LOW-COST STEREO VISION SYSTEM FOR AUTONOMOUS MOBILE ROBOTS

A Thesis

Presented to

the Faculty of California Polytechnic State University

San Luis Obispo

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Computer Science

by

Connor Citron

August 2014

COMMITTEE MEMBERSHIP

TITLE:	Low-Cost Stereo Vision System for Autonomous Mobile Robots
AUTHOR:	Connor Citron
DATE SUBMITTED:	August 2014
COMMITTEE CHAIR:	Professor John Seng, Ph.D., Department of Computer Science
COMMITTEE MEMBER:	Professor Franz Kurfess, Ph.D., Department of Computer Science
COMMITTEE MEMBER:	Professor Chris Lupo, Ph.D., Department of Computer Science

ABSTRACT

Low-Cost Stereo Vision System for Autonomous Mobile Robots

Connor Citron

Something, something, robots. that

ACKNOWLEDGMENTS

I would like to especially thank my parents and family for their love and support.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

Introduction

Introducing ...

CHAPTER 2

Background

This chapter presents some general information on stereo vision that be useful for understanding the decision that were made in developing this stereo vision system.

2.1 Computer Stereo Vision

Computer vision is concerned with using computers to understand and use information that is within visual images [13]. There are many different types of computer vision, which range from using one image to multiple images to obtain information. One image is not enough to determine the three dimensional properties of the objects within the image.

Stereo vision uses multiple images of the same scene in order to construct a three dimensional representation of the objects in the images [11]. Comparing multiple images together for their similarities and differences allows for the depth to be obtained.

Binocular stereo [17] involves comparing a pair of images. These images are normally acquired simultaneously from a scene. By searching for corresponding pairs of pixels between the two images, depth information can be determined [17]. Pixel based comparisons can require substantial amount of computational power and time. Certain assumptions are made because of the computational resources required. Camera calibration and epipolar lines [cite 14-14 and define better] are common assumptions. For example, two images of the same scene are 640 x 480 pixels in size. Each image therefore contains 307,200 pixels, which is over 600,000 pixels between the two images for one frame. For a real-time application, say 30 frames per second for example, that becomes over 18 million pixels between the two images that would need to be processed every second.

Computational requirements for real-time applications can be reduced in several ways. First, by lowering the number of pixels in the images reduces the number of pixel comparison per second. Images at a size of 320 x 240 pixels would require a quarter of the number of computations at the cost of losing some amount of detail in the images. Also, reducing the number of frames per second will decrease the amount of computing needed. Going much below 30 frames per second is noticeable to a person and can be annoying to observe a slow frame rate. A robot on the other hand, depending on its task and how fast its moving, might only need a few frames per second in order to function within a desired range. So image resolution could be more important than frames per second for a robot if details are more important than speed.

Figure 2.1 below represents a simplified illustration of binocular stereo vision. The two cameras are held at a known fixed distance from each other and are used to triangulate different the distance of objects in the images they create. The points $\mathrm{U_L}$ and $\mathrm{U_R}$ in the left and right images, respectively, are 2D represents the point P that is in 3D space. By comparing the offset of between $\mathrm{U_L}$ and $\mathrm{U_R}$ in the two images, it is possible to obtain the distance of point P away from the cameras [1].

The closer an object is to the stereo vision system, the greater the offset of corresponding pixels will be. If an object is too close to the system, it is possible

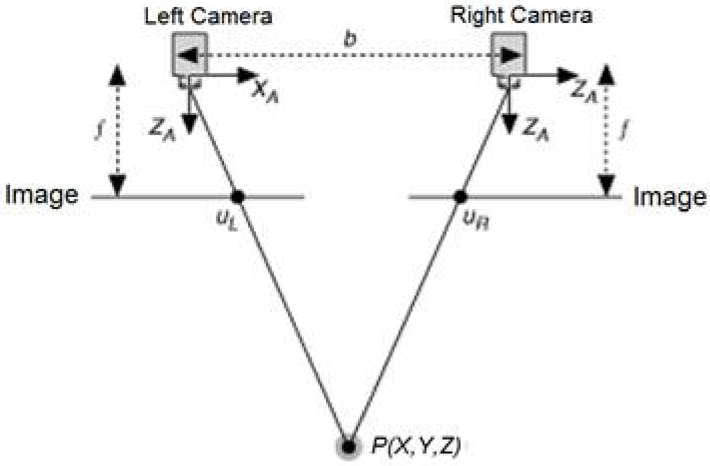


Figure 2.1: Simplified binocular stereo vision system [1].

for one camera to see part of an object that the other camera cannot. The farther an object is away from the stereo vision system, the smaller the offset of corresponding pixels will be. If an object is far enough away, it is possible for an object to be in almost the exact same location in both images. You can show this to yourself by holding a finger up close to your face, close one eye, and then alternate between which eye is open and which eye is closed. Your finger should appear to move a noticeable amount. Next, hold your finger as far away from you as you can and again alternate between which eye is open and which is closed. You should notice that your finger appears to move significantly less than it did when your finger was close to your face. That is how stereo vision works. The distance of an object is inversely proportional to the amount of offset between the two images.

