EVALUATION OF STEREO MATCHING FOR MOBILE PLATFORMS WITH APPLICATIONS FOR ASSISTING THE VISUALLY IMPAIRED

BY

BENJAMIN WILSON CHIDESTER

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2013

Urbana, Illinois

Adviser:

Associate Professor Minh Do

ABSTRACT

The rising interest in immersive entertainment and enhanced image and video content, along with the development of stereo cameras for mobile platforms, motivates the presented evaluation of stereo matching algorithms for mobile devices. This work investigates this potential for stereo matching on a mobile device for real-time applications, in terms of computation time and quality of depth inference. Several algorithms are tested on an Android tablet housing a Tegra 3 processor using images captured from the on-board consumer-grade cameras. Despite distortions incurred by the lower quality cameras and the computational constraints of a tablet, results show that a simple block matching approach can perform reasonable inference at a rate of 10 frames-per-second. Other methods are shown to be too computationally demanding for real-time applications, as even the fastest alternative local method, using adaptive support weights, requires up to 20 seconds per frame on a 320x360 image. Results also show the impact of lower quality "real-world" images on inference performance on algorithms.

Additionally, real-time stereo matching on a mobile device is applied to a novel application of assisting the visually impaired with navigation. A system is proposed using a simple block matching algorithm that infers the depth of the scene and communicates the presence of obstacles via sound to the user. The system is housed entirely within the mobile device, overcoming a primary hindrance of many users of assistive technology. The use of a mobile device also allows for an intuitive, interactive experience for the user with the depth information directly via the touch screen of the device. This system demonstrates the added functionality of real-time depth estimation on a mobile device and the potential for aiding the visually impaired with navigation.

Nevertheless, I am continually with you;
you hold my right hand.
You guide me with your counsel,
and afterward you will receive me to glory.
Whom have I in heaven but you?
And there is nothing on earth that I desire besides you.
My flesh and my heart may fail,
but God is the strength of my heart and my portion forever.
- Psalm 73:23-26

To my mom and dad.

ACKNOWLEDGMENTS

Firstly, I would like to express my gratitude to my adviser, Minh Do. His confidence in me as his student has encouraged me in my research, and his thoughtful guidance has provided me with confidence in pursuing challenging and interesting problems. I greatly appreciate his sincere concern for me and my success as a graduate student.

I would also like to thank my labmates, both in my current research group and in my previous research group under the advising of Pierre Moulin. I enjoy their comradery and encouragement as friends in research and studies. I also thank Johan Vu at ADSC for providing test images for the experiments in this thesis.

I am also thankful to the Graduate Christian Fellowship for providing a great community that has encouraged me both in my studies and in my faith, as well as their integration. I am thankful for the many friendships that I have been fortunate to have in GCF, and of those friends, I would like to give special thanks to Michael DiPasquale and Michael Culbertson for their ongoing encouragement and support. I would also like to thank the Monday prayer gathering for interceding on my behalf for rest and productivity as I have worked on this thesis.

I must also thank my loving family, whose support I treasure greatly. Especially, I thank my mom and dad for more than I can write down, but what stands above all else is their selfless love and care. I am indebted to them in everything. I hope in all things to make them proud.

Lastly, I thank God, my Lord and Savior, Jesus Christ, who has loved me and gave himself for me. Though I am flawed, his grace is sufficient for me. Though I am not adequate in and of myself, he equips and qualifies me. I hope that I would not boast in any praise or recognition that I may receive from my research, but rather that the gaze of onlookers might be directed toward him.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Motivation

Stereo matching is a well researched problem within the computer vision community, as it long stood as the primary method of inferring the structure of a scene from captured images. Recently, there has been a growing interest in depth information for entertainment and image processing purposes, as the desire for more immersive entertainment and communication experiences grows. For example, depth cameras are becoming popular enhancements to gaming systems and video content, such as movies and TV shows [1, 2, 3, 4]. There is significant interest surrounding the exploration of enhancing the experience of cameras, webcams, gaming devices, and other devices with depth information. In particular, the potential of depth information to enhance the experience of a mobile device is yet to be seen.

Until recently, mobile devices lacked the computational power to be a viable platform for computer vision tools, such as depth inference via stereo matching, but Pulli *et al.* [5] demonstrated that this limitation no longer exists. Additionally, depth inference requires the presence of multiple cameras on the mobile device, which has only recently been pursued in response to the current growing interest in the enhancement of depth information. Several companies, such as LG, have recently released 3D phones, which house a pair of stereo cameras. NVIDIA has also developed prototype tablets, which house stereo cameras, to investigate the added value that such camera rigs can provide.

As mobile devices are becoming increasingly smaller, more portable, and more powerful, the consumer demand for these devices is growing significantly, and the evaluation of the potential for depth information on mobile devices is becoming increasingly more relevant. It is expected that tablets

will continue to grow as the dominant share of the computing market, and smart phones are now the standard for phones.

For many of the future developments of mobile devices, fast stereo matching algorithms would be a vital component. There are several reasons that motivate this. Firstly, in the effort to make devices smaller, industry and academic research are currently working to develop imaging systems comprised of arrays of cameras. Creating a high resolution image from a single camera requires that the focal length of the camera be prohibitively long for mobile devices, as the focal length is a determining factor of the thickness of the device. With an array of cameras, each individual camera can have a small focal length, allowing the device to be thinner, and a high resolution image can be constructed from a fusion of the data of the camera array. However, the images captured from each camera need to be aligned, which can be accomplished via stereo matching. The fast-rising start-up company, Pelican Imaging, was formed to solve this exact problem, and it has received a great amount of industry interest. Mobile devices with such camera arrays would have great potential as tools for depth inference via stereo matching.

Secondly, depth information extracted through stereo matching can be used for image enhancement, such as artistic effects like depth-of-field, or immersive experiences, such as view interpolation and 3D video. Computational photography, a recently growing area of research within computer vision and graphics, has many applications that could be brought to widespread consumer usage on a mobile device with the addition of depth information. Recently, there has been much interest in depth cameras to provide depth information, but they are currently too large to conform to a thin, mobile device. Depth cameras also function only in indoor environments, limiting possible applications.

Beyond adding value to consumer devices, depth inference on mobile devices also has a practical application as an assistive tool for navigation for the visually impaired. According to a recent study on navigation assistive devices [6], though many technological devices have been proposed, much of the modern technology has not made a significant impact on the visually impaired community, and many of these devices have either not matured beyond the prototype and development phase or have left the market. The primary undesirable qualities of such devices are that they are inconvenient, socially embarrassing, or not useful enough to warrant adoption by users [7, 8]. Since

a mobile device is likely already owned by a visually impaired user and could be used inconspicuously, it offers great potential as an assistive tool for navigation, which motivates the development of such a tool using depth inference via stereo matching.

A challenge with an assistive device is communicating spatial information to the user in a way that is not overly intrusive to other sensory inputs, such as touch or sound, while being useful and intuitive to understand. Currently, it is not well understood which sense is best to use in this communication [9]. However, a system on a mobile device could be interfaced into tactile sensors for touch signals or any kind of speakers for sound, depending upon which sense is more effective.

1.2 Related Work

The work of Pulli *et al.* [5] is the most closely related study for mobile devices; however, it addressed computer vision tasks generally, and did not thoroughly examine the timings and performance of the many stereo matching algorithms that exist. Vu *et al.* [10] proposed an application of depth on mobile devices to render a synthetic depth-of-field effect, but the application has not yet been ported to the mobile device to test timings. To the author's knowledge, no other work exists that considers stereo matching on mobile devices.

There is a great amount of prior work that considers the application of technology and even computer vision to assisting the visually impaired. Many of these systems require additional mounted hardware and do not infer depth inference from the on-board cameras of a mobile device, such as the stereo vision system developed by the University of Southern California [11], which uses a bumblebee camera system mounted to the head of the user. Also, estimating depth inference in these systems is done on a laptop and not on a smart phone or tablet.

The "Listen2dRoom" system [8], developed at Georgia Tech's Sonification Lab, is related, in that it is a stand-alone mobile app, requiring no additional hardware. However, this system does not attempt to infer depth information, rather, it detects objects in the room and informs the user of their presence and position relative to one-another.

Another well-known visual assistance system that resembles the proposed system in this thesis is "The vOICe," which was created by Peter Meijer. This system translates visual information of every pixel into sound by modulating a complex pattern of audible sine waves, in which the visual information is conveyed in the frequency and amplitude of the waves. This system differs from the one proposed primarily in that it uses only the color information captured from the camera and does not infer the depth.

Some systems have been proposed that use small tactile sensors to encode visual information, such as the BrainPort [12], which encodes spatial information via an electrode array placed on the tongue. However, some users have voiced a dislike for such stimulators [7], and they can require an extensive amount of training. The proposed system does not use touch stimulators, such as the BrianPort or the system developed by USC, as they may be unpleasant for the user and may hinder adoption.

