2008

Miniaturized embedded stereo vision system (MESVS)

Bahador Khaleghi
University of Windsor

Miniaturized Embedded Stereo Vision System (MESVS)

by

Bahador Khaleghi

A Thesis
Submitted to the Faculty of Graduate Studies through the
Department of Electrical and Computer Engineering in Partial Fulfillment
of the Requirements for the Degree of Master of Science at the
University of Windsor

Windsor, Ontario, Canada
2008

Your file Votre référence
ISBN: 978-0-494-57616-8
Our file Notre référence
ISBN: 978-0-494-57616-8

Declaration of Co-Authorship/Previous Publications

0.1 Co-Authorship Declaration

I hereby declare that this thesis incorporates material that is result of joint research, as follows:

This thesis also incorporates the outcome of a joint research undertaken in collaboration with Mr. Siddhant Ahuja under the supervision of Professor Jonathan Wu. The collaboration is covered in Chapter 4 of the thesis as the design and development of the hardware for this work has been done by Mr. Siddhant Ahuja. In all rest of the cases, the key ideas, primary contributions, experimental designs, data analysis and interpretation, were performed by the author, and the contribution of co-authors was primarily through helping in obtaining experimental results presented in section 5.2 and also providing some of the ideas presented in section 4.3.3.

I am aware of the University of Windsor Senate Policy on Authorship and I certify that I have properly acknowledged the contribution of other researchers to my thesis, and have obtained written permission from each of the co-author(s) to include the above material(s) in my thesis.

I certify that, with the above qualification, this thesis, and the research to which it refers, is the product of my own work.

0.2 Declaration of Previous Publication

This thesis borrows some of its material from two original papers that have been previously published, as follows:

1. B. Khaleghi, S. Ahuja, and Q. M. Jonathan Wu, "A New Miniaturized Embedded Stereo-Vision System (MESVS-I)", CRV 2008 (to appear).
2. B. Khaleghi, S. Ahuja, and Q. M. Jonathan Wu, "An Improved Real-Time Miniaturized Embedded Stereo-Vision System (MESVS-II)", IEEE CVPR Workshop on Embedded Computer Vision 2008 (to appear).

I certify that I have obtained a written permission from the copyright owner(s) to include the above published material(s) in my thesis. I certify that the above material describes work completed during my registration as graduate student at the University of Windsor.

I declare that, to the best of my knowledge, my thesis does not infringe upon anyone's copyright nor violate any proprietary rights and that any ideas, techniques, quotations, or any other material from the work of other people included in my thesis, published or otherwise, are fully acknowledged in accordance with the standard referencing practices. Furthermore, to the extent that I have included copyrighted material that surpasses the bounds of fair dealing within the meaning of the Canada Copyright Act, I certify that I have obtained a written permission from the copyright owner(s) to include such material(s) in my thesis. I declare that this is a true copy of my thesis, including any final revisions, as approved by my thesis committee and the Graduate Studies office, and that this thesis has not been submitted for a higher degree to any other University of Institution.

Abstract

Stereo vision is one of the fundamental problems of computer vision. It is also one of the oldest and heavily investigated areas of 3D vision. Recent advances of stereo matching methodologies and availability of high performance and efficient algorithms along with availability of fast and affordable hardware technology, have allowed researchers to develop several stereo vision systems capable of operating at real-time. Although a multitude of such systems exist in the literature, the majority of them concentrates only on raw performance and quality rather than factors such as dimension, and power requirement, which are of significant importance in the embedded settings.

In this thesis a new miniaturized embedded stereo vision system (MESVS) is presented, which is miniaturized to fit within a package of 5x5cm, is power efficient, and cost-effective. Furthermore, through application of embedded programming techniques and careful optimization, MESVS achieves the real-time performance of 20 frames per second. This work discusses the various challenges involved regarding design and implementation of this system and the measures taken to tackle them.

Acknowledgements

I would like to express my sincere gratitude to my supervisor Dr. Q. M. Jonathan Wu whose support, encouragement, and novel ideas enabled me to successfully complete this thesis. I also wish to thank Dr. M. Ahmadi for his helpful comments and directions. I like to give my sincere appreciation to Dr. B. Boufama for his invaluable recommendations, and cooperation. Finally, I should send my gratitude to my colleague Mr. Siddhant Ahuja as without his support and collaboration, it would be impossible to realize this research work.

This research has been supported in part by the Canada Research Chair Program, AUTO21 NCE, and the NSERC Discovery grant.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Problem Statement

Stereo vision, i.e. the process of inferring 3D scene geometry from two or more images taken from different viewpoints, is an active research area of computer vision. Stereo vision literature has significantly advanced, especially during the last decade, and several new matching methodologies capable of producing high-quality results have proven to be extremely useful in a variety of applications [1, 2, 19]. Nevertheless, due to the complexity, cost, and power requirements of the hardware needed to implement them, the fundamental choice of the algorithm is restricted especially when real-time performance is desired [19].

There are numerous examples of real-time operational stereo vision systems in the literature. Most of these systems rely on FPGA [4, 3], PC [5, 6], Commodity Graphics [7] or a combination of those [8] as their computational platform. Some researchers have also attempted to utilize the VLSI technology for developing application specific integrated circuit (ASIC) based systems [27]. However, because of their high cost, lack of flexibility, and the time-consuming and complicated design process, such systems are rarely implemented. Majority of the aforementioned systems have solely focused on raw performance and not on the size and power demand, which are crucial in embedded settings and applications [19].

The aim of this thesis is to develop a fully integrated miniaturized embedded stereo vision system (MESVS) that is suitable as intelligent 3D range sensor in applications that demand for miniaturization, power efficiency, and low cost such as miniaturized mobile robotics. The main focus in designing MESVS has been on keeping the overall dimension, power consumption, and cost of the system as small as possible while maintaining an acceptable level of accuracy and performance.

1.2 Motivation

There are several examples of miniaturized mobile robotic platforms in the research community such as Keppera [28], Tinyphoon [29], and the Swarm-bots project [30]. Nevertheless, in most of these systems robots have quite limited range measurement capabilities through mechanisms such as SONAR or IR-based sensors that totally restrict their performance. The main contribution of this thesis is design and practical implementation of an embedded miniaturized stereo vision system that can provide these tiny mobile robots with affordable 2D range data.

Stereo vision is the preferable choice for range sensing in small mobile robots over other range sensing technologies such as ultrasonic range sensors, IR-based sensors, and Laser range finders. Common disadvantage of the ultrasonic ranging systems is the difficulty associated with handling spurious readings, crosstalk, higher-order and multiple reflections [32]. IR-based sensors rely on the time of flight (TOF) technique along with the reflection of emitted infra red beam to determine distance to the desired objects. They also have drawbacks such as susceptibility to the intervention from other IR sources, especially sunlight, and therefore are only suitable for indoor applications. Moreover, they normally have limited range of operability. Finally, laser range finders do not fit the requirements of miniaturized mobile robotics since they are typically too bulky and heavy and also expensive to be deployed in such settings.

Stereo vision on the other hand is a suitable choice for 3D sensing technology in robots, especially mobile robots, for the following reasons. First, it is a robust and efficient way to extract range information from the environment and can be effectively implemented in real-time on low-cost hardware. Second, it is passive in nature, therefore, does not cause any interference with other sensory devices of the robot. Lastly, it can be easily integrated with other computer vision methodologies commonly used by robots such as object recognition and tracking, which would in turn improve the overall performance of the vision-based sensing module of the robots [31]. Yet stereo vision has been rarely used because solving the correspondence problem is difficult and computationally expensive. The latest advances in the area of computational stereo vision and embedded media processors, makes it feasible to design and implement a small embedded stereo vision system that can be deployed as 3D range sensing module in miniaturized mobile robots common to the areas such as collective and swarm robotics.

1.3 Thesis Organization

Following the introduction, the second chapter provides a review of the current stereo vision literature including recent advances and also state-of-the-art methodologies. Regarding the concentration of this thesis that is dense stereo matching, only algorithms capable of producing dense disparity maps are discussed. This chapter explains some basic concepts required to understanding stereo vision such as disparity, triangulation, and epipolar geometry. Then, categorizes dense matching algorithms into local and global matching methods and presents different algorithms proposed so far belonging to each category along with a discussion of their features, advantages, drawbacks, and overall performance.

Chapter three is dedicated to stereo vision algorithms and systems capable of operating at real-time or near real-time that exist in the literature. Most of the methods presented in this chapter belong to the local matching group of methods. Because as will be discussed in chapter two local matching methods are typically less computationally expensive as compared to their counterpart (i.e. global matching methods) and are therefore a better candidate for fast and efficient implementation required in real-time stereo vision systems. Although, some variants of global matching methods have also been shown to be a feasible solution and therefore are included as a part of discussion.

In chapter four the miniaturized embedded stereo vision system (MESVS) developed for

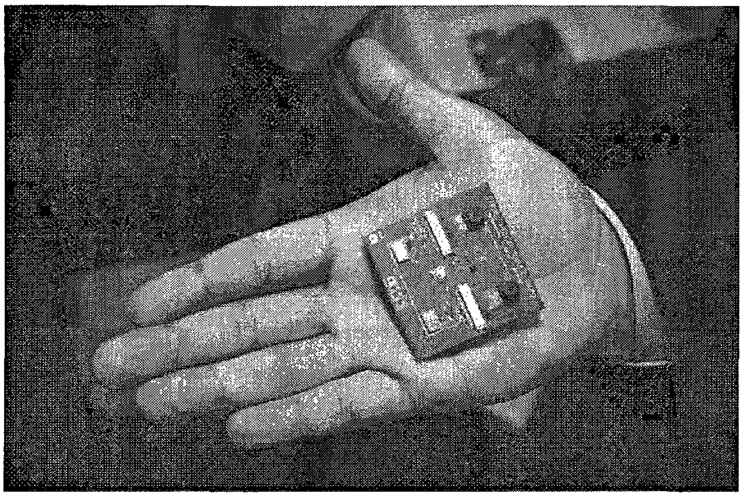


Figure 1.1: MESVS prototype

this thesis is explained (see Figure 1.1). It begins with a analysis of range accuracy versus horopter. The next subsection presents a discussion of the hardware platform selection for MESVS. It argues that regarding the design constraints and available options on the market, the new generation of embedded media processors is the best choice and then elaborates on the specific features and components of MESVS. Details of hardware implementation are also described. The system firmware (i.e. stereo matching engine) is discussed in the next subsection. At first the offline calibration procedure used to obtain the internal and external parameters of the stereo rig is explained. Next, each of the building blocks of the stereo matching engine including image sub-sampling, stereo rectification, pre-precessing, matching, and postprocessing steps are described. The special considerations made in order to improve the performance and accuracy of the system firmware within the embedded programming context are also detailed in this chapter.

The qualitative and quantitative experimental results presented in chapter five, illustrate the efficiency and robustness of the stereo vision system developed. The qualitative results show that our stereo matching engine can retrieve the proper shape for the foreground objects within a close vicinity of the system as supported by theoretical analysis in section 4.2.1. Furthermore, the quantitative results presented for the novel post processing algorithm demonstrate its ability to detect and remove most of mismatches caused by textureless and depth discontinuity regions in the scene.

Finally, chapter six provides the conclusion of this work along with suggestions regarding the future avenues of development for our system.

Chapter 2

Stereo Vision Literature

2.1 Stereo Vision

Stereo vision is one of the oldest and most heavily investigated problems of computer vision. it tries to retrieve depth information of a scene using two (or more) images of that scene taken with cameras separated by a distance. Numerous potential application areas such as 3D map building, view synthesis, image based rendering, and industrial automation keep stereo vision researchers motivated for further improvement of the literature. Every stereo vision system typically involves three major stages. The first step is the stereo calibration in which the extrinsic (transformation from world to camera coordinate system) and intrinsic parameters of the cameras (e.g. focal length, distortion, center of projection) are estimated. Correspondence problem also referred to as stereo matching problem is the second step, which involves identifying the corresponding points between images. The output of this step is called disparity map; a 2D image that specifies the amount of shift (disparity) between corresponding pixels. The last step is the 3D reconstruction, which deploys different geometrical methods such as triangulation to estimate the 3D geometry of the objects in the scene. Complexity of the natural scenes makes the stereo correspondence problem an ill-posed problem such that it is usually the most time-consuming and difficult step of a stereo vision system. Therefore, majority of research work in this field has been focused on development of robust and efficient algorithms to tackle this issue. Following sections provide a detailed discussion of the current state of the stereo matching literature including challenges of correspondence problem, common constraints and assumptions applied to facilitate the matching process, and a broad overview of existing stereo matching algorithms in the literature.

2.1.1 Taxonomy of dense stereo matching methodologies

In order to make the following review of stereo matching literature (presented in section 2.2) more clear, a taxonomy of dense stereo matching methods is introduced [12]. This taxonomy is used to evaluate and compare different components and design strategies applied in individual stereo matching algorithms. Since this work fits within the category of dense matching methods, the feature-based matching methods that provide only sparse disparity maps are not discussed here. Most of the stereo matching algorithms generally perform (subset) of the followings four steps (see figure 2.1) [12]:

- Matching cost computation: in this step some dissimilarity measure is deployed to compute pixel-wise matching cost. Examples of dissimilarity measures commonly used are Euclidean (squared difference), and Manhattan (absolute difference) metrics. the output of this step is the calculated matching cost values over all pixels and all disparities and forms the initial disparity space image $C_0(x, y, d)$ (considering only two-frame methods).

- Cost aggregation: this step is performed by summing or averaging over a support region in the disparity space image. The support region can be chosen as 2D (at a fixed disparity) in order to favor fronto-parallel surfaces or 3D in (x-y-d space) in order to favor slanted surfaces. Depending on the type of support region (2D or 3D) applied, aggregation may be implemented using different methods such as square windows, shiftable windows, and Gaussian convolution in case of 2D region. In case of 3D regions, the limited disparity difference [33], limited disparity gradient [34], and

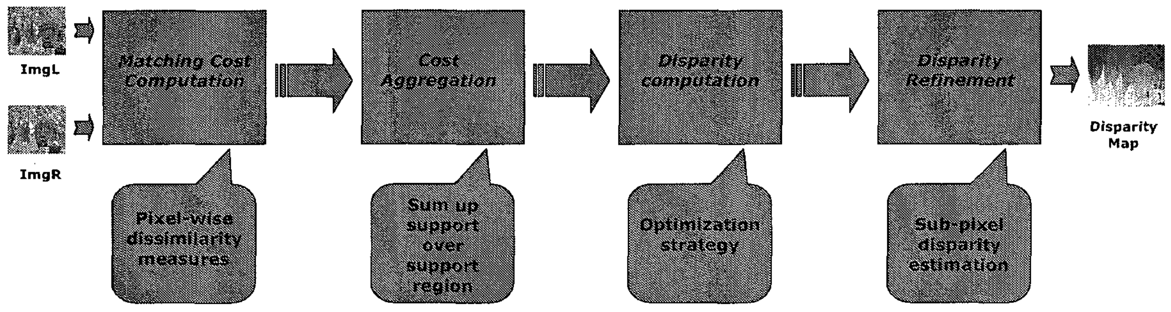

ImgL
ImgR
Matching Cost Computation
Cost Aggregation
Disparity computation
Disparity Refinement
Disparity Map
Pixel-wise dissimilarity measures
Sum up support over support region
Optimization strategy
Sub-pixel disparity estimation

Figure 2.1: Taxonomy of dense stereo matching algorithms

Prazdnys coherence principle [35] may be applied. The aggregated disparity space image $C(x, y, d)$ is the output of this step.

- Disparity computation/optimization: this step can be as simple as choosing the disparity associated with the minimum cost for each pixel (as done in the local matching methods, see section 2.2.1). On the other hand, this step can also be the main emphasis of the matching algorithm and based on complex global optimization methods when disparity computation is formulated as an energy minimization problem (as done in most global matching methods; see section 2.2.2). This step computes the initial disparity map.

- Disparity refinement: this last step is typically further performed to improve the quality of disparity estimates and may involve sub-pixel disparity estimation using iterative gradient descent method, fitting a curve to the matching costs, removal of occluded regions through cross-checking, or application of median filter in order to eliminate spurious mismatches. The final disparity map is produced in this step.

2.1.2 Challenges

Stereo matching algorithms have to deal with a multitude of challenges caused by the complexity of the natural scenes, environmental and systematic shortcomings, or simply the nature of stereo matching algorithm itself. As a result, there is no general solution to this problem and each stereo matching algorithm incorporates some simplifying assumptions (as explained in the next section) and solves the problem in a constrained manner. Following are some of the most important challenges involved in the stereo matching process:

- Depth discontinuities: as mentioned before most stereo matching algorithms operate by aggregating local support within a correlation window [12]; making the implicit assumption that all pixels within the window have similar disparity values. such assumption is violated when aggregating over depth discontinuity regions of the scene (usually object boundaries). Thus, results produced for these regions are mostly not reliable.

- Half-occluded regions: these are pixels that are only visible in one of the images (in case of binocular stereo) and therefore have no corresponding pixel in the other image. As a result disparity value may not be properly estimated for such regions. Stereo matching algorithms typically treat half-occlusion by detecting them through a variety of methods [17] and then propagating neighbor disparity values to fill these regions in the disparity map.

- Low texture regions: since most of the pixels within these regions are quite similar, correlation values computed for different candidate disparities would be very close, which makes it difficult for stereo matching algorithm to designate the best candidate and come up with the correct disparity. this is even more problematic for methods based on winner take it all (WTA) strategy.

- Variable illumination: in some applications stereo imagery are taken under different lighting or exposure conditions resulting in local and/or global radiometric variations in between corresponding pixels of the image pair. That is, corresponding pixels will not have exact same values making it difficult for the stereo matching algorithm to produce proper disparities.

- Specularities in the scene: this problem is caused by non-Lambertian surfaces in the scene. i.e. surfaces for which the reflection properties depend on the angle of the emitted light. For example, such surfaces may look dull in one image while seem shinny in the other (due to change at the angle of light source). This phenomenon usually causes many spurious matches within the disparity map unless treated appropriately.

- Image noise: noise may be caused by a variety of sources such as the difference between cameras gain and/or bias, errors introduced by image pixelization or sub-sampling process, and finally environmental noise that harms quality of images produced by the cameras. This issue may be handled through a combination of methods. For example, devising sampling insensitive measures [36], careful calibration of the cameras and estimating their gain/bias or deploying cost functions that are invariant to difference in gain/bias such as local non-parametric transforms [14]. Finally, several filters may be applied to the image in order to reduce the effect of noise.

2.1.3 Constraints and Assumptions

There are a wide variety of constraints and assumptions used to design stereo matching algorithms reflected in the broad range of methods that have been developed to solve this problem. Some of the most commonly used constraints are as follows:

- Epipolar constraint: one of the most commonly used and useful stereo constraints in the literature that is also used in this work. It states that for each given point $p1$ in the left image, the object point P projected onto point p lies somewhere on the ray that crosses points $C1$ and $p1$. Projecting this ray onto the right image results in epipolar line defined by the corresponding point $p2$ and the epipole $E2$. That is, the matching point of point $p1$ lies on the corresponding epipolar line in the right image. Figure 2.2 shows a geometrical representation of this constraint.

  Applying epipolar constraint helps decrease search space from 2D (entire image) into 1D (along epipolar line) and therefore significantly reduces the computational complexity of the matching algorithm. For example, for an image with VGA (640x480) resolution, the search over 640x480=307,200 pixels will be reduced to only 800 pixels (a diagonal epipolar line in the worst case).

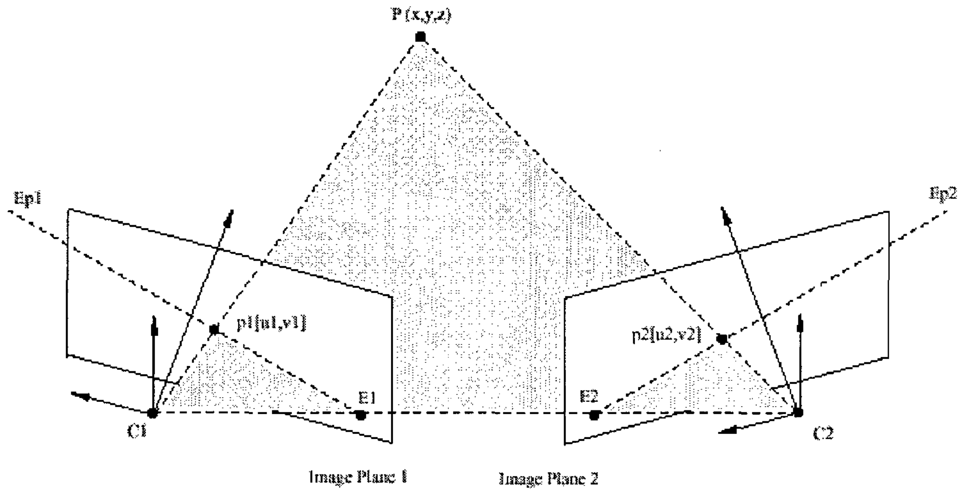


Figure 2.2: The epipolar constraint

- Smoothness constraint: this constraint (also referred to as continuity constraint) is widely used in stereo matching algorithms dictating that for each disparity estimated for a pixel, the neighboring disparity values do not abruptly change. Obviously, this assumption does not hold for disparities belonging to the object boundaries. Therefore, special considerations must be made to deal with such regions if this constraint is applied.

- Ordering constraint: The ordering constraint assumes that the order of points along the according epipolar line is preserved by the stereo projection. i.e. for a given point n on the right side of point m on the epipolar line, the matching point n' of n on the other image must lie on the right side of the matching point m' of m as shown in Figure 2.3.

  If used effectively, the ordering constraint would help to further reduce the search neighborhood and produce less ambiguous and more accurate results. Although ordering constraint is generally useful, one should be careful incorporating it into the matching process for two reasons. First, it may propagate incorrect information produced by previous mismatches through relying on their value to ascertain the future

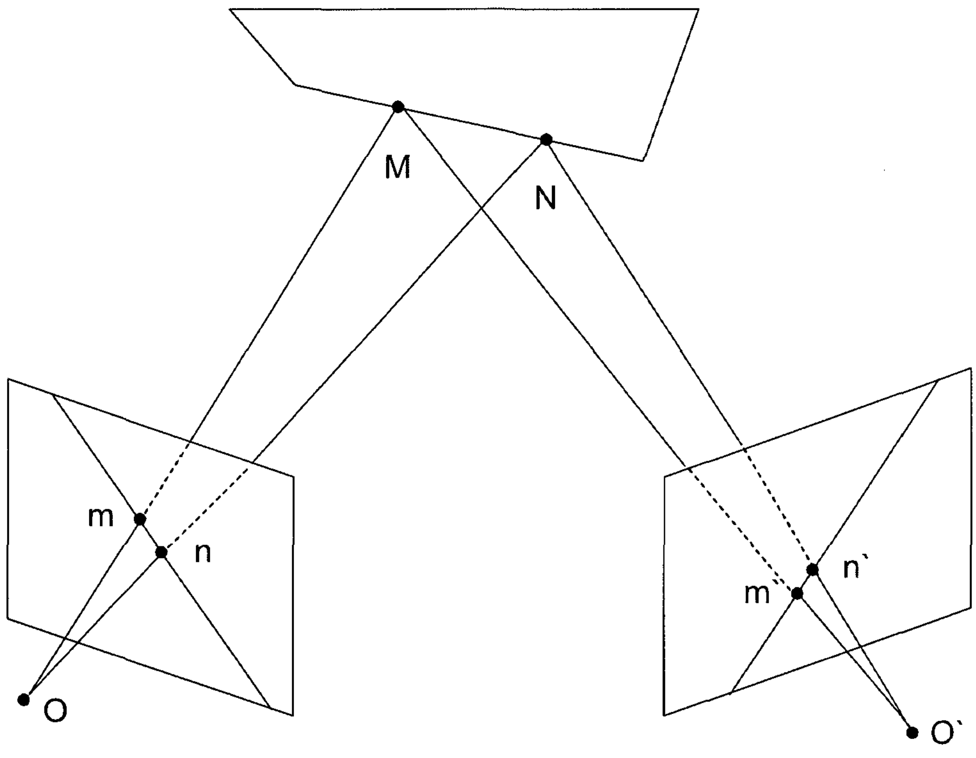


Figure 2.3: The ordering constraint

neighbor disparities. Second, this constraint does not always hold and is violated when we have thin foreground objects in the scene.

- Uniqueness constraint: This constraint states that for every point in the left image there is at most one matching point in the right image. It is based on the geometrical observation that it is impossible for an object point to project into more than one point (if non-occluded) in any given image. This constraint may be deployed to simplify the matching process or validate obtained results. One should note that this constraint is also violated when dealing with slanted surfaces as for such cases one point may be matched to many others.

- Light consistency constraint: This constraint assumes all object surfaces to have the Lambertian property, i.e. having reflection properties independent of the angle of light emission.

