Normalized Convolution Network and Dataset Generation for Refining Stereo Disparity Maps

Filip Skarfelt and Daniel Cranston
Master of Science Thesis in Electrical Engineering

Normalized Convolution Network and Dataset Generation for Refining Stereo Disparity Maps:

Filip Skarfelt and Daniel Cranston
LiTH-ISY-EX--19/5252--SE

Supervisor: Felix Järemo-Lawin
ISY, Linköping University
Jimmy Jonsson
Saab Dynamics

Examiner: Prof. Michael Felsberg
ISY, Linköping University

Computer Vision Laboratory
Department of Electrical Engineering
Linköping University
SE-581 83 Linköping, Sweden

Copyright © 2019 Filip Skarfelt and Daniel Cranston
Abstract

Finding disparity maps between stereo images is a well studied topic within computer vision. While both classical and machine learning approaches exist in the literature, they frequently struggle to correctly solve the disparity in regions with low texture, sharp edges or occlusions. Finding approximate solutions to these problem areas is frequently referred to as disparity refinement, and is usually carried out separately after an initial disparity map has been generated.

In the recent literature, the use of Normalized Convolution in Convolutional Neural Networks have shown remarkable results when applied to the task of stereo depth completion. This thesis investigates how well this approach performs in the case of disparity refinement. Specifically, we investigate how well such a method can improve the initial disparity maps generated by the stereo matching algorithm developed at Saab Dynamics using a rectified stereo rig.

To this end, a dataset of ground truth disparity maps was created using equipment at Saab, namely a setup for structured light and the stereo rig cameras. Because the end goal is a dataset fit for training networks, we investigate an approach that allows for efficient creation of significant quantities of dense ground truth disparities.

The method for generating ground truth disparities generates several disparity maps for every scene measured by using several stereo pairs. A densified disparity map is generated by merging the disparity maps from the neighbouring stereo pairs. This resulted in a dataset of 26 scenes and 104 dense and accurate disparity maps.

Our evaluation results show that the chosen Normalized Convolution Network based method can be adapted for disparity map refinement, but is dependent on the quality of the input disparity map.
Acknowledgments

We would like to thank our supervisors, Felix Järemo-Lawin and Jimmy Jons-son, as well as our examiner, Michael Felsberg, for their support throughout this work. We would also like to thank Fredrik Lundell, Viktor Ringdahl and Henrik Carlqvist for helping us getting acquainted with the systems at Saab Dynamics, and Hans Holmgren for giving us the opportunity to carry out this work at his department. Finally, we would like to extend our gratitude to Abdelrahman El-desokey, the principal author of the main refinement method used in this work, for his invaluable discussions and feedback.

Linköping, June 2019
Filip Skarfelt and Daniel Cranston
## Contents

1 Introduction 1
   1.1 Objective ......................................................... 2
   1.2 Problem Formulation ............................................. 3
   1.3 Proposed Methodology .......................................... 3
   1.4 Limitations ....................................................... 4
   1.5 Report Structure ............................................... 4
   1.6 Division of Labour .............................................. 4

2 Stereo Disparity Refinement 5
   2.1 Definitions ...................................................... 5
   2.2 Related Work ................................................... 6
      2.2.1 Classical Approaches ..................................... 6
      2.2.2 Machine Learning Approaches ............................ 7
   2.3 Selection of Method ............................................. 8
   2.4 Initial Disparity ................................................ 9
      2.4.1 Noteworthy Characteristics ................................. 9
      2.4.2 Accompanying Confidence Measures ..................... 10
      2.4.3 Synthetic Initial Disparity ............................... 11
   2.5 Normalized Convolution ........................................ 13
      2.5.1 Normalized Averaging ..................................... 14
   2.6 Method 1: Refinement Network ................................ 14
      2.6.1 Unguided Network .......................................... 15
      2.6.2 Guided Network ............................................. 18
      2.6.3 Training Strategy .......................................... 19
   2.7 Method 2: Superpixel-Based Refinement ....................... 21
      2.7.1 Superpixel Segmentation .................................. 22
      2.7.2 Global Optimization Layer ................................ 23
      2.7.3 Local Optimization Layer ................................ 24
   2.8 Method 3: Baseline ............................................... 24
   2.9 Evaluation ....................................................... 24

3 Dataset Generation 29
   3.1 Related Work .................................................... 29
<table>
<thead>
<tr>
<th>Contents</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.2 Method Used</td>
<td>30</td>
</tr>
<tr>
<td>3.2.1 Structured Light</td>
<td>30</td>
</tr>
<tr>
<td>3.2.2 Rectification</td>
<td>31</td>
</tr>
<tr>
<td>3.2.3 The Scenes and Final Setup</td>
<td>33</td>
</tr>
<tr>
<td>3.2.4 Merging of Disparities</td>
<td>34</td>
</tr>
<tr>
<td>3.2.5 Evaluation</td>
<td>37</td>
</tr>
<tr>
<td>4 Results</td>
<td>41</td>
</tr>
<tr>
<td>4.1 Disparity Refinement</td>
<td>41</td>
</tr>
<tr>
<td>4.1.1 Pre-Training Results</td>
<td>42</td>
</tr>
<tr>
<td>4.1.2 Evaluation Results: Own Dataset</td>
<td>45</td>
</tr>
<tr>
<td>4.1.3 Evaluation Results: Middlebury V3</td>
<td>50</td>
</tr>
<tr>
<td>4.1.4 Summary and Qualitative Comparisons</td>
<td>55</td>
</tr>
<tr>
<td>4.2 Dataset</td>
<td>60</td>
</tr>
<tr>
<td>4.2.1 Disparity Maps in General</td>
<td>60</td>
</tr>
<tr>
<td>4.2.2 Evaluation</td>
<td>64</td>
</tr>
<tr>
<td>5 Discussion</td>
<td>69</td>
</tr>
<tr>
<td>5.1 Disparity Refinement</td>
<td>69</td>
</tr>
<tr>
<td>5.1.1 Initial Disparity</td>
<td>70</td>
</tr>
<tr>
<td>5.1.2 Choice of Penalty Function</td>
<td>71</td>
</tr>
<tr>
<td>5.1.3 Categorization of Ground Truth</td>
<td>72</td>
</tr>
<tr>
<td>5.1.4 Closing Remarks and Suggestions for Future Work</td>
<td>72</td>
</tr>
<tr>
<td>5.2 Dataset</td>
<td>73</td>
</tr>
<tr>
<td>5.2.1 Problems and Improvements</td>
<td>73</td>
</tr>
<tr>
<td>5.2.2 Merging Process</td>
<td>74</td>
</tr>
<tr>
<td>5.2.3 Future Work</td>
<td>75</td>
</tr>
<tr>
<td>6 Conclusions</td>
<td>77</td>
</tr>
<tr>
<td>6.1 Disparity Refinement</td>
<td>77</td>
</tr>
<tr>
<td>6.2 Dataset</td>
<td>78</td>
</tr>
<tr>
<td>Bibliography</td>
<td>79</td>
</tr>
</tbody>
</table>
Stereo matching is a well-studied topic within computer vision. It tackles the problem of extracting 3D information from 2D images. By observing the scene from two different viewpoints and comparing the relative positions of objects in the 2D images, depth information can be extracted. While the topic has existed for some time, the approaches to solving stereo matching have historically been centered largely around classical computer vision and image processing techniques. That is, until recently. With the popularization of machine learning techniques as a means of solving stereo matching, such methods have come to compete with, and in many ways outperform, classical computer vision methods.

Generally, stereo matching algorithms can be structured into four different steps: matching cost, cost aggregation, disparity selection and disparity refinement, as was described in [1]. This thesis mainly tackles the fourth step: disparity refinement. An initial but incomplete estimation of the disparity map is thus assumed.

In the case of this thesis, the algorithm used to produce the raw disparity map is the one used at Saab Dynamics. Due to the corporate ownership of this algorithm not much can be revealed about it other than that it uses a classical approach, as opposed to a learning algorithm, and also produces confidence measures. The proposed technique for the refinement step is however learning based.

The main areas that are improved upon in the disparity refinement step are holes in the disparity map due to mismatches and occlusion. The holes can also appear due to lack of intrinsic dimensionality in those areas. Occlusions however, appear because there are areas in the image pairs that are not viewable from both cameras. An example of an initial disparity map is shown in figure 1.1.
Figure 1.1: Example of output from the stereo algorithm at Saab. Notice how the map contains holes in some regions (black pixels).

When evaluating the performance of disparity map estimation or refinement methods, or when training the parameters of machine learning approaches, it is very useful to have a ground truth of the disparity map. Datasets containing ground truths vary greatly in size, characteristics and density. A dataset with dense ground truth disparity maps is therefore created for use with the refinement method, and will be released to the public following the publication of this work.

1.1 Objective

The objective of this thesis work is twofold. First, we investigate the suitability of applying a modified version of the network proposed by Eldesokey et al. [2] in order to refine the initial disparity maps generated by the stereo matching algorithm used at Saab Dynamics. This method has shown state-of-the-art results for the related task of depth completion. Therefore, it is of interest to see if this method can perform well at disparity refinement as well. Furthermore, this method has comparatively few parameters and considers confidence values of each disparity measurement in the map.

The input that was originally used for this method was projected LiDaR measurements. These measurements are sparse and approximately uniformly distributed in the image, and are generally reliable. Conversely, the input data used in this thesis, i.e. Saab’s, contains more outliers and is generally not as reliable as LiDaR.

Second, we create a dataset of ground truth disparity maps using equipment at Saab, namely a setup for structured light and the stereo rig cameras. This dataset acts both as a means of fine-tuning the network and as a benchmark upon which the method can be evaluated.

There are already good datasets, such as Middlebury [3], for evaluation of stereo algorithms. There is however a shortage of datasets that are suited for learning algorithms. In this thesis we attempt to develop a method for creating dense ground truth datasets in an efficient manner, such that enough images can
be generated for use by learning algorithms.

1.2 Problem Formulation

The main questions that are answered in this thesis are:

1. Is the depth completion network proposed by Eldesokey et al. [2] able to produce state-of-the-art results also for disparity map refinement?

2. Since the input data is not as reliable as that of LiDaR, what measures should be taken to ensure stable training of the network, and what kind of performance can be expected?

3. What challenges exist when creating a ground truth dataset for disparity matching, and how can these be overcome?

4. How performant can we make the ground truth dataset, with a process that allows generation of enough images to be used for fine-tuning?

1.3 Proposed Methodology

In order to answer the above questions, the efforts in this investigation are split into three separate areas:

1. Construct and train a network similar to [2] that takes an initial disparity map (and associated RGB image and disparity confidence measures as guidance) and outputs a dense, more refined disparity map.

2. Implement a classical method that acts as a baseline for comparison.

3. Create a dataset of stereo images and ground truth disparities using equipment provided by Saab.

In order to properly compare the results of the above network, a non-learning method is also implemented. The main source for comparison is therefore the disparity refinement results when the non-learning method was used. This method is described in section 2.7. Finally, a straight-forward and simple method is also implemented to act as the baseline. This method is described in section 2.8.

These methods are evaluated on images from the produced dataset as well as the training images from the Middlebury V3 dataset [3], as the ground truth for these images are the only ones publicly available. Details pertaining to the evaluation are described in section 2.9. The method for creating the dataset is explained in section 3.2. The quality of the dataset itself is evaluated using various metrics explained in section 3.2.5.
1.4 Limitations

1. The raw disparity map is assumed to have been produced at an earlier stage.
2. The size of the created dataset is limited by the available resources.
3. The type of images producable in the dataset is limited by what the equipment allows.
4. Although some stereo matching algorithms do not necessarily need to perform any image rectification, this thesis report and all the theory presented herein is based on the rectified case.
5. Since the task is to refine initial disparity maps, no end-to-end methods are considered.

1.5 Report Structure

This report is structured in the following manner. In chapter 2, the underlying theory and related work on stereo disparity refinement is explained, followed by a thorough explanation of the implemented methods. The chapter ends with an explanation of how the methods are evaluated. Chapter 3 is dedicated to dataset generation. Similarly to the previous chapter, chapter 3 begins with a problem definition and study of previous related work, followed by a detailed description of the methods used to create the dataset. Chapter 4 shows the comparative results of the implemented refinement methods as well as the results from the dataset generation. This is followed by a discussion and suggestions for future work in chapter 5. Finally, conclusions are presented in chapter 6.

1.6 Division of Labour

Since this thesis has two authors, a clear division of labour is necessary. The report structure makes it easy to follow the division of labour. Chapters or sections pertaining to the refinement of disparity maps are written by Daniel Cranston, while those pertaining to dataset generation are written by Filip Skarfelt. The exception is this introduction chapter which is a common effort.
In this chapter, the proposed methods for refining the disparity maps are presented. A few key concepts are first defined, followed by a presentation of related work. We state the proposed methods to be investigated and argue for their suitability. The methods are then explained in depth. Finally, we explain how the performance of the methods are evaluated.

2.1 Definitions

To give a better understanding of the problem at hand, a few key concepts need to be defined.

Rectified Image Pair

Consider two images taken from cameras that observe the same scene but from different viewpoints. The theory of epipolar geometry states that for each point in one image there exists a corresponding epipolar line in the other image. This line constitutes all the points in the second image that might be in correspondence with the first point. Rectified stereo images imply that the epipolar lines are horizontal. This is achieved by horizontally aligning the two cameras, or by applying rectifying homographies, e.g. [4]. This simplifies the task of finding corresponding points, since the search area is reduced to a single row. An illustration of image rectification is shown in figure 3.2.

Stereo Matching

Stereo matching deals with extracting depth information from two (or more) images, based on the visual disparity between corresponding points. Two points
Stereo Disparity Refinement

(one in each image) are said to be corresponding points if they are projections of the same 3D point. Assuming a rectified image pair, the disparity value for a pixel is the relative horizontal position between the pixel and its corresponding point in the other image. By computing this for every pixel in an image, a disparity map $D$ can be obtained. Using this map, the focal length of the camera $f$ and the baseline distance between the cameras $b$, depth values $z$ can then be computed for each pixel $x$ as

$$z(x) = \frac{b \cdot f}{D(x)}.$$  \hfill (2.1)

Disparity Refinement

Stereo matching algorithms generally struggle to find correct correspondences in areas of occlusion, low texture or sharp edges. Therefore, a refinement step is usually performed on the initial disparity maps, and it is here that the focus of this part of the thesis work lies.

2.2 Related Work

As described by Scharstein et al. [1], stereo matching algorithms can be structured into four different steps: matching cost, cost aggregation, disparity selection and disparity refinement. As explained by R.A. Hamzah and H. Ibrahim in [5], stereo algorithms, including refinement methods, can in broad terms be divided into local and global methods. Global methods produce a disparity map by minimizing a global energy function over all disparities. These are generally not suited for real-time systems due to their computational complexity, but they usually produce good results. Local methods, or window methods, only consider a local area when calculating the matching cost for a pixel. The disparities are assigned through winner takes all optimization.

Below follows our literature study on stereo disparity refinement. Two fundamental approaches exist: classical computer vision approaches and machine learning approaches. These are treated separately in the following two subsections.

2.2.1 Classical Approaches

While machine learning approaches are definitely in focus in recent years, several promising classical approaches have also been proposed. A selection of such approaches are presented below. For a survey of classical algorithms between 2004 and 2015, see [5].

Ye et al. [6] identifies incorrect correspondences from the initial disparity map by means of Left Right Consistency (LRC) checking and divides them into occlusions and mismatches, upon which their multi-step refinement framework is applied. This framework makes use of and expands on several different methods in
2.2 Related Work

the literature, such as the cross-based support region [7], cross-region voting [8], and disparity inheritance [9].

Chang and Ho. [10] further augment LRC checking with a local consistency check, and refine inconsistent pixels through the use of the distance transform and a probabilistic support-and-decision styled voting scheme. Building on this work are Zhao et al. [11], who make use of optical flow to further enhance the result, and Jang et al. [12], who incorporate a coarse-to-fine segmentation approach.

S. Drouyer et al. [13] propose a method where the scene is assumed to contain several objects that are composed of multiple simple shapes. These simple shapes are considered to be representable as planes. These planes are estimated from the raw disparity map, by converting the disparities to 3D points. The holes in the disparity map are filled in by converting the 3D points from the model back to the disparity map. A hierarchical segmentation approach, which is similar to a binary partition tree [14], is used in order to handle regions of different characteristics, which is not possible by simply choosing a level of the partition tree.

In the recent work by Yan et al. [15], a two-step approach is employed on a superpixel level. The first step can be described as a global method where mean disparities for each superpixel is estimated from the initial disparities using Markov Random Field inference. In the second step, RANSAC plane fitting is used on each superpixel, constrained by the superpixels mean disparity to prevent degeneracy in the cases where the initial disparity values of the superpixel are noisy or sparse. This is followed by a Bayesian inference and prediction based refinement step. Finally, adaptive mean and median filters are applied.

