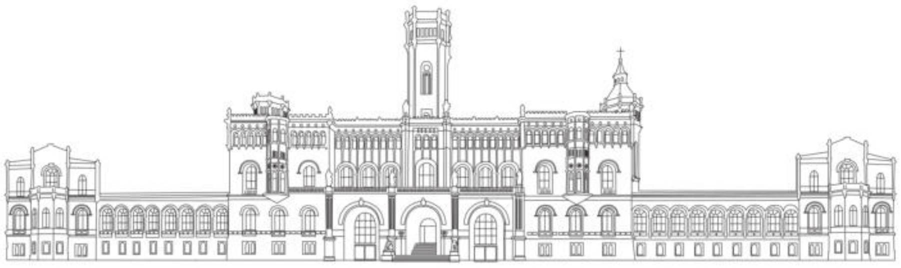
Uncertainty Awareness in Multi-View Stereo Reconstruction:

An Efficient Evidential Deep Learning Approach

Christian Grannemann
3129960

Institute of Photogrammetry and Geoinformation
Leibniz University Hanover



First Examiner: Prof. Dr.-Ing. habil. Christian Heipke
Second Examiner and Supervisor: Dr.-Ing. Max Mehltretter

To obtain the academic degree
Master of Science (M.Sc.)

14. August 2024

Abstract

The reconstruction of 3D models from multi-view images is a critical task in various fields such as autonomous driving and medical imaging. The accuracy and reliability of these reconstructions are paramount, particularly in those safety-critical real-time applications. This thesis addresses the challenge by introducing an Evidential Deep Learning (EDL) approach to Multi-View Stereo (MVS) reconstruction, enhancing the prediction of depth and associated uncertainty metrics without significantly increasing resource requirements.

Traditional MVS methods have achieved substantial progress by leveraging neural networks, yet the explicit prediction of uncertainty remains underexplored. Uncertainty in deep learning can be categorized into aleatoric uncertainty, which arises from data noise, and epistemic uncertainty, which stems from model limitations. Existing methods either overlook epistemic uncertainty or employ resource-intensive techniques to estimate it.

This thesis proposes a novel pipeline that integrates EDL into the MVS framework. EDL, based on the Dempster-Shafer Theory and probabilistic distributions like Dirichlet and Normal-inverse-Gamma, provides a robust measure of confidence by interpreting neural network outputs as parameters of an evidential distribution. This approach not only predicts depth but also quantifies the associated uncertainty, making it particularly suitable for applications requiring high trustworthiness.

The developed network, Evidential Multi-view Stereo Network (EMVSNet), incorporates EDL to produce both depth estimates and uncertainty metrics. Experiments conducted on standard datasets demonstrate that EMVSNet achieves competitive performance in depth prediction while providing meaningful uncertainty estimates. These metrics are crucial for identifying unreliable regions in the depth maps, thus enhancing the overall reliability of the 3D reconstruction.

The thesis begins with a review of foundational concepts and related work in MVS and uncertainty in neural networks. This is followed by a detailed presentation of the EMVSNet architecture and its components. Extensive experiments and results are then discussed to highlight the effectiveness of the proposed approach in various scenarios. Finally, the thesis concludes with an outlook on future research directions, emphasizing the potential of EDL to improve the robustness and applicability of MVS systems in critical real-world applications.

In summary, this work pioneers the integration of EDL into MVS, providing a comprehensive framework for accurate and reliable 3D model reconstruction with uncertainty awareness. This advancement paves the way for safer and more effective deployment of depth prediction systems in diverse fields.

Contents

Acronyms

AA-RMVSNet Adaptive Aggregation Recurrent Multi-view Stereo Network. 35, 48, 50, 58, 59, 61, 63, 90, 93

ADAM Adaptive Moment Estimation. 51

AUC Area under the curve. 63

BN Batch Normalization. 41, 45

BNN Bayesian Neural Network. 13, 14

BPA Basic Probability Assignment. 16, 17

CNN Convolutional Neural Network. 37, 41, 43

CV Computer Vision. 10, 29, 34

CVPR Computer Vision and Pattern Recognition. 1

DCN Deformable Convolutional Network. 38, 39

DLT Direct Linear Transform. 6

DSM Deep Stereo Matching. 31

DST Dempster-Shafer Theory. 15–18, 21

EDL Evidential Deep Learning. ii, 1–3, 12, 15, 18–23, 27, 28, 30, 31, 34–37, 43, 47–49, 58, 88–90, 92, 93

EER Equal Error Rate. 62

ELU Exponential Linear Unit. 47, 48

EMVSNet Evidential Multi-view Stereo Network. ii, 2, 35, 36, 43, 46, 48–50, 52, 53, 58, 59, 61–63, 67, 70, 71, 84, 85, 93

FIEDL Fisher Information-based Evidential Deep Learning. 31

FPR False Positive Rate. 49, 56

GN Group Normalization. 41

GPU Graphics Processing Unit. 50

ICCV International Conference on Computer Vision. 35

LLM Large Language Model. 30

LR Learning Rate. 51, 52

LSTM Long Short-Term Memory. 24, 36, 37, 41, 43

MAE Mean Absolute Error. 48, 54, 59, 61, 62

MCD Monte Carlo Dropout. 14, 29, 47, 48, 52, 53, 58, 88, 89, 93

ML Machine Learning. 25

MSE Mean Squared Error. 48, 53, 54, 61

MVS Multi-View Stereo. ii, 1, 2, 6, 9, 27–30, 32–35, 38, 50, 90, 92, 93

NiG Normal-inverse-Gamma. 15, 20–22, 30, 31, 35, 36, 43, 44, 46, 48

NLL Negative Log-Likelihood. 23

OOD Out of Distribution. 10, 31, 34, 49, 80, 92

OpenCV Open Source Computer Vision Library. 7

OSR Open Set Recognition. 30

PDF Probability Density Function. 18, 21

PPV Positive Predictive Value. 56

PR Precision-Recall. 49, 54–56, 59, 62, 63, 85

PSFA Plane Sweep Focus Algorithm. 8, 9

ReLU Rectified Linear Unit. 26, 38, 39, 47

RMSE Root Mean Squared Error. 54, 61

RNN Recurrent Neural Network. 37, 41, 43

ROC Receiver Operating Characteristic. 49, 54, 55, 63, 85

SfM Structure from Motion. 28, 29, 33

SIFT Scale-Invariant Feature Transform. 28

SotA State of the Art. 1, 30

SVD Singular Value Decomposition. 6

TnT Tanks and Temples. 33, 51, 58, 75–81, 92

TPR True Positive Rate. 49, 56

VRAM Video Random Access Memory. 79

Glossary

camera principal axis The camera's principal axis is a straight line that passes through the center of the lens and is perpendicular to the lens surface, serving as the path along which light rays travel undeviated. 3, 4

checkerboard artifact A visual distortion in image processing, particularly noticeable after deconvolution or transposed convolution operations, characterized by a grid-like pattern due to uneven overlap and gaps in the upsampled feature maps. 70, 92

chromatic aberration Chromatic aberration is a lens distortion where different colors focus at varying distances, causing colored fringes at high-contrast boundaries in an image. 6

extrinsic camera matrix The extrinsic camera matrix describes the transformation from the world coordinate system to the camera coordinate system, capturing the camera's position and orientation in the world. 3–5, 8

focal length The focal length is the distance between a camera's lens and its image sensor when the lens is focused at infinity. 5

intrinsic camera matrix The intrinsic camera matrix represents the internal parameters of a camera, such as focal length and optical center, that relate the camera's 3D world coordinates to its 2D image coordinates. 3–5, 7, 8

percentile curve A percentile curve is a graph that shows the percentage of data points that fall below each value on the x-axis, providing a visual representation of the cumulative distribution of the data. 71

principal point The principal point is the point on the image plane where the camera's optical axis intersects. 6

projection center The projection center is the (virtual) point inside the camera where all incoming rays intersect. 3–5, 8

PyTorch An open-source machine learning library by Facebook's AI Research lab, known for its dynamic computational graphs, GPU support, and extensive ecosystem for deep learning applications. 48, 50, 52

relative orientation The spatial relationship between two cameras, defined by their relative rotation and translation. 8

semantic segmentation Semantic segmentation is a computer vision task that involves assigning specific class labels to each pixel in an image, enabling the identification and differentiation of objects and regions based on their semantic meaning. 32

vanishing gradient problem The vanishing gradient problem occurs in deep neural networks when gradients of the network's loss function become too small as they are propagated back through the layers during training. This results in the weights in earlier layers changing very little or not at all, which significantly slows down the learning process or causes it to stall completely. 41, 44

voxel A voxel is a three-dimensional pixel used to represent values on a regular grid in 3D space, analogous to a pixel in 2D space. 28

Institut für Photogrammetrie
und GeoInformation

Leibniz
Universität
Hannover

Uncertainty-aware 3D Reconstruction from Multi-view Stereo

Proposal for a Master thesis topic (DE/EN)

Nienburger Straße 1, 30167 Hannover
Fakultät für Bauingenieurwesen und Geodäsie

Institut für Photogrammetrie
und GeoInformation

Prof. Dr.-Ing. habil. Christian Heipke

Dr.-Ing. Max Mehltretter

Tel.: +49 511 762-2981
Fax: +49 511 762-2483
E-Mail:
mehltretter@ipi.uni-hannover.de

23 January 2023

Multi-view stereo matching is commonly one of the first steps of a reconstruction pipeline aiming to reconstruct the 3D geometry of a scene using multiple images taken from different viewpoints. Inaccurate or erroneous results of this first step often have a negative impact on the subsequent processing steps and thus on the final 3D reconstruction itself if such errors are propagated unknowingly. One way to overcome this limitation is the estimation of the uncertainty associated to the depth estimates that are the result of the multi-view stereo matching process. While a number of approaches for uncertainty estimation have been presented in the literature for the binocular stereo case in recent years, the more general case of multi-view stereo has mostly been neglected.

The objective of this master thesis is to develop a methodology for predicting the uncertainty associated to multi-view stereo matching-based depth estimates, using a Convolutional Neural Network (CNN). For this purpose, first, a suitable baseline addressing the task of multi-view stereo matching is to be defined based on methodology presented in the literature. Afterwards, this baseline is to be extended to allow the prediction of both, depth and its associated uncertainty. The developed approach is to be evaluated with respect to the quality of the predicted uncertainties using different datasets to investigate the suitability of the employed model as well as the approach's general validity.

Thus, in addition to the conceptual elaboration, this master thesis also includes a practical implementation of the developed methodology. As basis for this implementation, methodology and software is provided, that was developed earlier at the Institute of Photogrammetry and GeoInformation. For the purpose of evaluation, the developed approach is to be tested on different publicly available datasets with known reference for the scene geometry. The acquisition of own data is not necessary within the scope of this work.

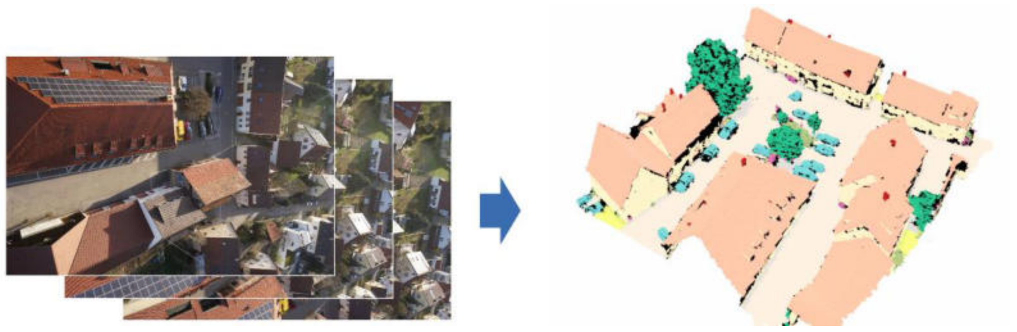


Figure 1: Example of a multi-view stereo reconstruction. Multiple images of the same urban scene taken from different viewpoints under slightly varying conditions serve as input to the multi-view stereo procedure to be developed to reconstruct the 3D geometry of this scene.

This thesis will be supervised by Max Mehltretter.

Besucheradresse:
Nienburger Straße 1
30167 Hannover
www.ipi.uni-hannover.de

1 Introduction

The reconstruction of 3D models from multi-view images has become one of the most prominent topics at the CVPR conference [1], reflecting a significant surge in interest and research within this field. This heightened focus is driven by the vast array of potential applications across various industries. For instance, autonomous vehicles rely on precise depth maps for navigation and obstacle avoidance, ensuring passenger safety and efficient route planning. In medical imaging, accurate 3D reconstructions are crucial for precise diagnosis and effective treatment planning. Given the critical nature of these applications, the trustworthiness of the results produced by such systems is paramount. It is often more essential to identify predictions that may be unreliable than merely achieving high reconstruction performance, especially when these systems are deployed in real-time applications.

As a foundational step in achieving complete 3D reconstruction from multiple images of an object, Multi-View Stereo (MVS) refers to the estimation of depth by incorporating multiple images from different perspectives. Various approaches have been developed to tackle this task, with recent methods predominantly utilizing neural networks to enhance performance.

Despite the significant advancements in MVS methods, the prediction of uncertainty remains underexplored. While some studies have addressed the prediction of uncertainty alongside depth estimation to better manage regions with unreliable data, the epistemic type of uncertainty is frequently overlooked. In instances where epistemic uncertainty is considered, the employed techniques are often resource-intensive.

This thesis seeks to enhance this practice by developing a pipeline that predicts depth in a reference image from MVS, accompanied by relevant uncertainty metrics, while minimizing additional resource requirements. Given that State of the Art (SotA) methods utilize neural networks, this research investigates deep learning approaches to address the challenges of uncertainty prediction.

The concept of Evidential Deep Learning (EDL), introduced by Sensoy et al. [2] in 2018, integrates uncertainty estimation into deep neural networks by interpreting the network's outputs as parameters of an evidential distribution. This approach captures both the predictive mean and the associated uncertainty, providing a more robust measure of confidence in the predictions. Neural networks using this technique can be trained end-to-end with an adapted loss function and can provide uncertainty estimations during inference, minimizing additional resource requirements.

To the best of the author's knowledge, this work introduces EDL into the field of MVS for the first time, presenting a novel approach with several potential advantages for real-time applications. The utilization of EDL offers multiple benefits, which will be elaborated upon in this thesis:

The reader will be introduce to fundamental image models and the different types of uncertainty in neural networks in chapter 2. Additionally, the mathematical background and the overall concept of EDL will be explained. Given the substantial interest and research efforts dedicated to MVS methods, chapter 3 introduces various established concepts in this field. This chapter also covers related applications of EDL, other uncertainty determination concepts, and datasets for MVS. Chapter 4 then presents the network structure developed in the course of this work, along with suggestions for adjustments and extensions. Building on the theoretical background and the network referred to as Evidential Multi-view Stereo Network (EMVSNet), setup and training process to conduct various experiments is introduced in chapter 5. Chapter 6 then details about those experiments designed to measure the network's prediction performance and compare the newly introduced concept of EDL to traditional methods of uncertainty prediction.

In summary, this thesis seeks to significantly advance the field of 3D model reconstruction from multi-view images by incorporating uncertainty prediction within the MVS pipeline using EDL. This innovative approach is designed to improve the reliability and robustness of depth prediction systems, ensuring their effectiveness in critical applications requiring accurate 3D reconstruction. By providing a framework for accurately estimating and managing uncertainty, the proposed method aims to pave the way for safer and more precise implementations in real-world scenarios, thereby enhancing the overall trustworthiness and performance of these systems.

2 Basics

This chapter introduces foundational concepts essential for understanding the subsequent sections of this thesis. The discussion begins with the camera model in section 2.1, detailing how a camera captures light information and the mathematical relationship between the real world and the image sensor. This includes the extrinsic camera matrix for transforming world coordinates to camera coordinates, and the intrinsic camera matrix for internal camera parameters, as well as addressing lens distortion. Next, the Plane Sweep Focus Algorithm is delved into in section 2.2, which is used for depth estimation by leveraging well-calibrated cameras and known extrinsic parameters, reducing the 2D search problem in stereo images to a 1D line search problem. The types of uncertainties, namely aleatoric and epistemic, that affect deep neural networks are examined in section 2.3. Section 2.4 introduces the EDL framework, that incorporates uncertainty estimation into neural networks. Based on the mathematical concepts discussed before, this concept should be suitable to enhance prediction accuracy and uncertainty estimation. Finally, section 2.5 presents a fundamental network layout and mathematical function used inside the network's structure.

2.1 Camera model

In general, a camera captures the information of light emitted by surfaces in the real world on its sensor. To describe the relationship between the coordinates of a point in the real world and the orientation of the camera, points can be described as P_w and P_c with

$$P = \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{2.1}$$

as homogeneous 3D coordinates in either coordinate system.

Extrinsic camera matrix

The extrinsic camera matrix then describes the transformation from world coordinates to camera coordinates. The camera coordinate system is located in the projection center. In most cases, the optical axis Z_c from the projection center to the origin of the camera coordinate system is aligned with the camera principal axis. The extrinsic matrix can be described by the concatenation of the rotation matrices (2.2-2.4) around each axis [4].

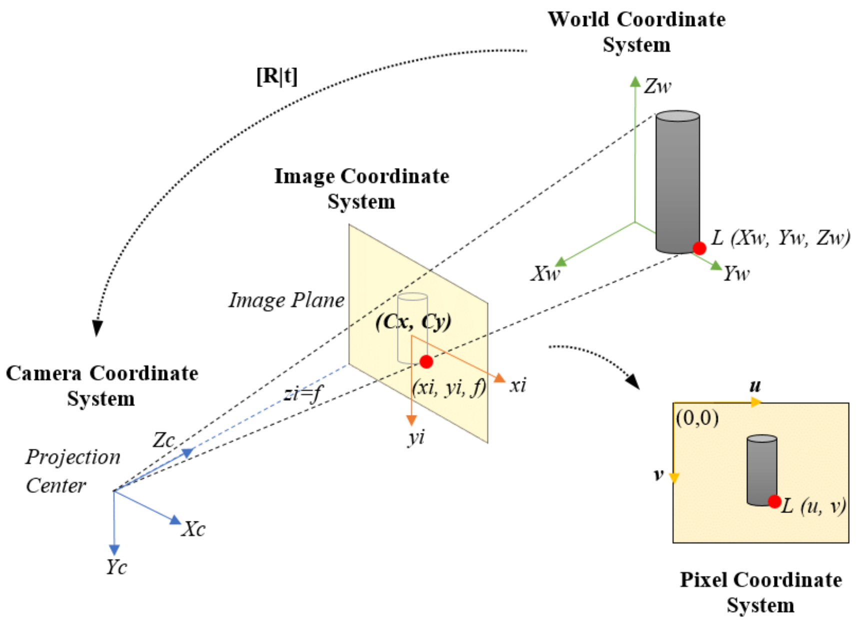


Figure 2.1: Model of different coordinate systems to describe the relationship between points in the real world and the pixel on the image sensor. The extrinsic camera matrix is used for conversion from the World Coordinate System to the Camera Coordinate System. The intrinsic camera matrix is used for further conversion to pixel coordinates. This illustration is simplified, in reality the image plane lies behind the projection center in direction of the camera principal axis. [3]

$$R_x = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\phi) & -\sin(\phi) & 0 \\ 0 & \sin(\phi) & \cos(\phi) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{2.2}$$

$$R_y = \begin{pmatrix} \cos(\theta) & 0 & \sin(\theta) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{2.3}$$

$$R_z = \begin{pmatrix} \cos(\psi) & -\sin(\psi) & 0 & 0 \\ \sin(\psi) & \cos(\psi) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{2.4}$$

To include the translation (2.5), typically the representation in homogeneous coordinates is used.

$$T_{xyz} = \begin{pmatrix} 0 & 0 & 0 & t_x \\ 0 & 0 & 0 & t_y \\ 0 & 0 & 0 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{2.5}$$

The concatenation of these matrices gives the 4x4 extrinsic camera matrix which encapsulates rotation and translation from world to camera coordinates.

$$[R \mid t] = R_x \times R_y \times R_z \times T_{xyz} \tag{2.6}$$

Intrinsic camera matrix

The fundamental intrinsic parameters of a camera can be described by the pinhole camera model. In this model, the law of similar triangles leads to the relationship between a point P_c in the 3D camera coordinate system and a point P_i in the two-dimensional image coordinate system located on the cameras image sensor. Equation 2.7 shows how this law leads to the coordinate X_i in image coordinates. The distance from the image sensor to the projection center is given as focal length f.

$$\frac{X_i}{f} = \frac{X_C}{Z_C} \Rightarrow X_i = f\frac{X_C}{Z_C} \tag{2.7}$$

To account for potential non-square pixels or different scaling factors in the x and y directions of the image sensor, the focal length is split into f_x and f_y. With this fundamental connection the main diagonal of the intrinsic camera matrix K (2.8) can be filled. The pixel coordinate system has its origin at the top-left corner, so c_x and c_y are the translation from this origin of the pixel coordinate system to the principal point.

$$K = \begin{pmatrix} f_x & s & c_x & 0 \\ 0 & f_y & c_y & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \tag{2.8}$$

In addition, a skew-parameter s is introduced, accounting for non-orthoganility between the x and y axis of the image sensor with angle α between the axis. In modern cameras, this angle can typically assumed to be 90 degrees.

$$s = -f_x \cdot \tan(\alpha) \tag{2.9}$$

Complete model

The complete camera model describes the rotation and translation from the world coordinate system to the pixel coordinate system 2.10.

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{pmatrix} f_x & s & c_x & 0 \\ 0 & f_y & c_y & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \cdot \left(\begin{array}{ccc|c} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{array}\right) \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{2.10}$$

As a result, the intrinsic matrix is given as an upper right triangular matrix. In concatenation with the extrinsic matrix as orthonormal matrix, Singular Value Decomposition can be used to determine the components of these matrices by measuring 6 point correspondencies and performing Direct Linear Transform. As in many cases the intrinsic camera calibration is already given and 3D pose estimation of the camera are done separately in two steps, the following paragraphs delve deeper into this procedure.

Note that, due to the quality of digital sensors, one rarely estimates the 11 parameters of the projection matrix. In particular, pixels are assumed to have no skew ($s = 0$), and be square ($f_x = f_y$). Also, if an image has not been cropped, it is safe to assume the principal point is at the center of the image. As a result, a common pinhole camera model is just composed of 7 parameters: the focal length f, the rotation matrix R and the translation vector T.

If using rectifying wide-angle images, resampling artifacts will be introduced as well as field of view cropping. To avoid these issues, MVS pipelines can support radial distortion and more complicated camera models directly, at the expense of extra complexity.

Lens distortion

The equation 2.10 only accounts for the pinhole camera model and does not include various distorting effects like chromatic aberration introduced by the lens [5]. In basic camera models, mainly radial and tangential distortion are considered. Radial distortion (2.11) can be seen as curving of straight lines in an images, either inward (pincushion distortion) or outward (barrel distortion).

$$\begin{aligned} x_{\text{radial, distorted}} &= x\left(1 + k_1 r^2 + k_2 r^4 + k_3 r^6\right) \\ y_{\text{radial, distorted}} &= y\left(1 + k_1 r^2 + k_2 r^4 + k_3 r^6\right) \end{aligned} \tag{2.11}$$

The corrected coordinates in equation 2.11 depend on the radial distortion coefficients k_1, k_2 and k_3 and the euclidean radius r from the image center. Often k_3 can be neglected if the lens is not wide-angle.

Tangential distortion (2.12) occurs when the lens and image plane are not parallel and is visible in an image by tilting straight lines. The tangential distortion coefficients p_1 and p_2 are considered in the model.

$$\begin{aligned} x_{\text{tangentail, distorted}} &= x + \left[2p_1xy + p_2\left(r^2 + 2x^2\right)\right] \\ y_{\text{tangentail, distorted}} &= y + \left[p_1\left(r^2 + 2y^2\right) + 2p_2xy\right] \end{aligned} \tag{2.12}$$

Camera calibration

For the task of calibrating a camera, the method introduced by Zhang [6] in the year 2000 still is used widely, e.g. by OpenCV, to receive the intrinsic camera matrix and lens distortion parameters. In general, arbitrary points with known position in a 3D world coordinate system could be used for camera calibration. But measuring the position of these points and finding their correspondences in multiple images automatically suggests to better use a repetitive 2D pattern with straight lines like a checkerboard for the task of camera calibration.

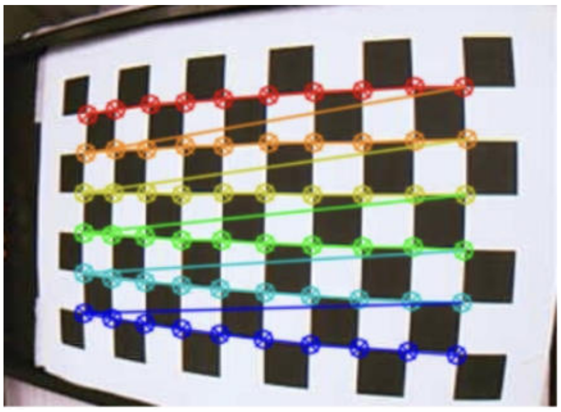


Figure 2.2: Checkerboard used for camera calibration.

As a trick, for every frame used for calibration, the orientation of the coordinate system is set with the checkerboard oriented in the x-y plane. This simplifies equation 2.10 to 2.13.

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{pmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \times \left(\begin{array}{cc|c} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ r_{31} & r_{32} & t_z \end{array}\right) \times \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = H \times P \tag{2.13}$$

The homography matrix H then needs at least 4 point correspondences in each image. To solve the given equations, at least three views of the plane must be given. The parameters of the extrinsic camera matrix then stay the same for every point extracted from a single point of view, while the parameters of the intrinsic camera matrix are consistent for all observations. The exact algorithm is well explained in full detail by Cyrill Stachniss. [7]

2.2 Plane Sweep Focus Algorithm

With well-calibrated cameras and known relative orientation between two camera views, epipolar geometry can constrain the 2D search problem of finding corresponding points in both images to a 1D line search problem. Figure 2.3 shows the epipolar geometry for a pair of left and right images. In addition, the fundamental assumptions of this constraint

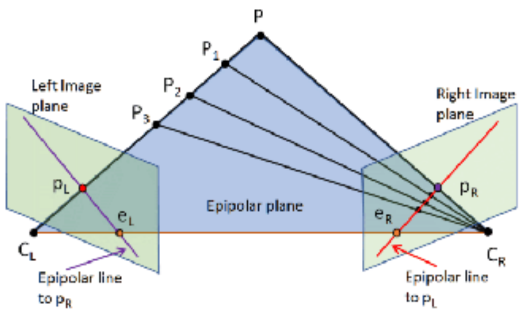


Figure 2.3: Epipolar geometry reduces the search region for point correspondences to a line. The epipolar plane is formed by the projection center C_1 and C_2 of both cameras as well as the point P in the real world. This is the foundation for the Plane Sweep Focus Algorithm (PSFA). [8]

can be used to perform the PSFA. If the intrinsic and extrinsic parameters of both views are known, the view from Camera 2 can be transformed to match the view of Camera 1 through a process known as image rectification, which aligns the epipolar lines but does not necessarily represent a homography. Image rectification modifies the images from both cameras so that the corresponding epipolar lines in each image become parallel and horizontal, simplifying the matching process between images. The transformation is typically represented by a 3×3 matrix that adjusts the orientation and scale but retains the perspective of the original images.

The following explanation is rather untypical for the PSFA but should clarify how the algorithm works. As visible in figure 2.3, the depth of a point P determines where on the epipolar line the point's projection in the second image plane can be found. The virtual points $P_{1..3}$ now represent one pixel in the first image but different pixels in the second image. As the projected image plane is swept through different depth hypotheses, the projection of a point only aligns with the pixel's position in the second image if the depth of the point in 3D space is correct for this depth hypothesis. The sweeping through different depth hypotheses does not typically result in visual focus or blur but involves comparing different alignments or disparities to determine the best depth estimate. Misalignments or incorrect depths will not show the typical visual effects of focus but rather a lack of correspondence between the points in the image pairs.

Cost Volume Construction

The PSFA is used to construct Cost Volumes, which are crucial data structures in MVS techniques, particularly when implemented in neural networks. Cost Volumes are 3D volumes where each element corresponds to a potential depth value at each pixel location in the image. This construction starts by defining a set of depth hypotheses across the scene's possible depth range. For each depth hypothesis, the algorithm transforms or *warps* the images from multiple views to what they would look like if the scene was located at that particular depth. This warping is performed using the known camera parameters and a simplified projection model assuming the depth hypothesis is correct. Once all images are warped to a common depth hypothesis, the next step is to calculate the similarity or *matching cost* between the corresponding pixels of these warped images. The similarity measurement typically involves comparing features extracted from the images, such as gradients, colors, or more complex descriptors. These features are designed to be robust against changes in viewpoint and lighting, enhancing the reliability of the matching cost. The calculated costs for each pixel across all depth hypotheses are then compiled into the Cost Volume, where the value at each location provides a measure of how well the pixels from different views agree with that depth hypothesis. Only if the actual depth of a real-world point corresponds to one of the depth hypotheses will the features from the images align well, resulting in a low cost. The ideal scenario is when pixels representing the same 3D point in different images overlap perfectly, indicating that the depth hypothesis might be accurate for that point.

However, several challenges affect the reliability of Cost Volumes. Non-Lambertian surfaces, which do not reflect light uniformly in all directions, and untextured areas, which lack sufficient detail for matching, complicate the creation of accurate Cost Volumes. These issues can lead to ambiguous cost measurements along the epipolar line, resulting in errors in depth estimation. Moreover, the resolution of the Cost Volume and the granularity of depth hypotheses directly impact the precision and computational requirements of the MVS system. A higher number of depth levels increases the resolution but also the computational load and memory usage. Thus, optimizing these parameters is crucial for balancing accuracy and performance in real-world applications.

In advanced implementations, neural networks are employed to refine the initial estimates from the Cost Volumes. These networks can learn to identify and correct typical errors by analyzing large datasets of images, thereby enhancing the depth estimates beyond what traditional matching algorithms can achieve. The integration of machine learning thus represents a significant advancement in the field of MVS, opening up new possibilities for accurate 3D reconstruction in diverse applications.

2.3 Uncertainty in deep neural networks

The continual advancements in computational power, along with the availability of large amounts of rapidly accessible memory, have made it feasible to tailor neural networks to a diverse array of tasks. Neural networks consist of layers of interconnected nodes, modeled after biological neurons, organized in a repetitive architecture. Theoretically, sufficiently deep neural networks have the capacity to approximate any linear or nonlinear function [9]. Consequently, they are increasingly employed in complex Computer Vision (CV) tasks that were previously infeasible to perform artificially.

The following subsection 2.3.1 explores the various types of uncertainty that can be identified in the context of neural networks. Subsection 2.3.2 introduces several methods commonly used to quantify uncertainty according to these identified categories.

2.3.1 Categorization

When training neural networks, it is often assumed that testing data at least falls into the same category as the training data. A neural network exclusively trained on dog images will never be able to distinguish between a *Mustang* or a *Golf* or even a car and a phone, as the network has never had any information about these instances and is simply unable to tell them apart. Even worse, it might happen that these car test images get related to a specific dog's race with high probability because this so called Out of Distribution (OOD) data has not been seen before.

As neural networks are more commonly deployed in the real world and potentially are used in safety-critical areas of application, the determination how uncertain a model's prediction is becomes more important. It is especially important to distinguish between the confidence of a model and a model's uncertainty about a prediction. The confidence of a prediction can be seen as the probability the network assigned to this outcome. Especially in classification tasks the remaining problem can be seen as there might be classes that have not been learned by the network - so they can not be predicted. But the network is predicting a result for a given input and its confidence could be high for a known class although it should know that the input does not fit to the information it was trained on.

As discussed in the MIT lecture [10], it can be distinguished between four different types of uncertainty:

- **Known Knowns:** Things we are certain of.
- **Known Unknowns:** We know there are things we can not predict.
- **Unknown Knowns:** Others know but we don't know.
- **Unknown Unknowns:** Completely unexpected or unforeseeable events.

These categories provide a useful framework for understanding different kinds of uncertainty. In the context of neural networks, this framework can be mapped to two specific types of uncertainty, namely aleatoric and epistemic.

Aleatoric uncertainty, also known as statistical uncertainty, corresponds to the inherent noise in the data. It is similar to *Known Unknowns*, meaning there are aspects that cannot be predicted due to the variability in the data itself. This type of uncertainty remains even if an infinite amount of data was available.

Epistemic uncertainty, on the other hand, refers to the uncertainty in the model parameters and structure, similar to *Unknown Knowns* and *Unknown Unknowns*. This type of uncertainty can be reduced with more data or a better model. It represents what the model does not know due to limitations in its training data or design.

Following the approach of Kendall and Gal [11], this work distinguishes between these two types of uncertainty to better manage and understand model predictions.

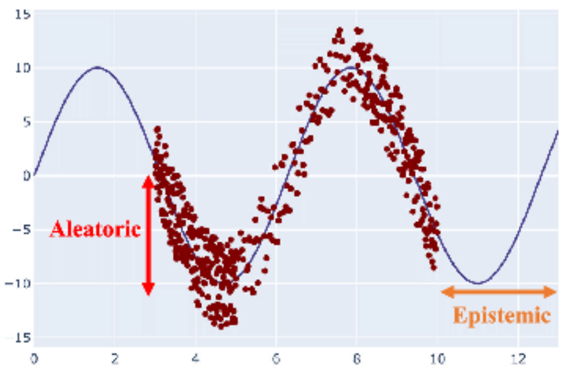


Figure 2.4: Data points with noise given to extrapolate a trigonometric function. The noise inherent in the data describes the aleatoric uncertainty, while epistemic uncertainty can be found in parts where no data is given. [12]

Aleatoric Uncertainty

Aleatoric uncertainty describes an irreducible type of uncertainty inherent in the data itself. This uncertainty arises from the randomness or inherent variability in the system being modeled, such as sensor noise or other forms of observational imprecision. A real-world example of aleatoric uncertainty is the unpredictable nature of rolling dice, where the outcome relies on factors impossible to measure with certainty. In the context of machine learning, aleatoric uncertainty manifests in the data used to train neural networks, where it remains present even if the volume of training data is increased. This is because the variations lie within the data itself, not in the knowledge or understanding of the data. Figure 2.4 shows data points, e.g., taken from noisy measurements, representing a function. It can be observed that aleatoric uncertainty prevails where data is given due to the noise in the data. While the function can be extrapolated from the data provided, a certain level of uncertainty always remains, indicating the irreducible nature of this type of uncertainty.

Furthermore, *aleatoric uncertainty can be decomposed into homoscedastic uncertainty* *and heteroscedastic uncertainty* [13]. Homoscedastic uncertainty is uniform across all observations and does not change with the input data. Heteroscedastic uncertainty, in contrast, varies with the input data, becoming more pronounced in certain regions of the input space than in others.

For instance, in financial markets, distinguishing between homoscedastic and heteroscedastic aleatoric uncertainty is crucial for developing predictive models. During volatile periods, such as economic downturns or major political events, financial instruments exhibit wide price fluctuations. This heightened volatility leads to greater variability in price movements, or heteroscedastic uncertainty, increasing the overall aleatoric uncertainty. Models trained on financial data need to incorporate this variability to accurately estimate risks during these unstable periods. Conversely, during stable market phases characterized by minor price changes, the forecasting uncertainty is typically lower. The inputs — market conditions — are more consistent and predictable, which reduces prediction fluctuations. In such scenarios, understanding and modeling heteroscedastic uncertainty allows financial models to adjust their confidence levels appropriately, adapting to the prevailing market conditions.

In the course of this work, one methodology for discrimination between the homoscedastic and heteroscedastic type of aleatoric uncertainty using EDL will be considered.

Epistemic uncertainty

Epistemic uncertainty, or model uncertainty, describes uncertainty that could potentially be reduced with more data or better models. This type of uncertainty is inherent in the model itself and manifests when there is insufficient data to support predictions in certain areas. Figure 2.4 illustrates this concept, showing regions where the model extrapolates beyond the available data, marking these areas as having high epistemic uncertainty. If additional data were provided in these areas, the epistemic uncertainty could be reduced.

Like aleatoric uncertainty, different sub types of epistemic uncertainty can be distinguished from each other. The following list provides an overview about possible subtypes.

