Hardware-Oriented
Multiple Camera Disparity Map
Estimation

by
Ipek Baz

Master Thesis Project
submitted to
School of Engineering
École Polytechnique Fédérale de Lausanne (EPFL)
in fulfillment of the thesis requirements
for the degree of

Master of Science
In
Electrical and Electronics Engineering

supervised by
Prof. Yusuf Leblebici
Dr. MER Alexandre Schmid
Abdulkadir Akin
Microelectronic Systems Laboratory - STI/LSM, EPFL

Friday 21st June, 2013
Acknowledgements

I would like to thank Prof. Yusuf Leblebici who gave me the opportunity to carry out my master thesis at LSM and Dr. Alexandre Schmid who gave us helpful comments in our papers. I also thank Abdulkadir Akin who advise and gave me helpful comments in this thesis. He also provided me necessary background about the principles of hardware implementations. In addition to his supervision, Abdulkadir encouraged me to develop and implement my own ideas during my study.

I would also like to thank individually Kerem Seyid, Tugba Demirci, Omer Cogal, and Gozen Koklu who are PhD students and scientists at LSM, not only for their advices and contributions during my thesis study but also for providing me such a friendly working environment and the many great times spent together.

Finally, very special thanks go to my family, my mother Filiz, my father Ibrahim and my sister Irem for their love and supports not only during my studies but also during all my life.
Abstract

Depth information is used in a variety of 3D based signal processing applications such as autonomous navigation, robot and driving systems, object detection and tracking, 3D television, and disparity-based rendering. In these applications, high accuracy and speed performances are required for depth map estimation. Depth maps can be generated by using disparity estimation methods, which are obtained from stereo matching between the stereo images.

The computational complexity of disparity estimation algorithms and the need of large size memory make the real-time processing of disparity estimation challenging, especially for high resolution images. This thesis proposes binocular (AWDE) and trinocular (T-AWDE) hardware-oriented adaptive window size disparity estimation algorithms, which target high resolution video with high quality disparity results. Furthermore, in depth map estimation based applications, disparity estimation should be performed in real-time. To reach real-time, the algorithms also should be suitable for hardware implementation. The proposed binocular and trinocular algorithms can be implemented in hardware very efficiently.

We first propose a novel binocular disparity estimation algorithm that is a hybrid solution involving the Binary Window Sum of Absolute Differences and the Census cost computation methods to vote and select the best suitable disparity candidates. It utilizes a pixel intensity based refinement step to remove faulty disparity computations. The AWDE algorithm dynamically adapts the window size considering the local texture of the image to increase the disparity estimation quality. We then propose a new trinocular adaptive window size disparity estimation algorithm. Our trinocular algorithm is the extended version of the binocular algorithm. A novel disparity map fusion method is developed. The proposed trinocular algorithm carefully handles the problem in the binocular disparity estimation results by adding a third camera into the system.

Finally, the algorithms are evaluated on the different stereo datasets. The results demonstrate that the proposed AWDE and T-AWDE algorithms are suitable for real-time hardware implementation and their reconfigurable hardware can be used in consumer electronics products where high-quality real time disparity estimation is needed for high resolution video.
# Contents

## Contents

List of Figures vii  
List of Tables ix  

## 1 Introduction

1.1 Motivation .................................. 1  
1.2 Thesis Outline .................................... 2

## 2 State of the Art

2.1 Binocular Stereo Matching ......................... 5  
2.1.1 Preprocessing .................................... 6  
2.1.2 Matching Cost Computation ......................... 14  
2.1.3 Cost Aggregation .................................. 16  
2.1.4 Disparity Computation and Optimization ............... 16  
2.1.5 Disparity Refinement ................................ 17  
2.1.6 Hardware Adaptable Stereo Matching Methods ........... 17  
2.2 Trinocular Stereo Matching ......................... 18  
2.2.1 Multi-Camera Calibration and Rectification .......... 18  
2.2.2 Trinocular Disparity Estimation ...................... 20

## 3 Hardware-Oriented Adaptive Window Size Disparity Estimation Algorithms

3.1 Binocular Hardware-Oriented Adaptive Window Size Disparity Estimation Algorithm .................................. 21  
3.1.1 Preprocessing .................................... 22  
3.1.2 Window Size Determination ......................... 23  
3.1.3 Matching Cost Calculation ......................... 24  
3.1.4 Disparity Selection .................................. 26  
3.1.5 Disparity Refinement ................................ 28  
3.2 Trinocular Hardware-Oriented Adaptive Window Size Disparity Estimation Algorithm .................................. 29  
3.2.1 Preprocessing .................................... 30  
3.2.2 Window Size Determination ......................... 31  
3.2.3 Matching Cost Calculation ......................... 31  
3.2.4 Disparity Selection .................................. 32  
3.2.5 Disparity Refinement ................................ 33  
3.2.6 Fusion of the Disparity Maps ......................... 33
3.2.7 Final Disparity Refinement . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 35

4 Experimental Results and Evaluations 37
   4.1 Implementation Results of the Binocular and Trinocular Stereo Rectification . . 37
      4.1.1 The Binocular Stereo Rectification . . . . . . . . . . . . . . . . . . . . . . . . 37
      4.1.2 The Trinocular Stereo Rectification . . . . . . . . . . . . . . . . . . . . . . . 43
   4.2 Implementation Results of the Binocular Stereo . . . . . . . . . . . . . . . . . . 44
   4.3 Implementation Results of the Trinocular Stereo . . . . . . . . . . . . . . . . . . 49
   4.4 Hardware Implementation Results of the Proposed AWDE Algorithm . . . . . . 53

5 Conclusion 55

Bibliography 57
List of Figures

2.1 Coordinate systems for pinhole camera model. 7
2.2 Pinhole camera model. 7
2.3 Image planes of undistorted rectified cameras. 10
2.4 Mathematical alignment of the two cameras into an unique image plane. 10
2.5 Inverse relationship between the depth and disparity. 12
2.6 The inverse relationship is found the by similarity of the triangles. 12
2.7 Inverse mapping with fractional precision coordinates. Corners indicate integer pixel coordinates. 13
2.8 Forward mapping with fractional precision coordinates. 13
2.9 The geometrical relationship between the cameras of the trinocular stereo system. 19
3.2 The selection of nearest source pixels from fractional inverse mapping and extraction of forward mapping with integer coordinates. 23
3.3 Matching cost calculation. 24
3.4 9 selected pixels in a block for BW-SAD calculation. 25
3.5 Census transform computation. 26
3.6 Disparity selection. 27
3.7 Disparity space image. 28
3.8 Examples for selecting 17 contributing pixels for 7x7, 13x13 and 25x25 window sizes during the disparity refinement process (yellow (1): 7×7, green (2): 13×13 and blue (3): 25×25). 29
3.9 Trinocular camera system. 29
3.10 Disparity search for the trinocular stereo system. 31
4.1 High resolution Middlebury benchmarks. 38
4.2 Low resolution Middlebury benchmarks. 39
4.3 The real images captured by the trinocular camera system. 39
4.4 Calibration images. 40
4.5 Original left and right images have distortions as observed near the lamp, bag, folder and cup; horizontal epipolar lines are demonstrated near the edge of these objects. 41
4.6 CLUT-R corrects distortions as observed near the lamp, bag, folder and cup. 41
4.7 Disparity estimation results of (a)Mini-census (b) AWDE for the rectified images of CLUT-R. 42
List of Figures

4.8 Original left, center and right images have distortions as observed near the tableau, monitor and bag; horizontal epipolar lines are demonstrated near the edge of these objects. ................................................................. 44
4.9 The trinocular rectification corrects distortions as observed near the tableau, monitor and bag. ................................................................. 44
4.10 Binocular disparity estimation results of (a) fix window 7×7, (b) fix window 13×13, (c) fix window 25×25, (d) AWDE ......................................................... 47
4.11 Visual disparity estimation results of the AWDE algorithm for HR benchmarks . . 48
4.12 Visual disparity estimation results of the AWDE algorithm for real images captured by the camera system ......................................................... 49
4.13 Visual disparity estimation results of the T-AWDE algorithm for HR benchmarks . 50
4.14 Visual disparity estimation results of the T-AWDE algorithm for real images captured by the camera system ......................................................... 51
4.15 Visual disparity estimation results of algorithmic steps of T-AWDE. ................. 52
<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>PSNR (dB) with the Rectified Images Produced By Matlab Calibration Toolbox</td>
<td>42</td>
</tr>
<tr>
<td>4.2</td>
<td>PSNR (dB) Comparison of the Rectified Images Produced by CLUT-R and Matlab</td>
<td>42</td>
</tr>
<tr>
<td></td>
<td>Calibration Toolbox Using Different Disparity Estimation Algorithms</td>
<td></td>
</tr>
<tr>
<td>4.3</td>
<td>Hardware Resource Comparison of the Rectification Hardware Implementations</td>
<td>43</td>
</tr>
<tr>
<td>4.4</td>
<td>Parameters of the AWDE</td>
<td>45</td>
</tr>
<tr>
<td>4.5</td>
<td>Disparity Estimation Performance Comparisons</td>
<td>46</td>
</tr>
<tr>
<td>4.6</td>
<td>Comparison of Disparity Estimation Results between the AWDE and T-AWDE</td>
<td>50</td>
</tr>
<tr>
<td>4.7</td>
<td>Hardware Performance Comparison</td>
<td>54</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

1.1 Motivation

Depth estimation is an algorithmic step in a variety of applications such as autonomous navigation, robot and driving systems, 3D geographic information systems, object detection and tracking, medical imaging, computer games and advanced graphic applications, 3D holography, 3D television, multi-view coding for stereoscopic video compression, and disparity-based rendering. These applications require high accuracy and speed performances for depth estimation.

Depth estimation can be performed by exploiting three main techniques: time-of-flight (TOF) camera, LIDAR sensor and stereo camera. A TOF camera easily measures the distance between the object and camera using a sensor, circumventing the need of intricate digital image processing hardware [1]. However, it does not provide efficient results when the distance between the object and camera is high. Moreover, the resolution of TOF cameras is usually very low (200×200) [1] when it is compared to the Full HD display standard (1920x1080). Furthermore, their commercial price is much higher than the CMOS and CCD cameras. LIDAR sensors compute the depth by using laser scanning mechanisms but they are also very expensive compared to CMOS and CCD cameras. Due to laser scanning hardware, LIDAR sensors are heavy and bulky devices. Therefore, they can be used mainly for static images. Consequently, in order to compute depth map, the majority of research focus on extracting the disparity information using two or more synchronized images taken from different viewpoints, using CMOS or CCD cameras [2].

Many disparity estimation algorithms have been developed with the goal to provide high-quality depth map results. Consequently, disparity estimation is one of the most active research areas in computer vision. Although a large number of algorithms for disparity estimation have been developed, relatively little work has been done for hardware implementations. The computational complexity of disparity estimation algorithms and the need of large size and bandwidth for the external and internal memory make the real-time processing of disparity estimation challenging, especially for High Resolution (HR) images.

Real-time disparity estimation for high resolution images offers some crucial advantages compared to low resolution disparity estimation. First, processing high resolution stereo images increases the disparity map resolution which improves the quality of the object definition. Second, disparity estimation for high resolution stereo images is able to define the disparity with sub-pixel efficiency compared to the disparity estimation for low resolution image. Therefore, the disparity estimation for high resolution provides more precise depth measurement than the disparity estimation for low resolution. Third, disparity values between 0-2 can be
1. Introduction

considered as background for low resolution images. In high resolution such disparities are defined within a larger disparity range; thus, the depth of far objects can be distinguished more precisely.

Despite the advantages of high resolution disparity estimation, the use of high resolution stereo images brings some challenges. Disparity estimation needs to be assigned pixel by pixel for high-quality disparity estimation. Pixel-wise operations cause a sharp increase in computational complexity when the disparity estimation targets high resolution stereo video. Moreover, disparity estimation for high resolution stereo images requires stereo matching checks with larger number of candidate pixels than the disparity estimation for low resolution images. The large amount of candidates increases the challenge to reach real-time performance for high resolution images. Furthermore, high-quality disparity estimation may require multiple reads of input images or intermediate results, which poses severe demands on off-chip and on-chip memory size and bandwidth especially for high resolution images.

This thesis proposes hardware-oriented adaptive window size binocular disparity estimation (AWDE) and trinocular disparity estimation (T-AWDE) algorithms that target high resolution video with high quality disparity results for binocular and collinear trinocular multiple camera systems. The proposed algorithms are a hybrid solution involving the Binary Window Sum of Absolute Differences (BW-SAD) and the Census cost computation methods to vote and select the best suitable disparity candidates. The algorithms combine the strengths of the Census transform and Binary Window SAD. Moreover, the use of BW-SAD provides better disparity estimation results than the SAD for the depth discontinuities [3].

The proposed algorithms dynamically adapt the window size considering the local texture of the image to increase the disparity estimation quality. The algorithms provide dynamic and static configurability to have satisfactory disparity estimation quality for the images with different contents. It provides dynamic reconfigurability to switch between window sizes of $7 \times 7$, $13 \times 13$ and $25 \times 25$ pixels in run-time to adapt to the texture of the image for its hardware implementation [4]. The benefit of using different window sizes for different texture features on the image is observed from the disparity estimation results in [3]. The selection of a large window size improves the algorithm performance in textureless regions while requiring higher computational load. However, the usage of small window sizes provides better disparity results where the image has a texture. Furthermore, the proposed algorithms utilize a pixel intensity based refinement step to remove faulty disparity computations. The computational complexity of the proposed refinement steps are not high, thus the refinement steps can be implemented in parallel.

The trinocular disparity estimation algorithm (T-AWDE) is the extended version of the binocular algorithm. The T-AWDE propose a novel approach that combines the strengths of the two disparity maps that are calculated for two stereo pairs of the trinocular camera system. The T-AWDE algorithm results clearly show the effect of adding the third camera in to the stereo system. Briefly, this thesis presents hardware adaptable, robust and accurate disparity estimation algorithms for binocular and multiple camera systems.

1.2 Thesis Outline

The thesis is organized in 5 chapters which explain the state of the art of stereo vision, algorithmic detail of the proposed binocular and trinocular disparity estimation methods, and the experimental and implementation results of the AWDE and T-AWDE algorithms.

