AUTOMATIC DISPARITY SEARCH RANGE ESTIMATION IN DEEP LEARNING STEREO

A thesis submitted by

Ruveen Perera, B.Sc. Eng. (Hons)

For the award of

Doctor of Philosophy

2021

ABSTRACT

This research concerns the deep stereo networks used for inferring depth from images captured with a stereo pair of cameras. Central to the process is the measurement of disparity between the images, with the computational effort depending on the limit of the "disparity search range (DSR)" over which the search is to be performed. Like the traditional stereo techniques, deep learning stereo methods also require users to specify the upper bound of DSR, commonly known as the "Maximum Disparity", manually. Selecting a substantially lower or a higher maximum disparity than necessary for a given scene can lead to disparity estimation errors or performance degradations during disparity inference.

This thesis presents an automatic disparity search range estimation technique for deep learning stereo which can be seamlessly embedded into the stereo algorithm itself without requiring pre-processing or explicit configuration by the users. The method incorporates a novel metric referred to as the Sum of New Cost Extrema (SNCE). The stated metric can be estimated on a per-layer basis during the cost volume construction phase of the stereo method. This can serve as the criterion on which to decide whether to continue the cost volume creation process at a given disparity, or to terminate it. In this way, the maximum disparity for a given scene is determined.

The SCNE metric is further utilised in a deep stereo algorithm which produces accurate disparity maps while estimating the disparity search range automatically without user intervention. The memory efficient design of the algorithm makes it possible to optimise memory for standard desktop computers and consumer-grade graphics processing hardware. Evaluations conducted using the benchmark stereo datasets indicate that the algorithm is able to produce accurate results for synthetic and real stereo image sequences without requiring users to set any parameter values during disparity inference, setting a new state-of-the-art benchmark in stereo disparity estimation. Results indicate improvements in performance due to the computational efficiency arising from the optimal cost volume sizes. When used with standard stereo image sequences, the algorithm performed up to 50% faster compared to a reference implementation with a fixed cost volume size. Extended testing with common stereo evaluation metrics on various real-world stereo datasets showed that the algorithm

can produce accurate results under varying scene conditions. Additional tests on images captured with a custom-built stereo camera confirmed the generalization capabilities of the algorithm while demonstrating the possibility of achieving complete independence from user specified parameters values when inferring depth from real world stereo imagery.

CERTIFICATION OF THESIS

This thesis is entirely the work of Ruveen Perera except where otherwise acknowledged. The work is original and has not previously been submitted for any other award, except where acknowledged.

Principal Supervisor	Dr. Tobias Low
Associate Supervisor	Prof. John Billingsley

Student and supervisors' signatures of endorsement are held at the university

ACKNOWLEDGEMENT

I would like to acknowledge the support I received from my supervisors, staff at USQ, friends and my family throughout this arduous but rewarding doctoral study. This would not have been possible without having such amazing people around me.

I owe my deepest gratitude to Dr. Tobias Low for guiding me through this process as my principal supervisor, mentor, teacher, and a facilitator. Your knowledge in the field, insights, trust, and patience have been instrumental in shaping up this research study. I was fortunate to be part of the productive and stimulating discussions we had regularly. Thank you for believing in my abilities and allowing me to challenge myself to broaden my horizons.

I am also extremely grateful to the associate supervisor Prof. John Billingsley for the support, invaluable advice, practical suggestions, and words of encouragement. I must also thank you for inspiring me to pursue interpersonal skills in addition to research skills to become a better person overall.

I would like to express my deepest appreciation to the examiners for their vitally important feedback that improved the presentation of the thesis considerably. I also sincerely acknowledge the helpful advice and constructive feedback from Dr. Ray Malpress, Dr. Kithsiri Perera and Dr. Andrew Maxwell during and after the confirmation process. My special thanks go to Mr. Mark Phythian for employing me as part of the teaching staff on a regular basis.

I can never forget my friends, colleagues, and the members of the "Toowoomba City Toastmasters Club" who provided all the positive and happy distractions to keep the social being inside me, alive during the past 4 years.

Finally, I would like to extend my sincere gratitude to my parents, wife and daughter for their unparalleled love and support. I am forever indebted to you for the sacrifices you have made and the hardships you endured to help me achieve my dreams.

This research has been supported by the Australian Government Research Training Program Scholarship. Therefore, I am ever so grateful to the people of Australia for the generosity and support towards scientific research.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

DEFINITIONS

Camera Calibration – *Camera calibration is the process of determining intrinsic and extrinsic parameters of a camera that consists of a lens and an imaging sensor. Intrinsic parameters refer to the internal characteristics such as focal length, centre, and distortion. Extrinsic parameters determine the position and the orientation of the camera with respect to the world.*

Stereo Image Rectification – *Transformation of images from mutually displaced cameras into a parallel camera geometry so that the pixels which correspond to the same point in 3D will lie in the same horizontal line in the two images.*

Disparity – *Apparent displacement between matched features in two views from two cameras.*

Matching Cost – *Matching cost refers to a numerical measure of similarity or dissimilarity (singular valued or multi-dimensional) between two pixels being matched.*

Legacy Stereo Pipeline – *Legacy stereo pipeline also referred to as the conventional or traditional stereo pipeline is a collective name for the 4-stage disparity estimation process used by the conventional stereo disparity estimation algorithms which includes: matching cost computation, cost aggregation, disparity selection and disparity refinement.*

Epipolar Plane – *The geometric plane that goes through a point of concern in the world and the two optical centres of a binocular stereo camera system.*

Epipolar Lines – *Lines of intersection between the epipolar plane and the image planes of cameras in a stereo camera system.*

Disparity Search Range – *Disparity search range refers to the range of pixel positions scanned along the epipolar lines in one image of a stereo pair, to find a match for a pixel in the other image. It consists of a floor value and a ceiling value.*

Maximum Disparity – *This is the upper bound (i.e., ceiling value) of the disparity search range. In most stereo algorithms, the users are expected to provide a value for maximum disparity.*

Minimum Disparity – *This is the lower bound (i.e., floor value) of the disparity search range and in most deep stereo algorithms it is taken as zero. However, it is possible to have negative minimum disparity because of calibration artefacts. In the scope of the thesis, the minimum disparity is considered to be zero.*

Cost Volume – *A 3D or higher dimensional arrangement of computed matching costs by stereo algorithms. The dimensions of a cost volume are determined by the image width, image height, maximum disparity, and the dimensionality of the individual matching costs. Singular valued matching costs result in 3D cost volumes whereas 2D or higher dimensionalities lead to 4D or higher cost volumes.*

Binocular Stereo Algorithms – *Binocular stereo algorithms estimate disparity or depth from images captured by using two mutually-displaced cameras. In this thesis, stereo algorithms refer to binocular stereo algorithms unless stated otherwise.*

Deep Stereo Algorithms – *Stereo disparity/depth estimation algorithms which are based on deep neural networks.*

Training Stage – *Training phase of a supervised deep stereo algorithm during which the gradient of the loss is used for backpropagation.*

Disparity Inference – *This refers to the phase after training when the trained model is used for estimating disparity maps from stereo images. During disparity inference, supervised deep stereo methods do not use backpropagation to update the model parameters.*

Automatic Disparity Search Range – *Automatic Disparity Search Range refers to the automatic estimation of the disparity search range without user intervention. In stereo algorithms which use zero as the minimum disparity, estimation of the maximum disparity results in full disparity search range estimation.*

Sum of New Cost Extrema – *Sum of New Cost Extrema is the novel metric introduced in this thesis. It refers to the number of matching cost extrema locations which move to the latest layer of a matching cost volume during a layer-wise cost volume creation process.*

Population Maximum (Disparity) – *This refers to the maximum ground-truth disparity across all pixels in the entire stereo dataset in concern*.

Consumer Grade Hardware – *This refers to the commonly available consumer graphics processing units or hardware often used for recreational/gaming activities by home users (at the time when the experiments were conducted). NVIDIA RTX2080 and NVIDIA GTX1070 are examples for devices referred to as consumer grade hardware in the thesis.*

LiDAR – *A method used for measuring distance by measuring the return time of a reflected laser beam.*

ABBREVIATIONS

AD – *Absolute Difference*

ADSR – *Automatic Disparity Search Range*

BRISK – *Binary Robust Invariant Scalable Points*

CNN – *Convolutional Neural Network*

CPU – *Central Processing Unit*

CT – *Census Transform*

CUDA – *Compute Unified Device Architecture*

DSR – *Disparity Search Range*

EPE – *End Point Error*

GPU – *Graphics Processing Unit*

LSTM – *Long Short-Term Memory*

MRF – *Markov Random Field*

NCC – *Normalized Cross Correlation*

ReLU – *Rectified Linear Unit*

RNN – *Recurrent Neural Networks*

RT – *Rank Transform*

SAD – *Sum of Absolute Difference*

SGM – *Semi Global Matching*

SIFT – *Scale Invariant Feature Transform*

SNCE – *Sum of New Cost Extrema*

SOTA – *State of the Art*

SPP – *Spatial Pyramid Pooling*

SSD – *Sum of Squared Difference*

SURF – *Speeded Up Robust Features*

WTA – *Winner Takes All*

CHAPTER 01 - INTRODUCTION

Binocular stereo vision involves extracting depth information from two views of a scene captured with a mutually displaced pair of cameras. The existence of biological stereo vision, as a reliable and self-sufficient form of perception in nature, continues to provide a benchmark for the machine-based binocular stereo vision systems [1]. One of the basic differences between biological and computer stereo vision is the prior configuration requirements. In biological stereo vision, the configuration process is autonomous and is driven by the environmental encounters by the species [2], whereas in computer stereo vision, the configurations must often be carried out by the users.

The traditional computer stereo algorithms rely on a multitude of parameters which determine their accuracy and performance [3]. In contrast, deep learning-based stereo methods require fewer parameters to be configured by the users during disparity inference[1]. The "maximum disparity" which is the ceiling value of the disparity search range (DSR), still remains as a user-configured parameter in most of the state-of-the-art deep learning techniques. That includes the top performing techniques on stereo vision benchmarks [4], [5], [6], [7], [8]. In addition, all such techniques use the minimum disparity (which is the floor value of DSR) as zero. Therefore, if the maximum disparity can be estimated automatically as part of the deep stereo pipeline, deep stereo networks can be independent of the user-specified parameters during stereo disparity inference. The development of such a technique would lead to new possibilities such as:

- *Failsafe stereo vision systems which can be deployed into unknown environments without explicit configuration by the users*
- *Reduced disparity estimation errors in the absence of overestimation or underestimation of parameters by the users*
- *Plug-and-play stereo sensors in robotics which do not require users to be aware of the internal configuration of stereo algorithms or their parameters*

[1] In deep learning stereo, "disparity inference" refers to the use of a trained deep stereo model to produce disparity maps from stereo images. In supervised deep stereo methods, disparity inference takes place after the training process is complete.

- *Higher performance and better utilization of computing resources during disparity inference*

Past studies on "automatic disparity search range estimation" have resulted in methods which can be used with traditional stereo methods to extract a suitable disparity search range (or the maximum disparity) from stereo image data. However, such methods depend on either some initial disparity value extracted by using feature descriptors [9], [10], [11], [12], limiting the subsequent disparity search with seed points [13], [14], [15], setting user-defined initial values [16], [17], limiting search space with image pyramids [18], [19], [20], [21] or observing historical values in image sequences [22], [23], [24]. Moreover, they all are required to be implemented as separate pre-processes to the stereo disparity estimation process which prolongs the forward propagation time while still being user dependent.

As far as the state-of-the-art deep stereo networks are concerned, most of the research work in the domain has been focussed on improving the accuracy of disparity estimations [25]. Consequently, their output accuracy during inference has been left dependent on the ability of the human users to determine a suitable maximum disparity value using their own judgement, trial-and-error or using methods that are external to the stereo disparity estimation network. The research study presented in this thesis aims to examine the possibility of making deep stereo networks independent of such user-specified parameter values, by making the disparity search range estimation an integral part of a deep learning network.

1.1 Fundamentals of Stereo Vision

Binocular stereo vision attempts to recover depth from the positional differences between matched points from two images of the same scene captured with mutually displaced cameras as shown in Figure 1.1. The illustration shows two cameras (*Camera 1* and *Camera 2*) registering image points (P_1 and P_2 on their respective image planes) for a point P on an object. The optical centres of the two cameras are denoted by C_1 and C_2 and the distance between C_1 and C_2 is known as the baseline of the stereo system. The plane that goes through P, C_1 and C_2 is called the "epipolar plane" while the points of intersection (E_1 and E_2) between the image planes (shaded in grey) and the baseline (C_1C_2) are called "epipoles". According to the geometry of the illustration,

image points (P_1 and P_2) must be located on the lines of intersection between the epipolar plane and the image planes which are called “epipolar lines”. The constraint introduced by the geometry of the epipolar lines is referred to as “epipolar constraint”.

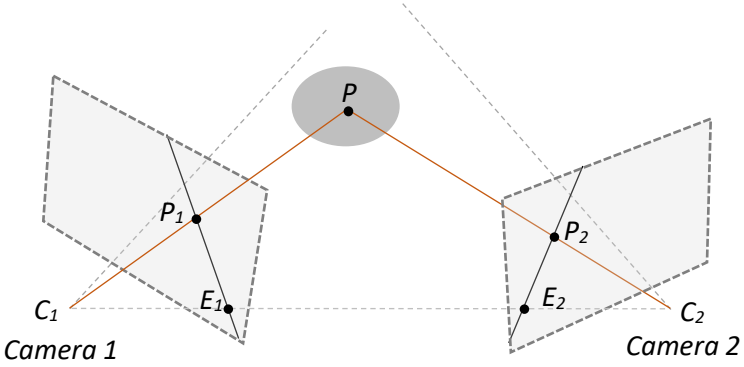


Figure 1.1: Two camera stereo system showing the epipolar geometry

One of the key tasks in stereo vision is to establish the matches between the points in the two images. If unconstrained, search for a pixel in one image would have to include all the pixels in the second image which would be computationally expensive. However, due to the epipolar constraint, it is sufficient to search along the epipolar lines. In a typical stereo camera set-up, the cameras are arranged in what is known as the “rectified configuration” which makes the epipolar lines parallel to the horizontal axis of the image planes. Therefore, when cameras are in rectified configuration, the search for a match for a pixel in one image, can be carried out along the same row in the other image. However, if the two optical axes are misaligned, horizontal and parallel epipolar lines can still be achieved through a rectification process which involves geometric transformation. Figure 1.2, depicts a camera pair in a rectified configuration.

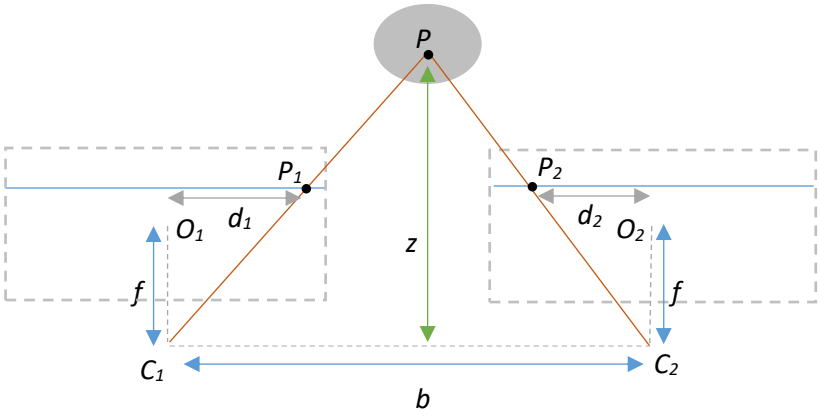


Figure 1.2: Two cameras in rectified configuration having horizontal epipolar lines resulting in correspondence search along the horizontal rows of image points

In Figure 1.2, d_1 and d_1 denote the horizontal offsets of the image points from the image centre whereas *z* denotes the perpendicular distance (or depth) to the point *P* from the baseline *b*. The focal length of the two cameras is equal to *f*. The centres of the image planes of the two cameras C_1 and C_2 are denoted by O_1 and O_1 respectively.

If the total disparity between the two image points (P_1 and P_2) in Figure 1.2 is denoted by *d* which is equal to the sum of d_1 and d_2 (or the difference between the *x* coordinates of P_1 and P_2 in pixels). The relationship between the depth *z*, baseline *b*, focal length *f* and the disparity *d* is given by Equation (1.1) below.

$$z = \frac{bf}{d} \tag{1.1}$$

If a calibrated stereo camera with fixed baseline and focal length is concerned, the depth to an object point can be derived by computing the disparity between the two corresponding image points using Equation (1.1). Therefore, the binocular stereo-based depth estimation problem can be solved by estimating the disparity between matching pixels in the stereo images.

1.2 Disparity Estimation

Disparity estimation in binocular stereo vision needs to find answers to the following fundamental questions.

1. How to measure the similarity or dissimilarity between two pixels?
2. How much to search for a match along the epipolar lines?
3. How to select the best match given a set of closely matching points?
4. How to handle any exceptions (ambiguous matching, occlusions etc.)?

Most of the traditional disparity estimation techniques attempt to solve the above using the conventional/legacy stereo pipeline [26] while state-of-the-art techniques utilize end-to-end deep neural networks [25] to achieve the same. However, both types of techniques depend on user inputs to choose parameter values, essentially depending on the user when considering the fundamental questions above.

1.2.1 Parameterization in Disparity Estimation

Parameterization is not the enemy when studying the science related to any phenomenon. In fact, parameterization provides a basis for systematic analysis of the

same. However, the issue with the parameterization is that the values of the parameters must be known in advance before the systems can be used. Unless specific algorithms are developed to learn parameters exclusively from the input data, the accuracy of the results may depend on user inputs.

Recent advances in machine learning and deep learning in general have considerably reduced the requirement for user configuration of parameters in vision systems including binocular as well as monocular systems. However, in stereo vision, there is a key parameter referred to as the maximum disparity (i.e., the ceiling value of the disparity search range) which is still required to be configured by the users.

This thesis is based on a study conducted in order to develop a stereo disparity estimation method which does not require users to configure maximum disparity when inferring disparity maps from rectified stereo images.

1.3 Research Objectives

This study is aimed at achieving the three main objectives outlined below with the respective courses of action.

1. Developing a methodology to automatically estimate the maximum disparity parameter in stereo vision algorithms without requiring user inputs

   This is to be achieved through an analysis of the matching cost volume creation process in stereo, using traditional stereo techniques. This includes experiments with standard stereo datasets and rudimentary stereo algorithms to observe the movement of matching cost extrema within a cost volume when it is constructed one layer at a time. The results are then used to study the correlation between the movement of the matching cost extrema and the maximum ground-truth disparity from the datasets. Based on the findings, the possibility of deriving a metric which can serve as the cost volume termination criterion, is explored.

2. Formulating machine learning friendly metrics which can be used to eliminate the manual configuration of disparity search range

Based on the identified relationships between the cost extrema movement and maximum disparity, a metric is formulated in such a way that it can be efficiently calculated at each layer of the cost volume while it is being built. The mathematical foundation of the metric needs to be established to provide a theoretical basis to determine the applicability of the metric with varying scene conditions. Standard stereo datasets are used with traditional stereo methods as well as a custom-built deep learning stereo method to determine the feasibility of using the metric for maximum disparity predictions.

3. Developing a fully autonomous stereo vision algorithm that does not require any pre-configuration by users during inference

   The developed metric is used with an improved deep learning stereo network to automatically terminate the cost volume creation process as soon as the maximum disparity is detected. Once the complete automation is achieved, the algorithm is used with standard stereo datasets to determine the accuracy of the maximum disparity predictions in sequences of images with ground-truth disparity. Furthermore, the disparity accuracy is investigated using the standard stereo accuracy measures while also analysing the possible cumulative gains in performance.

1.4 Contributions

The research work outlined in the thesis aims to advance the field of research through the contributions made by introducing the following.

- The first deep learning-based stereo disparity estimation method to be completely independent of user-specified parameters during stereo disparity inference
- A metric that can be used with existing state-of-the-art stereo vision algorithms so that they can also estimate the maximum disparity automatically as part of the forward propagation phase

- A new benchmark for fully autonomous stereo vision applications which is different from the existing benchmarks that mainly focus on disparity estimation accuracy
- A method to better utilize computational resources in stereo disparity estimation especially in parallel computing (GPU) environments
- The first deep learning-based stereo algorithm that does not require users to reconfigure in response to the changes in resolution and baseline selection (in multi-baseline systems)

1.5 Thesis Overview

Chapter 2 introduces the background of stereo vision both in terms of traditional and state-of-the-art deep learning-based stereo techniques covering the basics of the conventional stereo pipeline and the evolution of deep stereo methods. A background analysis of automatic disparity search range estimation follows, with the chapter summarising the findings which forms the basis for the research study. Research methodologies, research questions, and the main contributions of the study are discussed in Chapter 3.

Chapter 4 outlines the development of a novel metric which can be used to approximately estimate the maximum disparity for a scene. A probability based mathematical formulation is discussed in detail before the metric is tested on stereo datasets to identify the conditions under which, totally deterministic results can be achieved. The impact of the metric on existing stereo techniques in terms of computational time complexity is discussed in Chapter 5. The work includes a detailed algorithmic analysis of CPU and GPU based stereo vision algorithms to investigate the feasibility of using the metric to estimate the maximum disparity without causing significant delays.

In Chapter 6, a foundational level deep learning-based stereo method is developed step-by-step to test the feasibility of using the developed metric with deep learning stereo. The aim is to estimate the metric at each disparity during the forward propagation phase of the deep network. Although, the metric is not used to terminate the cost volume creation process at this stage of the study, Chapter 6 includes

experiments with stereo image sequences to verify the correlation between the value of the metric and the maximum values reported in ground-truth data.

Chapter 7 introduces an end-to-end deep learning technique which utilizes the metric to terminate a layer-wise cost volume creation process at a suitable maximum disparity to achieve complete automation. A special training regime is introduced which is aimed at improving the accuracy of the disparity predictions and the maximum disparity predictions simultaneously. A 4-stage extensive evaluation of the trained model is conducted using standard stereo datasets and stereo images captured with a custom-built stereo camera.

Finally, Chapter 8 includes a summary discussion on how the study managed to answer the research questions, achievement of the research objectives and future work related to the metrics and algorithms developed as part of this study.

CHAPTER 02 - LITERATURE REVIEW

This chapter starts with an overview of the traditional stereo vision techniques focussing on the legacy stereo pipeline (in Section 2.1) which also served as the foundation for early deep learning stereo techniques. A comprehensive review of the evolution of deep learning-based stereo techniques is presented in Section 2.2 which outlines the key milestones in development and significant contributions that made the state-of-the-art possible. Section 2.3 shows how the existing stereo disparity estimation techniques still require users to configure certain parameters and provides a list of state-of-the-art techniques which also rely on user-configured "maximum disparity" parameter. Section 2.4 contains the results of a focused review of literature on automatic disparity search range detection techniques and outlines their merits and drawbacks. A summary of the findings is provided at the end of the chapter in Section 2.5.

2.1 Traditional Stereo Vision

Stereo vision has remained one of the most important and extensively studied areas of research in machine vision for decades [25], [27]. Like stereopsis in animal binocular vision [1], computer stereo vision techniques rely on the positional differences of the objects in images captured by two or more imaging devices. Unlike biological vision which benefits from additional positional information from phenomena like vergence movements [28] of the eyes, computer stereo vision depth estimations often have to rely on image differences alone [29], except when using some additional active sensors like laser, in a sensor fusion application with stereo [30].

Depending on the extent to which the disparity information is recovered from the left and right images, stereo techniques can be broadly segregated into two branches: dense stereo and sparse stereo methods [31] with some intermediate algorithms being referred to as semi-dense stereo methods [32]. Dense methods attempt to recover depth for all the points in the images whereas sparse techniques focus on interest points or specific objects of interest in the scene. Semi-dense methods in contrast, involve selective dense matching in a subset of image points [33] (e.g., matching textured areas while leaving the texture-less regions unmatched).

2.1.1 The Conventional Stereo Pipeline

The conventional stereo algorithms follow the characteristic processing stages which are collectively referred to as the legacy stereo pipeline. It is comprised of the following four stages connected in a cascading arrangement [26].

1. Matching cost computation
2. Cost aggregation
3. Disparity selection
4. Disparity refinement

The traditional two-frame-stereo process starts with establishing a similarity (or dissimilarity) measure which can provide a numeric representation that serves as the matching cost between two pixels. Common techniques for calculating the matching costs include Absolute Difference (AD), Sum-of-Absolute-Difference (SAD), Sum-of-Squared-Difference (SSD) and Normalized Cross Correlation (NCC) with non-parametric measures such as Rank Transform (RT) and Census Transform (CT) [26]. When the costs are calculated at each pixel location for all possible disparities, they are usually organized into a 3D (or higher dimensional) data structure which is known as a cost volume [34].

Based on the cost aggregation and disparity selection stages, conventional stereo methods can be further classified into three groups: “Local” methods, “Global” methods and “Semi-Global” methods [25]. Cost aggregation in local methods involve combining the matching costs over a support region (also referred to as a support window, mask, or an aggregation window [35]) which helps reduce matching uncertainties. Global techniques on the other hand, often skip the same (stage 2) and feed the computed costs to the disparity selection stage directly [26].

During the disparity selection stage, the most fitting disparities are selected based on the matching costs associated with each pixel. Local methods employ a “Winner-Takes-All (WTA)” approach by selecting the disparities with the lowest associated cost. In contrast, global methods search for disparity assignments that minimize a global cost function (i.e. energy function) involving all costs at all disparities [36] via optimization techniques such as Graph Cuts [37] or Belief Propagation [38]. Semi-global methods attempt to minimize the combination of dissimilarity cost and

regularization cost along a limited set of paths or directions across the cost volume to achieve computational efficiency [39].

Once the disparities have been selected, they are further refined during the disparity refinement stage (which is also called post-processing or fine-tuning stage). This final stage includes regularization and occlusion detection [26]. Left-to-right consistency check is a common practice in stereo vision where two disparity maps (left and right disparity maps depending on which image is being used as the reference image for matching cost computation) are compared to identify the false matches [40]. If inconsistencies are found, various strategies can be used to remove them. For example, inconsistent disparities can be replaced with consistent values extracted (or interpolated) from the neighbouring pixels [39].

Until the deep stereo methods came to the forefront, the accuracy improvements in traditional stereo continued through confidence estimation methods for stereo [41], [42]. Accuracy of the confidence estimation techniques themselves kept improving with strategies such as training random forest classifiers to combine confidence measures [43] and supervised training to predict the correctness of matches and confidence estimates [44].

2.2 Evolution of Deep Stereo Methods

The advent of the artificial neural networks in machine vision has significantly benefited the stereo vision techniques in terms of performance and accuracy [27]. However, the early deep stereo vision algorithms did not replace the full stereo pipeline. Instead, the matching cost computation was one of the earliest to get replaced by neural networks [25]. In one of the earliest examples [45], the researchers used a Siamese network to produce matching scores for image patches surrounding the pixels being matched. Their solution used cross based cost aggregation followed by semi-global matching and a left-to-right consistency check for disparity refinement. Although considerable gains in accuracy were achieved at the time, the technique required additional processing to find the best match from a series of closely matching image patches.

Improvements to the visual correspondence between image patches continued with the addition of multi-scale feature extraction [46] which allowed comparisons to be

made in terms of spatial features instead of pixel intensity-based measures. Another similar deviant of the Siamese networks-based stereo methods used convolution over image patches to produce feature representations (i.e., feature volumes) before combining them into a matching score by using an inner product layer [47]. As reported by the authors, the change led to more accurate results with faster processing times. However, their technique still relied on conventional methods for cost aggregation and post-processing.

The availability of training data with ground-truth remained a hurdle for the early work in deep stereo vision. Although the stereo benchmark datasets such as Middlebury 2001-2014 [48], [49], [50], [51], [52] and KITTI 2012-2015 [53], [54] were extensively used for supervised learning at the time, the number of image pairs available was still not large enough. The introduction of the Scene Flow dataset [55] in 2016 with over 35,000 stereo image pairs with ground-truth data was a significant contribution to the stereo vision community. Later additions such as ETH3D [56], Driving Stereo [57] and GENUA PASTO [58] have made training data more accessible to the researchers.

In order to create adaptable stereo vision models (and to provide a solution for the scarcity of training data), some researchers proposed self-supervised network models which can learn from data during inference, without pre-training. Most of such methods used left-to-right consistency in stereo as the guiding rule for learning [59], [60] while some used active sensing technologies like IR – cameras later [61]. The work presented in [60] shows an attempt to use image warping error instead of disparity error to train a deep stereo vision model and demonstrates the self-improving ability. Since the data used for training is not known in advance, achieving a higher accuracy with self-supervised stereo remained a challenge.

2.2.1 End-to-End Learning

When the Scene Flow dataset was introduced, a disparity estimation technique called DispNet [55] was also introduced. DispNet was capable of regressing disparities from images directly using an encoder-decoder architecture (made possible by a series of convolutional and deconvolution layers). Some variations of the encoder-decoder architecture-based methods used the same at multiple stages. The work in [62] includes two such networks connected in series. However, they all still required a

correlation layer for scalability (to produce smooth disparity maps across scenes with large variations in disparity) and therefore rely on user inputs to determine the extent to which correlation is tested.

Another end-to-end stereo framework was introduced in [63] which used a highway network to produce a cost volume and a disparity refinement network made of convolutional and fully connected layers to produce a disparity map from the cost volume. However, the feature and disparity refinement networks required separate training which affected the scalability. The overall architecture also included conventional post-processing steps such as left-right consistency, sub-pixel optimization and smoothing with filters.

A scalable deep stereo method marked a key milestone in 2017 when GC-Net [64] incorporated geometry and context information (learned through high level feature representations of image data) to produce disparity predictions using an end-to-end framework. The key contribution of their work was the introduction of "differentiable disparity regression" which allowed gradients to be backpropagated through the disparity regression network by using the gradient friendly "softargmin" operation in place of the previously used "argmin" operation. Inspired by the success of residual learning, GC-Net also featured residual blocks of network layers to enable high frequency information to be used by the downstream network layers when making predictions which simultaneously improved the gradient flow.

2.2.2 Deep Learning Stereo Pipeline

The ability to replace all stages of the legacy stereo pipeline with an end-to-end deep network gave rise to a new class of deep learning-based stereo vision algorithms. The resulting architectural pattern is referred to as "3D regularization structure" in literature [25]. The algorithms which follow the 3D regularization structure, can be identified by their three characteristic stages of processing which include: (1) feature extraction, (2) cost volume creation and (3) regularization. The following sections outline the different algorithms which have achieved better results by improving one or more stages of the "3D regularization structure".

2.2.2.1 Improved Feature Extraction

EdgeStereo [65] has improved feature extraction using edge contours and related constraints by incorporating edge cues into the loss function used for training, thereby improving the overall accuracy of disparity estimations. FADNet [66] also optimizes feature extraction further through the introduction of point-wise correlation layers and dual-residual blocks concatenated at multiple scales. CFP-Net [67] is another example of a multi-scale feature matching network with improved correspondence. Similarly, SegStereo [68] relies on the feature network to extract semantic information which can enhance disparity predictions through better matching while also featuring the ability to learn both supervised and unsupervised. In contrast, Guided Stereo [69] uses external LiDAR measurements to enhance features learned by the convolutional network.

2.2.2.2 Optimizing Multiples Stages

Certain models enhance feature extraction and cost volume at the same time. The network presented in [70] which is called AMNet, uses a modified version of the ResNet [71] feature extractor to produce depth separable features. AMNet then creates a multi-scale cost volume which consists of sub-volumes for feature distances, concatenated features, and depth wise correlation. In contrast, the work in [72] incorporates a confidence estimation network to alter the generated cost volume which enhances the unimodality of the cost distribution which is helpful when regressing disparities with the “softargmin” operation [64].

GWC-Net [73] includes a group wise correlation stage where the extracted feature vectors are divided into segments before calculating a correlation score which is then combined with the concatenated feature volume to create a combined cost volume. SSPCV-Net [74] includes a pyramid cost volume to capture semantic information as well as multi-scale spatial information. Cost volumes with multi-scale information with or without pyramid cost volumes, require large amounts of memory [75] which can become a bottleneck during training and inference [25].

Moreover, there are networks that incorporate improvements to both the feature extraction and regularization stages. For example, PSM-Net [76] uses a spatial pyramid pooling (SPP) module to learn context information at different scales by having multiple convolutional layers in parallel, with each parallel path having different

feature lengths. A 3D convolutional encoder-decoder architecture is then used for cost aggregation to achieve better accuracy compared to regularization using basic 3D convolutional layers.

Meanwhile, some SOTA algorithms use additional cost aggregation layers within the end-to-end deep learning-based architecture. GA-Net (Guided Aggregation Network [7] with variants like [77]) is one such network which has implemented semi-global cost aggregation in the form of a deep neural network. However, the resource utilization problem can be made worse by having multiple cost volumes or elaborate aggregation stages especially when the maximum disparity is high [5].

2.2.3 Stereo with Recurrent Neural Networks

Based on the recurrent neural networks (or RNNs), some researchers have attempted to integrate left-to-right consistency into the learning process. One such study [78] uses "Highway" networks [79] to produce a cost volume which is used as input to a stacked convolutional LSTM model called LRCR (Left-Right Comparative Recurrent). LRCR is trained to evaluate left-right consistency to produce more refined disparities. However, the training process must be conducted in two stages due to the increased complexity of the network.

An RNN-based stereo matching network for video is demonstrated in [80]. It uses a concatenated feature volume from the input images and disparity estimates by an encoder-decoder network as input to LSTM modules to learn temporal relationships in both input images and the disparity maps. Despite achieving a higher level of accuracy compared to other self-supervised methods, the feature volume construction phase depends on user input for disparity range.

Recurrent neural networks have also been used for confidence estimation and matching cost refinement in stereo vision. For example, the study in [81] uses a set of LSTM modules to predict a confidence score for a pixel location given its associated cost vector whereas the work in [82] demonstrates the use of a recurrent neural network to correctly identify the location of match points.

2.2.4 Recent Studies in Deep Stereo

The available literature shows that the recent studies have focussed on incremental developments and additions to the 3D-regularization structure to achieve higher levels of accuracy, performance, and optimal resource utilization. For example, some key studies have used confidence and uncertainty-based measures [83], [84], [85], [86], adaptive confidence estimation networks [87], [88], [72], active inputs [69], [89] and multi-level robust feature representations [90] to improve disparity accuracy in deep stereo. On the other hand, certain hierarchical approaches [4], [5] have improved performance in addition to the accuracy. It is noteworthy how the network in [5] does not explicitly store a 3D (or higher dimensional) cost volume which leads to the efficient use of GPU memory. However, its multi-level initialization stage computes the matching costs at all disparities up to a maximum value chosen by the user. In another unique hierarchical network found in [4], Neural Architecture Search (NAS) has been used to incorporate human knowledge of deep networks to select the best feature and cost aggregation network structures for a deep stereo network. Their study marks a significant milestone due to the automatic selection of the optimal network architecture in deep stereo. Nevertheless, the solution depends on user-defined parameter values for maximum disparity like many other deep stereo techniques discussed so far.

2.3 Reliance on Parameters in Stereo Vision

The traditional stereo methods (local, global and semi-global alike) use parameters such as the mask size [91], maximum disparity (and related parameters in the form of maximum disparity range, disparity levels, displacement range) [92], penalties [39] and MRF parameters [93]. In addition, there are other custom defined parameters in some handcrafted cost functions used by various stereo techniques [51]. An effort to estimate parameters from the input images can be seen in [94] which features a MRF parameter estimation technique for global stereo for algorithms such as Graph-Cuts and Belief Propagation. SGM-Net [95] on the other hand, uses a deep learning network to learn penalty parameters in semi-global-matching which has resulted in accuracy improvements.

In contrast, end-to-end deep learning stereo methods use only a limited number of user-configurable parameters during inference [27]. Most of the parameters associated with such methods are usually selected and optimized during the training phase. Once the training is complete, only a few parameters are required to be set by the users for disparity inference. However, during this background study it was found that to this date, the maximum disparity remains a user-configured parameter in many deep learning stereo techniques. Table 2.1 includes a list of top performing algorithms on the KITTI 2015 stereo benchmark at the time of writing. They all require users to specify a suitable maximum disparity value during disparity inference.

The publicly available source code for some methods in Table 2.1 (e.g., [4], [6], [7], [67], [72], [73], [76], [96]) and the published method descriptions indicate that most of the methods define their cost volumes in the form of fixed-sized tensors in memory during the forward propagation through the network. Subsequently, the elements in the cost volumes get filled with the corresponding features or concatenated features at each disparity in a sequential manner (starting from disparity zero to some maximum disparity value specified by the user). The finished cost volumes are then passed on to the regularization stage during which the costs get further aggregated by the cost aggregation layers. Finally, disparity regression and up-sampling operations produce disparity maps at the original image resolution.

Name	*Title*	*Manual Max. Disp.*	*Typical Value/Values*
LEAStereo [4]	*Learning Effective Architecture Stereo*	Yes	192
HITNet [5]	*Hierarchical Iterative Tile-refinement Network*	Yes	320, 160, 256
CSPN [6]	*Convolutional Spatial Propagation Network*	Yes	192
GA-Net [7]	*Guided Aggregation Network*	Yes	192
AMNet [70]	*Deep Atrous Multi-scale Stereo Disparity Estimation Network*	Yes	192
AcfNet [72]	*Adaptive Unimodal Cost Filtering for Deep Stereo Matching*	Yes	192
AANet [96]	*Adaptive Aggregation Network*	Yes	192
Edge Stereo [65]	*Multi-Task Learning Network for Stereo Matching and Edge Detection*	Yes	192
SSPCV-Net [74]	*Semantic Stereo Matching with Pyramid Cost Volumes*	Yes	256,192
HSM [97]	*Hierarchical Deep Stereo Matching*	Yes	384,512
PSM-Net [76]	*Pyramid Stereo Matching Network*	Yes	192
PDS [98]	*Practical Deep Stereo*	Yes	255
CRL [62]	*Cascade Residual Learning*	Yes	160/40
GWCNet [73]	*Group Wise Correlation Network*	Yes	192
FADNet [66]	*Fast and Accurate Network for Disparity Estimation*	Yes	192
CFP-Net [67]	*Cross-form Pyramid Network*	Yes	Not Given
SegStereo [68]	*Exploiting Semantic Information for Disparity Estimation*	Yes	96
DispNetC [55]	*DispNet with Correlation Layer*	Yes	160/40
LRCR [78]	*Left-Right Comparative Recurrent Model for Stereo Matching*	Yes	228

Table 2.1: A list of state-of-the-art stereo disparity estimation techniques which rely on users to configure a maximum disparity value / disparity search range. *Table contains deep stereo networks (with a published method) from the top 100 algorithms on KITTI 2015 benchmark (i.e., anonymous submissions with no method details have been omitted while variants of the same network have also been removed).*

2.3.1 Deviations and Exceptions

The background study also found some deep stereo methods which can reduce the disparity search space. For example DeepPruner network [99] which is based on PatchMatch stereo [16], is able to reduce the disparity search range on a per pixel basis with the help of a deep neural network. However, the confidence estimation process still requires the minimum and the maximum values to be specified by the user. Advantages of the method include the savings in processing power due to the reduced size of the cost volume. The main drawback however is that the output accuracy depends on the manual parameter selection by the users.

Some encoder-decoder architecture-based algorithms can predict disparities without requiring users to provide the maximum disparity explicitly. For example, in a stereo vision-based robotic surgery application [100], researchers have been able to employ an end-to-end network based on the same paradigm to regress disparities directly. Their method uses the output disparity predictions to reconstruct the source images so that the reconstruction error can in turn be used to train the network. In another similar approach [101], the disparity predictions by the network are used to shift the pixels in the right image, creating a warped image. The warped image is then compared against the reference image for training. Such methods are suitable when the maximum disparity is limited (like in the case of robotic surgery where inherently small baseline leads to smaller disparities between the left and right views). In contrast, the DispNet [55] which also uses the encoder-decoder architecture, includes a correlation layer to handle large disparities. Correlation layers can efficiently handle large horizontal displacements provided that a suitable range is set by the users.

2.4 Automatic Disparity Search Range Estimation in Stereo

In deep stereo methods listed in Table 2.1 above, the maximum disparity corresponds to the disparity dimension of the cost volume. Therefore, in deep stereo methods that build a cost volume, the maximum disparity determines the disparity search range in full (as the minimum disparity is generally assumed to be zero). Although the automatic DSR estimation has not been studied much in deep stereo (as found in the background study), many traditional techniques in the past have attempted to estimate the same from the input data.

Such techniques can be categorized into two groups:

1. Scene-based DSR estimation techniques
2. Progressive DSR estimation techniques

2.4.1 Scene Based DSR Estimation Techniques

The scene based DSR estimation techniques aim to recover a suitable DSR from each individual stereo image pair. They commonly use feature descriptors, disparity priors, support points, stochastic properties or coarse-to-fine pyramid approaches.

2.4.1.1 Feature Descriptor Based Techniques

Past studies [9], [10] show efforts to estimate the disparity search range with feature descriptors like SIFT [102] and SURF [103]. For the results to be accurate, the relevant feature descriptors need to be able to match key points in the images associated with the foreground objects in the scene, which correspond to the largest disparity. Although such techniques can automatically find the maximum disparity, feature descriptors have their own parameters and thresholds [10] which must be specified and fine-tuned by the users to achieve reliable results. Furthermore, the time complexity of the stereo algorithms can be affected by the processing delays associated with such feature descriptors [11] when used in-line with the disparity estimation process.

2.4.1.2 Stochastic Methods

Authors of the work presented in [17] have used the correlation between the left and right images as a function of the distance between the images to form a variogram which has been subsequently used to estimate the maximum disparity. They have managed to obtain reasonably accurate DSR estimations using a hand-crafted heuristic rule developed by experimentation.

PatchMatch stereo [16] has been developed based on the assumption that large groups of pixels in digital images can be assigned to a fewer number of planes (inspired by the PatchMatch correspondence algorithm in image editing [104]). The method attempts to assign pixels iteratively to a set of planes starting from a random association and converges well under most practical scenarios. However, PatchMatch stereo relies on a user-specified value in the form of the “maximum allowed disparity” which gets reduced by half during each iteration of the refinement process. This makes

the initial value crucial for the overall performance. However, the technique highlights the feasibility of using the law of large numbers for disparity assignments in the face of uncertainty.

2.4.1.3 Disparity Priors and Support Points

The work in [13] shows an effort to use a set of initial support points to eliminate the ambiguities associated with disparities of the remaining points. The objective is to build a disparity prior based on triangulation of sparse but well-established match points which in turn determines the disparity search range for the ambiguous points. Robustness of the point correspondences is determined using a handcrafted method that measures the L1 distance between filtered grids of blocks in two images. Those blocks are also checked for left-to-right consistency before selection. However, the study indicates that the process must be initialized with a user-defined maximum disparity. When obtaining the published results, the researchers have in fact used an initial value equal to half the image width.

In a similar approach, the study presented in [15] uses robustly established matches to add constraints to the subsequent calculations of disparity. Initial set of reliably matched points have been obtained using normalized cross correlation and a threshold value. The method which is based on the principle of smooth variation of disparity, can remove the ambiguities gradually.

The algorithm presented in [12] starts with a limited set of seeds produced by using a pre-matcher and then grows to produce a semi-dense disparity map which is robust against the inaccuracies in the initial seeds. The results indicate that the computational time depends on the quality of the initial correspondence seeds despite the output disparity maps are unaffected. This can lead to convergence issues at higher resolution levels with larger DSR values.

Adaptive selection of a DSR value based on the other pixels in the neighbourhood is demonstrated in [14]. The process uses disparity values from the previous image rows to limit the DSR for the pixels which are processed later. However, at least one row must be processed with an initial DSR value defined by the user. Moreover, an incorrect initial setting can cause error propagation into the subsequent iterations.

2.4.1.4 Coarse-to-Fine and Image Pyramid Approaches

The study outlined in [18] has used an image pyramid in which the low-resolution layers are used to create disparity maps with an initial disparity search range before using the results to better estimate the disparity search space for the pixels at higher resolution levels. Nevertheless, their methodology requires some disparity search range to start the process and the images must be processed at multi resolution levels which adds to the overall computational complexity.

In the methodology published in [19], the process starts with an initial disparity search range and then keeps expanding the range iteratively until the original image size is reached. Their method is based on the findings of [20] which introduces a confidently stable matching approach that helps identify the largest unambiguous regions in matching. However, the matching process must be repeated at many different resolution levels until the final value is chosen which can be time consuming if used in combination with an already complex stereo disparity estimation algorithm. The accuracy of the results also depends on the initial disparity search range selection.

A modified coarse-to-fine pyramid-based approach is proposed in [21] with enhancements to subsampling scale selection and tuning of the histogram threshold. Although incremental developments in accuracy and temporal disparity accuracy is observed, initial disparity range selection must still be made by the users.

2.4.2 Progressive DSR Estimation Techniques

Progressive techniques estimate disparity search range for image pairs from the previous stereo images in a sequence. These methods are typically used for recovering depth from videos. The objective is to add constraints on disparity based on the predicted disparity flow values as shown in [22]. A recent study [23] demonstrates how progressive disparity range estimation can be used in an actual application involving a driver assistance system. They use BRISK based feature point matching between the left and the right images to create a sparse v-disparity [24] histogram from which the disparity search range can be estimated at the beginning. Subsequent estimations are then calculated by parabola fitting to the row-wise dense estimations and choosing the best fitting path. This is another example method that relies on a key

point extraction algorithm for its accuracy which requires careful selection of thresholds and other parameters related to the key point extraction method itself.

2.5 Summary of Findings

In this chapter, the past studies on stereo vision were discussed. They indicate that the conventional stereo methods implement the stages of the legacy stereo pipeline which include matching cost computation, cost aggregation, disparity selection and disparity refinement. Early neural networks-based stereo vision methods also followed the same architectural paradigm and replaced some conventional stages (especially the matching cost computation) with deep learning-based counterparts. Introduction of the differentiable "Softargmin" operation to replace the commonly used "argmin" operation allowed disparity regression to be carried out end-to-end in a gradient-friendly and scalable manner. The possibility of producing disparity maps in a single forward pass through a deep network led to a new architectural pattern called "3D Regularization Structure".

The new end-to-end architectural blueprint has three characteristic stages of processing which include feature extraction, cost volume creation and disparity regularization. Based on the 3D regularization architecture, the progression of deep stereo networks continued with some techniques incrementally optimizing one or more stages. For example, certain techniques have used elaborate feature extraction networks while others have utilized complex cost aggregation blocks, regularization networks and higher dimensional cost volumes. Studies indicate that the use of higher dimensional cost volumes lead to excessive resource utilization during both training and disparity inference. More recent studies have focussed on optimizing the performance and computational resource utilization of the stereo algorithms in addition to accuracy by introducing architectural changes to the stages of the deep stereo pipeline.

During the literature review, it was further observed that the conventional methods have various user-specified parameters that are required to be set or configured by the users in order to obtain accurate disparity maps. In contrast, deep learning-based algorithms have only a fewer number of stereo vision related user-specified parameters. However, the maximum disparity or the ceiling value of DSR, is still

required to be configured by the users based on their understanding of the scene structure. This includes even the top performing deep stereo algorithms on stereo benchmarks such as KITTI and Middlebury. Further, investigations into the methods which do not expect the maximum disparity as part of input, revealed that they rely on other user-configured parameters.

The review revealed two types of automatic DSR estimation techniques. The first uses individual scene-based methods to estimate DSR from individual stereo image pairs. They use feature descriptors, disparity priors/seeds, user-defined initial DSR estimates and coarse-to-fine image pyramid approaches to estimate a suitable DSR for a given scene. In contrast, progressive DSR estimation methods use DSR values from previous images of a sequence to limit the disparity search in subsequent calculations. However, the most common drawback of both types of methods according to the available literature is that they depend on other user-defined parameters or initial manual DSR estimates.

CHAPTER 03 - RESEARCH METHODOLOGY

Introduction

The literature review in Chapter 2 revealed that the traditional stereo techniques have many user-configured parameters such as the maximum disparity or disparity search range (DSR) while most deep learning stereo methods require the maximum disparity alone as a user-configured input during disparity inference. Therefore, it is imperative that if a deep stereo network can estimate the maximum disparity parameter automatically by itself without user intervention, then full autonomy in stereo disparity estimation can be achieved. However, the literature review also revealed that the existing automatic DSR estimation techniques have drawbacks of their own such as the additional computational complexity and custom parameters or thresholds which require selection and fine-tuning by the user before accurate results can be obtained. This thesis presents a new technique to estimate the maximum disparity automatically as part of the forward propagation stage of a deep stereo network. The current chapter outlines the research methodology used to develop the technique to meet the objectives identified in Chapter 1.

3.1 Research Questions

To achieve the objectives of the study, the following research questions need to be answered.

1. What phenomenon or associated metric can be used to determine the size of a cost volume in deep learning-based stereo algorithms, without user intervention?

   In the background study, it was found that most of the deep stereo techniques use a cost volume creation process which starts by defining a fixed-sized cost volume which is filled with layers of features or concatenated features learned from the stereo images, up to the maximum disparity defined by the user. If the cost volume can be built layer by layer and some layer-wise metric is able to provide an indication of the point at which that process should terminate, then that would determine the maximum disparity for the scene automatically.

2. Would such a metric be computationally efficient enough to be incorporated into a stereo method?

   Stereo disparity estimation process is a computationally demanding operation on its own. Deep stereo techniques have an inherently higher order of growth in computational complexity which is the main reason why they usually require specialized hardware (GPUs) for training and even inference in most cases. Therefore, any additional computations introduced to the already complex process can have a negative impact on the performance of the overall network. Therefore, it is not unreasonable to question if such a metric would introduce too much of a burden especially in terms of the order of growth of computations pertaining to the overall algorithm.

3. Can such a metric leverage on the stereo matching accuracy of the deep stereo network for accuracy of maximum disparity predictions?

   Deep stereo networks are usually trained for the output accuracy in terms of the disparity maps they produce. The training process uses the accuracy (or error) of the predictions to adjust the network parameters. Hence, it is important to ask if the accuracy of the predicted maximum disparity (by the metric) would improve with the matching accuracy of the network. If not, other optional criteria may have to be used to train the network.

4. If such a metric is integrated into a deep learning network, will the model require a special training regime to converge?

   As found in the literature review, the state-of-the-art deep stereo techniques often use elaborate 3D convolutional networks for cost aggregation. Hence, it is important to investigate the impact of cost aggregation networks on the convergence of a metric which works at the cost volume stage, and vice-versa.

5. What gains could be achieved by using the automatic maximum disparity prediction with a deep learning network?

   A universal maximum disparity assignment would lead to a waste of computational power especially when the stereo image sequences have

a wide variety of disparity ranges. On the other hand, a layer-wise metric on a cost volume can result in performance degradation. Therefore, it is important to ask if maximum disparity estimation with a metric can lead to significant gains in performance in the long run. Next, it is also important to investigate for other potential gains associated with such an approach.

3.2 Methodology Overview

A five-stage research methodology is adopted to find answers to the five research questions. The individual steps (i.e., phases or stages) of the methodology include:

1. Development of a metric to provide termination criteria for a cost volume
2. Evaluate the feasibility of the metric in terms of computational complexity
3. Analyse the convergence of the metric using a deep stereo method
4. Develop an end-to-end stereo network and embed the metric
5. Training and evaluation using standard and custom stereo data

A simplified graphical illustration of the five phases can be found in Figure 3.1 and the individual phases are explained in the sections to follow.

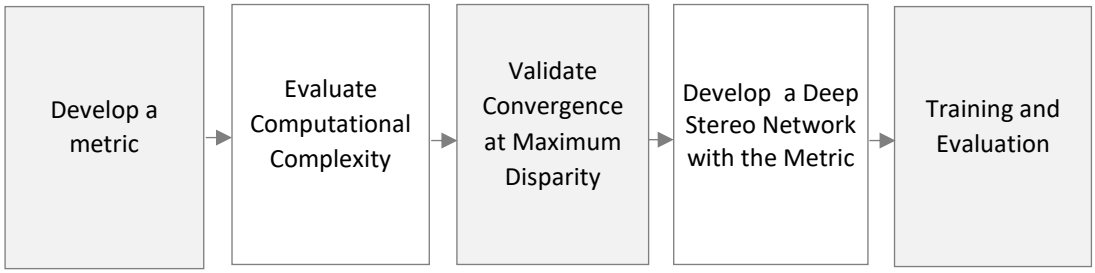


Figure 3.1: The five main steps of the methodology

3.2.1 Development of a Metric to Provide Termination Criteria for Cost Volume

The objective of this stage is to develop a metric which can be computed at each disparity to provide a universal termination criterion or criteria to end a layer-wise cost volume creation process, thereby determining the maximum disparity for any given scene. Initially, fundamental stereo matching techniques such as AD, SAD, SSD, RT, CT and NCC are used to study the changes in local matching cost extrema, in response to a layer-wise cost volume construction process. The intention here is to identify any phenomena associated with the matching cost extrema that coincide with the maximum disparity of a given stereo image pair. Standard stereo images from

Middlebury, KITTI, Scene Flow and custom-created stereo images with ground-truth disparity maps are used for the analysis. Based on the findings, a new metric (called *Sum of New Cost Extrema* - SNCE) is developed which can be estimated at each disparity to determine whether the cost volume construction should continue or terminate at a particular disparity. The mathematical foundation of the metric is then established to identify any conditions under which universal and user-independent convergence of the of the metric (i.e., the value of the SNCE metric meeting the termination criterion/criteria) can be achieved to reliably locate the maximum disparity. Subsequently, the developed metric is evaluated using synthetic and real datasets from stereo vision benchmark datasets (KITTI, Middlebury, Scene Flow) to validate whether the convergence of the SNCE metric coincides with the maximum disparity. Whenever necessary, custom stereo image pairs are synthesized and used to test the metric under challenging scenarios.

Definition: ***SNCE Convergence****: If the value of the SNCE metric computed at a particular disparity meets the termination criteria/criterion identified as part of the study, then it is referred to as "SNCE Convergence". The disparity value at which this happens, is a candidate for maximum disparity for the stereo image pair in concern.*

3.2.2 Evaluation of the Computational Complexity of the SNCE Metric

During this stage, the developed metric is further studied with both algorithmic analysis and empirical methods to estimate the additional computational complexity introduced by the metric on top of the already iterative computations associated with stereo disparity estimation. In order to achieve the same, the new metric is incorporated into basic stereo disparity estimation algorithms in both CPU and GPU environments. Initially, this is done with pseudo-code for algorithmic analysis and then the pseudo-code is implemented in Python and C++ for empirical analysis with real data to validate the theoretical findings. As part of the process, stereo images from the benchmark stereo datasets are used at four different resolution levels to investigate if the new metric affects the order of growth of computations associated with stereo algorithms. Furthermore, the overall impact of the metric on the

computational time of a stereo disparity estimation algorithm (on both CPU and GPU architectures) is analysed to determine the feasibility of the metric.

3.2.3 Analyse the Convergence of the Metric using a Deep Stereo Network

In the third phase of the overall methodology, a foundational deep stereo network is developed which can produce disparity maps along with a series of SNCE values computed at each disparity (i.e., at each layer along the disparity dimension of the cost volume). The developed network is trained using the Scene Flow "Driving" dataset using hyper parameters selected through experimentation with an optimizer. Next, the trained network is used to predict disparity maps for the validation/test stereo images while computing the SNCE metric at each disparity up to a value beyond the maximum disparity reported in the ground-truth data. The output disparity maps are then analysed qualitatively to ensure that the disparity predictions closely match the ground-truth data. The resulting SNCE variations are compared with the maximum disparity values extracted from the ground-truth data to verify if the convergence of the SNCE metric, occurs at the maximum disparity for a given scene. The same steps are repeated to further evaluate the metric using a series of stereo image sequences extracted from the validation data. Image cropping is used to improve the diversity of the maximum disparity values observed in test data. Finally, the disparity values that coincide with SNCE convergence are recorded and compared against the maximum disparity values extracted from the ground-truth data.

3.2.4 Develop an End-to-end Deep Stereo Method and Embed the Metric

In the fourth stage of the methodology, an end-to-end deep stereo network is developed based on the "*3D Regularization Structure*" in such a way that the network is able to:

- ✓ Build and allocate memory on-demand to the cost volume

  In a typical deep stereo network, a fixed-sized cost volume is constructed using the features extracted from the left and right stereo images by computing the matching costs or concatenating the features at each disparity starting from zero up to a user-specified maximum disparity. Such algorithms can afford to pre-allocate memory to hold the cost volume as the size is known in advance.

However, when using the SNCE metric to determine the maximum disparity automatically, the size of the cost volume is unknown in advance. Therefore, the network needs to be able to allocate memory on demand so that the layers of cost can be added to the cost volume until termination criteria is met. Also, due to the possibility of large maximum disparity values, the network needs to be memory efficient so that any over-estimation of the maximum disparity does not result in memory allocation errors.

- ✓ Compute the SNCE metric at every new layer added to the cost volume

  Each time a new layer is added to the cost volume the network should also be able to efficiently estimate the value of the SNCE metric for the newest layer.

