

## ABSTRACT

Title of thesis: AN ARCHITECTURE FOR HIGH-THROUGHPUT  
AND IMPROVED-QUALITY STEREO VISION  
PROCESSOR

Sang-Kyo Han, Master of Science, 2010

Thesis directed by: Gang Qu, Professor  
Department of Electrical and Computer Engineering

This paper presents the VLSI architecture to achieve high-throughput and improved-quality stereo vision for real applications. The stereo vision processor generates gray-scale output images with depth information from input images taken by two CMOS Image Sensors (CIS). The depth estimator using the sum of absolute differences (SAD) algorithm as stereo matching technique is implemented on hardware by exploiting pipelining and parallelism. To produce depth maps with improved-quality at real-time, pre- and post-processing units are adopted, and to enhance the adaptability of the system to real environments, special function registers (SFRs) are assigned to vision parameters. The design using 0.18 $\mu$ m standard CMOS technology can operate at 120MHz clock, achieving over 140 frames/sec depth maps with 320 by 240 image size and 64 disparity levels. Experimental results based on images taken in real world and the Middlebury data set will be presented. Comparison data with existing hardware systems and hardware specifications of the proposed processor will be given.

**AN ARCHITECTURE FOR HIGH-THROUGHPUT AND IMPROVED-  
QUALITY STEREO VISION PROCESSOR**

By

Sang-Kyo Han

Thesis submitted to the Faculty of the Graduate School of the  
University of Maryland, College Park, in partial fulfillment  
of the requirements for the degree of  
Master of Science  
2010

Advisory Committee:

Professor Gang Qu, Chair/Advisor  
Professor Shuvra S. Bhattacharyya  
Professor Ankur Srivastava

© Copyright by  
Sang-Kyo Han  
2010

To my parents and brother

## Acknowledgments

I would like to thank Dr. Gang Qu for advising and supporting me. He has always been considerate to me, and encouraging me to find better ways on the degree course. I could complete this work with his help and understanding. I also would like to thank the members of the committee, Dr. Shuvra S. Bhattacharyya and Dr. Ankur Srivastava. Their interests and constructive discussions on my research were helpful to me. It is my pleasure to thank the Korea Institute of Science and Technology (KIST). The work experience in the KIST deepened my knowledge and competence in the field of stereo vision systems. I would like to give my deepest gratitude to my parents and brother. Their endless support and love have motivated me not to give up in my life.

The life in the University of Maryland was the most painful time I have ever gone through. However, I would like to believe that it is a present from the God, and I am convinced that to open the present in the future will become delight.

# Table of Contents

1. Introduction .....	1
1.1. Research Backgrounds .....	1
1.2. Thesis Organization.....	3
2. Backgrounds of Stereo Vision.....	4
2.1. Image and Camera Model.....	4
2.1.1. Digital Image and Image Coordinate System .....	4
2.1.2. Pinhole Camera Model and Projection Matrix.....	5
2.1.3. Camera Parameters .....	6
2.2. Camera Calibration and Lens Distortion .....	6
2.2.1. Linear Approach to Camera Calibration.....	7
2.2.2. Lens Distortion .....	7
2.3. Epipolar Geometry .....	8
2.4. Image Rectification .....	9
2.4.1. Control of Homography Matrix by Special Function Registers .....	11
2.5. Image Filtering .....	11
2.5.1. Smoothing .....	12
2.5.2. Edge Detection .....	13
2.5.3. Bilateral Filtering.....	13
2.5.4. Control of Bilateral Filter Offset by SFR.....	15
2.6. Stereo Matching and 3-D Reconstruction .....	15
2.6.1. Ambiguity in Correspondence Problem.....	15
2.6.2. Stereo Matching by Correlation .....	16
2.6.3. 3-D Reconstruction .....	17
2.7. Occlusion Region and Handling It.....	18
2.7.1. LR Consistency Check.....	18
2.7.2. Control of LR Consistency Check by SFR .....	19
2.7.3. Median Filtering .....	19
3. Architecture of Proposed Stereo Vision Processor.....	21
3.1. System Prototype.....	21
3.2. Top level Architecture .....	21
3.3. Image Rectifier.....	22
3.3.1. RECT Core Block.....	23
3.4. Bilateral Filter .....	24
3.4.1. Weight Calculation Block .....	25
3.4.2. SUM Block.....	26
3.5. Depth Estimator and LR Consistency Check.....	27
3.5.1. Pipelined and Parallelized Implementation of SAD Algorithm.....	28
3.5.2. Left-Right SAD Block .....	29
3.6. Median Filter .....	31
4. Performance Evaluation and Comparison with Previous Works .....	33
4.1. Experiments based on Middlebury Stereo Data and Its Analysis.....	33
4.2. Experiments based on Real Scene and Its Analysis .....	36
4.3. Hardware Specifications .....	38

4.4. Comparison with Previous Hardware Works .....	39
5. Conclusions .....	41
Bibliography.....	42

# Chapter 1: Introduction

## 1.1. Research Backgrounds

Neurons sensitive to binocular disparity have been found in the visual cortex of many mammals, and respond differentially to binocular stimuli providing cues for stereoscopic depth perception [1][2]. This biological evidence has been the base for stereo vision systems. Stereo vision can produce depth information from two images taken on slightly different positions. The advantage over prevalent distance measurement devices like ultrasonic or laser equipments is that a complete scene is captured at once, yielding a contact-free acquisition of the spatial impression [14]. Therefore, it is useful and important in many visual applications such as 3-D TV, autonomous navigation, object recognition and surveillance systems to provide more detailed information than devices based on wave reflection. These application domains require the stereo vision system to guarantee high-throughput and high-quality depth maps at real time.

Generating the depth information from two images by triangulation requires the establishment of correspondence between a pair of images known as stereo matching. Since stereo matching has computational complexity and the matching technique directly affects the accuracy of disparity map, various matching algorithms have been devised. They can be classified as area-based [3], feature-based [4] and phase-based [5][6]. In the area-based approach, the correspondence problem is solved by matching image intensity patterns. The feature-based methods first detect edges and then seek matches between intersections of these edges. Resulting maps are not as detailed as the area-based type to calculate the depth for every pixel. This technique also becomes less effective in image regions without edges. In the phase-based algorithm, the disparity is estimated in terms of phase differences in the spectral components of stereo image pair.

Recently, these algorithms have been implemented on hardware instead of software in order to achieve the real-time performance. [7], [8] and [9] employed the area-based approaches such as dynamic programming, sum of absolute differences (SAD) algorithm and Census transform respectively on FPGAs. [10] and [11]



configured FPGA-based stereo systems using the phase-based method. [12]-[16], [27][28] reported ASIC design of stereo vision processors. However, further improvement is required for stereo vision to be used in practical applications. Since the stereo matching is the process of comparing everywhere of two images in order to find out the corresponding points, it has a very high computation demand. This burden can be drastically mitigated by laying image rectification prior to the matching process. On the other hand, the stereo vision system is exposed to various environments and variations in real applications. For example, two input imagers may not be identical, and furthermore they may produce noisy images. These problems should be alleviated to avoid the accuracy degradation of depth map.

Our research aims to design a stereo vision processor to provide high-throughput and high-quality depth maps. We integrate a depth estimator based on SAD algorithm, an image rectifier, a bilateral filter, a left-right (LR) consistency check, a median filter, and special function registers on a single processor. This design can run at 120MHz clock frequency, achieving over 140 fps dense depth maps with image size of 320\*240 and 8-bit disparities. Our processor has the following advantages over existing stereo vision hardware systems.

First, an image rectifier as pre-processing was employed to solve the correspondence problem more efficiently. Image rectification is to transform the images so that epipolar lines are aligned horizontally, and thus it simplifies stereo matching from two dimensional (2-D) area searches to one dimensional (1-D) searches along the epipolar line. In case of arbitrary placement of the camera, which is common, the epipolar line is skewed and the 1-D search will be ineffective. Therefore, the rectification is indispensable for stereo vision to be applicable to real world.

Secondly, we adopt a bilateral filter to improve the accuracy of depth maps. Using image filtering as another pre-processing is obviously necessary for preserving signal details while removing the noise. Besides, any stereo algorithm should compensate photometric variations between cameras of the stereo rig. Considered for the real-time requirement, conventional filters with iterative property in algorithm are not suitable to vision applications. It has been verified that the bilateral filtering is suitable for these goals [18].

Thirdly, post-processing units such as LR consistency check and median filter were integrated for better image accuracy. The LR consistency checker compares results of two operations, LR matching and RL matching. Matched points are confirmed only when the minimum disparity of LR matching is very close to that of RL matching. Consequently, the technique can detect incorrect matches caused by occlusion regions where areas visible to one camera only exist. The median filter is effective in diminishing outliers in disparity maps, which are caused by occlusion regions [24], and in eliminating erroneous matches.

Finally, the processor had the flexibility to control the accuracy of depth map. This is achieved by assigning vision parameters to special function registers. The function not only enhances the adaptability of processor in various stereo vision systems, but also eliminates complication to reconfigure FPGAs. We can easily control the quality of depth map by adjusting register values in each block.

## 1.2. Thesis Organization

This paper is organized as follows. Chapter 2 reviews the backgrounds of stereo vision applied to our processor such as projection matrix, camera's parameters, epipolar geometry, image rectification, bilateral filtering, SAD matching algorithm, and so on. Chapter 3 describes the hardware architecture of proposed stereo vision processor. First, its top level architecture is presented, and then the function of each component will be described with block diagrams. Chapter 4 shows experimental results based on Middlebury stereo data as well as images taken from real scene, and analyzes their output images. Besides, hardware specifications of designed processor and comparison data with previous hardware designs will be given. In chapter 5 the conclusion of this work will be found.

## Chapter 2: Backgrounds of Stereo Vision

### 2.1. Image and Camera Model

The image is obtained by projection of the object in three dimensions on camera. As a camera model, a pinhole camera model is usually used, and the image by this model is called perspective projection. The camera has not only intrinsic parameters such as focal length and coordinate of image center, but extrinsic parameters such as rotation and translation. The extrinsic parameters explain the transformation between camera coordinate system and world coordinate system. The projection and camera parameters can be concisely described using linear algebra.

#### 2.1.1. Digital Image and Image Coordinate System

The digital image is produced from the arrays of two dimensional pixels. In grayscale images, the brightness of each pixel has 8-bit data and its level is one of 256 values ranging from 0 to 255. In case of color images each pixel is composed of three small and very close but still separated RGB light sources. Each 8-bit RGB component can have 256 possible values ranging from 0 to 255. Figure 2.1 shows the coordinate system of digital image. The origin is at upper-left corner. The horizontal axis of right direction is  $u$  and the perpendicular axis of downward direction is  $v$ , and  $(u, v)$  is called the digital image coordinate.

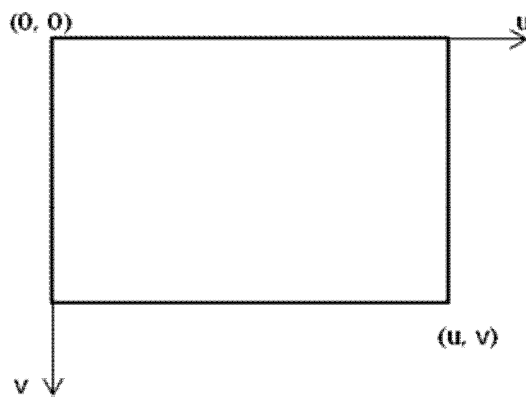


Figure 2.1: The Coordinate System of Digital Image

### 2.1.2. Pinhole Camera Model and Projection Matrix

The pinhole camera model with image plane in front of a focal point (or pinhole) is given in Figure 2.2. This model explains the relation between coordinate of object in three dimensions and its two dimensional image as follows.

$$x = f \cdot \frac{X}{Z}, \quad y = f \cdot \frac{Y}{Z} \quad (1)$$

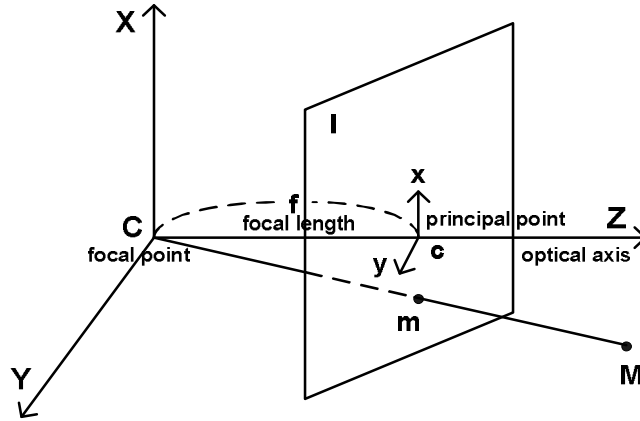


Figure 2.2: Pinhole Camera Model

We call this projection as the perspective projection in which a point  $M$  on the object, a pinhole and a point  $m$  on image plane  $I$  are in the straight line. Equation (1) can be expressed as (2) in the linear form

$$s \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}, \quad (2)$$

where  $s$  is the scale value. By letting  $\mathbf{m} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$ ,  $\mathbf{P} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$  and  $\mathbf{M} = \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$ , we obtain

$s\mathbf{m} = \mathbf{P}\mathbf{M}$ . The 3 by 4 matrix  $\mathbf{P}$  is called the projection matrix of perspective projection.