2.2.2 Machine Learning Approaches

Lots of learning algorithms have been developed recently in order to tackle the stereo matching problem. As discussed in [16], some integrate deep CNNs as components [17], and some convert the whole stereo pipeline into an end-to-end deep CNN-algorithm [18]. The refinement stage in particular can also be implemented as a CNN, acting directly on the disparity map [19]. In many stereo matching methods, the disparity refinement stage is dependent on the end-to-end solution of the method and is not always easily included into other frameworks. Therefore, mainly algorithms that were easily separable into a refinement stage were considered in this thesis.

Batsos and Mordohai [16] explain that the performance of deep networks is highly correlated to the number of their parameters. However, Eldesokey et al. showed that it is possible to get superior performance while only requiring 5% of the number of parameters compared to state-of-the-art methods [2], in the closely related problem of depth completion. This was done by propagating confidences by utilizing normalized convolution layers, and using the structural information in the image to further guide the network.

The notion of using such secondary information when training networks is not unique to [2]. The use of a binary validity mask to provide information of
missing pixels has been used in a wide range of related topics, such as optical flow estimation and image inpainting [20], [21]. These early works do not propagate the validity masks through the network, but leave them constant. Uhrig et al. [22] proposes a way to propagate the validity mask by means of max pooling. This approach is however sub-optimal for sparse data and deep nets, as is shown in the work by Jarlitz et al. [23]. Building on these observations, [24] and [2] instead consider the validity mask continuously and propagate the mask through the network using Normalized Convolution [25], which mitigates the problems from [22].

2.3 Selection of Method

In this thesis, there are a few points aside from the accuracy of the method that have to be taken into consideration. Since the results of this thesis is meant to be used as an extension to the stereo matching algorithm used at Saab, we also consider the needs of Saab in our selection of methods. One such need is real-time performance. This is not a central theme in this thesis, but because of this, any method or group of methods that are known to have long computation times are not considered for selection.

The method proposed by Eldesokey et al. [2], but adapted for disparity refinement, is chosen as the main method because it promises the performance of deep neural networks while also being very lightweight. Having comparatively few parameters to train should reduce the amount of training data needed. The fact that the network considers confidence measures of the raw disparity map is another positive. Further, the selection of this method ties in nicely with the other part of this thesis (chapter 3) which addresses the generation of a dataset. The aim is to use this dataset when training the network.

As we wish to compare the above method with a classical (non-learning) state-of-the-art method, we choose the method proposed by Yan et al. [15] to fill this role. In their paper, the initial disparity used as input was the predictions made from the CNN-based stereo matching network developed by Zbontar and LeCun [17]. This is here replaced by the initial disparity from Saab’s algorithm. Since the method operates directly on the raw disparity map and does not perform any costly correlation between left and right images, it fits the demands and limitations set out by Saab. While Markov Random Field inference can be costly, the method performs this inference on a superpixel level which speeds up the computation significantly. As such we get the benefits of this global method at a comparatively low computational cost.

Lastly, we want to compare the above methods to a simpler and more straightforward approach. For this purpose, we choose the inpainting method described in the paper by A. Telea [26].
2.4 Initial Disparity

Before diving into the details of the methods, it is important to understand the nature and characteristics of the input data.

Disparity refinement, as the term suggests, implies that there exists some initial disparity that we aim to refine. In this thesis, this initial disparity refers to disparity that is produced by the stereo algorithm at Saab. Other terms for this are sometimes used throughout the thesis, like "raw disparity", "unrefined disparity" or "Saab’s initial disparity". These terms are all equivalent and refer to "initial disparity" as is defined here.

2.4.1 Noteworthy Characteristics

Examples of initial disparities produced when Saab's stereo algorithm was applied to a range of different scenes and datasets is shown in figure 2.1.

![Figure 2.1: Examples of output disparity from Saab's stereo algorithm using different datasets. Top: Middlebury V3 [3]. Middle: Dataset created in this thesis (chapter 3). Bottom: FlyingThings3D [18]](image)
As can be seen in figure 2.1, the disparity map contains holes and general areas with no disparity values. It is apparent that the results of the algorithm vary greatly depending on the scene. Noticable re-occuring patterns are (1) disparities of foreground objects appear bloated/dilated and (2) disparity values are not found close to disparity (depth) discontinuities. Another important note to make is that Saab’s stereo algorithm generally requires parameter tuning for each scene, something that would be too time consuming when training on large datasets. Since the stereo algorithm is the property of Saab, its details are not disclosed in this thesis. However, it can be described loosely as follows:

1. It is a classical (non-learning) method
2. It is made to run in real-time, meaning that accuracy is to some extent sacrificed in favour of speed
3. It only produces a disparity map expressed in the left image
4. It produces a certainty mask for each outputted pixel, meaning that each pixel is accompanied with some measure of confidence.

These fundamental characteristics of the initial disparity was influential to our choice of method. An important observation can be made from #3. It is common for stereo algorithms to exploit information from both left and right disparity maps. Since Saab’s stereo algorithm only produces one disparity map, we would not be able to exploit such methods. The possibility of expanding Saab’s algorithm ourselves is of course a possibility, but such an approach would in a sense mean a departure from the goal of this part of the thesis, which is to refine the output of Saab’s stereo algorithm "as is". In regards to #2 and #4, it is difficult to assess beforehand the extent to which accuracy is sacrificed, and how reliable the produced confidence measures are.

### 2.4.2 Accompanying Confidence Measures

As mentioned earlier, the algorithm produces a confidence measure for each pixel in the estimated disparity map. While it is not uncommon for confidence measures to be expressed as a binary mask, indicating for each pixel the presence or absence of a measurement, these measures hold continuous values between 0 and 1.

The confidences outputted from Saab’s algorithm conforms to the usual representation of such masks as explained in section 2.5, i.e. they hold values between zero (0% confidence) and one (100% confidence). However, most of the pixels hold confidence values above 50%. Comparing the actual disparities similarity to the ground truth we quickly find that these confidence values should not be considered as percentage of certainties. Some form of pre-processing of the initial confidences need to be performed before feeding it into the network. In this thesis, we investigate two such approaches:

1. Thresholding the certainties using a cut-off $\alpha$
2.4 Initial Disparity

In the above, $\beta$ is set to 4, and $\alpha$ is not disclosed. The reason for why $\alpha$ is not disclosed is that it is more closely tied to Saab’s algorithm. These approaches are evaluated in chapter 4. An example of a confidence map from Saab’s stereo algorithm is shown in figure 2.2.

![Figure 2.2: An example of a output confidence mask from Saab's stereo algorithm](image)

2.4.3 Synthetic Initial Disparity

In addition to the initial disparity created by Saab’s algorithm, with its characteristics that were explained in section 2.4.1, we also create synthetic disparities that mimic what a general stereo algorithm might produce. These disparities act as a safer and more controlled input to the network, while still approximating the output of a general stereo algorithm. This allows us to train on the large synthetic dataset more easily since using Saab’s algorithm would demand that we fine tune for each scene in the dataset to ensure that the output is true to what it would output in a real scenario. Furthermore, this synthetic disparity allows us to more easily compare different design choices, since we have full assurance that the input data is trustworthy.

How we create this synthetic disparity is hinged on two assumptions. First, we assume that general stereo algorithms are good at finding correct disparities in areas with high intrinsic dimensionality. Such areas are due to structure in the texture of a surface, or depth discontinuities in the image. In the latter case, algorithms can struggle to find the exact border of the discontinuity. Secondly, we assume that disparities at leftover areas (intrinsically low dimensional areas) are very sparse, but still accurate. This second assumption is a strong one and might not reflect real stereo algorithms. Nevertheless it is necessary to make such an assumption if we wish to create a safe and stable input for the network.

We create this disparity by observing the corresponding RGB image and ground truth. We begin by performing Harris corner detection [27] on the RGB image. Thresholding the resulting image with a cut-off $c_{rgb}$ yields a mask $M_{rgb}$ that highlights areas with high intrinsic dimensionality, meaning either areas at depth discontinuities or high-textured areas. To model how stereo algorithms struggle with depth discontinuities, we create a similar mask by this time perform-
Harris corner detection on the ground truth, followed by thresholding with a cut-off $c_{\text{depth}}$. This mask, $M_{\text{depth}}$, highlights the exact borders of depth discontinuity regions in the image. The threshold cut-off value is dependent on the scenes, and after investigating many different choices we choose $c_{\text{depth}} = 10000$ and $c_{\text{rgb}} = 0.00001$.

By subtracting $M_{\text{depth}}$ from $M_{\text{rgb}}$, we obtain a mask $M_{\text{final}}$ that highlights high-texture areas while leaving the exact borders of depth discontinuities un-highlighted. We let this mask be the confidence mask of our synthetic disparity. The disparity itself then follows directly, we simply apply the mask to the ground truth. To model some imperfection in the disparities, we add a small amount of noise ($\pm 2$ pixels) to the ground truth values before applying the mask. Figure 2.3 summarizes the above by providing an overview of how the synthetic initial disparity is constructed.

![Figure 2.3: Creation of synthetic initial disparity, illustrated on an item from the FlyingThings3D [18] dataset. RGB image and Ground truth is sampled in a strategic fashion to produce initial disparities that can be used during training on synthetic datasets. Gray pixels indicate 0% confidence.](image)

Thus we have created, from one RGB and ground truth image, a plausible and well behaved initial disparity. This procedure is applied to every image of all the datasets we use. In fact, it is incorporated as a feature in our data-loaders, which means that we can toggle the use of this synthetic disparity or those of Saab’s algorithm with ease.
2.5 Normalized Convolution

This chapter is dedicated to describing the theory of Normalized Convolution, since it is an integral part of the chosen refinement network of method 1.

Normalized Convolution is defined in the work by Knutsson and Westin [25] (using slightly different notations) as:

\[ U = N^{-1}D = (B^T D_a D_c B)^{-1} B^T D_a D_c f \]  \hspace{1cm} (2.2)

Where
- \( D_a \) is a \( n \times n \) diagonal matrix containing the applicability function,
- \( D_c \) is a \( n \times n \) diagonal matrix containing the certainties of the signal,
- \( B \) is the matrix containing the basis vectors in its columns,
- \( f \) is the input signal consisting of \( n \) samples, and
- \( U \) is the result of the normalized convolution.

Normalized Convolution is a technique that enables convolution over sparse or partially unknown data. Initial disparity maps fit this description, as disparity data is noisy or partially missing. Each measurement in the data need to be accompanied with a value describing the confidence of the measurement. In the case of images this is usually expressed as a confidence map covering the entirety of the image. This map is often normalized between zero and one, where a value of one indicates 100% confidence in the measurement (i.e. pixel) and a value of zero indicates that the value of the pixel is missing.

Below follows an explanation of normalized convolution in the case of images. Consider an image \( F \) with partially missing data and a confidence map \( C \) describing the reliability of each pixel. The neighbourhood around each pixel can be expressed by a finite vector \( f_k \) and an accompanying confidence vector \( c_k \). Using this representation, \( F \) can be modelled locally by projecting each neighbourhood \( f_k \) onto a subspace spanned by some basis functions \( b_i \). For simplicity, these basis functions are packed into a matrix \( B \), holding each basis function in its columns:

\[ B = \begin{pmatrix} \ldots & b_i & \ldots \end{pmatrix} \]  \hspace{1cm} (2.3)

The local modeling of each \( f_k \) then becomes:

\[ f_k' = Br \]  \hspace{1cm} (2.4)

where \( r \) holds the coordinates of \( f_k' \) with respect to \( B \). Note that \( f_k' \neq f_k \) since \( f_k' \) is a projection of \( f_k \) onto a subspace. However, given the basis \( B \), a \( f_k' \) can be found that minimizes the mean-square error \( \| f_k' - f_k \| \) by choosing \( r \) to be:

\[ r = (B^T G_0 B)^{-1} B^T G_0 f_k \]  \hspace{1cm} (2.5)
Where $G_0$ is the metric that defines the scalar product used. In general normalized convolution, this metric holds the applicability and confidence measures. Since the confidence measures are position-dependent, the metric itself is also position-dependent and is defined for each position $k$ as:

$$G_0[k] = \text{diag}(a \cdot c_k) \quad (2.6)$$

The applicability $a$ acts as a sort of weighting or localization on the basis functions and must be positive. What is a suitable choice depends on the application. A common choice for the applicability is a Gaussian.

### 2.5.1 Normalized Averaging

In the previous explanation, not much was said about the basis functions $B$. These vary depending on application, but in general you want them to span a space that covers the interesting part of the signal as much as possible. The most simple case is when there is only one basis function $B = 1$. This case is commonly referred to as Normalized Averaging. This means that each position (pixel) is modeled to a single scalar value, the mean value of the local neighbourhood weighted by the certainty and applicability. Further, because there is only one basis function (with all elements = 1), equation (2.2) can be more compactly described as:

$$U[k] = \frac{a \ast (F \cdot C)}{a \ast C}[k] \quad (2.7)$$

Where $\ast$ denotes convolution and $\cdot$ denotes element-wise multiplication. An interesting observation is that normalized averaging is equivalent to regular convolution when $C$ is constant, and the applicability $a$ is then analogous to the filter coefficients. As we shall see shortly, the refinement network makes frequent use of normalized convolution.

### 2.6 Method 1: Refinement Network

This section explains the network architecture of the disparity refinement network used in this thesis. The reader is assumed to already have some knowledge of key concepts related to convolutional neural networks, such as weights, biases, activation, pooling, loss and back-propagation. As such, these concepts are not explained here.

As is mentioned in chapter 2.3, the disparity refinement network is based on the RGB guided depth completion network proposed by Eldesokey et al. [2]. This network can be divided into two main parts: an unguided network and a guided network that takes the output of the unguided network and the RGB image as input. "Guidance" in this case refers to the additional structural information that the RGB image provides. The unguided network does not consider the RGB image whereas the guided network does. A simple block diagram of the network is shown in figure 2.4.
2.6 Method 1: Refinement Network

Figure 2.4: A simple diagram of the components that make up the Disparity Refinement Network. Initial disparity and confidence maps from Saab’s algorithm are fed to the unguided part of the network, producing refined (intermediate) outputs. These outputs together with the RGB image is fed to the guided network which produces the final disparity map.

2.6.1 Unguided Network

Normalized Convolution Layer

The workhorse of the unguided network is the normalized convolution layer as described in [2]. It can be seen as an extension or generalization of the standard convolution layer, taking as input not only data but confidence of data as well. Here, "data" refers to the initial disparity map and "confidence of data" refers to the corresponding confidence mask.

Furthermore, while the standard convolution layer performs regular convolution, the normalized convolution layer replaces this operation with normalized averaging, equation (2.7). While the task of the standard convolution layer is to learn optimal filter coefficients, the normalized convolution layer instead learns the coefficients of the applicability. This means that each weight in the layer corresponds to a coefficient in the applicability.

As was explained in section 2.5, the applicability must be positive. In order to enforce this, the SoftPlus function $\Gamma$ is applied to the weights before the forward pass. The SoftPlus function is defined as follows:

$$\Gamma(z) = \log(1 + \exp(\beta z))$$

Where $z$ is the input variable to the SoftPlus function and $\beta$ is a scalar parameter. This effectively means that the applicability $a$ that gets used in the normalized convolution operation becomes:

$$a = \Gamma(W)$$
Where $W$ is the weights of the layer. The normalized convolution layer outputs not only the result of normalized averaging onto the input data (data refinement), but also produces a refined confidence mask (confidence propagation). The refined confidence mask is calculated according to equation (2.10).

$$C_{out} = \frac{(a \ast C_{in}) + \epsilon}{\sum a}$$  

(2.10)

Where $\ast$ denotes convolution and $\sum a$ implies summation of all elements in the applicability, which in turn was described in equation (2.9). For completeness, the result of normalized averaging onto the input data is expressed in equation (2.11).

$$F_{out} = \frac{a \ast (F_{in} \cdot C_{in})}{a \ast C_{in} + \epsilon}$$  

(2.11)

Where $F_{out}$ is the output disparity map, $F_{in}$ is the input disparity map and $\epsilon$ is a small number preventing division by zero. Notice the similarity with equation (2.7). Also note that $a$ in equations (2.10) and (2.11) are one and the same. In other words, the same weights are applied to both the data refinement (2.11) and confidence propagation (2.10). These two together can be viewed as constituting the forward propagation step of a normalized convolution layer.

**Unguided Network Architecture**

The Unguided Network is then constructed by a series of normalized convolution layers in the same manner as U-Net [28]. This results in a compact multi-scale architecture that shares weights between different layers. An illustration of this architecture is shown in figure 4 of [2]. With the permission of the author, it is included here as well and can be seen in figure 2.5.

Figure 2.5: Illustration of the Unguided Network, humbly borrowed from [2]. Here, $Z_n$ is the disparity map at different stages $n$ and $C_n$ is the corresponding confidence map. This means for example that $Z_0$ is the input disparity and $Z_{13}$ is the refined disparity outputted from the unguided network.
The depth of the network (scale depth) is in itself a parameter, which we choose to set to 3 in accordance with [2].