- ✓ Terminate the process at any disparity based on the computed value of the metric

  If the estimated value of the SNCE metric meets the termination criteria, the network should be able to terminate the cost volume creation process and start the cost aggregation stage. However, the necessary precautions must be taken to meet the input dimensionality requirements of the aggregation layers which may have strict requirements in terms of the input tensor sizes to guarantee a fixed dimensionality at the output (e.g., Transposed Convolutional Layers). In such cases, termination of the cost volume creation process can be delayed until the additional criteria is met.

3.2.5 Training and Evaluation

The convergence of the SNCE metric at the maximum disparity cannot be guaranteed before training the deep stereo network. In addition, the training process needs to be conducted with a large dataset over several iterations (or epochs). Hence, the use of automatic maximum disparity during training can result in delays due to over-estimation of maximum disparity especially at the beginning when the network is partially trained. Such delays can lead to training times which are prohibitively longer and impractical. Also, it is not feasible to train a deep stereo network with all potential

maximum disparity values up to the stereo image width. Ideally the network should be able to learn with a small maximum disparity and then detect any suitable maximum disparity value during the disparity inference. Therefore, the SNCE-based automatic disparity estimation must be disabled during the training phase and a small fixed maximum disparity value must be used to achieve faster processing times so that the overall training time can be maintained within practical time limits. Only the pixels having disparities equal to or below the selected fixed maximum disparity must be used for backpropagation to update weights and biases in the network. Once the training phase is finished, automatic maximum disparity detection must be enabled for disparity inference which is the ultimate objective (i.e., develop a deep stereo algorithm which can produce stereo disparity maps independently of user-defined parameters).

3.2.5.1 Improving the Accuracy of the Maximum Disparity and Dense Disparity Estimations

Unlike the deep stereo networks which are optimized for the accuracy of the output disparity maps, the deep learning model developed at this stage requires to ensure that the accuracy of both the final disparity maps as well as the maximum disparity estimations improve simultaneously during training. Hence, the network loss calculation during training needs to include both the output disparity accuracy as well as some additional criteria which can help improve the accuracy of the maximum disparity estimations with SNCE. This is achieved by using a special training process called “Clamped Training”.

In clamped training, an additional disparity map is obtained by sending a copy of the cost volume through a “Softargmin” based disparity regression layer and an up-sampling layer. Then the intermediate and the final disparity maps are compared against the ground-truth to obtain disparity errors which are combined to estimate the overall loss.

3.2.5.2 Training and Evaluation Stages

The developed network is trained with clamped training in a 3-stage process, using three datasets to improve the accuracy of the maximum disparity predictions and final disparity maps while ensuring generalization capabilities of the network.

The 3 stages of training are:

1. Initial Training
2. Extended Training
3. Fine Tuning

NVIDIA RTX2080 and NVIDIA GTX1070 GPUs are used as the hardware for training. At the beginning of each training stage, automatic maximum disparity estimation with SNCE is disabled and the maximum disparity is fixed to a value of 256 pixels. Then only the pixels with ground-truth disparities equal to or lower than 256 pixels are used to train the network with clamped training. At the end of each training stage, SNCE-based automatic disparity estimation is enabled to conduct testing and evaluation.

Moreover, the intermediate regression and up-sampling layers (used to obtain intermediate disparity maps for clamped training) are disconnected from the network during performance evaluations. At the end of each training stage, an evaluation of the network is conducted using the validation data. Upon completion of all 3 training stages, a comprehensive evaluation is conducted using 3 additional benchmark datasets (ETH3D, Middlebury 2014 and KITTI 2012) and custom stereo images captured with a custom-built stereo camera.

3.2.5.2.1 Initial Training and Evaluation

The aim of the initial training phase is to ensure that the network is able to converge in terms of the accuracy of the maximum disparity estimations and the dense disparity maps. Initial training is conducted using 600 stereo image pairs (with ground-truth) from the Scene Flow “Driving” dataset. The data is segregated into training and validation datasets in an 80/20 split. At the end of the initial training phase, the validation data is used to analyse the disparity maps qualitatively and quantitatively. The qualitative analysis involves visual inspection of the predicted and ground-truth disparity maps for inconsistencies. The quantitative analysis of the output disparity maps is conducted using the following commonly used stereo disparity accuracy measurements:

1. Average Error (End Point Error)– *Average error refers to the absolute difference between the predicted and ground-truth disparity maps taken as an average over the whole pixel space.*
2. Three Pixel Error (3-Pixel Error) – *Three-pixel error refers to the percentage of pixels in the disparity prediction with associated error of more than 3 pixels compared to the ground-truth.*

During the initial training phase, additional tests are conducted using sequences of cropped stereo image pairs from the validation dataset to verify if the maximum disparity estimations using SNCE match the maximum disparity reported in the ground-truth. Image cropping is used to enhance the diversity of the maximum disparity values encountered during evaluation. A much smaller cropping size is used at this stage to introduce more variation in maximum disparity. Finally, a performance evaluation is conducted using 5 cropped stereo image sequences each having 60 stereo image pairs from the validation dataset. Firstly, the average processing time per stereo image pair is computed for the 5 image sequences while keeping the SNCE-based automatic maximum disparity estimation enabled. Then the same experiment is repeated with the cost volume size fixed to the largest maximum disparity value observed in the dataset. A comparative analysis is then conducted to check for gains or losses in performance.

3.2.5.2.2 Extended Training and Evaluation

The objective of the extended training stage is to train the novel stereo network with a large dataset to enhance the diversity of stereo data encountered during training. The Scene Flow "Flying Things 3D" dataset with 22,390 stereo image pairs (with ground-truth) is used to conduct extended clamped training. Data is separated into training and validation datasets with a ratio of 80/20. Instead of running through the whole image dataset at each epoch, 1000 randomly selected images from the training dataset are used for each training cycle. The overall training is conducted for 1000 such iterations. This is done to limit the GPU memory required to keep intermediate data (e.g., tensors) for debugging and analysis. At the end of extended training, the validation dataset is used to conduct a qualitative and quantitative analysis on the

disparity maps predicted by the trained network. The same stereo disparity accuracy measurements are used for the quantitative analysis.

3.2.5.2.3 Fine-Tuning and Evaluation

Ideally the network should be trained on a large dataset of real-world stereo images with ground-truth so that it is able to generalize across any calibrated stereo image pair. However, the same effect can be achieved by obtaining the network model saved after extended training and retraining it with a small real-world stereo dataset. For that purpose, KITTI 2015 dataset with 200 stereo images (with ground-truth) is used with a 90/10 split between the training and validation datasets. Upon completion of the fine-tuning stage, a qualitative and quantitative analysis is conducted using the validation data in a similar way to the previous stages. Upon completion of the fine-tuning stage, the network is ready for evaluation for its generalization capabilities on additional datasets and images captured with an experimental stereo camera setup.

3.2.5.3 Evaluation on Additional Datasets

Upon successful completion of the three training stages, the network is further evaluated using three additional datasets: ETH3D, Middlebury 2014 and KITTI 2012. The objective is to test the trained model on different types of stereo images (e.g., grayscale and colour) which have been captured with different cameras (i.e., different baselines, resolution levels etc.) and scenes with diverse structure in terms of the proximity of the objects. A qualitative analysis is then conducted using the predicted disparity maps and the available ground-truth data. It is followed up by a quantitative analysis using the average error and three-pixel error metrics.

3.2.5.4 Evaluation on Custom Stereo Images

Finally, the generalization capabilities of network are tested with stereo images captured using a custom-built stereo camera system. The objective here is to analyse the network's response to imperfect stereo image pairs captured with an experimental stereo camera. The setup consists of a pair of individual FLIR Chameleon3 cameras with fisheye lenses mounted on a custom-built stereo rig as shown in Figure 3.2.

Prior to capturing images for evaluation, stereo calibration is performed using a series of images of a checkerboard pattern captured with each of the cameras. Upon

completion, calibration parameters are stored for later use when performing fisheye distortion correction and stereo calibration. Subsequently, the system is used to obtain calibrated stereo images of structured indoor scenes as well as sequences of outdoor scenes. The foreground objects are manipulated, or new objects are introduced into the foreground of the scenes to vary the maximum disparity. The network is then used to predict disparity maps for static scenes as well as sequences of images before analysing the disparity maps qualitatively.

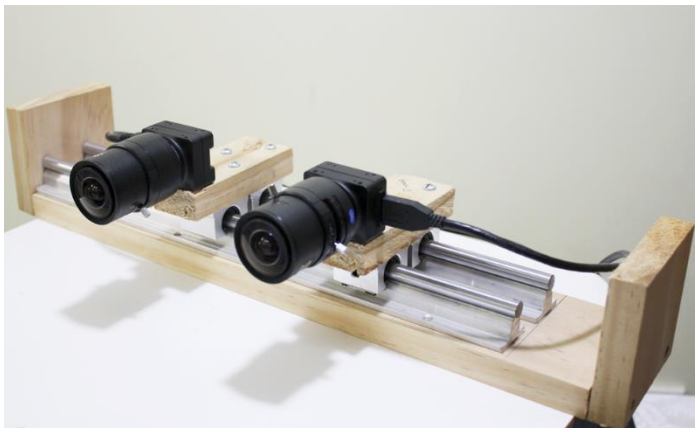


Figure 3.2: Custom built stereo camera setup to obtain calibrated stereo image pairs for evaluating generalization capabilities of the novel deep stereo network. System consists of two FLIR Chameleon CM3-U3-13S2C-CS cameras fitted with Fisheye lenses having manually variable focal length.

3.3 Chapter Summary

This chapter introduced a 5-stage methodology to achieve the objectives outlined in Chapter 1. In the first stage of the methodology, experiments will be conducted using basic stereo algorithms and standard stereo datasets to study the movement of local matching cost extrema during a layer-wise cost volume creation process. The objective is to find any phenomenon that coincide with the maximum disparity. Based on the findings a new metric called Sum of New Cost Extrema (SNCE) will be developed. A mathematical model for the metric will establish the termination criteria for a layer-wise cost volume creation process based on the value of the metric. Standard stereo benchmark datasets with ground-truth will be used to evaluate the metric for its convergence.

During the stage 2, an algorithmic analysis will be conducted and followed up with an empirical analysis to determine the additional computational complexity introduced

by the metric. Pseudo code of the basic stereo algorithms is then modified to include the metric and implemented in code for analysis using stereo images of varying resolution. The order of growth of computations associated with the modified algorithms (with and without the metric) is then determined. Experiments in both CPU and GPU environments will be completed to study the feasibility of the metric for use with deep stereo algorithms.

In the 3rd stage of the process, a basic deep stereo algorithm is developed in such a way that it can compute the new metric at every layer during the forward propagation through the network. The developed network model is then trained and evaluated with cropped stereo images and image sequences from the Scene Flow "Driving" dataset to validate SNCE convergence at maximum disparity. Although the SNCE metric is computed at each disparity, the cost volume size is kept fixed during this stage so that the value of the metric can be analysed at the maximum ground-truth disparity and above.

A fully-fledged deep stereo network is developed during the 4th stage of the methodology. This involves developing capabilities such as progressive memory allocation on GPU for the cost volume and termination or scheduled termination of the cost volume creation process based on the value of the metric.

During the 5th and final stage, the "clamped training" process will be introduced which is aimed at ensuring both SNCE convergence and output disparity accuracy improve simultaneously during the training process. A 3-stage training process is then used to train the end-to-end network with the objective of improving its generalization capabilities. The three training stages include initial training, extended training and fine-tuning. Scene Flow "Driving", Scene Flow "Flying Things 3D" and KITTI 2015 datasets are used during the training stages, respectively. At the end of each training process, the resulting network is evaluated using the validation data both qualitatively and quantitatively. Upon completion of the training process, the final network model is evaluated using stereo images from 3 additional stereo datasets (ETH3D, Middlebury 2014 and KITTI 2012) and stereo images captured using an experimental stereo camera system.

CHAPTER 04 - AUTOMATIC ESTIMATION OF DSR

Introduction

The literature review in Chapter 2 revealed that in many stereo algorithms, disparity search range (DSR) remains a user-configured parameter that relies on user's intuition and understanding of the scene structure. Furthermore, it was observed that even the modern state-of-the-art deep stereo techniques accept the ceiling value of DSR (i.e., the "Maximum Disparity") in the form of user input. This chapter aims to develop a metric which determines the termination criteria for a layer-wise cost volume creation process, thereby effectively determining the "Maximum Disparity" without user intervention.

The study in this chapter starts with an in-depth analysis of the matching cost extrema movements within a partially built cost volume when new layers of costs are added to the same. A metric called Sum of New Cost Extrema (SNCE) is then developed to capture the information about the movement of the local cost extrema. The novel metric is tested with foundational stereo vision techniques by using standard stereo datasets and custom-created synthetic image data. A mathematical model is developed to explain the behaviour of the metric and to study the reliability, certainties and uncertainties associated with the maximum disparity predictions made by using the metric. Subsequently, the unimodality of the matching costs is discussed which can lead to completely deterministic results when using the SNCE metric.

4.1 Local Cost Extrema Movements in a Partially Built Cost Volume

The size of a typical 3-dimensional cost volume used in most traditional stereo algorithms, is determined by the image width, height and the maximum disparity chosen by the user. For every pixel in the reference image of a stereo image pair, the corresponding 3D cost volume has an array of matching costs, having a length equal to the maximum disparity. Hence, the minimum or the maximum matching cost (i.e., matching cost extrema) for a given pixel, can be located anywhere within the disparity search range from 0 [2] to the maximum disparity. In algorithms which assign disparities

[2] Usually, a disparity value of zero is considered invalid as it indicates an infinite distance according to Equation (1.1). However, when matching finite pixels between two images, the possibility of a match being found at the same location on the other image could not be ruled out. Therefore, in this study the minimum possible disparity for a scene is taken as zero.

based on a Winner-Take-All (WTA) approach, the location of the cost extremum is chosen as the best-match disparity for a given pixel. For example, **Figure 4.1** shows the Sum-of-Absolute-Difference (SAD) based matching cost distributions for 5 pixels (P1 to P5) selected from the popular Middlebury "Cones" stereo image pair. The vertical green markers in the figure indicate the positions of the most stable cost extrema within the disparity search range from 0 to 60 pixels. The individual locations along the disparity dimension are mentioned inside the square labels of the markers.

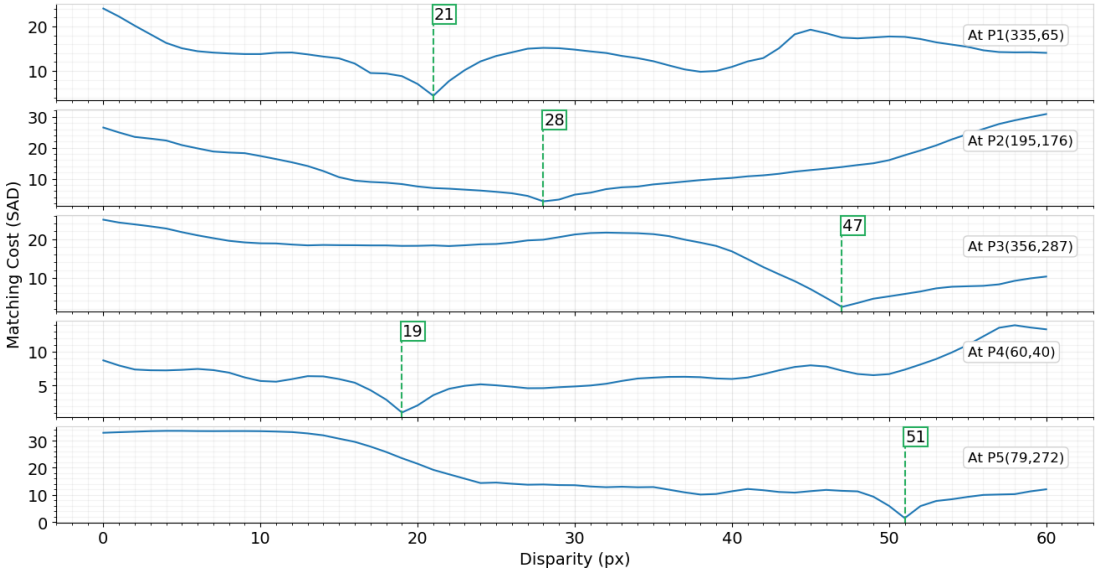


Figure 4.1: SAD-based matching cost distributions (computed over a 11x11px mask) for 5 random pixel locations in the Middlebury "Cones" 2003 stereo image pair at a resolution of 450x375 and a maximum disparity of 60 pixels. The green markers indicate the locations of the global cost extrema over the disparity search range of 0 – 60.

According to the five subplots provided in **Figure 4.1**, the cost distributions have distinct extrema established within the disparity search range (DSR) which spans from 0 to 60 pixels (here 60 is a user-defined value). It is important to note that the use of SAD for cost aggregation is arbitrary. If Normalized Cross Correlation (NCC) is used with the same set of points, distinct maxima (instead of minima) would be established at the same disparities as shown in **Figure 4.2**.

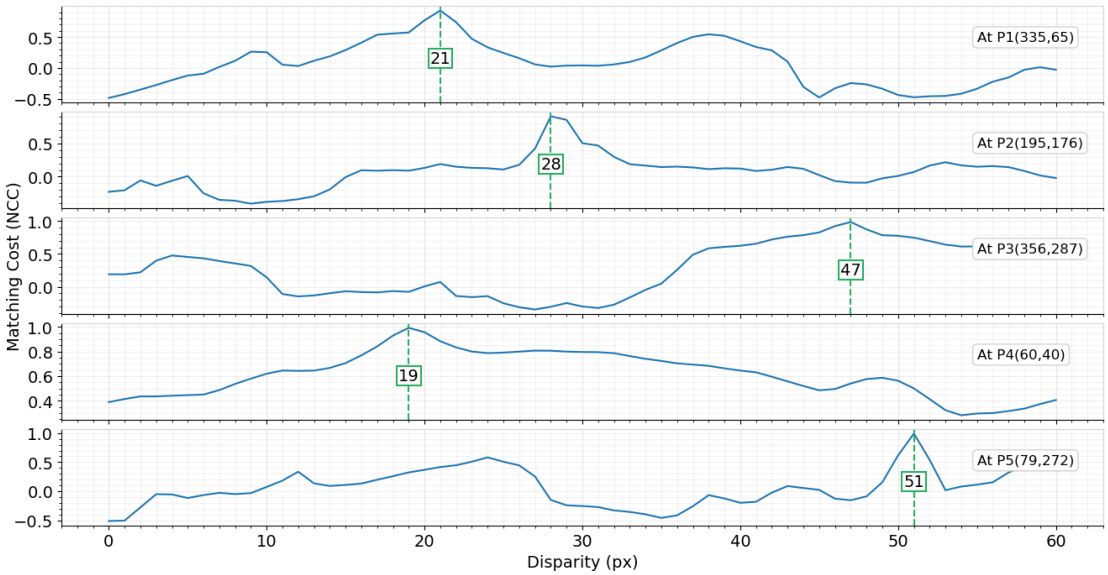


Figure 4.2: NCC based matching cost (aggregated over a 11x11px mask) distributions for the same 5 random pixel locations (w.r.t left image) from the "Cones" stereo image pair - Middlebury 2003 benchmark. The green coloured markers indicate the locations of the NCC-based matching cost extrema.

In a typical stereo vision scenario, the cost extrema in **Figure 4.1** and **Figure 4.2** correspond to the best match disparities for the five pixels (i.e., assuming disparity assignment with a WTA approach). For the same set of points as earlier, if the user sets the maximum disparity to 30 pixels, then the locations of some matching cost extrema would change as shown in **Figure 4.3**.

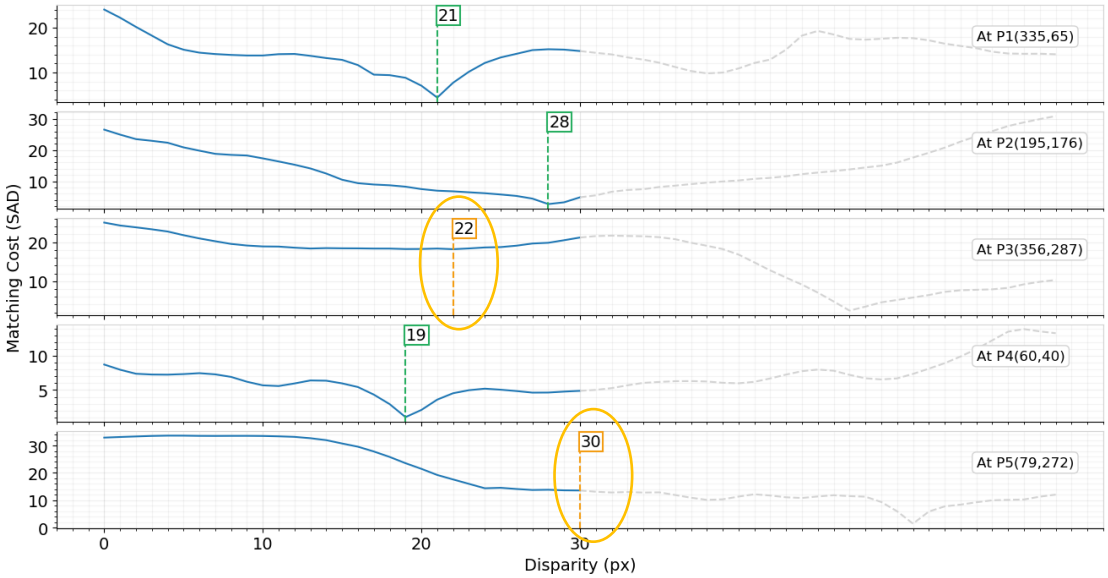


Figure 4.3: SAD-based matching cost extrema (i.e., green markers) for the same set of points (P1-P5) with a user-defined maximum disparity of only 30 pixels (i.e., smaller DSR of 0-30 instead of 0-60 used earlier). Cost minima locations of P3 and P5 are now different compared to the locations observed earlier when the maximum disparity was set to 60 pixels.

In fact, **Figure 4.3** can be thought of as the matching cost distribution for five points inside a partially built cost volume which has only 30 layers along the disparity dimension. **Figure 4.4** shows a partially built cost volume with the local extrema for the five points from P1 to P5 established in the range of disparity values from 0 to 30. As more layers are being added, these positions may or may not change depending on the matching cost distribution at each pixel position.

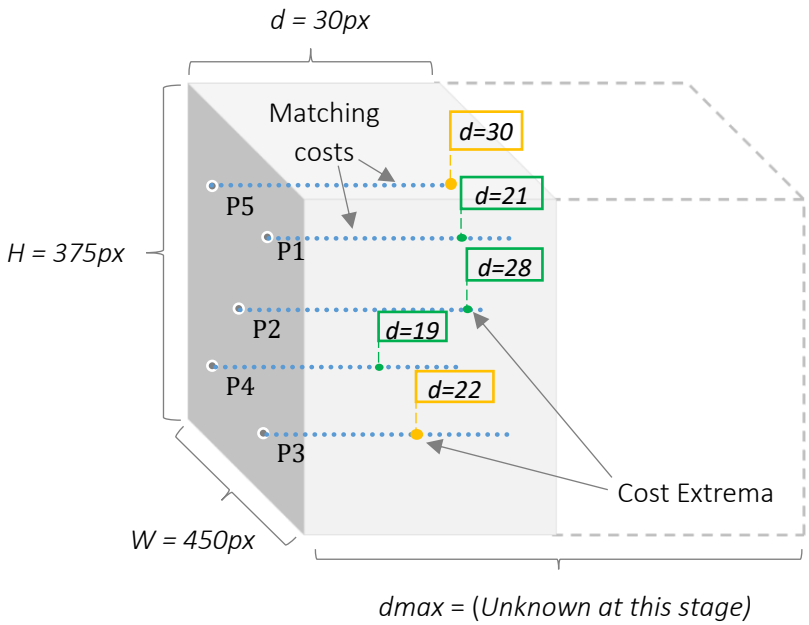


Figure 4.4: A graphical illustration of the matching cost distributions of points P1 to P5 inside a partially built cost volume which has 30 layers along the disparity dimension. Stable and unstable cost extrema are marked with green and orange markers. H, W and dmax stand for the image width, height and the best-match maximum disparity, which is unknown at the given partially built stage, respectively.

As a further illustration, **Figure 4.5** shows what happens to the cost extrema associated with the point P3 located at [356,287] when the maximum disparity is changed from 10 to 60 in increments of 10 while keeping the minimum disparity equal to zero.

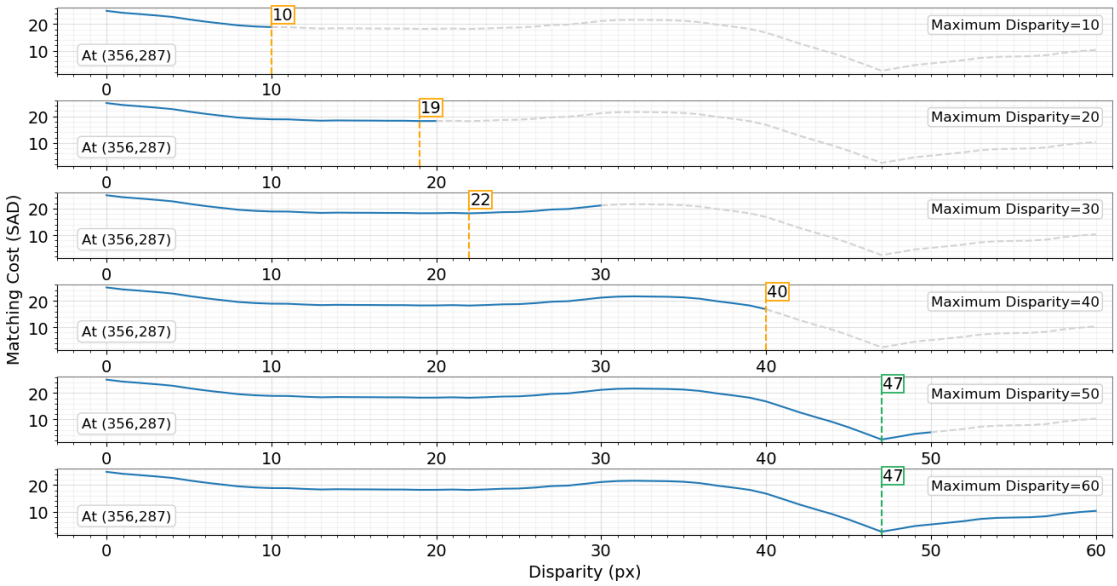


Figure 4.5: Variation of SAD-based cost minima for the point P3 when the maximum disparity is changed from 10 to 60 in increments of 10 pixels at a time while keeping the minimum disparity at 0. The orange-coloured markers indicate where the cost minima were located temporarily before settling at the most stable location marked with green markers.

In the scenario depicted in **Figure 4.5**, changing the maximum disparity from 10 to 60 in increments of 10 pixels, has resulted in the cost extrema shifting to relatively unstable locations at disparity 10, 19, 22 and 40 (marked with orange markers), before finally settling at 47 (marked with green markers) which is the most stable cost extrema location for point P3. This provides a glimpse of how the extrema associated with image points keep shifting (or moving) to new locations when the disparity search range is expanded by choosing progressively larger values for maximum disparity.

<u>Convention:</u> This chapter adopts the convention of using orange-coloured markers to indicate unstable matching cost extrema locations and green coloured markers to point to the most stable cost extrema for a given matching cost distribution.

4.2 Building the Cost Volume Layer-by-Layer

It is possible to simulate layer-wise construction of a cost volume by setting the step size to 1 (instead of 10 used earlier) and then progressively increasing the ceiling value of DSR, one step at a time. **Figure 4.6** shows what happens to the matching cost extrema of the same 5 random points (P1 to P5) on the Middlebury "Cones" image pair, when the maximum disparity is changed in steps of 1 pixel from 0 to 60. In accordance with the adopted convention, the orange-coloured markers indicate the

locations of the relatively unstable cost extrema while the green markers show the final stable cost extrema for each matching cost distribution.

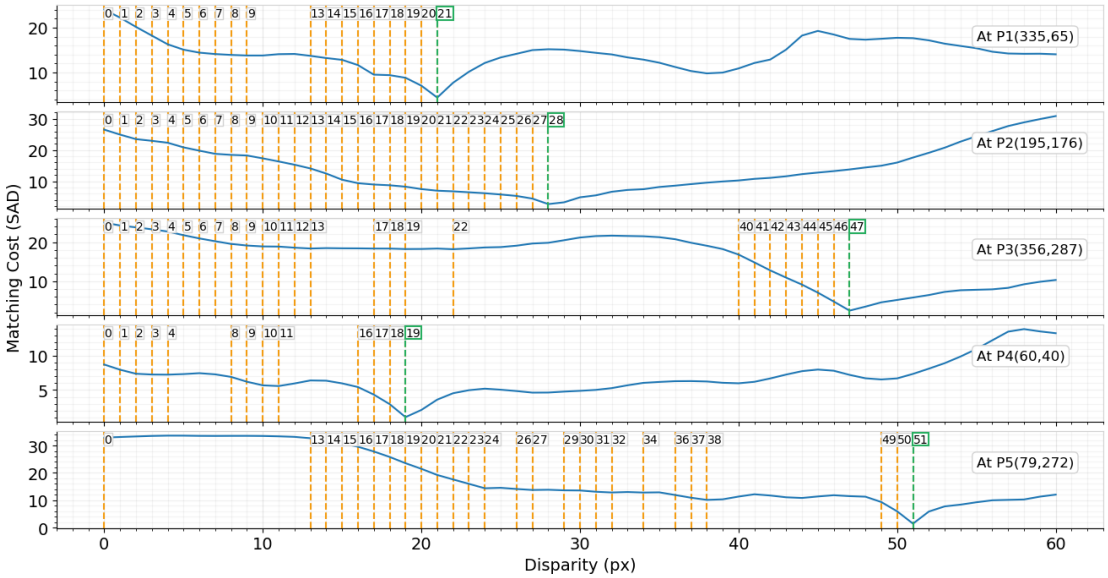


Figure 4.6: Changes in matching cost extrema locations for the same 5 points (P1-P5) from the Middlebury Cones image pair when the maximum disparity is changed from 0 to 60 in steps of 1 pixel at a time which simulates a cost volume that is being built one layer at a time up to 60 layers. The orange markers indicate the locations of the previous unstable cost extrema and the green markers indicate the locations of the final stable matching cost extrema locations for each of the 5 points.

As it can be seen from **Figure 4.6**, when the maximum disparity is set to 0 (which represents a cost volume with only one layer), the cost extrema for all points have remained in the first and only layer of the cost volume. Hence, there are orange-coloured markers at disparity 0 on each of the five subplots. Upon setting the maximum disparity to 1, the matching cost extrema of four points (P1, P2, P3 and P4) have moved to the 2nd layer of the cost volume. As a result, there are four vertical orange markers at disparity 1 in subplots of P1, P2, P3 and P4. Similarly, when the maximum disparity is set to 2, the cost extrema of the first 4 points (P1 to P4) have again moved to the 3rd layer; hence another 4 orange markers at disparity 2 on plots from P1 to P4. This appears to continue at most disparities until the most stable extrema locations are reached. The final stable minima locations are marked with green markers and no further shifts in cost extrema could be observed after reaching the most stable minima locations.

Figure 4.7 shows the total number of cost extrema shifts or movements at each disparity across all five points from P1 to P5 during the whole process (i.e., when increasing the maximum disparity from 0 to 60 in steps of 1 pixel). In other words, Figure 4.7 plots the total number of dashed markers at each disparity location across all five subplots in Figure 4.6. For example, there are 5 markers at disparity zero in Figure 4.6. Therefore, Figure 4.7 has a value of 5 at disparity zero. Similarly, at disparity 1, Figure 4.6 has 4 markers in the subplots from P1 to P4. Hence, Figure 4.7 has a value of 4 at disparity 1. The dashed green line at disparity 51 in Figure 4.7, indicates the largest value among the disparities associated with the five points (based on a WTA disparity assignment).

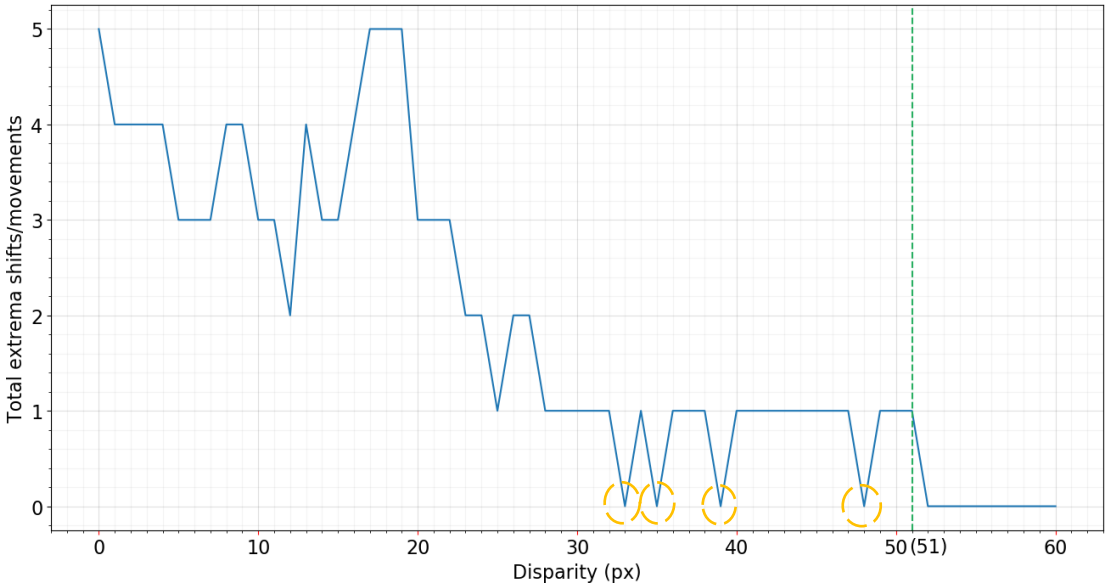


Figure 4.7: A graph showing the total number of matching cost extrema location changes (associated with the 5 random pixels P1-P5) at each disparity when the cost volume is constructed one layer at a time up to 60 layers along the disparity dimension. The largest disparity for the five pixels (selected based on WTA at the lowest cost) is marked with a dashed green line. Orange coloured circles mark the disparities (up to the largest disparity marked in green) with zero extrema movements/shifts.

Figure 4.7 indicates that the positions of the cost extrema associated with the five points (P1-P5) have changed many times at most of the disparities from 0 to 51 during the simulated layer-wise cost volume construction process. In fact, the cost extrema movements associated with just 5 pixels, have taken place at most of the disparities up to the largest disparity among the points P1 to P5 (i.e., 51) except at a few disparities marked with orange-coloured circles in Figure 4.7. This chapter aims to study the behaviour of the extrema movements shown in the figure, to identify the conditions under which the phenomenon can be used to determine a suitable

maximum disparity value automatically. The metric is expected to identify the occurrence of the last stable matching cost extremum/extrema for a given stereo image pair during a layer-wise cost volume creation process.

4.3 Sum of New Cost Extrema

The total sum of extrema movements plotted in Figure 4.7, is defined as a metric called "Sum of New Cost Extrema" or SNCE for further analysis in this chapter. Therefore, the following definition is used throughout this chapter and the rest of the thesis.

Definition: ***Sum of New Cost Extrema****: In a partially built matching cost volume, the total number of pixels with their matching cost extrema located at the current layer (layer added last) of the cost volume is called the "Sum of New Cost Extrema" or SNCE. The SNCE value can also be computed in the form of "Sum of New Cost Minima" or "Sum of New Cost Maxima" (SNCM) based on the cost aggregation method used. Nevertheless, SNCM and SNCE refers to the same metric and the terms can be used interchangeably.*

4.4 A Mathematical Expression for SNCE

If each stereo image contains (N) number of pixels and the matching cost variation for the ith pixel is given by C_i, then the minima based SNCE value for the dth disparity of a partially built cost volume can be expressed using the following:

$$SNCE(d) = \sum_{i=1}^{N} [argmin(C_i) == d] \tag{4.1}$$

$$\text{where } [I] = \begin{cases} 1 & if\ I\ is\ True \\ 0 & otherwise \end{cases}$$

The value of the "Iverson Bracket" or function $[I]$ in Equation (4.1) is equal to 1 when the cost minima for the given pixel is located at disparity d of a partially built cost volume. Otherwise, the value will be equal to 0. If the cost maxima are used instead of cost minima, the SNCE metric can also be expressed as shown in Equation (4.2).

$$SNCE(d) = \sum_{i=1}^{N} [argmax(C_i) == d] \tag{4.2}$$

4.5 SNCE on Standard Stereo Datasets

The Middlebury 2003 "Cones" stereo image pair includes 168,750 pixels at quarter resolution (i.e., 375x450). According to the ground-truth data, the largest integer disparity associated with the pixels in the stereo area[3] of the "Cones" image pair is equal to 54 pixels. Tests with 5 pixels (P1 to P5) earlier, showed that the extrema movements occur at most disparities up to the largest WTA disparity associated with the pixels in concern. **Figure 4.8** shows the variation of SNCE across all pixels against disparity when a SAD-based cost volume is constructed one layer at a time from 0 to 60 layers using a mask size of 11x11 pixels.

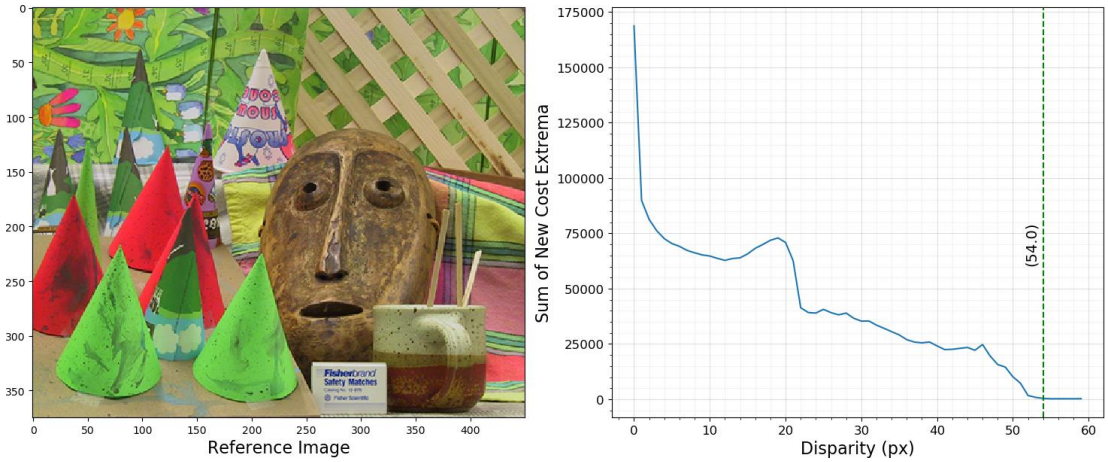


Figure 4.8: A plot showing the Sum of New Cost Extrema (SNCE) variation associated with all 168,750 pixels in the Middlebury 2003 Cones stereo image pair when the cost volume is constructed up to 60 layers (one layer at a time). The SNCE value appears to stay high until the maximum ground-truth disparity is reached. The vertical dashed-green line shows the largest ground-truth disparity across all pixels (which is 54 pixels in the "stereo area" of the "Cones" image pair).

According to **Figure 4.8**, when there was only one layer in the cost volume, the cost extrema associated with all 168,750 pixels have remained in the first layer (at disparity 0). Then with the addition of new layers to the cost volume, some of the cost extrema have shifted to the newly added layers. For example, approximately 25,000 extrema movements have occurred at a disparity of 40 pixels. In other words, when the 40th layer was added to the cost volume, the cost minima of around 25,000 pixels have moved to the 40th layer.

It is also important to note that the SNCE value in **Figure 4.8** appears to have become smaller around the maximum ground-truth disparity among all pixels which is 54

[3] Stereo area is the area of overlap between the images of a stereo pair. For pixels located outside the stereo area of one image, no matches can be found within the image boundaries of the other.

pixels. This is more visible from **Figure 4.9** which shows an enlarged view of the SNCE variation closer to the disparity value of 54. The diagram also shows that, at a disparity of 50 pixels, there have been around 10,000 extrema shifts. Then from disparity 50 to 54, the value has become significantly smaller particularly closer to the maximum ground-truth disparity marked with the vertical green line.

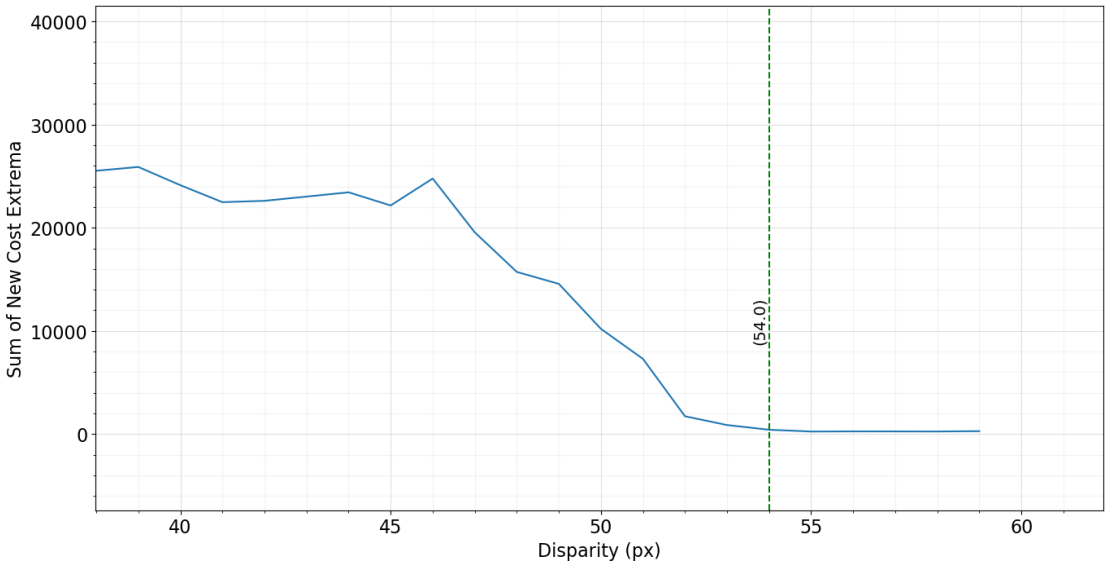
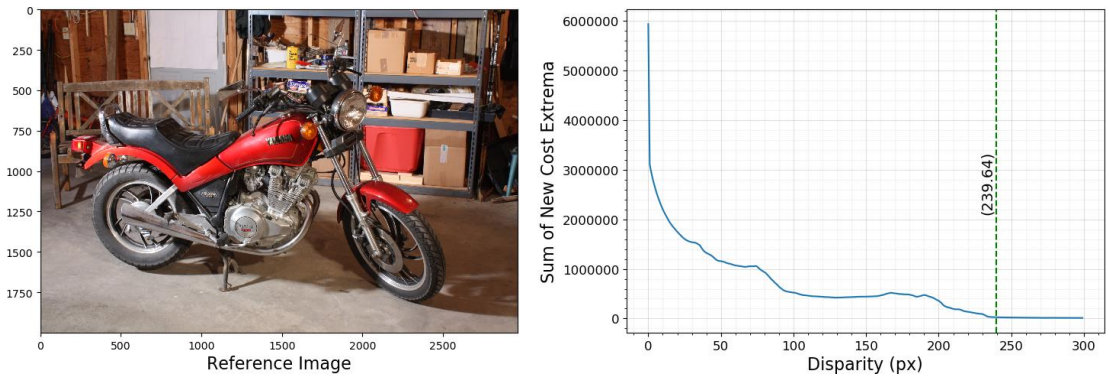


Figure 4.9: Enlarged view of the SNCE variation closer to the disparity value of 54 pixels for the Middlebury 2003 Cones stereo image pair, when the cost volume is created one layer at a time up to 60 layers. SAD has been used with a mask size of 11x11 to produce the matching costs.

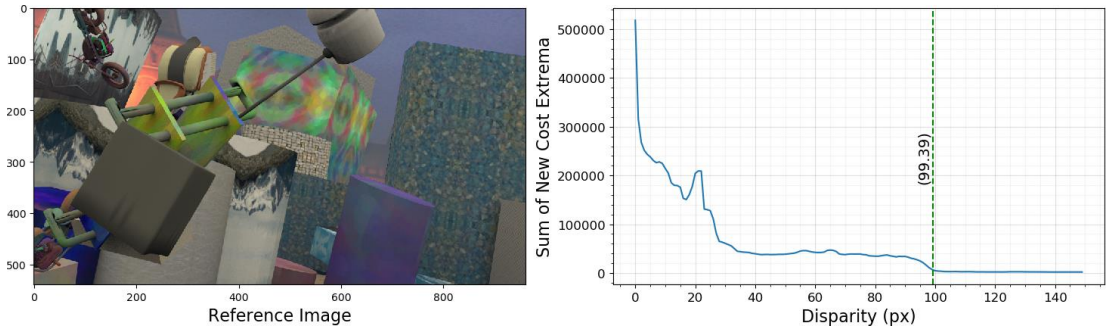
4.5.1 SNCE on Other Datasets

The series of subplots from (a) to (d) in **Figure 4.10** show the variation of SNCE for 4 sample stereo image pairs from the Middlebury, Scene Flow and KITTI benchmark datasets. The maximum ground-truth disparity associated with the 4 scenes are different from each other with values spanning from 76 to 385.26 according to the ground-truth data. However, the value of SNCE appears to reduce to some smaller value (or values) around the largest ground-truth disparity for each scene marked with dashed-vertical-green lines. When considering the scale of the vertical axis, the graphs also indicate that there are thousands of pixels with their cost extrema switching positions at each disparity up to the disparity values marked in green.

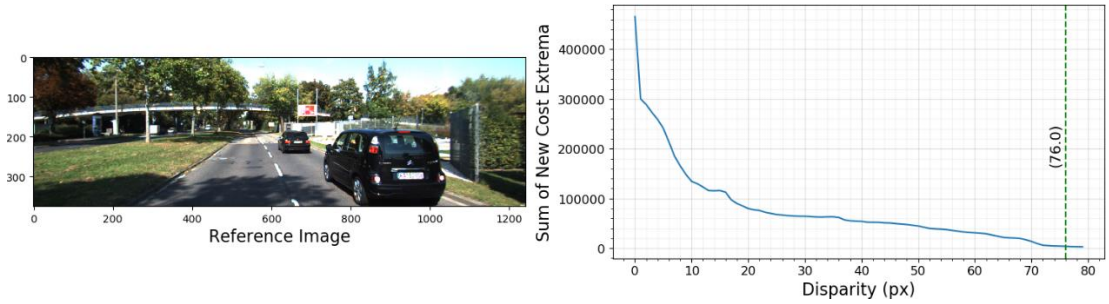
Important: At this point a clear correlation between the maximum ground-truth disparity and the small SNCE values cannot be established. The next section aims to develop a mathematical model which can be used to first explain the results shown in **Figure 4.10** including the dependency on the scene structure.



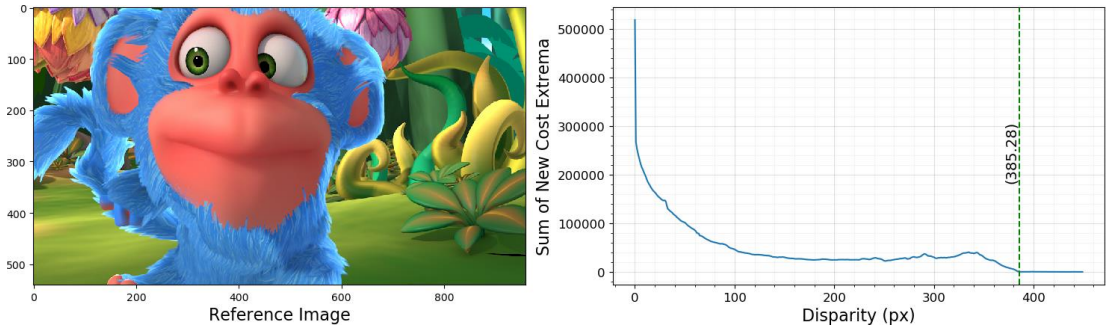
(a) Middlebury 2014 – "Bike"



(b) A Sample from Scene Flow "Flying Things 3D" Data



(c) A Sample from "KITTI 2015 Stereo" Data



(d) A Sample from "KITTI 2015 Stereo" Data

Figure 4.10: SNCE Variations across all image pixels for sample stereo image pairs from (a) Middlebury 2014, (b) Scene Flow Flying Things 3D, (c) Scene Flow Driving and (d) KITTI 2015 stereo datasets. The SNCE value appears to reach a lower value around the maximum ground-truth disparity reported in the stereo area of the scenes (shown with dashed-green markers with their values specified within the brackets).

4.6 Mathematical Representation

Consider the following depiction (**Figure 4.11**) of a partially built cost volume having the dimensions of *W x H x* d_{max} (where *W* = width, *H* = height and d_{max} = the maximum expected disparity for the scene which is yet to be found). At its current state, the cost volume has only *d* number of layers along the disparity dimension. The point *O (x, y, 0)* is the matching cost at zero-disparity for a random pixel at ***(x, y)*** on the reference image. The point *D (x, y, d)* is the matching cost for the same pixel at a disparity *d*. The point *D (x, y, d)* is located on the d^{th} layer of the cost volume.

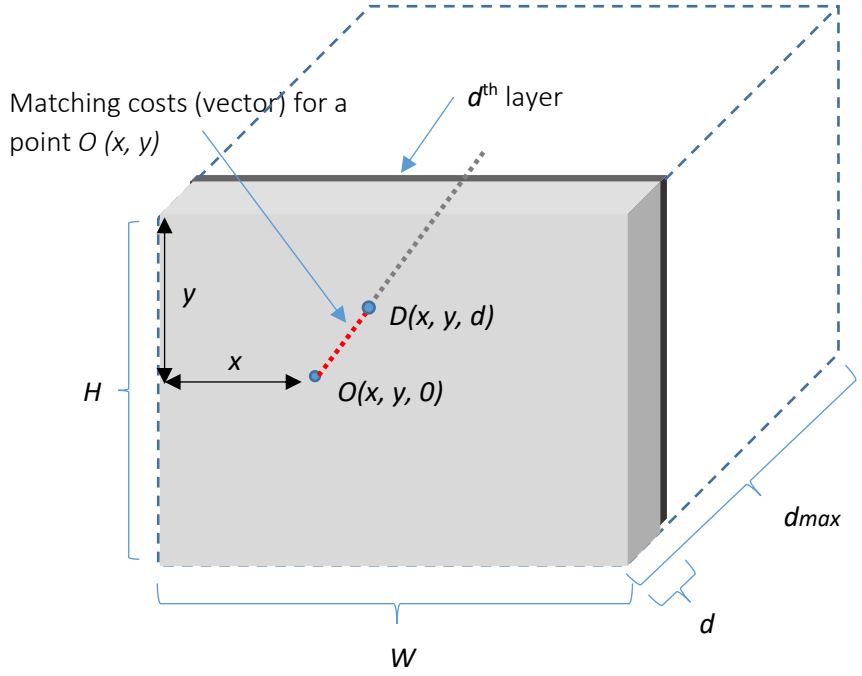


Figure 4.11: A partially built cost volume (with the d-th layer being the latest)

Now, the local cost extrema for the matching costs associated with a pixel at *(x, y)* on the reference image, must reside within the series of values from *O* to *D* (referred to as *OD* hereafter). If the disparity values along the *OD* direction have an equal likelihood of being the local cost extrema location on *OD* (exceptions to this assumption will be addressed in ***Section 4.6.1*** on scene structure), then the probability of the matching cost extrema along *OD* being located at d^{th} layer is given by **Equation (4.3)** below.

$$P(Extrema\ on\ OD\ located\ at\ D) = \frac{1}{d} \tag{4.3}$$

Conversely, the probability of not having cost extrema located at the same layer is given by:

$$P(No\ Extrema\ at\ D) = 1 - P(Extrema\ on\ OD\ located\ at\ D)$$
$$= 1 - \frac{1}{d}$$
$$= \frac{d-1}{d} \quad (4.4)$$

When the whole image is concerned, there are (WxH) number of points on any layer of the cost volume. Therefore, the probability of not having any extrema located on the d^{th} layer can be calculated as:

$$P(No\ Extrema\ at\ d'th\ layer) = \left(\frac{d-1}{d}\right)^{(WH)} \quad (4.5)$$

Furthermore, the probability of having at least one extremum located at the d^{th} layer can be defined as:

$$P(At\ least\ one\ extremum\ at\ d'th\ layer) = 1 - \left(\frac{d-1}{d}\right)^{(WH)} \quad (4.6)$$

As per the conditions for the above equation to stay valid, the disparity *d* should be less than the maximum expected disparity $dmax$ for the scene. Therefore, the complete equation for all disparities including values below and higher than the $dmax$ can be written as:

$$P\begin{pmatrix} At\ least\ one \\ extremum\ at \\ d^{th}\ layer \end{pmatrix} = \begin{cases} 1 - \left(\frac{d-1}{d}\right)^{WH}, & 0 < d \leq dmax \\ 0\ ^{4}, & d > dmax \end{cases} \quad (4.7)$$

From the equation, it can be deduced that the probability of having at least one extremum at the newest layer of a partially built cost volume, depends on the total number of pixels in the image as well as the current disparity (when the current disparity is lower than the maximum expected disparity for the scene). Specifically, a lower disparity value *d* makes the probability higher whereas a higher number of pixels also leads to a higher probability.

[4] Assuming stronger stereo matching with reliable disparity assignment capabilities.

4.6.1 Effect of the Scene Structure

Depending on the scene structure it may not be possible to assume that the cost extrema are randomly distributed in the range from 0 to some disparity *d* for all pixel locations. For example, some of the cost extrema may have already reached their most stable positions by disparity *d*. In such instances, only the remaining unstable extrema are likely to move to the newest layer of the cost volume. If $N(d)$ represents the number of unstable cost extrema at disparity *d*, then the **Equation (4.7)** can be modified as given below:

$$P\begin{pmatrix} At\ least\ one \\ extremum\ at \\ d^{th}\ layer \end{pmatrix} = \begin{cases} 1 - \left(\frac{d-1}{d}\right)^{N(d)}, & 0 < d \leq dmax \\ 0, & d > dmax \end{cases} \tag{4.8}$$

According to **Equation (4.8)**, if the value of *N(d)* is large enough and the *dmax* for the scene has not yet been reached, then there is still a higher probability of locating at least one cost extrema at the newest layer of the cost volume. Since disparity is inversely proportional to the depth, the value of *N(d)* can be perceived as an estimate of the number of pixels associated with the foreground of the scene with respect to the depth at disparity *d*. This can be better understood by plotting the SNCE variation with the ground-truth frequency distribution as a histogram as show in **Figure 4.12**.

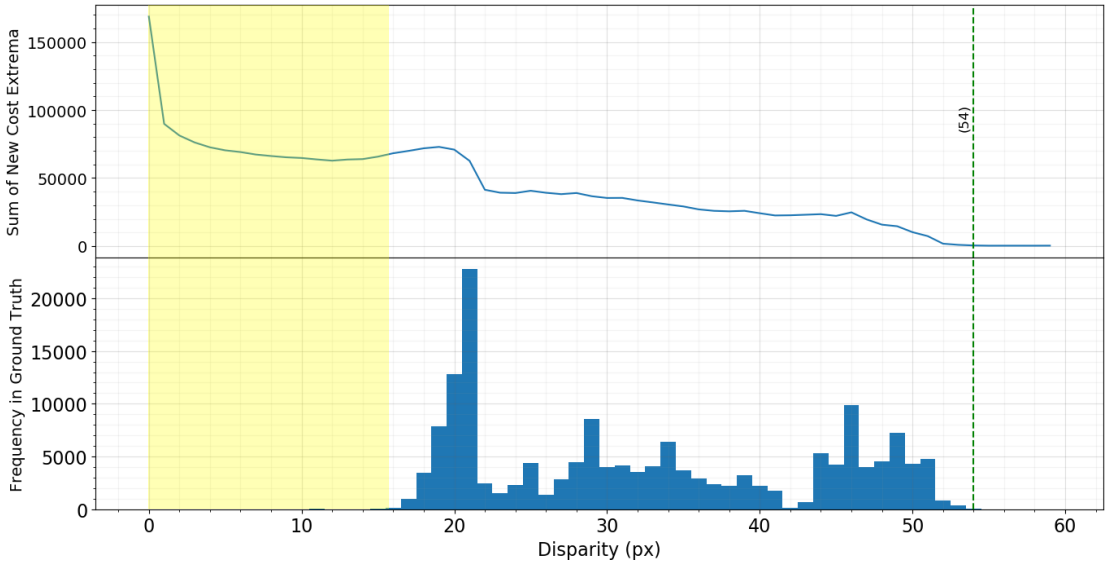


Figure 4.12: SNCE Variation and the ground-truth disparity frequency distribution for the Middlebury 2003 "Cones" image pair at quarter resolution. SAD was used for cost aggregation over a 11x11 mask when computing SNCE for disparities from 0 to 60. The ground-truth disparity frequency is low in the disparity range shaded in yellow (0-16). However, the SNCE value has remained high due to the large number of pixels which are yet to reach their stable extrema by disparity 16.

