STEREO MATCHING WITH TEMPORAL CONSISTENCY USING AN UPRIGHT PINHOLE MODEL

BY

YIBO JIANG

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Bachelor of Science in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2017

Urbana, Illinois

Adviser:

Minh N. Do

ABSTRACT

Stereo vision, as a subfield of computer vision, has been researched for over 20 years. However, most research efforts have been devoted to single-frame estimation. With the rising interest in autonomous vehicles, more attention should be paid to temporal consistency within stereo matching as depth matching in this case will be used in a video context. In this thesis, temporal consistency in stereo vision will be studied in an effort to reduce time or increase accuracy by utilizing a simple upright camera model. The camera model is used for disparity prediction, which also serves as initialization for different stereo matching frameworks such as local methods and belief propagation. In particular, this thesis proposes a new algorithm based on this model and sped-up patchMatch belief propagation (SPM-BF). The results have demonstrated that the proposed method can reduce computation and convergence time.

Subject Keywords: Stereo Vision, Temporal Consistency

To my parents, for their love and support

ACKNOWLEDGMENTS

I would like to express my gratitude to my advisor Prof. Minh N. Do for his guidance during the course of this project. I also want to thank Chen Chen for his support and knowledge for helping me finish my thesis.

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

AD	Absolute Difference
BF	Belief Propagation
BM	Block Matching
CBP	Context Guided Belief Propagation
FGI	Fast Guided Global Interpolation
HBP	Hierarchical Belief Propagation
MBM	Multi-Block Matching
MRF	Markov Random Fields
NCC	Normalized Cross-correlation
RANSAC	Random Sample Consensus
SAD	Sum of the Absolute Differences
SCD	Sum of Squared Differences
SLIC	Simple Linear Iterative Clustering
SPM-BF	Sped-up PatchMatch Belief Propagation
SURF	Speeded Up Robust Features

CHAPTER 1

INTRODUCTION

1.1 Motivation

Stereo vision, as a branch of computer vision, has been studied for over 20 years. It is the process of estimating depth information from a dual camera. There are other competing systems such as LIDAR that can extract depth information as well. The drawbacks of the stereo camera system are the high computational intensity, and the need for good lighting conditions of the environment which are not required by the LIDAR system. However, the LIDAR system is extremely expensive compared to a pair of high-resolution cameras. Stereo vision can provide high spatial resolution and lay the foundation for other usages like object detection and ego estimation [1]. Also, from a biological standpoint, humans use binocular depth cues as a primary source to sense the world without additional aids used by other animals like bats, which further shows that stereo matching can be an effective tool for depth sensing.

Modern research into stereo vision can be divided into three main categories. The traditional two branches are the local methods and the global methods. The local methods compute the disparity value at a given point depending only on intensity values within a finite window, and usually make implicit smoothness assumption by aggregating cost [2]. On the other hand, the global methods have explicit expression for a smooth term to be minimized. In fact, global methods always aim to minimize the energy function formulated as below:

$$Energy(D) = DataCost(D) + SmoothCost(D)$$

where D is the disparity map. Obviously, the direct minimization of energy function can be computationally challenging. Methods like loopy belief propagation [3] and many variants of belief propagation (BF) like hierarchical belief propagation (HBP) [4], context guided BP (CBP) [5] have been proposed to solve this problem, which can provide better disparity maps with less computational cost. With the recent success in neural networks, a new direction of stereo matching research has been focused on using the deep learning architecture for depth estimation, like Flownet, which has a correlation layer that explicitly provides matching capabilities [6], [7].

However, most research efforts have been devoted into studying the computation of disparity maps in one frame, even though the depth maps of consecutive frames are highly correlated. Studying how to incorporate temporal consistency into a current stereo matching framework can help increase the stability of the system and potentially reduce the computational cost. In this thesis, different current stereo vision frameworks will be tested by incorporating temporal consistency.

1.2 Thesis Outline and Contribution

The thesis is outlined as follows. Chapter 2 reviews two frameworks of stereo vision: the local methods and belief propagation. Chapter 3 describes the algorithm used in this thesis. Chapter 4 gives the experimental results. Chapter 5 concludes the thesis and Chapter 6 presents plans for future work.

The thesis starts with testing ego estimation to predict disparity in the next frame. However, this method is only limited to estimate static background. The limitation is partially caused by the unpredictability of objects moving in and out of the frame. This observation implies that it is practically impossible to do a perfect disparity prediction. In fact, the focus should be using temporal information to improve the current results. With this intention, the thesis proposes a new algorithm that uses a upright pinhole model as initialization to speed up belief propagation. The result is further improved by using Fast Global Guided Interpolation (FGI) [8] to interpolate downsampled depth map.

CHAPTER 2

RELATED WORK

2.1 Local Method: Block Matching

2.1.1 Overview

The stereo vision system is usually fed with two images from dual-cameras, and it uses the binocular information to estimate depth. Suppose the left image from the camera is I_L and right image is I_R. Based on simple parallax geometry, the corresponding points in the two images will be shifted by a disparity vector $\mathbf{d}$. The image pairs are usually rectified, thus the epipolar lines are aligned with the horizontal image axis [5]. This implies that if there is a pixel point $\mathbf{p}$ in I_L, we need to find a corresponding $\mathbf{q}$ in I_R such that $\mathbf{q} = \mathbf{p} - \mathbf{d} = \mathbf{p} + (d_x, 0)$. d_x is the x value of $\mathbf{d}$. Therefore, in this thesis, disparity value will be denoted as d which is equal to d_x. Then the depth can be inferred based on simple geometric relation in this formula:

$$\text{depth} = B\frac{f}{d_x} \tag{2.1}$$

where B is the baseline distance between the two cameras, and f is the focal length. Because of this, instead of directly deriving the depth map, the stereo vision system will be only focused on finding the disparity map based on the left and right images. In practice, the images are discretized into pixels, which often leaves the choice of the possible disparity value candidates to a discrete set as well. Particularly, the disparity value $d \in L$ where $L = \{1, ...l\}$. L is often referred to as the label space.

In the taxonomy by Scharstein and Szeliski [2], they described the basic stereo matching algorithm in four steps:

1. matching cost computation

2. cost (support) aggregation

3. disparity computation/optimization

4. disparity refinement

Step 3 is often just selecting the disparity labels that minimize the cost. The rest of this section will be devoted to the remaining three steps.

2.1.2 Matching Cost Computation

Suppose the images are rectified, and disparity vector is denoted as $\mathbf{d} = (d_x, 0)$. Then for any pixel $\mathbf{p}$ in the left image, the corresponding pixel in the right image is $\mathbf{p} - \mathbf{d}$.

One of the commonly used cost functions is absolute value (AD) [9] which simply calculates the norm of the corresponding vector difference. Its simplest implementation would be a straight pixel-wise computation. But local methods tend to use its variant, sum of the absolute differences (SAD) [9], that sums over all the pixels in the neighbor N_p of pixel $\mathbf{p}$. The formula of SAD is as follows:

$$C_{SAD}(\mathbf{p}, \mathbf{d}) = \sum_{\mathbf{q} \in N_p} |I_L(\mathbf{q}) - I_R(\mathbf{q} - \mathbf{d})| \tag{2.2}$$

Similarly, sum of squared differences (SSD) has the following form:

$$C_{SSD}(\mathbf{p}, \mathbf{d}) = \sum_{\mathbf{q} \in N_p} (I_L(\mathbf{q}) - I_R(\mathbf{q} - \mathbf{d}))^2 \tag{2.3}$$

Another popular metric combines truncated absolute differences of the color and the gradient at the matching points, and this model has been shown to be robust to illumination changes [10] and it is often paired with guided filter in practice:

$$\begin{aligned} C(\mathbf{p}, \mathbf{d}) = (1 - \alpha) \cdot \min(||I_L(\mathbf{q}) - I_R(\mathbf{q} - \mathbf{d})||, \tau_{col}) \\ + \alpha \cdot \min(||\nabla_x I_L(\mathbf{q}) - \nabla_x I_R(\mathbf{q} - \mathbf{d})||, \tau_{grad}) \end{aligned} \tag{2.4}$$

In this case, ∇_x is the gradient in x direction, α balances the color and gradient terms and τ_{col}, τ_{grad} are truncation values [10].