In the above, we have highlighted how the normalized convolution layer is fundamentally different from its standard counterpart. This raises the question of what a good loss function is for a network using such layers. Loss functions which are common for standard convolution neural networks like the L1 or L2 norm do not take confidences into account. Realizing this, Eldesokey et al. [2] proposed a new loss function consisting of a data term and a confidence term, and it is this norm that we adopt as our loss in this thesis. The data term is simply the huber norm [29] between the output of the last normalized convolution layer $Z_{last}$ and the ground truth disparity map $T$:

$$E_{data} = \|Z_{last} - T\|_H$$

(2.12)

The huber norm is defined as follows:

$$\|x\|_H = \begin{cases} \frac{1}{2}(x)^2, & \text{if } |x| < \delta \\ \delta |x| - \frac{1}{2}\delta^2, & \text{otherwise} \end{cases}$$

(2.13)

Where $\delta$ is a scalar parameter. The huber norm can be seen as a combination of the L1 and L2 norms, where $\delta$ decides the threshold where L1 should be used instead of L2. Eldesokey et al. [2] explains that this norm is a good choice since it can help prevent exploding gradients, thus improving the stability of convergence.

The confidence term of the loss function is designed to maximize the output confidence and is dependent on the data term. It is defined in (2.14) below.

$$E_{conf.} = -\frac{1}{p}(C_{last} - E_{data}C_{last})$$

(2.14)

Where $C_{last}$ is the output confidence from the last layer and $p$ is the current epoch number. Bringing (2.12) and (2.14) together, the total loss then becomes:

$$E_{tot} = E_{data} + E_{conf.}$$

(2.15)

Note in (2.14) that the confidence term gets inversely scaled by the current epoch number. The implication of this is that $E_{conf.}$ grows smaller (decays) with each epoch. This is to prevent this term from dominating the loss function once a few epochs have elapsed and the data term has started to converge. In order to be able to refer back to this loss moving forward, it is henceforth called the ConfLossDecay loss (sometimes shortened to ConfLoss for brevity).

Now that the loss function is defined, back-propagation is no different from that of a standard convolutional network. There is however one detail worth noting, which is that the SoftPlus function $\Gamma$ must be included in the gradient calculations. For a given layer $l$, the gradients of the weights in that layer, $W_l$, can be computed according to the chain rule as:

$$\frac{\delta E_{tot}}{\delta W_l} = \sum \frac{\delta E_{tot}}{\delta Z_l} \cdot \frac{\delta Z_l}{\delta \Gamma(W_l)} \cdot \frac{\delta \Gamma(W_l)}{\delta W_l}$$

(2.16)
This concludes the description of the Unguided Network. In summation, we observe that there are several parameters that need to be set. In this thesis, we adopt the same choices as in [2]:

- Choice of loss function: $\texttt{ConfLossDecay}$ (2.15)
- Choice of Huber (2.13) parameter $\delta$: 1
- Choice of $\texttt{SoftPlus}$ (2.8) parameter $\beta$: 10
- Choice of scale depth: 3

A final note is that this network provides a first refinement of the initial disparity at a low computational cost. It can be considered its own self-contained network. As such, it is independently evaluated and compared to both the superpixel-based method and the Disparity Refinement Network in its entirety. The results of these evaluations are shown in chapter 4.

### 2.6.2 Guided Network

The Guided Network takes as input 3 things. The first two are the outputs of the unguided network: a refined (intermediate) disparity map and a refined confidence map. The third input is the RGB image. As mentioned earlier, the unguided network does not consider the information in the RGB image. Therefore, one should not expect it to produce correct disparities at discontinuities in the image. This is instead the task of the guided network, to use the structure in the RGB image to refine the disparity values around edges.

An illustration of the guided network is shown in figure 2.6. The blue blocks indicate series of standard CNN layers. Note that the network does not make any refinements to the confidences; it only refines the disparity map.

**Figure 2.6:** Illustration of the Guided Network. The blue blocks indicate standard CNN layers. The refined (intermediate) disparity map is fed through a series of CNN layers while the refined confidence map is concatenated with the RGB image and fed to a feature extraction network. The output from these two are concatenated and fed to a fusion network that produces the final disparity map.
This is identical to [2] in the overall setup and characteristics of the layers. The parameters of the guided network are summarized in table 2.1. The arrows in the "Channels" and "Kernel size" columns indicate how these attributes change through the layers. For a more complete description, see chapter 5 of [2].

Table 2.1: Summary of parameters for the guided network

<table>
<thead>
<tr>
<th>Block name</th>
<th>#Layers</th>
<th>Channels</th>
<th>Kernel size</th>
<th>Activation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disparity</td>
<td>6</td>
<td>1-&gt;16</td>
<td>3x3</td>
<td>ReLu</td>
</tr>
<tr>
<td>Feature Extraction</td>
<td>6</td>
<td>4-&gt;64</td>
<td>3x3</td>
<td>ReLu</td>
</tr>
<tr>
<td>Fusion</td>
<td>6</td>
<td>80-&gt;32-&gt;1</td>
<td>3x3-&gt;1x1</td>
<td>ReLu</td>
</tr>
</tbody>
</table>

It is worth noting that [2] compares several choices of architecture. The one denoted Multi-Stream (Late Fusion) (abbreviated as MS-Net[LF]) provided the best results, and it is this architecture we have described here. In the results, this network architecture is denoted simply as MS.

Apart from MS-Net[LF], we also evaluate the architecture denoted Encoder-Decoder (Early Fusion) (abbreviated as EncDec-Net[EF]). This architecture concatenates the intermediate disparity and confidence maps with the RGB image and feeds it through a network similar to U-Net [28] with skip-connections, using ReLu activation in the encoder and Leaky ReLu activation in the decoder. An illustration of this architecture can be seen in figure 7d of [2]. In the results, this network architecture is denoted simply as Enc-Dec.

2.6.3 Training Strategy

To investigate which design choices are suitable, we implement 5 versions of the network. These versions vary in the choice of network architecture and loss function used. The implemented architecture/loss combinations are as follows:

- Guided (Enc-Dec), using the L2 loss.
- Guided (Enc-Dec), using the SmoothL1 loss.
- Guided (MS), using the L2 loss.
- Guided (MS), using the SmoothL1 loss.
- Unguided, using the ConfLossDecay loss.

The SmoothL1 loss is simply the Huber loss (2.13) with $\delta = 1$. Each of these versions are trained using the three different inputs described in section 2.4:

- Saab Initial Disparity, using thresholded confidence values.
- Saab Initial Disparity, using scaled confidence values.
- Synthetic Initial Disparity
This results in a total of 15 different training regimes, since each of the 5 network versions are trained on 3 different input. The training is split into two stages, a pre-training stage and a fine-tuning stage. The details of these stages are described below.

**Pre-Training**

In the pre-training stage, we train the networks using the large synthetic dataset FlyingThings3D [18]. With almost 22000 training images and 4000 validation images, this dataset provides the means to pre-train the networks to a good overall state. However, since many images depict varied and complex scenes, there are some images that Saab’s stereo algorithm fails to find good initial disparities for. Examples of such images are shown in figure 4.11. As can be seen, these disparity maps look nothing like the ground truth; they are very sparse and the vast majority of the given values are incorrect. We consider the refinement of such disparity maps to not be within the scope of this thesis, as the disparity maps are simply too poor to be considered valid initial solutions to be refined. We reinforce this by referring to Mayer et al. [30], who also make similar conclusions on the FlyingThings3D dataset.

Therefore, we discard these faulty initial disparity maps from the dataset before training. To do this, we aim to rank each image $x$ in the dataset according to some penalty function $P$ and discard the images with a high penalty according to this function. There are many ways to choose such a function. The function that we choose in this thesis is shown in equation (2.17).

$$P(I_x, T_x) = \frac{1}{N} \sum_{k \in \Omega} |I_x(k) - T_x(k)|$$  \hspace{1cm} (2.17)

Where $I_x$ is the initial disparity map created by applying Saab’s stereo algorithm to image $x$, $T_x$ is the corresponding ground truth, $\Omega$ is the set of pixels that have values in both $I_x$ and $T_x$, and $N$ is the total number of pixels in $\Omega$. We interpret this function as penalizing images where Saab’s stereo algorithm produces a disparity map for which the average disparity values deviate greatly from the corresponding ground truth. The higher the penalty of an image, the more likely that Saab’s algorithm failed to produce a good initial disparity for it. Thus, by calculating the penalty $P$ for each image $x$ and sorting them by magnitude, we could visually inspect the images to decide which ones to keep. We chose to discard the images that had among the 25% largest penalties, leaving roughly 16000 images left to train on. As we also evaluate the pre-training step separately, we perform an identical ranking on the validation images and use the 800 best images during evaluations.

**Fine-Tuning**

As a final preparation for the evaluations (see section 2.9), we fine-tune the network using all but the last 5 scenes of the dataset created in this thesis (chapter 3). Here, we only use Saab’s initial disparity as input to the network, since this is
the input that is used during the evaluations. Because the dataset is fairly small (26 scenes, 104 disparity images), we limit the fine-tuning to 20 epochs.

2.7 Method 2: Superpixel-Based Refinement

This section explains the state-of-the-art method used for comparison against the refinement network presented in the previous section. As mentioned in section 2.3, it is the classical (non-learning) method proposed by Yan et al. [15].

This method takes as input an initial disparity map and the corresponding RGB image to produce a dense and refined disparity map. The method can be summarized by the following 6 steps:

1. Over-segment the RGB image into superpixels

2. Calculate mean disparities for each superpixel

3. Compute a "neighbourhood system" describing the relation between superpixels

4. Perform constrained RANSAC plane-fitting for each superpixel

5. Refine each superpixels plane by observing its neighbours in a probabilistic fashion

6. Apply adaptive mean and median filtering

The first observation to make is that the method performs all its computations on superpixels rather than on individual pixels, which is crucial since it speeds up the computations significantly. Secondly, we observe that the method requires no information from the right image, an attribute shared with the employed refinement network of Eldesokey et al. [2]. Lastly (and contrary to [2]), we observe that the method does not consider confidences related to the disparity values.

An big assumption that is made in the method is that each superpixel corresponds to a planar surface. In the original paper [15], steps 2 and 3 are denoted as "the global optimization layer" and steps 4 and 5 as "the local optimization layer". We also adopt this naming convention in the explanations that follow. An overview of the method is shown in figure 2.7.
2.7 Stereo Disparity Refinement

Figure 2.7: Illustration of the superpixel-based refinement method.

Each step of the method is described in the following subsections. Note that, for sake of brevity, the explanations are not exhaustive in all cases. In such cases we refer the reader to the original paper [15] for more in-depth information.

2.7.1 Superpixel Segmentation

As both the global and local optimization layers work on a superpixel level, the first step is naturally to perform the superpixel segmentation. This is done on the RGB image using the method proposed by Felzenszwalb and Huttenlocher [31]. This segmentation technique can be described as a graph-based one where each pixel is a node in a undirected graph. The weights on the edges between nodes correspond to the (dis)similarity between two neighbouring pixels. Two pixels are considered neighbours if they are next to each other in the 8-connected sense. The segmentation algorithm is fast compared to most other segmentation algorithms, running in $O(n \log n)$ time (where $n$ is the number of pixels in the image).

There is one parameter of interest in this algorithm. It is denoted $k$ and controls how boundaries are created between superpixels, thus implicitly dictating how many superpixels are created. If $k$ is small, the boundary creation is more discriminative, resulting in more superpixels. Although this infers larger computational costs, a higher level of segmentation is necessary in this method. Thus we choose $k = 30$. An illustration of segmentation results on an image from the dataset created in this thesis for different choices of $k$ is shown in figure 2.8.
2.7 Method 2: Superpixel-Based Refinement

(a) Input image  (b) $k=300$  (c) $k=150$  (d) $k=30$

Figure 2.8: Comparison of superpixel segmentation results on an image from the dataset created in this thesis (chapter 3), using different choices of the parameter $k$.

In summary, this step produces superpixels $s$, where each pixel in the image belongs to exactly one superpixel. The number of superpixels created depend on the parameter $k$. Once created, they are fed as input to the Global Optimization Layer.

2.7.2 Global Optimization Layer

The next step is to assign a mean disparity value to each superpixel $s$. The naivé way would be to simply calculate the mean value of those pixels in the superpixel that have disparity values in the initial disparity map. Instead, the disparities within each superpixel $s$ is modeled as a gaussian with mean $\mu_s$ and variance $\sigma_s$. Each $\mu_s$ is simultaneously estimated through Markov Random Field (MRF) optimization, minimizing the following energy function:

$$ E(\mu) = \sum_{s \in \Omega} \Phi(\mu_s) + \lambda \sum_{(s,t) \in \mathcal{N}} \Psi_{st}(\mu_s, \mu_t) $$

(2.18)

Where $\Omega$ is the full set of superpixels $s$, $\mathcal{N}$ is the set of neighbouring superpixels, $\Phi(\mu_s)$ is called the data term, $\Psi_{st}(\mu_s, \mu_t)$ is called the smoothing term and $\lambda$ is a parameter that tunes the significance of the smoothing term. For a full explanation of the two terms, see chapter IV of the original paper [15].

After all $\mu_s$ have been estimated through MRF, this information is used to create a new set $\mathcal{N}_{3D}$. This set groups superpixels with similar $\mu_s$. While the previously mentioned $\mathcal{N}$ describes superpixels that are neighbours in a 2D spatial sense, $\mathcal{N}_{3D}$ can be interpreted as describing neighbours in depth. More formally, $\mathcal{N}_{3D}$ is created as follows:

$$ \mathcal{N}_{3D} = \{(s, t) \in \mathcal{N} \mid |\mu_s - \mu_t| < L\} $$

(2.19)

Where $\mu_s$ and $\mu_t$ are mean disparities belonging to superpixels $s$ and $t$ respectively, and $L$ is a positive scalar.

In summation, this step produces a mean disparity $\mu_s$ for each superpixel $s$ as well as the 3D neighbourhood set $\mathcal{N}_{3D}$. The creation of mean disparities for each superpixel can be interpreted as fitting a front-parallel plane to each superpixel.
This interpretation is adopted in the following step, which aims to refine these planes into planes of arbitrary orientation.

### 2.7.3 Local Optimization Layer

At this stage, each superpixel has been assigned a front-parallel plane, resulting in a disparity map that is piece-wise constant. The next step is to create slanted plane $\pi_s$ for each superpixel $s$. This is done by observing the disparity measurements inside the superpixel and performing RANSAC plane-fitting. To prevent degenerate cases, the plane-fitting is constrained by disallowing the constructed plane to deviate far from the front-parallel plane.

Since this plane-fitting is performed independently for each superpixel, no local information has been incorporated and no attempts have been made to maintain smoothness between neighbouring superpixels. Therefore, a final refinement is made on all $\pi_s$ using Bayesian inference and the information that $N_{3D}$ provides. The full details of this refinement is omitted for brevity and can be found in chapter V-B of the original paper [15]. The result is a dense and refined disparity map. As a final step, adaptive mean and median filtering is applied to the disparity map to yield the final result.

### 2.8 Method 3: Baseline

As mentioned in section 1.3, a simple method is also implemented and evaluated. For this task, we use the OpenCV implementation of the inpainting method by A. Telea [26].

### 2.9 Evaluation

The above three methods are evaluated on the dataset produced in this thesis (chapter 3) as well as the training images from the Middlebury V3 dataset [3], as the ground truth disparity for these images are publicly available. The results of the evaluations are shown in sections 4.1.2 and 4.1.3 respectively. These two sections make up the main results, whereas 4.1.1 presents the intermediate results from the pre-training of method 1.

Although the different versions of method 1 use different input during pre-training, all versions are evaluated based on the predictions the networks make on Saabs initial disparity (thresholded and scaled confidences), not synthetic.

As a means of visualizing the results, we use the end-point-error (EPE) map. This map is created by taking the absolute value of the difference of the output and ground truth. Such a map provides a means of visualizing how much each pixel of the output deviates from the ground truth. This deviation is also referred to as disparity error. We let disparity errors close to 0 be represented by dark blue and disparity error of some threshold $\delta$ and above be shown in yellow. Disparity errors between 0 and $\delta$ are then shown by an appropriate interpolation between
the two colours. We refer to an EPE map with $\delta = 20$ and $\delta = 4$ as EPE-20 and EPE-4 respectively. An example of these EPE maps can be seen in figure 4.5.

**Evaluation Metrics**

The metrics used in the evaluations are the common Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) as well as the $bad_4$ and $bad_2$ metrics from the Middlebury evaluations [3]. MAE and RMSE are defined in (2.20) and (2.21) respectively.

\[
MAE = \frac{1}{N} \sum_{i} |D(i) - T(i)| \tag{2.20}
\]

\[
RMSE = \sqrt{\frac{1}{N} \sum_{i} (D(i) - T(i))^2} \tag{2.21}
\]

Here, $D$ is the methods output disparity map, $T$ is the corresponding ground truth and $N$ is the number of valid pixels in the ground truth. Both these metrics are in essence pixel-wise comparisons between the output and ground truth.

MAE gives a good sense of the average error while RMSE punishes outliers harder. Outliers in this case are disparity values that deviate greatly from the ground truth. The $bad_4$ and $bad_2$ metrics show the percentage of pixels that have a disparity error larger than 4 and 2 pixels, respectively.