1.3 Contribution

The primary contribution of this thesis is to provide a thorough analysis of the current potential of stereo matching on mobile devices. In previous work on stereo matching, proposed algorithms have been analyzed on data sets of high resolution and high quality images, such as the well-known Middlebury data set [13], using the computational power of desktop PCs, whereas the evaluation of this thesis considers real-world images captured from inexpensive cameras on a tablet and the constraint of mobile computing power. The platform for this evaluation will be the previously mentioned NVIDIA developer tablet, which houses an NVIDIA Tegra 3 processor, the standard processor on current Android devices.

Secondly, this thesis will demonstrate the potential for depth information on mobile devices to aid the visually impaired in navigation. It will be shown that reliable depth can be estimated in real-time on a mobile device and relayed to the user to help with obstacle detection and avoidance. To keep the system nonintrusive and intuitive, a novel depth-to-sound interface is proposed that allows the user to interact with several parameters of the mapping.

1.4 Outline of Thesis

This thesis is outlined as follows. First, in Chapter 2, the various stereo matching methods and algorithms that have been proposed in the literature are grouped and described, taking note of their computational complexity and estimation accuracy. Chapter 3 describes the method used for evaluating several of these algorithms and reports their performance on a mobile device. Chapter 4 describes the integration of stereo matching into a proposed system on a mobile device for assisting the visually impaired in navigation. Chapter 5 concludes the thesis.

CHAPTER 2

STEREO MATCHING REVIEW

Before considering the use of stereo matching on mobile devices and its usefulness for navigation for the visually impaired, it is necessary to review the stereo matching problem and the approaches that have been proposed to solve it.

2.1 Stereo Matching Problem and Method

A stereo system requires a pair of calibrated stereo cameras. Given a stereo pair of images captured by the imaging system, the objective of stereo matching is to infer the disparity between pixels in the left and right images. Let I_L and I_R denote the images from the left and right viewing perspectives, respectively. To restrict the search complexity of stereo matching, the images are aligned, according to a rotation and translation, so that corresponding points lie along the same scanline. This process is known as rectification.

Once the stereo image pair is rectified, the depth of each point in the scene can be inferred from the disparity between its location in the two image planes. This disparity exists because of parallax, as objects in the scene that are close to the camera will shift in location from one image to the other. Objects that are closer to the camera will experience a greater shift between images than objects that are further in the background. Since the images are rectified, this disparity will only be horizontal, and for a point in the left image, its corresponding point in the right image will be shifted toward the *left*.

For a given pixel $\vec{p} = (p_x, p_y)$ in the left image I_L, with intensity $I_L(\vec{p})$, its corresponding location in the right image I_R will be $\vec{p} - \vec{d}$, where $\vec{d} = (d, 0)$ is the disparity induced by the parallax. Based on the baseline distance between the two cameras and the camera focal length, the depth is related to the

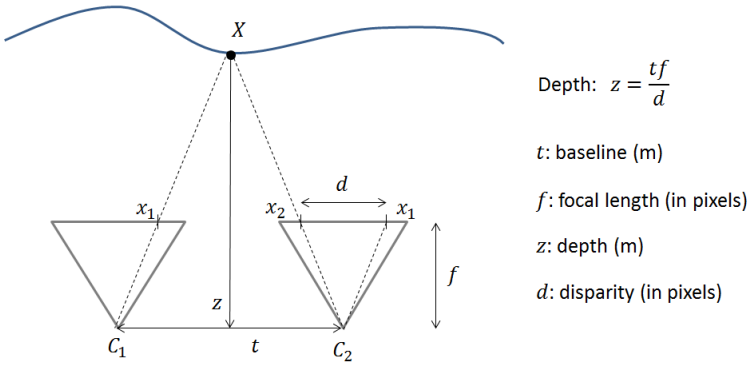


Figure 2.1: Relation of depth, focal length, baseline, and disparity in stereo-based depth inference.

disparity as shown in Fig. 2.1. To limit computation, a reasonable maximum resolvable depth is determined, which corresponds to a maximum disparity D, restricting the disparity search space for the disparity $\{0, 1, ..., D\}$. It should be noted that the disparity search space could include fractional shifts for subpixel accuracy. After the disparity at each pixel is inferred, these disparities comprise a *disparity map*, $\mathcal{D}$.

To determine the disparity for a given pixel between the two images, we must match the pixel $\vec{p}$ in the left image with its corresponding pixel in the right image. The method to determine this matching is what differentiates the many approaches of stereo matching. In the taxonomy by Scharstein and Szeliski [14], these approaches are categorized into *local* and *global* methods. Despite the differences of these two classes of approach, nearly all methods are comprised of the following four stages, also described by Scharstein and Szeliski:

1. matching cost computation
2. cost aggregation
3. optimization or cost minimization
4. disparity refinement

Generally, *local* methods are more suited for real-time performance, but they do not produce as accurate disparity maps as *global* methods, which are more computationally expensive. The following sections elaborate upon

the various methods within the *local* and *global* classes and consider the benefits and disadvantages of each, particularly within the context of mobile computing.

In this discussion, computational complexity will be a critical element for the evaluation of algorithms. For the purpose of notation, let N be the number of pixels in an image, W the window size, and D the number of possible disparity values.

2.2 Local Methods

Methods that infer the disparity at each pixel independently and do not invoke any global regularization or constraints are called *local* methods. Consequentially, these methods require the least computation and have the most potential for operating in real-time. Gong *et al.* [15] published a thorough study in 2007 on the variety of local methods that have been proposed. Even within this category, there exists a trade-off between computational cost and inference quality that must be assessed for the target platform of mobile devices.

In local methods, pixel correspondences are determined by optimizing a similarity metric between windows surrounding the pixel and candidate pixels. Two of the computationally simplest similarity metrics are (SAD) or sum of squared differences (SSD) which are defined as follows:

$$SSD(p,d) = \sum_{q \in \omega_p} \left(I_L(q) - I_R(q-d)\right)^2,$$
$$SAD(p,d) = \sum_{q \in \omega_p} \left|I_L(q) - I_R(q-d)\right|.$$

For these metrics, the matching cost is the absolute difference or squared difference, respectively, and the cost aggregation step is the summation of the matching cost of pixels within the windowed neighborhood of the reference pixel p. Another standard similarity metric is the normalized cross-correlation (NCC). Using this metric, the matching cost computation and

cost aggregation steps are merged. The NCC metric is defined as:

$$NCC(p,d) = \frac{\sum_{q\in\omega_p} I_L(q) I_R(q-d)}{\sqrt{\sum_{q\in\omega_p} \left(I_L(q)\right)^2 \sum_{q\in\omega_p} \left(I_R(q-d)\right)^2}}.$$

Although NCC is more computationally expensive than SSD or SAD, it is more robust to illumination variation between images. Since changes in lighting can be a common issue in stereo matching, still more advanced metrics have been proposed, including mutual information [16] and the rank or census transform. The census transform represents each pixel as a bit string based on the intensity structure of the surrounding region and the similarity between two pixels is the Hamming distance, which can be computed efficiently and has made this method popular for its trade-off of computation and disparity map quality. For subpixel accuracy, Birchfield and Tomasi [17] proposed a similarity metric based on linear interpolation, which is also computationally efficient, although it is still more expensive than SSD or SAD. For rectangular windows, the cost aggregation step is performed efficiently using the integral image technique [18] in $O(N)$ time, which is crucial for real-time performance.

These window-based matching methods struggle due to several primary reasons. (1) Finding correspondences is inherently ambiguous, as occlusion makes matching ill-defined. Additionally, when aggregating the cost within windows, occluding pixels may contribute to the aggregated cost, causing mismatches. (2) Pixels corresponding to different objects will experience different shifts between the two stereo images, as objects in the foreground will shift more than objects in the background, whereas window matching works well only under the assumption that pixels within a window shift by the same amount. (3) An additional challenge is "textureless regions," which are image regions of mostly uniform intensity. For such reasons, it is difficult to distinguish between different shift amounts because the content of the window does not change significantly throughout the uniform region. The first two of these challenges are addressed using what can be categorized as *adaptive support weights*. The latter challenge will be addressed by *global* methods.

2.2.1 Adaptive Support Weights

To address the first two challenges, local methods have been modified based upon the general intuition that pixels of similar intensity are likely to be of the same object, and therefore the same disparity, within a local window. Removing the contribution of objects of a different disparity than that of the current correspondence pixel candidate would overcome this issue. In 2006, Yoon and Kweon [19] prosed one of the most successful local methods in terms of disparity accuracy according to the results of their algorithm on the Middlebury benchmark [13]. Their method introduced adaptive support weights for windows based on spatial and intensity similarity. Since this method has produced significant disparity map quality, most recent local methods have been based on some form of this same intuition. However, adaptive support weights can be considered a form of linear translation-variant filtering, which generally requires $O(WN)$ computation. The following methods have attempted to lower the computational complexity by approximations to achieve real-time performance.