### 2.1.3. Camera Parameters

The world and camera coordinate systems are related by a set of physical parameters, such as the focal length of lens, the size of pixels, the position of principal point, and the position and orientation of camera. The projection matrix can be represented as a general form by using camera's intrinsic and extrinsic parameters like

$$\mathbf{P}=\mathbf{A}[\mathbf{R} \mid \mathbf{t}]. \quad (3)$$

The projection matrix means that one 3-D point described by the world coordinate system is projected on image plane through the intrinsic parameters ( $\mathbf{A}$ ) and the transformation ( $[\mathbf{R}, \mathbf{t}]$ ) to the camera coordinate system. The camera's intrinsic matrix  $\mathbf{A}$  in equation (3) consists of intrinsic parameters only and is given as

$$\begin{bmatrix} fk_u & -fk_u \cot \theta & u_0 \\ 0 & fk_v / \sin \theta & v_0 \\ 0 & 0 & 1 \end{bmatrix}$$
 where  $f$  is the focal length,  $k_u$  and  $k_v$  indicate unit length of each

axis on image plane,  $\theta$  is an angle between two axes of image plane, and  $(u_0, v_0)$  tells the coordinate of image center. Recently, we may regard the angle  $\theta$  as  $\frac{\pi}{2}$  and the shape of pixel as square due to the advance of manufacturing technology. Therefore, the intrinsic parameters are now composed of three elements: the focal length and the coordinate of image center. On the other hand, the  $3 \times 3$  matrix  $\mathbf{R}$  and the  $3 \times 1$  matrix  $\mathbf{t}$  denote rotation and translation respectively for 3-D displacement from the world coordinate system to the camera coordinate system. The camera coordinate system can be obtained by rotating the world coordinate system first and then translating it, and the two matrices are called the camera's extrinsic parameters.

## 2.2. Camera Calibration and Lens Distortion

The geometric camera calibration is a process to estimate the intrinsic and extrinsic parameters of a camera. We assume that the camera observes a set of features such as points or lines with known positions in fixed world coordinate system. The camera calibration can be modeled as an optimization process, where the discrepancy

between observed image features and their theoretical positions is minimized with respect to the camera's intrinsic and extrinsic parameters. Figure 2.3 shows the setup for camera calibration. The calibration frame is formed by three grids drawn on orthogonal planes. Other patterns such as lines or geometric figures could be used as well. The calibration process can be decomposed into the computation of the perspective projection matrix ( $\mathbf{P}$ ) and the estimation of camera's intrinsic and extrinsic parameters.

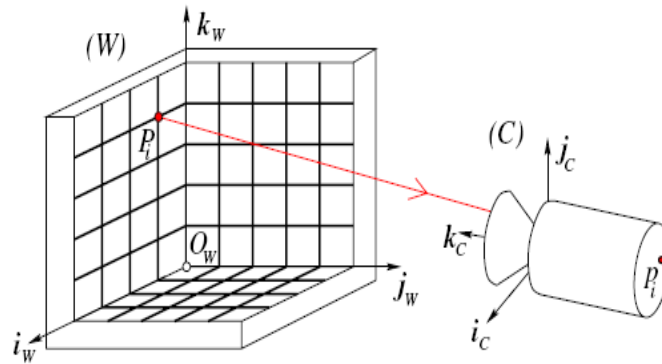


Figure 2.3: Camera Calibration Setup

### 2.2.1. Linear Approach to Camera Calibration

The most commonly used camera calibration method is perhaps the direct linear transformation (DLT) method using a set of control points whose object plane coordinates are known. The control points are normally fixed to a calibration rig. The problem is essentially to calculate the mapping between 2-D image coordinates ( $\mathbf{m}_i$ ) and 3-D object space coordinates ( $\mathbf{M}_i$ ). For correspondence between the 3-D and the 2-D, the mapping takes the form of a  $3 \times 4$  projection matrix  $\mathbf{P}$  such that  $\mathbf{m}_i = \mathbf{P}\mathbf{M}_i$  for all  $i$ . [21] presents mathematical explanations associated with this method.

### 2.2.2. Lens Distortion

We have assumed that the camera is equipped with a perfect lens. Real lenses, however, suffer from radial distortion by which the camera itself may produce distorted images and will lead to wrong correspondences especially in the outer regions of the

image. The radial distortion is a type of aberration to depend on the distance separating the optical axis from the point of interest. The applications using wide-angle or low-cost lenses suffer from more distortions. It has been reported that a typical 0.5-1% distortion is equivalent to a positional error of 1.25 to 2.5 pixels at the extremes of the image [20]. The lens's radial distortion parameters should be estimated along with the camera's intrinsic and extrinsic parameters. [21] deals with mathematical expansions from the projection process.

### 2.3. Epipolar Geometry

The epipolar geometry came from the correspondence problem in stereo vision. Figure 2.4 shows the epipolar geometry between two camera systems. We assume that the intrinsic parameters of each camera are known. Let  $O_L$  and  $O_R$  be the optical center of left and right cameras respectively. A 3-D point  $\mathbf{M}$  and two optical centers build the epipolar plane, and the intersections of this plane and each image become the epipolar lines denoted by  $l$  and  $l'$ . Given a point  $\mathbf{m}$  in the left image, the corresponding point on the right image is constrained to lie on the epipolar line of  $\mathbf{m}'$ , called the epipolar constraint. The points  $\mathbf{e}$  and  $\mathbf{e}'$  are known as the epipoles of left and right image, respectively. We observe that all epipolar lines on the left image pass through  $\mathbf{e}$  and similarly all epipolar lines on the right image pass through  $\mathbf{e}'$ .

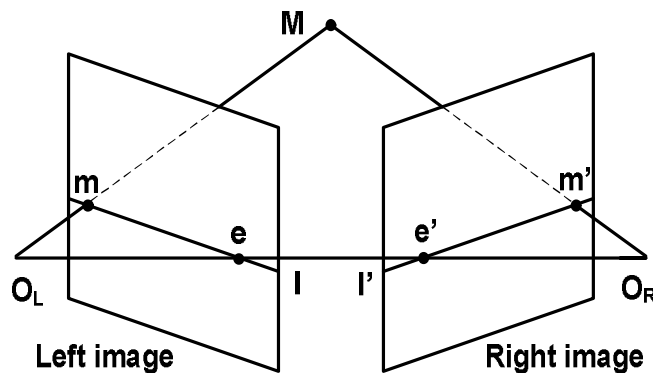


Figure 2.4: Epipolar Geometry between a Pair of Images

Let's suppose that the 3-D coordinates in camera coordinate system are  $\bar{\mathbf{m}}$  and  $\bar{\mathbf{m}}'$  for the image correspondences  $\mathbf{m}$  and  $\mathbf{m}'$ , respectively. The movement of one camera to any position of the other one is represented with the rotation ( $\mathbf{R}$ ) and the translation ( $\mathbf{t}$ ). Since image correspondences and lens centers are built on same plane, the three vectors  $\bar{\mathbf{m}}$ ,  $\mathbf{t}$  and  $\mathbf{R}\bar{\mathbf{m}}' + \mathbf{t}$  are coplanar. Therefore, one of them must lie in the plane spanned by the other two, or

$$\bar{\mathbf{m}}^T (\mathbf{t} \times (\mathbf{R}\bar{\mathbf{m}}' + \mathbf{t})) = 0. \quad (4)$$

Here,  $\mathbf{R}\bar{\mathbf{m}}' + \mathbf{t}$  denotes the coordinate of point  $\mathbf{m}'$  on the camera coordinate system for the left image. Using the transform

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}_x = \begin{bmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{bmatrix},$$

the equation (4) is written as

$$\bar{\mathbf{m}}^T (\mathbf{t} \times (\mathbf{R}\bar{\mathbf{m}}' + \mathbf{t})) = \bar{\mathbf{m}}^T [\mathbf{t}]_x (\mathbf{R}\bar{\mathbf{m}}' + \mathbf{t}) = \bar{\mathbf{m}}^T \mathbf{E} \bar{\mathbf{m}}' = 0, \quad (5)$$

where  $[\mathbf{t}]_x \mathbf{R}$  is the essential matrix  $\mathbf{E}$ , and the notation  $[\mathbf{t}]_x$  indicates the skew-symmetric matrix such that  $[\mathbf{t}]_x \mathbf{x} = \mathbf{t} \times \mathbf{x}$  is the cross-product of the vectors  $\mathbf{t}$  and  $\mathbf{x}$ . The equation (5) is called the epipolar equation, and takes 2-D image coordinates and camera movements only as the variables.

## 2.4. Image Rectification

The image rectification is an essential operation to simplify the correspondence problem. The task transforms the images so that the epipolar lines are aligned horizontally, which is equivalent to mapping the epipole  $\mathbf{e}$  to a predetermined point  $\mathbf{i} = [1 \ 0 \ 0]^T$  (a point at  $\infty$  of horizontal line) by applying the homography matrix  $\mathbf{H}$  to each image. Figure 2.5 illustrates the stereo pair to rectify figure 2.4. Through this process stereo matching can easily take advantage of the epipolar constraint, and the



search area is reduced from two dimensional areas to one dimensional horizontal line. Moreover, image rectification is indispensable in real applications where the epipolar line may be skewed by the arbitrary arrangement of cameras. The following formally describes the rectification of stereo images.

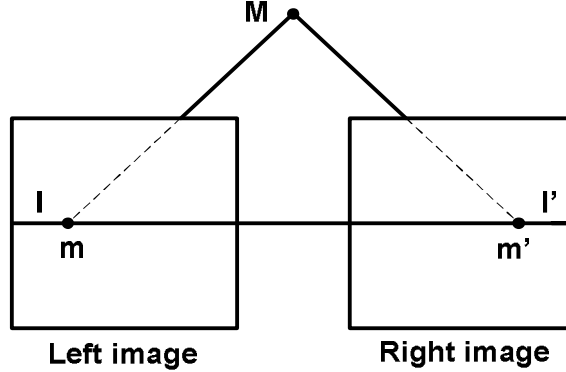


Figure 2.5: A Rectified Stereo Pair

Given the homography matrices  $\mathbf{H}$  and  $\mathbf{H}'$  to be applied to images  $I$  and  $I'$  respectively, let  $\mathbf{m}$  and  $\mathbf{m}'$  be corresponding points to satisfy equation (4). Rectified image points can be expressed as

$$\bar{\mathbf{m}} = \mathbf{H}\mathbf{m} \quad \text{and} \quad \bar{\mathbf{m}}' = \mathbf{H}'\mathbf{m}'.$$

From equation (4), we obtain

$$\bar{\mathbf{m}}'^T \bar{\mathbf{F}} \bar{\mathbf{m}} = [\mathbf{H}'\mathbf{m}']^T \bar{\mathbf{F}} [\mathbf{H}\mathbf{m}] = \mathbf{m}'^T [\mathbf{H}'^T \bar{\mathbf{F}} \mathbf{H}] \mathbf{m} = \mathbf{m}'^T \mathbf{F} \mathbf{m} = 0,$$

where it is a convention that the fundamental matrix for a rectified image is defined as

$$\bar{\mathbf{F}} = [\mathbf{i}]_x = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix}.$$

This points out that any point  $\mathbf{m}'$  corresponding with  $\mathbf{m}$  must line on the epipolar line  $\mathbf{F}\mathbf{m}$ . On the other hand, the fundamental matrix and the projection matrix satisfy the following relation [19]:

$$\mathbf{F} = [\mathbf{P}\mathbf{p}'^\perp]_x \mathbf{P}\mathbf{P}'^+, \quad (6)$$

where  $\mathbf{P}'^{\perp}$  is a null vector of matrix  $\mathbf{P}'$  and  $\mathbf{P}'^+$  is the pseudo-inverse of  $\mathbf{P}'$ . Note that the fundamental matrix and the homography matrix satisfy a relation

$$\mathbf{F} = \mathbf{H}'^T [\mathbf{i}]_k \mathbf{H} . \quad (7)$$

Therefore, we can find a pair of the homography matrix from the equations (3), (6) and (7), using camera's intrinsic and extrinsic parameters known by calibration.

