# A Real-Time Stereo Vision System with FPGA

Yosuke Miyajima and Tsutomu Maruyama

Institute of Engineering Mechanics and Systems, University of Tsukuba,
1-1-1 Ten-ou-dai Tsukuba Ibaraki 305-8573 Japan,
miyajima@darwin.esys.tsukuba.ac.jp

**Abstract.** In this paper, we describe a compact stereo vision system which consists of one off-the-shelf FPGA board with one FPGA. This system supports (1) camera calibration for easy use and for simplifying the circuit, and (2) left-right consistency check for reconstructing correct 3-D geometry from the images taken by the cameras. The performance of the system is limited by the calibration (which is, however, a must for practical use) because only one pixel data can be allowed to read in owing to the calibration. The performance is, however, 20 frame per second (when the size of images is $640 \times 480$, and 80 frames per second when the size of images is $320 \times 240$), which is fast enough for practical use such as vision systems for autonomous robots. This high performance can be realized by the recent progress of FPGAs and wide memory access to external RAMs (eight memory banks) on the FPGA board.

## 1 Introduction

The aim of stereo vision systems is to reconstruct the 3-D geometry of a scene from two (or more) images, which we call *left* and *right*, taken by cameras. Many dedicated hardware systems have been developed for real-time processing, and a stereo vision system with FPGA[1] has achieved real-time processing because of the recent progress of the size and the performance of FPGAs.

Compact systems for stereo vision are especially important for autonomous robots. FPGAs are ideal devices for the compact systems. Depending on situations, a robot may try to reconstruct the 3-D geometry, to find out moving objects which are coming to it, and to find out marker objects to check its position. FPGAs can support all these functions by reconfiguration.

In this paper, as the first step toward a vision system for autonomous robots, we describe a compact real-time stereo vision system which

1. supports camera calibration for easily obtaining correct results with simple circuit, and
2. checks *Left-Right Consistency* to find out occlusions without duplicating the circuit (by only adding another circuit for finding minimum value).

These functions are very imporant to obtain correct 3-D geometry. This system also supports filters for smoothing and eliminating noises to improve the system performance. In order to achieve these functions while exploiting maximum

P.Y.K. Cheung et al. (Eds.): FPL 2003, LNCS 2778, pp. 448–457, 2003.
© Springer-Verlag Berlin Heidelberg 2003

performance of FPGAs (avoiding memory access conflicts), we need one latest FPGA and eight memory banks on the FPGA board which are just supported on latest off-the-shelf FPGA boards.

The performance of the system is more than 20 frames per second (640×480 inputs and 640×480 output with disparity up to 200), which is much faster than previous works (more than 80 frames if the size of images is 320×240).

This paper is organized as follows. Section 2 describes the overview of stereo vision systems, and details of our system is given in section 3. The performance of the system is discussed in section 4. In section 5, conclusions are given.

## 2 Overview of Stereo Vision Systems

In order to reconstruct the 3-D geometry of a scene from two images (left and right) taken by two cameras, it is searched which pixels in the two images are projections of the same locations in the scene in the stereo vision systems. In this section, we first discuss the calibration of cameras, which is a very important step to simplify matching computation which is the most time exhaustive part in stereo vision systems, and then discuss matching algorithms and *left-right consistency* to suppress infeasible matches.

### 2.1 Calibration

Even if the same type of cameras are used to obtain left and right images, the characteristics of the cameras are different, and horizontal (vertical) lines in real-world may not be horizontal (vertical) in the images taken by cemeras. The aim of the calibration is to find out a realtionship (perspective projection) between the 3-D points in real-world and their different camera images. This is a crucial stage in order to simplify the following stages in the stereo vision systems and to obtain correct matching.

### 2.2 Matching Algorithm

Area-based (or correlation-based) algorithms match small windows centered at a given pixel to find corresponding points between the two images. They yield dense depth maps, but fail within occluded areas. Feature-based algorithms match local cues (e.g., edges, lines, corners) and can provide robust, but sparse disparity maps which requires interpolation. In hardware systems, area-based algorithms are widely used, because the operations required in those algorithms are very regular and simple.