2.1.1 Parallelism in Stereo Vision

Processing images for stereo vision allows for a high degree of parallelism. Locating the corresponding position of a pair of pixels is independent of finding another corresponding pair of pixels. This independent nature allows for the ability to process different parts of the same images at the same time, if there is hardware to support it.

Field Programmable Gate Arrays (FPGAs) allow for parallel processing to be implemented of the images. In section **(Implementation)** the amount of parallel processing used for the modular stereo vision system presented in this paper is discussed.

2.2 Stereo Vision Algorithms

Stereo vision algorithms can be placed into one of three different categories: pixel-based methods, area-based methods, and feature-based methods [9]. Pixel-based methods utilize pixel by pixel comparisons. They can produce dense disparity **(define!)** maps, but at the cost of higher computation complexity and higher noise sensitivity [9]. Area-based methods utilize block by block comparisons. They can also produce dense disparity maps and are less sensitive to noise, however, accuracy tends to be low in areas that are not smooth [9]. Feature-based methods utilize features, such as edges and lines for comparisons. They cannot produce dense disparity maps, but have a lower computational complexity and are insensitive to noise [9].

There are a lot of stereo vision algorithms out there [20]. In the taxonomy of [20], 20 different stereo vision algorithms were compared against each other using various reference images. Many algorithms are based on either the sum of absolute differences (SAD) or correlation algorithms [16].

An algorithm that is similar to SAD is Sum of the Square Differences (SSD). Both of these algorithms produce similar results and contain around the same amount of error [9]. SAD was chosen over the other algorithms to implement because it is simpler to implement in hardware. SSD requires squaring the difference between corresponding pixels and summing it up. Since squaring a number is the number multiplied by itself, the number will be added to itself that many times to produce the squared value. This is a lot more over head, and more hardware, than just taking the absolute difference of the difference of each corresponding pair.

2.2.1 Sum of the Absolute Differences Algorithm

SAD is a pixel-based matching method [16]. Stereo vision uses this algorithm to compare a group of pixels called a window from one picture with a window in another picture. The SAD algorithm, shown in Equation 2.1 [16], takes the absolute difference between each pair of corresponding pixels and sums all of those values together to create a SAD value. One SAD value by itself does not give any useful information about those two corresponding windows. Several SAD values will be calculated from different candidate windows for each reference window. Out of the all the SAD values calculated for the reference window, the SAD value with the smallest value (all of them are positive because of the absolute part in the equation) is determined to contain the matching pixel. Figure 2.2 shows for one reference window, there are several candidate windows used. The line that

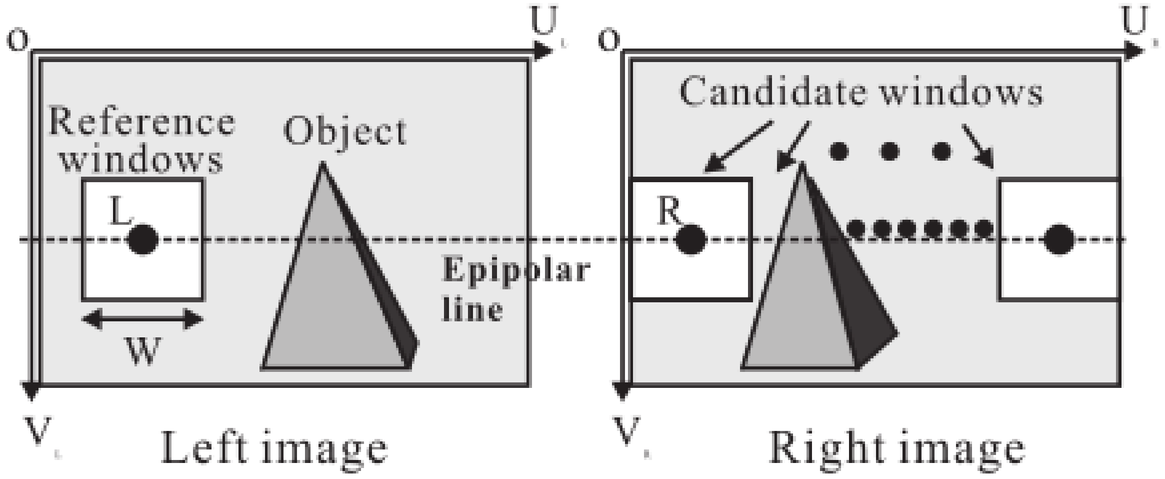


Figure 2.2: Searching for corresponding points between the two images [14].