Normalized cross-correlation (NCC) is another common cost metric as shown below:

$$C_{NCC}(\mathbf{p}, \mathbf{d}) = \frac{\sum_{\mathbf{q}\in N_p} I_L(\mathbf{q}) I_R(\mathbf{q}-\mathbf{d})}{\sqrt[2]{\sum_{\mathbf{q}\in N_p} I_L(\mathbf{q})^2 \sum_{\mathbf{q}\in N_p} I_R(\mathbf{q}-\mathbf{d})^2}} \tag{2.5}$$

Compared to other cost functions, NCC seems to be the most expensive one in terms of computation. However, it works well regardless of the lighting situation of the images.

2.1.3 Cost (Support) Aggregation

Raw cost computation usually fails to enforce smoothness within an object. To address this problem, different filtering techniques have been proposed, and two of the most popular methods are presented below.

1. Guided Filter

   In [10], a filter-framework which efficiently achieves high-quality solutions is proposed based on guided filter imaging [11]. Based on last step, cost computation, cost volume can be computed as a 3-D dimensional array, with each element $C_{\mathbf{p},d}$ representing the cost for pixel $\mathbf{p} = (x, y)$ of choosing a disparity value d. The general filtering follows this formula:

   $$E_{\mathbf{p},d} = \sum_{\mathbf{q}} W_{\mathbf{p},\mathbf{q}}(I) C_{\mathbf{p},d} \tag{2.6}$$

   where $E_{\mathbf{p},d}$ represents the cost volume after the filtering. $W_{\mathbf{p},\mathbf{q}}(I)$ is the weighted filter that depends on the guidance image, and $C_{\mathbf{p},d}$ is the raw cost volume produced the cost computation step.

   As in the case of the guided filter, the weights are defined as follows [10]:

   $$W_{\mathbf{p},\mathbf{q}} = \frac{1}{|\omega|^2} \sum_{\mathbf{k}\in\omega_k} (1 + (I_{\mathbf{p}} - \mu_{\mathbf{k}})^T (\Sigma_{\mathbf{k}} + \epsilon U)^{-1} (I_{\mathbf{q}} - \mu_{\mathbf{k}})) \tag{2.7}$$

   Here, $\mu_{\mathbf{k}}$, $\Sigma_{\mathbf{k}}$ are the mean and variance matrix of image I in a squared $r \times r$ window ω_k centered at $\mathbf{k}$. U is an identity matrix, and ϵ controls the strength of the filtering. $|\omega|$ is the number of pixels within the window. $I_{\mathbf{p}}$ and $I_{\mathbf{q}}$ are the pixel values at $\mathbf{p}$ and $\mathbf{q}$ at image I.

This filter preserves edges because $(I_{\mathbf{p}} - \mu_{\mathbf{k}})^T(I_{\mathbf{q}} - \mu_{\mathbf{k}})$ will have a positive sign if and only if $\mathbf{p}$ and $\mathbf{q}$ are on the same side of $\mu_{\mathbf{k}}$. With $1 + (I_{\mathbf{p}} - \mu_{\mathbf{k}})^T(\Sigma_{\mathbf{k}} + \epsilon U)^{-1}(I_{\mathbf{q}} - \mu_{\mathbf{k}})$, the weight can be controlled so that its value will diminish around the edges. See Fig. 2.1 for a one-dimension illustration. I_i and I_j are the pixels in the one dimension example with mean μ and variance σ. As shown in the graph, the filter value would be smaller if I_i and I_j are on the same side of the edge as supposed to different sides.

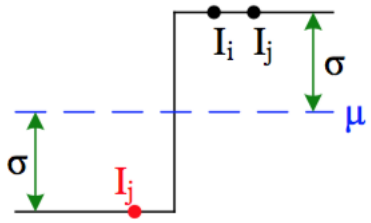


Figure 2.1: An one-dimensional example centered at pixel $\mathbf{k}$ with mean μ and variance σ. Figure adapted from [11].

2. Multi-Block Matching

   The Standard block-matching (BM) stereo matching algorithm just sums over a square-shaped window with the intention of correlating image patches [1]. The basic relation can be formulated as follows:

$$\hat{C}(\mathbf{p}, d) = \sum_{\mathbf{q} \in N_p} C(\mathbf{q}, d) \tag{2.8}$$

   where $\mathbf{p}, \mathbf{q}$ represent pixels and d is the disparity value. N_p is the neighbor of $\mathbf{p}$. $\hat{C}(\mathbf{p}, d)$ is filtered cost value after BM algorithm and $C(\mathbf{q}, d)$ is the raw cost value.

   One of the main problems with block-matching stereo is the inherent homogeneity assumption, which means that all pixels in the block have the same disparity value [1]. Of course, this assumption is often violated in real-world slanted surfaces. In that situation, the BM algorithm can actually dissimilar correlated image patches by summing over the squared-block.

   Multi-block matching (MBM) tackles this problem by utilizing

matching blocks of different shapes and sizes and combines them in a probabilistic fashion [1]. Suppose B is the set of all the blocks candidates, then MBM works like this:

$$\widetilde{C}(\mathbf{p}, d) = \prod_{b \in B} \frac{\hat{C}_b(\mathbf{p}, d)}{S_b(\mathbf{p})} \tag{2.9}$$

$$\hat{C}_b(\mathbf{p}, d) = \sum_{\mathbf{q} \in N_p^b} C(\mathbf{q}, d) \tag{2.10}$$

$$S_b(\mathbf{p}) = \sum_{d} \hat{C}_b(\mathbf{p}, d) \tag{2.11}$$

where $\hat{C}_b(\mathbf{p}, d)$ is the filtered cost of pixel $\mathbf{p}$ with disparity d using the block neighbor N_p^b. $S_b(\mathbf{p})$ is the summation of $\hat{C}_b(\mathbf{p}, d)$ over the entire label space.

Notice that each term $\frac{\hat{C}_b(\mathbf{p},d)}{S_b(\mathbf{p})}$ that corresponds to a pixel and one matching block in the set actually resembles a probability distribution over all the disparity labels. Suppose all matching blocks are independent, then we can multiply the "probability" of all matching blocks to get a probability of pixel $\mathbf{p}$ to have disparity label d.

Usually, the disparity is chosen by finding the label that minimizes the cost after cost aggregation. But in this case, the label is chosen by maximizing the probability, and in order to achieve that, different cost metrics will be modified to suit this purpose. For example, $\hat{C}_{SAD} = 255 - C_{SAD}$, where C_{SAD} and $\hat{C}_{SAD}$ represent the cost before and after the modification.

The MBM algorithm can boost the performance by keeping the advantages of the BM algorithm while at the same time still preserving the geometric properties of the image by various matching blocks.

2.1.4 Disparity refinement

One of the most widely-used post-processing technique is called occlusion detection and filling. To detect occlusion, a similar disparity map of the right image is acquired in the same fashion. A pixel is marked as occluded if the left and right pixel label does not match. The occluded pixels are then assigned to the lowest disparity value [10].

One of the advanced disparity refinement techniques is called slanted-plane smoothing. Slanted-plane smoothing constructs an image segmentation, a slanted plane for each segment, an outlier flag for each pixel, and a line label for each pair of neighboring segments [12]. In particular, if $\theta_i = (a_i, b_i, c_i)$ defines a disparity plane, then the disparity value for pixel $\mathbf{p} = (x, y)$ can be computed as:

$$d(\mathbf{p}, \theta_i) = a_i x + b_i y + c_i \tag{2.12}$$

The energy term can be formulated to be the sum of energies encoding appearance, location, disparity, smoothness, and boundary energies [12]. The direct minimization is NP-hard, and [12] proposed a block coordinate descent algorithm to achieve that. In practice, this algorithm can improve the accuracy significantly.

2.2 Belief Propagation and Its Variants

2.2.1 Belief Propagation

Global stereo matching algorithm is commonly formulated as a energy minimization framework [13]. Belief Propagation (BP) is, among others, a widely used algorithm for this problem. It is a Bayesian approach in a Markov random field (MRF).