**Categorization of Ground Truth**

When deciding how to evaluate the methods, we identify two principal objectives. One the one hand, we want the methods to correctly fill in holes and missing areas. On the other, we want them to refine areas that Saab’s stereo algorithm managed to detect. Based on these objectives, we divide the pixels of the initial disparity made by Saab’s stereo algorithm into three categories:

- **Category 1 - Correct predictions**: Pixels where Saab’s stereo algorithm find disparity values, deviating less than 4 pixels from the ground truth.

- **Category 2 - Incorrect predictions**: Pixels where Saab’s stereo algorithm find disparity values, deviating more than 4 pixels from the ground truth.

- **Category 3 - Missing predictions (Holes)**: Pixels where Saab’s stereo algorithm fail to find disparity values.

An illustration of this categorization is shown in figure 2.9.
In this way, we separate the ground truth into these three categories and evaluate them separately. This allows us to evaluate the performance of each method in the following three ways: preservation of correct disparities (category 1), refinement of incorrect disparities (category 2) and hole filling (category 3). In addition to this, we also evaluate on the entire ground truth, which provides an overall measure of performance.

**Proposed Metrics for Qualitative Comparisons**

Drawing from the implications of this categorization, we also introduce and make use of two new metrics, which we call $C_{abs}$ and $C_{rel}$. The $C$ stands for Correctness. $C_{abs}$ is the opposite of bad4, meaning the percentage of pixels that have disparity values within 4 pixels from the ground truth. Despite its similarity to bad4, $C_{abs}$ is included as it provides a tangible measure of how successful the given method is. A definition of $C_{abs}$ is given in (2.22).

$$C_{abs}(X) = \frac{N_{X,correct}}{N_{X,total}}$$  \hspace{1cm} (2.22)

Where $X$ is a disparity map, $N_{X,correct}$ is the number of correct pixels (disparity error less than 4 pixels) in $X$, and $N_{X,total}$ is the total number of pixels in $X$. As an example, a disparity map with $C_{abs} = 0\%$ would mean that it does not have a single correct disparity value, and $C_{abs} = 100\%$ would mean that it is more or less identical to the ground truth.

An important realisation is that $C_{abs}$ carries no information of the relative improvement that was made from input disparity to output disparity. To highlight this attractive piece of information, we define our second metric $C_{rel}$ as follows:

$$C_{rel}(D) = \frac{C_{abs}(D)}{C_{abs}(I)} - 1$$  \hspace{1cm} (2.23)
Where $D$ is the methods output disparity map and $I$ is the corresponding initial disparity map that was used to create $D$. These metrics are used in the qualitative results and comparisons of section 4.1.4. A illustration of these metrics is shown in figure 2.10.

Figure 2.10: Illustration of the usefulness of the proposed metrics. Consider the situation where Saab’s stereo algorithm has produced the initial disparity shown in (c). 39.49% of pixels in the input disparity are correct (having disparity error below 4 pixels). The initial disparity is fed to a method that produces the output disparity shown in (d). This new disparity has 63.51% correct pixels, which is a relative improvement of 60.83%. Using our metrics, we can make the same statement as above simply by saying that the output disparity in (d) has metrics $C_{\text{abs}}=63.51\%$ and $C_{\text{rel}}=60.83\%$. 

(a) RGB image  (b) Ground truth  (c) Input disparity. $C_{\text{abs}} = 39.49\%$. Only pixels with disparity error less than 4 are shown.  (d) Output disp. $C_{\text{abs}}=63.51\%$, $C_{\text{rel}}=60.83\%$. Only pixels with disparity error less than 4 are shown.
This chapter covers the method used for generating the dataset accompanying this thesis. The chapter starts with reviewing related theory. Afterwards, a review of previous work pertaining to dataset generation for stereo matching is presented. Finally, the method used to create the dataset in this thesis is explained.

In the context of this thesis work, a dataset is a collection of stereo images accompanied by ground truth disparity maps.

### 3.1 Related Work

Generating datasets with ground truth for disparity estimation is a more difficult task than for classification, since accurate manual labelling is not feasible for most scenes [30].

In the literature it is very common to use datasets with sparse ground truth disparity maps such as KITTI [32–34], even when evaluating dense matching algorithms or refinement methods [16, 35].

There are datasets that have dense ground truth, but there are usually other types of problems with these datasets. In the case of the much used Middlebury dataset [1, 3], it is simply too small for many applications such as neural network training. Synthetic datasets such as FlyingThings3D [18] have explicit access to the ground truth, but are limited by the fact that they do not contain real images.

In his investigation into synthetic datasets, Mayer et al. found that there is great value in using synthetic datasets as a part of the training process [30]. They found that the results improved if a mix of synthetic and real data was used. While increasing the complexity of the objects in the scene for the training data did increase the performance in benchmarks, very simple objects with simple motions performed surprisingly well. A variety of textures was also found to be important.
The MPI-Sintel Dataset [36] was created from images from an open source movie. Although the dataset initially held ground truth for optical flow, disparity was also added at a later stage. Up until the introduction of the FlyingThings3D, as well as the Monkaa and Driving datasets in a paper by Mayer et al. [18], the MPI-Sintel Dataset was the largest dataset for disparity estimation. In their paper, they praise the MPI-Sintel Dataset as a robust and reliable dataset in terms of its ground truth.

The NYU Depth V2 dataset [37] focuses chiefly on object segmentation, assisted by depth data, rather than disparity estimation. The dataset consists of 1449 RGBD images of indoor scenes captured from a Microsoft Kinect. Disparity values are not explicitly included in the dataset, but can be acquired from the depth information. Still, considering that the dataset was made with a different purpose in mind, it is unclear if this dataset would be suitable as training data or ground truth for disparity estimation. This notion is further reinforced by the observation that depth images from a Kinect depth camera usually do not have high accuracy.

Similar conclusions can be made about the SUN3D dataset [38], which focuses on scene understanding. Instead of using a depth camera, structure-from-motion was used to create a 3D reconstruction of the observed scene, which yields depth information as a result.

A lesser known stereo disparity dataset is the one kept by the University of Glasgow, created as part of a research project related to automating the handling of fabrics [39]. The dataset is limited in its usability, however, since there is little diversity or variation in the images.

In a paper by Herakleous et al. [40], a structured-light scanning system is introduced and each step of its creation is detailed. The main contribution is the open-source software "3DUNDERWORLD-SLS", which implements the proposed techniques. To estimate the accuracy and robustness of the system, 4 evaluation metrics are proposed, and the output is compared to that of a high-end commercial 3D scanner. Other important contributions, that tie in well with this thesis, is their discussions on inherent difficulties of creating a structured-light 3D scanning system, and their proposed metrics for evaluating such a system.

## 3.2 Method Used

In this section, everything related to the method for generating the dataset is explained.

### 3.2.1 Structured Light

The method used for creating the dataset is a structured light based method used at Saab Dynamics. The exact algorithm being used is a trade secret. Due to the method being based on structured light, there are a few strengths and weaknesses that have to be considered. In order for a disparity value to be obtainable for a given pixel, a few requirements have to be met.
3.2 Method Used

Figure 3.1: The left camera has a very steep viewing angle of the side of the ball, causing poor measurements.

1. The landmark has to be visible from both cameras in the stereo pair
2. The landmark has to be shone upon by the structured light light source
3. The local area around the landmark has to be sufficiently clear

It is possible to fulfill all these requirements for most pixels in an image in a single measurement, but not all pixels, unless the scene is exceedingly simple. Requirement 1 and 2 could also be described as: both cameras and the light source must have an unobstructed view of the landmark. Requirement 3 is broken around edges, where the viewing angle of one of the cameras is too steep to discern the structure in the structured light (figure 3.1). Another way to describe requirement 3 is that the SNR near the landmark has to be sufficiently high. This is not true for darker areas, or areas where most of the structured light has been reflected off the surface of the object being measured.

The above paragraph describes the requirements needed for a disparity to be obtainable. However there is a distinct phenomenon that can cause faulty disparity values; reflections. There are cases where the structured light is reflected off a reflective surface A onto another surface B. This causes the observed structure in surface B to be corrupted, which can cause faulty disparities.

3.2.2 Rectification

In order to reduce the amount of computations needed to produce a disparity map for a stereo rig image pair, rectified images are produced. This is achieved by aligning the epipolar lines, making them parallel. This means that the disparities are non-zero along one axis, in this case the horizontal axis.

In a rectified stereo rig the epipoles are at a point in infinity, making the epipolar lines parallel. If the principal axes of both cameras are parallel, and they have the same roll rotation perpendicular to both the principal axes and the baseline,
the epipolar lines are on the same row in both images. This means that the
disparity map values are one-dimensional. In general however, the stereo cameras
are not perfectly aligned in this way, meaning that they are not rectified.

Cameras that share a common camera center are called equivalent, and 3D-
points projected on their respective image planes only differ by a homography
transformation. This means that a new camera can be constructed that satisfies
the requirements to be rectified, since the homography can transform the epipo-
lar lines to be aligned and parallel. The expression in (3.1) [41] shows the relation
between two equivalent cameras, where $H$ is the rectifying homography, $K_1$ and
$K_2$ are the intrinsic camera parameters of the two cameras, and $R$ is the relative
rotation between the cameras.

$$C_1' = HC_1 = K_2RK_1^{-1}$$  \hspace{1cm} (3.1)

One method for camera rectification consists of the following steps:

1. Remove lens distortions
2. Find an $R$ for each camera to make epipolar lines parallel
3. Equalize the FoV's to make the epipolar lines appear on the same row in
   both images

Point 1 can easily be done by calibrating the cameras followed by a recon-
struction of the image with the distortion parameters taken into consideration. A
possible rotation in point 2 can be found by constructing an orthonormal base
with the baseline as one of the axes. The other two axes can be chosen such that
they maximize the common viewing area. In (3.2), the basis vectors for such a
rotation is given, where $b$ is the baseline, $n_A$ and $n_B$ are the camera centers, and
$f_A$ and $f_B$ the optical axes for camera A and B.
3.2 Method Used

\[
\begin{align*}
  b &= \frac{n_B - n_A}{\|n_B - n_A\|} \\
  d &= \frac{(f_A + f_B) \times b}{\|(f_A + f_B) \times b\|} \\
  f &= b \times d
\end{align*}
\] (3.2)

Point 3 can be done by changing the internal parameters of the equivalent cameras. The FoV’s should be close to the original values to make sure the rectified images are not too zoomed in or out.

If the images were not approximately rectified before the rectification, it can be helpful to offset the principal point in the image, to minimize the pixels that end up outside the FoV due to the rectifying rotation. This offset has to be taken into account when converting between disparity values and depth values.

3.2.3 The Scenes and Final Setup

As mentioned by Hamzah and Ibrahim [5], a varied set of objects and textures in the scenes is preferable, especially for learning algorithms. However, the measuring equipment is fairly immobile, and the measuring itself takes place in a protected environment where filming is strictly regulated. This unfortunately means that the scenes are limited to what the camera is permitted to be pointed at. In concrete terms, the scenes are indoors and contain various objects found in the office. Due to limitations in the structured light algorithm itself, the possible set of textures and colours in one scene is also limited to non-reflective surfaces, and colours with similar light absorbing properties.

Each scene is measured using three stereo pairs, consisting of a total of four cameras mounted on a rail. Two measurements are done for each stereo pair; one for each position of the light source. The light source is placed either on the left or the right side of the cameras, in order to provide good lighting from all views. An illustration of the measuring setup can be found in figure 3.3. The reference stereo pair refers to the stereo pair made up from the two center cameras. The coordinate system is based upon the position and orientation of the left reference camera. The name is chosen because they are the stereo pair which the merged disparities explained in 3.2.4 are described. The resolution of the cameras is 1440x1088 pixels.
As mentioned in 3.2.1, areas around edges have poor measurements, and this remains true even if the position of the light source is changed. In order to fill in the edges with valid values, another stereo pair is required. Optimally, there should be a stereo pair left, right, above and below the reference stereo pair, in order to provide good measurements from all angles. This would likely also require additional positions of the light source to sufficiently illuminate all visible surfaces. This is not very practical for several reasons, such as availability of cameras, additional manual labour due to moving the light source as well as extra time spent of doing measurements for one scene means a smaller total of scenes created. For comparison, in the Middlebury dataset [3], they used 12 projector positions for the Motorcycle scene, which could help explain the small size of the dataset. The setup mentioned above seems to be a sweet spot between time and labour spent on one scene, and additional density gained in the merged disparity map. Due to only four cameras being available for use, only edges from the right direction can be filled in, since there is a distinct stereo pair to the right of the left reference camera. Since the left reference camera is part of the leftmost stereo pair, there is no stereo pair that has good measurements where the measurements of the left reference camera are poor. Another camera would be needed to fill in edges from the left. Areas with missing measurements are referred to as holes in the disparity map.

### 3.2.4 Merging of Disparities

One way to fill the holes that occur due camera occlusions and shadows is to use multiple measurements. The shadows can be filled in by moving the light source.
source, and does not require a different camera pose. In order to fill in camera occlusions and near-edge areas however, a different pose is needed, as shown in figure 3.4a. The disparity information from the new poses can then be warped to the original pose, where we want to fill in the holes. There are a few things to take into consideration when doing this:

1. There can be several conflicting disparity values for a given pixel, due to background objects being visible from other stereo pairs.

2. Inlier values are affected by measuring noise, and do not have exactly the same disparity values.

3. Areas with missing disparity values from the reference stereo pair may get disparity values from a background object visible from the other stereo pair, as illustrated in figure 3.4b.

![Diagram](image.png)

(a) Situation where a previously occluded object can be measured by adding another stereo pair where the object is visible from both cameras.

(b) The left stereo pair fills a hole in the disparity map from an object which is not in the foreground in the reference camera (marked in black). This can happen if the right stereo pair has a poor measurement of the foreground object.

**Figure 3.4:** Illustration of situations that are solved or are problematic, that appear when introducing a merge from several stereo pairs. Reference stereo pair marked in black and gray.

There are other works where similar problems are handled. In a paper by Newcombe et al. on KinectFusion [42], a volumetric integration method, described by Brian Curless and Marc Levoy [43], is used in favor of point cloud fusion. This method transfers depth values to a voxelized 3D-space. In order to avoid further quantization noise, a very dense voxel field would be required for our dataset. The machine learning network OctNetFusion [44] is also based on voxelization and suffers from the same issue. As mentioned by Riegler et al. [44], volumetric integration methods also typically require a large amount of input views, which we do not have.
In order for the situation in figure 3.4b to occur we need to both have poor measurements of the foreground object, and successful measurements of the background object. The above methods are designed to handle situations where this is a fairly common occurrence, such as when measurements may have been done from all angles. This phenomenon is much more likely to happen if we have stereo pairs in very different positions where the background object is visible. If the distance between the stereo pairs is small compared to the distance to the scene, only areas close to the edges of objects are at risk of being corrupted by background disparities, since none of the stereo cameras can see the background object.

The ratio between the distance between the stereo pairs and the distance to the scene is around 1 : 15. Due to this, and since we only have 3 views, we use a simple approach to merging, which also helps with attenuation of noise in regions with several measurements. In areas where the measured disparity differs greatly, the measurements with smaller disparities are discarded. The final disparity value is obtained by averaging the remaining disparity values, as shown in figure 3.5.

![Figure 3.5: Illustration of the merging process for a given pixel. Each cross represents a measurement from one of the views. Small disparities (background objects) are discarded in favor of an averaging of larger disparities (foreground objects) in an interval.](image)

Due the merging process being fairly simple, it does at this stage not possess the same level of robustness to faulty disparities from background objects where the foreground object has poor measurements, as some of the volumetric integration methods brought up in [44]. A method to tackle this is therefore needed. The proposed method is to compare the RGB values of the reference image to those of a warped RGB image from a different stereo pair. If the disparity values are correct, then the RGB values are also expected to be very similar, with interpolation effects from the warp transformation likely being the largest source of difference. If the disparities stem from a background object however, the RGB values in the warped image will come from the background object. Unless the foreground object and the background object are chromatically very similar, we should be able to identify faulty disparities. The obvious limit of this method is that it can not handle the case where the background object and the foreground object are chromatically similar. We refer to this as chromatic consistency filtering. The discriminating function is presented below as $f(x)$, where $x$ is some pixel where we want to check the chromatic consistency. $I_{\text{ref}}(x_i)$ and $I_{\text{side}}(x_i)$ is an RGB value in the reference image and the warped image from another view,
3.2 Method Used

in some pixel coordinate \( x_i \), where \( x_i \) is some pixel from the set \( \Omega(x) \) of pixels neighbouring \( x \).

\[
f(x) = \begin{cases} 
1 & \text{if } g(x) \leq \epsilon \\
0 & \text{if } g(x) > \epsilon 
\end{cases}
\]

where \( g(x) = \left\| \sum_{x_i \in \Omega(x)} (I_{ref}(x_i) - I_{side}(x_i)) \right\| \) (3.3)

The dataset consists of the non-warped disparities from each stereo pair, which should contain a minimal amount of outliers. The merged disparities warped from several views may contain outliers where the foreground object precedence and chromatic consistency filtering fails, but they are also more dense. These are added to the dataset as separate disparity maps, and the difference between these sets are evaluated.

