2020

Comparative Study of Model-Based and Learning-Based Disparity Map Fusion Methods

Douglas E. Kerr Jr.
West Virginia University, dkerr2@mix.wvu.edu

2020

Comparative Study of Model-Based and Learning-Based Disparity Map Fusion Methods

Douglas E. Kerr Jr.

Comparative Study of Model-Based and Learning-Based Disparity Map Fusion Methods

Douglas Kerr, Jr.

Problem Report submitted
to the Benjamin M. Statler College of Engineering and Mineral Resources
at West Virginia University

in partial fulfillment of the requirements for the degree of

Master of Science
in
Electrical Engineering

Xin Li, PhD, Chair
David Graham, PhD
Natalia Schmid, PhD

Lane Department of Computer Science and Electrical Engineering

Morgantown, West Virginia
2020

Keywords: Stereo Matching, Disparity Maps, Fusion Methods, Machine Vision, Depth from Stereo Images

Abstract

Comparative Study of Model-Based and Learning-Based Disparity Map Fusion Methods

Douglas Kerr, Jr.

Creating an accurate depth map has several, valuable applications including augmented/virtual reality, autonomous navigation, indoor/outdoor mapping, object segmentation, and aerial topography. Current hardware solutions for precise 3D scanning are relatively expensive. To combat hardware costs, software alternatives based on stereoscopic images have previously been proposed. However, software solutions are less accurate than hardware solutions, such as laser scanning, and are subject to a variety of irregularities. Notably, disparity maps generated from stereo images typically fall short in cases of occlusion, near object boundaries, and on repetitive texture regions or texture-less regions.

Several post-processing methods are examined in an effort to combine strong algorithm results and alleviate erroneous disparity regions. These methods include basic statistical combinations, histogram-based voting, edge detection guidance, support vector machines (SVMs), and bagged trees. Individual errors and average errors are compared between the newly introduced fusion methods and the existing disparity algorithms. Several acceptable solutions are identified to bridge the gap between 3D scanning and stereo imaging. It is shown that fusing disparity maps can result in lower error rates than individual algorithms across the dataset while maintaining a high level of robustness.

To my family & friends…

Acknowledgements

I would like to acknowledge and thank my research advisor, Dr. Xin Li, for his incredible guidance during this work. Dr. Li is responsible for my interest in image and video processing. I am very thankful for his role in my education. It was truly inspiring to see how quickly he could solve problems, expand ideas, and offer research papers for further reading.

I would also like to extend my gratitude to my committee members. Dr. David Graham and Dr. Natalia Schmid both demonstrate an incredible passion for teaching. It has been refreshing, to say the least, to attend their lectures and learn from them.

Further, I would like to thank Dr. Jeremy Dawson for his guidance in the beginning of my graduate program. He opened many doors that allowed me to meet many helpful, intelligent people and make several, very good friends.

Lastly, I would like to thank some of my fellow students, Matthew Pachol, Stallone Sabatier, Kenney Hite, Casey Norville, Kyle Smith, Morgan Trester, and Catherine O'Hearn, who kept me motivated and encouraged me when I needed it most. My experience would not have been the same without you.

Table of Contents

List of Figures

List of Tables

1 Introduction

Many algorithms exist to interpret disparity from stereoscopic images and work well to accomplish this task. This is evidenced by the numerous, evaluated algorithms presented on the Middlebury website [1] [2] [3]. Middlebury provides datasets and ranks submitted algorithms based on several criteria. This work compares rankings after performing a variety of post-processing techniques.

The term “image fusion” is usually used to refer to the combining of images obtained from multiple, and possibly dissimilar, sensors to create a clearer or more reliable representation than any of the individual sensors alone. The following discussion is notably different in regard to the origin of the data to be fused. The length of the discussion is based on fusing results or outputs obtained from the application of different algorithms on sensor data as opposed to raw sensor data itself. Specifically, we are looking at unique, post-processed, disparity maps calculated from identically sourced, stereo images. New results are created either by patching select portions of inputs together or by interpolating entirely new data values from selected inputs.

1.1 Application Analysis

1.1.1 3D Scanning Hardware Replacement

Applications for low cost, high quality disparity maps are plentiful. Disparity maps can be obtained through 3D scanning hardware such as LIDAR. LIDAR offers the highest quality map but is currently the least cost-effective solution. For example, the Velodyne hdl-64e LIDAR contains 64 lasers, providing the highest resolution at a cost of up to $85,000 [4]. The Velodyne vlp-16 is the least expensive model on the product line and contains 16 lasers, providing lower resolution but at a more affordable price of $8,000 [4]. Large-scale distribution in consumer products is impractical at these costs.

Radar is another possible means of producing disparity maps. Radar is an order of magnitude less in price compared to LIDAR, ranging in the hundreds of dollars. However, radar does not provide the near-perfect maps accustomed to LIDAR. It is susceptible to echoing, which can create ghosting of objects, along with other distortions due to interference and noise.

Lastly, stereoscopic images can provide disparity maps at the lowest cost, but the algorithms used to calculate the maps can be inconsistent, as evidenced in this paper. The quality suffers in certain cases, as no single algorithm is both robust and accurate. Stereo matching is also subject to certain inherent faults such as occluded regions, reflection, differences between strong textured and texture-less regions, and object boundaries. The added benefit of this hardware is portability as demonstrated by the recent availability of two-camera configurations in high-end smartphones.

While hardware costs are likely to continue a downward trend, there is currently no promise of further size reduction or price reduction below a certain point. A cost analysis of LIDAR, radar, and stereo imaging shows at least an order of magnitude between each technology. The relative cost-to-performance ratio between each will likely remain the same unless a fundamental breakthrough occurs in production means, material makeup, or functionality for any of these technologies.

1.1.2 Augmented and Virtual Reality

Applications for high quality, portable 3D mapping include improvements for augmented and virtual reality. For augmented reality to be truly immerse, object distances and boundaries must be clearly distinguished. It is often necessary for a virtual object to appear partially occluded as it would in the real world if another object passed in front of it. Early applications have not made use of this, or rather, have not been able to. Rendered objects are limited to clear, flat surfaces and only appear as an overlay on the image. These initial limitations were evident in popular mobile games, such as Pokémon Go, and social media platforms, such as Snapchat. Augmented reality platforms simply lack the ability to fully make use of 3D environments since they have few cues to perceive depth. Currently, the size of the rendered object can be scaled based on the intrinsic lens properties and motion-based sensors. There is also often an observable flickering effect when a surface is disrupted due to foreground objects.

Virtual reality applications include being able to map portions of an environment at low cost with respect to hardware. This potentially allows for a more fluid virtual tour akin to Google Street View. 3D mapping could have a similar role in real-estate, allowing for an improved virtual

house tour. This would allow potential buyers to more accurately gauge the size of a home, quickly compare with other homes, and tour from a distance without any buying pressure or wasted time.

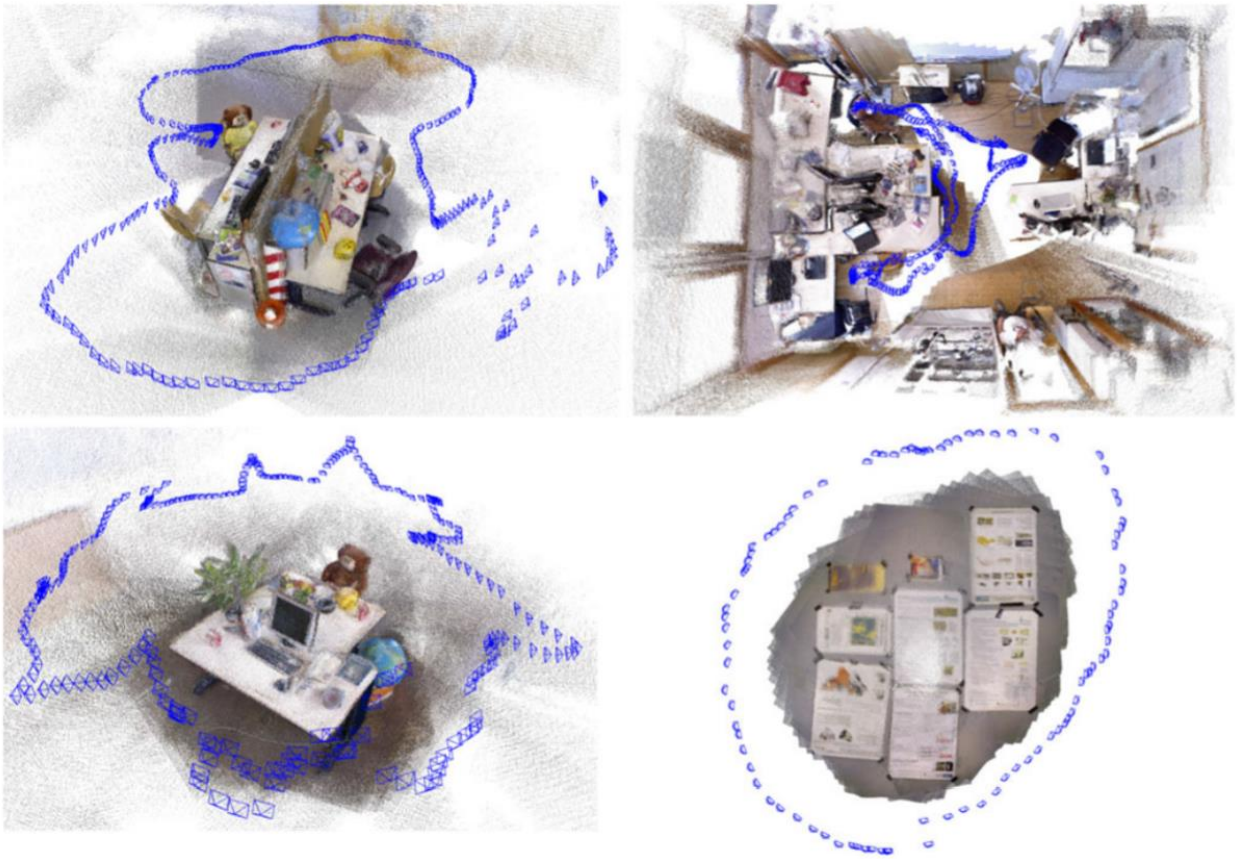

Figure 1.1 Instances of Pointcloud Reconstructions from Stereo Cameras. Adopted from [5]

3D scanning could allow an object to be scanned into virtual reality with a simple walk around. Simultaneous localization and mapping (SLAM) algorithms can become more accurate and useful with calibrated stereo cameras. ORB-SLAM2 and Stereo LSD-SLAM both detail ways of implementing stereo vision [5] [6]. Figure 1.1 provides examples of both full room virtualization as well as walk arounds of objects to create virtual objects.

1.1.3 Robotics

Stereo vision has proven useful in robotics for navigation purposes. NASA Mars exploration rovers *Spirit* and *Opportunity* used this technology for missions in 2004. Stereo vision was

beneficial because it relied on a passive sensor and no moving parts, limiting the risk of component failure. The cameras only needed to detect passive sunlight as opposed to LIDAR, which required light to be emitted before capturing its response. A software solution was deemed more power efficient for navigating complex environments. [7]

The use of drones has been proposed as a solution to mapping environments that are otherwise inaccessible [8]. This can include unstable mining sites and radioactive sites. The same application of stereo SLAM algorithms as discussed in Section 1.1.2 is used. The only difference being the stereo camera collection is done remotely as opposed to handheld data aggregation.

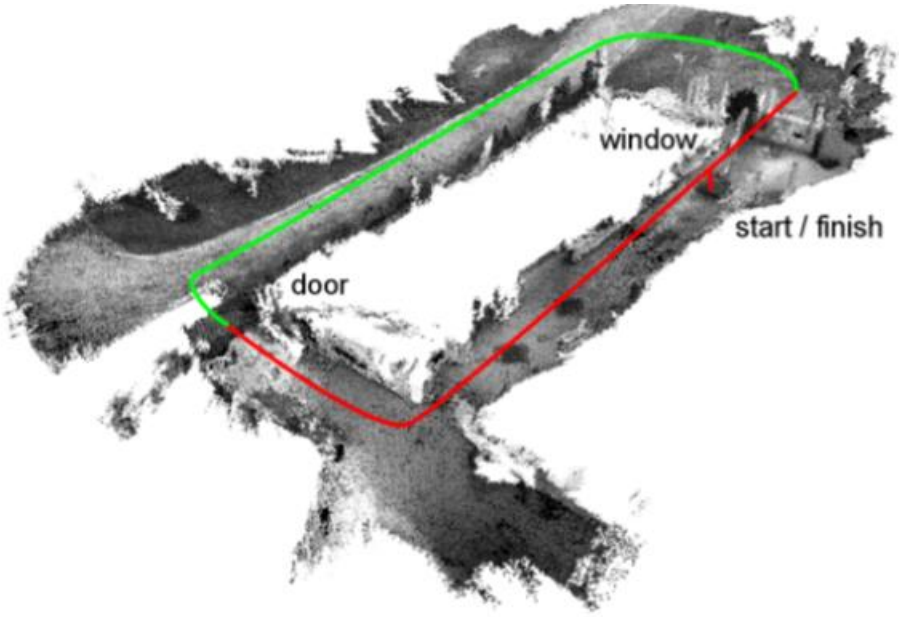

Figure 1.2 Indoor/Outdoor Mapping via Drone. Adopted from [8]

Figure 1.2 demonstrates work by Schmid et al. [8] to map both indoor and outdoor environments via a drone equipped with stereo cameras. This map was post-processed offsite to gain a higher resolution than onboard processing could accomplish.

1.1.4 Object Segmentation

Object segmentation implemented by traditional methods is subject to performance variations under fluctuating physical conditions. These limitations are apparent during illumination changes, ambiguous color overlap, and through motion in video. A patent by Y. A. Ivanov et al.

[9] highlights the benefits of using stereo vision to overcome these restrictions. Their patent details a method of using calibrated cameras to segment a person's gestures to interact with a dynamic projector as a form of control device.

Combing stereo vision with object knowledge has been shown to increase accuracy of mapping by creating more plausible surfaces [10]. Using object-level knowledge diminishes or removes some of the limitations of disparity mapping, such as ambiguity caused by reflective and transparent surfaces. F. Güney and A. Geiger apply this method to automotive applications to more accurately recover vehicle geometry that is degraded by windows. They claim a 50% reduction in error in reflective and texture-less regions [10]. This can be more clearly seen in Figure 1.3.

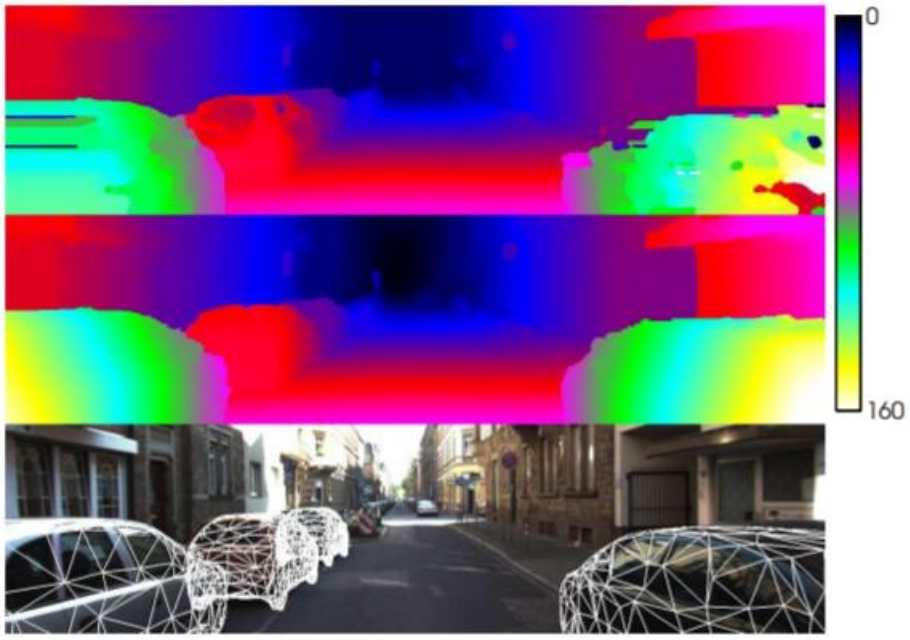

Figure 1.3 Disparity Mapping Using Stereo Vision (Top); Disparity Mapping Combined with Object Knowledge (Center); Recovered Geometry (Bottom) Adopted from [10]

Sumi et al. propose a method to segment partially occluded, known objects, including free-form objects, using stereo vision [11]. After an object is recognized, boundary features are extracted from the stereo cameras and an estimate of the orientation of the object is derived from the

corresponding, established 3D model. A representation of this technique can be seen in Figure 1.4.

Figure 1.4 Mesh Model Segmenting Partially Occluded Object. Adopted from [11]

1.1.5 Aerial Topography

A system has been proposed for the landing of unmanned aerial vehicles (UAVs) in ambiguous terrain which uses dense elevation mapping from stereo cameras along with motion estimation and other sensor information [12].

1.2 Contributions

Many different algorithms exist for stereo matching, each with a different degree of focus on improving and solving problematic areas. Three key difficulties are interpolating occluded regions, deciding object boundaries, and distinguishing fine-detailed or similarly textured objects. Assuming a selection of algorithms has enough diversity, each should have a unique combination of inherent strengths and weaknesses based on their underlying functional method of action. A set of proposed solutions from several, differing methods should, therefore, allow a consensus to be formed for a more robust, consistent, and accurate disparity map than any individual algorithm result. Several fusion methods are proposed to support this thesis. The methods that are proposed attempt to statically or adaptively select the best algorithm, or

combination of algorithm traits, to achieve this outcome. To clarify, static methods imply that a simple operation is applied uniformly. Static methods rely on no other sources of information beside the corresponding pixel values at a given location. Adaptive methods consider certain situational factors or assume prior knowledge to determine weighted rules.

Static methods include all the elementary statistical methods detailed in Section 3.1 except the weighted mean and weighted median. These approaches use a form of outside context to determine weights since they assume an approximation known to be close to the ground truth. The histogram method is also considered static since it is simply a voting method. The edge-guided approach detailed in Section 3.3 is considered adaptive since edge information is selected and treated differently based on the corresponding RGB image. The learning approaches in Section 4 are also considered adaptive since they depend on weights learned from previous inputs.

1.3 Structure of Problem Report

Chapter 2 presents a comprehensive literature review. A formal structure for stereo-matching is thoroughly defined and current trends differing from the historical process are presented. All algorithms used for fusion are briefly reviewed.

Chapter 3 explores model-based fusion methods. Simple statistical methods are outlined followed by more advanced approaches. A histogram-based fusion approach is detailed and a method to use edge extraction on RGB image data to guide algorithm selection.

Chapter 4 shifts focus toward learning-based fusion methods. Several popular approaches are examined. Support vector machines (SVMs) and bagged decision trees are described.

Chapter 5 introduces the dataset and the error metric used to evaluate all algorithms. It also discusses the reasoning for selecting the algorithms being compared. It continues to compare the results of the implementations of methods detailed in chapters 3 & 4.

Chapter 6 draws conclusions based on the results described in chapter 5. Limitations of fusion methods are discussed along with future areas of possible improvement.