Bilateral Filter

The bilateral filter is a popular image processing technique used often for edge-preserving filtering and denoising. In the context of stereo matching, it is used to aggregate contributions of pixel differences within a window that belong to the same object. Given a pixel location $p \in \mathbb{R}^2$ in the left image I_L, the matching cost for a pixel in the right image I_R for a given disparity d is given by

$$e_p(d) = \min\left\{|I_L(p) - I_R(p-d)|, \sigma\right\}.$$

The absolute difference could be replaced by another metric, such as squared difference or the census transform. This formulation also introduces truncation, which is used in most algorithms during the matching cost computation to reduce the effect of outliers [15]. The resulting $3D$ matrix is commonly referred to as the *disparity space image* (DSI). The elements of the DSI that lie within the window of p are weighted according to the weight kernel w_p,

and the aggregated cost is calculated as

$$C_p(d) = \sum_{q \in N_p} w_p(q) e_q(d). \tag{2.1}$$

In the standard SAD method, these weights would be set to 1. $C_p(d)$ is also a $3D$ matrix and is often referred to as the *cost volume*. Based on the intuition mentioned previously, it is desirable to give more weight to pixels that have similar intensity to p, along with pixels that are more spatially similar to p. The weights for each pixel are defined by the joint bilateral filter, given by

$$w_p(q) = \frac{1}{K_p} \exp\left(-\frac{|p-q|^2}{\sigma_s^2}\right) \exp\left(-\frac{|I_p - I_q|^2}{\sigma_I^2}\right), \tag{2.2}$$

where the constant K_p is a normalizing constant such that $\sum_{q \in N_p} w_p(q) = 1$ and the parameters σ_s^2 and σ_I^2 determine the variance of the kernel in the spatial and intensity dimensions, respectively.

An exact implementation of the joint bilateral filter is too computationally expensive for real-time applications, as it requires translation-variant filtering on the DSI. Some methods have proposed a more efficient implementation using a histogram approximation of the filter. In another approach, from 2011, Min *et al.* [20] proposed several computation reductions to the cost aggregation step, including spatial sampling of the matching window and a compact representation of the DSI.

Guided Filter

Another attempt at real-time performance was proposed by Rhemann *et al.* [21] in 2011 based on the recently published Guided Filter [22], which is also an edge-preserving filter, similar to the joint bilateral filter but with $O(N)$ computation complexity. The key assumption used in this approach is that, within a local window, the cost volume for each disparity is approximately a linear function of the reference image. This assumption stems from the intuition that edges in the reference image should also correspond to edges in the disparity map. The linear model for the cost volume is given as

$$C'_d(q) = a_p I(q) + b_p, \forall q \in \omega_p. \tag{2.3}$$

The optimal value of the parameters $\{a_p, b_p\}$ are those which minimize the squared error between the pre-filtered cost volume $C_d(q)$ and the filtered cost volume $C'_d(q)$. The energy function for the regularized squared error is given as

$$E(a_p, b_p) = \sum_{q \in \omega_p} \left((a_p I(q) + b_p - C_d(q))^2 + \epsilon a_p^2 \right), \tag{2.4}$$

where ϵ is a regularization parameter on the size of a_p. The model in Equation (2.3) can indeed be written in the form of linear filtering as in Equation (2.1), where the weights corresponding to the optimal parameters are given by

$$w_p(q) = \frac{1}{|\omega|} \sum_{\omega_k} \left(1 + \frac{(I(p) - \mu_k)(I(q) - \mu_k)}{\sigma_k^2 + \epsilon} \right).$$

The key to the Guided Filter is that the filtering operation can be written in terms of box filtering operations, which can be computed efficiently via the integral image technique in $O(N)$ time.

2.2.2 Adaptive Support Regions

The same intuition that is used in adaptive support weights, that neighboring pixels of similar similarity likely have the same depth, has led to another approach proposed by Lu *et al.* [23]. In their approach, the matching window is constructed based on intensity similarity, but weighted filtering is not used, which avoids this additional computational complexity. For each pixel p, a cross is defined $\left\{h_p^-, h_p^+, v_p^-, v_p^+\right\}$, where these elements are the minimum distances horizontally and vertically from p to a pixel of significantly different intensity. The computational complexity of this method is $O(N)$, and with the aid of GPU optimization, it can achieve real-time performance for small images with a small baseline.

2.3 Global Methods

Global methods generate a disparity map by minimizing a global energy function instead of minimizing window matching costs as in local methods. The intuition behind global methods is that, since scenes are generally comprised of objects of nearly constant disparity, the disparity map should be piece-

wise smooth. To enforce this piece-wise smooth criterion, the global energy function includes a penalty on the gradient of the disparity map in addition to the penalty for pixel intensity dissimilarity.

In general, the energy function can be written as:

$$E(\mathcal{D}) = E_{data}(\mathcal{D}) + \lambda E_{smooth}(\mathcal{D}). \tag{2.5}$$

The data term $E_{data}(\mathcal{D})$ should be relatively smaller for pixels that are more similar in intensity. The smoothness term $E_{smooth}(\mathcal{D})$ measures the amount of variation of the disparity map within small neighborhoods of pixels and penalizes for high variation.

Although such truly global approaches result in the best quality disparity maps, according to the Middlebury benchmark [13], they are computationally expensive, since solving for the optimal disparity map to minimize the total energy is NP-hard. Even solving for the approximate minimizer using a method such as loopy belief propagation or graph cut requires significantly more computation then local methods [24]. Therefore, such methods are not generally considered for real-time applications, though, for applications on a mobile device that are not processing video data, global methods could still be a viable option.

To make solving the global energy function in (2.5) computationally feasible, *semi-global* methods have been proposed that relax the global assumption by reducing the problem to a $1D$ optimization problem, enforcing the smoothness constraint only along horizontal scanlines [14]. This relaxed formulation of the energy minimization problem can be solved efficiently using dynamic programming.

2.3.1 Dynamic Programming

Dynamic programming (DP) has become one of the classic methods for stereo correspondence, with the earliest work dating back to 1985 by Ohta and Kanade [25]. In the DP approach, each horizontal scanline is represented by a sequence of features $\{m\}_{i=1}^{M}$, which could be line segments [26], edges, or simply the pixels themselves [27]. These features are then matched between the left and right images to minimize a $1D$ optimization function to form the disparity map. The key to the efficiency of DP methods is the recursive

nature of the energy function. The general form for the energy function is given as:

$$E_{i+1}(d_i) = \min_{d_{i+1} \in D} \left(m(d_{i+1}) + s(d_{i+1}, d_i) + E_{i+2}(d_{i+1}) \right), \tag{2.6}$$

where $m(d_{i+1})$ is the matching cost for disparity assignment d_{i+1} and $s(d_{i+1}, d_i)$ is the penalty for the similarity between the consecutive disparities d_{i+1} and d_i. The matching cost could be based on absolute difference, squared difference, or any other similarity function. The disparity similarity term is usually of the form

$$s(d_{i+1}, d_i) = \begin{cases} 0 & \text{if } d_{i+1} = d_i, \\ \lambda_{i,i+1} & \text{otherwise,} \end{cases} \tag{2.7}$$

though a unique lower penalty is assigned in some algorithms for the case of $|d_{i+1} - d_i| = 1$ to allow for planar surfaces.

Though the reduction of the global smoothness constraint to $1D$ scanlines reduces the computational complexity involved, these $1D$ DP methods often suffer from poorer quality disparity maps. Particularly, the generated maps can contain what are commonly known as "streaking artifacts," since information between scanlines is ignored and interscanline disparity consistency is not enforced. These streaks are commonly long, isolated, horizontal segments of constant disparity.

Other DP methods have been proposed that optimize a closer approximation to the global energy function while maintaining a low computational complexity. These methods are reviewed in the following sections.

Tree Dynamic Programming

In 2005, Veksler [27] proposed an approach using DP on a tree, and it is illustrative of the DP method. Considering the 4-connected grid of pixels, each pixel p is represented as a node in the tree, and the potential edges are the directions in the 4-connected grid. Formally, let $G(V, E)$ be a tree graph, where V is the set of vertices and E is the set of edges. Let v be the current vertex, d_v the disparity at vertex v, and $p(v)$ the parent node of v.