3.2.5 Evaluation

The dataset is meant to be usable as ground truth, meaning that we must have a high confidence that the disparities that we generate actually agree with the scenes we measured. In the case of normal stereo algorithms, one may compare the performance of the algorithm to a ground truth dataset, but that is obviously not possible here. Instead we have to measure objects with known geometry and compare these with the measured geometry. To accomplish this, a few metrics are brought forward for use in this comparison with different objects. These metrics are largely based on the evaluation metrics used by Herakleous et al. [40], but with some additions.

1. Linearity
2. Orthogonality
3. Accuracy
4. Calibration board comparison

The linearity metric is procured by plane fitting a flat object, and calculating the mean absolute error (MAE) (3.4) and the root mean square error (RMSE) (3.5) of the residuals. In the equations below \( N \) denotes the number of samples used in the averaging, and \( \epsilon_i \) is a residual value. The normal of the estimated plane is denoted as \( n \), which has its shortest distance \(-\Delta\) from the origin. Finally, \( x_i \) is a 3D point corresponding to the residual value \( \epsilon_i \).

\[
MAE = \frac{1}{N} \sum_{i} |\epsilon_i| = \frac{1}{N} \sum_{i} \left| \begin{bmatrix} n \\ -\Delta \end{bmatrix} \cdot \begin{bmatrix} x_i \\ 1 \end{bmatrix} \right| \quad (3.4)
\]
RMSE = \sqrt{\frac{1}{N} \sum_{i}^{N} \varepsilon^2_i} \tag{3.5}

The orthogonality metric can be acquired by taking the scalar product of estimated normals of a cubic object, and calculating the deviation from the expected 90 deg angle between the normals \(3.6\). The normals of the two planes compared are denoted \(\mathbf{n}_i\) and \(\mathbf{n}_j\), and the angle between them is \(\theta_{i,j}\).

\[\mathbf{n}_i \cdot \mathbf{n}_j = \cos \theta_{i,j} \implies \theta_{i,j} = \arccos (\mathbf{n}_i \cdot \mathbf{n}_j)\]

\[E_{\text{ortho, deg},i,j} = |\theta_{i,j} - 90^\circ|\] \tag{3.6}

The accuracy metric is simply calculated by comparing the measured length of an object to its true length. The object is measured several times and the mean error of the measured value and the known value \(3.7\) is calculated. The relative error is also calculated \(3.8\). \(L_{\text{known}}\) and \(L_{\text{meas},i}\) denotes the known length of an object and its measured length.

\[E_{\text{acc}} = \frac{1}{N} \sum_{i}^{N} L_{\text{meas},i} - L_{\text{known}}\] \tag{3.7}

\[E_{\text{relative}} = \frac{E_{\text{acc}}}{L_{\text{known}}}\] \tag{3.8}

The final metric, calibration board comparison, is a kind of extension of the linearity metric and uses the residuals from a plane fitting of a calibration board obtained when calibrating the camera in an earlier stage. These residuals are spatially very sparse, as only the corners of the chess tiles on the calibration board were included in the plane fitting, but the measurements themselves can be considered reliable since around 20-30 measurements per camera were made. The calibration board is not perfectly flat, and has some curvature. This final metric is an attempt to evaluate the ability of the structured light algorithm to reproduce the minute deformations of the calibration board. In order to make the comparison with the dense residuals from the structured light measurements, the

\textbf{Figure 3.6:} Illustration of the calibration board comparison, as seen from the side. The straight line shows the estimated plane. The curved line shows the actual geometry of the calibration board, which is exaggerated for clarity. The arrows illustrate the residuals of the plane fit with the calibration board.
calibration measurement residuals are upscaled, by applying a diffusion on the original residuals. Figure 3.6 shows a simplified illustration of the calibration board plane fitting. The upscaling of the sparse residuals can be seen as interpolation between the arrows in order to fill in the missing data. The difference between this metric and the linearity metric is that in the linearity metric, the object is expected to be perfectly flat, which corresponds to all of the residuals being 0. In the calibration comparison, the residuals from the plane fitting are not expected to be 0, but instead the difference of the plane fitting residuals from the calibration stage and the generated scene is expected to be 0.

The metrics used for the calibration board comparison are like MAE and RMSE in the linearity metric, but instead there is a known value $\epsilon_{\text{diff},i}$ from the calibration measurements and the structured light measurements, as shown in equations (3.9) and (3.10).

$$MAE = \frac{1}{N} \sum_{i} |\epsilon_i - \epsilon_{\text{diff},i}|$$ (3.9)

$$RMSE = \sqrt{\frac{1}{N} \sum_{i} (\epsilon_i - \epsilon_{\text{diff},i})^2}$$ (3.10)

The above metrics are used for evaluation in the cases where a single stereo measurement is used, and where the disparity merging has been performed. This lets us evaluate the noise attenuating effects of the averaging.
In this chapter, the results of this thesis are presented.

4.1 Disparity Refinement

This section presents the evaluation results pertaining to the disparity refinement methods described in chapter 2. These methods are:

- Method 1: Refinement Network [2]
- Method 2: Superpixel-Based Refinement [15]
- Method 3: Baseline (Inpainting) [26]

This section is split into 4 parts. In the first part, we present a quantitative evaluation of the results of the pre-training of method 1. All 15 training regimes (5 different network versions, 3 different inputs) as described in section 2.6.3 are presented. This is followed by a short summary of observations that can be made from the results. In parts 2 and 3, we present the results of the evaluations of all three methods (sections 2.6, 2.7 and 2.8) on the dataset created in this thesis, and the training images of Middlebury V3 [1]. For method 1, only the networks that were among the 5 best performing during the pre-training are chosen for evaluation. In the forth and final part, we summarize the results presented in parts 2 and 3, and present further comparisons that highlights some implications of the presented results.

The evaluation metrics used are MAE, RMSE, bad4 and bad2, as explained in section 2.9. We also explained the dividing of pixels in the ground truth into three categories, which allows us to evaluate the methods on the following sub-areas: preservation of correct disparities (category 1), refinement of incorrect dis-
parities (category 2) and hole filling (category 3). An illustration of this categorization is shown in figure 2.9. For visualization, we use the EPE-4 and EPE-20 error maps defined in section 2.9.

### 4.1.1 Pre-Training Results

Here, we present the evaluation results of the pre-training of the different network versions of method 1. The evaluation is performed on the FlyingThings3D pre-training validation set as described in section 2.6.3. As also mentioned in section 2.6.3, the 5 different network versions were trained using the three different inputs explained in section 2.4:

- Saab Initial Disparity, using **thresholded** confidence values.

- Saab Initial Disparity, using **scaled** confidence values.

- Synthetic Initial Disparity

The results of the pre-training are presented in tables 4.1 to 4.3. Each table pertains to different initial disparity. Table 4.1 pertains to Saab Initial Disparity using **thresholded** confidence values, table 4.2 to Saab Initial Disparity using **scaled** confidence values, and table 4.3 to Synthetic Initial Disparity. Since this section is related to the pre-training of method 1, method 2 is not shown for comparison in these tables, but is included in the main evaluation results of sections 4.1.2 and 4.1.3. The tables are followed by figure 4.1, showing a comparison of outputs from 5 different network versions.

**Table 4.1: Pre-training Results - Saab (Thresh. Conf.)**

<table>
<thead>
<tr>
<th>Architecture</th>
<th>Loss</th>
<th>MAE</th>
<th>RMSE</th>
<th>bad4</th>
<th>bad2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enc-Dec</td>
<td>L2</td>
<td>11.081</td>
<td>22.347</td>
<td>39.28</td>
<td>49.17</td>
</tr>
<tr>
<td>SmoothL1</td>
<td><strong>4.450</strong></td>
<td><strong>11.817</strong></td>
<td><strong>18.62</strong></td>
<td><strong>25.33</strong></td>
<td></td>
</tr>
<tr>
<td>MS</td>
<td>L2</td>
<td>6.448</td>
<td>13.853</td>
<td>30.52</td>
<td>40.46</td>
</tr>
<tr>
<td>SmoothL1</td>
<td>5.296</td>
<td>13.038</td>
<td>23.19</td>
<td>30.09</td>
<td></td>
</tr>
<tr>
<td>Unguided</td>
<td>ConfLoss</td>
<td>10.274</td>
<td>20.832</td>
<td>38.19</td>
<td>48.82</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method 3: Baseline [26]</th>
<th>Loss</th>
<th>MAE</th>
<th>RMSE</th>
<th>bad4</th>
<th>bad2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>11.724</td>
<td>24.128</td>
<td>30.68</td>
<td>34.68</td>
<td></td>
</tr>
</tbody>
</table>
4.1 Disparity Refinement

Table 4.2: Pre-training Results - Saab (Scaled Conf.)

<table>
<thead>
<tr>
<th>Method 1: Metrics</th>
<th>Architecture</th>
<th>Loss</th>
<th>MAE</th>
<th>RMSE</th>
<th>bad4</th>
<th>bad2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enc-Dec</td>
<td>L2</td>
<td>7.231</td>
<td>14.961</td>
<td>32.47</td>
<td>44.64</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SmoothL1</td>
<td>4.642</td>
<td>12.298</td>
<td>18.54</td>
<td>25.32</td>
<td></td>
</tr>
<tr>
<td>MS</td>
<td>L2</td>
<td>4.832</td>
<td><strong>10.283</strong></td>
<td>26.27</td>
<td>37.21</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SmoothL1</td>
<td>4.248</td>
<td>10.982</td>
<td>20.42</td>
<td>26.50</td>
<td></td>
</tr>
<tr>
<td>Unguided</td>
<td>ConfLoss</td>
<td>10.203</td>
<td>21.069</td>
<td>35.67</td>
<td>43.84</td>
<td></td>
</tr>
</tbody>
</table>

Method 3: Baseline [26] | 11.724| 24.128| 30.68| 34.68|

Table 4.3: Pre-training Results - Synthetic (Thresh. Conf.)

<table>
<thead>
<tr>
<th>Method 1: Metrics</th>
<th>Architecture</th>
<th>Loss</th>
<th>MAE</th>
<th>RMSE</th>
<th>bad4</th>
<th>bad2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enc-Dec</td>
<td>L2</td>
<td>1.242</td>
<td><strong>3.740</strong></td>
<td>4.93</td>
<td>10.40</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SmoothL1</td>
<td><strong>1.015</strong></td>
<td>4.166</td>
<td><strong>3.46</strong></td>
<td><strong>6.62</strong></td>
<td></td>
</tr>
<tr>
<td>MS</td>
<td>L2</td>
<td>1.623</td>
<td>4.186</td>
<td>5.95</td>
<td>13.49</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SmoothL1</td>
<td>1.175</td>
<td>4.461</td>
<td>4.16</td>
<td>7.59</td>
<td></td>
</tr>
<tr>
<td>Unguided</td>
<td>ConfLoss</td>
<td>2.562</td>
<td>8.568</td>
<td>8.14</td>
<td>13.86</td>
<td></td>
</tr>
</tbody>
</table>

Method 3: Baseline [26] | 2.541| 17.267| 6.27| 9.18|

In the case of Saab Initial Disparity using thresholded confidences, the Enc-Dec using the SmoothL1 loss function performs the best in all metrics, as shown in table 4.1. For the scaled confidences of table 4.2, however, MS performs the best on MAE and RMSE. We conclude that the type of confidence does not drastically affect the results in the case of Saab Initial Disparity.

The results when using Synthetic Initial Disparity are significantly better compared to Saab’s, as shown in table 4.3. For example, the MAE, bad4 and bad2 metrics are reduced by a factor of 4. This is expected since this initial disparity is essentially sparsely sampled ground truth with added noise, and is thus much more well-behaved than Saab’s. Once again Enc-Dec using the SmoothL1 loss performs the best overall. A pattern that can be seen in all three tables is that the guided networks outperform the unguided one as well as the baseline method in all cases.
Figure 4.1 provides illustrations of predictions made by 5 different network versions. The unguided network is clearly inferior to the guided versions, which is expected since it is merely a intermediate step upon which the guided architectures are based. As shown in the rightmost row, using synthetic initial disparity provides the best results. From this we conclude that the most important characteristic of initial disparity is not measurement density, but accuracy and good distribution over key areas of the image. The synthetic initial disparity is sparse, but it always gives correct disparities. Saab Initial Disparity, on the other hand, while being more dense overall, contains a higher ratio of incorrect disparities. Further, there are often large areas that have no disparity values at all.
4.1.2 Evaluation Results: Own Dataset

Tables 4.4 through 4.7 show the evaluation results of the three methods on our dataset. In the case of method 1, only the 5 network versions that performed the best during pre-training in each input category are shown. As explained in section 2.9, the evaluation is split up into the 3 categories (see figure 2.9) and evaluated separately.

Tables 4.4, 4.5 and 4.6 pertain to categories 1, 2 and 3 respectively. Table 4.7 shows the evaluation results when the entire ground truth is taken into account. The best performing version of method 1 is highlighted in bold. Each table is followed by a figure (figures 4.2 through 4.5) providing illustrative comparisons of the methods.

**Table 4.4: Results on Our Dataset - Category 1 (Preservation of Correct Disparities)**

<table>
<thead>
<tr>
<th>Method 1 (Best Performing Versions)</th>
<th>Metrics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pre-trained On</td>
<td>Network</td>
</tr>
<tr>
<td>Saab (thr. conf.)</td>
<td>Enc-Dec</td>
</tr>
<tr>
<td>Saab (sc. conf.)</td>
<td>Enc-Dec</td>
</tr>
<tr>
<td>Saab (sc. conf.)</td>
<td>MS</td>
</tr>
<tr>
<td>Saab (sc. conf.)</td>
<td>MS</td>
</tr>
<tr>
<td>Synth. (thr. conf.)</td>
<td>Enc-Dec</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Non-learning methods</th>
<th>Metrics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>MAE</td>
</tr>
<tr>
<td>Method 2: Superpixel-Based [15]</td>
<td>0.669</td>
</tr>
<tr>
<td>Method 3: Baseline (Inpainting) [26]</td>
<td>0.442</td>
</tr>
</tbody>
</table>

**Figure 4.2: Method-by-method comparison of performance results in Category 1 (Preservation of Correct Disparities) on our dataset, in the form of EPE-4 error maps from the results of each method. Only the pixels which fall into this category are shown. (a) and (b) show the best and worst performing versions of method 1 (in the MAE sense), (c) shows method 2 and (d) shows method 3. The reference image can be seen in figure 4.3e.**
Table 4.5: Results on Our Dataset - Category 2 (Refinement of Incorrect Disparities)

<table>
<thead>
<tr>
<th>Method 1 (Best Performing Versions)</th>
<th>Pre-trained On</th>
<th>Network</th>
<th>Loss</th>
<th>MAE</th>
<th>RMSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Saab (thr. conf.)</td>
<td>Enc-Dec</td>
<td>SmoothL1</td>
<td>8.845</td>
<td>11.764</td>
<td></td>
</tr>
<tr>
<td>Saab (sc. conf.)</td>
<td>Enc-Dec</td>
<td>SmoothL1</td>
<td>8.665</td>
<td>11.420</td>
<td></td>
</tr>
<tr>
<td>Saab (sc. conf.)</td>
<td>MS</td>
<td>L2</td>
<td>9.845</td>
<td>12.647</td>
<td></td>
</tr>
<tr>
<td>Saab (sc. conf.)</td>
<td>MS</td>
<td>SmoothL1</td>
<td>9.968</td>
<td>12.914</td>
<td></td>
</tr>
<tr>
<td>Synth. (thr. conf.)</td>
<td>Enc-Dec</td>
<td>SmoothL1</td>
<td>10.353</td>
<td>13.077</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Non-learning methods</th>
<th>Name</th>
<th>MAE</th>
<th>RMSE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Method 3: Baseline (Inpainting) [26]</td>
<td>10.729</td>
<td>13.371</td>
</tr>
</tbody>
</table>

Figure 4.3: Method-by-method comparison of performance results in Category 2 (Refinement of Incorrect Disparities) on our dataset, in the form of EPE-20 error maps from the results of each method. Only the pixels which fall into this category are shown. (a) and (b) show the best and worst performing versions of method 1 (in the MAE sense), (c) shows method 2 and (d) shows method 3. (e) is the reference image.
### Table 4.6: Results on Our Dataset - Category 3 (Hole Filling)

<table>
<thead>
<tr>
<th>Pre-trained On</th>
<th>Network</th>
<th>Loss</th>
<th>MAE</th>
<th>RMSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Saab (thr. conf.)</td>
<td>Enc-Dec</td>
<td>SmoothL1</td>
<td>4.057</td>
<td>8.150</td>
</tr>
<tr>
<td>Saab (sc. conf.)</td>
<td>Enc-Dec</td>
<td>SmoothL1</td>
<td>3.975</td>
<td>8.116</td>
</tr>
<tr>
<td>Saab (sc. conf.)</td>
<td>MS</td>
<td>L2</td>
<td>6.107</td>
<td>9.800</td>
</tr>
<tr>
<td>Saab (sc. conf.)</td>
<td>MS</td>
<td>SmoothL1</td>
<td>4.641</td>
<td>9.441</td>
</tr>
<tr>
<td>Synth. (thr. conf.)</td>
<td>Enc-Dec</td>
<td>SmoothL1</td>
<td><strong>3.752</strong></td>
<td><strong>7.592</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>MAE</th>
<th>RMSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method 2: Superpixel-Based [15]</td>
<td>3.331</td>
<td>10.194</td>
</tr>
<tr>
<td>Method 3: Baseline (Inpainting) [26]</td>
<td>4.615</td>
<td>10.678</td>
</tr>
</tbody>
</table>

Figure 4.4: Method-by-method comparison of performance results in Category 3 (Hole Filling) on our dataset, in the form of EPE-20 error maps from the results of each method. Only the pixels which fall into this category are shown. (a) and (b) show the best and worst performing versions of method 1 (in the MAE sense), (c) shows method 2 and (d) shows method 3. (e) is the reference image.
Table 4.4 shows that the Enc-Dec using the SmoothL1 loss function, pre-trained on Synthetic initial disparity, is best at category 1 (preservation of correct disparities). It is only marginally better than the baseline method, which practically does not attempt any corrections to these disparities at all. The table also shows that method 2, as well as the MS using the L2 loss function, deteriorates disparities of this category. The illustrations provided by figure 4.2 reinforce the fact that these two approaches are inferior to the best performing network as well as the baseline. We conclude that method 1 preserves correct disparities while method 2 deteriorates them slightly.