#### 2.4.1. Control of Homography Matrix by Special Function Registers

We have assumed that the  $\mathbf{H}$  matrices in one vision system don't change. However, it is worth noticing that positions of two cameras may not be always fixed. This occurs sometimes by camera replacement, or vibration generated from vision system's movement, and these makes the epipolar line be skewed so that the 1-D search will be ineffective. In addition, the unwanted situation can come from camera itself. Even though a pair of camera is highly analogous in appearance, they are not identical so that they have different focal lengths and different coordinates of image center. Consequently, camera's intrinsic and extrinsic parameters can vary according to each camera and stereo rig, which require the change of homography matrices. The former processors using FPGAs cannot avoid source code modifications and its logic synthesis in order to reconfigure the devices every that situation. In proposed design, however, we can control the parameters outside the processor without hardware reconfiguration because elements in the homography are assigned as special function registers.

### 2.5. Image Filtering

Pixel's point operations are a very simple class among the techniques of image enhancement which makes images look better, but they don't mean a change in a small neighborhood. New classes of operations are necessary which combine the pixels of a local neighborhood in an appropriate manner and yield a result to form a new image. If gray value doesn't change in a small neighborhood, it implies an area of constant gray values such as an object. If the value changes, it may be at the edge of an object. Edge detection aims at estimating the boundaries of the object, while smoothing gives adequate averages for the gray values within the object. The followings describe the

base to handle a wide range of image processing tasks: simple smoothing and edge detection, and introduce the bilateral filter employed to our stereo vision system as pre-processing.

### 2.5.1. Smoothing

The smoothing operations are used primarily for suppressing spurious effects produced by poor sampling systems or transmission errors. We briefly present smoothing techniques in both spatial and frequency domains.

Neighborhood averaging is a spatial-domain method for image smoothing. Given an  $N \times N$  image  $f(x, y)$ , the procedure is to produce a smoothed image  $g(x, y)$  whose gray level at every point  $(x, y)$  is obtained by averaging the gray-level values of the pixels of  $f$  contained in a predefined neighborhood of  $(x, y)$ . In other words, the smoothed image is obtained by using the relation

$$g(x, y) = \frac{1}{M} \sum_{(n, m) \in S} f(n, m) \quad (8)$$

for  $x, y = 0, 1, \dots, N-1$ .  $S$  is the set of coordinates of points in the neighborhood of (but not including) the point  $(x, y)$ , and  $M$  is the total number of points defined by the coordinates in  $S$ .

Edges and other sharp transitions such as noise in the gray levels of an image contribute heavily to the high frequency content of its Fourier transform. Therefore, blurring can be achieved by attenuating a specified range of high frequency components in the transform of a given image. Let  $g(x, y)$  be an image formed by the convolution of an image  $f(x, y)$  and a position-invariant operator  $h(x, y)$  like

$$g(x, y) = h(x, y) * f(x, y) \quad (9)$$

From the convolution theorem, we have the frequency-domain relation

$$G(u, v) = H(u, v)F(u, v) \quad (10)$$

where  $F(u, v)$  is the Fourier transform of the image  $f(x, y)$ . The problem is to select a function  $H(u, v)$  which yields  $G(u, v)$  by attenuating the high frequency components of  $F(u, v)$ . The inverse transform of  $G(u, v)$  will then generate the desired smoothed image  $g(x, y)$ . Since high frequency components are filtered out, and information in the low

frequency range is passed without attenuation, this method is commonly called low-pass filtering. The function  $H(u,v)$  is referred to as transfer function of a filter.

### 2.5.2. Edge Detection

The edges are points where brightness in an image sharply changes. Edge detection [22] is a sharpening technique to be useful as enhancement tools. Sharpening is analogous to differentiation, while averaging to integration. Detecting edges requires a filter operation to emphasize the change in gray values and to suppress areas with constant gray values. The derivative operators are suitable for such an operation. The first derivative has an extremum at the edge, while the second derivative crosses zero where the edge has the steepest ascent. Both of them can be used to detect edges. In two dimensions, the sum of the two second partial derivatives is called the Laplace operator and is denoted by  $\Delta$ . The Laplacian of a two-variable function  $f(x,y)$  is defined as

$$\Delta f = \nabla^2 f(x,y) = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}. \quad (11)$$

On the other hand, there is a filter called a Laplacian of Gaussian (LoG) which first smoothes the image with the Gaussian operator and then applies the Laplacian operator. The LoG edge detectors are isotropic so that its response is composed of an average across an edge and one along the edge. This means that adding some percentage of this response back to the original image yields a picture in which edges have been sharpened and details are easier to see.

### 2.5.3. Bilateral Filtering

Toward an improved stereo vision, a filtering that does not blur across range discontinuities like edge and compensates photometric variations between a pair of cameras takes an important position. The method aims at preserving the signal details while removing the noise. This is achieved by locally adaptive recovery methods such as Anisotropic Diffusion (AD), Weighted Least Squares (WLS), or Robust Estimation (RE). However, since all these methods have an iterative property, there has been no low-cost mechanism for smoothing in homogeneous regions while sharply preserving

edges. Complex schemes to extract this information would conflict with the real-time requirement. In 1998, Tomasi and Manduchi [17] proposed an alternative non-iterative bilateral filter. As an edge-preserving smoother, this is a weighted average of local neighborhood samples, where the weights are computed based on spatial and intensity differences between the central pixel and its neighbors. In [18], Ansar suggested a novel approach incorporating a bilateral filter into a background subtraction step, and verified that it is superior to conventional methods such as Laplacian filtering, Difference of Gaussians filtering (DoG), Rank and Census Filtering. We adopt the bilateral filter as another pre-processing stage, and the following describes its algorithm.

Typical background subtraction method is composed of a low-pass filtering and a subtraction of the low-pass filtered image from an original image as follows;

$$I' = I - I * G(\sigma_{large}) \quad (12)$$

where  $I$  is an original image,  $G(\sigma_{large})$  is a Gaussian mask with a variance  $\sigma_{large}$ . In Ansar's approach, the low-pass filtering with a Gaussian mask is replaced with a bilateral filtering. The filter returns  $B(x)$  at  $x$  in  $I$  as follows;

$$B(x) = \frac{\sum_{\xi \in \Omega} I(\xi) c(\xi, x) s(I(\xi), I(x)) d\xi}{\sum_{\xi \in \Omega} c(\xi, x) s(I(\xi), I(x)) d\xi} \quad (13)$$

where  $\Omega$  is the filter support and  $I(x)$  is the intensity at  $x$ .  $c(\xi, x)$  measures the geometric closeness between a center  $x$  and a nearby point  $\xi$ , while  $s(I(\xi), I(x))$  measures the photometric similarity between them. The weight functions  $c(\xi, x)$  and  $s(I(\xi), I(x))$  are typically Gaussian distributions defined as

$$c(\xi, x) = \exp\left(-\frac{1}{2} \left(\frac{|\xi - x|}{\sigma_c}\right)^2\right) \quad (14)$$

$$s(I(\xi), I(x)) = \exp\left(-\frac{1}{2} \left(\frac{|I(\xi) - I(x)|}{\sigma_s}\right)^2\right) \quad (15)$$

where  $\sigma_c$  and  $\sigma_s$  are the standard deviations of spatial component and intensity component, respectively. The geometric spread  $\sigma_c$  is chosen based on the desired amount of low-pass filtering. A large  $\sigma_c$  blurs more, that is, it combines values from

more distant image locations. Similarly, the photometric spread  $\sigma_s$  is set to achieve the desired amount of combination of pixel values. Note that the closer the Euclidean distance between the central pixel and its neighbor is, the larger  $c(\xi, x)$  will be, and the smaller the intensity difference between them is, the larger  $s(I(\xi), I(x))$  will be as well. Thus, the filtered output is finally obtained as

$$I' = I - B . \quad (16)$$

#### 2.5.4. Control of Bilateral Filter Offset by SFR

When input imagers take pictures, they are affected by background illumination. The effect can propagate to disparity maps, resulting in increase of erroneous matches or unmatches. Thus, we integrated the function able to alleviate the effect of variant surrounding illumination by register controls. Located between the bilateral background subtraction in the equation (16) and the output of bilateral filter, one SFR can add or subtract light intensity on the output image.

## 2.6. Stereo Matching and 3-D Reconstruction

Combining the images produced by a pair of camera and exploiting the disparity between them provides us depth information. Stereo vision involves two processes: the correspondence of features observed by two cameras and the reconstruction of their 3-D information.

### 2.6.1. Ambiguity in Correspondence Problem

Figure 2.6 shows the ambiguity in correspondence problem of stereopsis. There are three points on the epipolar line of each image. The nine points in 3-D are possible as the combination of corresponding points, and **A** to **I** denote their positions. Considered all possible correspondences for each point on left image, only three points on the epipolar line of right image can be candidates. However, if each picture consists of hundreds of thousands of pixels with lots of image features, the number of possible combination will be drastically increased. Consequently, some techniques are necessary

in order to establish the correct correspondences and avoid erroneous depth measurements. In principle, three candidates to each point can be answers, but essentially human's eyes see one candidate only. This is because the disparity of matches varies smoothly almost everywhere in the image [3].

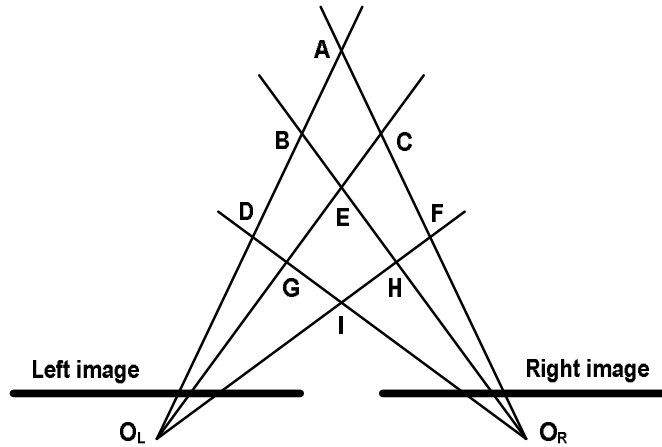


Figure 2.6: Ambiguity in Correspondence Problem

### 2.6.2. Stereo Matching by Correlation

It is called the stereo matching to establish correspondences between a pair of images. In the area-based matching algorithms the correspondence problem is solved by matching image intensity patterns, and this is also called the correlation technique. It finds pixel-wise image correspondences by comparing intensity profiles in the neighborhood of potential matches. Our stereo matching is based on a correlation method known as sum of absolute differences (SAD). Figure 2.7 depicts how to find the corresponding pixel in the SAD approach. Suppose that a pair of images is rectified by rectification process. The baseline  $T$  denotes a line connecting centers of two lenses, and the focal length  $f$  means a distance from a center of lens to image plane. One point  $P$  on a 3-D object is projected onto two image planes by the perspective projection as indicated by  $A$  and  $B$ , respectively. Then,  $A$  and  $B$  are called the corresponding pixel to each other. Once two images are rectified, each point in one image must be observed in the other image along the epipolar line by the epipolar constraint.

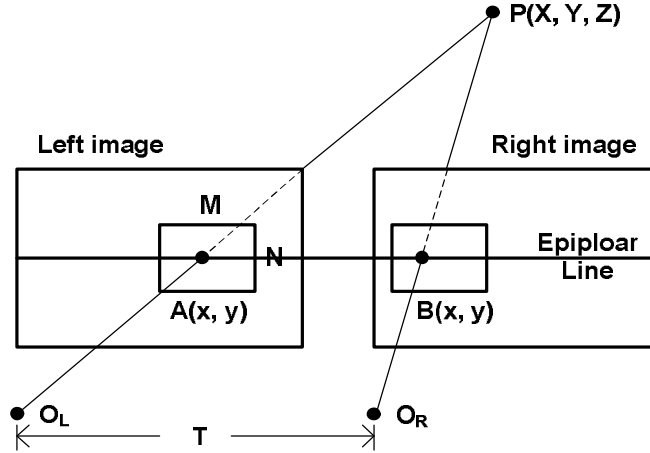


Figure 2.7: Finding Corresponding Pixels in SAD Algorithm

Consider windows of  $M$  by  $N$  size on each image as shown in Figure 2.7. When the disparity  $d$  denotes a distance between the centers of two windows, the SAD algorithm is defined as

$$SAD(x, y, d) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} |I_L(x+u, y+v) - I_R(x+u+d, y+v)| \quad (17)$$

where  $I_L$  and  $I_R$  are intensities in left and right image, respectively. In order to find out the corresponding point of  $A$  from the right image, the SAD between two windows is calculated with the manner that the right window moves from zero to the maximum disparity level  $d_{\max}$ , while the left window is fixed at the position centered at point  $A$ . Corresponding points are determined on the disparity exposing the smallest SAD. If the SAD computation shows zero value, it means the exact matching of two windows, that is, full correlation.