2 Literature Review

2.1 Stereo-Matching

Historically, the most basic form of computational stereo-matching has roughly followed the same set of principles. These are outlined as follows [13]:

- Image acquisition
- Camera modeling
- Feature acquisition
- Image rectification
- Initial distance determination
- Interpolation between feature points

First, a pair of images must be acquired. Depending on how this data is sourced, a large amount of variability may be introduced. Images that are not taken at approximately the same time are more likely to be subject to lighting and scenery changes. The degree of difference between viewing distances and viewing angles between the camera locations will also introduce irregularities. With this, the field of view also needs to be considered. The amount of useable information is limited to the overlap of stereoscopic images. The use of different cameras can present different photometry between images as well. This will change the relative brightness seen from each image acquisition point. Slight differences in image resolution can have an effect depending on the scene and amount of variance. [13]

2.1.1 Camera Modeling with Epipolar Geometry

At the heart of stereo-matching is the problem of finding corresponding feature points between images. To achieve an accurate map and reduce the outliers in the sets of corresponding points, it is necessary to have a proper camera model. This will map the geometry between physical camera locations and provide a narrow map in pixel space to search for corresponding feature points. The narrowed search region defined by epipolar geometry of the cameras is known as the epipolar constraint [14]. Figure 2.1 illustrates the epipolar constraint in a two-camera setup. X. Chai et al. [15] describe the geometry as follows: The point **P** is a scenery point in 3-D space. Points $\mathbf{p}_l$ and

$\mathbf{p}_r$ are projections onto image planes $\boldsymbol{I}_l$ and $\boldsymbol{I}_r$, respectively. $\mathbf{C}_l$ and $\mathbf{C}_r$ are the optical centers of each respective left and right camera. The base line connects the optical centers of each image. The epipolar plane $\boldsymbol{\pi}$ is defined by spatial points **P**, $\mathbf{C}_l$ and $\mathbf{C}_r$. Any plane containing the base line is considered an epipolar plane since **P** is arbitrary. The intersection of an epipolar plane and an image plane is deemed an epipolar line. The epipolar line formed by epipolar plane $\boldsymbol{\pi}$ and image plane $\boldsymbol{I}_l$ is denoted as $\mathbf{l}_{pl}$. The point $\mathbf{p}_r$ corresponding to $\mathbf{p}_l$ in image plane $\boldsymbol{I}_l$ must fall along the epipolar line $\mathbf{l}_{pr}$. This limits the search for matching points to the epipolar lines if the camera model is known, thereby reducing search complexity and computational cost. In order to find the epipolar lines, $\mathbf{l}_{pl}$ and $\mathbf{l}_{pr}$, and epipoles, $\mathbf{e}_l$ and $\mathbf{e}_r$, the image planes must be aligned via matrix transformation.

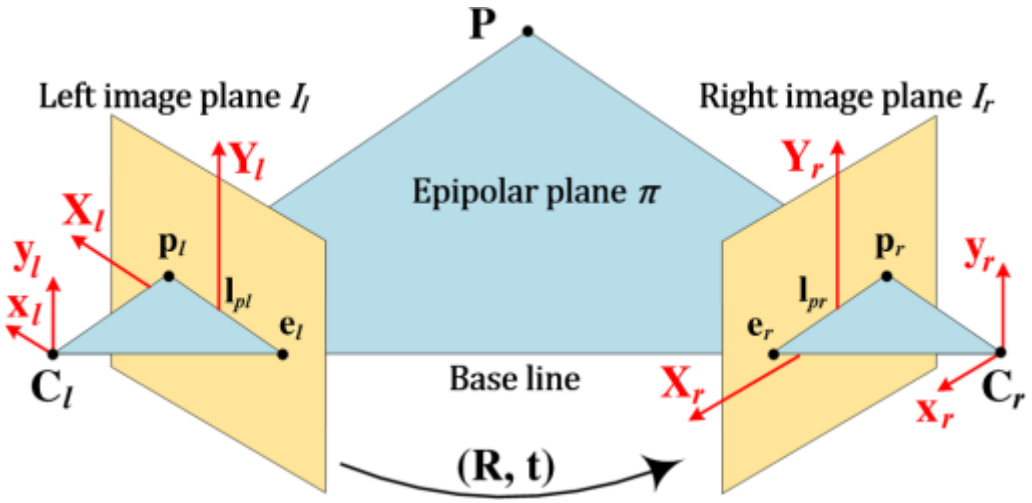

Figure 2.1 Visualization of Epipolar Geometry; Adopted from [15]

The *essential matrix* **E** contains the information to characterize the translation **T** and rotation **R** from the first image plane to the second, as shown in Figure 2.2 [16]. This gives the location of the second camera in terms of the first camera. The *fundamental matrix* **F** contains the same information as **E** but includes intrinsic properties of the camera as well [16]. This allows **E** to relate the image planes in pixel space as opposed to physical space.

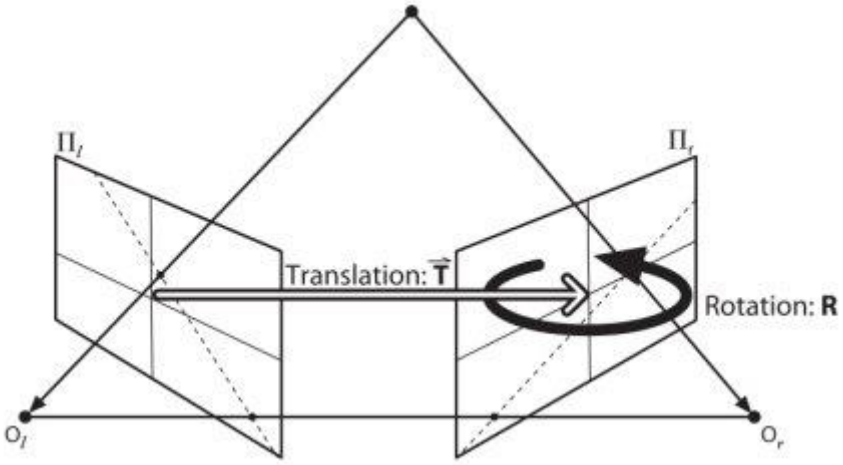

Figure 2.2 Image Plane Transformation; Adopted from [16]

The following derivation of the essential matrix and fundamental matrix closely follow the work of Bradski and Kaehler published in [16]. Point **P** viewed from the right camera location in terms of the left camera location is given by $P_r = R(P_l - T)$, where **T** is the origin of the other camera and **R** is the relative rotation. Next the epipolar plane needs to be introduced. All points **x** on a plane with normal vector **n** and passing through point **a** are constrained by:

$$(x - a) \cdot n = 0 \qquad 2.1$$

The epipolar plane contains vectors $\mathbf{P_l}$ and **T**; We can use a vector perpendicular to both to represent **n** in the above equation. We also replace the dot product with matrix multiplication by the transpose of the normal vector.

$$(P_l - T)^T (T \times P_l) = 0 \qquad 2.2$$

The relational equality defined earlier, $P_r = R(P_l - T)$, can be rewritten as $(P_l - T) = R^{-1}P_r$. Making this substitution and using the relation $R^T = R^{-1}$ yields:

$$(R^T P_r)^T (T \times P_l) = 0 \qquad 2.3$$

Now we re-write the cross product as a matrix multiplication and define matrix **S**.

$$T \times P_l = SP_l \;\Rightarrow\; S = \begin{bmatrix} 0 & -T_z & T_y \\ T_z & 0 & -T_x \\ -T_y & T_x & 0 \end{bmatrix} \qquad 2.4$$

Now making the substitution for the cross product, we have:

$$(P_r)^T RSP_l = 0 \qquad 2.5$$

The product *RS* is what is defined to be the essential matrix **E.** We would rather relate the image planes to one another, as opposed to the camera locations. This can be done by substituting the projection equations $p_l = (f_l P_l)/ Z_l$ and $p_r = (f_r P_r)/ Z_r$ and then dividing the result by $(Z_l Z_r)/ (f_l f_r)$. This leaves us with the final equation:

$$p_r{}^T E p_l = 0 \qquad 2.6$$

To find the relationship between a pixel in one image and the corresponding epipolar line in the other image, we need to introduce the intrinsic camera parameters mentioned earlier. To do this, we introduce **q** and relate it to the pixel coordinate **p** by means of intrinsic matrix **M,** which contains the camera parameters. This can be written as $q=Mp$, or $p=M^{-1}q$. Substituting this into the essential matrix equation, we are left with:

$$q_r{}^T \left(M_r{}^{-1}\right)^T E M_l{}^{-1} q_l = 0 \qquad 2.7$$

The fundamental matrix is then defined as:

$$F = \left(M_r{}^{-1}\right)^T E M_l{}^{-1} \qquad 2.8$$

Simplifying this, we now have an equation of the same form as the essential matrix equation.

$$q_r{}^T F q_l = 0 \qquad 2.9$$

Figure 2.3 demonstrates the estimated camera models and epipolar lines in two images. The top image shows the scene from the perspective of the left camera. It shows an estimate of the vanishing point off screen where the right image epipole would be. The bottom image shows the opposite from the right camera's perspective.

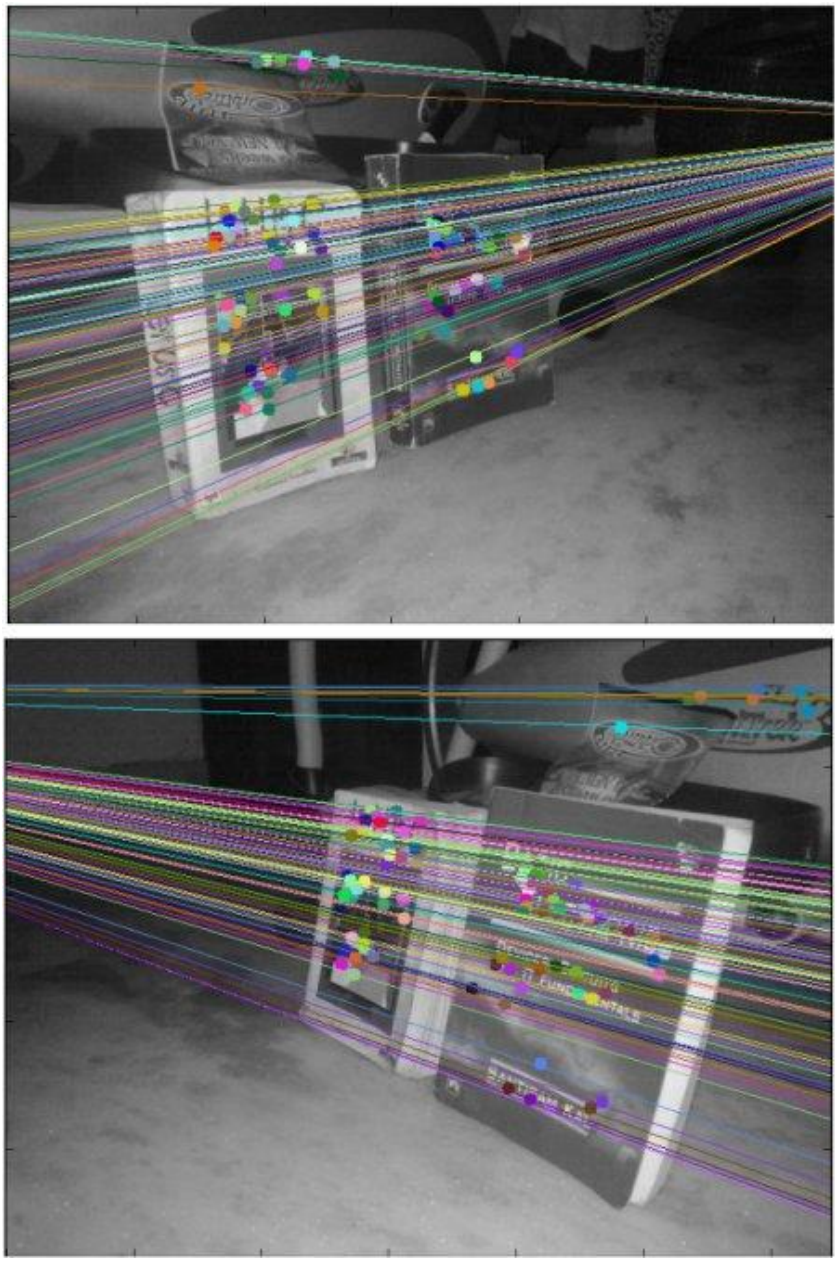

Figure 2.3 Estimated Epipolar Lines; Adopted from [14]

2.1.2 Feature Point Acquisition and Matching

Feature points must be determined between stereo images to estimate the fundamental matrix for the camera model. This, in turn, limits the number of feature points to the inliers of the epipolar constraint and increases the accuracy of disparity measurements [17]. Point-like features are usually used to obtain the camera model followed by more extensive feature matching once the

images are rectified [13]. Popular feature detection algorithms include Speeded Up Robust Features (SURF), Scale-Invariant Feature Transform (SIFT), and Oriented FAST and rotated BRIEF (ORB) due to their fast processing times and good matching rates [18]. Out of these algorithms ORB is shown to be the fasted while SIFT performs the best under most scenarios such as rotation, intensity, distortion, and shearing [18]. Current trends are relying on high resolution datasets that have minimal degrees of transformation between images and are already rectified. With this, research has transitioned to using deep learning to minimize cost functions and learn the correlation of features between images [19] [20] [21] [22]. SIFT and SURF are still popular feature extractors for image rectification in image processing toolkits such as MATLAB and OpenCV [17] [14]. Figure 2.4, below, shows epipolar constrained inliers calculated using SURF.

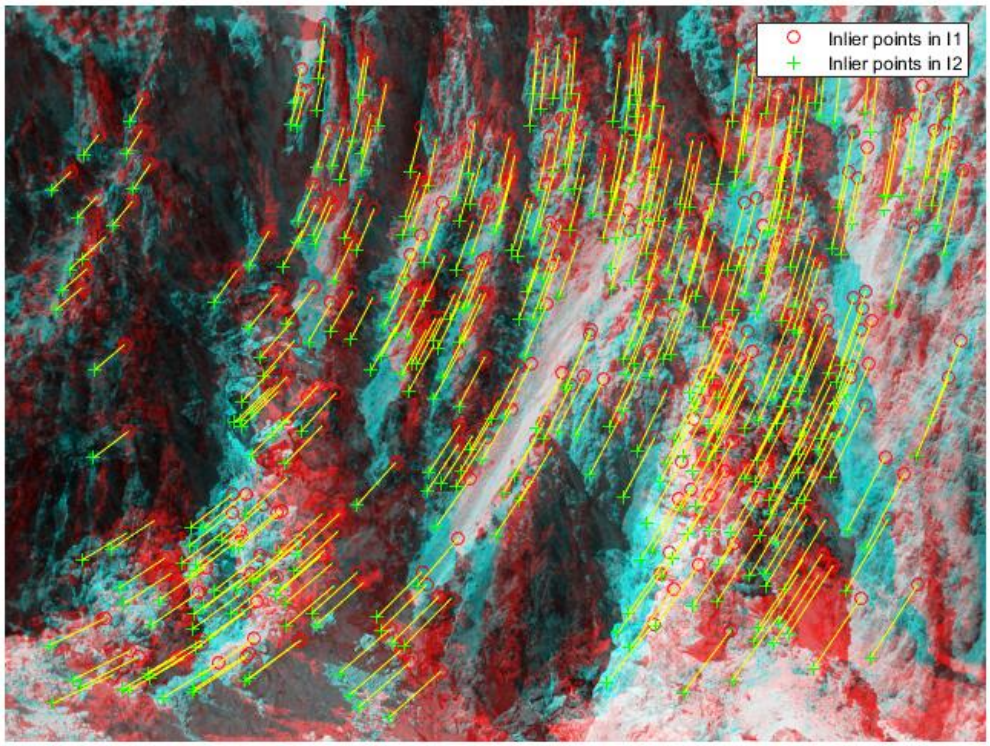

Figure 2.4 Epipolar Constrained SURF Features; Adopted from [17]

2.1.3 Image Rectification

A rectification transform shifts all the matching feature points so that they appear in the same row. It changes the perceived camera perspectives to be parallel with one another on an equal

vertical axis. Initial distance estimation is largely a matter of triangulation once the images are rectified and corresponding points are found. The rectified images in Figure 2.5 can be viewed with magenta-cyan filtered glasses to observe a 3D effect.

Figure 2.5 Rectified Stereo Images (Magenta-left, Cyan-right); Adopted from [17]

It is worth mentioning Barnard and Fischler state that "error in stereo distance measurement is directly proportional to the positional error of the matches and inversely proportional to the length of the stereo baseline" [13]. A longer translational distance between cameras increases the complexity of feature matching but improves distance estimation [13].

2.1.4 Initial Distance Determination and Interpolation

Once the images are rectified, disparity can be estimated on a pixel-by-pixel basis or by block matching. Many methods exist to minimize the matching cost function. The most common cost function is the sum of absolute differences (SAD) [23]. This is a block matching method to find the most similar kernels between images. Block matching suffers from "speckle" near object boundaries due to the foreground and background falling in the same block [16]. This can be

minimized via thresholding or filtering. More complicated methods have proposed maximization of mutual information and energy cost minimization [24].

2.2 Descriptions of Algorithms Utilized in This Work

Stereo matching is a well-studied issue in low-level computer vision; however, it is still challenging to produce accurate disparities in cases of occlusion, near object boundaries, and on repetitive texture regions or texture-less regions [25]. Numerous stereo matching algorithms have been proposed and researched to surmount these areas of shortcomings.

2.2.1 3DMST

L. Li et al. propose a stereo matching algorithm called 3D Minimum Spanning Tree (3DMST). Their algorithm offers a cost aggregation method that efficiently splices together minimum spanning tree (MST)-based support region filtering and PatchMatch-based 3D label search [21]. They use the raw matching cost from a matching cost with a convolutional neural network (MC-CNN).

2.2.2 JMR

JMR is a hybrid model proposed by P. Knöbelreiter et al. [26]. Their model combines conditional random fields (CRFs) and CNNs. The CNN is used to determine feature points and distinct edges which are then used to compute unary and binary costs of the CRF [26].

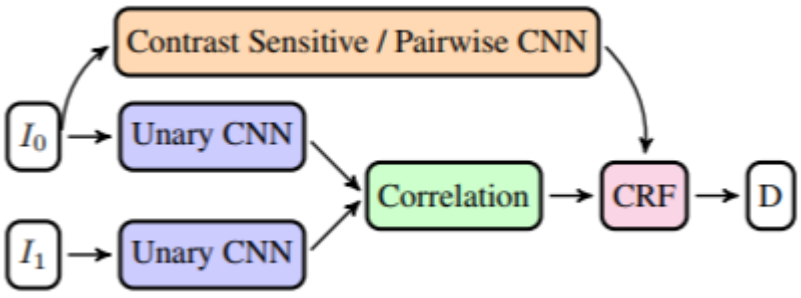

Figure 2.6 Architecture of JMR [26]

Figure 2.6 illustrates the unary CNN features of each image which are then compared in the correlation layer. From this, the matching cost becomes the unary cost of the CRF.

2.2.3 MC-CNN-acrt

MC-CNN Accurate Architecture, proposed by Zbontar and LeCun, uses a CNN on small patches with known disparity to learn the matching cost [22]. The left and right input are passed through several convolutional layers followed by rectified linear units (ReLU). The output of the CNN is a single number fed into a sigmoid nonlinearity to produce a similarity score between 0 and 1. The similarity score is then used to initialize the matching cost.

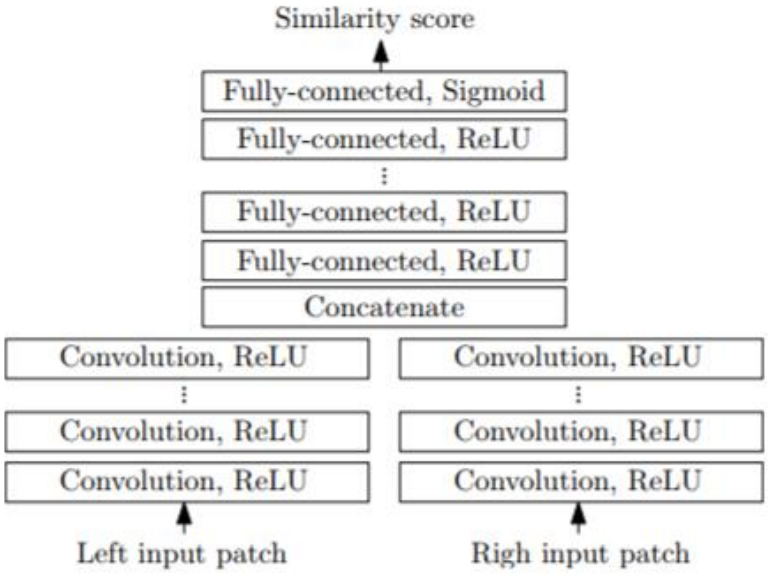

Figure 2.7 Architecture of MC-CNN Accurate [22]

A series of post-processing steps are included as follows: cross-based cost aggregation, semiglobal matching, a left-right consistency check, subpixel enhancement, a median, and a bilateral filter.