The energy equation for a node v that is not the root node is given by

$$E_v(d_{p(v)}) = \min_{d_v \in D} \left(m(d_v) + s(d_v, d_{p(v)}) + \sum_{u \in C_v} E_u(d_v) \right), \tag{2.8}$$

where C_v is the set of children of v. The only difference between Equation (2.6) and Equation (2.8) is that in Equation (2.6), $|C_v| = 1$. The optimal disparity of the root node r evaluates to

$$D_r^* = \arg\min_{d_r \in D} \left\{ m(d_r) + \sum_{u \in C_r} E_u(d_r) \right\}. \tag{2.9}$$

To solve for the optimal disparity map D, D_v is evaluated for each node of the tree, beginning at the lowest level of the tree and working upward. Since for leaf nodes, the set C_v is empty, and the energy can be evaluated directly, they can be evaluated independently of all other nodes in the tree. D_v is then evaluated at the next highest level of the tree, since all of the children of nodes at this level will have already been evaluated, and the evaluation of the optimal disparity depends only on the child nodes. When the root node is reached, its optimal disparity D_r can be solved by Equation (2.9). The disparity chosen at the root is then propagated to the second level, and this propagation continues for each level until the lowest level of the tree is reached, at which point the optimal disparity map has been found.

A *minimum spanning tree* (MST) representation of the image meets the DP assumptions of Equation (2.8). To construct MST from the graph G, each edge is assigned a weight $v_{pq} = |I(p) - I(q)|$, so that weights are related to the pixel intensity similarity of the corresponding connected pixels. If we consider the similarity function in (2.9) of adjacent disparities, connected pixels with small intensities differences will have the largest weight $_{pq}$. The MST is constructed to retain the edges with the largest weights, as these edges will contribute the most to the overall energy of the graph. This also enforces the intuition that connected pixels of similar color will likely have the same disparity, and their disparity values should therefore inform one another. Computing the MST can be done in almost linear time, keeping the computation in the algorithm low.

In general, the computational complexity of solving the tree-based DP method is $O(ND^2)$, since the number of nodes in the graph is N and for

each node v, for each disparity $d_{p(v)}$, all disparities d_v must be considered. However, Veksler [27] proposed the use of the disparity similarity metric in Equation (2.7) to reduce the computational complexity. With this similarity function, for each disparity $d_{p(v)}$, only $d_v = d_{p(v)}$ and d_v^* which minimizes $m(d_v) + \sum_{u \in C_v} E_u(d_v)$ need to be considered among disparities, and this search can be done in constant time. Therefore, the computation time is only $O(ND)$, which is of the same order as the faster local stereo methods.

Tree Dynamic Programming with Line Segments

Although the proposed approach of Veksler [27] is efficient and an improvement over the $1D$ relaxation of the global energy function, still much of the information between neighboring pixels is lost in creating the MST. In 2006, Deng *et al.* [26] proposed a similar tree approach using line segments as nodes, instead of individual pixels. An efficient preprocessing step segments each horizontal scanline into regions of constant intensity. An MST is also created, based on the intuition that nodes with similar pixel intensity should be kept in the tree and the additional intuition for line segments that edges between segments with many neighboring pixels should also be kept. To account for planar surfaces in the disparity map, a 3-parameter linear transform space is used to represent the disparity of each segment. This method produces quality disparity maps at near real-time speeds.

Semi-Global Matching

In 2008, Hirschmüller [16] proposed a semiglobal matching approach that simultaneously considers $1D$ paths in every direction through the image, in order to maintain the reduced computational complexity of the 1D relaxation while enforcing more global consistency. This method helps to reduce significantly the streaking effect that plagues 1D relaxations.

2.4 Disparity Refinement

Independent of the stereo matching algorithm that is chosen, the estimated disparity map can be further refined during and after estimation by subject-

ing it in other ways to certain a priori assumptions, such as an assumption that the disparity map should generally be piece-wise smooth. Most of these methods are employed after inference has been completed, and generally, they do not add significant computation time to the overall process of disparity estimation.

One of the most common and intuitive refinement methods is called the *left-and-right consistency check*. Stereo matching requires the designation of a reference and target image, and outputs only one disparity map for this pairing. However, the roles of reference and target can be changed, which produces a disparity map for each view of stereo pair. This check enforces the assumption that the generated maps for both views should be consistent by setting the disparity to be zero at any matched points that do not share the same disparity in the two maps.

Since we know a priori that the disparity map is likely to be generally homogenous with edges at object boundaries, a common method to remove stray disparities is to apply a filter, such as a speckle filter or median filter. The median filter is more robust than local averaging to the presence of a small number of outlier pixels because it replaces pixel intensities with the median of a windowed region, which is unaffected by the presence of a small number of outliers, and not the average.

Another common technique is to enforce a *confidence interval* for matching scores. The motivation behind this technique is that if the algorithm cannot confidently assert what disparity is correct for a given pixel, then it is more robust to assign a disparity value of zero than to risk assigning an incorrect disparity. Furthermore, this situation is often encountered along large regions of uniform texture, such as the sky or a wall in a scene, in which case the region is likely in the background, so a disparity of zero is an intelligent estimate.

CHAPTER 3

STEREO MATCHING ON MOBILE DEVICES

Having reviewed the various methods that have been proposed for stereo matching, these methods will now be investigated for translation onto a mobile device. Since the interest of this evaluation is methods that have the potential to perform in near-real-time on a mobile device, any methods that are known to be prohibitively slow on modern desktops are not considered. Except for some semi-global methods, this set of candidate methods consists mostly of local methods.

A major advantage of running stereo matching on the mobile devices described in this thesis is the small baseline between the stereo cameras. While most of the literature reports on the Middlebury dataset or applications that require a large baseline, the mobile devices that are currently developed or envisioned have a small baseline, which is in some cases, such as a smart phone, defined by the physical limitations of the size of the device. For instance, the NVIDIA tablet that is used in the experiments in this thesis has a baseline of 65 mm. For the camera array developed by Pelican imaging, the baseline may only be a few millimeters. Although a small baseline limits the maximum disparity that can be inferred, even a baseline of a few millimeters still allows for inference of up to several meters. Table 3.1 shows the relationship between disparity and depth for the NVIDIA tablet and the maximum inferable depth. This small baseline is crucial to giving hope to achieving real-time computation on the mobile platform.

A disadvantage of the mobile platform is the consumer-grade quality of the cameras housed in these devices, which generate significantly more noise and less sharpness than the cameras that are often used in the stereo matching literature. Since most all algorithms are tested on the Middlebury dataset, which is of significantly higher quality than the images that will be generated for the experiments in this thesis, they are able to achieve highly accurate disparity maps. However, on mobile devices, such accuracy will be more

Table 3.1: Conversion of disparity to depth for the NVIDIA tablet with a baseline of 65 mm.

Disparity (pixels)	Depth (meters)
1	38.15
2	19.08
5	7.63
10	3.82
30	1.27
50	0.76

Figure 3.1: NVIDIA developer tablet with Tegra 3 processor and stereo cameras.

difficult to achieve.

3.1 Tegra 3 Specifications

Until just this year, the Tegra 3 processor was the standard processor across Android-based tablets, which makes it the primary candidate for a testbed. It is a quad-core processor with a maximum frequency of 1.6 Hz when using all four cores, and it is capable of holding up to 2 GB of RAM. It also houses a GPU, which could be used to further enhance performance of computer vision algorithms, such as the cross-based adaptive region method of Lu *et al.* [23]. In addition, the Tegra 3 processor also supports ARM's advanced SIMD extension called NEON, which allows for faster computation and was not available on the Tegra 2 processor of the previous generation. Figure 3.1 shows the NVIDIA tablet, which houses the Tegra 3 processor, that is used in this evaluation.

3.2 Rectification

To rectify the stereo cameras on the NVIDIA tablet, a set of 16 images of a checkerboard were captured for both the left and right views. Since consumer-grade cameras on a mobile device likely incur more distortion, the rectification process may not be able to estimate the camera parameters as accurately as on the Middlebury dataset. For the NVIDIA tablet, the resulting rectification produced an average pixel error of less than one pixel, which is acceptable for stereo matching. Figure 3.2 shows a rectified pair of stereo images of the checkerboard.

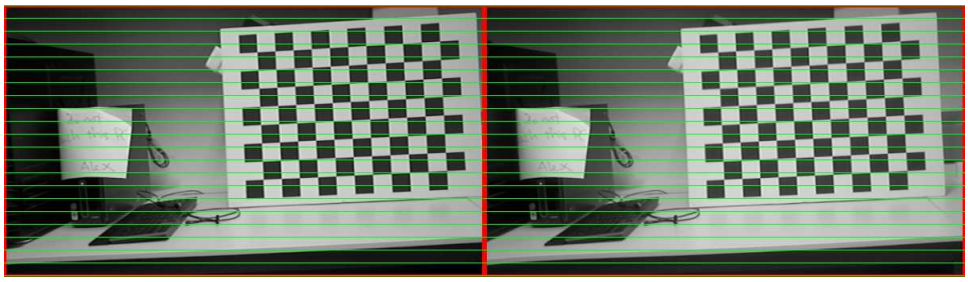


Figure 3.2: A rectified image of a checkerboard captured by the NVIDIA tablet and used for calibration. The horizontal lines allow for a visual check that the disparity between the left and right images is only horizontal.