Let D be the smooth disparity field, L be the line segment that indicates the depth discontinuity, and O indicate occlusion regions. Then $\{D, L, O\}$ can define a disparity map. Let $I = \{I_L, I_R\}$. Finding the disparity value is the same as maximizing the following probability based on the Bayesian rule:

$$P(D, L, O|I) = \frac{P(I, |D, L, O)P(D, L, O)}{P(I)} \tag{2.13}$$

In fact, it can be proven [2] that,

$$P(D|I) \propto \prod_p \exp(-\rho_d(d_p)) \prod_p \prod_{q \in N(p)} \exp(-\rho_s(d_p, d_q)) \tag{2.14}$$

$$\rho_d(d_p) = -\ln((1 - e_d)\exp(-\frac{|F(p, d_p, I)|}{\sigma_d}) + e_d) \tag{2.15}$$

$$\rho_s(d_p, d_q) = -\ln((1 - e_s)\exp(-\frac{|d_p - d_q|}{\sigma_s}) + e_s) \tag{2.16}$$

where $F(p, d_p, I)$ is the matching cost of pixel p with disparity d_p given I. By changing parameters e_d, e_s, σ_d and σ_s, various function can be used in BP. Moreover, if we take negative log of both sides of Eq. 2.12, the problem can be reformulated as minimizing energy function in the following form:

$$E = \sum_p E_p(d_p) + \sum_p \sum_{q \in N_p} E_{pq}(d_p, d_q) \tag{2.17}$$

where $E_p(d_p)$ can be viewed as the data term and $E_{pq}(d_p, d_q)$ can be viewed as the smoothness term.

On the other hand, the system can also be viewed as a Markov network. Fig. 2.2 shows the basic setup for the system. In this graph, X is the hidden nodes whereas Y is the observed nodes [2]. And $P(X|Y)$ can be factored as follows:

$$P(X|Y) \propto \prod_p \psi_s(x_p, y_p) \prod_p \prod_{q \in N(p)} \psi_{pq}(x_p, x_q) \tag{2.18}$$

and with the following definition

$$\psi_{pq}(x_p, x_q) = \exp(-\rho_s(d_p, d_q)) \tag{2.19}$$

$$\psi_s(x_p, y_p) = p(y_p|x_p) \propto \exp(-\rho_d(d_p)) \tag{2.20}$$

Now the global method of stereo vision becomes an MRF problem. But the exact inference can be computationally hard. BP minimizes Ep 2.15 by iteratively passing messages on a loopy graph [14].

Let $m_{qp}(d_p)$ be the message showing the q's opinion of p having label d_p and L be the label space, which can be calculated as follows:

$$m_{qp}^t(d_p) = \min_{d_p \in L}(E_{pq}(d_p, d_q) + E_q(d_q) + \sum_{s \in N_q \setminus p} m_{sq}^{t-1}(d_q)) \tag{2.21}$$

The t is the timestamp. And after T number of iterations, disbelief can be acquired as:

$$B_p(d_p) = E_p(d_p) + \sum_{q \in N_p} m_{qp}^T(d_p) \tag{2.22}$$

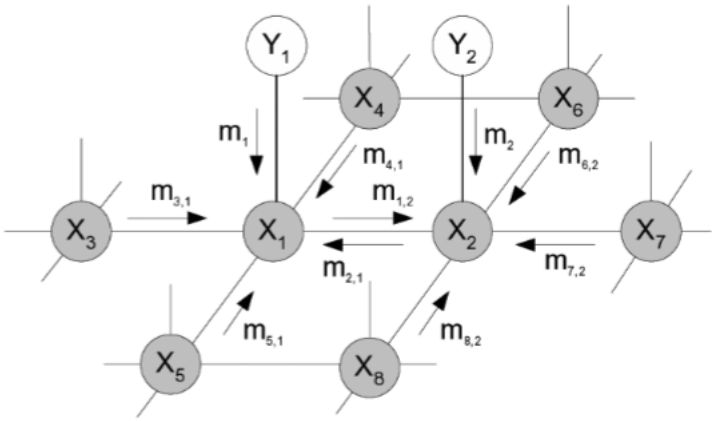


Figure 2.2: Message Passing in Markov Network.
Figure adapted from [2].

And the final label is selected by minimizing disbelief:

$$d_p = argmin_{d_p \in L} B_p(d_p) \tag{2.23}$$

2.2.2 Sped-up PatchMatch Belief Propagation

The BP algorithm as in the branch of global algorithm cannot easily achieve the speed of the local algorithm. With the recent success of local methods like MBM [1], there is a new direction of combining the advantages of local method with BP to achieve better accuracy and efficiency [13].

On the other hand, the computational difficulty is largely due to the huge label space L. To address that, [15] proposed to associate each pixel $\mathbf{p}$ with a sparse set of labels named particle R_p.

Combining these two ideas, [14] proposed a more efficient BP algorithm. The basic idea involves dividing the images into superpixels through segmentation. Because of the similarity of the pixels within a superpixel, the cost computation and aggregation can be applied to all the pixels in that by testing all the labels in the superpixel particle. Then the cost and the labels in the superpixel particle is passed to the pixels for message passing to derive final labels. The results of this algorithm has been tested to be faster while still maintaining a low error rate.

CHAPTER 3

METHODS

3.1 Ego Motion Estimation and Disparity Predication

The first approach that has been tested is based on the regular pinhole camera model [16]. The basic idea is that most pixels in consecutive frames are highly correlated as they are basically the same scene taken from different viewpoints and the correspondence can be modeled by a pinhole camera.

The algorithm starts with features matching as they form the sample data to estimate the parameters of the camera model. The images to consider consist of two pairs of images from consecutive frames, namely I_R^0, I_L^0 of the first frame and I_R^1, I_L^1 of the second frame with the subscript representing left and right images. The first feature to consider as proposed in the paper [16] is corners generated by Harris corner detector and non-maximum matching but other features like speeded up robust features (SURF) have also been tried.

After the feature points have been calculated, they can be transformed into 3D points. The transformation from 2D point to 3D point follows this formula.

$$\begin{cases} X = \frac{B(u-u_0)}{d} \\ Y = \frac{Bf_x(v-v_0)}{f_y d} \\ Z = \frac{Bf_x}{d} \end{cases} \tag{3.1}$$

where (u, v) is the inhomogeneous points of the image, (u_0, v_0) is the principle points of the image acquired from the calibration file (see Chapter 4 for more details), B is the baseline, f_x and f_y are the focal length in the x and y direction respectively and d is the disparity value.

After two sets of 3D points $\{\mathbf{X}^0\}$, $\{\mathbf{X}^1\}$ are attained, random sample

consensus (RANSAC) method is applied to minimize the following equation:

$$E = \frac{1}{N}\sum_{i=1}^{N} ||\{\mathrm{X}_i^1 - (R\mathrm{X}_i^0 + TR)\}||^2 \tag{3.2}$$

where R is the rotation matrix, and TR is the translation matrix of the camera model. RANSAC is chosen because of its ability to remove outliers and only return the parameters of inliers which in this case is the estimation of the transformation of static background. The basic assumption of RANSAC is that there are more inliers than outliers, so that it is more likely to choose inliers for estimation. However, in the practice of this experiment, this is not always the case, which can cause error. More details will be discussed later. Because R is the rotation matrix of the camera and TR is the translation matrix of the camera, only 3 non-collinear points and their matches are needed to estimate R and TR. During the experiments, the threshold of E is set to be 0.01 and the iteration number is 1000.

After rotation matrix R and translation matrix TR are obtained, the relation between the pixels in two different frames can be established. The projection relation is expressed as below:

$$\begin{cases} Z^0 x^0 = K[I \quad 0]X^0 \\ Z^1 x^1 = K[R \quad TR]X^1 \end{cases} \tag{3.3}$$

where Z is the value on z-direction on a 3D space, x is the inhomogeneous coordinates in the image, K is the intrinsic matrix of the camera, and X is the coordinates in the 3D space. The subscript represents the frame number. After some algebraic derivation, we can get Eq 3.4 as below,

$$\lambda \begin{pmatrix} x^1 \\ d^1 \end{pmatrix} = \begin{pmatrix} KRK^{-1} & \frac{KTR}{Bf_x} \\ 0_{1\times 3} & 1 \end{pmatrix} \begin{pmatrix} x^0 \\ d^0 \end{pmatrix} \tag{3.4}$$

where $\lambda = \frac{z^1}{z^0}$.

Eq 3.4 can be used to find consistent pixels and circle out the moving region. It can also be used to calculate the new disparity prediction of the static background and moving region will be updated using standard stereo vision method.