2.2.4 LW-CNN

H. Park and K.M. Lee propose a “look wide” CNN (LW-CNN) designed to mimic the way humans visually match two similar images [20]. They attempt to create a wider window for the

matching cost function that is intelligent enough to ignore irrelevant information around the target pixel. They claim the CNN's per-pixel pyramid-pooling layer provides robustness against weak texture, depth discontinuity, illumination and exposure differences [20]. This work is built on the work of [22] by following the structure for foundational layers of the CNN as seen in Figure 2.7.

2.2.5 MeshStereoExt

Zhang et al. focus on interpolating areas of occlusion. Their algorithm, MeshStereo [27], creates a triangular mesh surface map to interpolate patches of occlusion. The steps can be seen with visualizations in Figure 2.8.

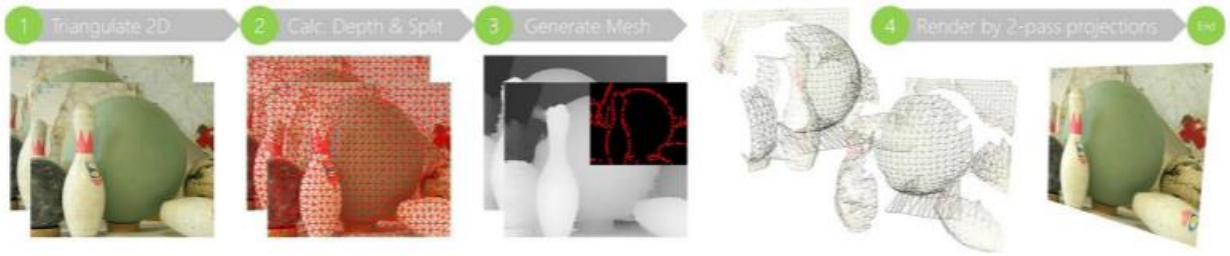

Figure 2.8 Architecture of MeshStereo [27]

From the stereo input images, 2D triangulations are created and split into a two-layer Markov random field (MRF) to handle vertices at discontinuities in the depth map. The upper layer models the splitting properties of the vertices and the lower layer optimizes region-based stereo matching [27]. The 3D mesh is lifted from the 2D triangulations according to piecewise planar disparity maps and splitting probabilities. Textures are then mapped to the meshes and new vantage points are synthesized.

2.2.6 NTDE

Non-textured Denoised Edges (NTDE) [25] proposes adaptive smoothness constraints using texture and edge information from the input image. NTDE determines non-textured regions and penalizes depth discontinuity while complementing CNN matching costs using color-based cost. From the two input images, edge maps are extracted and combined with a preliminary disparity

map to create denoised edges corresponding to depth discontinuities with high probabilities. Minor differences of neighboring disparities are penalized along denoised edges. [25]

2.2.7 PMSC

PatchMatch-Based Superpixel Cut (PMSC) [19] uses a two-layer matching cost with the goal of assigning 3D labels more accurately. The bottom layer measures the similarity between small, square patches locally by exploiting a pre-trained CNN. The top layer is developed to assemble the local matching costs in large, irregular windows induced by the tangent planes of object surfaces. Optimization stems from a multi-layer superpixel structure used to group preliminary label sets into candidate assignments, which can then be efficiently fused by α-expansion graph cut. [19]

2.2.8 SNP-RSM

Surface Normal Prediction Robust Stereo Matching (SNP-RSM) [28] proposes surface normal prediction through deep learning, overviewed in Figure 2.9. With a preliminary stereo matching disparity map and edge fusion strategy, the predicted surface normal map is converted to a disparity map by solving a least squares problem. The calculated disparity map is refined iteratively by bilateral filtering-based completion and edge feature refinement. [28]

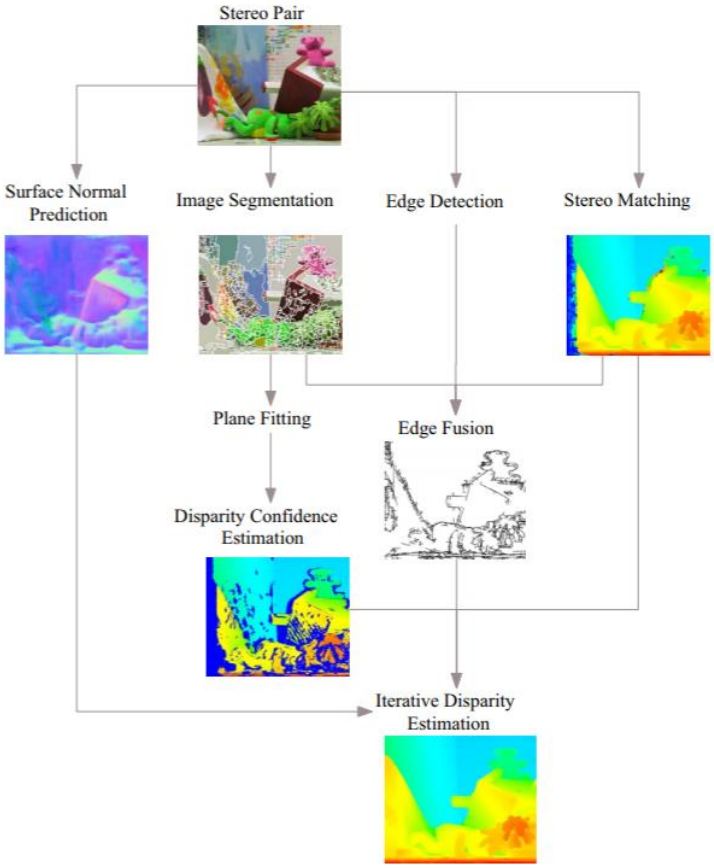

Figure 2.9 Framework of SNP-RSM [28]

2.2.9 APAP-Stereo

As-Planar-As-Possible (APAP-Stereo) is a unique algorithm proposed by M.G. Park and K.J. Yoon [3]. They describe their work as "exploiting local and dominant plane hypotheses to estimate APAP disparity maps" [3]. There is no publication available for this algorithm, but it is worth mentioning based on the merit of its benchmark results.

2.2.10 LPU

LPU is another non-published work and is only described as a "3D labeling stereo matching with content aware adaptive windows" [1]. It is unclear what the abbreviation LPU signifies.

2.2.11 MC-CNN Layout

Matching Cost Convolutional Neural Network (MC-CNN) Layout is an anonymous and unpublished algorithm that utilizes scene layout information to refine depth maps with positive results [2].

3 Model-Based Fusion Methods

3.1 Elementary Statistical Approaches

For the methods described in this section, the images to be fused were stacked in a matrix configuration where each layer represented the result of a different stereo algorithm. Corresponding pixels from each layer were grouped in a set and analyzed using the methods described below to select or create a candidate element, where each candidate was used to form a new, fused image. Therefore, each calculated pixel only had access to information from the algorithm results at its corresponding location and did not infer anything from surrounding pixels.

3.1.1 Mode

Mode is defined as the value in a set that occurs with the highest frequency. This is determined by first sorting the set followed by finding the maximum frequency of occurrences.
For this analysis the lowest valued mode was accepted. If no mode existed, the lowest valued element was accepted.

3.1.2 Mean

The statistical mean $\bar{x}$ for a set $\{x_1, x_2, \dots, x_n\}$ with n elements is defined as

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \qquad 3.1$$

Given a normal distribution, mean can be a strong predictor. However, it is naive to expect a normal distribution from the selection of algorithms without outliers skewing the mean toward results outside of a central grouping.

3.1.3 Median

The median is defined as the center element in an ordered set $\{x_1, x_2, \dots, x_n\}$ with n elements. For an odd set, the index of the median is $(n+1)/2$. For an even set, the mean of the values corresponding to the indices $floor(n+1)/2$ and $ceil(n+1)/2$ is determined as the median. Median offers some protection against outliers, unlike statistical mean.

This is a well-tested technique used by photographers to remove outliers (tourists) from busy scenes of landmarks by using multiple shots [29] [30]. This idea can be applied to calculated disparity maps, given that there is a general consensus where more than half of the images are within an acceptable range or the outliers are balanced in either direction.

3.1.4 Weighted Mean

The weighted mean $\bar{x}$ of a set of elements $\{x_1, x_2, \dots, x_n\}$ with respective weights $\{w_1, w_2, \dots, w_n\}$ is defined as

$$\bar{x} = \frac{\{\sum_{i=1}^{n} w_i x_i\}}{\{\sum_{i=1}^{n} w_i\}} \quad 3.2$$

The set of weights were determined by first choosing the median as a surrogate for the ground truth. A set of residual errors $\{r_1, r_2, \dots, r_n\}$ were calculated as the absolute value of the difference between the set of elements $\{x_1, x_2, \dots, x_n\}$ and the surrogate. The set of corresponding weights were then calculated as

$$w_i = e^{-r_i} \quad 3.3$$

3.1.5 Weighted Median

Weighted median differs from median in the sense that its goal is to determine the element that most evenly balances the weights of ordered elements rather the raw count of ordered elements. Generally, given a set of ordered elements $\{\mathrm{x}_1, \mathrm{x}_2, \dots, \mathrm{x}_\mathrm{n}\}$ with respective weights $\{\mathrm{w}_1, \mathrm{w}_2, \dots, \mathrm{w}_\mathrm{n}\}$ that satisfy

$$\sum_{i=1}^{n} w_i = 1 \quad 3.4$$

the weighted median is the element x_k that satisfies the following conditions:

$$\sum_{i=1}^{k-1} w_i \leq 1/2 \tag{3.5}$$

$$\sum_{i=k+1}^{n} w_i \leq 1/2 \tag{3.6}$$

More specifically, the weighted median can be realized as a lower-weighted median or an upper-weighted median if either of the above conditions are equal to $1/2$. The lower-weighted median, originally proposed by Edgeworth [31], is defined by the conditions:

$$\sum_{i=1}^{k-1} w_i < 1/2 \tag{3.7}$$

$$\sum_{i=k+1}^{n} w_i = 1/2 \tag{3.8}$$

Conversely, the upper-weighted median is defined by the conditions:

$$\sum_{i=1}^{k-1} w_i = 1/2 \tag{3.9}$$

$$\sum_{i=k+1}^{n} w_i < 1/2 \tag{3.10}$$

The set of weights were chosen using the same process described in Section 3.1.4 for weighted mean. If both the lower-weighted median and the upper-weighted median existed, the mean was determined and assigned a weight of zero. This new value was accepted as the weighted median since it equally pivots the total weights on either side of the ordered element, making it a true median. The basis for this method is that it will always produce the same value as the median

given equally distributed weights. Weighted median offers a similar robustness against outliers as median.

3.2 Histogram

Histograms follow a basic rule that organizes a given number of elements, $\boldsymbol{n}$, into a set number of bins, $\boldsymbol{k}$. The number of observations in each bin, $\boldsymbol{m_i}$, are totaled in the form of a bar graph. This rule is written as follows:

$$n = \sum_{i=1}^{k} m_i \qquad 3.11$$

This method provides a simple way for algorithms to vote on which range of values each pixel should be in. This is also an effective way to remove outliers since it is a majority voting scheme. The difficulties are determining the proper number of bins to provide useful insight. If the bin size is too large, it may not remove outliers. If the bin size is too small, the ranges may be too specific and not form any consensus.

Since the useful number of bins may be different depending on the range of the set, it was decided to set the size of the bin instead of the number. This was done by finding the maximum value in the stack of images and dividing it by the desired size of each bin. This is presumed to be more useful because the level of agreement between algorithms is similar regardless of the range of the disparity data.

The values in the largest bin are extracted for each pixel location. The mean of these values is used to interpolate a new value and create a fused disparity map based on majority voting.

3.3 Edge-guided

This method is unique in its approach since it uses outside information to guide the fusion. More specifically, the RGB input from the left camera is used to clean up the disparity result. This is done by applying edge map information to the disparity map.

Sobel edge detection was used for finding edge maps on each of the disparity results and on the left RGB camera input. Simple AND logic was used to find where the RGB edge mask, $\boldsymbol{E_{RGB}}$, aligned with each of the disparity result edge masks, $\boldsymbol{E_i}$, in image stack $\boldsymbol{I}$. If there was no overlap found between the RGB edge mask and any of the other edge masks, the median was taken between all algorithms for that point. If there were matching edges, only those algorithms were used to determine the composite disparity map at those points. The points with overlap were assumed to be close enough in agreement, due to common edges, that the mean would suffice as a fusion device. These operations are described in the two equations below with the mean and median taken along the third dimension, iterated by $\boldsymbol{i}$.

$$mean\{I_i(E_{RGB} \wedge E_i), I_{i+1}(E_{RGB} \wedge E_{i+1}), \dots, I_{end}(E_{RGB} \wedge E_{end})\} \qquad 3.12$$

$$median\{I_i\big(\neg(E_{RGB} \wedge E_i)\big), I_{i+1}\big(\neg(E_{RGB} \wedge E_{i+1})\big), \dots, I_{end}\big(\neg(E_{RGB} \wedge E_{end})\big)\} \qquad 3.13$$

A median filter was used to remove any speckle noise created by the new edge boundaries and from error caused by matching edges that did not provide disparity separation boundary information. These were false edges found within textures on flat surfaces or that were part of the same plane in the RGB image.

4 Learning-Based Fusion Methods

The inputs used for learning-based methods were approximately 150,000 sets of 55 features. Features were chosen from a simple patch comprised of the top, bottom, left, and right pixels with respect to the center pixel to be interpolated. This scheme was created to account for small vertical and horizontal offsets of information. These pixel values were gathered from each of the 11 algorithms to make a predictor vector. Patches were chosen from random locations within a test image. Points that fell within the occlusion mask were not used for training since they could not be reliably determined from the stereo images.

4.1 Support Vector Machines

At their root, support vector machines (SVMs) decide classification boundaries by maximizing the marginal distance between data points. SVMs can be used for regression with a few modifications. This is sometimes referred to as Support Vector Regression (SVR) [32]. Here we use an epsilon-intensive loss function, meaning the cost of all training points within the boundaries of the epsilon band is zero. Figure 4.1 shows a two-dimensional, linear example of the epsilon band and cost of points outside the band.

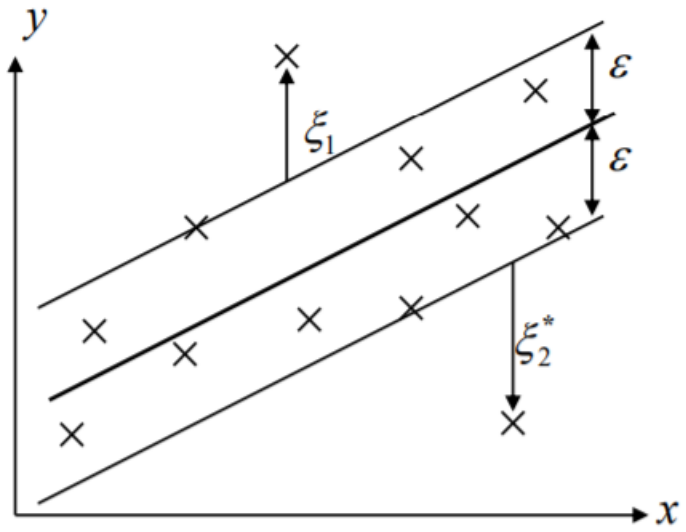

Figure 4.1 SVR Example with Epsilon Band [33]

The center of the band is denoted:

$$y = wx + b \tag{4.1}$$

Where *w* is a list of coefficients and *b* is the intercept. The goal of SVR is to find the solution that minimizes:

$$\frac{1}{2}\|w\|^2 + C\sum_{i=1}^{m}(\xi_i + \xi_i^*) \tag{4.2}$$

Where ξ_i, ξ_i^* are the deviations above and below the epsilon band, respectively.
Under the constraints:

$$\begin{gathered} y_i - wx_i - b \leq \varepsilon + \xi_i \\ wx_i + b - y_i \leq \varepsilon + \xi_i^* \\ \xi_i, \xi_i^* \geq 0 \end{gathered} \tag{4.3}$$

Both ξ_i and *C* may be optimized using gird search [32].

For a non-linear SVR a kernel is used to map features to a higher dimensional space, so linear operations can then be performed. A positive definite kernel, called a Radial Basis Function (RBF) was used. This kernel mapping function, $K(x_i, x_j)$,
is a gaussian kernel taking the form [32] [33] [34]:

$$K(x_i, x_j) = exp\left(\frac{-\|x_i - x_j\|^2}{2\sigma^2}\right) \tag{4.4}$$

Where σ is the standard deviation of the gaussian distribution.
After applying the kernel, the transformed linear equation becomes the following:

$$y = \sum_{i=1}^{m}(\alpha_i - \alpha_i^*)K(x_i, x_j) + b \tag{4.5}$$

Where m is the number of points in the training data and α_i, α_i^* are associated costs for each point above or below the epsilon band, respectively. α_i, α_i^*>0 for points outside the epsilon band and α_i, α_i^*=0 for points within the epsilon band.
For this implementation, epsilon was determined automatically via MATLAB optimization. A very fine gaussian kernel was implemented by setting the kernel scale factor to

$$\frac{1}{8}\sqrt{P} \qquad 4.6$$

where P is the number of predictors or features.

The model was trained using five-fold cross-validation. Principle component analysis (PCA) was also used in the training. This was set to account for 97% of the variance. This reduced training times and error between cross-validation. The data was also standardized to have mean zero and standard deviation one.

4.2 Bagged Trees

Decision trees offer some distinct advantages that make them appealing regression learners. They are fast predictors and offer insight into the strongest predicting features [35]. They are considered a "white box" model, meaning the decision-making logic can easily be viewed and interpreted after training [36].

Bagging, or bootstrap aggregation, averages trees created from many samples of data to lower variance and, in effect, smooth decision boundaries. Minimum leaf size was determined by evaluating the mean squared error (MSE) of several sizes over 50 trees as seen in Figure 4.2 below. The bagged tree model was implemented with 300 learners and a minimum leaf size of one.

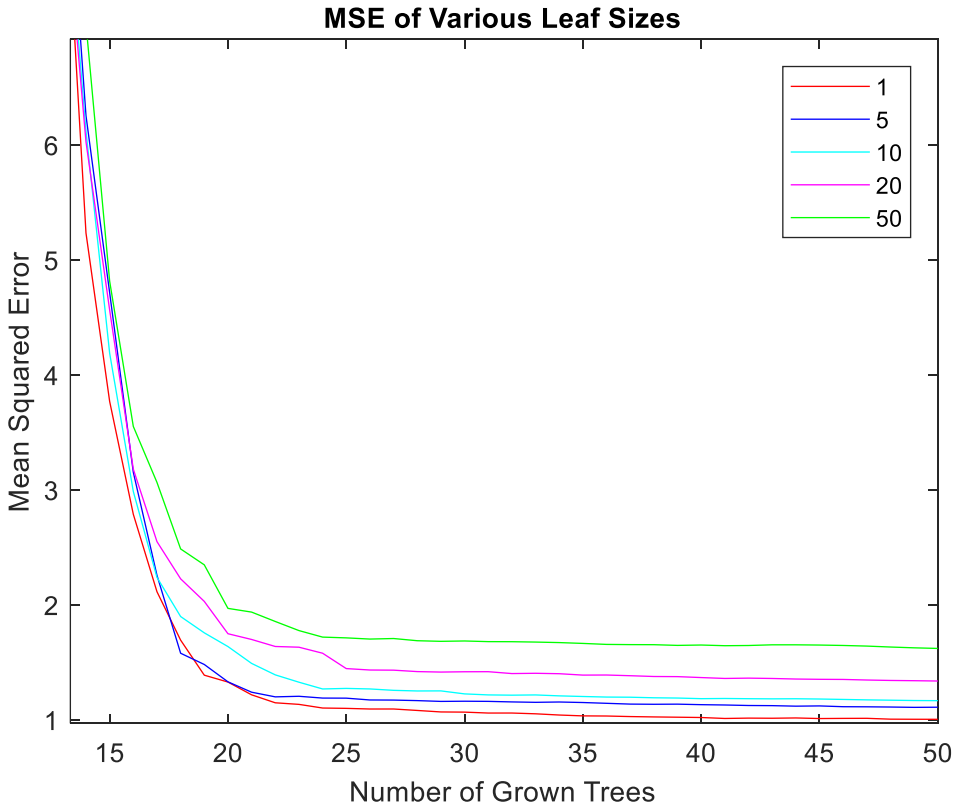

Figure 4.2 Mean Squared Error of Various Leaf Sizes

5 Experimental Results

5.1 Dataset

The dataset used for this work is a combination of three datasets from Middlebury [37] [23] [38]. A sample of the dataset with the left camera image on top and the corresponding ground truth beneath it is shown in Figure 5.1.