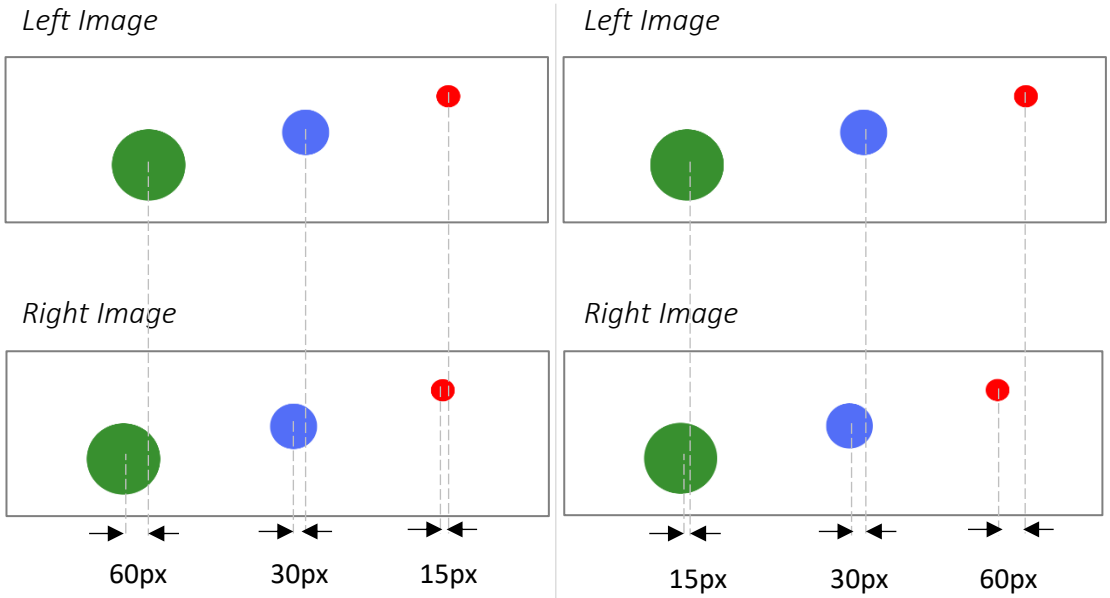
Figure 4.12 shows both the ground-truth frequency distribution and the SNCE variation for the Middlebury 2003 "Cones" image pair in the same diagram. According to the histogram shown at the bottom of the figure, there are only a very few pixels having any ground-truth disparities in the range from 0 to 16 (area shaded in yellow). Therefore, at disparities below 16, most of the pixels in the image correspond to the foreground with respect to a depth that corresponds to disparity 16. This leads to a higher probability of locating cost extrema of such pixels in the newest layer. As a result, the SNCE values in the disparity range from 0 to 16 have remained comparatively high. It is also noteworthy, how the SNCE value declines steeply at disparities (e.g., disparity 21 and 46) which correspond to the larger ground-truth disparity frequencies. It is a result of many cost extrema settling at their respective stable positions at 21 and 46.

4.6.2 Empirical Analysis – SNCE vs. Foreground Objects

In order to verify the effect of the foreground objects, two synthetic stereo scenes were created which include three planar objects with each having different sizes and associated disparities between the corresponding pixels. All images have a white texture-less background. Figure 4.13 shows the synthetic stereo images with dashed lines and arrows showing the different disparities assigned to the red, blue and green objects. The image pair on the left (Figure 4.13 (a)) has its largest object (green surface) as the foreground object whereas the stereo image pair on the right (Figure 4.13 (b)) has its smallest object (red surface) in the foreground.

Next, two SAD-based cost volumes were created using the synthetic image pairs, in a layer-wise manner while aggregating the costs with a mask size of 11x11 pixels. At every step, the number of pixels with their cost extrema located at the newest layer, was recorded as the SNCE value. The SNCE values at each disparity from the two scenes, were then plotted on the same graph (Figure 4.14) for comparison. In the plots, the large SNCE values at disparity 0 have been clipped to make the rest of the SNCE variations more visible.

(Please refer to *Appendix B* to locate the synthetic images, source code and a software executable used to produce the results)



(a) Scene 1: Large Object in the Foreground *(b) Scene 2 – Small Object in the Foreground*

Figure 4.13: Synthetic stereo image pairs depicting 3 mutually displaced planar objects with different disparities. The pair of images on the left (a) has the largest object in the foreground and the image pair on the right (b) has the smallest object in the foreground (simulated using the disparities shown with the dashed lines and arrows which are not part of the images). All images have a size of 11242x375.

As seen from the results in **Figure 4.14**, the SNCE values of "Scene 1" have been consistently higher than the values from "Scene 2" due to the larger foreground object in "Scene 1". This stems from the fact that there are many pixels associated with the green object (compared to the red object in "Scene 2") which are unlikely to reach stable cost extrema before the corresponding disparity of 60 pixels which also happens to be the largest disparity for the scenes. A larger foreground object leads to a higher $N(d)$ in **Equation (4.8)** at disparities below 60 which means that there is a higher probability of having at least one pixel with their cost extrema moving to the newest layer of a cost volume.

In addition, the SNCE values also appear to increase gradually during the ground-truth disparity ranges such as 1 – 15, 16 – 30 and 31 – 60. Moreover, the increase is much more noticeable during wider ranges (e.g., more during 31 - 60 than 16 - 30). This is due to the widening of the disparity search space which increases the probability of mismatches which in turn leads to more shifts in matching cost extrema (hence a higher SNCE value).

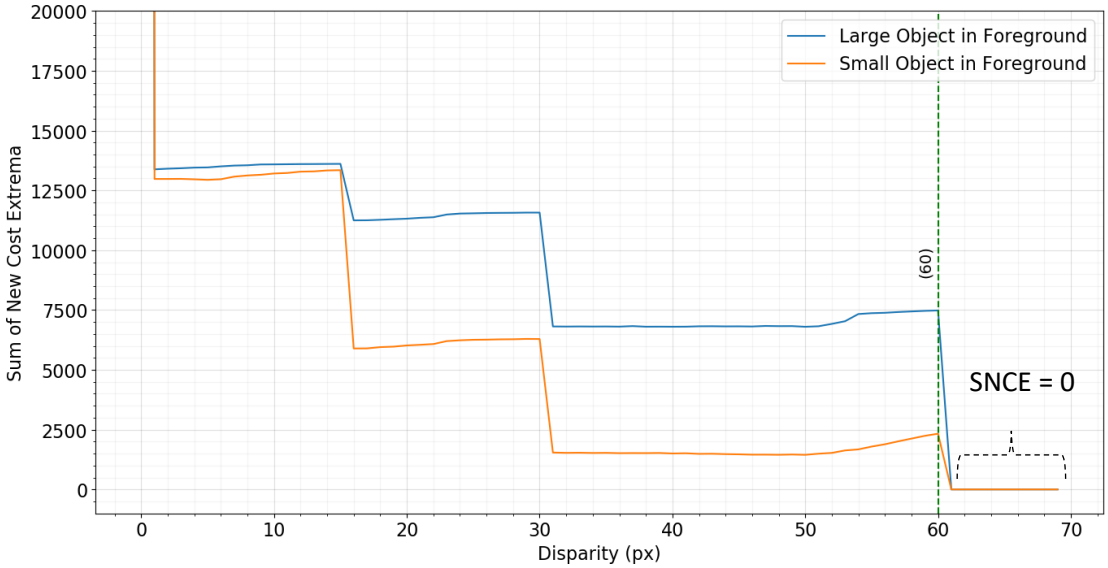


Figure 4.14: SNCE Variations for the two synthetic scenes shown in Figure 4.13. As expected according to Equation (4.8), the scene with the largest object in the foreground has a higher SNCE value towards the end. The steeper decline in SNCE at ground-truth disparities 15, 30 and 60 can also be observed clearly.

In both SNCE profiles (for "Scene 1" and "Scene 2"), the SNCE value has reached zero at disparity 61 just after the largest disparity for the scene which is 60 pixels. However, this was not the case in the SNCE profiles for the stereo images such as Middlebury "Cones", "Bike" and other examples provided earlier. They apparently reached some smaller SNCE value or values around the largest ground-truth disparity for the scene. A quick experiment with noise confirms that with the introduction of even a little bit of noise, the results for the synthetic images also begin to look like the results observed with the standard stereo datasets.

Figure 4.15 shows the updated SNCE variation when a small amount of normally distributed Gaussian noise is added to the synthetic stereo images. The addition of noise introduces more challenges to the SAD-based matching which appears to make the point of convergence of the SNCE profile uncertain. Therefore, the accuracy of the matching techniques should also be included in the analysis.

Important: If the closest foreground object in Figure 4.13 (b) *is made even smaller with just a handful of pixels associated with it, then the SNCE value may become zero or smaller prematurely before reaching the largest stable matching cost extrema for the scene.*

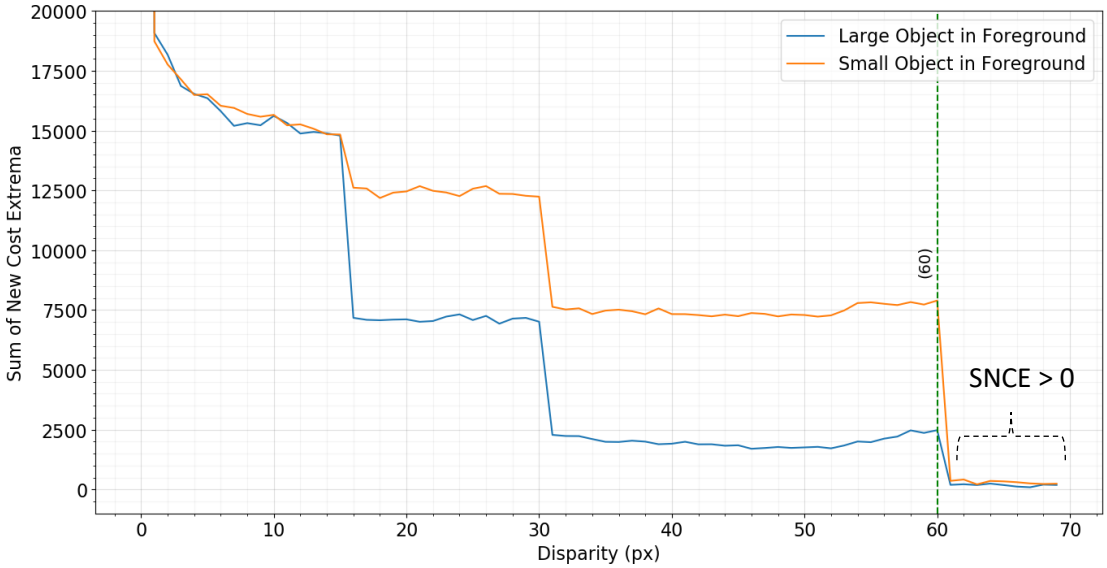


Figure 4.15: SNCE variation profiles for the synthetic stereo images in Figure 4.13 with added Gaussian noise (normally distributed). The noise added to the left and right images are different. In contrast to the previous noise-free scenario, SNCE profiles have not reached a value of zero immediately after the maximum disparity which is reminiscent of the results obtained with the standard stereo images from Middlebury, KITTI and Scene Flow datasets.

4.7 Deterministic Estimation of Maximum Disparity

The experiments and results so far suggest that the value of the SNCE metric (based on the rudimentary aggregation techniques SAD) may reduce to some lower value or values at around the largest disparity for a given scene and then possibly remain low afterwards. However, it is not yet clear how a suitable maximum disparity value can be extracted from the SNCE variations especially due to the challenges such as:

1. The reliance on the foreground objects/scene structure as per the **Equation (4.8)**

   As seen during the development of the equations from 4.3 to 4.8, the probability of locating extrema on the newest layer of the cost volume at disparities below the maximum, depends on the size of the objects in the foreground in terms of the number of pixels they occupy in the image space. However, this is not an assumption which can be made about all the scenes. There can be scenes with only a handful of pixels representing the foreground objects. Therefore, more improvements are required to make it possible to identify a suitable maximum disparity value in a scene independent manner.

2. Non-deterministic results

   Even with larger foreground objects, there exists a certain probability of not locating any extrema at disparities below the largest disparity associated with the pixels. This can make any SNCE-based maximum disparity estimations stochastic in nature. Such uncertain estimates may not be practical for critical applications that depend on stereo vision accuracy. Therefore, further investigations are required to eliminate the uncertainty.

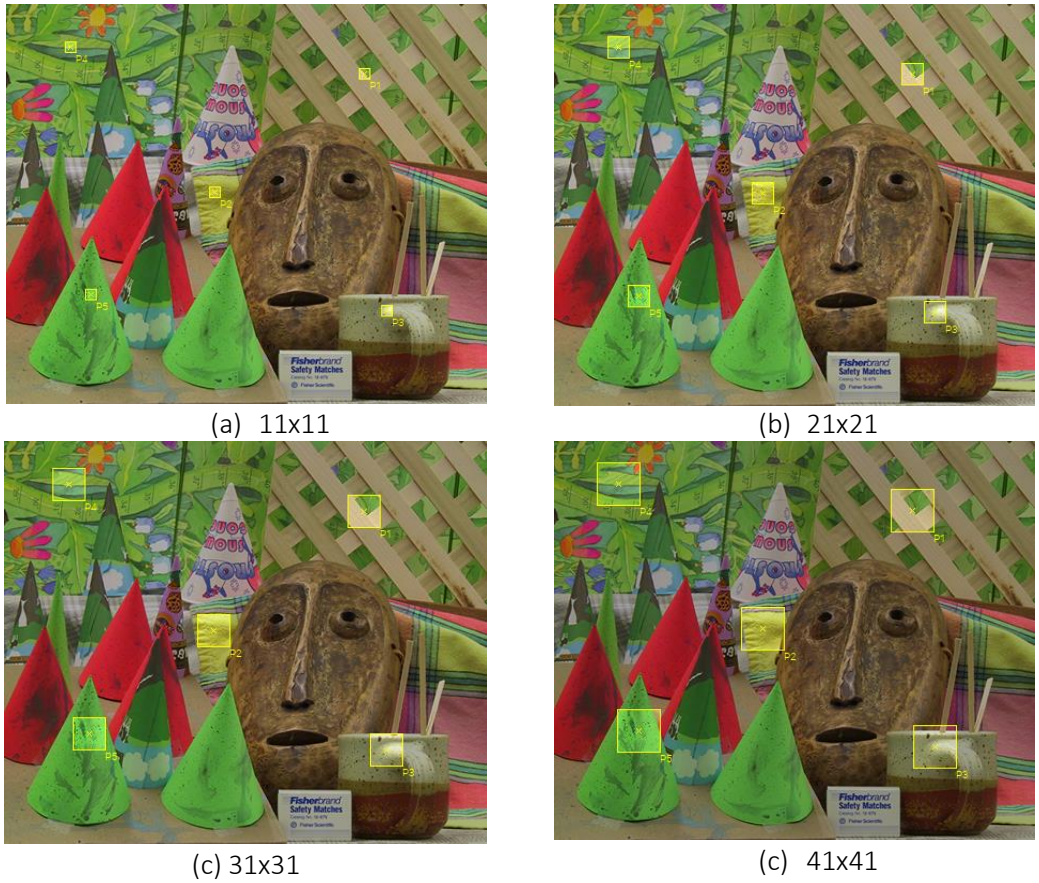
3. Difficulty in extracting a suitable maximum disparity from the SNCE variations observed so far, as they do not converge to a specific value

   So far, the only time the SNCE could be used to uniquely identify a suitable maximum disparity was during the tests with synthetic images. In the absence of noise, the SNCE value for the synthetic stereo image pairs became zero just after maximum disparity. In other examples, SNCE value appeared to reach some non-specific value or values around the largest ground-truth disparity. In such instances, it is difficult to decide when to terminate the cost volume creation process based on the SNCE value. At this point it is tempting to utilize a user-specified threshold to extract a candidate maximum disparity as shown in one of the author's published work in [105]. *However, this thesis further investigates the use of the SNCE metric to find user independent, scene independent and fully deterministic criteria which can be used to terminate a layer-wise cost volume creation process without requiring custom thresholds.*

To address the challenges mentioned above, the focus of this chapter turns towards the stereo matching technique. In all experiments so far, the pixel intensity differences were aggregated using the elementary SAD method with a fixed mask size of 11x11 pixels. As the first step, the effect of the mask size is studied in detail to establish its relationship with the SNCE variation.

4.7.1 SNCE vs. Mask Size

The success of traditional local stereo matching techniques depends on how well the local intensity landscape around a certain pixel is uniquely captured by the aggregation operation. In case of SAD-based stereo matching, the intensity differences between the corresponding points are aggregated over a window of values given by the mask size. Generally, the use of a large mask size yields good results due to the mask encompassing additional information about the pixels in the neighbourhood of the specific pixels being matched as shown in Figure 4.16.



(a) 11x11 (b) 21x21

(c) 31x31 (c) 41x41

Figure 4.16: An illustration of how different mask sizes encompass different levels of information. The points shown are the same 5 points from P1 to P5 used for analysis earlier in the chapter. Large mask sizes often help enclose additional information for comparison.

However, when using traditional techniques like SAD under certain scene conditions such as depth discontinuities/edges, occluded areas and repetitive structures, additional information from the pixels in the neighbourhood can have a negative effect on matching accuracy (depending on the mask size). While acknowledging the positive and negative effects of mask size on matching, the current section aims to explore the effect of different mask sizes on SNCE variation.

4.7.1.1 Extrema Shifts vs. Mask Size

Figure 4.17 shows the effect of a slightly larger mask size of 21x21 on cost extrema movements associated with the five points (P1-P5) during a layer-wise cost volume construction process in which the layers of SAD-based matching costs are added to the cost volume from disparity 0 to 60 sequentially. From the figure, it is notable how the extrema movements associated with the points P1, P2 and P4 have taken place at every disparity in the areas shaded in yellow which span from disparity 0 to the most stable cost minima of the respective points. Moreover, the uniqueness of the cost extrema appears to have improved when compared with the cost distributions obtained with a mask size of just 11x11 pixels shown earlier in **Figure 4.6**.

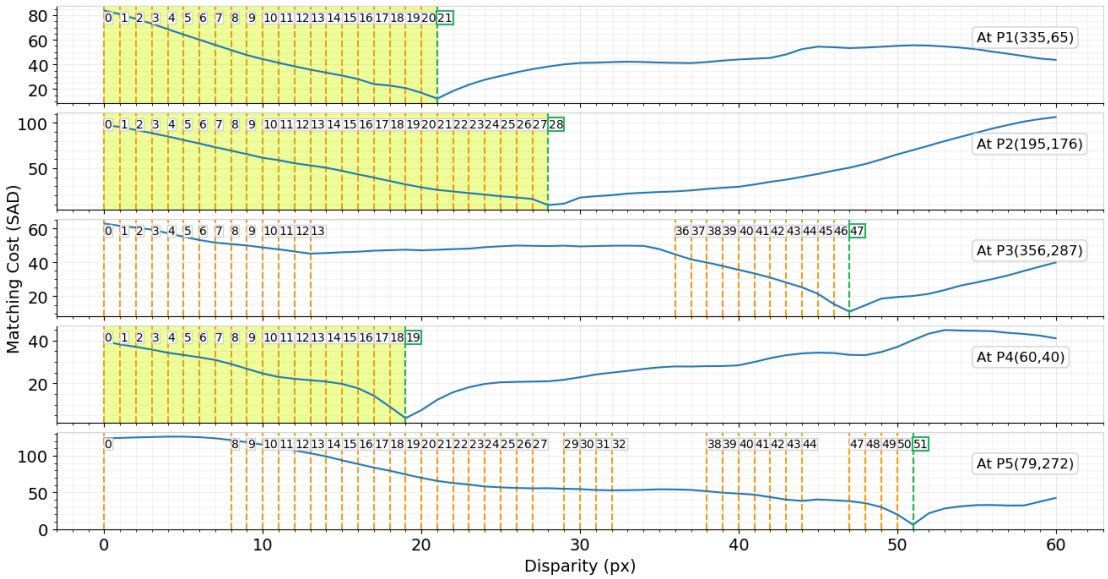


Figure 4.17: Extrema movements observed in the matching cost distributions (calculated using SAD with a mask size of 21x21 compared to 11x11 mask used previously) of 5 points (P1-P5), during the layer-wise cost volume construction process. In the regions highlighted in yellow, the subplots of three points (P1, P2 and P4) have extrema shifts (orange markers) at every disparity up to their respective stable cost minima locations (green markers).

Figure 4.18 (a) and (b) below, show the matching cost distributions and the associated extrema movements (i.e., during a layer-wise cost volume construction process) obtained by increasing the mask size further up to 31x31 and 41x41 pixels, respectively.

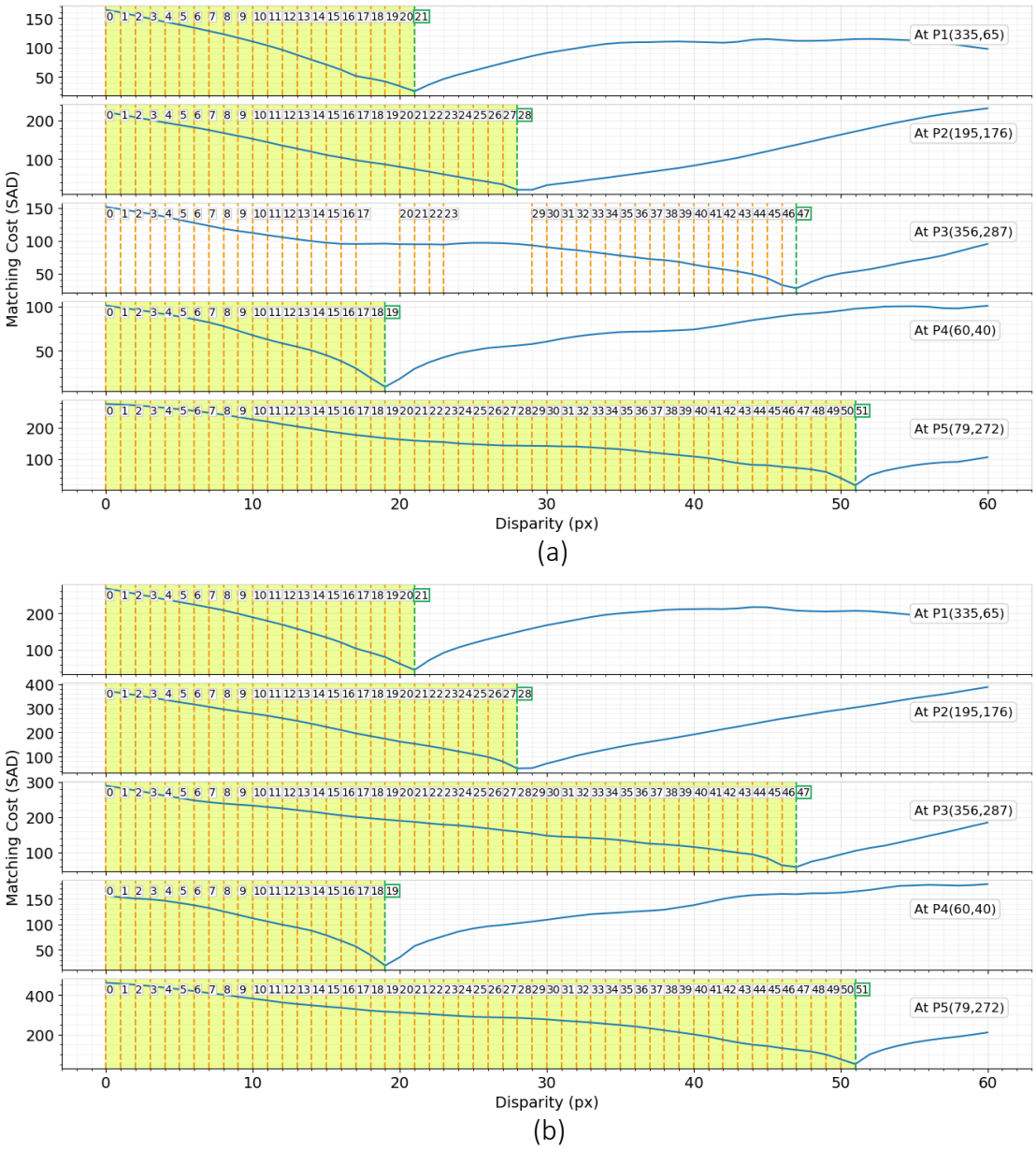


Figure 4.18: Extrema location changes observed in the matching cost distributions of 5 points (P1-P5), during a layer-wise cost volume construction process involving SAD-based costs with a mask size of (a) 31x31 and (b) 41x41 pixels. In the regions highlighted in yellow, extrema movements (orange markers) have taken place at every consecutive disparity up to the most stable cost minima (green markers).

The cost distributions in **Figure 4.17**, **Figure 4.18** (a) and **Figure 4.18** (b) show how the increasing mask size has resulted in more and more pixels having consecutive extrema shifts up to the most stable extrema. Moreover, the matching cost values at disparities after the cost extrema, appear to increase continuously. For example, all cost distributions in **Figure 4.18** (b) have consecutive extrema shifts towards the clearly established unique cost extrema of each pixel. The continuous descent towards a unique minima and subsequent increase in value of a cost distribution is a tell-tale sign of a "Unimodal Cost Distribution".

Definition: **Unimodal Function**: *A function f(x) is said to be "Unimodal" if for some specific value of (x) given by (a):*

- ✓ *the value of f(x) monotonically decreases towards some minima f(a) for all x<a and then monotonically increases for all x>a* **OR**
- ✓ *the value of f(x) monotonically increases towards some maxima f(a) for all x<a and then monotonically decreases for all x>a*

Unimodal cost distributions have significant advantages when it comes to the SNCE estimations compared to non-unimodal (or multi-modal) cost distributions according to the definition given by Equation (4.1). If the unimodality of the cost distributions can be assured, then the Iverson block inside the Equation (4.1) will be non-zero, for at least one of the pixels, until the most stable cost extrema location for all the pixels are reached.

Important: If the cost distributions of a group of pixels are unimodal, then the SNCE value for the same group of pixels at any disparity will converge to zero, if and only if the current disparity is higher than the largest disparity among the most stable cost extrema locations.

From the cost volume standpoint, when layers of costs are added to the cost volume, the unimodal cost distributions introduce extrema changes at every disparity until the stable extrema are reached. As a result, a SNCE variation profile obtained with any such pixel locations, stays non-zero (at least 1 or higher) until the largest disparity among them is reached. In addition, once the largest disparity is reached, then the SNCE value becomes zero and stays zero afterwards. This concept can be verified by plotting the SNCE profile for the 5 cost distributions at a mask size of 41x41 as shown in Figure 4.19.

According to Figure 4.19, it is not possible for the SNCE value to become zero until every cost distribution has reached the respective unique extrema (marked with vertical dashed lines in grey and green).

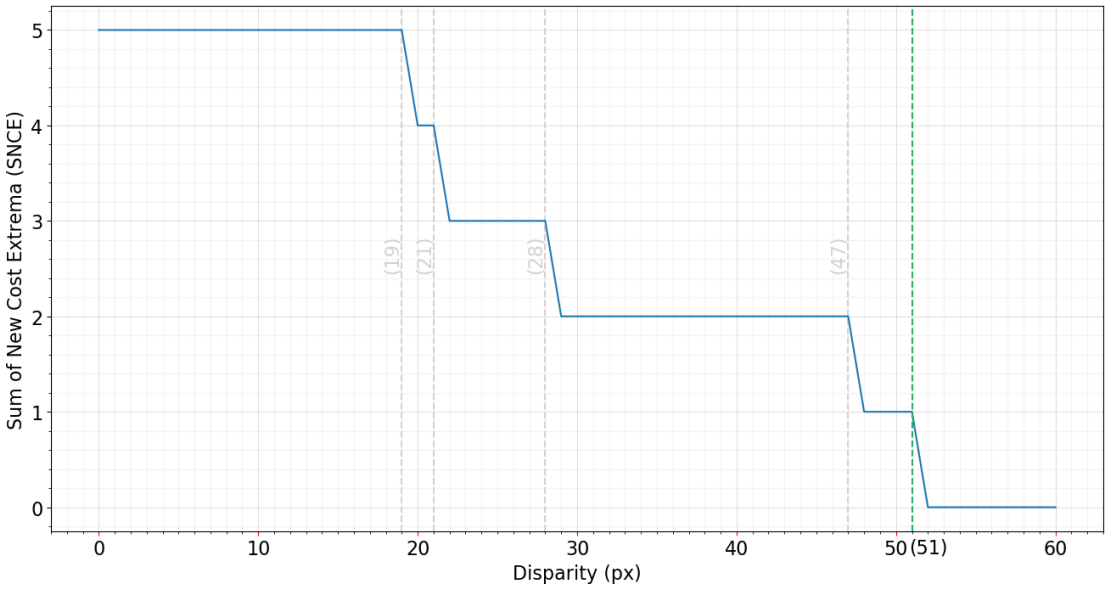


Figure 4.19: SNCE Variation for the unimodal cost distributions of the points P1 to P5 (obtained with SAD over 41x41pixel mask). The vertical dashed-grey lines show the locations of the stable cost extrema of P1 to P4. The largest ground-truth disparity among the 5 pixels is shown with a dashed green line which is the location of the most stable extremum of P5. The SNCE profile converges to zero immediately after the maximum value of 51.

4.8 Addressing the Challenges

Based on **Equation (4.1)** and the results shown in **Figure 4.18** (b) and **Figure 4.19**, any subset of the given 5 pixels in any combination are going to have the same result if their cost distributions are unimodal. In fact, the SNCE variation of a single pixel would stay as 1 until the best-match disparity for the pixel is reached and will remain zero thereafter. Consequently, if all the pixels in the stereo area of a stereo image pair is concerned, it should be possible to find the largest disparity among them based on the SNCE value reaching zero. Hence, as explained below in **Table 4.1**, unimodal cost distribution can address the challenges outlined in the previous section.

	Challenge	Effect of Unimodal Cost Distributions
1	SNCE relies on the size of the foreground objects according to **Equation (4.8)**	*If the cost distributions are unimodal, then the SNCE value will reach zero only after all cost distributions have reached their most stable cost extrema. Therefore, it does not depend on the number of pixels and it would stay valid even with just one single cost distribution.*

2	Non-deterministic or stochastic nature of the results	*Due to the continuously increasing or decreasing nature of the unimodal distributions, the local cost extrema for every pixel, moves to the newest layer of the cost volume until the most stable cost extrema are reached. This ensures a 100% probability of locating at least one extremum at the newest layer, until after all the pixels have reached their stable cost extrema. As a result, the probability given by* ***Equation (4.6)*** *becomes 100% which results in a deterministic outcome.*
3	Difficulty in extracting a suitable maximum disparity from a SNCE variation	*If the cost distributions are unimodal, then the individual cost distributions are going to have unique cost extrema. Therefore, the SNCE metric is not going to report any value other than zero after all the stable cost extrema locations have been reached. That provides a unique and unambiguous candidate for maximum disparity for the group of pixels in concern. Therefore, if the layers of unimodal costs are added to a cost volume (one layer at a time) and the SNCE value at each layer is computed, the first occurrence of a zero SNCE will signify the last stable cost extrema for the pixels in concern. Therefore, the cost volume creation process can terminate immediately.*

Table 4.1: How unimodal cost distributions address the challenges associated with SNCE when estimating a suitable maximum disparity value for a scene

4.9 Achieving Unimodality Across All Pixels

The study so far has revealed the feasibility of using SNCE for detecting the maximum disparity among a group of pixel locations in a stereo image pair subjected to the unimodality of the cost distributions. However, it remains to be asked whether it is possible to achieve unimodal cost distributions for all pixel locations in a pair of stereo images. Previously, it was shown how the increase in mask size resulted in SAD-based traditional stereo matching costs becoming more unimodal. Therefore, it is important to study the effect of mask size on unimodality of the matching costs.

4.9.1 Unimodality vs. Mask Size

To test the effect of mask size on unimodality of the cost distributions, 15 SAD-based cost volumes were created for the same Middlebury 2003 "Cones" stereo image pair using different mask sizes starting from 11x11 pixels up to extremely large 201x201 pixels. The stereo images were padded with zeros in instances where the area covered by the mask exceeded the image boundaries. The depth of the cost volume was set to 60 layers. Then, every individual matching cost distribution was tested for the unimodality using the following criteria.

- ✓ Existence of a clearly unique cost minima
- ✓ Consecutive changes in cost minima location for disparities starting from zero up to the most stable cost minima

The total number of pixels with unimodal distributions were recorded as a percentage of the total number of pixels in the Middlebury 2003 "Cones" image pair at quarter resolution which is 168,750 pixels. A plot of the results is shown in Figure 4.20.

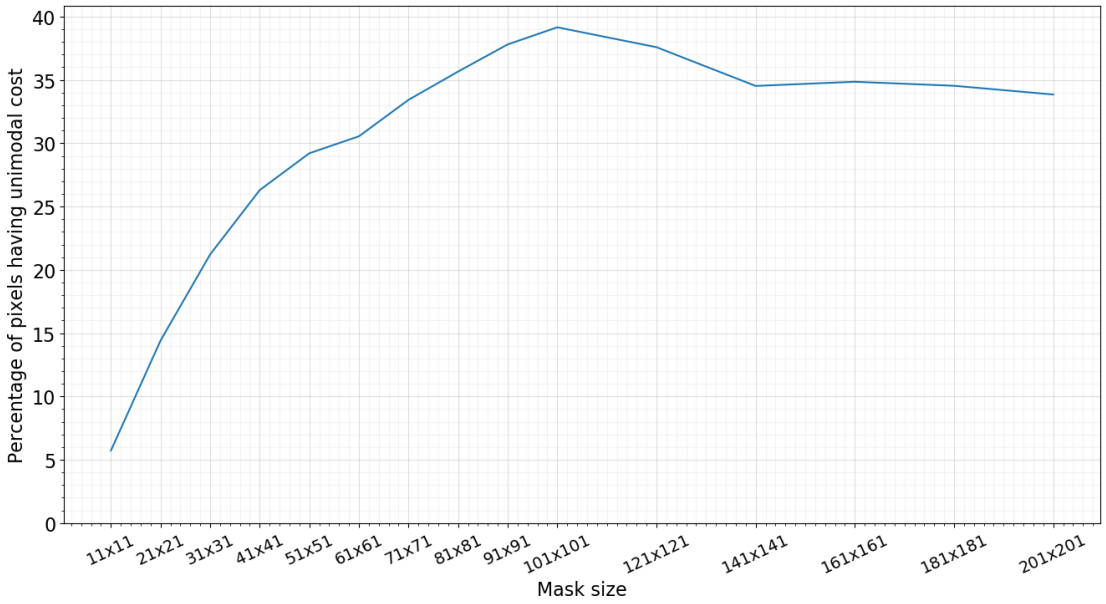


Figure 4.20: Percentage of pixels with unimodal cost distributions in a SAD-based cost volume when built using different mask sizes. Middlebury 2003 "Cones" image pair at quarter resolution (450x375) has been used for the experiment. Areas of the masks that exceed the image boundaries have been padded with zeros during the process.

As it can be seen from the figure, the percentage of unimodal cost distributions for SAD-based costs has increased with the mask size and the value has peaked at a mask size of 101x101 pixels. Thereafter, the percentage has started to decrease which is an indication of some cost distributions transitioning from being unimodal to non-unimodal (i.e., multimodal). According to the graph, the maximum percentage of unimodal cost distributions that can be achieved by changing the mask size is slightly lower than 40% of the total number of pixels. That again is at a large mask size of 101x101 pixels compared to the image size which is only 450x375 pixels. In summary, increases in mask size is not feasible from the practical standpoint and does not further improve the percentage of unimodal distributions.

4.9.2 Unimodality vs. Traditional Stereo Matching

Until now, only the SAD based cost aggregation has been used for generating the matching costs. However, there are other types of traditional cost aggregation techniques such as Sum of Squared Difference (SSD), Normalized Cross Correlation (NCC), Rank Transform (RT) and Census Transform (CT). Figure 4.21 shows the variation of unimodality as a percentage with changing mask sizes for SAD and other cost aggregation techniques, which have been obtained by following the same process as earlier. According to the plots, the algorithms such as SSD, NCC, RT and CT, lag SAD in

terms of the percentage of pixels with unimodal cost distributions. Particularly the Rank Transform RT and Census Transform techniques have reported very low percentage of unimodal cost distributions even at larger mask sizes.

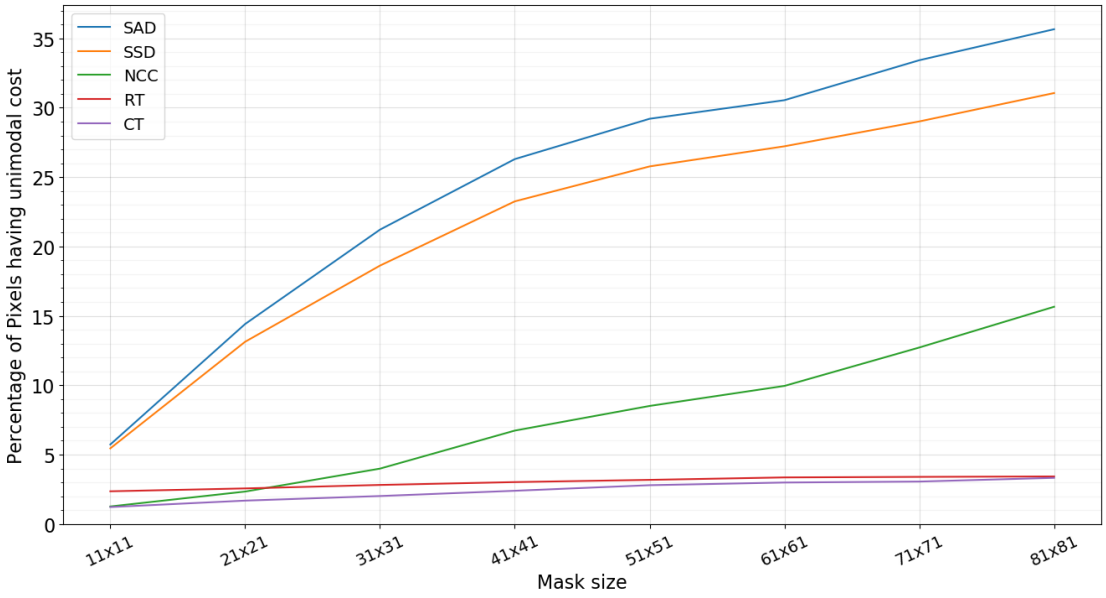


Figure 4.21: A comparison of the percentage of pixels with unimodal cost distributions in cost volumes constructed with different cost aggregation techniques and mask sizes. Middlebury 2003 "Cones" image pair at quarter resolution (450x375) has been used for building the cost volumes. Zero-padding has been used in areas of the masks that exceed the image boundaries.

Further tests including the best performing techniques above (i.e., SAD, SSD and NCC) reveal that even at extremely large mask sizes, the results do not vary much. The SAD-based aggregation still produces the highest percentage of unimodal cost distributions. The SSD and NCC methods do catch up at larger mask sizes only to decline later as shown in **Figure 4.22**.

According to **Figure 4.22**, SSD and NCC techniques also show a decline in the percentage of unimodal cost distributions after some mask sizes (SSD at 121x121 and NCC at 181x181). That is a clear indication of cost distributions for some pixel locations transitioning from being unimodal to non-unimodal cost distributions. The key take-away here is the impracticality of finding a single mask size or a cost aggregation algorithm that can produce unimodal cost distributions across all pixels.

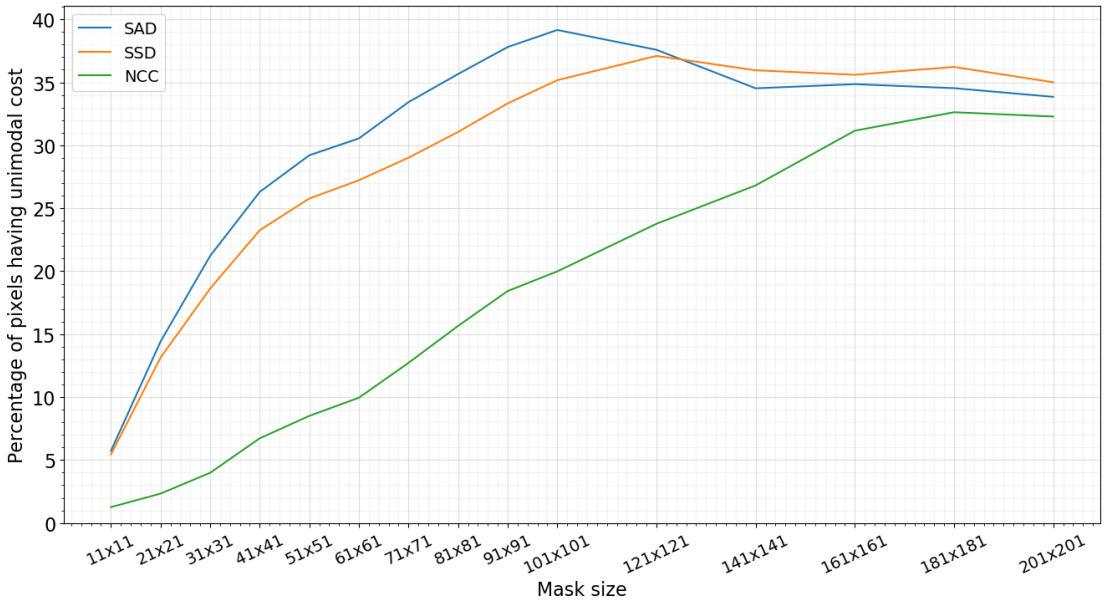


Figure 4.22: A comparison of the percentage of pixels with unimodal cost distributions in cost volumes constructed with SAD, SSD and NCC cost aggregation techniques at different mask sizes. Middlebury 2003 "Cones" image pair at quarter resolution (450x375) has been used for building the cost volumes. Zero-padding has been used in areas of the masks that exceed the image boundaries.

4.9.3 Practical Methods for Achieving Unimodality

Local stereo matching techniques depend on the use of masks (adaptive or fixed-sized) to capture additional information in the neighbourhood of a given pixel when matching. One potential avenue for obtaining unimodal cost distributions with local matching techniques would be to use adaptive mask sizes. On the other hand, global stereo methods depend on a fully built cost volume for subsequent optimization in an iterative process. Therefore, they are naturally incompatible with a layer-wise cost volume construction process.

However, there is a much more feasible solution which was already discussed during the literature review. Most of the modern end-to-end deep learning-based stereo techniques fundamentally depend on unimodality of the matching cost distributions due to the use of the differentiable "Softargmin" operation. Most importantly, they do not require users to configure any scene or user dependent parameters to achieve unimodal distributions. Instead, they can learn to produce unimodal distributions through supervised learning. This inherent property of deep learning techniques needs to be examined to determine the potential of using the same to achieve unimodal cost distributions. That would in turn lead to deterministic results when using the SNCE metric to predict a suitable maximum disparity value for a given scene.

4.10 Chapter Summary

Chapter 4 started with an in-depth analysis of the matching cost distributions within a partially built cost volumes to study the movements of SAD-based matching cost extrema during a layer-wise cost volume construction process. A metric called "Sum of New Cost Extrema (SNCE)" was developed which captures the movement of matching cost extrema across all the pixels being matched. Tests with stereo image samples from standard stereo datasets such as Middlebury, KITTI and Scene Flow revealed that the metric appears to become smaller in value once the most stable cost extrema for all the pixels have been reached. However, a clear point of convergence could not be established.

A probability based mathematical model was developed to explain the observed behaviour of the SNCE metric. The model revealed that the SNCE metric when taken as it is, depends on the number of image pixels occupied by the foreground objects. Tests using synthetic data confirmed the findings. Additional tests with different mask sizes revealed that unimodal cost distributions lead to the SNCE metric reaching a value of zero once all the stable cost extrema is reached. Thus, it was apparent that the SNCE metric can be used to deterministically estimate the largest disparity associated with a group of pixels subjected to the unimodality of cost distributions.

However, achieving unimodality across all pixel locations using traditional local stereo matching techniques was found to be challenging. Even with larger mask sizes, the peak percentage of unimodal pixels attainable was slightly lower than 40% of the total number of image pixels. On the other hand, global stereo methods depend on a fully built cost volume for the subsequent optimization in an iterative process which makes them incompatible with a layer-wise cost volume construction process or the SNCE metric. According to the background study in Chapter 2, a better alternative for achieving unimodal cost distributions is available via deep learning-based stereo vision. Deep stereo techniques depend on the unimodality of matching costs for accurate disparity regression through the differentiable "Softargmin" operation. Hence, the convergence of the SNCE metric on deep stereo networks must be analysed next, provided that the SNCE metric can be computed efficiently as part of the already complex computations associated with deep stereo algorithms.

CHAPTER 05 - EFFICIENT ESTIMATION OF THE SNCE METRIC

Introduction

This chapter aims to analyse the computational complexity of the SNCE metric which was introduced in the previous chapter. The main objective is to investigate the feasibility of using the metric with stereo vision algorithms. First, a comprehensive algorithmic analysis is conduced to identify the order of growth of computations associated with the SNCE metric. Experiments are then conducted to validate the same empirically using C++ and CUDA based implementations of a basic stereo algorithm with embedded SNCE. Finally, the results are analysed to estimate the computational overhead of the SNCE process on both CPU and GPU architectures.

5.1 Time Complexity of SNCE

The time complexity associated with the SNCE metric can be determined by analysing the operations contained in the mathematical expression below (**Equation (5.1)**), which was developed in the previous chapter. **Equation (5.1)** contains an Iverson block followed by a summation operation over the pixel space *N*. The Iverson block encloses a comparison between the current disparity *d* and the result of the argmin operation along the one-dimensional array of matching costs C_i associated with the i^{th} pixel. Therefore, the analysis should start with the "argmin" operation before incorporating the comparison and summation operations.

$$SNCE(d) = \sum_{i=1}^{N} [argmin(C_i) == d] \tag{5.1}$$

5.1.1 Algorithmic Complexity of the Enclosed "Argmin" Operation

A simple algorithm for the operations contained in the Iverson block in **Equation (5.1)** is shown in the pseudo code below.

ALGORITHM 01: *Computing the Iverson Block in Equation 5.1 above.*

```
//Calculate the disparity at which the minimum matching cost is located
//Input: Array D_i[1 × d] matching costs associated with any pixel up to a disparity (d)
//Output: Integer 1 or 0; An output of 1 if the minimum index equals to (d-1); 0 otherwise
01 : min_index ← D_i[0]
02 : for i ← 0 to (d − 1) do
03 :     if D_i[i] < min _index:
04 :         min_index ← i
05 : if min_index eq (d − 1):
06 :     return 1
07 : return 0
```

As per the pseudo code for ALGORITHM 01, there are two basic operations that take place during the computation. If t_f is the execution time for the comparison with assignment within the loop (shaded in yellow in pseudo code), then the total execution time T_f for the loop (at d^{th} disparity) can be approximated by the following equation (Equation (5.2)).

$$T_f(d) \approx \sum_{0}^{d-1} t_f \tag{5.2}$$

If the comparison operation or any follow up calculation after the *for* loop takes t_c time to execute (as shaded in blue in ALGORITHM 01 above), then the total execution time for the Iverson block (at d^{th} disparity) in Equation (5.1) can be estimated to be:

$$T(d) \approx \sum_{0}^{d-1} t_f \; + \; t_c \tag{5.3}$$

According to Equation (5.3), the total number of calculations that needs to be performed is given by *d + 1* which includes *d* number of operations within the loop and 1 operation outside. Therefore, the complexity of operations contained in the Iverson block has an order of growth equal to *d*.

5.1.2 "Argmin" Operation Over the Total Image Space

The calculation enclosed in the Iverson block of Equation (5.1) must be computed for all *N* pixels in the stereo images. If the width and height of the images are denoted by (w) and (h), then the pseudo code below outlines the computation of the minima based SNCE at a disparity value denoted by *d*.

ALGORITHM 02: *Computing minima based SNCE*
//Calculate the SNCE at a disparity value of d
//Input: 3D Array $CV_i[h \times w \times d]$ *matching costs*
//associated with any pixel up to a disparity d
//Output: Integer ($snce\_value$*); SNCE value at disparity d*

```
snce_value ← 0
for m ← 0 to (h − 1) do:
    for n ← 0 to (w − 1) do:
        min_index ← CV_i[m, n, 0]
        for i ← 0 to (d − 1) do
            if CV_i[m, n, i] < min _index:
                min_index ← i
        if min_index eq (d − 1):
            snce value ← snce_value + 1
return snce_value
```

As shown in the pseudo code, the total execution time for SNCE calculation at a certain disparity *d* increases with the number of pixels in the images. Therefore, the total time it takes to complete all the computations can be approximated by the following expression.

$$T_f(d) \approx \sum_{0}^{h-1} \sum_{0}^{w-1} \left(\sum_{0}^{d-1} t_f + t_c \right) \tag{5.4}$$

5.1.3 Estimating SNCE at Each Disparity

It is important to note that the SNCE values must be calculated for all disparities starting from 0 to some higher disparity value $dmax$ which is expected to be found using the SNCE metric itself. At zero disparity, the total number of calculations required to estimate SNCE is proportional to the number of pixels in the image which is equal to *n*. At a disparity value of *d,* the same estimation expands to *n* x *d* or *w* x *h* x *d* times. Hence, it is apparent that there is a best-case-scenario (at disparity 0) and a worst-case-scenario (at the maximum disparity). In the best-case scenario at zero disparity, the computational time for estimating SNCE value can be expressed by:

$$T_0 = T_f(0) \approx \sum_{0}^{h-1} \sum_{0}^{w-1} t_c \tag{5.5}$$

At the maximum disparity of $dmax$ the computational time estimation reaches the worst-case-scenario which is given by the following expression.

$$T_{Dmax} = T_f(Dmax) \approx \sum_{0}^{h-1} \sum_{0}^{w-1} \left(\sum_{0}^{dmax-1} t_f + t_c \right) \tag{5.6}$$

Therefore, the total computational time T_{total} for the estimation of SNCE values for all disparities from zero to the maximum disparity equals to the sum of all estimations (from the best case to the worst-case scenarios).

$$T_{total}(Dmax) \approx T_0 + T_1 + \cdots + T_{Dmax} \tag{5.7}$$

$$T_{total}(dmax) \approx \sum_{0}^{h-1}\sum_{0}^{w-1} t_c + \sum_{0}^{h-1}\sum_{0}^{w-1}\left(\sum_{0}^{1} t_f + t_c\right) + \cdots + \sum_{0}^{h-1}\sum_{0}^{w-1}\left(\sum_{0}^{dmax-1} t_f + t_c\right) \quad (5.8)$$

5.2 The Order of Growth in Computations

The order of growth of an algorithm refers to the growth in the number of computational steps when the size of the inputs is increased. In order to estimate the order of growth in computations, it is important to establish the total number of computational steps as a function of the input dimensions.

The total number of steps that leads to $T_{total}(dmax)$ in Equation (5.8) can be calculated by summing the number of computations associated with each of the disparities. If the same is denoted by C_{total}, it can be expressed in the following manner.

$$C_{total} = \{w \times h \times 1\} + \{w \times h \times 2\} + \{w \times h \times 3\} + \cdots + \{w \times h \times (dmax + 1)\}$$

$$C_{total} = (w \times h)\{1 + 2 + 3 + \cdots + (dmax + 1)\}$$

$$C_{total} = \frac{(w \times h)(dmax + 1)(dmax + 2)}{2} \quad (5.9)$$

According to the expression for C_{total}, the order of growth of the calculations increases with the input size (which is determined by the height and width) and the quadratic value of the disparity search range. If *n* denotes the size of the input in a single dimension, the order of growth (when estimating minima-based SNCE values for all disparities up to *d* can be incorporated into an expression using the standard "Big-oh" (Big-O) notation.

$$C(n, d) \in O\ (n^2 d^2) \quad (5.10)$$

5.2.1 Order of Growth in Stereo Disparity Estimation

From the analysis so far, the order of growth of the SNCE estimation was found to be extremely high. However, it needs to be compared with the complexity of stereo vision algorithms which already have a higher order of growth and contain most of the computations required by the SNCE metric. For example, consider the pseudo code of a Sum-of-Absolute-Difference (SAD) based stereo disparity estimation method (for grayscale images) which is given below. The algorithm uses a mask size of $(k \times k)$ for cost aggregation with SAD with winner-takes-all (WTA) disparity selection.

ALGORITHM 03: *Computing a disparity map from a stereo image pair*
//Estimate the disparity map using two calibrated grayscale stereo images
//Input: Maximum disparity (dmax), mask size $[k \times k]$,
//Stereo image pair $Img_1[h \times (w + dmax)]$ *(padded with zeros),* $Img_2[h \times w]$
//Output: $disparity\_map[h \times w]$

$disparity \leftarrow zeros[h \times w]$
$minval \leftarrow zeros[h \times w]$
$r \leftarrow \boldsymbol{int}(k/2 - 1)$
$\boldsymbol{for}\ d \leftarrow 0\ \boldsymbol{to}\ (dmax - 1)\ \boldsymbol{do}$:
    $\boldsymbol{for}\ m \leftarrow r\ \boldsymbol{to}\ (h - 1)\ \boldsymbol{do}$:
        $\boldsymbol{for}\ n \leftarrow r\ \boldsymbol{to}\ (w - 1)\ \boldsymbol{do}$:
            $sad = 0$
            $\boldsymbol{for}\ a \leftarrow 0\ \boldsymbol{to}\ (k - 1)\ \boldsymbol{do}$:
                $\boldsymbol{for}\ b \leftarrow 0\ \boldsymbol{to}\ (k - 1)\ \boldsymbol{do}$:
                    $dif \leftarrow abs\begin{pmatrix} img_1[m - r + a,\ n - r + b] \\ -img_2[m - r + a,\ n - r + b] \end{pmatrix}$
                    $sad \leftarrow sad + dif$
            $\boldsymbol{if}\ d\ \boldsymbol{eq}\ 0$:
                $minval[m, n] \leftarrow sad$
            $\boldsymbol{else}$:
                $\boldsymbol{if}\ sad\ \boldsymbol{less\ than}\ minval[m, n]$:
                    $minval[m, n] \leftarrow sad$
                    $disparity[m, n] \leftarrow d$
$\boldsymbol{return}\ disparity$

Following a similar process as before, the order of growth of the computations of the given stereo algorithm can be estimated from the pseudo code to be the following (Equation (5.11)).

$$C_{SAD\ stereo}(n, d, k) \in O(n^2 d\, k^2) \tag{5.11}$$

5.2.2 Embedding SNCE in Stereo

One clear observation from the pseudo code of the SAD-based stereo algorithm (ALGORITHM 03) is the fact that, most of the calculations required for estimating SNCE already exist in the stereo estimation algorithm itself. Therefore, it is possible to embed SNCE calculation in the stereo vision algorithm without adding a $O(n^2d^2)$ computational burden on top of the stereo algorithm that already have an order of growth equal to $O(n^2d\,k^2)$ which would be prohibitively expensive.

Consider the following pseudo code which shows a modified version of the stereo algorithm which includes an embedded SNCE estimation process. The changes are marked with boxes shaded in blue and yellow.

ALGORITHM 04: *Computing a disparity map from a stereo image pair while calculating SNCE*
//Estimate the disparity map using a calibrated grayscale stereo image pair
//while calculating SNCE at each disparity.
//Input: Maximum disparity (dmax), mask size $[k \times k]$,
//Stereo image pair $Img_1[h \times (w + dmax)]$ *(padded with zeros),* $Img_2[h \times w]$
//Output: $disparity\_map[h \times w]$

```
01 : disparity ← zeros[h × w]
02 : minval ← zeros[h × w]
03 : snce ← zeros[dmax]
04 : r ← int(k/2 − 1)
05 : for d ← 0 to (dmax − 1) do:
07 :     for m ← r to (h − 1) do:
08 :         for n ← r to (w − 1) do:
09 :             sad = 0
10 :             for a ← 0 to (k − 1) do:
11 :                 for b ← 0 to (k − 1) do:
12 :                     dif ← abs( img1[m − r + a, n − r + b]
                                  −img2[m − r + a, n − r + b] )
13 :                     sad ← sad + dif
14 :             if d eq 0:
15 :                 minval[m, n] ← sad
16 :                 snce[d] ← snce[d] + 1
17 :             else:
18 :                 if sad less than minval[m, n]:
19 :                     minval[m, n] ← sad
20 :                     disparity[m, n] ← d
21 :                     snce[d] ← snce[d] + 1
22 : return disparity
```

As shown in the modified pseudo code, with only 2 additional steps, the whole SNCE process can be incorporated into the stereo disparity estimation process without changing the order of growth of computational steps. This contrasts with the naive SNCE estimation process which had a growth in the order of $O(n^2d^2)$. If small, fixed mask sizes are concerned, the order of growth of the SNCE embedded-stereo method can be lower than the earlier estimation of $O(n^2d^2)$. In ALGORITHM 04, a 1D array is used to store the SNCE values. If the maximum disparity is unknown at the beginning, then the size of the array can be set to the image width. At the end of each iteration, some SNCE-based exit criteria can be used to terminate the process as shown in green in the modified pseudo code (ALGORITHM 05) below.