3.2 Upright Pinhole Camera Model

This approach utilizes a upright pinhole model [17], which can be used for one of the main application of stereo vision, autonomous vehicles. The assumption is that the cameras of autonomous vehicles are often up-right to the ground which in most circumstances is flat. Suppose the real height of the two frames are H_t and H_{t+1} respectively, the corresponding height in the images are y_t and y_{t+1}, the depth of the two images are Z_t and Z_{t+1}. Based on the geometric relation shown in Fig 3.1, the following can be derived:

$$\frac{y_t}{H_t} = \frac{f}{Z_t} \tag{3.5}$$

$$\frac{y_{t+1}}{H_{t+1}} = \frac{f}{Z_{t+1}} \tag{3.6}$$

The assumption implies that the real height in both frames should be the same, namely $H_t = H_{t+1}$. Then,

$$y_t Z_t = y_{t+1} Z_{t+1} \tag{3.7}$$

Furthermore, based on Eq 2.1, the relation between y_t and the disparity value d_t can be derived as:

$$\frac{y_t}{d_t} = \frac{y_{t+1}}{d_{t+1}} \Longrightarrow d_{t+1} = d_t \frac{y_{t+1}}{y_t} \tag{3.8}$$

Eq 3.8 provides a simple relation for disparity prediction. The only quantity needs to be evaluated at this point is $\frac{y_{t+1}}{y_t}$, which can be easily computed given optical flow information. In practice, optical flow is usually available or easy to acquire. For example, FPGA-based real-time optical-flow system [18] can provide the information pretty efficiently.

This estimation cannot serve as a direct depth map for the new frame because of occlusion caused by the moving objects and new objects coming into the new frame. But it serves as a good initialization to deal with stereo vision and can theoretically reduce the disparity calculation range by half. The experiments in this part is most concerned with using the estimation to make the error rate as low as possible. However, the accuracy of the depth estimation in the new frame is still largely bounded by the error propagated from the last frame, which will be discussed later.

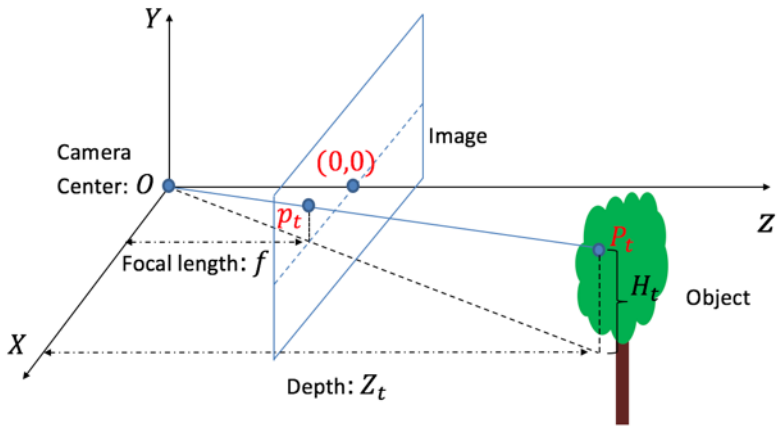


Figure 3.1: Upright Pinhole Camera Model. Figure courtesy of Chen Chen.

The basic pipeline involves three basic steps based on the depth map of the previous frame:

1. The first step calculates optical flow based on images of consecutive frames.

2. New prediction is formed by the aforementioned relation.

3. The prediction as initial value is used to calculate the disparity value of the new frame.

All the tests and experiments are devoted to Step 2 and 3 in an effort to reduce the propagated error in the prediction caused by the optical flow and the predication algorithm and also to find more efficient algorithm to utilize the initialization depth map. This method is mostly tested with EpicFlow [19]. Further details are discussed in Chapter 4.

3.3 Modified Sped-up PatchMatch Belief Propagation with Upright Pinhole Model

This section proposes a new algorithm that is based on SPM-BP [14] that incorporates temporal consistency and proves to improve convergence. Moreover, this algorithm utilizes FGI [8] to further speed up the process.

3.3.1 Disparity Prediction

Disparity prediction uses the upright pinhole camera model as introduced in Section 3.1. And the associated optical flow algorithm is EpicFlow [19] which can be proved to have lower error rate as shown in the next chapter.

3.3.2 Downsampling the image and Constructing a two-layer graph

One of the main problems with global methods like belief propagation is speed limitation. Because BP operates on the pixel level, its speed is proportional to the size of the image. To reduce the time, the proposed method begins by downsampling the image first and later uses FGI to upsample the depth map. Apparently, resizing images would result in information loss, which limits the downsample scale. Nevertheless, this approach is proved to work well with scaling factor $\lambda_{scale} = \frac{1}{2}$.

After downsampling the image, the algorithm follows the step in SPM-BP by partitioning the input image into b non-overlapping superpixels $S = (S_1, S_2, ...S_b)$ [14]. The algorithm utilizes simple linear iterative clustering (SLIC) [20] for the segmentation. The superpixels forms the first-layer of the graph and the original pixel forms the second layer. The superpixel-layer will provide label candidates stored in the superpixel particle and the standard message passing is still performed in the pixel-level. However, superpixel will now act as a basic unit for computing data cost which can be used for all the pixels inside the superpixel. Moreover, this formation allows filters like guided filter [10] to perform cost aggregation.

3.3.3 Particle Initialization by Voting and Perturbation

Unlike the standard SPM-BP algorithm which randomly sampled K labels from the label space at the beginning. The new proposed method uses the disparity prediction as a start to initialize the particle of each superpixel.

Consider the set of all disparity predictions of the pixels within a superpixel. There are three situations to consider. If the set happens to contain only K candidates, then those K candidates are used as the labels in the superpixel particle. If the set has more labels than K candidates, then the

particle is chosen by voting, which means the top K frequent labels shown within the superpixel are selected. Finally, if there are less than K labels in the set. Then besides these K labels, the rest labels are collected by adding a perturbation term to the most frequent labels, namely d'. And the perturbation term is a random variable uniformly sampled from the subset of a sample space $N = \{1, 2, ..., n\}$.

After the particle of superpixel have been generated, the particle of the pixels will be initialized to the same set as the superpixel they belongs to.

3.3.4 Particle propagation and random search

After the initialization, the rest of the algorithm will be ran in loops for T times so that the result can converge. And the first step of each loop is particle propagation and random search. Unlike the SPM-BP algorithm, these two separate steps are combined here as one particle generation step to save time.

As in [14], for each superpixel S_i, the new particle is generated first by sampling the neighboring superpixels. From each neighboring superpixels, a pixel p_i within that will be randomly selected which has particle R_{p_i}. And the new proposal will be the union set $RS_1 = \cup R_{p_i}$. RS_1 is the proposal generated by particle propagation.

Besides the proposal derived from neighbooring superpixels, a random search is performed to prevent local minima. To achieve this, a pixel p_i form S_i will be randomly chosen. And the new proposal will be $RS_2 = \{l + \frac{R}{2^i} | i = 1, ..., M\}$ $\forall i \in R_{p_i}$. R is a random variable sampled form the entire label space. RS_2 is the proposal generated by random search.

Finally, the new particle proposal will be $RS(i) = RS_1 \cup RS_2$, where $RS(i)$ is the new particle for superpixel S_i.

3.3.5 Data cost computation

Before computing the data cost for the proposal of each superpixel, a bounding box needs to be acquired first whose borders should be extended further by the width of filter used in cost aggregation step to avoid filtering artifacts around the boundaries [14]. Then the data cost is computed for all the pixels

within the bounding box with the labels in the proposal. After the raw cost has been computed, cost aggregation will be performed through guided filter.

3.3.6 Message Update and Final label selection

The data cost and label proposal of superpixels will be passed on to all the pixels within for message updates and label selections.