Subtypes of Epistemic uncertainty:

- **Parameter uncertainty**: This involves uncertainty about the optimal parameters within a model, reflecting a lack of precise knowledge about the best values for these parameters.
- **Model Structure uncertainty**: Concerns the uncertainty regarding whether the chosen model structure adequately captures the underlying data patterns.
- **Data uncertainty**: Reflects the limitations in the datasets representativeness or size, influencing how well the model can generalize.
- **Algorithmic uncertainty**: Arises from the limitations in the algorithms used

to train models, including convergence issues or biases introduced by the training process.

- **Scenario uncertainty**: Involves uncertainties due to unknown future conditions or actions that are not captured in the model.

In the scope of this thesis, the last three types of uncertainty will be explored and compared. Modeling *Parameter* and *Model Structure* uncertainty can be especially hard and is out of scope for this work.

Summary

In mathematics and statistics, uncertainty is frequently characterized in terms of a probability distribution, which serves as a fundamental concept for quantifying and managing uncertainty in predictive modeling. While epistemic uncertainty reflects a lack of knowledge about the most appropriate probability distribution to model the underlying process, aleatoric uncertainty relates to the inherent variability present in the system being observed. Both types of uncertainty play pivotal roles in the development and application of statistical models.

Understanding epistemic uncertainty can lead to improved model selection and a more precise estimation of model confidence, which is particularly vital in areas where data is scarce or highly complex. On the other hand, effectively capturing aleatoric uncertainty is crucial for realistic risk assessment and informed decision-making under conditions of inherent variability.

By distinguishing between these uncertainties and recognizing their implications, statisticians and data scientists can tailor their models to better reflect the realities of the data, enhancing the reliability, interpretability, and overall effectiveness of their analytical outcomes.

2.3.2 Traditional approaches

The exploration of uncertainty within the realm of neural networks remains an area of open challenges to overcome. This section introduces methods to estimate different types of uncertainty in the context of neural networks, while looking at the challenges, especially in form of increased computational resource demands and the intricacies of integrating uncertainty estimates into decision-making frameworks. The differentiation between aleatoric and epistemic uncertainty, which has been introduced in the previous subsection 2.3.1, remains relevant in context of this section as there are different approaches needed to quantify the different types of uncertainty.

Bayesian Neural Networks

Bayesian Neural Networks (BNNs) are an extension to the traditional approach of modeling weights and biases in neural networks. They extend the traditional method by incorporating Bayesian models, which allow a probabilistic interpretation of the model's parameters [14]. In contrast to a traditional neural network, BNNs treat weights as random variables, described by probabilistic distributions. A key element in this approach is the Bayes Theorem (2.14), which updates the probability estimate for a hypothesis as additional evidence is given.

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)} \tag{2.14}$$

In the context of BNNs, A typically represents a specific set of values for weights and biases as network's parameters, and B represents the observed data. Initially, the network's parameters are assigned prior distributions that express the beliefs about these parameters before any data has been given. As data is processed by the network, the Bayesian update mechanism uses the Likelihood of the observed data under different parameter configurations to update the posterior distribution of the parameters. This posterior distribution encapsulates everything the network has learned from the data, balancing the prior beliefs and the Likelihood of the observed data. BNNs can be particularly useful to model uncertainty and improve decision making in scenarios where data may be scarce or noisy.

The update mechanism in BNNs operates iteratively. Each new piece of data provides evidence that is used to update the posterior distribution of the network's parameters. This process is known as the already mentioned Bayesian updating mechanism. Mathematically, it involves recalculating the posterior distribution using the new data's Likelihood combined with the prior distribution, which is then normalized by the marginal probability of the new data. This iterative process continues as more data is observed, with the posterior distribution after processing one data point becoming the prior distribution for the next. Over time, this method adjusts the neural network's parameters to better model the underlying data structure, improving the network's predictions and uncertainty estimations.

Monte Carlo Dropout

A seminal paper by Gal and Ghahramani [15], titled *Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning* from 2016 presents dropout as a way to approximate Bayesian inference in deep neural networks. This work is foundational in linking dropout to epistemic uncertainty.

As one approach that can be categorized under the Bayesian based approaches, this technique called Monte Carlo Dropout (MCD) has been established [15]. Using dropout while training is a regularization technique to prevent overfitting, with dropout referring to deactivating some neurons with their connection. When introducing this technique into inference of a network if running multiple times, the variance in the model's predictions can quantify the underlying uncertainty. This would include aleatoric as well as epistemic uncertainty. To distinguish between both types of uncertainty, one approach could be to use the variability of multiple measurements as indicator for aleatoric uncertainty. If the total uncertainty is determined by the presented dropout technique, epistemic uncertainty could be calculated by subtracting the aleatoric uncertainty from the total uncertainty gathered by multiple forward passes through the same network using Monte Carlo Dropout.

Deep Ensembles

Another, rather computationally intensive approach is known as Deep Ensembles. This method involves training of multiple neural networks, potentially with varying initial weight parameters, and comparing their outcome [16]. Furthermore, there could be different subsets of the same training data or different hyperparameters be used for training of the different models. On one hand, this makes the ensemble more robust to overfitting by cancelling out the faulty predictions of single members when combining the results. On the other hand, this provides the opportunity to quantify the uncertainty inherit in the data or the model. One approach to estimate aleatoric uncertainty would be to train the network directly to predict parameters of a probability distribution rather than a simple estimation. With different members of the same underlying network, the variability in the predictions from the members reflects the epistemic uncertainty involved in the process. If multiple members of the same network, all trained slightly different, conclude to nearly the same outcome for a sample, in general they are certain about their prediction and the estimation can be classified as robust. If the prediction varies widely although there is only slight difference between the members of the model, it can be assumed that the prediction of the ensemble has high epistemic uncertainty.

While providing a robust framework for estimation of uncertainty, especially the epistemic type, using multiple models while training and inference introduces a massive overhead computationally and prevents this method, like Deep Ensembles, from real-time applications as inference of dozens of models is rarely feasible in parallel.

2.4 Evidential Deep Learning

The concept of Evidential Deep Learning (EDL) was introduced in 2018 by Sensoy et al. [2]. It adapts the Theory of Belief function from the Dempster-Shafer Theory (DST) to the broader field of neural networks. As it marks the foundation for the concept, DST is introduced first in section 2.4.1 to understand the basics of the concept. Using the Dirichlet or Normal-inverse-Gamma (NiG) distribution, the following two sections 2.4.2 and 2.4.3 provide the mathematical background by diving deeper into the characteristics of these distributions. The following section 2.4.4 explains how this approach of handling probabilities is incorporated into deep learning and can be used to obtain measures of aleatoric as well as epistemic uncertainty. In addition, variations of the original concept as well as examples for the usage of EDL are presented.

2.4.1 Dempster–Shafer Theory

The DST, which is a general framework for reasoning with uncertainty, was published in 1976 by Shafer [17] after initial work done by Dempster [18] in 1967. Inside most

Bayesian probability modeling frameworks, uncertainty is represented by the distribution of probabilities for different outcomes, which have to sum up to one. This traditional type of probability modeling updates a prior belief using the observed data, resulting in updated probabilities for the initial belief.

Mathematical concept

This concept differs substantially from the DST's way of modeling uncertainty. Uncertainty could occur from sources of different evidence or from under circumstances where the relationship between cause and effect is unclear. The restriction of a probability sum of one is not given for the DST where belief can also be assigned to a set of outcomes, which includes the option to assign a probability to the fact that the outcome for a given input is unknown. In DST, besides the belief for a certain outcome there is plausibility for this outcome given as well. The belief function *Bel* describes the probability for different outcomes from the frame of discernment Θ. The frame of discernment describes a set of alternatives that are complete and incompatible, so at least and at most one statement is true. The belief function (equation 2.15) now accumulates all probability masses that support the belief.

$$\mathrm{Bel}(A) = \sum_{X \subseteq A} m_X \tag{2.15}$$

The plausibility function *Pl* (equation 2.16) on the other hand measures how much the evidence does not refute each possible alternative. It is calculated by the sum over all Basic Probability Assignments (BPAs) that are not reasoning against the belief of a given output A.

$$\mathrm{Pl}(A) = 1 - \mathrm{Bel}(\neg A) \tag{2.16}$$

In DST, for the entirety of Θ there can be a BPA assigned as well. This would mean there is no evidence for a subset of Θ, thus describing a scenario of complete uncertainty. For the following example with possible weather conditions, this would mean any of the included weather conditions is given. As this does not stand against any of the subsets of Θ, this would be included in the plausibility function of all weather conditions.

To calculate the joint mass of belief functions, Dempster's rule of combination, shown in equation 2.17, is used.

$$m(A) = \begin{cases} 0, & \text{if } A = \emptyset, \\ \frac{1}{1-K} \sum_{B \cap C = A} m_1(B) \cdot m_2(C), & \text{otherwise,} \end{cases} \tag{2.17}$$

For this equation, K can be seen as conflict between the two mass functions. This normalization factor covers the belief for all cases that do not support the combined statement A given by the combination of B and C.

$$K = \sum_{B \cap C = \emptyset} m_1(B) \cdot m_2(C) \tag{2.18}$$

Real-world example

To clarify the concept behind DST and its special attributes, a frame of discernment Θ with different weather events is given as an example.

$$\Theta = \{Cloudy, Rainy, Windy\} \tag{2.19}$$

Assuming there are the following BPAs which have to sum up to one as they represent proportions of the total belief. Each of the three weather conditions has its own probability to occur, but some combinations like weather that is *Cloudy or Rainy* has its own source of evidence and is especially likely. In this example, this event gets its own probability.

$$\begin{aligned} m(\{Cloudy\}) &= 0.1 \\ m(\{Rainy\}) &= 0.2 \\ m(\{Windy\}) &= 0.2 \\ m(\{Cloudy, Rainy\}) &= 0.5 \end{aligned}$$

The belief function for the event *Cloudy* is given by the probability for this event. The plausibility function on the other hand includes all BPAs that support this belief, though do not stand against it. The belief and plausibility functions are then computed as follows:

$$\begin{aligned} Bel(\{Cloudy, Rainy\}) &= m(\{Cloudy, Rainy\}) = 0.5 \\ Pl(\{Cloudy, Rainy\}) &= 1 - m(\{Windy\}) = 0.8 \end{aligned}$$

If it is assumed that this is not the only source giving evidence for upcoming weather events, the joint mass of belief functions and Dempster's rule of combination can be used to fuse the BPAs of two sources of evidence into one model. In this second source of evidence, there is only the belief for a *Rainy* day given, the rest of the probability mass function is uncertainty U.

$$\begin{aligned} m_2(\{Rainy\}) &= 0.01 \\ m_2(\{Windy\}) &= 0.95 \\ Uncertainty &= 0.04 \end{aligned}$$

The resulting belief mass calculated by Dempster's rule of combination is given by m_{1+2}. As the second source only has *Rainy* and *Windy* as possible weather conditions, all other options from the first source are not considered anymore and resulting BPAs are given as:

$$\begin{aligned} m_{1+2}(\{Rainy\}) &= 0.030 \\ m_{1+2}(\{Windy\}) &= 0.802 \\ m_{1+2}(\{Uncertainty\}) &= 0.169 \end{aligned}$$

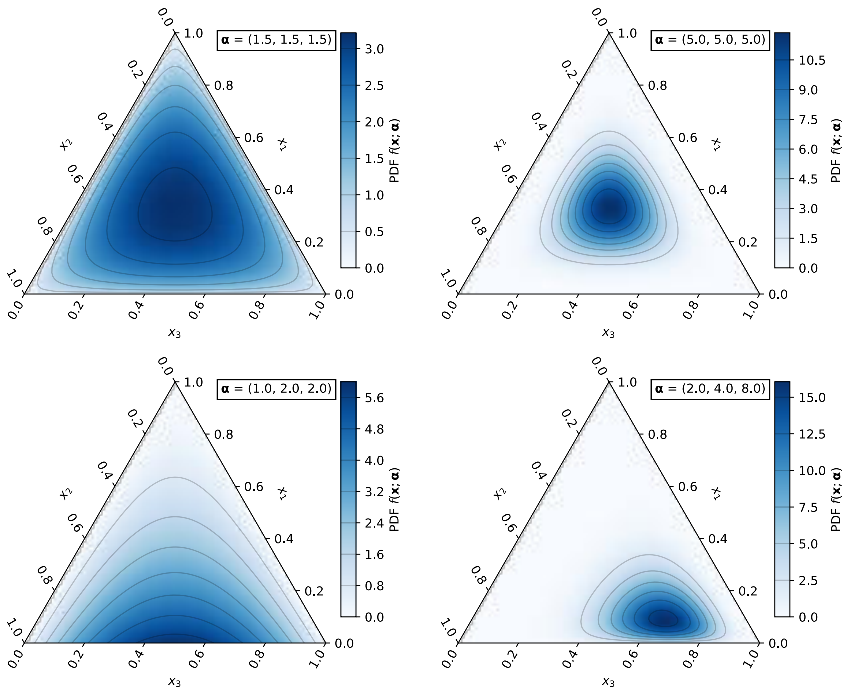


Figure 2.5: Probability Density Function (PDF) of the Dirichlet distribution. Three values of α, which indicate probabilities for different classes, show how the shape of the PDF is altered in respect to these values. [19]

For this calculation, the conflict is calculated by adding up the products of belief masses from $m1$ and $m2$ that have no intersection as they contradict each other. As the two sources have relatively contrary evidences, the conflict is given as $K = 0.763$. In this example, some of the drawbacks of the DST can be seen. Given is a high belief in *windy* weather given by the second source, resulting in a high normalization factor K. With such a high normalization factor, there can occur unintuitive shifts in belief for one option. Another problem might occur if the uncertainty is high, so no evidence can be given to any of the options. A high level of uncertainty can make it difficult to draw clear conclusions or make decisions based on the combined evidence.

In the end, DST provides a framework to handle evidence from different sources, including the ability to deal with the uncertainty of events. There might occur problems in extreme cases where two sources of evidence are giving contrary or highly conflicting information. In such scenarios, Dempster's Rule of Combination can lead to results that are difficult to interpret or use for decision-making. The framework's sensitivity to the initial assignment of belief masses means that small changes in these assignments can significantly affect the combined outcome. As the basis for EDL, DST is a part of a larger toolkit for dealing with uncertain information and should be understood to accomplish for its strengths and weaknesses.

2.4.2 Dirichlet distribution

The Dirichlet distribution is a multivariate generalization of the Beta distribution and is commonly used as prior distributions in Bayesian statistics. It is characterized as steady and multivariate. The distribution is particularly useful in Bayesian models where one needs to estimate multiple probabilities that sum to one, such as the probabilities of different categories in categorical data.

$$f(x) = \begin{cases} \frac{1}{\Gamma(\alpha)\beta^{\alpha}} x^{\alpha-1} e^{-\frac{x}{\beta}}, & \text{if } 0 < x < \infty \\ 0, & \text{otherwise} \end{cases} \tag{2.20}$$

Equation 2.20 shows the definition of a Gamma distribution, a key building block for understanding the Dirichlet distribution since it can be conceptualized as a distribution over a simplex, which represents a vector space where the components sum to one, constructed from multiple independent Gamma-distributed components. The definition for the Gamma function, which is integral in both the Gamma and Dirichlet distributions, is given in equation 2.21.

$$\Gamma(\alpha) = \int_0^{\infty} t^{\alpha-1} e^{-t} dt \tag{2.21}$$

For the Dirichlet distribution, consider a vector $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, ..., \alpha_K)$ of positive parameters, where each α_i corresponds to the *concentration* of the i-th category. The probability density function of the Dirichlet distribution is then defined for a vector $x = (x_1, x_2, ..., x_K)$ in the K-dimensional simplex (i.e., $x_i \geq 0$ for all i, and $\sum_{i=1}^{K} x_i = 1$) as follows:

$$f(\mathbf{x}; \boldsymbol{\alpha}) = \frac{\Gamma(\sum_{i=1}^{K} \alpha_i)}{\prod_{i=1}^{K} \Gamma(\alpha_i)} \prod_{i=1}^{K} x_i^{\alpha_i - 1} \tag{2.22}$$

This distribution is extremely flexible due to the vector of parameters α_i. Different values of α_i can shape the distribution differently, influencing how concentrated the distribution is around certain points of the simplex. When all α_i are equal and greater than 1, the distribution is centered and fairly spread out within the simplex. If any α_i is less than 1, the corresponding x_i tends to be closer to 0, pushing the distribution towards the edges of the simplex. This flexibility makes the Dirichlet distribution a widely used tool for expressing prior beliefs about the proportions in multiclass problems, including EDL

Figure 2.5 shows how the distribution behaves under different settings of concentration parameters α. Each plot represents a probability simplex, which can be seen as a triangle in 2D space, for three categories. The axes x_1, x_2, and x_3 satisfy the condition $x_1 + x_2 + x_3 = 1$. The first plot with parameters $(1.5, 1.5, 1.5)$ shows a relatively uniform distribution across the simplex with a slight density increase towards the center, illustrating the symmetry and equal Likelihood of all categories. In the second plot, where each parameter is increased to 5.0, the PDF is sharply peaked towards the middle, indicating a reduction in variance and a concentrated distribution around the mean due to higher symmetric α values. The third plot with asymmetric parameters $(1.0, 2.0, 2.0)$

shows a distribution more spread towards the higher values, biasing towards the second and third categories over the first, reflecting the influence of unequal α values. The fourth plot $(2.0, 4.0, 8.0)$ further demonstrates this effect by heavily skewing the probability mass towards the third category, which has the highest parameter, indicating a strong preference or expected frequency for this category.

These visualizations show how different parameter configurations affect category frequencies when dealing with categorical data, which in the end is learned by the model if applied to EDL.

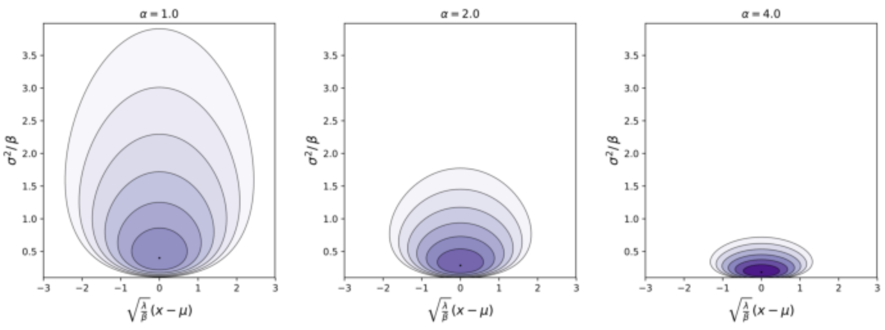


Figure 2.6: Probability Density Function of the NiG distribution. [20]

2.4.3 Normal-inverse-Gamma distribution

Another distribution used in context of EDL is the NiG distribution which can be used as conjugate prior to the Normal or Gaussian distribution. As its function as prior describes, it can be used if mean and variance are unknown and update the beliefs about these parameters after new data is observed. There are four parameters that describe this distribution:

- α **(Shape Parameter)**: This parameter influences the tail-heaviness and thereby determinating the uncertainty about the variance. It shapes how the variance is distributed, with higher values indicating less uncertainty.
- β **(Scale Parameter)**: This parameter determines the rate of tail decay and directly influences the expected variance of the normal distribution, thereby affecting the dispersion of variance values.
- γ **(Prior Mean)**: Represents the prior expected value of the normal distribution's mean, also described as μ_0. It serves as the baseline for the mean, indicating where the mean values are centered before any data observation.
- ν **(Precision of the Prior Mean)**: Affects the precision (inverse of variance) of the mean's prior distribution, declared as λ in figure 2.7. Higher values indicate a stronger and more precise belief in the prior mean, leading to less spread in the distribution of γ.

Figure 2.7 visualizes the PDFs of the NiG distribution, illustrating how the distribution changes with different values of the shape parameter α.

The contour plots in the left, middle, and right panels represent the distribution of the variance σ^2 relative to the beta parameter β on the y-axis and the standardized deviation $\sqrt{\nu/\beta}(x-\gamma)$ on the x-axis, for α values of 1.0, 2.0, and 4.0, respectively. In the left panel ($\alpha = 1.0$), the distribution indicates moderate uncertainty about variance with less spread about the mean γ, centered at 0 on the x-axis. As α increases to 2.0 in the middle panel, the spread of the variance decreases, showing narrower bands along the y-axis and a tighter concentration around the mean, reflecting increased confidence in both variance estimation and the precision around the mean γ. In the right panel, the distribution becomes more concentrated with significantly less variability in variance, visible by reduction in y-axis spread, and an even tighter clustering around the mean, suggesting a high level of certainty about both the variance and the mean.

The NiG distribution combines a Normal distribution for the mean γ given the variance σ^2, and an inverse gamma distribution for σ^2. Parameters γ and ν in the plots represent the mean and the precision of the mean under the normal distribution part of the NiG. As α increases, it affects both the shape of the inverse gamma by making it more peaked and less spread and indirectly influences the normal distribution's spread around γ, leading to less variability on the x-axis.

The vertical spread (σ^2/β) indicates the level of uncertainty in the variance, where, with increasing α, smaller ranges suggest more confidence in variance estimations. The horizontal spread ($\sqrt{\nu/\beta}(x-\gamma)$) shows how the data is spread around the mean γ, becoming narrower as α increases, indicating more confidence in the mean's position.

This demonstrates the NIG's utility in Bayesian inference, particularly in modeling uncertainty explicitly, such as in regression problems or robust predictive modeling.

2.4.4 The concept

The practical implementation of the EDL concept uses the Dempster-Shafer Theory combined with the Dirichlet or NiG Distribution to effectively quantify the uncertainty of a model's output. These functions used as a conjugate prior within the DST framework allow for a comprehensive probabilistic representation. Figure 2.7 shows the influence of the NiG distribution on the lower order Likelihood.

The probability density function, defined in formula 2.23, uses a concentration vector $\alpha = [\alpha_1, \alpha_2, \ldots, \alpha_K]$ for K hypotheses, representing prior observation counts for each category, effectively quantifying the strength of the evidence for each outcome. The vector $p = [p_1, \ldots, p_K]$ denotes the probabilities of each category, maintaining the constraint that $\sum_{i=1}^{K} p_i = 1$.

$$f(p;\alpha) = \frac{1}{B(\alpha)} \prod_{i=1}^{K} p_i^{\alpha_i - 1} \tag{2.23}$$

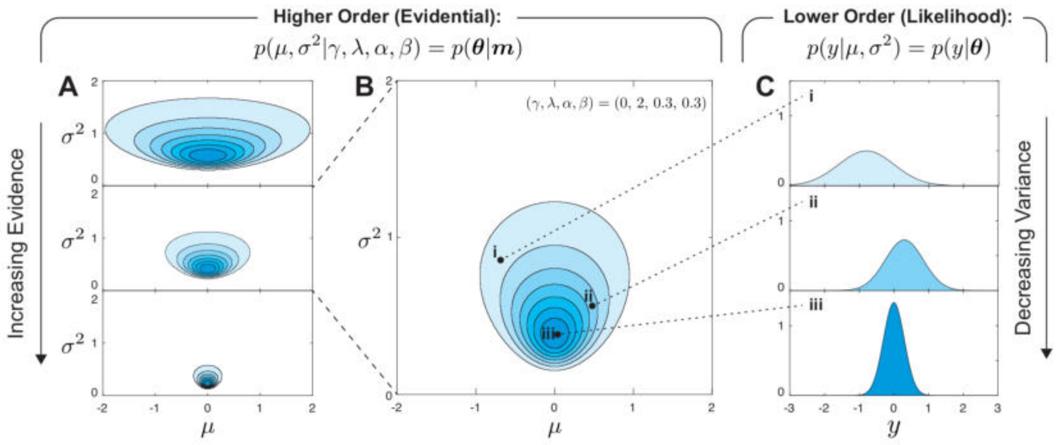


Figure 2.7: Different shapes of the evidential prior, based on parameters α, β, γ and λ, are presented on the left. For one set of values, the resulting lower order Likelihood with decreasing variance when sampled more centric is shown. The simplex of the prior directly influences the shape of the lower order Likelihood, getting a variation of the Normal distribution out of the NiG distribution. [21]

By manipulating the concentration parameters α, the model adapts to different levels of uncertainty and evidence within the observed data. Higher values of α correspond to strong evidence and lower uncertainty, while lower values indicate sparse data and higher uncertainty. This dynamic adaptation enables EDL to handle various degrees of ambiguity in the data inputs.

Mathematical representation

Following the Bayesian inference, *the likelihood of an observation y_i given ω follows a t-distribution with 2_{α_i} degrees of freedom* [22], with $\omega = (\gamma, \nu, \alpha, \beta)$ denoting a set of the four parameters describing the distribution. The Student's t-distribution is commonly used in statistics for datasets with small sample sizes or unknown variances, and is more robust to outliers than the Normal distribution. Equation 2.24 states the Likelihood function L_i^{NIG} of this distribution.

$$L_i^{\text{NIG}} = \text{St}_{2\alpha_i}\left(y_i \mid \gamma_i, \frac{\beta_i(1+\nu_i)}{\nu_i\alpha_i}\right) \tag{2.24}$$

The observed parameter is denoted y_i as an observation under the model defined. γ_i describes the location parameter, usually the mean or median. The two parameters α_i and β_i influence the scale and shape of the distribution, while ν_i adjusts the weighting of them.

In real implementation using the NiG distribution, the constraint for positivity of the three parameters ν, α, β is enforced by a SoftPlus function while additionally ensuring $\alpha > 1$.

Loss function

During training, the loss function in EDL focuses on enhancing the accuracy of uncertainty estimations as well as the predictions. The loss function $\mathcal{L}$ is formulated as the negative logarithm of model evidence, with the goal of balancing precise predictions and reliable uncertainty assessments. The following equations detail the components of the loss function where ω represents the set of the estimated distribution parameters.

$$\Omega = 2\beta(1+\nu) \tag{2.25}$$

The term Ω is defined to regularize the evidence, which influences the model's confidence in its predictions. Here, β and ν are the parameters that adjust the regularization strength, ensuring that the evidence is appropriately scaled.

$$\mathrm{L}_i^{\mathrm{NLL}}(w) = \tfrac{1}{2}\log\left(\tfrac{\pi}{\nu}\right) - \alpha\log(\Omega) + \left(\alpha + \tfrac{1}{2}\right)\log\left((y_i - \gamma)^2\nu + \Omega\right) + \log\left(\frac{\Gamma(\alpha)}{\Gamma\left(\alpha+\frac{1}{2}\right)}\right) \tag{2.26}$$

The Negative Log-Likelihood (NLL) component, $\mathcal{L}_i^{\mathrm{NLL}}(w)$, measures the discrepancy between the predicted distribution and the actual outcome. This term incorporates the model's uncertainty through parameters α and ν, where α represents the shape parameter and ν the scale parameter of the predictive distribution. The Gamma function Γ is used to normalize the probability distribution. This component ensures that the model fits the data well while adjusting its confidence based on the evidence.

$$\mathcal{L}_i^{\mathrm{R}}(w) = |y_i - \mathbb{E}[\gamma_i]| \cdot \Phi = |y_i - \gamma| \cdot (2\nu + \alpha) \tag{2.27}$$

The regularization term $\mathcal{L}_i^{\mathrm{R}}(w)$ penalizes large deviations between the predicted mean $\mathbb{E}[\gamma_i]$ and the actual outcome y_i. The regularization factor Φ is a function of the model's estimated variance ν and the parameter α. This term helps to prevent overfitting by discouraging predictions that stray too far from the observed values.

$$\mathcal{L}_i(w) = \mathcal{L}_i^{\mathrm{NLL}}(w) + \nu\mathcal{L}_i^{\mathrm{R}}(w) \tag{2.28}$$

The total loss, $\mathcal{L}_i(w)$, is a weighted sum of the NLL and the regularization term, controlled by the hyperparameter ν. This combined loss function explicitly penalizes deviations from the expected outcomes and encourages precise uncertainty assessments alongside accurate predictions. By optimizing this loss function, the model is trained to be both reliable and confident in its outputs, providing a balanced approach to EDL.

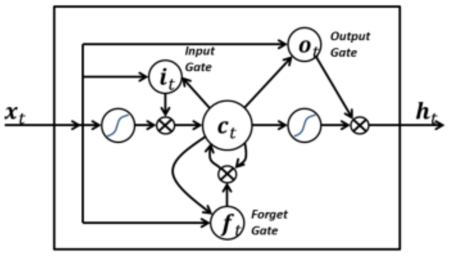


Figure 2.8: Schematic representation of the functionality of a LSTM Cell with input gate i_t, output gate o_t, forget gate f_t and current cell state c_t. x_t denotes the input and h_t the output.[23]

2.5 Network layouts and functions

The proposed network uses a Long Short-Term Memory (LSTM) Cell setup and some specific activation and loss functions, which are explained here in advance.

LSTM Cell

The LSTM Cell often is used in networks where spatiotemporal data is being processed, as it keeps track of information by providing - as the name suggest - some kind of memory. Although not using spatiotemporal data, this kind of neural network architecture will be used in the realm of this work. Figure 2.8 provides an overview of an LSTM Cell's general layout and functionality. The equation for one of the gates inside the LSTM network is given by:

$$f_t = \sigma_g(W * x_t + U * h_{t-1} + V \circ c_{t-1} + b) \tag{2.29}$$

Where:

- f_t is the output of the gate at time t.
- σ_g is a sigmoid function that controls the gate activation.
- W, U, and V are the weight matrices applied to the input, previous hidden state, and previous cell state, respectively.
- x_t is the input at time t.
- h_{t-1} is the hidden state from the previous time step.
- c_{t-1} is the cell state from the previous time step.
- b is the bias term associated with the gate.
- The operator $*$ represents convolution, used to combine the input with the weights.

- The operator $\circ$ represents element-wise multiplication, used to modulate the influence of the previous cell state by the current gate.

As described by this equation, each cell uses three learned matrices W, U and V and a fixed bias of b.

The input x_t is fed into all three gates as it generally is the foundation for further operation inside the cell.

The **Input Gate** combines the current input and the previous hidden state to create a candidate memory which represents a possible update to the current cell state.

The **Forget Gate** plays a role for the decision, which information from the previous cell state c_{t-1} should be kept and what would be discarded. Generally it takes the new input x_t and the previous cell state c_{t-1} as input. After processing with convolutions, the sigmoid function creates a mask ranging from 0 to 1 which is than multiplied with c_{t-1}. This determines how much influence different parts of the previous cell state c_{t-1} have for the calculation of the next state.

The **Output Gate** determines which parts of the current cell state c_t should be used to generate the current hidden state H_t. Like the forget gate, it uses x_t and c_t as inputs but the result of the convolution inside the cell is directly concatenated with the state c_t processed beforehand by an activation function.

Functions

The Softmax Activation (equation 2.30) is often used to covert the output of a neural network's layer into a probability distribution due to the characteristics of its output. The values are normalized to sum up to one, which checks a preliminary requirement for a probability distribution. In addition, it *softens* the distribution over probabilities by increasing the resulting values for probabilities that are high compared to others.

$$f(s)_i = \frac{e^{s_i}}{\sum_j^C e^{s_j}} \tag{2.30}$$

The Cross Entropy Loss (equation 2.31) is also widely used in the field of Machine Learning (ML). Usually a one hot encoding vector t_i is used where the correct class is assigned a value of 1 and all other elements are 0.

$$CE = -\sum_i^C t_i \log\left(f(s)_i\right) \tag{2.31}$$

Softmax Loss now is actually just a Softmax Activation plus a Cross Entropy Loss. In the end, the goal is to assign a high probability to the right class while reducing the probability for all other classes. This is ensured by the Softmax Activation in combination with Cross Entropy Loss by minimizing the negative sum.

The **Mish Activation Function** is defined by the equation:

$$\mathrm{Mish}(x) = x \cdot \tanh(\log(1 + e^{x})) \tag{2.32}$$

Mish is a smooth, non-monotonic function that combines the properties of Rectified Linear Unit (ReLU) and softplus. It enhances model learning capabilities by maintaining a smooth gradient flow over a wider range of values, helping prevent the loss of information during backpropagation, which is a common issue with ReLU when inputs are negative.

3 Related Work

The task of reconstructing a 3D scene from images has been addressed through various approaches and techniques, with deep learning methods significantly improving performance in recent years [24]. This chapter provides a comprehensive review of the related work, focusing on methods that incorporate uncertainty in depth reconstruction.

In section 3.1, different approaches for the task of MVS are presented, emphasizing methods that consider uncertainty. This section covers both, traditional techniques and modern deep learning based methods, and explores where uncertainty is already considered in MVS pipelines. An overview can be found in appendix 8.15. The subsequent section 3.2 delves into methods incorporating the concept ofEDL from various fields, outlining the foundation for the aimed for integration into MVS in this work. Datasets that can be found in context of MVS, some of which are used in the conducted experiments in chapter 6, are presented in section 3.3 with an overview presented in appendix 8.16. Finally, section 3.4 concludes with a discussion on notable concepts and identifies gaps in the current research. This last section aims to provide insights into potential areas to integrate uncertainty in 3D reconstruction methods.

3.1 Multi-View Stereo

The term Multi-View Stereo (MVS) describes techniques utilized to construct a detailed 3D model from multiple images captured from various viewpoints of a scene. Typically, MVS techniques require known camera parameters; this includes both intrinsic and extrinsic parameters, as well as the relative orientation of each camera. How to gather these prerequisites is outlined in chapter 2. Unlike single-view reconstruction techniques, MVS does not rely on a prior shape model but instead uses multiple images to extract depth information through the comparison of multiple perspectives. This approach allows MVS to effectively reconstruct complex surfaces and intricate details that might be lost in other methods.

MVS algorithms function by pinpointing correspondences among multiple images to ascertain the three-dimensional configuration of the scene. They exploit the overlap in images, utilizing distinctive features such as edges, textures, or colors to align points across different views. Once these correspondences are confirmed, methods like triangulation are employed to determine the depth and position of points within space, ultimately producing a comprehensive three-dimensional model.

The advantage of MVS over other stereo methods lies in its robustness and ability to reconstruct scenes with high accuracy. Because it integrates information from multiple views, it can resolve ambiguities and occlusions more effectively than two-view stereo systems. There are some related areas of research that should be distinguished from MVS, as they rely on a different set of input parameters. Methods within the MVS domain are typically applicable in scenarios where cameras are configured in a fixed, rather experimental setup. Examples include multiple cameras mounted on a vehicle or cameras attached to a handheld or camera arm. These setups are similar to those used in the datasets discussed in section 3.3.

Related areas of research

If the camera parameters are not known in advance, related methods to get a 3D structure of a scene from images are referred to as Structure from Motion (SfM). It can be used as a first step in a 3D reconstruction pipeline to get the required parameters and perform MVS. In SfM, typically distinct feature points are searched in each image, eg. by using algorithms like Scale-Invariant Feature Transform (SIFT). This method tries to identify image regions that are invariant to scaling and rotation, which can than be matched in different images. While SIFT can identify regions by including the gradients of neighbouring pixels, matching can be performed down to pixel level accuracy. When SIFT features are matched, epipolar geometry can be applied to better find corresponding features and filter out mismatched features. If the needed camera parameters are not given in advance, Zhang's method [6] can be used for calibration.

For the related task of monocular depth estimation there are some approaches that consider the uncertainty of their results [25, 26]. In particular, Poggi et al. [27] estimate the uncertainty in their model for the self-supervised case. They find that the consideration of uncertainty improves the accuracy of the depth estimation and can be used for the proposed learning techniques. Hornauer and Belagiannis [28] use an auxiliary loss function to estimate the uncertainty while using an already trained neural network. Wang et al. [29] dive deep into the consideration of aleatoric uncertainty on a pixel-wise level.

In the case of stereo images the work by Mehltretter [30] stands out as it combines aleatoric and epistemic uncertainty and introduces both in a neural network with the ability to be trained end-to-end. Wang et al. [31] introduce EDL to depth estimation from stereo images. They use two loss functions to improve the results of uncertainty estimation and to ensure smoothness of the uncertainty estimates in regions where the disparity changes smoothly.

Conventional methods

Early MVS methods include the usage of volumetric methods [32] where a 3D grid of voxels is used to represent the hull of the object. If a voxel contains surface pixels from multiple viewpoints, it is considered as part of the objects surface. Another method is space carving [33] where an initial hypothesis of an objects location in 3D space is iterated until its appearance is consistent with all views. Parts of the object that are not visible from the provided cameras perspectives are discarded. As the algorithm starts with an initial large, unshaped volume and defines it step by step, the process was referred to as *carving*. Another technique beside the nowadays established usage of deep learning are patch-based methods [34]. These algorithms start with feature extraction from the different views and matches them over these views. These matched features form the corners of small patches that grow in an iterative way while checking for photo-consistency in each step. Newer and more sophisticated methods introduce various concepts of deep learning to the field of MVS to extract and match features of different viewpoints.