ALGORITHM 05: *Computing a disparity map with SNCE based maximum disparity*
//Estimate the disparity map using a calibrated grayscale stereo image pair while
//using SNCE to determine the maximum disparity based on some SNCE threshold
//Input: Maximum disparity (dmax), mask size $[k \times k]$*,SNCE threshold (threshold)*
//Stereo image pair $Img_1[h \times (w + dmax)]$ *(padded with zeros),* $Img_2[h \times w]$
//Output: $disparity\_map[h \times w]$ *Output of 1 if the minimum index equals to (d); 0*
//otherwise

01 : $disparity \leftarrow zeros[h \times w]$
02 : $minval \leftarrow zeros[h \times w]$
03 : $snce \leftarrow \text{zeros}[w]$
04 : $criteria \leftarrow (some\ user\ defined\ criteria)$
05 : $r \leftarrow \textbf{int}(k/2 - 1)$
06 : $\textbf{for}\ d \leftarrow 0\ \textbf{to}\ (w-1)\ \textbf{do}$:
07 : $\quad \textbf{for}\ m \leftarrow r\ \textbf{to}\ (h-1)\ \textbf{do}$:
08 : $\quad\quad \textbf{for}\ n \leftarrow r\ \textbf{to}\ (w-1)\ \textbf{do}$:
09 : $\quad\quad\quad sad = 0$
10 : $\quad\quad\quad \textbf{for}\ a \leftarrow 0\ \textbf{to}\ (k-1)\ \textbf{do}$:
11 : $\quad\quad\quad\quad \textbf{for}\ b \leftarrow 0\ \textbf{to}\ (k-1)\ \textbf{do}$:
12 : $\quad\quad\quad\quad\quad dif \leftarrow abs\begin{pmatrix} img_1[m-r+a,\ n-r+b] \\ -img_2[m-r+a,\ n-r+b] \end{pmatrix}$
13 : $\quad\quad\quad\quad\quad sad \leftarrow sad + dif$
14 : $\quad\quad\quad \textbf{if}\ d\ \textbf{eq}\ 0$:
15 : $\quad\quad\quad\quad minval[m,n] \leftarrow sad$
16 : $\quad\quad\quad\quad snce[d] \leftarrow snce[d] + 1$
17 : $\quad\quad\quad \textbf{else}$:
18 : $\quad\quad\quad\quad \textbf{if}\ sad\ \textbf{less than}\ minval[m,n]$:
19 : $\quad\quad\quad\quad\quad minval[m,n] \leftarrow sad$
20 : $\quad\quad\quad\quad\quad disparity[m,n] \leftarrow d$
21 : $\quad\quad\quad\quad\quad snce[d] \leftarrow snce[d] + 1$
22 : $\quad \textbf{if}\ snce(d)\ \textbf{meets}\ criteria$:
23 : $\quad\quad \textbf{break}$
24 : $\textbf{return}\ disparity$

5.3 Empirical Analysis

So far, the theoretical analysis of a basic stereo vision algorithm has shown that the SNCE estimation can be embedded without increasing the order of growth of the computations. However, it is important to validate the same using empirical methods and find out exactly how much additional time complexity is added by SNCE. To investigate the same, this section includes experiments involving C++ based implementations of the algorithms presented in the preceding sections.

5.3.1 Methodology

Four sets of stereo image pairs were created using the "Bike" image pair from the Middlebury 2014 dataset by progressively resizing the images and reducing the width and height by half. The first image pair has the same width and height as the original (2964x2000 pixels) whereas the last image pair has 1/8th of the initial width and height. The maximum disparity was set to 280 pixels at the full resolution and then reduced by half every time the images were rescaled by a factor of two. Hence, at half-quarter resolution, maximum disparity is limited to 35 pixels. Mask size was set to 3x3. Disparity maps were then estimated using the SAD-based basic stereo algorithm on a 1.8 GHz AMD CPU while measuring the computational time. Algorithm was executed 5 times at each resolution to obtain an average value. The final values were analysed to verify if they match the order of growth given by $O(n^2 d\ k^2)$. Subsequently, the SNCE related changes were introduced to the algorithm and the experiments were repeated to verify if the new set of results follow the same order of growth. Please refer to *Appendix B* to locate source code and the software used.

5.3.2 Results

5.3.2.1 Order of Growth of the Stereo Algorithm

The Table 5.1 below shows the results when using the stereo algorithm alone (without the SNCE operation). As seen from the results, algorithm has taken close to 3 minutes (178.29 seconds) to compute the disparity map at full resolution (2964x2000) while taking only (0.33 seconds) at half-quarter resolution of 370x250.

According to the theoretical analysis earlier, the order of growth of the computational time was found to be $O(n^2 d\ k^2)$ which indicate that the computational time should decrease by 8 (2^3) times, every time the width, height and disparity are reduced by half (while keeping the mask size constant at 3x3). The last row in Table 5.1, shows the

execution time for half, quarter and half-quarter resolution levels multiplied by 8. It is apparent from the results that the computational time has reduced by approximately 8 times. Therefore, it can be concluded that the order of growth found through algorithmic analysis, is approximately the same as the actual order of growth observed when the basic SAD-based stereo algorithm was run on actual computing hardware.

	Execution Time in Seconds (s)			
Resolution	*Full (2964x2000)*	*Half (1482x1000)*	*Quarter (741x500)*	*Half of Quarter (370x250)*
Take 1	175.80	22.28	2.71	0.33
Take 2	182.45	22.65	2.76	0.34
Take 3	180.36	22.38	2.74	0.35
Take 4	175.99	21.94	2.70	0.34
Take 5	176.84	22.10	2.75	0.34
Average	178.29	22.27	2.73	0.34
x8		178.18	21.83	2.73

Table 5.1: Execution times in seconds for SAD based stereo disparity estimation algorithm using the Middlebury 2014 "Bike" image pair at 4 different resolution levels with a maximum disparity of 280 at full resolution. Maximum disparity was halved each time the width and height were reduced by half. The arrow connections show how the execution times have reduced by approximately 8 times at each level.

5.3.2.2 Order of Growth of the Stereo Algorithm with SNCE

When the same experiments were repeated with changes to accommodate SNCE computations, the average computational time did not increase significantly (as shown in Table 5.2). Like in the previous experiments, the reduction of maximum disparity, width and height by half has resulted in computational times improving by approximately 8 times.

	Execution Time in Seconds (s)			
Resolution	*Full (2964x2000)*	*Half (1482x1000)*	*Quarter (741x500)*	*Half of Quarter (370x250)*
Take 1	180.50	22.29	2.83	0.34
Take 2	177.14	22.29	2.75	0.36
Take 3	181.61	22.20	2.76	0.34
Take 4	196.13	23.04	2.75	0.35
Take 5	180.66	22.45	2.81	0.36
Average	183.21	22.45	2.78	0.35
x8		179.64	22.25	2.81

Table 5.2: Execution times in seconds for SAD based stereo disparity estimation algorithm with embedded SNCE computations using the Middlebury 2014 "Bike" image pair at 4 different resolution levels with a maximum disparity of 280 at full resolution. Maximum disparity was halved each time the width and height were reduced by half. The arrow connections show how the execution times have been reduced by approximately 8 times at each level.

It is also important to note that the overall computational time at each resolution level has not increased significantly despite calculating SNCE at each disparity. For example, at half-quarter resolution, the time has only increased by 10 ms (from 340 ms to 350

ms). Table 5.3 below shows the increase in computational time as a percentage of the average execution time in the absence of SNCE calculation.

	Execution Time in Seconds (s)			
Resolution	*Full (2964x2000)*	*Half (1482x1000)*	*Quarter (741x500)*	*Half of Quarter (370x250)*
Stereo Only	178.29	22.27	2.73	0.34
Stereo with SNCE	183.21	22.45	2.78	0.35
Increase in Time (seconds)	4.92	0.18	0.05	0.01
Percentage Increase	2.76	0.82	1.78	2.90

Table 5.3: A comparison of the execution times between the basic stereo and SNCE-based stereo

As seen from the results in Table 5.3, the increase in computational time due to SNCE is less than 3% of the total execution time of the algorithm.

5.4 Parallelization of SNCE Estimation

Deep learning-based state-of-the-art algorithms use the power of parallel processing on GPUs to expedite the training and inference times of stereo vision algorithms. Embedding SNCE into such deep learning methods requires the SNCE metric to be suitable for parallel processing. Therefore, the fittingness of SNCE for parallel processing is investigated in this section.

As shown at the beginning of the chapter, SNCE calculation for a particular disparity involves an "argmin" operation followed by a comparison, results of which must be summed over all the pixels. Therefore, an intuitive solution for parallelization of SNCE would be to assign the operations enclosed in the Iverson brackets of Equation (5.1), to individual threads on a GPU. However, the summation of the individual results requires the threads to be synchronized.

5.4.1 CUDA Programming

All parallel computing related experiments in this chapter are conducted using the CUDA (Compute Unified Device Architecture) parallel computing platform using NVIDIA devices. CUDA programming is based on C-language and the development depends on the architecture of the GPU device being used. The pieces of code written for NVIDIA devices that support CUDA are called "kernels" or "kernel functions" which are executed on many parallel threads in the GPU. Threads in a GPU are structured as grids of thread blocks. Therefore, when executing a kernel, thread requirements can be specified by specifying the block dimension in terms of threads, and the grid

dimension in terms of the number of such blocks. For example, a 4x4 grid of 4x4 blocks would indicate 16 blocks with 16 threads each (i.e., grids and blocks can have 1D, 2D or 3D dimensions). However, the "grids and blocks" structure is important for accessing threads and assigning tasks to each using CUDA programming. CUDA provides several variables to access the grid-block architecture. For example, "*threadIdx*" variable can be used to identify threads so that "*threadIdx.x*", "*threadIdx.y*" and "*threadIdx.z*" refer to the location of the thread in the thread block and they provide the coordinates along *X*, *Y* and *Z* dimensions, respectively. The "*blockIdx*" variable works the same way but identifies the block within the grid. In addition, "*blockDim*" provides the dimensions of the blocks.

5.4.2 Basic Stereo on GPU

Before SNCE estimation on a GPU, it is important to come up with a stereo disparity estimation algorithm which is suitable for GPU architectures. The following pseudo code of a CUDA based kernel outlines an algorithm which can be used to obtain a disparity map from a pair of rectified stereo images on a GPU.

ALGORITHM 06*: Computing Stereo Disparity on GPU.*

//Calculate a disparity map given a pair of rectified grayscale stereo images
//Input: Maximum disparity (dmax), mask size $[k \times k]$,
//Stereo image pair $Img_1[h \times (w + dmax)]$ *(padded with zeros),* $Img_2[h \times w]$
//Output: $disparity\_map[h \times w]$

```
01 : disparity ← zeros[h × w]
02 : minval ← zeros[h × w]
03 : i ← blockIdx.y * blockDim.y + threadIdx.y
04 : j ← blockIdx.x * blockDim.x + threadIdx.x
05 : if k < i < h − k and k < j < w − k:
06 :     for d ← 0 to (dmax − 1) do
07 :         mc ← matchingCost(i, j, d):
08 :         if d eq 0:
09 :             minval(i, j) ← mc
10 :         else:
11 :             if mc < minval(i, j):
12 :                 minval(i, j) ← mc
13 :                 disparity(i, j) ← d
14 : return disparity
```

In the pseudo code provided above, the segment with the yellow background shows the use of kernel variables to uniquely identify a thread in the grid and block architecture so that the tasks can be assigned to each. The segment with a blue background shows the tasks carried out by each thread which are repeated at each

disparity (due to the *"for"* loop). The "matchingCost" function refers to any calculation that involves a computation of a similarity measure with a mask size of (k x k). The most important difference between the GPU and CPU implementations is the simplicity. Due to the parallel architecture, the stereo computations have been reduced to the correspondence search at each pixel by a thread in the GPU. As a result, the order of growth of computations of an algorithm depends only on the order of growth of the computations carried out by each thread and the number of threads. Therefore, the order of growth of computations of the given GPU-based stereo example should remain within $O(n^2 d\ k^2)$ in big-oh notation.

5.4.3 Incorporating SNCE Calculations

Like in the case of SNCE calculation for SAD-based stereo outlined earlier, SNCE can be included as part of the GPU-based implementation as well. This can be demonstrated with only a few changes to the pseudo code of the GPU-based stereo example in the previous section. The following pseudo code segment shows a GPU-based stereo algorithm which is able to compute SNCE at each disparity.

ALGORITHM 07*: Computing Stereo Disparity with SNCE on GPU.*

//Calculate a disparity map given a pair of rectified grayscale stereo images
//Input: Maximum disparity (dmax), mask size $[k \times k]$,
//Stereo image pair $Img_1[h \times (w + dmax)]$ *(padded with zeros),* $Img_2[h \times w]$
//Output: $disparity\_map[h \times w]$

```
01 : disparity ← zeros[h × w]
02 : minval ← zeros[h × w]
03 : snce ← zeros[dmax]
04 : i ← blockIdx.y * blockDim.y + threadIdx.y
05 : j ← blockIdx.x * blockDim.x + threadIdx.x
06 : if k < i < h − k and k < j < w − k:
07 :     for d ← 0 to (dmax − 1) do
08 :         mc ← matchingCost(i, j, d):
09 :         if d eq 0:
10 :             minval(i, j) ← mc
11 :             atomicAdd(snce[d], 1)
12 :         else:
13 :             if mc < minval(i, j):
14 :                 minval(i, j) ← mc
15 :                 disparity(i, j) ← d
16 :                 atomicAdd(snce[d], 1)
17 : return disparity
```

The changes from the previous section include the addition of a 1D array called "snce" (shown with a yellow background) to hold the SNCE results. Each element in the "snce"

array corresponds to the SNCE values at each disparity. The additional "*AtomicAdd*" operations (highlighted in green) are used to provide a thread safe method to increment each value in the "snce" array.

5.4.4 Atomicadd Operation

According to the pseudo code above, each thread in a GPU that computes some stereo matching score at a certain disparity, needs to be able to increment the corresponding element in the "snce" array if the current matching score is the minimum score found so far (up to the current disparity). However, a large number of threads attempting to update the same element can lead to what is termed as a "race-condition". This can result in the updates to the elements being inconsistent. Therefore, the example implementation uses the "Atomicadd" operation provided by the CUDA platform which ensures that multiple threads are allowed to update the respective elements in the "snce" array without causing inconsistencies.

5.4.5 Order of Growth of Computations

Like in the example of CPU based stereo method, the order of growth of computations in the parallel stereo calculation method with SNCE remains at $O(n^2 d\ k^2)$ even after the introduction of SNCE provided that the "AtomicAdd" operations do not lead to increase in the order of growth. In the next section, empirical methods are used to experimentally verify the same.

5.4.6 Methodology

Using the same dataset in Section 1.4.1, a GPU-based stereo vision technique was used to derive disparity maps while using the GPU thread configurations shown in the Table 5.4. The maximum disparity was set to be 280 pixels at full resolution and progressively reduced by half, every time the resolution is reduced. Time to generate the disparity map was recorded 5 times at each resolution and an average value was taken. The same steps were repeated with the SNCE-enabled version of the algorithm to record execution times at each resolution level. An NVIDIA RTX2080 GPU with 8 GB memory and 2944 CUDA cores, was used as the hardware platform for the experiment. Please refer to *Appendix B* to locate source code and the software used.

Resolution	*Maximum Disparity*	*Block*	*Grid*
2964 x 2000	280	16 x 16 x 1	186 x 125 x 1
1482 x 1000	140	16 x 16 x 1	93 x 63 x 1
741 x 500	70	16 x 16 x 1	47 x 32 x 1
371 x 250	35	16 x 16 x 1	24 x 16 x 1

Table 5.4: Allocation of thread blocks at each resolution. Block configuration is kept constant and the grid configuration is changed to accommodate sufficient threads to cover the resolution.

5.4.7 Results

5.4.7.1 SAD-based Stereo on GPU without SNCE

Table 5.5 lists the observed algorithm-execution-time when the GPU-based stereo algorithm was used 5 times at each resolution. As seen from the table, the average time at quarter resolution is close to 8 times the value at half-of-quarter resolution. The last two rows confirm that the same observation stands for other resolution levels as well indicating that the order of growth of the computational time remains approximately the same even when the SAD-based stereo algorithm is implemented on GPU.

Resolution	Execution Time in Seconds (s)			
	Full (2964x2000)	*Half (1482x1000)*	*Quarter (741x500)*	*Half of Quarter (370x250)*
Take 1	2.0673	0.2217	0.0281	0.0038
Take 2	2.1485	0.2280	0.0305	0.0039
Take 3	2.0738	0.2362	0.0270	0.0043
Take 4	2.1004	0.2217	0.0308	0.0043
Take 5	2.0765	0.2221	0.0274	0.0043
Average	2.0933	0.2259	0.0288	0.0041
X8		1.8076	0.2302	0.0330

Table 5.5: Execution times for SAD-based stereo disparity estimation algorithm on GPU using Middlebury 2014 bike image pair at 4 different resolution levels (All times are given in seconds). The arrows indicate how the execution times have reduced by approximately 8 times when width, height and disparity were reduced by half.

5.4.7.2 SAD-based Stereo on GPU with SNCE

Table 5.6 presents the results for the SAD based stereo vision algorithm on GPU with embedded SNCE computations. Like the previous example, the experiments have been run for 5 times and the individual values as well as average values are shown in table. The last row contains the execution times for half, quarter and half-of-quarter resolutions, multiplied by 8.

Resolution	Execution Time in Seconds (s)			
	Full (2964x2000)	*Half (1482x1000)*	*Quarter (741x500)*	*Half of Quarter (370x250)*
Take 1	2.6784	0.3014	0.0369	0.0048
Take 2	2.7530	0.3042	0.0357	0.0047
Take 3	2.7524	0.3020	0.0393	0.0052
Take 4	2.6955	0.3138	0.0351	0.0047
Take 5	2.7179	0.2958	0.0350	0.0046
Average	2.7195	0.3035	0.0364	0.0048
X8		2.4651	0.3081	0.0385

Table 5.6: Execution times for SAD-based stereo disparity estimation algorithm with SNCE on GPU using Middlebury 2014 bike image pair at 4 different resolution levels (All times are given in seconds). The arrows indicate how the execution times have reduced by approximately 8 times when width, height and disparity were reduced by half.

As depicted by the connection arrows, average execution times in consecutive columns have increased by approximately 8 times which indicate that the order of growth of the approximate computational times are consistent with $O(n^2 d\ k^2)$.

5.5 Computational Time with and without SNCE

The next table of results (Table 5.7) contains the average execution times for the SAD-based stereo disparity estimation algorithm on GPU (before and after embedding SNCE). It is important to note that the values are significantly smaller at lower resolution. For example, at half-of-quarter resolution, the disparity estimation has only taken approximately 4 milliseconds. Even at the highest resolution of 2964x2000, the average time to compute the disparity map with a maximum disparity of 280 pixels has only taken approximately 3 seconds which is a significant improvement compared to the results on CPU (Table 5.3).

To analyse the results in the table further, it is important to understand the cost volume resolution in deep stereo networks. For example, even the top performing algorithms [6], [7], [76] on stereo benchmarks usually build cost volumes at 1/4th of the original image resolution. Therefore, a cost volume resolution of 741x500 translates to an input image resolution of 2964x2000. According to the table, SNCE estimation at 741x500 has resulted in an additional delay of only 7.6 milliseconds. In fact, even the lowest cost volume resolution level used for this experiment (370x250) is higher than the quarter-resolution of images from some stereo benchmarks. For example, the KITTI 2015 dataset has a resolution of 1242x375 which corresponds to a

cost volume resolution of 314x94 whereas in the Scene Flow dataset the same corresponds to 240x135 (i.e., due to the original image resolution of 960x540). According to the table, at a maximum disparity of 280, the additional delay introduced by SNCE when processing such datasets with an SAD based algorithm on a GPU would be lower than 0.7 milliseconds.

	Execution Time in Seconds (s)			
Cost Volume Resolution	*Full (2964x2000)*	*Half (1482x1000)*	*Quarter (741x500)*	*Half of Quarter (370x250)*
SAD Stereo Only	2.0933	0.2259	0.0288	0.0041
SAD Stereo with SNCE	2.7195	0.3035	0.0364	0.0048
Increase in Time (seconds)	0.6262	0.0776	0.0076	0.0007

Table 5.7: A comparison of the execution times (on GPU) for the basic SAD-based stereo with and without SNCE. The last row indicates the net increase in computational time due to SNCE.

Chapter Summary

The study outlined in Chapter 5 started with an algorithmic analysis of the SNCE metric to identify its computational complexity in terms of the order of growth of the computations. When analysing the pseudo-code, it was observed that even the basic stereo algorithms already contain most of the computations required by the SNCE metric. The analysis also showed that the SNCE metric can be efficiently embedded into the existing stereo algorithms without changing their order of growth in computations. An empirical analysis using a C++ based implementation of a basic stereo algorithm (with the ability to compute the SNCE metric at every disparity), confirmed the theoretical findings. The study was then extended to parallel processing (GPU) environments with a CUDA based implementation. The results obtained using an NVIDIA RTX2080 GPU confirmed that the order of growth remains unchanged as expected. However, a significantly faster performance was achieved with very little added-latency due to the SNCE metric (at quarter resolution levels of the datasets such as Middlebury, KITTI and Scene Flow). Since the SNCE metric is all about extrema movements, the findings of this chapter should equally apply to extrema movements in a cost volume of a deep stereo network.

CHAPTER 06 - SNCE CONVERGENCE IN DEEP STEREO

Introduction

The last chapter included experiments which demonstrated the feasibility of estimating the SNCE metric as part of a rudimentary stereo algorithm on sequential and parallel computing architectures. The results indicated that the performance achieved on consumer hardware is acceptable for deep learning-based algorithms. The next step is to leverage on the unimodal cost distributions of a deep stereo network to achieve SNCE convergence, so that the maximum disparity for a scene can be detected during the forward propagation phase of the network.

This chapter starts by outlining the architecture of the proposed network detailing the key components and their functionality. The training process is outlined next, with emphasis on data selection, data preparation, hyper-parameter selection and monitoring of progress. An evaluation conducted with the weights and biases saved at different stages of the training process follows, which is aimed at finding out whether the accuracy of the SNCE-based maximum disparity estimations improves with training. Finally, the robustness of the method is validated using tests on stereo image sequences.

6.1 Network Architecture

The overall network architecture used for the experiments consists of four stages: feature extraction, cost volume creation, disparity regression and up-sampling. A multi-layer 2D convolutional neural network is used to extract features from the input stereo images after down-sampling them into a lower resolution ($1/4^{th}$ of the original). A cost volume creation module (CV Module) then concatenates the features and aggregates them using another multilayer 2D convolutional block to produce a 3D cost volume. Subsequently, a disparity regression layer extracts suitable disparities which are then up sampled by an up-sampling layer to produce a disparity map at original image resolution. The overall architecture of the deep stereo model used in this chapter is shown in **Figure 6.1**. The Python source code (PyTorch [106] based) for the individual components and the complete network is included in *Appendix C*.

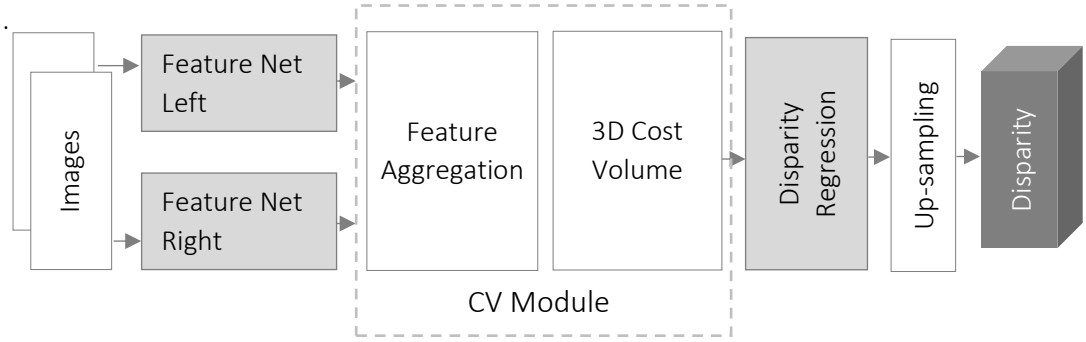


Figure 6.1: Architecture of the deep learning network used for the experiments

As outlined in Chapter 2, modern deep stereo networks which follow the "3D regularization structure", commonly use **3D convolutional costs aggregation layers** to aggregate the matching costs further. However, such further aggregation is omitted in this chapter to focus on the cost volume (and the cost extrema movements) which is central to the SNCE metric and its convergence.

6.1.1 Feature Network

The feature extraction network (referred to as "Feature Net" in the diagram) is built from four different types of 2D convolutional neural network (CNN) blocks shown in **Figure 6.2**. It starts with an input layer, which includes a 2D convolutional layer with a stride size of 4 which is used to reduce the resolution of the grayscale input images to $1/4^{th}$ of the original image size. This helps ensure that the computational time and space requirements of the network can be maintained within the practical limits.

The input layer is then connected to a series of network blocks (called "Type A" blocks in **Figure 6.2**). Each forward path in "Type A" blocks has a combination of 2D convolutional (with different kernels sizes – 3x3, 5x5 and 7x7), batch-normalization and ReLU layers. Two of the forward paths use dilated convolution with larger kernel sizes to enlarge the receptive field of the network. The residual connection makes it possible to connect several blocks in series without hindering gradient propagation while also allowing predictions of upstream layers to be passed on to the downstream layers. The output of the seven "Type A" blocks is then connected to a set of 7 "Type B" blocks connected in series. "Type B" blocks contain only two parallel paths with only one of them having dilation enabled. They also have a residual connection allowing more blocks to be connected to create a much deeper network. The final output of

the feature network is produced by the “output block” which is a single 2D convolutional layer. The notable difference in the output block from the other blocks is the exclusion of batch normalization and ReLU layers to aid in pre-scaling of the matching costs in the quest for achieving unimodal cost distributions. As found in Chapter 4, unimodality is the main requirement for deterministic convergence of the SNCE metric.

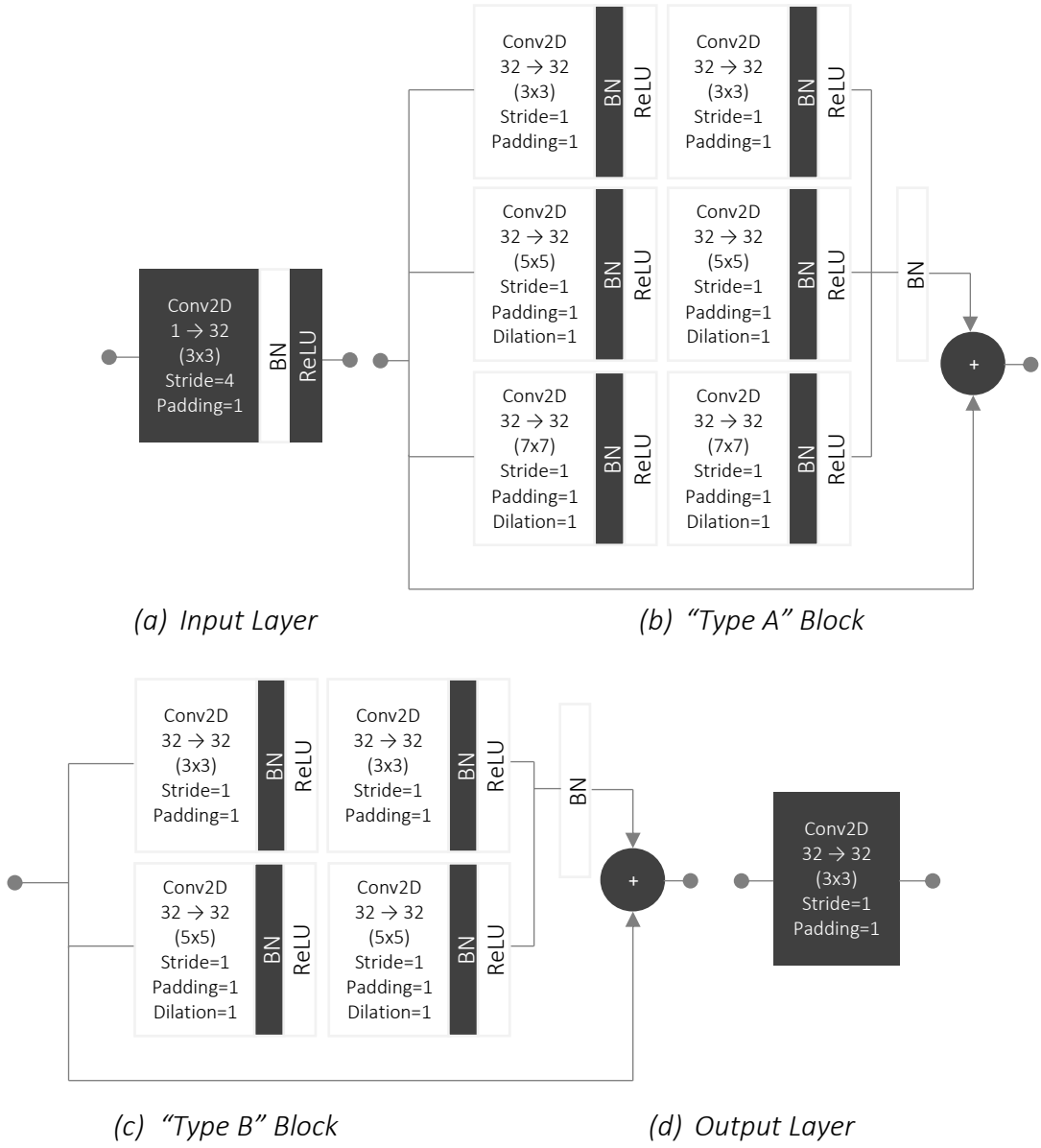


Figure 6.2: Different types of CNN blocks used to build the "Feature Net" of the deep learning network used for experiments in this chapter

All input and output feature lengths (channels) are set to 32 except in the case of the input layer which uses an input feature length of 1 for the grayscale input images. "Type A" or "Type B" blocks do not change the dimensionality of the input tensors. The overall architecture of the feature extraction network is shown in Figure 6.3.

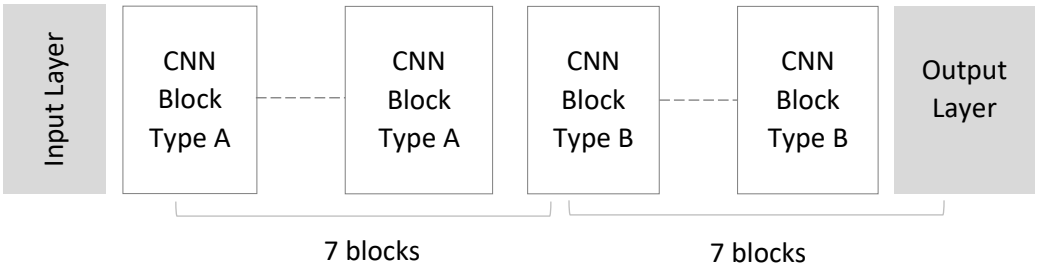


Figure 6.3: Overall architecture of the Feature Network

6.1.2 Cost Volume Module (CV Module)

The main task of the "CV Module" is to build a cost volume using the output feature volumes provided by the "Feature Network" and compute the SNCE metric at every layer. However, only a fixed-sized cost volume is built during the study outlined in this chapter. (i.e., before estimating the maximum disparity automatically via a layer-wise cost volume creation process, it is important to empirically verify whether a deep learning technique can produce unimodal cost distributions which are essential for deterministic SNCE convergence). As observed during the literature review, many deep stereo networks use cost volumes by concatenating features on a per pixel basis at each disparity which results in a 4D cost volume. However, as shown in Chapter 4, the SNCE metric is calculated on singular valued matching costs so that the extrema can be traced at each layer. Therefore, a CNN-based feature aggregation network (shown in Figure 6.4) is used to learn to predict a single valued representation from a pair of concatenated feature vectors, which can then be stored in a 3D cost volume.

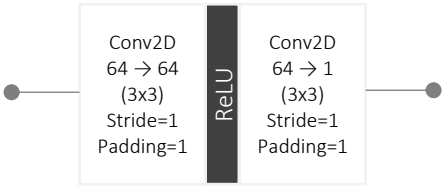


Figure 6.4: The feature aggregation network used by the cost volume module to produce single valued matching costs for the 3D cost volume. The network accepts a concatenated feature vector with a length of 64 and uses a combination of 2D convolutional and ReLU layers to output a single valued cost for each pixel at each disparity.

Figure 6.5 provides a graphical illustration of how two feature vectors (associated with two pixels being matched) are concatenated and then aggregated by the aggregation CNN to produce a single valued matching cost in the cost volume.

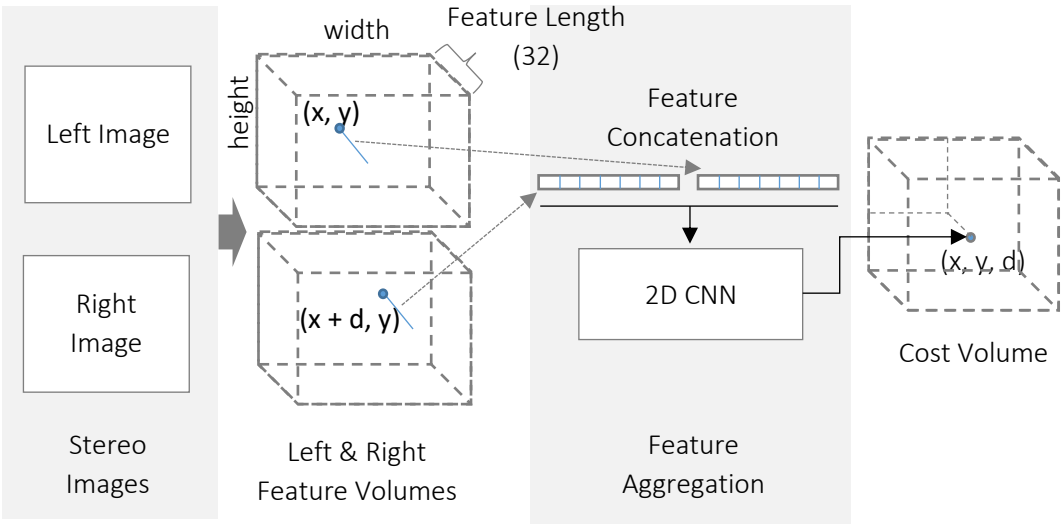


Figure 6.5: A graphical illustration showing how a 2D CNN is used to derive a single valued matching cost at point (x, y, d) in the cost volume by using the two feature vectors associated with the two points{ (x, y) & (x + d, y) } on stereo images with a mutual displacement of (d) pixels along the horizontal epipolar lines.

6.1.3 Regression Module

In the proposed network, disparity regression module carries out the task of regressing disparities from the CV module output. Due to being non-differentiable, the conventional approach of getting the index of the matching cost extrema (referred to as the "argmin" operation in literature) is not suitable for gradient propagation in a deep learning network. However, a gradient friendly, differentiable equivalent can be composed by using the differentiable "softargmin" operation to extract a matching disparity from a series of matching scores [64]. The "softargmin" operation on a cost distribution C is given by the following Equation (6.1).

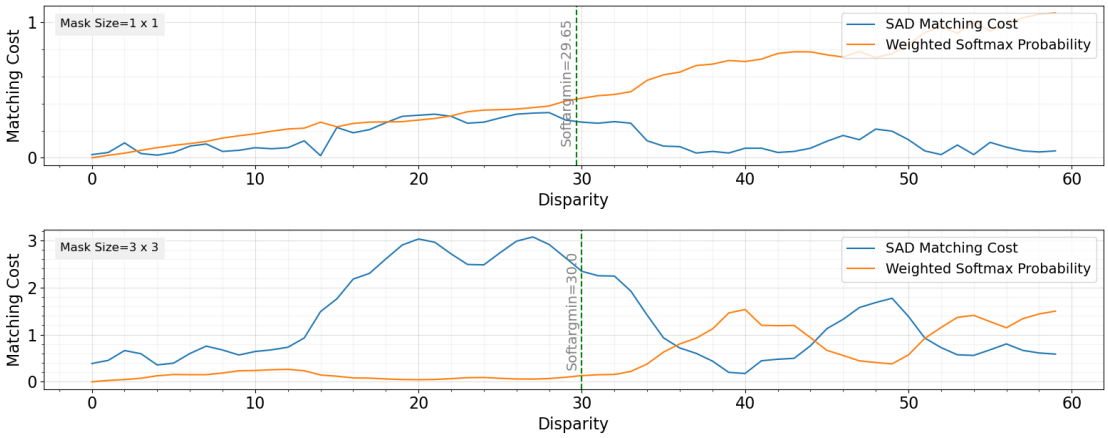
$$Softargmin = \sum_{d=0}^{dmax} d \times softmax(-C^d) \tag{6.1}$$

(d=disparity, dmax = maximum disparity and $-C^d$=negative value of the matching score at d^{th} disparity)

6.1.3.1 Convergence of Softargmin Operation

Although the softargmin operation is differentiable, it is vulnerable to multi-modal matching cost distributions under certain conditions. Thus, the cost distributions must be unimodal for the softargmin operation to be feasible for use in deep stereo networks. As an example, **Figure 6.6** shows six SAD-based matching cost distributions associated with a random pixel location with the coordinates (157,89) on "Middlebury 2003 Cones" stereo image pair. The costs have been calculated up to a maximum disparity of 60 pixels by changing the aggregation mask size from 1x1 to 15x15. In the 6 subplots, the matching cost distributions are shown in blue while their weighted softmax probability functions are plotted in orange. The blue vertical lines mark the locations of the minima predicted by the "softargmin" function in **Equation (6.1)**.

From the figure, it is apparent that when a larger mask size is used (which makes the cost distributions more unimodal as shown in Chapter 4), the softargmin based prediction tends to align well with the actual minima. It is also important to note how scaling (i.e., values on the vertical axis progressively increasing with the mask size) has some correlation with the softargmin convergence. In fact, **Figure 6.6** shows how the scaling of matching cost values has a positive effect on the unimodality of cost distributions which in turn can improve the softargmin convergence. Therefore, both the output layer of the feature network (**Figure 6.2** (d)) and the feature aggregation network (**Figure 6.4**), do not contain "Batch-normalization" layers in order to allow the network to learn scaling during the training process.



(Figure extends to the next page)

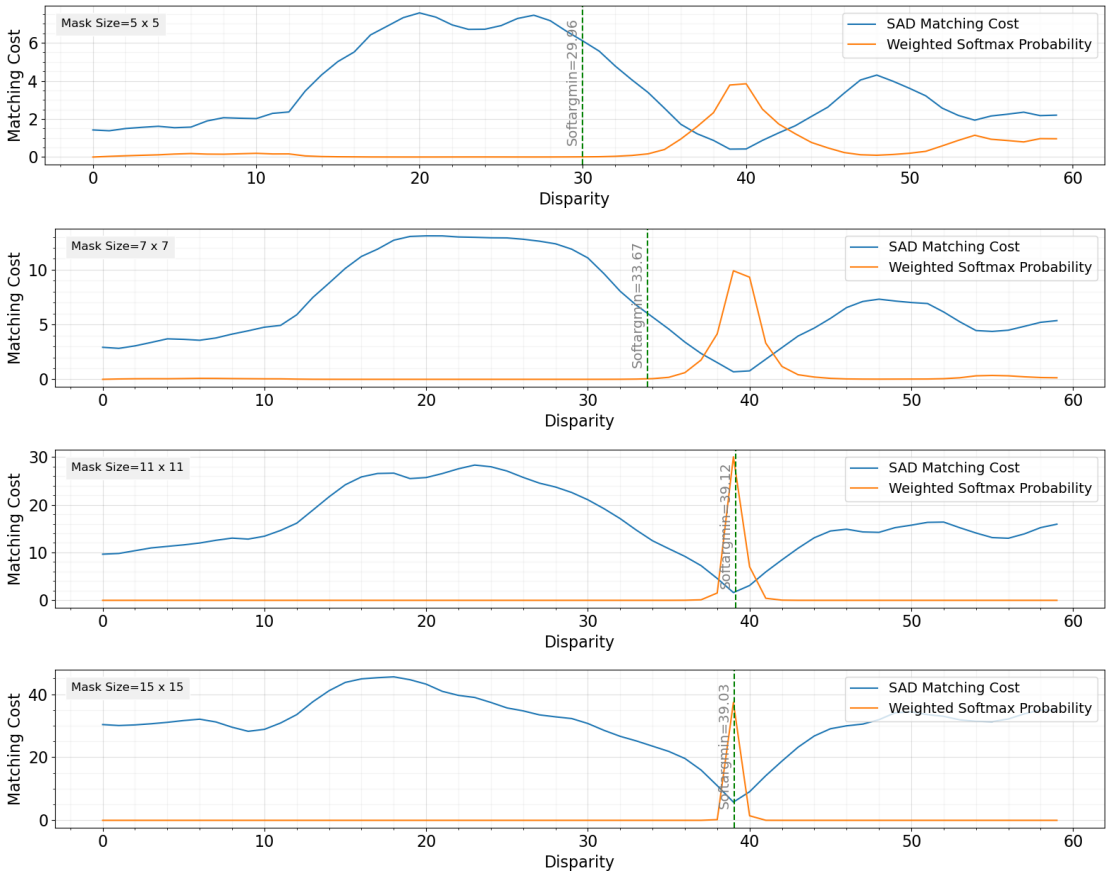


Figure 6.6: The sub-figures from top to bottom show the SAD-based matching cost variations for a random pixel from the "Middlebury 2003 Cones" image pair, when the mask size is changed from 1x1 to 15x15. The blue vertical lines show the locations of the minima predicted by the "softargmin" operation. When the mask size increases, the predictions start to align well with the actual minima.

6.1.4 Up-Sampling Module

As mentioned earlier, the input layer reduces the dimensionality of the feature network which leads to faster learning with less resources on GPU hardware. However, it also makes the output disparity resolution lower than the original resolution of the images. Therefore, an up-sampling layer is used to increase the resolution of the predicted disparity maps back to the original resolution. Due to the feature extraction network operating at $1/4^{th}$ of the image width and height, the up-sampled disparities must be multiplied by 4 to obtain the final disparities.

6.2 Data Preparation

Data preparation for training is one of the most crucial stages of the supervised-training process in deep neural networks. When fitting a neural network model to a problem, the overall success depends on how well the data is selected and suitably prepared for training. The model for the current experiment is required to learn

features from images instead of remembering the values (a phenomenon which is known as over-fitting). The training process involves the adjustment of weights and biases of the network which determine the features that the network is capable of learning. During backpropagation, these weights and biases get updated based on the gradient of the output loss. If the features in input data are not scaled consistently, then there can be discrepancies in how the weights and biases get updated in response to the network encountering similar features. That makes it harder for the network to learn. The process of re-scaling the features in the input is known as normalization. When training the basic deep stereo network outlined in this chapter, Z-score normalization (which is also known as standardization) was used to normalize the input image data.

6.2.1 Z-Score Normalization

The z-score normalization (or standardization) involves scaling of the image data so that they contain their pixel intensity values distributed with a mean value of zero and a standard deviation of one. Since the experiments are carried out using grayscale images, the features are required to be normalized based on intensities of a single channel. The following equation (Equation (6.2)) shows the relationship between the pixel intensities and their normalized values.

$$Normalized\ Intensity\ at\ Point\ (P) = \frac{\text{Pixel intensity at (P)} - \text{ Mean intensity}}{Standard\ deviation} \qquad (6.2)$$

As per the Equation (6.2), the mean and the standard deviation values must be available before the image data can be normalized. These measures can be estimated for the entire image dataset or on a per-image basis (i.e., for each image). For the training process outlined in this chapter, normalization of the grayscale images has been carried out using the mean and standard deviation computed on a per-image basis.

6.2.2 Dataset Selection

The “Scene Flow – Driving” dataset was selected for the experiment, which contains synthetic driving scenes captured at various focal lengths, directions and capture intervals. From the “Scene Flow – Driving” dataset captured at a focal length of 15 mm and a faster simulated vehicle speed, images listed under the “forward” and

"backward" categories were compiled into a single dataset with 600 image pairs. Then 80% of the images (480 image pairs) were used as the training batch whereas the remaining 120 images were reserved for validation and testing with 60 images each. Validation images were used for tracking the validation accuracy during training while the 60 test images were set aside for testing and evaluation after training.

The images in the dataset contain a rich mix of scenes which are usually challenging to any stereo vision method due to the presence of shadows, glare, reflections and very fine structures. However, such phenomena do not occur in every image pair of the dataset. Therefore, it was important to ensure that the training dataset contains a mix of scene conditions expected during inference. Otherwise, if the training dataset does not contain a particular type of scene condition, then the network would be unable produce good results when it eventually encounters unfamiliar scene conditions during testing. For example, if the training dataset did not include images with shadows, it would be hard for the network to produce good inference output for the test data with scenes containing shadows.

The Scene Flow dataset being a driving dataset, contains multiple image pairs with similar scene conditions in sequence. Therefore, every 5^{th} image pair in the forward and backward sequences was selected to be part of the testing and validation datasets while reserving all other images for training.

6.2.3 Data Augmentation

Using only 480 image pairs for training can result in the network over-fitting to the data because of the limited size of the dataset. Moreover, there is only a limited amount of variation in the maximum ground-truth disparity in Scene Flow image pairs at original image resolution. This can be seen in Figure 6.7, which shows the variation of the maximum ground-truth disparity associated with all 600 images in the dataset used for the experiments. As shown in the graph, the maximum ground-truth disparity at native resolution often remains closer to the mean value with only a few infrequent changes. However, by randomly cropping the original images into a smaller size, many different images with a wider range of maximum ground-truth disparity values can be obtained. Having a wider variation in maximum ground-truth disparity is important when verifying the convergence of the SNCE metric.

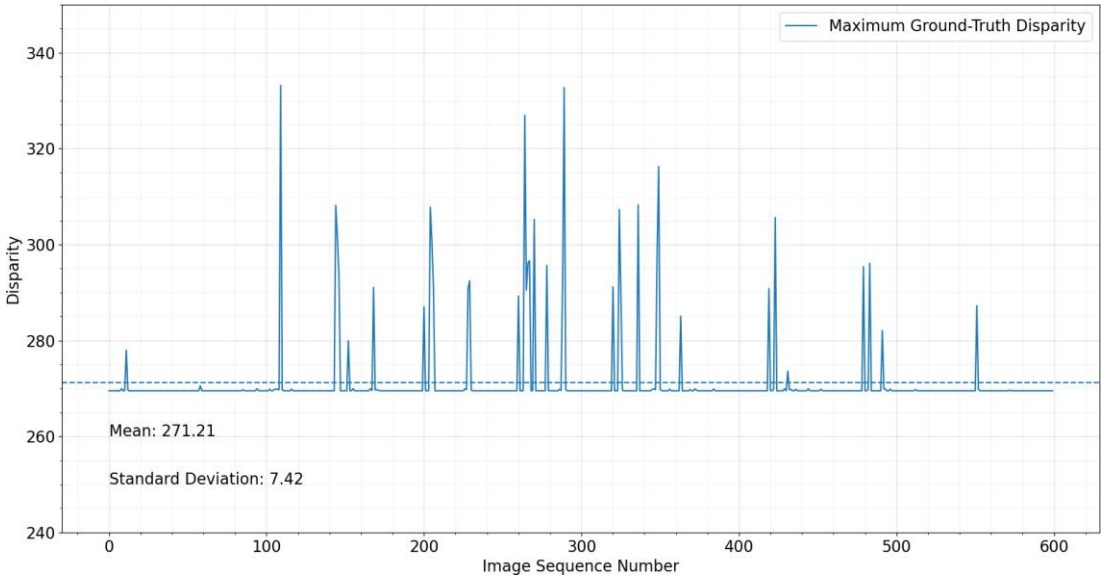


Figure 6.7: Variation of the maximum ground-truth disparity across all 600 stereo image pairs at native resolution (960x540). The standard deviation is smaller which indicates infrequent variation.

The dataset was further augmented by randomly cropping the images to a size of 800x480 pixels. Figure 6.8 depicts the variation of the maximum ground-truth disparity for the augmented dataset. As seen from the plot, the new dataset has a much better variation in maximum ground-truth disparity. This is evident from the value of the standard deviation which is 18.47 for the augmented dataset as opposed to only 7.42 at native resolution.

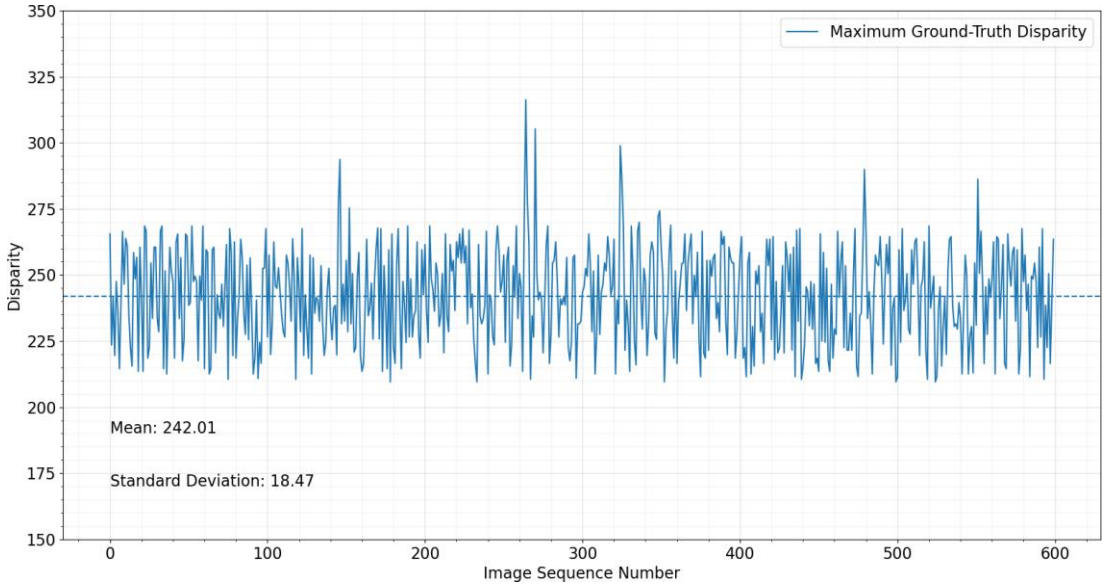


Figure 6.8: Distribution of the maximum ground-truth disparities across the dataset (600 images) with random cropping to 800x480. Values vary frequently over a larger range.

6.3 Training

When training a neural network, it is required to estimate how much the network output is different from the expected output which is the purpose of having a loss function. The proposed network is expected to produce a 2D disparity map from a stereo image pair which needs to be compared against the ground-truth disparity map for the same scene. Due to being robust against the outliers, "SmoothL1 Loss" which is also known as the "Huber Loss" was selected as the loss function which is given by the following equation.

$$loss(x, y) = \begin{cases} 0.5(x-y)^2/\beta, & if\ |x-y| < \beta \\ |x-y| - 0.5\beta, & otherwise \end{cases} \quad (6.3)$$

In Equation (6.3), the predicted output is given by x and the expected output is given by y. β is typically set to 1. According to the equation, SmoothL1 loss behaves very similar to the L1 loss (which is the absolute difference between the predicted and expected values) when the value of L1 loss is more than β. However, when the L1 loss is less than β, the Huber loss takes the form of L2 loss which is the squared difference between the values. The objective is to prevent the potential exploding gradients due to larger discrepancies between the predicted and expected values.

6.3.1 Learning Rate and Optimizer

The learning rate determines the extent to which the weights and biases are updated when the gradient of the loss is backpropagated through the network. Usually the learning rate or the "step size" is set within an optimizer which is the actual algorithm that updates the weights based on the gradient of the loss. For the current experiment, the learning rate was set to 0.0001 with other default settings (Adam *beta parameters= [0.9, 0.999], weight decay=0*) on Adam optimizer [107]).

6.3.2 Validation and Monitoring of Progress

The disparity dimension of the cost volume module was fixed[5] at 96 which translates to 384 pixels at original image resolution. An NVIDIA GTX1070 GPU with 8 GB on board memory was used as the hardware platform for training. The network model was

[5] The reliability of the SNCE metric at predicting a suitable candidate maximum disparity is yet to be verified at this point. That is the main purpose of the current chapter. Therefore, as outlined in the methodology in Chapter 3, a fixed-sized cost volume is used for both training and evaluation process explained in the current chapter.

trained with the augmented dataset for 500 training cycles (epochs). At the end of each cycle, the model parameters were saved for later analysis. The images from the validation dataset were used to calculate the validation loss at the end of each epoch and the values were recorded to track the progress. It was observed that the validation loss continued to decrease reaching a value of around 2.80 at 500 epochs. Figure 6.9 shows the variation of the validation loss with the number of training cycles.

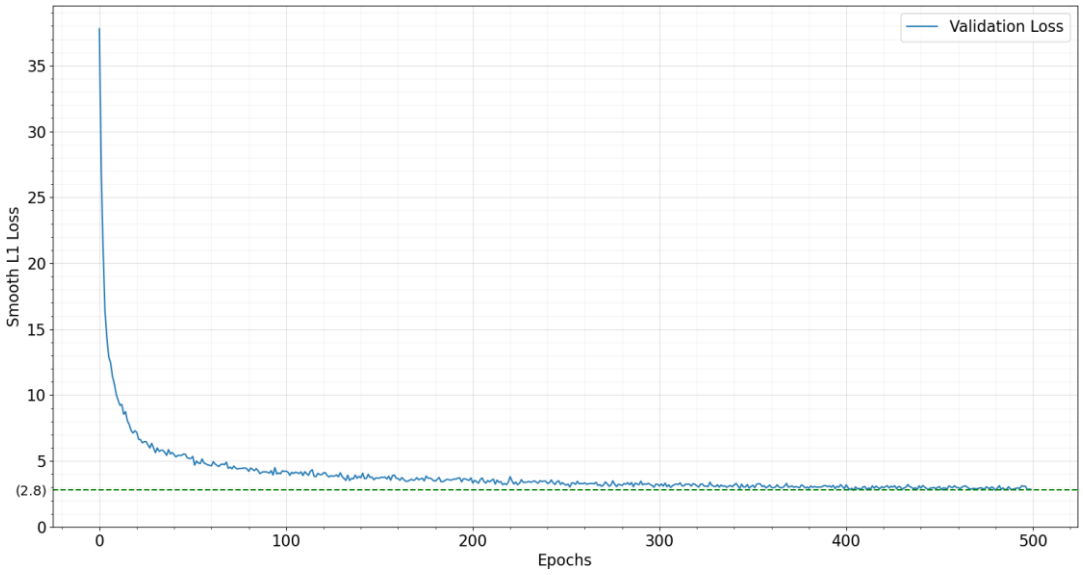


Figure 6.9: Variation of the validation loss during training. Validation loss continued to decrease reaching a value of 2.80 at 500 epochs.

6.4 Results and Evaluation

After training for 500 epochs, the trained network was used to derive disparity maps for 60 cropped stereo image pairs (i.e., with a crop size of 800x480 pixels) from 60 test images. Some sample disparity maps are given in Figure 6.10 which contains the results for 5 stereo image pairs. In the figure, grayscale left images are shown in the leftmost column for reference. The ground-truth disparity maps are shown in the middle while the predicted disparity maps are shown in the rightmost column.

6.4.1 Qualitative Analysis

A quick visual inspection of Figure 6.10 reveals that the network has managed to closely predict disparity maps despite challenging conditions such as shadows, glare and occluded areas. The relatively smoother disparity variation around the leftmost edges of the predicted maps indicates that the model has been able to reasonably regularize the regions which are located outside the stereo area. However, when compared to the ground-truth disparity maps, there are some instances of missing details, blurry

edges and erroneous predictions. The visually identifiable obvious errors are marked on the disparity maps themselves.