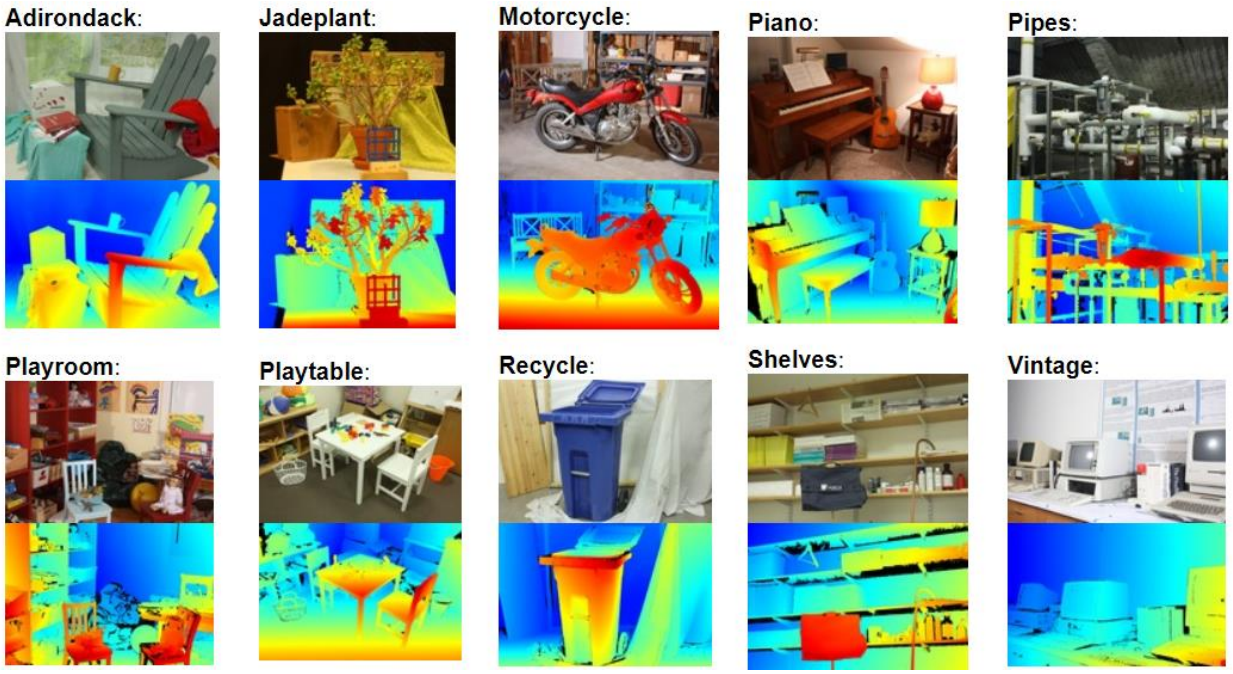

Figure 5.1 Sample of Dataset with Ground Truth

Using a laser scanner can limit the coverage of a scene and suffer calibration issues when relating the ground truth to the input images [38]. For this reason, it is not the best solution to use a separate device to achieve a ground truth. D. Scharstein et al. [38] build on the ideas proposed by D. Scharstein and R. Szeliski [39] to achieve a disparity directly from input views and illumination disparities in half-occluded regions.

Standard checkerboard calibration schemes were used to acquire calibration images. Coded images were created from the stereo cameras and projected Gray codes on the scene. The correspondence of the coded images was used to find 2D view disparities that were then used to

rectify the images. They were cycled back through the pipeline to create the 1D view disparities. Merging these disparities allowed for the creation of 1D illumination disparities to calibrate the projector and lead to a final, subpixel-accurate disparity. Each disparity pair was in agreeance of within 0.5 pixels. D. Scharstein et al. report an accuracy of 0.2 pixels for most surfaces. [38]

A broader overview of the pipleline can be seen in Figure 5.2. Slightly warped calibrations and disparites were created to demostrate how quickly large errors can occur when imperfect settings are used with stereo algorithms.

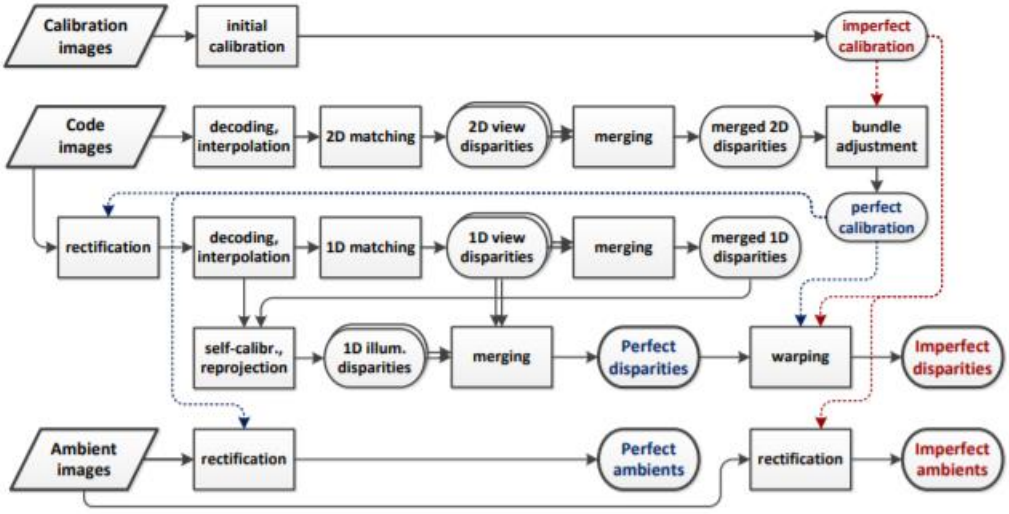

Figure 5.2 Pipeline of Ground Truth Derivation. Adopted from [38]

The Middlebury dataset provided full-resolution, half-resolution, and quarter-resolution images. Each pair of images was obtained from a six-megapixel DLSR camera [38]. The half-resolution images used in this evaluation were not all equal. The maximum resolution was 994 x 1482. While most other images were close to this size, two images mixed in from an older dataset were about half this size. For reference, these images are entitled “ArtL” and “Teddy.” Images in the dataset varied in size primarily due to necessary cropping that was performed to eliminate erroneous edges resulting from the image rectification transform.

5.1.1 Description of Error Metrics

As standard convention, disparity maps were calculated from the perspective of the left image, and error was calculated from that perspective. There are four levels of benchmarking to gauge the number of erroneous pixels. These are determined by hard limits. The error is defined by the percentage of pixels whose absolute difference from the ground truth is greater than the threshold. The threshold allowances in order of strictness are 0.5, 1, 2, and 4 units. For the sake of data reduction, results are shown for strict thresholds of pixels whose error is greater than 0.5 and for a more lenient middle ground of errors greater than 2. Occlusion masks were provided for each image so that only the nonoccluded area was compared against the ground truth when calculating error.

A weighted average is used to benchmark the overall performance of an algorithm. Each pair of images is given an equal weight of eight percent except for three pairs that introduce irregularities, such as difference lighting conditions; these are discounted to half the normal weight – four percent. The weights of all benchmarked disparity maps sum to one hundred percent.

One caveat of this work is that all benchmarked algorithms are based off training data that has the ground truth publicly available. This is the only way to be able to measure the performance of fusion methods against the algorithms in a fair, relative manner. Using results from training data should not have any effect on the outcomes of fusion methods since these are post-processing steps that do not directly gather any information from the ground truth but rather look for a consensus in the data.

The range of ground truth values for each scene is different, as it is dependent on the situation. This may cause visualizations to be slightly misleading. Each visualization is normalized by its highest value. This was done in order to provide the highest contrast in each image. While this is more visually appealing, it is more difficult to see any possible effect that the range of values has on algorithm performance. Normalizing each image by the highest overall value may offer more insight into whether there is a degradation on scenes with high disparity vs low disparity, but this insight is better achieved with residual error maps.

5.1.2 Selection of Algorithms

Eleven algorithms were selected based on their performance on the loosest error threshold. This was done to remove the most serious outliers and guarantee most of the error was contained within a threshold of four. Certain revised algorithms were removed in order to remove redundancy. It would not be possible to obtain an even spread of strengths and weaknesses if multiple algorithms were based on the same technique. Figure 5.3 shows sorted algorithm performance that was used to determine the cutoff point. "LPU" was the last accepted algorithm since it appeared at the end of a short plateau. The algorithms that were utilized were current as of March 2017.

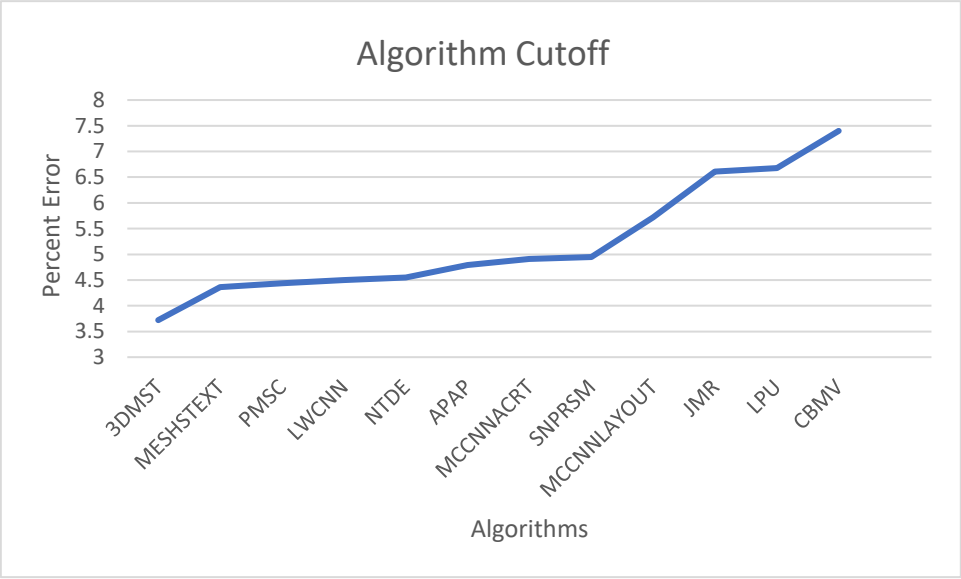

Figure 5.3 Sorted Error of Algorithms

Due to the overwhelming number of algorithms with top-performing results available for half-resolution images, that resolution was chosen for analysis. This also helped save on compute time for image manipulation and training; however, it should be noted that Middlebury upscales all results to full-resolution when scoring [40]. Therefore, it is possible that very small discrepancies may be seen in Table 5.1 and Table 5.2 as they were scored based on their current half-resolution to avoid any further information changes due to upsizing an image.

		Weighted Average'	Adirondack'	ArtL'	Jadeplant'	Motorcycle'	MotorcycleE'	Piano'	PianoL'	Pipes'	Playroom'	Playtable'	PlaytableP'	Recycle'	Shelves'	Teddy'	Vintage'
	Algorithms / Weights		8%	8%	8%	8%	8%	8%	4%	8%	4%	4%	8%	8%	4%	8%	4%
1	'3DMST'	**38.04%**	**24.27%**	49.35%	45.04%	38.00%	31.38%	**35.72%**	**45.36%**	36.41%	48.21%	**27.91%**	25.20%	**34.72%**	60.25%	47.42%	**34.31%**
2	'APAP-Stereo'	49.39%	42.76%	50.89%	54.07%	55.63%	53.67%	40.85%	47.11%	50.63%	56.10%	39.75%	34.45%	51.54%	62.03%	51.85%	57.17%
3	'LPU'	40.80%	25.80%	45.95%	**38.52%**	49.03%	37.42%	39.44%	53.55%	26.73%	52.97%	46.23%	**24.77%**	42.44%	67.11%	43.74%	52.42%
4	'LW-CNN'	39.89%	32.85%	**43.82%**	41.90%	35.59%	29.41%	43.32%	53.38%	32.08%	51.93%	35.65%	32.43%	39.33%	60.63%	43.92%	46.33%
5	'MCCNN_Layout'	39.28%	28.34%	47.44%	39.94%	38.55%	**28.22%**	39.82%	50.69%	26.38%	52.73%	38.51%	36.52%	35.30%	**57.66%**	**43.32%**	54.70%
6	'MC-CNN-acrt'	39.92%	30.19%	46.60%	39.96%	38.84%	28.60%	41.14%	50.23%	**24.27%**	54.92%	38.84%	36.97%	35.81%	60.65%	43.88%	60.91%
7	'MeshStereoExt'	41.47%	28.50%	48.83%	51.09%	42.84%	35.48%	37.62%	49.06%	37.70%	**47.57%**	32.36%	29.40%	42.75%	59.89%	49.08%	41.41%
8	'NTDE'	41.22%	34.96%	45.26%	41.82%	**34.99%**	32.27%	43.64%	53.15%	29.32%	53.97%	45.63%	36.85%	38.41%	62.12%	44.39%	51.78%
9	'PMSC'	39.48%	25.45%	47.27%	44.00%	39.22%	32.13%	37.36%	48.87%	38.29%	50.64%	29.94%	27.55%	36.11%	60.84%	48.33%	45.37%
10	'JMR'	56.16%	50.56%	63.23%	70.09%	49.05%	46.18%	51.80%	60.70%	45.41%	60.03%	50.62%	48.41%	53.99%	73.72%	61.80%	78.00%
11	'SNP-RSM'	41.25%	31.85%	50.11%	45.37%	39.39%	31.00%	41.56%	50.26%	27.96%	55.74%	39.67%	37.45%	36.63%	60.84%	44.06%	54.04%

Table 5.1 Algorithm-Image Pairs with Error Threshold of 0.5

		Weighted Average'	Adirondack'	ArtL'	Jadeplant'	Motorcycle'	MotorcycleE'	Piano'	PianoL'	Pipes'	Playroom'	Playtable'	PlaytableP'	Recycle'	Shelves'	Teddy'	Vintage'
	Algorithms / Weights		8%	8%	8%	8%	8%	8%	4%	8%	4%	4%	8%	8%	4%	8%	4%
1	'3DMST'	**7.28%**	1.57%	5.59%	**10.94%**	4.09%	4.40%	10.12%	15.69%	5.15%	**9.93%**	**5.73%**	**5.29%**	**6.51%**	29.98%	**3.20%**	**6.95%**
2	'APAP-Stereo'	7.88%	3.07%	7.56%	13.85%	4.59%	4.78%	10.75%	16.12%	5.54%	10.25%	8.68%	8.14%	7.83%	**12.39%**	4.70%	7.93%
3	'LPU'	10.55%	3.28%	7.02%	11.88%	6.03%	6.51%	13.69%	26.16%	7.66%	15.56%	9.79%	6.61%	10.81%	36.10%	3.64%	21.77%
4	'LW-CNN'	8.56%	2.89%	6.04%	13.24%	**3.28%**	**3.41%**	11.81%	17.46%	**3.85%**	12.05%	10.41%	9.70%	7.08%	30.63%	3.23%	14.31%
5	'MCCNN_Layout'	9.73%	3.58%	8.36%	14.81%	4.13%	4.37%	12.71%	15.62%	4.80%	12.46%	14.91%	12.87%	7.93%	25.08%	5.05%	18.02%
6	'MC-CNN-acrt'	10.42%	3.42%	8.51%	16.41%	3.91%	3.92%	12.67%	18.54%	4.49%	14.70%	15.14%	13.39%	7.05%	30.63%	4.57%	24.85%
7	'MeshStereoExt'	9.58%	3.59%	6.91%	18.39%	5.48%	5.97%	**8.90%**	**13.83%**	8.27%	11.21%	8.88%	8.29%	10.64%	31.38%	4.50%	12.19%
8	'NTDE'	10.21%	4.58%	7.38%	16.03%	4.15%	4.49%	13.38%	19.38%	5.35%	14.48%	12.10%	11.77%	8.46%	33.58%	3.35%	17.81%
9	'PMSC'	8.39%	**1.50%**	**4.62%**	11.47%	3.81%	4.12%	11.98%	18.25%	5.41%	12.61%	8.02%	6.90%	7.67%	31.72%	3.22%	17.74%
10	'JMR'	10.96%	5.36%	6.77%	16.78%	4.26%	4.45%	14.30%	21.92%	5.61%	15.14%	11.28%	9.27%	8.02%	35.81%	4.93%	30.32%
11	'SNP-RSM'	11.23%	5.57%	11.94%	19.88%	5.63%	4.99%	12.66%	16.95%	6.10%	15.32%	14.86%	12.80%	7.71%	31.30%	4.54%	18.59%

Table 5.2 Algorithm-Image Pairs with Error Threshold of 2

In order to provide insight and verify how useful each selected algorithm was the best performing algorithms were selected on a pixel-by-pixel basis relative to the ground truth and then visualized over a sample image. This also provided a way to visualize and possibly identify which areas algorithms were well-suited for. Each algorithm was assigned an indexing number, as seen in Table 5.1 and Table 5.2, and color-coded by column for better visualization. Figure 5.4 shows a visual representation of which algorithm had the lowest error in each pixel.

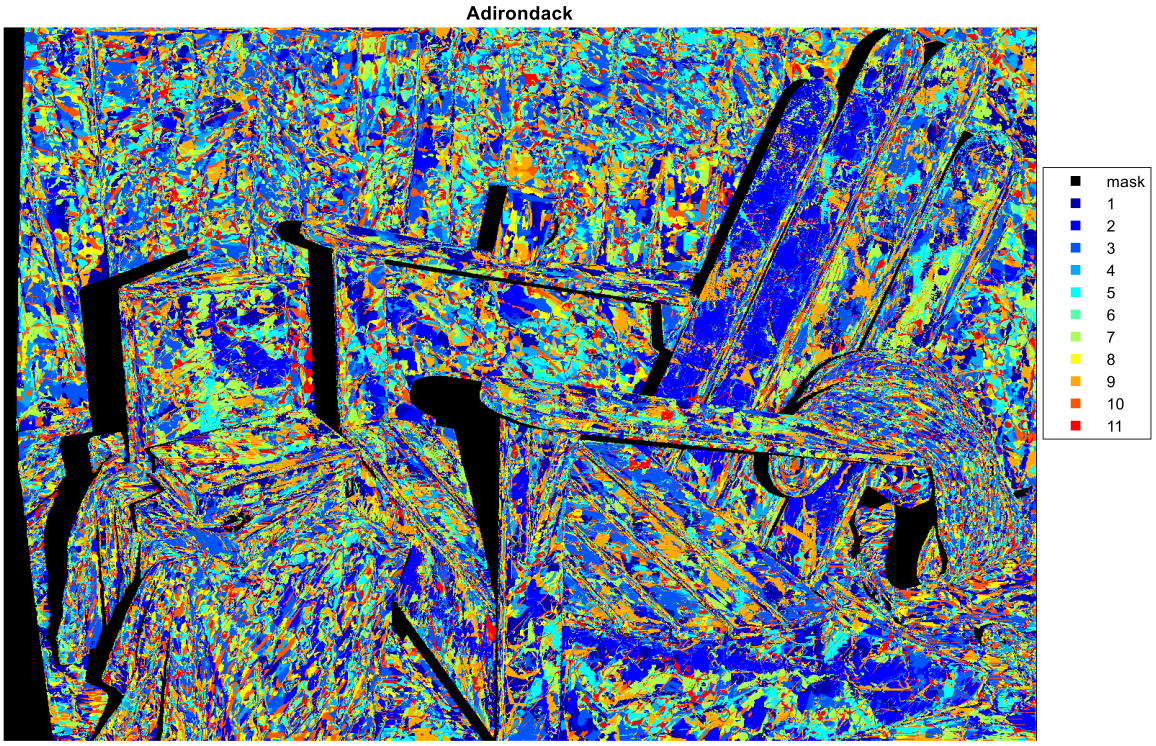

Figure 5.4 Color-Coded Selection of the Top-Performing Algorithm per Pixel

Different algorithms perform the best between regions of object separation. These discontinuities make it possible to distinguish objects when only looking at color-coded algorithms. Also note that each algorithm appears in small clusters that are distributed throughout the image. These clusters get much smaller on textured objects such as the towel draped over the arm of the Adirondack as seen above.