- Visibility constraint: This constraint is originally designed in order to handle occlusion and is derived from the definition of occlusion stating that a pixel in the left image will be visible in both images if there is at least one pixel in the right image matching it [37]. Visibility constraint is more general (and weaker) compared to both ordering and uniqueness constraints as it permits many to one matching, unlike the uniqueness constraint. Furthermore, the ordering constraint need not to be satisfied.

2.2 Stereo Matching Methodologies

As mentioned before stereo vision has been the subject of intense research for a long period of time (several decades). As a result, stereo vision literature is filled with a wide variety of methods developed with different level of efficiency and performance. Generally, most of these algorithms can be broadly categorized into local and global matching methods. Local methods typically calculate the disparity map using local image correlation over a certain area (window). Global methods on the other hand, model the disparity map constraints explicitly using energy functions and infer it through a global (usually iterative) optimization process. The following sections provide a brief review along with a comparison and analysis of both of the aforementioned matching algorithm categories.

Before proceeding to the next section, it is worth to briefly discuss phase-based stereo matching methods. Phase-based stereo matching techniques have been the center of attention during the last decade [38]. Their idea is to use phase information instead of image intensities to establish correspondence and estimate disparities. More specifically, disparity is defined as the shift required to align the phase values of the two signals. Phase-based methods are robust to contrast variations and imbalance between stereo image pair because phase information is amplitude invariant. Nevertheless, the main disadvantage of phase-based matching algorithms is their high sensitivity to noise, which makes them unsuitable for applications with low signal to noise ratio (i.e. less than 0.9-0.95) [38].

2.2.1 Local Matching

Local matching methods perform matching at a local level (hence the title local) and typically involve the following steps:

1. Apply some dissimilarity measure to compute the pixel-wise matching cost
2. Aggregate matching cost over correlation window along the epipolar line within search window for all candidate disparity values
3. Determine the disparity based on WTA (Winner-Take-All) strategy, i.e. choose the disparity associated with the minimum cost at each pixel
4. Post-process results to remove outliers

Figure 2.4 depicts an example of performing local matching trying to find the corresponding point for a point p in the left image. As shown a local neighborhood of pixels surrounding point p is considered (the correlation window) in the left image. The same area (in terms of size and shape) is considered in the right image and is moved over for d different cases (d is the disparity range) along the epipolar line (presented as a dashed line). The entire area in the right image covered by this process is called the search window. Each time the correlation window in the right image is moved, the new correlation value associated to that window is computed and stored. Finally, the displacement (disparity) producing the best (usually highest) correlation value is chosen.

Local matching methods have the advantage of being fast and easy to implement yet they produce less accurate and rather unreliable results. A multitude of local matching methods exist in the literature, which differ in the way they perform steps 1 and 2 as the optimization step for these methods is trivial. Examples of different dissimilarity measures used by these methods include the absolute difference, squared difference, non-parametric local transforms (rank and census), normalized cross correlation (NCC), gradient-based measures, and Laplacian of Gaussian (LoG). Also, several variants of correlation windows have been proposed such as adaptive window [40], multiple window [41], adaptive support weight [39], and segment-based window [25].

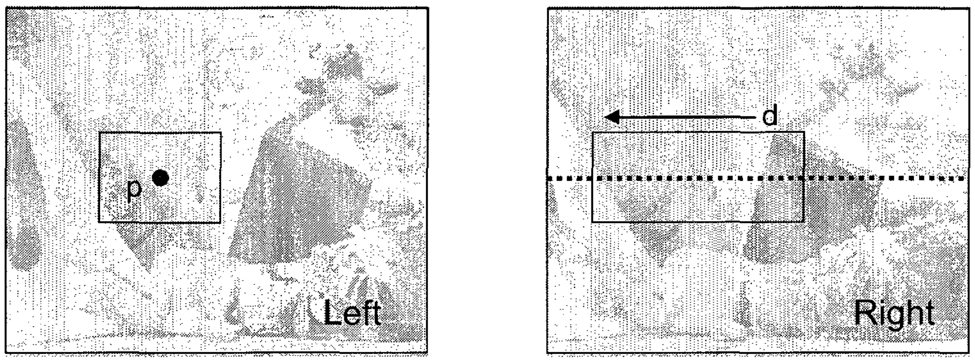


Figure 2.4: Pattern matching technique used in most local matching methods

2.2.2 Global Matching

Global matching methods formulate stereo matching problem in an energy optimization framework and are capable of yielding high quality results in the cost of being more complicated and therefore slower (as compared to local methods). Following is a summary of global methods features:

- Use stereo image pair as given evidence
- Often skip cost aggregation step
- Enforce the matching constraints through some potential functions incorporated into the energy function as shown in Equation (2.1):
  - Data term measures how well estimated disparity agrees with the input image pair
  - Smoothness term encodes smoothness assumptions regarding the disparity map
- Compute disparity map using a global optimization method (either 1D or 2D) to find the disparity map corresponding to the minimum energy level

(2.1)

$$D = \arg\min_d E(d)$$
$$E(d) = E_{Data}(d) + \lambda E_{Smooth}(d)$$

The data term $E_{Data}(d)$ used in the energy function (also referred to as likelihood within the Bayesian framework) is a measure of how well the disparity function $d(x, y)$ agrees with the stereo image pair. It is computed based on the $C(x, y, d(x, y))$ function, which is the initial (or aggregated) matching cost function calculated in the previous steps. The data term is generally defined as follows [12]:

$$E_{Data}(d) = \sum_{x,y} C(x, y, d(x, y))$$

The $E_{Smooth}(d)$ is the second term typically incorporated into the global energy function, which encodes the smoothness assumptions regarding the desired disparity map. Many global matching methods keep the smoothness function simple (e.g. by measuring only the differences between neighboring pixels disparities) in order to make the following optimization process computationally tractable. Equation (2.2) presents a generalized form of the smoothness function commonly used in the literature [12].

$$E_{Smooth}(d) = \sum_{x,y} \rho(d(x, y) - d(x + 1, y)) + \rho(d(x, y) - d(x, y + 1)) \tag{2.2}$$

The ρ is typically a monotonically increasing function of the disparity difference. If designed poorly, the smoothness term may smooth the disparity map everywhere and therefore lead to poor results at object boundaries. Global matching methods that rely on robust smoothness function and do not have such an issue are called $Discontinuity-preserving$. An example of such robust functions is illustrated in Equation (2.3), which relies on the pixel intensity differences to reduce the smoothness costs at high intensity gradients [12].

$$E_{Smooth}(d) = \sum_{x,y} \rho(d(x, y) - d(x + 1, y)).\rho_I(||I(x, y) - I(x + 1, y)||) + \rho(d(x, y) - d(x, y + 1)).\rho_I(||I(x, y) - I(x, y + 1)||) \tag{2.3}$$

The above formulation is based on the observation that object boundaries (depth discontinuity regions) usually coincide with the sudden changes in image gradients (edges). Thus to preserve depth discontinuities the smoothness term should not penalize such regions; i.e. ρ_I is a monotonically decreasing function of intensity differences.

Examples of the smoothness functions widely used in the literature are truncated linear and quadratic model [12], the Potts model (also symmetric version) [42], and more recently the differential geometry consistency based method that is based on surface normal instead of the individual disparity values and can successfully recover both slanted and curved surfaces in the scene [43].

In majority of the global matching methods the main focus has been on improving the cost function optimization (i.e. minimization) step. In fact, there are numerous global stereo matching algorithms each based on a different energy optimization approach. These approaches can be further divided into 1D and 2D optimization methods depending on the level at which optimization is performed. In 1D methods such as dynamic programming [8], scanline optimization [44], and genetic algorithm [45] the optimization is performed for pixels within the image scanline at a time. In contrast, in 2D methods such as simulated annealing [47], belief propagation (BP) [1], graph cuts (GC) [2], and synchronous optimization [46] the entire image is considered at once. For example, in BP-based methods the disparity surface is modeled as a Markov random field (MRF). Then, assuming a Bayesian inference framework, the minimum energy level of the matching cost function corresponding to the Maximum A Posterior (MAP), i.e. the most probable disparity surface given input stereo image pair as evidence, is obtained deploying belief propagation approach iteratively. Figure 2.5 shows a summary of stereo matching categorization discussed here.

Recently, stereo vision researchers have diverted their center of attention from development and experimentation with efficient energy minimization methodologies into attempting to improve the cost function definition. For example Yoon and Kweon [42] propose a new symmetric stereo cost function that treats both the reference and target images equally to improve performance without explicitly modeling occlusion or using color segmentation information [42]. Another prominent example is the application of differential geomet-

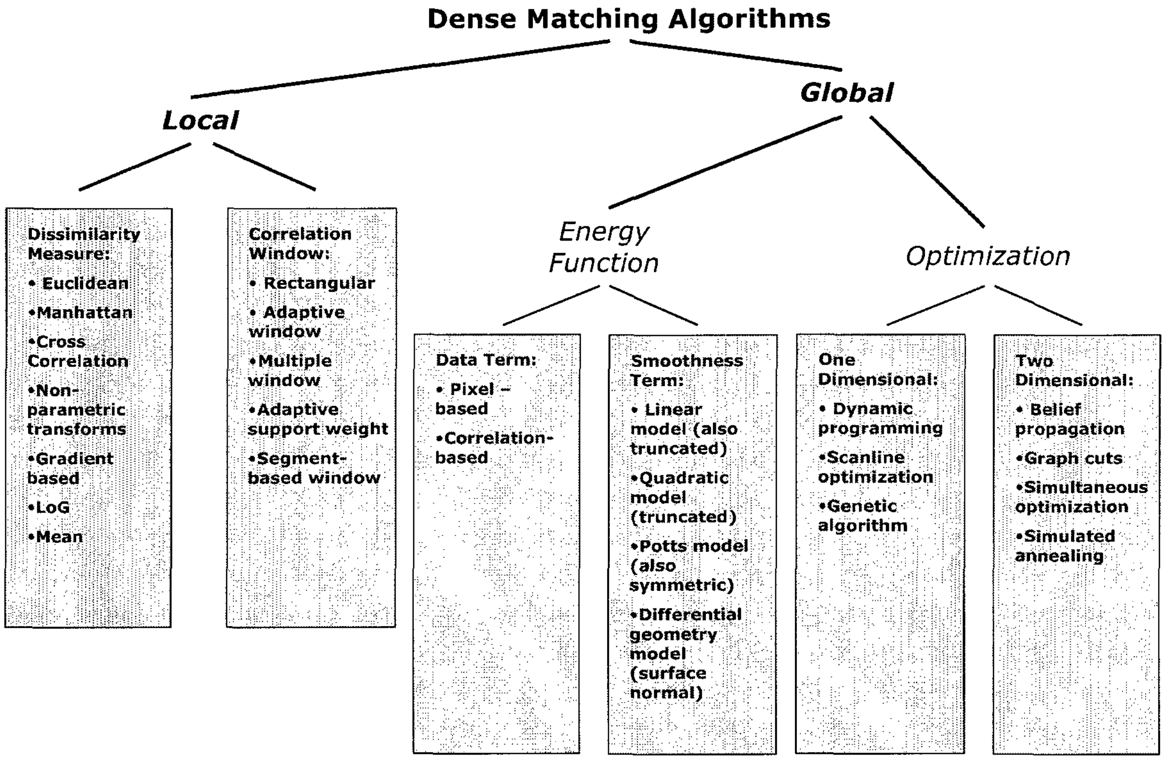
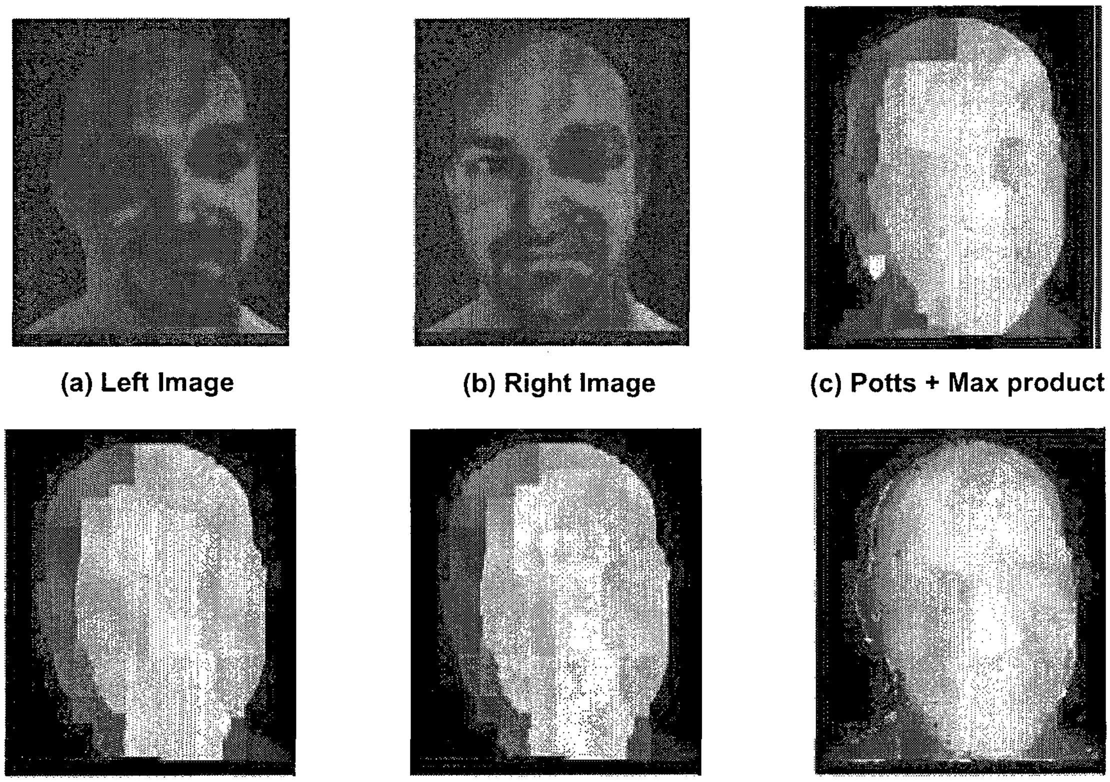


Figure 2.5: Summary of dense stereo matching methodologies

ric consistency as smoothness term, which helps eliminating the systematic error associated with modeling slanted and curved surfaces (common in the majority of matching algorithms assuming only fronto-parallel surfaces). This new method enforces geometric consistency for both depth and surface normal [43]. The experimental results presented in Figure 2.6, prove the efficiency and superior performance of this improved cost function in recovering depth information of human face as compared to two of the models commonly applied in the literature; i.e. truncated linear and Potts model. In all cases Bayesian belief propagation in max-product mode is used to estimate the MAP of the matching cost function.

2.2.3 Local vs. Global

Local matching methods are inherently simpler than their global counterparts and therefore faster. They may also be implemented easier and more efficiently. As mentioned the dispar-



(a) Left Image (b) Right Image (c) Potts + Max product

(d) Linear + Max product (e) Linear + Max product (with subpixel refinement) (f) Surface geometry + Max product

Figure 2.6: Comparison of the surface geometry model to the truncated linear and Potts models [43]

ity computation step in local matching methods is often trivial (based on WTA strategy) and they do most of their work in the first two steps of matching process, namely, matching cost computation and support aggregation. The main difficulty of applying local methods is that it is hard to determine the proper (optimal) size and shape of the support region. Large correlation windows tend to blur object borders and small correlation windows are sensitive to noise and increase the number of mismatches. Furthermore, due to their simpler structure, local methods often produce less accurate results in contrast to global methods. Although, some recent work such as [39] provides results comparable to global matching methods but at the cost of a relatively high computational complexity.

Table 2.1: Brief comparison of local vs. global matching methods

Methodology	Local	Global
Emphasis	Matching cost computation Cost aggregation	disparity computation
Advantages	Fast and efficient implementation	High quality results Easy to incorporate constraints
Drawbacks	Window size and shape determination is difficult Less accurate	Complicated Slow

Global matching methods have the advantages of being capable of producing high quality results, and ease of incorporating new constraints regarding the desired disparity map into the cost function. Figure 2.7 shows the disparity maps produced by applying two global matching algorithms (i.e. BP and GC) and a local matching method (sum of squared differences + mean filter) to tsukuba and venus image pair taken from Middlebury stereo webpage [48]. Results of dynamic programming approach are also included. One can see that due to considering only one scanline at a time, this method has difficulty maintaining consistency of results between image rows. A problem typically referred to as "streaking effect". Comparing the obtained results to the ground truth, it is clear that global matching methods are able to properly recover fine details of the scene whereas local matching methods have more difficulty dealing with object boundaries and textureless regions and therefore produce less reliable results.

The major drawback of these methods is that global optimization is typically a complicated and time-consuming process based on iterative probabilistic inference methods. Therefore the performance of stereo matching systems based on global methods is usually far from real-time unless implemented on specialized hardware. Although, some work has been reported on successful implementation of real-time stereo vision systems based on 1D optimization methods such as dynamic programming [8]. Real-time stereo vision is further

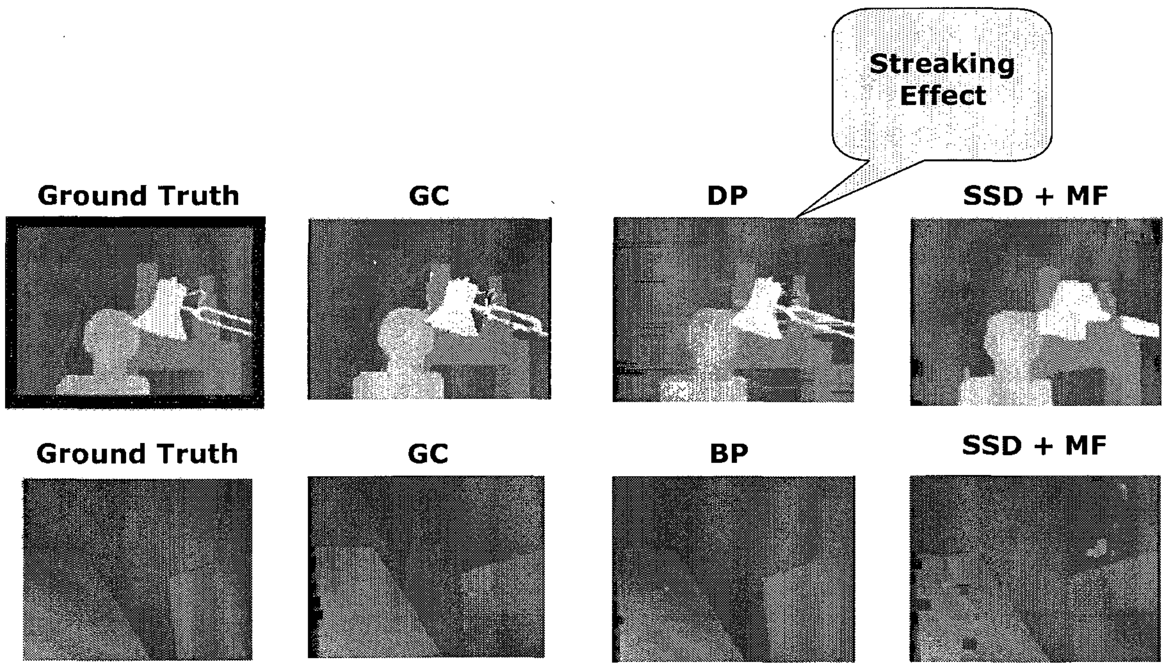


Figure 2.7: Qualitative comparison of local vs. global matching methods

discussed in chapter 3. Table 2.1 provides a brief comparison of local and global matching algorithms highlighting the main features for each group.

2.2.4 State of the Art

As mentioned in section 1.1, stereo vision research community has experienced significant contributions, mainly throughout the last couple of years, and many new high performance matching methods now exist in the literature. Most of these top-notch methods are global matching methods that deploy well-defined and robust cost (energy) functions along with an efficient optimization (inference) algorithm to compute the disparity map. Following is a summary of the main features/assumptions found in state of the art methodologies:

- Input image pair is assumed to be rectified
- Most of the methods have only considered static scenes
- The overall accuracy of top-performing methods is quite good (90% or more)

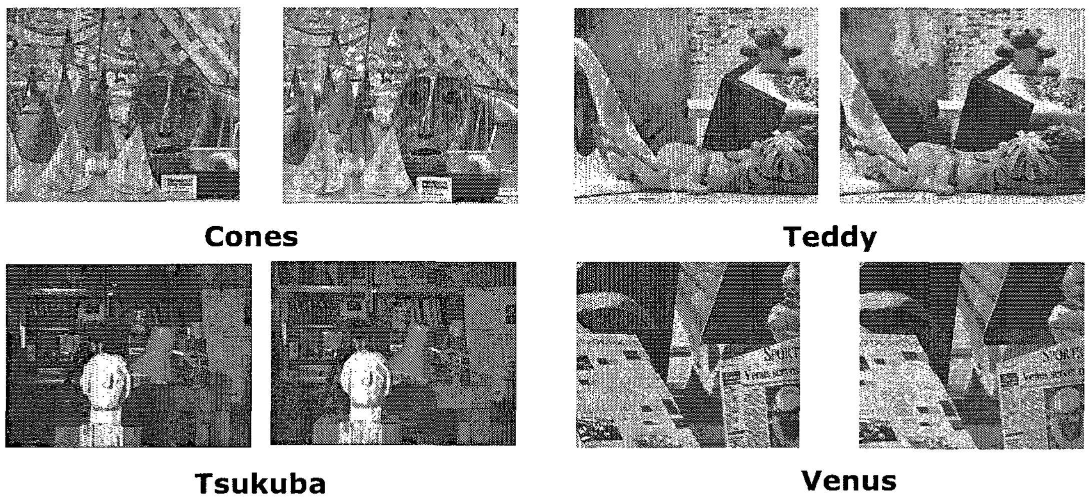
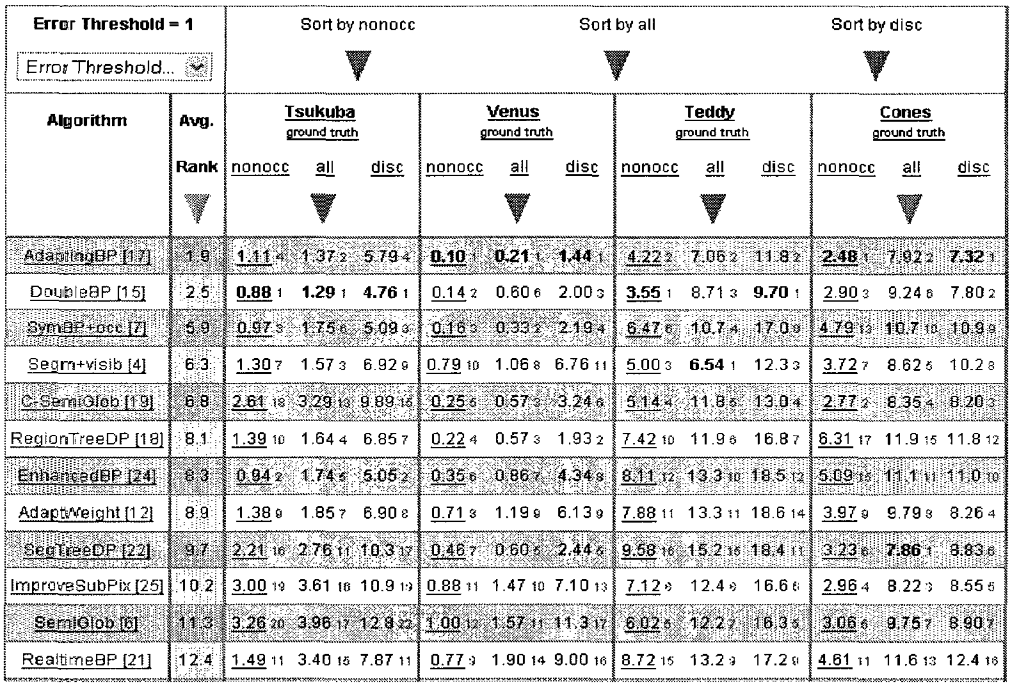


Figure 2.8: Test image pairs used in the Middlebury ranking of stereo matching methods

- Depth discontinuity areas are the most challenging regions resulting in highest error rates
- Most of top-performing methods rely on Bayesian belief propagation as inference method
- Recent methods attempt incorporating color segmentation data, enhanced similarity metrics, and cost functions to improve results
- Most of the them are computationally expensive and therefore relatively slow

A good representative of the stereo matching state of the art methodologies is the ranking of best performing algorithms based on the taxonomy developed to evaluate their performance quantitatively [48]. Figure 2.9 illustrates the latest ranking of stereo matching methods presented on Middlebury college stereo webpage. As shown the rank of an algorithm depends on the average error of the algorithm when applied to four different test image pairs, namely, Teddy, Cones, Venus, and Tsukuba (see Figure 2.8). Error percentages are computed for three categories, discontinuity regions (disc), non-occluded regions (nonocc), and all pixels of the image (all).