In Chapter 2, we describe the state of the art. A taxonomy and evaluation of the binocular and trinocular disparity estimation algorithms in literature are explained in detail. In addition to that, multi-camera calibration and rectification methods are introduced.
In Chapter 3, we firstly describe a novel algorithm for binocular hardware oriented adaptive window size disparity estimation, which is named AWDE. Then, to solve problems in the binocular disparity estimation, a novel trinocular disparity estimation method is explained in detail.

In Chapter 4, experimental and implementation results of the proposed algorithms are represented in both visual and quantitative. Furthermore, the comparisons between the proposed algorithms and other similar existing algorithms in literature are given and discussed.

In Chapter 5, a summary of the master project and conclusion are given.
2.1 Binocular Stereo Matching

In order to compute depth map, the majority of researches focus on estimating the disparity information by taking two or more synchronized images from different viewpoints. In human vision system, the term disparity is described as the difference in location of corresponding points that are seen by left and right eyes [5]. There is an inverse relationship between disparity and depth. For example, the farthest object gives the smallest horizontal location change, in other words, disparity result. Generally, horizontal displacements are used in disparity estimation, but vertical disparity estimation can also be studied based on the camera setup. As in the common disparity estimation camera setups, in our system, cameras are collinearly mounted into the surface. Therefore, horizontal disparity estimation phenomenon was studied. In disparity estimation, the (x,y) coordinates of the disparity map is calculated by finding the relationship between a pixel coordinates (x,y) of the reference images and its corresponding coordinates (x1,y1) in matching image [2]. The relationship is formulized as follows:

\[ x_1 = x + s d(x, y), \quad y_1 = y \]

(2.1)

where; \( s = \pm 1 \) is a sign based on the choose of reference and matching image to make disparity positive.

Stereo matching algorithms are mainly classified into two categories such as local and global approaches. Local methods compute each pixel disparity independently based on the intensity similarity over the support window. Depend on the algorithms, matching costs are aggregated over the support region. Then the disparity value, which gives the smallest cost, is generally selected as the disparity of the pixel by using winner-take-all approach. In local methods, selection of support window plays significant role on the performance of the algorithm. It is known that in high textured regions and object boundaries, small support window size provides good performance but large window size fails. Large window size causes to blur in the object boundaries. However, in low textured regions, large window size gives good results. Therefore, some recent algorithms apply adaptable window size to get more accurate disparity estimation results.

In global methods, disparity estimation problem is considered as minimizing the energy function which consists of data and smoothness terms through various optimization techniques. Global algorithms outperform the local algorithms because global methods decreases the matching ambiguities caused by various factors. Although the global methods give better results than the local methods, they require very expensive computational cost to minimize
the energy function. For this reason, the use of this kind of algorithms is impractical in hardware implementation of disparity estimation.

Based on the Middlebury taxonomy [2] of stereo algorithms, disparity estimation algorithms usually divide the stereo matching into the following five steps as follows:

- Preprocessing
- Matching cost computation
- Cost aggregation
- Disparity selection and optimization
- Disparity refinement

2.1.1 Preprocessing

Stereo image calibration and rectification are pre-processing steps of disparity estimation intended to remove image distortions and enable stereo matching along an epipolar line. In disparity estimation, epipolar line geometry is required to search correspondence pixel in a horizontal line. Therefore, alignment mismatches between cameras in a stereo vision system make disparity estimation operation more difficult by preventing the horizontal search assumption.

The method of calibration is to take the images on a known structure that has many individual and easily detectable points. Chessboard can be used as known structure in calibration. By viewing this structure from a variety of angles, it is possible to then compute the (relative) location and orientation of the camera as well as the intrinsic parameters of the camera at the time of each image [6].

To get perfect alignment between cameras in a camera system, firstly internal calibration is applied to compute the internal camera parameters and then external camera calibration is used to find the individual camera orientations and geometrical relationship between cameras in a system by calculating the rotation and translation matrices. In order to clearly describe these operations, firstly camera model and calibration will be expressed and then rectification will be described in detail.

**Camera Model and Calibration**

In this section, some basic concepts about camera model are presented. The basic camera model, the pinhole camera model, is used to describe the calibration procedure. In this model, light is envisioned as entering from the scene or a distant object, but only a single ray enters from any particular point. This point is then “projected” onto an imaging surface [6]. In other words, image formation is done by projecting 3D points into the image plane based on the perspective projection. The pinhole camera can be modeled by using intrinsic and extrinsic parameters. To clearly describe these parameters, firstly three orthogonal coordinate systems should be defined. These three coordinate systems are the world coordinate system (X,Y,Z), the camera coordinate system (Xc,Yc,Zc), and the image coordinates system as (x,y) respectively as shown in Figure 2.1.

The intrinsic parameters represent the internal characteristic of the camera. These are focal length, principal point and skew coefficient. According to pinhole camera model, focal length can be defined as the distance from the pinhole aperture to the screen. This is shown in Figure 2.2, where f is the focal length of the camera, Z is the distance from the camera to the object. Principle point can be defined as coordinates of the center of the image in image coordinate system. In other words, it is a point which is intersection of the optical axis with
the image plane. The skew coefficient is the angle between the x and y pixel axes. Following these definitions, a camera matrix containing internal characteristics of the camera can be formulated as follows:
2. State of the Art

\[ A = \begin{bmatrix} f_x & 0 & c_x' \\ 0 & f_y & c_y' \\ 0 & 0 & 1 \end{bmatrix} \] (2.2)

Here: \((c_x', c_y')\) and \((f_x, f_y)\) represent the principal point and the focal lengths in pixel-related units. The intrinsic camera matrix parameters do not depend on the scene view. It is calculated one time and can be used many times until parameters change.

Real lenses generally have some distortion such as radial and tangential distortions. In the case of radial distortion, the location of pixels near the edges of the imager is noticeably distorted by the lenses of real cameras. It is known that, rays farther from the center of the lens are bent more than those closer to the center. That means, although the distortion is 0 at the (optical) center of the imager it increases towards the edges of the imager. The second distortion is tangential distortion and is due to manufacturing defects resulting from the lens not being exactly parallel to the imaging plane. [6]

Therefore, prior to mathematical modeling of any camera, a calibration process is needed to eliminate or correct these radial and tangential distortions caused by the lens. These distortions can be corrected by Brown’s distortion model as given in equations (2.3) and (2.4). In this thesis, we used the Matlab [7] and Open-CV [8] calibration toolboxes for determining the internal calibration parameters, \(k_1, k_2, k_3\) for radial distortion and \(p_1, p_2\) for tangential distortions respectively. According to Brown’s model, the radial and tangential distortion correction is formulated as follows:

\[ x_{corrected} = x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \]
\[ y_{corrected} = y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \] (2.3)

\[ x_{corrected} = x + [2p_1 y + p_2(r^2 + 2x^2)] \]
\[ y_{corrected} = y + [2p_1(r^2 + 2y^2) + 2p_2x] \] (2.4)

Once the image coordinates are corrected for the distortions then the extrinsic parameters of the camera are needed to provide us all the geometrical model that is required to find the transformation between the camera coordinate system and the scene coordinate system [9]. The transformation formula can be shown in as follows:

\[ sm' = A[R|t]M' \]

or

\[ s \begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \] (2.5)

where: \((X,Y,Z)\) are the scene coordinates of a 3D point in the world coordinate system. \((u,v)\) are the coordinates in image plane in pixels. \(R\) and \(T\) are extrinsic parameters. \(R\) is the rotation matrix and \(T\) is the translation matrix. The transformation in above can also be formulated in a different manner as follows:
2.1. Binocular Stereo Matching

\[
\begin{bmatrix}
  x \\
  y \\
  z
\end{bmatrix}
= R
\begin{bmatrix}
  X \\
  Y \\
  Z
\end{bmatrix}
+ t
\]

\[x' = x/z\]
\[y' = y/z\]

To correct the tangential and radial distortion, the transformation formula given above is extended as follows:

\[
\begin{bmatrix}
  x \\
  y \\
  z
\end{bmatrix}
= R
\begin{bmatrix}
  X \\
  Y \\
  Z
\end{bmatrix}
+ t
\]

\[x' = x/z\]
\[y' = y/z\]

\[x'' = x'(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + [2p_1 x'y' + p_2(r^2 + 2x'^2)]\]
\[y'' = y'(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + [p_1(r^2 + 2y'^2) + 2p_2 x'y']\]

where
\[r^2 = x'^2 + y'^2\]
\[u = f_x * x'' + c_x\]
\[v = f_y * y'' + c_y\]

Rectification

In practice, after intrinsic and extrinsic calibration, the angles and distances between cameras are adjusted. This process is called rectification. Output of this process is the creation of row-aligned images from the initial undistorted and calibrated images as shown in Figure 2.3. If the images are not rectified taking the geometrical relationship between cameras and internal parameters into consideration, the search operation for corresponding points can be computationally expensive. By using our knowledge of the geometry of the camera system, search space is reduced to minimum as much as possible.

The final rectified images can be generated by calculating new transformation matrixes from the scene coordinates to image coordinate by rotating the old ones around their optical center until the focal planes become coplanar [10] as shown in Figure 2.4. In this manner, point in left image and their correspondence points in right image are brought to the same vertical coordinate. At the end of the rectification process, intrinsic parameters become same for both cameras. This process can clearly be done by Matlab [7] or Open-CV [8] stereo calibration toolboxes.

In these toolboxes, Bouguet’s algorithm is used. The algorithm for stereo rectification simply attempts to minimize the amount of change reprojection produces for each of the two images and reprojection distortions, while maximizing common viewing area. [6]. The rotation matrix R that gives the rotation from the right camera’s image plane into the left camera’s image plane is divided in half between the two cameras. As a result of this division, the distortions that come from reprojection are minimized. The two resulting rotation matrixes are represented as R_L and R_R for the left and right camera respectively.
Figure 2.3: Image planes of undistorted rectified cameras.

Figure 2.4: Mathematical alignment of the two cameras into an unique image plane.
According to this process, left and right images are rotated by a half. At the end of the rotation, the images are put into coplanar alignment. However, to get row aligned image planes, left camera’s epipoles must be put into infinity. The formula of the rotation matrix for row aligned image planes is as follow:

\[
e_1 = \frac{T}{||T||}
\]
\[
e_2 = \frac{[-TyTx0]^T}{\sqrt{T_x^2 + T_y^2}}
\]
\[
e_3 = e_1 \times e_2
\]
\[
R_{\text{rect}} = \begin{bmatrix}
(e_1)^T \\
(e_2)^T \\
(e_3)^T
\end{bmatrix}
\]

The row aligned two image planes are then created by following rotation matrixes:

\[
R_l = R_{\text{rect}} R_l
\]
\[
R_r = R_{\text{rect}} R_r
\]

Finally, the projection matrixes are expressed as follows:

\[
P_l = M_{\text{rect},l} P'_l = \begin{bmatrix}
f_{x,l} & \alpha_l & c_{x,l} \\
0 & f_{y,l} & c_{y,l} \\
0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0
\end{bmatrix}
\]
\[
P_r = M_{\text{rect},r} P'_r = \begin{bmatrix}
f_{x,r} & \alpha_r & c_{x,r} \\
0 & f_{y,r} & c_{y,r} \\
0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
1 & 0 & 0 & T_x \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0
\end{bmatrix}
\]

where: \(M_{\text{rect}}\) and \(M_{\text{rect}}\) are the rectified left and right camera matrixes.

The Relation Between Disparity and Depth

The disparity map is computed by using the images that are taken from the left and right imagers. \(x_l\) and \(x_r\) are the horizontal positions of the points in the left and right image planes respectively. Depth map is a grayscale visualization of the distances of the objects to the reference observer. The geometrical relation between the left and right cameras allows us to show that the depth is inversely proportional to the disparity between left and right images as shown in Figure 2.5.

The disparity is defined simply by horizontal position shift \(d = x_l - x_r\). This inverse proportional relationship is shown in Figure 2.6, where we can easily derive the depth \(Z\) by using relationship between the triangles [6].

In this figure, the left and right imager pixels coordinates are shown as \((x_l, y_l)\) and \((x_r, y_r)\), respectively. The center of projection are represented as \(O_l\) and \(O_r\). \((c_x, c_y)\) is principle point. \(F\) represents the focal length. \(T\) indicates the distance between the cameras which is called as baseline. After rectification, as mentioned above, the intrinsic parameters are same for both cameras, as well as focal length. Based on this setup, the mathematical relation between the depth and disparity is shown as follows:

\[
\frac{T - (x_l' - x_r')}{Z - f} = \frac{T}{Z} \Rightarrow Z = \frac{fT}{x_l' - x_r'}
\]
2. STATE OF THE ART

Figure 2.5: Inverse relationship between the depth and disparity.

Figure 2.6: The inverse relationship is found by similarity of the triangles.

Hardware Adaptable Rectification Methods

Image rectification is one of the most essential pre-processing parts of a disparity estimation algorithm. Nevertheless, many real-time stereo-matching hardware implementations [4], [11] and [12, 13] prove their disparity estimation efficiency using already calibrated and rectified
2.1. Binocular Stereo Matching

benchmarks of the Middlebury evaluation dataset [14]. Furthermore, although some of them propose that they use rectification module in their system, they do not provide detailed information about their rectification procedures to get row aligned image pairs [15].

![Inverse Mapping](image1)

**Figure 2.7:** Inverse mapping with fractional precision coordinates. Corners indicate integer pixel coordinates.

![Forward Mapping](image2)

**Figure 2.8:** Forward mapping with fractional precision coordinates.

A real-time disparity estimation system needs to perform real-time rectification which requires solving the models of lens distortions, image translations and rotations. Look-up-table based rectification algorithms allow image rectification without demanding high complexity operations. However, in terms of hardware implementation, they require an external memory to store large size look-up-tables.

A look-up-table based approach is a straightforward approach that enables saving hardware resources [16, 17, 18]. In [16, 17, 18], the mappings between original image pixel coordinates and rectified image pixel coordinates are pre-computed and the pre-computations are used as look-up-tables. Due to the significant amount of generated data, these tables are stored in an external memory such as a DDR or SRAM [16, 17]. Using an external memory for the image rectification process may cause an additional cost for the disparity estimation hardware system or impose additional external memory bandwidth limitations on the system.