1. The incoming messages will be calculated for all pixels p within superpixel S_i, all $d_p \in RS(i) \cup R_p$ and $d_q \in RS(i)$ using the following formula, R_p it the particle of pixel **p**:

$$m_{qp}^{t}(d_p) = \min_{d_p \in L}(E_{pq}(d_p, d_q) + E_q(d_q) + \sum_{s \in N_q \setminus p} m_{sq}^{t-1}(d_q)) \tag{3.9}$$

2. The disbelief will be computed:

$$B_p(d_p) = E_p(d_p) + \sum_{q \in N_p} m_{qp}^{T}(d_p) \tag{3.10}$$

3. The new particles for pixel p will be chosen by selecting top K disbelief:

$$d_p = argminK_{d_p \in RS(i) \cup R_p} B_p(d_p) \tag{3.11}$$

After T iterations, the the final label is selected by minimizing disbelief:

$$d_p = argmin_{d_p \in R_p} B_p(d_p) \tag{3.12}$$

Notice that the neighboring pixels given one reference pixel is its nearest four neighbors from left, right, up and down directions.

3.3.7 Depth upsampling and Post-processing

After the result converges, the depth map can be upsampled to the regular size using FGI [8] followed by post-processing algorithms like slanted plane smoothing.

CHAPTER 4

EXPERIMENTAL RESULTS

4.1 KITTI Dataset

Any experimental results shown in this thesis were tested on the KITTI [21] 2015 training dataset. The specification of the camera can be referred in the calibration file. In particular, the projection matrix which is named P_{rect} in the calibration text file is formulated in the following way.

$$P_{rect} = \begin{bmatrix} f_x & 0 & u_0 & -f_x B \\ 0 & f_y & v_0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

where u_0 and v_0 are the coordinates of the principle points, f_x and f_y are the focal length in the x and y direction respectively, and B is the baseline. All the parameters can therefore be acquired in this way. In particular, the baseline B can be computed by $-\frac{P_{rect}(1,4)}{P_{rect}(1,1)}$. For example, the parameters associated with the first pair images is the following: $f_x = f_y = 721.5377px$, $u_0 = 609.5593px$, $v_0 = 172.8540px$ and $B = 0.5372m$.

The KITTI dataset is the most widely-used dataset of stereo vision for autonomous vehicles. However, each pair of images only have two frames to work with. As a result, the experiments in this thesis are not tested for multiple frame disparity prediction.

4.2 Ego Motion Estimation and Disparity Predication

The first algorithm is tested on the KITTI dataset. Using the mechanism described in Section 3.2, 46 pairs of matching can be acquired. However, nearly half of them are extremely far from the camera. But those feature

pairs are relatively close to each other, which can cause more error when converting to the 3D coordinates. So those error-prone matches are removed and it ends up having only 27 pairs. Those points are then converted to 3D coordinates based on Eq. 3.1. For consistency, the disparity values used in the equation are all calculated by the difference of matched points instead of using the value from depth map which is only available for the first frame. Those 3D points are then passed to the RANSAC algorithm by randomly selecting 3 matches to estimate R and TR matrices. The threshold is set to be 0.01 and 1000 iterations are executed. The generated R and TR matches well with 18 pairs out of 27 which is around 67%. The R and TR are applied to find the moving regions which are updated by the standard stereo-matching algorithm and the rest is updated using Eq. 3.4. The result can be seen in Fig 4.1.

One of the main problem occurs when the number of pairs is really

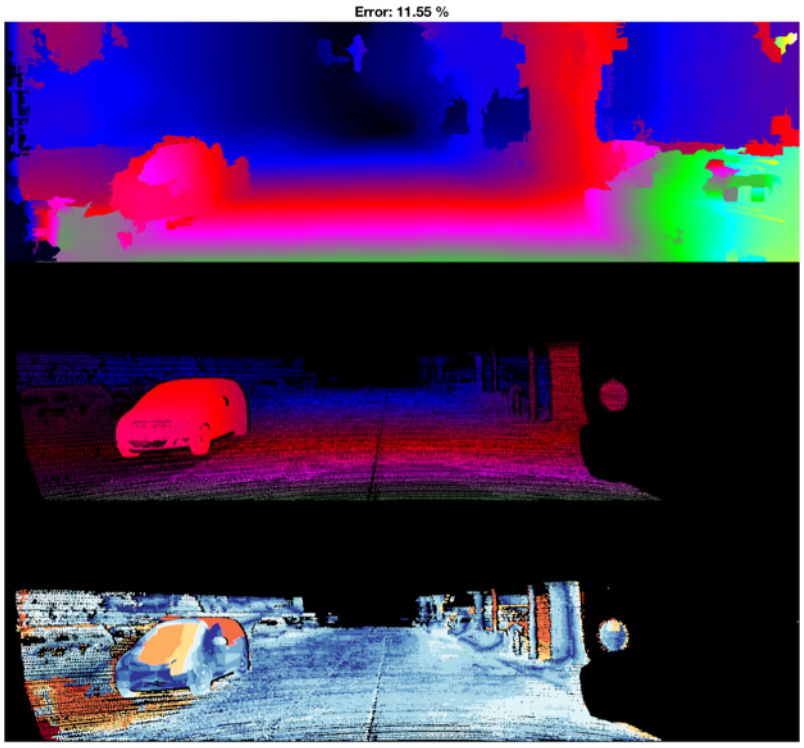


Figure 4.1: The top image is the depth map produced by the ego estimation and disparity algorithm with the regular pinhole camera model and corner as feature, the middle one is the ground truth, and the last one is the difference.

small and the RANSAC estimation suffers from more uncertainty, because

RANSAC relies on the assumption that inliers outnumbered outliers in a significant way so that inliers are more likely to be selected as samples by the algorithm.

To address this problem, other features have also been tested. The first one is SURF and it tends to generate more pairs. Nevertheless, the final accuracy is actually worse. The problem is that many matches failed the epipolar constraint tests and are removed during the filtering and the quality of the remaining matches are not as good as those generated by corner detection.

With the intention to gather more matching points and following the

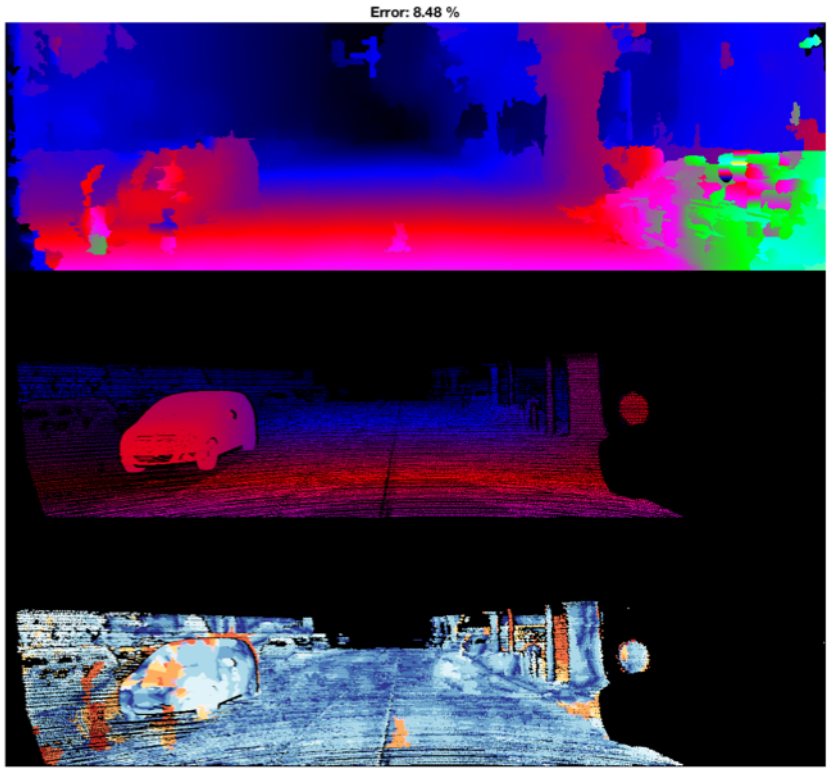


Figure 4.2: The top image is the depth map the ego estimation and disparity algorithm with the regular pinhole camera model using optical flow to generate matched points, the middle one is the ground truth, and the last one is the difference.

direction of the last experiments, optical flow is then tested so that every pixel in the image can have a match. Optical flow is again obtained from EpicFlow, and different pairs of pixels from consecutive frames are calculated based on that. There may be pairs that share the same pixels, but it is highly unlikely they will all contribute to the estimation of R and TR as they cannot

all be inliers. Because the image is of size 375×1242, there are 465750 possible match candidates. After using RANSAC scheme to estimate R and TR, there are 116122 pairs that matches the R and TR well with threshold at 0.01 and iteration number at 1000. The result is shown in Figure 4.2, and it seems slightly better with less error rate. The R and TR matrix can be better estimated using this framework compared to the previous one using corner as features, but the loss on error comes with an increase in computational time. The other problem is that, RANSAC scheme does not converge well with too many pairs even after 1000 iterations. Although there are 116122 pairs under the threshold, it only counts 24.93% of the candidates, so it does not fit with the assumption well, leaving an uncertain factor in the algorithm. Fortunately, different R and TR estimation value using RANSAC all eventually give similar accuracy.