Before the broad utilization of neural networks, COLMAP [35] made a huge contribution to the field of CV by delivering a robust algorithm for various tasks like SfM and MVS. It proved its scalability through extensive testing on a large scale dataset of Rome and its key contributions are still used in recent works [36]. The DeMoN network [37] introduced deep learning techniques, in particular an encoder-decoder structure, to the task of SfM. The authors include the training by artificially generated variations of the original input data to generate more training examples and make the network more robust.

Deep learning methods

As a remarkable starting point for the usage of neural networks, MVSNet [38] was introduced in 2018 and has since been used in various following works [39, 40] with different variations to the original method. They extract features from the images, use Differentiable Homography and optimize the selection of the right depth hypothesis with the Softmax function. The work by Li et al. [41] from 2020 combines the concept of PatchMatch [42] and deep learning. It is especially suitable for high resolution images and and has another focus on the prediction of the results confidence. Similarly, the Deepc-MVS network by Kuhn et al. [43] considers uncertainty as it focuses on the prediction of the network's confidence. The refinement of 3D reconstruction results by consideration of uncertainty measures boosts the networks performance and shows how uncertainty prediction can also be a viable solution to increase a network's confidence and prediction results. Cheng et al. [44] make uncertainty consideration a crucial part of their network architecture as well as they use multiple stages to refine their depth prediction based on the uncertainty found in the previous stage. In their network setup, first a feature extractor with multiple stages is used to get feature representations of the images in varying resolutions. These features are warped into one reference view to form a Cost Volume which is then processed by a 3D-CNN to form a Probability Volume. Then they consider the uncertainty included in the Probability Volume to form a Cost Volume with different depth, called *Adaptive Thin Volumes*, in the concurrent stages.

In 2021, two interesting works have been presented. The paper *Digging into Uncertainty in Self-supervised Multi-view Stereo* by Xu et al. [45] introduces the consideration of epistemic uncertainty by using the technique of MCD. They use a self supervised learning approach and exclude pixels with high uncertainty from the contribution to learn. The work by Wei et al. [46] on the other hand does not consider uncertainty but uses an Attention mechanism to make context-aware feature processing possible. In 2021 it was one of the best performing approaches for the most recognized benchmarks. As Transformers have been introduced to many areas of research [47] including nowadays commonly known Large Language Models (LLMs) like ChatGPT, they are as well used in context of MVS [48, 49]. The work by Su et al. [50] named *Uncertainty Guided Multi-View Stereo Network for Depth Estimation* combines the training of a uncertainty map and the depth map. They use a Cost Volume and *adaptively determine the depth search range for finer stages*. Two rather memory efficient methods are introduced by Wan et al. [51] and Mi et al. [52]. The former declares the search process for the right depth hypothesis as a binary search problem. Vis-MVSNet [53] especially considers the visibility of pixels and incorporates a view selection mechanism where information from the most informative view is taken. The recent work by Song et al. [54] uses a Bayesian Probability Volume to consider the aleatoric uncertainty. GeoMVSNet [55] can be considered as one of the top-performing SotA networks with special attention to capture the geometric structure given in a scene. Chen et al. [56] do consider epistemic uncertainty in their network and propagate the uncertainty information through the network for different depth hypothesis. Liao and Waslander [57] propagate the uncertainty information through the whole 3D reconstruction pipeline until they can enrich the final object's mesh with uncertainty information.

3.2 Evidential Deep Learning

The work of Sensoy et al. [2], published in 2018, layed foundation for further development of EDL, introducing this new technique to quantify uncertainty in neural networks. Based on the theoretical concept developed by Dempster [18] and Shafer [17], they used the Dirichlet distribution, introduced in section 2.4.2, as prior. A survey from 2021 [58] outlines the following development. In a follow up work, Sensoy et al. [59] trained a similar model in 2020 *using a contrastive loss with artificial OOD samples from a Variational Autoencoder*. While these posterior networks are able to model epistemic and aleatoric uncertainty by incorporating observed data, also prior networks have been introduced [60, 61, 62]. These networks only incorporate epistemic uncertainty estimation in advance for an easier to train procedure.

Regression based methods like the work presented by Amini et al. [21] in 2020 use a different set of evidential priors. In this case, they use a NiG distribution as prior over a Gaussian likelihood function. The model was trained to predict molecular properties and showed reliable and robust results. Other regression based methods also incorporate other prior distributions [63] and can depict other distributions than Gaussian [64].

Applications

The concept of EDL is used in graph neural networks [61] and for Open Set Recognition (OSR) [65] where samples belonging to unkown classes are identified. The ability of EDL to model these unknown classes can be marked as one of the main advantages over traditional methods of deep learning. Pandey and Yu [66] use EDL in context of meta learning where a model should adapt to a minimal amount of new training data rapidly. Another way to adapt the network's parameters is active learning where some instance, for example a user, is asked to provide labels for data and thus provide the information for further training to the network. In context of EDL active learning was used by Hemmer et al. [67]. As a practical implementation used in real world scenarios, Soleimany et al. [68] use EDL to predict a molecule's property.

Lou et al. [69] use EDL for the task of Deep Stereo Matching (DSM). They find that a *costvolume-based model is not robust in regions with large illumination changes while transformer-based model does not make full use of complex local textures*. To overcome these issues, their model architecture uses a Cost Volume as well as a transformer based approach, each using the principle of EDL to gather the Dirichlet distribution's variables and fuse them in an *Inter evidential fusion* model. In the work by Wang et al. [31], EDL is used for DSM in a simpler way.

Aguilar et al. [70] use EDL in a continual learning framework called *CEDL* which integrates OOD detection. The same use case has been covered by Cao and Zhang [71]. Their approach improves OOD detection by combining eight normalization and a novel network architecture, achieving perfect results on difficult datasets.

Variations

Based on the somewhat dated work by Amini et al. [21], recent contributions have built upon and improved the original research, advancing scientific knowledge.

Meinert et al. [22] question why the concept behind EDL seems to perform alright in practice while they find some theoretical flaws in the mathematical background of the concept. They question the ability of the presented Loss function to find a global minimum due to the dependency on a specific parameter. In addition, they suggest new formulas to describe the aleatoric and epistemic uncertainty out of the NiG's parameters. Their new interpretation of these types of uncertainty influenced this work, as their set of formulae is used as Default implementation method in this work. Their way to mathematically describe aleatoric and epistemic uncertainty will be compared against the original paper's implementation in chapter 6.

In 2021 the paper *Multivariate Deep Evidential Regression* has been published by Meinert and Lavin [72]. It enhances the framework by adapting to multivariate regression tasks, like found in traffic flow analysis. They use a loss function specifically adapted to this task.

Nagahama [73] introduces a concept called *modified-EDL*. It extends the classical approach to perform EDL with the ability to better deal with new, unknown classes. It adds an additional class to the set of possible outcomes which represents unknown classes and performs better in scenarios where unknown classes do occur.

Deng et al. [74] argue that traditional mean-square error-based learning approaches impede the acquisition of evidence, particularly in instances of samples characterized by high data uncertainty. To address this limitation, they implement a Fisher Information-based Evidential Deep Learning (FIEDL) framework, which quantifies the informational content inherent in individual data samples. This framework utilizes Fisher Information to assess the informativeness of data and accordingly adjusts the weighting of loss terms. This method is notably effective in modulating the learning process for samples that exhibit significant aleatoric uncertainty if they are represented by one-hot vectors.

3.3 Datasets

Over the years, several publicly accessible datasets in the domain of MVS have been released, as detailed in Appendix 8.16. Notably, the Middlebury MVS dataset [75] stands out as the initial dataset widely acknowledged and utilized in this field. Despite its limitation of featuring only two objects, the diversity of viewing angles available within this dataset is commendable.

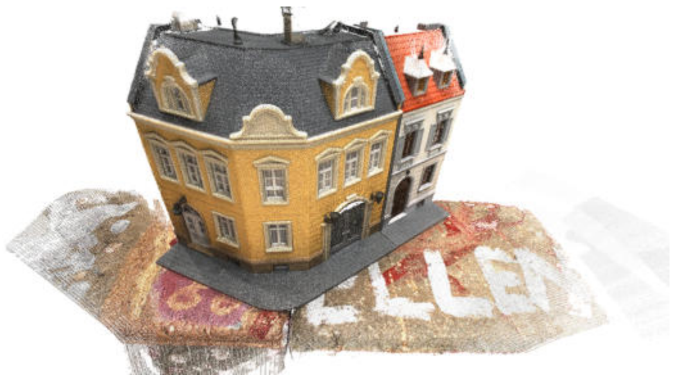


Figure 3.1: A representative scene from the DTU dataset, widely used among researchers.

Ten years after its release, in 2016, the quasi standard benchmark to this date and foundation for the training procedures of many machine learning approaches, the DTU dataset [76], has been released. In comparison, it does not sample synthetic images from a retrieved 3D model but is using a high precision camera arm to gain photos of small objects from exact positions. In addition to variations in the camera arms position, also the lighting intensity and direction is altered, so algorithms can be optimized to deal with the various challenges introduced by shadows.

Both datasets could be categorized as laboratory setups capturing small objects, while the three datasets released in 2017, namely ETH3D [77], ScanNet [78] and Tanks and Temples [79], focus on indoor and outdoor scenes. Remarkably, the ETH3D dataset has semantic annotations for objects that can be found in the scene, so in theory one could use its annotations to reconstruct individual objects, too. This dataset also uses relatively cheap sensors to retrieve its 3D information and has a large amount of images included in the dataset. Besides the task of MVS it is often used for performance analysis of methods in the field of semantic segmentation. In addition to high resolution images and precisely matched LiDAR scans, the ETH3D dataset provides video sequences of the scenery, including various fields of view. This can especially be advantageous in some

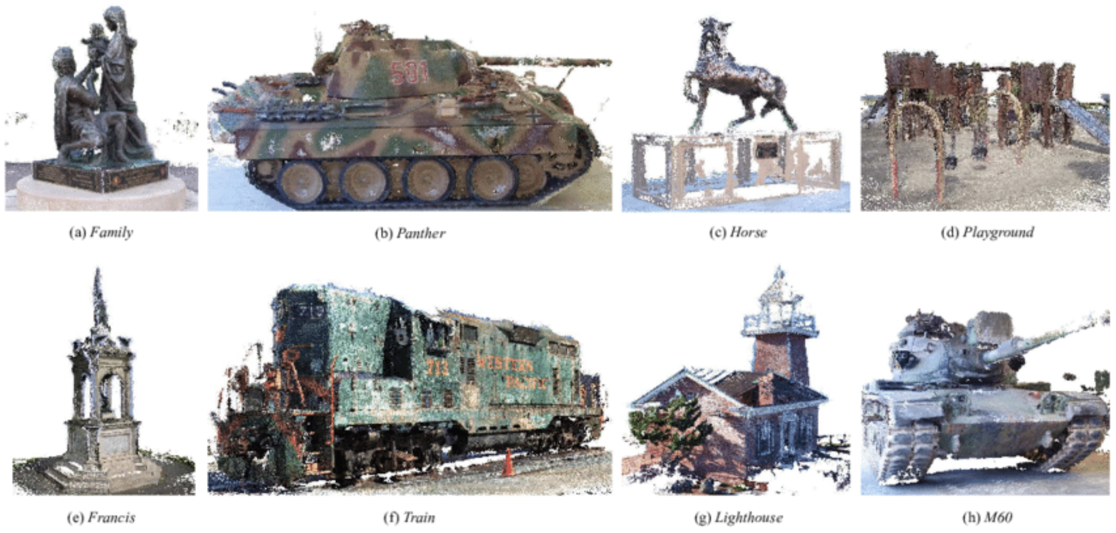


Figure 3.2: Reconstruction resuluts of various scenes included in the Tanks and Temples (TnT) dataset. [38]

approaches for SfM which require video-like image sequences with small inter-frame variations.

The Tanks and Temples (TnT) dataset has been established itself as a remarkable benchmark and widely used option for inter-benchmark performance testing in the field of MVS. Its data is made of high precision LiDAR scans enriched with RGB data from high resolution video cameras. Other than the name suggests, it not only contains tanks and old statue, but also scenes like the *Ballroom* with huge dimensions. As a key feature, the goal was to capture scenarios of the real world and they - in comparison to the DTU benchmark - went from a laboratory setting to capturing their data in the real world.

The Blended MVS dataset [80] was introduced in 2020 alongside an expansion, known as Blended MVG dataset, which was released within the same year. This dataset employs a specialized 3D reconstruction pipeline to extract three-dimensional information from images across diverse scenes. It encompasses a broad spectrum of scenes including drone-captured images of buildings and detailed captures of smaller objects, offering a substantial variety and volume of data.

In 2022, the UrbanScenes3D [81] and the GigaMVS [82] datasets have been released. Both focus on large sceneries, in case of UrbanScenes3D complemented with synthetic CAD models of such scenes. The GigaMVS dataset is truly remarkable for its scale and use of high resolution photos. The creators of this dataset especially want to challenge MVS methods to deal with these enormous amounts of data with sceneries taking up to 32.000 m^2 and billions of points in their reference point cloud. Its scenes contain parts of urban areas, buildings and archaeological sites.

As a relatively new dataset, Skoltech3D [83] from 2023 can be mentioned in the context of MVS datasets. It has been captured in a laboratory environment with multiple scanners and camera mounted to a rig, which makes it interesting for comparing the results of various capturing methods and resolutions. Like the DTU dataset it includes various lighting conditions and one hundred viewing directions for each object scanned.

3.4 Discussion

While CV in general and 3D reconstruction from Stereo as well as Multi-view Stereo in particular present a field of research with remarkable interest, the estimation and integration of methods to determine uncertainty has not been tackled in many approaches. Especially for deployment in risk averse areas, knowledge about the uncertainty of a prediction or the recognition of a model's inability to produce a trustworthy prediction could be crucial to even consider real-world application of such systems. The need for these systems might be missing to this point as roll out of real-world applications for safety-critical MVS applications might be held back due to other limitations. But knowing about cases or areas where a deep neural network's predictions are not trustworthy might be essential one day.

Works like the one presented by Su et al. [50] do incorporate estimated uncertainty predictions to refine the estimated depth with success for the stereo case. As this seems promising for the case of MVS as well, there can various methods be found that incorporate aleatoric uncertainty, as the found in appendix 8.15 suggests. But most of those methods do lack incorporation of epistemic uncertainty, which seems to be a major problem to reliably detect OOD data. Methods that do incorporate epistemic uncertainty [45, 56] use techniques that lead to a massive computational overhead. This is a major drawback and can make those setups completely unusable for application beside an experimental laboratory environment. If modern networks for 3D reconstruction from MVS scratch the abilities of cutting edge hardware, training of multiple networks or inference on dozens of members of one network often is not possible, especially not in real-world applications needing near real time execution.

Due to limitations in existing methodologies, the integration of EDL into the domain of MVS has emerged as a promising research area, attracting considerable attention for the reasons detailed previously. Although Wang et al. [31] have demonstrated the effectiveness of EDL in Stereo Vision, its application in MVS remains unexplored. A notable advantage of EDL, as discussed in Chapter 2, is its efficiency in terms of memory usage and computational demand, while adeptly handling both aleatoric and epistemic uncertainties. Consequently, EDL represents a compelling method for quantifying uncertainty in 3D reconstruction tasks within MVS, devoid of many limitations characteristic of alternative approaches.

4 EMVSNet

The methodologies developed in this work are designed to model uncertainty along with the reconstruction of scenes in 3D from Multi-view Stereo. Due to the factors mentioned earlier in this work, the usage of EDL seemed to be a reasonable choice, only relying on methods introduced by deep learning and marking an approach never before introduced to the field of MVS.

The structure of the Neural Network developed in this thesis called EMVSNet relies as foundation on the work of Wei et al. [46] presented on International Conference on Computer Vision (ICCV) 2021. Their Adaptive Aggregation Recurrent Multi-view Stereo Network (AA-RMVSNet) can be used as a solid previous work proven to effectively overcome the challenges introduced by the task of 3D reconstruction from MVS. The approach has a modular structure which allows for variations in the amount of viewing directions used while maintaining a comprehensive performance to other methods introduced to the field of MVS.

As secondary part of the proposed EMVSNet, a work by Lou et al. [69] delivers the fundamental concept for the part of the network which estimates the NiG distribution's parameters. Originally, this concept uses a 3D Cost Aggregation as well as Stereo Transformer to extract the evidential prediction parameters at different stages of the network and fuse them for a combined prediction. Various concepts used by these preliminary works are reused in this work and will be explained in the following in more detail.

Final goal of the proposed method is the determination of estimated depth and aleatoric as well as epistemic uncertainty. This is achieved using a set of input RGB images of dimensions $w \times h$, with one reference view and associated source views of the same dimensions. The reference view is chosen so it represents the central frame, capturing movement in all directions within a continuous set of images. This set of images is also characterized as set of stereo images, so overlapping image areas must exist.

The depth and uncertainty predictions are performed on a per pixel level, associating each pixel of the reference view a predicted depth based on a predefined depth range r_d. Aleatoric and epistemic uncertainty are defined on a per pixel level as well, calculated through the set of parameters $p = (\alpha, \beta, \gamma, \nu)$ for each pixel.

With the chosen NiG distribution as a prior, the distribution of the input data is assumed to be Gaussian. This assumption encompasses image noise, disparity uncertainties within views, and uncertainties about the model's parameters. Furthermore, it is assumed that the individual observations are independent of each other and that the uncertainties can be adequately described by the NiG distribution.

Section 4.1 presents the specific network structure and explains important design choices. In section 4.2, the chosen mathematical representation for uncertainty and Loss are presented, while section 4.3 deals with possible adaptions of the presented method, including outlook for follow-up work.

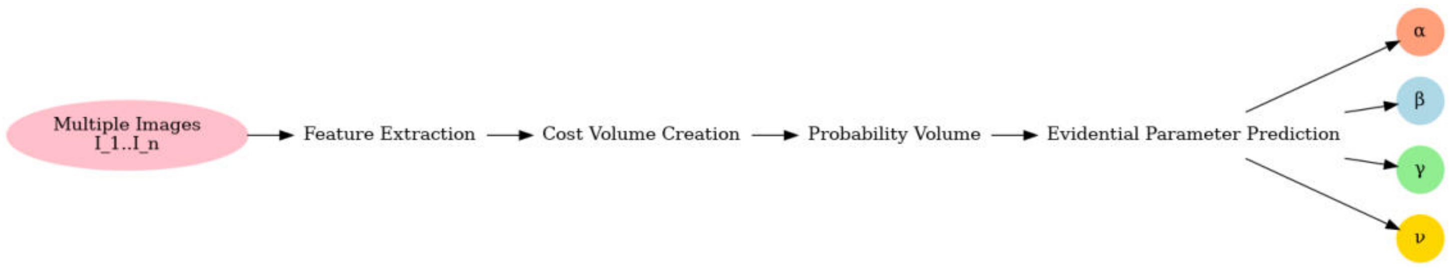


Figure 4.1: Overview over the proposed EMVSNet network structure. For multiple images as input, parameters of the NiG distribution are predicted per pixel, which are then used to calculate aleatoric and epistemic uncertainty.

4.1 Network structure

The structure of the proposed network is split into following parts: The feature extraction including the Intra-view Adaptive Aggregation Module, followed by homography and a preliminary Cost Volume construction which is presented in section 4.1.1. Regularization and refinement of this network structure leads to a Probability Volume which is as preliminary step processed by an LSTM network (section 4.1.2). Based on the Probability Volume available at this point, the prediction of parameters used for EDL follows, which is explained in detail in section 4.1.3.

4.1.1 Feature extraction and Cost Volume creation

Input for the given network is a set of images $I_{1..k}$ for k source views from different directions. For the homographic warping executed in following steps, the extrinsic camera parameters between source and reference view must be known as well.

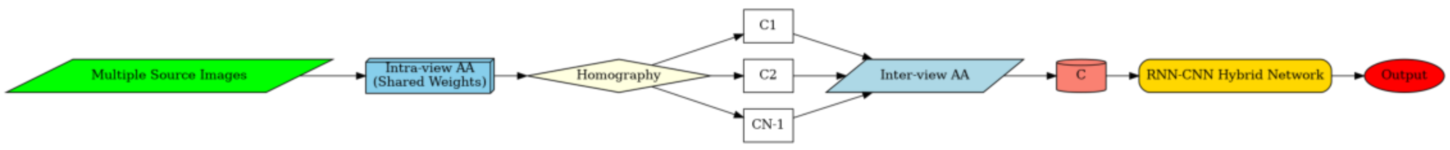


Figure 4.2: Feature extraction pipeline of EMVSNet. Per image, one Cost Volume is created and warped into the reference view. Those Cost Volumes are combined and refined as Probability Volume, marking the input for the EDL part of the network.

Overview first part

Multiple images from different viewpoints, represented as $W \times H \times 3$ matrices containing RGB information, in combination with their extrinsic rotation matrices, mark the input for the presented network structure. First step of the pipeline is the extraction of relevant features from each input image, including processing inside an Intra-view Adaptive Aggregation Module. The feature maps, enriched with contextual information about the image, are afterwards transformed to match the viewing direction of the reference view using homography. As a result, warped Cost Volumes $C^w_{1..k}$ for the difference between reference view and source views can be calculated. These mark the input for an Intra-view Adaptive Aggregation Module which weights the information and matching cost of the different viewpoints, combining the information into one Cost Volume C^c. This Cost Volume is further refined by a Recurrent Neural Network (RNN)-Convolutional Neural Network (CNN) Hybrid Network which incorporates LSTM network layout. As result a refined Cost Volume C^r marks the output of the first part of the network and is used for further processing and extraction of the EDL parameters. The processes and network layout are introduced in section 4.1.3.

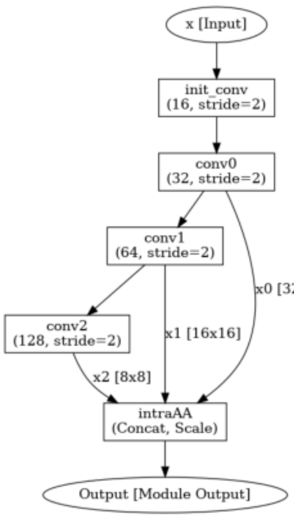


Figure 4.3: Feature extraction pipeline of the network.

Feature extraction

From each individual image for each viewing direction, features are extracted using a shared backbone feature extraction block. Figure 4.3 visualizes the feature extraction pipeline for each individual image marking the input. After various convolutions, a specialized *Intra-view Adaptive Aggregation Module* follows until the output marks a feature representation of each image, still represented in its original viewing direction.

Intra-view Adaptive Aggregation Module

The *Intra-view Adaptive Aggregation Module* (fig. 4.4) marks an interesting approach to

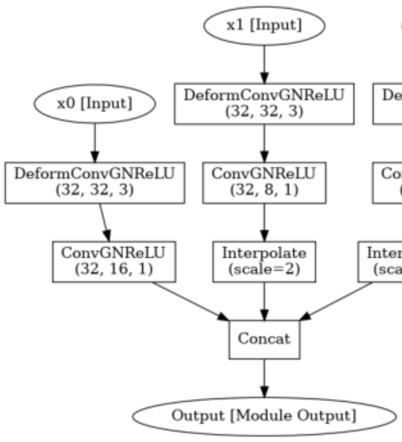


Figure 4.4: Layout of the Intra-view Adaptive Aggregation Module.

tackle problems often faced in MVS where surfaces contain areas without much texture, lacking information. For these regions of an image, or more precisely, parts of surfaces visible from multiple viewpoints, point-to-point matching can be challenging due to indistinguishable features. Also variations in the scene like changing lighting conditions such as reflections represent challenges that must be overcome to perform MVS.

The **DeformConvGNReLu** layer marks important part of the network. Deformable Convolutional Networks (DCNs) allow for adaptable receptive fields, which means they can modify the spatial sampling locations in the input feature maps according to the input data. This characteristic is particularly beneficial for handling geometric deformations within images, such as those caused by reflections or irregular surfaces. This is done by learning an offset beside the usual stride used in regular convolutions. This allows for the concatenation of features from different regions and finally allows for unique features in regions where otherwise feature uniformity would result in loss of detail and inability to distinguish between different objects or phenomena within the same spatial context.

Additionally, the module uses **Feature Pyramiding** and **Hierarchical Processing**. The module processes inputs of different resolution (x0, x1, x2) through separate pathways initially, which are then merged. This approach is known as a feature pyramid, where features extracted at various scales or resolutions are combined. By doing so, it effectively captures both high-resolution details and broader contextual information. This can be especially useful when dealing with reflections, as it helps in distinguishing the reflected features from the actual features by considering information at different scales.

The **ConvGNReLu** Layer is build from a combination of convolutional layers in addition to group normalization while using ReLU as activation function. This is a typical layout for a layer in a neural network to perceive visual information. The convolutional layer extracts features which are than normalized to ensure stable and cosistent training. The ReLU activation function ensures the necessary non-linearity to learn complex, non-linear pattern efficiently.

To be able to match features of different resolution, **Interpolation** and **Up-Sampling** is being used. Due to down-sampling or due to the deformation handling in previous layers information is being lost, so combining low resolution and higher resolution feature maps is unavoidable. Interpolating to a higher resolution and then combining it with other feature maps ensures that the network utilizes both local detail and global contextual information, crucial for analyzing complex textures and surfaces like shiny or reflective ones.

The final **Concatenation** of the upscaled feature maps from different resolutions with applied DCN results in a rich diversified feature map representation of the input image even in regions of uniformity in RGB space due to the processing in different resolutions with features taken from different parts of the image.

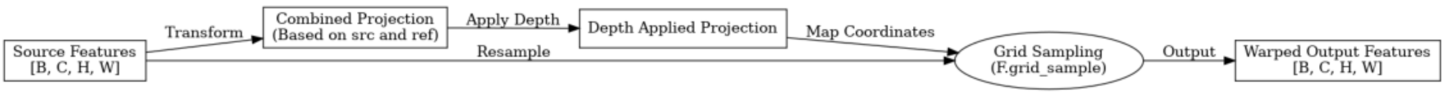


Figure 4.5: Homography warping is performed depthwise after feature extraction.

Homography

After extraction of relevant and rich feature maps $F_{1..n}$ for each input image I_x, views are morphed using homography so all of them match the reference viewing direction. Overview of the process is provided in figure 4.5.

The homography warping block performs depth-aware image warping based on source and reference projection matrices, allowing for perspective transformations of source features into a target or reference view. Initially, the function computes a combined projection matrix by multiplying the source projection matrix with the inverse of the reference projection matrix. This combined matrix encapsulates the transformation needed to map points from the source to the reference coordinate frame.

The function then applies a depth value to each point in the source feature map. This step involves generating a meshgrid of pixel coordinates, which are converted into homogeneous coordinates and transformed using the computed rotation and translation components of the projection matrix. As result, the feature map is available in reference view direction. To get a representation in 3D space, the 2D feature map is scaled by the provided depth values, resulting in a feature matrix with depth of the desired amount of depth values, repeatedly filled with the 2D feature maps. In the end, this process projects each pixel's location into the 3D space defined by the depth and camera parameters.

After projecting the coordinates, the function normalizes these to ensure they fall within the image boundaries, preparing them for the final resampling step. Source features are warped onto the reference view based on the computed grid. This sampling uses bilinear interpolation, preserving continuity and reducing artifacts. The output is a new set of feature maps that represent the source image as viewed from the perspective of the reference camera, adjusted for depth.

Cost volumes

This set of feature maps can now be used to calculate warped Cost Volumes C^w. For this the squared difference between warped features from the k-th view F_k^w and the reference view F^r is calculated.

$$\mathrm{C}_k^w = (F_k^w - F^r)^2 \tag{4.1}$$

After this step, a reweighting takes place that is performed inside the cost regularization and refinement part of the network.

4.1.2 Regularization and refinement

Set up with Cost Volumes describing matching cost between the features of each source view and the reference view, this part of the network takes care of comparing the information from the different views. For this purpose, one Cost Volume is created and then refined to get a distribution over probabilities for the depth of each pixel.

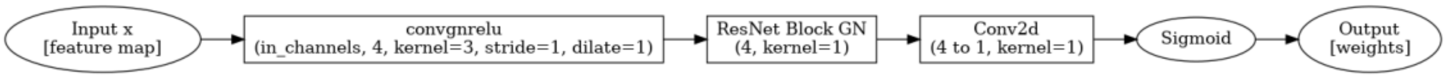


Figure 4.6: Layout of the Inter-view Adaptive Aggregation Module to combine features of different viewpoints.

Inter-view Adaptive Aggregation Module

After the homography transformation has been applied to align the feature maps across different views, the Inter-view Adaptive Aggregation Module performs a critical step in refining the stereo matching process. This module undertakes the reweighting of the homography-transformed Cost Volumes. The purpose of this reweighting is to adaptively adjust the contribution of different pixels based on their alignment and consistency across views. By applying learned or predefined weights to these Cost Volumes, the module effectively enhances the robustness and accuracy of depth estimation, focusing on areas with high confidence and suppressing potential errors due to misalignments or occlusions in the input images. This approach is essential for generating precise and reliable disparity maps from the processed Cost Volumes.

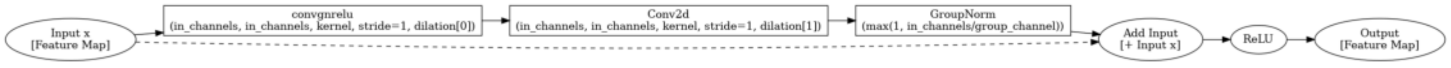

Input x
[Feature Map]
convgnrelu
(in_channels, in_channels, kernel, stride=1, dilation[0])
Conv2d
(in_channels, in_channels, kernel, stride=1, dilation[1])
GroupNorm
(max(1, in_channels/group_channel))
Add Input
[+ Input x]
ReLU
Output
[Feature Map]

Figure 4.7: The Resnet Block with Group Normalization.

Resnet Block with Group Normalization

The **Resnet Block with Group Normalization** module is a refined architectural component designed to optimize the performance of the network by incorporating two fundamental innovations in deep learning: residual learning and group normalization, both of which address specific challenges in training very deep networks. An overview of the setup is given in figure 4.7

Residual learning is implemented to combat the vanishing gradient problem that typically occurs in deep networks. By introducing shortcut connections that bypass one or more layers, the Resnet Block allows gradients to flow directly through these shortcuts, making it easier to train deeper networks effectively. The shortcut essentially performs identity mapping, and its output is added to the output of the layers being bypassed. This setup not only preserves the integrity of the input data throughout the network but also encourages feature reuse, which is beneficial for learning complex patterns without increasing computational burden.

Group Normalization (GN) is another key feature of this module and represents a shift from the conventional Batch Normalization (BN), which relies on batch size to compute the mean and variance used for normalization. Instead, GN divides each feature channel into groups and normalizes the features within each group based on their mean and variance. This method is particularly advantageous in situations where small batch sizes are used, as it maintains consistent training and inference performance regardless of batch size variations.

Together, these features enable the Resnet Block with Group Normalization to maintain stable training dynamics, facilitate faster convergence, and improve the generalization of the network. The inclusion of GN also simplifies the optimization landscape by reducing dependency on batch size, making the module adaptable to different training environments and hardware configurations. This adaptability, coupled with the enhanced training efficiency provided by residual connections, makes this part an essential component of the network's ability to adapt to different amounts of viewing directions used.

Cost Volume Creation

As stated in equation 4.2, the reweighted C^r and warped C^w volumes are then combined to form one volume C^g.

$$C^g = \sum_{i=1}^{k}((C_i^r + 1) \cdot C_i^w) \tag{4.2}$$

The general Cost Volume C^g now incorporates information from all viewpoints and has dimensions of $D \times W \times H$ referring to the number of possible depth hypothesis and width and height of the image.

U-Net with Convolutional Long Short-Term Memory Module

The general Cost Volume C^g is further processed by a RNN-CNN hybrid network that has a layout like U-Net with LSTM structure. This module is used to *leverage spatial*

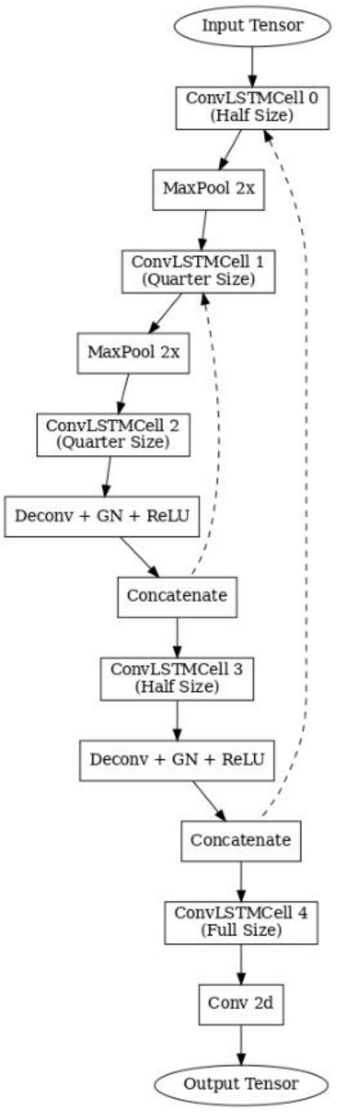

Input Tensor
ConvLSTMCell 0 (Half Size)
MaxPool 2x
ConvLSTMCell 1 (Quarter Size)
MaxPool 2x
ConvLSTMCell 2 (Quarter Size)
Deconv + GN + ReLU
Concatenate
ConvLSTMCell 3 (Half Size)
Deconv + GN + ReLU
Concatenate
ConvLSTMCell 4 (Full Size)
Conv 2d
Output Tensor

Figure 4.8: U-Net structure with LSTM cells.

context information and to turn matching costs into a probability distribution of D depth hypotheses [46]. This layout allows the network to capture visual information included in the features while keeping memory of previous inputs. So after this step, in the original network layout the output of the RNN-CNN module represents a probability distribution over possible depth levels.

Convolutional LSTM Cell

The ConvLSTMCell combines convolutional and recurrent neural network functionalities to process sequencial data effectively. It initializes with parameters that define the spatial dimensions and the depth of input and output channels, incorporating convolution operations within the LSTMs gating mechanism. Specifically, it uses a single convolution layer to simultaneously handle both the input tensor and the hidden state, producing an output with four times the channels of the hidden dimension, corresponding to the LSTM's input, forget, output, and cell state update gates. These gates - each activated by specific functions like sigmoid for the input, forget, and output gates, and tanh for the cell update gate - allow the cell to decide how much new information to accept, how much old information to retain, and how much of the cell state to output.

As the cost volume C^r already incorporates information from all viewpoints, the usage of an LSTM module which stores information from one processing step to another seems to be unintuitive first. But the way this part works explains the refinement character of the module. Processing inside the LSTM cell is performed by sweeping through the cost volume C^g constructed in the preliminary steps. So for each depth hypothesis inside the Cost Volume, this module ensures spatial coherence for adjacent pixels which are likely to be close in the spatial dimension of the real world. The processing inside the Convolutional LSTM Cell can also help with occluded areas as it keeps track of feature similarities from previous depth hypothesis, which allows this module to output rough estimates of possible depth levels if there is a transition from matching features on one side of an edge to the other side.

Finally, after feature extraction, processing and refinement, the refined Cost Volume C^r is fed into a Softmax function to form the Probability Volume P. In the original network layout, this volume would now be seen as probability distribution over the possible depth hypothesis and classification, meaning the depth with highest probability is chosen as final depth value for this pixel. Inside EMVSNet, this Probability Volume P marks the input for the second part of the network, whose layout will be explained in the following section 4.1.3.

4.1.3 Prediction of EDL parameters

The structure of the network until the creation of the Probability Volume P does not incorporate the prediction of the parameters needed for EDL. The task of the second part of the network, discussed in this section, is to predict the NiG parameters based on the Probability Volume created. Appendix 8.2 provides a general graphical overview over this part, which is further analyzed here.

Volume creation and Interpolation

As a first step, there are three volumes of different resolution created based on the Probability Volume P. These Volumes are denoted as $V_{1..3}$ and have the same, one half and one quarter the resolution of P. For interpolation, the trilinear type is used with usage of aligned corners. This method ensures the volume to keep values at corners and prevents shifts or distortions introduced by the interpolation process. As will be described in the following, the Volumes $V_{1..3}$ at different resolutions are used to extract the parameters α, β, γ and ν of the NiG distribution. This ensures that smaller and wider area around a pixel are incorporated into the prediction of the uncertainty of the depth estimation.