### 2.6.3. 3-D Reconstruction

Once the correspondence between images is established, we can calculate the depth of 3-D point using both the disparity measured by SAD and the triangulation  $Z = Tf(1/d)$  as presented in Figure 2.8. The 3-D scene is now reconstructed by assigning light intensities proportional to disparity and then collecting them. Here, a zero disparity indicates the depth of infinity.



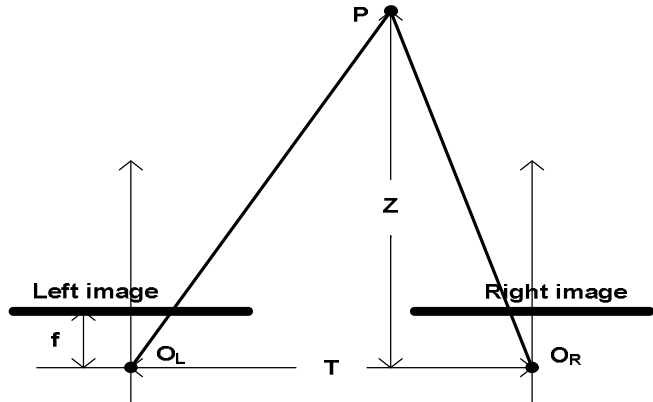


Figure 2.8: Relation between a 3-D point and 2-D images

## 2.7. Occlusion Region and Handling It

Since stereo images are taken from a different position, there is an area to be visible to one camera only. This is because protruded objects hide objects in the background. It is called the occlusion region and leads to erroneous or unmatched pixels. Figure 2.9 illustrates the situation that one point **B** is visible to both cameras but the other **A** to left camera only. The point **A** is called half-occluded because it is occluded in one of the views and not the other. This problem can be detected by left-right (LR) consistency check and be alleviated by median filtering. The methods were adopted in our design as post-processing to improve the accuracy of disparity map.

### 2.7.1. LR Consistency Check

The LR consistency check is also referred to as bidirectional matching, in which stereo matching is performed twice from left to right (LR) and from right to left (RL). Matches are confirmed only when LR matching is consistent with RL matching. The technique has been proven to be effective in discarding the erroneous matches by area-based algorithms in presence of occlusions [23]. However, this approach is characterized by a significant computational cost due to two matching process.

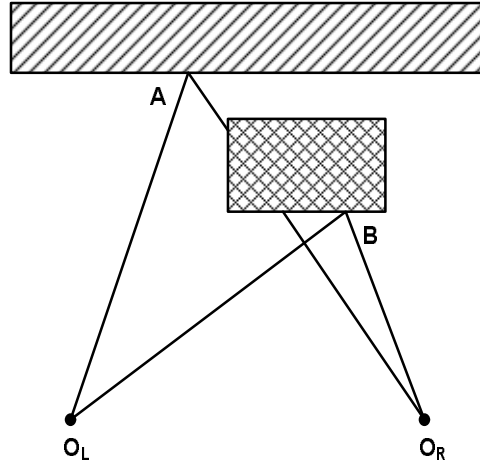


Figure 2.9: Occlusion Region

### 2.7.2. Control of LR Consistency Check by SFR

In LR consistency check, LR SAD can differ from RL SAD, especially, in the occlusion regions. Therefore, it is important in the accuracy of depth map to decide how much difference is acceptable. Our system parameterized the difference by SFR. When the difference between two SAD computations falls into a threshold, the pixels are finally recognized as matching points. High threshold increases the number of candidates of matches while the accuracy of disparity map decreases. On the contrary, low threshold decreases the number of candidates and even some points can be left as unmatched points. Consequently, the vision processor with the parameterized threshold will be able to control the quality of depth map. Our design assigned the threshold as SFR so that we can write the value outside the processor.

### 2.7.3. Median Filtering

The median filtering is commonly used to diminish outliers in disparity maps, which are often caused by occlusion regions [24], and to eliminate erroneous matches. It selects a medium value within a filter mask. Finding the medium requires the sorting of pixels according to intensity, thus the median filter belongs to the rank value filter. Figure 2.10 shows how the filter with the window size of 3\*3 works. In order to filter one pixel, 9 pixels including its neighbors are sorted, and then the medium is chosen as

the filtered output. Intuitively, we can know that the performance of this filter depends on the window size and a sorting algorithm.

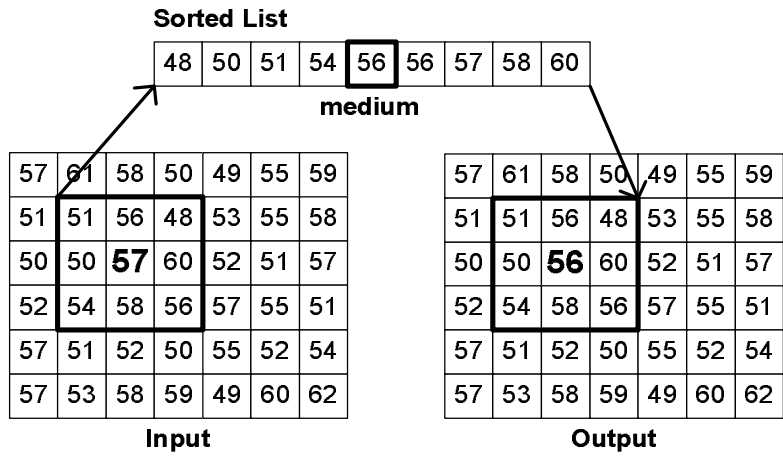


Figure 2.10: 3 by 3 Median Filter

## Chapter 3: Architecture of Proposed Stereo Vision Processor

### 3.1. System Prototype

Figure 3.1 shows the prototype of our stereo vision system. This consists of camera module, Xilinx FPGA-based board, external interface, and power system. The camera module is composed of two CMOS Image Sensor (CIS) cameras generating the color image of VGA (640\*480) size at the speed of 30 frames per second. The camera's lens has focal length of 6 mm, and the baseline distance between two cameras is 9 cm. The Xilinx FPGA board emulates not only stereo vision processor to integrate image rectification, bilateral filtering, depth estimator, and post-processing units, but also CIS interface. The external interface allows us to communicate with personal computers through IEEE 1394 protocol.

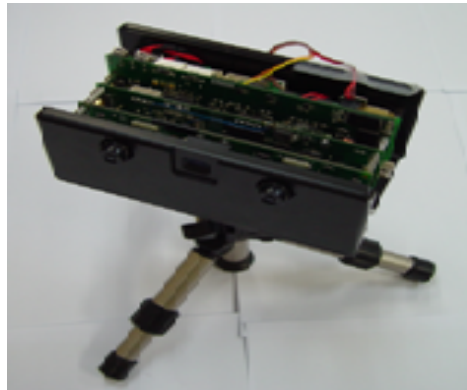


Figure 3.1: Prototype of Our Stereo Vision System

### 3.2. Top level Architecture

Figure 3.2 presents the top level hardware architecture of our stereo vision processor. Image Rectifier, Bilateral Filter, Depth Estimator, Median Filter, and SFRs are integrated on a single processor. With the 0.18 $\mu$ m CMOS technology, the proposed stereo vision processor can operate at 120 MHz clock and achieve 144 frames/sec depth maps with 320 by 240 image size and 64 disparity levels. A pair of CIS camera producing the color image of VGA resolution (640 by 480) consists of a stereo camera.

CIS Interface first synchronizes data and timing signals coming from two CIS cameras, and downsizes image data to 320 by 240. This block makes the processor independent on the kind of input imagers by providing a consistent image size as well as consistent frame, line synchronization signals. In Image Rectifier, images delivered from CIS Interface are rectified using the homography matrices. Bilateral Filter produces smooth and edge-preserving images from the rectified outputs. In Depth Estimator, bilateral filtered images are processed to extract disparity maps with 3-D information. Median Filter diminishes outliers and erroneous matches in disparity maps. External Interface transfers dense depth maps with 8-bit disparity to the host or user's machine, and receives host's inputs in order to set SFRs.

In the design phase of the processor, we build models to imitate two CIS cameras and the host, and use them in the verification phases including a gate-level simulation. The CIS model exactly provides sequences of YCbCr color data as well as timing signals such as frame, line syncs and their blanking times. Moreover, it imitates the situation that two imagers generate images at different time each other, which is common. The host model imitates to capture the output image of each stage stored in FIFO memory of External Interface, and to deliver parameterizable values to SFRs.

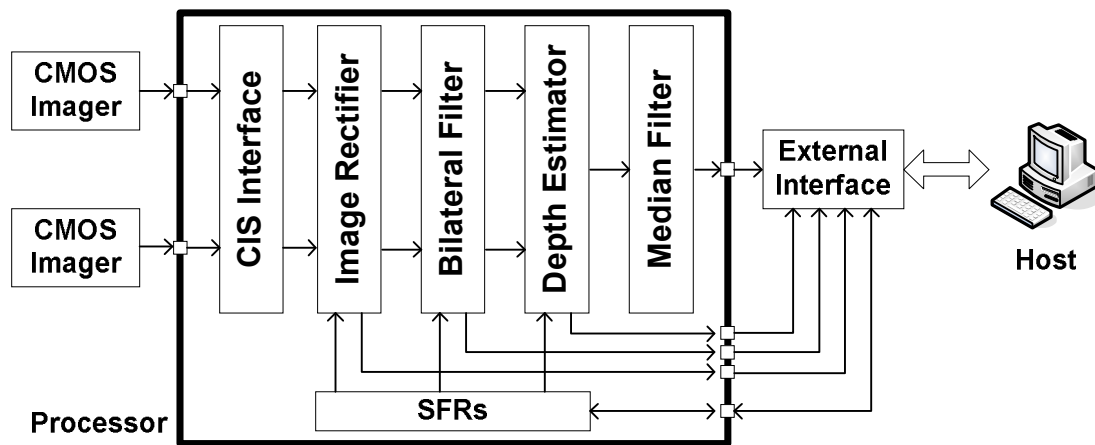


Figure 3.2: Top Level Architecture of Proposed Stereo Vision Processor

### 3.3. Image Rectifier

Figure 3.3 shows the hardware architecture of image rectifier. Two rectifiers are needed in stereo vision systems. RECT Control embeds the finite state machine to control the overall operations of rectification block. The controller generates control signals such as address, enable, byte select in order to store input stream into frame buffer, and provides the 2-D coordinate ( $xidx$ ,  $yidx$ ) of input pixel to RECT Core by counting the number of line sync and pixels within it. This block also detects a new frame and let Output Control know it by transferring a signal. Frame Buffer stores output images from CIS Interface, and sends data to RECT Core to be interpolated. Since our rectifier was designed to transform the color image and to store approximately half of input image to the buffer, the buffer size of 96Kbyte is required for each camera. RECT Core finds the position of pixel transformed by the homography, and produces its data. Output Control makes rectified pixel, frame and line sync signals synchronized, and delivers them to next stage.

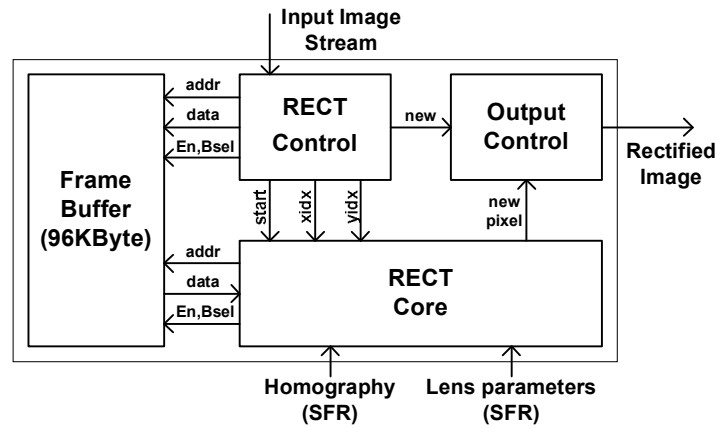


Figure 3.3: Hardware Architecture of Image Rectifier

#### 3.3.1. RECT Core Block

RECT Core performs main operations in image rectifier. The diagram shown in Figure 3.4 incorporates the pixel transformation, the compensation of radial distortion and the bilinear interpolation. Matrix Calculation presented in Figure 3.5 depicts how to obtain the position of transformed pixel by the homography  $\mathbf{H}$  given through SFR, and delivers it to Radial Distortion block that offsets radial distortion of lens using lens

parameters [20]. Note that  $xidx$ ,  $yidx$ ,  $x$ ,  $y$ ,  $rx$  and  $ry$  are the pixel positions in the camera coordinate system, not data. Essentially,  $rx$  and  $ry$  are outputs in RECT Core block, but they usually have floating point values so that the bilinear interpolation was employed in order to get fine data corresponding to its coordinate. Fetch Pixels finds addresses of 4 neighboring pixels with an integer position around the coordinate  $(rx, ry)$ , and fetches their data  $fa$ ,  $fb$ ,  $fc$  and  $fd$  from Frame Buffer. Finally, the rectified image comes out after interpolation based on 4 pixels in Bilinear Interpolation block. Remind that computations are mainly composed of finding address because the rectification is a mapping of pixel onto a new location.