Overall, these methods tends to have more error than the ones proposed below because it only focuses on estimating the camera model to readjust the disparity of static background, whereas the following algorithms can adapt to different influence of the disparity change.

4.3 Upright Pinhole Camera Model

All the experiments in this part were tested on KITTI dataset [5] using the 200 training image pairs. Based on the stereo matching steps described in Chapter 2, the standard baseline algorithm suite includes NCC for cost computation, MBM for cost aggregation, and slanted plane smoothing for disparity refinement.

The first method uses image warping to form a single combined image based on the left and right image pair and the prediction of disparity value using the upright pinhole camera model. This image is then used as the new left image to feed into the standard stereo vision algorithm. One potential problem with this approach is the information loss. As shown in Figure 4.3, the glitch in the image can be a potential cause for error. However, the advantage of this method is that it can directly shorten the guess range and therefore time without a lot of changes to the standard algorithm. In fact, the guess range can be reduced from 128 to 64. This method is tested using SIFT flow [22] and EpicFlow [19] respectively. The SIFT flow has around

10% error rate while the EpicFlow can reduce the error rate to 8.89%.

(a) Original Image

(b) Warped Image

Figure 4.3: Original Image and Warped Image

The second method just simply set all the costs in the cost volume associated with disparity that is out of bound for each pixel to zero. Tested with EpicFlow, the error rate can further drop to 7.72%. Notice that there are parts of the cost volume associated with certain disparity value that is completely zero for all pixels. So there is no need to calculate them and the disparity guess range can be obtained by selecting the largest and the smallest disparity guess value. Theoretically, this may not improve any speed. However, empirically, the guess range can in effect be reduced from 128 to around 92, which is still a 28% decrease.

The error propagated from the previous disparity calculation, optical flow calculation and the prediction calculation can be significant. Fast guided global interpolation (FGI) [8] has been tested to reduce this type of error. The first test tried to interpolate optical flow by selecting matched points

Table 4.1: Summary of two methods associated with the upright pinhole model

Method	Error Rate	Search Range Reduction
Using warped image	8.89%	50%
Setting out-of-range cost to zero	7.72%	28%

whose SAD difference are below a threshold which is 4 in this test, and then passing though the FGI motion interpolation algorithm. However, this only proves to be working for some images. For example, the error rate for the second pair of image after flow interpolation is 7.47% compared to 9.19% without flow interpolation. The reason is probably that EpicFlow itself can produce rather reliable optical flow value without interpolation that may in fact hurt the results obtained.

More efforts were devoted to directly reduce the uncertainty in the disparity prediction. One approach is to downsample the depth map first and then use FGI to upsample the map. The intuition is that FGI uses image of the new frame as global guide, which may help reduce the error in prediction. So the main problem is focused on how to downsample the depth map. Direct downsampling can cause significant information loss, so a filter must be applied first to blur the map so that every point in the map can obtain information about its surrounding pixels. Two different filter schemes have been experimented on the first 20 image pairs. As shown in Table 4.2, the Gaussian filter cannot provide performance boost, but the box filter seems to work well with a 0.67% error rate drop. However, the accuracy improvement is still limited and only work on a case by case basis without a universal improvement on the test data. As shown in Fig 4.4, the result with interpolation tends to resemble the depth map of baseline stereo vision techniques evidenced by the similarity in the error images (the bottom one) and can achieve better accuracy than the ones without interpolation.

Table 4.2: Average error rate for the first 20 image pairs using different interpolation scheme

	Baseline	Gaussian Filter with Radius 8	Box Filter with Width 32
Error Rate	9.32%	9.38%	8.65%

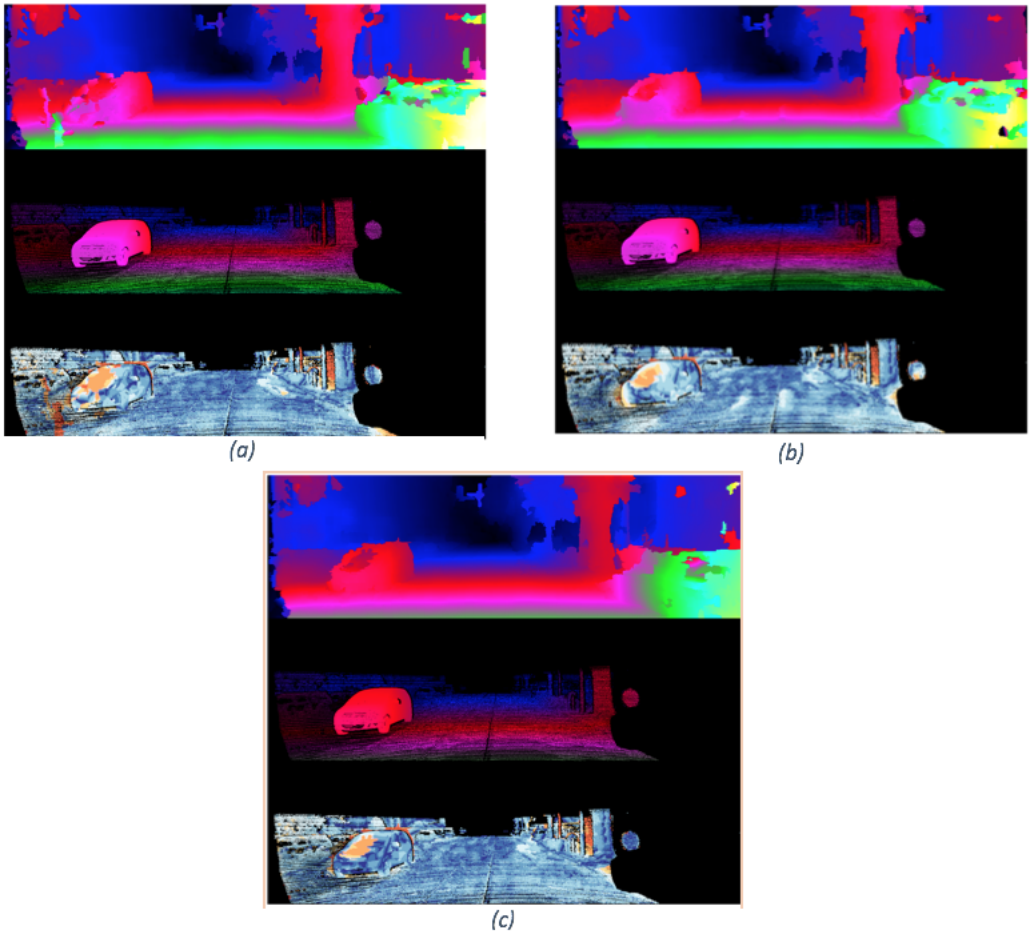


Figure 4.4: (a) The result of stereo vision with the upright pinhole model for disparity prediction and no interpolation (b) The result of stereo vision with the upright pinhole model for disparity prediction including interpolation with 32 size box filter (c) The result for the baseline algorithm with no disparity prediction

4.4 Modified Sped-up PatchMatch Belief Propagation with Upright Pinhole Model

4.4.1 Parameter and Cost Function Selection

Following the previous examples, NCC will be used as cost function in this case, which preforms better for KITTI dataset than the sum the truncated absolute difference of the color and the gradient at the matching points as proposed in [14].

The smoothness term is defined with two pixels p and q by the following formula:

$$E_{pq}(l_p, l_q) = \lambda * \exp(-(||I_p - I_q||)/\sigma) * min(|l_p - l_q|, \tau) \tag{4.1}$$

In this experiment, λ is set to be 0.01, σ to be 10 and τ to be 2. Guided filter will be used for cost aggregation, and window size is chosen to be 9 evident by the following graph Fig 4.5 of the relations between raw error rate (before post processing) and the window size. The number of loops is set to be 3 even though 2 seems to be sufficient. And the superpixel number is set to be 500.