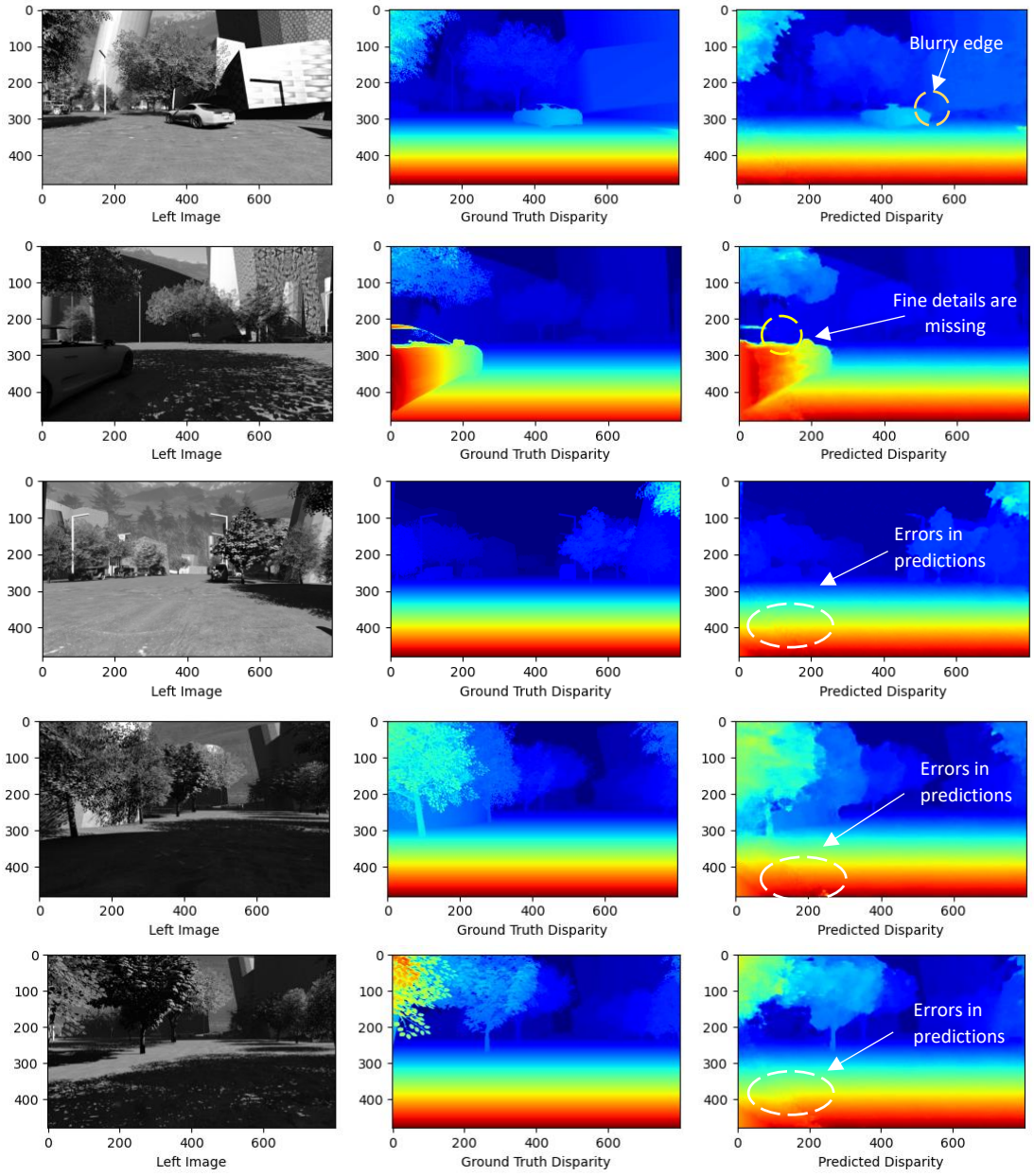


Figure 6.10: Sample disparity predictions using the described network. Grayscale left images have been provided as a reference on the left-most column. The ground-truth disparity maps are shown in the middle with the predicted disparity maps on the right. The colour spectrum used for the disparity maps is given in ***Figure 6.11****.*

Figure 6.11: Colour spectrum used for disparity maps included in this thesis. The red colour indicates the object points in the foreground of the scene whereas the blue colour pixels indicate the object points in the background. The other colours indicate the object points in between.

6.4.2 Evaluation Methodology

The model parameters saved after 1 training cycle were loaded into the network and the CV module was configured to use a maximum disparity value of 96. Then a randomly selected stereo image pair (Figure 6.12) from the Scene Flow "Driving" test dataset was used as input to the network and the SNCE values for the disparity range from 0 to 96 were extracted from the CV module. The process was repeated with model parameters saved after 2, 5, 10, 50, 100 and 500 cycles to obtain 6 more SNCE profiles. All seven SNCE profiles were plotted on the same graph along with the maximum ground-truth disparity for the scene in the stereo area.

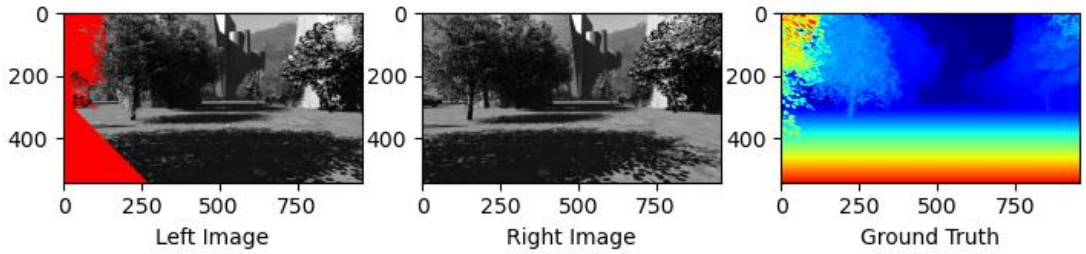


Figure 6.12: The selected stereo test image pair (with ground-truth) from the Scene Flow "Driving" dataset. The area shaded in red on the left image shows the pixels which have their corresponding pixels located outside of the image boundaries of the right image (i.e., according to the left-ground-truth-disparity map). The objective here is to extract a maximum disparity value for the stereo area in the scene. Therefore, when obtaining the maximum disparity for comparison, disparities in the (red) shaded area have been excluded.

6.4.3 Evaluation - SNCE Convergence

Figure 6.13 shows the series of seven SNCE plots obtained with network models saved after 1, 2, 5, 10, 50, 100 and 500 training cycles. It is important to note that the figure contains SNCE variations at cost volume resolution (i.e., $1/4^{th}$ of the width and height of the original image). The maximum ground-truth disparity for the scene in the stereo area was found to be 269.5 pixels which corresponds to approximately 67 pixels at cost volume resolution. It is marked with a dashed vertical line with the label "Max GT Disparity". According to the graph, it appears as if all SNCE variation profiles have reached some smaller value at around the maximum ground-truth disparity. However, the scale of the vertical axis makes the smaller values appear to be closer to zero which can be misleading. Figure 6.14, shows an enlarged view of the same graph around the neighbourhood of the maximum ground-truth disparity which reveals that only some of the graphs have reached zero while the others have remained non-zero even after the maximum ground-truth disparity for the scene has been reached.

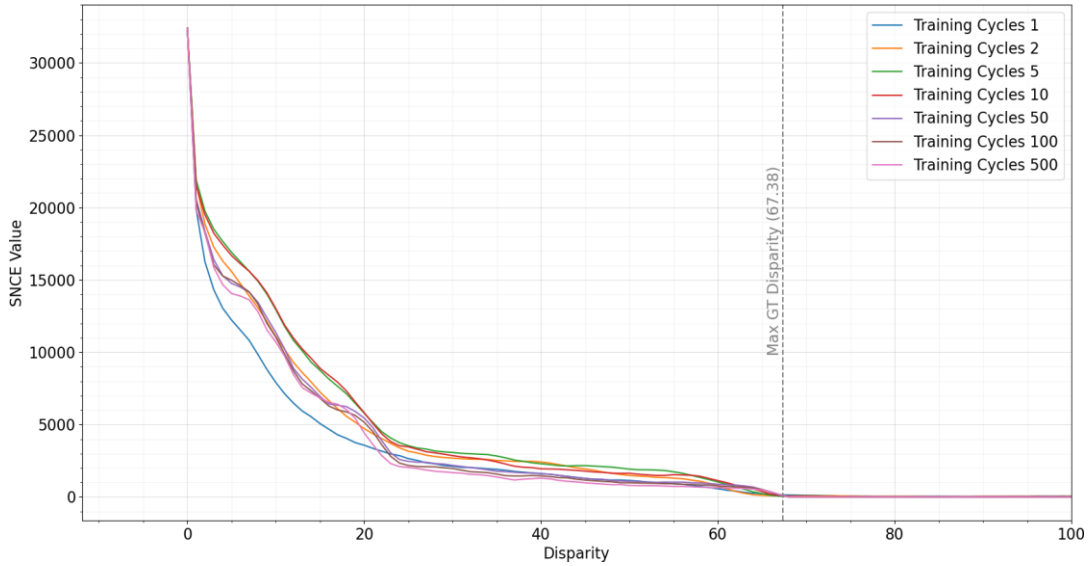


*Figure 6.13: SNCE profiles for the scene in **Figure 6.12** obtained with the network models saved after different number of training cycles (at cost volume resolution or 1/4th of the original image resolution). All SNCE variations appear to converge to some smaller value around the maximum ground-truth disparity for the stereo image pair.*

For example, the plot shown in blue (in Figure 6.14) which corresponds to the model trained for only one cycle, has not reached zero at any disparity. In contrast, the SNCE profiles obtained with the models trained for 100 and 500 training cycles have reached zero immediately after reaching the maximum ground-truth disparity and have remained constant thereafter.

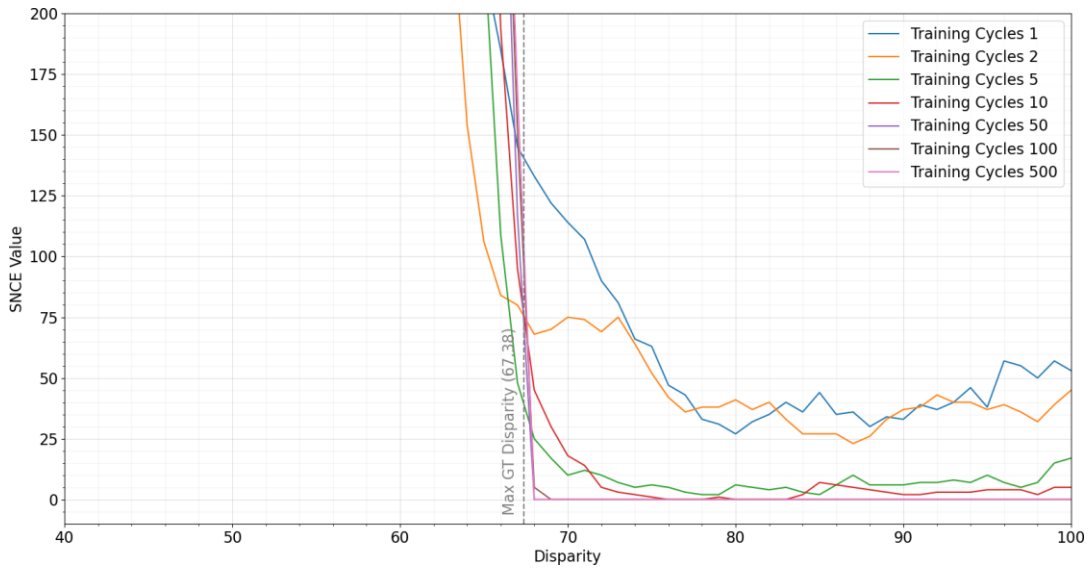


*Figure 6.14: An enlarged view of the SNCE profiles for the scene in **Figure 6.12** obtained with the network models saved after different number of training cycles at the cost volume resolution. The SNCE profiles that correspond to the well-trained networks (100, 500 epochs) show convergence to zero immediately after the maximum ground-truth disparity.*

The SNCE graphs obtained with the models trained for 100 and 500 cycles are barely distinguishable from each other in the neighbourhood of the maximum ground-truth disparity as they lie on top each other. The plot shown in purple which corresponds to the model trained for 500 epochs, provides a better view of SNCE convergence around maximum ground-truth disparity.

6.4.4 SNCE Convergence in Image Sequences

The results for the chosen test image pair suggest that the SNCE value computed using the fixed-sized cost volume converges to zero at a suitable candidate maximum disparity which is equal to or slightly above the maximum ground-truth disparity for the scene. Furthermore, the SNCE convergence appears to improve with the number of training cycles. Therefore, the next step is to verify if the findings would hold for a sequence of images with a diverse set of maximum ground-truth disparity values.

6.4.4.1 Methodology

Using seeded-random cropping, 3 sets of image sequences (each having 60 images) were created from the Scene Flow "Driving" test dataset which was reserved earlier. The deep stereo network was loaded with the parameters saved after 500 training cycles. The CV module was set to a maximum disparity of 96 at cost volume resolution which is higher than the maximum ground-truth disparity of the entire Scene Flow "Driving" dataset at quarter resolution. Using the model, SNCE profiles were obtained for all image pairs in the image sequences. The value at which each SNCE variation becomes zero was recorded as the "estimated candidate maximum disparity" value for each stereo image pair. The extracted candidate maximum disparity values were compared with the maximum disparity values observed in ground-truth data. Like the previous experiment, the maximum ground-truth disparity value for each image pair was estimated by considering the stereo area only.

6.4.5 Results on Image Sequences

The results for the first image sequence are shown in Figure 6.15. According to the figure, the plot in orange shows the candidate maximum disparity estimations for each image pair in the sequence. The maximum ground-truth disparity in the stereo area for each image pair is shown in blue. The horizontal axis indicates the indices of the corresponding image pairs in the sequence. As seen from the results, the SNCE value

has reached zero at or just above the maximum ground-truth disparity for all cropped image pairs.

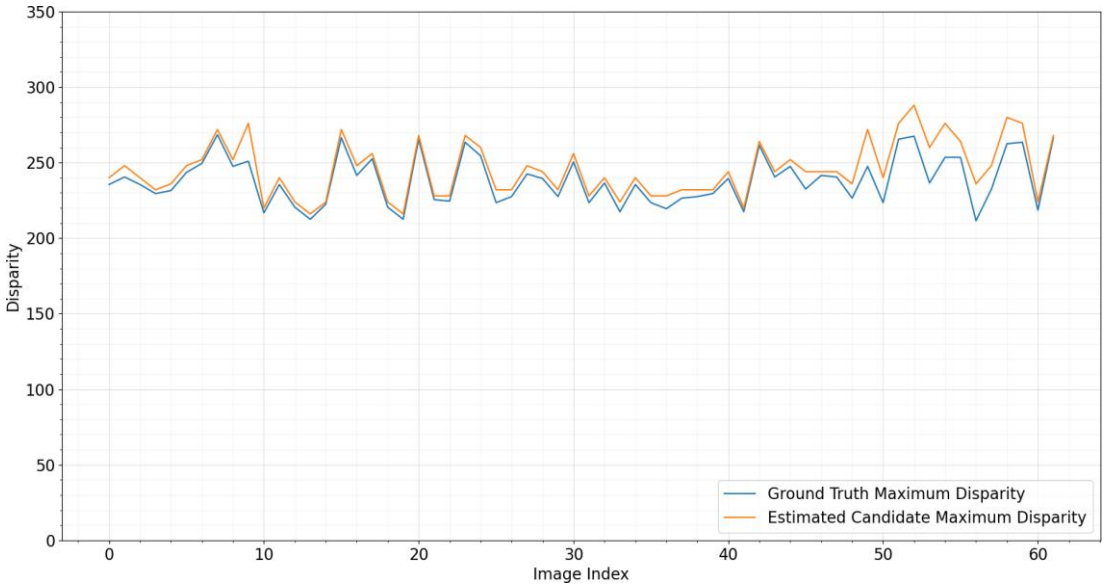


Figure 6.15: Results for the first image sequence. Estimated candidate maximum disparities align well with the maximum ground-truth disparity values reported in the stereo area of the scenes showing a stronger correlation.

The results for the second pseudo-random image sequence are shown in Figure 6.16. From the graph, it is apparent that a clear majority of the estimated candidate maximum disparity values have been either equal to or slightly higher than the maximum reported in the ground-truth, except for one instance circled in red.

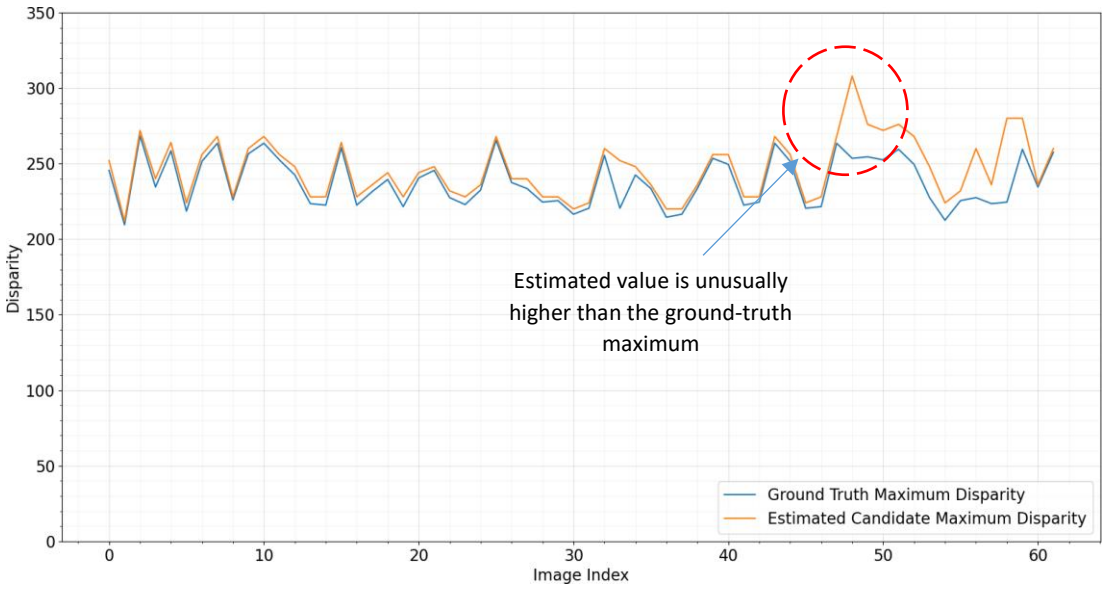


Figure 6.16: Results for the second image sequence showing that the disparities at which the SNCE value reached zero closely matches the maximum ground-truth disparity in most cases. All values are equal to or slightly higher which is acceptable. An unusually high estimation is marked with a dashed-red circle.

6.4.5.1 Unusually Higher Estimations

The results for the third image sequence are shown in **Figure 6.17**. Although the results appear to indicate a very close correlation between the maximum ground-truth disparity values and the candidate maximum disparity estimates in most cases, there is another exception similar to the results of the second image sequence. In both image sequence 2 and 3, the SNCE metric has reached zero at an unusually higher disparity for the image pairs marked with dashed red circles in **Figure 6.16** and **Figure 6.17**. Therefore, a closer inspection of the image pairs and the corresponding disparity maps is required to find the root cause for the deviations.

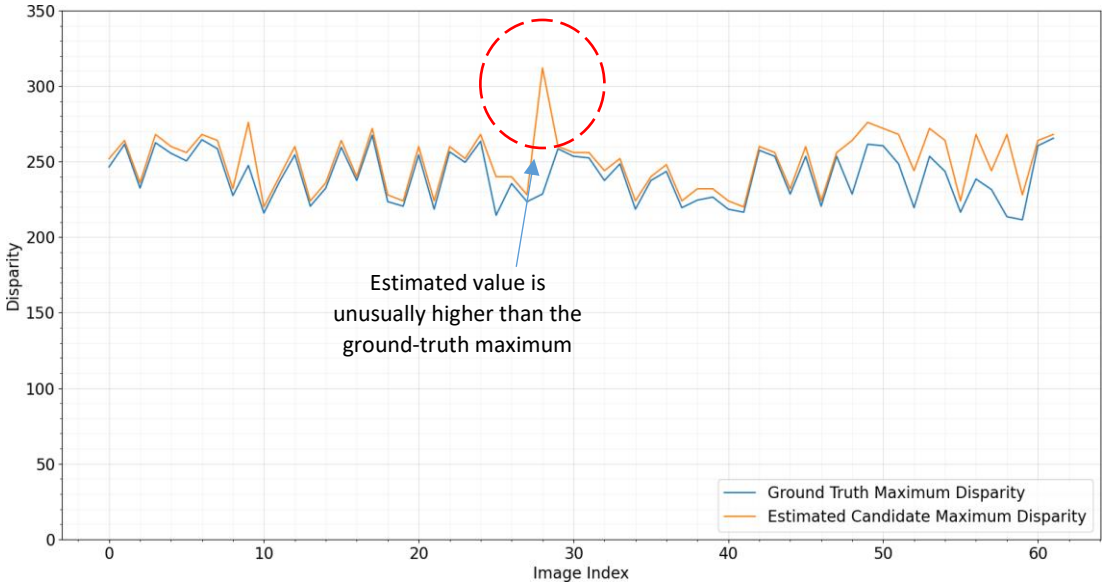


Figure 6.17: Results for the third image sequence. Most estimations align well while there is one candidate maximum disparity estimation which is unusually higher than the corresponding ground-truth maximum.

6.4.5.1.1 Methodology for Exceptions Analysis

The cropped stereo image pairs which correspond to the exceptions were located using the pseudo-random sequences. Then the located image pairs were used as input to the deep stereo network loaded with parameters saved after 500 training cycles, to produce disparity maps. The input images, output disparity maps and the corresponding ground-truth disparity maps were then inspected for any overestimation of disparity.

6.4.5.1.2 Root Cause for Unusually Higher Estimations

Figure 6.18 shows the stereo image pair, the ground-truth disparity map and the predicted disparity map for the exceptional scenario marked with a circle in **Figure 6.16**. The stereo pair is a challenging one for stereo correspondence due the existence of glare. Area in the predicted disparity map, where the estimated disparities for some pixels are larger than the maximum ground-truth disparity, is marked with a yellow rectangle[6]. The corresponding areas in the stereo image pair and the ground-truth disparity map are also marked with bounding boxes. According to the figure, the network has overestimated disparities for some pixels in the area containing glare related image artefacts. A further investigation into the exceptional scenario in the third image sequence also revealed the presence of discrepancies in the estimated disparity values in an area affected by glare as shown in **Figure 6.19**. In both scenarios, the SNCE value has been affected by the existence of cost extrema at higher disparities than the maximum values in ground-truth data, due to challenging scene conditions.

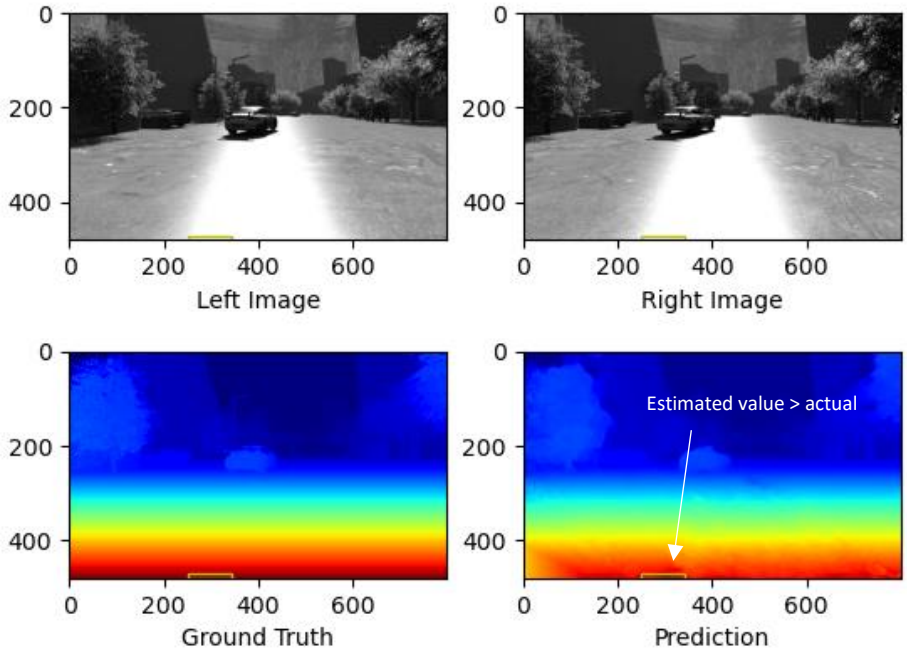


Figure 6.18: Stereo images, ground-truth disparity map and the output disparity map for the exception reported in the second image sequence. Bounding boxes enclose the pixels for which the estimated disparity is larger than the ground-truth maximum. The stereo pair is a challenging one due to glare, therefore the cost volume produced by the model has some cost extrema located at higher disparities than the maximum ground-truth disparity. This has resulted in the SNCE value reaching zero at a higher disparity than the ground-truth maximum.

[6] The yellow rectangle has been drawn based on the minimum and maximum values of the x and y coordinates of the affected pixels. Therefore, it does not mean all the pixels within the boundaries of the yellow rectangle have reported unusually high values. Only some of them have.

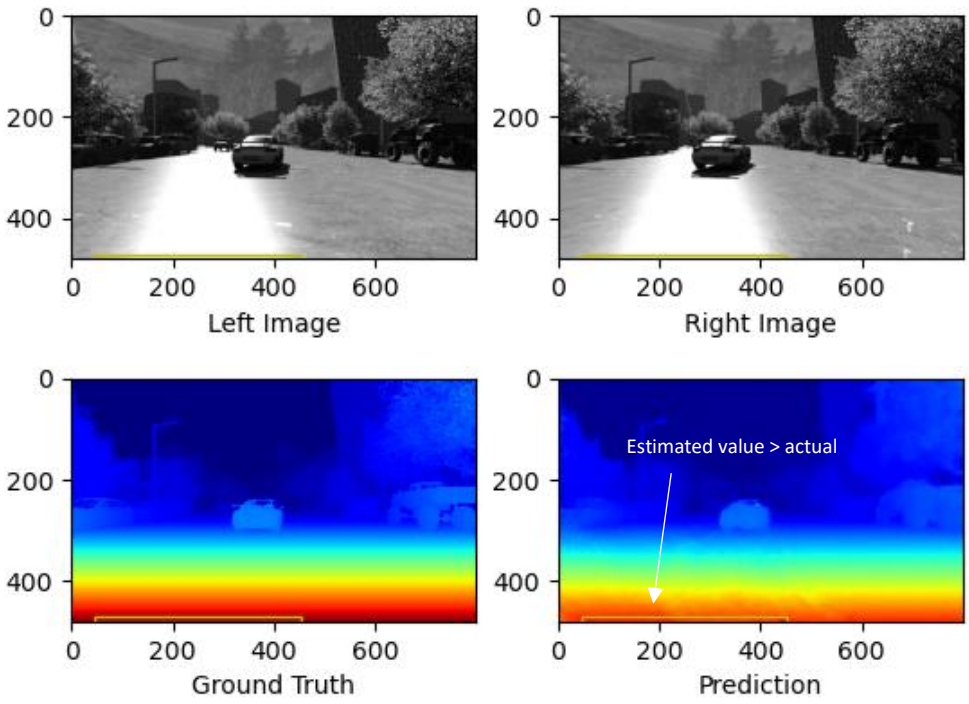


Figure 6.19: Stereo images, ground-truth disparity map and the output disparity map for the exception reported in the third image sequence. Bounding boxes enclose the pixels for which the estimated disparity is larger than the ground-truth maximum. The stereo pair again is a challenging one due to glare, therefore the cost volume produced by the model has some cost extrema located at higher disparities than the maximum ground-truth disparity. This has resulted in the SNCE value reaching zero at a higher disparity than the ground-truth maximum.

6.4.5.2 Predictions Beyond the Stereo Area

Until now, the maximum ground-truth disparity was estimated by considering the stereo area only. However, deep stereo algorithms can also estimate disparities for pixels outside the stereo area as seen from the estimated disparity maps shown earlier in Figure 6.10. However, when a pixel is outside the stereo area, two match points cannot be established in the pair of stereo images which makes the disparity unknown. The only way the algorithm can estimate a disparity value for such a pixel is by using its regularization capabilities based on the already established matches. To investigate the performance of the basic deep stereo network outside the stereo area, the 1st image sequence was re-evaluated with the maximum ground-truth disparity values estimated over the entire pixel space of the respective image pairs. Figure 6.20 shows the plots for the SNCE-based candidate maximum disparity estimations and the maximum ground-truth disparity (over the entire pixel space) for each cropped image pair in the 1st image sequence. The updated results reveal that most of the candidate maximum disparity estimations are still equal to or slightly higher than the maximum

ground-truth disparity (over the entire pixel space). This is probably due to the pixels with the highest ground-truth disparity being located in the stereo area already. However, there is one exception (marked with a dashed-red circle in Figure 6.20) in which the SNCE-based candidate maximum disparity estimation is lower than the ground-truth disparity which merits further investigation using the disparity maps.

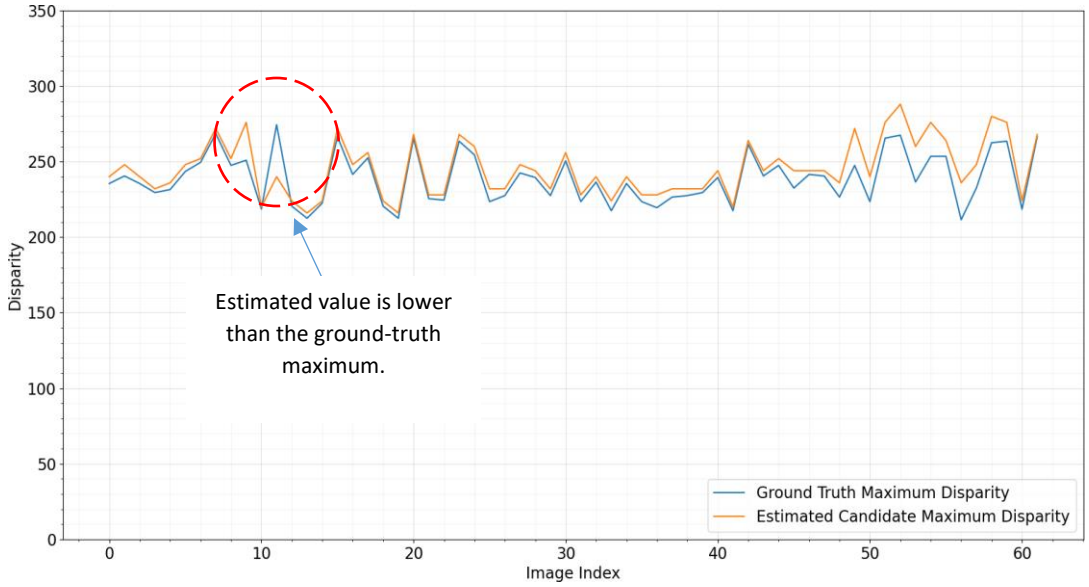


Figure 6.20: Updated results for the first image sequence with the areas outside the stereo area also considered when computing the maximum ground-truth disparity. Most estimations still align well while there is one instance (circled in red) which shows that the SNCE-based candidate maximum disparity estimation is lower than the maximum ground-truth disparity.

6.4.5.2.1 Root Cause for Lower Estimations

By following the same methodology as earlier, the stereo image pair, ground-truth disparity map and the predicted disparity map were extracted for the exceptional scenario in the updated results for the first image sequence. The results are shown in Figure 6.21. In the area marked with blue bounding boxes, the ground-truth disparity values are higher than the candidate maximum disparity value for the scene which was estimated using the SNCE metric[7]. A closer look at the left and the right images indicate that the affected area is located outside of the stereo area. As seen from the images, the part of the car marked with a yellow bounding box in the left image in Figure 6.21 is not visible in the right image. This means that the cost extrema in the cost volume

[7] This again does not mean that the ground-truth disparity is higher than the candidate maximum for all pixels within the bounding box. Only some of the pixels meet the condition. However, the bounding box has been drawn by using the maximum and the minimum values of the x and y coordinates of the corresponding pixels.

have been established at lower disparities (than the ground-truth disparity values) for the same pixel locations. This has caused premature SNCE convergence before reaching the maximum ground-truth disparity for the whole scene.

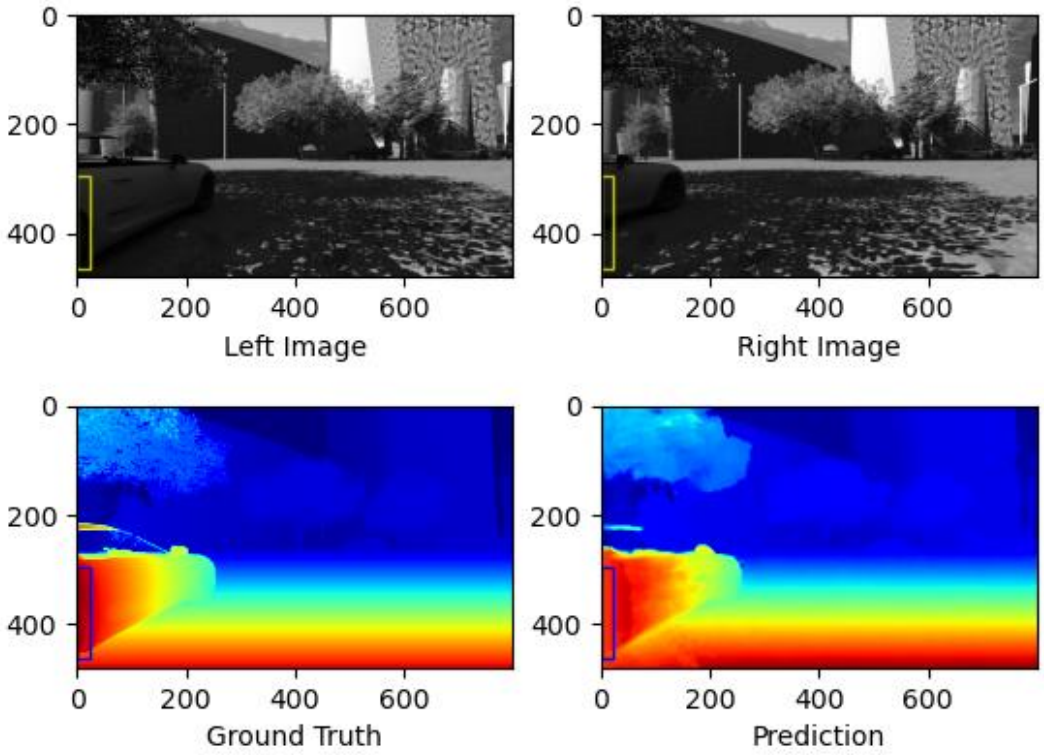


Figure 6.21: Stereo images, ground-truth disparity map and the output disparity map for the exception reported in the updated results of the first image sequence (with the area outside the stereo area included when computing the maximum ground-truth disparity). Bounding boxes enclose the pixels for which the estimated candidate maximum disparity is smaller than the ground-truth maximum. The affected pixels are outside the stereo area.

Chapter Summary

The feasibility of achieving SNCE convergence with a deep stereo network was studied in Chapter 6. A deep stereo network that can produce a 3D cost volume and estimate the SNCE metric at each disparity was developed at the beginning. The network was then trained using the Scene Flow "Driving" dataset while saving the weights and biases after each training iteration. Upon completion of the training process, the trained model was used to obtain disparity maps for sample stereo images from the test dataset. A qualitative analysis of the results indicated that the network has been able to produce disparity maps with reasonable accuracy in terms of the details captured by the disparity maps. However, it was also observed that the challenging scene conditions such as image glare can produce errors in the output.

An experiment conducted using the network parameters saved at different stages of the training process, revealed that the SNCE convergence gets better with training. It also showed that the value of the metric indeed becomes zero immediately after reaching the maximum ground-truth disparity as long as the network is sufficiently trained.

Further tests with stereo image sequences with ground-truth data indicated that the disparity at which the SNCE metric reaches zero, coincides reasonably well with the maximum ground-truth disparity. The study also showed that inaccurate stereo matching can lead to SNCE convergence at a higher disparity than the maximum reported in ground-truth data. This implies that the SNCE metric is vulnerable to the stereo matching errors. It was also observed that the SNCE metric can underestimate disparities for pixels outside the stereo area.

The basic deep stereo network presented in this chapter only used a fixed-sized cost volume. Due to the stronger correlation between the SNCE-based candidate maximum disparity estimations and the maximum ground-truth disparity in the stereo area, progressive cost volume construction with SNCE-based termination of the process is attempted next in Chapter 7.

CHAPTER 07 - AUTOMATIC DSR WITH DEEP LEARNING

Introduction

In the previous chapter, it was shown that the SNCE convergence can be achieved with the inherently unimodal cost distributions in deep stereo networks. The network used for the experiments included a feature extraction network and a basic feature aggregation module which produced a fixed-sized cost volume. The next challenge is to improve it further so that the cost volume can be built one layer at a time and the process can be terminated when the SNCE value reaches zero. Hence, this chapter introduces a novel deep stereo network called ADSR-Net (Automatic Disparity Search Range Network) which can automatically estimate a suitable maximum disparity value for a given scene, during a progressive cost volume creation stage.

The study starts by identifying the design objectives to guide the development process. The network architecture is introduced next, and it is implemented in PyTorch. A performance analysis is conducted using the untrained network to ensure that the forward propagation time of the network remains practical for training and inference. Clamped Training is introduced to improve the unimodality of the matching costs during training, in the presence of a stronger cost aggregation network.

The network is trained in a 3-stage process which involves training with stereo images from the Scene Flow “Driving”, “Flying Things 3D” and KITTI 2015 stereo datasets sequentially. At the end of the first stage, an evaluation is conducted using cropped stereo image sequences to assess the accuracy of the maximum disparity predictions. In addition, a comparative analysis is conducted with a reference implementation that uses a fixed-sized cost volume, to identify any cumulative gains in performance when using automatic maximum disparity estimation. After each training stage, the model is evaluated using the validation datasets to determine the accuracy of the disparity predictions. Upon completion of all three stages, an extended evaluation is carried out using additional datasets such as the Middlebury 2014, ETH3D and KITTI 2012 to analyse the output accuracy both quantitatively and qualitatively. The study concludes with a demonstration of the generalization capabilities and user-parameter independence of the ADSR-Net through a qualitative analysis involving a custom-built stereo camera system. Source code for the ADSR-Net is included in *Appendix C*.

7.1 Design Objectives

A clear set of objectives must be defined in advance to guide the successful development of any deep learning-based algorithm. In this study, the proposed stereo disparity estimation network is expected to achieve the following objectives.

1. Faster average inference times when processing stereo image sequences

   The deep stereo model should be able to choose optimum cost volume sizes due to automatic estimation of a suitable maximum disparity for each scene in a stereo image sequence. This should result in computational time savings compared to when using a fixed large maximum disparity value that suits all scenes.

2. The model should be able to learn at a lower disparity range

   It is computationally efficient to train a network at a fixed small maximum disparity which limits the size of the cost volume which in turn reduces the number of calculations during back-propagation. However, during the inference stage, the proposed system should be able to adapt to disparity values larger than the value used during training. By using the SNCE metric, the system should be able to automatically estimate a suitable maximum disparity without the user specifying a value.

3. Automatic maximum disparity detection during inference only

   If automatic maximum disparity detection is used during training, the size of the cost volumes can be different from one scene to another making it difficult to use mini-batch gradient descent. Therefore, automatic maximum disparity estimation should only be used during disparity inference and NOT during the training stage (Please refer Appendix D for a detailed explanation).

4. The network should have a low memory footprint so that it can be trained and used on consumer-grade GPU hardware.

   If present, larger maximum disparity values (up to image width) can cause large cost volume sizes which result in higher memory utilization by the algorithm. Therefore, a design goal is set to make the algorithm more memory efficient so that it can be trained and run-on consumer grade GPU devices. Most importantly the network must be developed

in such a way that it can safely handle over-estimation of maximum disparity values under exceptional circumstances.

7.2 Network Architecture

The architecture of the ADSR-Net also follows the "3D regularization structure". It is comprised of a 2D convolutional feature extraction network, a layer-wise cost volume creation module (with SNCE-based termination capability) and a regularization module. Figure 7.1 shows a block diagram of the new architecture and the proceeding sections outline the design of the individual modules in detail.

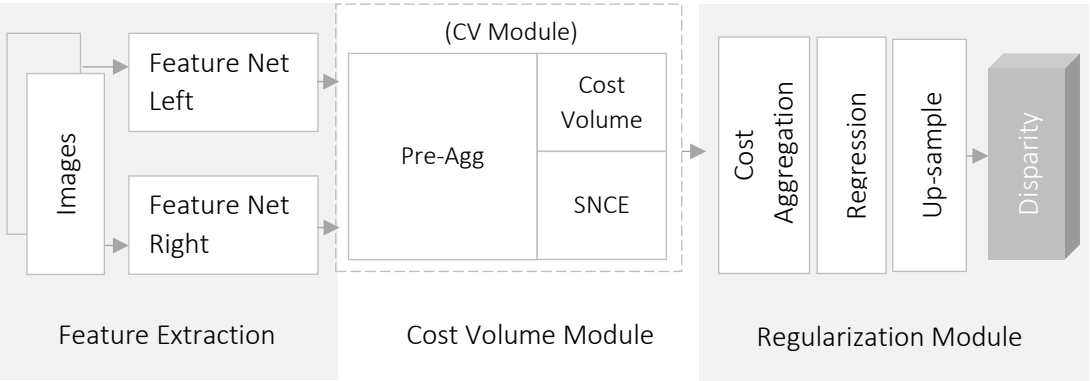


Figure 7.1: High level architecture of the state-of-the-art stereo disparity estimation network (ADSR-Net). Overall layout follows the "3D Regularization Structure". SNCE module within the CV module is only activated during inference and remains switched off during training in order to facilitate training with mini batches of data (i.e., mini-batch gradient descent).

7.2.2 Improved Feature Net

According to the results from the previous chapter, there were a few errors in candidate maximum disparity estimations due to matching errors. Therefore, the feature extraction network in the ADSR-Net model needs to be strengthened to minimize the possibility of such errors. Feature network used for earlier experiments, included 14 layers of "Type A" and "Type B" blocks (Figure 6.2). In the new feature network, the number of "Type A" blocks is increased up to 10 while keeping the number of "Type B" blocks the same at 7. Therefore, the only difference in Feature network from the previous chapter is the addition of three "Type A" blocks. Input and output feature length of all blocks is maintained at 32. Furthermore, the input layer has an input feature length of 3 to accommodate the 3 colour channels to facilitate training with colour images. The output feature length of the input layer matches the input feature length of "Type A" blocks, which is 32.

7.2.3 Cost Volume Module (CV Module)

The CV module of the new architecture must accomplish the following two tasks to achieve full automation in terms of automatically detecting a suitable maximum disparity.

I. Build a 3D cost volume progressively (one layer at a time) in GPU memory

*In the absence of any prior knowledge about the maximum disparity applicable for a scene, the CV module must build the cost volume one layer at a time until the termination criteria are met (i.e., SNCE value reaching zero and any additional criteria). Therefore, the CV module is responsible for progressively allocating memory to the growing cost volume in an efficient manner. The cost volume construction process of the ADSR-Net uses the 2D convolutional feature aggregation network from the previous chapter (**Figure 6.4**) and refer to it as the pre-aggregation network (or Pre-Agg) to improve the clarity of the explanations. By using the Pre-Agg network, feature correspondences from the left and right images are compiled into cost volume layers which are progressively stacked on top of each other to form a variable-sized cost volume.*

II. Compute and use SNCE to detect maximum disparity during inference

Every time a new layer is added to the cost volume, the CV module needs to calculate the SNCE value for the new layer and verify if the metric has reached zero in which case the cost volume can be deemed to have reached its final size. The CV module is then required to terminate the cost volume creation process or schedule termination in such a way that it does not affect any downstream components in the network. Scheduled termination may be necessary if downstream network layers have restrictions on the dimensionality of the input tensors.

In the ADSR-Net, the SNCE computation is carried out at every disparity during stereo inference only. SNCE based termination remains switched off during training of the network for computational efficiency and to allow the use of mini-batch gradient descent. PyTorch-based source code for the cost volume module is provided in *Appendix C* which shows the implementation details of the features outlined so far.

7.2.4 Regularization Module

The regularization module consists of three blocks: the cost aggregation network, disparity regression layer and the up-sampling layer. It accepts a variable-sized cost volume from the CV module and aggregates the costs with the cost aggregation network before using the disparity regression and up-sampling layers to produce a disparity map at the original image resolution.

7.2.4.1 Cost Aggregation Network

The matching cost volume produced by the CV module is further aggregated by a modular cost aggregation network in the ADSR-Net architecture. The main pre-requisite for a candidate cost aggregation network for the ADSR-Net is the ability to accept variable sized cost volumes produced by the CV module. Therefore, the cost aggregation network of the ADSR-Net has been designed as a combination of three types of blocks comprised of 3D convolution, 3D transpose convolution, 3D batch normalization and ReLU layers. The first block type (input layer) which is shown in Figure 7.2, accepts a 3D cost volume. It then converts the 3D cost volume into a 4D tensor which is repeatedly processed by the downstream layers of the cost aggregation network. The 3D convolution operation in the input block is followed by 3D batch normalization and ReLU operations.

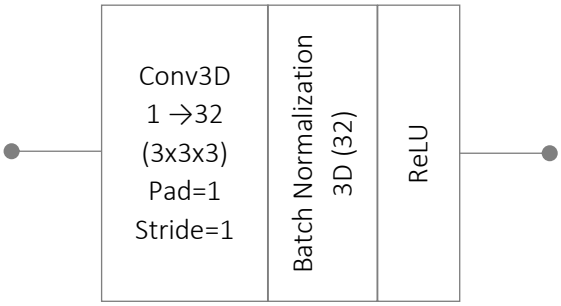


Figure 7.2: Input layer of the 3D convolutional cost aggregation network

To improve the scalability of the aggregation architecture, the rest of the aggregation network includes the "aggregation blocks" shown in Figure 7.3. Unlike the input block, aggregation blocks consist of two parallel branches (Branch-A and Branch-B) of 3D convolutional and transpose convolutional layers connected with intermediate 3D batch normalization and ReLU layers. Furthermore, there is a residual connection from input to the output which allows the high frequency information to be used for the predictions by the downstream layers. The most noteworthy feature of the

aggregation block is the existence of a 3D transpose convolutional layer in Branch-B. The regular 3D convolutional layer in Branch-B has a stride size of 2 pixels along the disparity dimension which reduces the output resolution by half along the disparity dimension of the input tensor. Thereafter, the 3D transpose convolutional layer which also has its stride size set to 2 along the disparity dimension, restores the original resolution. What the design does is providing 3 paths for the data to flow. The residual path does not alter the information whereas the Branch-A attempts to process the input tensor with a detailed view along the disparity dimension. The Branch-B learns to process the same with an aggregated view along the same dimension. The ADSR-Net uses 5 such aggregation blocks which are connected in a cascading arrangement after the input layer. If needed, the number of such blocks can be increased based on the availability of GPU memory.

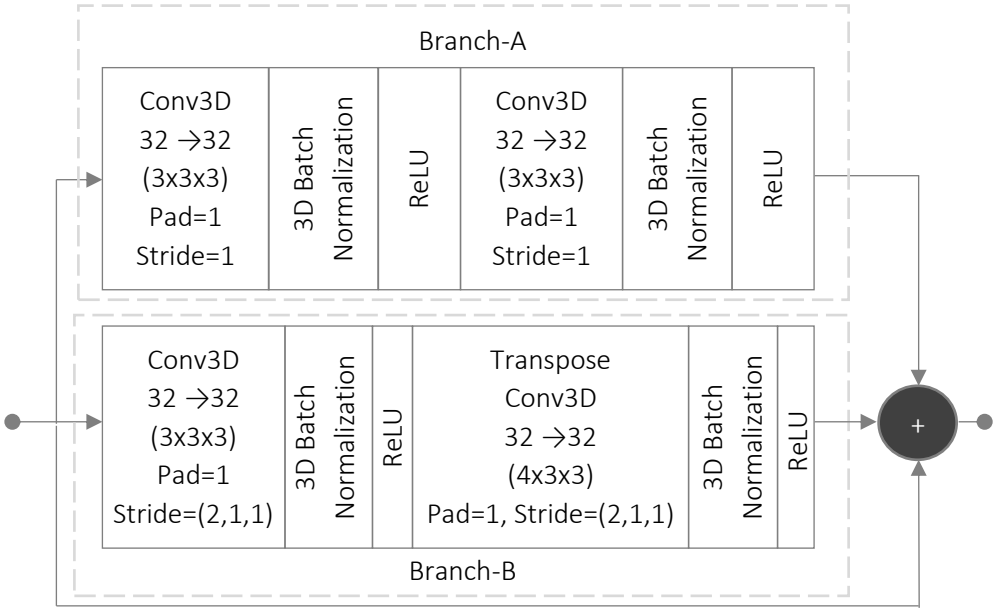


Figure 7.3: An aggregation block with a residual connection. There are 5 of them in the regularization module of the ADSR-Net model used for the experiments in this chapter.

At the end of the chain of aggregation blocks, an "output block" helps reduce the dimensionality of the input tensor back to 3D from 4D by using the last 3D convolutional layer as shown in Figure 7.4. The output block needs to produce a 3D tensor as input to the regression layer of the regularization module. To permit variability of the values, the output block does not include batch normalization or ReLU operations after the final 3D convolution operation. Therefore, the final matching

costs can take any value without the network imposing restrictions (this is important when ensuring unimodality of the aggregated costs).

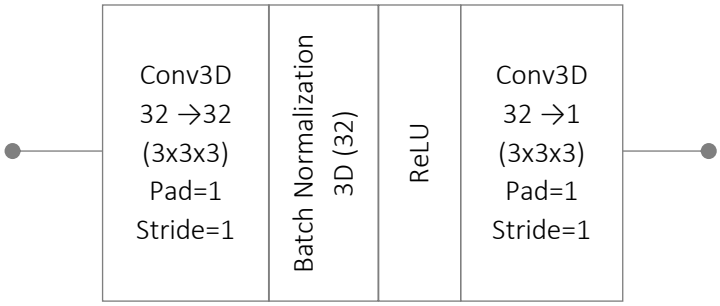


Figure 7.4: Layers in the output block in the aggregation network of ADSR-Net. Note the last 3D convolutional layer which does not have batch normalization or ReLU operations after it.

7.2.4.2 Regression and Up-Sampling Layers

The aggregated cost volume coming out of the output block of the cost aggregation network is used by the regression layer to regress disparities through a differentiable "softargmin" approximation to the traditional "argmin" operation. Starting from the down-sampled output from the input layer of the feature extraction network, all operations take place at $1/4^{th}$ the original resolution (i.e., $1/4^{th}$ width and $1/4^{th}$ height). Therefore, to produce disparity maps at the original image resolution, an up-sampling layer is also used at the very end of the regularization operation.

7.2.5 Efficiency of the Cost Volume Module

Before training the network, it is important to evaluate the performance of the cost volume module to ensure that it does not cause significant delays during the forward pass through the network. Longer delays can make the inference times prohibitively longer rendering the network unfeasible. Therefore, in a preliminary study, the forward propagation latency through the ADSR-Net was measured and compared against the latency data acquired by replacing its CV module with fixed-sized cost volumes. Two types of such fixed-sized cost volumes (3D and 4D) were used to obtain two sets of statistics for comparison.

First, the untrained network was used to produce disparity maps for ten randomly selected stereo image pairs from the Scene Flow- "Driving" dataset. Since the network was untrained at this point (hence no SNCE convergence), disparity cut-off had to be programmatically initiated at pre-configured values of 960, 480, 240 and 120 which

correspond to the full, half, quarter and half-quarter width of the original image. Then the forward propagation time on an NVIDIA GTX1070 GPU was measured for each image pair at the given disparity cut-off points. An average value for the latency (over 10 images) was obtained and recorded for analysis. Then the same process was repeated with fixed-sized (define once) cost volumes (3D and 4D) in place of the CV module.

7.2.5.1 Forward Propagation Times on Scene Flow Data

The results of the experiment involving 10 images from the Scene Flow driving data are shown in the Table 7.1 below.

Cost Volume Type	Cost Volume Construction Method	Mean Forward Propagation Time (ms)			
		Max Disparity (960)	Max Disparity (480)	Max Disparity (240)	Max Disparity (120)
CV Module	*Layer-wise*	*2281.18*	*1176.51*	*636.34*	*371.50*
3D Cost Volume	*Define once/Fixed*	*2116.59*	*1117.97*	*602.82*	*355.49*
4D Cost Volume	*Define once/Fixed*	*2140.44*	*1130.88*	*618.34*	*362.37*

Table 7.1: Forward time delay for the network before and after replacing the cost volume creation process with fixed-sized 3D and 4D cost volumes (i.e., Obtained using the Scene Flow Driving dataset with an original image resolution of 960x540)

The difference in forward propagation times between the layer-wise cost volume construction (first row of results in Table 7.1) and the fixed 3D cost volume (second row), can be attributed to the SNCE estimation delay and the time taken for the layer-stacking operation. For example, according to the results obtained at a maximum disparity of 120 pixels, the additional delay introduced by those operations is approximately 16ms which is about 4.5% increase from the total time associated with a fixed-sized 3D cost volume (pre-defined in memory) with a maximum disparity of 120 pixels. Similarly, at the theoretical maximum disparity (i.e., image width of 960 pixels), the same amounts to around 7.7% increase. When compared with the processing times associated with a 4D cost volume, the additional delay introduced by the CV module ranges from 2.5% to 6.5% of the total time (when DSR is changed from half-quarter to full image width).

7.2.5.2 Forward Propagation Times on KITTI 2015 Data

The same experiment was re-run using stereo images from the KITTI 2015 driving dataset to test against the changes in image resolution. The KITTI images have a

resolution of 1242x375 as opposed to the Scene Flow images with a resolution of 960x540. The results are shown in the Table 7.2 below.

Cost Volume Creation Method	*Mean Forward Propagation Time (ms)*			
	Max Disparity (1242)	Max Disparity (620)	Max Disparity (308)	Max Disparity (152)
CV Module	*2768.97*	*1401.97*	*736.16*	*408.57*
Fixed 3D Cost Volume	*2568.72*	*1310.00*	*695.93*	*391.16*
Fixed 4D Cost Volume	*Memory Error*	*1336.49*	*706.07*	*393.12*

Table 7.2: Forward propagation times for the network before and after replacing the layer-wise cost volume creation process with fixed-sized 3D and 4D cost volumes (i.e., Obtained using the KITTI 2015 Driving dataset with original image resolution of 1242x375)

As per the results, the increase in delay introduced by SNCE estimation and tensor stacking operations, has varied from 4.5% to 7.7% approximately when the maximum disparity was changed from 152 to 1242 pixels. Another significant observation from the results is the "Memory Error" associated with the 4D cost volume which justifies the use of a pre-aggregation network in the ADSR-Net to reduce the dimensionality of the cost volume before regularization. On-board memory available on NVIDIA GTX1070 GPU (8 GB) has been insufficient in the case of the 4D cost volume when the maximum disparity was set to image width. This validates the use of a 3D cost volume in the ADSR-Net. If a higher dimensional cost volume is used, then there can be abrupt terminations of the inference process due to over-estimated maximum disparity under exceptional circumstances. As shown in the previous chapter, there can be exceptions when the system encounters a matching error or unfamiliar data.

Therefore, in summary, the following conclusions can be made based on the results of the experiments involving the CV module of the ADSR-Net.

- ✓ The processing delay introduced by the layer-wise cost volume construction and SNCE calculations does not lead to significant increase in overall forward propagation time (typically less than 10% on standard stereo datasets)
- ✓ Smaller dimensionality of the cost volume produced by the ADSR-Net leads to efficient use of on-board memory, making it better suited for layer-wise cost volume construction in GPU memory.

7.3 ADSR-Net Training

After determining that the CV module is able to process larger maximum disparity values without introducing significant delays, the next step is to train the model so that it can accurately estimate output disparity maps while using the SNCE metric to estimate a suitable maximum disparity value during disparity inference. As learned from Chapter 4, the accuracy of the SNCE-based maximum disparity predictions depends on the unimodality of the matching costs. Therefore, the training process needs to improve the overall matching accuracy of the network while ensuring unimodality of the matching costs used by the SNCE metric.

7.3.1 Training Objectives

Unlike the other deep learning-based stereo techniques which focus on the accuracy of the output disparity maps alone, the ADSR-Net needs to achieve a higher level of accuracy in maximum disparity estimations and dense disparity predictions simultaneously. Furthermore, the ADSR-Net model must be trained at a smaller fixed maximum disparity in order to use mini-batch gradient descent (*Appendix D)* and to complete training within practical time limits. As shown in Table 7.1, if the estimated maximum disparity reaches the full image width during training, then the processing time can be more than 2 seconds per image pair on a consumer grade GPU. That would make the training time impractical for large datasets.

7.3.2 Network Preparation

For the regression layer in the new architecture to work effectively, the cost aggregation network must produce an aggregated-cost volume with unimodal cost distributions. However, it does not necessarily mean that the CV module output will also be unimodal. For the SNCE-based maximum disparity estimations to be accurate, the matching costs produced by the CV module must also be unimodal in nature. Thus, to achieve unimodality in the CV module output and the aggregation network output, both must have “Softargmin” based disparity regression layers connected to their respective outputs during training. The cost aggregation network already has a regression layer connected although the CV module does not have one. The solution is to connect a secondary regression layer to the CV module output separately as shown in Figure 7.5. By adding an up-sampling layer, another supplementary disparity

output can be obtained at original image resolution as shown in the figure. Now, if both disparity outputs can be used during the loss computation, it is possible to ensure unimodality in CV output while improving the final disparity output accuracy through supervised training. Nevertheless, it is important to note that this modified architecture is only used during training.