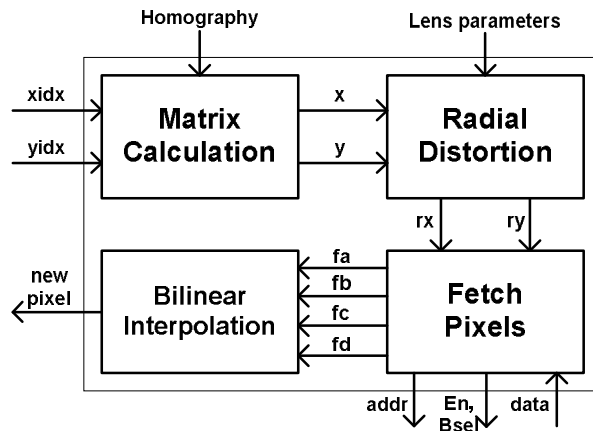


Figure 3.4: RECT Core Block in Image Rectifier

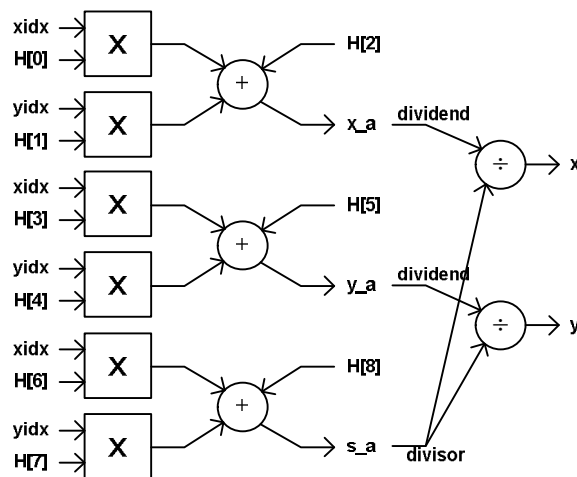


Figure 3.5: Matrix Calculation Block in RECT Core

### 3.4. Bilateral Filter

We adopted the bilateral filter with  $\sigma_c = 3.7$ ,  $\sigma_s = 30$  in equations (14) and (15), and the window size of 11\*11. Figure 3.6 illustrates how the bilateral filter works. In order to filter one pixel, 11\*11 pixels are required and they are stored at vertical buffers. The vertical buffers were implemented by shift registers to discard the oldest one pixel and simultaneously to fill one new pixel every clock cycle. Weight Calculation blocks find weights depending on both spatial and intensity difference between the central pixel and its neighbors. The spatial and intensity difference values are stored at tables with 121 levels and 256 levels, respectively. Divider performs the division in equation (13), taking the summation of weights over intensity difference as a numerator and that over spatial difference as a denominator. In ABS block, absolute difference between the weighted average and the original image pixel is computed, that is,  $I' = I - B$  in equation (16). This logic also has the SFR to alleviate the effect of variant surrounding illumination on disparity maps.

#### 3.4.1. Weight Calculation Block

Our bilateral filter has 121 weight calculation blocks since the window size is 11 by 11. As shown in Figure 3.7, SUB and ABS blocks calculate the spatial difference between a central pixel (M061) and its local point (Mxxx). The difference is used as index to look-up the table of EXP\_CAM containing values of the weight function  $c(\xi, x)$ . The input EXP\_D denotes the value of the weight function  $s(I(\xi), I(x))$  which was already computed according to intensity difference between two pixels. Thus, the first multiplier performs multiplication of  $c(\xi, x)$  and  $s(I(\xi), I(x))$ , and its result becomes the WEIGHT. The second multiplier produces the CP\_WEIGHT, the output of multiplication of  $I(\xi)$  and  $c(\xi, x)s(I(\xi), I(x))$ . index to look-up the table of EXP\_CAM containing values of the weight function  $c(\xi, x)$ . The input EXP\_D denotes the value of the weight function  $s(I(\xi), I(x))$  which was already computed according to intensity difference between two pixels. Thus, the upper multiplier performs multiplication of  $c(\xi, x)$  and  $s(I(\xi), I(x))$ , and its result becomes the WEIGHT. The lower multiplier produces the CP\_WEIGHT, the output of multiplication of  $I(\xi)$  and  $c(\xi, x)s(I(\xi), I(x))$ .



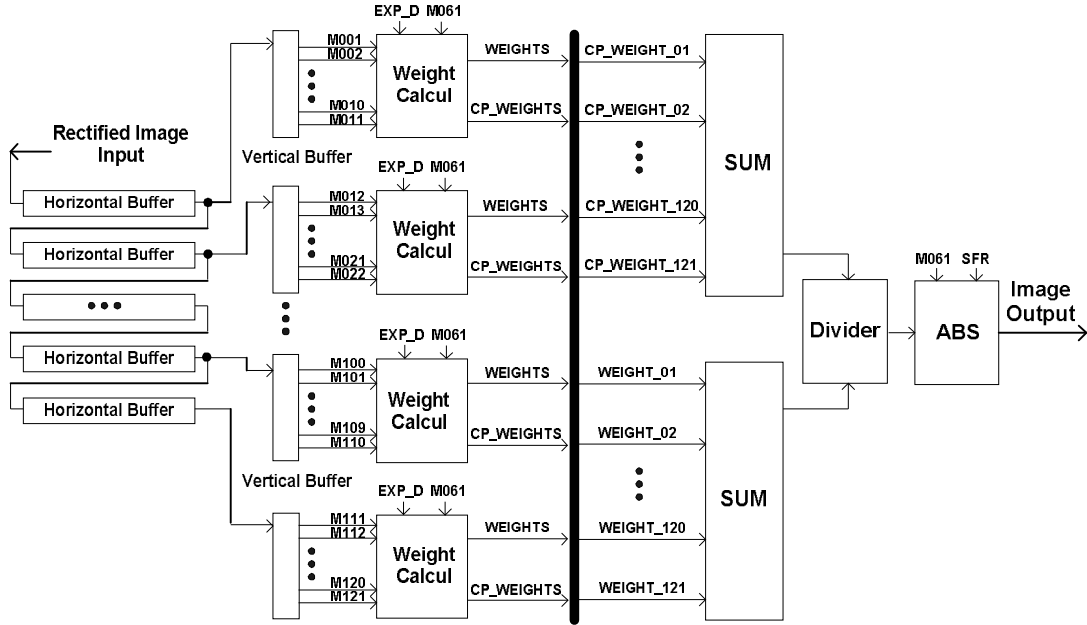


Figure 3.6: Hardware Architecture of Bilateral Filter

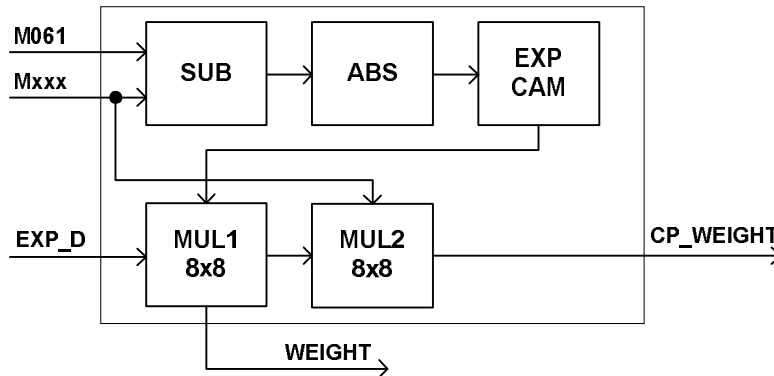


Figure 3.7: Weight Calculation Block in Bilateral Filter

### 3.4.2. SUM Block

The SUM block illustrated in Figure 3.8 performs the summation of weight functions in equation (13). The operation is very simple because it is composed of additions only. However, note that the block has the large number of inputs. This requires the high fan-in of a digital logic gate so that the speed of the gate can be degraded. The rising (or falling) time of two identical transistors connected in series will be approximately double that for a single transistor with the same capacitive load.

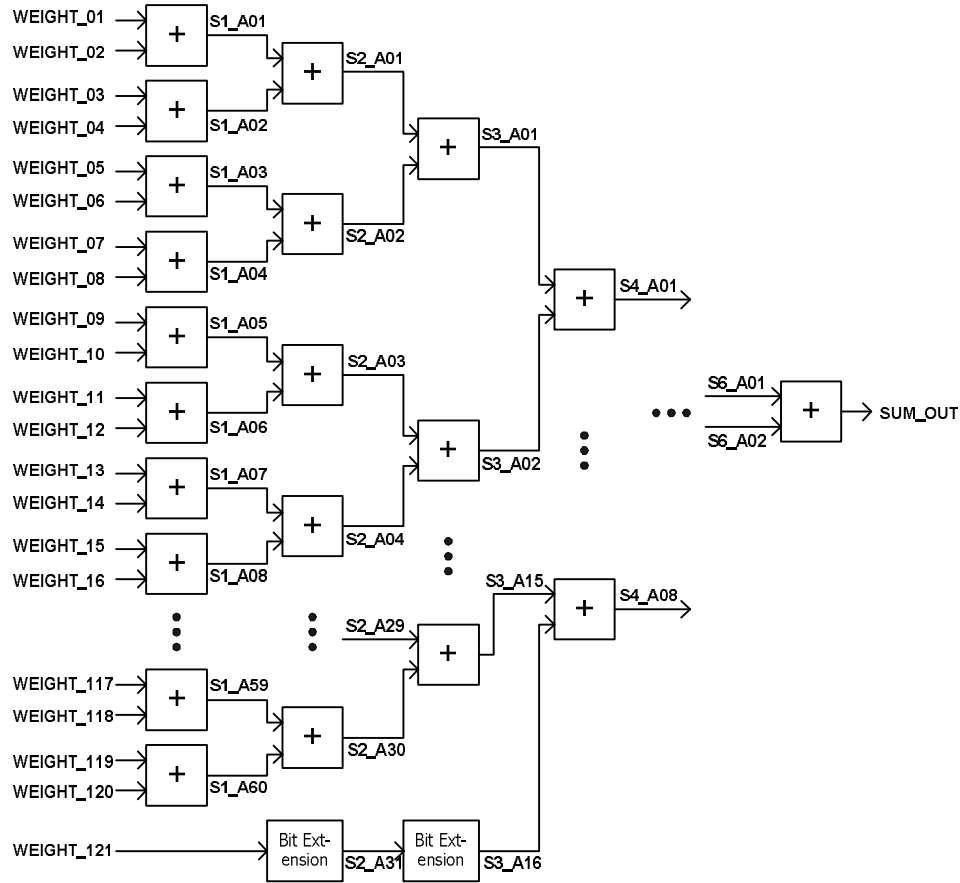


Figure 3.8: SUM Block in Bilateral Filter

In order to achieve the best speed-performance we restrained the number of inputs to two as the figure.

### 3.5. Depth Estimator and LR Consistency Check

The depth estimator performs stereo matching to establish the correspondences between a pair of image. To find matching points efficiently and effectively, it is necessary for input images to be rectified and filtered, respectively, before matching. So we call those steps as pre-processing. Also to enhance the accuracy of depth maps and to avoid erroneous matches, several techniques such as LR consistency check and median filtering are employed after matching, called post-processing. The depth

estimator integrated in our processor has the structure presented in Figure 3.9. Its input is rectified and filtered image, and output is the disparity between a pair of image. Left and Right Buffer store scan-line data of image and dispatch them to SAD blocks synchronizing to the control signals of hard-wired Controller. The Controller not only

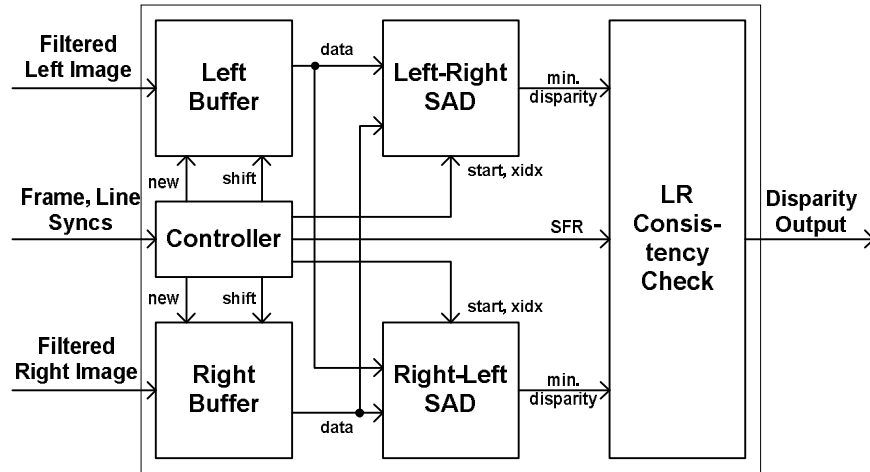


Figure 3.9: Hardware Architecture of Depth Estimator

determines output timing of two Buffers and two SAD blocks, but delivers the value of special function register to LR Consistency Check block. In LR SAD block, a window with the size of  $11 \times 11$  around candidate pixel is taken at each image along epipolar lines, and then the SAD is computed with the manner of left fixed window and right moving window by 64 disparity levels. RL SAD block also computes the disparity, however, with the opposite manner of left moving window and right fixed window. In LR Consistency Check block, the LR SAD value is compared with the RL SAD one. Only when the difference falls into a threshold, the matching is successful. Otherwise, the result is discarded regarding as unmatched point. Consequently, the threshold affects the quality of depth map so that it should be parameterized if the LR consistency check is employed. In our processor the parameter is controlled by SFR.