Look-up-table based rectification methods can be distinguished by two different image warping flows: forward mapping and inverse mapping. Forward mapping computes the rectified target pixel locations considering the given pixel locations in the original image. Inverse mapping computes the original source pixel locations considering the given pixel locations in the rectified image. The mapping requires separate tables for X and Y coordinates, and for the
right and left images. Therefore, four tables are required. The formulations for forward and inverse mappings are presented in equations (2.12) and (2.13), respectively. In these equations, ForwT is the forward mapping table, InvT is the inverse mapping table, Ori represents the original image taken from the camera, Rec represents the rectified image. \(Y_\text{Rec}, X_\text{Rec}, Y_\text{ori}\) and \(X_\text{ori}\) represent the Y and X coordinates.

\[
\text{Forward} : (Y_\text{Rec}, X_\text{Rec}) = (\text{Forw}_y(Y_\text{Ori}, X_\text{Ori}), \text{Forw}_x(Y_\text{Ori}, X_\text{Ori}))
\]

\[
\text{Rec}(y, x) = \text{linear interpolation(nearest neighbours of } \text{Rec}(y, x)) \tag{2.12}
\]

\[
\text{Inverse} : (Y_\text{Ori}, X_\text{Ori}) = (\text{Inv}_y(Y_\text{Rec}, X_\text{Rec}), \text{Inv}_x(Y_\text{Rec}, X_\text{Rec}))
\]

\[
\text{Rec}(y, x) = \text{linear interpolation(nearest neighbours of } \text{Ori}(y_{\text{Ori}}, x_{\text{Ori}})) \tag{2.13}
\]

Moreover, a typical rectification process utilizes fractional pixel precision which requires the linear interpolation of four pixels. The linear interpolation schemes for inverse and forward mappings are represented in Figure 2.7 and Figure 2.8, respectively. The linear interpolation process for forward mapping is more complex than the linear interpolation process of inverse mapping, since it requires additional computation and memory consumption to find the closest target pixels in the rectified image. The look-up-table based rectification hardware architectures presented in [16, 17, 18] use the inverse mapping due to its simplicity.

The size of the look-up-table depends on the size of the rectified image and the fractional precision. For example, for the rectification of 1024×768 resolution stereo images with 6 bits fractional precision, only the rectification map requires approximately 6 MB of space in a memory. This amount of data is excessive to fit into the on-chip memory of a mid-range FPGA in hardware implementation. Therefore, as expressed in [19], external memory usage can be required in hardware implementations [16, 17, 18].

### 2.1.2 Matching Cost Computation

The second step of the stereo matching algorithms is called matching cost computation. To identify the correspondence pixel in the right image, a matching cost is computed for each candidate pixels in the image. The absolute differences (AD), sum of absolute difference (SAD), squared differences (SD), sum of square difference (SSD) and census transform are the most common used pixel-based matching costs. In addition to them, recently more robust matching cost calculations are published in literature such as truncated quadratics and contaminated Gaussians [2].

The some common used matching costs are expressed in this Section. Assuming that pixel in the gray-level left image is \(I_l(x_l,y_l)\), and the candidate pixel in the gray-level right image is \(I_r(x_l-d,y)\) then the SD and AD matching costs between these pixels can be written as follows:

\[
SD(x_l, y, d) = (I_l(x_l, y) - I_r(x_l - d, y))^2 \tag{2.14}
\]

\[
AD(x_l, y, d) = (I_l(x_l, y) - I_r(x_l - d, y)) \tag{2.15}
\]

The SAD and SSD matching costs are calculated over the search window \(((2w+1)\times(2w+1))\) centered at \((x_l,y)\) pixel and expressed as follows:

\[
SAD(x_l, y, d) = \sum_{i=-w}^{w} \sum_{j=-w}^{w} |I_l(x_l + i, y + j) - I_r(x_l + i - d, y + j)| \tag{2.16}
\]
2.1. Binocular Stereo Matching

According to matching costs calculation, it is assumed that the gray-levels pixel in reference image and its correspondence pixel in the matching image are equal. Since this research study has been carried out on multiple camera system, a difference between camera gains or bias is expected. Therefore, although the matching cost computation is based on the intensity difference in the support window region, the matching costs must be insensitive to differences in camera gains or bias. Gradient-based measures and non-parametric measures such as rank and census transforms can be given as examples of the intensity insensitive matching costs [2].

The matching cost is calculated over a support window. During the calculation it has been assumed that pixels in a support region have similar intensity and also disparity values. Therefore, selection of the window size and shape plays crucial role on the accuracy of the algorithm. For the accurate disparity estimation result, the support window must be large enough to cover sufficient intensity variations. But it must also be selected small enough to cover only the pixels having same intensity values. Because, the use of fix window size and shape for pixels within the image does not give good results, therefore researchers working on this matter have focused on the use of different window size and shape at different pixels within the image.

For example, within the object boundaries, test results showed that the use of small window size is required to prevent the blurriness in the boundaries, and the use of large window size is required to get enough intensity variation within the window in low textured regions. In the case of presence of texture less areas in the images, the matching cost does not provide local minimum at the correct disparity. If any one of the matching costs expressed above is applied in very low textured region, the calculated matching cost is not able to find the correct correspondence in the other image. The main reason for that is the gray-level of the left and right images are the same along the corresponding horizontal lines in very low textured areas. Therefore, the larger window must be used in the textureless areas to get enough intensity variation.

In addition to the use and importance of size of the support window, the window shape and its determination are also very significant for matching cost calculations. The pixels near disparity discontinuities may have very different intensity values. In matching cost calculation, it is assumed that the pixel within the support window must have similar intensity and disparity values. Thus, the shape of supporting window should be chosen by avoiding the neighboring pixels that have very different intensity value than the pixel in the center of this window.

Kanade and Okutomi [20] proposed an adaptive window based method on the initial estimation of the disparity map. It is updated iteratively for each pixel to find the size and shape of the window. In this work, the intensity and disparity variance are used to choose the best window for the pixel. The disadvantage of this method is its dependence on the initial disparity estimation and therefore the algorithm used here is very sensitive to the initial estimation. Fusiello et al. [21] also proposed a method that uses multiple but limited number of windows. Thus, use of limited number of window does not cover whole regions within the image. On the other hand, Yoon and Kweon [22] proposed a locally adaptive support weight approach. In this method, a weight is calculated for every pixel within the support region based on their color dissimilarity and spatial distance to the center pixel of the window. Although this method gives very accurate result it is computationally very expensive. Hence, it is not suitable for hardware adaptability constraints.
2.1.3 Cost Aggregation

In local methods, the cost aggregation step is generally used especially in top performer algorithms. In the cost aggregation, matching costs are summed or averaged over a support region that is generally three dimensional in x-y-d space (d is the disparity range). Although the cost aggregation step provides decrease in the disparity estimation mismatches because of the noise and low texture variation, it computationally requires more complexity and memory usage. Thus, its use is not suitable for hardware adaptable disparity estimation algorithms.

2.1.4 Disparity Computation and Optimization

Local Methods

In local methods, the most important steps are matching cost computation and aggregation. Disparity selection part is trivial. In most of the local methods, the disparity value which gives the minimum cost over the disparity range is selected as final disparity at each pixel. This operation is called as “winner-take-all” (WTA). As expected, there must be uniqueness constraint of disparity matches. However, WTA can be caused a problem which a point in matching image can be matched to multiple points. For this reason, in addition matching cost, confidence metrics are used to select the correct disparity result.

The confidence metrics can be separated into five groups according to their matching costs [23, 24]. In calculation of confidence metrics, \( c_1 \) represents the minimum matching cost and \( d_1 \) is the corresponding disparity value of the minimum matching cost. Furthermore, \( c_2 \) symbolizes the second minimum matching cost and \( d_2 \) is the corresponding disparity value of the second minimum matching cost. The first one is the minimum matching cost which is also used in WTA method. Second confidence metric is based on local properties of the cost curve. The sharpness or flatness around the minimum is measured by the following equation:

\[
C_{CUR} = -2c(d_1) + c(d_1 - 1) + c(d_1 + 1)
\]  

The third one is about characteristic of the local minima. This metric provides us the strength of the minima. If there are other strong costs within the disparity range, it means that there is ambiguity in the selected minima. This metric is shown as one of the top methods in literature and require small computational complexity according to evaluation of the confidence metrics on stereo disparity estimation results by [23, 24]. It is defined as:

\[
C_{PKRN} = \frac{c_2}{c_1}
\]  

The fourth one is calculated on the entire cost curve. This metric create probability distribution of the cost curve. There are different methods in representing the probability distribution function. The last one checks the consistency between the left and right disparity maps. It is generally used to test the correctness of the calculated disparity map. For this metric, two disparity maps are computed. In the first map, the left image is used as reference image and the right image is used as matching image. In contrast to first disparity map, the right image is used as reference image and the left image is used as matching image in the computation of the second disparity map. These disparity maps are checked as follows:

\[
C_{LRC}(x, y) = -|d_1 - D_R(x - d_1, y)|
\]  

Although this metric gives very good results especially in occluded region. It requires very high computational complexity. For disparity map estimation, two different disparity maps must be computed. Therefore, it doubles the computational cost for disparity estimation.
Global Methods

In contrast to local methods, the most important part of global disparity estimation algorithms is disparity computation and optimization. Many global methods define disparity estimation as energy-minimization problem [2]. The main aim of these methods is to find a disparity function that minimizes the global energy as follows:

\[ E(d) = E_{data}(d) + \lambda E_{smooth}(d) \]  \hspace{1cm} (2.21)

The energy function consists of two terms such as data and smoothness. The data term formulates that how the disparity function match with input image pair by using the matching cost. The second term includes the smoothness assumption of the algorithm.

After global energy has been defined, many algorithms are used to minimize the energy function such as Markov Random fields, dynamic programming, max-flow and graph-cut [2]. Most top performer algorithms in the Middlebury benchmark were designed based on the global methods. Global algorithms give very good disparity estimation result but they require very expensive computational complexity and high memory bandwidth to minimize the global energy. Hence, global methods should not be preferred in hardware adaptable algorithm.

2.1.5 Disparity Refinement

Disparity refinement step is the post processing of the disparity estimation map. There are many ways to refine the calculated disparity map such as sub-pixel disparity estimation, left-right cross check, median filtering and disparity voting [2]. In some applications, image based rendering and view synthesis, high resolution disparity map is required. Therefore, sub-pixel disparity estimate method is used to increase the resolution of the disparity map.

Beside sub-pixel disparity estimation, left-right crosscheck is used to solve one of the most crucial problems in disparity map estimation. Many disparity estimation algorithms do not give good result in occluded regions. Occluded region means a part of the image which seen in one image but does not seen in the other image. Therefore, disparity values in occlude regions are not confident. In left-right crosscheck, left-to-right and right-to-left disparity maps are compared to solve the occlusion problem.

Median filter is another method to refine the disparity maps. It fills the holes and cleans the mismatches in disparity map. Disparity voting is another disparity refinement method. It basically finds the most frequently used disparity value within a window which is chosen by intensity variation and spatial distance to the center pixel of the window.

2.1.6 Hardware Adaptable Stereo Matching Methods

Many Disparity Estimation algorithms have been developed with the goal to provide high-quality disparity results. These are ranked with respect to their performance in the evaluation of Middlebury benchmarks [14]. Although top performer algorithms provide impressive visual and quantitative results [25, 26], their implementations in real-time High Resolution stereo video are challenging due to their complex multi-step refinement processes or their global processing requirements that demand huge memory size and bandwidth. For example, the AD-Census algorithm [25], currently the top published performer, provides successful results that are very close to the ground truths. However, this algorithm consists of multi disparity enhancement sub algorithms, and implementing them into a mid-range FPGA is very challenging both in terms of hardware resource and memory limitations.
Various hardware architectures that are presented in literature provide real-time DE [11, 27, 28, 12, 13, 15]. Some implemented hardware architectures only target CIF or VGA video [11, 27, 28, 12]. The hardware proposed in [11] only claims real time for CIF video. It uses the Census transform and currently provides the highest quality disparity results compared to real time hardware implementations in ASICs and FPGAs. The hardware presented in [11] uses low complexity Mini Census method to determine the matching cost, and aggregates the Hamming costs following the method in [25]. Due to high complexity cost aggregation, the hardware proposed in [11] requires high memory bandwidth and intense hardware resource utilization, even for Low Resolution (LR) video. Therefore, it is able to reach less than 3 frames per second (fps) when its performance is scaled to 1024×768 video resolution and 120 pixel disparity range.

The systems proposed in [12, 13, 15] claim to reach real time for HR video. Still, their quality results in terms of the HR benchmarks given in [14] are not provided. [12] claims to reach 550 fps for 80 pixel disparity range at a 800×600 video resolution, but it requires extremely large hardware resources. A simple edge-directed method presented in [13] reaches 50 fps at a 1280×1024 video resolution and 120 pixel disparity range, but does not provide satisfactory DE results due to low-complexity architecture. In [15], a hierarchical structure with respect to image resolution is presented to reach 30 fps at a 1920×1080 video resolution and 256 pixel disparity range, but it does not provide high quality disparity estimation for HR.

In this thesis, a hardware-oriented adaptive window size disparity estimation (AWDE) algorithm is presented to process HR stereo video with high-quality disparity estimation results for binocular camera system. Its real-time reconfigurable hardware implementation proves that the proposed algorithm is efficiently implemented in hardware and gives very good results in terms of both accuracy and speed [4]. The proposed algorithm combines the strengths of the Census Transform and the Binary Window SAD (BW-SAD) methods, thus enables an efficient hybrid solution for disparity estimation. Although the low-complexity Census method can find the disparity of the pixels where the image has a texture, mismatches are observed in textureless regions. Moreover, due to a 1-bit representation of neighboring pixels, the Census easily selects wrong disparity results. In order to correct these mismatches, AWDE algorithm proposal developed here uses the support of the BW-SAD, instead of using the complex cost aggregation method [25, 11].

The benefit of using different window sizes for different texture features on the image is observed from the DE results in [3]. The selection of a large window size improves the algorithm performance in textureless regions while requiring higher computational load. However, the usage of small window sizes provides better disparity results where the image has a texture as explained in Section2.1.2. Moreover, the use of BW-SAD provides better disparity estimation results than the SAD for the depth discontinuities. The hardware presented in [3] is not able to dynamically change the window size, since it requires to re synthesize the hardware for using different window sizes.