3.3 Pre-Processing

In any stereo rig of cameras, each sensor will have its own color response or characteristics, and these differences have the potential to disrupt the matching process. To handle this, the images can be processed prior to performing stereo matching using a color-balancing algorithm, which should equalize the red, green, and blue content of both sensors. If a colorchecker is not available, a simple and effective color-balancing algorithm is the Grey World algorithm, described as follows:

$$\bar{G} = (\bar{r} + \bar{g} + \bar{b})/3, \tag{3.1}$$

$$r' = r(\bar{G}/\bar{r}), \tag{3.2}$$

$$g' = g(\bar{G}/\bar{g}), \tag{3.3}$$

$$b' = b(\bar{G}/\bar{b}), \tag{3.4}$$

where $\bar{r}$, $\bar{g}$, and $\bar{b}$ are the average pixel intensities of the red, green, and blue channels, respectively.

3.4 Experiment Set-Up on the Tegra 3 Tablet

The performance, in terms of computation time and estimation accuracy, of stereo matching was tested on the NVIDIA Tegra 3 tablet using rectified and white-balanced images. All processing, including rectification, white-balancing, and stereo matching, was conducted on the tablet and is including in the timing results. Some algorithms did not have an implementation available, and checking for correct implementation can be difficult, which limited the set of candidate algorithms. However, the following algorithms, which provide the most hope for achieving real-time performance on the tablet, are representative of the many stereo matching algorithms and provide an effective comparison for other algorithms:

- Semi-Global Block Matching (SGBM)
- SAD-based Block Matching (SBM)
- NCC-based Block Matching (NCC)
- Cost-Volume Filtering (CVF)

The SGBM algorithm is based upon the method of Hirschmüller [16] and uses the implementation of OpenCV. It is also the fastest global or semi-global method available for testing. The SBM algorithm also uses the implementation of OpenCV, which is highly optimized, making use of the NEON library for ARM processors, and is the fastest and simplest stereo matching method. The NCC algorithm was tested to investigate the effectiveness of NCC in

comparison to SAD on real-world images as a matching cost. Although NCC incurs more computational complexity, NCC has been reported to handle more robustly the color variation between cameras that often arises in real-world experiments. The CVF method is an implementation of the algorithm proposed by Rhemann *et al.* [21], which is the one of the fastest and most accurate algorithms that employs adaptive weights.

Several parameters of stereo matching are considered in this evaluation, which trade off speed and disparity accuracy: downsampling, disparity refinement, and window size. Disparity refinement of left-and-right consistency check, uniqueness ratio, and speckle filtering are considered for the SBM and SGBM methods. The maximum disparity is set based on the baseline of the stereo pair of cameras, which is 64 pixels for an image of width 640 pixels and 32 pixels for an image of width 320 pixels. Downsampling is performed only in the horizontal direction. The image height of both full-scale and downsampled images is 360 pixels. A testset of 22 images, captured from the NVIDIA tablet, were used in the evaluation. Timings that are reported are the run-times of the algorithm, averaged across these 22 images.

3.5 Results

The first question of interest in the evaluation is the computation time of the algorithms. Figures 3.3, 3.4, and 3.5 are plots of the average run-times for the four tested algorithms for different parameter settings. SBM is surprisingly efficient, and when operating on the downsampled version of the image (a resolution of 320x360), it is able to obtain average speeds of near or below 100 ms, which could equate to 10 frames-per-second while streaming, an acceptable rate for real-time processing. Since this method uses the integral image technique to efficiently compute matching costs of windows, its computational complexity does not depend on the window size, which is consistent with the plot. It is also worth noting that the disparity refinement does not incur a significant penalty in terms of computation time.

The next closest method in computation time is SGBM, which obtained average run-times of between 700 and 750 ms on the downsampled images and is therefore not a competitor for real-time processing. Therefore, the only candidate method for real-time processing is the simple and highly optimized

SBM method. NCC and CVF fare significantly worse, obtaining average run-times of between 4 and 5 seconds on the downsampled images and between 17 and 20 seconds on the full resolution images.

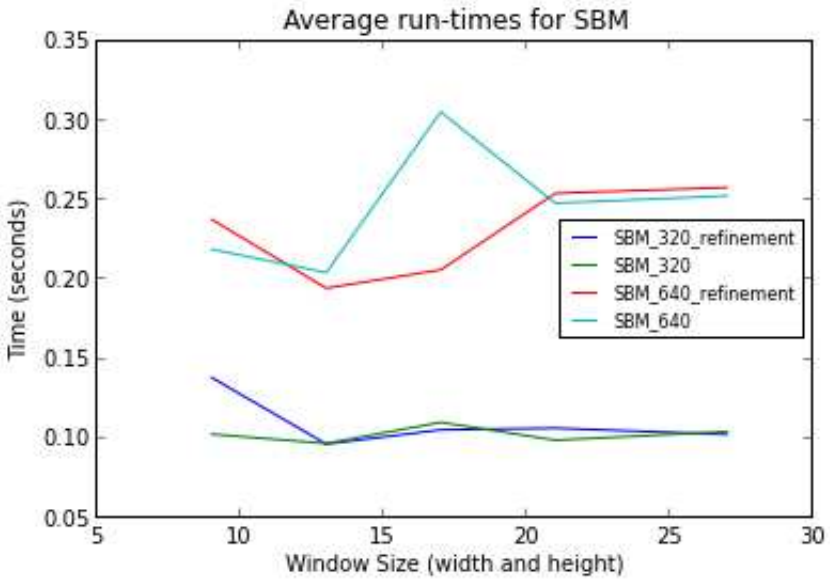


Figure 3.3: Run-times of SBM averaged across the dataset of 22 images. The parameters of window size, image resolution, and disparity refinement were considered. The suffix "320" indicates that the image width was 320 pixels.

The second aspect of the evaluation is the quality of inference of the tested algorithms. Since SBM was the only method to process frames in real-time, more attention is given to its produced disparity maps. Figure 3.6 shows the resulting disparity maps under a variety of different parameter settings, including downsampling, window size, and disparity refinement, for the motorbike image of the dataset, which has a balance of near and far objects and intricate object boundaries. As expected, the algorithm struggles along the large textureless regions of the structure in the left half of the image. Fortunately, with the assistance of the uniqueness ratio, the algorithm is able to avoid possibly erroneous estimates in this region by setting the estimates to zero disparity. Block matching methods are also notorious for widening the edges of foreground objects, which can be seen in the resulting disparity maps. However, overall, the disparity map is impressive given the less sophisticated algorithm of SBM. The consistency check handles occlusion at object boundaries, as seen along the right side of the person in the background.

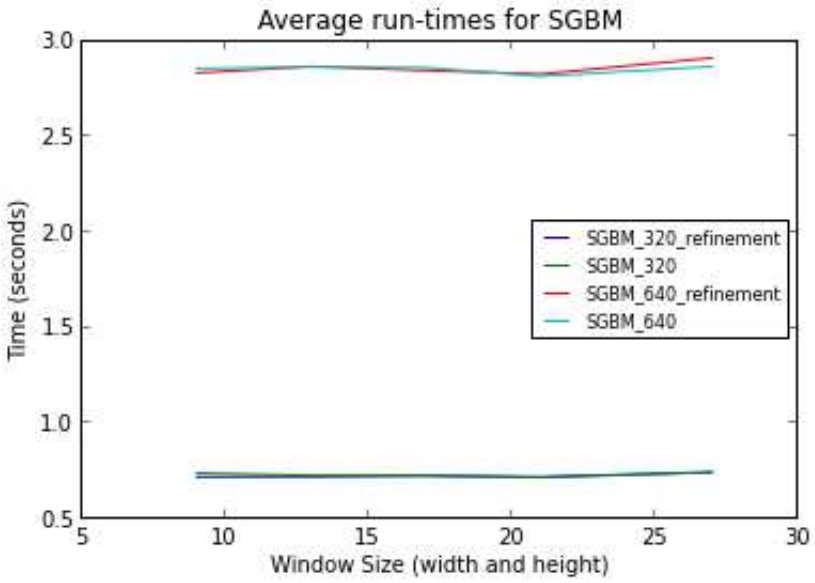


Figure 3.4: Run-times of SGBM averaged across the dataset of 22 images. The parameters of window size, image resolution, and disparity refinement were considered. The suffix "320" indicates that the image width was 320 pixels.

The algorithm also performs surprisingly well on the downsampled images, and the discrepancy of the performance on the full resolution images may not be significant for some applications. Note that the disparity maps that are shown have been rescaled to the full resolution of 640x360.