Error was calculated for each of the composite images made up of the best performing algorithm per pixel. Table 5.3 provides the maximum score possible if the best algorithm could be correctly picked for any given circumstance. Error rates were cut substantially in both cases showing that it is possible to achieve marginally better results given a proper selection mechanism. This provides a reasonable upper bound for how well a combination of algorithms could potentially work. However, this does not reflect the improvement possible if new values are interpolated between algorithms.

	Weighted															
	Average'	Adirondack'	ArtL'	Jadeplant'	Motorcycle'	MotorcycleE'	Piano'	PianoL'	Pipes'	Playroom'	Playtable'	PlaytableP'	Recycle'	Shelves'	Teddy'	Vintage'
Weights		8%	8%	8%	8%	8%	8%	4%	8%	4%	4%	8%	8%	4%	8%	4%
Error Threshold 0.5	8.49%	2.61%	11.15%	11.08%	7.18%	5.87%	8.29%	14.42%	5.01%	9.90%	7.04%	5.33%	7.82%	23.71%	10.49%	7.62%
Error Threshold 2	1.68%	0.18%	1.52%	2.65%	1.16%	1.17%	2.40%	4.84%	1.14%	1.79%	1.47%	1.11%	1.60%	5.96%	0.59%	0.88%

Table 5.3 Error of Composite Images Using Best Known Values

From the color-coded images, histograms were produced to show the number of "best" pixels contributed by each algorithm. This does not guarantee that they are within an acceptable range of error, only that they have the minimum possible amount with respect to the select of algorithm results. Figure 5.5 shows a collection of histograms for the first four images.

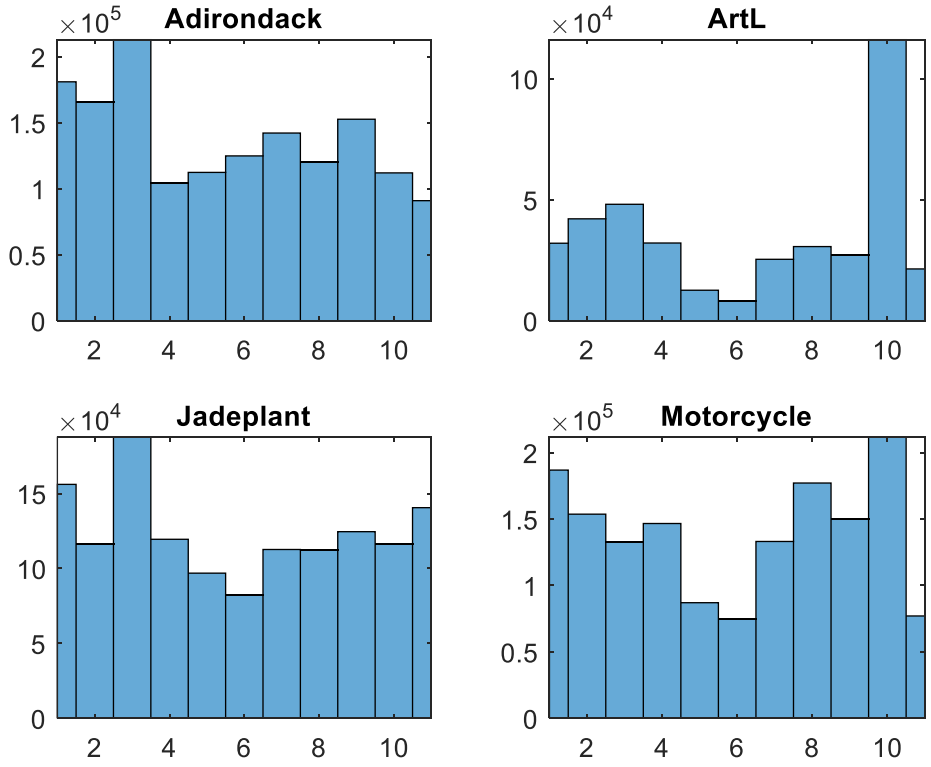

Figure 5.5 Histograms of Contributed Pixels per Algorithm

An algorithm that contributed a large number of pixels closest to the ground truth does not necessarily mean it scored the best when calculated for error. It is important to remember that it was possible for an algorithm to have more pixels that fall within the error threshold but not contribute many pixels with the best absolute error. An algorithm that contributed the most pixels with the lowest absolute error may still have a considerable number of pixels that fall outside of

the error threshold. This can be seen when comparing the tenth algorithm of "ArtL" in Figure 5.5 with the results in Table 5.1. The highest contributor was the worst scoring. In other words, the tenth algorithm, JMR, was extremely accurate for many pixels, but well outside the error threshold for the remaining pixels. This insight into its imbalance made it a promising candidate for fusion with other algorithms. Removing outliers from JMR and keeping its accurate pixels for consensus permitted good results discussed near the end of Section 5.2.4.

5.2 Model-Based

A summary of all model-based results can be seen in Table 5.4 and Table 5.5, below. Weighted mean shows very consistent performance across all images for both error thresholds. Conversely, mean is among the worst in individual error rates and the worst in overall performance for both error thresholds. Weighted median is a close second in overall performance if mode is disregarded. Mode has the lowest weighted average when measuring with the strictest error threshold. This is wholly due to two images that it performs exceptionally well on, thereby skewing the average. Results are discussed in greater detail for each model-based fusion method in the remainder of this section.

	Weighted Average'	Adirondack'	ArtL'	Jadeplant'	Motorcycle'	MotorcycleE'	Piano'	PianoL'	Pipes'	Playroom'	Playtable'	PlaytableP'	Recycle'	Shelves'	Teddy'	Vintage'
Weights		8%	8%	8%	8%	8%	8%	4%	8%	4%	4%	8%	8%	4%	8%	4%
Mode	**33.70%**	24.32%	**25.68%**	38.17%	35.88%	29.43%	37.87%	47.28%	27.40%	49.66%	32.38%	28.40%	34.32%	57.63%	**26.22%**	40.22%
Mean	37.35%	23.14%	47.53%	41.70%	37.62%	27.48%	36.16%	49.47%	27.48%	47.58%	33.65%	30.02%	33.30%	59.04%	44.03%	47.22%
Median	34.04%	21.29%	43.94%	35.15%	35.17%	25.65%	34.03%	44.56%	23.00%	45.76%	29.28%	26.03%	31.75%	56.75%	42.68%	37.34%
Weighted Mean	33.74%	**20.89%**	44.09%	**34.43%**	35.39%	**25.20%**	**33.43%**	**43.66%**	**22.40%**	**45.02%**	**28.61%**	**25.49%**	**31.44%**	**56.39%**	42.90%	38.48%
Weighted Median	33.88%	21.22%	43.73%	34.98%	**35.07%**	25.55%	33.91%	44.38%	22.88%	45.49%	28.96%	25.72%	31.68%	56.49%	42.61%	**37.01%**
Histogram	34.39%	21.78%	44.57%	35.29%	35.89%	25.51%	33.67%	44.06%	22.87%	45.11%	29.50%	26.44%	32.37%	56.77%	43.18%	41.19%
Edge Guided	34.12%	21.33%	44.01%	35.21%	35.24%	25.73%	34.10%	44.58%	23.09%	45.90%	29.44%	26.26%	31.78%	56.76%	42.69%	37.33%

Table 5.4 Model-Based Fusion with Error Threshold of 0.5

	Weighted Average'	Adirondack'	ArtL'	Jadeplant'	Motorcycle'	MotorcycleE'	Piano'	PianoL'	Pipes'	Playroom'	Playtable'	PlaytableP'	Recycle'	Shelves'	Teddy'	Vintage'
Weights		8%	8%	8%	8%	8%	8%	4%	8%	4%	4%	8%	8%	4%	8%	4%
Mode	7.36%	2.06%	**5.05%**	10.21%	3.37%	3.53%	10.96%	15.31%	3.84%	11.20%	9.18%	7.39%	6.61%	26.55%	**2.77%**	10.25%
Mean	11.35%	3.18%	10.75%	21.32%	5.72%	6.08%	12.89%	20.81%	10.72%	14.14%	11.98%	9.71%	7.64%	31.70%	4.61%	19.99%
Median	6.73%	1.79%	5.80%	9.62%	3.35%	3.48%	9.72%	14.29%	3.83%	9.74%	6.64%	5.47%	5.95%	26.66%	2.90%	7.02%
Weighted Mean	**6.57%**	**1.67%**	5.75%	**9.34%**	**3.26%**	3.41%	**9.59%**	**13.93%**	3.76%	**9.40%**	**6.32%**	**5.29%**	**5.76%**	**26.31%**	2.91%	**6.68%**
Weighted Median	6.69%	1.80%	5.67%	9.49%	3.31%	3.44%	9.79%	14.16%	3.81%	9.72%	6.56%	5.48%	5.89%	26.56%	2.88%	7.06%
Histogram	6.71%	1.79%	5.47%	9.71%	3.28%	**3.37%**	9.75%	14.16%	**3.66%**	9.69%	6.94%	5.73%	6.03%	26.34%	2.96%	7.14%
Edge Guided	6.80%	1.86%	6.00%	9.73%	3.40%	3.52%	9.76%	14.31%	3.88%	9.81%	6.78%	5.60%	5.98%	26.69%	2.94%	7.00%

Table 5.5 Model-Based Fusion with Error Threshold of 2

5.2.1 Mode

For determining image pixel values, mode is among the least consistent for establishing a consensus. The values produced by each algorithm result are extremely specific and there are a relatively small number of algorithm results compared here. This makes the likelihood of agreement extremely low, as each algorithm can take any number of values.

In order to reach an agreement, values were rounded to the nearest quarter. This was determined to be enough to reach a consensus while limiting the introduction of artificial error. Increasing the order of magnitude or the number of results would increase the likelihood of consensus and therefore allow a smaller degree of rounding. This still is not especially useful from an application point-of-view since it would require significantly more pre-processing.

It can be concluded that mode is unlikely to improve troublesome areas since it is a majority voting method and many algorithms will fall short in the same areas but to a slightly different degree. The poor consistency is likely explained by the lack of finding matching values. Consequently, the minimum value was accepted to be the mode by default, which does not offer any benefits.

Interestingly, mode performs exceptionally well for two images, “Teddy” and “ArtL.” This is possibly coincidental, and these are definite outliers when comparing other algorithms and methods. It may be worth investigating these cases further, but it should be noted that these were two images mixed in from an older dataset. They used a less reliable ground truthing mechanism and are comparably smaller in size.

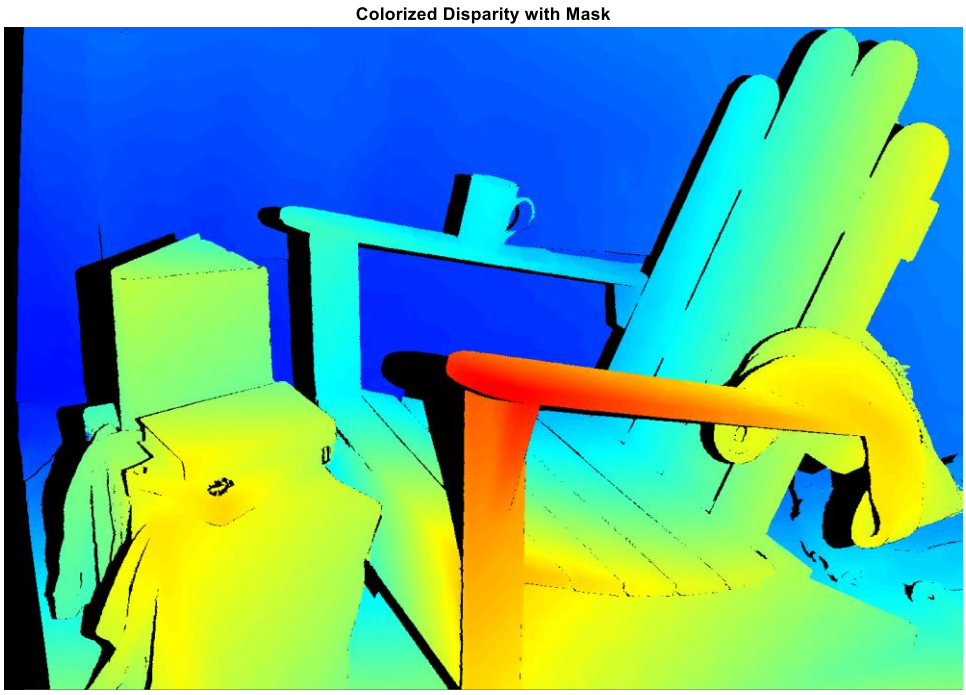

Figure 5.6 Mode Disparity

The residual error in Figure 5.7 shows a mix of misinterpreted object boundaries as well as some shifting on object surfaces. This further illustrates that mode does not perform well on troublesome areas because it needs at least a small level of agreement to make a decision.

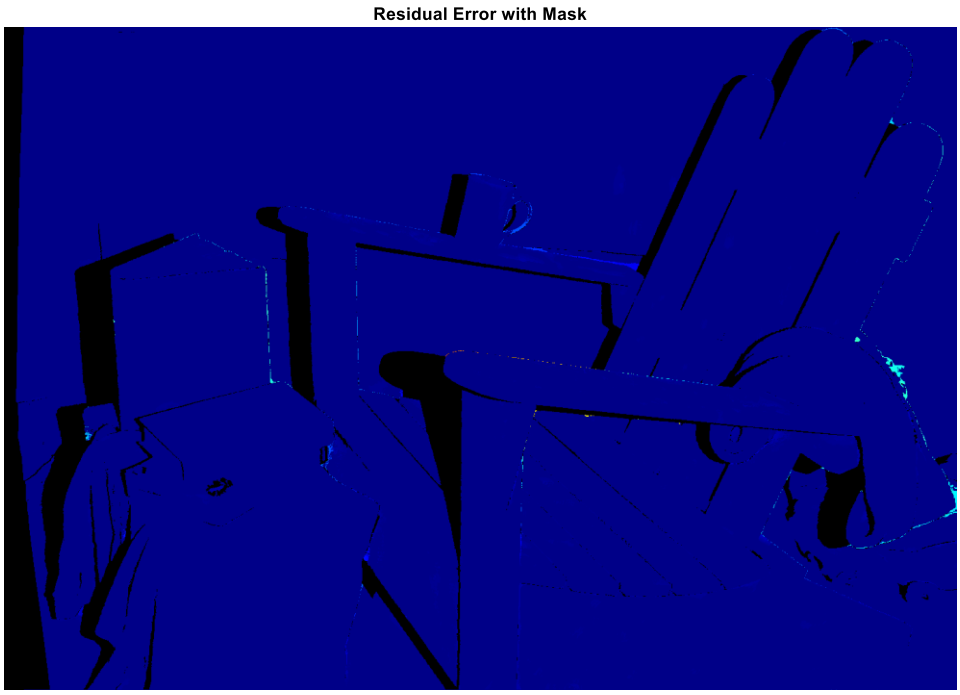

Figure 5.7 Mode Disparity Residual Error

5.2.2 Mean

Mean performs poorly as a method to interpolate new disparity values from the data. It is consistently the lowest performing fusion method. It likely falls short due to outliers causing skewness. Mean alone has no way of discounting outliers. It is worth noting that, although overall performance is comparably poor, this is one of the few disparities that provide a fully connected handle on the coffee mug.

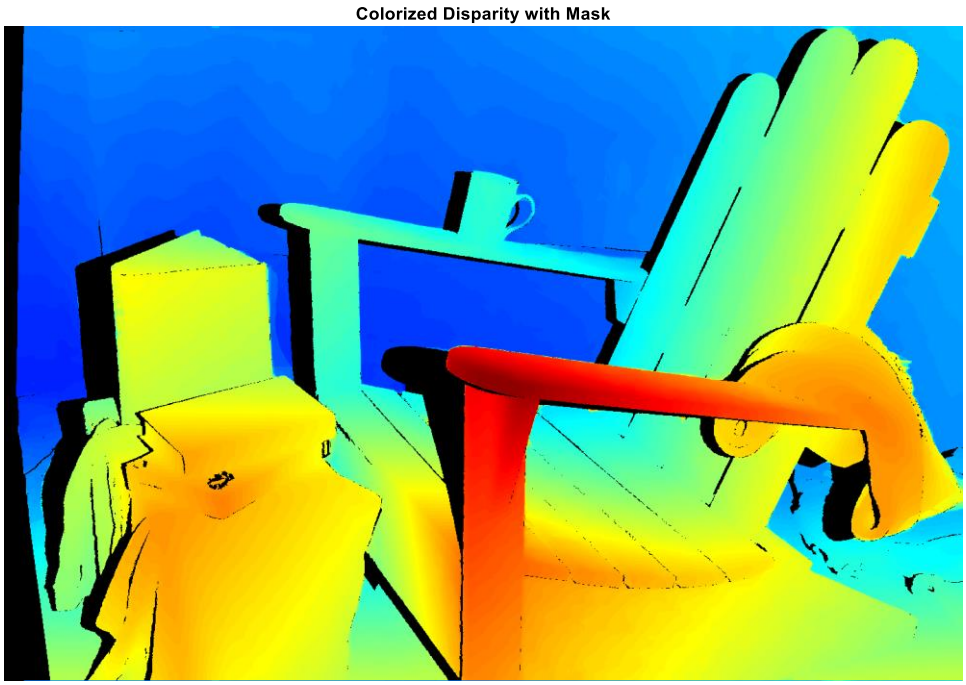

Figure 5.8 Mean Disparity

The residual error of the mean is unique, as it shows no extreme points of error, but rather, lower and smoother error regions. This seems to make the image more visually appealing in some cases such as the previously mentioned, finer details of the mug. The visual appeal of this method is due to the bimodal distribution near object boundaries. One peak represents the foreground object and the other represents the background. Taking the mean of these two peaks creates a smooth transition gradient and minimizes the absolute error in these transition regions, but unfortunately stretches these uncertain regions, creating more error overall. Therefore, Figure 5.9 does not show very many, if any, points of extreme error.