the candidate windows move across are called epipolar lines.

$$\sum_{(i,j)\in W} |I_1(i,j) - I_2(x+i, y+j)| \tag{2.1}$$

In stereo vision, epipolar lines are created from the two cameras capturing images from the same scene. Figure 2.3 show the epipolar line that point X must be on in the corresponding images. This is useful because if the epipolar lines are known for both images, then it is possible to know the line that two corresponding points are on. It reduces the problem of finding the the same two points from a 2D area to a 1D line. Now, if the epipolar lines in both images are horizontal as they are in Fig. 2.2 as opposed to them being at a diagonal as they are in Fig. 2.3 then Eq. 2.1 reduces to Equation 2.2. For cameras that are not perfectly aligned, rectification is often used in order to align epipolar lines between images [15]. However, many stereo vision algorithms will assume that the epipolar lines are rectified to simplify the overall processing required.

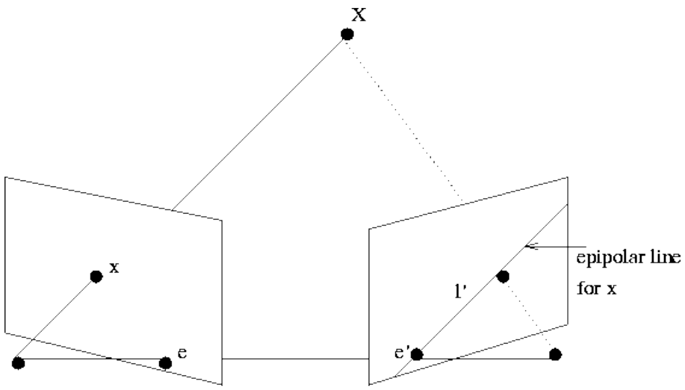
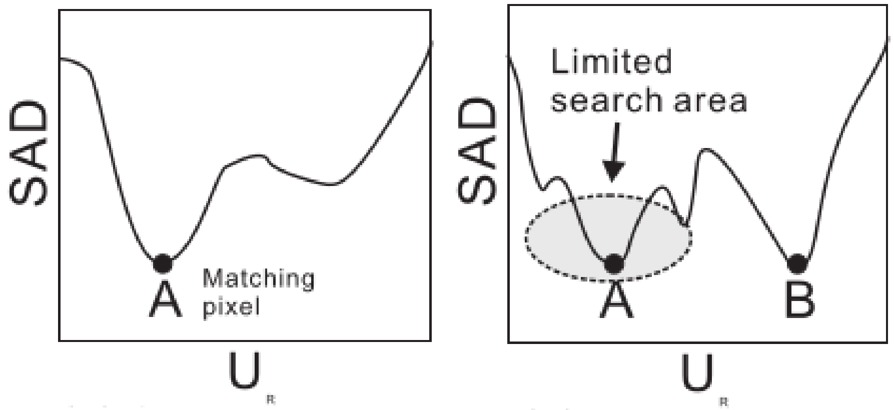


Figure 2.3: The epipolar line that point X is on for both images [8].

$$\sum_{(i,j)\in W} |I_1(i,j) - I_2(x+i,j)| \tag{2.2}$$

The disparity is the amount of offset between two corresponding pixels. The disparity range is the range that the candidate window will move through the image and is represented by the value 'x' in Eq. 2.2. It corresponds to the amount of SAD values that will be calculated. Figure 2.4 shows two types of SAD search methods. Fig. 2.4a selects the overall SAD value with the lowest value to be the matching pixel. However, Fig. 2.4b limits the search region to a specific area. This helps to avoid issues of similar looking areas that are not near the reference window from being falsely identified as matching. The downside to this is that if an object gets too close, meaning it would have high disparity values, and if the search region is not large enough, then the objects distance will be miss classified. It is important to determine a window size and a search region that are not too small and are not too big.