Results of disparity estimation of all four algorithms on several different test images are shown in Figures 3.7, 3.8, 3.9, and 3.10. With the help of a spatial smoothness penalty, SGBM is able to produce more contiguous estimates of objects than SBM, as seen in the truck and road of the truck image and the boxes of the shelf image. The NCC and CVF methods do not use a uniqueness ratio or left-and-right consistency check, which could both help in removing erroneous estimates. As seen in Figure 3.9, these methods both struggle with the background wall, although NCC does well in estimating the depth of the people. In Figure 3.8, CVF struggles with the road, another large region of uniform texture. In many of the images, only SBM is able to remove the erroneous large disparity estimates of the background and walls. For certain applications, the accurate results of NCC along object boundaries or the estimate of other more sophisticated methods may be needed, for instance applications that use the depth to guide image

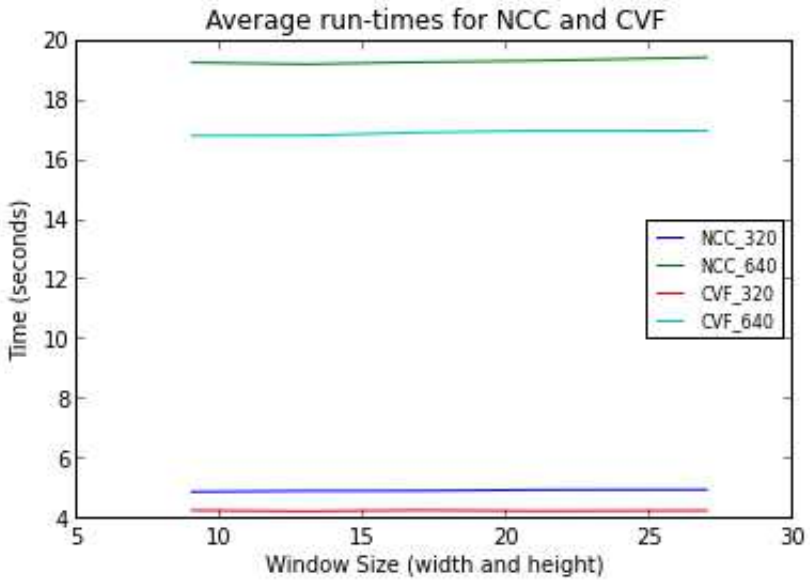
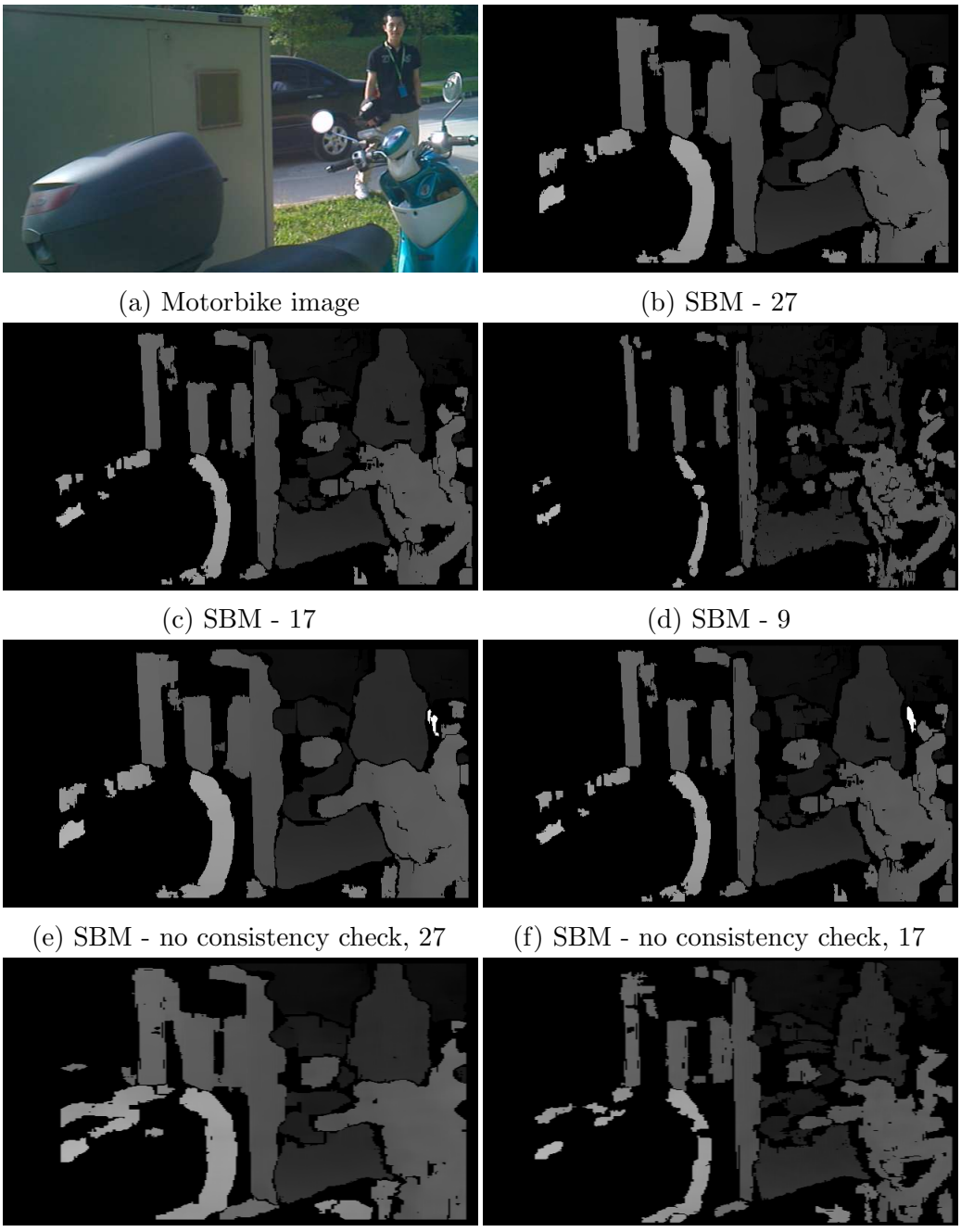


Figure 3.5: Run-times of NCC and CVF averaged across the dataset of 22 images. The algorithms were run both on the full resolution images and downsampled images. The suffix "320" indicates that the image width was 320 pixels.

filtering, but for the navigation assistance application to be described in Chapter 4, the results of SBM, which lack erroneous large disparity estimates, are favorable.



(a) Motorbike image (b) SBM - 27

(c) SBM - 17 (d) SBM - 9

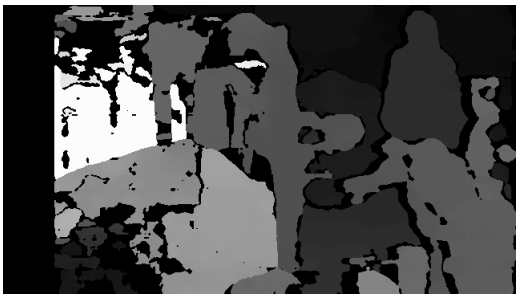
(e) SBM - no consistency check, 27 (f) SBM - no consistency check, 17

(g) SBM - downsampled, 17 (h) SBM - downsampled, 9

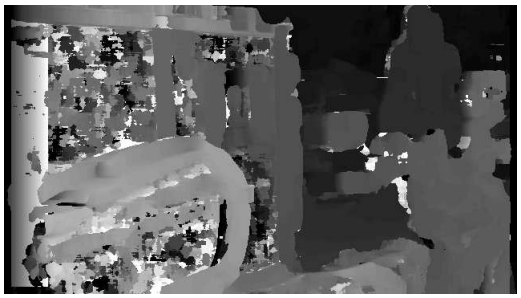
Figure 3.6: Disparity estimation results of the SBM algorithm on the motorbike image for a variety of different parameter settings. The number in the caption indicates the window size, in pixels, that was used for matching. Images (e) and (f) show the result without using the left-and-right consistency check. Images (g) and (h) show the result on the downsampled (320x340) image.

(a) Motorbike image

(b) SBM



(c) SGBM



(d) NCC

(e) CVF

Figure 3.7: Disparity estimation results on the full resolution (640x360) motorbike image with a window size of 17 pixels.

(a) Truck image

(b) SBM

(c) SGBM

(d) NCC

(e) CVF

Figure 3.8: Disparity estimation results on the full resolution (640x360) truck image with a window size of 17 pixels.

(a) Standing image

(b) SBM

(c) SGBM

(d) NCC

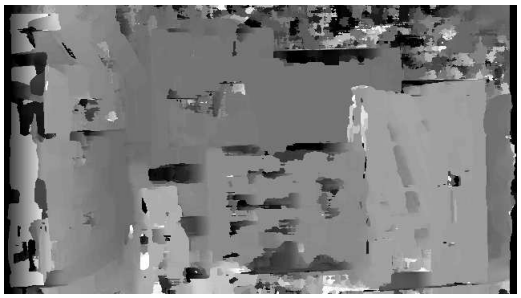
(e) CVF

Figure 3.9: Disparity estimation results on the full resolution (640x360) standing image with a window size of 17 pixels.