2.2 Trinocular Stereo Matching

2.2.1 Multi-Camera Calibration and Rectification

In contrast to binocular stereo calibration and rectification, there is a few works in literature about multi-camera calibration [29, 30, 31] and rectification methods [32, 33, 34]. These methods are mainly based on single or stereo camera calibration and rectification procedure. In multiple camera calibration, one of the cameras in the system is determined as reference camera and binocular camera calibration procedures are applied between the reference camera and the other cameras in the system.
2.2. Trinocular Stereo Matching

In trinocular stereo system, firstly the two camera pairs are determined and then positional relationships of the two camera pairs are obtained by using the binocular calibration parameters. In the system, the leftmost camera, the reference camera, is called as left camera, the camera in the middle is called as center camera and the rightmost camera is called as right camera. The trinocular camera system is shown in Figure 2.9. The transformation matrixes from the world coordinate system to image coordinate system for left, center and right cameras are represented as follows:

\[
\begin{bmatrix}
  x_l \\
  y_l \\
  z_l
\end{bmatrix} = M_l \begin{bmatrix}
  X \\
  Y \\
  Z \\
  1
\end{bmatrix} = \begin{bmatrix}
  r_{l1} & r_{l2} & r_{l3} & t_{lx} \\
  r_{l4} & r_{l5} & r_{l6} & t_{ly} \\
  r_{l7} & r_{l8} & r_{l9} & t_{lz}
\end{bmatrix} \begin{bmatrix}
  X \\
  Y \\
  Z \\
  1
\end{bmatrix} \tag{2.22}
\]

\[M_l = [R_lT_l]\]

\[
\begin{bmatrix}
  x_c \\
  y_c \\
  z_c
\end{bmatrix} = M_c \begin{bmatrix}
  X \\
  Y \\
  Z \\
  1
\end{bmatrix} = \begin{bmatrix}
  r_{c1} & r_{c2} & r_{c3} & t_{cx} \\
  r_{c4} & r_{c5} & r_{c6} & t_{cy} \\
  r_{c7} & r_{c8} & r_{c9} & t_{cz}
\end{bmatrix} \begin{bmatrix}
  X \\
  Y \\
  Z \\
  1
\end{bmatrix} \tag{2.23}
\]

\[M_c = [R_cT_c]\]

\[
\begin{bmatrix}
  x_r \\
  y_r \\
  z_r
\end{bmatrix} = M_r \begin{bmatrix}
  X \\
  Y \\
  Z \\
  1
\end{bmatrix} = \begin{bmatrix}
  r_{r1} & r_{r2} & r_{r3} & t_{rx} \\
  r_{r4} & r_{r5} & r_{r6} & t_{ry} \\
  r_{r7} & r_{r8} & r_{r9} & t_{rz}
\end{bmatrix} \begin{bmatrix}
  X \\
  Y \\
  Z \\
  1
\end{bmatrix} \tag{2.24}
\]

\[M_r = [R_rT_r]\]

Figure 2.9: The geometrical relationship between the cameras of the trinocular stereo system.
where $R_l$, $R_c$, and $R_r$ are rotation matrixes of the left, right and center cameras respectively. $T_l$, $T_c$ and $T_r$ are the translation matrixes of the cameras. Based on these transformation matrixes, the corresponding relationship between the left and right cameras and the corresponding relationship between the left and center cameras is expressed in (2.25) and (2.26) respectively.

\[
\begin{bmatrix}
    x_l \\
    y_l \\
    z_l
\end{bmatrix}
= R_l R^{-1}_r
\begin{bmatrix}
    x_r \\
    y_r \\
    z_r
\end{bmatrix}
+ T_l - R_l R^{-1}_r T_r
\]

(2.25)

\[
R_{rl} = R_l R^{-1}_r
\]

\[
T_{rl} = T_l - R_l R^{-1}_r T_r
\]

where: $R_{rl}$ and $T_{rl}$ are the rotation and translation matrixes from center camera coordinate system to left camera system.

\[
\begin{bmatrix}
    x_l \\
    y_l \\
    z_l
\end{bmatrix}
= R_l R^{-1}_c
\begin{bmatrix}
    x_c \\
    y_c \\
    z_c
\end{bmatrix}
+ T_l - R_l R^{-1}_c T_c
\]

(2.26)

\[
R_{cl} = R_l R^{-1}_c
\]

\[
T_{cl} = T_l - R_l R^{-1}_c T_c
\]

where: $R_{cl}$ and $T_{cl}$ are the rotation and translation matrixes from right camera coordinate system to left camera system. Based on these rotation and translation matrixes, the global calibration is done by optimizing parameters and minimizing the errors. At the end of global calibration, rectification is applied. During the rectification, three camera coordinate systems are reprojected into one coordinate system according to multi camera calibration parameters [32, 33, 34]. Firstly a reference camera is chosen. Then, the common coordinate system is determined based on the reference camera. After that, all cameras in the system is reprojected into the common coordinate system by using the above equations which provides the geometrical relationship between the cameras. Finally, an error minimization method is used to optimize the rectification results [8].

### 2.2.2 Trinocular Disparity Estimation

There are some drawbacks in binocular stereo matching algorithms. Binocular stereo algorithms do not give good results in occluded regions which are not visible from both cameras [35, 36, 37, 38, 39, 40]. This problem may be solved by using very high computationally complex operations such as scene segmentation. As oppose to binocular vision system, trinocular vision system get three separate view from different viewpoints of the same scene. Hence, if there is an occluded region in one of the binocular pairs, this region may be visible in the other image pair.

Trinocular disparity estimation can be simply defined as the extensive version of binocular system. In trinocular system, the center image is used as reference image and two binocular disparity estimation maps are calculated for both center-right and center-left image pairs. The most important part of trinocular disparity estimation is to find a good fusion method to merge two binocular disparity maps into robust and accurate trinocular disparity map. The main aim is to determine which binocular disparity map gives the best disparity result for searched pixel.
Chapter 3

Hardware-Oriented Adaptive Window Size Disparity Estimation Algorithms

3.1 Binocular Hardware-Oriented Adaptive Window Size Disparity Estimation Algorithm

The main focus of the AWDE algorithm is its compatibility with real-time hardware implementation while providing high quality disparity estimation results for high resolution images. The algorithm is designed to be efficiently parallelized to require minimal on-chip memory size and external memory bandwidth in its hardware implementation [4].

Figure 3.1: 49 selected pixels of adaptive windows (yellow (1): $7 \times 7$, green (2): $13 \times 13$ and blue (3): $25 \times 25$).
As a terminology, we use the term “block” to define the 49 pixels in the left image that are processed in parallel. The term “window” is used to define the 49 sampled neighboring pixels of any pixel in the right or left images with variable sizes of $7 \times 7$, $13 \times 13$ or $25 \times 25$. The pixels in the window are used to calculate the Census and BW-SAD cost metrics during the search process.

As a general rule, increasing the window size increases the algorithm and hardware complexity. As shown in Figure 3.1, in order to provide constant hardware complexity over the three different window sizes, 49 neighbors are constantly sampled for different window sizes in our proposed algorithm. “1”, “2” and “3” indicate the 49 pixels used for the different window sizes $7 \times 7$, $13 \times 13$ and $25 \times 25$, respectively.

The algorithm consists of five main parts: preprocessing, window size determination, matching cost calculation, disparity selection and disparity refinement.

### 3.1.1 Preprocessing

Rectification process requires to model distortions of the lens and the rotation of the image planes due to mechanical misalignment. Modeling of the lens distortion requires internal calibration of the cameras and the modeling of the rotation requires external calibration. Internal and External calibrations need to be processed before running a disparity estimation algorithm. In this thesis, the Matlab and Open-CV calibration toolboxes are used for external and internal calibrations [7, 8]. The pinhole camera model and calibration methods that are used by these calibration toolboxes are particularly described in Chapter 2.

In this thesis, we present an intermediate solution that compresses the rectification information. Its hardware is implemented in [19]. This novel method provides the look-up-table fit into the on-chip memory of a Virtex-5 FPGA. The low-complexity de-compression process is used to create mapping operation. This algorithm is hardware adaptable because the algorithm requires a negligible amount of hardware resources for low complexity de-compression algorithm and does not require using an external memory for its hardware implementation [19].

In contrast to the selection of the hardware implementations of [16, 17, 18], a forward mapping based rectification scheme is selected for the proposed compressed look-up-table based rectification (CLUT-R) algorithm. The forward mapping is created by inverse mapping. In CLUT-R, fractional precision is ignored and the nearest pixel is selected as source pixel. These are shown in Figure 3.2. Therefore, CLUT-R causes performance loss in the disparity estimation. This performance loss is evaluated and its negligible distortion is analyzed in Chapter 4. Ignoring fractional precision allows an efficient compression scheme.

After creating the forward mapping, the method governing compressed rectification is similar to the run-length encoding technique. In the proposed coding scheme, instead of coding the run-length of the regular order, the locations where the regular order changes are encoded. These locations are called breakpoints. Moreover, the proposed scheme includes additional specific techniques to compress the integer precision forward mapping efficiently. The regular order of the mapping of Y coordinates is encoded following a row-by-row scheme, and the regular order of mapping of X coordinates is encoded following a column-by-column scheme. Then, mappings are compressed and easily stored. In this manner, the low-complexity de-compression process is used to create mappings. Finally, Thanks to the compressed image rectification algorithm, as our hardware adaptable disparity estimation algorithm, our rectification algorithm also becomes hardware adaptable especially in terms of memory usage.
3.1. Binocular Hardware-Oriented Adaptive Window Size Disparity Estimation Algorithm

3.1.2 Window Size Determination

One of the main novel methods that are used in this algorithm is the window size determination for pixels to cover a search window with enough intensity variations. The window size determination is computationally complex operation. Therefore, to be able to design hardware adaptable algorithm, the window size of the 49 pixels in each block is adaptively determined according to the Mean Absolute Deviation (MAD) of the pixel in the center of the block with its neighbors. Although, the window size is not determined for each pixel separately, the AWDE algorithm gives very good results. This negligible effect will be shown in Chapter 4.

The equations of the MAD are presented in (3.1) for different window sizes, where \( c \) is the center pixel of the block and \( q \) is the pixel in the 49 sampled neighboring pixels within the window, \( N_c \). The center of the block is the pixel located at block(4, 4). In this algorithm, three different window sizes are used such as \( 7 \times 7, 13 \times 13 \) and \( 25 \times 25 \). The disparity can be properly found, if there is enough intensity deviation within this search window. For this reason, the determination of search window size plays significant role on disparity estimation map quality.

\[
\begin{align*}
\text{MAD}_{7 \times 7}(c) & = \frac{1}{48} \times \sum_{q \in N_c} |I_L(q) - I_L(c)| \times \text{mask}_{7 \times 7} \\
\text{MAD}_{13 \times 13}(c) & = \frac{1}{48} \times \sum_{q \in N_c} |I_L(q) - I_L(c)| \times \text{mask}_{13 \times 13} \\
\text{MAD}_{25 \times 25}(c) & = \frac{1}{48} \times \sum_{q \in N_c} |I_L(q) - I_L(c)| \times \text{mask}_{25 \times 25}
\end{align*}
\]

(3.1)

Figure 3.2: The selection of nearest source pixels from fractional inverse mapping and extraction of forward mapping with integer coordinates.
where: mask\textsubscript{7x7}, mask\textsubscript{13x13} and mask\textsubscript{25x25} represent the 49 sampled pixels from 7x7, 13x13 and 25x25 windows.

The high MAD value is a sign of high texture content and the low MAD value is a sign of low texture content. Regions with low texture content require higher window sizes to get enough intensity variation. As expressed in (3.2), a 7x7 window is used if the MAD of the center pixel is high, and a 25x25 window is used if the MAD is very low.

\[
\text{window size} = \begin{cases} 
7 \times 7 & \text{if } \text{MAD}_{7x7}(c) > tr_{7x7} \\
13 \times 13 & \text{elseif } \text{MAD}_{13x13}(c) > tr_{13x13} \\
25 \times 25 & \text{else}
\end{cases}
\] (3.2)

In the window size determination procedure, firstly the MAD is calculated over 7x7 window and then it is compared with threshold value, tr\textsubscript{7x7}. If the MAD\textsubscript{7x7} is bigger than the threshold value, it means that there is enough intensity deviation within the window 7x7 for the centered pixel and 7x7 window size is selected as suitable window size for this pixel. If the MAD\textsubscript{7x7} is less than the threshold value, tr\textsubscript{7x7}, it means that there is not enough intensity deviation within the 7x7 window and so the window size is increased to 13x13. Same operations are applied for 13x13 window as applied in 7x7 window. If there is enough intensity deviation within 13x13 window, 13x13 window size is selected as suitable window size for this pixel. If not, the 25x25 window size is selected to be able to cover enough intensity variation.

### 3.1.3 Matching Cost Calculation

In this thesis, local disparity estimation method is used. Matching cost calculation is window-based operation. Matching cost is calculated between the window around the center pixel in left image and the windows around the candidate center pixels in the right image. This search operation is done in the corresponding horizontal scan line thanks to the rectification. The matching cost is computed between the pixel in left image and each candidate in the right image by shifting the window over all possible disparity value. In the algorithm developed in this study, the matching costs of 49 pixels within the same block are calculated in parallel as shown in Figure 3.3.

![Matching cost calculation](image)

**Figure 3.3**: Matching cost calculation.
As mentioned in Chapter 2, there are different kinds of matching cost computation methods in disparity estimation. Some of them require computationally complex operations. To design hardware adaptable algorithm, efficient and less computational complexity required costs are commonly chosen.

For each matching costs, there are some advantages and disadvantages. For example, there may be some problem in real images or there may be some intensity difference between images that are taken from different cameras. To solve different kinds of problems, a hybrid solution involving the Binary Window SAD and Census cost computation methods is presented to benefit from their combined advantages.