For example, Figure 2.5 shows a template window (Figure 2.5a) from one image and the search area (Figure 2.5b) in the other window. The disparity range is three, or zero to two. There are three 3x3 windows within the search



(a) Global SAD search

(b) Local SAD search

Figure 2.4: The SAD between a reference window and several candidate windows [14].

region in Fig. 2.5b. From left to right the three search windows have the center pixel as 4, 6, and 5, respectively.

Comparing corresponding pixels in the template window with the first search window (let's call it S0) gives the absolute differences for all nine pixels going from left to right and top to bottom of 8, 1, 1, 2, 1, 0, 1, 2, and 2. So the SAD value for S0 of 18 is obtained by adding up all nine of those values. The SAD value for the second search window (S1) is 6 and the last search window (S2) is 13. The template window has the smallest difference between S1, which means that the center pixel in S1 is determined to be the corresponding pixel for the center pixel in the template window.

The disparity value is 1 (how far the matching search window was shifted to the left). The disparity value is used to create a disparity map. Each disparity value in the disparity map is at the same relative location that the center pixel of its corresponding template window is located.

1	2	3
4	5	6
7	8	9

(a) Template Window

9	1	2	4	5
2	4	6	5	3
8	6	7	8	7

(b) Search Region

Figure 2.5: Template (reference) window and search (candidate) window.

CHAPTER 3

Related Works

There are several different ways to implement a stereo vision system. Many stereo vision systems are implemented on field-programmable gate arrays (FPGAs). FPGAs allow for parallelization when processing images. Systems that use FPGAs generally can achieve a high frames per second on a decent or good image quality, but most of these systems are expensive.

FPGA Design and Implementation of a Real-Time Stereo Vision System [16] uses an Altera Stratix IV GX DE4 FPGA board to process the right and left images that come from the cameras that were attached to it. [16] uses the Sum of Absolute Differences (SAD) algorithm to compute distances. This system allows for real time speeds up to 15 frames per second at an image resolution of 1280x1024. However, the Altera Stratix IV GX DE4 FPGA board costs over $4000, [4] which makes the system impractical for non-high budget projects.

Improved Real-time Correlation-based FPGA Stereo Vision System [12] uses a Xilinx Virtex-5 board to process images. [12] uses a correlation-based algorithm, which is based on the Census Transform, to obtain the depth in images. The algorithm is fast, but there are some inherent weaknesses to it. This system can run at 70 frames per second for images at a resolution of 512x512. Unfortunately, the Xilinx Virtex-5 board costs more than $1000, [5] which is still quite expensive.

Low-Cost Stereo Vision on an FPGA [18] uses a Xilinx Spartan-3 XC3S2000 board. [18] uses the Census Transform algorithm for image processing. This allows images with a resolution of 320x240 to be processed at 150 frames per second. The total hardware for the low-cost prototype used in [18] costs just over $1000, which is a bit too pricy for a lot of projects.

An Embedded Stereo Vision Module For Industrial Vehicles Automation [10] uses a Xilinx Spartan-3A-DSP FGPA board. [10] uses an Extended Kalman Filter (EKF) based visual simultaneous localization and mapping (SLAM) algorithm. The accuracy of this system directly varied with speed and distance of detected object. The Xilinx Spartan-3A-DSP FGPA board is around $600, [7] which is fairly expensive still.

Several commercial stereo vision systems exist presently [9]. Most of them are quite capable of producing good quality depth maps of their surroundings. However, the cost of these products can be relatively expensive, especially from a club or hobbyist standpoint. The Bumblebee2 [3] from Point Gray is able to produce disparity maps at a rate of 48 frames per second for an image size of 640x480, but it costs somewhere around $1000 or so. Having been involved with the Cal Poly Robotics Club for 6 years and seen the budgets each project in the club gets, $1000 would be most of their budget for the year. That kind of money could be better spent elsewhere on the project.

During the course of this thesis, a stereo vision surveillance application paper [19] was published that used the Digilent Atlys board [2]. A stereo camera module, VmodCAM [6], can be purchased with the Atlys board and was also used. The Atlys board is relatively cheap, at least by the standards presented thus far, at $230 for academic use. With the VmodCam included, the price goes around $350, which is still significantly cheaper than the other FPGA boards presented from other papers. This is why the Atlys board was selected for use in this thesis (the selection was independent of the surveillance paper). The surveillance paper used the AD Census Transform to calculate distance. Their board output the disparity map data over HDMI to a monitor. The output image is rather noisy, but it is very easy for a human to understand what is in the image.

CHAPTER 4

Implementation

Architectural stuff

CHAPTER 5

Experiments and Results

Experiments and Results

CHAPTER 6

Conclusions

Concluded.