(a) Shelf image

(b) SBM

(c) SGBM



(d) NCC

(e) CVF

Figure 3.10: Disparity estimation results on the full resolution (640x360) standing image with a window size of 17 pixels.

CHAPTER 4

A SYSTEM FOR ASSISTING THE VISUALLY IMPAIRED

Having established the potential of stereo matching for real-time applications on a mobile platform, through use of the SBM method, this chapter examines its potential impact for assisting the visually impaired by proposing a novel system using stereo-matching.

4.1 System Overview

The proposed system is an entirely self-contained application on a mobile device. A flow-chart of the system is shown in Figure 4.1. Using the stereo cameras on-board the device, stereo images are captured and rectified and given as input to a stereo matching algorithm, which infers the depth of the scene. The depth is then converted to sound to be communicated to the user, though mappings to touch could also be used. The mapping of depth-to-sound relates the average depth in a region of interest in the inferred disparity map to the frequency of a sinusoidal sound wave that is output by the device. A novel aspect of this system is an interface based around the touch screen that allows the user to interact with the mapping by adjusting the scale of the region over which averaging is performed and the location of the region. The system is developed on the NVIDIA tablet, as shown in Figure 4.2, though it could be ported to any stereo-equipped tablet or smart phone.

The depth estimation is performed via stereo matching as evaluated in Chapter 3. The depth-to-sound transformation and user interface are described in the following sections.

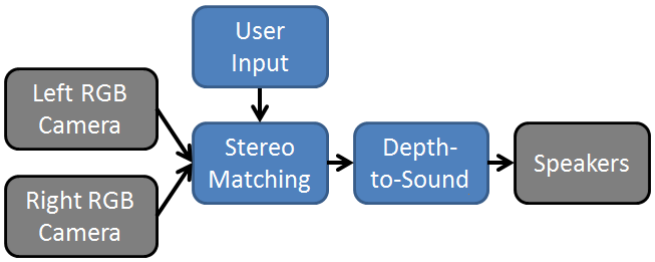


Figure 4.1: Flowchart of the proposed assistance system.

Figure 4.2: Use of the NVIDIA developer tablet for navigation assistance.

4.2 Depth-To-Sound Transformation

Once the depth of the scene has been estimated, it must be mapped to another sensory input to be communicated to the user. Both sound and touch have been studied as channels to communicate information to the blind, but neither sense has been deemed more well-suited for the task [9].

The advantage of touch as a channel is that it does not interfere with the user's ability to listen to the surroundings, which is known to be a crucial component of navigation for the visually impaired. Disadvantages of communication via touch are the lack of resolution of a human's touch sensory inputs and the possible physical or social discomfort caused by devices that communicate via touch. Touch is commonly transmitted through a tactile sensor or a vibrating motor that can be worn around the torso, which is the approach of the system of USC [11], or even on the tongue, which is the approach of the BrainPort [12]. Exotic sensors, such as the BrainPort, that encode visual information through complex spatial patterns can require extensive training and their effectiveness is still in question, which is another possible pitfall of depth-to-touch.

The transformation from data to sound is more straightforward if creating a low-fidelity signal of the visual information. However, richer descriptions are also challenging to capture in sound, and the research area known as *sonification* has developed to study such transformations. Although sound is a vital sense for the visually impaired user, the proposed system uses sound as a channel for communicating visual information and the presence of obstacles because the mapping is more straightforward for simple signals and requires no additional hardware, which allows the system to be contained entirely to the mobile device, making it more comfortable for use.

Inherently, the depth information that is inferred must be compressed before it is communicated to the user, as it is not possible to convey it in its fullness through sound. Though systems such as "The vOICe" have attempted to accomplish this, such systems have not been widely adopted, likely due to the difficulty of learning to interpret the generated sounds. A mapping to sound must also avoid over-interpreting the data for the user and be robust. For this proof-of-concept system, a simple transformation of averaging a region of the image and mapping the average depth to frequency is proposed. This mapping provides the user with a one-dimensional signal that encodes the distance of the content in the scene that is before them or in whatever direction the tablet is facing. To understand the surroundings more fully, the user can sweep the tablet in various directions to discover the depth of objects in those directions as well. This loss of dimensionality in the transformation from depth to sound requires that the depth inference of stereo matching on the mobile device need not be as precise as in other applications that are often studied in the literature. Therefore, though the resulting depth estimation of the evaluation on mobile devices is not as precise as those reported on the high-quality Middlebury dataset, especially at object boundaries, such precision is not necessary for this application, and estimation of the real-time SBM method will be sufficiently accurate.

To allow for greater scene understanding, the proposed depth averaging scheme is also variable in scale. The user can adjust the scale of the averaging window, which expands or restricts the field-of-view for which depth information is communicated. The most straightforward output sound to communicate the depth is a sine wave with frequency modulated by the depth, or possibly a beeping pattern with beeping frequency modulated by

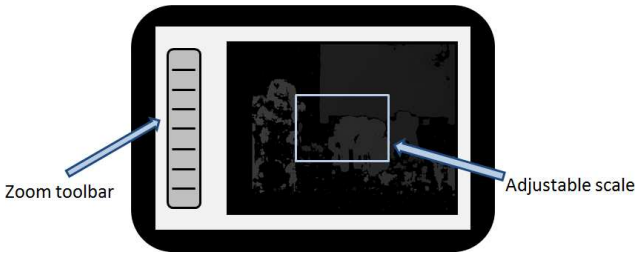


Figure 4.3: User interface on the tablet.

depth. The case of a sine wave can be described as follows:

$$s(t) = A \sin\left(\left(1 + \alpha_d\right) f_0 t\right), \tag{4.1}$$

where α_d is a monotonically increasing function of the disparity, d, ranging from 0 to α_{max}, and f_0 is a base frequency.

This mapping is also advantageous in that it is simple to compute, reducing possible overhead that would limit the frame processing rate of the application and it avoids any issues of robustness that a more intricate mapping would bring. Another advantage of deploying this system on a mobile device without any additional hardware is that many users would have access to the system without any barrier to entry and could test this mapping and other mappings to provide feedback as to what mappings are most useful.

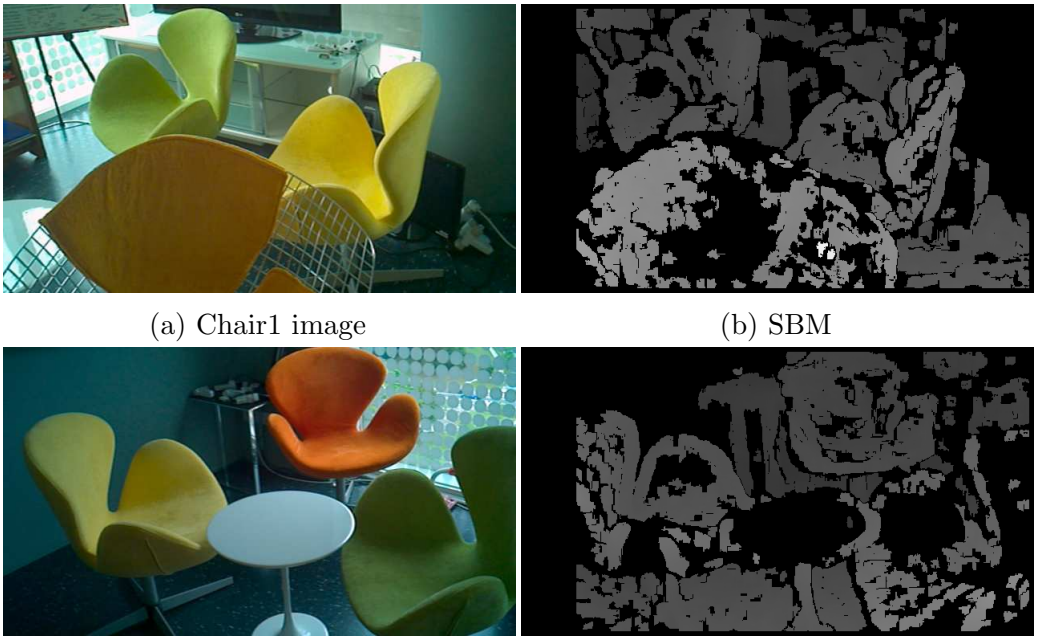
4.3 User Interface

Access to a touch screen allows for easy interaction between the user and the system to adjust two key parameters of the depth-to-sound mapping. The interface contains a toolbar that controls the scale of the region to be averaged, which allows the user to quickly adjust the field-of-view of the content that is being communicated while operating the device. Likely, a small field-of-view would be most informative about the scene.

By touching the screen, the user can also change the location of the averaging window, which allows the user to scan the depth of the scene without having to change the perspective of the tablet. Figure 4.3 shows the proposed user interface for the tablet.