Error Threshold = 1

Error Threshold...

Sort by nonocc · Sort by all · Sort by disc

Algorithm	Avg. Rank	Tsukuba nonocc	Tsukuba all	Tsukuba disc	Venus nonocc	Venus all	Venus disc	Teddy nonocc	Teddy all	Teddy disc	Cones nonocc	Cones all	Cones disc
AdaptingBP [17]	1.9	1.11 4	1.37 2	5.79 4	**0.10** 1	**0.21** 1	**1.44** 1	4.22 2	7.06 2	11.8 2	**2.48** 1	7.92 2	**7.32** 1
DoubleBP [15]	2.5	**0.88** 1	**1.29** 1	**4.76** 1	0.14 2	0.60 6	2.00 3	**3.55** 1	8.71 3	**9.70** 1	2.90 3	9.24 8	7.80 2
SymBP+occ [7]	5.9	0.97 3	1.75 6	5.09 3	0.16 3	0.33 2	2.19 4	6.47 6	10.7 4	17.0 9	4.79 13	10.7 10	10.9 9
Segm+visib [4]	6.3	1.30 7	1.57 3	6.92 9	0.79 10	1.06 8	6.76 11	5.00 3	**6.54** 1	12.3 3	3.72 7	8.62 5	10.2 8
C-SemiGlob [19]	6.8	2.61 18	3.29 13	9.89 15	0.25 5	0.57 3	3.24 6	5.14 4	11.8 5	13.0 4	2.77 2	8.35 4	8.20 3
RegionTreeDP [18]	8.1	1.39 10	1.64 4	6.85 7	0.22 4	0.57 3	1.93 2	7.42 10	11.9 6	16.8 7	6.31 17	11.9 15	11.8 12
EnhancedBP [24]	8.3	0.94 2	1.74 5	5.05 2	0.35 6	0.86 7	4.34 8	8.11 12	13.3 10	18.5 12	5.09 15	11.1 11	11.0 10
AdaptWeight [12]	8.9	1.38 9	1.85 7	6.90 8	0.71 8	1.19 9	6.13 9	7.88 11	13.3 11	18.6 14	3.97 9	9.79 8	8.26 4
SegTreeDP [22]	9.7	2.21 16	2.76 11	10.3 17	0.46 7	0.60 5	2.44 5	9.58 16	15.2 15	18.4 11	3.23 6	**7.86** 1	8.83 6
ImproveSubPix [25]	10.2	3.00 19	3.61 16	10.9 19	0.88 11	1.47 10	7.10 13	7.12 8	12.4 8	16.6 6	2.96 4	8.22 3	8.55 5
SemiGlob [6]	11.3	3.26 20	3.96 17	12.8 22	1.00 12	1.57 11	11.3 17	6.02 5	12.2 7	16.3 5	3.06 5	9.75 7	8.90 7
RealtimeBP [21]	12.4	1.49 11	3.40 15	7.87 11	0.77 9	1.90 14	9.00 16	8.72 15	13.2 9	17.2 9	4.61 11	11.6 13	12.4 16

Figure 2.9: Latest ranking of state of the art stereo matching algorithms [48]

The percentages of mismatches (erroneous matches) B for each category is computed by counting the number of pixels for which the estimated disparity values differ from the correct one (determined by the ground truth) by more than a threshold value δ_d [12] as shown below:

$$B = \frac{1}{N}\sum_{(x,y)}(|d_C(x,y) - d_T(x,y)| > \delta_d)$$

Where N is the total number of pixels belonging to the region of interest, and $d_C(x,y)$ and $d_T(x,y)$ are the computed and true disparity maps, respectively. It can be seen that the error percentages are higher for teddy and cones image pair as they are more challenging and include more complex objects and larger disparity range . Regarding the results presented in Figure 2.9, it can be concluded that top-performing stereo matching methods are all global methods mostly relying on belief propagation for disparity computation and their associated accuracy is quite satisfactory. This probably demands for more challenges test image data that enables stereo vision researchers to assess their algorithms under more difficult situations, identify the major shortcomings, and attempt to resolve them.

Chapter 3

Real-time Stereo Vision

3.1 Fast Stereo Matching

Numerous stereo matching algorithms with real-time or near real-time performance exist in the stereo research community. These algorithms can be generally classified into two groups: area-based methods and feature-based methods. Feature-based methods operate by extracting viewpoint invariant features (e.g. edge, corner, and line segment) from the image and trying to match them. They have very fast and efficient implementation, however, can only yield a sparse disparity map. Area-based methods calculate disparity map using local image correlation over a certain area (window). They are usually preferred as they can find a dense disparity map in contrast to the feature-based methods.

Most of the dense real-time stereo vision systems in the literature rely on local (area-based) matching methods or some specific global matching methods such as dynamic programming to achieve an acceptable performance level. Global matching methods such as Belief propagation (BP) or Graph cuts (GC) based techniques are typically too computationally demanding to be applicable in real-time stereo and be implemented on affordable hardware. Although some recent preliminary work in this area seems promising [49]. Dynamic programming (DP) is the most common global matching method that is used to

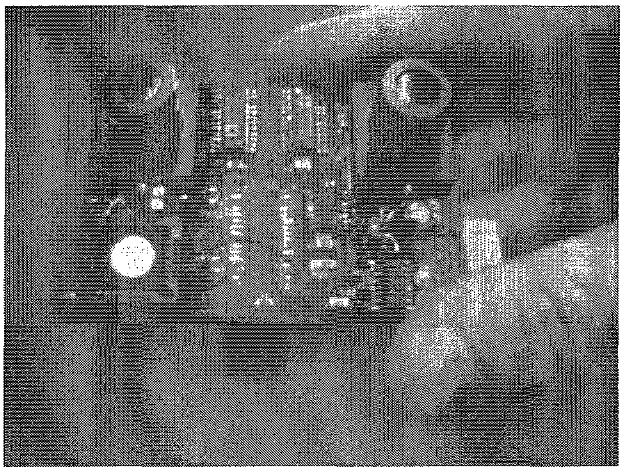


Figure 3.1: SRI small vision module [56]

realize high quality yet real-time stereo vision systems and algorithms [50, 51, 52]. As mentioned in section 2.2.3, DP has the advantage of performing the optimization in a one-dimensional manner (along the scanline) in contrast to the more complex two-dimensional optimization of BP and GC based methods. Nevertheless, as previously discussed it has drawbacks such as suffering from erroneous horizontal strokes (streaking effect) as well as being unable to deal with vertical displacement between input images [50]. On the other hand, local matching methods especially correlation-based matching techniques have already proven to be a feasible solution for real-time stereo in various software [53] and hardware applications [54, 55].

3.1.1 Correlation-based Matching

As mentioned in the previous section, correlation-based stereo has a long history of being used as real-time stereo matching solution in numerous applications. It consists of four main steps;

1. Use some dissimilarity measure to compute pixel-wise matching cost (Manhattan and Euclidean distances are the most commonly used measures deployed in this step).
2. Aggregate matching cost over correlation window along the epipolar line within the

search window.

3. Determine initial disparity map by finding the best disparity candidate through Winner-Take-All (WTA) strategy.

4. Post-process obtained results to remove potential outliers or produce sub-pixel disparity estimates.

While some algorithms in this area try to improve the accuracy of the correlation-based stereo through incorporating constraints such as ordering, uniqueness, and visibility into the matching process [37, 57], more recent methods take a more systematic approach and develop more advanced pixel dissimilarity metrics and/or cost aggregation methods to enhance the overall performance of the matching algorithm in problematic areas such as poorly textured regions and disparity discontinuity regions [39, 24, 58, 23].

Performance of the correlation-based matching algorithms is highly dependent on the cost aggregation approach deployed. Indeed several new cost aggregation methods have been proposed in the literature that produce higher quality results than many global matching methods [?, 60, 39]. Although, some of these methods can not achieve real-time performance. A recent study provides a comparison of several aggregation methods in terms of both computational cost and quality of disparity maps produced [59]. The conclusive remakes presented, serve as useful guidelines for developing novel real-time stereo vision systems. For example, it is shown that sampling insensitive matching method [36] and shiftable windows, commonly used in offline stereo matching algorithms, are of no benefit for real-time applications. It is due to the fact that for the sake of the processing speed, majority of real-time stereo algorithms represent matching costs as single bytes. As a result, small cost differences can not be distinguished due to the limited available precision.

Some of the most prominent work in the field of real-time correlation-based stereo matching is briefly discussed in the following. Most of these methods attempt to improve upon the quality of matching algorithm while maintaining its real-time efficiency. Hirschmuller proposes three approaches designed to enhance the obtained disparity map [23]. First, a novel multiple window approach is presented that reduces errors on object boundaries. Also,

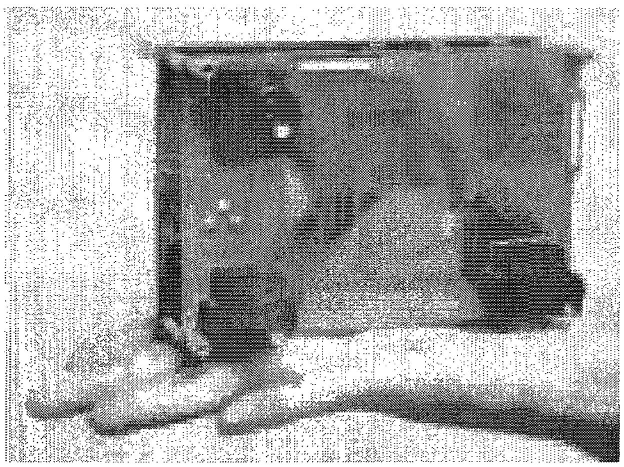


Figure 3.2: Miniature Stereo Vision Machine based on FPGA (MSVM III) [3]

a border correction method is introduced that further improves results on object boundaries. Finally, a general error filter based on correlation function is described that allows for detection and removal of uncertain matches. Authors show that integration of these methods into the matching process reduces the error by 50% while still maintaining the real-time suitability of correlation-based algorithms. Madain et al. propose a correlation-based method in which the neighbor pixels not similar to the pixel of interest (centeral one) are not considered for computing the correlation value. This equals to adjusting the shape and size of the correlation window adaptively based on local content. They argue that their method is suitable for real-time implementation using specialized hardware. Furthermore, the experimental results provided show that their method is capable of producing stable results in textureless areas and also alleviates the blurring effect at object boundaries [24]. Gong and Yang developed an image-gradient-guided cost aggregation scheme inspired by the success of stereo matching methods based on color segments. The proposed scheme specifically targets the new generation of graphics processing units (GPUs) and is designed to fit their architecture. As a result, their new algorithm achieves remarkable improvement in terms of accuracy (especially for depth discontinuity regions) without sacrificing real-time performance [[25]]. Another interesting work is by Hirschmuller, which presents a new stereo matching method based on mutual information using semi-global optimization [26].

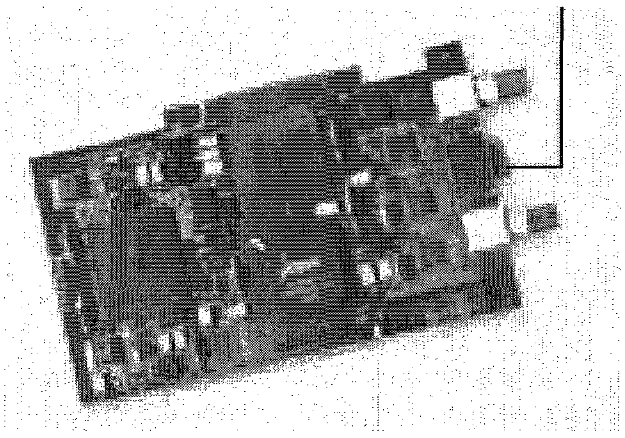


Figure 3.3: The Acadia-I PCI vision accelerator board [71]

The proposed algorithm objective is to develop a stereo matching method that produces accurate results (especially at object boundaries), is invariant to illumination changes (due to application of the mutual information), and can be implemented efficiently. In their paper, authors first introduce a hierarchical scheme for computation of mutual information based matching, which makes it virtually as fast as intensity based matching. Then, propose a semi-global matching method, which is basically an approximation of global cost calculation with linear complexity with respect to the number of pixels and disparities.

3.2 Previous Work

There are many stereo matching algorithms that claim to be capable of producing robust results within the constraints of a real-time stereo vision system. In fact, due to the significant advances in the processing power of commercial hardware available on the market, many new stereo vision systems operating at real-time and near real-time have been lately presented in the literature. A wide variety of hardware platforms including CPUs, Graphics Processing Units (GPUs), FPGAs, and Digital Signal Processors (DSPs) have been used to develop these stereo vision machines. Examples of such systems are Small Vision Module (SVM) developed at Stanford research institute (SRI) [56] that is based on DSP and achieves 8fps, and MSVM III that uses FPGAs and is capable of processing image pairs at up to 60 fps [3] (see Figures 3.2 and 3.1). Some researchers have also attempted utilizing the VLSI technology for developing application specific integrated circuit (ASIC) based vision systems. Acadia vision processor is an example of such systems illustrated in Figure 3.3. However, due to their high cost, lack of flexibility, and rather time-consuming and complicated design process, such systems are rarely implemented. Several PC-based stereo vision systems have also been recently reported that achieve their efficiency through incremental calculation schemes aimed to avoid redundant calculations and parallelizing the computationally expensive portion of the algorithm with Single Instruction Multiple Data (SIMD) parallel instructions, available nowadays in almost any state-of-the-art general purpose microprocessors [5] or utilizing the vector processing capability and parallelism in commodity graphics hardware to speed up the matching process [8].

The majority of aforementioned systems focus on raw performance and not on the size and power demand, which are crucial in embedded settings and applications like miniaturized mobile robotics.

Chapter 4

Miniaturized Embedded Stereo Vision System (MESVS)

4.1 System Overview

This chapter describes the design and implementation of a new miniaturized and fully integrated embedded stereo vision system (MESVS) developed for this thesis. The MESVS vision system is compact, power-efficient and of low-cost and is based on the state of the art dual core embedded media processor as computational platform. The main aim of this work has been to develop a small, efficient, real-time, and intelligent stereo-vision sensor that can be applied in a wide variety of imaging applications in real-world environments. The MESVS shown in Figure 4.1, fits within a tiny package of only 5x5cm, is capable of producing accurate and quality disparity maps at 20fps while operating at 600MHz per core. Furthermore, it is quite power-efficient and consumes very low power (i.e. 700mA@3.3V).

The system firmware consists of five main steps as follows:

1. Image sub-sampling

2. Stereo rectification

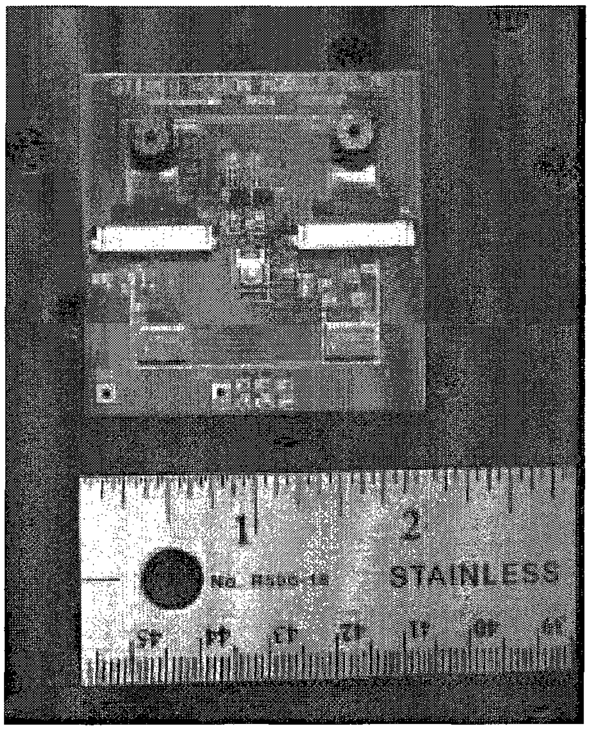


Figure 4.1: Miniaturized Embeddded Stereo Vision System developed for this work

3. Pre-processing step based on local non-parametric transforms

4. Correlation-based matching with three levels of recursion followed by cross checking to detect and remove half-occluded regions of the scene

5. Novel postprocessing algorithm to improve obtained results around object boundaries and reduce number of mismatches caused by low-texture areas

The high level of performance, quality and accuracy of MESVS firmware was achieved through efficient implementation of the algorithms, improved memory and data management, in-place processing scheme, careful code optimization, and deploying pipe-lined programming model, that leverages the advantages of dual-core architecture of the embedded vision processor used.

In the following sections, first the details of hardware implementation of the system is explained and then the stereo matching engine developed considering special constraints and limitations of embedded settings and also design aims of the system such as real-time performance and high accuracy are discussed.

4.2 Hardware Description

4.2.1 Horopter vs. range accuracy analysis

As mentioned before the major focus of this work has been on keeping the overall dimensions, power consumption and cost of the system as small as possible, while improving the accuracy and performance. One of the impacts of miniaturizing the dimensions of the system is on the baseline, i.e. the distance between cameras centers of projection. Regarding the system dimension of only 5x5 cm and based on results of the following analysis, the baseline of about 28mm was chosen for the system. In fact, the small focal length of the compact camera modules along with the small baseline, necessitates analyzing the resulting effect on the range of measurable objects in the scene (Horopter) and also the corresponding accuracy (range uncertainty) to evaluate the feasibility of such a configuration. Technically, the horopter must be large enough to encompass the range of objects in the application. Furthermore, the corresponding range accuracy associated with the obtained disparities within the specified horopter must be acceptable.

As shown in Equation 4.1 the range uncertainty Δr is proportional to the range r, baseline b, focal length f, pixel size x, and the change in disparity ΔN. Regarding a baseline of 28mm, focal length of 28mm, and pixel size of 17um, setting system horopter to 5-35 pixels with disparity range of 30, yields a measurable range of about 15-100 cm (see Figure 4.2).

$$\Delta r = (\tfrac{r^2}{bf})x\Delta N \tag{4.1}$$

Also the range data can be extracted from disparities through triangulation as shown in Equation 4.2 below (N is the disparity value).

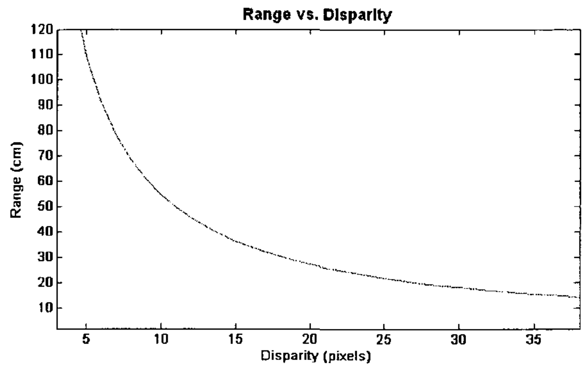


Figure 4.2: Analysis of range versus corresponding disparity

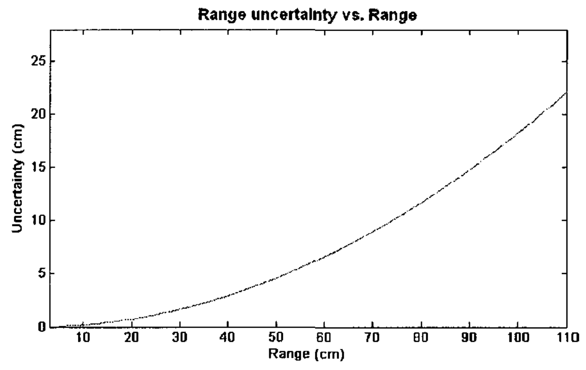


Figure 4.3: Analysis of range uncertainty versus corresponding disparity

$$r = \frac{bf}{Nx} \tag{4.2}$$

Figure 4.3 illustrates the diagram of range uncertainties vs. range for the available range of interest (i.e. 15-100cm). As shown for objects within close vicinity of the system (less than 50cm) the maximum uncertainty is about 5cm. The conclusion is that through a well-adjusted horopter along with a large enough disparity range, it is possible to compensate for the effect of the small baseline and compute range data with acceptable level of accuracy.

4.2.2 Choice of computational platform

Another important item that was carefully chosen is the hardware technology used as computational platform as it heavily influences the cost, performance, and the power consumption characteristics of the final system. Following is a list of the criteria taken into account for choosing the appropriate computational platform for this work:

- Performance characteristics (i.e. processing speed, memory architecture and features, data buses, power requirement, benchmark results, etc.)
- Development considerations (i.e. single vs. multi-core architecture, number and variety of I/O ports and peripherals supported, instruction set features, developer familiarity and learning curve, compatibility, customer support, etc.)
- Cost of integration
- Availability and future roadmap
- Packaging considerations

Current available options that are common to the computer vision community are FPGA, ASIC (VLSI technology), CPU, GPU, embedded processors, media processors, and ASSPs (Application Specific Signal Processors). Because the stereo vision system is going to operate in an embedded settings where power efficiency is crucial , CPU and GPU options are ruled out as they are typically power demanding and not designed for such configurations. Choosing ASIC has the advantage of being exactly tailored to the desired features of the system. However, it requires VLSI design expertise and is difficult and time-consuming to design, inflexible and usually expensive. Media processors such as ASIPs supply higher performance than most DSPs and GPPs (General Purpose Processors). They also provide enhanced functionality for video processing; however, they have complex programming models, higher developmental cost, and higher associated risk as their roadmaps are not clear [20]. ASSPs incorporate one or more processor types that are well matched to the application, and thus offer excellent performance, and energy efficiency. Nevertheless, they are typically inflexible, have a sharp learning curve and require extensive tuning. Their roadmap is also unclear and the benefits of low cost can only be realized when mass produced. FPGAs may seem like a promising alternative as they can be reconfigured dynamically, offer architectural flexibility, high throughput and performance; however, they have drawbacks such as requiring design skills beyond that found in many vision labs today [61], they also consume more power, cost higher [56] and have larger footprint when compared to state-of-the-art embedded microprocessor solutions. Therefore, although FPGAs offer higher computational power, our embedded vision system is based on an embedded processor with integrated DSP functionality as the processing unit mainly due to its lower power consumption and compact size, which are of outmost importance regarding the design constraints of the system mentioned before.

4.2.3 Hardware implementation

As explained in the previous section, embedded processor is chosen as the computational platform of the MESVS. Embedded processors can be subdivided into microprocessors and microcontrollers. Microcontrollers are typically just not fast enough for real-time vision applications as stereo vision is a computationally demanding task. Embedded microprocessors (also referred to as DSP) may be roughly categorized into three groups: low cost fixed-point, high performance fixed-point, and floating-point processors [62]. In general, low cost fixed-point DSPs are not particularly fast; they operate at modest clock speeds (typically 350 MHz or less) and are mostly single-MAC (Multiply-Accumulate) devices that closely resemble the traditional DSP architectures of the early 1990's [62]. Floating point DSPs such as Analog Devices SHARC and TigerSHARC families have more complex circuitry and therefore are not as energy-efficient as high performance fixed-point processors. Although they have the advantage of ease of programming due to their improved dynamic range and one does not have to worry about scaling the signal in order to avoid exceeding the dynamic range of the processor. Finally, high performance fixed-point processors provide a fast yet power-efficient processor within an affordable price range. Most of them support multiple instructions execution using VLIW (very long instructions word) techniques, and also multiple MACs per cycle. Therefore, they provide the best compromise between ease of programming and performance versus cost and energy efficiency.

There are three main vendors of high performance fixed-point DSP processors: Analog Devices, Texas Instruments, and Freescale. Analog Devices ADSP-BF5xx (Blackfin), Freescale's MSC81xx and MSC71xx, and Texas Instruments TMS320C64x family of processors are the three main competitors in the area of high performance fixed-point DSPs [62]. Although Freescale's MSC81xx and MSC71xx and Texas Instrument's TMS320C64x family of processors provide high performance levels (clock speed of up to 1GHz), they are not originally designed for multimedia applications and therefore do not support video capture and processing facilities found in Blackfin processors. Additionally, they are not as power-efficient as Blackfin processors and cost more on average.

i.MX processors from Freescale and DaVinci processors from Texas Instruments are comparable to Blackfin processors in the sense that both are designed for embedded multimedia applications and comprise in-built camera interface and video processing and enhancement sub-systems. The processors that support a digital camera interface have an advantage in terms of performance because the core processor does not have to care about low-level video synchronization and transfers. These interfaces all have DMA (Direct Memory Access) capability, which is the only efficient way of handling the huge amount of video data with no intervention of the core processor. This way, core processor can analyze a frame in the main memory while the DMA can grab the next frame and move it to the main memory [63]. i.MX processors only support one camera interface and therefore may not be used to implement stereo vision system, which requires at least two interfaces. DaVinci processors supply more processing power than Blackfin family (up to 7200 MMACs@900MHz), yet they cost more and also have higher power consumption and larger dimensions. As a result, Blackfin family of processors is the best choice regarding the aforementioned design requirements of MESVS.