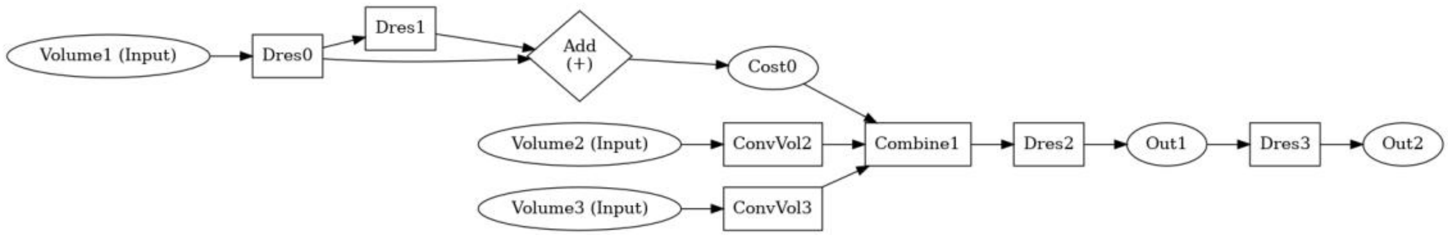


Figure 4.10: Dense Residual Block of the proposed network.

Dense Residual Block

The Dense Residual Block, visualized in figure 4.10, feeds the different inputs into the Hour Glass network with both dense and residual connections. This is done through dense connections, where the feature maps from all preceding layers are concatenated as input to each subsequent layer. Additionally, residual connections are incorporated to facilitate gradient flow and improve training stability. The shortcut connections allow the network to effectively learn residual functions with reference to the input, which mitigates the vanishing gradient problem. By combining dense and residual connections, the Block is able to receive complex patterns. Both Outputs of the Dense Residual Block are fed into the Hours Glass network, marking the next step of the pipeline.

Hour Glass Network (UP&Down)

The Hours Glass network is shown in appendix section 8.1. It is build as identical network structure but fed with different input feature maps. This type of network architecture is designed for tasks requiring precise spatial understanding, like it is necessary if analyzing the position of objects in 3D space. The encoder-decoder nature of this network part allow it to combine low-level details with high-level semantics. Even more fine-graded analysis of the content is possible through the different input sized of features fed into the networks by also combing the results in the end.

Classification Layer

The Classification Layer (figure 4.11) is applied 3 times total and for the first time in

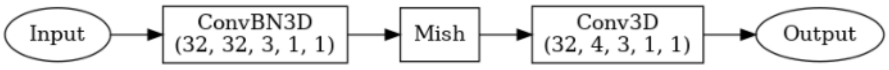


Figure 4.11: Classification Layer 1-3, all equally setup to predict the desired four chanels.

the network's flow outputs for each feature map the four parameters needed. *Cost* 0, *Out* 1 and *Out* 2 are used as input, providing different resolution feature map to capture information in multiple resolutions. The ConvBN3D Block (figure 4.12) just represents 3D Convolution followed by BN. The output of this layer is given as the logarithmic of α,

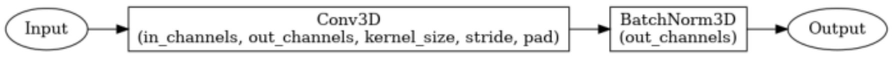


Figure 4.12: ConvBN3D Block made of 3D Convolution followed by Batch Normalization.

β, γ and ν in different resolution. To be able to combine them, they are upsampled using trilinear interpolation. The upsampled Cost Matrix is fed through a Softmax function resulting in a probability distribution. To extract the parameter γ, which represents the mean of expected values, this distribution of probabilities is used for regression by the predefined depth levels. The extracted probability distribution is also used to calculate the remaining parameters by elementwise multiplication.

As a last step, these parameters, which are seen as logarithmic versions to ensure positivity, are combined.

Combine uncertainties

The combination of the parameters is done iteratively, resulting in a single set of parameters $(\alpha_i, \beta_i, \gamma_i, \nu_i)$. The used fusion strategy has been adopted from *MoNIG* [84]. The following equations describe one iteration of the fusion process:

$$\alpha = \alpha_1 + \alpha_2 + \frac{1}{2}$$
$$\beta = \beta_1 + \beta_2 + \frac{1}{2}\gamma_1(\nu_1 - \nu)^2 + \frac{1}{2}\gamma_2(\nu_2 - \nu)^2$$
$$\gamma = \gamma_1 + \gamma_2$$
$$\nu = (\gamma_1 + \gamma_2)^{-1}(\gamma_1\nu_1 + \gamma_2\nu_2)$$

Calculation of β *is defined as not only the sum of* β_1 *and* β_2*, but also the variance between the combined distribution and each individual distribution, as it provides insight into aleatoric and epistemic uncertainties simultaneously* [84]. The parameter ν is also weighted by γ, which *measures the confidence of the expectation.*

4.2 Uncertainty and Loss

With the network structure set up to predict a set of NiG distribution parameters $(\alpha, \beta, \gamma, \nu)$, it remains to define the formulae for epistemic and aleatoric uncertainty as well as the Loss. In this regard, the proposed network structure has been influenced by the finding of Meinert et al. [22].

Equation 4.3 shows the **Default** formulae used to describe aleatoric and epistemic uncertainty. This set of formuae differs from the ones used in the original implementation by Amini et al. [21], which has been introduced in section 2.4.4 and will be used and described as **Alternative** set of equations, explained in the following section.

$$\underbrace{\mathrm{E}[\sigma^2] = \sqrt{\frac{\beta \cdot (\nu + 1)}{\nu \cdot \alpha}}}_{\text{aleatoric}}, \quad \underbrace{\mathrm{Var}[\mu] = \frac{1}{\sqrt{\nu}}}_{\text{epistemic}} \tag{4.3}$$

With the arguments against the original representation of Amini et al. [21], using the modified set of equations as default is a reasonable choice. But still it is shown that practice often the quantification of uncertainty using the traditional methods is rather good. The paper ows its title to this fact as the representation is being described as *unreasonable effective*.

The introduced default expression to describe aleatoric uncertainty is identical to the general shape parameter introduced in equation 2.24. Representation of the variance has been adjusted so it only depends on the weighing parameter ν. This is an interesting approach as it suggests that the overall shape of the Student t-distribution is the primary factor for aleatoric uncertainty or, in other words, the aleatoric and epistemic uncertainty follow a distribution of the same shape and the describing factor for the epistemic uncertainty is only given by ν.

As stated in the paper by Meinert et al. [22], as **Default** Loss function for EMVSNet, this function has been adopted as stated in equations 4.4 and 4.5.

$$\sigma_i^2 = \frac{\beta_i}{\nu_i} \tag{4.4}$$

$$L_i' = \log \sigma_i^2 + (1 + \lambda \nu_i) \frac{(y_i - \gamma_i)^2}{\sigma_i^2} \tag{4.5}$$

The authors say the Alternative Loss function suffers from overparameterization, which means it could be minimized independently of the data. This can lead to solutions that not properly disentangle the uncertainties. Furthermore, separation between aleatoric and epistemic uncertainty has been improved. The effect of the theoretical advantages this choice makes will be explored in chapter 6.

4.3 Adaptions

The proposed network structure is one way to handle the problem, but there is potential to adress serveral challenges by adjusting input data, network layout or processing procedure. As already mentioned above, section 4.3.1 introduces the usage of an Alternative set of formulae to describe aleatoric and epistemic uncertainty and the Loss function. Section 4.3.2 deals with the already experimentally proven advantages usage of the Exponential Linear Unit (ELU) function could provide. The final section 4.3.3 details the modifications introduced to integrate MCD into the network architecture, enabling the subsequent experiments.

4.3.1 Alternative formulae and Loss

The initially published paper by Amini et al. [21] about EDL introduced the formulae 4.6 to quantify the prediction as well as aleatoric and epistemic uncertainty.

$$\underbrace{\mathrm{E}[\mu] = \gamma,}_{\text{prediction}} \quad \underbrace{\mathrm{E}[\sigma^2] = \frac{\beta}{\alpha - 1},}_{\text{aleatoric}} \quad \underbrace{\mathrm{Var}[\mu] = \frac{\beta}{\nu(\alpha - 1)}}_{\text{epistemic}} \tag{4.6}$$

This set of formulae to describe aleatoric and epistemic uncertainty as well as the originally introduced Loss function as stated in equations 2.25-2.28 are used as **Alternative** method in course of the experiments.

4.3.2 Two stage network

Li et al. [85] find that *ROC AUC achieved by the EDL method is significantly lower than that obtained by cross-entropy loss.* Also they believe that *EDL tends to be sensitive to initialization and some hyper-parameters, where improper settings may lead to significantly degraded AUC and unreliable uncertainty estimation.* They suggest the usage of a two-stage network setup to train an underlying network using Cross Entropy Loss for superior learning of the prediction for the fundamental task and only estimating the parameters for EDL in a second step.

In the paper, the authors also replace the ReLU function with the ELU function, which leads to significant improvements in performance. The ReLU activation function (equation 4.7) is widely used in neural networks due to its computational efficiency.

$$\mathrm{ReLU}(x) = \max(0, x) \tag{4.7}$$

However, in certain scenarios, its use can lead to the dying ReLU problem, where neurons cease to learn and update their weights during training. This issue occurs when the input to the ReLU function is consistently negative, resulting in a ReLU output of zero. Consequently, the gradient through these neurons is zero, preventing any weight updates during backpropagation.

To address these limitations, the ELU activation function is employed, which can be described by the following equation:

$$\mathrm{ELU}(x) = \begin{cases} x & \text{if } x \geq 0 \\ \alpha(e^x - 1) & \text{if } x < 0 \end{cases} \tag{4.8}$$

where α is a hyperparameter that defines the value that ELU approaches as the input becomes large and negative.

The ELU function in this case offers several advantages including a non-zero gradient for negative inputs which mitigates the dying ReLU problem by allowing updates to the neuron weights even when the input is negative. It is also smooth for all values of x, aiding in the acceleration of the convergence of the learning process due to more predictable gradients throughout training. Additionally, it has been shown that networks using ELU tend to generalize better on unseen data compared to those using ReLU, likely because ELU can produce more robust features at each layer of a neural network. These attributes make ELU particularly suitable for the two-stage network proposed by Li et al. [85], where robust and reliable feature representation is crucial for effective uncertainty estimation in the second stage of training.

For EMVSNet, the network layout given by AA-RMVSNet is used as first stage. The following part is not explained in detail here as it has some flaws visible in the experiments. But basically a setup of convolutional layers combination with the proposed ELU function is used to determine the four parameters describing the NiG distribution.

4.3.3 Monte Carlo Dropout

To evaluate the EDL-based approach against a more traditional method using MCD, the network layout of EMVSNet was modified. Specifically, to the evidential components of the network two additional dropout layers were added. These 3D dropout layers were integrated into the HourGlass and HourGlassUp modules, applied to the original outputs of these modules as described in Section 4.1.3. The input to these dropout layers is of size $C \times D \times H \times W$, where the PyTorch dropout layer zeros out entire channels of the input with a probability p.

Maintaining the original network layout with the addition of dropout layers ensures comparability between the approaches. In practical applications, when using MCD to estimate uncertainty, it would typically be preferable to use Mean Squared Error (MSE) and Mean Absolute Error (MAE) instead of the EDL loss function discussed in section 4.2. However, in this case, retaining the proposed network layout without further modification additionally provides valuable insights into the potential combination of traditional methods with EDL.

5 Experimental setup

This chapter presents a comprehensive overview of the experimental setup for EMVSNet. The first section 5.1 explains the objectives of the analysis and conducted experiments. The following section 5.2 provides detailed explanation of the general setup including used hardware and dataset selection. The subsequent section 5.3 provides details about the parameters used in training and presents metrics recorded while training. The last section 5.4 explains details about metrics used.

5.1 Objectives of the analysis

The primary objective of the experiments conducted in chapter 6 is to evaluate whether the prediction of uncertainty can inform the reliability of a given depth estimation. As previously discussed, employing such a system could provide valuable insights and safety-related recommendations in practical applications. Moreover, integrating these results into 3D reconstruction can enhance reconstruction performance, particularly when the system prioritizes predictions with low uncertainty. The detailed reconstruction process for a complete 3D model is out of scope for this work, but if multiple contrary depth hypothesis for one point in the real world exist, determine unreliable hypothesis or weighting the individual calculations can tremendously increase performance and reliability [86].

The theoretical performance of such a determination system to filter unreliable predictions is illustrated using Precision-Recall (PR) plots generated under various conditions. Like explained in section 5.4, these plots are capable to show how well a system would predict erroneous regions. Generated Receiver Operating Characteristic (ROC) curves are are capable to determine a good balance between True Positive Rate (TPR) and False Positive Rate (FPR) for this task.

Additionally, a significant area of interest lies in comparing this approach with other methods for predicting uncertainty and assessing inter-dataset performance. In real-world applications, domain shifts between training and inference data are inevitable. Therefore, exploring the system's performance in processing OOD data is of particular importance.

Furthermore, a noteworthy aspect of EDL is the potentially minimal increase in runtime and hardware requirements. Comparison aganinst the underlying network's implementation in terms of run and training time is conducted.

In the course of chapter 6, it will be differentiated between aleatoric, epistemic and combined uncertainty. The combined uncertainty in this case describes the sum of aleatoric and epistemic uncertainty. This unit of measurement has been added because there might be cases where one or another type of uncertainty is mainly present. If the performance of the prediction is then compared against the error, only for one of the methods the prediction of an erroneous region would be true. But for combined uncertainty, it might be the case that the prediction of such cases is enhanced.

5.2 General setup

The general network setup has been adapted from AA-RMVSNet, so it is implemented in Python using PyTorch.

Hardware

The training procedure of the presented EMVSNet has been performed on a local machine with following hardware specifications:

- **OS**: Ubuntu 22.04.4 LTS
- **CPU**: Intel i7 12700K with 12 Cores/24 Threads
- **RAM**: 32 GB DDR4 3200 MHz
- **GPU**: 2x Nvidia RTX 3090 24GB
- **CUDA**: CUDA 11.7 / Driver version 535.171.04

As the system used is limited in PCIe lanes, the dual GPU setup is only able to use 8 PCIe lanes for each card. In addition, the power supply with maximum output of 850 watts required to apply a power limit to both Graphics Processing Units (GPUs) of 300 watt each to ensure stable operation. Even though this procedure limited performance of the components, the hardware has been more than sufficient to train the network.

Datasets

As dataset for training, a subset of 84 scenes of the proposed DTU dataset is being used. A full list of all scenes used is attached in the Appendix section 8.3. The remaining scenes of the DTU dataset are then used for the test (24 scenes) and validation (18 scenes) set, also specified in the appendix. In general, the DTU dataset seems to be the most broadly used in terms of MVS. Specifics of the dataset have already been presented in section 3.3, but with its characteristics as laboratory setup with high accuracy and challenging lightning conditions it naturally appears as good option for a training dataset.

In addition - to provide the opportunity of cross dataset experiments showing the impact of a domain transfer - the TnT dataset is used exclusively for testing purposes. From this dataset, three scenes are utilized:

- Lighthouse
- M60
- Panther

With large scale outdoor scenes captured in the real world they mark a tremendous shift in domain and the idea behind their utilization is a comparison in performance and the feasibility of employment on other datasets.

5.3 Training

The training process involves some more parameters for configuration, this list provides an overview of the most important parameters:

- **Total Steps**: ~ 240.000
- **Duration**: ~ 26 hours
- **Training Scenes**: 84
- **Depth levels**: 32
- **Number of views**: 3
- **Batch size**: 1
- **Optimizer**: Adaptive Moment Estimation (ADAM)
- **Input image resolution**: 128 x 160 pixel
- **Total parameters**: 4.494.115
- **Learning Rate Scheduler**: Cosine Annealing
- **Learning Rate**: $1 \times 10^{-3} \rightarrow 2 \times 10^{-6}$
- **Weighting factor**: $\lambda = 0.1$
- **Stopping criterion**: Early stopping

The Learning Rate (LR) initially starts at its initial value and is decreased following the Cosine Annealing Scheduler to the minimal value. This is done over the entire planned maximum epochs. This means, other than usually with Cosine Annealing Scheduling, the LR is not altered between high and low states but follows π of a cosine cycle, thus going from high LR to low back to high. In some cases, this prevents getting stuck in local minima and additionally provides information in the later phase about the optimal LR setting.

The parameters of the network are initialized like the default behaviour of PyTorch. The initialization methods for different layers are summarized in the table 5.1. The weighting factor has been adopted from literature, there have no experiments been performed to optimize this parameter. Stopping has been done based on the test set Loss, with stopping performed if decrease is below 0.2.

Training of both, EMVSNet and the proposed variation to incorporate MCD into the network has been performed according to the hardware presented in section 5.2. Details of the training results for EMVSNet incorporating MCD are presented below.

EMVSNet

In Appendix 8.8, graphs showing metrics in the network's training process are presented. Absolute error and loss as well as mean uncertainty predictions are captured and visualized using a smoothed graph. It can be seen that over the term of the total steps, error and loss decline as expected.

Additionally, error metrics are plotted, showing how many prediction are located outside a certain range of accepted error. These trends show a decrease within the training process. While these trends still decline slightly after the training, appendix 8.9 shows the same metrics for the testing set, with error and uncertainty graphs relatively stable when training was stopped. Interestingly, the general trend in predicted uncertainties tends to go upwards for all of these graphs, while it declines in the training set. This is a behaviour worth further investigation in future work.

Appendix 8.10 show the results based on the full validation set. These graphs show as well that training has been performed good enough to stop training. In general, to further improve performance of the network, testing different parameters and optimizing the training process would be recommended.

Layer Type	**Weights**	**Biases**
Convolutional	Kaiming Uniform	Zero
Linear	Kaiming Uniform	Zero
Batch Normalization	One	Zero
ConvLSTMCell	Uniform	Zero
Deformable Convolution	Kaiming Uniform	Zero
Group Normalization	One	Zero

Table 5.1: Used initialization methods for the parameters of different layers in EMVSNet.

Alternative

The Alternative set of formulae with own Loss function has been trained using basically the same parameters as shown for EMVSNet. Only difference is this factor:

- **Weighting factor**: $\lambda = 0.01$

No experiments have been conducted to optimize this factor, instead the default recommendation from the literature has been adopted.

The broad fluctuation in values when using the alternative method can be seen in these graphs as well as in the testing results in appendix 8.12. The results for the validation set are presented in appendix 8.13.

Monte Carlo Dropout

Training of the network incorporating additional MCD layers has been carried out using the same parameters as well. The only change is that - obviously - the parameter count is increased slightly. Additionally, for the dropout part, these parameters are applied:

- **Dropout rate:** 50%
- **Dropout Input dimensions:** $32 \times 32 \times 128 \times 160$

In appendix 8.14, the graphs for MCD network training are presented. No unusual behaviour compared to the normal training method of EMVSNet can be explored here, so incorporation of dropout layers seems not to hinder the training process.

5.4 Metrics

Chapter 6 uses different metrics, which are introduced here. These allow for comparison of different compensation functions and determination of the model's general performance.

Mean Squared Error

Mean Squared Error (MSE) is a common metric used to evaluate the accuracy of a predictive model. It calculates the average of the squares of the errors, where the error is the difference between the predicted and actual values. MSE is defined as:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

where y_i is the actual value, $\hat{y}_i$ is the predicted value, and n is the number of observations. MSE is sensitive to large errors, making it useful for highlighting significant deviations in predictions.

Root Mean Squared Error

Root Mean Squared Error (RMSE) is the square root of the MSE. It provides an indication of the magnitude of the errors in the same units as the original data, making it easier to interpret. RMSE is defined as:

$$\text{RMSE} = \sqrt{\text{MSE}} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

RMSE is particularly useful when comparing the performance of different models on the same dataset.

Mean Absolute Error

Mean Absolute Error (MAE) measures the average magnitude of the errors in a set of predictions, without considering their direction. It is calculated as the average of the absolute differences between the predicted and actual values. MAE is defined as:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

MAE is less sensitive to outliers compared to MSE, providing a more robust measure of model accuracy.

Coefficient of Determination (R²)

The Coefficient of Determination (R^2) is a statistical measure that indicates how well the regression predictions approximate the real data points. It provides an indication of the proportion of the variance in the dependent variable that is predictable from the independent variable(s). R^2 is defined as:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

where y_i is the actual value, $\hat{y}_i$ is the predicted value, $\bar{y}$ is the mean of the actual values, and n is the number of observations. An R^2 value closer to 1 indicates a better fit of the model to the data.

ROC and PR curve

ROC curve and PR curve provide excellent measurements to analyze the performance of the model's uncertainty prediction. To classify the depth and uncertainty predictions, two thresholds have to be applied:

- **Uncertainty Threshold (T^u):** This is the threshold below which predictions are considered sufficiently certain. It delineates the level of uncertainty acceptable for the model to classify its predictions as reliable.

- **Error Threshold (T^e)**: This threshold defines the maximum permissible error between the predicted and actual values, beyond which predictions are considered inaccurate.

These thresholds are then applied to categorize each pixel based on its depth and uncertainty estimations:

- **True Positives** (TP): Cases where the error between the predicted depth and the actual depth is below the Error Threshold (T^e), and the associated uncertainty is below the Uncertainty Threshold (T^u), signifying high confidence in the accurate prediction.
$$TP = \{\text{Error} \leq T^e \wedge \text{Uncertainty} \leq T^u\}$$

- **False Positives** (FP): Cases where the error exceeds the Error Threshold (T^e), but the associated uncertainty is below the Uncertainty Threshold (T^u), indicating incorrect confidence in an inaccurate prediction.
$$FP = \{\text{Error} > T^e \wedge \text{Uncertainty} \leq T^u\}$$

- **True Negatives** (TN): Cases where both the error between the predicted depth and the actual depth exceeds the Error Threshold (T^e) and the associated uncertainty is above the Uncertainty Threshold (T^u), correctly indicating a lack of confidence in inaccurate predictions.
$$TN = \{\text{Error} > T^e \wedge \text{Uncertainty} > T^u\}$$

- **False Negatives** (FN): Cases where the error exceeds the Error Threshold (T^e) and the uncertainty is below the Uncertainty Threshold (T^u), indicating a failure to accurately predict within the acceptable error range despite high confidence.
$$FN = \{\text{Error} > T^e \wedge \text{Uncertainty} \leq T^u\}$$

For both curves, from the defined values above there are key figures which must be specified and calculated:

- **Positive Predictive Value** (Precision): $P = \frac{TP}{TP+FP}$
- **True Positive Rate** (Recall): $TPR = \frac{TP}{TP+FN}$
- **False Positive Rate** (Specificity): $FPR = \frac{FP}{FP+TN}$

As the ROC curve and the PR curve are used for classification tasks, their representation of predictive performance aligns with the main interest of the measurements conducted. Classification allows for a decision weather a given depth estimation is trustworthy. If further usage of the data produced includes weighted incorporation of the results, for example used in the further 3D reconstruction pipeline, performance potentially might be even better than if strict thresholds are applied.

The **ROC curve** plots the TPR against the FPR across various threshold settings. The TPR measures the model's ability to accurately predict the depth while assigning a high certainty to these correct predictions. Conversely, the FPR quantifies the proportion of instances where predictions with significant errors are mistakenly classified as certain.

The **PR curve** focuses on the relationship between Positive Predictive Value (PPV) and TPR, which is particularly important in situations where classes are imbalanced. Precision is calculated as the proportion of correct positive identifications among all positive predictions made, measuring the accuracy with respect to the certainty. This curve becomes especially valuable when the cost of false positives is substantial, or when the positive class carries more significance than the negative class.

The PR Curve, using thresholds for error and uncertainty, categorizes predictions to assess if the model can reliably indicate when it is likely to be correct or incorrect. This method is essential for operational settings where decisions based on model predictions require high reliability. The PR curve's limitation in only evaluating the arrangement of errors, without indicating the types of error distributions, suggests that while it can show how well the model separates trustworthy predictions from uncertain ones, it doesn't reveal the underlying error characteristics or the exact nature of the uncertainties involved.

Accuracy and F1 Score

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{5.1}$$

Another interesting measurement is the predictive performance based on different thresholds for uncertainty. Summarily to the definition above, thresholds T^u for uncertainty and T^e for the prediction error are chosen to classify the predictions. Then, accuracy (equation 5.1) and F1-Score (equation 5.2) are calculated based on different values for T^u.

$$\text{F1-Score} = 2 \cdot \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{5.2}$$

The F1-Score can also visually be seen in PR curves, marking the point where precision and recall curve cross.

Histogram

The Histogram can be seen as an indicator for the direct relationship between uncertainty and error. If the histogram shows a strong positive correlation, it suggests that the model's uncertainty estimates are aligned with actual prediction errors, validating the model's ability to estimate its own reliability accurately. This tool is particularly useful for visualizing the spread and central tendency of errors as they relate to uncertainty, providing insights into whether the model consistently recognizes more difficult or uncertain cases.

Calibration Plot

The Calibration Plot offers an analytical view of how effectively the model quantifies its certainty. By plotting the error normalized by the standard deviation against predicted uncertainty, one can assess the calibration of uncertainty: whether the model's confidence in its predictions is justified based on actual outcomes. A well-calibrated model will show that higher uncertainties correlate with higher errors, indicating an accurate awareness of its prediction limits. This plot is helpful for evaluating the model's reliability across different levels of confidence and ensuring that it does not systematically overestimate or underestimate its capabilities.

6 Experiments and Results

The experiments outlined in this chapter assess various aspects of EMVSNet's performance. Both qualitative and quantitative analysis are conducted, utilizing diverse datasets and evaluating the impact of different formulae.

Within the first section 6.1, the performance of depth and uncertainty prediction using different metrics is analyzed. Section 6.2 extends this analysis with visual evaluation, which can play an important role in performance measurement. Section 6.3 presents experiments that are conducted using different amount of viewpoints to gather their influence. In in section 6.4, data from the TnT dataset is used to evaluate inter-dataset performance and specific characteristics of the uncertainty prediction with data unrelated to the training set. The following section 6.5 compares the Default EMVSNet approach to the Alternative set of formulae, presented in section 4.3.1. The subsequent section 6.6 uses MCD to compare the prediction quality of the EDL based approach against. This adaption of the network has been introduced in section 4.3.3. Performance of another adaption of the original network layout is shown in section 6.7, which has been introduced in section 4.3.2.

6.1 General performance

To analyze the network's performance of uncertainty prediction, as first intention, it is rational for a large set of samples, which is present in the form of many pixels per image, to connect uncertainty and error. Statistically, for many samples, higher uncertainty should lead to higher error. This assumption yields to some interesting performance indicators to consider:

- **Depth estimation:** To evaluate the general performance of the network, it is crucial to compare it against the network without the EDL part. This is done by comparing the DTU test set metrics against AA-RMVSNet.
- **Heatmap:** Uncertainty plotted against the real Error. Over many samples, a positive correlation can be assumed. Additionally, it is interesting to explore the nature of this relationship, whether it is quadratic, linear, or another form, to better understand how uncertainty and error are interconnected.
- **Calibration Plot:** Error divided by Standard Deviation, plotted against the predicted uncertainty. This plot can be used to determine weather bigger variance in the predictions corresponds to higher predicted uncertainty. This way it can be determined if the model is overconfident or under confident in its predictions.

Model	Views	D	Loss	Time [ms]	MAE [mm]	2mm ↓	4mm ↓	8mm ↓	16mm ↓	32mm ↓
EMVSNet	3	32	17.68	169	7.37mm	44.45%	23.39%	11.68%	6.49%	3.93%
AA-RMVSNet	3	32	1.09	132	20.25mm	79.88%	60.51%	29.94%	15.30%	11.57%
AA-RMVSNet	7	128	1.69	440	5.90mm	39.55%	17.89%	8.77%	5.51%	3.84%

Table 6.1: Performance comparison of EMVSNet and AA-RMVSNet on the DTU test set, showing the MAE and percentage of predictions outside of various error thresholds. Results are presented for different numbers of views and disparity levels D.

- **PR Curve:** Two thresholds T^e and T^u for error and uncertainty are applied. These thresholds categorize into the classes correct or incorrect prediction for T^e and trustworthy and uncertain prediction for T^u. This curve shows if the model can be used to determine if a prediction is trustworthy, so if the model can be used effectively. As limitation, this curve only evaluates the arrangement of error distributions — how errors are sorted, yet it does not specify the assumed types of these distributions.

These tools to measure the network's performance are presented in regard of theoretical meaningfulness as well as the results found.

The quantitative performance of the proposed network in this section is mostly analyzed in relation to the real error measured. Without ground truth data for uncertainty quantification, these experiments should express how uncertainty prediction could improve reliability of estimations.

Comparison against AA-RMVSNet

Table 6.1 presents the performance results on the DTU test set for both AA-RMVSNet and EMVSNet. The testing parameters for EMVSNet were carefully selected to align with its training parameters, as well as the default optimized testing parameters reported by Wei et al. [46].

It is noteworthy that the EMVSNet model, despite utilizing only 32 disparity levels D and 3 views, demonstrates performance comparable to AA-RMVSNet, which operates with $D = 128$ and 7 views. When AA-RMVSNet is evaluated on the test set with only 3 views and $D = 32$, its performance is significantly lower than that of EMVSNet. Overall, this comparison indicates that the two networks perform similarly, with EMVSNet likely surpassing AA-RMVSNet if their parameters were fully aligned. Further research could potentially confirm this finding.

While AA-RMVSNet demonstrates a significant advantage in inference time compared to the more complex EMVSNet, an even more notable difference emerges during training. Table 6.2 presents the average time required to train a single sample using a batch size of one. In this context, the more intricate architecture of EMVSNet reveals its limitations, as it leads to increased training time.

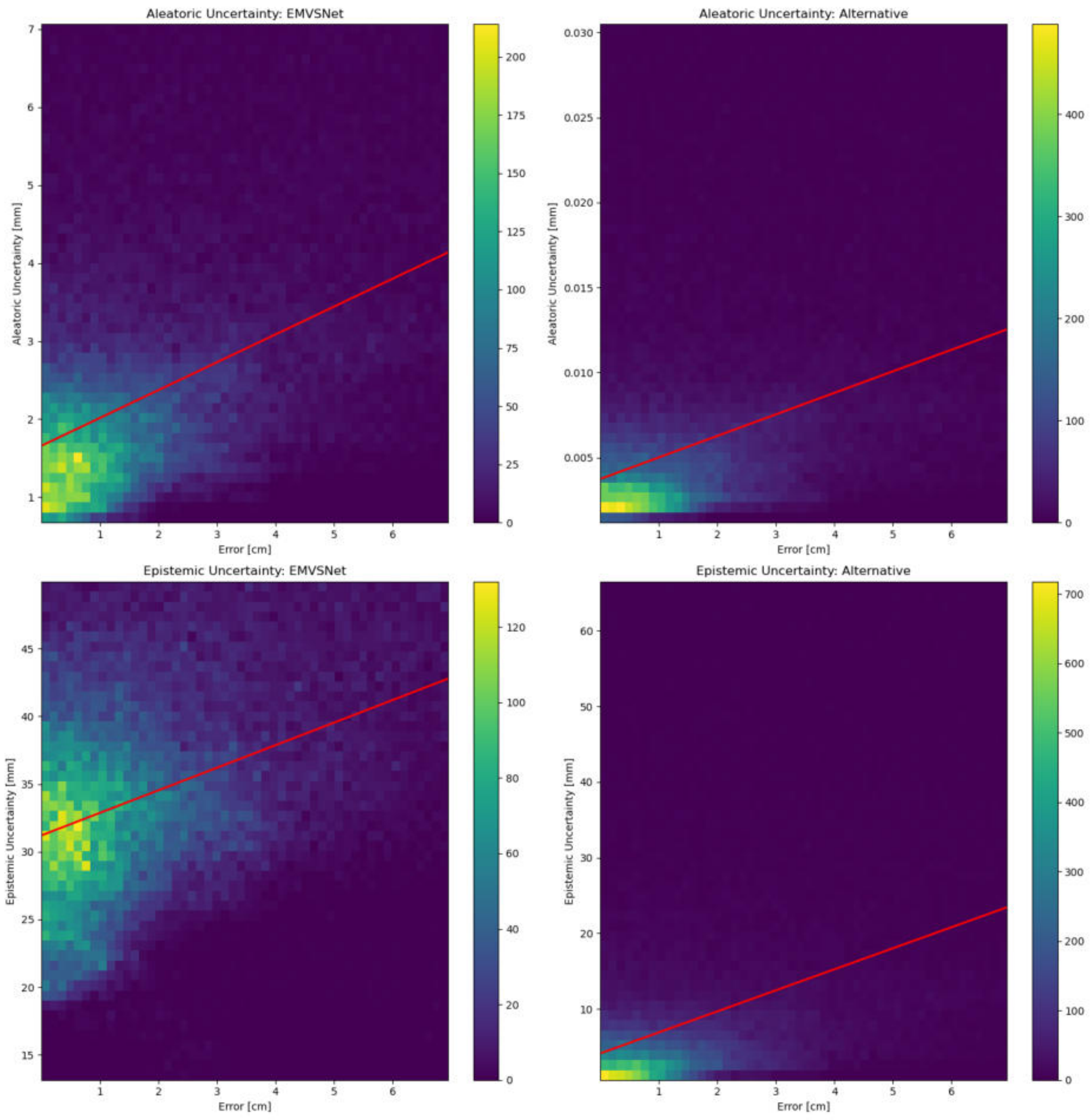


Figure 6.1: Visualization for 50 bins of uncertainty predictions versus actual error as Heatmap. The top 10% of errors and uncertainty values are removed to filter outliers and improve visibility in the relevant region. The red line is indicating linear regression. The colored scale indicates the amount of samples in the associated bin.

Model	Mean Time
AA-RMVSNet	248 ms
EMVSNet	721 ms

Table 6.2: Average training time per sample for AA-RMVSNet and EMVSNet, illustrating the computational demands of each model. EMVSNet requires significantly more time to train due to its complex architecture.

Heatmap

The heatmap of uncertainty against error is shown in figure 6.1. This plot also includes the red linear regression line which shows positive correlation between error and uncertainty. The heatmap shows how in general the low error areas are predominant and in comparison to the Calibration plot, the variation in predicted uncertainty for certain errors is clearly visible.

To further analyze how the error is connected to the uncertainty, cubic regression is compared to the linear regression to see which better suites the data. Table 6.3 compares the performance of linear and cubic regression models using MSE, RMSE, MAE, and R^2. The results indicate that there is not a significant difference between these regression types, although cubic regression appears to provide a slightly better fit.

Type	Metric	Linear [mm]	Cubic [mm]
Aleatoric	MSE	23.29	22.96
	RMSE	4.83	4.79
	MAE	2.05	2.04
	R^2	0.35	0.36
Epistemic	MSE	30.54	29.85
	RMSE	5.53	5.46
	MAE	2.30	2.18
	R^2	0.15	0.17

Table 6.3: Comparison of Linear and Cubic Regression using different metrics. MSE, RMSE and MAE are given in millimeters.

The R^2 score can also indicate weather homoscedastic or heteroscedastic uncertainty is more predominant, as a higher score would indicate more variability with varying input data. The value of 0.35 indicates a fair mix of both types, more leaning towards the homoscedastic type of uncertainty.

Calibration Plot

The Calibration plot in figure 6.2 demonstrates an almost linear correlation between uncertainty and error. Additionally, figure 6.3 illustrates the distribution of predicted uncertainty values across the three scenes. To mitigate the impact of outliers, the highest and lowest two percent of predicted uncertainty values have been excluded. It is observed that in these extreme cases, the correlation between predicted uncertainty and MAE worsens.

A possible explanation for this deviation in extreme cases could be that while high uncertainty is correctly predicted in certain areas, the prediction is more blurred compared to the actual regions of high error. In this scenario, the general concept of increased uncertainty is accurate, but the exact location is not, leading to a reduced correlation due to the pixel-by-pixel comparison method, where smearing of predictions occurs.

Despite the excellent correlation shown in the calibration plot, a considerable variation in values can be identified when considering the Heatmap results. However, it is important to note that the Calibration plot utilizes more data points, as the Heatmap excludes 10 percent of the values. This exclusion does not negatively affect performance, as a positive correlation is maintained across almost the entire range of predicted uncertainty values, applicable to both types of uncertainty.

PR Curve

The PR curves for all three test scenes are visualized for each type of uncertainty in appendix 8.5. To fully understand the performance metrics and their impact on real-world data, it is crucial to consider the distribution of errors, as presented in appendix 8.4.