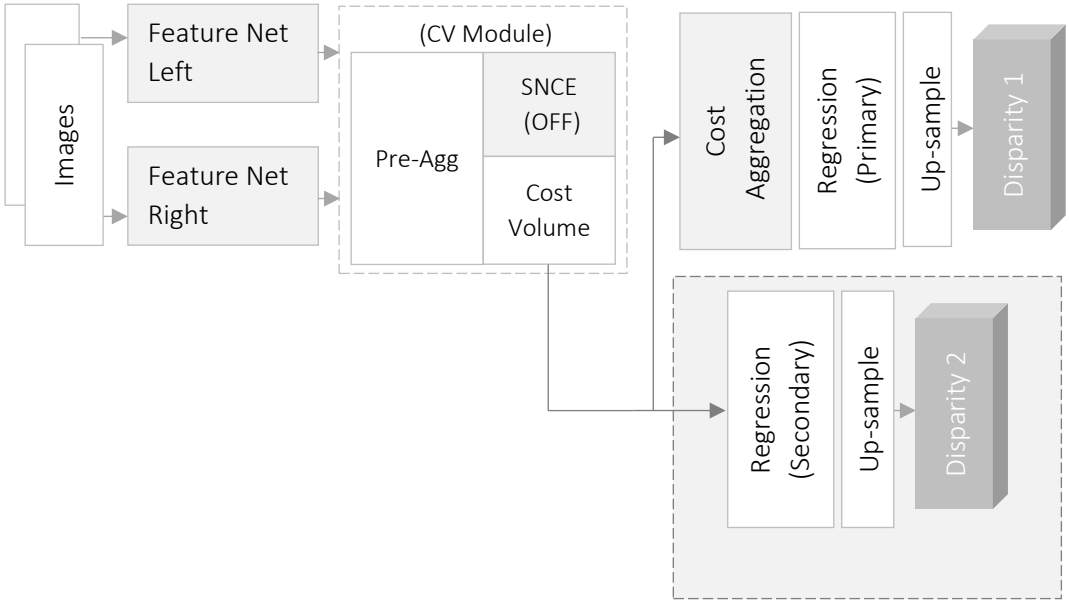


Figure 7.5: A graphical illustration of how the ADSR-Net is slightly changed during training to enforce SNCE convergence. It is important to note that the automatic maximum disparity estimation remains turned off during the training process which means a smaller fixed maximum disparity is enforced. Any pixel having its predicted disparity value higher than the selected maximum is excluded from backpropagation.

7.3.3 Clamped Training

Typical deep stereo disparity estimation algorithms use the difference (or error) between the predicted and expected disparity maps as input to the network loss calculation. With the ADSR-Net modified during training, the loss calculation needs to combine the two losses involving “Disparity 1” and “Disparity 2” outputs in Figure 7.5. This effectively clamps the final disparity prediction to the intermediate prediction from the pre-aggregation network, forcing the network to improve accuracy of both predictions. Therefore, in this thesis, the process is referred to as “Clamped Training”.

It is also possible to use weights to decide how much each disparity error estimate contributes to the final loss. However, in this study, each disparity error was set to contribute equally towards the overall loss without skewing towards one of the disparity outputs.

7.3.4 Prediction Loss/Accuracy

Using the same loss function (SmoothL1 Loss) the output loss calculation for the ADSR-Net was modified to incorporate the final and intermediate predictions ("Disparity 1" and "Disparity 2" in Figure 7.5). Hence, the final loss calculation ($ADSR\_Net\ Loss$) is given by:

$$ADSR\_Net\ Loss\ = \boldsymbol{loss}(Disparity1, gt) + \boldsymbol{loss}(Disparity2, gt) \qquad (7.1)$$

$$with \rightarrow \boldsymbol{loss}(x, y) = \begin{cases} 0.5(x-y)^2/\beta, & if\ |x-y| < \beta \\ |x-y| - 0.5\beta\ , & otherwise \end{cases}$$

7.3.5 Hyper-Parameters, Optimizer and GPU Hardware

Selecting an optimal learning rate for a network is a challenging exercise on its own and the supplementary disparity map ("Disparity 2") makes the margin of error even smaller. That makes clamped training susceptible to learning rate selection. Since two partially shared networks are learning to predict similar outputs at the same time, it is important to select a learning rate which is suitable for both networks. For the ADSR-Net, a learning rate of $1x10^{-4}$ was found to be a good starting point. Throughout the training process, image data was augmented to have an image size of 512x256 pixels and 256x256 during the fine-tuning stage. The same optimizer used in the previous chapter (Adam optimizer) was used with the selected learning rate and other default settings (beta parameters of the Adam optimizer= [0.9, 0.999], weight decay=0). The batch-size was set to 2. Initial tests with hardware revealed that training on NVIDIA RTX2080 GPU with 2944 CUDA cores and 8 GB on-board memory, was approximately 30% faster than training on NVIDIA GTX1070 due to the highe7r number of CUDA cores.

7.3.6 Data Preparation

The Scene Flow "Driving" image set, Scene Flow "Flying Things 3D" and KITTI 2015 stereo datasets were normalized using Z-score normalization before using them for training the ADSR-Net. Unlike the model developed in the previous chapter which used the mean and standard deviation at individual image level, the ADSR-Net was trained with images normalized using the channel-wise mean and standard deviation values calculated over each dataset. All images were then randomly cropped to a size of 512x256 pixels during training so that the training speed can be improved while

increasing the diversity of the image data encountered by the network. The channel-wise statistics used for image normalization can be found in *Appendix E*.

7.3.7 Training Regime

Due to the existence of multiple datasets and multiple training objectives, the training process had to be conducted in stages as outlined below.

I. Initial training stage

The objective of initial training was to train the ADSR-Net enough to evaluate the accuracy of the maximum disparity estimations and gains in performance as a result of estimating the maximum disparity automatically. Like the experiments used in the previous chapter, a selected set of stereo images from the Scene Flow "Driving Dataset" was used for initial training.

II. Extended training

The objective of extended training was to train the ADSR-Net over a large dataset so that it can predict disparity maps across various scenes without retraining. Ideally a large real-world dataset with dense disparity ground-truth data should have been used. However, the available real-world datasets such as KITTI 2012/2015, only contain a limited number of stereo images (with sparse ground-truth). Therefore, extended training was conducted using the synthetic "Flying Things 3D" dataset before fine-tuning with a real-world dataset.

III. Fine-Tuning

The ADSR-Net is expected to predict disparity maps for real-world stereo images captured with real cameras. To accomplish the same, the model trained with "Flying Things 3D" dataset during extended training, was trained further with the KITTI 2015 dataset.

It is important to note that a lower learning rate was used during the "Fine-Tuning" stage. The number of training cycles (epochs) was also limited. This was done to make sure that the fine-tuning process does not undo the learning during the previous

stages (i.e., does not change the weights and biases significantly). A summary of parameters associated with each of the training stages is provided in Table 7.3.

Parameter	Initial Training	Extended Training	Fine Tuning
Dataset	Scene Flow "Driving"	Scene Flow "Flying Things 3D"	KITTI 2015 Scene Flow
Image Size (px)	960x540	960x540	1242x375
Crop Size	512x256	512x256	256x256
Maximum Disparity at Cost Volume Resolution (*During Training Only*)	64	64	32
Batch Size	2	2	8
No of Images	600	22,390	200
Training/Testing Split	80/20	80/20	90/10
Epochs	500	1024	160
Learning Rate	0 - 300 epochs → 1e-4 300 - 500 → 1e-5	0 - 500 → 1e-4 500 - 800 → 1e-5 800 - 1024 → 1e-6	0 - 160 → 1e-7 (monitor and adjust)
Optimizer	Adam	Adam	Adam

Table 7.3: Summary of parameters associated with each of the training stages of the ADSR-Net.

7.4 Evaluation

In the sections to follow, the ADSR-Net is evaluated against the objectives outlined at the beginning of the chapter. To prepare the ADSR-Net for evaluation, SNCE is enabled in the CV module (as it was turned off during training). Furthermore, the additional layers such as regression and up-sampling layers which were connected to the Pre-Agg output of the network during training are disabled to stop them from causing any delays during inference, particularly when measuring delays to validate performance. The model parameters saved at different training-stages are loaded into the model and the updated model is used to produce disparity maps for the validation data splits as well as stereo images from datasets such as ETH3D, Middlebury 2014 and KITTI 2012. The results are then analysed for performance and accuracy in terms of maximum disparity and dense-disparity predictions. Finally, the generalization capabilities of the ADSR-Net are evaluated using the stereo images captured with a custom-built stereo camera.

The overall evaluation process can be divided into four steps. They are aimed at determining the following:

1. Accuracy of the maximum disparity estimations
2. Performance improvements due to automatic maximum disparity estimation
3. Accuracy of the dense disparity maps produced by the ADSR-Net
4. Generalization capabilities of the ADSR-Net when presented with custom stereo image data

The evaluation steps presented in this chapter, follow the order in which they were done in parallel to the training process. Since the accuracy of the maximum disparity estimations by the ADSR-Net is the main objective of this chapter, it is evaluated immediately after completing the initial training phase. The same is true about performance which must also be verified immediately after the initial training stage. On the other hand, the accuracy of the disparity maps is evaluated at multiple stages (at the end of every training stage and comprehensively after the training is complete).

However, the generalization capabilities of the ADSR-Net are only evaluated at the end of the training process, using the fully trained ADSR-Net. A more detailed view of the individual steps involved in evaluation is given in Table 7.4. The first column in the table shows the evaluation stage while the second column defines the corresponding objectives. The third column outlines how the ADSR-Net and the test data is prepared before the results are obtained. The fourth column shows the different types of analysis (quantitative and qualitative) conducted on the results obtained by using the prepared data and the trained network model.

#	Evaluation Step	Preparation	Analysis
1	Accuracy of the maximum disparity estimations	<u>Evaluation Data</u> *5 Sequences of stereo test image pairs from the Scene Flow "Driving" dataset (from the validation/test data split)* <u>Data Preparation</u> *Crop images and ground-truth data files to create pseudo random sequences of 60 images, each having the size 512x256 pixels* <u>Model Parameters</u> *ADSR-Net with parameters saved after the initial training stage* <u>Model Preparation</u> ✓ *Turn ON SNCE in the CV module*	<u>Quantitative</u> *Compare estimated vs. ground-truth maximum disparity* <u>Qualitative</u> *Visual analysis of the predicted dense disparity maps*
2	Performance improvements due to automatic disparity estimation (improvements in computational efficiency)	<u>Evaluation Data</u> Test image pairs from the Scene Flow "Driving" dataset (from the validation/test split) <u>Data Preparation</u> Crop images and ground-truth data files to create five pseudo random sequences of 60 images, each having the size 512x256 pixels <u>Model Parameters</u> ADSR-Net with parameters saved after the initial training stage <u>Model/Models Preparation</u> ✓ Turn ON SNCE in the CV module for measuring inference time for automatic disparity ✓ Turn OFF SNCE and set the maximum disparity to the population max in test data	<u>Quantitative</u> Estimate mean processing time over 60 images for the five image sequences and compare that with the processing time observed when using a fixed cost volume with maximum disparity set to the maximum value observed in the test data <u>Qualitative</u> N/A
3	Accuracy of the dense disparity maps	<u>Evaluation Data</u> ✓ *Stereo image pairs from the Scene Flow "Flying Things3D", Scene Flow "Driving" and KITTI 2015 stereo datasets – (from the validation/test split)* ✓ *Stereo image pairs (15 RGB stereo pairs) from Middlebury 2014 stereo dataset* ✓ *Stereo image pairs (27 pairs) from ETH3D grayscale stereo image dataset* <u>Data Preparation</u> *None* <u>Model Parameters</u> ✓ *ADSR-Net parameters saved after each training stage for Scene Flow "Driving", "Flying Things 3D" and KITTI 2015 datasets* ✓ *Fully trained ADSR-Net parameters saved after fine-tuning for stereo images from Middlebury 2014, ETH3D and KITTI 2012*	<u>Quantitative</u> *Estimate and compare common stereo disparity evaluation metrics* ✓ *End-Point-Error* ✓ *3-Pixel Error* <u>Qualitative</u> *Visual analysis of the predicted dense disparity maps*

		Model Preparation ✓ *Turn ON SNCE in the CV module*	
4	Generalization capabilities of ADSR-Net when presented with data captured with custom stereo image pairs	**Evaluation Data** ✓ *Stereo image captured with the custom-built stereo camera setup explained in chapter 3.* **Data Preparation** *Rectify images* **Model Parameters** ✓ *Fully trained ADSR-Net parameters saved after fine-tuning for stereo images from Middlebury 2014, ETH3D and KITTI 2012* **Model Preparation** ✓ *Turn ON SNCE in the CV module*	**Quantitative** N/A **Qualitative** ✓ *Visual analysis of the predicted dense disparity maps* ✓ *Identify challenging scene conditions*

Table 7.4: A detailed overview of the steps covered in this chapter when evaluating the ADSR-Net using the test data as well as additional stereo images from standard datasets and images captured with a custom stereo camera.

7.5 Results and Evaluation

The following sub-sections with subheadings starting from EVALUATION STEP 1 to EVALUATION STEP 4, include evaluations of the results obtained at each of the steps outlined in Table 7.4. Then at the end, the strengths and weaknesses of the algorithm will be discussed with examples.

7.5.1 **EVALUATION STEP 1**: Accuracy of Maximum Disparity Predictions

As shown under step 1 in Table 7.4, immediately after the initial training stage, the ADSR-Net was used to produce a series of disparity maps for 5 sets of cropped stereo image sequences from the Scene Flow "Driving" test data. Pseudo-random values were used for the positions of the cropped images in each of the stereo image pairs so that the tests can be repeated. The results were used to make the following measurements for further analysis:

- ✓ Maximum disparity detected by the ADSR-Net
- ✓ The maximum ground-truth disparity for the image pair

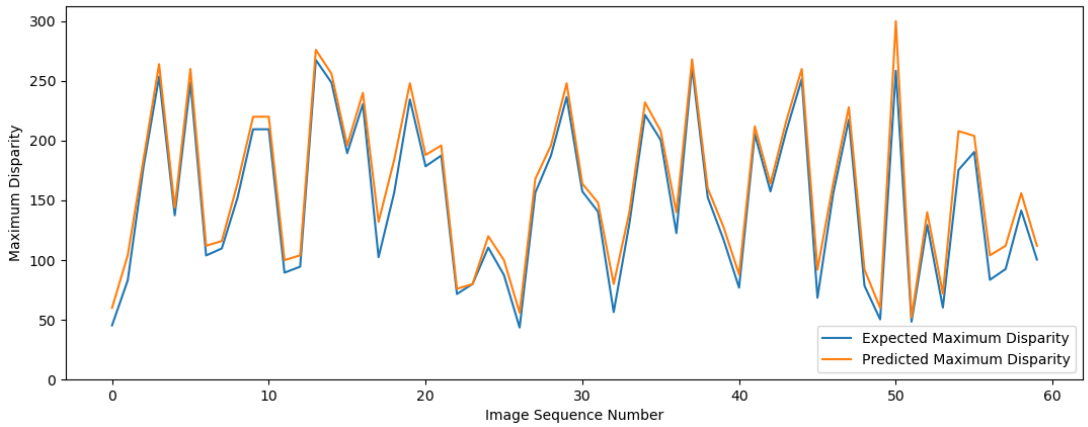


Figure 7.6: Variation of the maximum ground-truth disparity (Expected) and the maximum disparity detected by the ADSR-Net (Predicted) for the 1st pseudo-random image sequence from the Scene Flow validation data

Figure 7.6 contains the results of the first image sequence and it shows the variation of the maximum disparity values predicted by the ADSR-Net alongside the ground-truth maximum disparity values obtained from the Scene Flow validation data. The first observation that can be made from Figure 7.6 is the wider variation in maximum disparity compared to the image sequences used in the previous chapter. It is due to the much smaller cropping size during data augmentation. Analysis in the previous chapter included images cropped at 800x480 pixels as opposed to 512x256 pixels used here. The original image size of the Scene Flow dataset being 960x540, the smaller cropping size allows image patches to be extracted from different areas of the images that correspond to depths varying from low to high. The other main observation is how the ADSR-Net has managed to predict maximum disparity equal to or slightly higher than the ground-truth value for all image pairs. The same is true for the results of the 3rd image sequence shown in Figure 7.7.

Important: In practice users select maximum disparity values which are slightly larger than the expected maximum for different scenes. Any value smaller than the maximum or significantly larger can lead to errors in final disparity estimations. Therefore, the objective here is to automatically estimate a certain maximum disparity value which is equal to or slightly higher than the actual which is acceptable because of the presence of an additional cost aggregation network which is able to regularize the output further to produce accurate disparity maps.

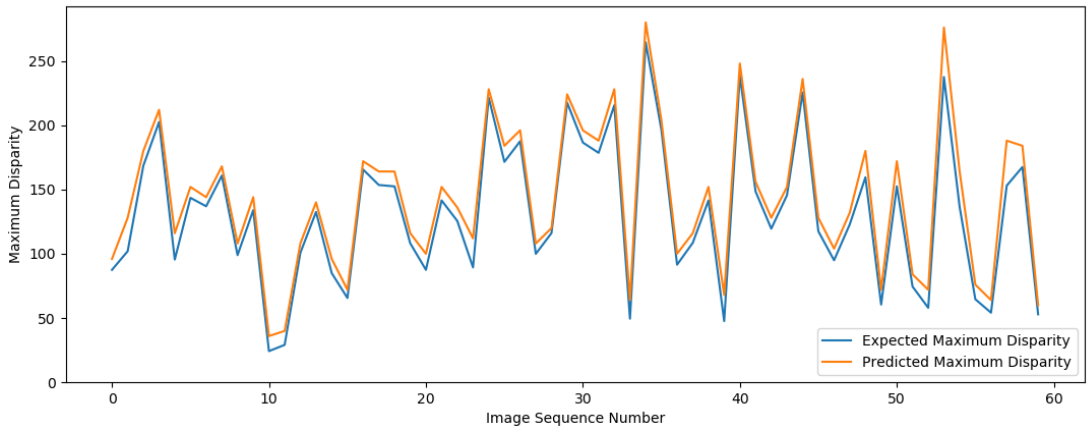


Figure 7.7: Variation of the maximum ground-truth disparity (Expected) and the maximum disparity detected by the ADSR-Net (Predicted) for the 3rd pseudo-random image sequence from Scene Flow "Driving" validation data

7.5.1.1 Overview of Exceptions in Maximum Disparity Estimations

Results for the 4th image sequence shown in **Figure 7.8**, indicate that the SNCE-based maximum disparity estimation for one of the cropped stereo image pairs has been uncharacteristically higher than the maximum ground-truth disparity. In order to find the root cause for the exception the corresponding final disparity map was analysed.

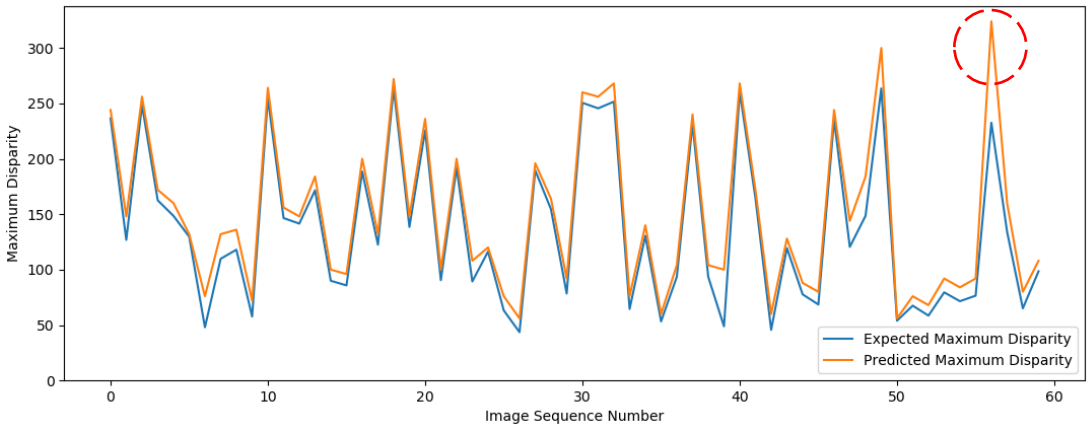


Figure 7.8: Variation of the maximum ground-truth disparity and the maximum disparity detected by the ADSR-Net for the 4th image sequence from Scene Flow validation data. There is one instance for which the ADSR-Net has considerably over-estimated the maximum disparity for the scene.

The input images, the ground-truth disparity map and the ADSR-Net prediction for the exception reported in the 4th image sequence is shown in **Figure 7.9**. It is clear from the input images that they have been affected by glare. Consequently, the output disparity map produced by ADSR-Net has also been affected. This is reminiscent of the results in the previous chapter that showed how image glare can make it unable for the network to reliably match the pixels in the affected area which increases the matching uncertainty, resulting in cost extrema movements past the maximum disparity.

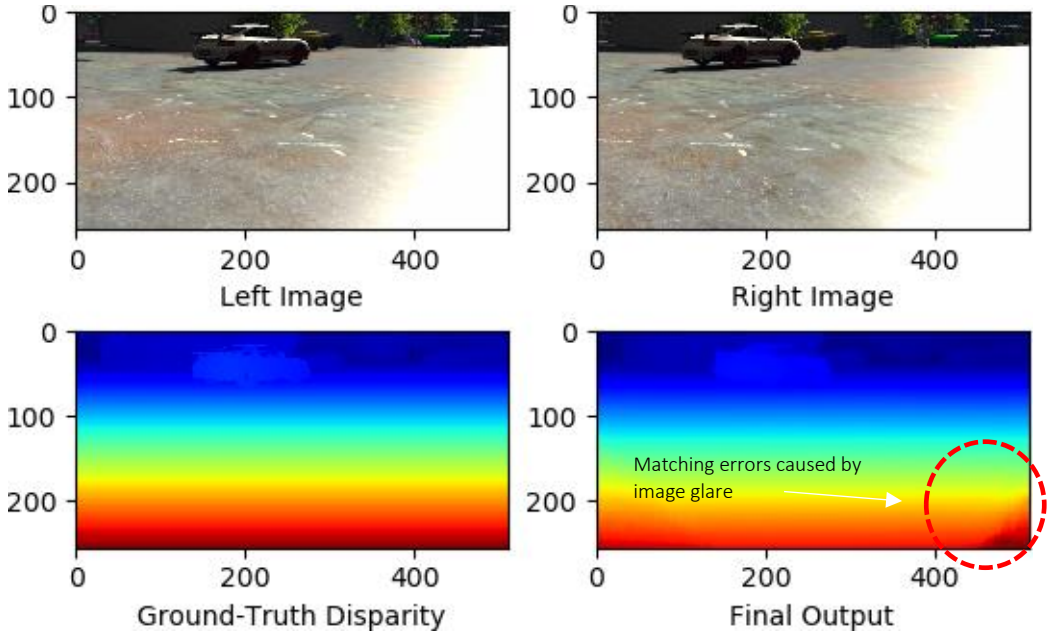


Figure 7.9: The scene from the 4th image sequence in which the ADSR-Net has considerably over-estimated the maximum disparity. Final disparity output (by the ADSR-Net) for the scene is shown at the bottom right-hand corner which suggests that there were matching errors due to the glare present in the input images.

Results of the 5th image sequence also suggest that the ADSR-Net has overestimated the maximum disparity for one of the cropped image pairs. The unusually high prediction is marked with a dashed red circle in Figure 7.10.

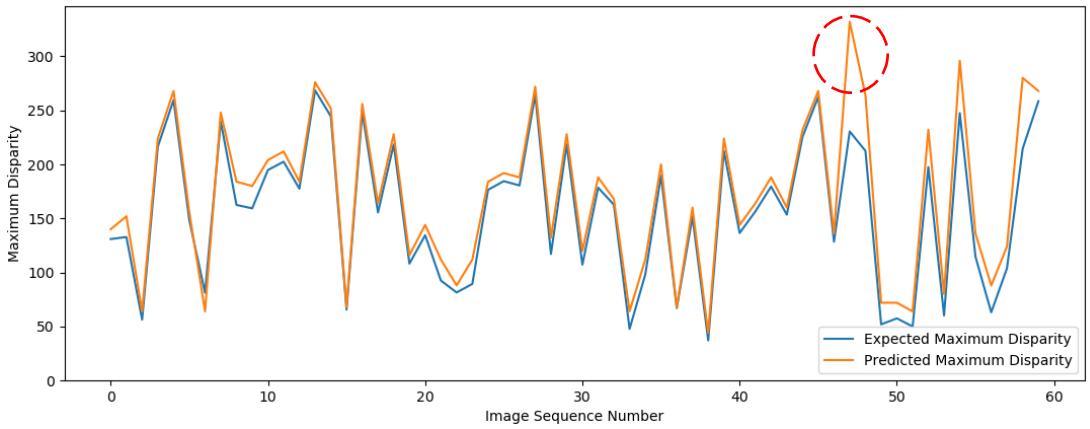


Figure 7.10: Variation of the maximum ground-truth disparity and the maximum disparity detected by the ADSR-Net for the 5th image sequence from Scene Flow validation data. There is one instance for which the ADSR-Net has considerably over-estimated the maximum disparity for the scene.

When the corresponding input images were extract from the 5th image sequence, they also indicated the presence of glare. However, unlike the exception in the 4th image sequence, the ADSR-Net output did not show any visible errors in the corresponding areas in the the final disparity output shown in Figure 7.11. However, by enabling the

additional disparity regression and up-sampling layers used for training, it was possible to obtain an intermediate disparity map which showed matching errors at pre-aggregation level due to the glare related artefacts in the input images. Since image glare has only affected a small region, the cost aggregation network has been able to filter out the exception from the final disparity prediction. However, as seen from the exception in the 4th image sequence, this may not be possible in all cases.

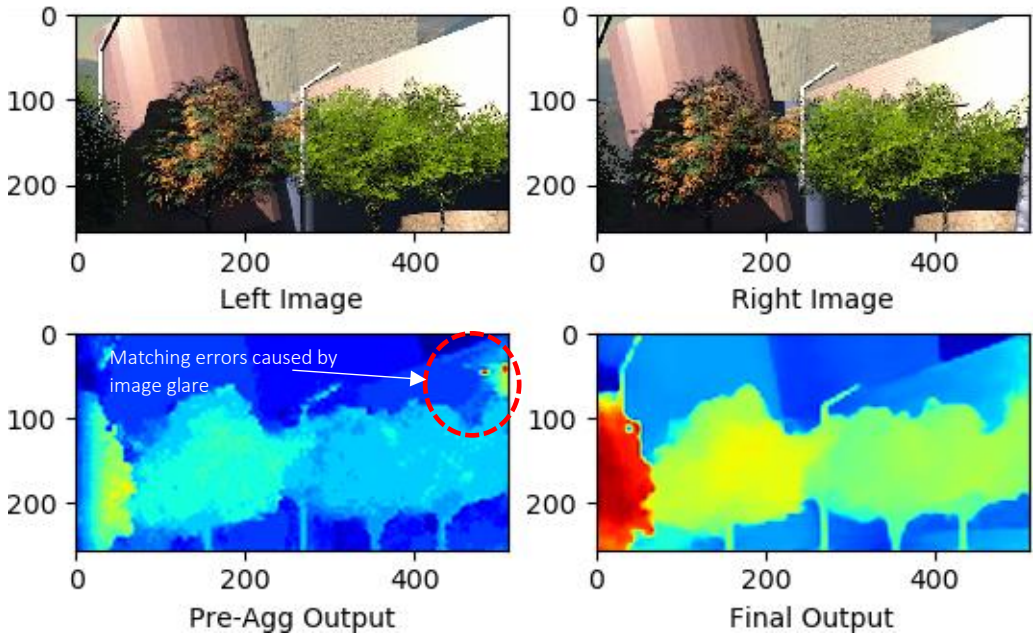


Figure 7.11: Scene from the 5th image sequence in which the ADSR-Net has considerably over-estimated the maximum disparity. Pre-aggregation output for the scene shown at the bottom left-hand corner shows errors caused by bad matching due to the glare present in the input images. Since only a small area has been affected, the cost aggregation network has been able to filter out the exception in the final disparity prediction.

A rare exception, in which the ADSR-Net has underestimated the maximum disparity, can be seen in the results of the 2nd image sequence shown in Figure 7.12.

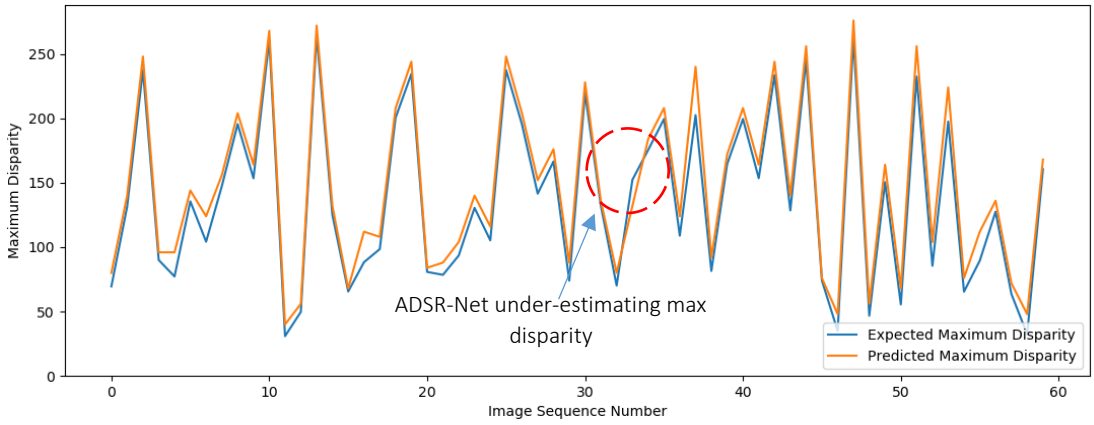


Figure 7.12: Variation of the maximum ground-truth disparity (Expected) and the maximum disparity detected by the ADSR-Net (Predicted) for the 2nd pseudo-random image sequence from Scene Flow "Driving" validation data. There is one instance (circled) in which the ADSR-Net has underestimated the maximum disparity value.

Since most issues encountered by the ADSR-Net were related to the matching accuracy of the pre-aggregation network, the same should be tested for the image pair related to the underestimation of the maximum disparity. To begin the triaging process, the SNCE was disabled on ADSR-Net and the maximum disparity was set to 152 pixels (or 38 pixels at cost volume resolution) which is the ground-truth maximum disparity for the image pair in concern. The additional regression and up-sampling layers (used for training) were also connected to the ADSR-Net to obtain a disparity map from the pre-aggregation network. The corresponding input images, the ground-truth disparity map and the intermediate disparity map are shown in **Figure 7.13**.

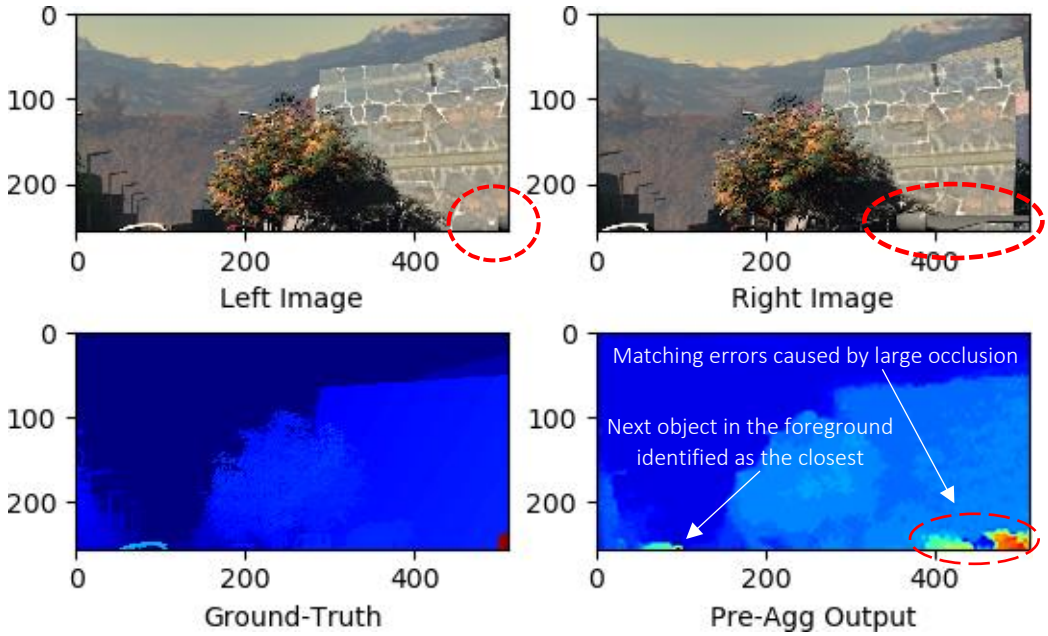


Figure 7.13: The scene from the 2nd sequence in which the ADSR-Net had under-estimated the maximum disparity. Pre-aggregation output for the scene suggests that the closest object has been mismatched (even with a fixed maximum disparity) due to the errors associated with the mostly occluded object (circled in red).

According to **Figure 7.13**, an object in the right image (a part of a passing vehicle) is occluding a large area (about 1/3rd the image width) in the right image. This has resulted in the pre-aggregation network registering errors in the corresponding region as shown in the "Pre-Agg" output. Even with a manually configured maximum disparity, pre-aggregation network has incorrectly identified the second closest object as the closest. Since the ADSR-Net has only completed initial training the exception can again be attributed to the ineffective matching at pre-aggregation level. Therefore, the network needs to be trained even further which is the objective of the extended training.

7.5.1.2 Qualitative Analysis of the Results after Initial Training

After confirming that the ADSR-Net is able to reasonably predict maximum disparity values in stereo image sequences with larger variance in maximum disparity (despite a few exceptions as noted earlier), the same ADSR-Model was used to predict disparity maps for test stereo images from the Scene Flow "Driving" dataset at their original resolution (of the size 960x540 pixels). Disparity maps for 3 sample images, produced by the ADSR-Net are given in Figure 7.14.

As seen from the figure, even after the initial training stage with only 480 stereo image pairs, the ADSR-Net has started producing reasonably accurate disparity maps without user-specified parameters such as the maximum disparity which is a significant achievement compared to other deep learning techniques which require at least the maximum disparity value to be specified by the users.

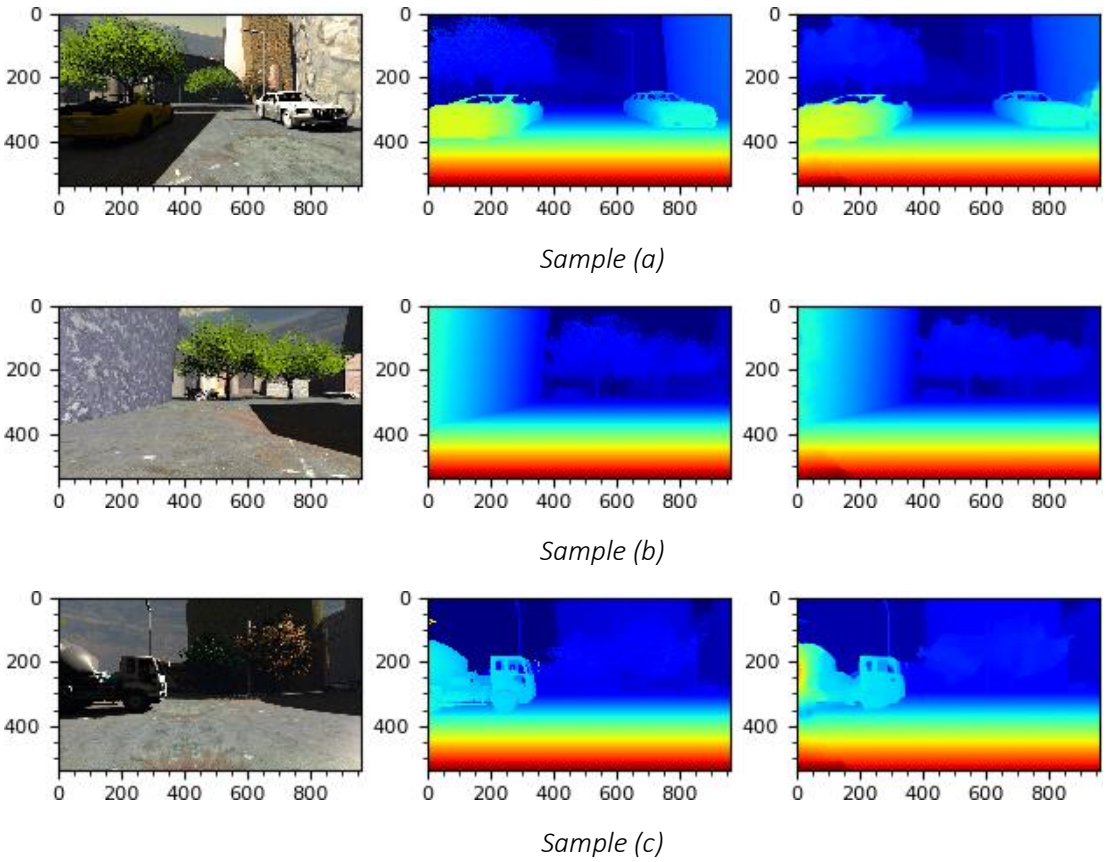


Figure 7.14: Three sample disparity maps produced by the ADSR-Net after the initial training stage which was conducted using the "Scene Flow – Driving" dataset. The shown maps have been selected from the ADSR-Net predictions for validation image pairs. The leftmost image in each row shows the reference image (left) in the stereo image pair. The disparity map in the middle shows the ground-truth disparity for the given stereo image pair. The output disparity map on the right, shows the ADSR-Net prediction.

7.5.2 EVALUATION STEP 2: ADSR-Net Performance

Based on the accuracy of the maximum disparity predictions by the ADSR-Net and its ability to produce disparity maps with reasonable quality, the model was determined to be ready for a performance evaluation. First, the trained ADSR-Net was used to produce disparity maps for 5 pseudo-randomly cropped stereo image sequences (from the validation dataset) while determining the maximum disparity automatically. The average/mean inference times over each image sequence was recorded for comparison. Next, the experiment was repeated with a reference implementation having a fixed-sized cost volume (i.e., by replacing the CV module of the ADSR-Net with a fixed-sized cost volume). The reference deep stereo network shared the feature-network and the regularization module with the ADSR-Net but used a fixed-sized cost volume. The maximum disparity of the fixed cost volume was set to the maximum disparity observed in validation data (84px at quarter resolution). The bar chart in **Figure 7.15** shows a comparison of the mean inference times across 5 image sequences. From the figure, it can be observed that the ADSR-Net (with automatic maximum disparity estimation) has produced the lowest mean inference times across all five image sequences. NVIDIA RTX2080 GPU was used for the evaluation.

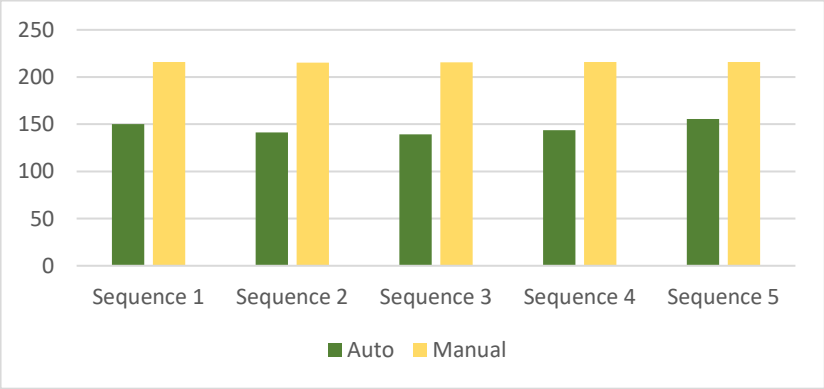


Figure 7.15: Difference in mean processing times between automatic and fixed maximum disparity for 5 pseudo-random image sequences from the Scene Flow "Driving" test data with each sequence having 60 images. Green bars correspond to the inference times of the ADSR-Net whereas the yellow bars correspond to an ADSR-Net equivalent with a fixed-sized cost volume in place of the CV module (reference implementation).

In fact, when the maximum disparity is manually set to the population maximum, the average processing time per image pair has increased by as much as 50% in some image sequences, compared to the processing times achieved through automatic maximum disparity with the ADSR-Net. This can be seen from **Figure 7.16** which shows

the percentage increase in processing times when the maximum disparity is set to a fixed value defined by the user. Although it is possible for the measurements to change depending on the distribution of maximum disparity in the dataset, the observed savings in processing time with ADSR-Net has exceeded 50% in 3 out of 5 image sequences used for the experiment.

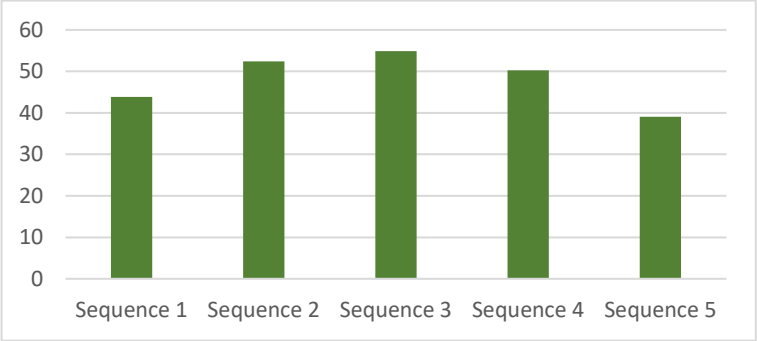


Figure 7.16: Percentage increase in processing time when the maximum disparity was set to the population maximum of the Scene Flow "Driving" test data, manually.

7.5.3 EVALUATION STEP 3: Dense Disparity Prediction Accuracy of the ADSR-Net

Due to the availability of multiple datasets for evaluation during and after training, the accuracy of the disparity predictions by the ADSR-Net was analysed in two steps.

1. Accuracy of Disparity Predictions on Validation Datasets

   At the end of each training stage, all validation datasets from Scene Flow "Driving", Scene Flow "Flying Things 3D" and KITTI 2015 were used to evaluate disparity prediction accuracy of the ADSR-Net.

2. Accuracy of Disparity Predictions on Additional Datasets

   Upon completion of all 3 training stages, additional datasets (i.e., Middlebury 2014, KITTI 2012 and ETH3D datasets) were used to check the accuracy of the ADSR-Net when presented with unfamiliar data.

Note: *Evaluation on Middlebury, ETH3D and KITTI 2012 is sufficient to validate the generalization capabilities of the ADSR-Net. However, the evaluation (Step 4) aims to introduce more challenges through stereo images captured by using a custom-built stereo camera.*

7.5.3.1 Accuracy of Disparity Predictions on Validation Datasets

At the end of each training stage, the ADSR-Net was used to predict disparity maps for the validation image pairs to validate the improvements in accuracy. Although the training was conducted at a small cropped-image size, the evaluation of disparity predictions was conducted at full image resolution. The predicted disparity maps were saved and used for qualitative and quantitative analysis.

7.5.3.1.1 Qualitative Analysis Disparity Maps on Validation Datasets

Figure 7.17 shows some example dense disparity maps produced by the ADSR-Net for stereo images from the "Flying Things 3D" validation data, just after finishing the extended training phase. A close visual inspection of the results in the figure, reveals that the ADSR-Net has managed to capture the fine details in both foreground (red) and background (blue) despite the challenging inputs. Furthermore, the foreground objects in ground-truth disparity maps match the same in the ADSR-Net output indicating that the maximum disparity has not been underestimated.

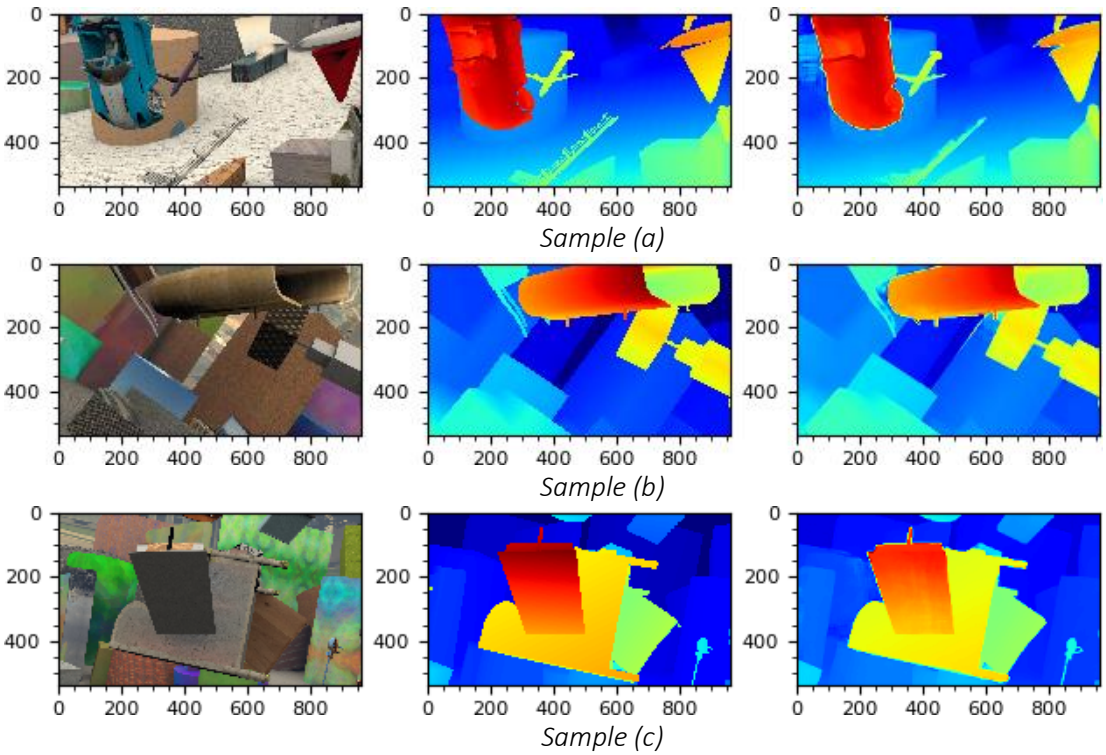


Figure 7.17: Three sample disparity maps produced by the ADSR-Net after the extended training stage which was conducted using the Scene Flow "Flying Things 3D" dataset. The shown maps have been selected from the ADSR-Net predictions for the validation image pairs. The leftmost image in each row shows the reference image (left). The disparity map in the middle shows the ground-truth disparity for the given stereo image pair. The disparity map on the right, shows the ADSR-Net prediction.

Similar results can be observed in the disparity maps obtained after the fine-tuning stage using the validation images from the KITTI 2015 dataset. Three sample disparity output maps from the same are shown in Figure 7.18. A quick visual inspection confirms that the ADSR-Net has successfully captured the foreground and background objects correctly. It is important to note that the ground-truth disparity maps for the KITTI 2015 test data have not been shown in the figure due to being sparse in nature and covering only about 2/3 of the image along the vertical dimension. However, the monocular cues available for the naked eye in the reference images are sufficient for the visual verification of finer details in the output disparity maps. For example, details of the trees in "Sample (b)" of Figure 7.18 have been captured well in the output disparity maps. However, a quantitative analysis of the output disparity maps is required to reliably assess the accuracy of the disparity maps.

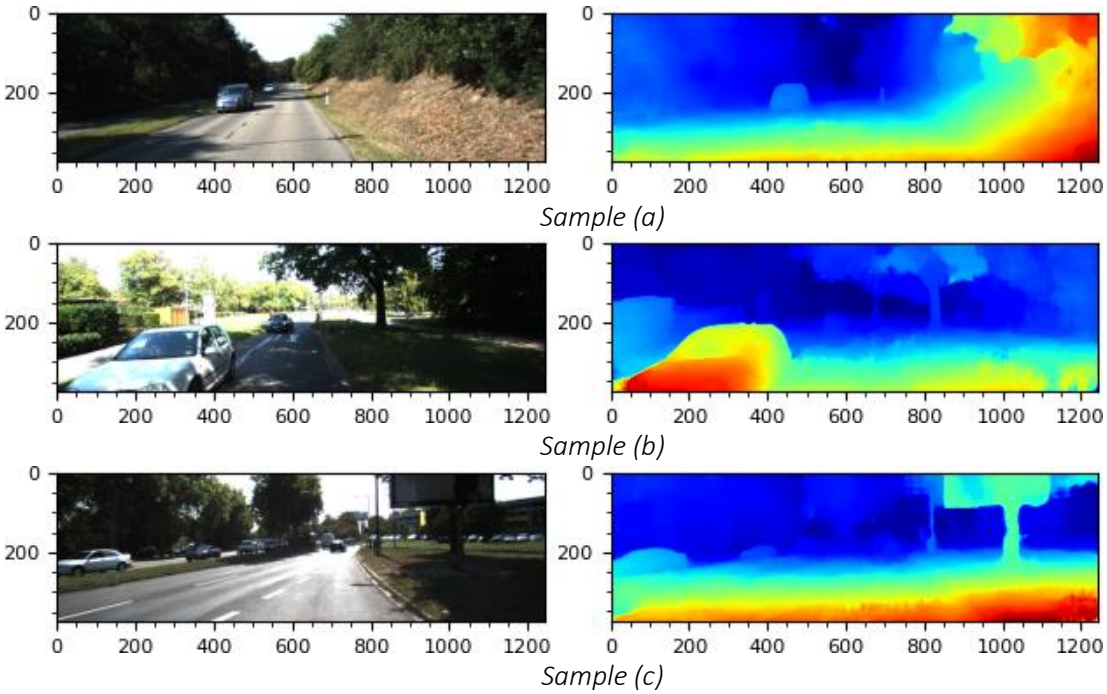


Figure 7.18: Three sample disparity maps produced by the ADSR-Net after fine-tuning with stereo image pairs from the KITTI 2015 stereo dataset. The shown maps have been selected from the ADSR-Net predictions for the validation image pairs. The leftmost image in each row shows the reference image (left). The disparity map on the right, shows the ADSR-Net prediction.

7.5.3.1.2 Quantitative Analysis of Disparity Maps on Test/Validation Datasets

There are many stereo vision related evaluation metrics which can be used to quantitatively evaluate the output disparity maps against the ground-truth data. For the current evaluation, the following two metrics are used.

1. Mean Error/End-Point-Error (EPE)

   The mean error estimates the average/mean value of the absolute difference between the estimated and expected (ground-truth) disparity maps. The value is given in pixels which provides an estimate of how much on average, the two disparity maps (predicted vs. ground-truth) differ from each other. When analysing datasets, the value was computed as an average over the entire dataset.

2. Mean 3-Pixel Error

   The 3-Pixel Error estimates the percentage of pixels in which the estimated disparity values differ by more than or equal to 3 pixels compared to the ground-truth disparity. This was also computed as an average over the entire dataset.

For dense disparity estimations, both the above metrics can be estimated based on the total number of pixels in the image. However, when the ground-truth data is semi-dense or sparse, using the total number of pixels with valid ground-truth disparity values can provide more meaningful results. Hence, the same is used in the analysis in the current section.

The ADSR-Net internally down-samples the input images by a factor of 4 along the length and height dimensions and up-samples the disparity prediction at the end of the stereo pipeline by the same factor. Therefore, an error of 1 pixel at the cost volume resolution (which is $1/4^{th}$ of the image resolution) corresponds to an error of 4 pixels at the original image resolution. Therefore, the 3-Pixel error can provide an indication of the percentage of pixels with their disparity estimations within the 1-pixel error margin at the cost volume resolution.

The Table 7.5 summarizes the average/mean values of the two metrics (3-Pixel error and Mean Error) for the validation datasets. A quick look at the 3-Pixel error shows that it remains between 9 to 13 percent. In other words, 87 to 91 percent of predictions have an error of less than 1 pixel at the cost volume resolution. The most important finding from the summary of the results is the fact that the ADSR-Net has managed to keep the mean error across all datasets around 2 pixels (approximately) while estimating the maximum disparity automatically. At this point, it needs to be

stated that the ADSR-Net has not been optimized for any dataset. Instead, it has been designed and trained to achieve autonomy during disparity inference with the ability to generalize across varying input data while using minimal computational resources.

Test Dataset	Image Type	No of Images	GT Type	Image Size	Mean 3-Pixel Error (%)	Mean EPE (px)
Scene Flow Driving	*Colour RGB*	*120*	*Dense*	*960x540*	*12.92*	*2.19*
Scene Flow Flying Things 3D	*Colour RGB*	*4478*	*Dense*	*960x540*	*9.07*	*2.14*
KITTI 2015	*Colour RGB*	*20*	*Sparse*	*~1242x375*	*12.73*	*1.88*

Table 7.5: Evaluation results obtained with 3 validation datasets. Mean error or the average end-point-error over the dataset and the mean 3-Pixel error for each dataset are mentioned along with other dataset specific information. The metrics have been estimated over the pixels with ground-truth data only.

Important: ADSR-Net has not been optimized for any dataset to achieve a rank on a stereo vision benchmark. Instead, it has been designed and trained with the objective of achieving autonomy during disparity inference with the ability to generalize across varying input data while using minimal computational resources.

7.5.3.2 Accuracy of the Disparity Predictions on Additional Datasets

For practical applications, it is much preferable to have deep learning algorithms that can predict disparity maps for data which is different from what they have been trained on. Although the evaluations on validation data provide clues about the disparity estimation accuracy of the network on unseen data, validation data still has a lot in common with the training data because of originating from the same datasets. Therefore, the best way to evaluate the accuracy of the ADSR-Net on unfamiliar data is to conduct tests with additional datasets like ETH3D, Middlebury and KITTI 2012. These datasets have been specifically chosen to introduce as much variation in input as possible which poses the following challenges to the ADSR-Net.

1. Differences in image types

   The ADSR-Net has been trained exclusively on RGB colour images from the Scene Flow and KITTI 2015 datasets. However, ETH3D and KITTI 2012 provide grayscale images to be used for testing which can introduce additional challenges to the ADSR-Net algorithm.

2. Differences in image sizes

   The ADSR-Net has been trained with fixed-sized cropped images extracted from datasets with a reasonably fixed image size across all image pairs. In contrast, Middlebury 2014 and ETH3D datasets have a large variance in image sizes to challenge the ADSR-Net in terms of the size and scale of the objects.

3. Difference in scene structure

   A key advantage of using diverse datasets is the ability to test the algorithm on scenes with different scene structures. This allows to test the ADSR-Net against dissimilar proximity levels especially with respect to the close-range objects.

4. Difference in stereo baseline

   As seen in chapter 1, the baseline of a stereo camera affects the maximum disparity. Therefore, the additional datasets introduce further variations in input data to the ADSR-Net as they have been captured with different stereo cameras with different baseline distances.

5. Short and Long-Range Views of the Same Scene

   The ETH3D dataset contains stereo images with multiple views (short range and long range) of the same scene which can be used to check if the ADSR-Net is able to correctly determine the obvious differences in maximum disparity.

7.5.3.3 Qualitative Analysis of Results on Additional Datasets

The results shown in Figure 7.19 through Figure 7.21 contain disparity predictions by the ADSR-Net for sample stereo image pairs from the three datasets: ETH3D, Middlebury and KITTI 2012, respectively. For every image pair, the figures include the reference image on the left, the ground-truth disparity map in the middle (except in Figure 7.21) and the ADSR-Net prediction on the rightmost column. A quick visual comparison against the ground-truth disparity maps confirms that the ADSR-Net has been able to handle varying inputs from different datasets accurately in terms of the details captured in the disparity maps. For example, in each of the scenes, the closest

objects to the camera have been clearly identified suggesting that the maximum disparity values have not been underestimated.