As shown in table 4.1, the candidate DSP from Blackfin family, i.e. ADSP-BF561, consumes less power and is cheaper and more compact in contrast to its counterpart processor from Texas Instruments DaVinci family. Although the Texas Instruments embedded processor offers higher computational power (based on higher MMACs) yet it costs twice

Table 4.1: Comparison of the embedded processors

DSP Model	Blackfin ADSP-BF561	TMS320C6418-600
Power estimate (mW)	1100	1850
RAM Memory (Kbytes)	328	548
Camera interface	2 (ITU-656 compatible)	2 (ITU-656 compatible)
Architecture	16 bit fixed-point	32 bit fixed-point
MMACs (Max)	2400	4800
Price range (1k)	27 USD	55 USD
Clock frequency (MHz)	600x2	600
Packaging	256-Pin MBGA	548-Pin PBGA
Core Voltage (V)	0.8 - 1.2	1.4
Dimensions (W x L)	12 x 12mm	23.1 x 23.1mm
Process technology (um)	0.13	0.13

as much, occupies almost four times more space and requires more power. The processors have some features in common such as supporting two camera interface ports compatible with ITU-R656 video signaling standard, being based on 0.13um CMOS technology, and operating at up to 600MHz. Considering the major requirements of our design, i.e. power efficiency, compactness and low cost, ADSP-BF561 has been chosen as the processing unit of our system.

Figure 4.4 illustrates a high-level block diagram of the miniaturized embedded stereo vision system measuring only 50x50mm including two small camera modules. The heart of the system is the ADSP-BF561 dual core processor, one of the latest members of 16bit embedded processors from Analog Devices featuring dual Blackfin cores with each core capable of 1200 MMACs@600 MHz (2400 MMACs total) and suitable for demanding still very space and power limited signal processing applications. The ADSP-BF561 processor possesses a flexible parallel port interface (PPI), which can be configured to support the ITU656 standard by hardware. One of the other useful features of the Blackfin processor is its powerful direct memory access (DMA) capabilities, which are used extensively to avoid

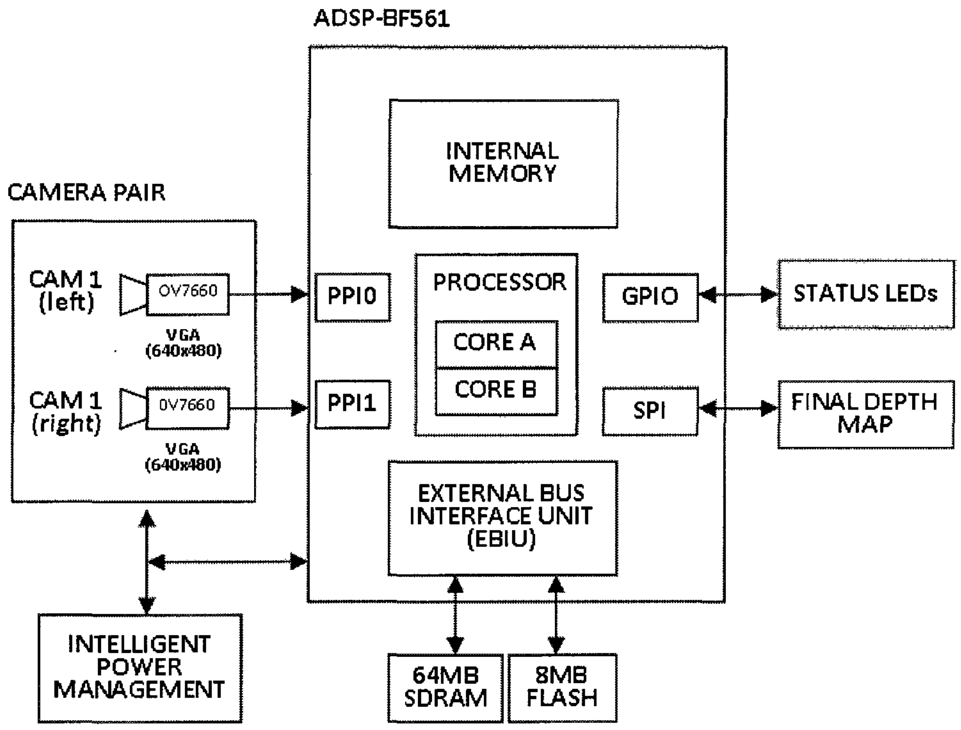


Figure 4.4: High level block diagram of MESVS

bottlenecks in video data movements within the system and keep the core processors away from performing slow memory accesses. The MESVS also includes 64 MB of SDRAM, 8 MB of addressable flash memory, and dynamic core voltage regulator that allows adjusting core voltage through software. Furthermore, system board is hosting two camera modules and their corresponding oscillators, and voltage regulation circuitry to allow standalone operation of the system. A JTAG interface is also incorporated used for debugging and downloading system firmware. Finally, the SPI port is used to enable connection of the embedded stereo vision system to the other units (modules) within an embedded setting.

The camera module is the OV7660-FSL CMOS camera from OmniVision designed for mobile applications where low power consumption and small profile are of utmost importance. It measures only 6.5 x 6.5 x 4.8mm, requires only 40 mW in full operation mode, and incorporates a DSP functionality with digital ITU656 interface support [64]. The in-built

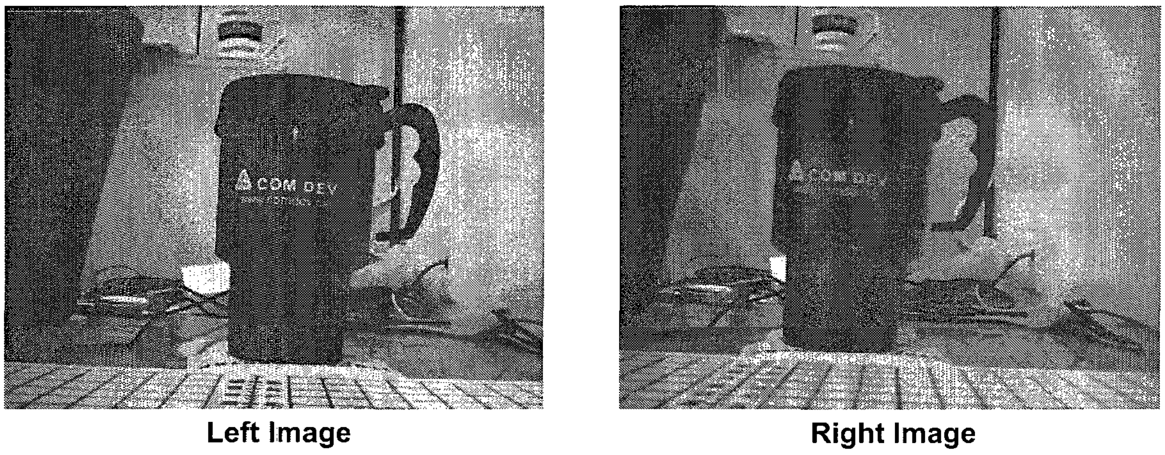


Figure 4.5: Sample stereo snapshots captured by the CMOS camera modules

DSP of the camera provides some basic image processing functionality such as auto exposure and white balance control, color and gamma correction, and color space conversion, e.g. from RGB to YUV color space. ITU656 is a digital video protocol that requires video timing signals such as horizontal and vertical synchronization signals to be embedded as reference codes within video data stream. This in turn results in simpler and more robust video capture process. The camera module was set to supply the graphic data in the YUV 4:2:2 format. To reduce the image size and the computational burden, only the luminance channel Y data is used and the captured images are stored as grayscale. Figure 4.5 shows sample binocular snapshots taken by the system.

4.2.4 Memory architecture

Due to limited memory resources in an embedded setting, memory and data management plays a crucial role in enhancing system performance. Blackfin family of embedded processors is based on modified Harvard architecture (separate data and code memories) along with a hierarchical memory structure. An overview of the memory architecture of the MESVS is shown in Figure 4.6. Each Blackfin processing core has access to 100KB of high-

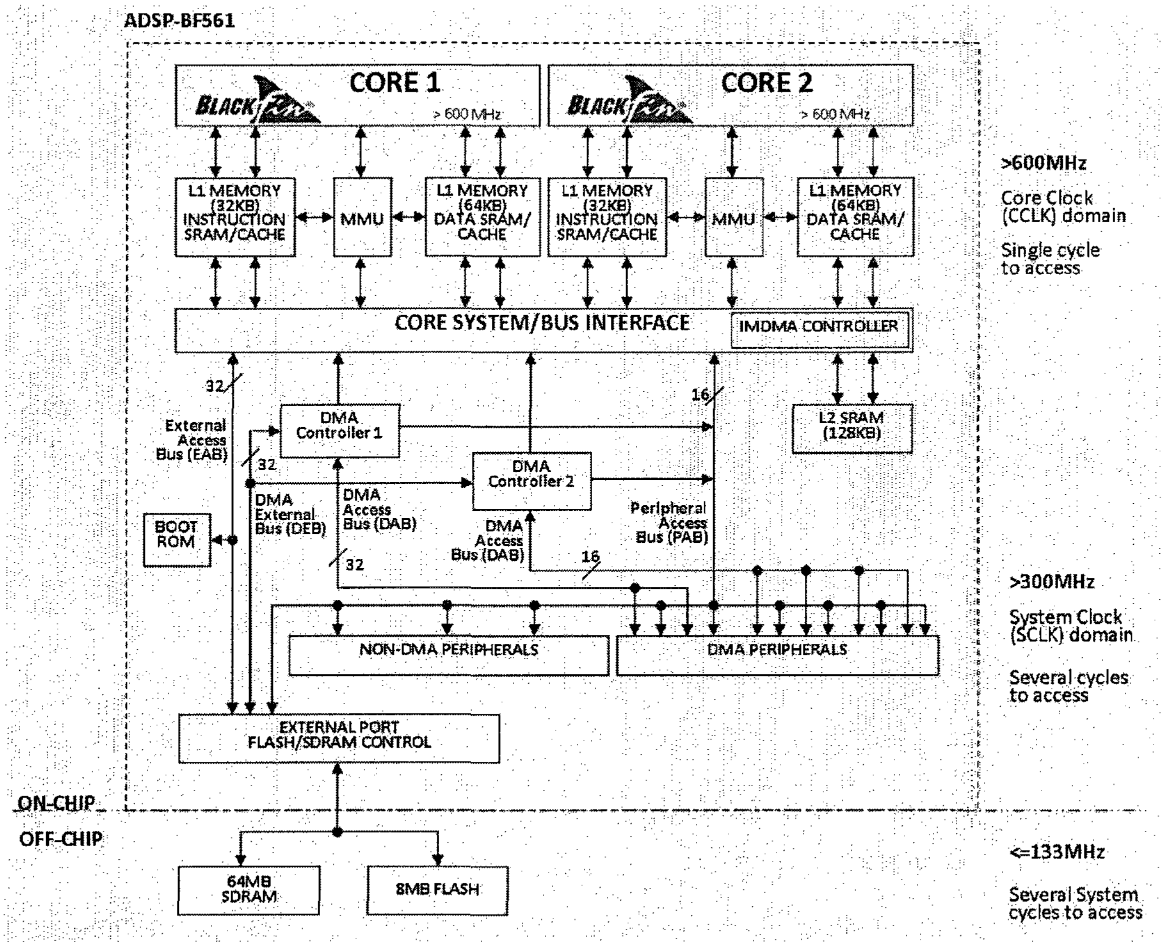


Figure 4.6: Overview of memory architecture of the MESVS

speed, high-performance, low-latency, Level 1 (L1) memory typically operating at the same speed as the processing core. The L1 memory consists of 64KB of data dedicated and 32KB of instruction dedicated memory. There is also 4KB of extra memory per core configured as scratchpad (omitted in Figure 4.6 for simplicity). The second memory unit is a unified Level 2 (L2) memory shared between both of the processing cores and operating at approximately half of the core-clock speed (up to 300MHz). As a result L2 memory accesses have a higher latency as compared to L1 memory. Memory Management Unit (MMU) defines the properties of a given memory space and protects the internal registers from unintended access for each processing core. External memory also referred to as Level 3 (L3) memory consists of

64MB of SDRAM and 8MB of flash memory. The systems firmware is burnt onto and kept on the flash memory through JTAG interface. The L3 memory is quite large. However, its access time is measured in System Clock Cycles (SCLK), which is usually much larger than the Core Clock Cycles (CCLK). Therefore, to assure optimum system performance, slow memory accesses to the external memory are mostly performed using DMA facility and majority of memory accesses of the processing cores is limited within the fast on-chip memory (L1 nd L2) of the system as explained in section 4.3.3.

Blackfin architecture supports several DMA channels that facilitate data movement between the peripherals and also within memory unit, without causing overhead to the processing cores. The DMA features are extensively deployed in order to improve system performance and reduce the run-time of the stereo matching, and rectification algorithms as discussed in section 4.3.3.

4.2.5 Power consumption

In most of the embedded systems especially mobile platforms that rely on batteries as the sole source of energy, power management requires special consideration. Because stereo matching algorithms are computationally expensive, the system has to operate at maximum capacity (system clock) in order to achieve a real-time processing rate. In addition, the matching algorithm must be carefully selected and tuned, to take advantage of the dynamic power management. Blackfin processors have the ability to vary the voltage and frequency dynamically, which facilitates reduction of power consumption. Dynamic power consumption is directly proportional to the square of the operating voltage V_{DD} and linearly proportional to the operating frequency f, as shown in the Equation 4.3 [18, 21] below (K is the system constant):

$$P_{dynamic} = KV_{DD}^2 \cdot f \tag{4.3}$$

According to the Equation 4.3 one way to reduce the power consumption is lowering the frequency. However, this would only be useful when the system is already operating at the lowest voltage allowed. Because lowering frequency also means that it takes more time for the firmware to execute. MESVS incorporates power saving features such as intelligent voltage regulation, dynamic management of frequency, flexible power management modes (sleep, hibernate, full-on and active), and isolated power domains for components. Having multiple power domains, one for the internal logic circuitry of the processor and the other for I/O, improves flexibility and power savings of the system. External components such as L3 memory chip were carefully selected, taking their power consumption into account. In addition, the IDE used to develop system firmware, i.e. Analog Devices's VisualDSP++ suite features a Statistical Profiler tool that can quantify exactly how much time is spent in any given segment of the firmware. Using this tool, the major segments of system firmware were carefully tuned to maintain a balance betwwen power demands and performance. Regarding all this, the average active power dissipation of MESVS is estimated (through adding the active power dissipated in individual power domains) to be about 2.3W (i.e. 700mA@3.3V).

4.3 System Firmware

This section provides a detailed explanation of the development process taken to implement the firmware for MESVS. As briefly discussed in section 4.1, MESVS relies on a multi-step recursive correlation-based stereo matching algorithm to achieve high performance and quality results. In the next sections first the offline stereo calibration procedure is explained followed by an in-depth discussion of stereo matching engine (elaborating on each of the steps) in section 4.3.2. Finally, in section 4.3.3 the measures taken to enhance system functionality in order to attain real-time performance are presented .

4.3.1 Stereo calibration

To calibrate the stereo vision system, the camera calibration MATLAB toolbox from the Institute of Robotics and Mechatronics at Caltech [13] is deployed. This toolbox is a user-friendly, powerful, and well-documented software that supports a comprehensive internal

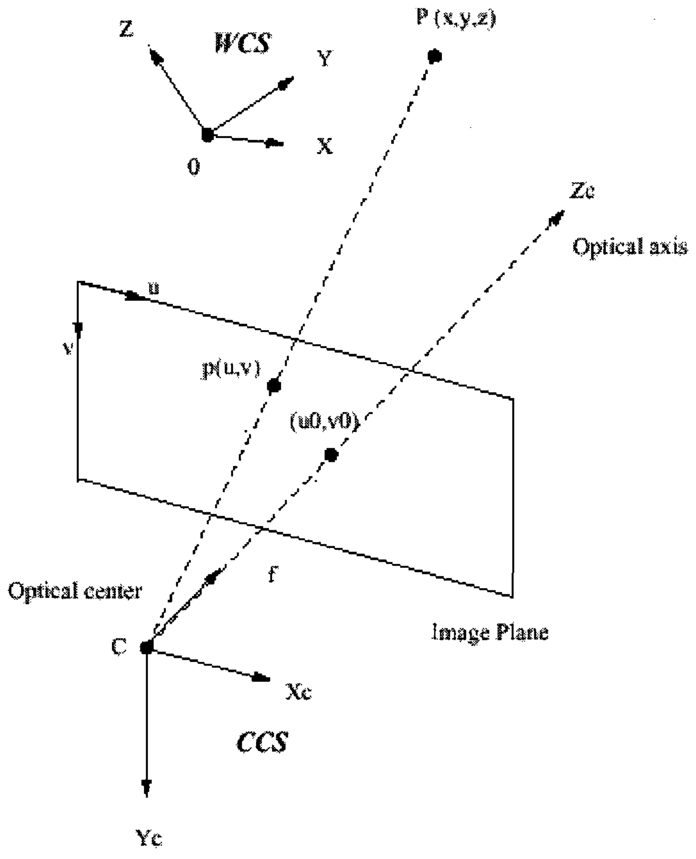


Figure 4.7: Pinhole camera model

camera model and therefore is capable of calibrating a wide range of camera systems. The camera model is very similar to that used by [65]. It is based on the pinhole camera model (see Figure 4.7) and considers both radial and tangential distortions using five coefficients (distortion model is also known as "Plumb Bob" [66]). Furthermore, it naturally supports non-square pixels and through estimation of skew factor allows the pixels to be even non-rectangular. In addition to calculating estimates for the intrinsic camera parameters such as focal length, projection center, pixel size, distortion parameters and extrinsic stereo parameters (i.e. rotation and transformation), the toolbox also returns estimates of the uncertainties associated to those parameters. User only has to specify the four extreme corners of the calibration pattern and software is able to extract the rest of the corners automatically, which in turn speeds up the calibration process. Figure 4.8 shows several snapshots of the calibration pattern used to calibrate the right camera module.

Through experiments, the stereo calibration process was found to be a difficult and rather time-consuming stage and very influential on the quality of the final disparity map obtained. In order to maximize the accuracy, we captured our calibration shots in the maximum resolution provided by our camera modules (i.e. VGA), which allowed to locate the four extreme grid corners in the most precise way possible. Moreover, the 3D positioning of the planar calibration pattern through the calibration shots must include a wide range of varieties along three axes (X, Y, and Z) and three orientations (pitch, yaw, and roll) to allow for external stereo parameters to be estimated with reasonable accuracy. Finally, it was proven experimentally that the radial distortion of the cameras must be compensated. Although, considering the tangential distortion in the model would slightly improve the results, it would considerably increase the computational complexity of the un-distortion algorithm. As a result, in the current implementation only radial distortion of the cameras is corrected in order to maintain a balance between performance and quality. One of the future avenues of research of this work would be to develop and implement an online self-calibration algorithm, which is necessary for the system to produce consistent and reliable results over the time.

4.3.2 Stereo matching engine

MESVS relies on an efficient and robust stereo matching engine to retrieve the depth information of the scene in real-time, i.e. about 20 fps, in the current implementation. This section provides an in-depth description of this stereo engine, focusing on challenges and required characteristics of the software dictated by both the hardware platform deployed and the embedded settings. As illustrated in Figure 4.9, there are five major stages that constitute the stereo matching engine, namely, image sub-sampling, stereo rectification, preprocessing, matching core, and post-processing. The stereo camera calibration is also initially performed in an offline manner as discussed in section 4.3.1. Following is an elab-

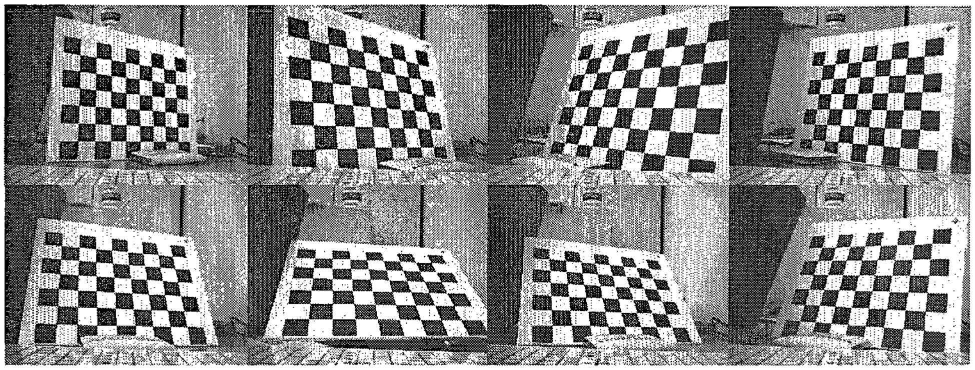


Figure 4.8: Sample snapshots of the calibration pattern used to calibrate the right camera module

oration on each of these five steps.

Image sub-sampling

One of the main challenges of programming for a vision application in an embedded setting is the very limited amount of fast on-chip memory available compared to the size of the image and video data. Although external memory may also be deployed to store and retrieve image data, it typically operates at much lower speed compared to the processing core and the high access latencies associated to it would make system performance to suffer. That is, processing cycles would be wasted waiting for the new data to be supplied by the slow external memory.

The original images produced by the cameras are in VGA resolution and require 300KB of storage space per image. As discussed in section 4.2.4, Blackfin memory architecture only provides 64KB of fast L1 memory per core that operates at 600MHz (same as the core clock speed) and 100KB of slower shared L2 memory that operates at 300MHz. There is also 64 MB of off-chip SDRAM memory available that is running at 133MHz and is several times slower than the on-chip memories. Therefore to ensure maximum performance and fit captured image pair within the on-chip memory of Blackfin, we sub-sample image pair into QQVGA (160x120) resolution. Sub-sampling is done efficiently using 2D DMA facility

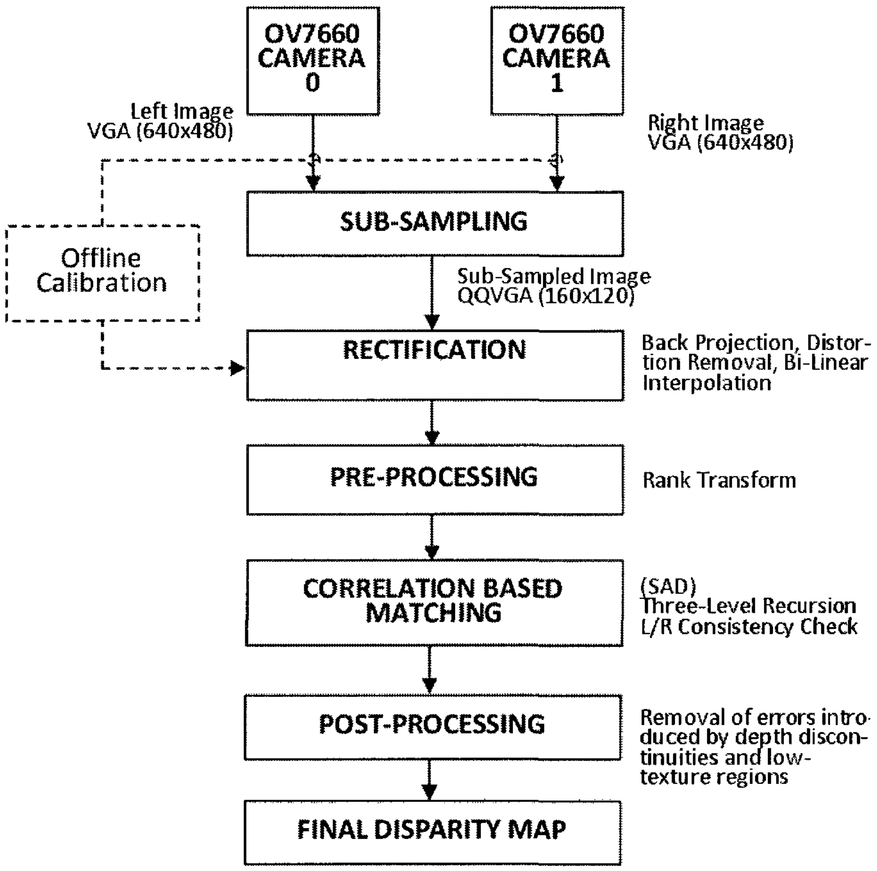

OV7660 CAMERA 0
OV7660 CAMERA 1
Left Image VGA (640x480)
Right Image VGA (640x480)
Offline Calibration
SUB-SAMPLING
Sub-Sampled Image QQVGA (160x120)
RECTIFICATION
Back Projection, Distortion Removal, Bi-Linear Interpolation
PRE-PROCESSING
Rank Transform
CORRELATION BASED MATCHING
(SAD) Three-Level Recursion L/R Consistency Check
POST-PROCESSING
Removal of errors introduced by depth discontinuities and low-texture regions
FINAL DISPARITY MAP

Figure 4.9: A step-by-step diagram of the stereo matching engine of MESVS

of the Blackfin processor by skipping every three rows and columns of the input image as shown in Figure 4.10.