In the context of EMVSNet, the most beneficial application might be identifying regions with relatively high errors, which are expected to be the most critical concerning safety. The error distribution reveals that, at a threshold of 6 mm, over 80 percent of inliers are present across all scenes. Conversely, a strict threshold of 2 mm excludes up to half of the predictions, resulting in a hard separation. This effect is also reflected in the performance, with Equal Error Rate (EER) values increasing at higher thresholds.

Considering the error distribution, the relatively mediocre performance of EMVSNet in scene one at a 2 mm threshold becomes understandable. Across all scenes, it is evident that a threshold T^e of 6 mm yields F1-Scores above 90%, demonstrating the model's excellent performance. Generally, epistemic uncertainty prediction appears to be less effective compared to aleatoric uncertainty. However, using a combined method mitigates the weaknesses of both types and enhances overall performance.

In some instances, EMVSNet exhibits fluctuations in precision, where the performance is inconsistent. This often occurs due to a small number of samples in specific regions, causing individual samples to have a significant impact on the overall graph.

ROC Curve

The ROC curves, illustrated in appendix 8.6 for each scene, demonstrate that aleatoric uncertainty prediction consistently yields the best results. As with the PR curves, increasing the threshold T^e improves model performance.

Although the 2 mm threshold excludes nearly 50% of predictions, the model still performs significantly better than random guessing. At higher thresholds, the Area under the curve (AUC) approaches 0.9, indicating excellent performance.

The ROC curves further illustrate the model's strong ability to identify erroneous depth estimates. Once again, epistemic uncertainty prediction underperforms compared to the aleatoric type.

The steep rise in the ROC curve, achieving a high True Positive Rate (TPR) with a low False Positive Rate (FPR), suggests that an optimal threshold can be found to accurately identify regions with high error. This behavior is particularly notable at 4 mm and 6 mm T^e thresholds.

Conclusion

The comparison with the underlying AA-RMVSNet model reveals comparable results, with EMVSNet even surpassing its performance in direct comparison. Further experiments would be beneficial at this stage, particularly considering that the more complex network structure of EMVSNet may contribute to its enhanced performance. However, to fully explore this potential, it would be necessary to experiment with different disparity levels D.

Both, the Heatmap and the Calibration plot, demonstrate a correlation between measured error and expected uncertainty. The Heatmap reveals a broader variation in values, which aligns with theoretical expectations since the correlation between uncertainty and error is of a statistical nature. Higher uncertainty indicates a higher likelihood of error, but it does not guarantee high error for every individual sample. Thus, the Correlation plot provides a reliable measure of average behavior across many samples, and it shows excellent correlation.

The PR and ROC curves further highlight how the classification of error and uncertainty can be used to filter out unreliable predictions. With F1-Scores exceeding 80%, and often reaching 90% for higher error thresholds, the model demonstrates excellent performance. Although epistemic uncertainty prediction often lags behind aleatoric uncertainty, combining both approaches proves effective, enhancing performance by addressing cases with predominant epistemic uncertainty.

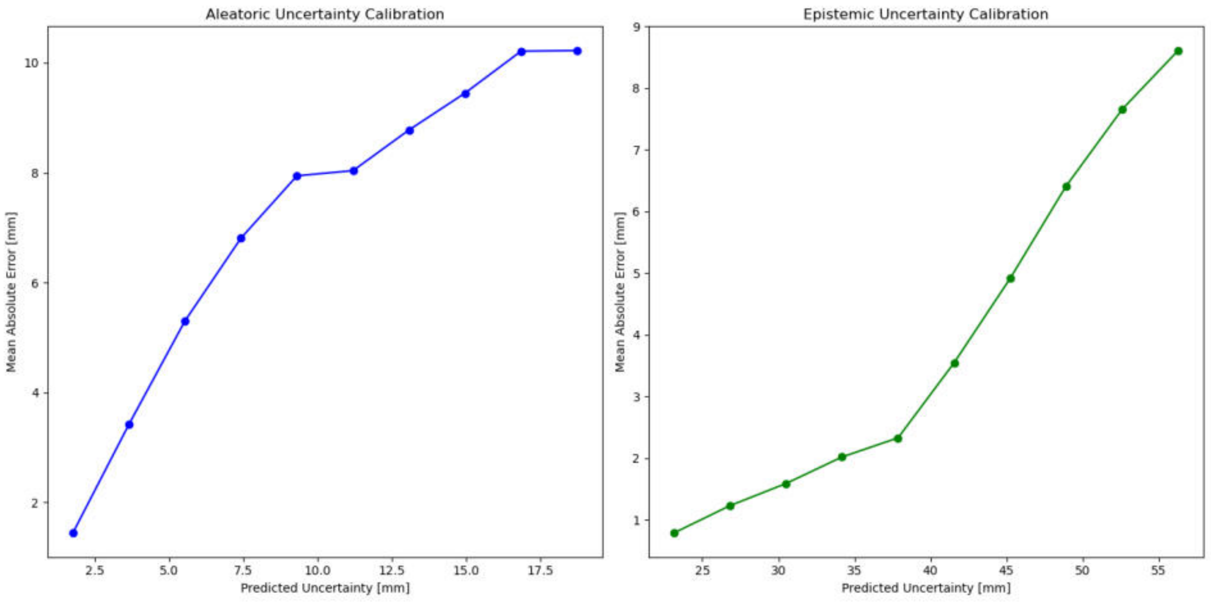


Figure 6.2: Calibration plot for aleatoric and epistemic uncertainty: To remove outliers, only values within the 2nd to 98th percentiles were used.

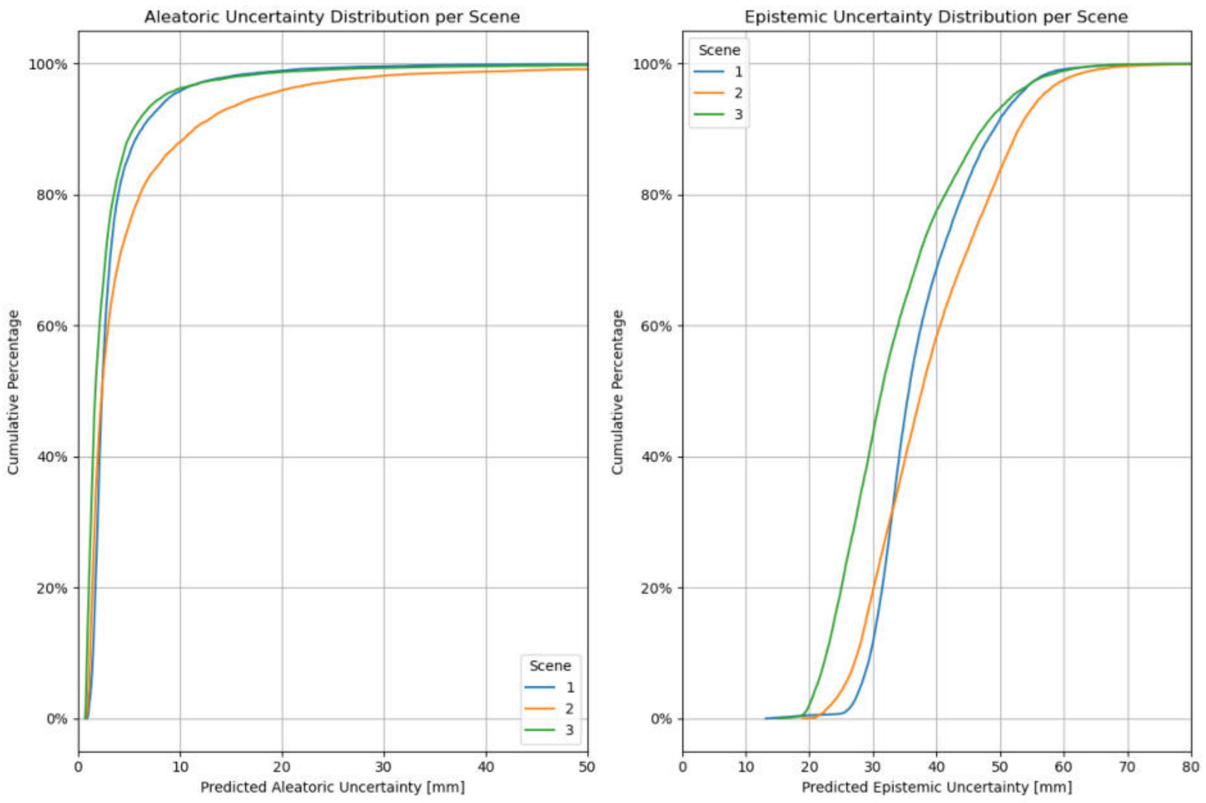


Figure 6.3: Distribution of predicted uncertainty values for all three scenes.

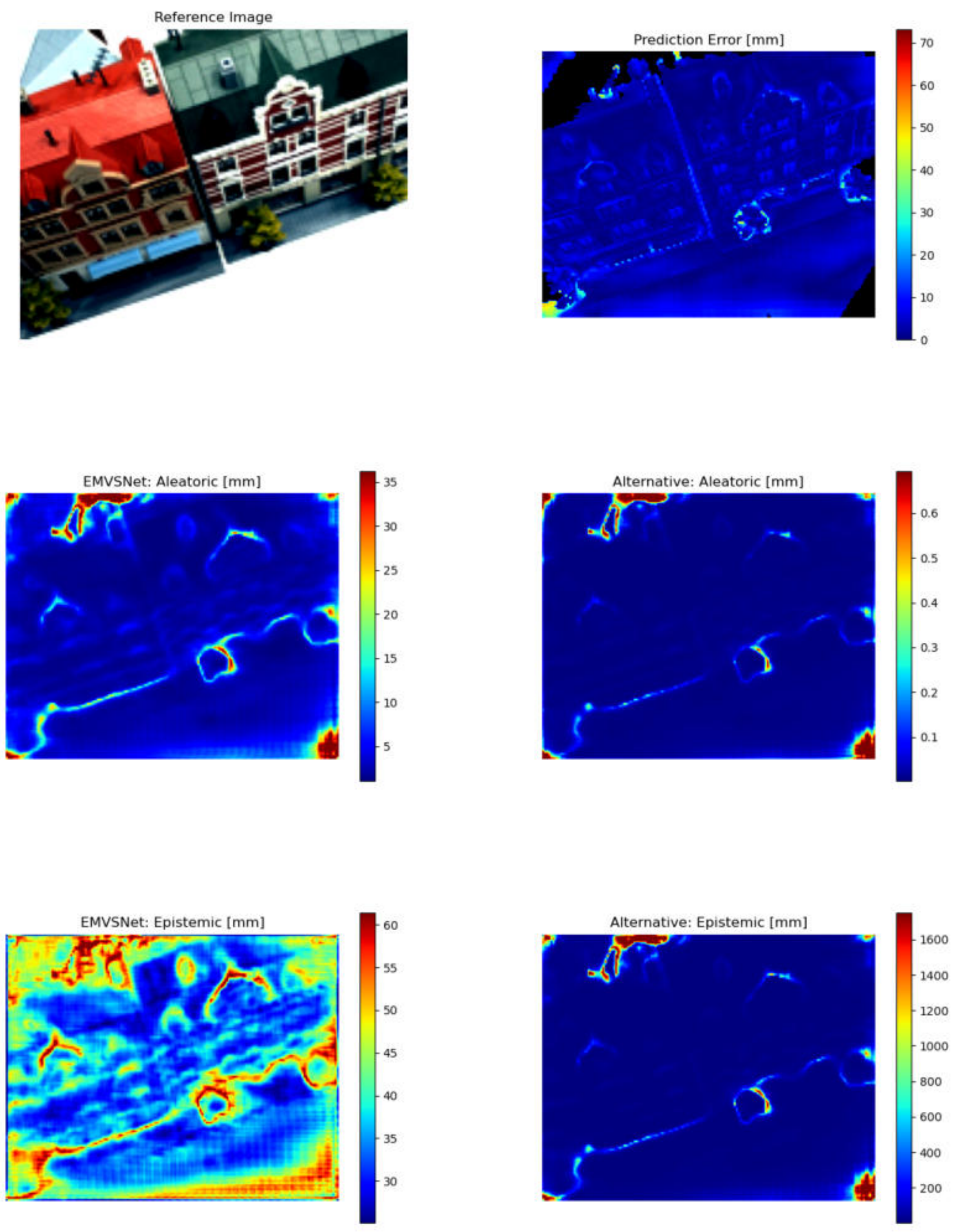


Figure 6.4: Visualizations for Scan 15 include the reference image, prediction error, and aleatoric and epistemic uncertainties with 1% clipping. The relative uncertainty measurements use a color scale, with deep red representing the highest uncertainty and deep blue the lowest, tailored to the specific method. The prediction error is similarly scaled, reflecting actual errors in millimeters. In the Prediction Error visualization, black regions indicate masked areas without ground truth, where error calculation isn't possible.

6.2 Visual evaluation

In addition to the qualitative results, the qualitative analysis is another major point that should not be overlooked. Visually identifying reasons for parts of the image that exhibit high uncertainty of either type might explain certain shortcomings found in the quantitative analysis. Additionally, it can be checked if parts of the image that theoretically should tend to higher uncertainty, for example non lambertian surfaces, really show higher uncertainty predictions.

The results presented for three scenes in figures 6.4-6.6 show uncertainty for EMVSNet, separated for the aleatoric and epistemic type.

General Analysis

Across all images, the outer borders where depth and material change can be clearly identified as regions with increased prediction error. Generally, the background is observed as the part of the image with the highest uncertainty. Since ground truth values for depth — and consequently for error - are only available for areas where objects are present, it is not possible to directly compare depth estimation in the background regions to uncertainty prediction.

However, the notable overlap between areas of high uncertainty and the masked background areas suggests a potential new metric for performance evaluation: whether the masked background areas could be predicted using uncertainty. Due to the limited scope of this thesis, this investigation is included in the outlook for future work.

Scan 15

In the first scene, it can be recognized that the change in depth at window projections leads to a higher error. The same can be said about the complex structure of bushes in front of the house. Especially pronounced, the tip of the bay window has high epistemic uncertainty. Remarkable is also the region in the top left corner, where some spots of high prediction error can be found. This area is also one of the few with masking applied and there higher uncertainty in the masked region can be found. The masking in the bottom right corner can be seen in all uncertainty predictions as well.

Overall, the aleatoric and epistemic uncertainty prediction do align well with measured error. The predicted epistemic uncertainty tends to more extreme values, so clipping has been performed in more areas.

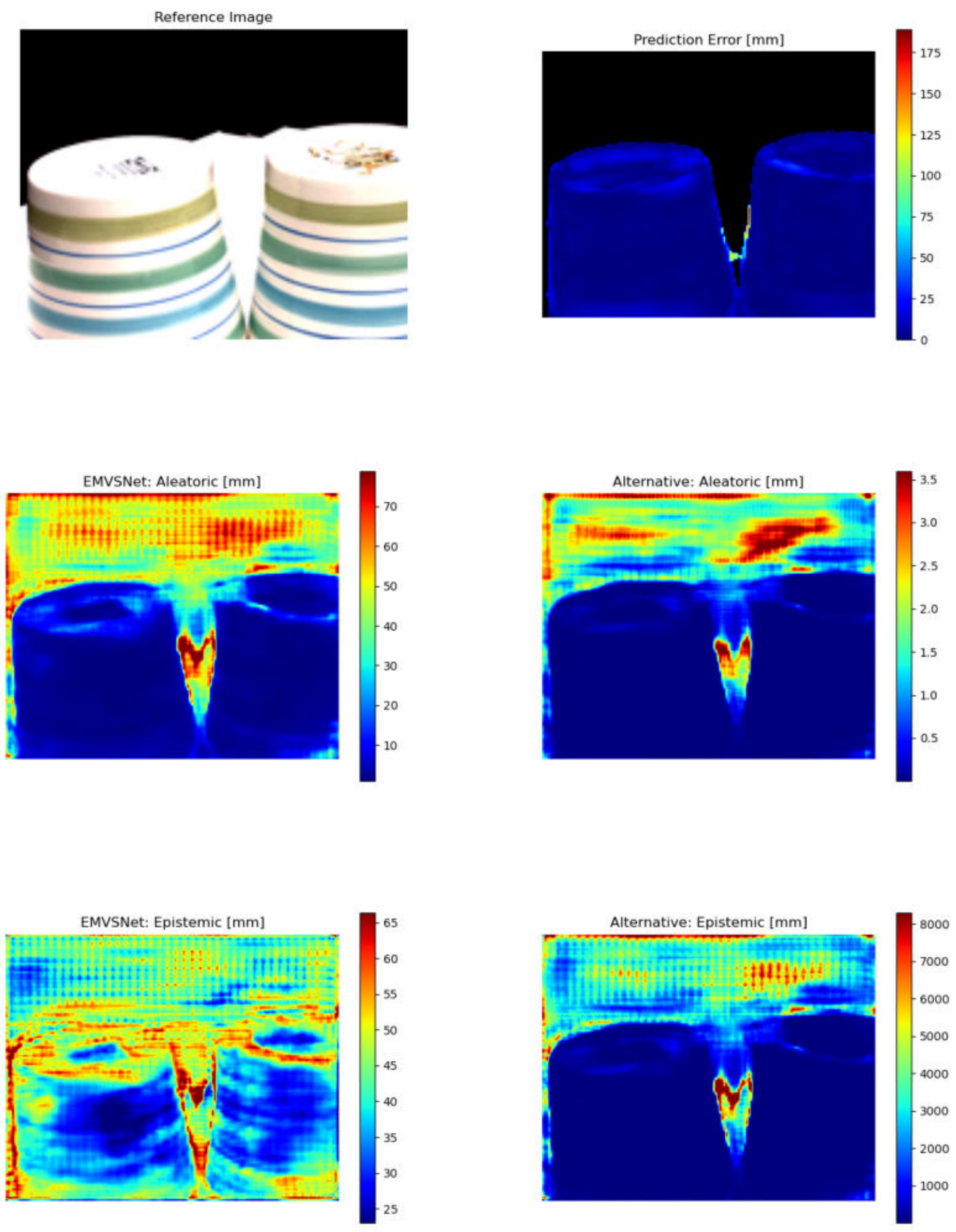


Figure 6.5: Visualizations for Scan 48 include the reference image, prediction error, and aleatoric and epistemic uncertainties with 1% clipping. The relative uncertainty measurements use a color scale, with deep red representing the highest uncertainty and deep blue the lowest, tailored to the specific method. The prediction error is similarly scaled, reflecting actual errors in millimeters. In the Prediction Error visualization, black regions indicate masked areas without ground truth, where error calculation isn't possible.

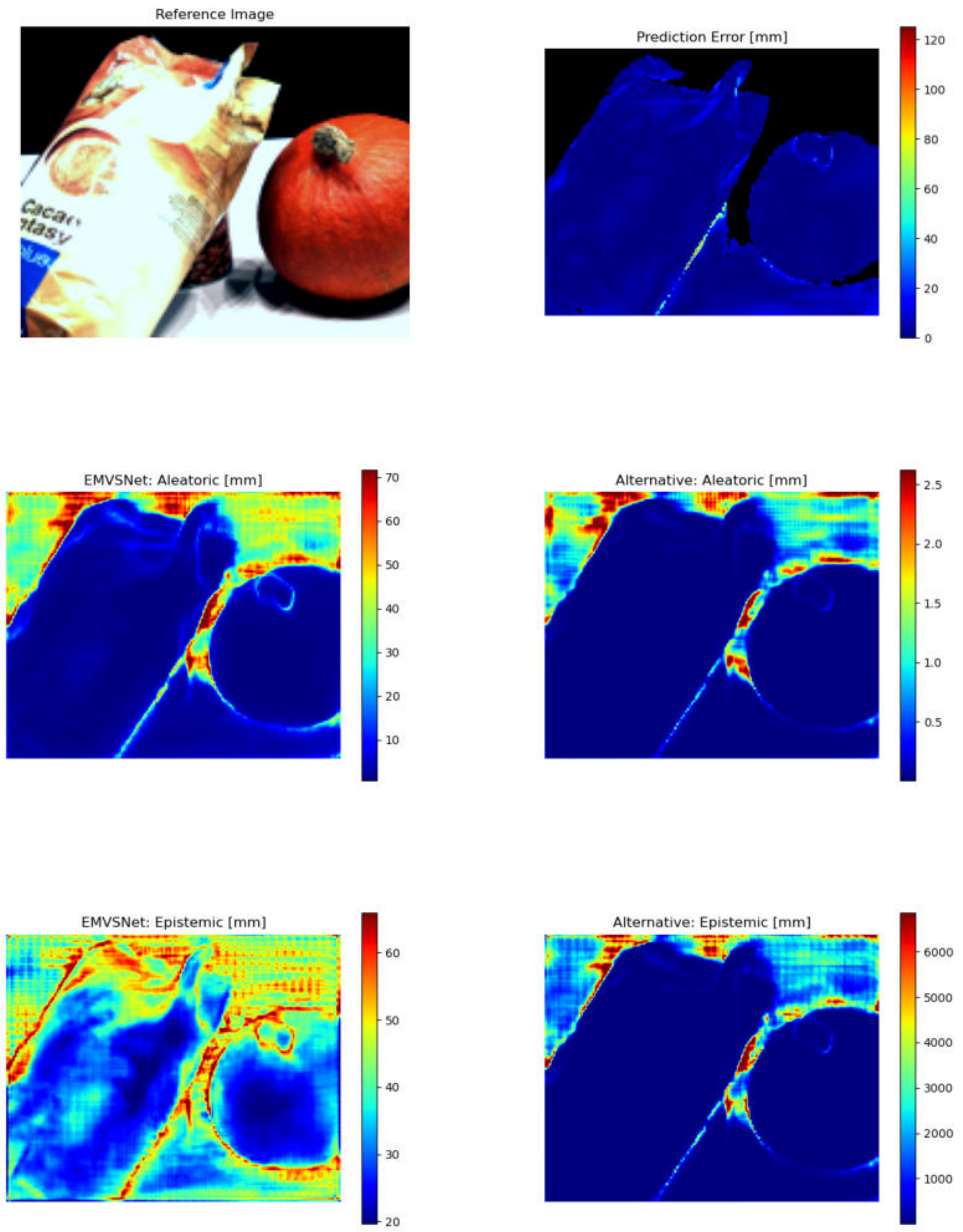


Figure 6.6: Visualizations for Scan 32 include the reference image, prediction error, and aleatoric and epistemic uncertainties with 1% clipping. The relative uncertainty measurements use a color scale, with deep red representing the highest uncertainty and deep blue the lowest, tailored to the specific method. The prediction error is similarly scaled, reflecting actual errors in millimeters. In the Prediction Error visualization, black regions indicate masked areas without ground truth, where error calculation isn't possible.

Scan 48

This scene again shows clear separation between the objects and background with epistemic uncertainty. In these images, a grid structure can be recognized. As strong suggestion this could be a phenomena known as checkerboard artifact. This can occur if up sampling is done using deconvolution. One strategy to overcome the issue would be to use bilinear interpolation followed by convolution, but as results are not to badly influenced here, this is reserved for the outlook.

Again, area of highest uncertainty matches well with the area of biggest error in case of aleatoric uncertainty where the area between the pots is most pronounced. What can further be seen in this example is higher uncertainty in the bright white areas on top of the pot due to unambiguouity and possibly non lambertian surfaces. In general, the epistemic type of uncertainty is much more pronounced in this example, although the ambiguity in white areas would indicate aleatoric uncertainty.

Scan 32

This scene again shows correlation of background and uncertainty as well as checkerboard artifacts. As area of high error the in part of the source images occluded part of the can behind the bag can be identified. For EMVSNet, variations in uncertainty prediction in the overexposed areas of the bag can be identified. These areas of increased uncertainty would be expected as there is no real visual information. Appendix 8.7 shows the images used for this scene. Potentially, increased uncertainty might also be introduced due to changes in lighting, but in this case, these is overexposure visible in all the frames. So most likely, the increased uncertainty is due to the visual ambiguity in this area.

With EMVSNet, in general edges can be identified as areas of higher uncertainty compared to those areas located on the object. This behaviour is to be expected as changing viewpoints especially influence the object's border by revealing or masking part of the object.

Interestingly, for the proposed scene 32 (appendix 8.7), it can clearly be identified that the increased movement of the bag in relation to the camera's position leads to higher uncertainty than the can hidden behind it not changing the position as much. It is also remarkable that the hidden can does not lead to higher uncertainty in this area, the right border of the can is visible in both aleatoric and epistemic plots but it could be anticipated that also part of the bag would be influenced by the changing features in this area. This marks as an indicator that the model was trained well on ignoring the features presented by a now hidden viewpoint.

The second image has the stripes printed on the vases slightly transferred to the predicted epistemic uncertainty. Why this might happen is an open question. What can nicely be identified is both, reduced uncertainty as well as reduced error on the printed bottom of the vases. Also, the over saturated area is exposed in both plots.

Including the visual abnormalities regarding background, masked areas and checkerboard artifacts, visual identification of areas with higher uncertainty seems to be well explained in almost all cases. As the theoretical consideration suggests, areas of uniformity, changing depth and background can be identified as high uncertain areas. Especially with the clear separation between background and object, real world application of this system for example for autonomous driving might be able to increase safety.

6.3 Amount of viewpoints

Incorporating more viewpoints on one hand increases computational power needed to process and combine the information. On the other hand, it might result in better results as more information is available and ambiguities from a certain perspective might be resolved through more information from another viewpoint. Not in every case anyhow must more information lead to better results - conflicting information from shadows or reflections could also result in decreased model performance.

Another interesting aspect is the given training procedure of the network described in chapter 5. Weights are trained using one amount of viewpoints, so altering them might influence the interpretation of the combined Cost Volume from individual features extracted. Although a shared backbone is used and combination of the results from each viewpoints makes it unlikely this will influence the network performance negatively, this is a fact worth checking.

For this test, one reference frame is kept while 1, 2, 4, 8 or 19 additional source frames are added. The prediction error and in each type of uncertainty values are plotted as percentile curves. This provides the opportunity to compare prediction accuracy as well as overall uncertainty for the different amounts of viewpoints.

Table 6.4 shows performance metrics for five experimental setups ranging from 2-20 viewpoints for EMVSNet. Best performance in this experiment is given when using 5 viewpoints, although the network was trained on less. In general, using uneven amounts of viewpoints enables the usage of equal amount of previous and following images in DTU dataset with the reference frame located in the middle.

In figures 6.7-6.9, for every amount of viewpoints precision and recall for all types of

Views	Test Loss	Time	MAE [mm] $\downarrow$	2mm $\downarrow$	4mm $\downarrow$	8mm $\downarrow$	16mm $\downarrow$	32mm $\downarrow$
2	8.14	0.181	11.73	55.6%	33.5%	18.9%	11.7%	7.4%
3	5.88	0.199	7.37	44.5%	23.4%	11.7%	6.5%	3.9%
5	5.54	0.218	6.79	41.2%	21.2%	10.3%	5.7%	3.6%
9	5.71	0.317	7.10	42.5%	21.4%	10.4%	5.9%	3.9%
20	5.90	0.368	7.45	44.3%	22.1%	10.8%	6.3%	4.2%

Table 6.4: Mean performance metrics for different amounts of viewpoints. The percentage values inside the right columns indicate how many depth estimations exceed an error of the given range in millimeter.

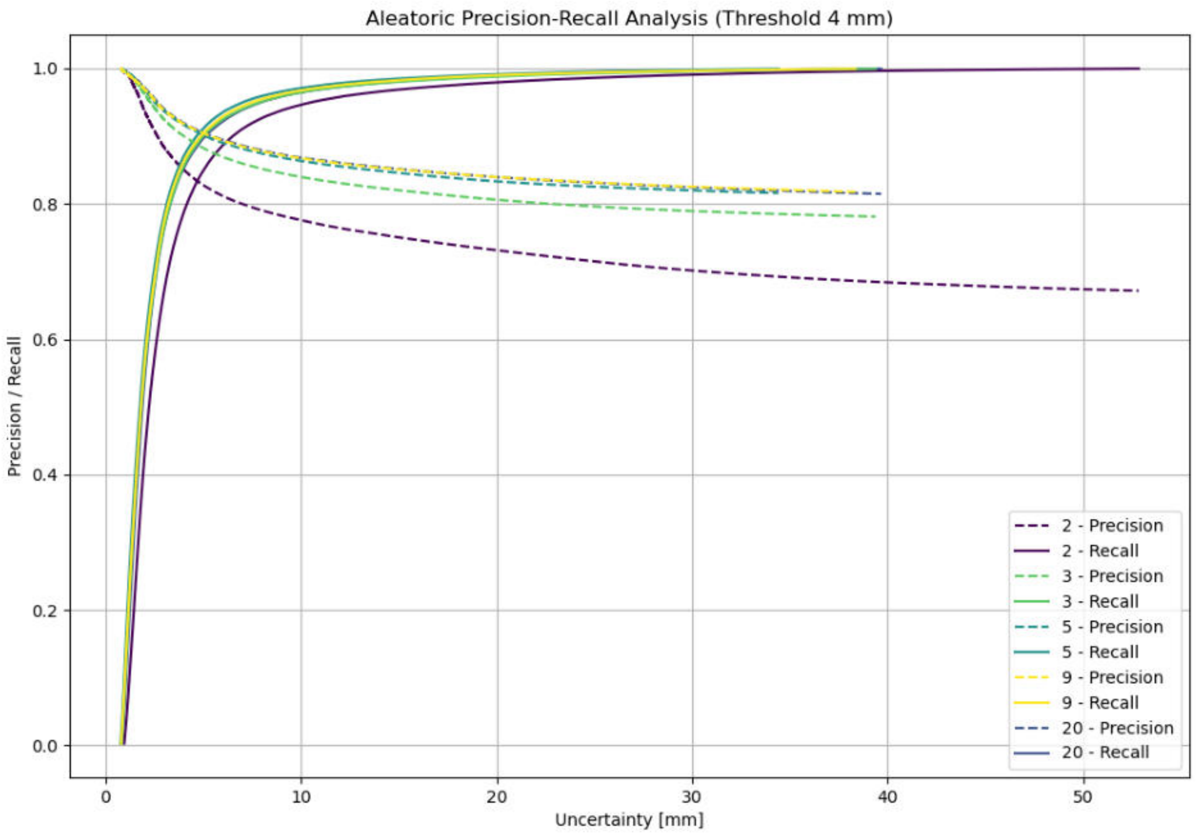


Figure 6.7: Comparison of Precision-Recall curves for aleatoric uncertainty. Different amounts of viewing directions used are shown, the error threshold T^e has been set to 4mm.

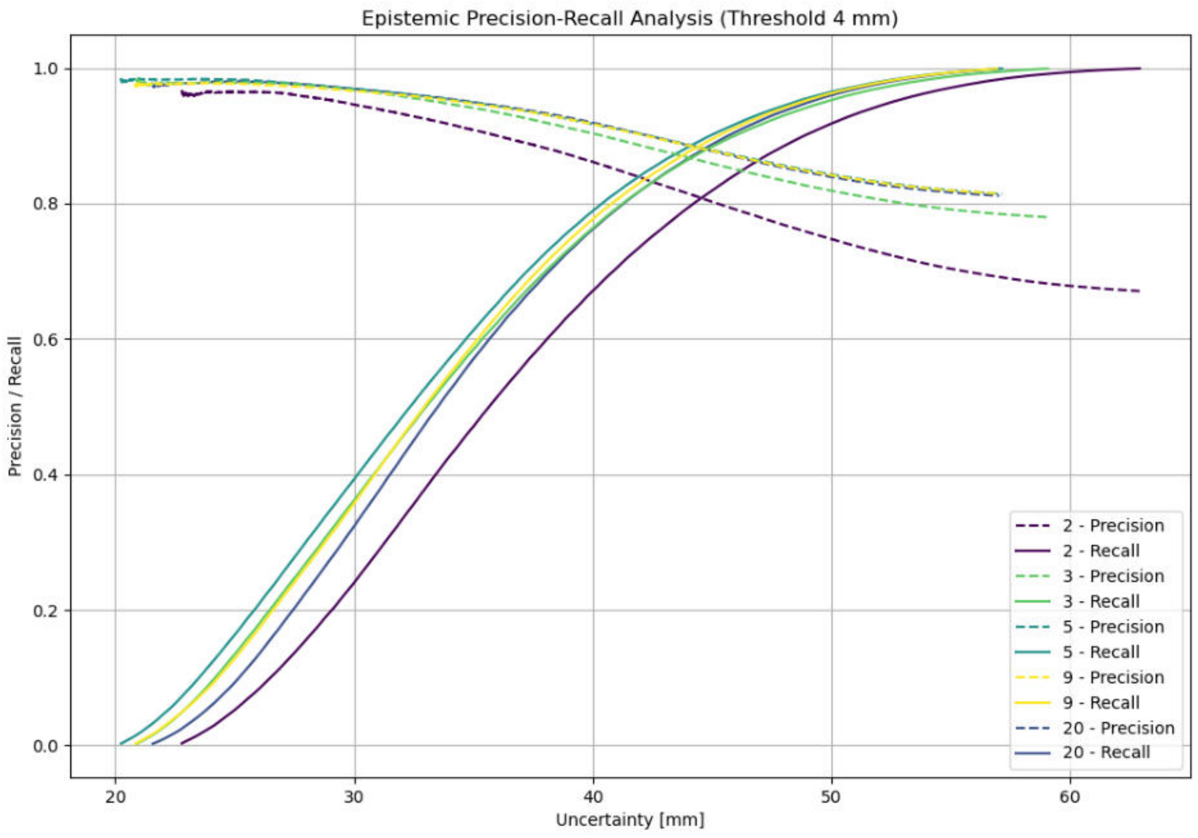


Figure 6.8: Comparison of Precision-Recall curves for epistemic uncertainty. Different amounts of viewing directions used are shown, the error threshold T^e has been set to 4mm.

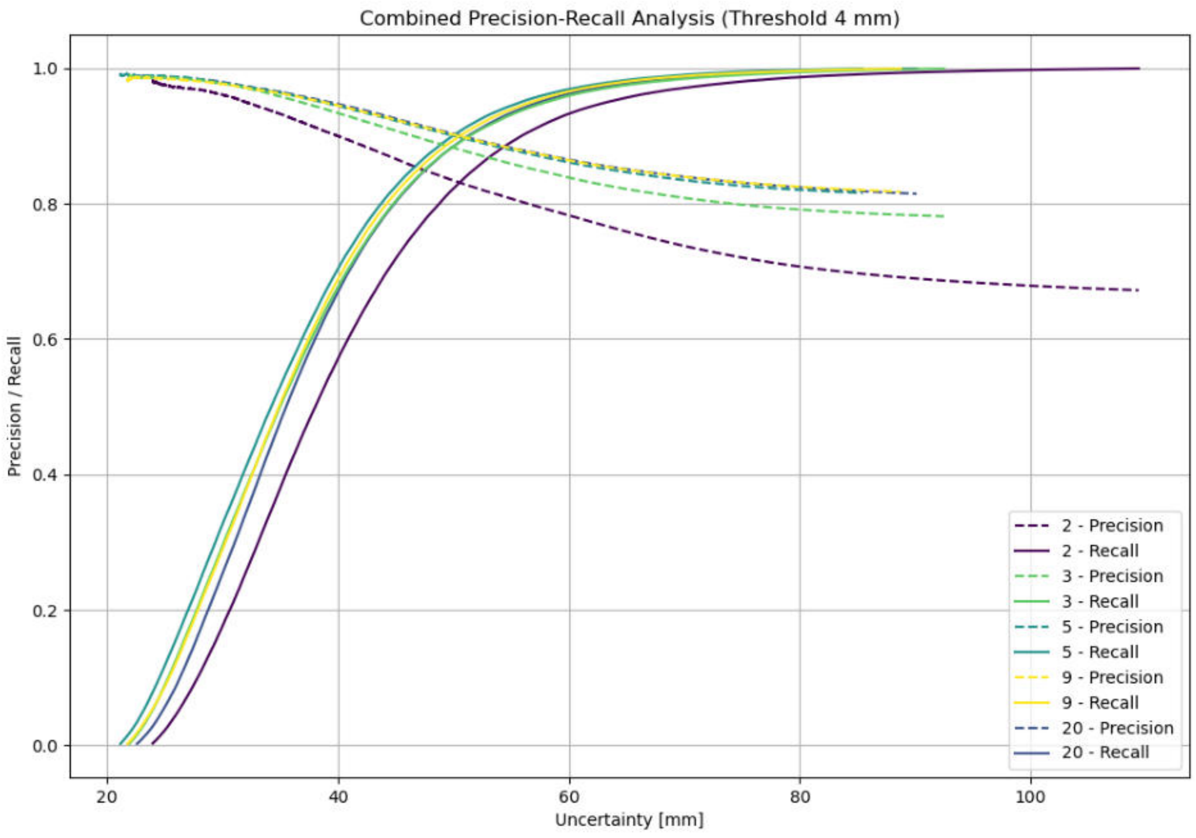


Figure 6.9: Comparison of Precision-Recall curves for the combined aleatoric and epistemic uncertainty. Different amounts of viewing directions used are shown, the error threshold T^e has been set to 4mm.

uncertainty including combined uncertainty is visualized. In this case a fixed, balanced threshold for the error of 4 mm is used. As direct performance comparison it can be recognized that more viewpoints lead to better performance with only marginal improvements after inclusion of 5 viewpoints. In case of the aleatoric uncertainty its also interesting that the chosen 90 percent of uncertainty values stretch to much higher values than in the other cases.