Regarding category 2 (refinement of incorrect disparities), table 4.5 shows that the Enc-Dec using the SmoothL1 loss function, pre-trained on Saab initial disparity, performs the best among the network versions of method 1. Its results are better than the baseline, but inferior to method 2. Figure 4.3 highlights that incorrect disparities are mainly along depth discontinuities in the image, and shows that no method is able to refine these areas perfectly. Method 2 has the best performance in this category. This is likely due to its super-pixel segmentation, which has the ability to enforce a sharp disparity discontinuity at correct places, provided a successful segmentation in those areas.

In category 3 (hole filling), the Enc-Dec using SmoothL1, pre-trained on the Synthetic initial disparity, shows the best performance among the network versions. This is apparent from table 4.6. Further, we conclude that the results from the best version of method 1 are comparative to those of method 2, since method 2 has superior MAE, while method 1 has superior RMSE. Both method 1 and method 2 outperform the baseline method. Illustrations provided in figure 4.4 show that all methods manage to fill in trivial holes such as the brown wall in the background, and that the difference instead lies mainly with disparities close to depth discontinuities.

**Table 4.7: Results on Our Dataset - All Categories (Whole Ground Truth)**

<table>
<thead>
<tr>
<th>Method 1 (Best Performing Versions)</th>
<th>Metrics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pre-trained On</td>
<td>Network</td>
</tr>
<tr>
<td>Saab (thr. conf.)</td>
<td>Enc-Dec</td>
</tr>
<tr>
<td>Saab (sc. conf.)</td>
<td>Enc-Dec</td>
</tr>
<tr>
<td>Saab (sc. conf.)</td>
<td>MS</td>
</tr>
<tr>
<td>Saab (sc. conf.)</td>
<td>MS</td>
</tr>
<tr>
<td>Synth. (thr. conf.)</td>
<td>Enc-Dec</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Non-learning methods</th>
<th>Metrics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>MAE</td>
</tr>
<tr>
<td>Method 3: Baseline (Inpainting) [26]</td>
<td>1.890</td>
</tr>
</tbody>
</table>
4.1 Disparity Refinement

![Comparison of output from the three methods on an image from our dataset. The rows from top-to-bottom correspond to method 1 (Enc-Dec SmoothL1, pre-trained on Synthetic initial disparity), method 2 and method 3. (a) shows the initial disparity used, (b) the output, (c) the error map thresholded at 20 pixels, and (d) the error map thresholded at 4 pixels. (e) shows the input image and ground truth for reference.](image)

**Figure 4.5:** Comparison of output from the three methods on an image from our dataset. The rows from top-to-bottom correspond to method 1 (Enc-Dec SmoothL1, pre-trained on Synthetic initial disparity), method 2 and method 3. (a) shows the initial disparity used, (b) the output, (c) the error map thresholded at 20 pixels, and (d) the error map thresholded at 4 pixels. (e) shows the input image and ground truth for reference.

Table 4.7 shows that the network version that performs the best on our dataset overall is the Enc-Dec with SmoothL1 loss function, pre-trained on Synthetic initial disparity. It outperforms methods 2 as well as the baseline method in the MAE and RMSE metrics, but suffers in the bad4 and bad2 metrics. Method 2 outperforms both other methods on bad4 and bad2. Looking at the comparisons in figure 4.5, we observe that the EPE-4 map corresponding to method 2 show significantly fewer incorrect disparities around the yellow object to the right of the image. We attribute the success of method 2 in the bad4 and bad2 metrics to the methods success in correctly segmenting the image.
4.1.3 Evaluation Results: Middlebury V3

Tables 4.8 through 4.11 show the evaluation results of the three methods on the training images of the Middlebury V3 dataset [3]. In the case of method 1, only the 5 network versions that performed the best during pre-training in each input category are shown. As explained in section 2.9, the evaluation is split up into the 3 categories (see figure 2.9) and evaluated separately.

Tables 4.8, 4.9 and 4.10 pertain to categories 1, 2 and 3 respectively. Table 4.11 shows the evaluation results when the entire ground truth is taken into account. Each table is followed by a figure (figures 4.6 through 4.9) providing illustrative comparisons of the methods. Finally, table 4.12 shows a comparison of how the methods perform on 7 individual items in the dataset: Adirondack, ArtL, Jade-plant, Motorcycle, Playroom, Playtable, and Teddy.

Table 4.8: Results on Middlebury V3 [3] - Category 1 (Preservation of Correct Disparities)

<table>
<thead>
<tr>
<th>Method 1 (Best Performing Versions)</th>
<th>Metrics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pre-trained On</td>
<td>Network</td>
</tr>
<tr>
<td>Saab (thr. conf.)</td>
<td>Enc-Dec</td>
</tr>
<tr>
<td>Saab (sc. conf.)</td>
<td>Enc-Dec</td>
</tr>
<tr>
<td>Saab (sc. conf.)</td>
<td>MS</td>
</tr>
<tr>
<td>Synth. (thr. conf.)</td>
<td>Enc-Dec</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Non-learning methods</th>
<th>Metrics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>MAE</td>
</tr>
<tr>
<td>Method 2: Superpixel-Based [15]</td>
<td>0.804</td>
</tr>
<tr>
<td>Method 3: Baseline (Inpainting) [26]</td>
<td>0.441</td>
</tr>
</tbody>
</table>

Figure 4.6: Method-by-method comparison of performance results in Category 1 (Preservation of Correct Disparities) on Middlebury V3, in the form of EPE-4 error maps from the results of each method. Only the pixels which fall into this category are shown. (a) and (b) show the best and worst performing versions of method 1 (in the MAE sense), (c) shows method 2 and (d) shows method 3. The reference image can be seen in figure 4.7e.
Table 4.9: Results on Middlebury V3 [3] - Category 2 (Refinement of Incorrect Disparities)

<table>
<thead>
<tr>
<th>Method 1 (Best Performing Versions)</th>
<th>Metrics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pre-trained On</td>
<td>Network</td>
</tr>
<tr>
<td>Saab (thr. conf.)</td>
<td>Enc-Dec</td>
</tr>
<tr>
<td>Saab (sc. conf.)</td>
<td>Enc-Dec</td>
</tr>
<tr>
<td>Saab (sc. conf.)</td>
<td>MS</td>
</tr>
<tr>
<td>Saab (sc. conf.)</td>
<td>MS</td>
</tr>
<tr>
<td>Synth. (thr. conf.)</td>
<td>Enc-Dec</td>
</tr>
</tbody>
</table>

Non-learning methods

<table>
<thead>
<tr>
<th>Metrics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>Method 3: Baseline (Inpainting) [26]</td>
</tr>
</tbody>
</table>

Figure 4.7: Method-by-method comparison of performance results in Category 2 (Refinement of Incorrect Disparities) on Middlebury V3, in the form of EPE-20 error maps from the results of each method. Only the pixels which fall into this category are shown. (a) and (b) show the best and worst performing verisons of method 1 (in the MAE sense), (c) shows method 2 and (d) shows method 3. (e) is the reference image.
Table 4.10: Results on Middlebury V3 [3] - Category 3 (Hole Filling)

<table>
<thead>
<tr>
<th>Method 1 (Best Performing Versions)</th>
<th>Metrics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pre-trained On</td>
<td>Network</td>
</tr>
<tr>
<td>Saab (thr. conf.)</td>
<td>Enc-Dec</td>
</tr>
<tr>
<td>Saab (sc. conf.)</td>
<td>Enc-Dec</td>
</tr>
<tr>
<td>Saab (sc. conf.)</td>
<td>MS</td>
</tr>
<tr>
<td>Saab (sc. conf.)</td>
<td>MS</td>
</tr>
<tr>
<td>Synth. (thr. conf.)</td>
<td>Enc-Dec</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Non-learning methods</th>
<th>Metrics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>MAE</td>
</tr>
<tr>
<td>Method 2: Superpixel-Based [15]</td>
<td>15.149</td>
</tr>
<tr>
<td>Method 3: Baseline (Inpainting) [26]</td>
<td>15.581</td>
</tr>
</tbody>
</table>

Figure 4.8: Method-by-method comparison of performance results in Category 3 (Hole Filling) on Middlebury V3, in the form of EPE-20 error maps from the results of each method. Only the pixels which fall into this category are shown. (a) and (b) show the best and worst performing versions of method 1 (in the MAE sense), (c) shows method 2 and (d) shows method 3. (e) is the reference image.
Table 4.8 shows that the Enc-Dec using SmoothL1, pre-trained on Synthetic initial disparity, is the best performing network version in category 1 (refinement of incorrect disparities). However, the best performing network version of method 1 still performs worse than the baseline method, albeit only slightly. Method 2 is the worst performing method in this category. We conclude that method 1 deteriorates correct disparities slightly, while method two deteriorates them significantly. From the illustrations in figure 4.6 we observe that the deterioration of disparities by the best performing network version of method 1 is visually negligible. We also observe that method 2 can suffer greatly if the segmentation fails, as seen by the incorrect disparities in the upper right portion of figure 4.6c.

In category 2 (refinement of incorrect disparities), the Enc-Dec using SmoothL1 loss, pre-trained on Saab initial disparity, performs the best. Its results are better than those of the baseline method, but worse than method 2. That being said, the $MAE$ and $RMSE$ values of all three methods are rather large, so although methods 1 and 2 are comparatively better than the baseline, we can not conclude that either of the two methods corrects the incorrect disparities to a satisfactory level. This is further reinforced in figure 4.7, which shows that the vast majority of incorrect pixels remain incorrect.

As shown in table 4.10, the Enc-Dec using SmoothL1, pre-trained on Saab initial disparity with scaled confidence, shows the best performance among the network versions in category 3 (hole filling). This performance is however inferior to those of method 2 and the baseline. Method 2 also struggles, having very similar $MAE$ and $RMSE$ metrics to the baseline method. Figure 4.8 shows that essentially all methods struggle greatly with hole filling. We conclude that the images are generally very challenging in the Middlebury V3 dataset, and that this is the cause of the difficulty shown by both methods to fill in holes.

| Table 4.11: Results on Middlebury V3 [3] - All Categories (Whole Ground Truth) |
|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|
| Method 1 (Best Performing Versions) | Metrics | Pre-trained On | Network | Loss | MAE | RMSE | bad4 | bad2 |
| Saab (thr. conf.) | Enc-Dec | SmoothL1 | 18.788 | 35.880 | 37.72 | 44.41 |
| Saab (sc. conf.) | Enc-Dec | SmoothL1 | 15.748 | 30.972 | 36.23 | 42.93 |
| Saab (sc. conf.) | MS | L2 | 18.736 | 33.933 | 43.31 | 54.66 |
| Saab (sc. conf.) | MS | SmoothL1 | 18.387 | 35.478 | 36.73 | 43.30 |
| Synth. (thr. conf.) | Enc-Dec | SmoothL1 | 19.705 | 39.474 | 37.70 | 43.67 |

<table>
<thead>
<tr>
<th>Non-learning methods</th>
<th>Metrics</th>
<th>Name</th>
<th>MAE</th>
<th>RMSE</th>
<th>bad4</th>
<th>bad2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method 3: Baseline (Inpainting) [26]</td>
<td>10.318</td>
<td>19.356</td>
<td>31.23</td>
<td>37.95</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Figure 4.9: Comparison of output from the three methods on the ArtL image from the Middlebury V3 dataset. The rows from top-to-bottom correspond to method 1, method 2 and method 3. (a) shows the initial disparity used, (b) the output, (c) the error map thresholded at 20 pixels, and (d) the error map thresholded at 4 pixels. (e) shows the input image and ground truth for reference.

Table 4.11 shows that the network version that performs the best on the Middlebury V3 dataset overall is the Enc-Dec with SmoothL1 loss function, pre-trained on Saab initial disparity with scaled confidences. The best method is however method 2, which shows a notable reduction in the bad4 and bad2 metrics. Method 1 shows approximately 50% worse results compared to the baseline across all four metrics. We attribute the poor performance of method 1 to the fact that the networks have never been trained on any images from the Middlebury V3 dataset, and that the images from this dataset are generally more challenging.
than (or at the very least quite different compared to) those used during training. Still, as shown in figure 4.9, method 1 does generalize to some extent, managing to recover the shape of the rings better than the other two methods.

**Table 4.12**: Individual evaluations on 7 items from Middlebury V3 [3]. Method 1 is the network version denoted Enc-Dec(SmoothL1), pre-trained on Saab(scaled conf).

<table>
<thead>
<tr>
<th>Metric: MAE</th>
<th>Name</th>
<th>Adiron</th>
<th>ArtL</th>
<th>Jadepl</th>
<th>Motor</th>
<th>Playr</th>
<th>Playt</th>
<th>Teddy</th>
</tr>
</thead>
</table>

|----------------------|-----------|--------|-------|--------|-------|-------|-------|-------|

<table>
<thead>
<tr>
<th>Metric: bad4</th>
<th>Method 1</th>
<th>26.97</th>
<th>35.15</th>
<th>67.77</th>
<th>29.29</th>
<th>40.88</th>
<th>31.59</th>
<th>16.33</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method 2</td>
<td>15.06</td>
<td>32.45</td>
<td>63.08</td>
<td>18.85</td>
<td>35.43</td>
<td>24.10</td>
<td>10.29</td>
<td></td>
</tr>
<tr>
<td>Method 3</td>
<td>22.07</td>
<td>36.32</td>
<td>65.85</td>
<td>20.07</td>
<td>37.48</td>
<td>28.42</td>
<td>15.57</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Metric: bad2</th>
<th>Method 1</th>
<th>34.46</th>
<th>42.53</th>
<th>72.01</th>
<th>34.33</th>
<th>48.48</th>
<th>40.68</th>
<th>21.21</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method 2</td>
<td>22.30</td>
<td>35.58</td>
<td>66.27</td>
<td>23.73</td>
<td>43.94</td>
<td>31.12</td>
<td>15.32</td>
<td></td>
</tr>
<tr>
<td>Method 3</td>
<td>29.32</td>
<td>40.14</td>
<td>68.91</td>
<td>24.82</td>
<td>45.39</td>
<td>37.55</td>
<td>19.86</td>
<td></td>
</tr>
</tbody>
</table>

The main takeaways from table 4.12 are that method 2 indeed seems to perform the best on the majority of the images in almost all metrics, and that method 1 seemingly performs poorly on all images in the dataset.

### 4.1.4 Summary and Qualitative Comparisons

Method 1 shows good results on our dataset, and it serves as a proof of concept that the depth completion network by Eldesokey et al. [2] can be adapted for disparity refinement. That being said, the results can not compete with the state-of-the-art. In the evaluations on our dataset, method 1 has lower MAE and RMSE than method 2, but has worse bad4 and bad2 metrics. On Middlebury V3, method 2 outperforms method 1 on all four metrics. Even the inpainting approach of method 3 beats method 1. The refinement networks have only trained on FlyingThings3D and training images from our own dataset. It would seem that the ability for the network to generalize to unseen images is low, since it performs so poorly on Middlebury V3. This is further reinforced by table 4.13, which
shows that fine-tuning on the training images of our dataset was indispensable to getting good results on the evaluation images.

**Table 4.13**: Comparison of the performance of the best performing network with and without fine-tuning. The Epoch column denotes the number of epochs spent fine-tuning. This result demonstrates the importance of fine-tuning as well as the networks poor generalization ability, which in turn helps explain the poor results on Middlebury V3.

<table>
<thead>
<tr>
<th>Method 1 (Best Performing Network)</th>
<th>Metrics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pre-trained On</td>
<td>Network</td>
</tr>
<tr>
<td>Synth. (thr. conf.)</td>
<td>Enc-Dec</td>
</tr>
<tr>
<td>Synth. (thr. conf.)</td>
<td>Enc-Dec</td>
</tr>
</tbody>
</table>

Still, considering the comparatively few trainable parameters in the network, perhaps a lack of generalization should be expected. As for method 2, it performs the best generally, and has the best results on bad4 and bad2 on both datasets.