Stereo rectification

Stereo rectification is the process in which the image planes of the left and right stereo images are transformed such that the pairs of conjugate epipolar lines become collinear and parallel to the horizontal image axes. Rectification is a very important and useful stage as it reduces the 2D correspondence search into a simpler 1D search along the epipolar lines. It constitutes back projection, image un-distortion, and bilinear interpolation steps. Listing 4.4 illustrates a simple pseudo-code for the back projection algorithm. As described

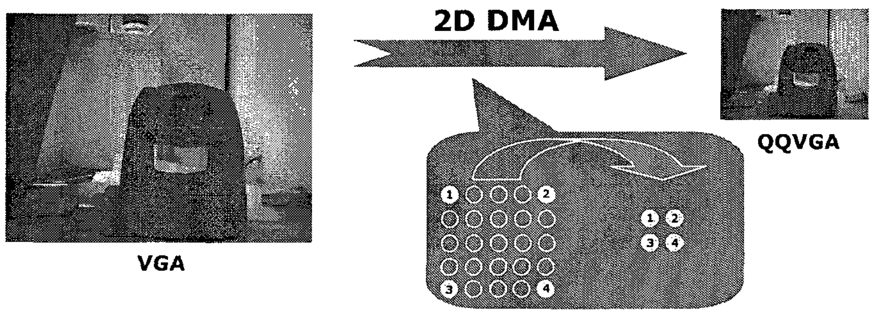


Figure 4.10: Image subsampling from VGA to QQVGA using 2D DMA facility

for each of the pixels in the image the back projected coordinate in the rectified image is computed through a matrix multiplication using homogeneous coordinate system and then is converted back to the regular (Cartesian) coordinate system. The elements of the *back projection matrix* are based on the parameters estimated in the calibration step and are real numbers (For more detailed explanation please refer to [67]). Therefore, this matrix should be represented using floating-point data meaning that implementing back projection step requires floating-point arithmetic.

(4.4)

```
1. $for\ i = 1\ to\ image\ height$
2.   $for\ j = 1\ to\ image\ width$
2.1      $rays = back\ projection\ matrix \cdot [j\ i\ 1]$
2.2      $back\ projected\ coordinate = [rays(1,1)\ /\ rays(3,1) \quad rays(2,1)\ /\ rays(3,1)]$
3. $end$
```

The main difficulty of implementing this step is the need for floating-point operations, which are not supported by the fixed-point embedded media processor used in MESVS. In fact, Blackfin supports floating-point operations through software emulation, which is available as a library implemented in C. However, it was found to be inefficient. A relaxed and faster assembly implementation of the basic floating-point arithmetic was deployed in order to facilitate optimum performance of the rectification algorithm.

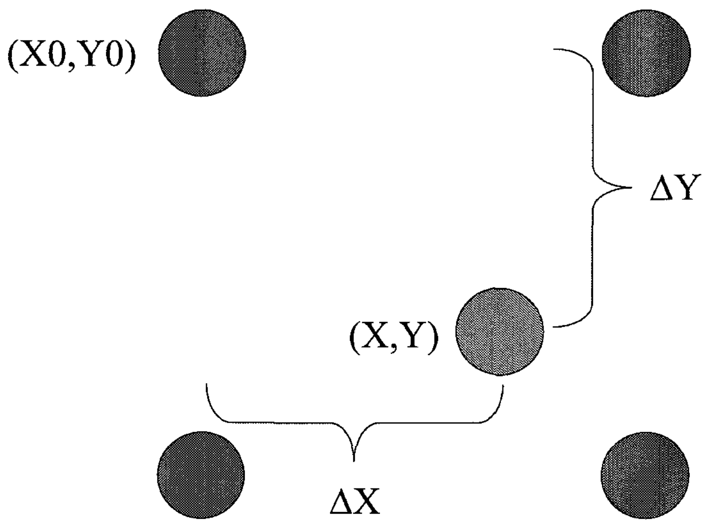


Figure 4.11: Bilinear interpolation process

The next step is the image un-distortion, which involves removal of the radial distortion of the cameras. The mathematical representation of this step is shown in Listing 4.5. The un-distortion step requires addition and multiplication of real numbers. In our implementation, we observed that range of data used in the un-distortion algorithm may be restricted to [-1 1]. Therefore, the fractional data type operations, which are natively supported by Blackfin and efficient, were used instead of the floating-point operations. This has resulted in additional improvement of the algorithm.

$$\begin{aligned}
&r_2 = back\ projected\ coordinate(1,1)^2 + back\ projected\ coordinate(2,1)^2\\
&r_4 = (r_2)^2\\
&r_6 = (r_2)^3\\
&offset = 1 + k_1 \cdot r_2 + k_2 \cdot r_4 + k_3 \cdot r_6\\
&undistorted_x = back\ projected\ coordinate(1,1) \cdot offset\\
&undistorted_y = back\ projected\ coordinate(2,1) \cdot offset
\end{aligned} \tag{4.5}$$

The last step is the bilinear interpolation as the $undistorted_x$ and $undistorted_y$ coordinates produced by the un-distortion process are typically non-integer. Thus, the intensity of the rectified pixel $I_{rect}(i, j)$ is determined considering contributions associated to the four neighbors (See Figure 4.11). There are four coefficients a_1, a_2, a_3, and a_4 computed as shown in Equations 4.6 to 4.9and used to perform the bilinear interpolation step (See Equation 4.10).

$$a_1 = (1 - \Delta X) \cdot (1 - \Delta Y) \tag{4.6}$$

$$a_2 = (\Delta X) \cdot (1 - \Delta Y) \tag{4.7}$$

$$a_3 = (1 - \Delta X) \cdot (\Delta Y) \tag{4.8}$$

$$a_4 = (\Delta X) \cdot (\Delta Y) \tag{4.9}$$

$$I_{rect}(i, j) = a_1 \cdot I(X0, Y0) + a_2 \cdot I(X0 + 1, Y0) + a_3 \cdot I(X0, Y0 + 1) + a_4 \cdot I(X0 + 1, Y0 + 1) \tag{4.10}$$

Similar to the back projection step, the bilinear interpolation algorithm is implemented using the relaxed and fast assembly implementation of the basic floating-point arithmetic. In the initial implementation, applying these strategies resulted in over 70% improvement in the execution time of the rectification algorithm and allowed the MESVS to achieve the frame rate of 10 [19].

In the second generation of the MESVS, the indexes computed in the un-distortion step (i.e. $undistorted_x$ and $undistorted_y$) and coefficients deployed in the bilinear interpolation step (a_1 to a_4) were computed once and buffered as look-up tables in the external memory. To avoid performance degradations caused by slow external memory accesses the double buffering technique along with DMA data transfers were deployed. This enhanced scheme resulted in 50% additional improvement of run-time for the rectification step. See section 4.3.3 for more details.

Preprocessing

The preprocessing step is incorporated to enhance the robustness of the matching algorithm especially with respect to the radiometric variations in the scene. The preprocessing algorithm was based on Rank transform in the first prototype of MESVS [19]. Rank transform is a form of non-parametric local transform, designed to be invariant to changes due to the difference of gain and bias between cameras. Rank transform R(P) of a pixel P is defined as the number of pixels $N(P)$ in the local region whose intensity $I(P')$ is less than the intensity of the central pixel $I(P)$ [14] (See Equation 4.11). In other words, Rank transform replaces the intensity of a pixel P with its Ranking among all the pixels within a predefined local neighborhood surrounding it.

$$R(P) = ||\{P' \in N(P) | I(P') < I(P)\}|| \tag{4.11}$$

Rank transform was chosen because according to the latest studies it outperforms other cost functions such as Laplacian of Gaussian (LoG), normalized cross correlation (NCC), mutual information (MI), and Mean filter once applied with correlation-based matching methods [15]. Furthermore, it could be implemented efficiently on an embedded media processor. The window size for the local neighborhood was set to 5 to provide a tradeoff between performance of the system and quality of the obtained results.

In the second generation of the system (MESVS-II) [20], Rank transform was replaced with Census transform (See Equation 4.12) to further improve the robustness of the system

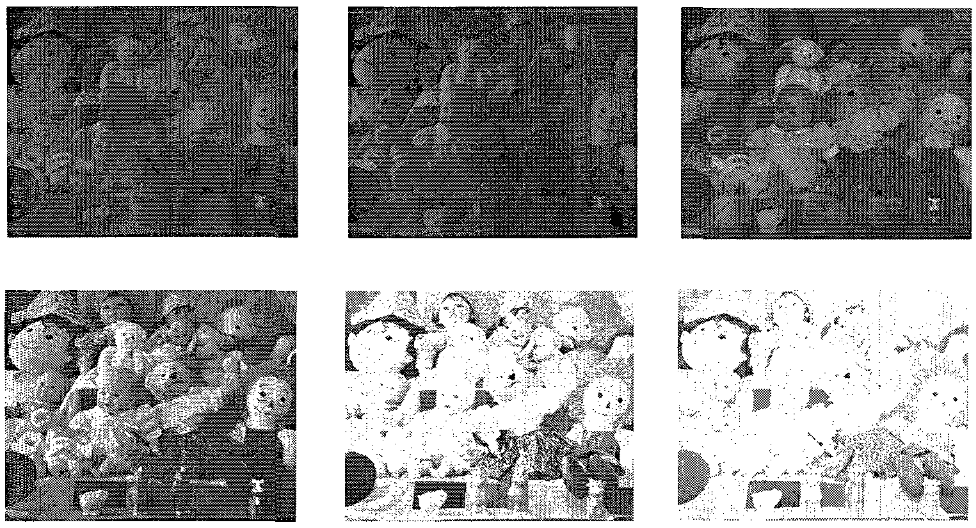


Figure 4.12: Sample shots with varying lighting conditions from Dolls dataset [15]

focusing on radiometric variations in the scene.

Rank transform has several drawbacks. It can not distinguish between rotations and reflections, and its results are the same for a variety of patterns [68]. These issues are caused by the fact that Rank transform loses spatial information regarding distribution of pixels surrounding the central pixel [69]. On the other hand, Census transform is capable of preserving the local structure of the image [14], as it encodes it into a bit stream. It is also proven to provide higher quality disparity results, due to lower mismatches as compared to the Rank transform [14, 69]. Equation 4.12 shows the mathematical representation of the Census transform R_T for a pixel P. The $\bigotimes$ represents the concatenation operator on a set of pixels with a set of displacements D within a square window. Also, the $\xi(P, P')$ is the comparison operator, which is equal to 1 when $I(P') < I(P)$ and 0 otherwise.

$$R_T(P) = \bigotimes_{[i,j]\in D} \xi(P, P + [i,j]) \tag{4.12}$$

Nevertheless, there is no work in the literature that provides a quantitative comparison of Rank versus Census transform in presence of radiometric changes (e.g., due to different lighting and exposure conditions). As a part of this work, the new stereo test datasets (2005 datasets) available at Middlebury stereo website [15] were used, to perform a quantitative comparison of the Rank versus Census transform under radiometric variations. It contains six datasets (Art, Books, Dolls, Laundry, Moebius, and Reindeer) each captured under three different exposures and lighting conditions, resulting in nine different combinations of image pairs. Figure 4.12 illustrates several sample images with varying lighting and exposure from Dolls dataset.

Figures 4.13(a-b) show the result of this comparison as average error percentage under exposure and lighting variations for each of the nine different combinations for Rank of size 3x3 versus Census of size 3x3. The average errors for each case were computed over all six datasets. Similarly, Figures 4.13 (c-d) show the obtained results for Rank of size 5x5 versus Census of size 3x3. As illustrated by the experiments, in both cases Census transform of size 3x3 outperforms both Rank of size 3x3 and 5x5 under varying radiometric conditions.

Census transform along with the Hamming distance (used as dissimilarity measure between the transformed pixels) are implemented efficiently through a mix of C and assembly code. More specifically, all critical loops of the algorithm where most of the processing cycles are dedicated to are implemented and carefully optimized in assembly. Hamming distance between two bit strings is defined as the number of bits in which the given bit strings differ. It is efficiently implemented using an exclusive-or instruction followed by a special in-built instruction of Blackfin family called ONES that provides the number of 1's contained in the given register. Both of these instructions (i.e. bitwise exclusive-or and ONES) requires only a single processing cycles, which makes the implementation of Hamming distance efficient. Equation 4.13 shows the mathematical formulation of the Hamming distance where $\bigotimes$ represents the boolean exclusive-or operator and N is the number of bits in the bit strings (i.e. R_{T1} and R_{T2}.

$$\mathcal{H}(R_{T1}, R_{T2}) = \sum_{i=1}^{N} R_{T1} \bigotimes R_{T2} \tag{4.13}$$

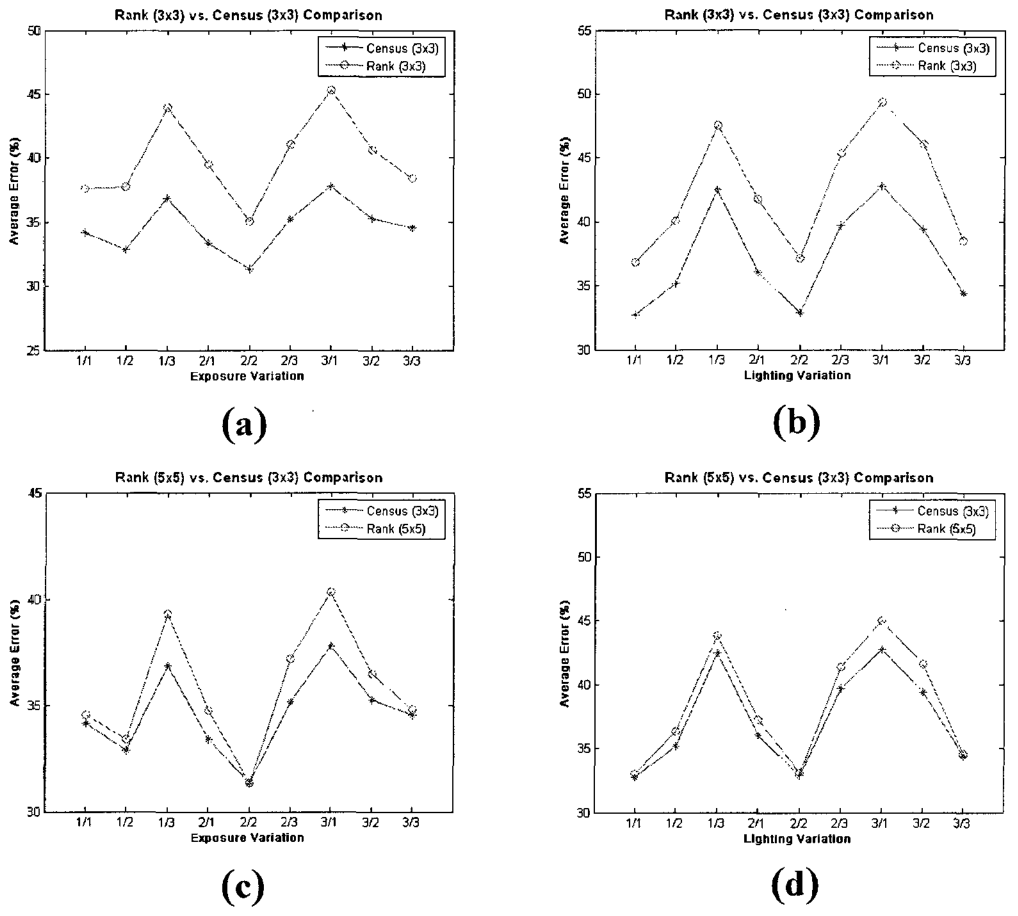


Figure 4.13: Comparison of Rank (size 3 and 5) vs. Census (size 3) transform for 3x3 left/right image combinations differing in exposure and lighting conditions

Using the bit counting facilities supplied by Blackfin processor along with the careful assembly level optimization of the code, memory management techniques such in-place processing (explained in section 4.3.3), and finally reducing the window size from 5 to 3 (Census of size 3 replaced by Rank of size 5), make it possible to decrease the number of cycles required by the preprocessing algorithm from about 25 million instructions (MI) in MESVS-I to only 7 MI in MESVS-II. Therefore, the new enhanced preprocessing algorithm is more than three times faster and also more robust as compared to the original system.

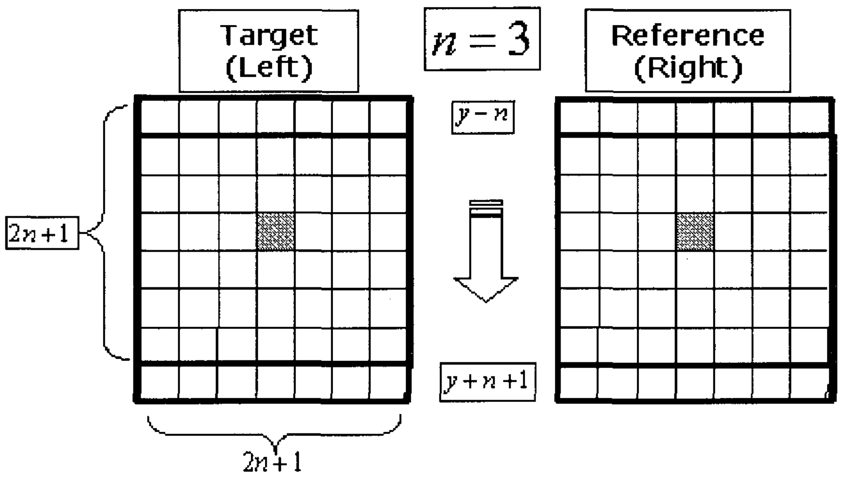


Figure 4.14: Vertical recursion in correlation calculation process

Matching core

The most important part of the stereo matching engine is the matching core, which is carefully selected, and then implemented for optimum performance while being compact, and robust. It is based on a fast correlation-based matching algorithm that combines three levels of recursion, namely, horizontal, vertical, and modular to avoid redundant computations and expedite the matching process [16]. The correlation measure deployed is the sum of Hamming distances $SHD(x, y, d)$. Equation 4.14 illustrates how this measure is computed assuming correlation window of size (n+1)x(n+1) centered at (x, y) in the left image L and at (x, y, d) in the right image R for the candidate disparity of d.

$$SHD(x,y,d) = \textstyle\sum_{i,j=-n}^{n} \mathcal{H}(L(x+j, y+i), R(x+j+d, y+i)) \tag{4.14}$$

The vertical recursion implies that the correlation measure of the subsequent row $SHD(x, y+1, d)$ can be computed recursively using its value for the current row $SHD(x, y, d)$ plus an update term $U(x, y+1, d)$ as shown by Equation 4.15 [16].

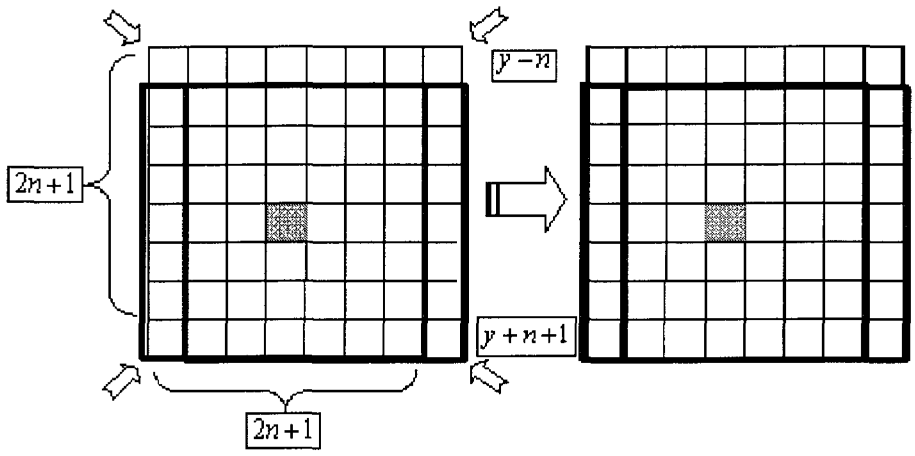


Figure 4.15: Combination of the horizontal and vertical recursion

$$SHD(x, y+1, d) = SHD(x, y, d) + U(x, y+1, d) \tag{4.15}$$

Where the update term $U(x, y+1, d)$ may be recursively computed as shown in Figure 4.14 (a) by excluding the contributions of the row above the correlation window, denoted by $y - n$, and taking into account the contributions from the new row within the shifted correlation window, denoted by $y + n + 1$ (assuming a correlation window of size $2n + 1$ with $n = 3$). Equation 4.16 demonstrates the mathematical formula used to compute the update term.

$$U(x, y+1, d) = \sum_{j=-n}^{n} |L(x+j, y+n+1) - R(x+d+j, y+n+1)| - \sum_{j=-n}^{n} |L(x+j, y-n) - R(x+d+j, y-n)| \tag{4.16}$$

The horizontal recursion has a similar implication as the vertical recursion for columns. That is, the update term $U(x+1, y, d)$ can be recursively calculated by excluding the contributions of the leftmost column of the previous correlation window (before horizontal shift), and considering the contributions from the newly included column, i.e. the rightmost column of the correlation window after the horizontal shift. Combining both the horizontal and vertical recursions, one can see that only the contributions associated with four pixels at the corners of the correlation windows (See Figure 4.15 and Equation 4.17) are required to recursively compute the update term values.

$$U(x,y+1,d) = U(x-1,y+1,d)+(|L(x+n,y+n+1)-R(x+d+n,y+n+1)|-|L(x+n,y-n)-R(x+d+n,y-n)|-|L(x-n-1,y+n+1)-R(x+d-n-1,y+n+1)|+|L(x-n-1,y-n)-R(x+d-n-1,y-n)|) \tag{4.17}$$

This adds to the system flexibility by making the run-time of the matching algorithm independent of the window size. Furthermore, using this method one only has to store and update the update terms of a row at a time. Therefore, the total memory required is WxD words, where W is the image width and D is the disparity range.

Finally, the modular recursion in incorporated to further speed up the matching process. It is based on the observation that contributions associated with the two pixels at the top right and bottom right corners of the correlation window are the same as the contributions of the top left and bottom left pixels $2n+1$ iterations later (See Figure 4.16). Therefore, the most recent $2n+1$ terms (associated to these contributions) can be temporarily stored within a 2D array $M(x,d)$ for each disparity value $d \in [0, d_{max}]$ to be re-used to compute the update term U $2n+1$ iterations later. Equation 4.18 shows the modified version of Equation 4.17 that incorporates the modular recursion.

$$U(x,y+1,d) = U(x-1,y+1,d) + |L(x+n,y+n+1) - R(x+d+n,y+n+1)| - |L(x+n,y-n) - R(x+d+n,y-n)| - M(x',d) \tag{4.18}$$

Whenever an item of the array $M(x,d)$ is accessed, it is updated modularly as shown in the Equations 4.19 and 4.20. More details on these recursion schemes is presented in [16].

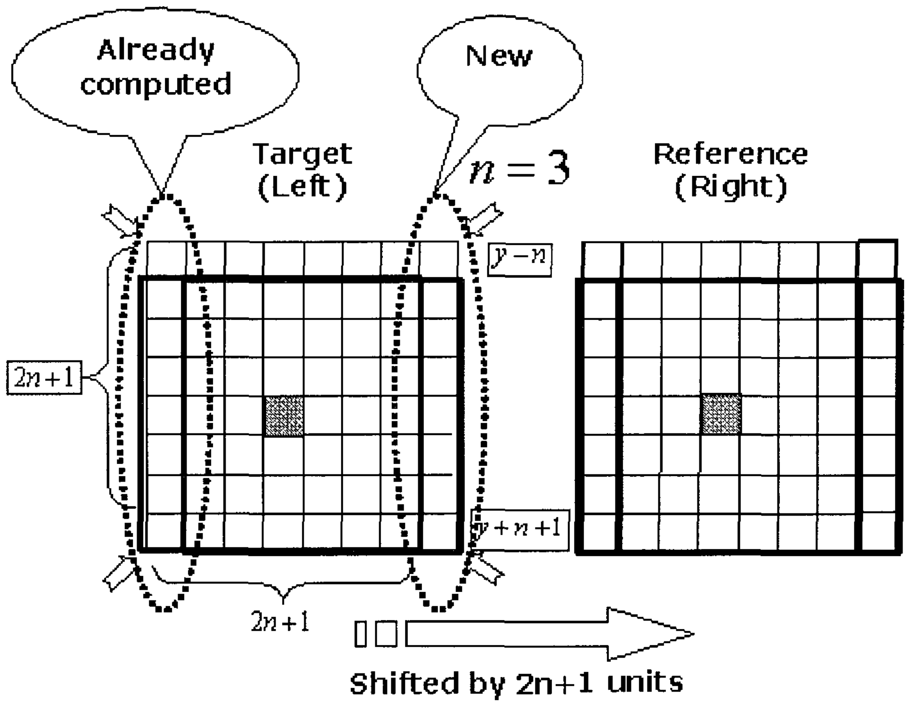


Figure 4.16: Modular recursion applied in the correlation process

(4.19)

$$M(x', d) =$$
$$|L(x-n-1, y+n+1) - R(x+d-n-1, y+n+1)| - |L(x-n-1, y-n) - R(x+d-n-1, y-n)|$$

(4.20)

$$x' = x \; mod(2n+1)$$

The left/right consistency check (LRC) method is widely accepted in the stereo vision community as an effective approach to detect half-occluded regions. LRC is based on the observation that for any given pixel on the left image scanline at position x_l, if one uses the left to right image disparity map $disp_{l_r}(x, y)$ to predict its position in the right image scanline using $x_r = x_l + disp_{l_r}(x_l, y)$ and then projects this position back to the left image scanline using right to left image disparity map $disp_{r_l}(x, y)$ and $x_{lp} = x_r + disp_{r_l}(x_r, y)$, the disparity values are consistent if and only if $x_l = x_{lp}$. In practice we typically allow x_l and x_{lp} to be slightly different limited through a threshold value. Also, disparity values from neighbor pixels may be propagated to substitute the disparity values invalidated by LRC. In contrast to the other occlusion detection methods available, LRC detects the majority of the occluded pixels and performs quite well in highly textured scenes [17]. Furthermore, it can be efficiently implemented by temporarily storing the correlation values for each row. The stereo matching engine incorporates an efficient implementation of LRC to handle half-occlusions. No additional memory is required to implement this extra validation step. For regions detected as occluded, the disparity values are invalidated, and set to zero.