In the area-based algorithms, *epipolar restriction* is used in order to decrease computational complexity. As shown in Figure 1, the corresponding point of a given point lies on its epipolar line in the other image, when the two cameras are arranged so that their principal axes are parallel. Corresponding points can then be found by comparing with every points on the epipolar line on the other image. To use this restriction, calibration of cameras is necessary to guarantee

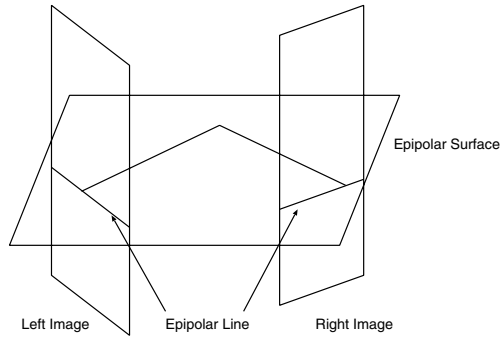



**Fig. 1.** Epiploar Geometry

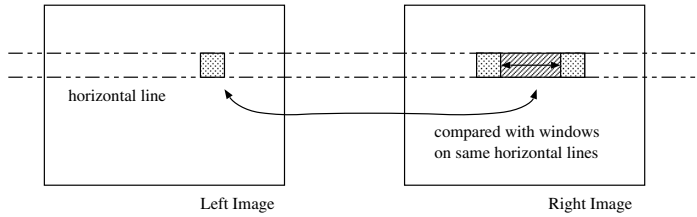



**Fig. 2.** Stereo Matching on Epilolar Constraint

that objects on a horizontal line in real-world also lie in the same horizontal lines in left and light images taken by the cameras. Then, we need to only compare windows on same horizontal lines in left and right images as shown in Figure 2.

The most traditional area-based matching algorithm is normalized cross-correlation [3], which requires more computation time than the following simplified algorithms. The most common pixel-based matching algorithm are squared intensity differences (SSD)[2] and absolute intensity differences (SAD)[4]. We used the SAD (Sum of Absolute Difference) algorithm because the algorithm is the simplest among them, and the results obtained by the algorithm is alomst same with other algorithms[4]. In SAD algorithm, the value of $d$ which minimizes the following equation is searched.

$$\sum_{i=-n}^{n} \sum_{j=-m}^{m} |I_r(x+i, y+j) - I_l(x+i+d, y+j)|$$

In the equation, $I_r$ and $I_l$ are the right and left image respectively, $n$ and $m$ are the size of the window centered at a given pixel (its position is $x$ and $y$). $d$ is the disparity, and its range decides how many pixels on the other image are compared with the given pixel. In order to find the corresponding points for an object which is closer to the vision system, larger range for $d$ becomes necessary, though it requires more hardware resources.

### 2.3 Occlusion and Left-Right Consistency

When pairs of images of objects are taken by two cameras (left and right), some parts of the objects appear in left (right) images and may not apper in right (left) images, depending on the positions and angles between the cameras and the objects. These occlusions are major source of errors in computational stero vision systems, though it has been reported that these occlusions help the human visual system in detecting object boundaries[5].

In many computational systems, one of left and right images is chosen as the base of the matching. Then, windows which include target pixels are selected in the base image, and the most similar windows are searched in another image. If the role of left and right images is reversed, different pairs of the windows may be selected. The so-called *left-right consistency constraint*[6] states that feasible window pairs are those found with both direct and reverse matching.

In our system, occlusions are detected by checking the left-right consistency. Figure 3 shows left image based matching and right image based matching. These matching can be executed in our system without duplicating whole circuit (by adding only another module which consists of comparators and selectors).