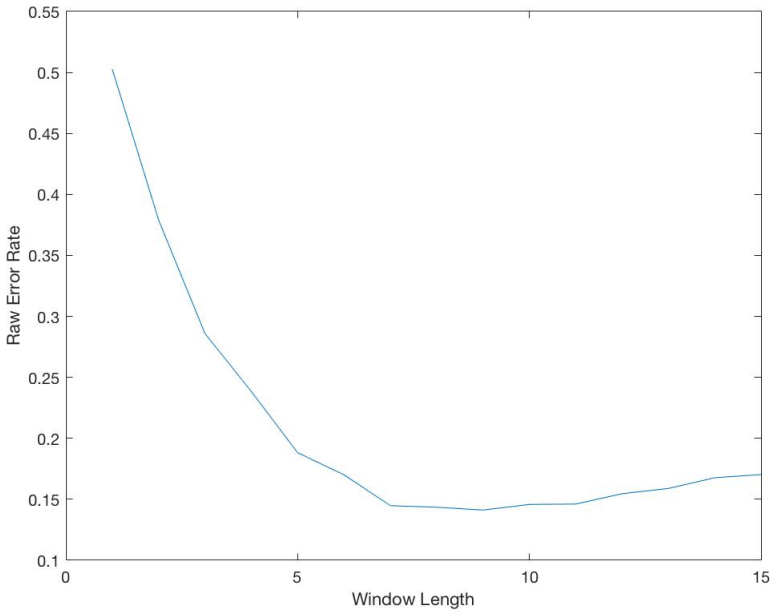


Figure 4.5: Raw Error Rate vs. Filter Window Size

4.4.2 Results

The original algorithm takes around 3 - 5 loops to converge, while the proposed algorithm only takes 2 - 3 loops to converge and sometimes only one iteration seems to be sufficient especially with the help of FGI upsampling and other post-processing techniques. The convergence of the algorithm is shown in Table 4.3, tested with the first 20 images of the KITTI training. The comparison is shown in the following graph Fig 4.6:

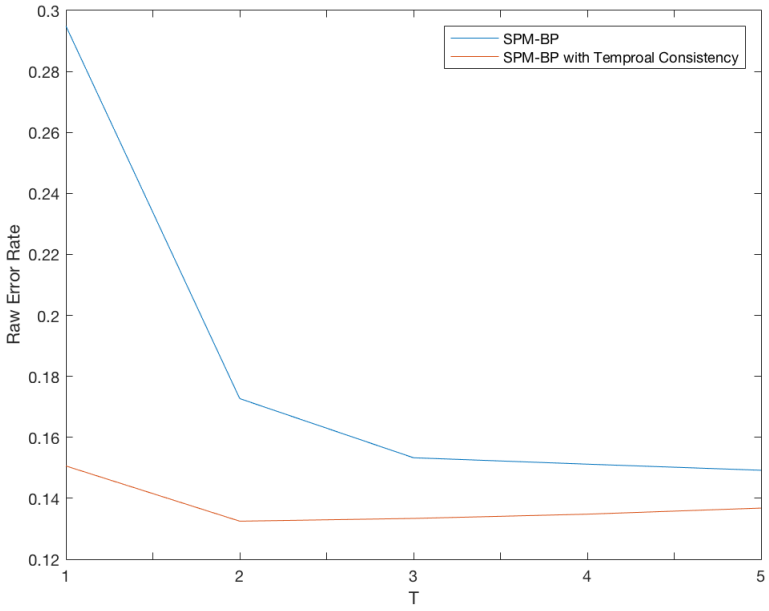


Figure 4.6: Raw Error Rate vs. Iterations T

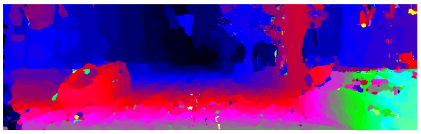
The graph shows the comparison of raw error rate (before post-processing) between SPM-BP without temporal consistency with the proposed algorithm. And it is evident that the new algorithm can converge faster and seems to have a slightly lower error rate. As shown in Fig 4.8, which is the disbelief

Table 4.3: Error rate in the process of the proposed algorithm, tested of the first 20 images in KITTI

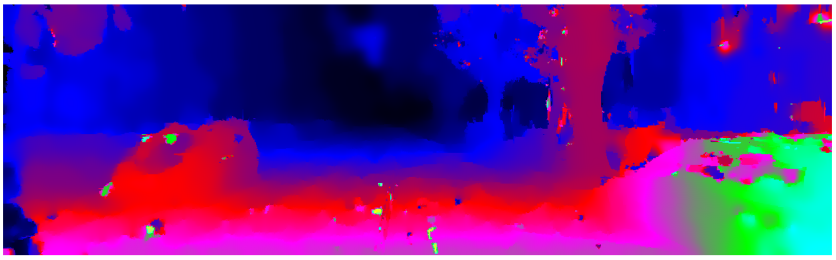
Methods	Error Rate
First Round Average Raw Error Rate	16.62%
Second Round Average Raw Error Rate	15.05%
Third Round Average Raw Error Rate	14.91%
Final Error Rate	9.14%

map of the image, the ones with temporal consistency tends to converge to a deeper disbelief sooner than the one without as indicated by the blue color which represents low value of disbelief.

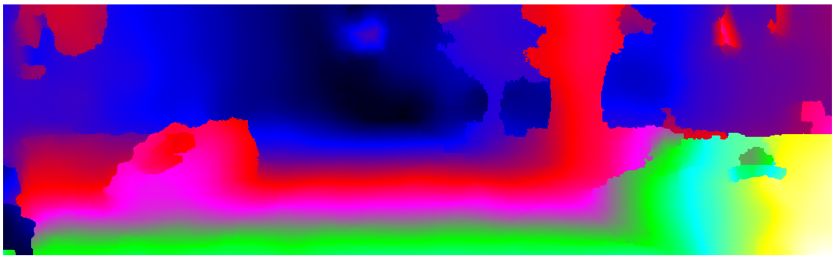
In particular, the third round average raw error rate is at 14.91%, and if tested with the original method, the error rate is 15.09%. So there is some accuracy increase. But the error difference for the two methods after post-processing is not that significant. Tested with the 100 training data in the KITTI dataset, the proposed algorithm gives an average error rate of 8.46%, whereas the SPM-BP algorithm with no temporal consistency gives 8.49%. In this case, the improvement is not significant. Nevertheless, the convergence speed is still improved. Finally, Fig 4.7 shows different stages of the proposed algorithm.



(a) Before Upsamping

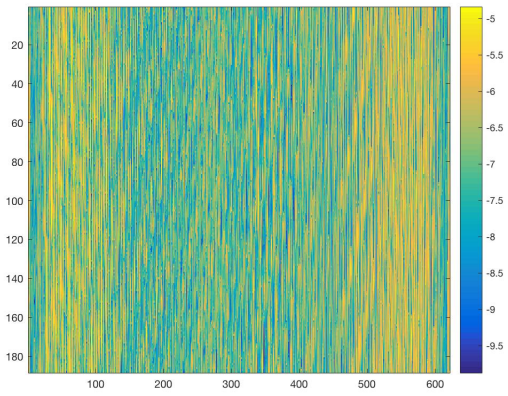


(b) After Upsamping

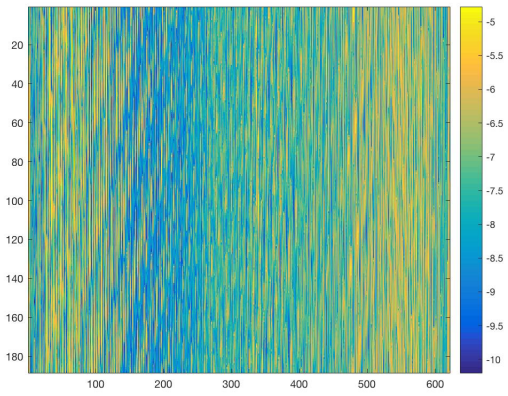


(c) After Post-processing

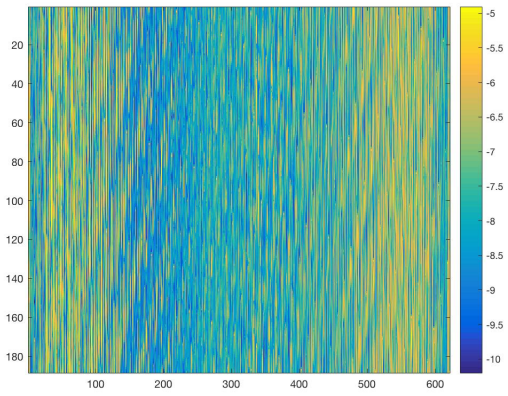
Figure 4.7: Different Stages of the Proposed Algorithm: Modified Sped-up PatchMatch Belief Propagation with Upright Pinhole Model