Postprocessing

Local matching methods based on correlation have a natural tendency to blur object boundaries (depth discontinuity regions). This issue is commonly know as "foreground fattening problem" [70]. It is caused by the assumption of smooth disparity variations within the correlation windows, which is violated around object boundaries.

The postprocessing algorithm is a novel method designed with the following objectives:

- Reduce errors due to the depth discontinuity regions of the scene, which typically overlap with the object boundaries
- Alleviate errors introduced by the poorly textured areas of the image
- Have efficient implementation with minimum overhead

The postprocessing algorithm takes the variance map of both target image and disparity map itself as input to process the disparity map and invalidate mismatches caused by the low-texture and depth discontinuity regions. One of the advantages of the proposed method is that the required variance maps are computed recursively using fast recursion schemes similar to the matching core (see section 4.3.2). As a result, the algorithm implementation

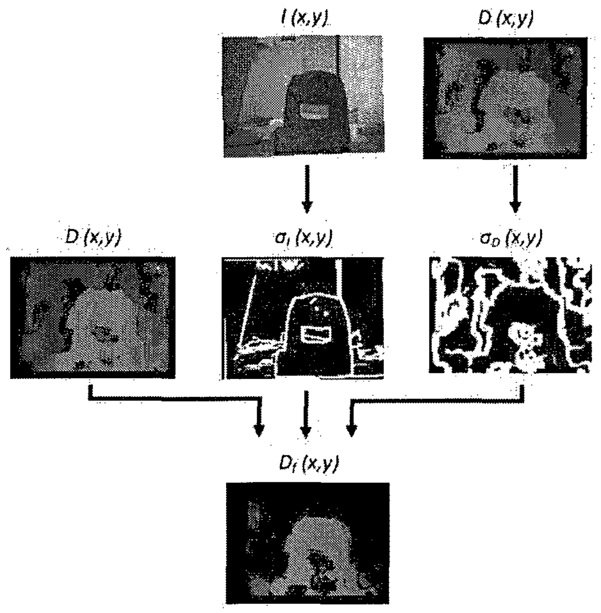


Figure 4.17: Illustration of the postprocessing algorithm

is quite efficient requiring only about 2 million cycles to run on the current embedded hardware platform.

Equations 4.21 and 4.22 show how the mean and variance of an image I are calculated for a local neighborhood of size NxN with $N = 2n + 1$.

$$\mu(x,y) = \tfrac{1}{N^2} \textstyle\sum_{i,j=-n}^{n} I(x+j, y+i) = \tfrac{1}{N^2} S_1(x,y) \tag{4.21}$$

$$\sigma^2(x,y) = \tfrac{1}{N^2} \textstyle\sum_{i,j=-n}^{n} (I(x+j, y+i) - \mu(x,y))^2 \tag{4.22}$$

Assuming a symmetric window (of local neighborhood), it is shown that Equation 4.22 used to compute variance of the image can be simplified as shown in Equation 4.23.

$$\sigma^2(x,y) = \frac{1}{N^2}\left(\sum_{i,j=-n}^{n} I^2(x+j,y+i)\right) - \mu^2(x,y) = \frac{1}{N^2}S_2(x,y) - \mu^2(x,y) \tag{4.23}$$

Similar to the section 4.3.2, the summation terms S_1 and S_2 are computed as shown in Equations 4.24 and 4.24 [16].

$$S_1(x,y+1) = S_1(x,y) + U_1(x,y+1) \tag{4.24}$$

$$S_2(x,y+1) = S_2(x,y) + U_2(x,y+1) \tag{4.25}$$

The update terms U_1 and U_2 introduced in the Equations 4.24 and 4.25 are computed using three levels of recursion as explained for the matching core.

The postprocessing algorithm is based on the idea that for a given pixel that lacks enough texture (measured by comparing the variation of intensities over a local neighborhood against a threshold) and its estimated disparity value is not in agreement with the surrounding pixels (measured by comparing variation of disparity values over a local neighborhood against a threshold), the estimated disparity for that pixel is most likely invalid and must be removed from the disparity map. The low-texture regions of image are blackened in the variance map of the target image $\sigma_I(x,y)$ as shown in Figure 4.17. Similarly, the regions of disparity map where neighboring disparity values do not agree with the estimated disparity are presented as white in the variance map the disparity map $\sigma_D(x,y)$ (See Figure 4.17).

Equation 4.26 provides a summary of the postprocessing algorithm in mathematical terms. As illustrated each pixel in the final disparity map $D_f(x,y)$ is invalidated and marked as zero whenever its texture $\sigma_I(x,y)$ is less than a predetermined threshold T_{txt} and its disparity variation $\sigma_D(x,y)$ is more than a predetermined threshold T_{disp}. Otherwise, it is marked as valid by being kept unchanged. The threshold values T_{txt} and T_{disp} are manually tuned for optimum performance in the current implementation.

$$D_f(x,y) = \begin{cases} 0, & if \begin{cases} \sigma_I(x,y) <= T_{txt} \\ \sigma_D(x,y) >= T_{disp} \end{cases} \\ D(x,y), & \text{otherwise} \end{cases} \tag{4.26}$$

As shown by the obtained experimental results (see section 5.2.2), the proposed post-processing algorithm method improves the quality of disparity maps, specifically around object boundaries. i.e. the overall recovered shape of the objects in the foreground is more clear and less affected by the foreground fattening problem common to correlation-based matching methods. As the last processing step, the disparities belonging to the background (more than 50cm away from the system) are removed (blackened) by checking against a constant threshold (see Figure 4.17).

4.3.3 Achieving real-time performance

This section elaborates on several optimization strategies applied in this work to enhance the performance of MESVS to the real-time level, i.e. 20 frames per second. In every embedded setting, optimization involves three major steps as follows [21].

- Compiler based optimization: this is typically the first step performed in order to maximize the speed and/or minimize the space requirements at the compile time. It is done by modifying the code to take advantage of the performance enhancement architectural features of the embedded processor. In case of Blackfin, these features include multi-stage interlocked pipelining, vectorization, and application of the compiler intrinsic functions such as $mult\_fr1x16$ and add_fr1x16 functions used in the implementation of the rectification algorithm.

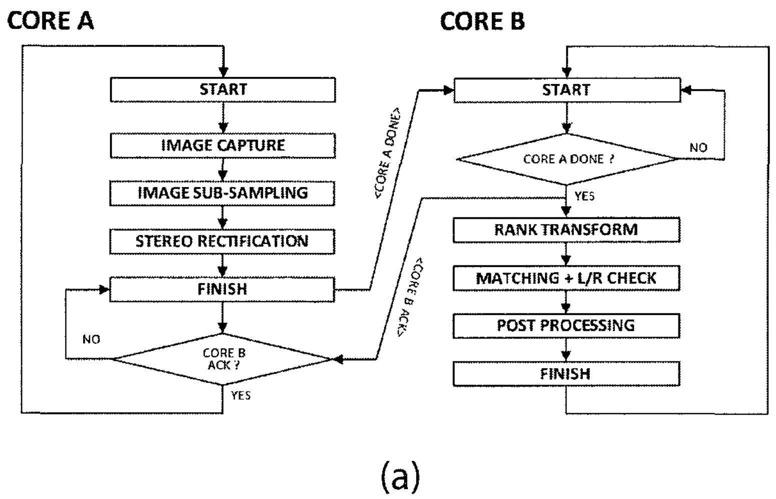
- Assembly level optimization: this is the lowest level of optimization. In the current implementation of MESVS, all of the critical loops of the system firmware, i.e. portions of the code that are most frequently executed, are detected using the statistical profiler suite of VDSP++ IDE. Then, system firmware is implemented as a mix of C and Assembly code, with critical loops implemented in assembly and carefully optimized for performance.

- System level optimization: it is achieved through partitioning system memory efficiently to streamline data flow. To achieve real-time performance, it is necessary to consider the bandwidth provided by the processor and system bus to determine the scheme data is handled within system memory. One of the key points regarding this observation is the trade-off between the memory access speed and the available physical size of the memory unit. For example although external off-chip memory is typically much larger than on-chip memory, its several times slower in terms of access time. Therefore, in the current implementation of MESVS firmware the main objective has been to limit all memory access of processing cores to the fast on-chip memories and avoid potential bottlenecks and performance degradations caused by the slow external memory accesses.

In the following sections first the two-staged pipelined processing model used to leverage the dual-core architecture of the embedded processor is explained and then the memory and data management schemes developed on Core A and B of the processor in order to achieve real-time performance are described.

Two-staged pipelined processing

In order to exploit the dual core processing capabilities of the Blackfin processor and enhance performance of the system firmware, the stereo matching engine is implemented as a two-staged pipeline, as shown in Figure 4.18. The core A of Blackfin is in charge of image capture, sub-sampling, and stereo rectification; while core B performs the preprocessing, matching and postprocessing steps. The mutual signaling mechanism implemented using shared data protected with locks is deployed to maintain synchronization between the processing cores (See Figure 4.18 (a)). Core A begins the process by performing its three tasks



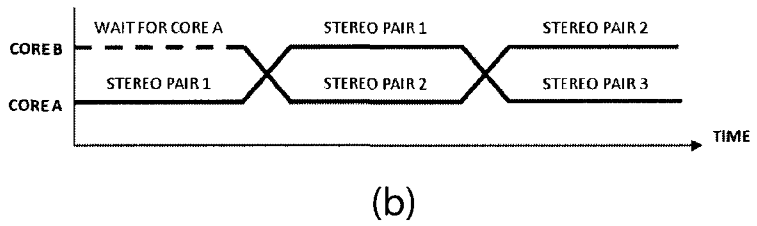


Figure 4.18: (a) Flowchart of the two-staged pipeplined processing model (b) Processing timeline of dual cores

on the first stereo image pair and then signals core B to start performing the rest of the steps. Once started, core B signals core A back, allowing it to continue with processing the second stereo image pair. Generally, while core B is processing image pair n, core A is processing image pair $n+1$ at the same time (See Figure 4.18 (b)). Using the pipelined programming model, both processing cores are fully operational all the time. Furthermore, it is simpler and has less overhead as compared to the other programming models [21].

Data management on Core A

Figure 4.19 illustrates how data flow is managed and image sub-sampling and rectification steps are handled within core A (only the left image processing is shown for the sake of

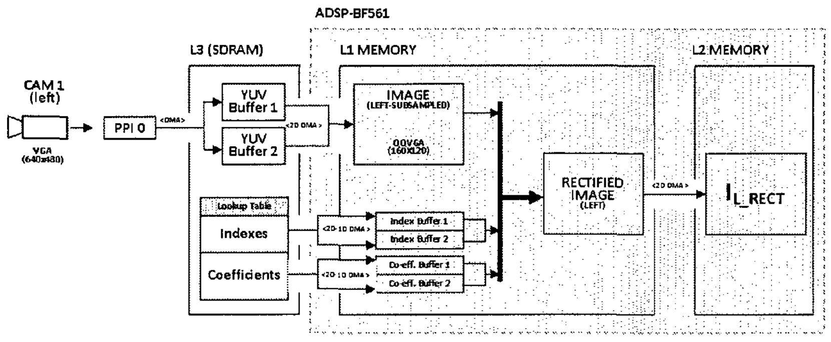


Figure 6. Memory and data flow management for Core-A.

Figure 4.19: Memory and data managemnet scheme implemented on core A

simplicity, the procedure is similar for the right image). The process starts with capturing image provided by the camera module using the PPI video port of Blackfin and transferring it to the L3 (external) memory using 2D DMA facility. Double buffering scheme is also used to allow core A process input images while DMA transfers next captured image to the external memory. This improves system performance since the processing cycles of core A will never be wasted waiting for the next image to be captured. Once captured, the image has to be sub-sampled to fit within the fast L1 memory of the system. Sub-sampling process is performed efficiently through the 2D DMA facility of Blackfin processor.

Rectification is the stage where an enhanced embedded processing model is implemented to reduce the execution time of the algorithm. The proposed scheme relies on the observation that image indexes computed during the back-projection step and coefficients used for bi-linear interpolation phase of the rectification algorithm are constant. So, instead of re-computing these parameters for each new image pair, they are computed only once and stored as indexes and coefficients tables. Regarding the resolution of QQVGA (160x120) for the stereo image pair, 37.5 KB (38,400 Bytes = 160 x 120 x 2 index values per pixel x 1 byte per index entity) of memory space is required for each indexes table (one for the left and one for the right image) and similarly 150 KB (153,600 Bytes = 160 x 120 x 4 coefficients per pixel x 2 bytes per coefficient entity) for each coefficients table. So, the total amount of memory required for both left and right image tables is 375 KB, meaning that they do not fit within the on-chip memory of the system. The rectification tables are stored in the external memory. Double buffering along with the parallel DMA transfers scheme is used to resolve the problem of long delays associated to the external memory accesses. The idea is to allocate four buffers within the fast L1 memory (two for indexes and two for coefficient table) (See Figure 4.19). Each time the rectification algorithm needs rectification parameters for a given row of the image, it uses for example the first buffers , i.e. index buffer 1 and co-eff. buffer 1; while simultaneously DMA is fetching data from the external memory and filling the other (second) buffers, i.e. index buffer 2 and co-eff. buffer 2. For the next row, the rectification algorithm and the DMA both switch buffers and so on. In our implementation we experienced a performance improvement of about 50% using the aforementioned techniques. The last step is to transfer the rectified image to the L2 shared memory to be processed by core B in the next round. This is accomplished through another 2D DMA stream.

Data management on Core B

In order to improve the performance of the matching algorithm implemented on core B, the system matching engine is modified to deploy in-place processing approach (See Figure 4.20). This has resulted in significant reduction of the amount of memory required by the matching algorithm. In-place processing means that source image data will be overwritten with the results of the algorithm, i.e. the source and destination memory locations remain the same. As a result, no extra (separate) memory is required to store processed data. As shown the rectified image pairs are first transferred from L2 into L1 memory using 2D DMA facility. Then, the Census transform is applied to each image (i.e. *L_RECT* and *R_RECT*) and it is overwritten with the obtained results. A linear buffer of size 1xW with W being the image width is used to temporarily store the Census transform results for the image row processed at the moment. Once done processing the current row, results are transferred from the linear buffer and written onto the source image. The same procedure is repeated for the rest of the rows of the image.

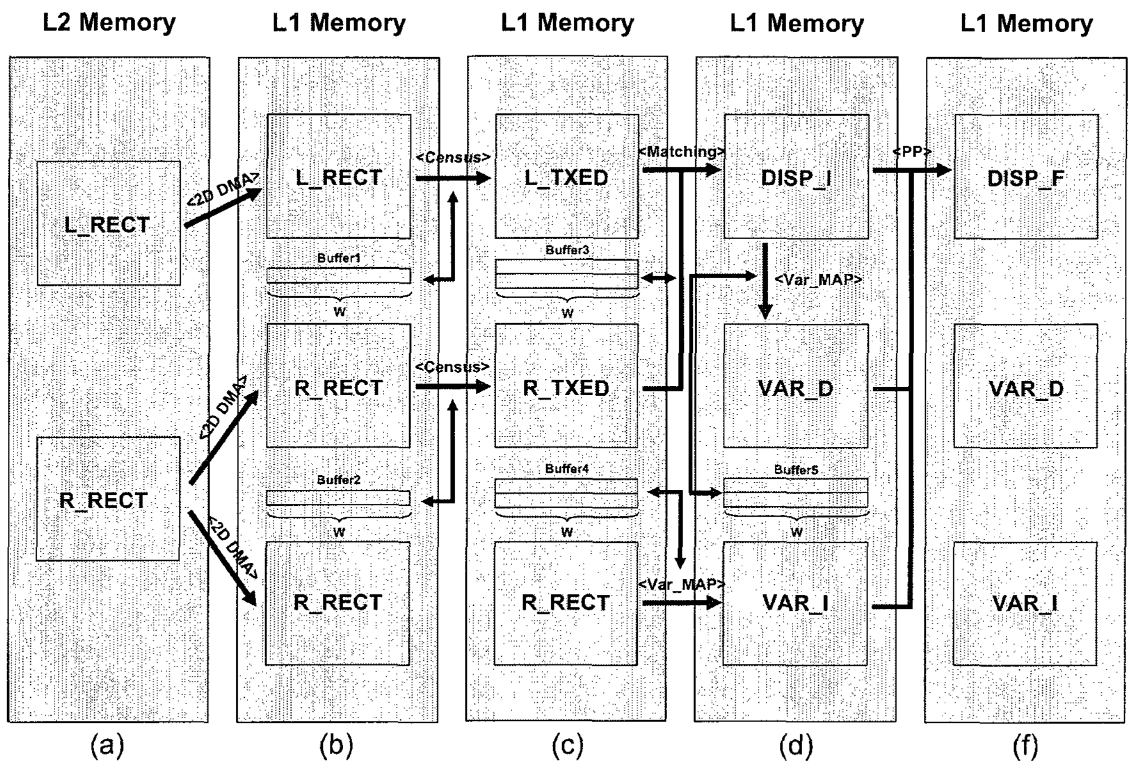


Figure 4.20: Memory and data managemnet scheme implemented on core B

The next step is to perform recursive correlation-based matching. This step is also implemented using the in-place processing scheme as shown in Figure 4.20. However, for this algorithm the linear buffer has to hold the results (disparity values) of both the current and previous processed row and thus becomes of size 2xW. This is caused by the fact that the matching algorithm requires pixel values for both previous row and current row in order to recursively compute correlation scores for the pixels on the current row (see section 4.3.2). Therefore, matching results (disparities) obtained for the previous row may not be overwritten until processing the current row of the image is done.

The variance maps needed by the postprocessing algorithm (i.e. VAR_D and VAR_I) are also computed using a similar recursive approach combined with in-place processing scheme. Again a buffer of size 2xW is used to temporarily store results of the current and previous row for each of the variance maps and the source images are overwritten by the maps produced. Lastly, the postprocessing algorithm is applied to produce the final disparity map $DISP\_F$. As illustrated deploying the in-place processing scheme has made it possible to keep the entire program data within the fast L1 memory only and therefore alleviated any performance bottleneck and degradations caused by the slow memory latencies.

Chapter 5

Experimental Results

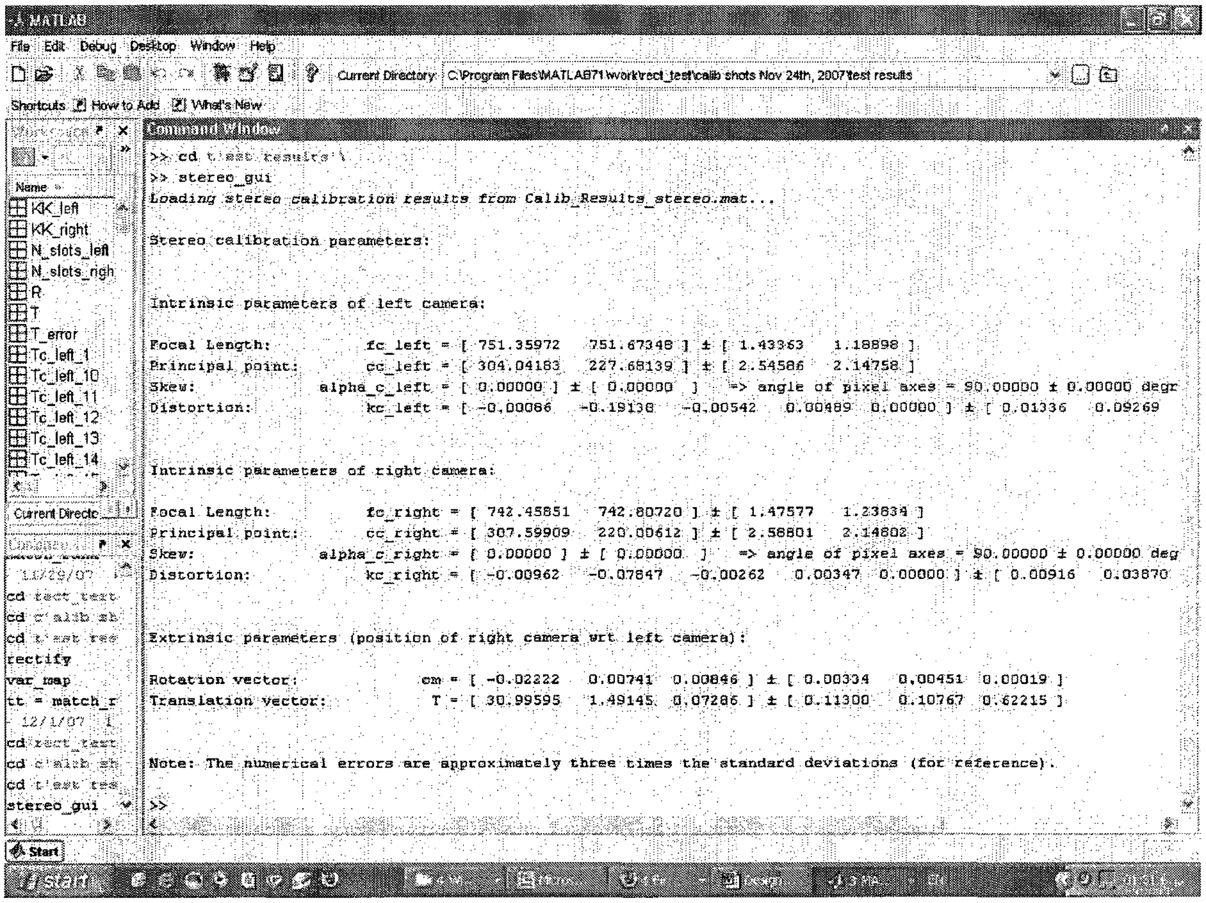
This chapter presents the practical experiments performed and their obtained results using MESVS. It begins with elaborating on the stereo calibration process and emphasizing on the importance of its accuracy on the quality of the final disparity maps produced. In the next section the experimental results obtained with MESVS in a variety of cases are presented. Finally, performance of the novel postprocessing algorithm is discussed and analyzed in section 5.2.2.

5.1 Performing Stereo Calibration

As mentioned in section 4.3.1, the stereo calibration for MESVS is accomplished using Caltech calibration toolbox [13]. According to the software documentation, the following steps were taken to perform calibration.