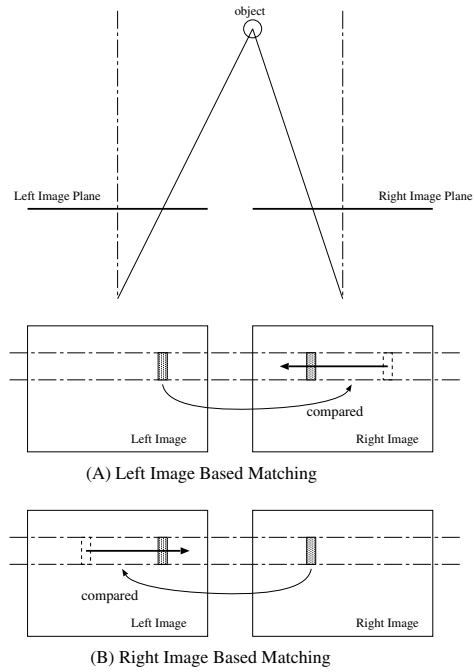



**Fig. 3.** Left Based / Right Based Matching

### 2.4 Filters

Filters are often used in the stereo matching for smoothing and eliminating noises to improve the system performance. We prepared Laplacian of Gaussian (LoG) filter for that purpose. Dual-port block RAMs make it possible to implement filters efficiently.

## 3 Details of the System

### 3.1 Overview

Figure 4 shows the system overview. Our system consists of a host computer, two cameras and one off-the-shelf FPGA board (ADM-XRC-II by Alpha Data with one additional SSRAM board). Left and right images taken by the two cameras are sent to external RAMs on the FPGA board.

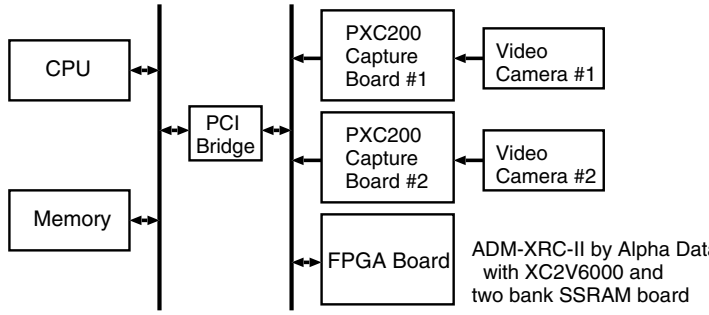



**Fig. 4.** System Overview

Figure 5 shows the structure of the FPGA board. The board has eight external memory banks (including two memory banks by the additional SSRAM board) which can be accessed independently. The first pair of images (left and right) sent from cameras are stored in bank0 and bank2 respectively, and next pair of images are stored in bank1 and bank3 while the images in bank0 and bank2 are processed by the circuit on the FPGA. The results by the circuit are written back to bank4 and bank5. When the data in bank4 is being sent to the host computer, FPGA writes new results to bank5. In order to exploit maximum performance of FPGA by avoiding memory access conflicts on these external memory banks, we need six memory banks for stereo matching its self. The rest two banks (bank6 and bank7) are used for the calibration described below. Thus, we need at least eight memory banks for our stereo vision system.

### 3.2 Calibration

In our system, calibration is performed using images of a predefined calibration grid. Before starting the system, the grid is given to left and right cameras (po-

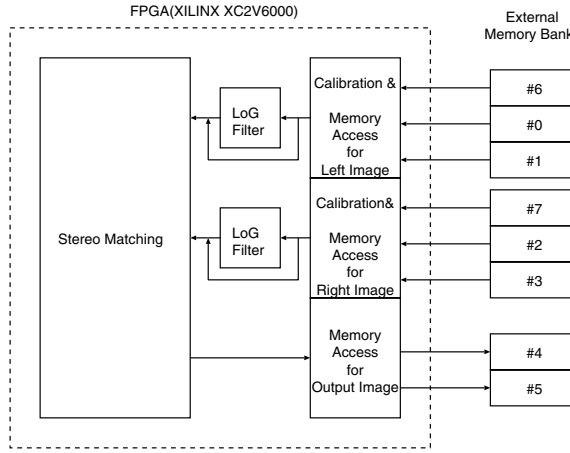



**Fig. 5.** FPGA Board

sitions of the cameras and the grid are fixed in advance), and the images of the grid taken by both cameras are sent to the host computer. Then, the host computer calculates which pixels on left and right images should be compared, and the positions of the pixels which should be compared are sent back to external RAMs on the FPGA board. These pixel position informations for left and right image are stored in bank6 and bank7, respectively (the size of the information is same with the size of images). In the later matching stages, FPGA first reads out the positions of the pixels which should be compared from the bank6 and bank7, and then the pixel data from bank0/1 and bank2/3 are read out using the positions.