Overall, more viewpoints seem to improve depth and uncertainty estimation. It can be seen that even though the network was trained on fewer viewpoints, performance increases with increased amount.

6.4 Inter-dataset performance

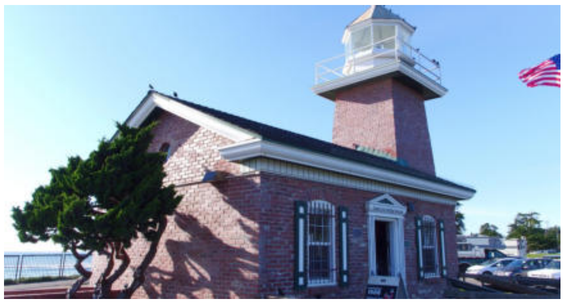
The practice of utilizing models trained on a specific dataset, or type of data, and applying them to another dataset with a significant domain gap often presents challenges. This is particularly pronounced in applications where domain-specific features significantly influence model performance. To investigate the general ability of a model not only to perform depth estimation, but also to handle uncertainty in measurements, the model trained on the DTU dataset is applied to test scenes from TnT. The test scenes selected from TnT for this purpose are:

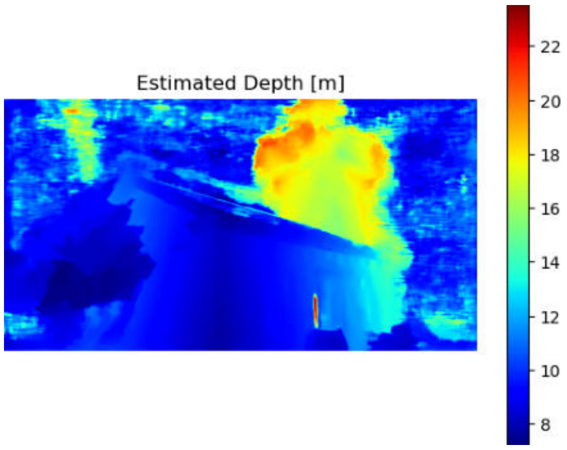
- Lighthouse
- M60
- Panther

Utilizing weights optimized for the DTU dataset in applications involving TnT data provides a valuable opportunity to assess the generalizability of the trained model across different domains. Notably, the DTU dataset predominantly features indoor, artificially illuminated scenes with small objects, whereas the TnT dataset comprises larger, outdoor scenes that include buildings and tanks, illustrating a substantial domain shift.

Theoretical Considerations

Theoretically, one might anticipate increased epistemic uncertainty when a model trained on one domain is applied to another distinctly different one. This arises due to the model's lack of knowledge about new domain features, which were not present during training. Conversely, aleatoric uncertainty, which is inherent to the observational data regardless of the model, might be lower in the TnT dataset. This is because the challenging lighting conditions often found in the DTU dataset are absent, and the TnT dataset benefits from richer textural information and more diverse surface interactions. Additionally, the different scales and object complexities between the two datasets might further exacerbate the epistemic uncertainty. For example, models trained on small, detailed objects might struggle with the scale and simplicity of large structures or vice versa.





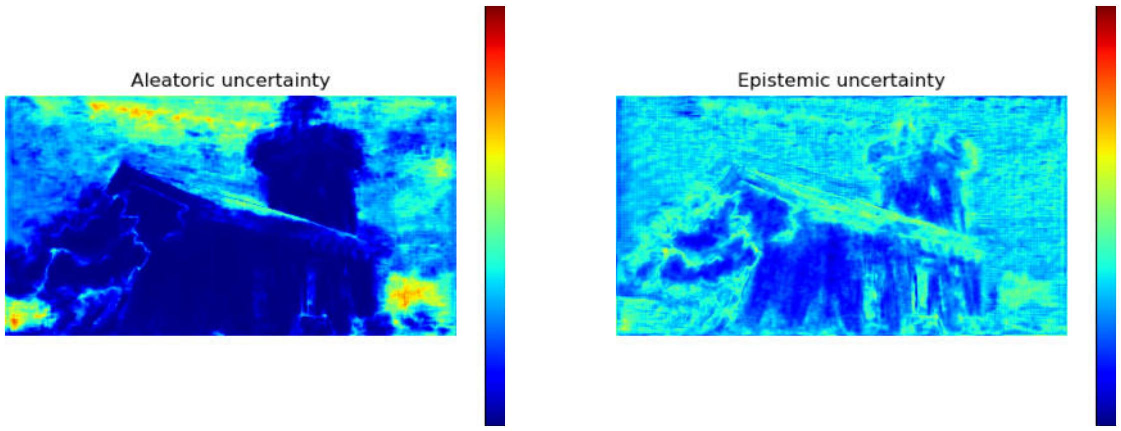
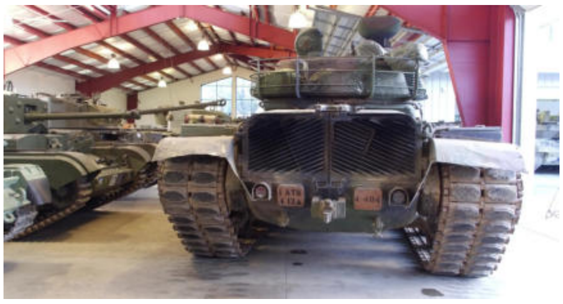
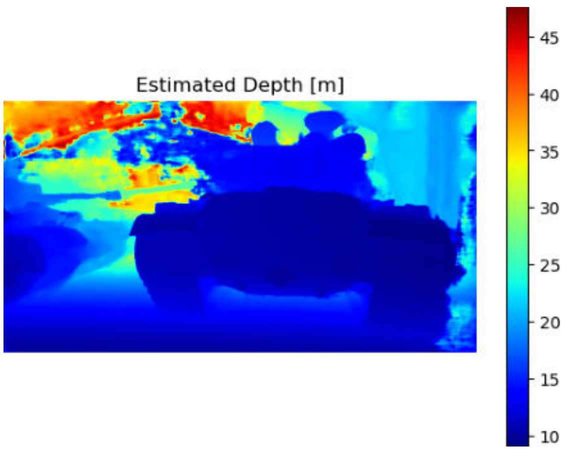


Figure 6.10: Visualization for the Lighthouse scene (image 48) from the TnT dataset. The reference image, estimated depth, and plots depicting relative areas of aleatoric and epistemic uncertainty are shown. Red represents high uncertainty, blue indicates low uncertainty.





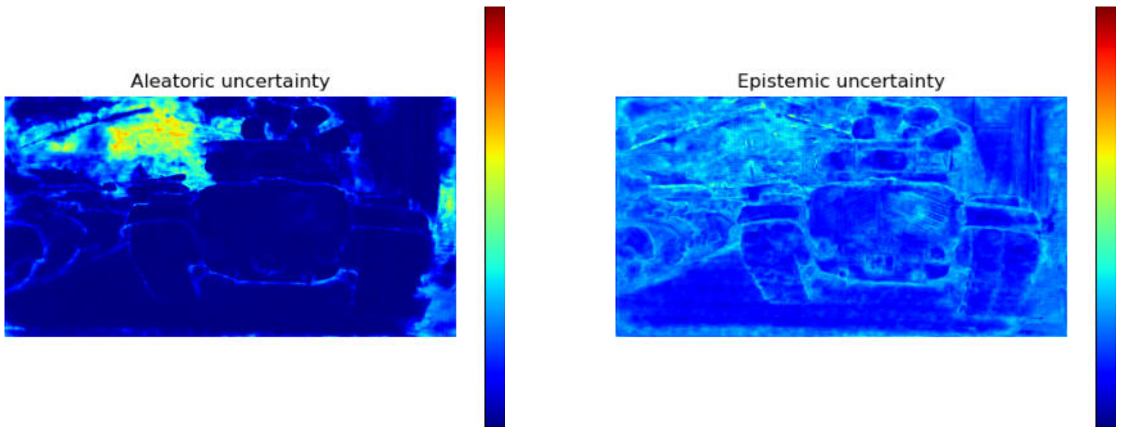


Figure 6.11: Visualization for the M60 scene (image 4) from the TnT dataset. The reference image, estimated depth, and plots depicting relative areas of aleatoric and epistemic uncertainty are shown. Red represents high uncertainty, blue indicates low uncertainty.

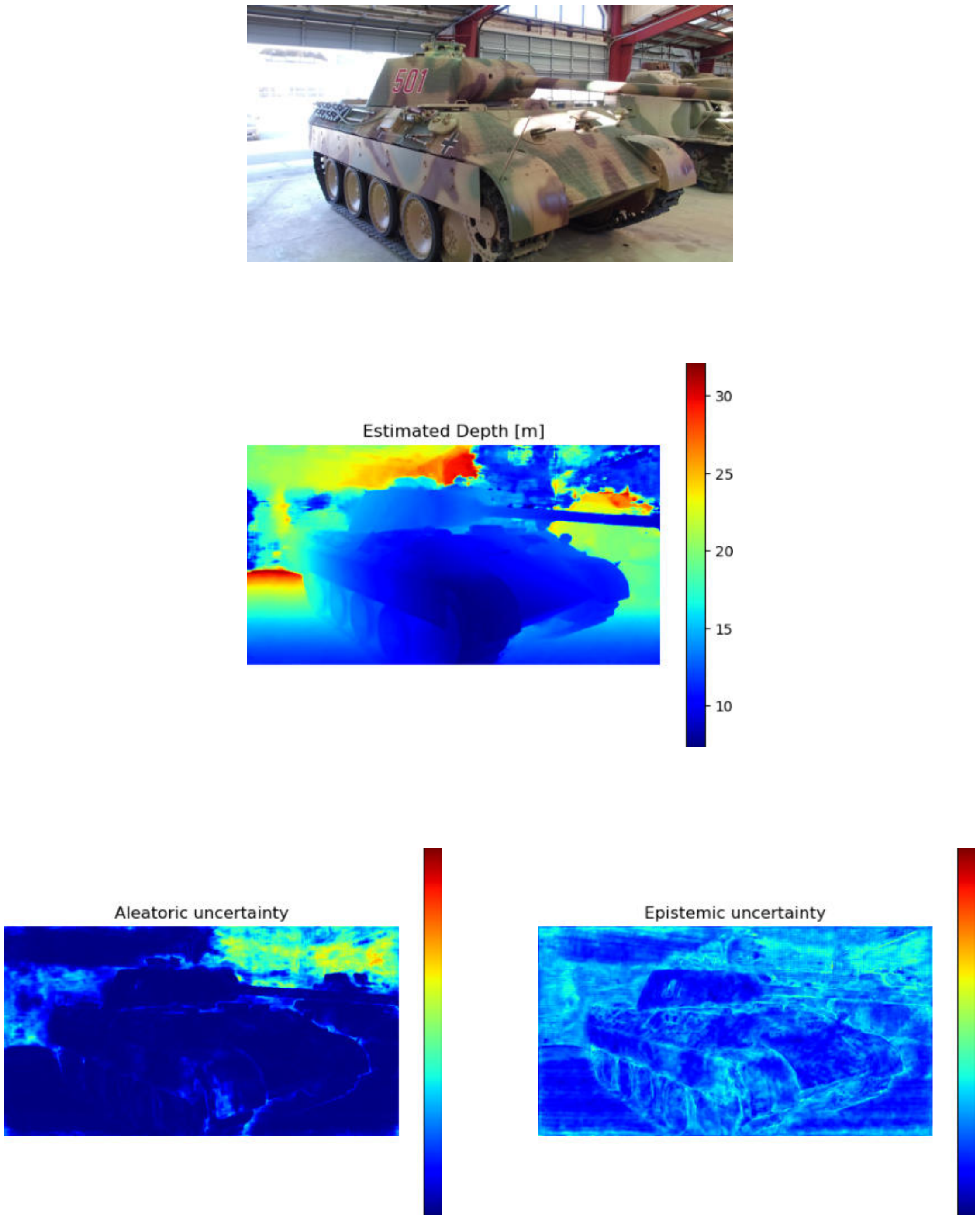


Figure 6.12: Visualization for the Panther scene (image 27) from the TnT dataset. The reference image, estimated depth, and plots depicting relative areas of aleatoric and epistemic uncertainty are shown. Red represents high uncertainty, blue indicates low uncertainty.

Specific experiments

The performance of the proposed neural network on this dataset is evaluated using these metrics:

- Visually comparing estimated depth and uncertainty
- Mean uncertainty

To proof the increased epistemic uncertainty due to the domain shift, mean uncertainty for three scenes from each dataset is compared. As the TnT dataset does not provide masks to separate the objects in question from the background, the comparison is done using the whole scene. This is reasonable as it can be expected that also background information in DTU is more common and known than the outdoor scenery marking the background in TnT.

Testing the inter-dataset performance based on data from the TnT dataset is done using the following settings:

- **Viewpoints:** 5
- **Depth levels:** 32
- **Resolution:** 544×1024

According to the experiment in section 6.3, the chosen number of viewpoints was already very good performing while limiting the need for computational resources. The increased resolution of the TnT dataset leads to increased memory consumption and without labour intensive fine-tuning of GPU memory, allocation of 24 GB Video Random Access Memory (VRAM) is only sufficient for 5 simultaneous viewpoints. Analyzing the influence of the number of viewpoints on inter-dataset performance has been left for future work at this point.

Visual comparison

Figures 6.10-6.12 display three scenes with the predicted depth and associated uncertainty. Due to the lack of integration of ground truth data in this work, quantitative analysis of the peculiarities mentioned here is not available. But truly remarkable is the bad depth prediction performance in regions where sky is visible with depth predictions on the complete false end of the spectrum. These problems especially occur in the *Lighthouse* scene widely spread, and this scene also is the one where aleatoric uncertainty prediction worst fits to the obviously visible error. In the *M60* and *Panther* scene the areas of increased aleatoric uncertainty align remarkably well with areas of high error. Even though the first scene has the sky as problematic area, visually the object should be extractable from the background using the aleatoric uncertainty estimation.

The predicted epistemic uncertainty visually almost presents itself like an edge detection algorithm. Increased uncertainty at object's edges on the other hand is plausible, so uncertainty prediction seem to align to problematic areas.

While depth prediction remarkably struggles especially with the outdoor scene in this test, uncertainty prediction seem to be able to identify the areas of increased error although there is the domain shift happening. While, again, perfect pixel matching is not given, broader areas of increased uncertainty do align with high error and should be able to identify erroneous regions.

Mean epistemic uncertainty

Table 6.5 shows the mean predicted uncertainty values for the three tested TnT scenes. As calculation of uncertainty is done using the probability distribution over the depth level bins and absolute values are extracted later using the depth rage, these values are directly comparable against results from DTU dataset.

Scene	Aleatoric	Epistemic
Lighthouse (TnT)	35.1	45.6
M60 (TnT)	22.9	44.3
Panther (TnT)	21.8	43.9
Scene 1 (DTU)	3.44	37.65
Scene 2 (DTU)	5.10	39.20
Scene 3 (DTU)	2.97	33.41

Table 6.5: Mean values of aleatoric and epistemic uncertainties for different scenes. Lighthouse, M60, and Panther are from the TnT dataset, while Scene 1, Scene 2, and Scene 3 are from the DTU dataset. All values are given on the scale of the DTU dataset and are comparable against each other.

Compared to the predicted uncertainty values for the three scenes from the DTU dataset, using the three TnT scenes results in 20% higher mean of epistemic uncertainty values. As discussed before, this is expected behaviour and shows the shift in dataset.

Unexpectedly the values of aleatoric uncertainty are an order of magnitude higher with the OOD dataset, which is not to be expected by itself. One explanation might be the truly visible difference in appearance of scenes in TnT. The broader field of view with big objects includes small details not found in the DTU dataset. Here, in general objects are more uniform and visual features seem not to change that much.

Further investigation into this behaviour might be necessary to explore the source of this behaviour. One way to do so could be the analysis of activation inside the evidential part of the network to see which parts of the image are essential for prediction of aleatoric uncertainty with the domain-shifted dataset. Another factor might be the inclusion of background, so using a OOD dataset with proper masks might reveal problems in that regard.

Strategies for Enhancing Model Adaptability

There are multiple ways to further refine and improve the comparison process only scatched at this point. Comparing the performance on high-detail scenes like 'Lighthouse' against the less textured scenes such as 'Panther' could reveal how well the model handles different levels of surface complexity and lighting conditions. Additionally, introducing perturbations such as artificial noise or varying light conditions in the test scenes could help in understanding the robustness of the model against real-world environmental variations.

Further testing could involve cross-dataset validation where the model is not only tested on TnT but also on other datasets that represent different challenges, such as varying degrees of motion blur or different weather conditions. This would help in identifying the specific limitations and strengths of the model across a broader spectrum of scenarios. Moreover, it would be worthwhile to consider the impact of training data diversity—integrating data from multiple sources during training could potentially enhance model robustness. The exploration of transfer learning and fine-tuning the model with minimal data from the target domain (such as TnT) could also provide insights into the practicality of deploying such a model in dynamic real-world applications where the domain conditions are continually changing.

6.5 Comparison of formulas

The experiments conducted in this section are designed to show differences between both formulae 4.3 and 4.6. These experiments should provide the opportunity to account for theoretical differences of these approaches in real world example and to show which is better suited, with respect to the specific task, to describe the types of uncertainty.

The difference in mathematical representation of uncertainty and the critique of Meinert et al. [22] about the original paper provide an interesting opportunity to compare both approaches. Specifically, as already stated in the paper, for many cases the actual performance of uncertainty prediction is alright using the approach of Amini et al. [21].

Uncertainty for three scenes

To measure the effects of various input data as well as the feasibility of the proposed network structure, the uncertainty predictions for three scenes for each dataset are analyzed. This is done as comparison between both formulas by:

- Comparing the mean of both aleatoric and epistemic uncertainty for three scenes.
- Comparing the relative difference in uncertainty prediction between three scenes.
- Comparing the predicted uncertainties as density plot over the distribution of values.
- Visualize all of the expected uncertainty for three scenes as heatmap showing how often a certain range of uncertainty is predicted for a certain error.

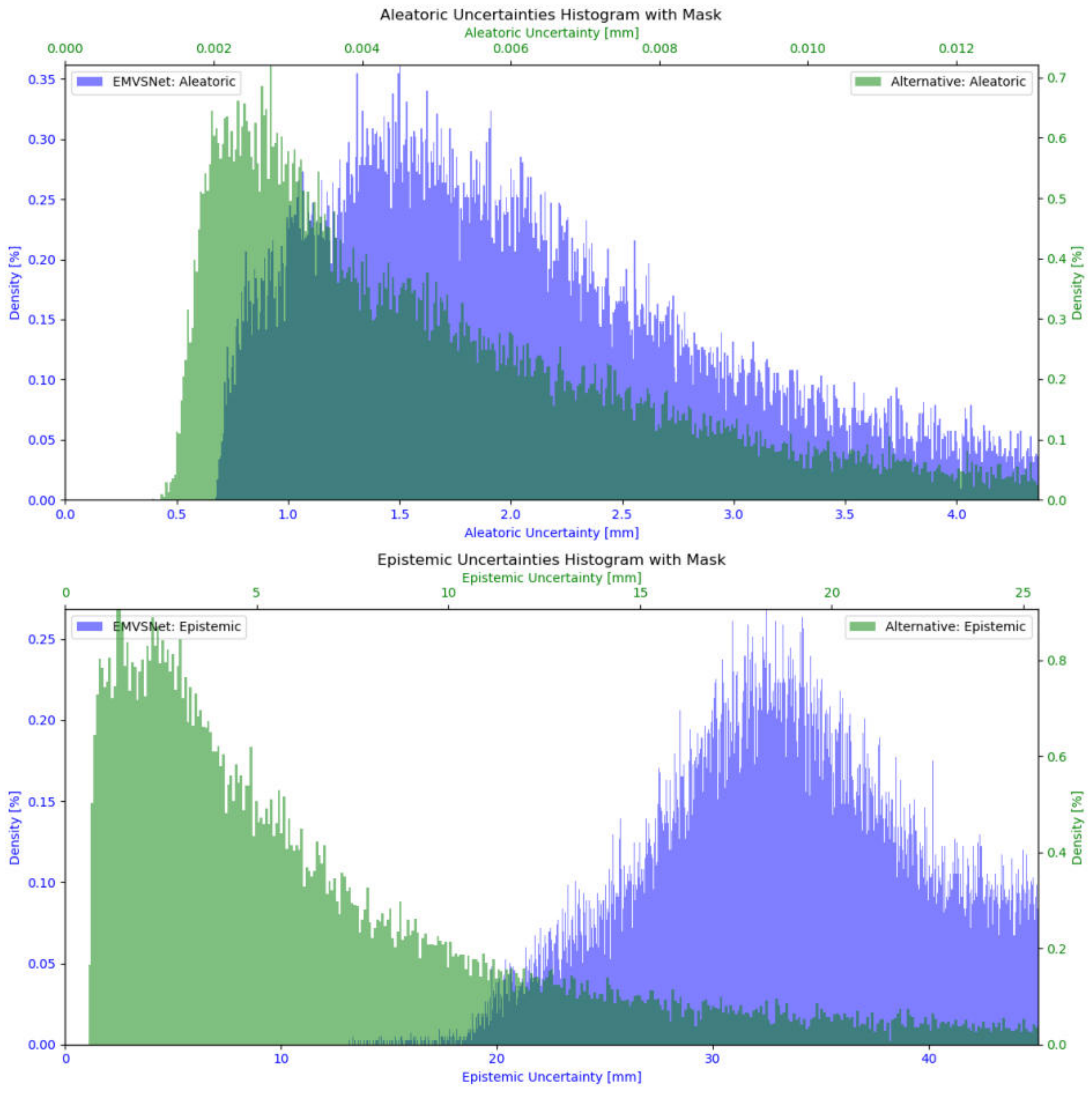
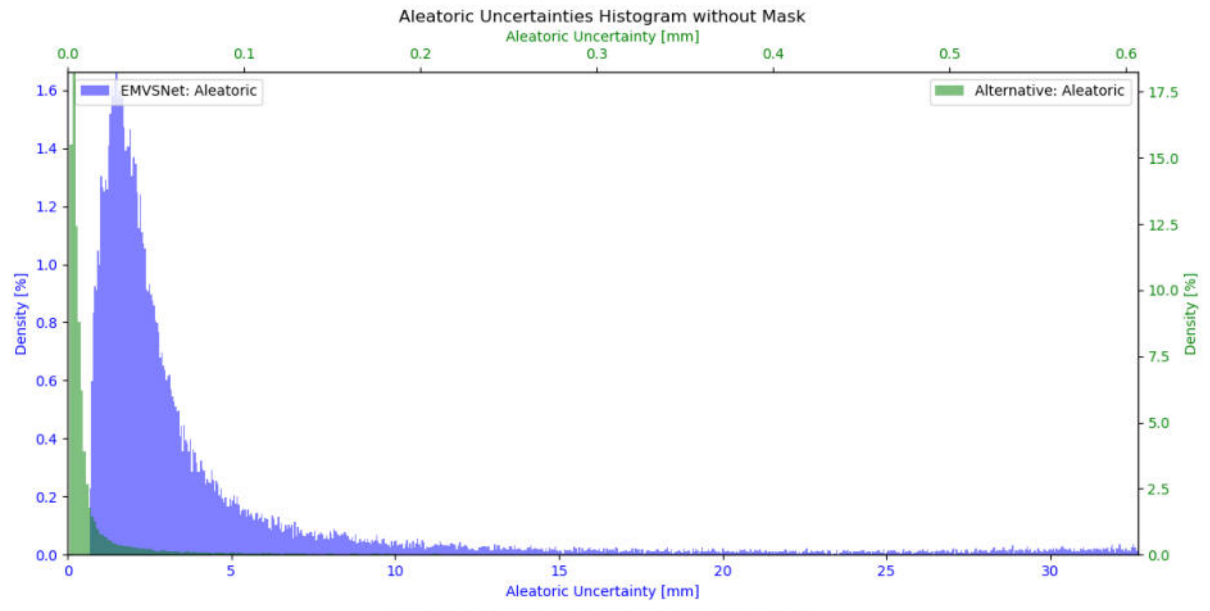


Figure 6.13: Density function of uncertainty values for the 3 scenes distinguishing between the two formulas with masking applied. Top 10 percent outliers are removed.



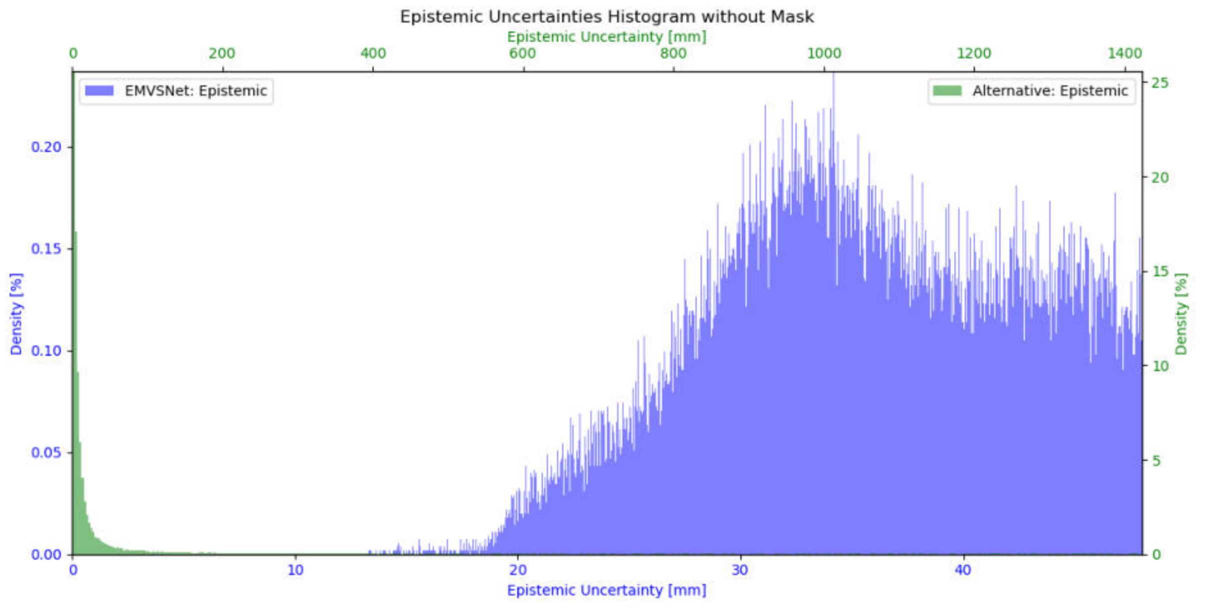


Figure 6.14: Density function of uncertainty values for the 3 scenes distinguishing between the two formulas without masking. Top 10 percent outliers are removed.

The comparison of mean uncertainty is meant to show differences between the two formulas in absolute measured values. By including the relative differences into the investigation, even with difference in total predicted value it might be shown that overall trend follows the same pattern. The density plot should show the distribution of uncertainty prediction, to show a theoretical wider spread in values which are not represented in mean values. In the context of this experiment, the mentioned error is seen as difference between real ground truth depth d^g and predicted depth d^p. The heatmap shows the relation of real error to expected uncertainty. As, in general, higher uncertainty leads to higher error, this heatmap should show a relation between these.

Mean uncertainty

Figure 6.15 shows the proposed comparison between EMVSNet and the Alternative formula as mean uncertainty values for the three scenes. In this plot already, a stark mismatch between aleatoric and epistemic uncertainty values for the Alternative can be spotted. Up to factor 2000 between aleatoric and epistemic uncertainty is much higher than the factor 10 that can be seen with EMVSNet. This is the observation that Meinert et al. [22] made as well. With the new formulation of uncertainty and Loss, the distribution between both uncertainty types is much more reasonable.

Relative uncertainty

The *unreasonable effectiveness* of the method proposed by Amini et al. [21] can to a degree already be spotted in this plot. Although there is this strong mismatch in aleatoric and epistemic uncertainty, if compared between scenes like in figure 6.16, the relative difference is much more reasonable and in line with results from EMVSNet.

Density plot

Figures 6.13 and 6.14 show a histogram of uncertainty values with and without the background included. Especially for the case with masking applied, EMVSNet seems to have a more even distribution across different uncertainty predictions in comparison to the Alternative method. It is also noteworthy that the Alternative has wider spread values. For the case without masking, values for EMVSNet stay reasonable while especially for epistemic uncertainty, for the Alternative everything gets concentrated at the lower spectrum of values.

Heatmap

Figure 6.1 illustrates the the bespoken histogram for both, EMVSNet and the Alternative. In this plot again, the wider range of values for the Alternative while mainly concentrating on the lower end of the spectrum is clearly visible. Also figures 6.4 to 6.6 show an enormous missmatch between aleatoric and epistemic uncetainty values

Conclusion

The experiments conducted in this section underscore the distinct differences between the two approaches to uncertainty quantification. The Alternative set of formulae, as critiqued by Meinert et al. [22], exhibits a significant imbalance between aleatoric and epistemic uncertainties, with the latter often being orders of magnitude greater. This stark discrepancy suggests that the Alternative method might be theoretically less robust compared to EMVSNet, which maintains a more balanced and reasonable distribution between these types of uncertainties.

However, despite this apparent theoretical flaw, the Alternative method demonstrates a paradoxical effectiveness in practical applications. The PR and ROC curves, discussed in Appendices 8.5 and 8.6, reveal that the Alternative method performs on par with, and in some cases even surpasses, the performance of EMVSNet. This phenomenon can be described as *unreasonably effective*, a term that echoes the observations made by Amini et al. [21]. Despite the model's theoretical inconsistencies, it remains highly capable of identifying areas of high uncertainty, which is crucial in many practical scenarios.

This paradox highlights that while EMVSNet offers a more theoretically consistent approach, the Alternative method's empirical effectiveness cannot be overlooked. Specifically, in scenarios where high uncertainty detection is critical, the Alternative method proves to be a surprisingly strong contender, regardless of its theoretical limitations. Therefore, the Alternative method can be seen as a practically viable, and in some cases superior, approach for uncertainty quantification, demonstrating that the practical effectiveness observed by Amini et al. [21] can indeed be considered *unreasonably effective*.

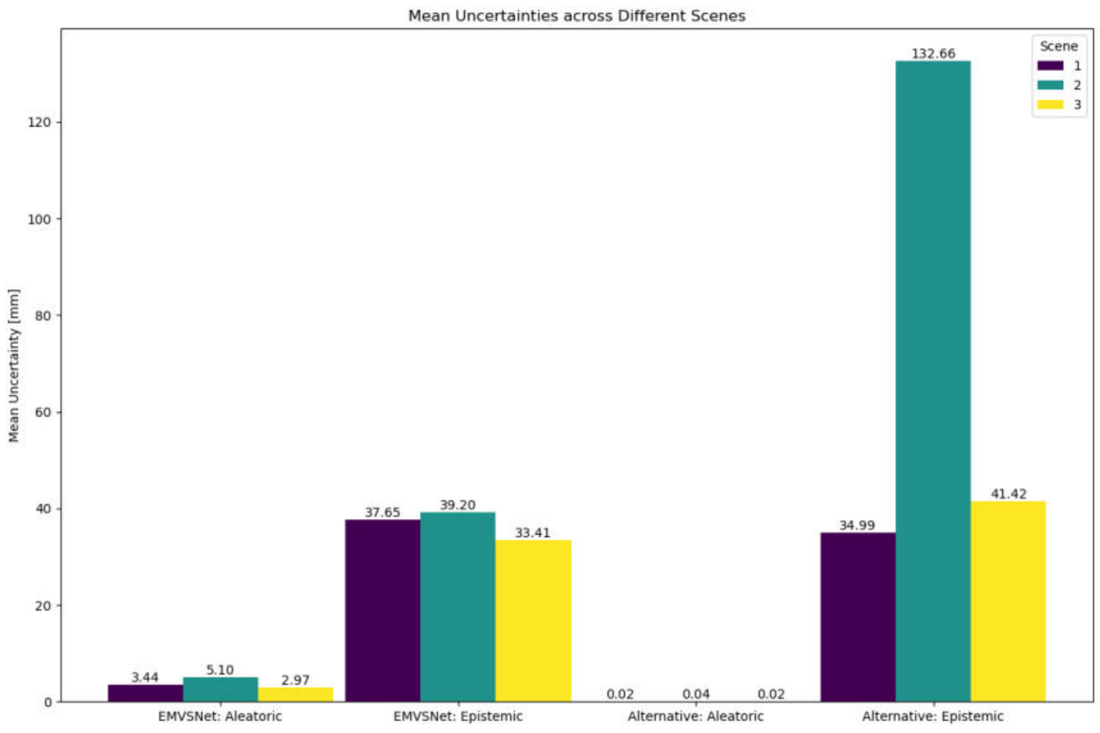


Figure 6.15: Absolute difference in uncertainty levels across the three scenes, given in millimeters.

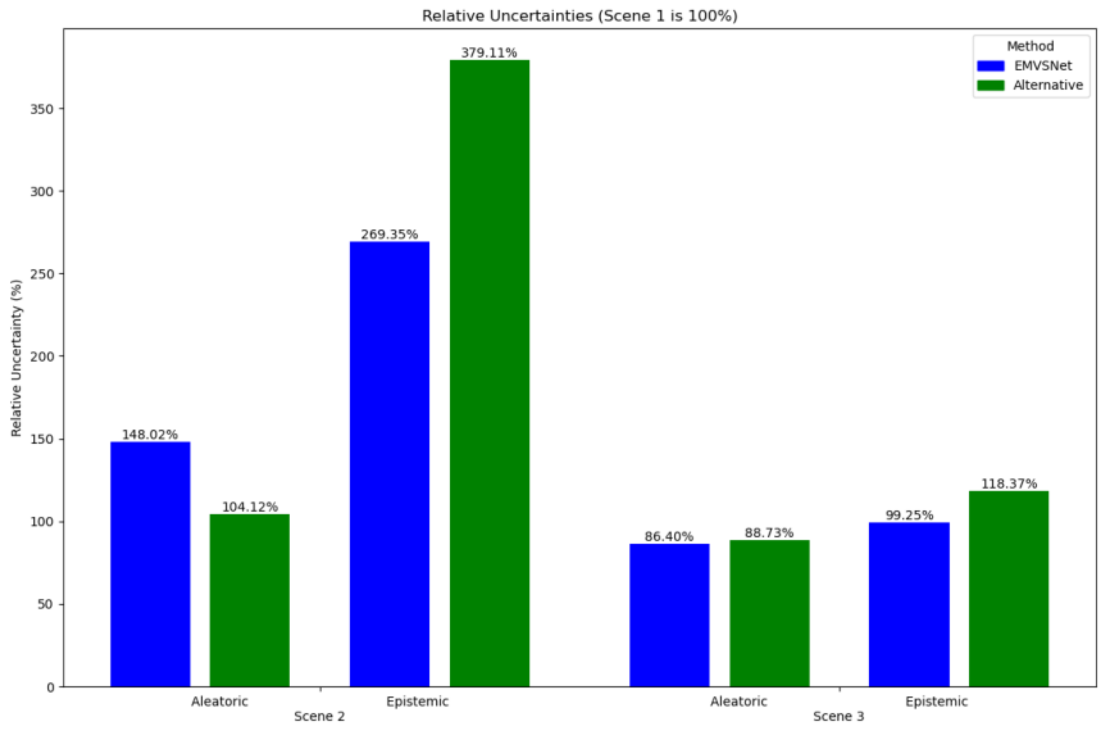


Figure 6.16: Relative distribution of uncertainty values across three scenes, normalized to 100% for scene 1.

6.6 Monte Carlo Dropout

The comparison between EDL and MCD was conducted on the DTU test set, focusing specifically on scene 4. Figure 6.17 presents a representative result from this scene.

The error maps generated by both methods generally align well, with EDL showing a slightly higher error overall. Both methods effectively highlight areas of increased uncertainty, particularly along the boundaries where depth changes occur, which correspond closely to the actual error distribution. However, it was necessary to apply a 2% clipping to the values due to outliers in the MCD variance, which significantly exceeded the typical range observed in the data.