7.5.3.3.1 ETH3D Dataset

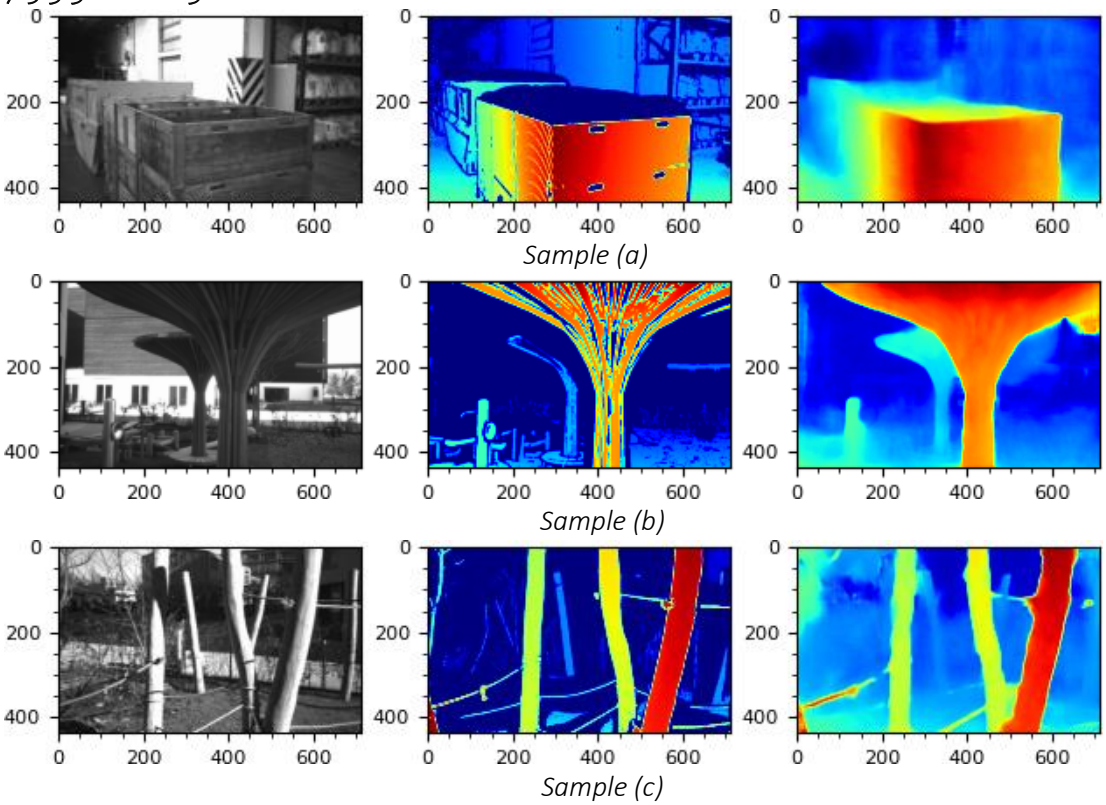


Figure 7.19: The ADSR-Net disparity predictions for 3 sample stereo image pairs from the ETH3D dataset. The results have been obtained using the ADSR-Net model trained with Scene Flow "Driving", "Flying Things 3D" and fine-tuned with the KITTI 2015 dataset. The leftmost image in each row shows the reference image (left) of the stereo image pair. The ground-truth disparity map is shown in the middle and the ADSR-Net output on the right. Results indicate reasonable disparity estimations for unfamiliar data.

ETH3D samples shown in Figure 7.19 present special challenges to the network. For example, all three scenes from sample (a) to (b) indicate insufficient lighting with some regions being clearly darker than the rest. In addition, the scene in Figure 7.19 (b), has its foreground object spanning the whole image width which tests the algorithm's ability to predict disparity maps in a scale independent manner. Nevertheless, the ADSR-Net has been able to handle the challenging scenarios which validates the success of the model design and the training process. However, the lack of details in the foreground objects such as the holes in the box in sample (a), vertical embossed stripes in sample (b) and thin branches in sample (c), indicate over-smoothing by the regularization module which is a drawback.

7.5.3.3.2 Middlebury 2014

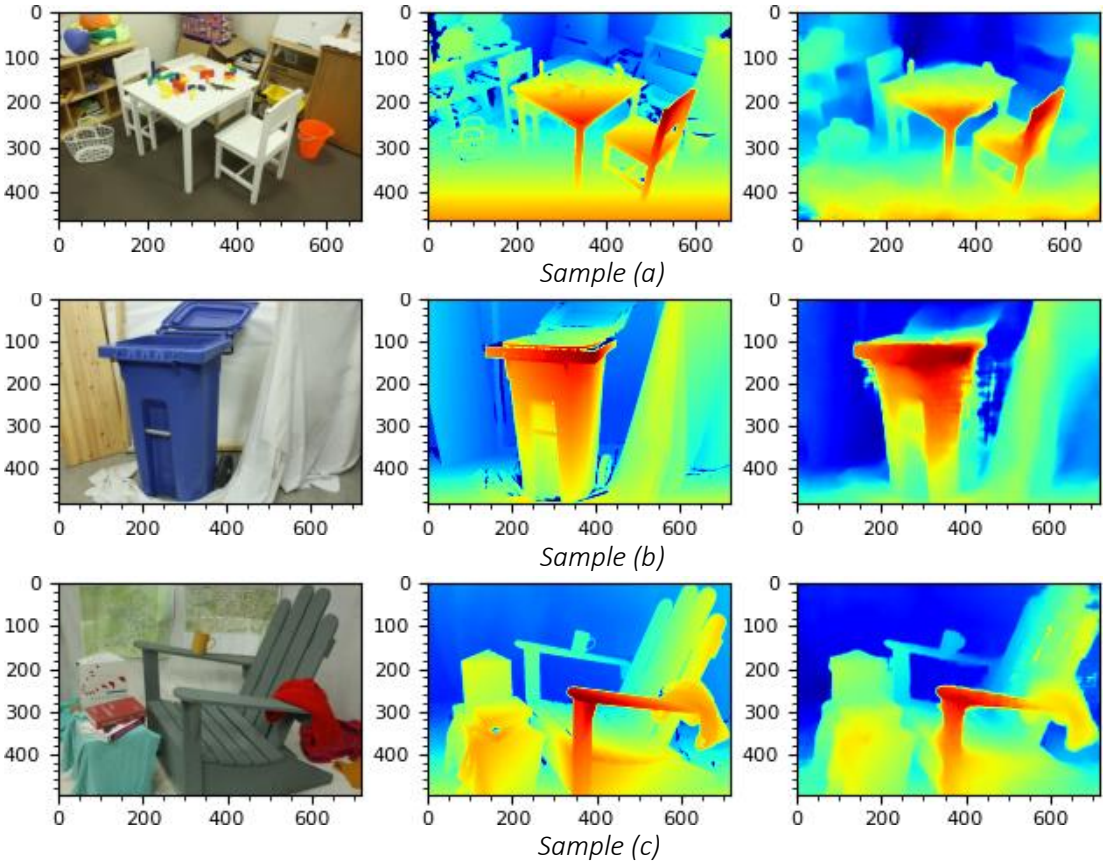


Figure 7.20: ADSR-Net disparity predictions for sample stereo image pairs from the Middlebury 2014 dataset. Reference left images are shown on the left with the ground-truth disparity maps in the middle and the ADSR-Net predictions on the right. From a qualitative standpoint, the ADSR-Net has produced disparity maps with reasonable accuracy despite noticeable errors close to the object boundaries.

Sample disparity outputs for the Middlebury 2014 dataset shown in **Figure 7.20** include close range and well-lit scenes in which the distribution of ground-truth disparities is skewed towards the maximum disparity of the scene. The reference images in **Figure 7.20** indicate the presence of texture-less areas which usually pose a challenge to the stereo vision algorithms. Nevertheless, the ADSR-Net has been able to estimate reasonably accurate disparity maps from a qualitative standpoint due to capturing the details of most of the objects when compared to the ground-truth disparity maps. However, disparity estimations around the edges of the objects lack in clarity, leading to blurry edges (e.g., recycle bin in Sample (b)).

7.5.3.3.3 KITTI 2012

Despite much wider variation in object proximity in input images (compared to the Middlebury 2014 data), results from the KITTI 2012 dataset shown in Figure 7.21, indicate a clear segregation of the background and the foreground objects with much accurate details in the predicted disparity maps.

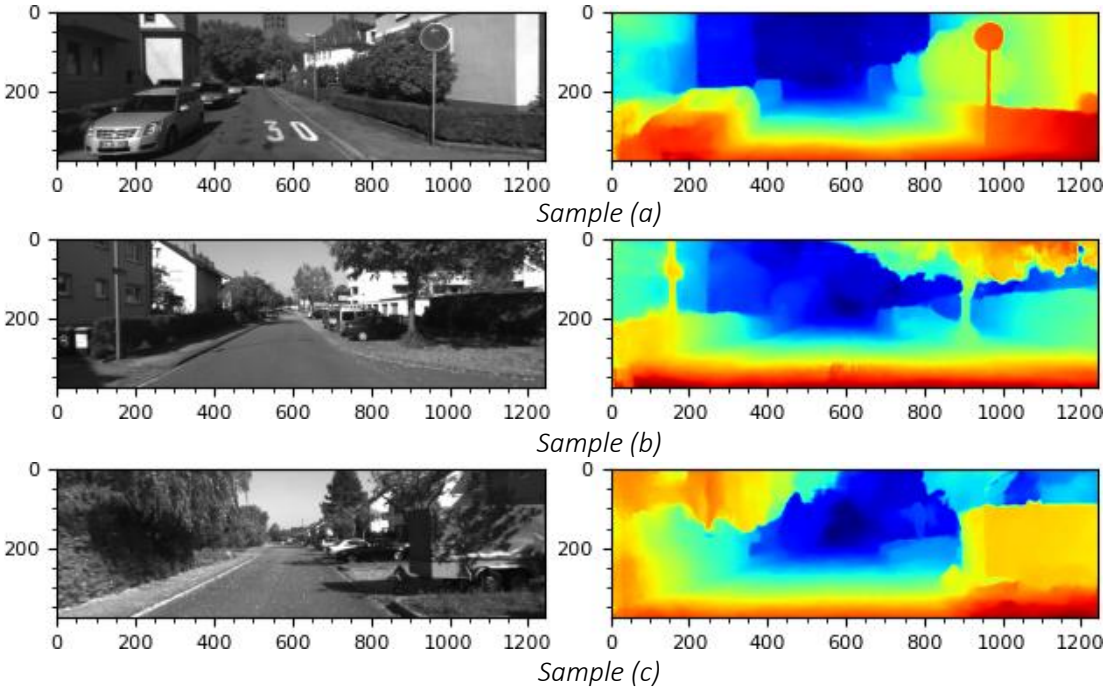


Figure 7.21: Sample ADSR-Net disparity predictions for KITTI 2012 stereo image data. Reference images are shown on the left with the ADSR-Net predictions shown on the right.

For example, the road sign in sample (a) has been clearly identified including the thin pole on which it is mounted. The same is true for the object in sample (b) which appears to be a lighting post. The overhanging tree branches in sample (c) can be clearly distinguished from the background. The trailer which is parked under the tree in sample (c) which is obscured by the shadows, is still clearly identifiable in the disparity map. Other darker, shadowy regions in all three samples do not appear to have caused the algorithm to register noticeable mismatches.

7.5.3.4 Quantitative Analysis of the Results on Additional Datasets

Like the analysis done for the data from the validation datasets, a quantitative analysis is required to gain better insights when studying the results for the additional datasets. Table 7.6 shows the mean 3-Pixel error and the end-point-error estimates for the 3 additional datasets. As it can be seen from the results, the average values reported by the evaluation metrics are slightly higher than the same reported earlier (with the

validation datasets) except in the case of ETH3D dataset. According to the mean 3-Pixel error, more than 97% of the disparities estimated by the ADSR-Net on ETH3D dataset, have an error of less than 1 pixel at the cost volume resolution (quarter resolution) despite the algorithm encountering these images for the first time. In contrast, the results for the Middlebury and KITTI 2012 datasets have shown an increase in the end-point-error and the mean 3-Pixel error. However, over 80% of the disparities estimated by the ADSR-Net on Middlebury 2014, still have an error of less than 1 pixel at cost volume resolution. For the KITTI 2012 dataset, the same is true for 88% of the pixels. Most importantly, the observed results have been obtained without any user-defined parameters.

Dataset	Image Type	No of Images	GT Type	Image Size	Mean 3-Pixel Error (%)	Mean EPE (px)
ETH3D	*Gray*	*27*	*Dense*	*Variable*	*2.65*	*0.63*
Middlebury 2014	*RGB*	*15*	*Dense*	*Variable*	*18.6*	*3.34*
KITTI 2012	*Gray*	*194*	*Sparse*	*1226x370*	*11.49*	*2.66*

Table 7.6: Results of the quantitative analysis using 3 additional datasets. Despite the differences in image type (colour/grey), scene structures and image size, the ADSR-Net did not require the user to set any parameter values when obtaining these quantitative results.

7.5.4 EVALUATION STEP 4: Evaluation on User-Captured Stereo Images

The 4th and the final step in the ADSR-Net evaluation was carried out using a custom-built stereo camera system (i.e., Figure 7.22). It is comprised of two FLIR Chameleon 3 monocular cameras fitted with variable focal-length FISHEYE lenses mounted on a handmade stereo camera rig. The details of the cameras and lenses are provided in *Appendix F*. Since the individual monocular cameras have not been specifically matched for stereo applications, the images produced by the camera can be challenging to any stereo vision algorithm. Furthermore, the FISHEYE lenses with manually variable focal length and iris can pose more challenges when matching. Unlike the datasets used for evaluations earlier (captured with well-matched stereo cameras or simulated as such), custom stereo images captured with an improvised stereo system can help test the ADSR-Net on more realistic scenes under non-ideal conditions (which the real-time stereo algorithms are most likely to encounter).

Figure 7.22: Stereo camera system used for capturing custom stereo images. The setup contains two FLIR Chameleon CM3-U3-13S2C-CS cameras fitted with Computar A4Z2812CS-MPIR FISHEYE lenses (with manually variable focal length 2.8mm-10mm and iris) mounted on a hand built stereo camera rig.

7.5.4.1 Stereo Calibration and Camera Preparation

Images captured by the individual cameras must first be undistorted and rectified before sending them through the ADSR-Net model. This was achieved using a one-time calibration process which was conducted before the rectification process. The process involved capturing a series of stereo images of a checker-board pattern in different orientations. The resulting images were used to obtain camera parameters so that the images could be undistorted (i.e. removing the fisheye distortion). The undistrorted image pairs were then used to perform stereo calibration. Stereo calibration parameters as well as the individual camera parameters were saved for later use when capturing new images. Upon capturing a new image pair using the custom stereo setup, the images were first undistorted by using the saved camera parameters before performing stereo calibration to obtain a calibrated pair of strereo images. Once calibrated, it was not required to change or set any parameters except following any structural changes to the camera rig in which case a recalibration of the sytem was requried.

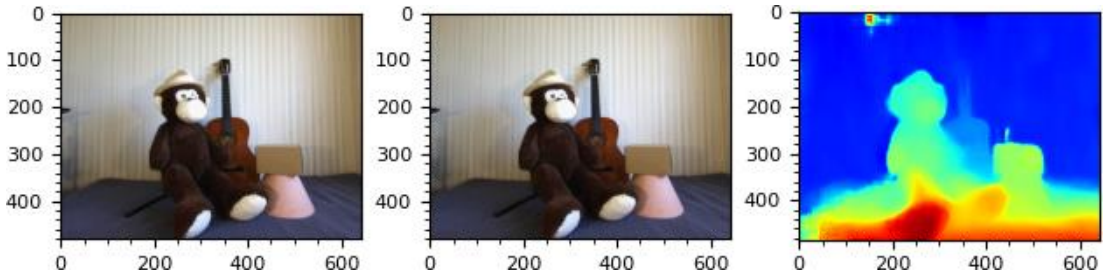
The other important aspects related to the image capturing process include exposure and gain selections for cameras. For structured static scenes, it is possible to use automatic gain and exposure settings because of the two cameras having the same internal algorithms for automatically adjusting the same (i.e. gain and exposure settings). However, in the case of stereo image sequences captured by moving the camera or objects, automatic gain and exposure can cause significant differences in

the output images. Therefore, in the experiments outlined in this section, images of some structured scenes (i.e. indoor) have been captured with automatic settings whereas the image sequences have been obtained with fixed exposure and gain settings only. Furthermore, the cameras had to be locked into a fixed baseline position before the calibration process and maintained fixed throughout (despite the baseline of the custom stereo rig being variable). Figure 7.23 shows the custom-built camera being used to capture a pair of images of a static scene.

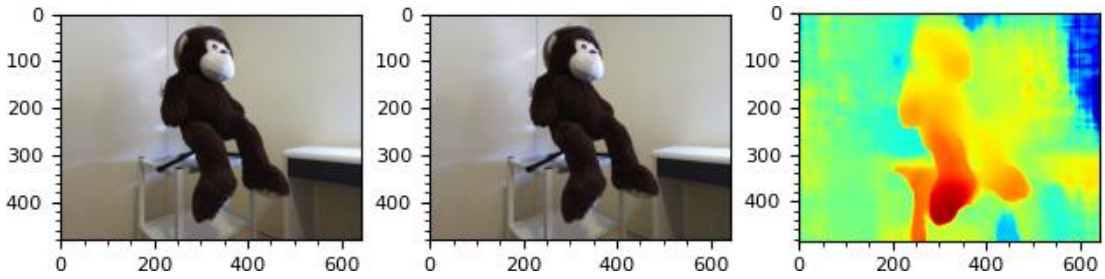
Figure 7.23: A static image being captured by using the custom-built stereo camera system.

7.5.4.2 Results on Structured Scenes

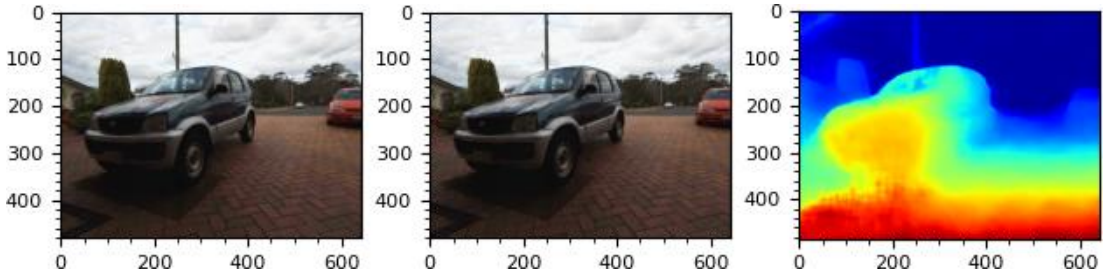
Figure 7.24 includes three calibrated stereo image pairs for 3 static scenes, captured using the custom stereo camera system, along with the disparity maps produced by the ADSR-Net. The static scene 1 in the figure contains a large soft-toy and uniquely identifiable objects placed infront of a textured background. The shapes of the objects and the soft-toy are clearly identifiable in the output disparity map produced by the ADSR-Net. The smooth transition of disparities, as evident from the gradual change in color gradient in the output disparity map (starting from the closest object points in the scene), indicates that the ADSR-Net has been able to estimate a suitable maximum disparity value for the scene automatically.



(a) *Static Scene 1*



(b) *Static Scene 2*



(c) *Static Scene 3*

Figure 7.24: Disparity predictions for custom stereo camera images of 3 static scenes captured using the stereo camera rig shown earlier. The left image is shown on the left with the right image in the middle. The disparity map predicted by the ADSR-Net is shown on the right.

The second example in Figure 7.24 (b) shows another static scene showing the same soft-toy figure in front of a backdrop comprised of glossy and texture-less areas with shadows and reflections. However, when analysing the disparity map of the scene produced by the ADSR-Net, it is clear that the contours of the foreground object (i.e., soft-toy figure) have been well identified by the algorithm. It is also noteworthy how the leading limb of the soft-toy figure is correctly associated with the largest disparity (as observed from the colour gradient in the disparity map). However, the disparity map also shows some noticeable errors in the texture-less regions and pixels that correspond to surfaces with glossy texture.

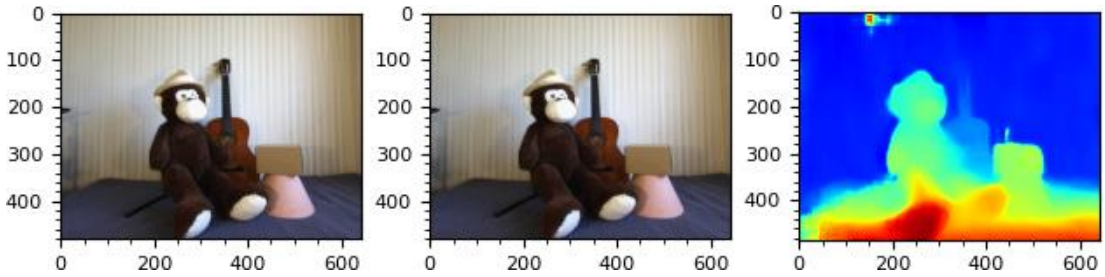
The third static scene in Figure 7.24 (c), shows the stereo images for an outdoor scene with vehicles parked on a driveway. As it can be noticed from the disparity map produced by the ADSR-Net, the contours of the objects (e.g., Cars, lighting pole, tree etc.) are clearly identifiable in the output. It is also important to note that the disparities associated with background do not include obvious errors. This can be attributed to the clearly distinguishable texture in the background objects including the trees and the clouds.

However, disparity estimates of the background objects which are located far away from the cameras, lack in fine details. This is due to the inherent depth resolution issue in stereo (Equation (1.1)). Furthermore, the repetitive patterns on the foreground (e.g., driveway) have not caused significant errors although the darker shadow in front of the car has resulted in noticeable errors in the corresponding area of the output disparity map. Despite all scenes in Figure 7.24 (a) to (c) having unevenly lit areas (with a generally gloomy outcast), the ADSR-Net appears to have been able to handle uneven brightness and contrast well.

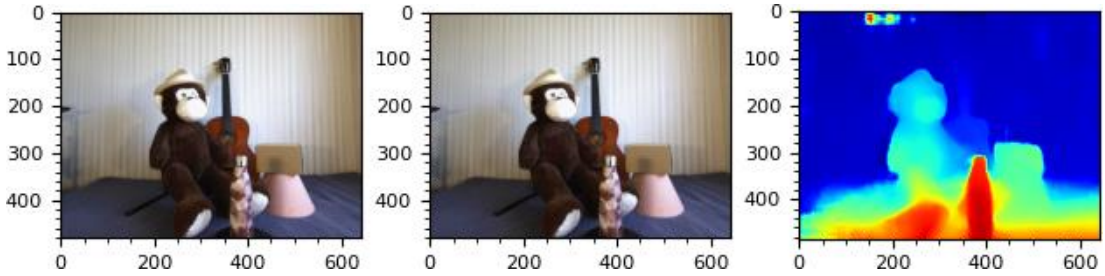
7.5.4.3 Changing Maximum Disparity by Adding Foreground Objects

Since the maximum disparity for a scene is directly related to the proximity of the foreground objects, the robustness of the ADSR-Net can be further challenged by manipulating the foreground objects of a scene. Figure 7.25 illustrates a scenario in which a new object is introduced to the foreground. Figure 7.25 (a) is the same static scene (Scene 1 in Figure 7.24) shown earlier and Figure 7.25 (b) depicts the same scene with a new object (a bottle) placed in the foreground.

As expected, the ADSR-Net disparity prediction for the augmented scene shows the new foreground object (bottle) as being closest to the camera than the leading limb of the soft-toy figure which was the closest object earlier (before the foreground of the scene was manipulated). This coupled with the absence of obvious errors in the foreground, indicates that the ADSR-Net has been able to readjust the maximum disparity to suit changes in scene structure.



(a) *Static Scene 1*



(a) *Static Scene 1 + new close-range object (bottle)*

Figure 7.25: Introducing a close-range object to a static scene. When a new object (i.e., a bottle) was introduced as a foreground object to the static scene 1 (a), the ADSR-Net was able to automatically increase its maximum disparity to incorporate the new object in the disparity prediction as shown in (b).

7.5.4.4 Results on Image Sequences

As observed during the EVALUATION STEP 1, the advantages of automatic maximum disparity estimation become more evident when processing image sequences. In image sequences with larger variations in object proximity, automatic maximum disparity estimation can lead to significant savings in computational time compared to when using a fixed maximum disparity. However, the stereo image sequences used for the earlier experiments were sourced from standard stereo image datasets. To harness the capabilities of automatic maximum disparity estimation in a real-world application, the ADSR-Net must first be able to produce accurate disparity maps for image sequences captured using an imperfect stereo camera. Such an image sequence is shown in Figure 7.26. The figure shows 5 stereo image pairs with the respective disparity predictions by the ADSR-Net. Images contain five consecutive scenes captured when panning the stereo camera from left to right starting from the static scene 3 (in Figure 7.24) earlier. As it can be seen from the image sequence, the scene structure changes with the camera movement. Yet, in spite of the changes, the ADSR-Net predictions show that it has been able to capture the changes reasonably well. This demonstrates the generalization capabilities of the ADSR-Net.

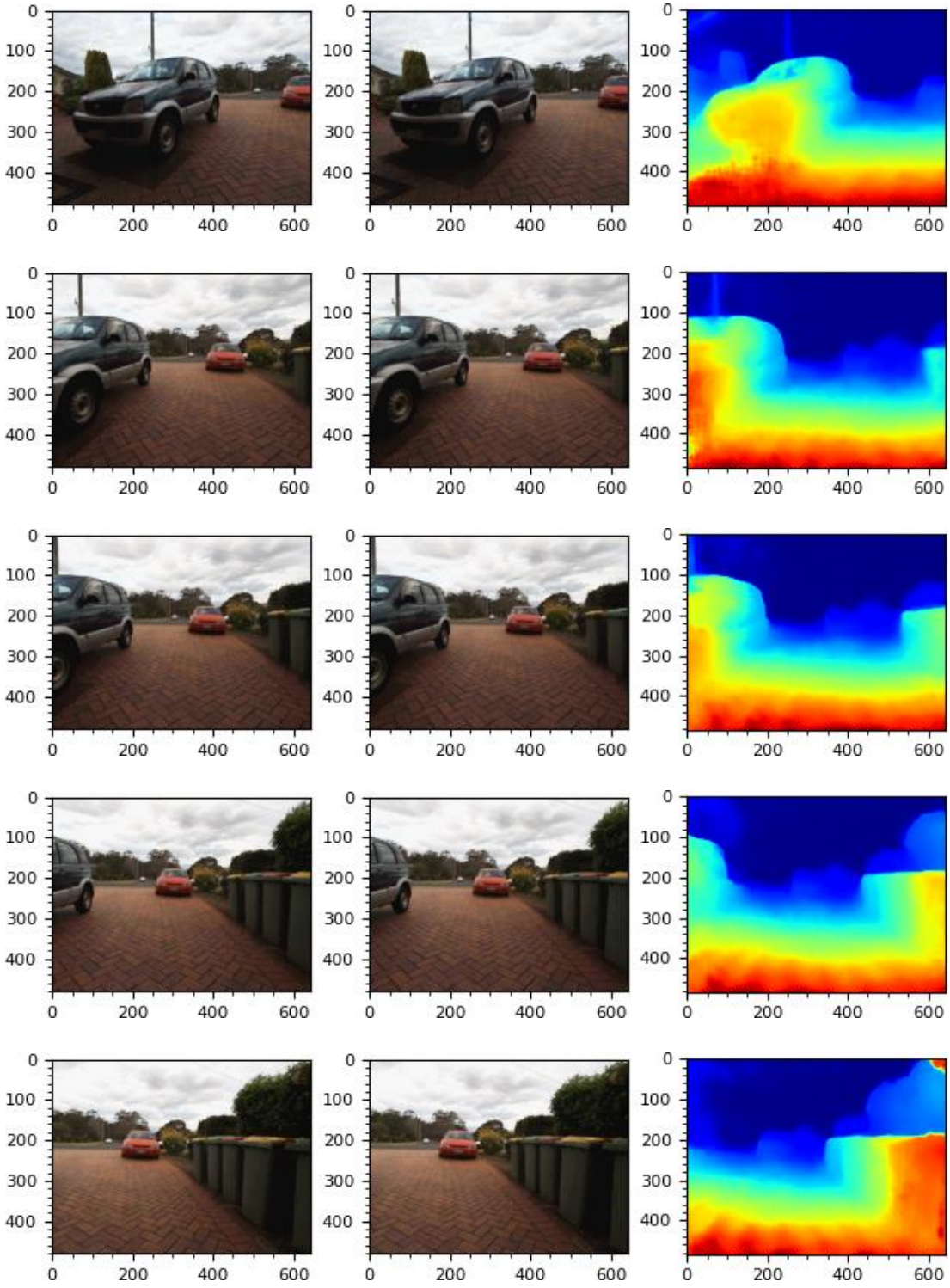


Figure 7.26: ADSR-Net disparity predictions for a sequence of 5 stereo image pairs captured by panning the stereo camera rig from left to right (images from top to bottom correspond to the images periodically captured when panning the stereo camera rig from left to right).

7.5.5 Strengths of the ADSR-Net

The following list of strengths separate ADSR-Net from the other deep learning stereo algorithms.

- ✓ Independence from user's understanding of the scene structure
- ✓ Ability to handle multiple resolution levels without reconfiguration
- ✓ Ability to handle multiple baseline stereo without user intervention

7.5.5.1 Independence from User's Understanding of the Scene Structure

As outlined in Chapter 2, most stereo algorithms (traditional and deep learning-based) require users to specify the maximum disparity based on user's knowledge and understanding of the scene structure. However, the ADSR-Net does not require such inputs during inference. Figure 7.27 shows the ADSR-Net output for two stereo image pairs from the ETH3D dataset which correspond to one close-up view and another zoomed-out view of the same scene. For a typical deep stereo algorithm to produce accurate results for the given stereo images ("Delivery Area 1l" and "Delivery Area1s" of ETH3D), a user must either specify suitable maximum values for each scene separately or select a large value which is at least equal to or more than the expected maximum disparity for the close-up view.

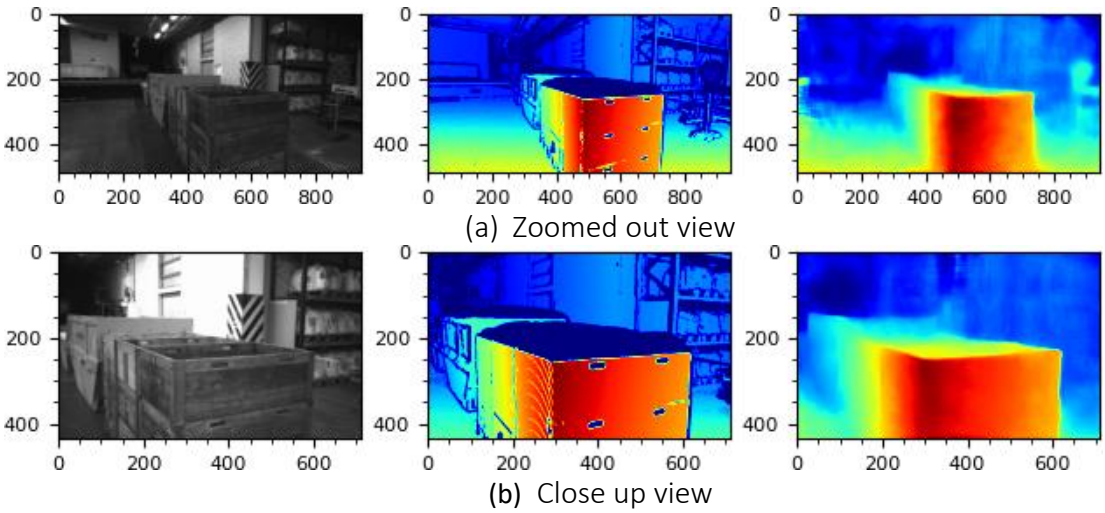


Figure 7.27: Two views of the same scene ("Delivery Area 1l" and "Delivery Area1s" from the ETH3D dataset). The grayscale reference images are shown on the left with the ground-truth disparity maps in the middle. The ADSR-Net predictions are shown on the right. The ADSR-Net has automatically detected a suitably larger maximum disparity value for the close-range scene compared to the zoomed-out version.

The ADSR-Net on the other hand has produced accurate disparity maps for both scenes (on visual comparison with the ground-truth disparity maps) without user-

specified parameters. Another similar scenario is given in Figure 7.28. When predicting disparity maps across four scenes, the ADSR-Net did not require any user intervention other than providing the two input images. However, when compared to the ground-truth disparity maps, the ADSR-Net has not only produced disparity maps which look similar to the ground-truth but match them quantitatively as well (i.e., due to being part of the ETH3D dataset, these image pairs were included in the summary of results presented in Table 7.6).

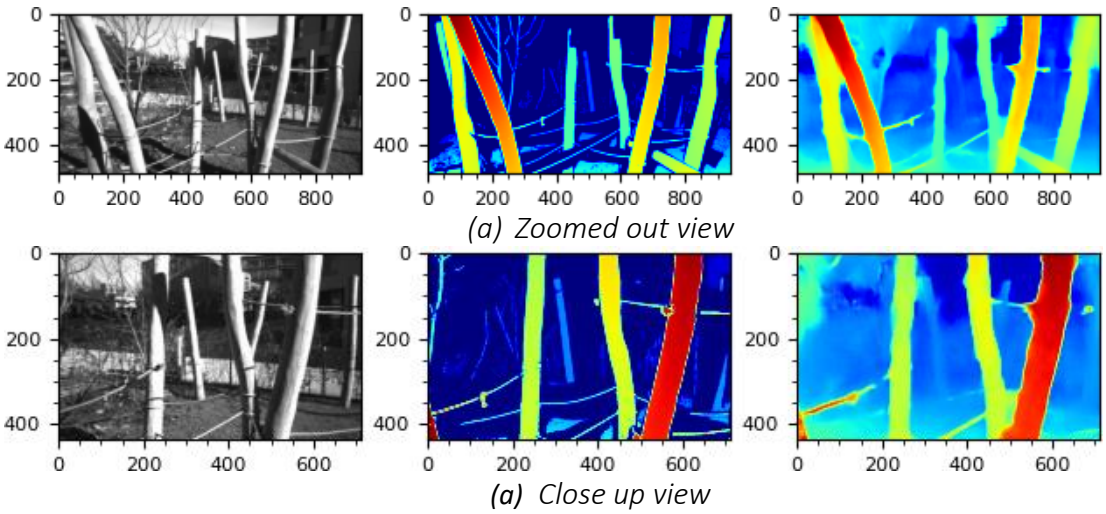
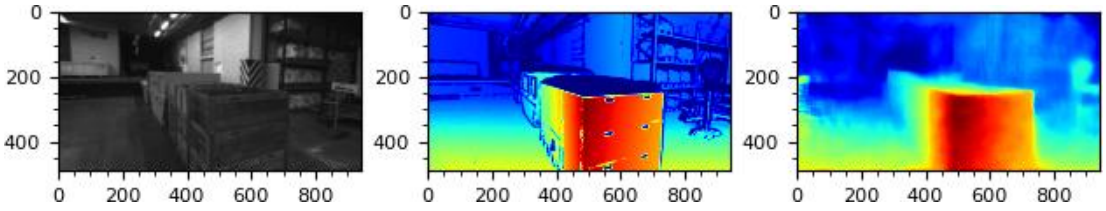


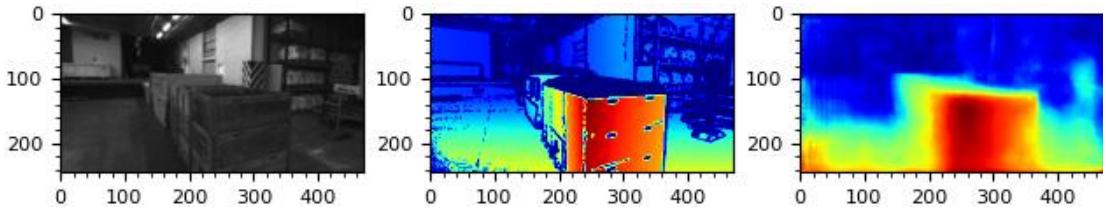
Figure 7.28: Two more views ("Playground 3l" and "Playground 3s") from the ETH3D dataset depicting a close-up view and a zoomed-out view of the same scene. The grayscale reference images are shown on the left with the ground-truth disparity maps in the middle. The ADSR-Net predictions are shown on the right. The ADSR-Net has automatically adjusted the maximum disparity value for the close-up scene.

7.5.5.2 Ability to Handle Multiple Resolution Levels without Requiring User Inputs

Figure 7.29 (a) shows another example from the grayscale ETH3D stereo dataset at original resolution along with the ground-truth disparity map and the corresponding ADSR-Net prediction. Figure 7.29 (b) shows the same at half resolution. In both cases, the ADSR-Net has been able to produce disparity maps which are qualitatively similar to the ground-truth disparity maps. In addition, the average error for the disparity estimates at full and half resolution was found to be 0.53 and 0.43 pixels respectfully (in the regions where the ground-truth disparity maps contain valid disparities).



(a) At original resolution



(b) At half resolution

Figure 7.29: The ADSR-Net disparity predictions for the "Delivery Area 1l" scene from the ETH3D data at two different resolution levels (original resolution [a] and half resolution [b]). The leftmost image shows the reference (left) image with the disparity ground-truth in the middle. The rightmost image shows the disparity prediction by the ADSR-Net. Unlike other deep stereo techniques, the ADSR-Net does not require users to adjust maximum disparity value depending on the resolution of the input images.

7.5.5.2.1 Ability to Handle Multiple Baseline Stereo without User Intervention

According to the basic stereo equation (**Equation (1.1)**) given in Chapter 1, varying baseline can change the disparity associated with a given object point. Therefore, when using typical deep stereo algorithms with multiple baseline stereo cameras (e.g., Bumblebee XB3), users must select two different disparity search ranges or maximum disparity values. Even then, depending on the proximity of the objects to the camera, the maximum disparity may have to be adjusted separately. If a single large maximum disparity value is used with the two baselines, the long-range accuracy gains of the larger baseline configuration can be lost due to potential errors introduced by the unnecessarily large maximum disparity.

On the other hand, throughout the evaluation steps from 1 to 4, the ADSR-Net was evaluated on datasets captured using stereo vision cameras with different baseline distances (including the custom-built stereo camera rig). It was shown that regardless of the type of images used, the ADSR-Net was able to produce disparity maps without the user having to specify the maximum disparity explicitly. Therefore, the ADSR-Net can make the disparity estimation process much simpler for multiple baseline configurations.

7.5.6 Challenging Scene Conditions / Phenomena

By analysing the results recorded as part of the evaluation steps from 1 to 4, it was possible to find anomalies in disparity predictions by the ADSR-Net which appear to have a strong correlation with the following:

1. Artefacts due to lens flare
2. Regions with glare in images
3. Low textured regions in stereo images

7.5.6.1 Effect of Lens Flare

For example, **Figure 7.30** shows the end-point-error associated with the ADSR-Net predictions for all 27 stereo image pairs from the ETH3D dataset. Among the values, the end-point-error for the scene labelled "playground_2l" stands out due to the relatively high value.

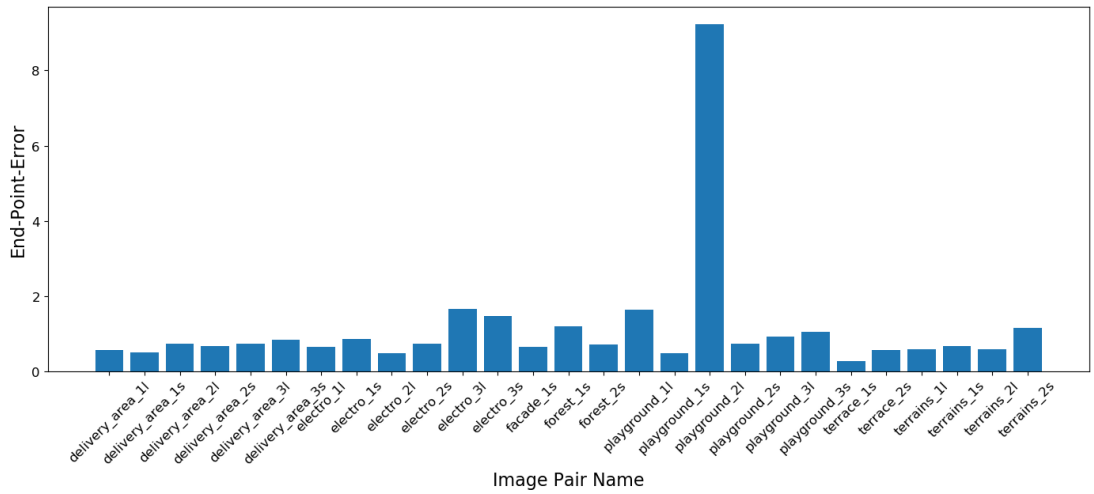
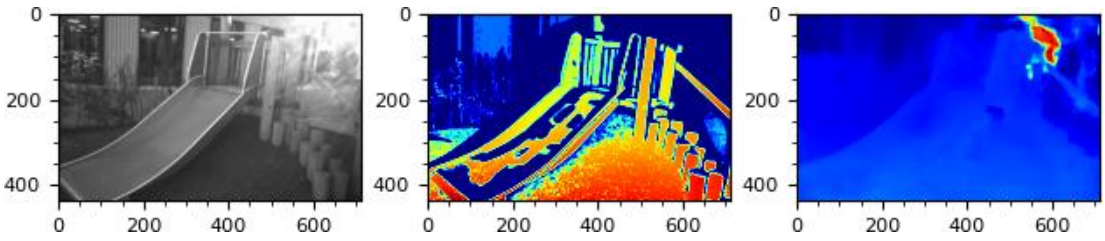


Figure 7.30: A bar-graph showing the variation of the "End-Point-Error" in disparity estimations for stereo image pairs from the ETH3D dataset at original resolution. The end-point-error of the ADSR-Net disparity estimation for the "Playground 2l" appears to be unusually high compared to the others.

A closer inspection of the scene in concern and the disparity prediction by ADSR-Net **(Figure 7.31)** reveals that there are noticeable discrepancies between the predicted and the ground-truth disparity maps for the scene. The exception appears to coincide with the lens flare related artefacts in the input images. As it can be seen from **Figure 7.31**, there is a cluster of disparity predictions in the corresponding area, which appear to be unusually larger in value. This together with the relatively accurate contours in the rest of the disparity predictions, indicate that the ADSR-Net has overestimated the maximum disparity for the region with lens-flare related artefacts.



(a) Reference Image | (b) Ground-truth Disparity | © ADSR-Net Prediction

Figure 7.31: Lens flare induced artefacts in the "Playground 2l" image pair of the ETH3D dataset appear to coincide with the unusually large and erroneous disparity predictions by the ADSR-Net.

A quick comparison between the series of maximum disparity predictions for all 27 ETH3D stereo image pairs and the respective ground-truth maximum values (**Figure 7.32**) shows how much the ADSR-Net has over-estimated the maximum disparity for the scene labelled "playground_2l". However, the unusually high maximum disparity has not resulted in abrupt termination of the algorithm due to the lack of GPU memory. This shows the importance of the memory efficient design of the ADSR-Net.

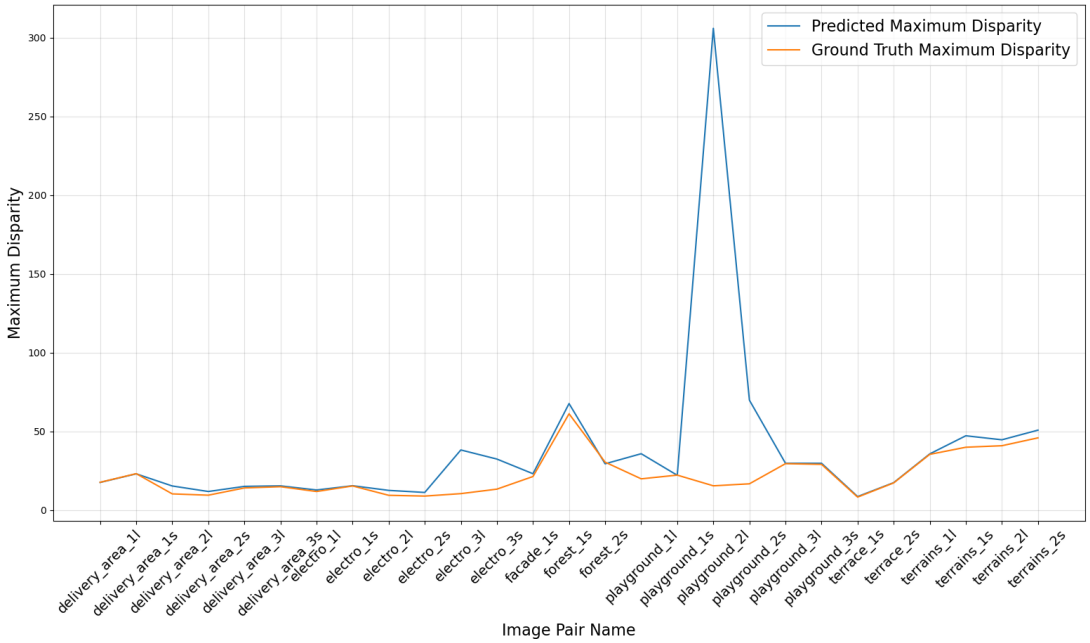
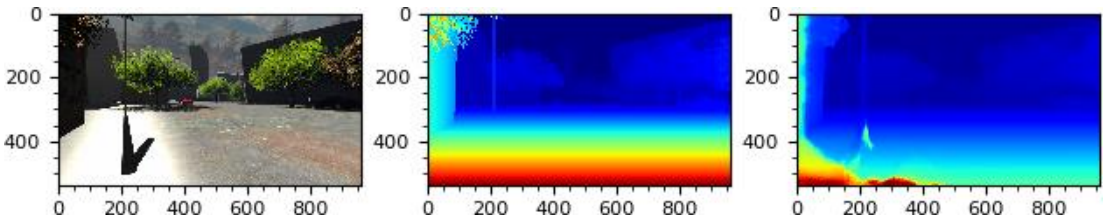


Figure 7.32: A comparison between the maximum ground-truth disparity and the maximum disparity predicted by the ADSR-Net for all 27 stereo image pairs from the ETH3D dataset. The maximum disparity for the scene labelled "playground_2l" as determined by the ADSR-Net (based on SNCE) is significantly larger than the ground-truth maximum. In other scenes, the ADSR-Net has predicted the maximum disparity closely.

7.5.6.2 Presence of Image Glare

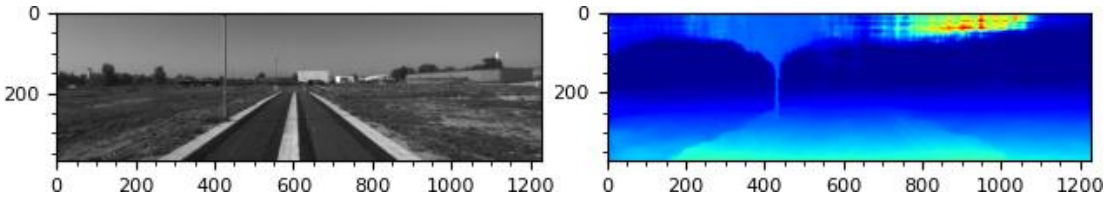
A similar procedure on other datasets revealed that, image glare in general has a negative effect on the disparity estimation accuracy of the ADSR-Net. For example, Figure 7.33 illustrates how glare in combination with a shadow can result in disparity estimation errors. According to the predicted disparity map, background objects have been clearly identified even in some areas affected by glare (e.g., ground plane). In contrast, the leftmost part of the glare affected region in the foreground, spans out of the stereo area making it harder for the algorithm to fill in the disparities in the affected area. An in-depth analysis is required to establish the correlation conclusively. However, such an analysis is beyond the scope of the thesis due to the universal nature of the negative effects of glare in computer vision techniques.



(a) Reference Image *(b) Ground-truth Disparity* *(c) ADSR-Net Prediction*

Figure 7.33: Glare induced errors in disparity predictions by the ADSR-Net (An example from the Scene Flow "Driving" dataset)

7.5.6.3 Low Textured Areas



(a) Reference image *(b) ADSR-Net Output*

Figure 7.34: An example from KITTI 2012 dataset in which a low-textured area has introduced errors to the disparity map produced by the ADSR-Net.

Low textured areas which affect the other stereo matching techniques, appear to have a negative impact on the ADSR-Net as well. This can be seen from the ADSR-Net output for a sample stereo image pair from the KITTI 2012 dataset shown in Figure 7.34. The area occupied by the sky in Figure 7.34 has very little texture. Therefore, the algorithm has produced an erroneous disparity map with the largest disparities located in the

low-textured region. In addition, it appears as if the ADSR-Net has attempted to fill in the remaining areas which has resulted in a utility pole being incorrectly identified as a tree due to the errors propagating from the texture-less region.

7.6 Chapter Summary

This chapter introduced an end-to-end deep learning-based stereo algorithm called ADSR-Net which utilizes the SNCE metric to detect a suitable maximum disparity for a given scene automatically. The new improved architecture which is based on the findings of the previous chapter included a much deeper feature network, a modular cost aggregation network and a layer-wise cost volume creation module with SNCE-based termination capability.

Preliminary tests before training showed that the use of a pre-aggregation network to produce a 3D cost volume leads to a memory efficient network design compared to similar implementations with 4D or higher dimensional cost volumes. However, in the presence of a cost aggregation network an improvised training technique called "clamped training" was required to enforce unimodality of the matching costs at pre-aggregation level.

The chapter also introduced a multi-stage training process using 3 stereo datasets, with the objective of improving the generalization capabilities of the ADSR-Net. At the end of each training stage, the network was evaluated for disparity estimation accuracy both qualitatively and quantitatively. Tests involving image sequences (which were conducted immediately after the initial training stage) showed that the automatic maximum disparity estimation capability of the algorithm can result in up to 50 percent savings in average computational time per stereo image pair compared to when using a fixed cost volume size that fits the population data.

Evaluations on validation data, additional datasets and the stereo images captured with a custom-built stereo camera system, revealed that the fully trained ADSR-Net is able to detect suitable maximum disparity values and predict accurate disparity maps under diverse scene conditions. The ADSR-Net was also able to handle variable resolution levels and multiple baseline distances without users having to specify parameter values explicitly.

Phenomena such as lens flare, glare and low textured areas which affect stereo matching techniques and other machine vision techniques in general, were found to cause errors in disparity estimations by the ADSR-Net. Extensive further evaluations focussed on each phenomenon are required to conclusively determine the exact cause of such errors which is beyond the scope of the current thesis.

Although the ADSR-Net produced qualitatively and quantitatively accurate results across different stereo image datasets (standard and custom), the model has a lot of room for improvement in terms of the network architecture and training. The ADSR-Net model used for the evaluation had only a smaller number of network blocks in the feature extraction and cost aggregation networks in order to maintain trainability on consumer grade GPUs such as GTX1070 and RTX2080. By increasing the number of blocks and by conducting training in high performance computing/multi-GPU environments with more resources, much better results can be achieved.

CHAPTER 08 - CONCLUSIONS AND FUTURE WORK

Introduction

The preceding chapters from 1 to 7 detailed the methodology, experiments, results and evaluations pertaining to the development of a deep stereo network called ADSR-Net (Automatic Disparity Search Range Network) that does not depend on user-specified parameter values to produce disparity maps. The overall process started with the development of the novel SNCE (Sum of New Cost Extrema) metric to serve as the termination criteria for a layer-wise cost volume construction stage. The feasibility of using the metric in CPU and GPU environments was then studied with algorithmic analysis and empirical methods. Upon positive confirmation, the metric was incorporated into an initial deep stereo method to validate the possibility of achieving SNCE convergence. Experiments with stereo image sequences extracted from the Scene Flow "Driving" dataset demonstrated that the SNCE value becomes zero at maximum disparity (or just above the maximum disparity). From this, the new metric was incorporated into an end-to-end deep stereo network which included a feature extraction module, a dynamic layer-wise cost volume creation module and a cost aggregation network. To train the network, a novel training scheme called "clamped training" was created to combine the final dense disparity prediction with an intermediate disparity prediction obtained after the cost volume creation stage.

The ADSR-Net was then trained with Scene Flow "Driving", Scene Flow "Flying Things 3D" and KITTI 2015 datasets in a 3-stage training process targeted at improving the generalization capabilities. Clamped training helped improve the accuracy of the maximum disparity estimations and the disparity maps simultaneously. Extensive post-training evaluations were carried out on the validation datasets, additional datasets (ETH3D, Middlebury 2014 and KITTI 2012) and stereo images captured using a custom-built stereo camera. A qualitative and quantitative analysis indicated accurate disparity predictions by the ADSR-Net under varying scene conditions. The results on stereo image sequences showed a significant improvement in performance compared to the same algorithm with maximum disparity set to the population maximum. This chapter aims to summarize the research work in terms of achieving the objectives, merits of the contributions and future work.

8.1 Achievements in Objectives

As elaborated in Chapter 1 and supported by the findings in Chapter 2, most deep stereo algorithms solve the stereo disparity estimation problem only partially on their own, due to their dependence on user-selected "maximum-disparity" or "disparity search range" parameters. In other words, users partially-solve the disparity estimation problem for the machines by providing the expected range of variation in disparity, manually. The work presented in this thesis was aimed at eliminating the need for users to specify the commonly used "maximum-disparity" parameter in deep stereo networks. The maximum disparity parameter can often be the only such user-specified parameter in many state-of-the-art stereo algorithms (Table 2.1) during disparity inference. Therefore, the automatic maximum disparity estimation in deep stereo is required for achieving complete independence from user's prior knowledge of the scene structure or the lack thereof. To achieve the same, the following objectives were identified in Chapter 1.

1. Developing a methodology to automatically estimate the maximum disparity parameter in stereo vision algorithms without requiring user input.
2. Formulating machine learning friendly metrics which can be used to eliminate the manual configuration of the disparity search range / maximum disparity.
3. Developing a fully autonomous stereo vision algorithm which does not require any pre-configuration by the users during disparity inference.

Chapter 4 introduced the novel SNCE metric which can reliably indicate the occurrence of the maximum disparity for a given scene, based on its value if the matching cost distribution remains unimodal. The analysis in Chapter 4 showed that the SNCE metric met the requirements of the *Objective 1*, even though achieving unimodality of cost distributions with traditional techniques (such as changing the mask size) was not feasible due to the dependency on user judgement and scene conditions. Based on the findings of the literature review in Chapter 2, it was also observed that the unimodality of cost volumes is a pre-requisite for differentiable disparity regression in deep stereo networks. The existence of a common pre-requisite for the SNCE metric and the deep learning stereo algorithms was favourable for the development of a new

stereo technique to determine the maximum disparity automatically, provided the metric could be estimated in a computationally efficient manner.

Chapter 5 showed that the SNCE metric can be estimated efficiently as part of the cost volume construction process in both CPU and GPU environments, without imposing a significant additional computational burden. Most importantly it was seen that the order of growth of the computations for a typical stereo disparity estimation algorithm does not change due to the introduction of the SNCE metric. That paved the way for incorporating the SNCE as part of the forward propagation phase of a deep stereo network directly.

Chapter 6 featured a basic deep stereo network which was able to compute the SNCE metric at each disparity. This was done to check whether the SNCE value converges at maximum disparity or at an appropriate value slightly above it when using a deep stereo network. Experiments using the Scene Flow "Driving" dataset with data augmentation, further enforced the outcome. It was also observed that the accuracy improves in tandem with the stereo matching accuracy of the overall network. Hence, the results in Chapter 6 confirmed that the pre-requisite unimodal cost distributions associated with deep stereo networks, indeed lead to deterministic convergence of the SNCE metric as anticipated in Chapter 4. At this stage however, the network was neither capable of building the cost volume one layer at a time nor terminating the cost volume automatically based on the SNCE metric. It was also not clear how an elaborate cost aggregation network would affect the overall process.

The progressive construction and the automatic termination of the cost volume at an arbitrary disparity required dynamic assignment of memory to the cost volume tensor. Chapter 7 introduced an improved deep stereo network with the ability to dynamically allocate memory to the cost volume one layer at a time. The new design was able to compute SNCE at every layer and then terminate the process based on the estimated value of the metric. The new network which was named ADSR-Net, also featured a cost aggregation stage immediately after the cost volume creation stage. The ADSR-Net demonstrated that the SNCE metric can be implemented in a machine learning friendly manner (*Objective 2* above). However, to achieve the *Objective 3*, the network had to be trained in such a way that the accuracy of both the maximum disparity

estimations as well as the output disparity estimations would improve simultaneously. Therefore, a unique training scheme called “clamped training” was adopted. In clamped training, the output accuracy of a network is combined with some supplementary criteria required to be optimized during the training process. Hence, when using clamped training with the ADSR-Net, the output disparity error of the network was combined with the disparity error associated with an intermediate disparity map extracted just before the cost aggregation stage (pre-aggregation level). This led to the effective “clamping” of the output accuracy to the pre-aggregation accuracy which in turn resulted in simultaneous improvement of both the maximum disparity estimations and the final disparity predictions.

Extensive evaluations with standard stereo datasets (i.e., KITTI, Middlebury, Scene Flow, ETH3D) and stereo images captured with a custom-built stereo camera system showed that the ADSR-Net is able to achieve complete automation in maximum disparity estimation during stereo inference. This eliminated the need for any user-specified parameter values when estimating disparity maps. The resulting deep stereo network set a parameter-less stereo benchmark on its own due to being the only deep stereo network with a variable-sized cost volume which is based on the dynamically determined maximum disparity.

8.2 Research Outcomes and Contributions

The novel deep stereo technique (ADSR-Net) provides a user independent solution to the stereo disparity estimation problem. This contrasts with many traditional and deep-learning-based stereo methods which have been designed to achieve higher matching accuracy on datasets such as KITTI and Middlebury, often with the help of users selecting a suitable maximum disparity value to optimize the output accuracy. Furthermore, by eliminating the most common user-defined parameter in deep stereo, this thesis has demonstrated complete user-parameter-independence in deep stereo inference with an example implementation that produced promising results. In addition to achieving full autonomy (which is the core objective behind this thesis), the ADSR-Net has also delivered the following positive research outcomes.