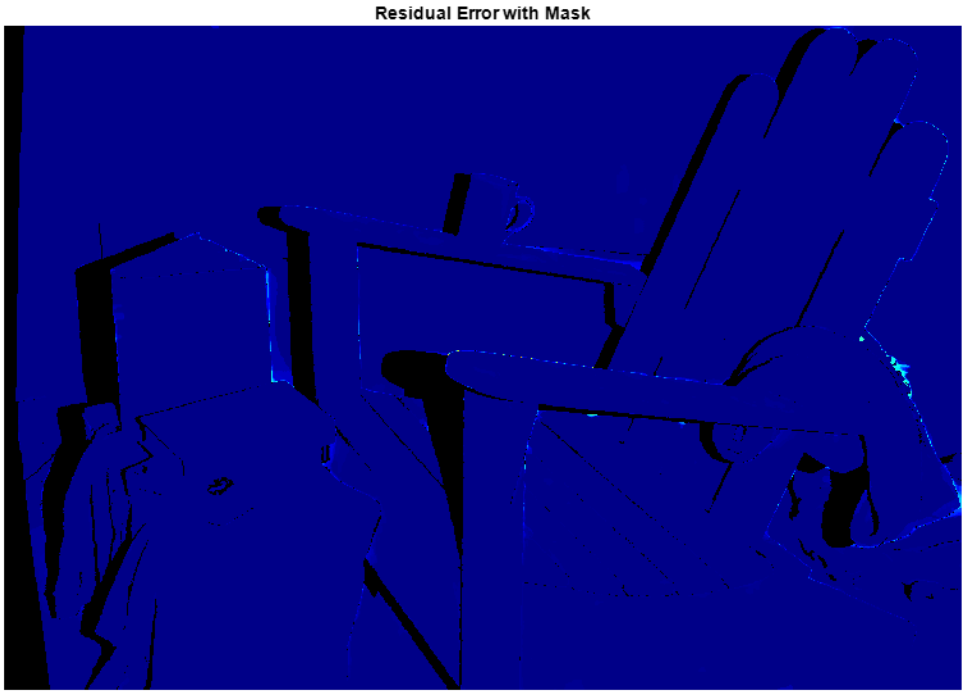

Figure 5.9 Mean Disparity Residual Error

5.2.3 Median

Median is somewhat robust to outliers and therefore displays the best results out of the three most basic statistical methods. It works well to find a central grouping of values and outliers often cause minimal shifting.

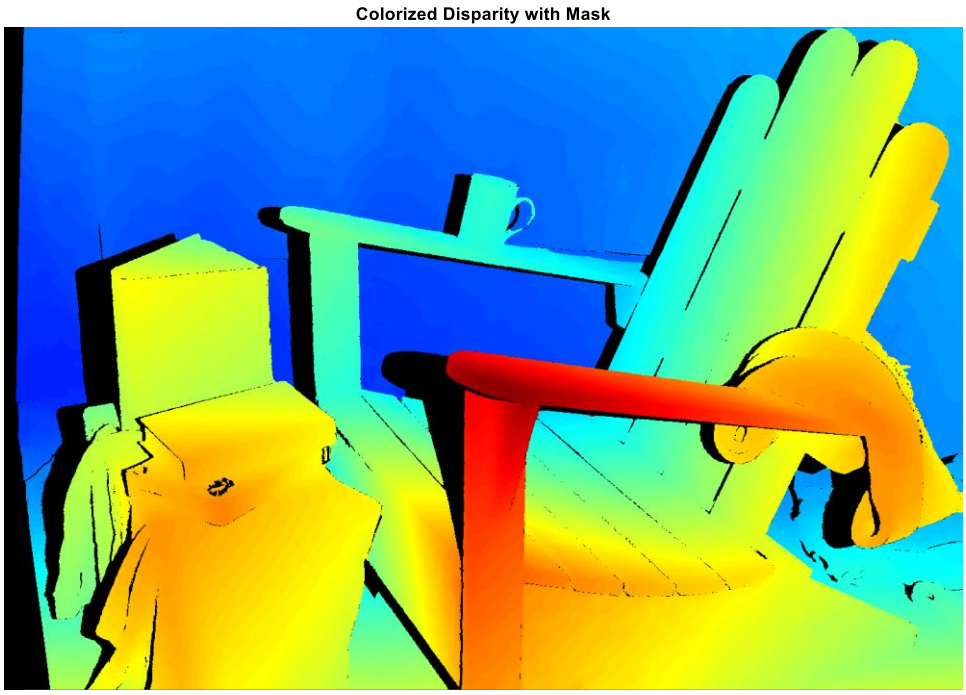

Figure 5.10 Median Disparity

Most residual error is narrowly concentrated along object boundaries. This is discernably different from the residual error of the mean disparity, which appears hazier and extends away from boundaries.

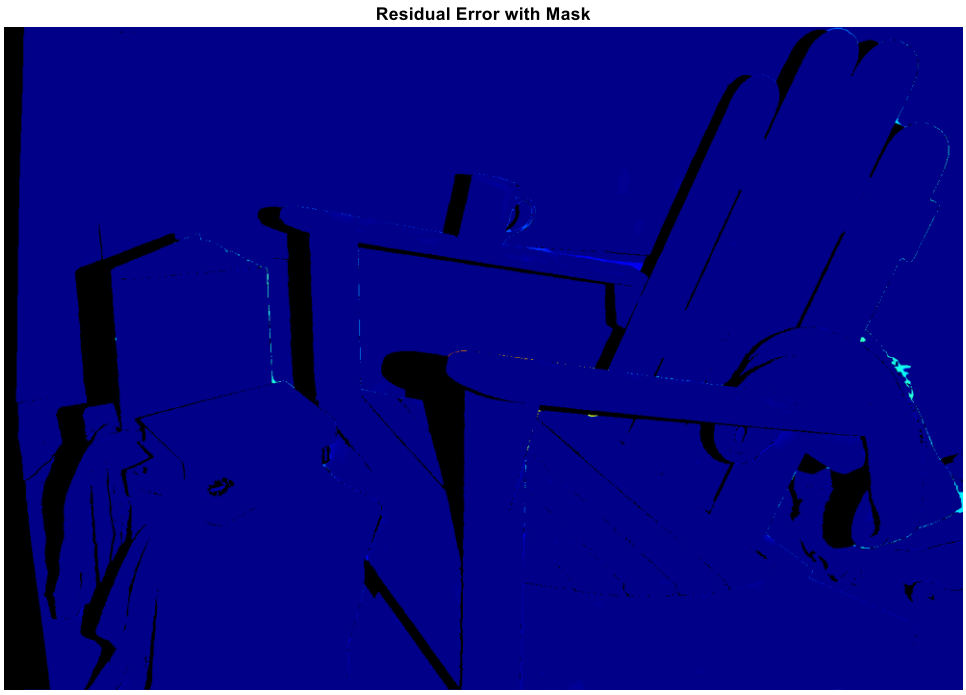

Figure 5.11 Median Disparity Residual Error

5.2.4 Weighted Mean

After examining the results of the mode, mean, and median approaches, it was determined that median would be the best surrogate for the ground truth to determine weights for each algorithm result. A surrogate is necessary since the ground truth would not be available in a real-world situation. Instead the ground truth is approximated as closely as possible.

To further increase the reliability of the approximated ground truth, an exhaustive computation was performed across a list of all algorithm combinations to find which combination yielded the lowest, weighted average error. This was computed using median for each error threshold with top performing results shown in Table 5.6. A combination of five algorithms was found to provide the lowest error; these algorithms were as follows: 3DMST, APAP-Stereo, LW-CNN, MeshStereoExt, and JMR.

	Weighted															
	Average'	Adirondack'	ArtL'	Jadeplant'	Motorcycle'	MotorcycleE'	Piano'	PianoL'	Pipes'	Playroom'	Playtable'	PlaytableP'	Recycle'	Shelves'	Teddy'	Vintage'
Error Threshold 0.5	32.61%	18.49%	43.04%	34.44%	34.45%	24.84%	31.39%	42.36%	23.50%	43.21%	26.57%	22.74%	30.30%	55.61%	42.81%	35.50%
Error Threshold 2	6.00%	1.42%	4.64%	9.23%	3.10%	3.30%	8.10%	12.92%	3.76%	8.02%	5.42%	4.88%	5.58%	23.67%	2.80%	6.37%

Table 5.6 Lowest Weighted Average from Exhaustive Search of All Algorithm Combinations

Since median offers some robustness to outliers, this trait now carries over to the determined weights. This allows mean to operate with less emphasis on extremities by normalizing the distribution with weights. The result is a new value closer to median which utilizes information from all algorithm values.

Weighted mean has the lowest overall weighted error and highest performance consistency of all model-based fusion methods, excluding mode, which has previously been discussed and discounted for its outliers. The error is also notably lower than any individual algorithm.

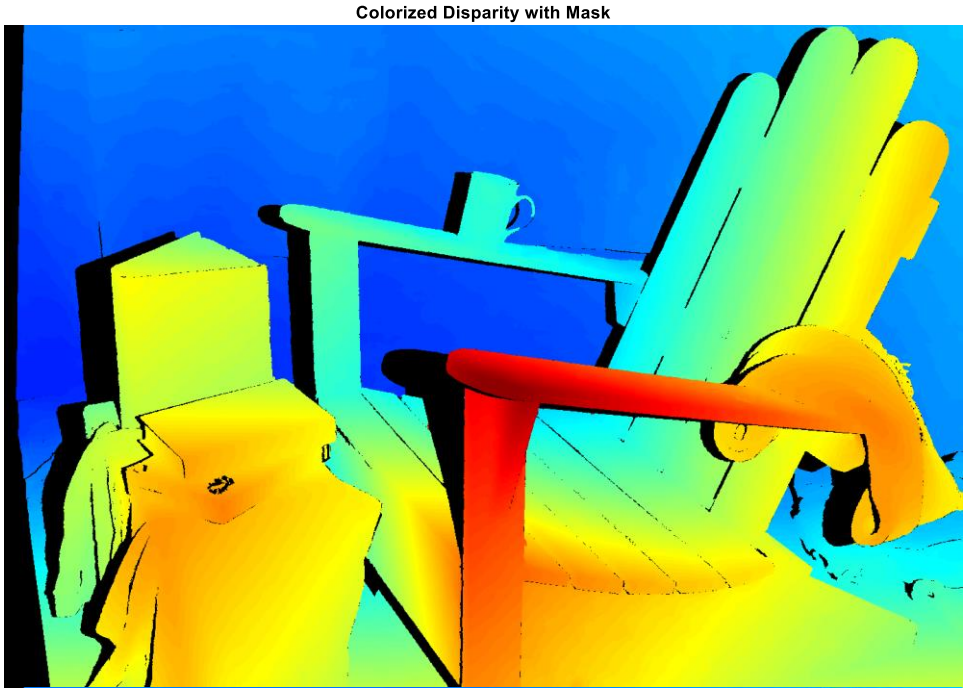

Figure 5.12 Weighted Mean Disparity

There is slightly less residual error seen near the edge boundaries of foreground objects when compared to other algorithms. This may be key to its high scoring nature. Incorrect assessment of foreground and background elements accounts for most of the error.

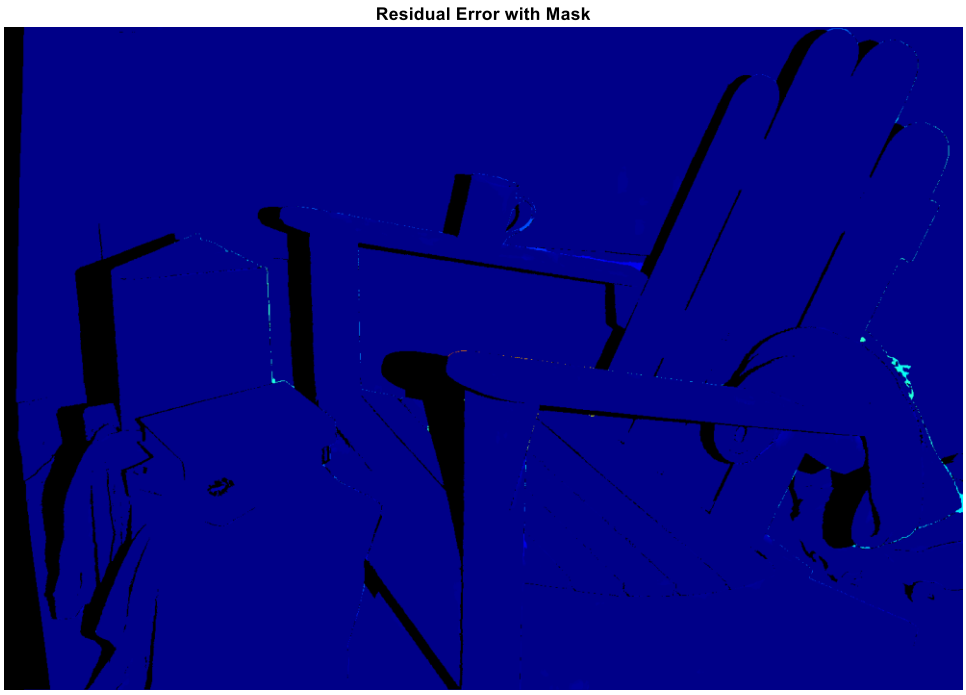

Figure 5.13 Weighted Mean Residual Error

5.2.5 Weighted Median

Weighted median exhibits similar outcomes in comparison to weighted mean. The main difference is the weighted median is choosing an existing algorithm value for most scenarios. Occasionally, it is taking the mean of two algorithm values to interpolate a new value. Weighted median inherits the same outlier robustness from the assigned weights as weighted mean. However, this is redundant in the sense that weighted median is an extension of median. This additional check for outliers could cause skewness in data that is already relatively clean.

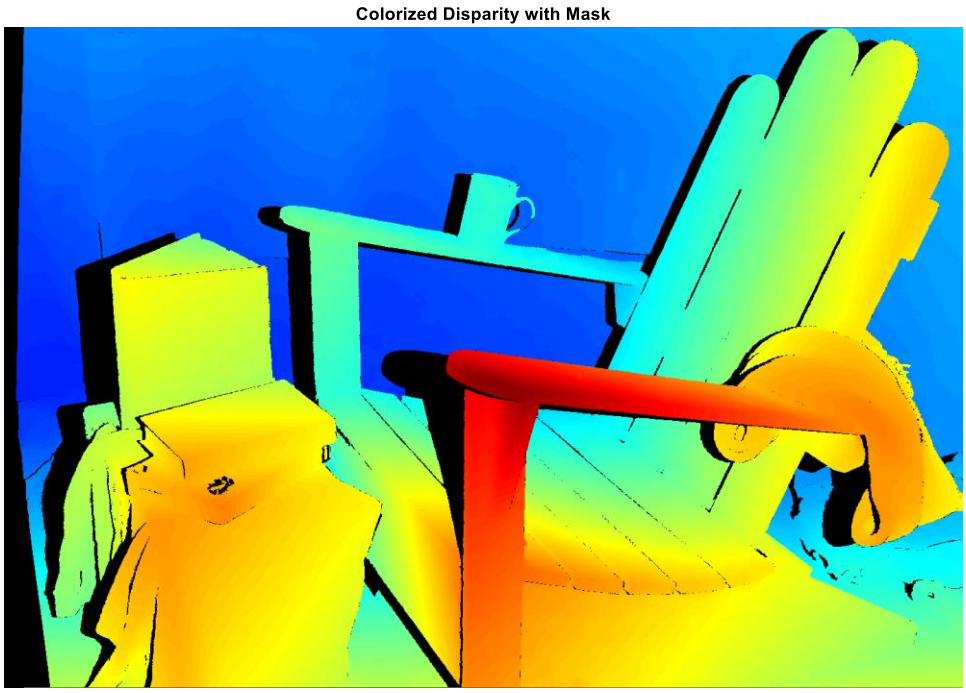

Figure 5.14 Weighted Median Disparity

The resulting residual error of weighted median does not differ significantly from weighted mean. There is a slight increase in residual error near the right side of Figure 5.15, but the two methods are otherwise consistent in error intensity and placement.

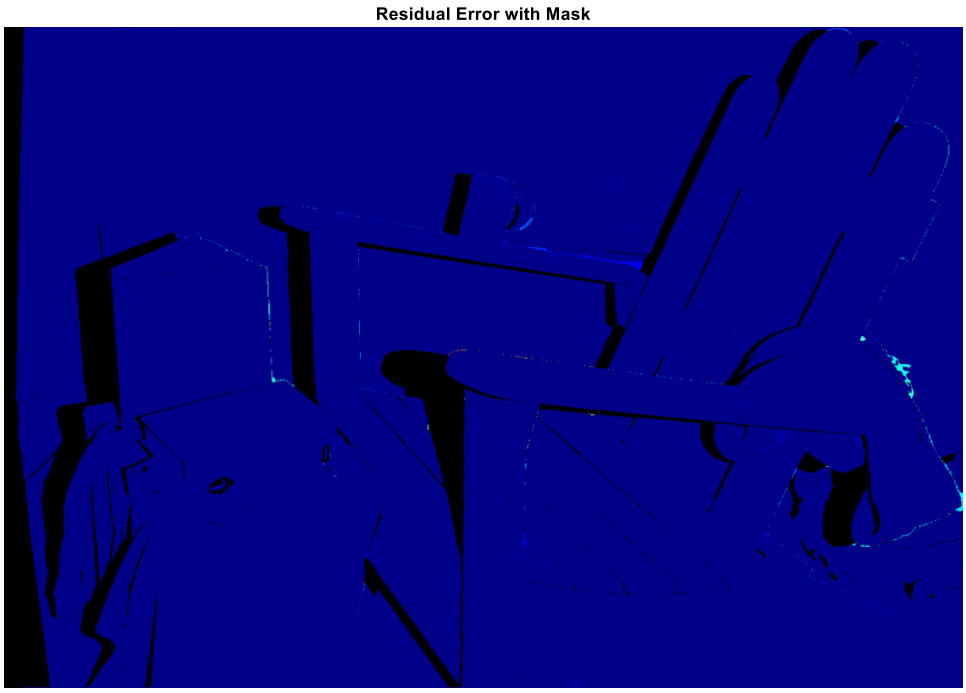

Figure 5.15 Weighted Median Residual Error

5.2.6 Histogram

Histograms do not have an ideal number of bins or bin size since the spread of datapoints can vary greatly. It is possible for the boundary to evenly divide a small cluster of points, in which case useful information may be lost since only one bin is chosen. If the bins are too small, there may not be enough points that fall within any bin to make it the largest. Large bins run the risk of capturing too broad of a range of points and watering down the precision of tight clusters.

A bin size of three was determined to be the best size to minimize average weighted error. The mean of the values in the first, largest bin was taken to decide the disparity value for each pixel. This means the background was favored in circumstances of arbitrary object boundaries. The range within a bin is small enough to disallow substantial deviations, so mean was deemed acceptable to represent an overall value for each bin.

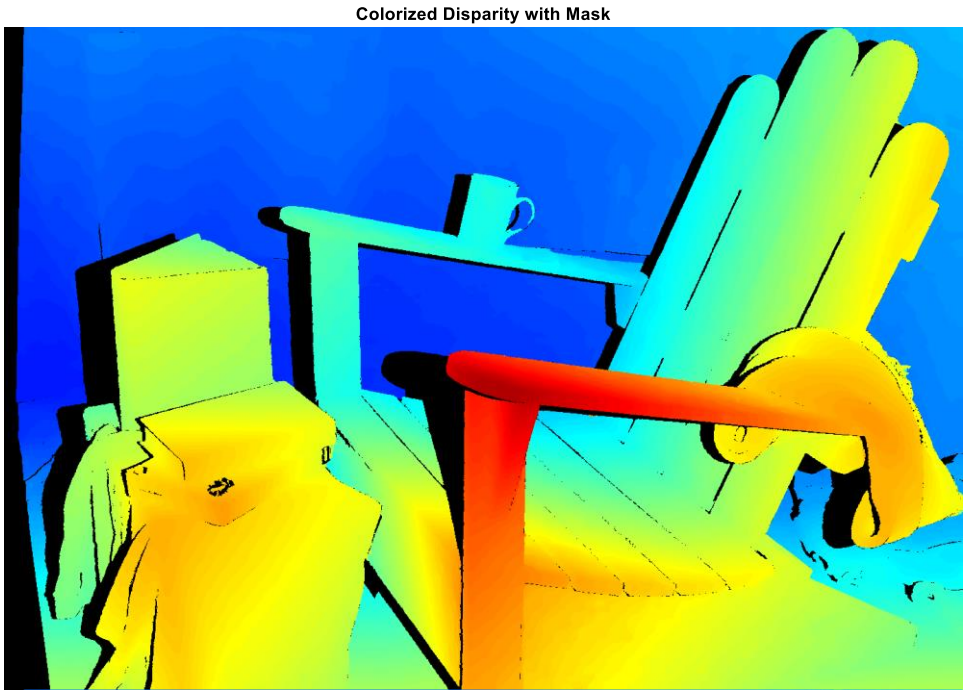

Figure 5.16 Histogram Disparity

Histogram-based fusion produced very similar results to median. This is to be expected, as both methods have a similar level of robustness to outliers. However, there is slightly more visible residual error in the mid-ground for histogram-based results and this is a more expensive task than simply taking the median.