### 3.5.1. Pipelined and Parallelized Implementation of SAD Algorithm

Figure 3.10 shows SAD computation based on 5 by 5 window sizes. The illustration implies that the SAD algorithm can be implemented with the techniques of

pipelining and parallelism. Let us suppose that here is only one processing element (PE) to calculate absolute difference between two pixels and the element takes one clock cycle to perform the pixel operation like Figure 3.10 (a). It takes five cycles to compute the first line in the window. If we configure one line-based processing element (LPE) with five PEs, the time cost will be decreased to one cycle for the line computation as drawn in Figure 3.10 (b), by exploiting parallelism based on more hardware resources. The LPE can process another line in one cycle by pipelining so that five cycles are taken for the window computation. Now consider an SAD computation with disparity ranging from 0 to 63 like Figure 3.10 (c). For one pixel matching, this requires 64 times as much computation as Figure 3.10 (b). Besides, note that image's line data come into the depth estimator sequentially even though they seem to get together in the window at one moment. In other words, the second line is actually delivered later than the first one to the estimator.

Due to this constraint and much computation we devised the pipelined and parallelized method using LPEs and its hardware architecture as shown in Figure 3.11 and Figure 3.12, respectively. The characteristic of this technique is that each LPE was assigned to process each line in left and right windows. The LPE1 processes the first lines only in two windows, and the LPE2 does the second lines and so on. This approach enables the SAD value to be computed every cycle after filling out pipeline. Figure 3.11 depicts first 3 cycles finding absolute differences between the two windows in left-right SAD computation. Our depth estimator has the window size of 11 by 11, but 5 by 5 was drawn for convenience. At cycle 1, LPE1 only operates for disparity level of zero. At cycle 2, LPE1 calculates absolute differences at disparity level of 1, while LPE2 does at disparity level of zero and the others don't work. Following this manner, the SAD computation at one disparity level, (for example,  $d=0$ ), take 11 cycles in our system. We know that latency between LPE1 and LPE11 operations is 10 cycles. Consequently, the depth estimator with 64 disparity levels consumes total 74 cycles to match one pixel, that is,  $(64+11-1)$  cycles. Due to pipelining, however, 73 cycles will be hidden, and the processor actually produces a disparity output every cycle.

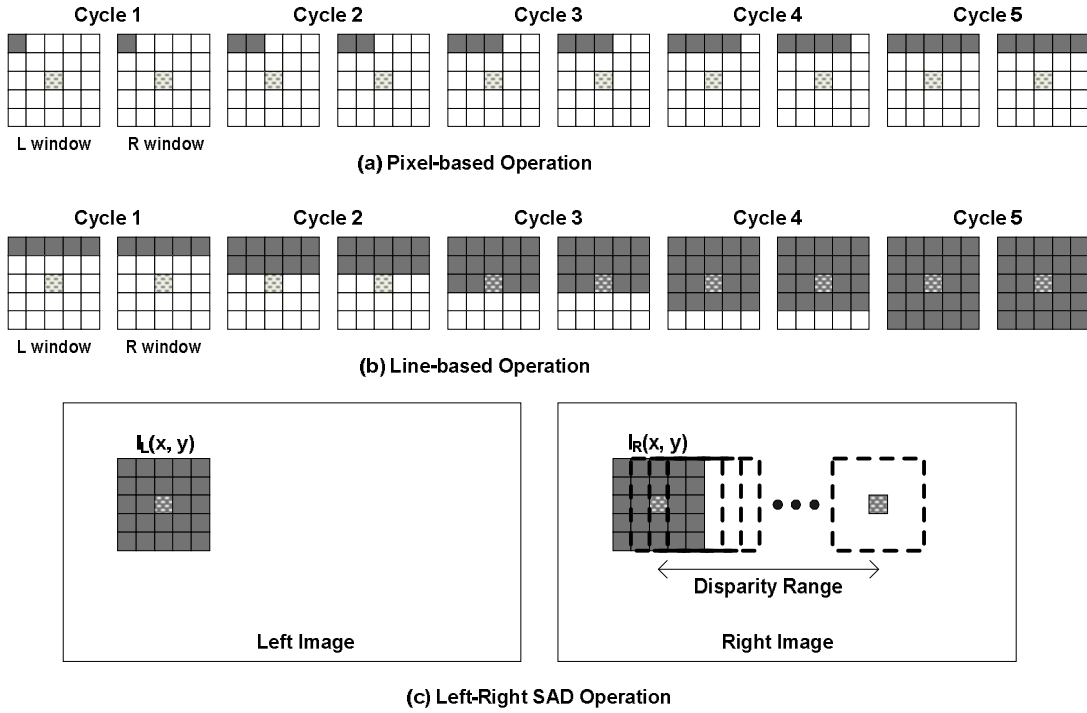


Figure 3.10: SAD Computation based on 5 by 5 Windows

		LPE1	LPE2	LPE3
Cycle 1		 d=0	No Operation	No Operation
Cycle 2		 d=1	 d=0	No Operation
Cycle 3		 d=2	 d=1	 d=0
	Left Image	Right Image		

Figure 3.11: Pipelined and Parallelized Operations of Line-based Processing Elements in Left-Right SAD Computation

### 3.5.2. Left-Right SAD Block

Figure 3.12 illustrates the structure of LR SAD block based on the concept in previous subsection. The hardware integrates total 11 LPEs, and each LPE embeds total 11 pixel-based processing elements (ABS) to find the intensity difference between two pixels. Since left image is reference in this block, left image data remain as same during the SAD operation of 64 disparity levels, while right image data come from Right Buffer every cycle. This manner is opposite in RL SAD block. The Disparity Range is dispatched by control logics to comparator, synchronized to output of shift registers and summation block. The values of absolute difference reaching SUM block at certain time are the computation results for same disparity level. The COMPAR block selects the minimum SAD among 64 candidates and sends its disparity to the LR Consistency Check block.

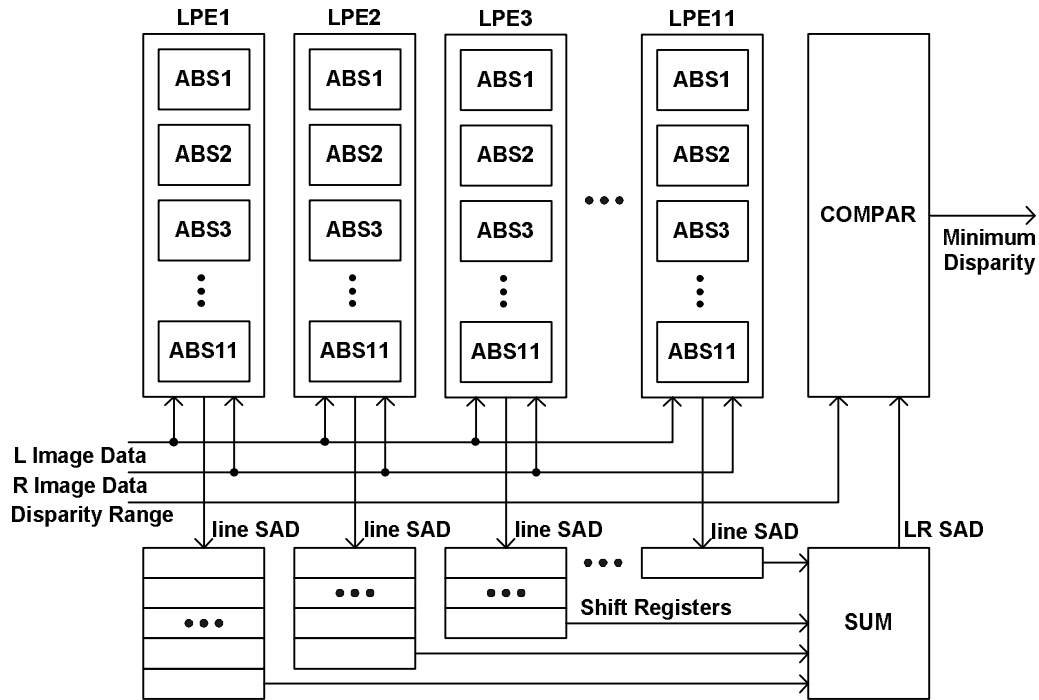


Figure 3.12: Left-Right SAD Block in Depth Estimator

### 3.6. Median Filter

We adopted the median filter with the window size of 3\*3 as another post-processing unit. Figure 3.13 shows the hardware architecture of median filter similar to

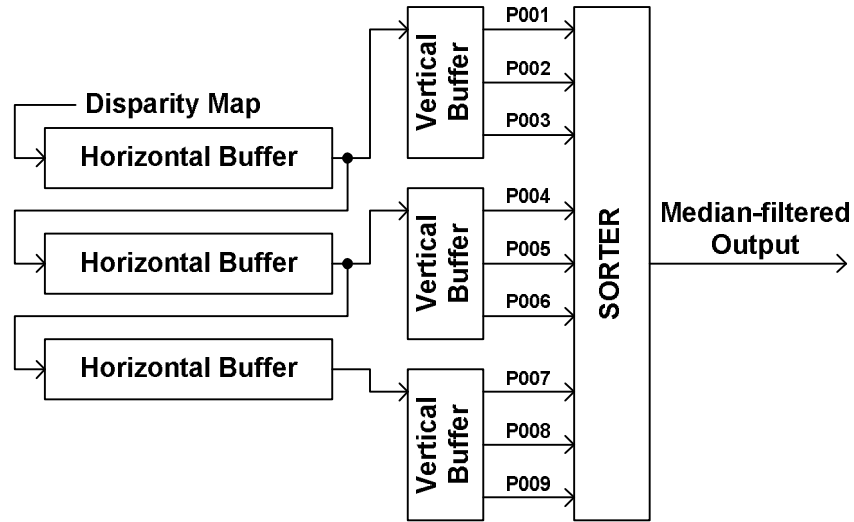


Figure 3.13: Hardware Architecture of Median Filter

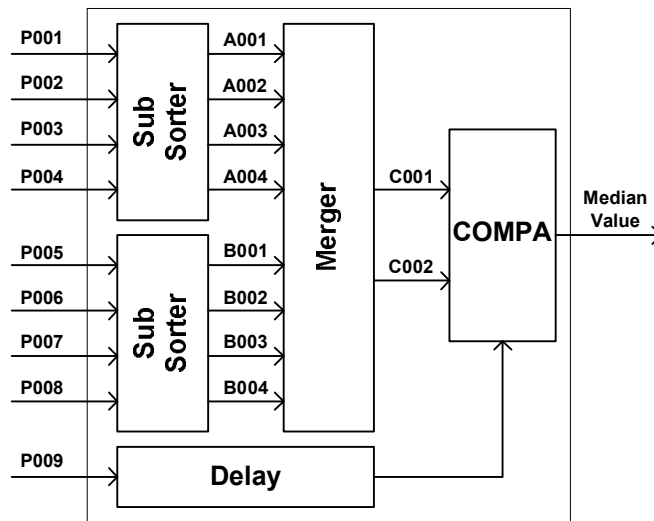


Figure 3.14: SORTER Block in Median Filter

the bilateral filter. In order to filter one pixel,  $3 \times 3$  pixels are required and they are stored at vertical buffers. The vertical buffers were implemented by shift registers to discard the oldest one pixel and simultaneously to fill the new one every clock cycle. The 9 pixels are given to the SORTER block that sorts the pixels by decreasing order, and then selects the medium as the filter output. Figure 3.14 presents how the SORTER block was implemented. Nine pixels conveyed from vertical buffers are divided into three groups. The first two groups are sorted with decreasing order at Sub-Sorter,

respectively. Merger block combines them with decreasing order, and extracts two middle values to send into COMPA logic. For data synchronization, the third one is delayed as much time as the computation of Sub-Sorter and Merger takes. It is 6 clock cycles. In COMPA block, *P009* is compared with *C001* and *C002*. The logic chooses the median value among them as the output of median filter.