This function is very important to obtain correct 3-D geometry with simple matching circuit, but this allows us to read only one pixel data in each clock cycle from bank0/1 and bank2/3, because the next pixel data which should be compared may not lies in the next address of the image data (when horizontal line in the image do not correspond to the true horizontal lines in real-world owing to the distortion of the lens of the camera and so on). Because of this restriction, the system performance is limited by the access time to the external RAMs on the FPGA board, and can not be improved by providing wider access to the external RAMs.

### 3.3 Matching and Left-Right Consisteny

Figure 6 shows the outline of the matching circuit. In Figure 6, suppose that window size is $n \times n$, and column data of windows ($n$ pixel data which are shown as $Li$ and $Ri$ in Figure 6) are read out at once in order to simplify the figure (in the actual system, only one pixel data is read out at once as described above).

Column data of windows in left image are broadcasted to all column modules, while column data of windows in right image are delayed by registers and then

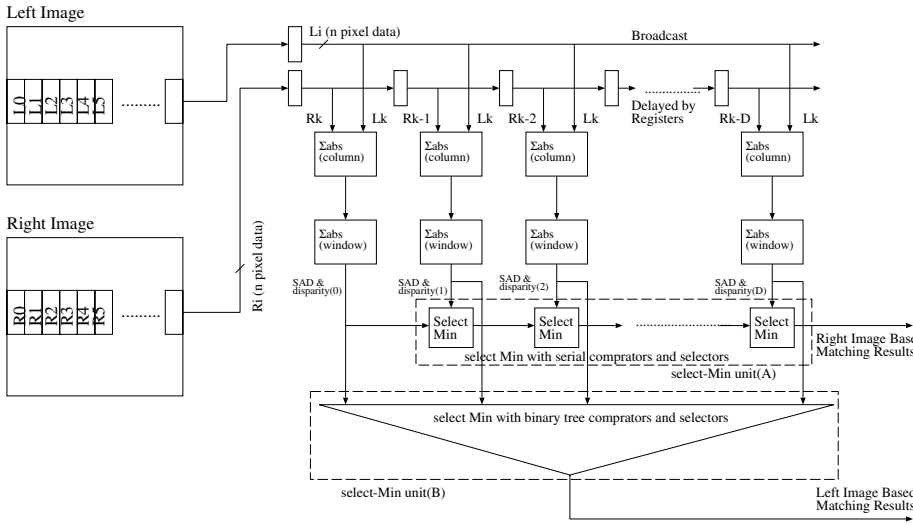



**Fig. 6.** Outline of the Matching Circuit

given to the column modules. In the column modules, sum of absolute difference of one column data is calculated. The outputs by the column modules are sent to window modules to sum up $n$ values (thus, sum of absolute difference of one window is calculated).

Outputs by the window modules are compared and minimum value and its disparity are selected by two kinds of units. In the select-minimum unit(A), outputs by the window modules are shifted (with daley), and compared with the outputs by the next window module. The smaller value and its disparity are selected and shifted to the next compare module. Then, the output by the last select-min module gives the minimum of sum of absolute difference and its disparity when the right image is chosen as base for the matching. In another select-minimum unit(B), all outputs by the window modules are compared by binary tree comparators and selectors, and minimum value and its disparity are selected. The output by this unit gives the minimum of sum of absolute difference and its disparity when the left image is chosen as base for the matching.

Figure 7 shows the outputs by the window modules when the window size is $5 \times 5$. In Figure 7, by comparing outputs of the window modules with shifting and delaying (parts covered by slanting lines in Figure 7), one window in the right image (window {R6-R10}) is compared with windows in the left image({L6-L10},{L7-L11},{L8-L12}...), while one window in left image({L11-L15}) is comapred with windows in the right image({R11-R15},{R10-R14},{R9-R13},...) by comparing outputs of window module at the same time (gray parts in Figure 7).