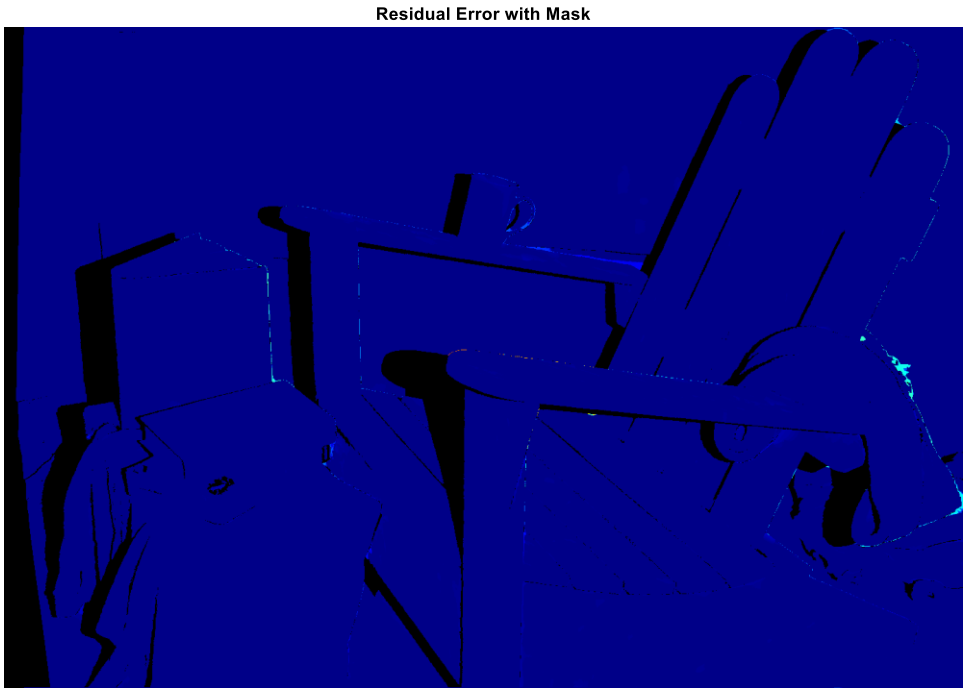

Figure 5.17 Histogram Residual Error

5.2.7 Edge-Guided

Edge-guided results are not significantly different from the median results that they are meant to improve upon. This is likely due to detected edges introducing more error than they clean up. When an incorrect edge is detected, it then mistakenly uses a poor combination of algorithms to select a value along that edge. These isolated cases can be smoothed out with a median filter to appear visually better, but error is still introduced.

Sobel edge detection was chosen over Canny due to the amount of background noise being picked up and misinterpreted as proper boundaries. The basis of this technique is demonstrated in Figure 5.18, comparing the overlap of ground truth edges with edges from 3DMST algorithm results.

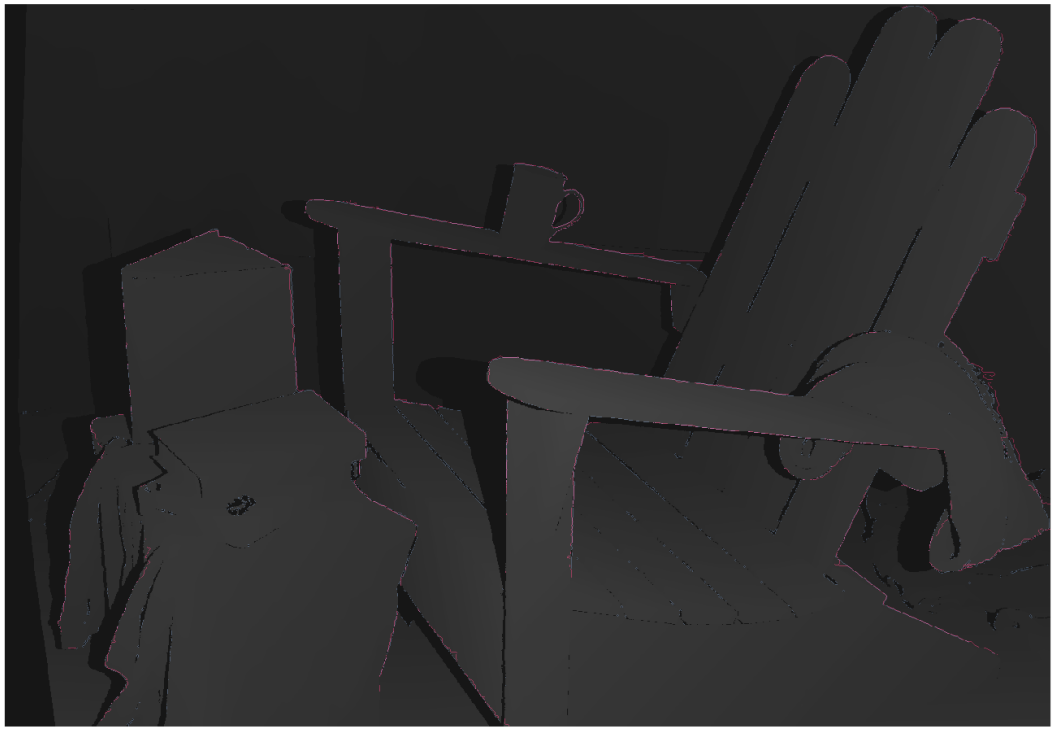

Figure 5.18 Ground Truth Edges (Blue) and 3DMST Edges (Red)

The ground truth edges appeared to line up well with the edges from the 3DMST algorithm result and most important edges were detected. In practice, however, using the RGB image as the ground truth for edges is very different and resulted in poor edge detection performance. For an RGB image, edges were detected mostly in textured regions and were not picked up on some more desirable object boundaries. Figure 5.19 demonstrates these shortcomings.

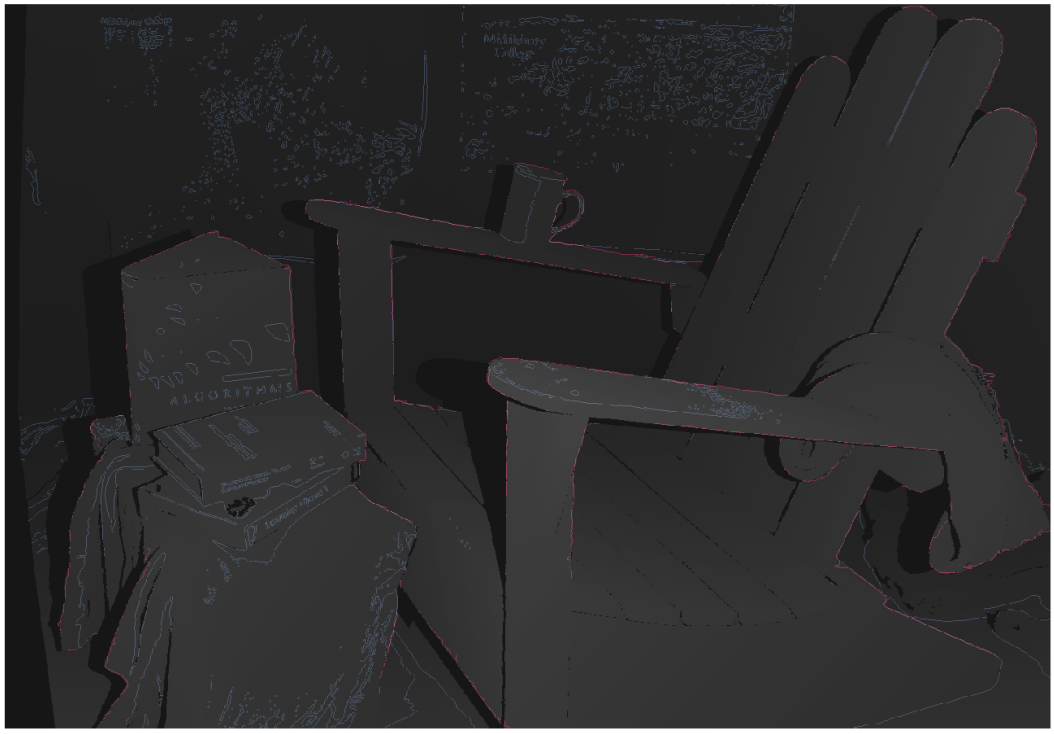

Figure 5.19 RGB Image Edges (Blue) and 3DMST Edges (Red)

A median filter was necessary to remove some extreme outliers on separation boundaries, yet inconsistencies along certain edges can still be seen in Figure 5.20. This can also be seen in Figure 5.21, which appears slightly hazy in some areas due to the error introduced during the filtering step.

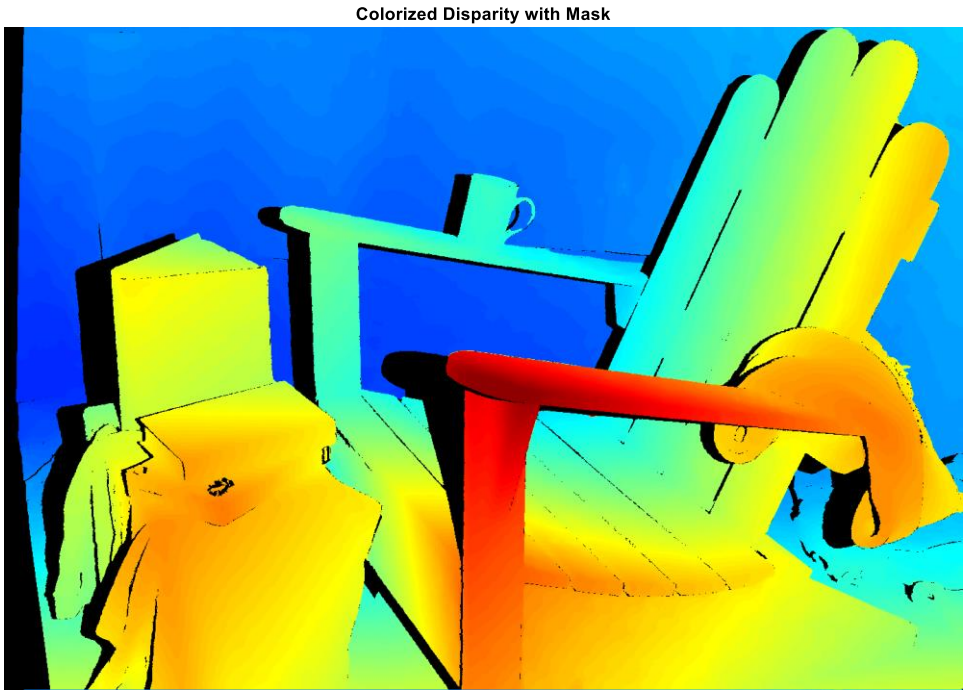

Figure 5.20 Edge-Guided Disparity

Slightly less extreme points of error are seen in Figure 5.21when compared to median residual error in Figure 5.14. This demonstrates that taking the mean around the edges can be effective at reducing error on object boundaries. Though, this is not a perfect implementation due to edges not always being correctly detected or overlapping as expected.

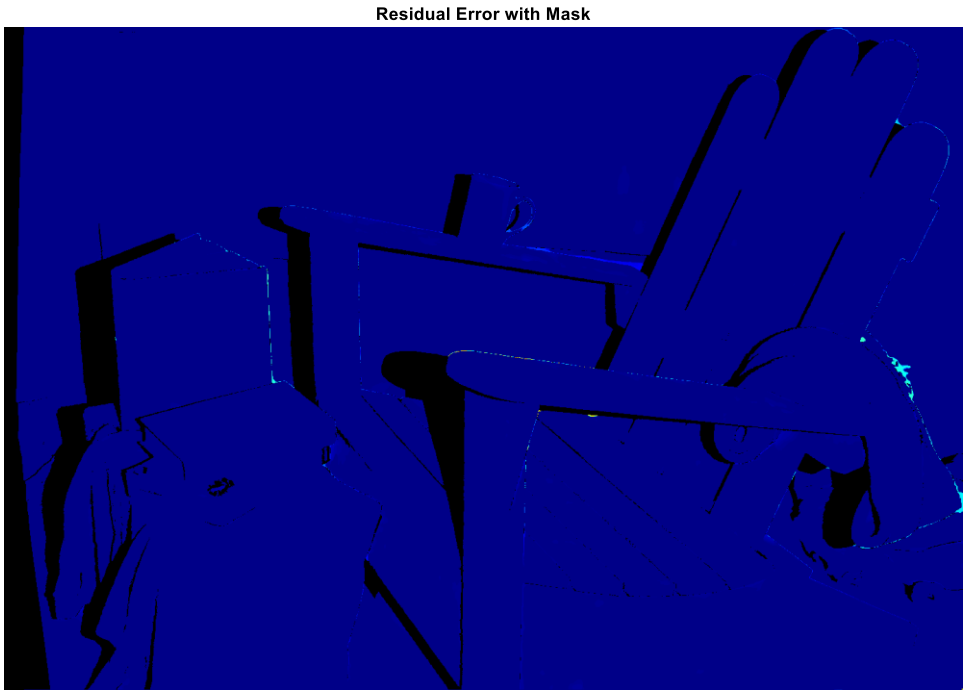

Figure 5.21 Edge-Guided Residual Error

5.3 Learning-Based

Approximately 150,000 random patches were sampled from the “Vintage” image to be used as training data. The actual number of patches was slightly smaller after removing noisy areas that fell within the image mask. This training image was selected due to its large range and high variance to ensure a robust set of features since the amount of data was limited.

A summary of all learning-based results are presented in **Error! Reference source not found.** and **Error! Reference source not found.**, below. The SVM model is essentially useless, considering its weighted average is at best ~97% and ~89% accurate, respectively, for error thresholds of 0.5 and 2. It is unable to mark any improvement over the individual algorithm results from its input. While it still provides a coherent disparity result, it is unable achieve an acceptable result even on its training data.

The bagged tree model performs well compared to the SVM model. It is inferred that bagged trees show favorable results as a method to reduce error from a pool of erroneous algorithm

results. Bagged trees are able to establish a path to accurate results on the training image "Vintage." This scales reasonably well to the other images in the dataset when considering the ratio of training to testing data. More training data is likely to provide more scalable results.

	Weighted Average	Adirondack	ArtL	Jadeplant	Motorcycle	MotorcycleE	Piano	PianoL	Pipes	Playroom	Playtable	PlaytableP	Recycle	Shelves	Teddy	Vintage*
Weights		8%	8%	8%	8%	8%	8%	4%	8%	4%	4%	8%	8%	4%	8%	4%
SVM	97.27%	96.86%	99.17%	96.76%	97.33%	98.61%	97.71%	98.01%	97.76%	97.53%	98.18%	98.24%	95.57%	95.16%	95.02%	96.81%
BaggedTree	**63.12%**	**62.04%**	**68.42%**	**69.76%**	**65.22%**	**59.09%**	**67.71%**	**72.18%**	**60.27%**	**66.54%**	**62.37%**	**58.44%**	**61.91%**	**70.30%**	**65.83%**	**29.16%**

Table 5.7 Learning-Based Fusion with Error Threshold of 0.5

	Weighted Average	Adirondack	ArtL	Jadeplant	Motorcycle	MotorcycleE	Piano	PianoL	Pipes	Playroom	Playtable	PlaytableP	Recycle	Shelves	Teddy	Vintage*
Weights		8%	8%	8%	8%	8%	8%	4%	8%	4%	4%	8%	8%	4%	8%	4%
SVM	88.91%	87.72%	95.13%	88.85%	89.32%	91.25%	87.27%	87.56%	88.97%	89.29%	91.56%	90.93%	85.82%	83.26%	86.40%	87.69%
BaggedTree	**17.14%**	**22.23%**	**11.81%**	**31.45%**	**13.44%**	**14.91%**	**18.59%**	**23.89%**	**15.70%**	**22.75%**	**15.92%**	**13.08%**	**12.37%**	**34.89%**	**10.11%**	**3.72%**

Table 5.8 Learning-Based Fusion with Error Threshold of 2

5.3.1 Support Vector Machines

Support Vector Regression displays poor performance. Many object boundaries are estimated to be farther into the foreground than the object itself. Some of this error may be explained because of the lack of training data. While parts of Figure 5.22 appear to be visually consistent, it is clear that the model did not generalize as well.

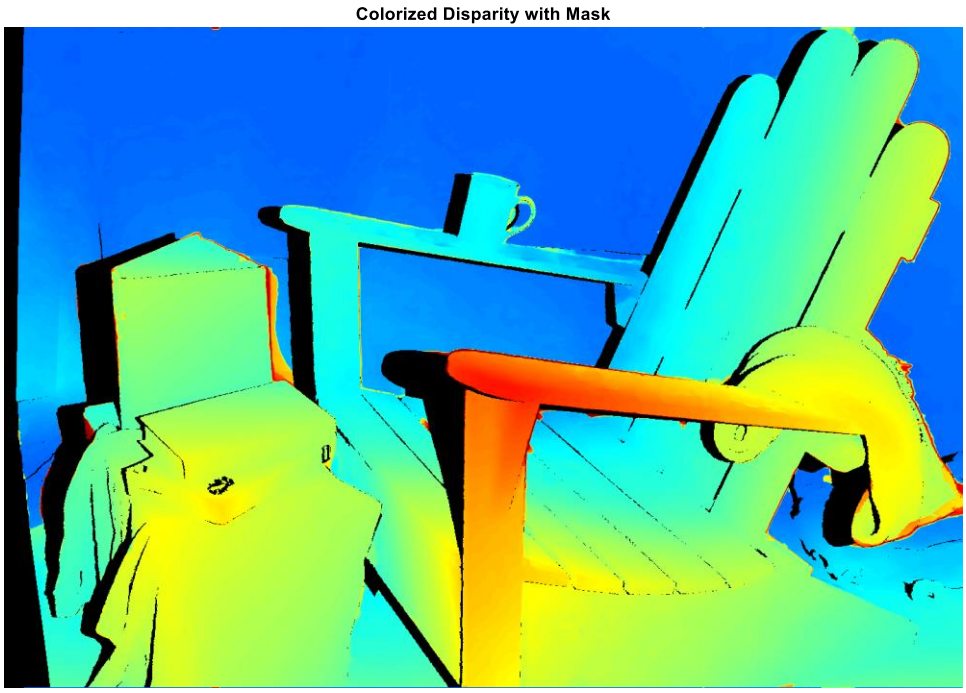

Figure 5.22 SVR Disparity Map

The residual error in Figure 5.23 shows that any object boundary with a reasonably large disparity is extremely misinterpreted. Again, these are finer details that could not be captured and generalized with the limited training set. Object surfaces, although appearing uniform, are also misestimated to a lesser degree. This is most apparent in the background where these values stray further from the ground truth.