## Chapter 4: Performance Evaluation and Comparison with Previous Works

In this chapter we evaluate the performance evaluation of the proposed processor by comparison with previous hardware works. The test images are Teddy obtained from Middlebury stereo data sets [26], and pictures taken at real world. [26] provides valuable test images able to apply to stereo vision systems so that many researchers associated with computer vision have used them as reference images. However, since they are very clear and rectified images, stereo vision systems applied to real world should be verified with different images taken at real world as we present. The evaluation results according to different inputs include output images of each block in processor and output images controlled by SFRs. We also confirm the functionality of bilateral filter and median filter.

### 4.1. Experiments based on Middlebury Stereo Data and Its Analysis

Figure 4.1 shows input image Teddy, and output image in each stage of our processor. Figure 4.1 (a) and (b), Middlebury color images, are converted to gray scale images with the size of 320 by 240 (QVGA) at the front part of bilateral filter. Remind that the rectification process is not required since [26] provides rectified images. Figure 4.1 (c) and (d) are outputs of bilateral filter to operate as an edge-preserving smoother. It is worth observing that the first five horizontal lines don't produce meaningful values because we employed the window size of 11\*11 in this filter. From the images we can confirm that the bilateral filter smoothes homogeneous regions while sharply preserves discontinuities like edges. Figure 4.1 (e) shows the depth map, output of depth estimator, where closer objects are represented with higher intensity. It is verified that the LR consistency check in area-based stereo algorithms is effective in detecting and discarding erroneous matches at occlusion regions and around edges, although there are a few erroneous ones like pepper noise. Median filter can mitigate the effect of

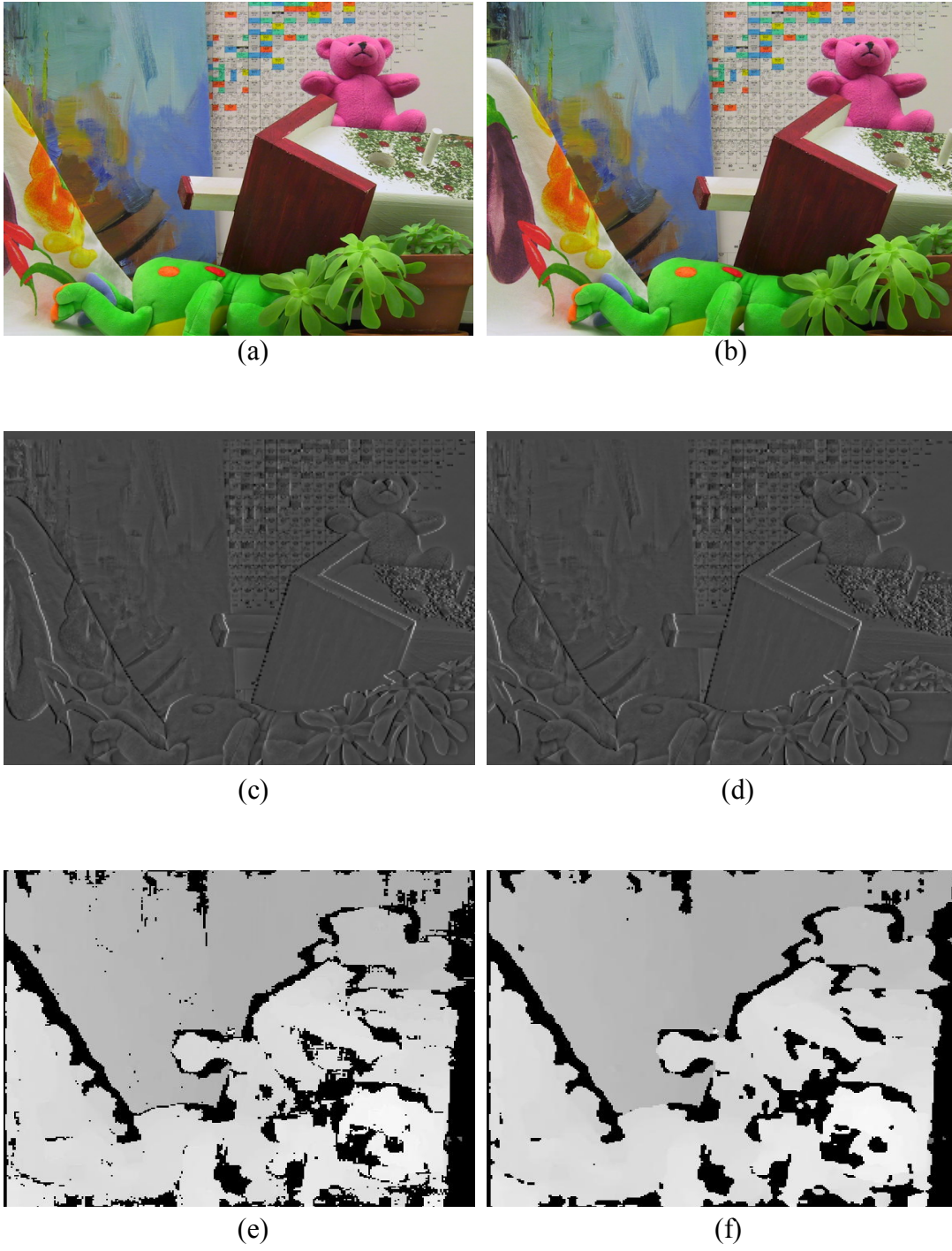


Figure 4.1: Experimental Results using Middlebury images in Designed Processor: (a) left input (b) right input (c) left bilateral filtering output (d) right bilateral filtering output (e) depth estimator output (f) median filtering output

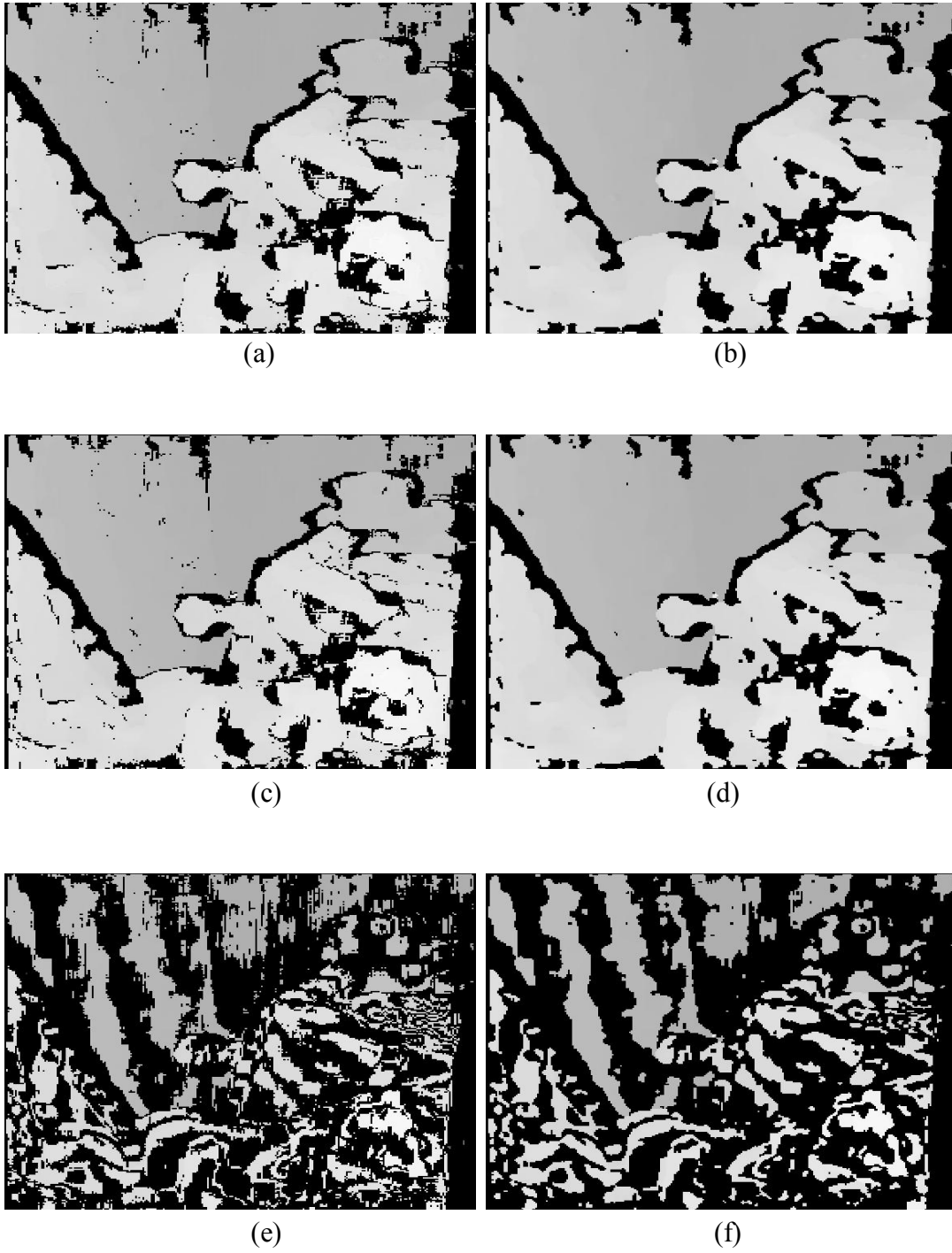


Figure 4.2: Results of SFR Control in Depth Estimator:

(a) high threshold (b) its median filtering output (c) middle threshold (d) its median filtering output (e) low threshold (f) its median filtering output

occlusion regions and erroneous matches by interpolating neighboring disparities [24], and Figure 4.1 (f) presents its result applied to the depth map.

Figure 4.2 illustrates how disparity map varies as we control one of SFRs in the processor. The value is a threshold to decide how close the LR SAD is to the RL SAD in LR consistency checker. High threshold implies that the disparity difference between them is large, and thus leads to many candidate matches. However, if the value is so large, the accuracy of depth map will be decreased. Figure 4.2 (a) has a threshold of 3, which means that if the disparity difference is smaller than 3, the match is finally confirmed as valid one. The value in Figure 4.2 (c) is 2, and Figure 4.2 (e) has 1. From the images we can know that as the threshold decreases, unmatched points increases. It is worth noticing that Figure 4.2 (e) is a result confirming matches only when there is no disparity difference between two SAD computations. Figure 4.2 (b), (d) and (f) present outputs of 3\*3 median filtering for each threshold case. Although Figure 4.2 (a) and (c) have significant difference in the accuracy, the median filter compensates them well so that it is not easy to find out distinctive difference between Figure 4.2 (b) and (d).

## 4.2. Experiments based on Real Scene and Its Analysis

In this section we verify the designed hardware with test scenes obtained at real world. The images were taken by two low cost CMOS Image Sensors (CIS) with VGA resolution. As shown in Figure 4.3 (a) and (b), unrectified and unfiltered stereo images are used as test inputs. Note that the left image is a little obscure due to photometric or geometric variations of left camera, which can lead to poor disparity map. Figure 4.3 (c) and (d) are color-rectified ones. We find that the bilateral filter operates as an edge-preserving smoother in Figure 4.3 (e) and (f). Figure 4.3 (g) is the depth map extracted from filtered images, and closer objects have more intensity in our system. Unmatched pixels around background objects came mainly from the obscurity of left input image generated by unfocused imaging. This blurs object's features like edge, and weakens object's intensity making it similar with neighbors. The obscurity propagates to the left