1. Gains in performance when processing stereo image sequences

The ADSR-Net's ability to dynamical choose a suitable maximum disparity value produced up to 50% savings in computational time when tested on sample stereo image sequences from the Scene Flow dataset. This was in comparison to the performance of the exact same algorithm with a fixed-sized cost volume set to a maximum disparity equal to the maximum ground-truth disparity observed in the dataset. In the conventional approach, a user is required to select a maximum disparity based on intuition in such a way that the chosen value covers all possible maximum disparity values for the scenes. Therefore, even for the long-range scenes with a small maximum disparity, such algorithms need to produce a fixed-sized cost volume with its disparity dimension equal to the user-selected large maximum disparity value. In contrast, the ADSR-Net is able to select a suitable value for each individual stereo pair automatically which leads to optimal use of the computing resources thus leading to savings in processing time.

2. Memory Efficient Deep Stereo Design Pattern

The SNCE metric requires single-valued matching costs between corresponding pixels. Therefore, the ADSR-Net uses a 3D cost volume having the dimensions determined by the image width, height and the maximum disparity estimated automatically by the SNCE metric. However, those single-valued costs are generated by a pre-aggregation network with trainable parameters. Hence, the ADSR-Net can learn to retain just enough information in a smaller-dimensional cost volume (i.e., 3D compared to 4D cost volumes in other deep stereo methods). Consequently, as observed in the results of Chapters 6 and 7, the ADSR-Net can produce accurate disparity maps while utilizing less memory resources to hold the cost volume tensors. This improves the trainability and inference capabilities of the network on consumer grade hardware like GTX1070 and RTX2080 (with 8 GB memory).

3. Portable design for integration with other deep stereo networks

The work presented in this thesis demonstrates the feasibility of achieving complete parameter independence during stereo inference using the 3D

regularization structure in such a way that it can also be easily adopted by the other deep stereo algorithm designs. The feature extraction network and cost regularization phases of the ADSR-Net are clearly segregated from the cost volume module. The feature network used in ADSR-Net is similar to the feature networks of other deep stereo networks that follow the 3D regularization architecture, in term of what it achieves. In addition, the cost volume module of the ADSR-Net was shown to be memory efficient which means that it can be integrated with other feature extraction and regularization designs to develop efficient deep stereo networks. Such models would result in improvements in performance while achieving user parameter independence during stereo inference.

8.3 Addressing the Research Questions

At the beginning of the study, five research questions were identified as being fundamental to the development of a deep stereo network that is independent from the user-specified parameters during stereo disparity inference. This thesis answers each individual research question as summarized below:

1. What phenomenon or associated metric can be used to determine the size of a cost volume in deep learning-based stereo algorithms, without user intervention?

The thesis introduced the novel SNCE *(Sum of New Cost Extrema)* metric which is an estimation of the number of corresponding pixels in a partially built cost volume with their cost extrema located at the current layer of matching costs. If unimodality of the matching cost distributions can be ensured, then the SNCE estimation at a given layer can be used to decide whether to continue building the cost volume by adding new layers or to stop the process. For example, if there is at least one pixel for which the matching cost extrema can be located at the current layer, then the process needs to continue. This provides a deterministic basis for a progressive cost volume construction process to continue or terminate at a given disparity which eliminates the need for the users to specify a value for the maximum disparity. As a result, the maximum disparity estimation can be carried out as part of the cost volume construction process which is integral to many stereo disparity estimation techniques.

2. Would such a metric be computationally efficient enough to be incorporated into a stereo method?

The SNCE metric can be implemented in such a way that it does not change the order of growth in computations associated with stereo disparity estimation in both sequential and parallel processing environments. ALGORITHM 05 in Chapter 5 shows how a typical CPU-based block matching stereo algorithm can be modified to include SNCE-based automatic maximum disparity estimation without affecting the order of growth of the computations of the overall algorithm. ALGORITHM 07 in Chapter 5 shows how the same can be achieved in a GPU environment. This is because most of the repetitive computations required by the SNCE metric are already contained in the stereo disparity estimation process itself. As a result, the SNCE metric can be integrated into a stereo algorithm without introducing significant additional delays. It stays true for deep learning stereo algorithms as well. Most of the existing deep stereo algorithms include a cost volume creation process during which the matching costs (1D or multi-dimensional) are concatenated for each pair of pixels being matched. Therefore, the same computations can be used for the SNCE estimation as well which results in the order of growth remaining unchanged.

3. Can such a metric leverage on the stereo matching accuracy of the deep stereo network for accuracy of maximum disparity predictions?

The SNCE metric ultimately depends on the unimodality of the matching cost distributions for accurate estimation of the maximum disparity for a given pair of stereo images. The deep stereo networks that follow the 3D regularization structure, naturally require the matching cost distributions to be unimodal so that the "Softargmin" operation can be used during disparity regression. Therefore, in deep stereo networks, the unimodality of the matching cost distributions improves with training. The increase in unimodality positively affects the SNCE-based maximum disparity estimations. Since the network is trained with loses calculated using the output disparity maps, the improvements in disparity estimation accuracy yield more unimodal

distributions which in turn improves the maximum disparity estimations using the SNCE metric.

4. If such a metric is integrated into a deep learning network, will the model require a special training regime to converge?

The ADSR-Net has an elaborate cost aggregation network between the cost volume module and the "Softargmin" based disparity regression. This eases the requirement for the cost volume coming out of the cost volume module to be unimodal. In other words, when an extended cost aggregation stage is present, it becomes the responsibility of the cost aggregation network to produce a unimodal matching cost distribution for the regression layers that use "Softargmin" operation. Hence, if the ADSR-Net is trained using the final disparity prediction accuracy alone, then the convergence of SNCE cannot be guaranteed. Therefore, a special training technique is required to ensure that the SNCE convergence takes place in parallel with the improvements to the final disparity accuracy. This is achieved by obtaining an intermediate disparity map from the cost volume (via additional regression and up-sampling layers) and adding the associated error to the final disparity estimation error of the network when computing the overall loss. Chapter 7 introduced the concept and defined it as "clamped training". When the intermediate and final disparity predictions are clamped together, both are required to improve simultaneously for the output loss of the ADSR-Net to be reduced.

5. What gains could be achieved by using the automatic maximum disparity prediction with a deep learning network?

With SNCE, a deep network can learn to predict the final disparity map without depending on user inputs. The overall outcome is an algorithm which can be used to obtain a disparity map without requiring prior knowledge of stereo vision or related parameters. It also frees the algorithm output from errors in user's judgment related to the maximum disparity. The existing deep stereo networks often require users to specify a different maximum disparity value if the scene structure, resolution or the stereo baseline is changed. The ADSR-

Net which is equipped with SNCE does not have any such dependencies. Even if a new stereo camera is plugged in, the ADSR-Net can produce disparity maps right away as seen from the results reported in Chapter 7. Furthermore, dynamic selection of maximum disparity leads to significant savings when processing stereo image sequences. The conventional approach requires users to specify a maximum disparity suitable for all scenes in a stereo image sequence. As a result, users must select the largest expected maximum disparity. This results in the algorithm having to process a larger cost volume even for a scene with a smaller disparity. It also leads to over-use of computational resources due to processing larger tensors than necessary. According to the evaluation results in Chapter 7, the average processing time per image pair improved by as much as 50% or more when automatic maximum disparity detection was used instead of a fixed-sized cost volume.

8.4 Future Work

While achieving the core objectives, the study gave rise to the following future work.

1. Improving the learning process and the loss function to incorporate unimodality

As elaborated in Chapter 7, the ADSR-Net training process is governed by a loss function which combines the disparity prediction error at both the intermediate and final output levels in order to ensure that the SNCE-based maximum disparity estimation accuracy and the output disparity accuracy improve at the same time. However, as pointed out in Chapter 4, the SNCE metric ultimately depends on unimodality of the cost distributions. Apart from relying on the unimodal cost distributions produced by the "Softargmin" based regression layers, the loss function used for the ADSR-Net training did not explicitly include the unimodality as part of the loss equation mathematically. The ADSR-Net training process can be further improved by introducing additional losses to explicitly force the algorithm to adjust its weights and biases to further improve the unimodality of the final cost distributions. Due to the computational complexity of the existing unimodality tests, they could not be feasibly implemented on consumer grade GPU hardware at the time of study.

2. Training for a larger number of iterations with high performance hardware or multi-GPU environments

As elaborated in Chapter 7, the ADSR-Net can be trained on consumer grade GPU hardware due to the efficient use of computational resources. However, a higher disparity estimation accuracy can be achieved by increasing the number of training cycles (epochs) and training in high performance computing environments. Due to the unavailability of such advanced computing resources, extensive training of the ADSR-Net was not feasible during the study. Even during the extended training stage outlined in Chapter 7, the ADSR-Net was only trained with batches of randomly selected image samples instead of the whole training data split from the Scene Flow "Flying Things 3D" data. By using advanced computing resources during training, the ADSR-Net can be trained for a larger number of epochs on full datasets which would further increase the accuracies reported in Chapter 7. Only after such extended training that the ADSR-Net can be compared properly against other deep stereo accuracy benchmarks such as KITTI and Middlebury. However, it must be stated that the ADSR-Net being fully autonomous in determining parameters during inference, sets itself apart from the other deep stereo networks in which the users are able to carefully select parameters to optimize the results for a given dataset. Nevertheless, further training and comparison against benchmark datasets would help validate the potential of the ADSR-Net.

3. Effect of challenging conditions which are common for stereo vision in general

It was also found in Chapter 7 that the accuracy of disparity predictions by the ADSR-Net can be affected by commonly observed challenges in stereo vision (and machine vision in general) such as low textured regions and phenomena like glare and lens flare. The work presented in thesis does not include an in-depth analysis of the algorithm's response to such conditions. The existing knowledge as well as emerging knowledge in such areas can also be incorporated into the ADSR-Net to improve the output disparity accuracy of its disparity predictions. However, it is beyond the scope of this thesis.

REFERENCES

[1] V. Nityananda and J. C. A. Read, “Stereopsis in animals: Evolution, function and mechanisms,” *J. Exp. Biol.*, vol. 220, no. 14, pp. 2502–2512, 2017, doi: 10.1242/jeb.143883.

[2] S. Sharma, P. Gupta, and C. M. Markan, “Adaptive Neuromorphic Circuit for Stereoscopic Disparity Using Ocular Dominance Map,” *Neurosci. J.*, vol. 2016, 2016.

[3] E. C. H. Cheung, J. Wong, J. Chan, and J. Pan, “Optimization-based automatic parameter tuning for stereo vision,” in *IEEE International Conference on Automation Science and Engineering*, 2015, vol. 2015-Octob, pp. 855–861, doi: 10.1109/CoASE.2015.7294188.

[4] X. Cheng *et al.*, “Hierarchical Neural Architecture Search for Deep Stereo Matching,” *arXiv Prepr. arXiv2010.13501*, 2020, [Online]. Available: http://arxiv.org/abs/2010.13501.

[5] V. Tankovich, C. Häne, Y. Zhang, A. Kowdle, S. Fanello, and S. Bouaziz, “HITNet: Hierarchical Iterative Tile Refinement Network for Real-time Stereo Matching,” *arXiv Prepr. arXiv2007.12140*, 2020, [Online]. Available: http://arxiv.org/abs/2007.12140.

[6] X. Cheng, P. Wang, and R. Yang, “Learning depth with convolutional spatial propagation network,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 10, pp. 2361–2379, 2019.

[7] F. Zhang, V. Prisacariu, R. Yang, and P. H. S. Torr, “Ga-net: Guided aggregation net for end-to-end stereo matching,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 185–194.

[8] X. Gu, Z. Fan, S. Zhu, Z. Dai, F. Tan, and P. Tan, “Cascade cost volume for high-resolution multi-view stereo and stereo matching,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 2495–2504.

[9] Y. Zhou, Y. Song, and J. Lu, “Stereo Image Dense Matching by Integrating Sift and Sgm Algorithm,” *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.*, vol. 42, p. 3, 2018.

[10] D. Min, S. Yea, Z. Arican, and A. Vetro, “Disparity search range estimation: enforcing temporal consistency,” in *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2010, pp. 2366–2369.

[11] A. Satnik, R. Hudec, P. Kamencay, J. Hlubik, and M. Benco, “A comparison of key-point descriptors for the stereo matching algorithm,” in *2016 26th International Conference Radioelektronika (RADIOELEKTRONIKA)*, 2016, pp. 292–295.

[12] J. Cech and R. Sara, “Efficient sampling of disparity space for fast and accurate matching,” in *2007 IEEE Conference on Computer Vision and Pattern Recognition*, 2007, pp. 1–8.

[13] A. Geiger, M. Roser, and R. Urtasun, “Efficient large-scale stereo matching,” in *Asian Conference on Computer Vision*, 2010, pp. 25–38.

[14] U. Ozgunalp, X. Ai, Z. Zhang, G. Koc, and N. Dahnoun, “Block-matching disparity map estimation using controlled search range,” in *2015 7th Computer Science and Electronic Engineering Conference (CEEC)*, 2015, pp. 35–40.

[15] Z. Zhang and Y. Shan, “A progressive scheme for stereo matching,” in *European Workshop on 3D Structure from Multiple Images of Large-Scale Environments*, 2000, pp. 68–85.

[16] M. Bleyer, C. Rhemann, and C. Rother, “PatchMatch stereo - Stereo matching with slanted support windows,” in *BMVC 2011 - Proceedings of the British Machine Vision Conference 2011*, 2011, vol. 11, pp. 1–11, doi: 10.5244/C25.14.

[17] B. Cyganek and J. Borgosz, “Maximum disparity threshold estimation for stereo imaging systems via variogram analysis,” in *International Conference on Computational Science*, 2003, pp. 591–600.

[18] M. Sizintsev and R. P. Wildes, “Coarse-to-fine stereo vision with accurate 3D boundaries,” *Image Vis. Comput.*, vol. 28, no. 3, pp. 352–366, 2010, doi: 10.1016/j.imavis.2009.06.008.

[19] C. Republic, “Automatic Disparity Search Range Estimation for Stereo Pairs of Unknown Scenes ∗,” *Proc. CVWW*, pp. 1–10, 2004.

[20] R. Šára, “Finding the largest unambiguous component of stereo matching,” in *European Conference on Computer Vision*, 2002, pp. 900–914.

[21] S. Smirnov, A. Gotchev, and M. Hannuksela, “A disparity range estimation technique for stereo-video streaming applications,” in *2013 IEEE International Conference on Multimedia and Expo Workshops (ICMEW)*, 2013, pp. 1–4.

[22] M. Gong, “Enforcing temporal consistency in real-time stereo estimation,” in *European Conference on Computer Vision*, 2006, pp. 564–577.

[23] H. Ma *et al.*, “Multiple lane detection algorithm based on optimised dense disparity map estimation,” in *2018 IEEE International Conference on Imaging Systems and Techniques (IST)*, 2018, pp. 1–5.

[24] R. Labayrade, D. Aubert, and J.-P. Tarel, “Real time obstacle detection in stereovision on non flat road geometry through" v-disparity" representation,” in *Intelligent Vehicle Symposium, 2002. IEEE*, 2002, vol. 2, pp. 646–651.

[25] K. Zhou, X. Meng, and B. Cheng, “Review of Stereo Matching Algorithms Based on Deep Learning,” *Comput. Intell. Neurosci.*, vol. 2020, 2020, doi: 10.1155/2020/8562323.

[26] R. A. Hamzah and H. Ibrahim, “Literature survey on stereo vision disparity map algorithms,” *J. Sensors*, vol. 2016, 2016.

[27] H. Laga, L. V. Jospin, F. Boussaid, and M. Bennamoun, “A survey on deep learning techniques for stereo-based depth estimation,” *IEEE Trans. Pattern Anal. Mach. Intell.*, 2020.

[28] A. Francisco, “Vergence micromovements and depth perception,” in *BMVC92*, Springer, 1992, pp. 367–376.

[29] L. Zhao and C. E. Thorpe, “Stereo-and neural network-based pedestrian detection,” *IEEE Trans. Intell. Transp. Syst.*, vol. 1, no. 3, pp. 148–154, 2000.

[30] Q. Baig, O. Aycard, T. D. Vu, and T. Fraichard, “Fusion between laser and stereo vision data for moving objects tracking in intersection like scenario,” in *2011 IEEE Intelligent Vehicles Symposium (IV)*, 2011, pp. 362–367.

[31] N. Lazaros, G. C. Sirakoulis, and A. Gasteratos, “Review of stereo vision algorithms: from software to hardware,” *Int. J. Optomechatronics*, vol. 2, no. 4, pp. 435–462, 2008.

[32] O. Veksler, “Semi-dense stereo correspondence with dense features,” in *Proceedings IEEE Workshop on Stereo and Multi-Baseline Vision (SMBV 2001)*, 2001, pp. 149–157.

[33] S. Pillai, S. Ramalingam, and J. J. Leonard, “High-performance and tunable stereo reconstruction,” in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 3188–3195.

[34] O.-E. Ng and V. Ganapathy, “A novel modular framework for stereo vision,” in *2009 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, 2009, pp. 857–862.

[35] A. Motten and L. Claesen, “A binary adaptable window SoC architecture for a stereo vision based depth field processor,” in *2010 18th IEEE/IFIP International Conference on VLSI and System-on-Chip*, 2010, pp. 25–30.

[36] M. Agrawal and L. S. Davis, “Window-based, discontinuity preserving stereo,” in *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, 2004, vol. 1, pp. I--I.

[37] V. Kolmogorov and R. Zabih, “Graph cut algorithms for binocular stereo with occlusions,” in *Handbook of Mathematical Models in Computer Vision*, Springer, 2006, pp. 423–437.

[38] J. Sun, N.-N. Zheng, and H.-Y. Shum, “Stereo matching using belief propagation,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 7, pp. 787–800, 2003.

[39] H. Hirschmuller, “Accurate and efficient stereo processing by semi-global matching and mutual information,” in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05)*, 2005, vol. 2, pp. 807–814.

[40] H. Hirschmuller, “Stereo processing by semiglobal matching and mutual information,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 2, pp. 328–341, 2007.

[41] X. Hu and P. Mordohai, “A quantitative evaluation of confidence measures for stereo vision,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 11, pp. 2121–2133, 2012.

[42] D. Pfeiffer, S. Gehrig, and N. Schneider, “Exploiting the power of stereo confidences,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 297–304.

[43] R. Haeusler, R. Nair, and D. Kondermann, “Ensemble learning for confidence measures in stereo vision,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 305–312.

[44] A. Spyropoulos, N. Komodakis, and P. Mordohai, “Learning to detect ground control points for improving the accuracy of stereo matching,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1621–1628.

[45] J. Zbontar, Y. LeCun, and others, “Stereo matching by training a convolutional neural network to compare image patches.,” *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 2287–2318, 2016.

[46] Z. Chen, X. Sun, L. Wang, Y. Yu, and C. Huang, “A deep visual correspondence embedding model for stereo matching costs,” in *Proceedings of the IEEE International*

Conference on Computer Vision, 2015, pp. 972–980.

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

[48] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *Int. J. Comput. Vis.*, vol. 47, no. 1, pp. 7–42, 2002.

[49] D. Scharstein and R. Szeliski, "High-accuracy stereo depth maps using structured light," in *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, 2003, vol. 1, pp. I--I.

[50] D. Scharstein and C. Pal, "Learning conditional random fields for stereo," in *2007 IEEE Conference on Computer Vision and Pattern Recognition*, 2007, pp. 1–8.

[51] H. Hirschmuller and D. Scharstein, "Evaluation of cost functions for stereo matching," in *2007 IEEE Conference on Computer Vision and Pattern Recognition*, 2007, pp. 1–8.

[52] D. Scharstein *et al.*, "High-resolution stereo datasets with subpixel-accurate ground truth," in *German Conference on Pattern Recognition*, 2014, pp. 31–42.

[53] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 3354–3361.

[54] M. Menze and A. Geiger, "Object scene flow for autonomous vehicles," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3061–3070.

[55] N. Mayer *et al.*, "A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4040–4048.

[56] T. Schops *et al.*, "A multi-view stereo benchmark with high-resolution images and multi-camera videos," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 3260–3269.

[57] G. Yang, X. Song, C. Huang, Z. Deng, J. Shi, and B. Zhou, "Drivingstereo: A large-scale dataset for stereo matching in autonomous driving scenarios," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 899–908.

[58] A. Canessa, A. Gibaldi, M. Chessa, M. Fato, F. Solari, and S. P. Sabatini, "A dataset of stereoscopic images and ground-truth disparity mimicking human fixations in peripersonal space," *Sci. Data*, vol. 4, no. 1, pp. 1–16, 2017, doi: 10.1038/sdata.2017.34.

[59] C. Zhou, H. Zhang, X. Shen, and J. Jia, "Unsupervised Learning of Stereo Matching," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, vol. 2017-Octob, pp. 1576–1584, doi: 10.1109/ICCV.2017.174.

[60] Y. Zhong, Y. Dai, and H. Li, "Self-Supervised Learning for Stereo Matching with Self-Improving Ability," *arXiv Prepr. arXiv1709.00930*, 2017, [Online]. Available: http://arxiv.org/abs/1709.00930.

[61] Y. Zhang *et al.*, "Activestereonet: End-to-end self-supervised learning for active stereo systems," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 784–801.

[62] J. Pang, W. Sun, J. S. J. Ren, C. Yang, and Q. Yan, “Cascade Residual Learning: A Two-Stage Convolutional Neural Network for Stereo Matching,” in *Proceedings - 2017 IEEE International Conference on Computer Vision Workshops, ICCVW 2017*, 2017, vol. 2018-Janua, pp. 878–886, doi: 10.1109/ICCVW.2017.108.

[63] A. Shaked and L. Wolf, “Improved stereo matching with constant highway networks and reflective confidence learning,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4641–4650.

[64] A. Kendall *et al.*, “End-to-end learning of geometry and context for deep stereo regression,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 66–75.

[65] X. Song, X. Zhao, H. Hu, and L. Fang, “Edgestereo: A context integrated residual pyramid network for stereo matching,” in *Asian Conference on Computer Vision*, 2018, pp. 20–35.

[66] Q. Wang, S. Shi, S. Zheng, K. Zhao, and X. Chu, “FADNet: A Fast and Accurate Network for Disparity Estimation,” in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 101–107.

[67] Z. Zhu, M. He, Y. Dai, Z. Rao, and B. Li, “Multi-scale cross-form pyramid network for stereo matching,” in *2019 14th IEEE Conference on Industrial Electronics and Applications (ICIEA)*, 2019, pp. 1789–1794.

[68] G. Yang, H. Zhao, J. Shi, Z. Deng, and J. Jia, “Segstereo: Exploiting semantic information for disparity estimation,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 636–651.

[69] M. Poggi, D. Pallotti, F. Tosi, and S. Mattoccia, “Guided stereo matching,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 979–988.

[70] X. Du, M. El-Khamy, and J. Lee, “AMNet: Deep Atrous Multiscale Stereo Disparity Estimation Networks,” *arXiv Prepr. arXiv1904.09099*, 2019, [Online]. Available: http://arxiv.org/abs/1904.09099.

[71] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[72] Y. Zhang *et al.*, “Adaptive unimodal cost volume filtering for deep stereo matching,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020, vol. 34, no. 07, pp. 12926–12934.

[73] X. Guo, K. Yang, W. Yang, X. Wang, and H. Li, “Group-wise correlation stereo network,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3273–3282.

[74] Z. Wu, X. Wu, X. Zhang, S. Wang, and L. Ju, “Semantic stereo matching with pyramid cost volumes,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 7484–7493.

[75] S. Khamis, S. Fanello, C. Rhemann, A. Kowdle, J. Valentin, and S. Izadi, “Stereonet: Guided hierarchical refinement for real-time edge-aware depth prediction,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 573–590.

[76] J.-R. Chang and Y.-S. Chen, “Pyramid stereo matching network,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 5410–5418.

[77] F. Zhang, X. Qi, R. Yang, V. Prisacariu, B. Wah, and P. Torr, “Domain-Invariant Stereo Matching Networks,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2020, vol. 12347 LNCS, pp. 420–439, doi: 10.1007/978-3-030-58536-5_25.

[78] Z. Jie *et al.*, “Left-Right Comparative Recurrent Model for Stereo Matching,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3838–3846, doi: 10.1109/CVPR.2018.00404.

[79] R. K. Srivastava, K. Greff, and J. Schmidhuber, “Highway Networks,” *arXiv Prepr. arXiv1505.00387*, 2015, [Online]. Available: http://arxiv.org/abs/1505.00387.

[80] Y. Zhong, H. Li, and Y. Dai, “Open-world stereo video matching with deep rnn,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 101–116.

[81] M. Shahzeb Khan Gul, M. Bätz, and J. Keinert, “Pixel-wise confidences for stereo disparities using recurrent neural networks,” in *30th British Machine Vision Conference 2019, BMVC 2019*, 2020, p. 23.

[82] A. Emlek and M. Peker, “Refinement of matching costs for stereo disparities using recurrent neural networks,” *EURASIP J. Image Video Process.*, vol. 2021, no. 1, pp. 1–19, 2021.

[83] S. Kim, D. Min, S. Kim, and K. Sohn, “Adversarial confidence estimation networks for robust stereo matching,” *IEEE Trans. Intell. Transp. Syst.*, 2020.

[84] S. Cheng *et al.*, “Deep stereo using adaptive thin volume representation with uncertainty awareness,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 2524–2534.

[85] M. Poggi, F. Tosi, and S. Mattoccia, “Learning a confidence measure in the disparity domain from O (1) features,” *Comput. Vis. Image Underst.*, vol. 193, p. 102905, 2020.

[86] F. Tosi, M. Poggi, A. Benincasa, and S. Mattoccia, “Beyond local reasoning for stereo confidence estimation with deep learning,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 319–334.

[87] M. Poggi, F. Aleotti, F. Tosi, G. Zaccaroni, and S. Mattoccia, “Self-adapting Confidence Estimation for Stereo,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 12369 LNCS, pp. 715–733, 2020, doi: 10.1007/978-3-030-58586-0_42.

[88] S. Kim, S. Kim, D. Min, and K. Sohn, “Laf-net: Locally adaptive fusion networks for stereo confidence estimation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 205–214.

[89] M. Poggi, G. Agresti, F. Tosi, P. Zanuttigh, and S. Mattoccia, “Confidence Estimation for ToF and Stereo Sensors and Its Application to Depth Data Fusion,” *IEEE Sens. J.*, vol. 20, no. 3, pp. 1411–1421, 2020, doi: 10.1109/JSEN.2019.2946591.

[90] G.-Y. Nie *et al.*, “Multi-level context ultra-aggregation for stereo matching,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3283–3291.

[91] A. Fusiello, V. Roberto, and E. Trucco, “Efficient stereo with multiple windowing,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1997, pp. 858–863.

[92] B. Tippetts, D. J. Lee, K. Lillywhite, and J. Archibald, “Review of stereo vision algorithms and their suitability for resource-limited systems,” *J. Real-Time Image Process.*, vol. 11, no. 1, pp. 5–25, 2016.

[93] M. F. Tappen and W. T. Freeman, “Comparison of graph cuts with belief propagation for stereo, using identical MRF parameters,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2003, vol. 2, pp. 900–907, doi: 10.1109/iccv.2003.1238444.

[94] L. Zhang and S. M. Seitz, “Parameter estimation for MRF stereo,” in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05)*, 2005, vol. 2, pp. 288–295.

[95] A. Seki and M. Pollefeys, “Sgm-nets: Semi-global matching with neural networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 231–240.

[96] H. Xu and J. Zhang, “Aanet: Adaptive aggregation network for efficient stereo matching,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 1959–1968.

[97] G. Yang, J. Manela, M. Happold, and D. Ramanan, “Hierarchical deep stereo matching on high-resolution images,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5515–5524.

[98] S. Tulyakov, A. Ivanov, and F. Fleuret, “Practical deep stereo (pds): Toward applications-friendly deep stereo matching,” *arXiv Prepr. arXiv1806.01677*, 2018.

[99] S. Duggal, S. Wang, W.-C. Ma, R. Hu, and R. Urtasun, “Deeppruner: Learning efficient stereo matching via differentiable patchmatch,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 4384–4393.

[100] M. Ye, E. Johns, A. Handa, L. Zhang, P. Pratt, and G.-Z. Yang, “Self-supervised siamese learning on stereo image pairs for depth estimation in robotic surgery,” *arXiv Prepr. arXiv1705.08260*, 2017.

[101] R. Garg, V. K. Bg, G. Carneiro, and I. Reid, “Unsupervised cnn for single view depth estimation: Geometry to the rescue,” in *European Conference on Computer Vision*, 2016, pp. 740–756.

[102] D. G. Lowe, “Object recognition from local scale-invariant features,” in *Proceedings of the Seventh IEEE International Conference on Computer Vision*, 1999, vol. 2, pp. 1150–1157.

[103] H. Bay, T. Tuytelaars, and L. Van Gool, “Surf: Speeded up robust features,” in *European Conference on Computer Vision*, 2006, pp. 404–417.

[104] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman, “PatchMatch: A randomized correspondence algorithm for structural image editing,” *ACM Trans. Graph.*, vol. 28, no. 3, p. 24, 2009.

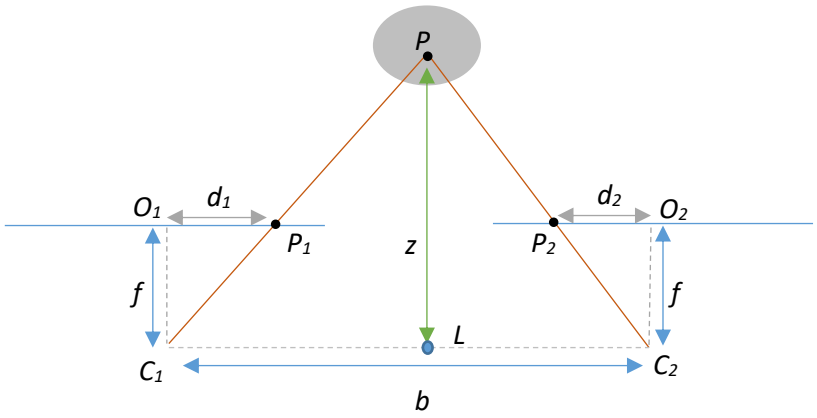
[105] R. Perera and T. Low, “Automatic Stereo Disparity Search Range Detection on Parallel Computing Architectures,” in *International Conference on Image Analysis and*

Recognition, 2020, pp. 404–416.

[106] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, and ..., “Automatic differentiation in pytorch,” 2017, [Online]. Available: https://openreview.net/forum?id=BJJsrmfCZ.

[107] D. P. Kingma and J. A. Ba, “A method for stochastic optimization. arXiv 2014,” *arXiv Prepr. arXiv1412.6980*, vol. 434, 2019.

Appendix A – Relationship Between Depth and Stereo Disparity



Appendix 1.1: Planar view of two cameras in rectified configuration having horizontal epipolar lines resulting in correspondence search along the horizontal rows of image points

The illustration shows the planner view of two cameras (*Camera 1* and *Camera 2*) in rectified configuration registering image points (P_1 and P_2 on their respective image planes) for a point P on an object. The optical centres of the two cameras are denoted by C_1 and C_2 whereas d_1 and d_1 denote the horizontal offsets of the image points from the image centre. Furthermore, z denotes the perpendicular distance (or depth) to the point P from the baseline b. The focal length of the two cameras is equal to f. The centres of the image planes of the two cameras C_1 and C_2 are denoted by O_1 and O_1 respectively. From the similar right-angled triangles of C_1PL *and* $C_1P_1O_1$:

$$\frac{C_1L}{z} = \frac{d_1}{f} \tag{A.1}$$

From the similar right-angled triangles of C_2PL *and* $C_2O_2P_2$:

$$\frac{C_2L}{z} = \frac{d_2}{f} \tag{A.2}$$

By combining Equation (A.1) with Equation (A.2) the following can be deduced:

$$b = C_1L + C_2L = \frac{z\,(d_1 + d_1)}{f} \tag{A.3}$$

If the d denotes the horizontal offset between the in pixels, then the relationship can be written as the following:

$$d = \frac{b\,f}{z} \tag{A.4}$$

As seen from Equation (A.4), the disparity *d* between two corresponding pixels is inversely proportional to the depth *z* to the point in the real world. Therefore, point objects which are located far away from the stereo camera lead to smaller disparities between the corresponding pixels whereas the points closer to the camera introduce much larger disparities. This is the basic principle behind stereo vision-based depth perception.

Appendix B – Supplementary Materials / Software

Supplementary Materials and Utility Software

All the supplementary materials and utility software for this research project are available via an archived folder which can be located by visiting the following link. Please use the "WINRAR" archiver (with the password given below) to access the files.

Link	https://drive.google.com/drive/folders/1DoeJRqT-oVsmpxl77m0bnPuoid8rBSm5?usp=sharing
Password	AD5R!N3T#2021

Chapter 4

Experiments in Chapter 4 required the SNCE metric to be estimated for any given stereo image pair up to a maximum disparity specified by the user. It was also required to be repeated using stereo cost aggregation techniques such as AD, SAD, SSD, NCC, RT and CT. This was achieved using a Window x64 based utility which was developed using C++. A copy of the utility (with source code) is available in a folder named "Chapter 4" in the archive. Please refer to the "README.TXT" in the same folder for additional details on usage, pre-requisites, and permissions.

Chapter 5

Experiments in Chapter 5 required the SNCE metric to be estimated in both CPU and GPU environments while computing the computational time for the disparity estimation process. For the scenarios involving CPUs, a Windows x64 based software utility was developed using C++ and a CUDA kernel was developed in Picadas for GPUs. Both the programs and the source code (GPU-version only as the CPU version is the same as earlier) are available from the same repository in a folder called "Chapter 5". Please refer to the "README.TXT" for additional details and instructions.

Chapter 6

The foundational deep stereo network which was developed as part of the study outlined in Chapter 6, is included as a Windows x64 based executable in a folder named "Chapter 6". The executable file can be used to estimate disparity maps for the files in the "imgs" folder. Trained model is saved in "saved/configs" folder. Please refer to the "README.TXT" for instructions and *Appendix C* for the source code.

Chapter 7

A demo version of the end-to-end deep stereo network (ADSR-Net) is included as a Windows x64 based executable in a folder called "Chapter 7" in the archive. The executable can be used to estimate disparity maps for images placed inside the "sample" folder (without requiring users to specify any parameters). The saved model can be found inside the "saved/configs" folder. Please refer to the "README.TXT" for instructions and *Appendix C* for the source code.

Appendix C - Source Code

Feature Extraction Networks

The feature extraction networks in Chapter 6 and Chapter 7, both use "Type A" and "Type B" blocks. The following two code snippets of show PyTorch implementations of "Type A" and "Type B" blocks.

"Type A" Block – Chapter 6 & 7

```
class TypeA(torch.nn.Module):
    def __init__(self):
        super().__init__()
        # Path 1 - (3x3)
        self.path1 = torch.nn.Sequential(
            torch.nn.Conv2d(32, 32, (3, 3), (1, 1), padding=(1, 1)),
            torch.nn.BatchNorm2d(32),
            torch.nn.ReLU(inplace=True),
            torch.nn.Conv2d(32, 32, (3, 3), (1, 1), padding=(1, 1)),
            torch.nn.BatchNorm2d(32),
            torch.nn.ReLU(inplace=True)
        )

        # Path 2 - (5x5)
        self.path2 = torch.nn.Sequential(
            torch.nn.Conv2d(32, 32, (5, 5), (1, 1), padding=(2, 2), dilation=1),
            torch.nn.BatchNorm2d(32),
            torch.nn.ReLU(inplace=True),
            torch.nn.Conv2d(32, 32, (5, 5), (1, 1), padding=(2, 2), dilation=1),
            torch.nn.BatchNorm2d(32),
            torch.nn.ReLU(inplace=True)
        )

        # Path 3 - (7x7)
        self.path3 = torch.nn.Sequential(
            torch.nn.Conv2d(32, 32, (7,7), (1, 1), padding=(3, 3), dilation=1),
            torch.nn.BatchNorm2d(32),
            torch.nn.ReLU(inplace=True),
            torch.nn.Conv2d(32, 32, (7,7), (1, 1), padding=(3, 3), dilation=1),
            torch.nn.BatchNorm2d(32),
            torch.nn.ReLU(inplace=True)
        )

        # common BN
        self.ball = torch.nn.BatchNorm2d(32)

        # Residual Connection
        self.residual = torch.nn.Sequential()

    def forward(self, x):
        out1 = self.path1(x)
        out2 = self.path2(x)
        out3 = self.path3(x)
        out = out1 + out2 + out3
        out = self.ball(out)
        out += self.residual(x)
        return out
```

“Type B” Block – Chapter 6 & 7

```
class TypeB(torch.nn.Module):
  def __init__(self):
    super().__init__()
    # Path 1 - (3x3)
    self.path1 = torch.nn.Sequential(
      torch.nn.Conv2d(32, 32, (3, 3), (1, 1), padding=(1, 1)),
      torch.nn.BatchNorm2d(32),
      torch.nn.ReLU(inplace=True),
      torch.nn.Conv2d(32, 32, (3, 3), (1, 1), padding=(1, 1)),
      torch.nn.BatchNorm2d(32),
      torch.nn.ReLU(inplace=True)
    )

    # Path 2 - (5x5)
    self.path2 = torch.nn.Sequential(
      torch.nn.Conv2d(32, 32, (5, 5), (1, 1), padding=(2, 2), dilation=1),
      torch.nn.BatchNorm2d(32),
      torch.nn.ReLU(inplace=True),
      torch.nn.Conv2d(32, 32, (5, 5), (1, 1), padding=(2, 2), dilation=1),
      torch.nn.BatchNorm2d(32),
      torch.nn.ReLU(inplace=True)
    )

    # common BN
    self.ball = torch.nn.BatchNorm2d(32)

    # Residual Connection
    self.residual = torch.nn.Sequential()

  def forward(self, x):
    out1 = self.path1(x)
    out2 = self.path2(x)
    out = out1 + out2
    out = self.ball(out)
    out += self.residual(x)
    return out
```

Feature Extraction Network - Chapter 6

The feature extraction network in Chapter 6 uses seven “Type A” blocks and seven “Type B” blocks connected in a cascading arrangement. In addition, the input layer of the network has only 1 input feature channel to accommodate grayscale images.

```
class FeatureNetII(torch.nn.Module):
  def __init__(self):
    super().__init__()
    # Feature Input Block (One channel for grayscale images)
    self.inblock = torch.nn.Sequential(
      torch.nn.Conv2d(1, 32, (5, 5), (4, 4), padding=(2, 2)),
      torch.nn.ReLU(inplace=True),
      torch.nn.BatchNorm2d(32)
    )

    # Seven TypeA layers
    layersA = []
    for i in range(7):
      layersA += [TypeA()]
    self.netA = nn.Sequential(*layersA)

    # Connection Layer
    self.midblock = torch.nn.Sequential(
      torch.nn.Conv2d(32, 32, (3, 3), (1, 1), padding=(1, 1)),
      torch.nn.ReLU(inplace=True),
      torch.nn.BatchNorm2d(32)
    )

    # Seven TypeA layers
    layersB = []
    for i in range(7):
      layersB += [TypeB()]
    self.netB = nn.Sequential(*layersB)

    # Output Layer
    self.clast = torch.nn.Conv2d(32, 32, (3, 3), (1, 1), padding=(1, 1))

  def forward(self, x):
    out = self.clast(self.netB(self.midblock(self.netA(self.inblock(x)))))
    return out
```

Feature Extraction Network – Chapter 7

The feature extraction network used in Chapter 7 uses ten "Type A" blocks and seven "Type B" blocks connected in a cascading arrangement. In addition, the input layer of the network has three input feature channels to accommodate RGB input images.

```
class FeatureNetII(torch.nn.Module):
  def __init__(self):
    super().__init__()
    # Feature Input Block (3 Channels for RGB images)
    self.inblock = torch.nn.Sequential(
      torch.nn.Conv2d(1, 32, (5, 5), (4, 4), padding=(2, 2)),
      torch.nn.ReLU(inplace=True),
      torch.nn.BatchNorm2d(32)
    )

    # Ten TypeA layers
    layersA = []
    for i in range(10):
      layersA += [TypeA()]
    self.netA = nn.Sequential(*layersA)

    # Connection Layer
    self.midblock = torch.nn.Sequential(
      torch.nn.Conv2d(32, 32, (3, 3), (1, 1), padding=(1, 1)),
      torch.nn.ReLU(inplace=True),
      torch.nn.BatchNorm2d(32)
    )

    # Seven TypeA layers
    layersB = []
    for i in range(7):
      layersB += [TypeB()]
    self.netB = nn.Sequential(*layersB)

    # Output Layer
    self.clast = torch.nn.Conv2d(32, 32, (3, 3), (1, 1), padding=(1, 1))

  def forward(self, x):
    out = self.clast(self.netB(self.midblock(self.netA(self.inblock(x)))))
    return out
```

Cost Volume Modules (CV Modules)

Two types of cost volume modules were developed as part of the study. The cost volume module developed in Chapter 6 declares a fixed-sized cost volume in memory and computes the matching costs. It also computes the SNCE values at each disparity for comparison against the maximum ground-truth disparity data.

Cost Volume Module - Chapter 6

```
# PyTorch based python function to compute SNCE metric at every layer of the
# fixed-sized cost volume of the network used for experiments in Chapter 6
# x -> Left feature map with dimensions = batch_size x channels x height x width
# y -> Right feature map with dimensions = batch_size x channels x height x width
# preagg aggregates features from channels (32) to (1) in a learnable way
def calCV(x, y, maxdisp, preagg):
        batch_size      = x.size()[0]     # Batch Size
        channels        = x.size()[1]     # Number of channels
        height          = x.size()[2]     # Image height
        width           = x.size()[3]     # Image width
        # Cost volume declared in advance in memory up to a depth of "maxdisp"
        covol           = x.new().resize_(batch_size, 1, maxdisp, height, width).zero_()
        # Temporary layer to store right featured at a given disparity
        RX              = y.new().resize_(batch_size, channels, height, width).zero_()
        sncm = []       # Array to hold SNCE value (Sum of New Cost Minima is used)
        # Loop through disparities up to "maxdisp"
        for i in range(0, maxdisp):
                RX[:, :, :, :]          = 0                     # Clear temporary feature layer
                RX[:, :, :, i:width]  = y[:, :, :, 0:width - i]     # Assign the current layer to RX
                # Compute and store matching costs at 0 disparity
                covol[:, :, i, :, :]    = preagg (torch.cat((x, RX), dim=1))

                if i > 0:   # No need to compute SNCE at disparity (equal to width x height)
                        # Compute the SNCE value at current disparity (i)
                        sval = (torch.sum((torch.min(covol[0, 0, :i, :, :], dim=0).indices) == (i -
                        1))).cpu().numpy()
                        sncm += [sval]    # Update SNCE array

        return covol, sncm
```

Cost Volume Module - Chapter 7

The cost volume module of the ADSR-Net developed in Chapter 7, can build a cost volume one layer at a time while computing matching costs and the SNCE metric at each disparity before scheduling termination of the process when the termination criteria is met.

```
# PyTorch based python function to generate a layer wise cost volume while computing
# the SNCE value at each layer and then terminate when the SNCE value reaches zero.
# x -> Left feature map with dimensions = batch_size x channels x height x width
# y -> Right feature map with dimensions = batch_size x channels x height x width
# preagg aggregates features from channels (32) to (1) in a learnable way
def calculateCVA(x, y, preagg):
        schedule        = False         # Scheduled termination flag
        batch_size      = x.size()[0]   # Stereo image pairs in a batch
        channels        = x.size()[1]   # Number of channels/filters
        height          = x.size()[2]   # Image height at quarter resolution
        width           = x.size()[3]   # Image width at quarter resolution
        # Initialize cost volume tensor with only one layer in it
        covol           = x.new().resize_(batch_size, 1, 1, height, width).zero_()
        # Temporary layer to store right featured at a given disparity
        RX              = y.new().resize_(batch_size, channels, height, width).zero_()
        # Expect worst case scenario
        Maxd            = width
        # Loop through all disparities until SNCE value reaches zero
        for i in range(0, width):
                RX[:, :, :, :]          = 0             # Clear temporary feature layer
                RX[:, :, :, i:width]  = y[:, :, :, 0:width - i]       # Assign the current layer to RX
                if i == 0: # If there is only one layer in the cost volume
                        # Compute and store matching costs at 0 disparity
                        covol[:, :, i, :, :]    = preagg(torch.cat((x,RX), dim=1))
                else:   # There are previous layers stacked up to (i)th disparity
                        # Compute the matching costs at disparity (i)
                        tmp             = preagg(torch.cat((x,RX), dim=1))
                        tmp             = torch.unsqueeze(tmp, dim=0)
                        # Stack up the current matching costs on top of the cost volume
                        covol           = torch.cat((covol,tmp), dim=2)
                if i > 0:  # No need to compute SNCE at disparity (equal to width x height)
                        # Compute the SNCE value at current disparity (i)
                        sval = (torch.sum((torch.min(covol[0, 0, :i, :, :], dim=0).indices) == (i-1)))
                        # Added for formatting. Could be part of the previous line
                        sval = sval.cpu().numpy()
                # If SNCE value is zero, schedule for termination
                if sval < 1:
                        schedule = True
                        print("Detected max disparity "+str(i*4))
                        maxd    = i * 4   # compute maximum disparity at original resolution
                # Check to see scheduled termination criteria is met
                if schedule and i%2 == 1:
                        print("Breaking at i="+str(i))
                        break
        return covol, maxd
```

Feature Aggregation Network - Chapter 6 & 7

Both networks in Chapter 6 and Chapter 7 use a 2D feature aggregation network when computing a 3D cost volume using the feature volumes from the feature extraction network. This is achieved by using a 2D convolutional network which can predict a singular value representation for two feature arrays for two pixels being matched.

```
class PreAgg(torch.nn.Module):
  def __init__(self, fsize):
    super().__init__()
    self.conv1 = torch.nn.Conv2d(fsize, fsize, 3, stride=1, padding=1)
    self.relu1 = torch.nn.ReLU(inplace=True)
    self.conv3 = torch.nn.Conv2d(fsize, 1, 3, stride=1, padding=1)

  def forward(self,x):
    out = self.conv1(x)
    out = self.relu1(out)
    out = self.conv3(out)
    return out
```

Differentiable Disparity Regression

End-to-end deep stereo networks rely on "Softargmin" operation to regress disparities from cost volumes without hindering the gradient backpropagation through the network. In the ADSR-Net and the foundational deep stereo network in Chapter 6, the process is implemented as a function. The following code snippet shows the PyTorch implementation of the regression operation.

Softargmin Based Disparity Regression - Chapter 6 & 7

```
def regress(covol):
  shape = covol.size()
  dblock = torch.ones(shape[2], 1, 1).cuda()
  dblock[:, 0, 0] = torch.arange(0, shape[2])
  dblock = dblock.repeat(shape[0], 1, 1, shape[3], shape[4])
  dmap = torch.sum(dblock * (F.softmax(-1 * covol, dim=2)), dim=2)
  return dmap
```

Complete Network Used in Chapter 6

Due to the modularity of the components developed as part of this PhD study, building complex deep stereo networks becomes easier. For example, the following code snippet shows the complete network (from Chapter 6).

End-to-End Network - Chapter 6

```
class MyNet(torch.nn.Module):
  def __init__(self,bsize,height,width,maxdisp):
    super().__init__()
    self.bsize   = bsize          # Batch Size (Use 1 to avoid tensor mismatch)
    self.height  = height         # Image Height
    self.width   = width          # Image Width
    self.maxdisp = maxdisp        # Fixed-sized cost needs max disparity
    self.fnet    = FeatureNet1()  # The Feature network
    self.preagg  = PreAgg(64)     # The feature aggregation network

    # The Up-sampling layer of the network
    self.upsample = torch.nn.Upsample(scale_factor=4, mode='bicubic', align_corners=False)

    # Network initialization code comes here

  def forward(self,left,right):

    left = self.fnet(left)    # Obtaining the left feature volume

    right = self.fnet(right)# Obtaining the right feature volume

    # Build a fixed-sized cost volume and return with the SNCE array
    covol, snce = calCV(left, right, self.preagg)

    # Aggregate the costs, regress disparities and upsample to get final disparity map

    dmap1 = self.upsample(regress(covol) * 4)
    return dmap1, sncm
```

Cost Aggregation Network of the ADSR-Net

The ADSR-Net uses multi-layer cost aggregation network to further aggregate the cost volume produced by the CV module. As outlined in Chapter 7, the cost aggregation network has 3 types of blocks: input block, aggregation block and the output block. PyTorch based implementations of the individual blocks and the full network is shown in the code snippets below.

Input Layer of the Cost Aggregation Network - Chapter 7

```
class Conv3DIn(torch.nn.Module):
  def __init__(self, inf, outf, mask, pad, stride):
    super().__init__()
    self.net = torch.nn.Sequential(
      torch.nn.Conv3d(inf, outf, kernel_size=mask, padding=pad, stride=stride,bias=False),
      torch.nn.BatchNorm3d(outf),
      torch.nn.ReLU(inplace=True),
    )

  def forward(self, x):
    return self.net(x)
```

Output Layer of the Cost Aggregation Network - Chapter 7

```
class Conv3DOut(torch.nn.Module):
  def __init__(self, inf, outf, mask, pad, step):
    super().__init__()
    self.net = torch.nn.Sequential(
      torch.nn.Conv3d(inf, inf, kernel_size=mask, padding=pad, stride=step),
      torch.nn.BatchNorm3d(inf),
      torch.nn.ReLU(inplace=True),
      torch.nn.Conv3d(inf, outf, kernel_size=mask, padding=pad, stride=step)
    )

  def forward(self, x):
    return self.net(x)
```

Aggregation Block from the Cost Aggregation Network - Chapter 7

```
class Conv3DMid(torch.nn.Module):
  def __init__(self, inf, outf, mask, pad, step):
    super().__init__()
    # Branch A of the aggregation block (with 3D Convolution)
    self.net = torch.nn.Sequential(
      torch.nn.Conv3d(inf, outf, kernel_size=mask, padding=pad, stride= step),
      torch.nn.BatchNorm3d(outf),
      torch.nn.ReLU(inplace=True),
      torch.nn.Conv3d(inf, outf, kernel_size=mask, padding=pad, stride= step),
      torch.nn.BatchNorm3d(outf),
      torch.nn.ReLU()
    )


    # Branch B of the aggregation block (with 3D Transposed Convolution)
```

```
        self.tnet = torch.nn.Sequential(
            torch.nn.Conv3d(inf, outf, kernel_size=mask, padding=pad, stride=(step *2, step, step)),
            torch.nn.BatchNorm3d(outf),
            torch.nn.ReLU(inplace=True),
            torch.nn.ConvTranspose3d(inf, outf, kernel_size=(4,3,3),
                                     stride=(step *2, step, step), padding=pad),
            torch.nn.BatchNorm3d(outf),
            torch.nn.ReLU()
        )

        # Residual connection from input to output
        self.bypass = torch.nn.Sequential()

    def forward(self, x):
        net = self.net(x)
        tnet = self.tnet(x)
        bpass = self.bypass(x)
        out = net + tnet + bpass
        return out
```

Full Cost Aggregation Network - Chapter 7

```
class My3D(torch.nn.Module):
    def __init__(self, nol):
        super().__init__()
        # Input layer of the aggregation network
        self.in3d = Conv3DIn(1, 32,3 , 1, 1)
        layers = []
        # Append (nol) number of Aggregation Blocks
        for i in range(nol):
            layers.append(Conv3DMid(32, 32, 3, 1, 1))
        self.mid3d = torch.nn.Sequential(*layers)

        # Output layer of the aggregation network
        self.out3d = Conv3DOut(32, 1, 3, 1, 1)

    def forward(self, x):
        return self.out3d(self.mid3d(self.in3d(x)))
```

End-to-End ADSR-Net Implementation

Again, due to the modularity of the components developed during the study, the implementation of the complete ADSR-Net in PyTorch is clean and concise. The code snippet below shows the end-to-end ADSR-Net implemented using the modules shown earlier.

ADSR-Net Full Implementation - Chapter 7

```
class MyNet(torch.nn.Module):
  def __init__(self,bsize,height,width):
    super().__init__()
    self.bsize    = bsize          # Batch Size (Use 1 to avoid tensor mismatch)
    self.height   = height         # Image Height
    self.width    = width          # Image Width
    self.fnet     = FeatureNetII() # The feature network
    self.preagg   = PreAgg(64)     # The feature aggregation network
    self.my3d     = My3D(4)        # The cost aggregation network
    # The Up-sampling layer of the network
    self.upsample = torch.nn.Upsample(scale_factor=4, mode='bicubic', align_corners=False)

    # Network initialization code comes here

  def forward(self,left,right):

    left = self.fnet(left)    # Obtaining the left feature volume
    right = self.fnet(right)# Obtaining the right feature volume

    # Build the cost volume progressively and terminate when SNCE==0 return
    # the cost volume and the detected maximum disparity
    covol, maxd = calCVA(left, right, self.preagg)

    # Obtaining an intermediate pre-agg level disparity map for testing
    dmap1 = self.upsample(regress(covol) * 4)

    # Aggregate the costs, regress disparities and upsample to get final disparity map
    dmap2 = self.upsample(regress(self.my3d(covol)) * 4)
    sncm = [maxd]
    return dmap1, dmap2, sncm
```

Appendix D – Auto DSR During Training

When training a neural network with datasets, network parameters can be updated in three different ways. In case of “Batch Gradient Descent”, the network parameters are updated with the mean gradient for the entire batch whereas if “Mini-Batch Gradient Descent” is used, then the parameters are updated using the mean of the gradients pertaining to each data mini-batch. In “Stochastic Gradient Descent”, network parameters are updated using the output gradient from each data sample.

If SNCE is used to selectively terminate cost volume creation process at different maximum disparities while training the ADSR-Net, that can lead to discrepancies in the sizes of the corresponding cost-volume-tensors. In such instances, calculations related to the gradient propagation through the network can be affected by the mismatches in tensor dimensions. When using standard deep learning frameworks such as PyTorch and TensorFlow, the backpropagation process can terminate with an error.

This can be avoided by using a batch size of 1 (Stochastic Gradient Descent) during training if that is acceptable. However, if a larger batch size is required during training (i.e., batch or mini-batch gradient descent is used), it is possible to use a suitable fixed-sized cost volume during training (since the disparity ground-truth information is readily available). Once the training is complete, the pre-set maximum disparity can be removed, thereby allowing SNCE to be used to automatically estimate the maximum disparity during inference.

Appendix E – Dataset Statistics

Dataset	Variant	No of Images	Mean			Standard Deviation		
			R	G	B	R	G	B
Scene Flow	Driving (Forward and Backwards)	600	73.6061	83.7100	83.9837	55.5312	59.1187	60.7565
Scene Flow	Flying Things 3D	22390	92.9623	101.9831	107.2283	40.6729	44.4135	48.2968
KITTI 2015 Stereo	Stereo/Scene flow dataset	200	98.6081	102.3069	97.2213	80.2671	78.5655	75.8632

Table AppxE.1: Dataset statistics for the data used for training and validation of ADSR-Net. The mean and standard deviation have been calculated over the entire batch of images in each dataset

Appendix F - Camera/Lens Specifications

Camera

Part No	CM3-U3-13S2C-CS
ADC	12-bit
Chroma	Colour
Frame Rate	30
Lens Mount	CS-Mount
Mega-Pixels	1.3
Pixel Size	3.75
Readout Method	Global Shutter
Max Resolution	1288x964
Sensor Format	1/3”
Sensor Type	CCD
Sensor Name	Sony ICX445
Exposure Range	0.046ms to 31.9s
Gain Range	-11 dB to 23.991 dB
Dynamic Range	58.77
Interface	USB3 Gen 1

Lens

Part No	A4Z2812CS-MPIR
Focal Length	Variable
Focal Length (min)	2.8mm
Focal Length (max)	10mm
Lens Mount	CS-Mount
Iris	Manual
Megapixel	3
Format	1.27”

Appendix G – Notes on Performance, Calibration and Accuracy

Since ADSR-Net is able to dynamically adopt a suitable disparity search range for a given scene, the execution time for the algorithm depends on the proximity of the closest objects in front of the camera. In other deep stereo variants, larger “maximum disparity” values lead to much larger (and higher dimensional) cost volumes that affect the performance negatively. In contrast, due to the automatic selection of the disparity search range and special cost volume structure (with pre-aggregation), ADSR-Net is able to perform better compared to any reference implementation with a fixed sized cost volume.

Results shown in Table 7.1 and Table 7.2 Chapter 7 can be used as guidelines when using ADSR-Net for any real-world application. However, caution must be taken to carry out preliminary tests to compute frame rates for close range objects to find out if the delivered frame rate is sufficient for the application in concern. This is especially important in applications like close range obstacle avoidance which require much faster frame rates for making faster control decisions.

When using ADSR-Net with a custom-built stereo camera, stereo calibration is required to save the parameters for stereo image rectification. In such situations, the image rectification time must also be considered when computing the frame rates. When using standard stereo cameras, the output stereo image pairs can be used directly as inputs to the network.

The overall accuracy depends on the training level of the ADSR-Net model used. Again, the evaluation results shown in Section 7.5 can be used as a guideline. If a higher accuracy is required, the model can be modified (using the source code provided in Appendix C) or the network can also be trained with more data.