We identify the biggest source of error for method 1 as the spurious artifacts shown in figure 4.10. These artifacts appear in very large areas for which the initial disparity has no disparity values (figure 4.10b). Our initial conclusion was that the pre-training had taught the network to assign large missing areas with large disparity values, but seeing as even the networks pre-trained on the synthetic input exhibit these artifacts, we believe that this behaviour is something that is inherent to the method rather than something that was learnt from the data. Upon examining the intermediate output from the unguided network, we noticed that it fails to propagate disparity values to all pixels (figure 4.10c). It is precisely these areas that exhibit these spurious artifacts (figure 4.10d).

**Figure 4.10**: Illustration of spurious artifacts that appear in the output from the refinement network. Upon studying the intermediate output from the unguided network, it was apparent that these artifacts appear where the unguided network did not manage to propagate disparity values, i.e. the black areas in (c). This could be remedied by increasing the scale depth or kernel size of the unguided network.
By either increasing the kernel size of the unguided network or increasing the scale depth, these situations can be avoided. Unfortunately this phenomenon was noticed at a late stage and therefore there was no time to re-train all the networks.

We also identify that the SmoothL1 loss function tries to minimize the prediction error averaged over all the pixels in the image, which could help explain why the outputs of method 1 have a sort of low-pass filtered aesthetic to them. A low-pass tendency is detrimental to the bad4 and bad2 metrics, which help explain why method 1 performs poorly on these, especially on Middlebury V3.

To highlight how the pre-training benefits the results, we redid the training for the best performing network, this time not performing any pre-training but instead increasing the number of epochs spent fine-tuning. How these two networks performed is shown in table 4.14 and demonstrates the usefulness of pre-training. As a reminder, "pre-training" refers to training on FlyingThings3D, and "fine-tuning" on our dataset.

**Table 4.14:** Comparison of the performance of the best performing network with and without pre-training. The Epoch column denotes the number of epochs spent fine-tuning. This result demonstrates the usefulness of pre-training, and that fine-tuning alone leads to a inferior result.

<table>
<thead>
<tr>
<th>Pre-trained On</th>
<th>Network</th>
<th>Epochs</th>
<th>MAE</th>
<th>RMSE</th>
<th>bad4</th>
<th>bad2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Synth. (thr. conf.)</td>
<td>Enc-Dec</td>
<td>20</td>
<td>1.640</td>
<td>4.594</td>
<td>7.85</td>
<td>12.40</td>
</tr>
<tr>
<td>None</td>
<td>Enc-Dec</td>
<td>500</td>
<td>2.032</td>
<td>5.670</td>
<td>9.43</td>
<td>15.80</td>
</tr>
</tbody>
</table>

As for the penalty function (equation (2.17)) used to discard unsuitable items from the FlyingThings3D dataset prior to pre-training, it successfully discarded the worst items as shown in figure 4.11. Conversely, the images that received some of the lowest penalties are shown in figure 4.12. However, as shown in the middle row of figure 4.12, images which initial disparity failed to detect foreground objects still persist. This suggests that the penalty function used is not optimal, and that other choices of this function could yield a better ranking. This topic is expanded on in the discussions of section 5.1.2.

In summation of the results from the disparity refinement network, we provide qualitative comparisons of the relative improvement from input to output that can be expected, using the metrics explained at the end of section 2.9. These examples are found in figure 4.13 and show that the refinement network indeed has potential to refine even very sparse disparity maps.
Figure 4.11: Examples of items from FlyingThings3D that received some of the highest penalties, and were discarded from pre-training because of it. Each row represents an item. The columns depict (from left-to-right) the left image, the right image, the initial disparity produced by Saab’s algorithm, and the corresponding ground truth disparity. Notice how the initial disparity is vastly different from the ground truth.

Figure 4.12: Examples of items from FlyingThings3D that received some of the lowest penalties. Each row represents an item. The columns depict (from left-to-right) the left image, the right image, the initial disparity produced by Saab’s algorithm, and the corresponding ground truth disparity. Notice how the initial disparity in the middle row, although having received a low penalty, contains a foreground object in the upper left without any disparities.
4.1 Disparity Refinement

Figure 4.13: Qualitative comparisons of the relative improvement gained from applying method 1 on various disparity maps, using the metrics defined in section 2.9. The columns from left-to-right show the left RGB image, Saab’s initial disparity, and the output from the best performing network of method 1. (a) shows a validation image from our dataset, (b) and (c) show images from FlyingThings3D, and (d) shows the Motorcycle scene from Middlebury V3.
4.2 Dataset

This section presents the results of the dataset generation. The experiment setup is explained in chapter 3, and the evaluation methods specifically are explained in section 3.2.5.

4.2.1 Disparity Maps in General

In this section, results pertaining to the generated disparity maps and various situations are presented. The dataset includes 26 different scenes from 3 different views, which together with the merged disparity maps make up a total of 104 disparity maps. A scene with a dense disparity map can be found in figure 4.15. This dense disparity map is used for comparison with less dense disparity maps, and showcases a mostly optimal final disparity map, given the proposed methodology. Figure 4.14 shows a 3D reconstruction of scene in the dataset, in order to visualize the results. The camera view does not correspond with any of the actual cameras, and thus areas with occluded measurements have missing values. The disparity from single measurements which make up the final merged disparity make can be found in figure 4.16. This figure shows that holes in horizontal discontinuities are easier to fill in than vertical discontinuities. Areas near the lamp arm are only partially filled in. Figure 4.17 shows two different problematic situations. First, a situation where a region in the image remains unlit by the light source in all measurements, resulting in a large hole in the disparity map in that region. Second, a situation where all measurements of a region in a foreground object were poor, but regions from a successfully measured background object that ought to be occluded were filled in instead. In figure 4.18, the amount of crevices in the scene means that there are many regions with common camera occlusions and light source shadows in all the measurements, resulting in many missing disparity values.
Figure 4.15: Case with an overall dense disparity map. This showcases a mostly optimal result, given the proposed methodology.

Figure 4.16: Disparity maps from each stereo pair and light source positions, warped to the reference stereo pair. The disparities from the left, middle and right stereo pairs are in that order in the columns. The rows differ in the position of the light source. The final row shows the merged disparity map, together with the RGB image from the reference stereo pair.
Figure 4.17: The large marked area shows a situation where a large portion of the disparity map below the lamp is missing due to the shadow being cast by the lamp from the light source in all measurements. The small marked area shows a situation where all measurements were poor for the waist part of the action figure, resulting in disparity values from a background object being filled in.

Figure 4.18: Situation where the scene is simply too complex for the amount of positions for the light positions.
4.2 Dataset

(a) A situation where faulty disparities are filled in from a single right stereo measurement.

(b) By performing the chromatic consistency filtering, the faulty disparities are identified and removed.

(c) The faulty disparities have in this case instead been discarded in favor of foreground measurements from other stereo pairs.

(d) The RGB image from the reference camera.

**Figure 4.19:** This figure illustrates how faulty disparities can be found and discarded in two separate stages in the merging process. The disparity values in (a) and (b) come from the right stereo pair, while (c) shows a fully merged disparity map. All three maps are expressed in the reference camera.

(a) Remaining faulty disparities remaining after filtering chromatically inconsistent pixels.

(b) Some faulty disparities remaining even after massively lowering the threshold of the chromatic consistency filter. At this threshold, a lot of correct disparity values are erroneously filtered away.

(c) RGB image from the right stereo pair warped to the reference camera. The RGB values corresponding to the faulty disparities look very similar to the correct values in (d).

(d) The RGB image from the reference camera.

**Figure 4.20:** This figure illustrates a situation where neither the merging process nor the chromatic consistency filtering manages to properly discard the faulty disparities.
Figure 4.19 and 4.20 show two situations that affect the disparity merging process and chromatic consistency filtering. In figure 4.19, both the preference of foreground objects in the merging process, and the chromatic consistency filtering manage to discard faulty disparities. In figure 4.20 however, since all measurements of the foreground object were poor, there is no foreground object to take preference over the background object. The chromatic consistency filtering can remove most of the faulty disparities by lowering the filter threshold, but this is at the cost of discarding many correct disparity values as well.

4.2.2 Evaluation

In this section, results pertaining to the evaluation of the dataset will be presented. Figure 4.21 shows the residuals as a heat map together with the normal vectors of the plane fitting of the sides of a cubic object, also shown in the figure. The colour scale for the residuals goes from -0.25 mm to 0.25 mm. Table 4.15 presents the linearity metrics which correspond to figure 4.21. A clear reduction in the size of the residuals, post merging, can be observed in both the figure and the table. That the left view has better metrics than the merge view for the right plane is likely to be a fluke, since it has considerably worse metrics for the other planes. The Middlebury dataset [3] does not provide RMSE for their measurements which is why the value is missing in a comparison of the linearity metrics, shown in table 4.16. We conclude that the disparity merging process has a positive effect on the linearity metric.

Table 4.17 shows the orthogonality metrics, which are unavailable for the Middlebury dataset. The 3DUNDERWORLD-SLS values in both these tables contain averages of the of metrics provided in the paper [40], in order to make them more comparable with our values, and those from the Middlebury dataset. While the sides of the cube were orthogonal up to less than tenth of a degree (table 4.17), the UNDERGROUND-SLS values were significantly better. Other than the difference in resolution and amount of measurements, the cubic object itself is slightly different. As is visible in figure 4.21c, two of the sides are far smaller than the top side, giving fewer measurements. The top side, while large, has a discolouration which negatively affects the structured light algorithm. This discolouration is visible in the residuals.

As in figure 4.21, figure 4.22 shows the residuals of a plane fitting, but this time with a calibration board. Figure 4.22e in particular is from the calibration stage, while the rest are from the structured light measurements. The calibration board pattern is visible in the residuals in the figure due to an unwanted side effect in the structured light algorithm. The colour scale of the residuals goes from -0.5 mm to 0.5 mm. The residuals from the structured light measurements should resemble those from calibration stage as much as possible. The resemblance is quite clear in the merged measurements, shown in figure 4.22d. The residuals from the left stereo pair are positive on the sides and negative in the middle, just like the calibration residuals, but slightly larger. This means that the nature of the shape is correct, but slightly exaggerated. The residuals from the merge view have attenuated the artifacts of the structured light algorithm,
and also looks most similar to the calibration residuals.

(a) Residuals and normal vectors from the middle stereo pair measurements.  (b) Residuals and normal vectors from the merged measurements.  (c) The cubic object used for the measurements.

Figure 4.21: Visualization of linearity metric residuals, where the residuals have been added as a texture to the piece-wise planar object. The red lines show the normal vectors of the planes.

Table 4.15: Linearity metric as measured on the cubic object.

<table>
<thead>
<tr>
<th>View/PlaneView/</th>
<th>Left MAE</th>
<th>Left RMSE</th>
<th>Top MAE</th>
<th>Top RMSE</th>
<th>Right MAE</th>
<th>Right RMSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Left</td>
<td>0.03614</td>
<td>0.04420</td>
<td>0.02081</td>
<td>0.02681</td>
<td>0.01212</td>
<td>0.01546</td>
</tr>
<tr>
<td>Right</td>
<td>0.01505</td>
<td>0.01961</td>
<td>0.02257</td>
<td>0.02920</td>
<td>0.01959</td>
<td>0.02454</td>
</tr>
<tr>
<td>Middle</td>
<td>0.04024</td>
<td>0.04907</td>
<td>0.02892</td>
<td>0.03770</td>
<td>0.02518</td>
<td>0.03158</td>
</tr>
<tr>
<td>Merged</td>
<td>0.01469</td>
<td>0.01935</td>
<td>0.01572</td>
<td>0.02068</td>
<td>0.01264</td>
<td>0.01580</td>
</tr>
</tbody>
</table>

Table 4.16: Linearity metric comparison.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>MAE</th>
<th>RMSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Middlebury [3]</td>
<td>0.032</td>
<td>-</td>
</tr>
<tr>
<td>Ours (average of merged)</td>
<td>0.01523</td>
<td>0.02001</td>
</tr>
<tr>
<td>3DUNDERWORLD-SLS [40] (averaged)</td>
<td><strong>0.005678</strong></td>
<td><strong>0.01161</strong></td>
</tr>
</tbody>
</table>

Table 4.17: Orthogonality metrics.

<table>
<thead>
<tr>
<th>View / Metric</th>
<th>$E_{\text{orth}, \deg, 1, 2}$</th>
<th>$E_{\text{orth}, \deg, 1, 3}$</th>
<th>$E_{\text{orth}, \deg, 2, 3}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Left</td>
<td>0.07596</td>
<td>0.08880</td>
<td><strong>0.01542</strong></td>
</tr>
<tr>
<td>Right</td>
<td>0.09563</td>
<td>0.01341</td>
<td>0.07054</td>
</tr>
<tr>
<td>Middle</td>
<td><strong>0.07151</strong></td>
<td>0.05684</td>
<td>0.05684</td>
</tr>
<tr>
<td>Merged</td>
<td>0.08084</td>
<td><strong>0.00699</strong></td>
<td>0.02947</td>
</tr>
<tr>
<td>3DUNDERWORLD-SLS</td>
<td>0.0002728</td>
<td>0.0004038</td>
<td>0.0021327</td>
</tr>
</tbody>
</table>
Results

(a) Residuals from the left stereo pair.
(b) Residuals from the right stereo pair.
(c) Residuals from the middle stereo pair.
(d) Residuals from merged measurements from all stereo pairs.
(e) Diffused residuals from the calibration stage.
(f) The calibration board used for measurements. The ruler at the bottom was used for the accuracy test.

Figure 4.22: Residuals of plane fitting of a calibration board. Values in (a-d) should resemble those in (e).
4.2 Dataset

Figure 4.23: Difference of calibration residuals and residuals from the different views. Values near zero are good.

Figure 4.23 shows the difference of the residuals from the structured light measurements, and the residuals from the calibration. This figure is meant to complement figure 4.22, to make it easier to judge the resemblance of the observed deformations of the calibration board from structured light measurements, and the measurements from the calibration stage. Values near zero mean both measurements agree on the nature of the deformations of the calibration board. The colour scale of the residuals goes from -0.5 mm to 0.5 mm.

From visual inspection in the calibration board residual figure (4.22), a clear improvement can be seen between the single stereo measurements and the merged measurements. This is further reinforced by looking at the residual difference figure (4.23) and the metrics table (4.18), which both show significant improvements after merging. The values of residual differences in figure 4.23d are smaller and the errors are more evenly distributed than in the single stereo counterparts. As expected from our observations of the residuals from the left stereo pair in figure 4.22a, which seemed to be too large, the difference of the two residuals in...
figure 4.23a almost appear to have inverted the observed deformation, showing negative values on the sides, and positive values in the middle. This effect is apparently mostly removed in the after merging, as seen in 4.23d. Since no other dataset provides an evaluation metric such as this, it is not possible to compare the relative performance of this metric compared to other datasets. It did provide expected results, at least for the merged measurements, so we conclude that including this evaluation metric had some value.

**Table 4.18:** MAE and RMSE of the difference of the calibration board plane fitting residuals for the different views.

<table>
<thead>
<tr>
<th>View / Metric</th>
<th>MAE (mm)</th>
<th>RMSE (mm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Left</td>
<td>0.11865</td>
<td>0.15056</td>
</tr>
<tr>
<td>Right</td>
<td>0.10807</td>
<td>0.13914</td>
</tr>
<tr>
<td>Middle</td>
<td>0.12319</td>
<td>0.16035</td>
</tr>
<tr>
<td>Merged</td>
<td><strong>0.06461</strong></td>
<td><strong>0.08372</strong></td>
</tr>
</tbody>
</table>

Table 4.19 shows the accuracy metrics. The values are an average of 5 measurements of a 30 cm ruler. The standard deviation of those measurements are shown in the second column. The accuracy metrics were good, but had some measuring noise, due to that the measurements were an average of several hand made measurements. The standard deviation values are not proper evaluation metrics, but are included to show the extent of the measuring noise. The standard deviations show that our measurements of this metric are not as precise as those in 3DUNDERWORLD-SLS [40], but there does not appear to be a bias to measuring shorter or longer objects, since there are both positive and negative relative errors. Since UNDERWORLD-SLS only provide absolute values, it is not possible to calculate the relative error for comparison.

**Table 4.19:** Accuracy metrics. The 3DUNDERWORLD-SLS value is an absolute value, and the standard deviation value is the standard deviation of absolute valued metric instead of the signed metric used by us.

<table>
<thead>
<tr>
<th>View / Metric</th>
<th>$E_{acc}$ (mm)</th>
<th>Standard deviation</th>
<th>$E_{relative}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Left</td>
<td>0.1360</td>
<td>0.1900</td>
<td>0.0453 %</td>
</tr>
<tr>
<td>Right</td>
<td>-0.0808</td>
<td>0.2654</td>
<td>-0.0269 %</td>
</tr>
<tr>
<td>Middle</td>
<td>0.2166</td>
<td>0.0818</td>
<td>0.0722 %</td>
</tr>
<tr>
<td>Merged</td>
<td><strong>-0.0206</strong></td>
<td>0.3156</td>
<td><strong>-0.0069 %</strong></td>
</tr>
<tr>
<td>3DUNDERWORLD-SLS (abs)</td>
<td>0.05200</td>
<td>0.03602</td>
<td>-</td>
</tr>
</tbody>
</table>
5 Discussion

In this chapter, we provide discussions related to the methods used as well as overall observations gathered throughout this thesis.