As described above, left-right consistency check (left image based matching and right image based matching) can be executed by only adding another com-

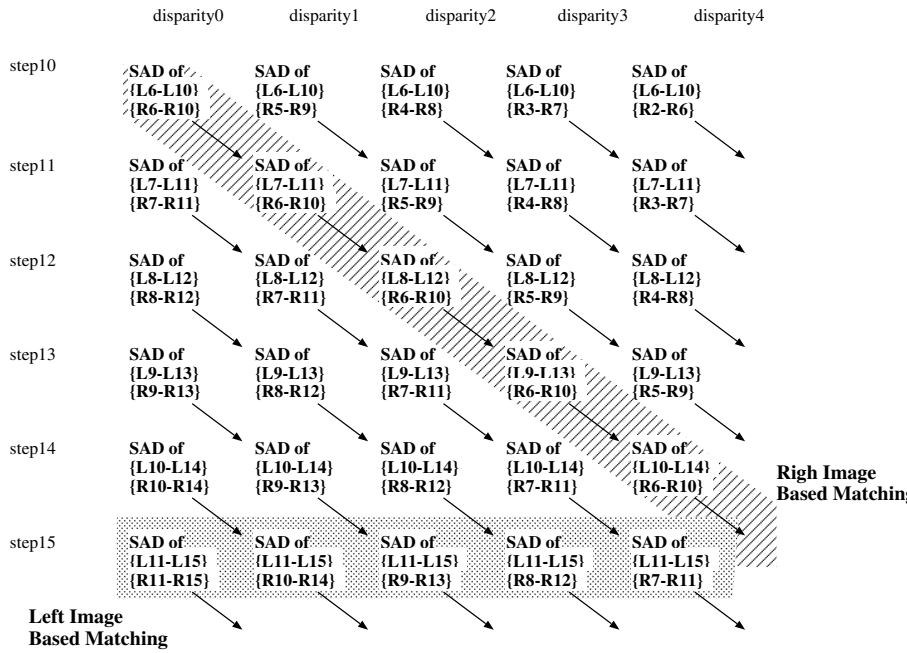



**Fig. 7.** Left-Right Consistency

pare unit which requires only D-1 comparators and selectors when D (number of window modules, namely maximum disparity) is $2^k$.

### 3.4 Details of the Modules

Figure 8 shows the details of the column module and the window module. In the column module, absolute difference of inputs from left image and right image (one pixel in each clock cycle as described above) is calculated, and summed up for $n$ times when the window size is $n \times n$. The outputs of the column module are sent to the window module, and $n$ outputs are summed up again to caluculate the SAD(sum of absolute difference) of $n \times n$ window. In the window module, new input value is accumulated, and input at (current_step - $n$) step is subtracted instead of adding $n$ previous values in order to reduce circuit size (previous values are stored in shift registers)[7]. As shown in Figure 7, the output of the window module is, for example, SAD of window {L6-L10}{R6-R10} at step 10, and SAD of {L7-L11}{R7-R11} at step 11. In this case, SAD of window {L7-L11}{R7-R11} can be calculated by the following equation.

$$\text{SAD of } \{\text{L7-L11}\}\{\text{R7-R11}\} = \text{SAD of } \{\text{L6-L10}\}\{\text{R6-R10}\} - \text{SAD of } \{\text{L6}\}\{\text{R6}\} + \text{SAD of } \{\text{L11}\}\{\text{R11}\}$$

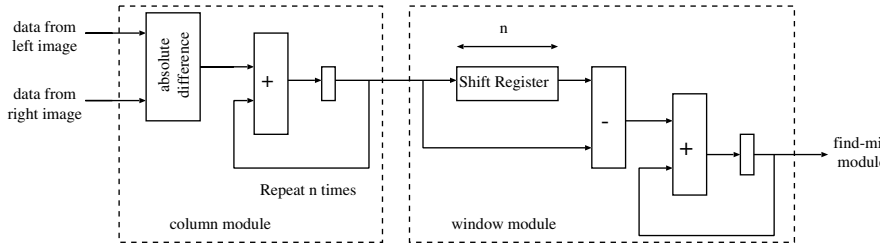



**Fig. 8.** Details of the Column Module and the Window Module

## 4 Performance

In our system, major factor that decreases system peformance is window size (because only one pixel is read from external memory owing to calibration, it takes n clock cycles to read one column data for the window), and factors that increase the circuit size is the maximum value of disparity which decide the number of modules.