1. Prepare a calibration pattern (a planar surface with checkerboard pattern)

2. Take a number of sample shots of the calibration pattern (about 18 shots was found to be enough for our system) in different positions and orientations with each camera (i.e. left and right).



```
>> cd test results\
>> stereo_gui
Loading stereo calibration results from Calib_Results_stereo.mat...

Stereo calibration parameters:


Intrinsic parameters of left camera:

Focal Length:          fc_left = [ 751.35972   751.67348 ] ± [ 1.43363   1.18898 ]
Principal point:       cc_left = [ 304.04183   227.68139 ] ± [ 2.54586   2.14758 ]
Skew:             alpha_c_left = [ 0.00000 ] ± [ 0.00000  ]   => angle of pixel axes = 90.00000 ± 0.00000 degr
Distortion:            kc_left = [ -0.00086   -0.19138   -0.00542   0.00489  0.00000 ] ± [ 0.01336   0.09269


Intrinsic parameters of right camera:

Focal Length:          fc_right = [ 742.45851   742.80720 ] ± [ 1.47577   1.23834 ]
Principal point:       cc_right = [ 307.59909   220.00612 ] ± [ 2.58801   2.14802 ]
Skew:             alpha_c_right = [ 0.00000 ] ± [ 0.00000  ]   => angle of pixel axes = 90.00000 ± 0.00000 deg
Distortion:            kc_right = [ -0.00962   -0.07847   -0.00262   0.00347  0.00000 ] ± [ 0.00916   0.03870


Extrinsic parameters (position of right camera wrt left camera):

Rotation vector:             om = [ -0.02222   0.00741  0.00846 ] ± [ 0.00334   0.00451  0.00019 ]
Translation vector:           T = [ 30.99595   1.49145  0.07286 ] ± [ 0.11300   0.10767  0.62215 ]


Note: The numerical errors are approximately three times the standard deviations (for reference).

>>
```

Figure 5.1: A sample screenshot of the calibration results obtained using Caltech toolbox

3. Calibrate each camera using calibration shots captured in the previous step. This is a rather time-consuming and difficult step as the corner extraction process is not fully automatic and requires the four extreme corners of the calibration pattern to be manually located by the user. In addition, to obtain maximum accuracy, user is allowed to manually enter the location for all corners. In our experiments we found this step very important and influential on quality of the final results as explained in section 5.1.1. The output of this step is the internal(intrinsic) parameters for each camera, i.e. the focal length f_c, principal point c_c, skew factor $alpha_c$, and distortion coefficients k_c (See Figure 5.1). The distortion model used for the calibration includes both radial and tangential distortions. Although in our experiments we found that compensating

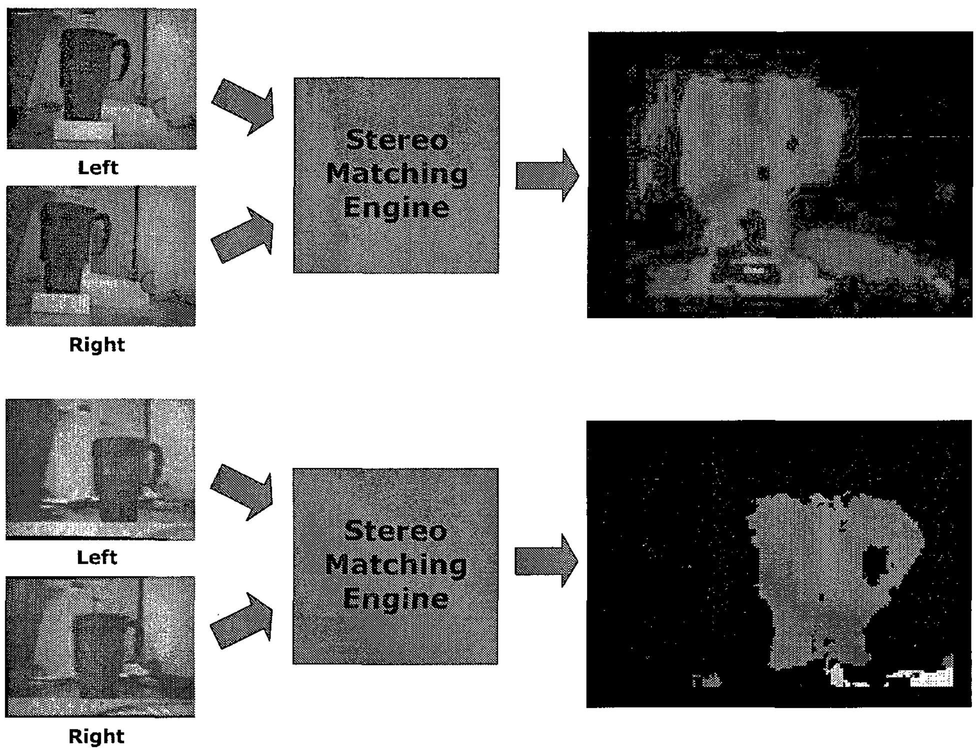


Figure 5.2: The effect of calibration accuracy on performance of the stereo matching engine

for tangential distortion is not desirable as it is computationally expensive, but does not significantly improve the obtained results.

4. Use the stereo calibration GUI along with the calibration parameters obtained for each camera in the previous stage to calibrate for the external parameters of the stereo system (i.e. the rotation vector *om* and translation vectors *T* that determine position of the right camera with respect to the left camera).

5.1.1 Calibration accuracy

Figure 5.2 presents a comparison of the disparity maps produced for two different calibration cases. The stereo image pairs in the lower section were rectified (and un-distorted) using more accurate parameters as compared to the upper section. One can observe that the disparity map produced by the stereo matching engine for the lower section has a better quality. For example, the handle of the mug is fully recovered and also the boundary of the object is less distorted.

The above observation makes calibration a very important step and influential on the overall efficiency of the system. Therefore, in our experiments we have tried to maximize the accuracy of the calibration process by manually fine tuning the corner extraction procedure and also reducing the systematic errors such as defects found on the calibration pattern (i.e. non-planarity or unequal distance between the grids).

5.2 Experiments

This section presents results of the experiments performed in order to evaluate performance of the MESVS stereo matching engine. First we present some qualitative results showing the superior efficiency of the Census compared to the Rank transform. We also assess the robustness of both transforms with respect to illumination variations in the scene. The second part of experiments is dedicated to the quantitative evaluation of the postprocessing algorithm.

5.2.1 Rank vs. Census results

As discussed and shown through extensive experiments (See section 4.3.2), the Census transform outperforms Rank transform when exposed to the radiometric variations. In this section the experimental results obtained using real world image pairs (captured with MESVS) are presented. The results are produced using both Rank (deployed in MESVS-I) and Census transform (deployed in MESVS-II) in order to compare their performance in a real situation.

Figure 5.3 shows the output of the stereo matching engine at each step using both Rank (right column) and Census transform (left column) as preprocessing algorithm. The object of interest (a mug) is shown in the rectified reference image (See Figure 5.3 (a)). Figure 5.3 (b-e) illustrate the transformed reference image, initial disparity map, results after applying LRC method, and the final disparity map produced by the postprocessing algorithm, respectively. It can be seen that in both cases the structure of the object is roughly recovered (Figure 5.3 (c)). Furthermore, applying LRC method helps to eliminate mismatches caused by the occluded areas (e.g. left boundary of the object). Finally, the postprocessing algorithm reduces the foreground fattening problem (please note that the background subtraction step is removed to facilitate comparison). Moreover, comparing the final disparity maps, one can see that Census transform provides higher quality results as the object boundaries are smoother and less distorted.

In order to further assess the robustness of the Rank and Census transforms, we have experimented under variable illumination conditions by modifying the light source intensity and changing its location. Figure 5.4 (a) shows the left and right image pairs of the same object of interest. The left image is captured with lights turned on and the right image is captured with lights turned off to mimic the effect of variation in illumination intensity. Figures 5.4 (b-e) present output of the stereo matching engine in different steps for the Rank and Census transforms similar to Figure 5.3. Both of the methods seem to be able to cope with illumination variations and produce acceptable results. Although Census transform is producing a slightly higher quality disparity map.

The results of changing the light source location are shown in Figure 5.5. Figure 5.5 (a) shows the rectified image pair with the light source positioned at the left side in the left image and positioned at the center in the right image. Both of the transforms seem to fail the test. Nevertheless, results obtained using Census transform are relatively more accurate and less noisy. For example the large area on the upper right corner of the image is mismatched using Rank transform whereas Census transform does not produce such mismatches (See Figure 5.5 (c)). In conclusion, Census transform proves to be more robust than Rank transform. Although both of the transforms fail under large illumination variations (especially as a result of changing the illumination source position). This demands for incorporating more robust preprocessing algorithm or possibly even matching method (e.g. a global matching algorithm) in order to make MESVS more suitable for real world applications.

Finally, Figure 5.6 shows more results produced through the Census transform. As shown both objects of interest (the mug and the camera) are correctly recovered. In addition applying LRC and postprocessing algorithms improves results by alleviating errors caused by occluded, low texture and depth discontinuity regions of the scene. One problem with the postprocessing algorithm is that it enlarges the black areas resulted from removing mismatched disparities using LRC method. This may be alleviated by applying a dilation operator or a similar filter to the obtained disparity map, which can be considered as a future work.

5.2.2 Performance of the postprocessing algorithm

In order to evaluate effectiveness of the postprocessing algorithm, we have applied the algorithm to a set of test image pairs with known ground truth disparity map. The image pairs used are Cones, Teddy, Venus, and Tsukuba (see Figure 2.8). Because the postprocessing algorithm is designed to target the depth discontinuity and low texture regions, the obtained results are collected for these areas separately. The occluded areas are also included to further examine the performance of the algorithm.

As mentioned before, the thresholds for the postprocessing algorithm are determined experimentally and tuned to provide the optimum functionality. A similar procedure is applied here for the test image data. We have experimented with a wide range of values for both the image T_{txt} and disparity T_{disp} thresholds. The experimental results are collected for each of the depth discontinuity, textureless, and occluded areas of the image. The accuracy of the algorithm, the ratio of correctly detected pixels to the total pixels detected (See Equation 5.1), is selected as the representative of its performance and gathered over the four test image pairs mentioned before. In our experiments we figured that a larger window size produces higher quality results. Nevertheless, regarding the real-time performance considerations of MESVS, the algorithm window size (used to define the local neighborhood deployed in the variance map computations) is limited to five.

$$Accuracy = \frac{True\ Negatives + True\ Positives}{True\ Negatives + True\ Positives + False\ Negatives + False\ Positives} \tag{5.1}$$

Figure 5.7 shows the obtained results for the occluded areas. As shown the maximum accuracy is achieved for the T_{txt} of about 35 and T_{disp} of about 8. One can see that the maximum obtainable accuracy for occluded regions is poor (about 45%), which is not important because the postprocessing algorithm is not designed to treat these areas of the image. The results for the depth discontinuity and textureless regions are illustrated in Figures 5.9 and 5.8, respectively. It can be seen that for the textureless regions the diagram is almost concave with the maximum accuracy obtained for T_{txt} of about 110 and T_{disp} of about 17. On the other hand, the results for the depth discontinuity regions are nonconclusive as for these areas we have the higher the threshold values, the better the accuracy of the algorithm. In order to tackle this issue, we have computed the average accuracy over both the textureless and depth discontinuity regions as shown in Figure 5.10. This solves the problem as the obtained diagram has the desirable concavity characteristics to an acceptable extent and the best accuracy of 64% is achieved for the T_{txt} of around 135 and T_{disp} of around 17.

Having tuned the thresholds for the postprocessing algorithm, we have calculated five performance measures, namely, sensitivity, specificity, negative predictive value (NPV), positive predictive value (PPV), and of course accuracy to analyze the efficiency of the algorithm. Table 5.1 presents the computed measures for various areas of the images. Sensitivity and specificity are statistical measures commonly used for binary classifiers to see how well they identify a condition. Sensitivity is defined as the proportion of true positives of all positive cases in the sample space as shown by Equation 5.2. In our experiment, sensitivity is indeed the probability of if a pixel belonging to a certain region (e.g. occluded) is an outlier, then it is detected as an outlier by the postprocessing algorithm. Similarly, the specificity is the proportion of true negatives of all negatives cases (See Equation 5.3). It is the probability of a pixel belonging to a certain region being valid and not detected as an outlier. Together, the sensitivity and specificity reveal how well the postprocessing algorithm predicts positive and negative cases.

(5.2)

Table 5.1: Quantitative results of the postprocessing algorithm for different regions of the image

Region	Sensitivity	Specificity	NPV	PPV	Accuracy
Discontinuity	69.34%	58.55%	81.21%	42.50%	61.85%
Texture-less	79.66%	61.82%	88.41%	45.39%	66.90%
Occluded	43.55%	45.17%	51.69%	37.27%	44.48%

$$Sensitivity = \frac{True\ Positives}{True\ Positives + False\ Negatives}$$

(5.3)

$$Specificity = \frac{True\ Negatives}{True\ Negatives + False\ Positives}$$

The other two measures computed are NPV and PPV. As illustrated in Equation 5.4 and 5.5, these measures are the probability of a negative and positive outcome of the algorithm to be correct, respectively. The last presented parameter is the accuracy (already introduced), which provides a measure of how reliable the results of the postprocessing algorithm are.

(5.4)

$$NPV = \frac{True\ Negatives}{True\ Negatives + False\ Negatives}$$

(5.5)

$$PPV = \frac{True\ Positives}{True\ Positives + False\ Positives}$$

Looking at table 5.1, one can see that the postprocessing algorithm delivers an acceptable level of performance for the depth discontinuity and textureless area. The average accuracy for both of these regions is more than 60% with rather high sensitivity and NPV parameters. The results for textureless areas are slightly better than depth discontinuities as supported by higher performance measure values. Another observation is that NPV values are almost twice as much as PPV values. That is, high number of false positives are the major cause of accuracy degradation for the postprocessing algorithm not the false negatives. This is a valuable clue to take into account for future improvement of the postprocessing algorithm. Regarding the occluded ares, postprocessing algorithm is performing rather poorly, which is not an issue since it is not targeting those regions.

In conclusion, as intended the postprocessing algorithm is capable of dealing with outliers casued by the depth discontinuity and low texture areas and has a decent level of efficiency, which may be further enhanced in the future.

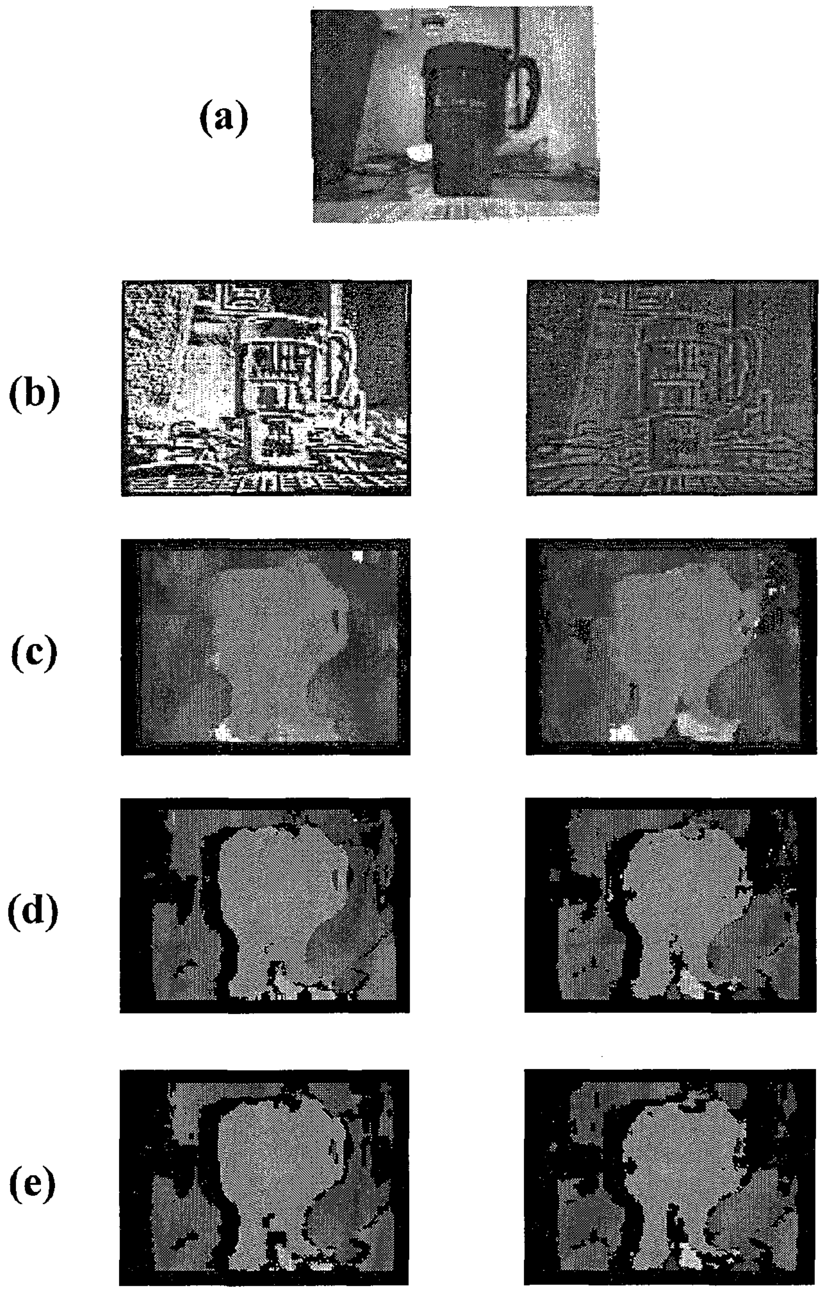


Figure 5.3: Comparison of Rank (right column) vs. Census (left column) results - (a) rectified target image (b) tranformed image (c) initial disparity map (d) disparity map after LRC (e) final disparity map

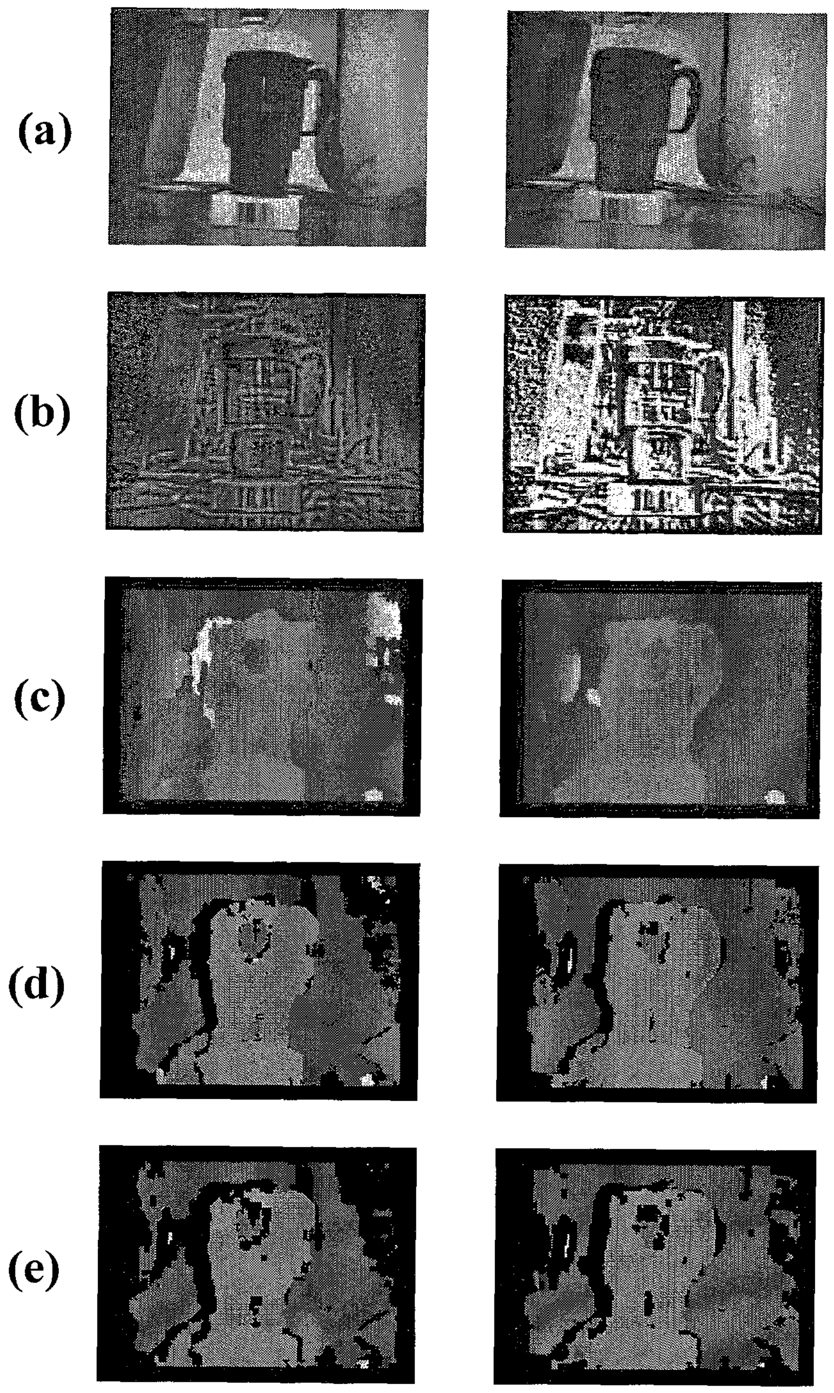


Figure 5.4: Comparison of Rank (left column) vs. Census (right column) with variable illumination - (a) rectified image pair (b) tranformed image (c) initial disparity map (d) disparity map after LRC (e) final disparity map

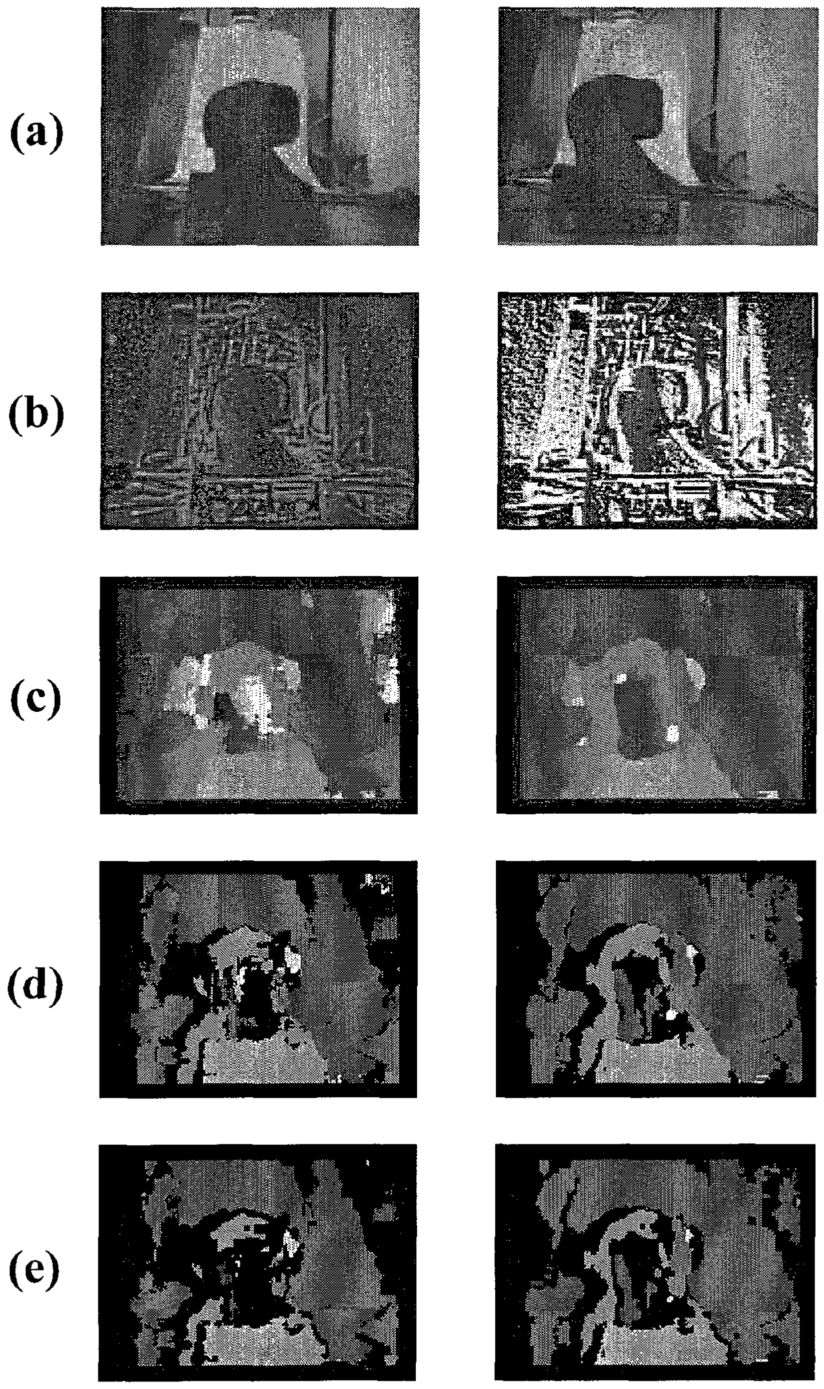


Figure 5.5: Comparison of Rank (left column) vs. Census (right column) with variable lighting source - (a) rectified image pair (b) tranformed image (c) initial disparity map (d) disparity map after LRC (e) final disparity map