This behavior is consistent with previous observations when using 10 samples for MCD. Although theoretically, increasing the number of samples should mitigate this effect, it remains evident in this scenario. The predictions for aleatoric and epistemic uncertainty tend to be more concentrated within a narrower range, indicating a more stable uncertainty estimation, albeit with some extreme variances.

Method	Mean Time [s]
EDL	0.190
MCD	5.436

Table 6.6: Mean processing time taken from 98 batches for EDL and MCD. For MCD, 30 samples are used.

Table 6.6 illustrates the processing time required for each method, highlighting the substantial advantage of EDL in terms of computational efficiency. The runtime for MCD is approximately 28 times longer than that of EDL, with the increase in processing time closely correlating with the number of samples used.

The implementation of only two dropout layers at a late stage of the network, combined with high dropout rates, resulted in reasonably accurate predictions. Although there are several opportunities to refine this approach, such as integrating dropout into more parts of the network, adjusting dropout rates, and adding regularization layers, the chosen setup serves as a solid baseline. Future work could explore these optimizations to enhance the performance and efficiency of MCD further.

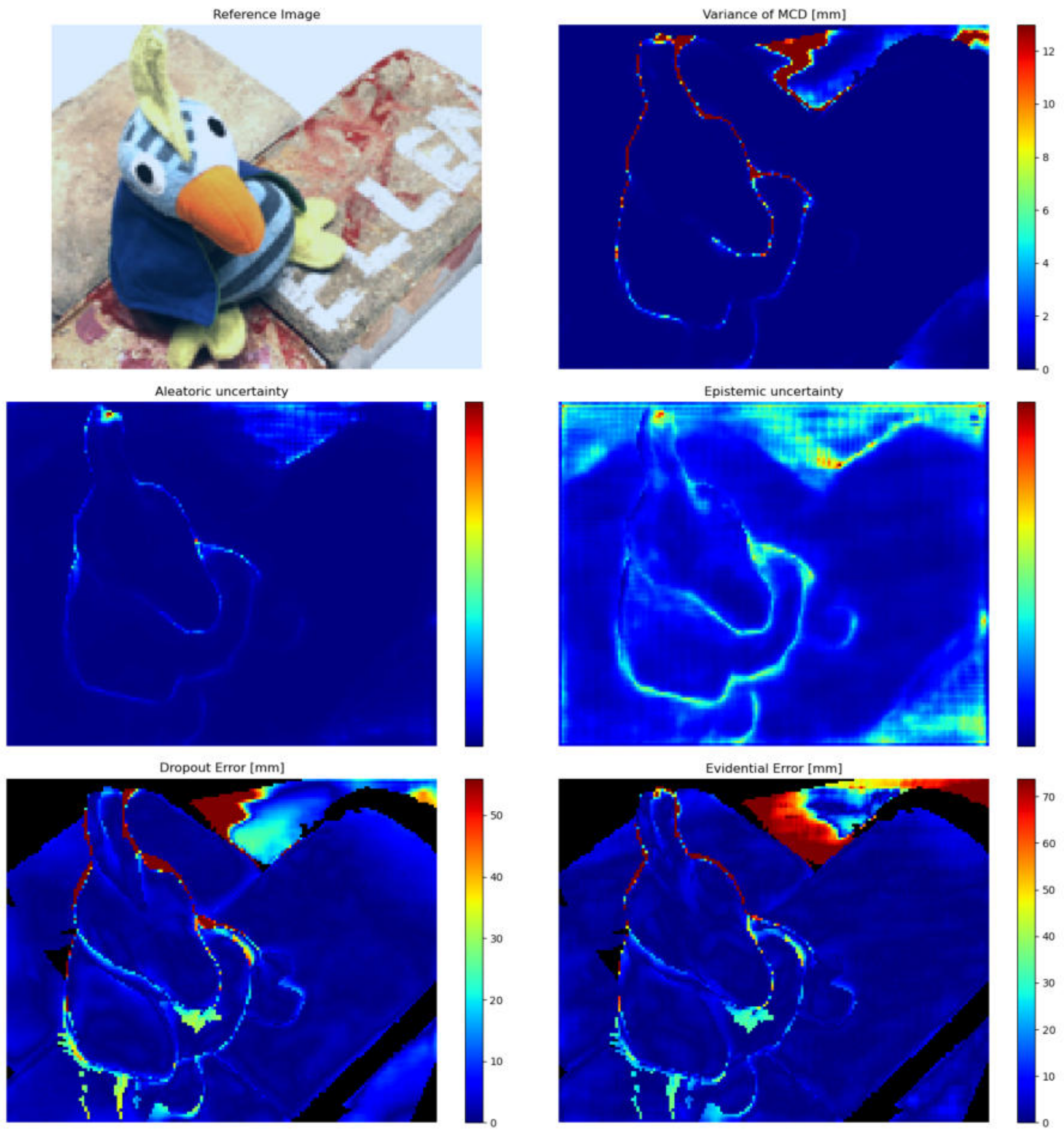


Figure 6.17: Comparison of prediction results from EDL and MCD for DTU scene 4. To improve visibility, 2% clipping has been performed for all plots.

6.7 Two-Stage Network

The two-stage approach with a modified activation function, as introduced in Section 4.3.2, presents an intriguing method to potentially overcome the limitations encountered in previous experiments. This approach was explored in the course of this work, and the results are illustrated in Figure 6.18.

While the specific details of the network architecture and configuration are not the focus here, this experiment serves to highlight alternative possibilities for integrating EDL into MVS. The example demonstrates both potential and ongoing challenges that need to be addressed when employing this two-stage approach.

Firstly, it is important to note that the depth estimation performance remains robust. Since the underlying network architecture was used without modification, with uncertainty prediction added as a secondary component, the depth estimation results are identical to those of the original AA-RMVSNet implementation.

However, the standard deviation of the depth hypothesis reveals a pattern suggestive of quantization effects, which also manifests in the uncertainty predictions, though with a different visual signature. Additionally, the estimated uncertainty range appears relatively narrow, indicating that while the approach is effective, it may be constrained by the current network configuration.

In conclusion, while this two-stage approach shows promise for enhancing both depth and uncertainty prediction, further research and development are necessary. Specifically, a network architecture that is explicitly designed and optimized for this dual-task approach may be required to fully realize the potential benefits and address the identified challenges.

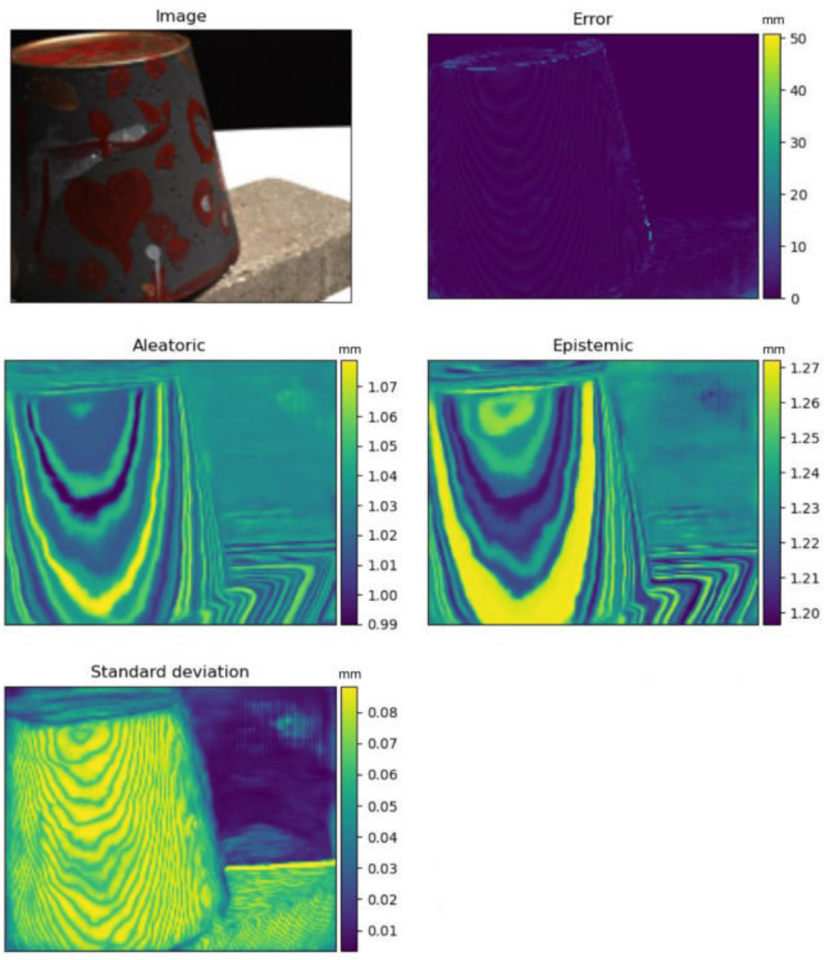


Figure 6.18: Results of the two-stage training approach, displaying the input image, depth estimation error, aleatoric and epistemic uncertainty maps and the standard deviation of the predicted depth.

7 Conclusion and Outlook

This work represents a foundational step in integrating Evidential Deep Learning (EDL) into Multi-View Stereo (MVS) systems, potentially broadening the applicability of MVS methods in real-world scenarios. It has been demonstrated that by extending the established network for depth prediction, it is possible to enhance predictions with uncertainty values in a relatively resource-efficient manner. The conducted experiments showed excellent performance on the training dataset, with the model achieving high precision in identifying areas with significant errors. Due to the lack of ground truth data for uncertainty, only a stochastically inspired comparison was possible. Future evaluations of potential datasets that include ground truth uncertainty data would be highly beneficial. Visually, the predicted uncertainties were well-explained and aligned with expected outcomes. In this context, checkerboard artifacts were observed, which could potentially be mitigated by modifying the network architecture, such as by switching to bilinear interpolation. Based on the results, an intriguing idea emerged: using the predicted uncertainty values not only for their intended application but also for masking, as they provide effective object-background separation, at least in cases of relatively uniform background.

Additionally, the network's performance was evaluated with varying numbers of viewpoints. This investigation did not reveal a dramatic change in performance; however, the training foundation remained unchanged. In the future, it may be worthwhile to adjust the number of viewpoints used in training to optimize the setup.

An anomaly was observed in the aleatoric uncertainty during Out of Distribution (OOD) tests on the Tanks and Temples (TnT) dataset, warranting further investigation. Otherwise, the domain transfer experiments lacked a ground truth comparison, which would be beneficial for examining the performance on OOD data. Regarding the TnT dataset, a comparison of the complete reconstruction results while incorporating the predicted uncertainty would be highly valuable. As this thesis represents an initial approach to integrating EDL into an MVS pipeline, the complete 3D scene reconstruction was not addressed, but the chosen approach and observed results suggest interesting possibilities for filtering or weighting depth predictions.

This work adapted a modified set of equations to express aleatoric and epistemic uncertainty, as well as an alternative loss function. Peculiarities and different behaviors were reported here, as in the original paper, so it would be interesting to further investigate why and in which cases these mathematical concepts are beneficial. The two-stage network approach was not particularly successful, and this approach requires further investigation and refinement.

The comparison of EDL to Monte Carlo Dropout (MCD) yielded similar results and demonstrated the applicability of Evidential Multi-view Stereo Network (EMVSNet). However, the investigation of this concept was relatively brief. Integrating MCD into the proposed method could be an intriguing combination to further refine uncertainty predictions, provided that computational efficiency and runtime are not critical factors. Conversely, if computational efficiency and runtime are paramount, further investigation and optimization of the network's efficiency would be essential.

In addition to these results, experiments demonstrated comparable performance to Adaptive Aggregation Recurrent Multi-view Stereo Network (AA-RMVSNet) in the task of depth estimation. Investigating whether incorporating uncertainty predictions could further enhance depth prediction performance would be advantageous. However, it is noteworthy that in the cases examined, the model's performance already surpasses that of the underlying architecture. Nonetheless, expanding training and inference to include more views and disparity levels would be beneficial for further evaluation.

Final statement

The reconstruction of 3D scenes is likely to become increasingly important in our daily lives. Consider the Apple Vision Pro, capable of capturing 3D videos and projecting three-dimensional information into our world. As technology evolves, the demand for more immersive and accurate 3D experiences will only grow, influencing industries ranging from entertainment to architecture, education, and beyond. Given the economic inefficiency of 3D sensors in many applications and the much more cost-effective alternative of using cameras, MVS methods will be essential for reconstructing 3D data. As these methods become more refined, they will play a critical role in shaping how we interact with digital content in the real world, enabling more seamless and intuitive user experiences.

While EDL has proven to be an efficient concept with good performance in terms of MVS, the results generated in this thesis suggest that its application might be of interest in other areas where neural networks are employed as well. The ability to quantify uncertainty with EDL could enhance decision-making processes in various fields, such as medical imaging, autonomous driving, and robotics, where understanding the confidence of a model's predictions is crucial.

In summary, while this thesis has made significant strides in enhancing the reliability of MVS through the integration of EDL, it also highlights the complexity and challenges inherent in accurately predicting and managing uncertainty in 3D reconstruction. This work paves the way for future innovations that will further bridge the gap between theoretical advancements and practical, real-world applications, ensuring that as technology progresses, so too does our confidence in its outcomes.

8 Appendix

8.1 Hour Glass network

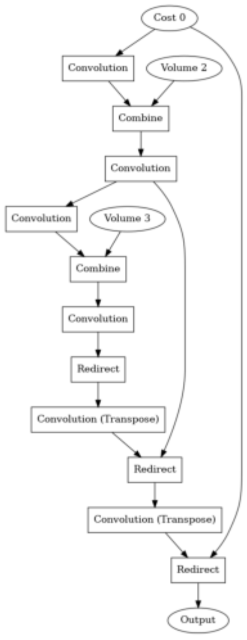


Figure 8.1: Hours Glass upward path of the network.

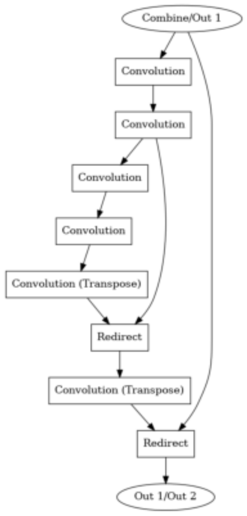


Figure 8.2: Hours Glass downward path of the network.

8.2 Evidential network structure

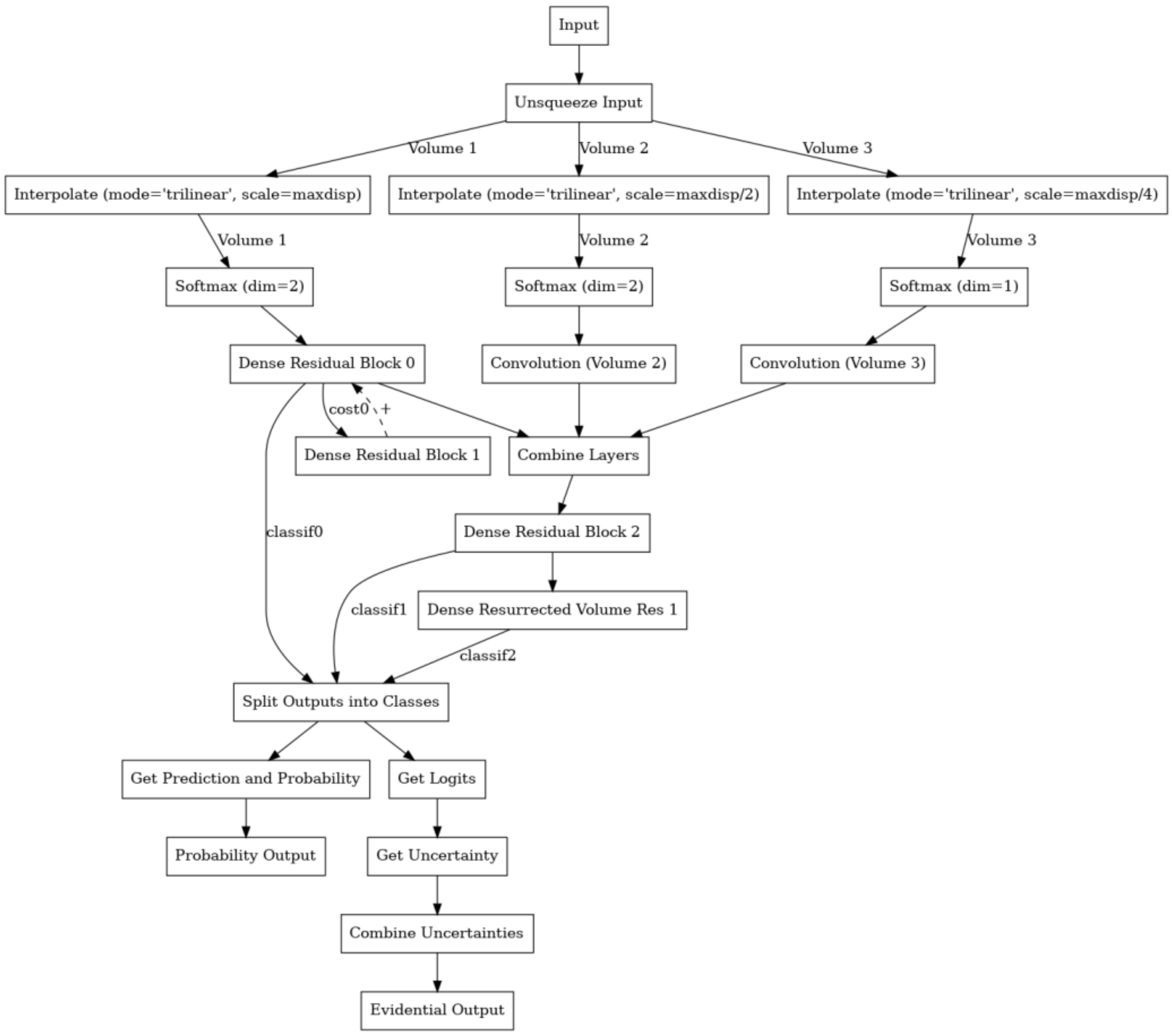


Figure 8.3: Overview of evidential part of the proposed network.

8.3 DTU scene selection

Training

- scan2
- scan6
- scan7
- scan8
- scan14
- scan16
- scan18
- scan19
- scan20
- scan22
- scan30
- scan31
- scan36
- scan39
- scan41
- scan42
- scan44
- scan45
- scan46
- scan47
- scan50
- scan51
- scan52
- scan53
- scan55
- scan57
- scan58
- scan60
- scan61
- scan63
- scan64
- scan65
- scan68
- scan69
- scan70
- scan71
- scan72
- scan74
- scan76
- scan83
- scan84
- scan85
- scan87
- scan88
- scan89
- scan90
- scan91
- scan92
- scan93
- scan94
- scan95
- scan96
- scan97
- scan98
- scan99
- scan100
- scan101
- scan102
- scan103
- scan104
- scan105
- scan107
- scan108
- scan109
- scan111
- scan112
- scan113
- scan115
- scan116
- scan119
- scan120
- scan121
- scan122
- scan123
- scan124
- scan125
- scan126
- scan127
- scan128

Testing

- scan1
- scan4
- scan9
- scan10
- scan11
- scan12
- scan13
- scan15
- scan23
- scan24
- scan29
- scan32
- scan33
- scan34
- scan48
- scan49
- scan62
- scan75
- scan77
- scan110
- scan114
- scan118

Validation

- scan3
- scan5
- scan17
- scan21
- scan28
- scan35
- scan37
- scan38
- scan40
- scan43
- scan56
- scan59
- scan66
- scan67
- scan82
- scan86
- scan106
- scan117

8.4 Error distribution

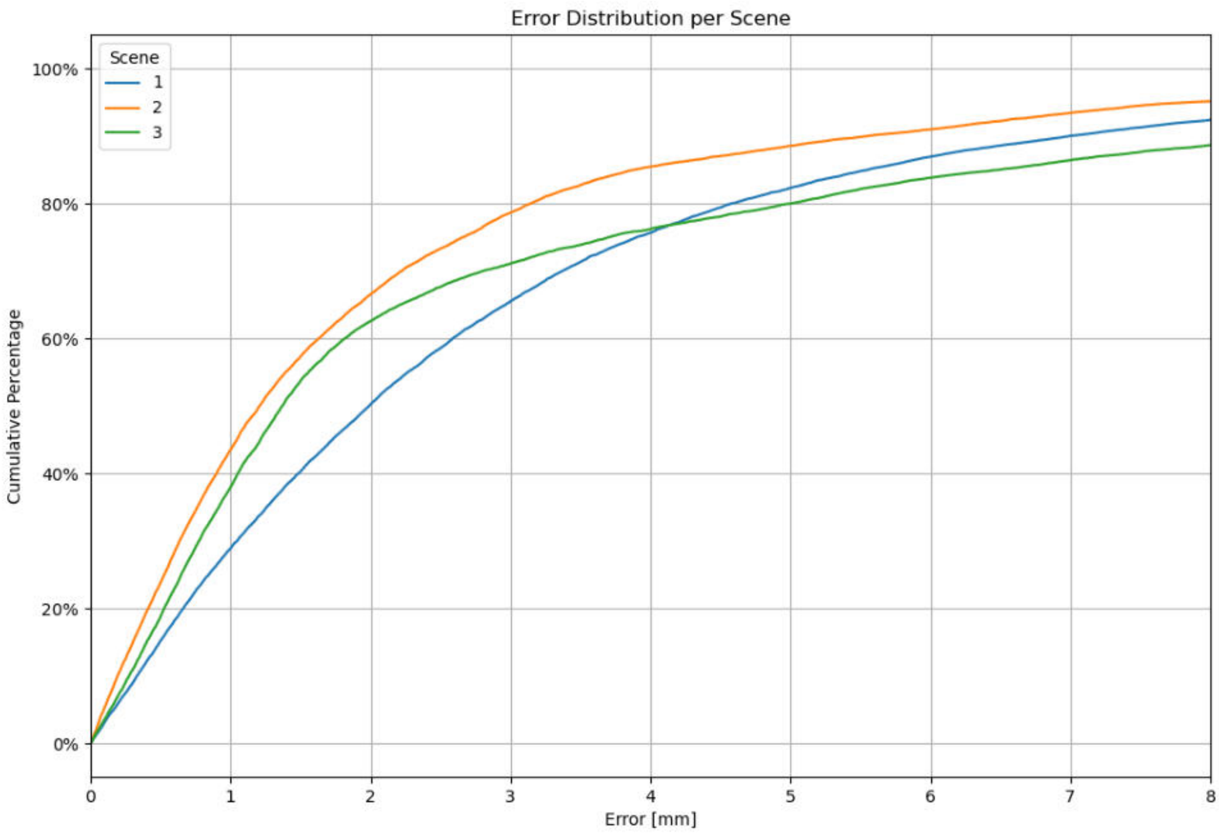
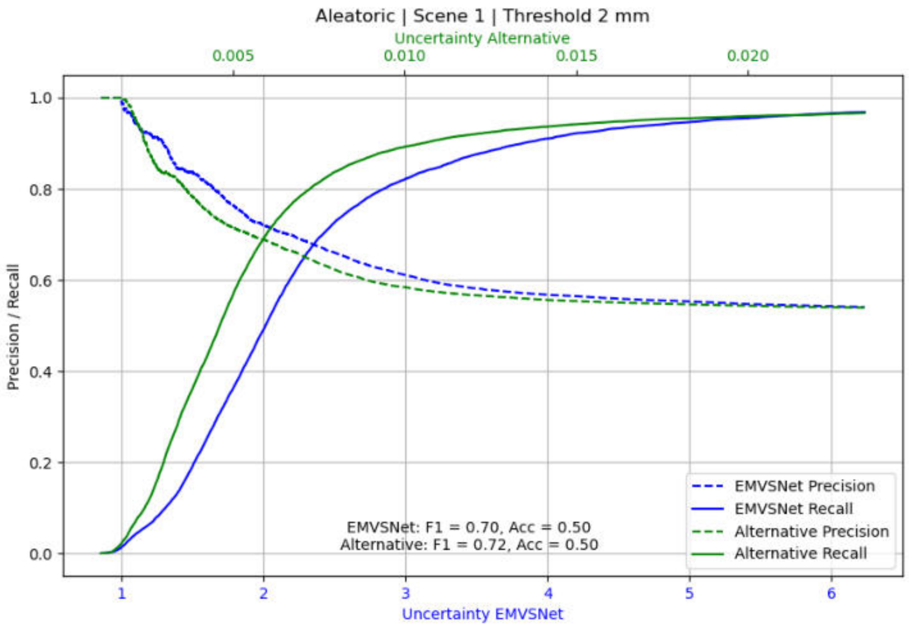
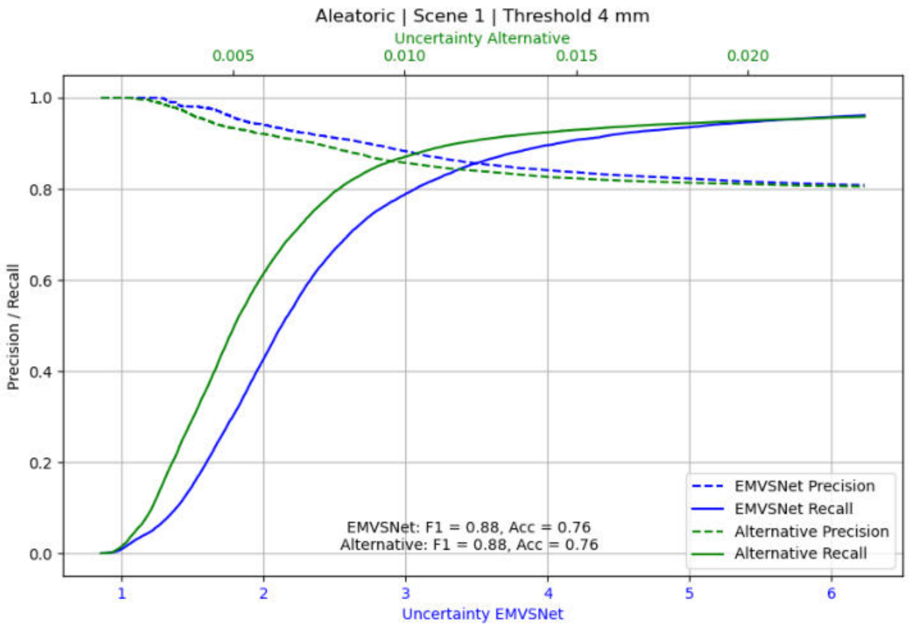


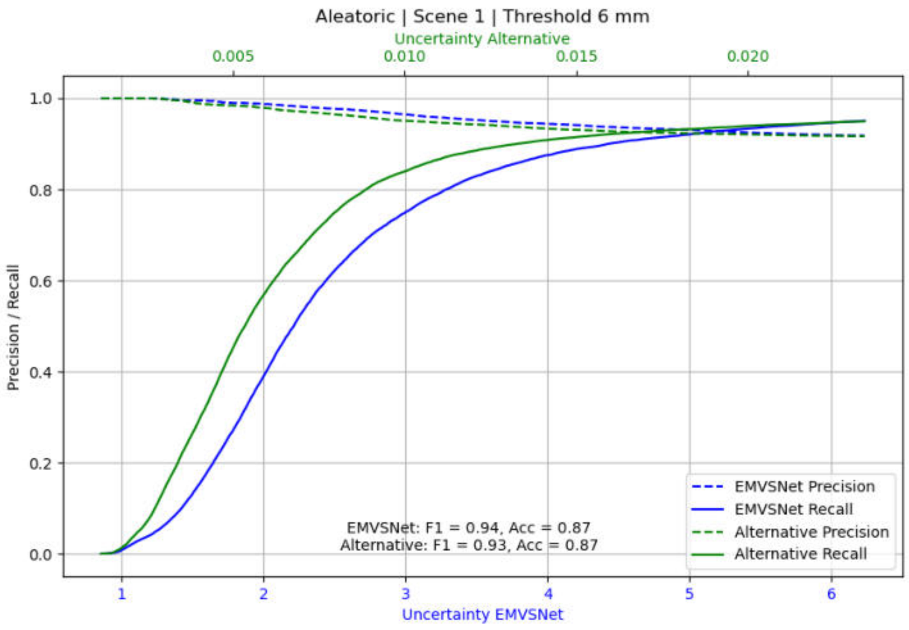
Figure 8.4: Cumulative distribution of error for the three test scenes.