The SAD is one of the most commonly used similarity metrics in literature. In this work, the Binary window sum of absolute (BW-SAD) is used as cost metrics instead of SAD. The use of BW-SAD provides better results than using the SAD when there is disparity discontinuity since it combines shape information with the SAD [3]. However, the computational complexity of the BW-SAD is high, thus result of this cost is provided for nine of the 49 pixels in a block and they are linearly interpolated to find the BW-SAD values for the remaining 40 pixels in a block. The selected nine pixels for the computation of BW-SAD are shown in Figure 3.4.

![Figure 3.4: 9 selected pixels in a block for BW-SAD calculation.](image)

The formula expressing the BW-SAD for a pixel \( p \) is shown in (3.4). The BW-SAD is calculated over all sampled pixels \( q \) of a neighborhood \( N_p \), where the notation \( d \) is used to denote the disparity and \( I_L \) and \( I_R \) represents the gray level intensity values of the left and right images respectively. The binary window, \( w \), is used to accumulate absolute differences of the pixels, if they have an intensity value which is similar to the intensity value of the center pixel of the window. The multiplication with \( w \) in (3.4) can be implemented as reset signal for the resulting absolute differences (AD). In the rest of the thesis, the term, “Shape” is indicated by \( w \).

\[
w = \begin{cases} 
0 & \text{if } |I_L(q) - I_L(p)| > \text{threshold}_w, q \in N_p \\
1 & \text{else}
\end{cases}
\]  

(3.3)

\[
BW - SAD(p, d) = \sum_{q \in N_p} |I_L(q) - I_R(q - d)| \cdot w
\]  

(3.4)

In contrast to BW-SAD, the low complexity Census metric is computed for all of the 49 pixels of a block. The formula expressing the Census for a pixel \( p \) is shown in (3.5). The Census is calculated over all sampled pixels \( q \) of a neighborhood \( N_p \), where the notation \( d \) is used to
denote the disparity. The Census transform maps the local neighborhood surrounding pixels to a bit string representation. If the neighboring pixel has intensity value which is less than the center pixel intensity, this pixel is coded as 1. If the neighboring pixel has intensity value which is bigger than the center pixel intensity, this pixel is coded as 0. The Census transform formula is shown as follow:

\[
\text{Census}(p, d) = \bigoplus_{q \in N_p} \varepsilon(I_L(p), I_L(q))
\]

\[
\varepsilon(I_L(p), I_L(q)) = \begin{cases} 
0 & \text{if } I_L(p) < I_L(q) \\
1 & \text{else} 
\end{cases}
\]  

(3.5)

where: \(\bigoplus\) operation represents concatenation of bit strings. Two bit strings for left and right images are created by using above formula. Figure 3.5 shows the census transform computation for two example 7\(\times\)7 windows which come from the left and right images respectively.

To compare the similarity between the two strings, the Hamming distance is used. The Hamming distance computes the number of different bits in two bit strings.

### 3.1.4 Disparity Selection

In matching cost calculation step, the BW-SAD and the Census transform costs are calculated for each pixel within the image. Depending on the texture of the image, the Census and BW-SAD have different strengths and sensibility for the disparity calculation. For this purpose, a hybrid selection method is used to combine them.

In real images, there are different problems such as difference in camera gains and bias, light reflection, and illumination difference between the images. These problems lead to bad disparity estimation results for many stereo matching algorithms. The matching cost that does
not directly depend on the intensity value of the pixel is required to solve these problems. The Census transform improve the performance of the stereo algorithms in this kind of situations [42]. The Census transform also make the disparity estimation insensitive to noises. In addition to its efficiency in disparity estimation, the Census transform is suitable for hardware adaptable real time systems.

As shown in (3.6) and (3.7), an adaptive penalty (ap) that depends on the texture observed in the image is applied to the cost of the Hamming differences between the Census transform values to increase or decrease the effect of the BW-SAD and Census transform costs on the hybrid cost.

\[
HC(p, d) = BW - SAD(p, d) + \text{hamming}(p, d) \times ap
\]  

(3.6)

\[
ap = \begin{cases} 
ap_{7x7} & \text{if window size}(p) == 7 \times 7 \\
ap_{13x13} & \text{else if window size}(p) == 13 \times 13 \\
ap_{25x25} & \text{else if window size}(p) == 25 \times 25 
\end{cases}
\]

(3.7)

The Hybrid Cost (HC) is computed by using the equation (3.6). 2's order penalty values are used to turn the multiplication operation into a shift operation in hardware implementation. If there is a texture on the block, the BW-SAD difference between the candidate disparities needs to be more convincing to change the decision of Census, thus a higher penalty value is applied. If there is no texture on the block, a small penalty value is applied since the BW-SAD metric is more reliable than the decision of Census. After applying adaptive penalty to the Census cost, the BW-SAD and census transform cost are merged by summing operation.

![Figure 3.6: Disparity selection.](image)

Subsequently, the disparity with the minimum Hybrid Cost (HC) is selected as the disparity of a searched pixel based on the winner-takes-all principle (WTA) as illustrated in Figure 3.6. The selection of the disparity at the pixel (x,y) which has the lowest matching cost is expressed as:
3. Hardware-Oriented Adaptive Window Size Disparity Estimation Algorithms

\[ d(x_L, y) = \arg \min_d (HC(x_L, y, d)) \]  

The three dimensional matrices Height × Weight × disparity range, are required to store matching costs, but it needs too much memory usage. The three dimensional matrices, which is described by the image size and disparity range and named as disparity space image, is illustrated in Figure 3.7 Therefore, in this algorithm, the matching costs are calculated for each pixel at a time and the only minimum cost is stored in memory.

![Disparity space image](image)

**Figure 3.7: Disparity space image.**

### 3.1.5 Disparity Refinement

While several disparity refinement steps for stereo matching have been proposed in literature, the most of them requires high computational complexity and are not suitable for hardware implementations. The proposed Disparity Refinement process assumes that neighboring pixels within the same Shape needs to have an identical disparity value, since they may belong to one unique object. In order to remove the faulty computations, the most frequent disparity value within the Shape is used.

As shown in Figure 3.8, since the proposed algorithm is processed seven rows in parallel during the search process of a block, the disparity refinement process only takes the disparity of pixels in the processed seven rows. The disparity refinement process of each pixel is complemented with the disparities of 16 neighbor pixels and its own disparity value. Finally, the most frequent disparity in the selected 17 contributors is replaced with the disparity of that pixel. In some situations, there is more than one most frequent value within the 17 contributors. In this case, the disparity value which is the closest pixel to the center pixel is replaced with the disparity of that pixel.

The selection of these 17 contributors proceeds as follows. The disparity of the processed pixel and the disparity of its four adjacent pixels always contribute to the selection of the most frequent disparity. Four farthest possible Shape locations are pre-computed as a mask. If these locations are activated by Shape, the disparity values of these corner locations and their two adjacent pixels also contribute. Therefore, at most 17 and at least 5 disparities contribute to the refinement process of each pixel.

In Figure 3.8, examples of the selection of contributing pixel locations are shown for three different window sizes. Considering the proposed contributor selection scheme, the pixels in
3.2 Trinocular Hardware-Oriented Adaptive Window Size Disparity Estimation Algorithm

Trinocular stereo vision overcomes the problems in the binocular stereo vision such as occlusion problems and erroneous match. The addition of third image provides additional measures and information to improve disparity estimation result especially in the problematic regions as mentioned in above.

Figure 3.8: Examples for selecting 17 contributing pixels for 7x7, 13x13 and 25x25 window sizes during the disparity refinement process (yellow (1): 7×7, green (2): 13×13 and blue (3): 25×25).

The same row with the same window size have identical masks. The masks for the seven rows of a block and three window sizes are different. Therefore, 21 different masks are applied in the refinement process. These masks turn out to simple wiring in hardware implementation.

The highest frequency selection is used for the refinement process since it can be implemented in hardware with low-complexity equality comparators and accumulators.

3.2 Trinocular Hardware-Oriented Adaptive Window Size Disparity Estimation Algorithm

Trinocular stereo vision overcomes the problems in the binocular stereo vision such as occlusion problems and erroneous match. The addition of third image provides additional measures and information to improve disparity estimation result especially in the problematic regions as mentioned in above.

Figure 3.9: Trinocular camera system.

In trinocular vision, three cameras are used to calculate disparity map. In this thesis, collinear trinocular stereo system is used. Two cameras left and right are matching cameras and the
center camera is the reference camera. The left, right and center views are denoted as $I_L$, $I_C$, and $I_R$ respectively. The collinear trinocular system is shown in Figure 3.9.

The trinocular stereo vision can simply be explained as the extended version of the binocular stereo vision. In trinocular system, two disparity maps are calculated between the left and center images and between the right and center images pairs. Therefore, there are two disparity estimates for the center camera that can be combined. The most important part of the trinocular stereo vision is the fusion of the two disparity maps.

The proposed trinocular hardware-oriented adaptive window size disparity estimation (T-AWDE) algorithm consists of seven main parts: preprocessing, window size determination, matching cost calculation, disparity selection, disparity refinement, fusion of the disparity maps, and final disparity refinement.

### 3.2.1 Preprocessing

Image rectification is one of the most essential preprocessing parts of disparity estimation. As described in Chapter 2, the rectification process requires to model distortions of the lens and the rotation of the image planes due to mechanical misalignment. To model the lens distortion, internal calibration of the cameras is required. The modeling of the rotation between the image planes requires external calibration. Therefore, the internal and external calibration is needed to be processed before the disparity estimation map computation to get row aligned images.

In this thesis, as stated before the Open-CV calibration toolbox [8] is used for external and internal calibration, for trinocular stereo system. The detailed information about the calibration methods and camera model can be seen in Chapter 2.

In trinocular stereo system, firstly the two camera pairs are determined and then positional relationships of the two camera pairs are obtained by using the binocular calibration parameters. In our trinocular camera calibration and rectification system, the cameras from the leftmost to the rightmost are called as left camera, center camera and right camera respectively. The left camera is set as the reference camera.

Based on the transformation matrixes from world coordinate system to image coordinate system of the cameras, which are given in Chapter 2, the corresponding relationship between the two camera pairs are as follows:

\[
R_{cl} = R_l R_c^{-1} \\
T_{cl} = T_l - R_l R_c^{-1} T_c
\]  

(3.9)

where: $R_{cl}$ and $T_{cl}$ are the rotation and translation matrixes from center camera coordinate system to left camera system.

\[
R_{rl} = R_l R_r^{-1} \\
T_{rl} = T_l - R_l R_r^{-1} T_r
\]  

(3.10)

where: $R_{rl}$ and $T_{rl}$ are the rotation and translation matrixes from right camera coordinate system to left camera system. By using these rotation and translation matrixes, the global calibration is applied. By using the parameters of the globally calibrated camera system, the rectification is processed to reproject three camera coordinate systems into uniform coordinate system.

The uniform coordinate system is determined by the geometrical relationship between the left and right camera. After determination of the uniform coordinate system, the transformation
3.2. Trinocular Hardware-Oriented Adaptive Window Size Disparity Estimation Algorithm

from center camera coordinate to the uniform coordinate system is calculated by using the above rotation and translation matrixes. Firstly, the center camera coordinate system is reprojected into the left camera coordinate system by using the $R_{cl}$ and $T_{cl}$ matrixes and then is reprojected into the uniform coordinate system. Finally, error minimization techniques are used to optimize the multi-camera rectification results.

3.2.2 Window Size Determination

In trinocular system, two disparity maps are calculated from the left image to the center image and from the right image to the center image. In both disparity estimations, the reference image is the center image. Therefore, the window size determination step in the T-AWDE algorithm is done just one time for the center image as explained in detail in Section 3.1.2. In other words, the window size determination step in trinocular system is same as the step in the binocular disparity estimation algorithm and so it does not require extra computational complexity.

3.2.3 Matching Cost Calculation

As explained in Section 3.1.3, the window-based local disparity estimation method is used in the proposed binocular disparity estimation. In binocular vision, the matching cost is calculated between the window around the center pixel in left image and the windows around the candidate pixels in the right image. In trinocular vision, there are two matching cost computations between the two stereo pairs. In both matching cost computations, the center image is used as reference image. The first matching cost is computed between the pixel in center image and each candidate pixels in the right image by shifting the window over all possible candidates. In this matching cost, as usual, the candidate pixels are searched in the left direction. If a point $(u,v)$ in the center image has a disparity value $d$, its corresponding point is $(u,v-d)$ in the right image. The second matching cost is computed between the pixel in center image and each candidate pixels in the left image by shifting the window over all possible candidates. In this matching cost, the candidate center pixels are searched in the right direction. If a point $(u,v)$ in the center image has a disparity value $d$, its corresponding point is $(u,v+d)$ in the left image ($d$ represents the disparity). Figure 3.10 illustrates matching cost calculation process.

![Figure 3.10: Disparity search for the trinocular stereo system.](image)

The BW-SAD and the Census costs are calculated for both image pairs as expressed in Section 3.1.3. The equation expressing the BW-SAD between the center and right images and between
the center and left images for a pixel \( p \) is shown in (3.13). The BW-SAD is calculated over all pixels \( q \) of a neighborhood \( N_p \), where the notation \( d \) is used to denote the disparity. The binary window, \( w \), is used to accumulate absolute differences of the pixels, if they have an intensity value which is similar to the intensity value of the center of the window. The shape is just computed over the center image.

\[
w = \begin{cases} 
0 & \text{if } |I_C(q) - I_C(p)| > \text{threshold}_w, q \in N_p \\
1 & \text{else }
\end{cases}
\]

(3.11)

\[
BW - SAD_{CR}(p,d) = \sum_{q \in N_p} |I_C(q) - I_R(q - d)| * w
\]

\[
BW - SAD_{CL}(p,d) = \sum_{q \in N_p} |I_C(q) - I_L(q + d)| * w
\]

(3.12)

In addition to the BW-SAD, the low complexity Census metric is also computed for both image pairs and the Hamming distance is used to compare the similarity between the results of the Census cost bit strings.