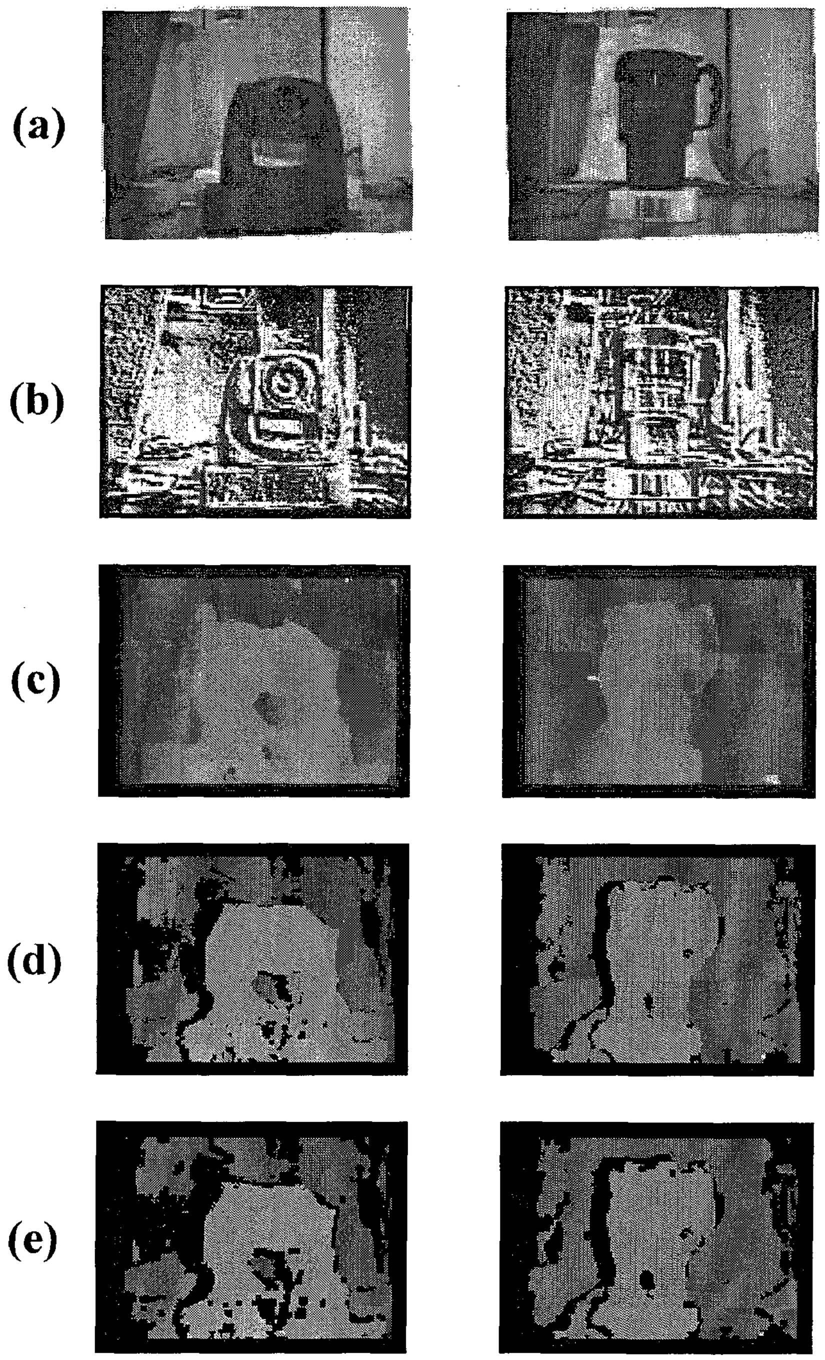


Figure 5.6: Sample experimental results using Census as preprocessing method - (a) rectified target image (b) tranformed image (c) initial disparity map (d) disparity map after LRC (e) final disparity map

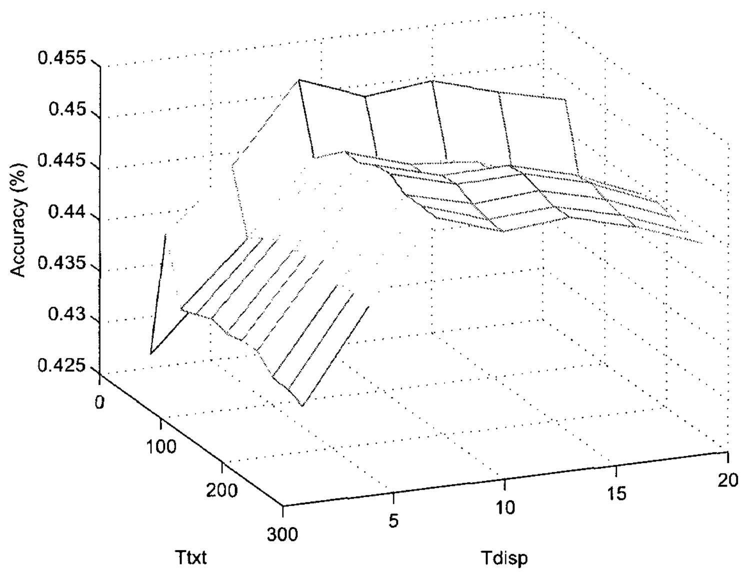


Figure 5.7: Accuracy versus thresholds (textureless and disparity) for occluded regions

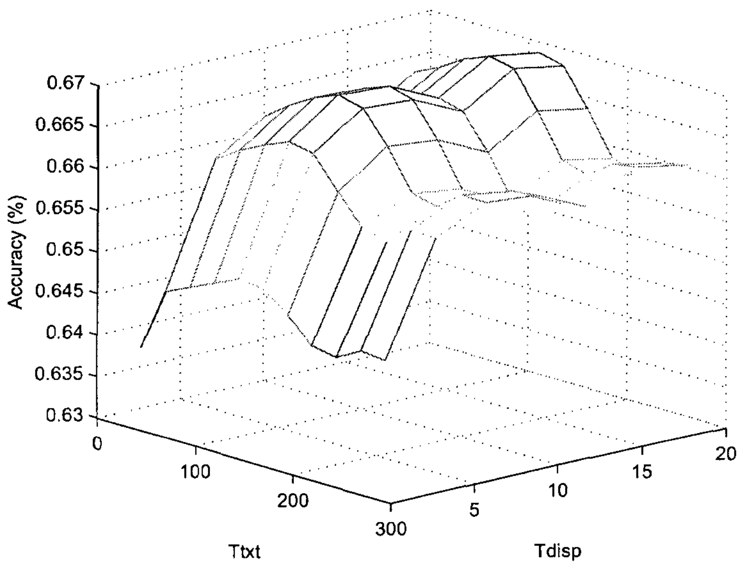


Figure 5.8: Accuracy versus thresholds (textureless and disparity) for textureless regions

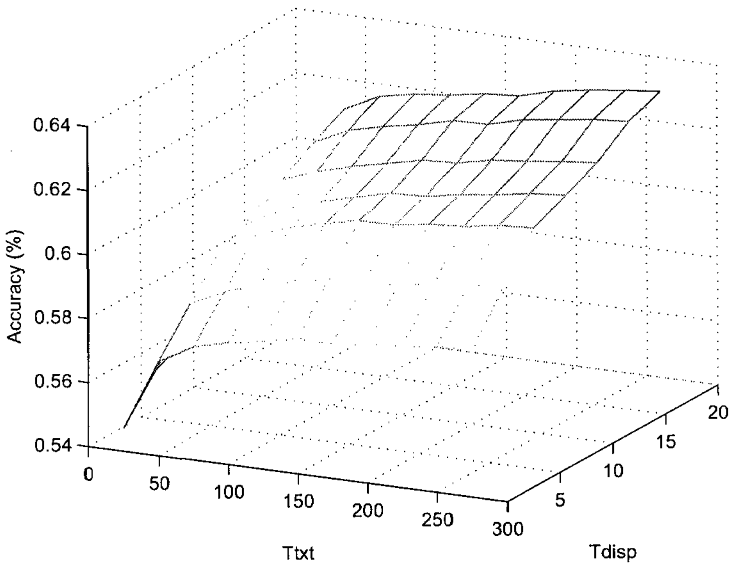


Figure 5.9: Accuracy versus thresholds (textureless and disparity) for depth discontinuity regions

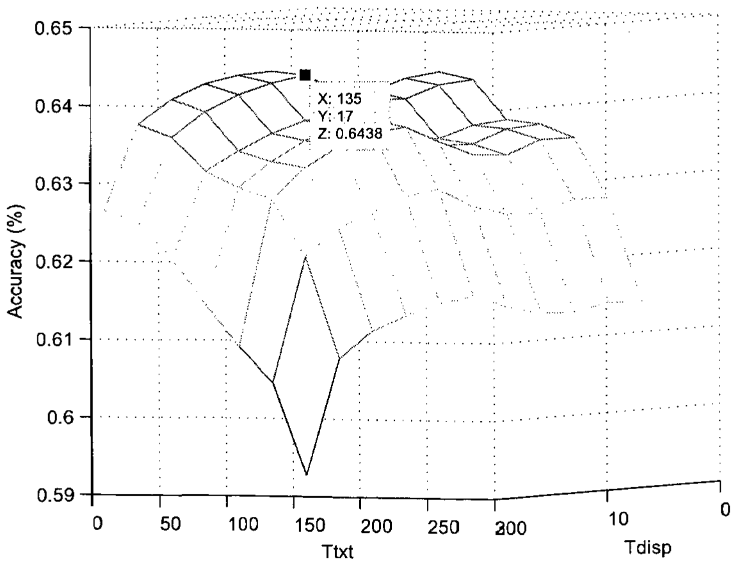


Figure 5.10: Average accuracy versus thresholds (textureless and disparity) for both textureless and depth discontinuity regions

Chapter 6

Conclusion and Future Work

6.1 Summary of Contributions

Real-time stereo vision systems with high performance and quality have many potential applications and therefore have an ever increasing demand. There are many stereo vision systems in the literature that claim to produce accurate results within the constraints of a real-time or near real-time system. The main concentration in most of these systems is the performance and/or quality without considering factors such as size, price, and power consumption, which are crucial in the embedded settings.

The major contribution of this work has been to develop a new stereo vision system, which is compact, capable of operating at real-time, power-efficient, and of low cost. System components have been carefully chosen in order to fit the design requirements of the system. The system processor is a recently introduced embedded media processor, which provides a fine balance between performance versus cost and power characteristics. Camera modules are also one of smallest in the market measuring only 6.5 x 6.5 x 4.84mm and consuming 40 mW in active mode. Enhanced power saving features of the system such as multiple power management modes (i.e. sleep, hibernate, full-on and active), and also the isolated power domains for components, add to the system flexibility and efficiency in terms of power

Table 6.1: Comparison of the MESVS-II vs. SVM

Feature	MESVS-II	SVM
Processor	ADSP-BF561 (up to 600MHz)	ADSP-2181 (up to 50 MHz)
Dimensions (cm)	5x5	5x7.5
Performance (fps)	20	8
Disparity range (pixels)	30	16

requirements.

The outcome of this thesis has been two generations of miniaturized embedded stereo vision systems, i.e. MESVS-I and MESVS-II. MESVS-I is the first system prototype operating at about 10 fps, measuring only 5x5cm, and consuming 2.3 W. MESVS-II is an improvement upon the original system, which achieves real-time performance of 20 fps through careful optimization of the system firmware and deployment of enhanced memory and data management schemes. In the current implementation each MESVS unit costs about $500, which can be further reduced if mass manufactured in large quantities.

We have also proposed a new postprocessing algorithm that is efficient, i.e. is based on a fast recursive implementation. Furthermore, as supported by the experimental results, our postprocessing algorithm effectively removes outliers caused by low-texture and depth discontinuity areas of the scene and alleviates the so-called foreground fattening problem of the area-based matching methods.

Table 6.1 presents a comparison of MESVS-II versus Small Vision Module (SVM) (See Figure 3.1) in terms of feature characteristics. SVM is developed at SRI in 1997 and is chosen for this comparison as indeed it is the closest competitor of our system due to also being miniaturized, relying on a DSP processor, and finally being one of the most popular small vision modules in the literature. As shown MESVS-II is more compact, has higher processing rate, and covers a larger disparity range using a new embedded media processor that is quite faster than one used in SVM.

6.2 Future Work

Following are some of the potential avenues of research that may be taken in the future to further enhance the performance or/and functionality of MESVS:

- Although current system is based on embedded media processor, future research may consider an FPGA based implementation of the system as the latest FPGA products on the market seem like a promising alternative.
- Development of an online self-calibration scheme (as compared to the offline method currently deployed) would allow the stereo matching engine to dynamically adapt to the slight changes of internal and/or external parameters of the system and produce consistent and accurate results over the time.
- The feasibility of applying some variant of global matching methods (e.g. fast algorithms based on dynamic programming) as the base of stereo matching engine may be explored.
- Replacing current camera modules with a higher resolution one (e.g. OV2640-FSL), may be considered in order to improve the offline calibration accuracy and therefore the quality of the final obtained disparity maps
- Develop an adaptive thresholding scheme for the thresholds used in the postprocessing algorithm to improve its flexibility
- Incorporate BT measure into the matching process in order to eliminate sensitivity of the matching algorithm to the image sampling noise may be useful

References

[1] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient Belief Propagation for Early Vision", International Journal of Computer Vision, Vol. 70, No. 1, October 2006, pp. 41-54.

[2] Y. Boykov, O. Veksler, and R. Zabih, "Fast Approximate Energy Minimization Via Graph Cuts", IEEE Transaction on Pattern Analysis and Machine Intelligence, Vol. 23, No. 11, November 2001, pp. 1222-1239.

[3] J. Yunde, Z. Xiaoxun, L. Mingxiang, and A. Luping, "A miniature stereo vision machine (MSVM-III) for dense disparity mapping", ICPR, August 2004, pp. 728-731.

[4] C. Murphy, D. Lindquist, A. M. Rynning, T. Cecil, S. Leavitt, and M. L. Chang, "Low-Cost Stereo Vision on an FPGA", IEEE FCCM, April 2007, pp. 333-334.

[5] L. D. Stefano, M. Marchionni, and S. Mattoccia, "A PC-based real-time stereo vision system", International Journal of Machine Graphics and Vision, Vol. 13, No. 3, January 2004, pp. 197-220.

[6] H. Kim, D. B. Min, S. Choi, and K. Sohn, "Real-time disparity estimation using foreground segmentation for stereo sequences", Optical Engineering, Vol. 45, No. 3, March 2006, pp 037402 (10 pages).

[7] Y. Ruigang, M. Pollefeys, and L. Sifang, "Improved Real-Time Stereo on Commodity Graphics Hardware", IEEE CVPR Workshop, June 2004, pp. 36-44.

[8] L. Wang, M. Liao, M. Gong, R. Yang, and D. Nister, "High-Quality Real-Time Stereo Using Adaptive Cost Aggregation and Dynamic Programming", 3DPVT 2006, pp. 798-805.

[9] G. van der Wal, M. Hansen, M. Piacentino, "The Acadia vision processor", Proceedings of 5th International Workshop on Computer Architecture for Machine Perception, Padova, Italy, 2001, pp. 31-40.

[10] N. Chang, T.M. Lin, T.H. Tsai, Y.C. Tseng, and T.S. Chang ,"Real-Time DSP Implementation on Local Stereo Matching", IEEE ICME, July 2007, pp. 2090-2093.

[11] "Blackfin Embedded Symmetric Multiprocessor ADSP-BF561 Datasheet" available online at www.analog.com.

[12] D. Scharstein and R. Szeliski, "A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms", International Journal of Computer Vision, Vol. 47, No. 1-3 , April 2002, pp. 7-42.

[13] "Camera Calibration Toolbox for Matlab" available online at www.vision.caltech.edu/bouguetj/calib_doc.

[14] R. Zabih and J. Woodfill, "Non-parametric Local Transforms for Computing Visual Correspondence", ECCV 2004, pp. 151-158.

[15] H. Hirschmuller and D. Scharstein, "Evaluation of Cost Functions for Stereo Matching", IEEE CVPR, June 2007, pp. 1-8.

[16] L. D. Stefano, M. Marchionni, and S. Mattoccia, "A fast area-based stereo matching algorithm", Journal of Image and Vision Computing, Vol. 22, No. 12, October 2004, pp. 983-1005.

[17] G. Egnal and R.P. Wildes, "Detecting Binocular Half-Occlusions: Empirical Comparisons of Five Approaches", IEEE Transaction on Pattern Analysis and Machine Intelligence, Vol. 24, No. 8, August 2002, pp. 1127-1133.

[18] "Estimating Power for ADSP-BF561 Blackfin Processors", Analog devices Engineer-to-Engineer note 293, available online at www.analog.com.

[19] B. Khaleghi, S. Ahuja, and Q. M. Jonathan Wu, "A New Miniaturized Embedded Stereo-Vision System (MESVS-I)", CRV 2008 (to appear).

[20] B. Khaleghi, S. Ahuja, and Q. M. Jonathan Wu, "An Improved Real-Time Miniaturized Embedded Stereo-Vision System (MESVS-II)", IEEE CVPR Workshop on Embedded Computer Vision 2008 (to appear).

[21] D. J. Katz, R. Gentile, "Embedded Media Processing", New York: Elsevier, 2006.

[22] J. Banks, P. Corke, "Quantitative Evaluation of Matching Methods and Validity Measures for Stereo Vision", IJRR, 2001; 20; 512.

[23] H. Hirschmuller, "Improvements in Real-Time Correlation-Based Stereo Vision", IEEE SMBV 2001, pp. 141-148.

[24] M.P. Patricio, F. Cabestaing, O. Colot, P. Bonnet, "A Similarity-based adaptive neighborhood method for correlation-based stereo matching", IEEE ICIP 2004, pp. 1341-1344.

[25] M. Gong, and R. Yang, "Image-gradient-guided Real-time Stereo on Graphics Hardware", IEEE 3DIM 2005, pp. 548-555.

[26] H. Hirschmuller, "Accurate and Efficient Stereo Processing by Semi-Global Matching and Mutual Information", IEEE CVPR 2005, pp. 807-814.

[27] P. J. Burt, "A pyramid-based front-end processor for dynamic vision applications", IEEE Proceedings 2002, Vol. 90, No. 7, pp. 1188-1200.

[28] www.k-team.com

[29] http://www.tinyphoon.com

[30] http://www.swarm-bots.org

[31] L. Iocchi and K. Konolige, "A multiresolution stereo vision system for mobile robots", 1998.

[32] B. Barshan, and D. Baskent, "Comparison of two methods of surface profile extraction from multiple ultrasonic range measurements", Measurement science technology, Vol. 11 2000 , pp. 833-844.

[33] W. E. L. Grimson, "Computational experiments with a feature based stereo algorithm", IEEE TPAMI, Vol. 7, No. 1, pp. 1734, 1985.

[34] S. B. Pollard, J. E. W. Mayhew, and J. P. Frisby "A stereo correspondence algorithm using a disparity gradient limit" Perception, Vol. 14, pp. 449470, 1985.

[35] K. Prazdny, "Detection of binocular disparities", Biological Cybernetics Vol. 52, No. 2, pp. 9399, 1985.

[36] S. Birchfield, and C. Tomasi, "A Pixel Dissimilarity Measure That Is Insensitive to Image Sampling", IEEE PAMI, Vol. 20, No. 4, pp. 401-406, 1998.

[37] J. Sun, Y Li, SB Kang, and HY Shum, IEEE CVPR 2005, "Symmetric stereo matching for occlusion handling", Vol. 2, pp. 399-406.

[38] A. Cozzi, B. Crespi, F. Valentinotti and F. Wrgtter, "Performance of phase-based algorithms for disparity estimation", Journal of machine vision and applications, Vol. 9, No. 5-6, pp. 334-340, 1997.

[39] K. J.Yoon, and I. S. Kweon, "Adaptive Support-Weight Approach for Correspondence Search", IEEE PAMI, Vol. 28, No. 4, pp. 650-656, 2006.

[40] T. Kanade, and M. Okutomi, " A stereo matching algorithm with an adaptive window: theory andexperiment", IEEE PAMI, Vol. 16, No. 9, pp. 920-932, 1994.

[41] A. Fusiello, V. Roberto, and E. Trucco, "Efficient Stereo with Multiple Windowing", IEEE CVPR 1997, pp. 858-863.

[42] K. J. Yoon, and I. S. Kweon, "Stereo Matching with Symmetric Cost Functions", IEEE CVPR 2006, pp. 2371-2377.

[43] G. Li, and S. W. Zucker, "Differential Geometric Consistency Extends Stereo to Curved Surfaces", ECCV 2006, pp. 44-57.

[44] P. N. Belhumeur, and D. Mumford, "A Bayesian treatment of the stereo correspondence problem using half-occluded regions", IEEE CVPR 1992, pp. 506512.

[45] H. Saito, and M. Mori, "Application of genetic algorithms to stereo matching of images", Pattern Recognition Letters, Vol. 16, No. 8, pp. 815-821, 1995.

[46] T. Aydin, and Y. S. Akgul, "3D Structure Recovery From Stereo Using Synchronous Optimization Processes", BMVC 2006.

[47] J. P. Pascual Starink, and E. Backer, "Finding point correspondences using simulated annealing", Pattern Recognition Letters, Vol. 28, No. 2, pp. 231-240, 1995.

[48] http://vision.middlebury.edu/stereo

[49] W. Yang, L. Wang, R. Yang, S. Wang, M. Liao, and D. Nister, "Real-time Global Stereo Matching Using Hierarchical Belief Propagation", BMVC 2006.

[50] S. Forstmann, Y. Kanou, J. Ohya, S. Thuering, and A. Schmitt "Real-Time Stereo by using Dynamic Programming", IEEE CVPR Workshop on Real-time 3D Sensors and Their Use 2004, pp. 29-29.

[51] G. Van Meerbergen, M. Vergauwen, M. Pollefeys, and L. Van Gool, "A Hierarchical Symmetric Stereo Algorithm Using Dynamic Programming", International Journal of Computer Vision, Vol. 47, No. 1-3, pp. 275-285, 2002.

[52] M. Gong, and Y. Yang, "Near real-time reliable stereo matching using programmable graphics hardware", IEEE CVPR 2005, Vol I, pp. 924-931.

[53] R. Yang, and M. Pollefeys, "Multi-Resolution Real-Time Stereo on Commodity Graphics Hardware Multi-Resolution Real-Time Stereo on Commodity Graphics Hardware", IEEE CVPR 2003, pp. 211-217.

[54] T. Kanade, A. Yoshida, K. Oda, H. Kano, and M. Tanaka, "A Stereo Machine for Video-Rate Dense Depth Mapping and Its New Applications ", IEEE CVPR 1996, page 196.

[55] J. Woodfill, and B. Von Herzen, "Real-Time Stereo Vision on the PARTS Reconfigurable Computer", IEEE Symposium on FPGAs for Custom Computing Machines 1997, page 201.

[56] K. Konolige, "Small Vision Systems: Hardware and Implementation", Eighth International Symposium on Robotics Research 1997.

[57] M. Harti, Y. Ruichek, and A. Koukam, "A voting strategy for high speed stereo matching", , ICCVG 2004, pp. 187-196.

[58] H. Hirschmller, "Real-Time Correlation-Based Stereo Vision with Reduced Border Errors", IJCV 2002.

[59] L. Wang, M. Gong, M. Gong, R. Yang, "How Far Can We Go with Local Optimization in Real-Time Stereo Matching", 3DPVT 2006, pp. 129-136.

[60] O. Veksler, "Fast variable window for stereo correspondence using integral images," IEEE CVPR 2003, pp. 556-561.

[61] W. J. MacLean ,"An Evaluation of the Suitability of FPGAs for Embedded Vision Systems", IEEE CVPRW 2005, pp. x-x.

[62] "Comparing Mainstream DSP Processors: A survey", available online at http://www.dspdesignline.com/howto/198800216.

[63] S. Mahlknecht, R. Oberhammer, G. Novak, "A Real-Time Image Recognition System for Tiny Autonomous Mobile Robots", Journal of Real-Time Systems, 2005.

[64] "OV7660FSG/OV7660FSL Color CMOS VGA (640 x 480) Concept Camera Module with OmniPixelTM Technology", Datasheet, Omnivision technical info, version 1.1, 2004.

[65] J. Heikkil, and O. Silvn, "Calibration procedure for short focal length off-the-shelf CCD cameras", ICPR 1996, pp. 166-170.

[66] D. C. Brown, "Decentering Distortion of Lenses", Photometric Engineering, Vol. 32, No. 3, pp. 444-462, 1966.

[67] "A compact rectification algorithm",

[68] J. Banks, and P. Corke, "Quantitative Evaluation of Matching Methods and Validity Measures for Stereo Vision", IJRR, Vol. 20, page 512, 2001.

[69] B. Cyganek, "Comparison of Nonparametric Transformations and Bit Vector Matching for Stereo Correlation", Springer LNCS 3322 (2004), pp. 534-547.

[70] J. Lu, S. Rogmans, G. Lafruit, and F. Catthoor, High-Speed Dense Stereo via Directional Center-Biased Windows on Graphics Hardware, 3DTV 2007, pp. 1-4.

[71] http://www.pyramidvision.com/products/acadia/index.asp

VITA AUCTORIS

Bahador Khaleghi was born in 1982 in Tehran, Iran. In 2005, he received his Bachelor of Applied Science degree in Computer Engineering from Sharif University of Technology in Tehran, Iran. From there he went to Canada to pursue graduate studies. He has been working as a research assistant in the Computer Vision and Sensor Systems Lab under the supervision of Professor Jonathan Wu (CRC in automotive sensor systems) for the last two years.