CHAPTER 7

Future Work

In the Future!

BIBLIOGRAPHY

[1] 3d imaging with ni labview. http://www.ni.com/white-paper/14103/en/, August 2013.

[2] Atlys spartan-6 fpga development board. http://www.digilentinc.com/Products/Detail.cfm?NavPath=2,400,836&Prod=ATLYS&CFID=5602753&CFTOKEN=ff475ab97d889237-27A8B1C4-5056-0201-02F29D3CC5564ED6, July 2014.

[3] Bumblebee2. http://ww2.ptgrey.com/stereo-vision/bumblebee-2, July 2014.

[4] Digi-key. http://www.digikey.com/product-detail/en/DK-DEV-4SGX230N/544-2594-ND/2054809?cur=USD, July 2014.

[5] Digi-key. http://www.xilinx.com/products/boards_kits/virtex5.htm, July 2014.

[6] Vmodcam - stereo camera module. http://www.digilentinc.com/Products/Detail.cfm?NavPath=2,648,931&Prod=VMOD-CAM, July 2014.

[7] Xtremedsp starter platform spartan-3a dsp 1800a edition. http://www.xilinx.com/products/boards-and-kits/HW-SD1800A-DSP-SB-UNI-G.htm, July 2014.

[8] Epipolar geometry. `http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/OWENS/LECT10/node3.html`, Accessed July 18, 2014.

[9] P. Ben-Tzvi and Xin Xu. An embedded feature-based stereo vision system for autonomous mobile robots. In *Robotic and Sensors Environments (ROSE), 2010 IEEE International Workshop on*, pages 1–6, Oct 2010.

[10] P. Ben-Tzvi and Xin Xu. An embedded feature-based stereo vision system for autonomous mobile robots. In *Robotic and Sensors Environments (ROSE), 2010 IEEE International Workshop on*, pages 1–6, Oct 2010.

[11] M.Z. Brown, D. Burschka, and G.D. Hager. Advances in computational stereo. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(8):993–1008, Aug 2003.

[12] Jingting Ding, Xin Du, Xinhuan Wang, and Jilin Liu. Improved real-time correlation-based fpga stereo vision system. In *Mechatronics and Automation (ICMA), 2010 International Conference on*, pages 104–108, Aug 2010.

[13] M. Gosta and M. Grgic. Accomplishments and challenges of computer stereo vision. In *ELMAR, 2010 PROCEEDINGS*, pages 57–64, Sept 2010.

[14] M. Hariyama, N. Yokoyama, M. Kameyama, and Y. Kobayashi. Fpga implementation of a stereo matching processor based on window-parallel-and-pixel-parallel architecture. In *Circuits and Systems, 2005. 48th Midwest Symposium on*, pages 1219–1222 Vol. 2, Aug 2005.

[15] Li He-xi, Wang Guo-rong, and Shi Yong-hua. Application of epipolar line rectification to the stereovision-based measurement of workpieces. In

Measuring Technology and Mechatronics Automation, 2009. ICMTMA '09. International Conference on, volume 2, pages 758–762, April 2009.

[16] N. Isakova, S. Basak, and AC. Sonmez. Fpga design and implementation of a real-time stereo vision system. In *Innovations in Intelligent Systems and Applications (INISTA), 2012 International Symposium on*, pages 1–5, July 2012.

[17] Seunghun Jin, Junguk Cho, Xuan Dai Pham, Kyoung-Mu Lee, Sung-Kee Park, Munsang Kim, and J.W. Jeon. Fpga design and implementation of a real-time stereo vision system. *Circuits and Systems for Video Technology, IEEE Transactions on*, 20(1):15–26, Jan 2010.

[18] C. Murphy, D. Lindquist, AM. Rynning, Thomas Cecil, S. Leavitt, and M.L. Chang. Low-cost stereo vision on an fpga. In *Field-Programmable Custom Computing Machines, 2007. FCCM 2007. 15th Annual IEEE Symposium on*, pages 333–334, April 2007.

[19] G. Rematska, K. Papadimitriou, and A Dollas. A low cost embedded real time 3d stereo matching system for surveillance applications. In *Bioinformatics and Bioengineering (BIBE), 2013 IEEE 13th International Conference on*, pages 1–6, Nov 2013.

[20] D. Scharstein, R. Szeliski, and R. Zabih. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. In *Stereo and Multi-Baseline Vision, 2001. (SMBV 2001). Proceedings. IEEE Workshop on*, pages 131–140, 2001.