### 3.2.4 Disparity Selection

In matching cost calculation step, the BW-SAD and the Census transform costs are calculated for each stereo pairs. Depending on the texture of the image, the Census and the BW-SAD have different strengths and sensibility for the disparity calculation. To this purpose, a hybrid selection method is used to combine them based on the texture information. In trinocular vision, there are two hybrid cost computations between the two stereo pairs. The first hybrid cost is computed between the pixel in center image and left image and the second hybrid cost is computed between the center image and right image as shown in (3.13).

\[
HC_{CL}(p,d) = BW - SAD_{CL}(p,d) + \text{hamming}_{CL}(p,d) \times \text{ap}
\]

\[
HC_{CR}(p,d) = BW - SAD_{CR}(p,d) + \text{hamming}_{CR}(p,d) \times \text{ap}
\]

(3.13)

\[
\text{ap} = \begin{cases} 
ap_{7x7} & \text{if window size}(p) == 7 \times 7 \\
ap_{13x13} & \text{else if window size}(p) == 13 \times 13 \\
ap_{25x25} & \text{else if window size}(p) == 25 \times 25 
\end{cases}
\]

(3.14)

The detail information about the computation of the hybrid cost is given in Section 3.1.4. Subsequently, the disparity with the minimum Hybrid Cost is selected as the disparity of a searched pixel based on the winner-takes-all principle (WTA) and initial disparity maps for both image pairs are computed. The minimum cost values for each pixel is stored in memory to the fusion of the two disparity maps.

Winner take all approach generally properly works on local disparity estimation methods. However, in some cases, the selected disparity value that has the minimum cost in the searched pixel is not correct. To get rid of from the mismatched selected disparities, confidence metrics are used to measure the confidence of the selected matching cost. As expressed in Chapter 2, there are several confidence metrics for stereo matching in literature. Some of them are suitable for hardware implementation, but some of them are not. In the proposed T-AWDE algorithm, naive peak ratio (PKRN) metrics is used to measure the confidence of the matching cost. The PKRN basically calculates the ratio between the second minimum cost and the first minimum cost which are represented by \( c_2 \) and \( c_1 \) respectively. The PKRN is expressed as follows:
3.2. Trinocular Hardware-Oriented Adaptive Window Size Disparity Estimation Algorithm

\[ C_{PKRN} = \frac{c_2}{c_1} \]  

\[ \text{Confidence} = \begin{cases} 
1 & \text{if } C_{PKRN}(p) > \text{thresh}_{\text{conf}} \\
0 & \text{else} 
\end{cases} \]  

In winner take all approach, it is assumed that there must be high ratio between the minimum cost value and the other calculated cost values in the searched pixel. If the ratio is smaller than the threshold, the selected disparity value for the pixel is detected as unreliable. In this manner, mismatched and outlier disparities can be eliminated. Therefore, in this step, PKRN metrics are calculated to measure the confidence of the matching costs.

3.2.5 Disparity Refinement

As explained in Section 3.1.5, the proposed Disparity Refinement process assumes that neighboring pixels within the same Shape needs to have an identical disparity value, since they may belong to same object. In addition to the Shape information, the confidence metrics, which are calculated in Section 3.2.4, are also used to only select the reliable disparity values within the Shape. In order to remove the miscalculated disparity values, the most frequent disparity value within the Shape is used.

Since the proposed algorithm is processed seven rows in parallel during the search process of a block, the disparity refinement process only takes the disparity of pixels in the processed seven rows. The disparity refinement process of each pixel is complemented with the disparities of neighbor pixels within the Shape and its own disparity value. The selected pixels are activated for finding the most frequent value within the shape according to their confidence matrix result. If the confidence matrix value in the pixel is 1, the disparity value in the pixel is activated. In this manner, the use of the confidence in the selection of the contributors provides us to prevent the propagation of the miscalculated initial disparity values.

Finally, the most frequent disparity in the selected contributors is replaced with the disparity of that pixel. If the disparity value in a pixel is updated in disparity refinement step, its minimum matching cost value is also updated according to the matching cost value of the most frequent and closest pixel to that pixel. In some situations, there is more than one most frequent value within the contributors. In this case, the disparity value which is the closest pixel to the center pixel is replaced with the disparity of that pixel.

3.2.6 Fusion of the Disparity Maps

Up to this step, two disparity maps from the center image to the left image and from the center image to the right image are computed. In this step, two disparity maps are fused to solve the problems in the binocular disparity estimation maps such as miscalculated disparity values due to lack of texture, occluded areas and outliers. The disparity values coming from the same pixel in two disparity maps may be same or different. The goal of this step is that benefit from combined advantages of the two disparity maps.

Some regions in the center image are only visible in the correspondence left image region or correspondence right image region. For this reason, in these regions, the appropriate camera selection plays significant role to overcome the occlusion problem in trinocular disparity map estimation. Furthermore, although the region is visible in three cameras, the most reliable camera pair must be selected to get more accurate disparity map.

For selecting the most proper camera pair for each pixel, a novel method is proposed. The sum of neighboring disparity difference, minimum matching cost and mean absolute neighboring
intensity difference features are used to merge the two disparity maps. Firstly, for each pixel, the binary window sum of neighboring disparity difference (BW-SNDD) is calculated as one of the feature for selecting the proper camera pair instead of SNDD. The use of the BW-SNDD provides better results than using the SNDD when there is disparity discontinuity since it combines shape information with the SNDD [38]. The BW-SNDD is calculated from two disparity maps. The formula expressing the BW-SNDD for a pixel $p$ is shown in (3.17). The BW-SAD is calculated over all pixels $q$ of a neighborhood $N_p$, where the notation $D$ is used to denote the disparity map. The calculation of the binary window, $w$, is shown in above.

$$BW - SNDD(p, d) = \sum_{q \in N_p} |D(p) - D(q)| \ast w$$  \hspace{1cm} (3.17)

The proposed camera selection process assumes that neighboring pixels within the same shape needs to have a similar disparity value, since they may belong to one unique object. If the disparity deviation within the shape is high, it means that the pixel value is not reliable. Therefore, the BW-SNDD values are computed for the pixels in both disparity maps and then these values are compared with each other. If the ratio between the BW-SNDD values of the pixel that are coming from the two disparity maps is higher than the threshold, one of the disparity values, which has the smallest BW-SNDD value, is selected as proper and reliable disparity result for the pixel.

Secondly, the minimum matching costs are used as another feature to select the proper camera pair. The minimum matching costs come from the previous step. If the matching cost of the pixel is small, it means that the disparity value of the pixel is reliable. Therefore, the minimum matching cost values are used in proper camera selection. If the ratio between the matching cost values of the pixel that are coming from the two disparity maps is higher than the threshold, one of the disparity values, which has the smallest matching cost value, is selected as proper and reliable disparity result for the pixel.

Finally, texture information is used to get more accurate disparity map estimation. The MAD, which is computed in window size determination step, gives us information about the texture in a pixel. The MAD calculates the sum of absolute intensity difference within the search window. The largest window size applied in the algorithm is $25 \times 25$. Especially in real images, there are some regions that $25 \times 25$ window size is not large enough to get appropriate intensity deviation. If there is not enough intensity deviation in a pixel and its neighbors, the disparity could not be correctly calculated. In other words, the disparity values in very low textured areas are not reliable. Consequently, if the MAD value of the pixel in the center image is higher than the threshold, the selected disparity value becomes reliable. To overcome this problem, the images that are taken from the trinocular system are down sampled by 4. The trinocular disparity map is computed for the down sampled images. The use of $25 \times 25$ or $13 \times 13$ window for the down sampled images may give enough intensity deviation in the correspondence regions of the very low textured regions in the original images. Therefore, in very low textured areas, the disparity estimation result, which is calculated on the down sampled images, is selected as reliable.

To compute the trinocular disparity map of the downsampled images, the window size determination, matching cost calculation, disparity selection, and disparity refinement steps of the T-AWDE algorithm is also applied on the down sampled images. For fusion of the disparity maps of the downsampled images, only the disparity maps between the center and left and between the center and right images are used. Then, the initial trinocular disparity map is refined using the same final disparity refinement step of the T-AWDE.

In contrast to the trinocular disparity estimation of the downsampled images, in the fusion of disparity maps step of the T-AWDE, trinocular disparity map is created by fusing the three...
disparity maps. These are two binocular disparity estimation maps, which are calculated for both center-right and center-left image pairs, and the trinocular disparity estimation of the downsampled images for very low textured regions. However, if the disparity value of the pixel is not reliable in three disparity maps, the disparity value of the pixel is not assigned in the trinocular disparity map. This step detects the highly reliable seed pixels from the three disparity maps and fuses them for each pixel.

3.2.7 Final Disparity Refinement

In this step, the disparities of the highly reliable pixel are propagated. This propagation operation is also window-based. The different line based propagation operations exist in literature, but they have streaking artifacts along the propagation line [43]. In these works, there is a few number of seed pixels and so they find the disparity of the pixel by linearly interpolating the closest seed pixels to that pixel. However, in the T-AWDE algorithm, there is less number of unseeded pixels and the unseeded pixels are updated by using the most frequent disparity voting method. The proposed Final Disparity Refinement process assumes that neighboring pixels within the same Shape should have an identical disparity value, because the pixels within the object may belong to same object. In order to remove the miscalculated disparity values and find the disparity of the unseeded pixels, the most frequent disparity value within the Shape is used.

The pixel and its neighbors in the Shape are processed in the final disparity refinement. Finally, the most frequent disparity in the selected contributors is replaced with the disparity of that pixel. If there is more than one most frequent value within the contributors, the disparity value, which is the closest pixel to the center pixel, is replaced with the disparity of that pixel.
Chapter 4

Experimental Results and Evaluations

In the previous chapter, we have presented binocular and trinocular disparity estimation methods. To assess the efficiency of our method, the proposed AWDE algorithm is compared with the algorithms in literature that are also implemented in hardware. The stereo pairs in Middlebury datasets are used to evaluate the performance of the algorithm developed here [14]. In this datasets, the stereo images and their corresponding disparity maps are given.

There are 24 datasets in this database. We have selected five of the datasets (‘Tsukuba’, ‘Venus’, ‘Aloe’, ‘Art’, and ‘Cloth’). ‘Aloe’, ‘Art’, and ‘Cloth’ are used as high resolution images. ‘Tsukuba’ and ‘Venus’ are used as low resolution images. The datasets are illustrated in Figure 4.1 and Figure 4.2.

In addition to Middlebury benchmarks, the real trinocular and binocular images are captured by our camera system. In our camera setup, the cameras are mounted on a collinear surface and the distance between the each camera pair is set to 20cm. In this system, the distance between the right and center camera and the distance between the center and left camera are same, so disparity value of a same pixel in the two image pairs are also same. The captured images from the camera system is shown in Figure 4.3.

In the first section of this chapter, the performance of the proposed binocular rectification and trinocular rectification is demonstrated. In the second section of the chapter, the rates of the correctly calculated pixels in disparity map are compared between the proposed binocular disparity estimation algorithm and other existing algorithms. In the third section, the performance of the proposed trinocular disparity estimation algorithm is shown in both visual and quantitative.

4.1 Implementation Results of the Binocular and Trinocular Stereo Rectification

4.1.1 The Binocular Stereo Rectification

The proposed compressed rectification algorithm (CLUT-R) is designed using Matlab. This algorithm is also implemented in hardware by using Verilog and is verified using Modelsim 6.6c [19].

The proposed compression and decompression algorithms are evaluated using the images taken by our stereo camera system. To compare the proposed rectification algorithm, the PSNR value between the original rectified image with Caltech algorithm in Matlab toolbox and the compressed rectified image is calculated. The PSNR is calculated in (4.1).
Figure 4.1: High resolution Middlebury benchmarks.
4.1. Implementation Results of the Binocular and Trinocular Stereo Rectification

Figure 4.2: Low resolution Middlebury benchmarks.

Figure 4.3: The real images captured by the trinocular camera system.
4. Experimental Results and Evaluations

\[ PSNR = 10 \log_{10} \frac{\text{MAX}^2}{\text{MSE}} \]
\[ \text{MSE} = \frac{1}{mn} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [I(i,j) - I'(i,j)]^2 \] \hspace{1cm} (4.1)

The 1024×768 size original left and right images are shown in Figure 4.5. The original images are rectified using the Caltech rectification algorithm [7] and the proposed CLUT-R algorithm. As can be seen in Figure 4.5, the original non-rectified left and right images are not row-aligned. Hence, the horizontal search assumption in the disparity estimation algorithm is not provided and it causes very low quality disparity estimation results. The rectification results of the CLUT-R are presented in Figure 4.6 and this figure represents that the CLUT-R algorithm provides us row-aligned image pairs. In this manner, the horizontal search operation is applied. In Figure 4.7, the disparity estimation results of the AWDE [4] and the mini-census [11] algorithms by using the rectified pictures of the CLUT-R are presented. The disparity maps demonstrate that the left and right images are precisely reprojected into a uniform coordinates system and become properly row-aligned stereo pairs.

Figure 4.4: Calibration images.

The PSNR between the rectification results of CLUT-R and Caltech rectification algorithm are evaluated in Table 4.1. The PSNR of the left image is 42.67 dB, and the PSNR of the right image is 41.87 dB. Generally, a PSNR larger than 30 dB is considered acceptable according to the human eye. Therefore, CLUT-R provides very high quality rectification results. The PSNR between the non-rectified original images and the rectification results of Caltech are
4.1. Implementation Results of the Binocular and Trinocular Stereo Rectification

Figure 4.5: Original left and right images have distortions as observed near the lamp, bag, folder and cup; horizontal epipolar lines are demonstrated near the edge of these objects.

Figure 4.6: CLUT-R corrects distortions as observed near the lamp, bag, folder and cup.

also provided in Table 1 for comparison. The PSNR comparison between the non-rectified original images and rectified images by the Caltech algorithm shows that there is really high misalignment between the left and right cameras.