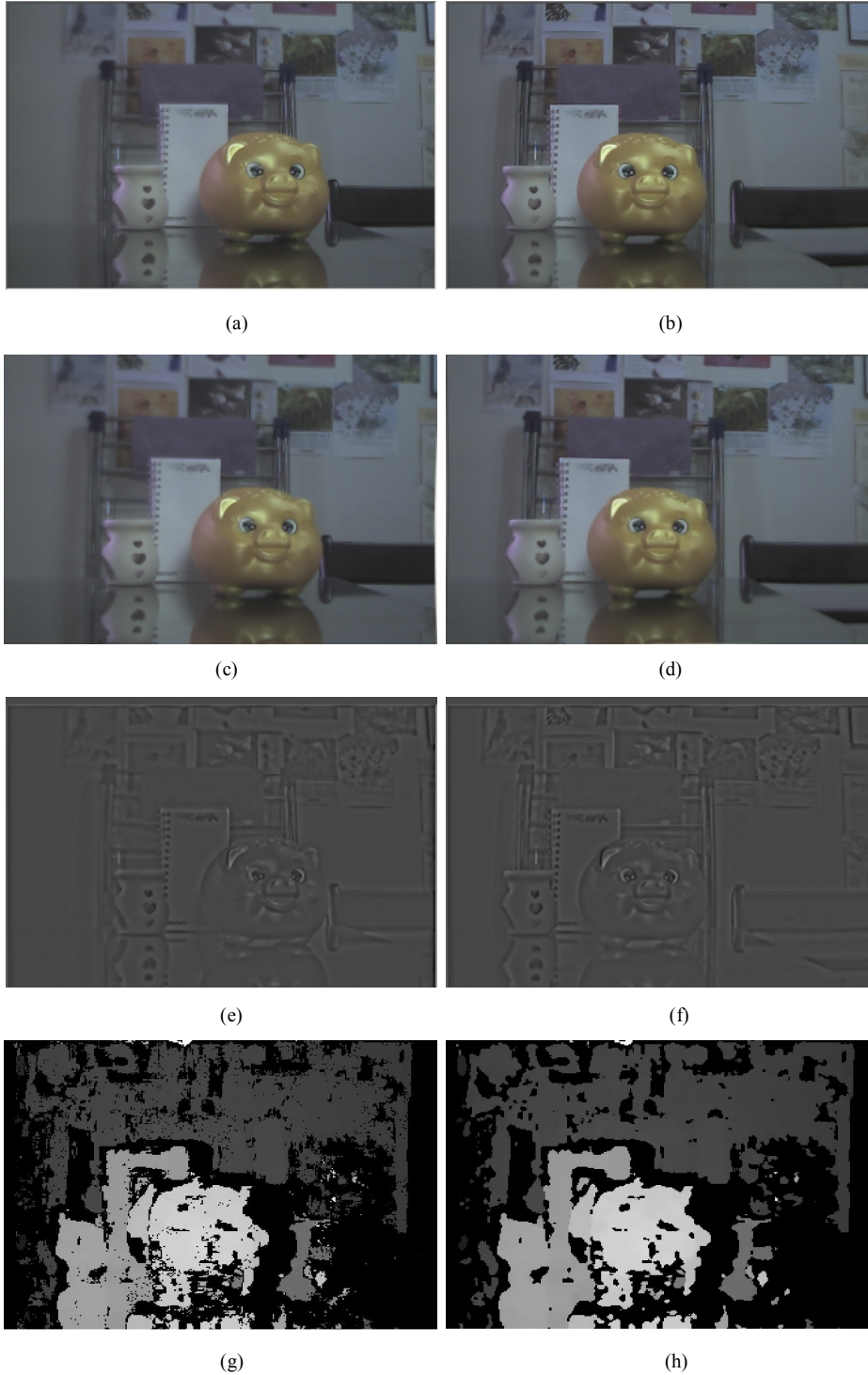


Figure 4.3: Test Results of Each Block in Designed Processor:  
 (a) left input (b) right input (c) left rectification output (d) right rectification output (e)  
 left bilateral filtering output (f) right bilateral filtering output  
 (g) depth estimator output (h) median filtering output

bilateral-filtered image as shown in Figure 4.3 (e), and causes incorrect matches or mismatches in depth estimator. Figure 4.3 (h) is a median-filtered output of the disparity map.

### 4.3. Hardware Specifications

The stereo vision processor described in this paper generates 320 by 240 depth images with 64 disparity levels. At the 120MHz operating clock frequency, dense disparity maps can be produced at the rate of 144 frames per second. The design integrates an image rectifier, a bilateral filter, a depth estimator with the LR consistency check, and a median filter using 0.18um CMOS technology. The hardware specification of the design including SFRs is summarized in TABLE I. At the condition of 60MHz system clock and 0.18um CMOS technology, the total gate count, memory usage and dynamic power consumption are estimated to 1.5 million, 235KBytes and 1.48W, respectively. As indicated in the table, two rectifiers of left and right color images take almost amount of memory embedded in the processor, since their memory were implemented able to accommodate approximately half of input images. Besides, their power consumption takes the largest portion in the processor, even though the logic count of depth estimator is more than that of image rectifiers. This is because the memory access in image rectifiers is more frequent than in depth estimator.

Table I: Hardware Specifications (0.18um CMOS Technology, 60MHz Clock)

	Gate Count	Memory Usage	Dynamic Power Consumption
CIS Interfaces	1.7K	8KB	2mW
Image Rectifiers	495K	194KB	680mW
Bilateral Filters	355K	7.8KB	290mW
Depth Estimator	661K	24KB	500mW
Median Filter	4K	1.4KB	3.37mW
<b>Total</b>	<b>1,516K</b>	<b>235.2KB</b>	<b>1,475mW</b>

#### 4.4. Comparison with Previous Hardware Works

In Table II, the performance of our system is compared with previous hardware works. It says that the proposed design shows better performance and integrates more hardware features enhancing the accuracy of disparity map. We apply one metric, Stereo Computation, to each system in order to compare the amount of computation used for stereo matching. It is obtained from the multiplication of the size of depth map output, the throughput, the window size and the disparity level. The metric is fairly reasonable in that the amount of stereo computation is proportional to all of them. Even though some of works presented in the table didn't clearly show the size of depth map, we calculate the amount assuming that the output size is same to the input size. Based on the result of multiplication ( $320 \times 240 \times 144 \times 11 \times 11 \times 64$ ), our processor has the better computational ability in stereo matching than any other work. (For the stereo computation of [25], we applied the disparity level of 20 from their actual experimental result.)

Table II: Comparison with Previous Hardware Implementations

	P. J. Burt [13]	M. Kuhn [14]	M. Hariyama [15]	M. Hariyama [27]	K. Ambrosch [8]	S. Longfield [25]	<b>Our Processor</b>
System Type	ASIC (0.25um)	ASIC (0.25um)	ASIC (0.18um)	ASIC (0.18um)	FPGA	FPGA	<b>ASIC (0.18um)</b>
Clock Freq.	100MHz	75MHz	125MHz	100MHz	65MHz	58MHz	<b>60MHz</b>
Input Image Size	512*480	256*192	320*240	32*32	320*240	320*240	<b>640*480</b>
Throughput (Max.)	(80GOPS)	50fps	10fps	1,000fps	425fps	300fps	<b>144fps</b>
Window Size	N/A	10*3	Variable: 3*3 to 5*5	4*4	3*3	Variable: 3*3 to 13*13	<b>11*11</b>
Disparity Level	N/A	25	N/A	N/A	100	Variable: 2 to 40	<b>64</b>
Stereo Computation*	N/A	1,843M	N/A	N/A	29,376M	77,875M	<b>85,642M</b>
Pre-processing	Filter	Without	Without	Without	Without	Without	<b>Rectifier, Bilat. Filter</b>
Stereo Algorithm	SAD	SSD & Census	SAD	SAD	SAD	Census	<b>SAD</b>
Post-processing	LR Check	LR Check, Med. Filter	Without	Without	Without	Without	<b>LR Check, Med. Filter</b>
Other Features	Motion Estimator	N/A	N/A	N/A	N/A	N/A	<b>SFRs</b>

## **Chapter 5: Conclusions**

We have presented a stereo vision processor extracting disparity maps with high-throughput and improved-quality. Since the system was designed considering uses in real applications, it integrates pre- and post-processing units such as rectifier, bilateral filter, LR consistency checker and median filter using 0.18um CMOS technology. The proposed processor has the flexibility to control the quality of depth maps according to real environments. This characteristic was achieved by assigning vision parameters to special function registers. The SAD algorithm as stereo matching technique was implemented on hardware exploiting pipelining and parallelism in order to achieve higher throughput. The proposed design is independent of the types of CIS camera or host computer, since we used typical models of them at the functional verification and the gate-level simulation. We evaluated the performance of the proposed stereo vision processor through experiments based on Middlebury data sets and images taken from real scenes.



## Bibliography

- [1] D. H. Hubel and T. N. Wiesel, "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex", *Journal of Physiology*, 160, pp. 106-154, 1962.
- [2] H. B. Barlow, C. Blackmore, J. D. Pettigrew, "The neural mechanism of binocular depth discrimination", *Journal of Physiology*, 193, pp. 327-42, 1967.
- [3] D. Marr and T. Poggio, "A Computational theory of Human Stereo Vision", *Proceedings of the Royal Society of London – Series B: Biological Sciences*, Vol. 204, pp. 301-328, 1979.
- [4] W. E. L. Grimson, "Computational Experiments with a Feature based Stereo Algorithm", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-7, 1985.
- [5] G. C. DeAngelis, I. Ohzawa, R. D. Freeman, "Depth is encoded in the visual cortex by a specialized receptive field structure", *Nature*, 11, 352(6331) pp. 156-159, 1991.
- [6] F. Solari, S. P. Sabatini, G. M. Bisio, "Fast technique for phase-based disparity estimation with no explicit calculation of phase", *IET Electronics Letters*, Vol. 37 (23), pp. 1382 -1383, 2001.
- [7] Sungchan Park, Hong Jeong, "High-speed parallel very large scale integration architecture for global stereo matching", *Journal of Electron Imaging*, Vol. 17, 2008
- [8] K. Ambrosch, M. Humenberger, W. Kubinger, A. Steininger, "Hardware implementation of an SAD based stereo vision algorithm", *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1-6, 2007.
- [9] J. Woodfill and B. V. Herzen, "Real-Time Stereo Vision on the PARTS Reconfigurable Computer", *IEEE Symposium on FPGA-Based Custom Computing Machines*, pp. 201, 1997.
- [10] A. Darabiha, J. Rose, W.J. MacLean, "Video-Rate Stereo Depth Measurement on Programmable Hardware", *IEEE Conference on Computer Vision and Pattern Recognition*, Vol. 1, June 2003.
- [11] D. K. Masrani, W. J. MacLean, "Expanding Disparity Range in an FPGA Stereo System While Keeping Resource Utilization Low", *IEEE Conference on Computer Vision and Pattern Recognition*, 2005.
- [12] J. Mikko Hakkarainen, Hae-Seung Lee, "A 40×40 CCD/CMOS absolute-value-of-difference processor for use in a stereo vision system", *IEEE Journal of Solid-State Circuits*, 1993.
- [13] P. J. Burt, "A pyramid-based front-end processor for dynamic vision applications", *Proceedings of the IEEE*, Vol. 90, pp. 1188-1200, 2002.
- [14] M. Kuhn et al., "Efficient ASIC Implementation of a Real-time Depth Mapping Stereo Vision System", *IEEE Midwest Symposium on Circuits and Systems*, Vol. 3, pp. 1478-1481, 2003.
- [15] Masanori Hariyama et al., "VLSI processor for reliable stereo matching based on window-parallel logic-in-memory architecture", *Symposium on VLSI Circuits*, pp. 166-169, 2004.
- [16] Ralf M. Philipp, Ralph Etienne-Cummings, "A 128×128 33mW 30frames/s Single-Chip Stereo Imager", *IEEE Solid-State Circuits Conference*, pp. 2050-2059, 2006.

- [17] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images", *IEEE International Conference on Computer Vision*, pp. 839-846, 1998.
- [18] Adnan Ansar, Andres Castano, Larry Matthies, "Enhanced Real-time Stereo Using Bilateral Filtering", *International Symposium on 3D Data Processing, Visualization, and Transmission*, pp. 455-462, 2004.
- [19] Zhengyou Zhang, "Determining the Epipolar Geometry and its Uncertainty: A Review", *International Journal of Computer Vision*, 1998.
- [20] J. Weng, P. Cohen, M. Herniou, "Camera calibration with distortion models and accuracy evaluation", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 14, pp. 965-980, 1992.
- [21] David A. Forsyth and Jean Ponce, "Computer Vision: A Modern Approach", Prentice Hall, 2002.
- [22] John Canny, "A computational approach to edge detection", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.: PAMI-8, pp. 679-698, 1986.
- [23] G. Egnal and R. P. Wildes, "Detecting binocular half-occlusions: empirical comparisons of five approaches", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 24, pp. 1127-1133, 2002.
- [24] M. Z. Brown, D. Burschka, G. D. Hager, "Advances in computational stereo", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 25, pp. 993-1008, 2003.
- [25] S. Longfield, Jr. and Mark L. Chang, "A Parameterized Stereo Vision Core for FPGAs", *IEEE Symposium Field Programmable Custom Computing Machines*, pp. 263-266, 2009.
- [26] <http://vision.middlebury.edu/stereo/data/>
- [27] M. Hariyama et al., "1000 frame/sec Stereo Matching VLSI Processor with Adaptive Window-Size Control", *IEEE Asian Solid-State Circuits Conference*, pp. 123-126, 2006.
- [28] Sang-Kyo Han et al., "Improved-Quality and Real-Time Stereo Vision Processor", *International Conference on VLSI Design*, pp. 287-292, 2009.