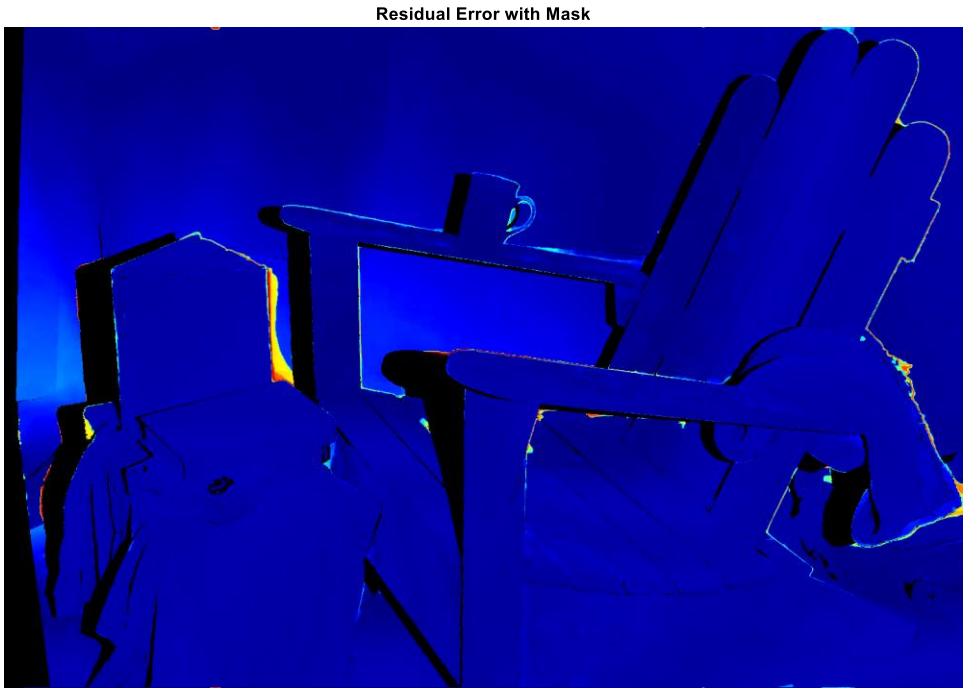

Figure 5.23 SVR Residual Error

5.3.2 Bagged Trees

Bagged trees did not show very promising results when the model was applied to test images. Performance was greatly improved compared the SVM model, but there are still clear problems around object boundaries.

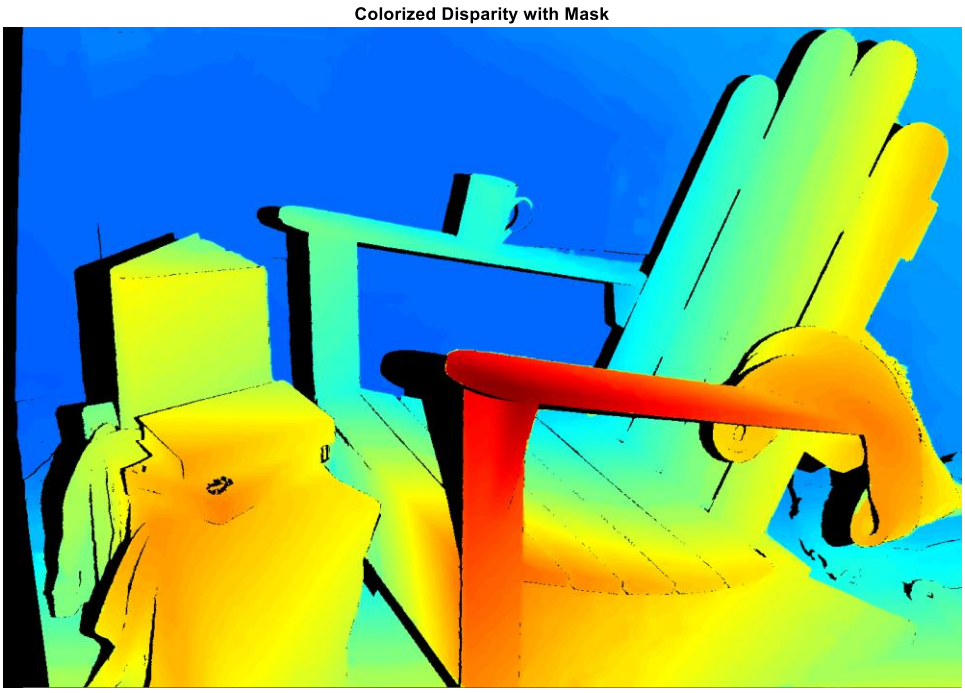

Figure 5.24 Bagged Tree Disparity

More error can be seen along object boundaries when compared with previous model-based methods. Much of the background is also skewed by some amount and appears as a light haze in Figure 5.25

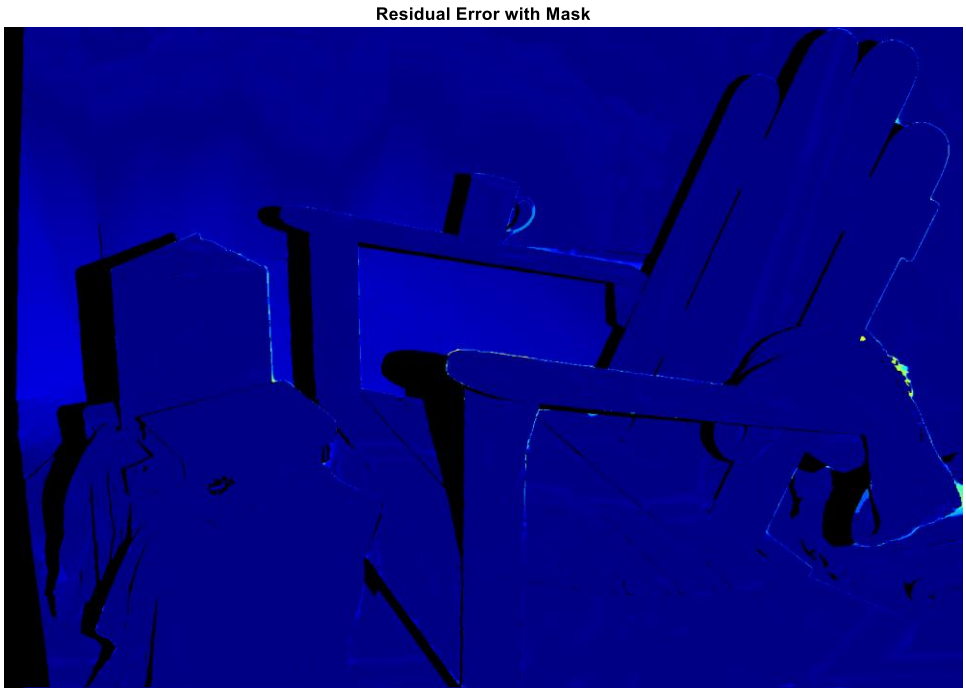

Figure 5.25 Bagged Tree Residual Error

The bagged tree model suffers from the same incorrect shift of background values. Object surfaces appear to have less overall error, but they exhibit less uniformity than expected. Stripes can be seen along the Adirondack surface in Figure 5.25, as these specific depths were not correctly learned.

6 Conclusions

This work presented a balance of different methods to fuse several erroneous disparity maps into a more accurate representation. Several statistical methods achieved this goal by producing maps with lower error on a per image basis, or in most cases, a lower overall weighted average error. It is shown that median, weighted mean, weighted median, and histogram approaches are all sufficient at the task of removing outliers in the algorithm results. Edge-guided results also fared well. Mode and mean did not offer any improvements and were found to be regressive in most cases.

Learning-based approaches compared in this work suffered from limited data. This was required in order to provide a fair comparison. Support vector regression was not able to infer accurate values even on the image it was trained on. Bagged trees showed potential and scaled relatively well compared to support vector regression. The bagged tree model was able to learn strong predictors on the training image. It would seem that with a more inclusive range of data, bagged trees could perform well across new images. Despite generally poor performance, the results for both learning-based models are still visually consistent. This demonstrates that they have a sense of linearity but with poor bias. More data is needed to create a successful model.

When looking at the results of limited algorithm combinations discussed in Section 5.2.4, it becomes clear that the correct balance of algorithms is essential and more algorithms is not necessarily better. Comparing the results of weighted mean and weighted median in Table 5.4 and Table 5.5 with Table 5.6 shows that no improvement was found by applying these methods. The median of five algorithms in Table 5.6 provides the best fusion.

Other considerations in this process have been explored as well. It may have been better to choose initial algorithms based on their average weighted error results across all four error thresholds as opposed to just one. The best scoring algorithms do not remain consistent across each error threshold due to their unique performance imbalances as discussed near the end of Section 5.1.2.

It may also have helped to select algorithms if they performed exceptionally well for one case. These approaches could have warranted a wider range of performances under different circumstances. This would build on the idea that an intelligent selector would be able to choose a high performing algorithm, or combination of algorithms, for any given case.

6.1 Limitations

The majority of algorithms suffer from poor context, or lack thereof, when determining proper decision boundaries between foreground and background objects. They fall short in their ability to recognize and segment objects at a level near human capability. This may be considered a chicken-and-egg problem in the sense that proper segmentation may be extremely useful to achieve fully accurate disparity mapping under all conditions; but robust, accurate disparity mapping could help to achieve much better object segmentation. The combination of segmentation and disparity has been demonstrated in [10] with promising results. This is a slightly more complex version of the edge-guided fusion introduced in Section 3.3. Both segmentation accuracy and edge detection accuracy limit how accurate a resulting disparity fusion can be near object boundaries.

The use of all proposed methods is dependent on the results of several other algorithms often doing redundant work. This is not feasible for real-world applications that often need real-time computations in order to serve their overarching purpose. Therefore, approaches demonstrated in this work are unfortunately limited to applications that are not time sensitive. There is currently a large trade-off between computation time and the gains in consistency and accuracy. While this trade-off restricts this work to a post-processing step, it does not make its results any less valid and it retains several use cases.

Model-based methods may be limited in some regards. Simple statistical methods follow very basic rules for every pixel location which does not offer flexibility. Mode is a primary example of not reaching its full potential. When a consensus cannot be reached, the first result in a sorted array is preferred. This favors the smallest disparity or background location.

Learning methods are limited due to the amount of training data available. There are not enough images to demonstrate the power of a neural network. Even results of simple regression learners do not translate well to new scenarios. There is also a limit on the amount of processing power and time it takes to train full images. Constructing half resolution images using a trained model can currently take tens of hours with parallel processing.

Bagged trees appeared to show signs of overtraining on the limited data. Using a selection of pixels over different images would certainly be more insightful. This was not deemed useful in this work a few reasons. The training time would have been too high, and the overall weighted average error would not have been valid since it includes trained data. It also makes it difficult to draw comparisons between other methods. For bagged trees to be investigated further, outside training data is necessary.

6.2 Future Work

This work demonstrates the strong points of a variety of algorithms, showing that together they can be more robust in their overall accuracy than any individual algorithm. As mentioned, this is not practical in an application sense. It has been shown that very good results can be realized using the combination of five algorithms. Future exploration has several imaginable areas of research to lead to improved disparity mapping. The first may be further limiting the number of algorithms required to create a new map while maintaining the overall level of robustness that has been demonstrated. It may be possible to find two algorithms with completely opposite strengths and weaknesses that do not achieve strong results independently.

It is likely possible to find a more intelligent method of correctly determining and selecting algorithm strengths and resolving object boundaries. A strong segmentation network may be able to provide a correction mechanism capable of producing visually better and nearly equivalent results with minimal dependence on the starting algorithm. This is one conceivable way to create a correction filter that could be further integrated if successful.

To further the idea of intelligent selection, there is likely more to be done with simple statistical approaches. From the results discussed in Section 3.1, there were apparent strengths and

weaknesses for mean, median, and mode. Analyzing the distribution peaks before selecting one of these methods may be beneficial. Mean appears to be the best solution for evenly distributed peaks where the object boundary is arbitrary, while median works well overall for most object surfaces. Mode could also be made to be more robust. If there is no consensus, defaulting to the previously mentioned approach could yield much lower error.

The opportunities for learning-based algorithms are still plentiful. Given sufficient data, insight into the cruxes of depth approximation may be identified and avoided. Isolating reflective textures and refining them as smoothed objects is an example of an issue that requires a higher level of insight.

Bibliography

[1] N.A., "LPU," 3 July 2016. [Online]. Available: http://vision.middlebury.edu/stereo/eval3/. [Accessed 8 November 2017].

[2] N.A., "Stereo Depth Map Refinement with Scene Layout Estimation," 21 January 2016. [Online]. Available: http://vision.middlebury.edu/stereo/eval3/. [Accessed 13 November 2017].

[3] M. Park and K. Yoon, "As-Planar-As-Possible Depth Map Estimation," 28 May 2016. [Online]. Available: http://vision.middlebury.edu/stereo/eval3/. [Accessed 8 November 2017].

[4] A. Davies, "Wired," 25 September 2014. [Online]. Available: http://velodynelidar.com/docs/news/This%20Palm-Sized%20Laser%20Could%20Make%20Self-Driving%20Cars%20Way%20Cheaper%20_%20WIRED.pdf. [Accessed 6 Novermber 2017].

[5] R. Mur-Artal and J. D. Tardos, "ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras," *IEEE Transactions on Robotics,* vol. 33, no. 5, pp. 1255-1267, 2017.

[6] J. Engel, J. Stückler and D. Cremers, "Large-Scale Direct SLAM with Stereo Cameras," in *IEEE/IRS International Conference on Intelligent Robots and Systems (IROS)*, Hamburg, Germany, 2015.

[7] S. B. Goldberg, M. W. Maimone and L. Matthies, "Stereo Vision and Rover Navigation Software for Planetary Exploration," in *IEEE Aerospace Conference Proceedings*, Big Sky, 2002.

[8] K. Schmid, T. Tomic, F. Ruess, H. Hirschmuller and M. Suppa, "Stereo Vision based indoor/outdoor Navigation for Flying Robots," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Tokyo, Japan, 2013.

[9] Y. A. Ivanov, A. P. Pentland and C. R. Wren, "Computer vision depth segmentation using virtual surface". United States Patent US6911995B2, 17 8 2001.

[10] F. Güney and A. Geiger, "Displets: Resolving Stereo Ambiguities Using Object Knowledge," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Boston, MA, 2015.

[11] Y. Sumi, Y. Kawai, T. Yoshimi and F. Tomita, "3D Object Recognition in Cluttered Environments by Segment-Based Stereo Vision," *International Journal of Computer Vision,* vol. 46, no. 1, pp. 5-23, 2002.

[12] M. Meingast, C. Geyer and S. Sastry, "Vision Based Terrain Recovery for Landing Unmanned Aerial Vehicles," in *IEEE Conference on Decision and Control (CDC)*, Nassau, 2004.

[13] S. T. Barnard and M. A. Fischler, "Computational Stereo," *ACM Computing Surveys,* vol. 14, no. 4, pp. 553-572, December 1982.

[14] "OpenCV: Epipolar Geometry," OpenCV, 23 December 2016. [Online]. Available: https://docs.opencv.org/3.2.0/da/de9/tutorial_py_epipolar_geometry.html. [Accessed 7 February 2018].

[15] X. Chai, F. Zhou and X. Chen, "Epipolar constraint of single-camera mirror binocular stereo vision systems," *Optical Engineering,* vol. 56, no. 8, 2017.

[16] G. Bradski and A. Kaehler, Learning OpenCV: Computer Vision with the OpenCV Library, Sebastopol, CA: O'Reilly Media, Inc., 2008.

[17] "Uncalibrated Stereo Image Rectification," Mathworks, 2018. [Online]. Available: https://www.mathworks.com/help/vision/examples/uncalibrated-stereo-image-rectification.html. [Accessed 16 February 2018].

[18] E. Karami, S. Prasad and M. Shehata, "Image Matching Using SIFT, SURF, BRIEF and ORB: Performance Comparison for Distorted Images," in *Newfoundland Electrical and Computer Engineering Conference*, St. Johns, Canada, 2015.

[19] L. Li, S. Zhang, X. Yu and L. Zhang, "PMSC: PatchMatch-Based Superpixel Cut for Accurate Stereo Matching," *IEEE Transactions on Circuits and Systems for Video Technology,* no. 99, 2016.

[20] H. Park and K. M. Lee, "Look Wider to Match Image Patches With Convolutional Neural Networks," *IEEE Signal Processing Letters,* vol. 24, no. 12, pp. 1788-1792, 2016.

[21] L. Li, X. Yu, S. Zhang, X. Zhao and L. Zhang, "3D cost aggregation with multiple minimum spanning trees for stereo matching," *Applied Optics,* vol. 56, no. 12, pp. 3411-3420, 2017.

[22] J. Žbontar and Y. LeCun, "Stereo Matching by Training a Convolutional Neural Network to Compare Image Patches," *Journal of Machine Learning Research,* vol. 17, no. 65, pp. 1-32, 28 August 2016.

[23] H. Hirschmüller and D. Scharstein, "Evaluation of Cost Functions for Stereo Matching," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2007)*, Minneapolis, MN, 2007.

[24] J. Kim, V. Kolmogorov and R. Zabih, "Visual Correspondence Using Energy Minimization and Mutual Information," in *IEEE International Conference on Computer Vision*, Nice, France, 2003.

[25] K.-R. Kim and C.-S. Kim, "Adaptive Smoothness Constraints for Efficient Stereo Matching Using Texture and Edge Information," in *IEEE International Conference on Image Processing (ICIP)*, Phoenix, 2016.

[26] P. Knöbelreiter, C. Reinbacher, A. Shekovtsov and T. Pock, "Cornell University Library," 3 May 2017. [Online]. Available: https://arxiv.org/pdf/1611.10229.pdf. [Accessed 8 November 2017].

[27] C. Zhang, Z. Li, Y. Cheng, R. Cai, H. Chao and Y. Rui, "MeshStereo: A Global Stereo Model with Mesh Alignment Regularization for View Interpolation," in *IEEE International Conference on Computer Vision (ICCV)*, Santiago, 2015.

[28] S. Zhang, W. Xie, G. Zhang, H. Bao and M. Kaess, "Robust Stereo Matching with Surface Normal Prediction," in *IEEE International Conference on Robotics and Automation (ICRA)*, Singapore, 2017.

[29] L. Juliff, "How to Remove People From Your Travel Photos Using Photoshop," 22 August 2013. [Online]. Available: https://toomanyadapters.com/how-to-remove-people-travel-photos-photoshop/. [Accessed May 2018].

[30] N. S. Young, "Remove Tourists from Images Using Photoshop," 24 February 2015. [Online]. Available: https://photofocus.com/2015/02/24/remove-tourists-from-images-using-photoshop/. [Accessed May 2018].

[31] F. Y. Edgeworth, "XXII. On a new method of reducing observations relating to several quantities," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science ,* vol. 25, no. 154, pp. 184-191, 2009.

[32] M. S. Yumlu and F. S. Gurden, "SVR for Time Series Prediction," in *Support Vector Machines: Data Analysis, Machine Learning and Applications*, B. H. Boyle, Ed., Istanbul, Nova Science Publishers, Inc., 2011, pp. 117-130.

[33] O. Chapelle and V. Vapnik, "Model Selection for Support Vector Machines," *Advances in Neural Information Processing Systems,* pp. 230-236, 2000.

[34] P. Paisitkriangkrai, "Linear Regression and Support Vector Regression," 24 October 2012. [Online]. Available: https://cs.adelaide.edu.au/~chhshen/teaching/ML_SVR.pdf. [Accessed April 2018].

[35] C. Shalizi, "Regression Trees," 11 October 2006. [Online]. Available: http://www.stat.cmu.edu/~cshalizi/350-2006/lecture-10.pdf. [Accessed May 2018].

[36] F. Gorunescu, Intelligent Systems Reference Library, vol. 12, J. Kacprzyk and L. C. Jain, Eds., Verlag Berlin Heidelberg: Springer, 2011.

[37] D. Scharstein and C. Pal, "Learning Conditional Random Fields for Stereo," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2007)*, Minneapolis, MN, 2007.

[38] D. Scharstein, H. Hirschmuller, Y. Kitajima, G. Krathwohl, X. Wang and P. Westling, "High-Resolution Stereo Datasets with Subpixel-Accurate Ground Truth," 2014. [Online]. Available: http://www.cs.middlebury.edu/~schar/papers/datasets-gcpr2014.pdf. [Accessed 31 January 2018].

[39] D. Scharstein and R. Szeliski, "High-Accuracy Stereo Depth Maps Using Structured Light," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Madison, WI, 2003.

[40] D. Scharstein, R. Szeliski and H. Hirschmüller, "vision.middlbury.edu," [Online]. Available: vision.middlebury.edu/stereo/. [Accessed November 2017].

[41] H. Hirschmuller and D. Scharstein, "Evaluation of Cost Functions for Stereo Matching," 2007.