, and J.J. Villanueva, "Trinocular stereo matching with composite disparity space image," in Proceedings IEEE ICIP-2009, 16th IEEE International Conference on Image Processing, 2009, pp.2089-2092.

[2] T. Ueshiba, "An efficient implementation technique of bidirectional matching for real-time trinocular stereo vision," in Proceedings IEEE ICPR-2006, 18th International Conference on Pattern Recognition, 2006, pp.1076-1079.

[3] K.J. Yoon, and I.S. Kweon, “Adaptive support-weight approach for correspondence search,” IEEE Trans. PAMI, vol. 28 (4), 2006, pp. 650-656.

[4] A. Motten, and L. Claesen, “A Binary Adaptable Window SoC Architecture for a Stereo Based Depth Field Processor,” in Proceedings IEEE VLSI-SOC-2010, 18th IEEE/IFIP International Conference on VLSI and System-on-Chip, Madrid, 27-29 September 2010, pp. 25 - 30.

[5] X. Hu, and P. Mordohai, “Evaluation of stereo confidence indoors and outdoors,” in Proceedings IEEE CVPR-2010, 23th IEEE conference on Computer Vision and Pattern Recognition, 2010, pp. 1466-1473.

[6] A. Motten, L. Claesen, and Y. Pan, “Binary confidence evaluation for a stereo vision based depth field processor SoC,” in Proceedings IEEE ACPR-2011, 1st Asian Conference on Pattern Recognition, Beijing, 28-30 November 2011, pp. 456 - 460.

[7] S. Jin, J. Cho, X. D. Pham, K.M. Lee, S. –K. Park, and J. W. Jeon, “FPGA Design and Implementation of a Real-Time Stereo Vision System,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 20 (1), 2010, pp. 15-26.

[8] A. Motten, and L. Claesen, "Low-cost real-time stereo vision hardware with binary confidence metric and disparity refinement," in Proceedings IEEE ICMT-2011, International Conference on Multimedia Technology, 2011, pp.3559-3562.

[9] Li Mingxiang, Jia Yunde, "Stereo vision system on programmable chip (SVSoC) for small robot navigation," in Proceedings IEEE/RSJ IROS-2006, International Conference on Intelligent Robots and Systems, 2006, pp.1359-1365.

[10] D. Scharstein, and R. Szeliski, “A taxonomy and evaluation of dense two-frame stereo correspondence algorithms,” International Journal of Computer Vision, vol. 47 (1), 2002, pp. 7-42.

[11] Z. Zhang, “Flexible Camera Calibration by Viewing a Plane from Unknown Orientations,” in Proceedings IEEE ICCV-1999, 7th IEEE International Conference on Computer Vision, Kerkyra, 20-25 September 1999, pp. 666 - 673.