The performance loss of CLUT-R is also evaluated for different disparity estimation algorithms. The disparity estimation algorithms that are implemented on real-time hardware are used for the evaluation [4, 11, 12, 13, 15]. The disparity estimation results obtained using the images that are rectified by Caltech rectification algorithm are assumed as the respective ground truths of the disparity estimation algorithms. These ground truths are compared with the disparity estimation results of the respective algorithms using the images that are rectified by CLUT-R. The PSNR results are provided in Table 4.2. CLUT-R provides 36.6 dB and 30.46 dB PSNR for the AWDE algorithm [4] and Ttofis et al. [13], respectively. Therefore, the proposed CLUT-R algorithm has a negligible effect on the quality of the disparity estimation, and it can be used in different disparity estimation systems. The PSNR between the disparity estimation results using the original images and the disparity estimation results using the rectified images by Caltech are also provided in Table 4.2 for the comparison.
4. Experimental Results and Evaluations

Figure 4.7: Disparity estimation results of (a) Mini-census (b) AWDE for the rectified images of CLUT-R

Table 4.1: PSNR (dB) with the Rectified Images Produced By Matlab Calibration Toolbox

<table>
<thead>
<tr>
<th>Comparison with Rectified Left Image by Matlab Calibration Toolbox</th>
<th>Comparison with Rectified Right Image by Matlab Calibration Toolbox</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original Image</td>
<td>15.69</td>
</tr>
<tr>
<td>Proposed (CLUT-R)</td>
<td>42.67</td>
</tr>
</tbody>
</table>

Table 4.2: PSNR (dB) Comparison of the Rectified Images Produced by CLUT-R and Matlab Calibration Toolbox Using Different Disparity Estimation Algorithms

<table>
<thead>
<tr>
<th>DE using Matlab Calibration Toolbox vs. DE using Original Images</th>
<th>DE using Matlab Calibration Toolbox vs. DE using CLUT-R</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ttofis [13]</td>
<td>17.85</td>
</tr>
<tr>
<td>Georgulas [12]</td>
<td>18.97</td>
</tr>
<tr>
<td>AWDE [4]</td>
<td>19.58</td>
</tr>
<tr>
<td>Greisen [15]</td>
<td>17.57</td>
</tr>
</tbody>
</table>
4.1. Implementation Results of the Binocular and Trinocular Stereo Rectification

The hardware of the CLUT-R is presented in [19]. The proposed hardware operates at 273 MHz. Therefore, it can process up to 347 fps at a 1024 × 768 XGA video resolution. The comparison between the hardware implementation of the proposed algorithm and the stereo image rectification hardware implementations in literature, which are given in [19], are presented in Table 4.3. The hardware architecture of [41] requires a significant amount of hardware resources to support complex operations for solving the lens distortion models. Hardware architectures of look-up-table based implementations [16] and [17] require a significant amount of resources to implement external memory (E.M.) controller. Hence, combining CLUT-R with BRAM controller consumes less LUT and DFF resources than [16, 17, 41]. The DFF and LUT consumption of [18] is not available (NA). Nevertheless, the capacity of CLUT-R to fit the look-up-tables into the on-chip memory of the Virtex-5 FPGA is approximately six times more efficient than [18], as a benefit of its efficient compression scheme. As a result, the proposed rectification algorithm is suitable for efficient hardware implementation. The detail information about hardware implantation of the CLUT-R is in [19].

Table 4.3: Hardware Resource Comparison of the Rectification Hardware Implementations

<table>
<thead>
<tr>
<th>Hardware</th>
<th>Device</th>
<th>Image Resolution</th>
<th>LUT</th>
<th>DFF</th>
<th>Memory (KB)</th>
<th>E.M.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Son[41]</td>
<td>Virtex-4</td>
<td>Resolution</td>
<td>3418</td>
<td>5932</td>
<td>0</td>
<td>✓</td>
</tr>
<tr>
<td>Vancea[16]</td>
<td>Virtex-E</td>
<td>752x480</td>
<td>2459</td>
<td>2075</td>
<td>99</td>
<td>✓</td>
</tr>
<tr>
<td>Gribbon[17]</td>
<td>Spartan-2</td>
<td>640x512</td>
<td>2396</td>
<td>2396</td>
<td>16</td>
<td>✓</td>
</tr>
<tr>
<td>Park[18]</td>
<td>Virtex-5</td>
<td>640x480</td>
<td>NA</td>
<td>NA</td>
<td>104</td>
<td>X</td>
</tr>
<tr>
<td>CLUT-R</td>
<td>Virtex-5</td>
<td>1280x720</td>
<td>227</td>
<td>197</td>
<td>1</td>
<td>X</td>
</tr>
<tr>
<td>2x(CLUT-R+BRAM Contr)</td>
<td>Virtex-5</td>
<td>1024x768</td>
<td>784</td>
<td>427</td>
<td>203</td>
<td>X</td>
</tr>
</tbody>
</table>

4.1.2 The Trinocular Stereo Rectification

The real trinocular images are captured from our camera system. Before running the disparity estimation algorithm, firstly, the cameras are calibrated and then the images are rectified based on the calibration parameters. Although, the same cameras are used in our system, there are some imperfections. The use of more than one camera causes to get different sensor response. In addition to that, physical alignment is generally not satisfactorily accurate in multi-camera system. Therefore, these non-idealities must be concerned in real images that are captured from the camera systems.

Although the distance between the cameras is set to 20cm to get same disparity results from the two disparity maps, the error occurs between the disparities that come from the two stereo pairs. Furthermore, currently, the multi-camera calibration and rectification is a hot topic and the existing algorithms in literature do not provide perfect results. For example, at the end of the calibration, in the theory, although the internal parameters of the camera must be same, the internal parameters of the cameras are close to each other but not exactly same for real camera systems. The reasons mentioned above cause some distortions on rectified images. The total distortion is formulated using the horizontal position difference between the two camera pairs.

The 1024 × 768 size original left, center and right images are represented in Figure 4.8. The original images are calibrated using the OpenCV calibration toolbox [8] and then rectified according to the rectification algorithm in Chapter 3. As shown in Figure 4.8, the original non-rectified left, center and right images are not row-aligned. Hence, the horizontal search assumption in the disparity estimation algorithm is not applied and it causes to very low qual-
ity disparity estimation results. Also, the fusion of the two disparity map operation cannot be used in the non-rectified image pairs. The rectification results of the proposed method are illustrated in Figure 4.9 and the figure represents that the multi-camera rectification algorithm provides us row-aligned image pairs. In this manner, the horizontal search operation is properly applied and good quality trinocular disparity estimation results are obtained.

Figure 4.8: Original left, center and right images have distortions as observed near the tableau, monitor and bag; horizontal epipolar lines are demonstrated near the edge of these objects.

Figure 4.9: The trinocular rectification corrects distortions as observed near the tableau, monitor and bag.

4.2 Implementation Results of the Binocular Stereo

To evaluate the proposed binocular stereo algorithm performance, the algorithm is firstly tested on Middlebury benchmarks. The ground truths of the stereo pairs also exist in the Middlebury. Hence, the disparity estimation results are shown in both quantitative and visual. The images in Middlebury datasets are not synthetic, but they are captured from very ideal conditions. These image sequences are taken at regular intervals with a camera by shifting in a horizontal line [2]. The all images in a sequence are taken in the same light conditions and same camera gain and bias. Nevertheless, these images include some problems, which are caused by using of a real camera, such as misalignment, noise and lens distortions.

The comparisons of the resulting disparities with the ground-truths are done as prescribed by the Middlebury evaluation module. If the estimated disparity value is not within a 1 range of the ground truth, the disparity estimation of the respective pixel is considered as erroneous.
In other words, the percentage of bad pixels is calculated. The quality measurement is formulated as follows:

\[
\text{Error rate} = \left( \frac{100}{N} \right) \star \sum_{q \in I} \left( |D(p) - D'(q)| > \varepsilon \right)
\]  

(4.2)

where: \( D(q) \) is the ground truth value at pixel coordinate \( q \), \( D'(q) \) is estimated disparity value in that pixel. \( \varepsilon \) is a threshold (it is set to 1), \( N \) is number of pixels in the image \( I \). In this calculation, 18 pixels located on the borders are neglected in the evaluation of low resolution benchmarks (Tsukuba and Venus), and a disparity range of 30 is applied for all algorithms. However, 30 pixels located on the borders are neglected in the evaluation of high resolution benchmarks (Aloe, Art and Clothes), and a disparity range of 120 is applied for all algorithms. In these experiments, the error rate between the ground truth disparity map and estimated map are calculated based on the above formula and specifications.

There are some parameters in the proposed algorithm. The parameters of the AWDE algorithm are shown in Table 4.4. Parameters are selected by sweeping to obtain high quality disparity estimation result considering different features. The adaptive penalty parameters, which are used to combine the BW-SAD and Census transform, is determined by parametric sweep with different values on the Venus. The Venus is chose for this experiment. The main reason is that the Venus has the different characteristic regions and it provides us determine the proper parameters. According to the parametric sweep, the use of 4, 16 and 32 for adaptive penalties \( 7 \times 7, 13 \times 13 \) and \( 25 \times 25 \) respectively gives the best disparity estimation result.

<table>
<thead>
<tr>
<th>( \text{tr}_{7 \times 7} )</th>
<th>( \text{tr}_{13 \times 13} )</th>
<th>( \text{ap}_{7 \times 7} )</th>
<th>( \text{ap}_{13 \times 13} )</th>
<th>( \text{ap}_{25 \times 25} )</th>
<th>( \text{threshold}_{\text{ww}} )</th>
<th>( \text{threshold}_{\text{conf}} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>2</td>
<td>32</td>
<td>16</td>
<td>4</td>
<td>8</td>
<td>0.1</td>
</tr>
</tbody>
</table>

Table 4.4: Parameters of the AWDE

Table 4.5 compares the disparity estimation performance of the AWDE algorithm with other existing algorithms that are also implemented in hardware [12, 13, 15]. One of the main targets of these algorithms is to become successful and run in real-time for high resolution images. The evaluation of the most of the existing algorithms in literature is presented in the Middlebury. Currently, the highest quality disparity estimation algorithm that targets low resolution is [11]. These papers do not provide the disparity estimation quality results for the high resolution benchmarks of the Middlebury data-set. Thus, [11], [12], and [15] are implemented in software, and the software implementation of [13] is obtained from the authors. The disparity estimation results for the Census and the BW-SAD metrics for different window sizes are also presented in Table 4.5.

The Census and BW-SAD results that are shown in Table 4.5 are provided by sampling 49 pixels in a window. Although the Census and the BW-SAD algorithms do not provide individually very efficient results, the combination of these algorithms into a reconfigurable hardware provides an efficient hybrid solution, as demonstrated from the AWDE results in Table 4.5. Furthermore, if a fix window size is applied in the disparity estimation algorithm, the results in Table 4.5 show that the accuracy of the algorithm results with fix window size is less the proposed adaptable window size algorithm.

The proposed algorithm uses adaptable window size according to the texture information of the search region such as \( 7 \times 7, 13 \times 13 \) and \( 25 \times 25 \) window sizes. Most of the algorithms in literature use fixe window size in all regions. It is known that, low textured region require
Table 4.5: Disparity Estimation Performance Comparisons

<table>
<thead>
<tr>
<th></th>
<th>Tsukuba (288x384)</th>
<th>Venus (383x434)</th>
<th>Aloe (1110x1282)</th>
<th>Art (1110x1390)</th>
<th>Clothes (1110x1300)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chang [11]</td>
<td>4.15</td>
<td>0.56</td>
<td>3.75</td>
<td>12.80</td>
<td>2.97</td>
</tr>
<tr>
<td>Ttofis [13]</td>
<td>13.21</td>
<td>4.56</td>
<td>8.88</td>
<td>32.18</td>
<td>7.67</td>
</tr>
<tr>
<td>Georgoulas [12]</td>
<td>12.38</td>
<td>15.20</td>
<td>6.97</td>
<td>23.75</td>
<td>9.15</td>
</tr>
<tr>
<td>Census7</td>
<td>26.05</td>
<td>30.80</td>
<td>20.36</td>
<td>45.39</td>
<td>21.80</td>
</tr>
<tr>
<td>Census13</td>
<td>18.19</td>
<td>18.83</td>
<td>11.21</td>
<td>31.65</td>
<td>9.36</td>
</tr>
<tr>
<td>Census25</td>
<td>15.94</td>
<td>15.38</td>
<td>10.41</td>
<td>29.66</td>
<td>7.16</td>
</tr>
<tr>
<td>BWSAD7</td>
<td>12.19</td>
<td>19.45</td>
<td>8.31</td>
<td>34.03</td>
<td>13.33</td>
</tr>
<tr>
<td>BWSAD13</td>
<td>11.23</td>
<td>15.16</td>
<td>7.13</td>
<td>28.57</td>
<td>9.27</td>
</tr>
<tr>
<td>BWSAD25</td>
<td>10.43</td>
<td>11.12</td>
<td>6.74</td>
<td>24.74</td>
<td>6.28</td>
</tr>
<tr>
<td>FWDE7</td>
<td>9.53</td>
<td>12.59</td>
<td>5.38</td>
<td>20.87</td>
<td>5.39</td>
</tr>
<tr>
<td>FWDE13</td>
<td>7.90</td>
<td>6.82</td>
<td>4.81</td>
<td>16.97</td>
<td>3.16</td>
</tr>
<tr>
<td>FWDE25</td>
<td>8.03</td>
<td>5.66</td>
<td>5.16</td>
<td>18.12</td>
<td>3.87</td>
</tr>
<tr>
<td>AWDE</td>
<td>7.64</td>
<td>5.53</td>
<td>4.94</td>
<td>16.33</td>
<td>2.89</td>
</tr>
<tr>
<td>AWDE-HC</td>
<td>7.47</td>
<td>4.73</td>
<td>4.92</td>
<td>16.17</td>
<td>2.95</td>
</tr>
</tbody>
</table>

big window but high textured regions require small window to be able to reach sufficient intensity difference. The fix window size hardware adaptable disparity estimation algorithm results are also shown in Table 4.5 for different window sizes such as 7×7, 13×13 and 25×25. The results indicate that the use of adaptable window size provides much better results than the use of fix window size for search window. In Figure 4.10, the adaptable window and fix window disparity estimation results of Cloth are shown. The disparity estimation result by using 7×7 fix window gives very accurate results near the edge region and high textured region but the algorithm fails especially in the low textured regions. Furthermore, although the disparity estimation result by using 13×13 fix window present poor disparity result near the edge regions, it gives very accurate disparity estimation results in low textured regions. The disparity estimation result by using 25×25 fix window also gives very poor disparity estimation result near the edge regions, but it provides us to correctly find the disparity of the very low textured regions. However, the AWDE algorithm combines the advantages of the use of adaptable window size and outperforms the fix window size disparity estimation results.