5.1 Disparity Refinement

As shown in section 4.1.1, the evaluation results from the pre-training show better results when compared to the results from our own dataset and Middlebury V3. This is reasonable considering that the large size of the dataset, and that all training and validation images in this part come from FlyingThings3D. These results show that, as long as large amounts of data is provided, the refinement network is able to infer a good disparity map from Saab’s initial disparity as well as synthetic initial disparity.

The refinement network that has the best overall performance is the Encoder-Decoder with SmoothL1 loss, using scaled (non-thresholded) confidences. This suggests that including continuous confidence values is beneficial, and it reinforces the usefulness of including RGB guidance in the network. One explanation for why Enc-Dec performs better than MS is that Enc-Dec considers different scales of the input whereas MS does not. Judging from the results, it would seem that the SmoothL1 loss is superior to the L2 loss. In fact, the best network trained using SmoothL1 outperforms the best network trained using L2 on the RMSE metric, a metric that the L2 loss in theory should minimize well. We conclude that the L2 loss is not suitable and that prioritizing outliers during training is not optimal for the input data we have.

It would have been interesting to try to increase the layer depth at various parts of the refinement network to see if performance could be improved that way, but lack of time prevented us from carrying out such experiments.

Further, if more time had been available, it would have been interesting to...
experiment with the construction of a loss function that considers the ground truth categories to give larger weight to refinement of incorrect disparities in the initial disparity map. The concept of a penalty function similar to the one we use to discard FlyingThings3D images might also be used for weighting the loss function.

While method 2 performs well on both datasets, it does not perform nearly as well as the results shown by the original author [15]. There, predictions from a CNN trained to perform disparity matching is used, which provides a dense and mostly accurate (albeit noisy) initial disparity. The characteristics of this disparity differ greatly from the initial disparity from Saabs algorithm. Considering that we applied method 2 to the training images of Middlebury V3 [3] in exactly the same way as the original author and get inferior results, we conclude that Saabs initial disparity is more difficult to refine. Having said that, it is unfair to compare a method that has real time demands (Saabs) with another that has no such demand (the disparity matching CNN).

5.1.1 Initial Disparity

The biggest source of error in Saabs initial disparity is that it sometimes fails to find even a single disparity value of a foreground object (figure 5.1), resulting in a hole in the area of the disparity map corresponding to the foreground object.

![Figure 5.1: Illustration of a problematic image from FlyingThings3D [30] that was not discarded. Many foreground objects are not detected by Saabs algorithm and as a result no disparity values are assigned for them. Images like this one should be discarded, but the penalty function (equation (2.17) assigned this image a comparatively low penalty and therefore was not removed.](image-url)

This is due to the need for fine-tuning of the parameters of Saabs stereo algorithm to fit each scene, something that was not possible for FlyingThings3D. Therefore, the blame can hardly be put on the algorithm, as this phenomenon mainly appears in images from FlyingThings3D. While the refinement network has some means of inferring that this hole is a separate object (based only on the RGB image), it has no way of inferring the correct disparity values of it. This is because the network needs initial disparity values, but not a single one was given. Instead the network tends to assign disparities in such holes to values that are common for foreground objects (figure 5.2). The result is a disparity map
where one can visually see the foreground object recovered to some extent, but the disparity values of the object are of poor quality.

![Illustration](image)

**Figure 5.2:** Illustration of the refinement networks ability to partially cope with completely missing foreground objects. The knife in the foreground has no disparity values in the initial disparity of (b), yet the network assigns plausible values for it (c). These assigned values are often of poor quality, but it at least speaks to the networks ability to reason about the scene.

In the end, the penalty function we used managed to remove the worst images, and the majority of the remaining FlyingThings3D images had perfectly acceptable initial disparities. We considered training with different subsets of the FlyingThings3D images (discarding 50% or even 75% of the images). However, we eventually decided against this due to lack of time.

### 5.1.2 Choice of Penalty Function

Despite our efforts to weed out images from FlyingThings3D that Saabs algorithm did not create a good initial disparity for, images which initial disparity failed to detect foreground objects still persisted. This means that the penalty function $P$ (equation (2.17)) that we used to rank and discard images was not optimal and deserves some critique.

The most obvious issue is our choice of $\Omega$, i.e. which pixels to compare disparity values for. Since we chose only to include pixels that have values in the initial disparity, this allows images, whose initial disparity fail to give disparity values to foreground objects, to receive a low penalty, as long as the disparity values that are given are sound. Also, it technically allows images whose initial disparity is extremely sparse to also receive low penalty, however the characteristics of Saabs algorithm makes this eventuality highly unlikely, and we did not identify any such images throughout the course of this work.

Originally, we considered a loss function that compared ground truth disparities not to the initial disparity, but to the result from an unguided (normalized convolution) network that had been trained for a few epochs on all images (or even to the result from applying inpainting on the initial disparity). Although such a function would penalize missing foreground objects, this approach was deemed unsuited because it would mean letting an unstable network decide the ranking.

Based on the above reflections, we identify three characteristics that a good penalty function should penalize. First, it should penalize disparity maps for
which a high percentage of disparity values given are incorrect, Secondly, it should penalize disparity maps that are sparse (a high percentage of pixels in the map have no disparity values). Lastly, it should penalize maps for which objects (foreground or otherwise) in the scene have no disparity values at all. The loss function used in this thesis only penalizes the first characteristic and is oblivious to the other two, which we believe reduced the quality of the pre-training. Designing a function that also considers the second characteristic (penalizing sparse maps) can be achieved by adding a second term to our penalty function (equation (2.17)) with some suitable weighting. The third characteristic (penalizing missing objects) is perhaps not as straightforward, since it requires some global understanding of the entire image. One approach could be to segment the image into superpixels and enforce some penalty if the disparity map lacks disparity values in one or many superpixels. A potential drawback is the assumption that each superpixel corresponds to an object, something that might not always hold true. The design of penalty function is an interesting topic that should be considered should similar work be done in the future.

5.1.3 Categorization of Ground Truth

Thanks to the categorization of the ground truth we could evaluate refinement of incorrect disparities and filling of missing disparities separately. We realise that the hole filling category could have been split into two, trivial holes and difficult holes. This has similarities to what we did when we split disparities that Saabs algorithm found into correct (preservation) and incorrect (refinement) ones. One way to achieve this is to let the holes be filled by inpainting, and put the disparities that inpainting correctly filled in into the "trivial holes" category. The remaining disparities that inpainting could not solve would then become the "hard holes" category.

Lastly, we question the usefulness of category 1 (preservation of correct disparities), since this objective has far less priority than refinement of incorrect disparities and hole filling. Nevertheless, we believe that this categorization enhanced the results and provided transparency into the behaviour of each method.

5.1.4 Closing Remarks and Suggestions for Future Work

As with all refinement methods, the success of a method relies to some extent on the quality and characteristics of the input, and it is difficult to assess if the results were limited by the input or by the method. However, considering refinement results on the synthetically created initial disparity maps were of particularly high quality, we conclude that the network indeed has promise. It is quite possible that better results can be achieved if another type of initial disparity was used. Therefore, it would be interesting to train method 1 using initial disparity created by some other method, for example the one that Yan et al. [15] (the original author of method 2) uses, i.e. the stereo matching CNN proposed by Zbontar and LeCun [17]. Although no explicit confidence measures are produced by the CNN, one could potentially use the (inverse) variance of the disparity values around a
pixel to produce a confidence measure. Ultimately, since this is the only work that performs refinement specifically on initial disparity from Saab's algorithm, it is difficult to say with certainty what kind of results should be expected.

Our suggestions for future work on disparity refinement include (1) investigating other loss functions for the refinement network based on the discussions on the penalty function in equation (2.17) and the ground truth categorization, (2) reformulating a better penalty function for ranking and discarding unsuitable images from the FlyingThings3D dataset, (3) training the refinement network using initial disparity of another type than that of Saab's, and (4) compounding the images from FlyingThings3D with those of other datasets to see if the refinement network can be made more robust to different types of input. The biggest takeaway is that the refinement network could be adapted for disparity refinement and therefore it is possible that further efforts could increase its performance.

5.2 Dataset

The evaluation metrics of the dataset are very satisfactory overall. The superior performance of 3DUNDERGROUND-SLS in the evaluation metrics can in part be explained by the fact that they used cameras with about 10 times higher resolution than ours. In addition to this over 40 measurements were used in their experiments, whereas we used a total of 6 measurements for the merged disparity maps. On the other hand, we performed well in the linearity metric against the Middlebury dataset which used a more similar setup. The merging process had a clear positive effect on the linearity metric and the calibration board comparison, but there was no clear improvement for the accuracy metric and the orthogonality metric. There is nothing that hints at the accuracy being poor, but due to the measurement noise of this metric, it is also difficult to objectively conclude that the dataset is accurate.

The dataset is well performant, efficient to generate and contains dense disparity maps. Some of the phenomena that negatively affect these characteristics are discussed below, along with some ideas for improvements.

5.2.1 Problems and Improvements

Some of the ground truth disparities suffer from light source shadows, which appear due to the light source being situated above the cameras (figure 4.17). It is in theory easy to fix this by adding another light source position below the cameras. However in practice, this small change would likely more than double the time required for us to measure one scene, due to limitations in the equipment available. The problem can also be somewhat mitigated by avoiding measuring objects with large depth discontinuities in the vertical direction. Since this thesis strives to find a fast way to create reliable ground truth disparities, in order to be able to generate sufficient quantities for use in learning algorithms, we decided to not slow down our scene generation process. However, since this issue is not a fundamental one, but one of practicality, it is certainly possible for some-
one with other equipment to be able to add an additional light source, without significantly affecting the scene generation time.

Scenes such as the complex scene in figure 4.18 would be a bit trickier to densify. For full density, both cameras in some stereo pair, and the light source would need to have a clear line of sight to the object visible in each pixel in the reference camera. If any single one of these do not have a clear line of sight, either camera occlusion or light source shadows will appear. This means that the required combination of positions of the cameras and light sources needed for high density can quickly increase with scene complexity. It is definitely possible to generate a dense disparity map with a low number of stereo pairs and light sources, such as the scene in figure 4.15, but one is ultimately limited in the set of possible scenes that one can generate.

There is one problem that occurs since we do not only add the final merged disparity to the dataset, but also the single stereo pair measurements. Scenes that require many stereo pairs in order to get high density inescapably means that these single stereo pair disparity maps will be even more sparse than for comparatively simple scenes. An arbitrary amount of stereo pairs can be used to get closer and closer to 100% density, but if the scene requires it to get an acceptable density, then this method might be sub-optimal. For somewhat simpler scenes however, a large amount of high density ground truth disparity maps can be quickly generated.

With our setup, we managed to create 26 scenes, for a total of 104 disparity maps, in less than a day. One of the limiting factors were finding new objects to include in the scenes, as to not simply show the same objects in different orientations over and over again. If HDR (High Dynamic Range) measurements were to be performed, it would greatly increase the variety of objects that can be included in the scenes, since objects with significantly differing light absorbing properties can not be successfully measured simultaneously. Since all objects would have better measurements, it would also reduce the amount of occurrences where foreground objects have missing measurements, and thus reduce the risk of introducing faulty disparities from background objects in the merging process.

5.2.2 Merging Process

We include two different methods for finding and discarding faulty disparities stemming from background objects. Foreground object preference, and chromatic consistency filtering. It is not trivial to evaluate these methods, since we do not know where these phenomena occur, and if we did, solving this problem would be trivial. The occurrences evaluated in the results section are found by studying the disparity maps manually. The fact that we only found two occurrences makes it reasonable to believe the assumption that this phenomenon becomes more uncommon if the ratio between distance between the stereo pairs and distance to the scene is small. In both cases included, either both methods succeed, or both methods either fail or produce unsatisfactory results. But since only two instances of this phenomenon were found, it is difficult to judge how well these methods perform generally. The foreground object preference in
the disparity map merging only need a single successful measurement from any stereo pair. This means that the probability of faulty disparities remaining after the merging process should become lower and lower the more measurements we add. The chromatic consistency filtering on the other hand does not improve with the amount of measurements, but simply relies on the assumption that the colours of the foreground object and the background object are distinct. As can be seen in figure 4.20, this obviously does not work well when the foreground object and background object are by happenstance chromatically similar.

Given a semantic segmentation algorithm, it should be possible to weed out faulty disparity values by comparing their labels. In the case of the situation in figure 4.20, the faulty disparity values on the waist could be discarded since that area would be labelled as "action figure" in the reference camera, but in the camera from where the disparities were warped, the label would be "cardboard". This check could be done by keeping track of the labels during the warp.

The merging process can be expected to scale well with additional stereo pairs as long as the ratio between the distance between the stereo pairs and the distance to the scene does not significantly increase. As long as this ratio does not significantly increase, we do not leave the domain where the merging process is expected to be sufficient. This should be useful if we want even more dense disparities, and even faster generation of disparity maps, since each new stereo pair gives rise to a new disparity map to add to the dataset. The limitation of this ratio should not be a problem in practice, since less and less useful depth information can be measured the further one goes from the initial camera pose. This is easy to visualize if we consider the extreme case where the 2 stereo pairs are on opposite sides of an object. The second stereo pair would only have measurements of the backside of the object, which are completely useless, or even destructive, when warped to the first stereo pair for merging.

### 5.2.3 Future Work

In this thesis we have only performed the merging of the disparity maps to the reference camera. But it is also possible to perform the merging to every camera in the setup. This would increase the density of the disparity maps from every view. This does have some diminishing returns since the reference camera has the most neighboring stereo pairs to fill in holes from other directions. There would also be times where the spatial distance between the stereo pairs involved in the merging is greater than it is in this thesis, e.g. if we would warp the disparities from the right stereo pair to the left stereo pair. This means that the observed scenes in the two views are even more different than it is in this thesis, and that there are more situations where we need to make the distinction between foreground objects and background objects correctly. But if done correctly it would give a larger number of high density disparity maps than in this dataset.
In this chapter, the conclusions of this thesis work are presented. The principal goals for this thesis are twofold. The first one is to investigate the potential of applying the depth completion network proposed by Eldesokey et al. [2]. The second one is to generate a dataset for use by learning algorithms. The conclusions related to these goals are separately presented below.

6.1 Disparity Refinement

In this part of the thesis, a network similar to [2] is created, where several architectures are considered and evaluated. Training consists of first pre-training on the large FlyingThings3D dataset [18], followed by fine-tuning on all but the last 5 scenes in the dataset created in this thesis. Evaluation is performed on the last 5 scenes as well as the training images from the Middlebury V3 dataset.

In order to train the network, initial disparities are created for all images of all datasets. This is done by applying the real-time stereo algorithm developed at Saab Dynamics. A second set of initial disparities are synthetically created for the pre-training phase by strategically sampling the ground truth and adding noise. One network is trained for each combination of architecture and initial disparities during pre-training.

To put the method of [2] in contrast, a non-learning, state-of-the-art method, as well as a basic baseline method, is implemented. These methods are evaluated in the same manner as [2]. The evaluation is split into three categories: preservation of correct disparities, refinement of incorrect disparities and hole filling accuracy.

Results show that the depth completion network proposed by Eldesokey et al. [2] can be adapted for disparity map refinement, but is dependent on the quality of the input disparity. In the case of initial disparity map from Saab's stereo...
algorithm, state-of-the-art performance is not achieved. However, refinement on the synthetically created initial disparity maps show very satisfactory results and suggests that the method has potential to reach higher performance given an input of higher quality.

6.2 Dataset

Part of the goal of this thesis is to effectively create a dataset for use with learning algorithms. This is done by measuring a scene with several stereo camera pairs and light sources. Several dense disparity maps are then generated from each scene. The most significant limitation comes from the set of possible objects to include in the scene, due to limitation in the structured light algorithm. It is very difficult to measure objects with specular surfaces, but measuring surfaces with varying light absorption properties is a challenge that can be overcome by performing several measurements with HDR.

The most prominent challenges are therefore related to improving the density of the disparity maps, and the most straightforward solution to them is to increase the number of cameras and light sources. Adding additional cameras introduces the problem of having several disparity values corresponding to different depths in each pixel. This problem is handled by preferring large disparity values over small ones, in order to discard values that correspond to background objects. Comparing RGB values from different views is also done to remove disparity values whose corresponding RGB values differ from each other.

The dataset performs well on the all 4 evaluation metrics. The first metric is a linearity metric where plane fitting is done on planar objects. The second metric is an orthogonality metric, where an object with orthogonal sides is measured. The scaling accuracy is evaluated by measuring an object with known length. In the final metric ability to reconstruct the deformations of a calibration board is measured.

Considering the density, performance of the evaluation metrics, and ease of creating a new scene, we conclude that this method is capable of creating ground truth data for use by learning algorithms.
Bibliography