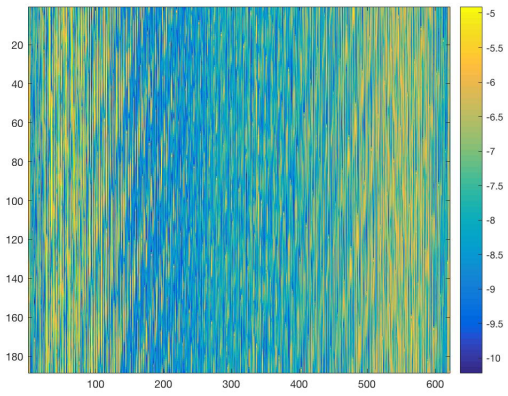
(a) First Round



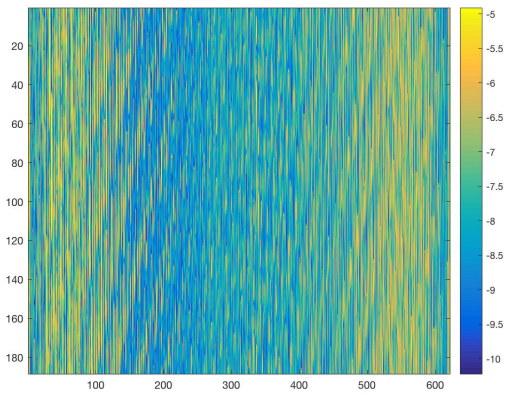
(b) First Round with Temporal Consistency



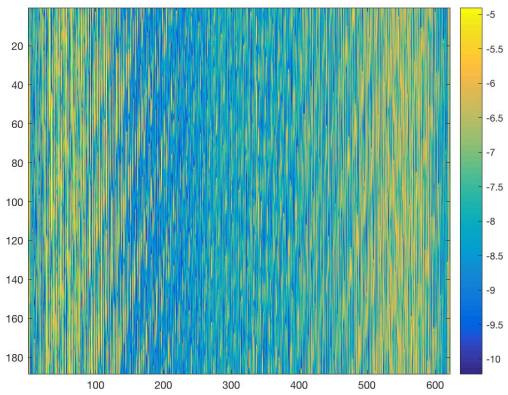
(c) Second Round



(d) Second Round with Temporal Consistency



(e) Third Round



(f) Third Round with Temporal Consistency

Figure 4.8: The DisBelief Graph of the Modified Sped-up PatchMatch Belief Propagation Algorithm with Upright Pinhole Model Tested with and without Temporal Consistency

CHAPTER 5

CONCLUSION

In this thesis, three different models of stereo matching with temporal consistency have been studied. The first one uses ego estimation for disparity prediction and focus mostly on the static background. Nevertheless, it proves to be less efficient and error-prone. The second one adopted a upright pinhole camera model by assuming flat surfaces. The assumption seems bold but is valid in most real-life circumstances. Moreover, the estimation provides a good initialization for stereo matching, and can reduce the runtime by using a limited search space after initialization with little or no loss in accuracy.

With the success of the upright pinhole model, the thesis proposes a new algorithm based on SPM-BP. The main contributions of the newly proposed algorithm include the addition of temporal consistency and further speed improvement by using FGI for depth map upsampling. And the results proves to be better in convergence speed with a slightly increase in overall accuracy.

CHAPTER 6

FUTURE WORK

One of the first thing that can be improved with the current model is still speed. Even though the proposed algorithm boost the performance in terms of time, it is still slower than the other local methods if running in Matlab on CPU. It would be useful to transport the serial code to parallel code and test the algorithm on GPU.

Besides the possible improvements on the implementation side, the algorithm can also benefit from more research. Currently, the disparity prediction model is simple and powerful under the assumption of slat surface. With a more robust prediction mechanism, the error rate and convergence speed may work better.

REFERENCES

[1] N. Einecke and J. Eggert, "A multi-block-matching approach for stereo," in *Intelligent Vehicles Symposium (IV), 2015 IEEE*. IEEE, 2015, pp. 585–592.

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

[3] J. Sun, N.-N. Zheng, and H.-Y. Shum, "Stereo matching using belief propagation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 7, pp. 787–800, 2003.

[4] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient belief propagation for early vision," *International Journal of Computer Vision*, vol. 70, no. 1, pp. 41–54, 2006.

[5] T. Mei, L. An, and B. Bhanu, "Context guided belief propagation for remote sensing image classification," *Applied optics*, vol. 54, no. 11, pp. 3372–3382, 2015.

[6] P. Fischer, A. Dosovitskiy, E. Ilg, P. Häusser, C. Hazırbaş, V. Golkov, P. van der Smagt, D. Cremers, and T. Brox, "Flownet: Learning optical flow with convolutional networks," *arXiv preprint arXiv:1504.06852*, 2015.

[7] W. Luo, A. G. Schwing, and R. Urtasun, "Efficient deep learning for stereo matching," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 5695–5703.

[8] Y. Li, D. Min, M. N. Do, and J. Lu, "Fast guided global interpolation for depth and motion," in *European Conference on Computer Vision*. Springer, 2016, pp. 717–733.

[9] H. Hirschmuller and D. Scharstein, "Evaluation of stereo matching costs on images with radiometric differences," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 9, pp. 1582–1599, 2009.

[10] A. Hosni, C. Rhemann, M. Bleyer, C. Rother, and M. Gelautz, "Fast cost-volume filtering for visual correspondence and beyond," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 2, pp. 504–511, 2013.

[11] K. He, J. Sun, and X. Tang, "Guided image filtering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 6, pp. 1397–1409, 2013.

[12] K. Yamaguchi, D. McAllester, and R. Urtasun, "Efficient joint segmentation, occlusion labeling, stereo and flow estimation," in *European Conference on Computer Vision*. Springer, 2014, pp. 756–771.

[13] X. Wang and Y. Liu, "Accurate and fast convergent initial-value belief propagation for stereo matching," *PloS one*, vol. 10, no. 9, p. e0137530, 2015.

[14] Y. Li, D. Min, M. S. Brown, M. N. Do, and J. Lu, "SPM-BP: Sped-up patchmatch belief propagation for continuous mrfs," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 4006–4014.

[15] R. Kothapa, J. Pacheco, E. Sudderth et al., "Max-product particle belief propagation," Brown University Dept. of Computer Science, Masters project report, 2011.

[16] J. Jiang, J. Cheng, B. Chen, and X. Wu, "Disparity prediction between adjacent frames for dynamic scenes," *Neurocomputing*, vol. 142, pp. 335–342, 2014.

[17] C. Chen, J. Lu, D.-K. Kwon, D. Moore, and M. N. Do, "Accelerated stereo matching for autonomous vehicles using an upright pinhole camera model," *Electronic Imaging*, vol. 2017, no. 19, pp. 18–21, 2017.

[18] J. Díaz, E. Ros, F. Pelayo, E. M. Ortigosa, and S. Mota, "FPGA-based real-time optical-flow system," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, no. 2, pp. 274–279, 2006.

[19] J. Revaud, P. Weinzaepfel, Z. Harchaoui, and C. Schmid, "EpicFlow: Edge-Preserving Interpolation of Correspondences for Optical Flow," in *Computer Vision and Pattern Recognition*, 2015.

[20] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "SLIC superpixels compared to state-of-the-art superpixel methods," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 11, pp. 2274–2282, 2012.

[21] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[22] C. Liu, J. Yuen, and A. Torralba, "Sift flow: Dense correspondence across scenes and its applications," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 5, pp. 978–994, 2011.