The sampling is applied in 13×13 and 25×25 windows to calculate the disparity map within the same number of pixel in the all window size. If the sampling is not applied and all the pixels in a window are used during the matching process, the complexity of the AWDE algorithm increases by 12 times. In this case, the algorithm, which does not apply the sampling, becomes unsuitable for hardware implementations. The result of the high complexity version of the AWDE algorithm (AWDE-HC) is also provided in Table 4.5 for comparison. The AWDE-HC provides almost same quality results as the AWDE. Considering the hardware overhead, the low complexity version of the algorithm, AWDE, is selected for hardware implementation, and its efficient reconfigurable hardware is presented [4].

The algorithm presented in [11] uses the Census matching cost with the cost aggregation method, and provides the best results for both low and high resolution stereo images except the high resolution benchmark Clothes. However, the algorithm is not suitable for hardware implementation due to the high-complexity of cost aggregation. In particular, the algorithm requires a lot of computation complexity in high resolution images. Although many stereo matching algorithms that are also implemented in hardware gives good results on low resolu-
4.2. Implementation Results of the Binocular Stereo

Figure 4.10: Binocular disparity estimation results of (a) fix window $7 \times 7$, (b) fix window $13 \times 13$, (c) fix window $25 \times 25$, (d) AWDE

tion images, they cannot run in real-time for high resolution images as efficiently as low resolution stereo images because of the drastically increase in computational complexity. None of the compared algorithms that have a real-time high resolution hardware implementation [12, 13, 15] is able to exceed the disparity estimation quality of the AWDE for high resolution images.

The visual results of the AWDE algorithm for the high resolution benchmarks Clothes, Art and Aloe are shown in Figure 4.11. Our algorithm provides both quantitative and visual satisfactory results and reaches real-time for high resolution images in hardware implementation [4]. Because the algorithm uses different window size according the texture information within the search window, one of the common problems in stereo matching algorithms is overcome in the proposed AWDE algorithm. Many algorithms in literature uses fix window size and shape, so they fail on the textureless and edge regions. However, the AWDE algorithm gives good result in both textureless and edge regions by using different window size and shape information. In particular, the shape information provides us very accurate and precise results in depth discontinuities. Furthermore, if the results are analyzed in detail, it is seen that the main errors come from the occlusion regions. Also, there are regions that are miscalculated
4. Experimental Results and Evaluations

Figure 4.11: Visual disparity estimation results of the AWDE algorithm for HR benchmarks

(a)

(b)

(c)
4.3 Implementation Results of the Trinocular Stereo

Figure 4.12: Visual disparity estimation results of the AWDE algorithm for real images captured by the camera system because of other problems.

To evaluate the proposed binocular stereo algorithm performance, the algorithm is secondly tested on the images that are captured from our real stereo camera system. The disparity map result of the AWDE algorithm for the 1024×768 resolution pictures taken by our stereo camera system is shown in Figure 4.12. In real images, there are much more textureless regions. In addition to that, the error of the calibration and rectification process, noise on the captured image, different sensor responses of the cameras and difference in gain and bias of the cameras causes the problems for the disparity estimation of the real images. Although these problems exist in our stereo pair, our algorithm provides visual satisfactory results. The AWDE algorithm gives good result in both textureless and edge regions by using different window size and shape information. To solve the problems in the binocular stereo vision, the trinocular disparity estimation algorithm is proposed.

4.3 Implementation Results of the Trinocular Stereo

Up to now, the researchers have not worked on creating benchmarks with their ground truths for trinocular or multi-camera systems. For this reason, the existing trinocular disparity estimation algorithms generally show their results for their own images that are taken from the camera systems and some of them show their results for Middlebury stereo dataset. Due to the lack of trinocular benchmarks and their ground truths, the results are compared only
4. Experimental Results and Evaluations

In Section 4.2, the problems in binocular disparity estimation algorithm have been discussed. In binocular disparity estimation, the problems especially come from the occlusion regions and very low textured regions. To solve this kind of problems, trinocular disparity estimation algorithm is proposed. Consequently, the main aim of these experiments is to show the effect of adding a third camera and comparison between the binocular and trinocular disparity estimation results.

Table 4.6: Comparison of Disparity Estimation Results between the AWDE and T-AWDE

<table>
<thead>
<tr>
<th></th>
<th>ALOE</th>
<th>ART</th>
<th>CLOTH</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWDE_Cr</td>
<td>5.41</td>
<td>14.22</td>
<td>0.75</td>
</tr>
<tr>
<td>AWDE_CL</td>
<td>5.62</td>
<td>15.08</td>
<td>0.41</td>
</tr>
<tr>
<td>T-AWDE</td>
<td>3.83</td>
<td>7.70</td>
<td>0.27</td>
</tr>
</tbody>
</table>

To evaluate the proposed trinocular stereo algorithm performance, the algorithm is firstly tested on Middlebury benchmarks which have their own ground truths. Therefore, the dispari-

Figure 4.13: Visual disparity estimation results of the T-AWDE algorithm for HR benchmarks.
4.3. Implementation Results of the Trinocular Stereo

Figure 4.14: Visual disparity estimation results of the T-AWDE algorithm for real images captured by the camera system.

The disparity estimation results are shown in both quantitative and visual. For the quantitative result, the error rate calculation formula has given in (4.2). In this calculation, 18 pixels and 30 pixels located on the borders are neglected in the evaluation of low resolution benchmarks and high resolution image respectively. The disparity ranges are set to 30 and 120 respectively for low resolution and high resolution images. The disparity map results of the T-AWDE algorithm and binocular disparity result of the image pairs for the Middlebury benchmarks are shown in Figure 4.13. The disparity maps between the center and left image pairs are shown in first column, the disparity maps between the center and right image pairs are shown in second column and trinocular disparity maps are shown in the third column.

In the second stage of the evaluation, the algorithm is tested on the images that are cap-
Figure 4.15: Visual disparity estimation results of algorithmic steps of T-AWDE.
tured from the trinocular stereo camera system. The trinocular disparity map result of the T-AWDE algorithm for the $1024 \times 768$ resolution pictures taken by our trinocular camera system is shown in Figure 4.14.

The real images have different characteristics than the benchmarks. There are much more textureless regions and imperfections because of the calibration and rectification errors, noise, different sensor responses of the cameras and difference in gain of the cameras. Although these problems exist in our stereo system, our algorithm provides visually satisfactory binocular disparity estimation results as shown in Figure 4.12. However, if the binocular disparity results are analyzed in detail, it is clearly seen that there are errors in disparity estimation results because of the problems in above. As shown in Figure 4.14, the problems in binocular disparity maps are overcome by adding the third camera into the stereo system.

In Figure 4.15, the visual disparity estimation results of the algorithmic steps of the T-AWDE are shown. The first image demonstrates the window size selection. The black, dark-gray and gray regions represent the $7 \times 7$, $13 \times 13$ and $25 \times 25$ window sizes. The white regions show the very low textured areas and in these regions the disparity is calculated from the downsampled images. The second and third images demonstrate the binocular disparity estimation results of Center-Left and Center-Right image pairs in the end of the disparity selection step. The fourth and fifth images show the binocular disparity estimation results of Center-Left and Center-Right image pairs in the end of the disparity refinement step. The sixth image represents the most confident selected pixels in the end of the fusion step. The seventh image illustrates the disparity estimation result in the end of the final disparity refinement step and the last image is the median filtered version of the final disparity map. If the disparity results of the each algorithmic step of the T-AWDE are analyzed, it is clearly seen that low texture, occlusion, and repetitive structure problems in binocular vision and other problems, which come from the use of real images, are almost solved as shown in the final trinocular disparity map.

4.4 Hardware Implementation Results of the Proposed AWDE Algorithm

The AWDE algorithm is also implemented in hardware and the detailed information about the hardware implementation is presented in [4]. In this study, the comparison of the hardware implementation [4] results of the AWDE architecture with other existing hardware implementations that targets high resolution [12, 13, 15] is shown. The reconfigurable hardware architecture of the proposed AWDE algorithm is implemented using Verilog HDL, verified using Modelsim 6.6c. The Verilog RTL models are mapped to a Virtex-5 XCUP110T FPGA comprising 69k Look-Up Tables (LUT), 69k DFFs and 144 Block RAMs (BRAM). The proposed hardware consumes 59% of the LUTs, 51% of the DFF resources and 42% of the BRAM resources of the Virtex-5 FPGA. The proposed hardware operates at 190 MHz after place & route and computes the disparities of 49 pixels in 195 clock cycles for 120 pixel disparity range. Therefore, it can process 60 fps at a $768 \times 1024$ XGA video resolution.

In section 4.2, the comparison between the binocular disparity estimation algorithms, which are also implemented in hardware, is presented. The hardware implementations of the same algorithms are compared in Table 4.7 [4]. This comparison can be summarized as follows. Due to the high-complexity of cost aggregation process, [11] only reaches 42 fps for CIF images, thereby consuming a large amount of hardware resource. If the performance of [11] is scaled to $1024 \times 768$ for disparity range of 120, less than 3 fps can be achieved. In terms of disparity estimation quality, the overall best quantitative results following the results of AWDE are obtained from [15] as shown in Table 4.7. The hardware presented in [15] provides high disparity range due to its hierarchical structure. However, this structure easily causes faulty computations when the disparity selection finds wrong matches in low resolution. The
hardware implementation of [12] provides the highest speed performance in our comparison. However this hardware consumes 60% of the resources of a Stratix-IV FPGA due to 480 SAD computations in parallel. In our hardware implementation, only 9 SAD computations are used in parallel for the same size window and this module consumes 16% of the resources of Virtex-5 FPGA on its own.

Table 4.7: Hardware Performance Comparison

<table>
<thead>
<tr>
<th>Hardware</th>
<th>Technology</th>
<th>Image Resolution</th>
<th>Disparity Range</th>
<th>fps</th>
<th>Clock Speed (MHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chang[11]</td>
<td>ASIC-90nm</td>
<td>352×288</td>
<td>64</td>
<td>42</td>
<td>95</td>
</tr>
<tr>
<td>Ttofis[13]</td>
<td>Virtex-5</td>
<td>1280×1024</td>
<td>120</td>
<td>50</td>
<td>100</td>
</tr>
<tr>
<td>Greis.[15]</td>
<td>Stratix-III</td>
<td>1920×1080</td>
<td>256</td>
<td>30</td>
<td>130</td>
</tr>
<tr>
<td>Georg.[12]</td>
<td>Stratix-IV</td>
<td>800×600</td>
<td>80</td>
<td>550</td>
<td>511</td>
</tr>
<tr>
<td>Proposed AWDE</td>
<td>Virtex-5</td>
<td>1024×768</td>
<td>120</td>
<td>60</td>
<td>190</td>
</tr>
</tbody>
</table>
Chapter 5

Conclusion

As the interest in depth map based visual systems and consumer applications is increased, the need of real-time hardware adaptable disparity estimation algorithms is also increased. The basic problem considered in this thesis is the following: “Developing a hardware-oriented disparity map estimation algorithm using multiple cameras”. The target algorithms process the binocular and trinocular stereo images captured from the camera system. Therefore, firstly a methodology is investigated to calibrate multiple cameras, secondly, the images are rectified to provide epipolar line geometry and finally, the algorithms are developed to provide disparity map estimation of the objects on the scene.

The first achievement of the thesis consists in binocular stereo disparity estimation. In this thesis, a hardware-oriented adaptive window size disparity estimation algorithm is presented. The algorithm is also implemented in hardware and its hardware works in real-time [4]. The proposed AWDE algorithm dynamically adapts the window size considering the local texture of the image to increase the disparity estimation quality. Currently, the AWDE algorithm and its real-time hardware implementation reach higher disparity estimation quality than the existing real-time disparity estimation hardware implementations for high resolution images. The proposed reconfigurable hardware can process 60 fps at a $1024 \times 768$ XGA video resolution for 120 pixel disparity range. It shows that the AWDE algorithm is hardware adaptable and its hardware can work in real-time for high resolution image. The results demonstrate that the AWDE algorithm and its reconfigurable hardware can be used in consumer electronic products where high-quality real-time disparity estimation is needed for high resolution.

The second achievement of the thesis consists in multi-camera disparity estimation. The analyses on the binocular disparity estimation results show that there are some problems in binocular stereo vision such as occlusion, textureless and erroneous disparity calculation. This thesis proposes a new trinocular disparity estimation algorithm to solve problems of the binocular disparity estimation algorithm. The trinocular algorithm can be defined as the extended version of the binocular algorithm. In trinocular method, two disparity maps are calculated and then they are combined in an efficient method. A novel fusion method to combine the disparity maps is proposed. As mentioned in above the proposed binocular algorithm is hardware adaptable. Therefore, we can claim that the trinocular method is also hardware adaptable and its hardware implementation can work in real-time. The T-AWDE algorithm results clearly show that adding a third camera into the stereo system provides us solving the problems in the binocular disparity estimation. The proposed T-AWDE algorithm properly selects correct binocular pair for each pixels and so fusion of the two disparity map results gives us very good trinocular disparity estimation results.

The final achievement of the thesis consists in calibration and rectification. In this thesis, a
A novel compressed look-up-table based image rectification algorithm is presented. The proposed method is based on off-line compression of the rectification information to be able to fit the tables into the on-chip memory of a Virtex-5 FPGA. The presented de-compression hardware consumes a negligible amount of computational complexity, and it does not require using any external memory to store the look-up-tables. The proposed hardware is advantageous if using external memory is considered as an additional cost, or if the disparity estimation system has external memory bandwidth limitations. Furthermore, the trinocular rectification can also be designed by using the same mapping methods in the binocular rectification. Therefore, the trinocular rectification method can easily be implemented in hardware. The presented binocular rectification hardware [19] would be even more profitable if it is adapted for high resolution multiple camera disparity estimation systems.

In addition to the contributions, there is still some works to be done. Firstly, the quality of the disparity estimation would be increased using multiple disparity refinement processes. Secondly, instead of direct use of the matching costs, the matching cost can be represented exponentially. In this manner, the matching costs become more robust. Finally, angular camera orientation would be taken into consideration in the future. In the proposed T-AWDE algorithm, the target orientation for multiple cameras is planar. In angular orientation, the cameras will be positioned on circle to provide large angle of view disparity estimation. The T-AWDE algorithm should be adapted according to angular camera orientation.


[40] M. Mozerov, A. Amato, and X. Roca, “Occlusion handling in trinocular stereo using composite disparity space image.”