Table 1 shows the system performance against the window size. The image size used for the evaluation is $640 \times 480$. As described above, the performance becomes worse as the window size becomes larger. The most often used window size in stereo vision systems is $7 \times 7$ or $9 \times 9$. The performace in Table 1 is calculated based on the maximum frequency reported by CAD. In practice, the system could process 20 frames per second in those window sizes, which is fast enough for autonomous robots. When, we need more performance, we can reduce the image size to $320 \times 240$ (which is widely used in other stereo vision systems). Then, the performace becomes four times faster without changing the circuit.

Table 2 shows the performance when we changed the maximum disparity. In this case, the circuit size becomes larger as the maximum disparity becomes larger, though the performance does not change as described above. Maximum disparity 200 is quite large compared with other stereo vision systems.

**Table 1.** Window Size and the Performance

| Window Size | $15 \times 15$ | $13 \times 13$ | $11 \times 11$ | $9 \times 9$ | $7 \times 7$ | $5 \times 5$ |
|---|---|---|---|---|---|---|
| Performance (frames per second) | 8.7 | 10.0 | 11.8 | 14.5 | 18.9 | 26.0 |

The size of left and right image is $640 \times 480$, and maximum disparity is 80.

**Table 2.** Maximum Disparity and Circuit Size

| Maximum Disparity | 80 | 160 | 200 |
|---|---|---|---|
| Circuit Size | 21% | 43% | 54% |
| Performance (FPS) | 18.9 | 18.9 | 18.9 |
| Operation Frequency (MHz) | 40 | 40 | 40 |

The size of left and right image is $640 \times 480$, and window size is $7 \times 7$.

## 5 Conclusions

In this paper, we described a compact stereo vision system with one off-the-shelf FPGA board with one FPGA. This system supports (1) camera calibration for easy use and for simplifying the circuit, and (2) left-right consistency check for reconstructing correct 3-D geometry. The performance of the system is limited by the calibration (which is a must for practical use) because only one pixel data can be allowed to read in owing to the calibration. The performance is, however, 20 frame per second (when the size of images is $640 \times 480$), which is fast enough for practical use such as vision systems for autonomous robots. The operation frequency of the system is still very slow. We are now improving the details of the circuit. We think that we can process more than 30 frames per second by this improvement.

This system became possible because of the continuous progress of FPGAs. We needed at least eight memory banks on the FPGA board to exploit maximum performance of FPGA avoiding memory access conflicts on the memory banks on the FPGA board while supporting calibration. We also needed the latest FPGA to support very large maximum disparity.

## References

1. M.Arias-Estrada, J.M.Xicotencatl, "Multiple Stereo Matching Using an Extended Architecture", FPL:2003-212, 2001.
2. P.Anandan, "A computational framework and an algorithm for the measurement of visual motion", IJCV, 2(3):283-310,1989.
3. T.W.Ryan, R.T.Gray, and B.R.Hunt, "Prediction of correlation errors in stereo-pair images", Optical Engineering, 19(3):312-322, 1980.
4. T.Kanade, "Development of a video-rate stereo machine" in IUW, pp. 549-557. 1994.
5. K.Nakayama and S.Shimojo, "Da Vinci stereopsis: Depth and subjective occluding contours from unpaired image points", Vision Research, 30:1811-1825, 1990
6. P.Fua, "Combining stereo and monocular information to compute dense depth maps that preserve depth discontinuities", IJCAI, 1991.
7. O.Faugeras, B.Hotz, H.Mathieu, T.Viville, Z.Zhang, P.Fua, E.Thron, L.Moll, G.Berry, J.Vuillemin, P.Bertin, and C.Proy, "Real time correlation-based stereo: algorithm, implementations and application,", Tech. Rep. 2013, INRIA, August 1993.