4.4 Discussion

A prototype system was created that uses the SBM algorithm for disparity estimation. Separate threads for the camera and stereo matching were used, enabling the system to operate at approximately 10 frames-per-second. The disparity estimation from the SBM algorithm provides useful depth information of the scene to the user, although, due to the matching ambiguity and the constraint of the uniqueness ratio, the disparity estimation of some objects is sparse. A few images of common obstacles in an office environment are shown in Figure 4.4. Some objects are difficult to detect, such as the specular and homogeneous surface of the table, but much of the scene is still inferred.



(a) Chair1 image (b) SBM

(c) Chair2 image (d) SBM

Figure 4.4: Disparity estimation results in an indoor environment.

In the future, user testing will provide feedback on the usefulness of the proposed depth-to-sound mapping and possibly other mappings, the inferred depth information, and the user interface. Further enhancements to the mapping may be useful, such as removing the depth content of the floor from the disparity map to decouple it from the depth information of objects. Another technique that could be investigated is the use of a blob-detection algorithm to look for near objects. These post-processing techniques can

also be performed efficiently without significantly reducing the frame rate of the system. Additionally, the mobile device platform offers the potential for the generation of other sound signals, including the use of 3D sound, which could convey richer descriptions of the depth of the scene without further interfering with the sounds of the environment.

CHAPTER 5

CONCLUSION

This thesis has evaluated the performance of various stereo matching algorithms on a standard tablet and demonstrated that, with a frame rate of 10 frames-per-second, simple block matching is the primary candidate algorithm for real-time applications. Although not all existing stereo matching algorithms could be tested, any other algorithm would need to be highly optimized for the ARM processor to compete with computation time of the block matching and semi-global block matching methods that were tested. Algorithms that are optimized for the GPU could also be used to possibly obtain faster run-times. Additionally, the evaluation demonstrated that, although stereo matching algorithms on real-world images captured from consumer-grade cameras on mobile devices suffer degradation from noise and distortions from the camera, the tested algorithms are still able to infer depth with reasonable accuracy, though not as well as they would on the Middlebury dataset. The more accurate local methods, such as Cost Volume Filtering, could be used for applications that require higher quality disparity maps, but operation times of several seconds must be acceptable. For applications of enhanced and immersive multimedia experience, this thesis has shown that stereo matching on mobile devices is a viable option for depth inference.

Additionally, the feasibility of providing a user with useful depth information in real-time using only a mobile device has been shown using stereo matching. The proposed novel system is able to provide the user with information about the depth of obstacles in the navigated environment in real-time. The widespread use of mobile devices offers an extensive user test-bed to test the system and to further refine the communication of depth information to the user. Additionally, mobile devices offer convenience, mobility, and functionality, and they pose no physical or social discomfort to the user, making the proposed system attractive for use by the visually impaired community.

REFERENCES

[1] S. Zinger, L. Do, and P. de With, "Free-viewpoint depth image based rendering," *Journal of Visual Communication and Image Representation*, vol. 21, no. 5-6, pp. 533 – 541, 2010, special issue on Multi-camera Imaging, Coding and Innovative Display.

[2] A. Smolic, K. Mueller, P. Merkle, P. Kauff, and T. Wiegand, "An overview of available and emerging 3d video formats and depth enhanced stereo as efficient generic solution," in *Picture Coding Symposium, 2009. PCS 2009*, May 2009, pp. 1 –4.

[3] Y. Mori, N. Fukushima, T. Yendo, T. Fujii, and M. Tanimoto, "View generation with 3d warping using depth information for FTV," *Signal Processing: Image Communication*, vol. 24, no. 1-2, pp. 65 – 72, 2009, special issue on advances in three-dimensional television and video.

[4] D. Min, D. Kim, S. Yun, and K. Sohn, "2d/3d freeview video generation for 3dTV system," *Signal Processing: Image Communication*, vol. 24, no. 1-2, pp. 31 – 48, 2009, special issue on advances in three-dimensional television and video.

[5] K. Pulli, A. Baksheev, K. Kornyakov, and V. Eruhimov, "Real-time computer vision with OpenCV," *Commun. ACM*, vol. 55, no. 6, pp. 61–69, June 2012. [Online]. Available: http://doi.acm.org/10.1145/2184319.2184337

[6] U. R. Roentgen, G. J. Gelderblom, M. Soede, and L. P. de Witte, "Inventory of electronic mobility aids for persons with visual impairments: A literature review," *Journal of Visual Impairment & Blindness*, vol. 102, no. 11, pp. 702–724, 2008.

[7] R. Golledge, R. Klatzky, J. Loomis, and J. Marston, "Stated preferences for components of a personal guidance system for nonvisual navigation," *Journal of Visual Impairment & Blindness (JVIB)*, vol. 98, no. 03, 2004.

[8] M. Jeon, N. Nazneen, O. Akanser, A. Ayala-Acevedo, and B. Walker, "'Listen2dRoom': helping blind individuals understand room layouts," in *CHI '12 Extended Abstracts on Human Factors in Computing Systems*, ser. CHI EA '12. New York, NY, USA: ACM, 2012. [Online]. Available: http://doi.acm.org/10.1145/2212776.2223675 pp. 1577–1582.

[9] J. M. Loomis, R. Klatzky, J. Rieser, D. Ashmead, F. Ebner, and A. Corn, "Functional equivalence of spatial representations from vision, touch, and hearing: Relevance for sensory substitution," in *Blindness and Brain Plasticity in Navigation and Object Perception*, J. Rieser, D. Ashmead, F. Ebner, and A. Corn, Eds. New York: Lawrence Erlbaum Associates, 2008, pp. 155–184.

[10] D. T. Vu, B. Chidester, J. Lu, and M. N. Do, "Scribble2focus: An interactive photo refocusing system based on mobile stereo imaging," in *Proceedings of the 1st IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, ser. GlobalSIP '13, 2013.

[11] V. Pradeep, G. Medioni, and J. Weiland, "Robot vision for the visually impaired," in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on*, June 2010, pp. 15 –22.

[12] "Brainport." [Online]. Available: http://www.wicab.com

[13] "Middlebury stereo page." [Online]. Available: http://vision.middlebury.edu/stereo/

[14] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *International Journal of Computer Vision*, vol. 47, pp. 7–42, 2002.

[15] M. Gong, R. Yang, L. Wang, and M. Gong, "A performance study on different cost aggregation approaches used in real-time stereo matching," *International Journal of Computer Vision*, vol. 75, pp. 283–296, 2007.

[16] H. Hirschmüller, "Stereo processing by semiglobal matching and mutual information," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 2, pp. 328–341, 2008.

[17] S. Birchfield and C. Tomasi, "A pixel dissimilarity measure that is insensitive to image sampling," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 20, no. 4, pp. 401 –406, April 1998.

[18] O. Veksler, "Fast variable window for stereo correspondence using integral images," in *Computer Vision and Pattern Recognition, 2003. CVPR 2003. IEEE Computer Society Conference on*, 2003, pp. 556–561.

[19] K.-J. Yoon and I. S. Kweon, "Adaptive support-weight approach for correspondence search," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 28, no. 4, pp. 650 –656, April 2006.

[20] D. Min, J. Lu, and M. N. Do, "A revisit to cost aggregation in stereo matching: How far can we reduce its computational redundancy?" in *International Conference on Computer Vision (ICCV)*, 2011. [Online]. Available: http://diml.yonsei.ac.kr/ forevertin/ICCV2011_Final.pdf

[21] C. Rhemann, A. Hosni, M. Bleyer, C. Rother, and M. Gelautz, "Fast cost-volume filtering for visual correspondence and beyond," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, June 2011, pp. 3017 –3024.

[22] K. He, J. Sun, and X. Tang, "Guided image filtering," in *Computer Vision – ECCV 2010*, ser. Lecture Notes in Computer Science, K. Daniilidis, P. Maragos, and N. Paragios, Eds. Springer Berlin / Heidelberg, 2010, vol. 6311, pp. 1–14.

[23] J. Lu, K. Zhang, G. Lafruit, and F. Catthoor, "Real-time stereo matching: A cross-based local approach," in *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*, April 2009, pp. 733 –736.

[24] M. Tappen and W. Freeman, "Comparison of graph cuts with belief propagation for stereo, using identical mrf parameters," in *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, Oct. 2003, pp. 900 –906 vol.2.

[25] Y. Ohta and T. Kanade, "Stereo by intra- and inter-scanline search using dynamic programming," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. PAMI-7, no. 2, pp. 139–154, March 1985.

[26] Y. Deng and X. Lin, "A fast line segment based dense stereo algorithm using tree dynamic programming," in *Computer Vision – ECCV 2006*, ser. Lecture Notes in Computer Science, A. Leonardis, H. Bischof, and A. Pinz, Eds. Springer Berlin / Heidelberg, 2006, vol. 3953, pp. 201–212.

[27] O. Veksler, "Stereo correspondence by dynamic programming on a tree," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 2, June 2005, pp. 384–390.