8.5 PR curves

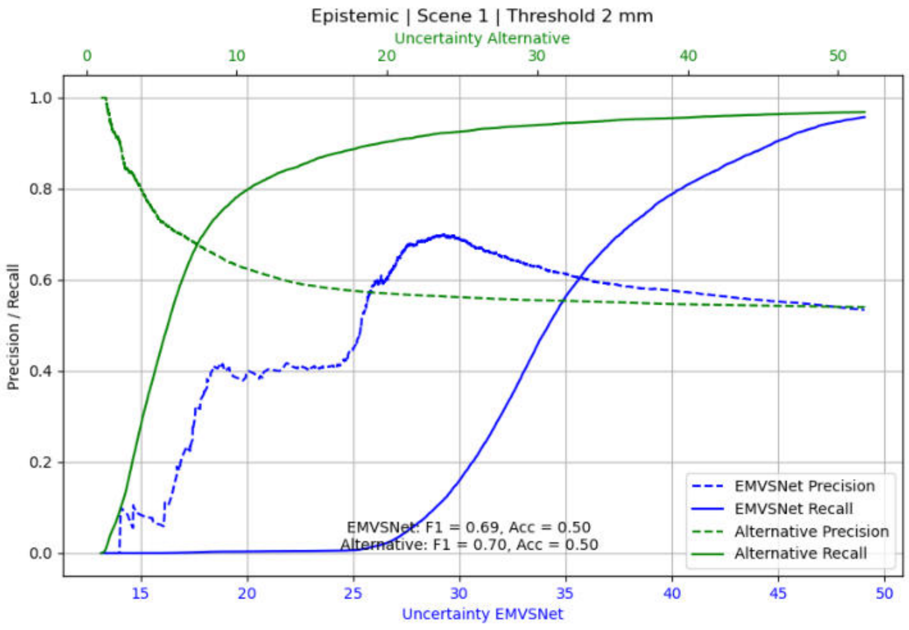
Aleatoric Uncertainty for Scene 1

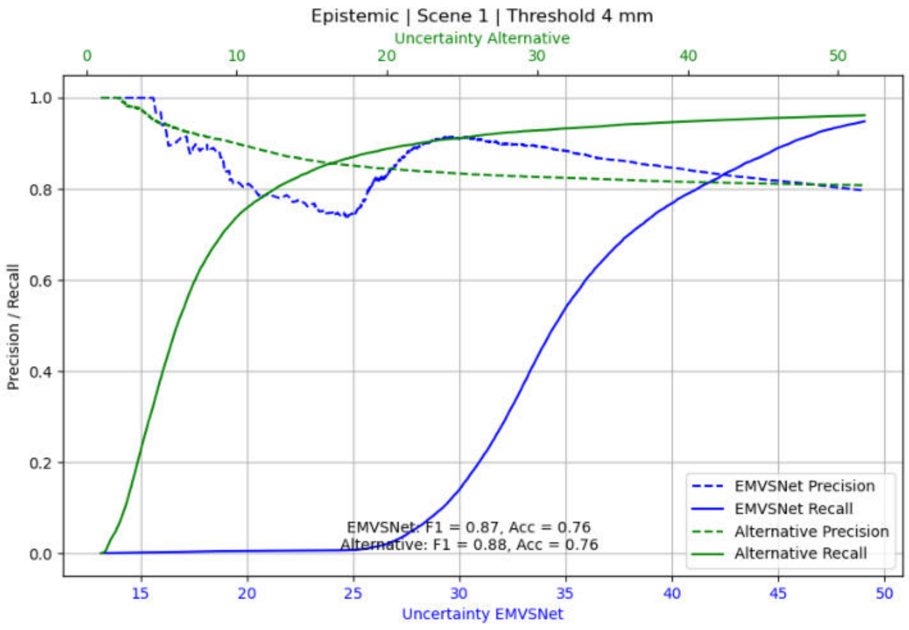


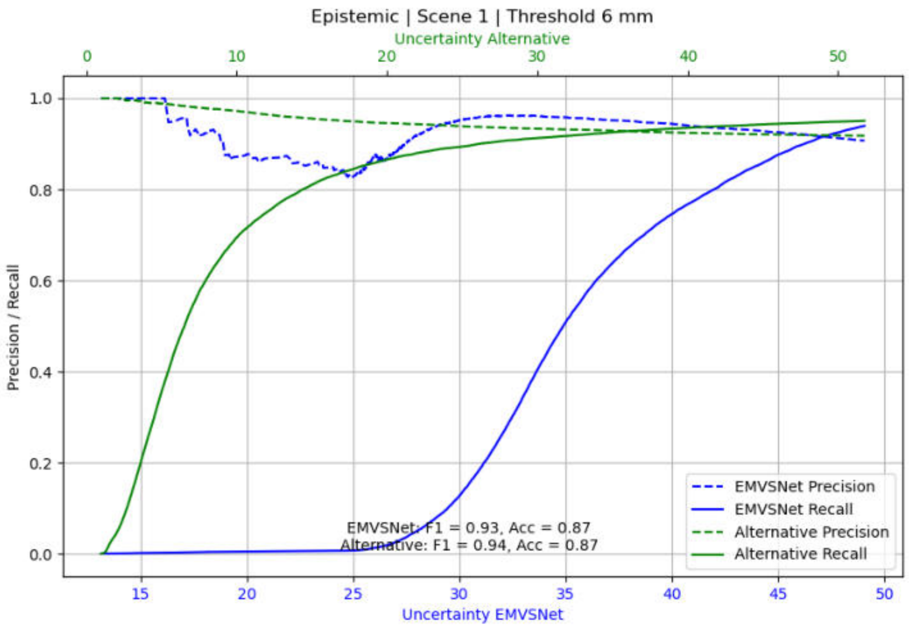




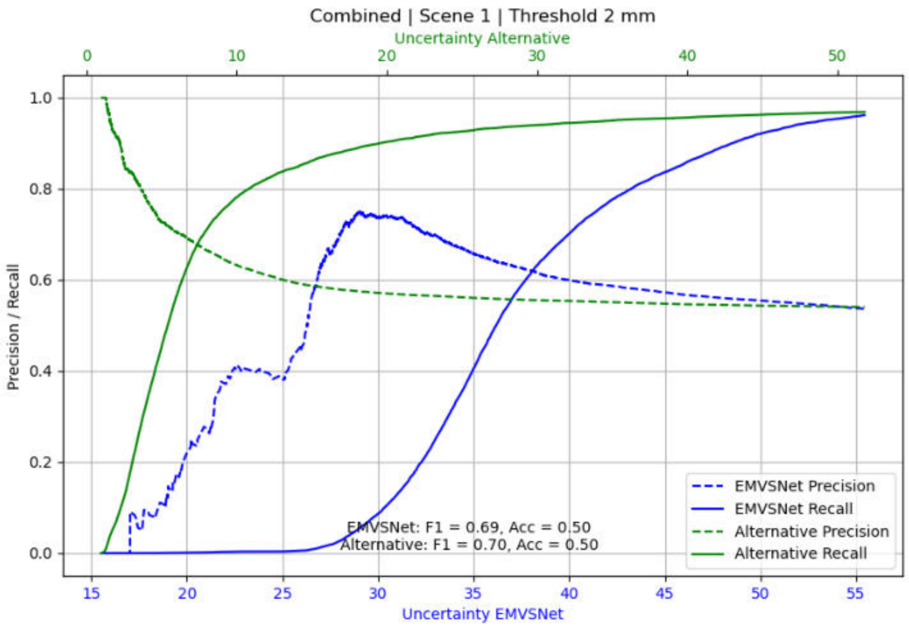
Epistemic Uncertainty for Scene 1

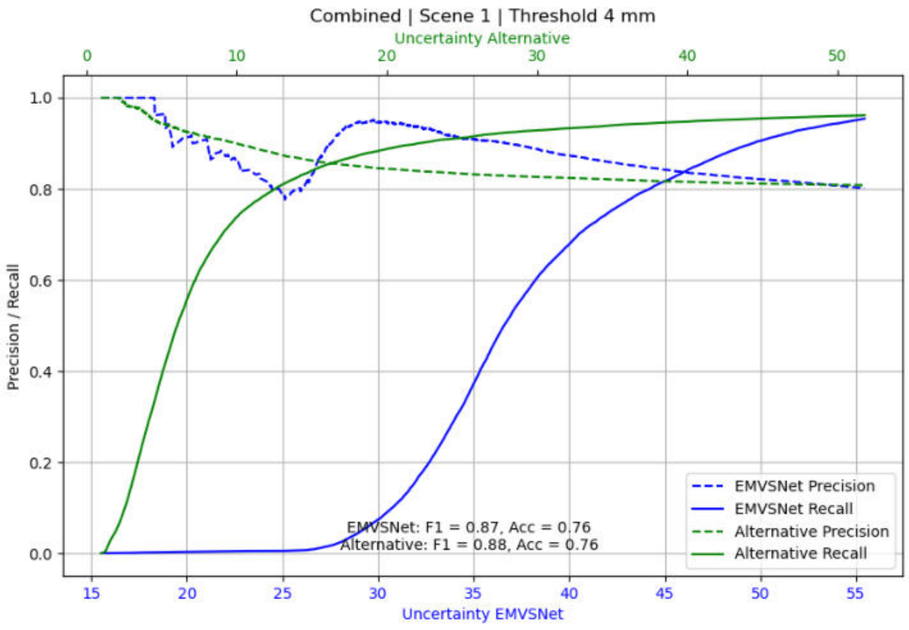


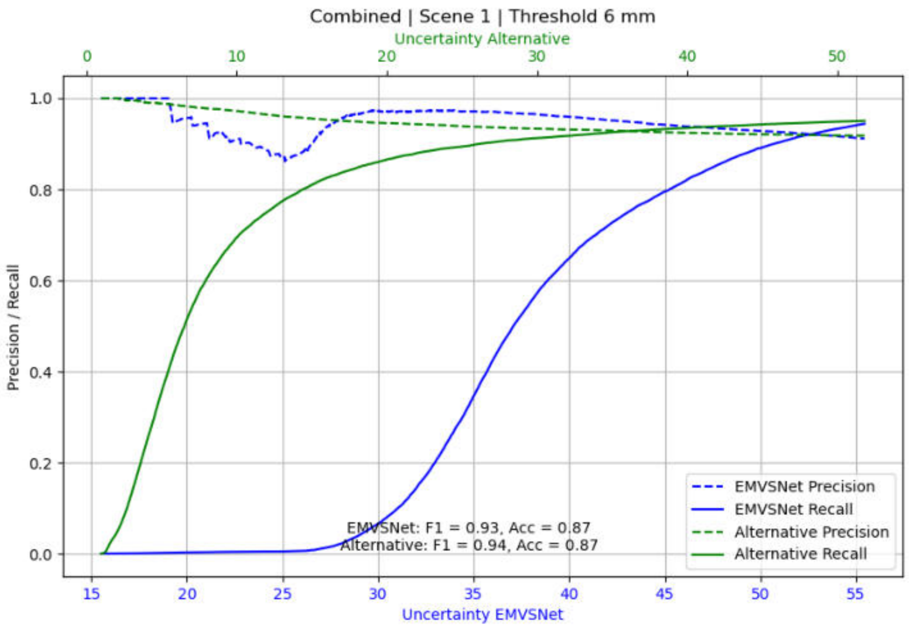




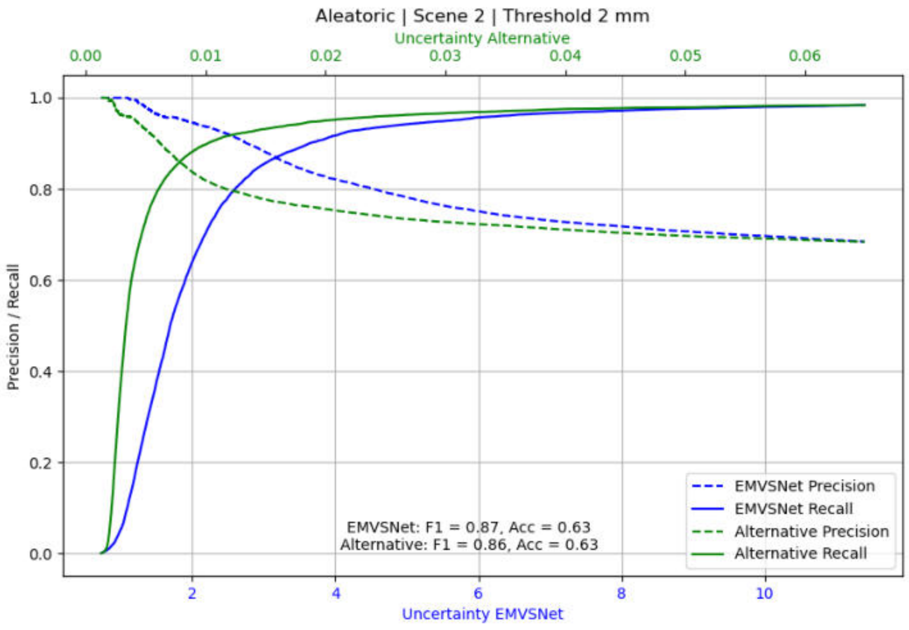
Combined Uncertainty for Scene 1

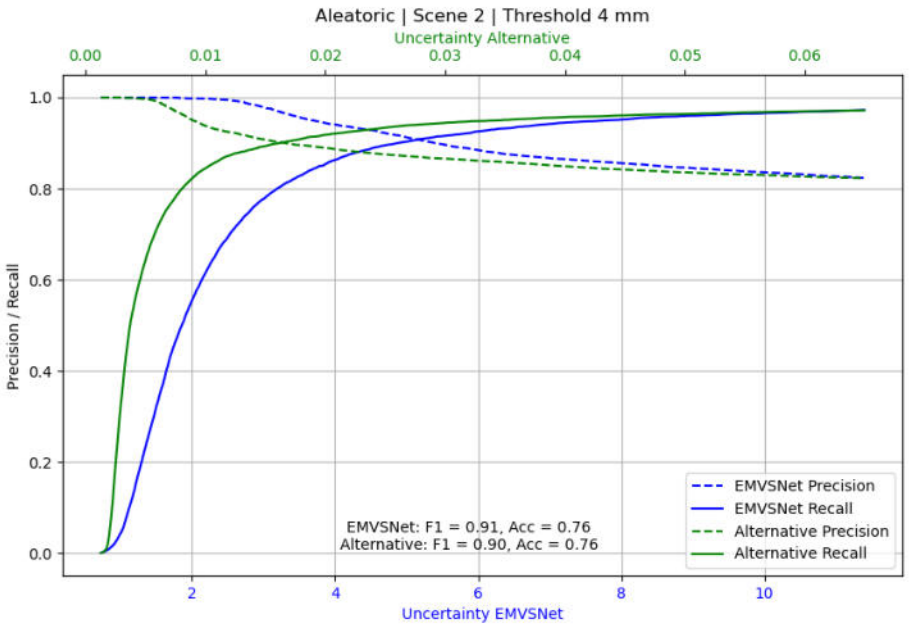


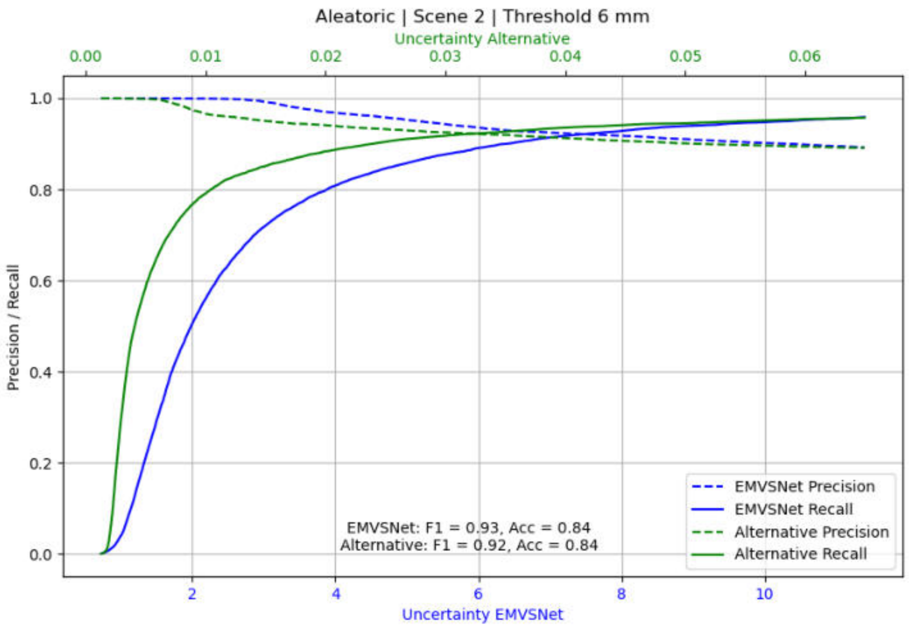




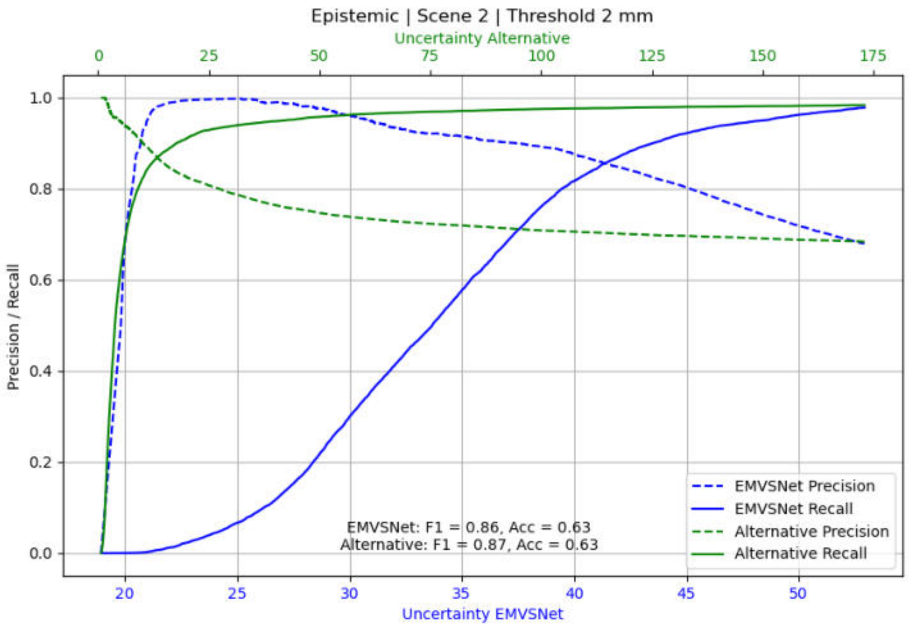
Aleatoric Uncertainty for Scene 2

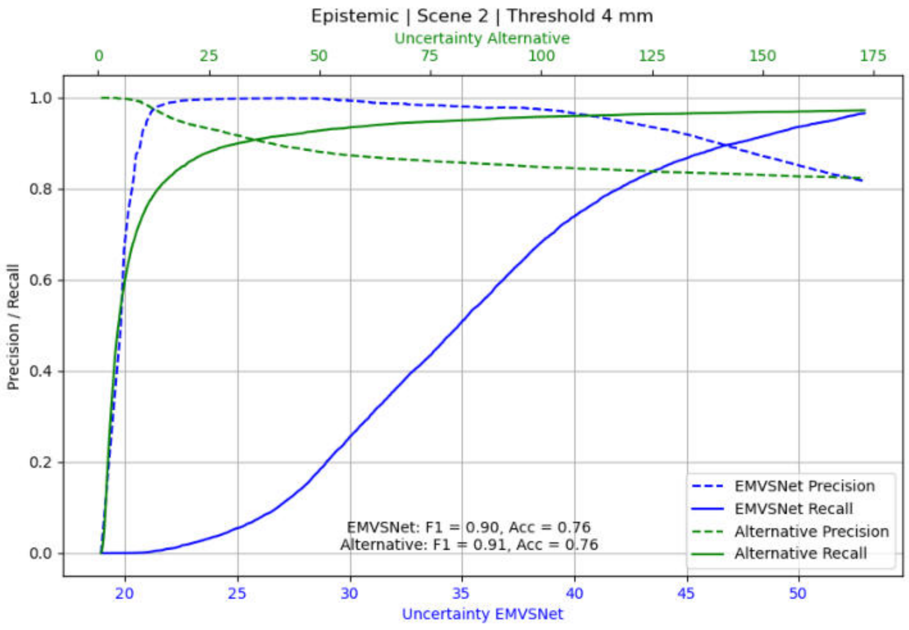


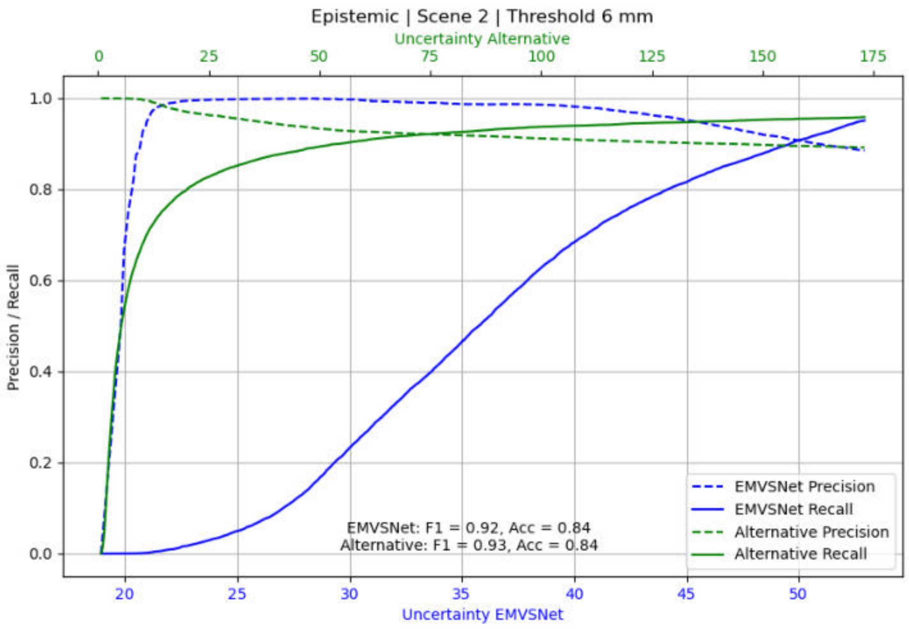




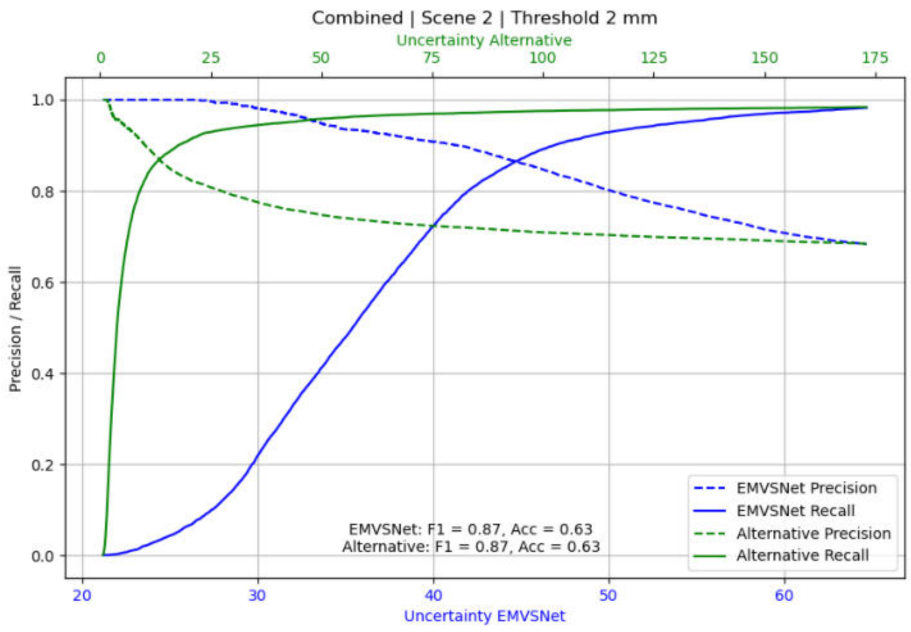
Epistemic Uncertainty for Scene 2

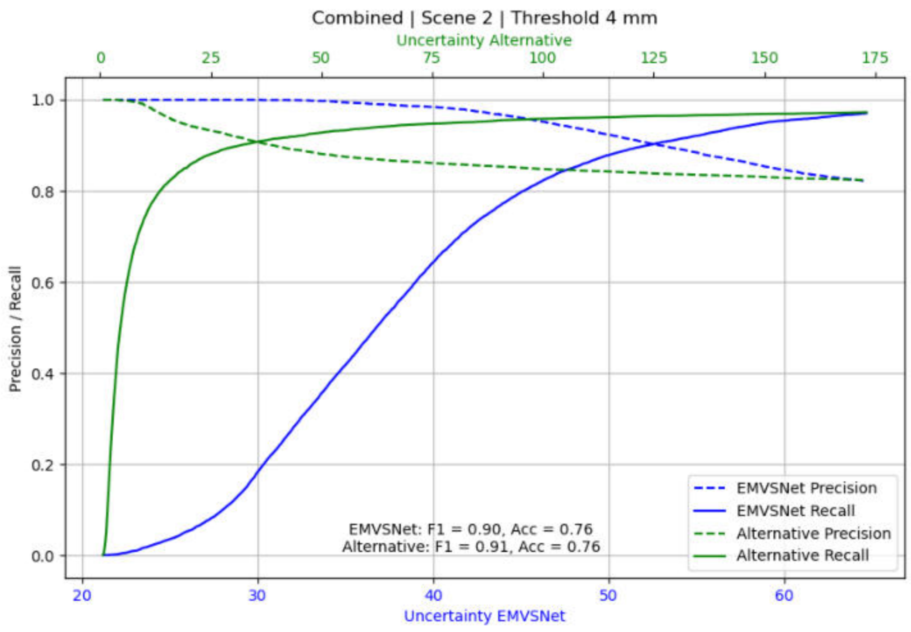


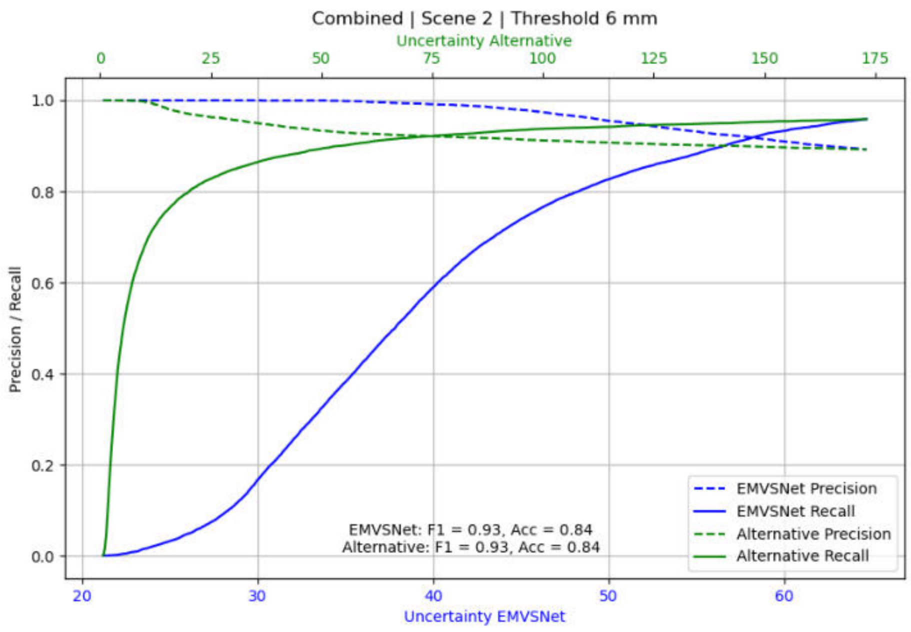




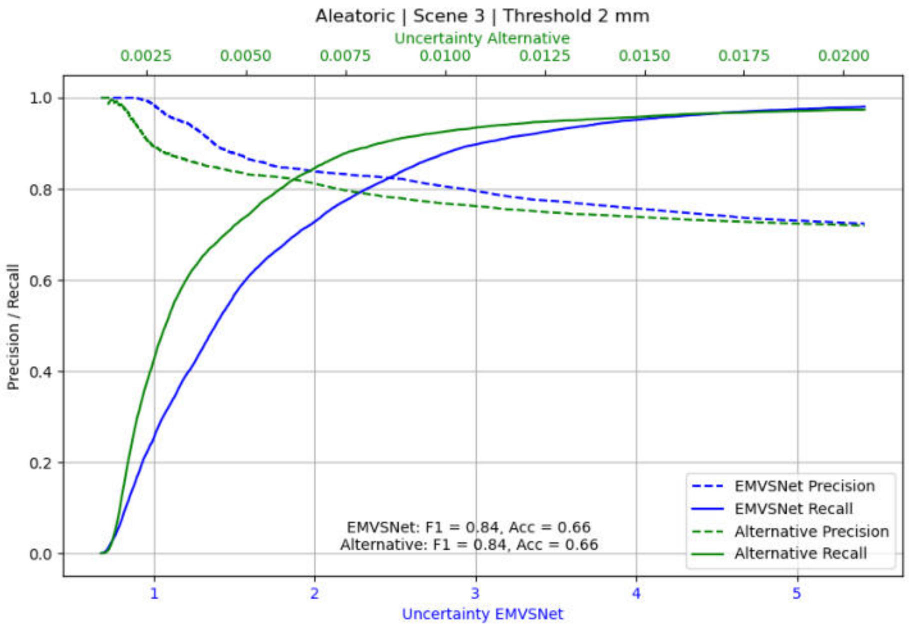
Combined Uncertainty for Scene 2

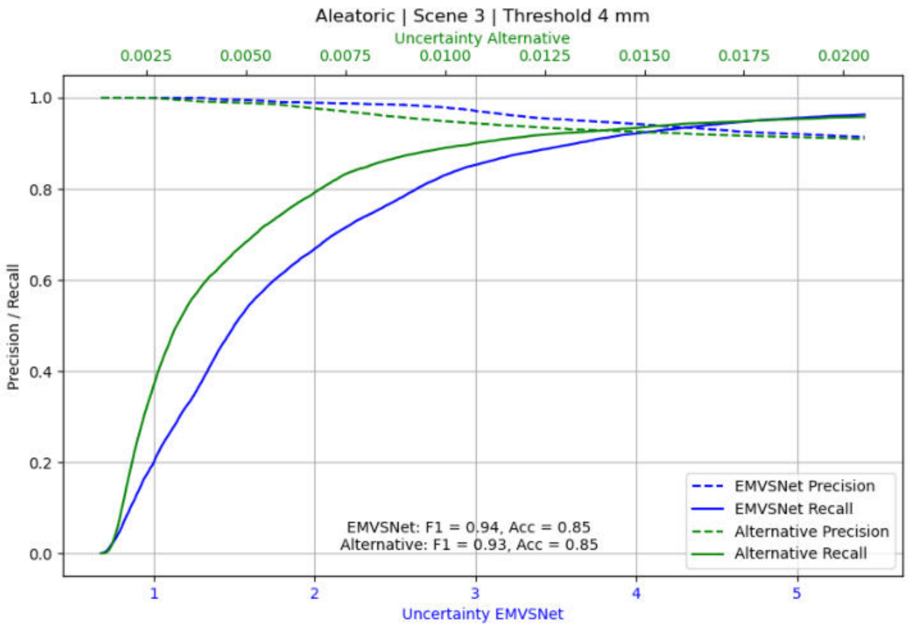


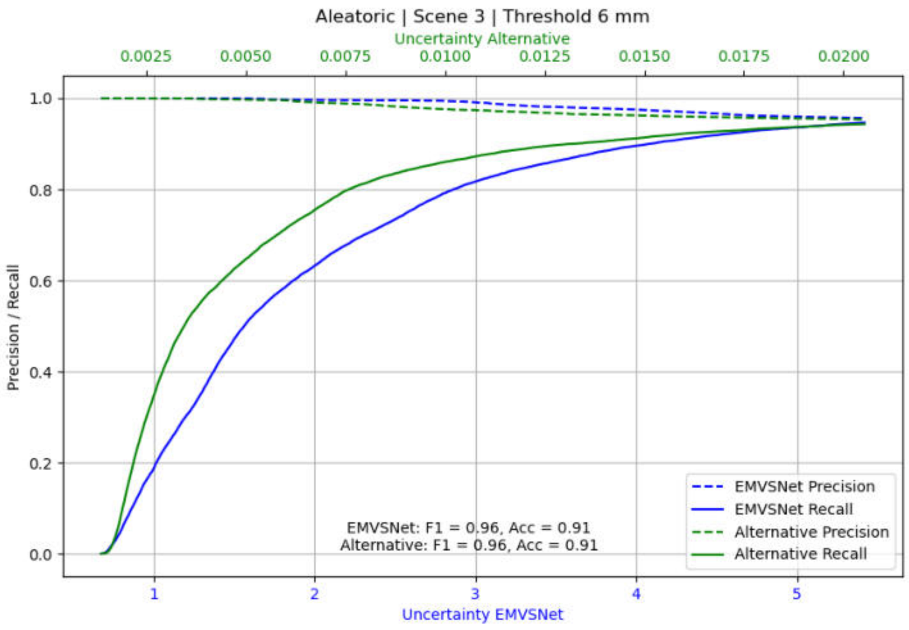




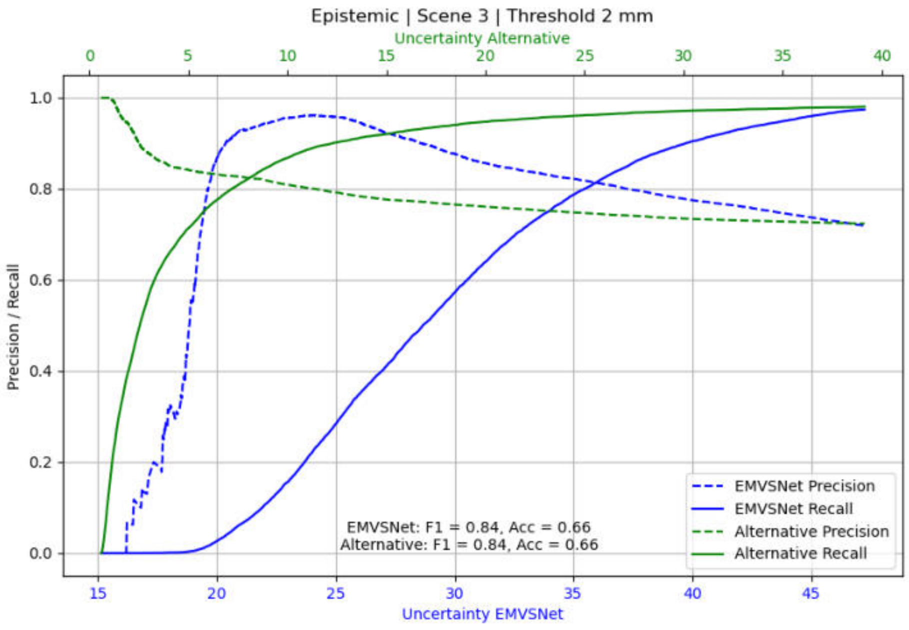
Aleatoric Uncertainty for Scene 3

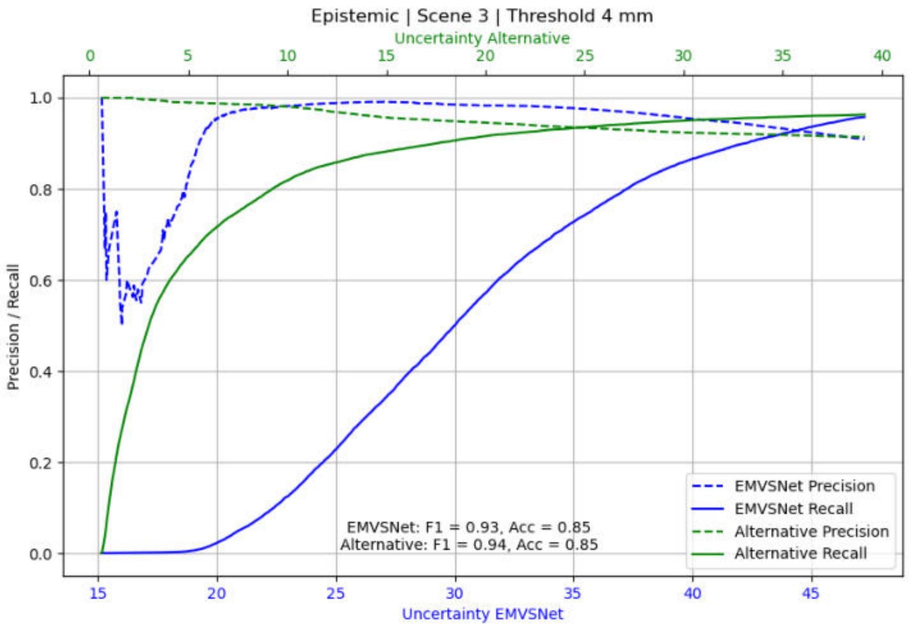


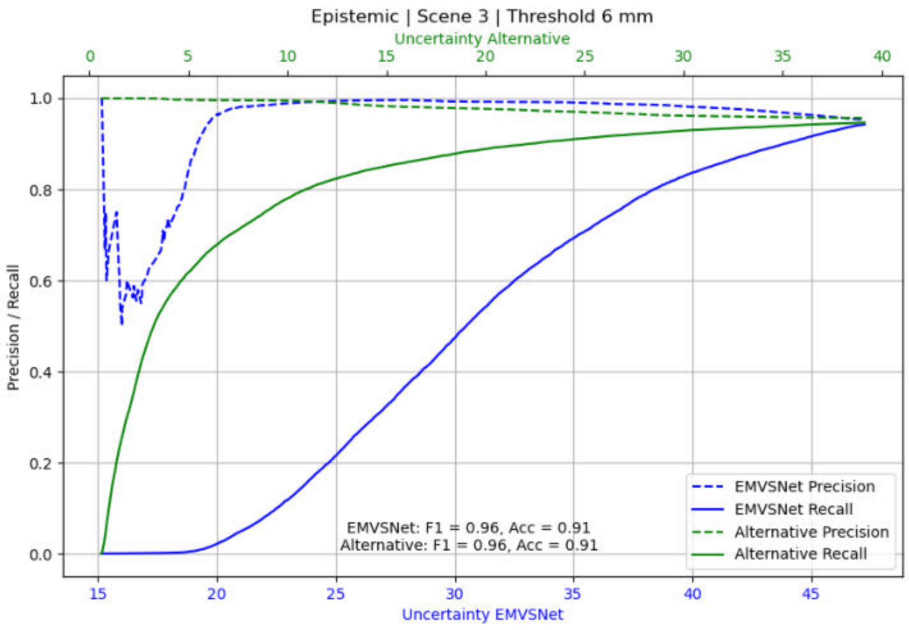




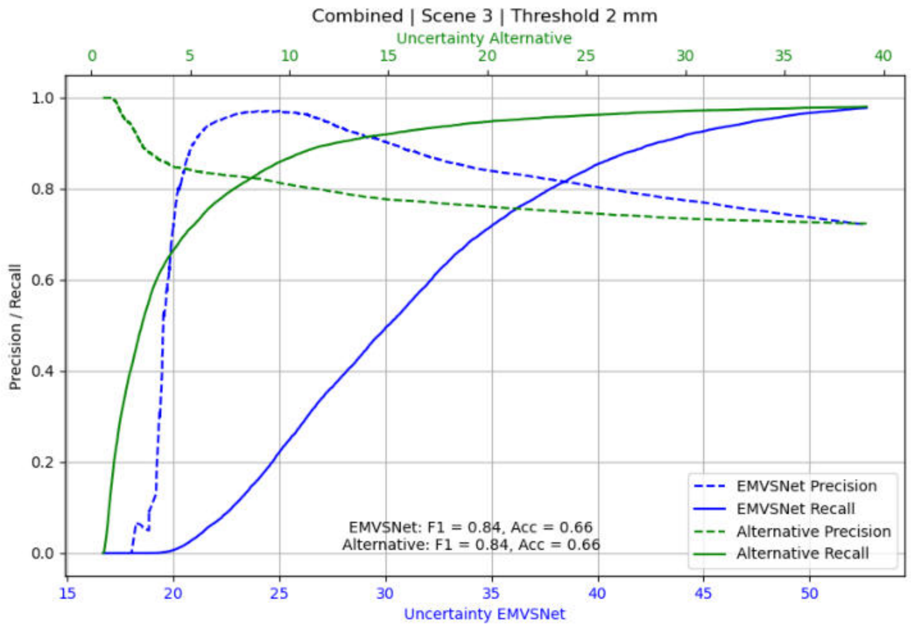
Epistemic Uncertainty for Scene 3

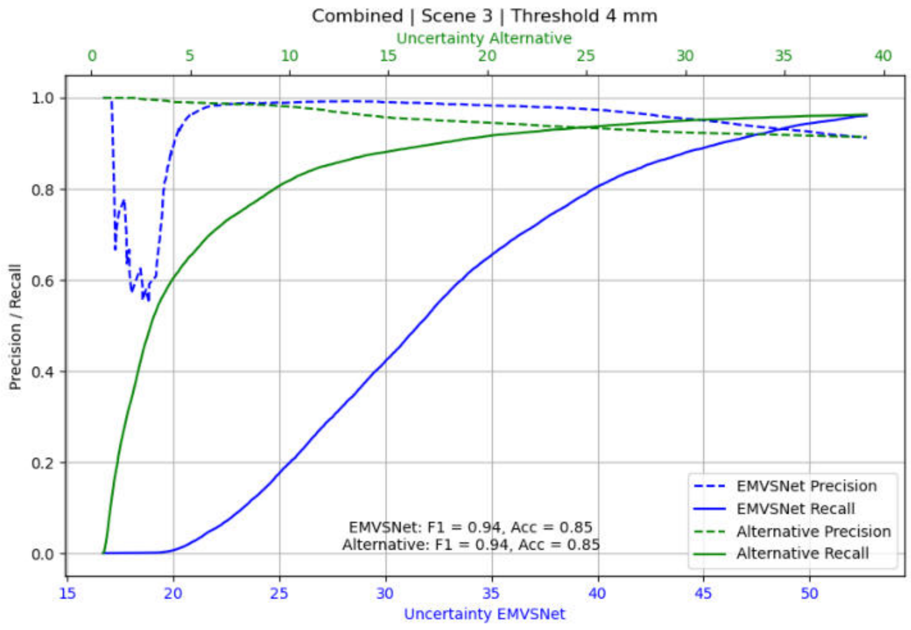


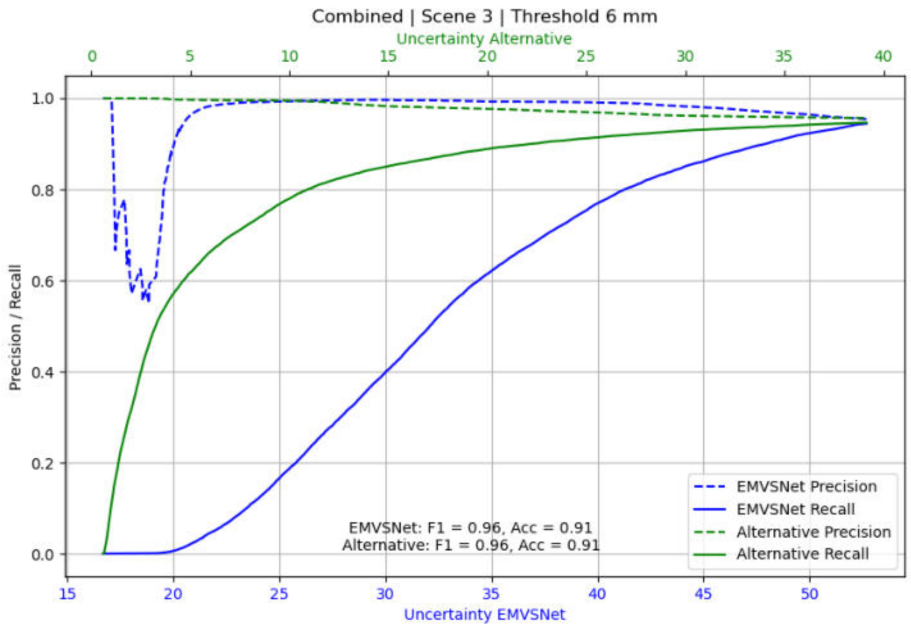




Combined Uncertainty for Scene 3

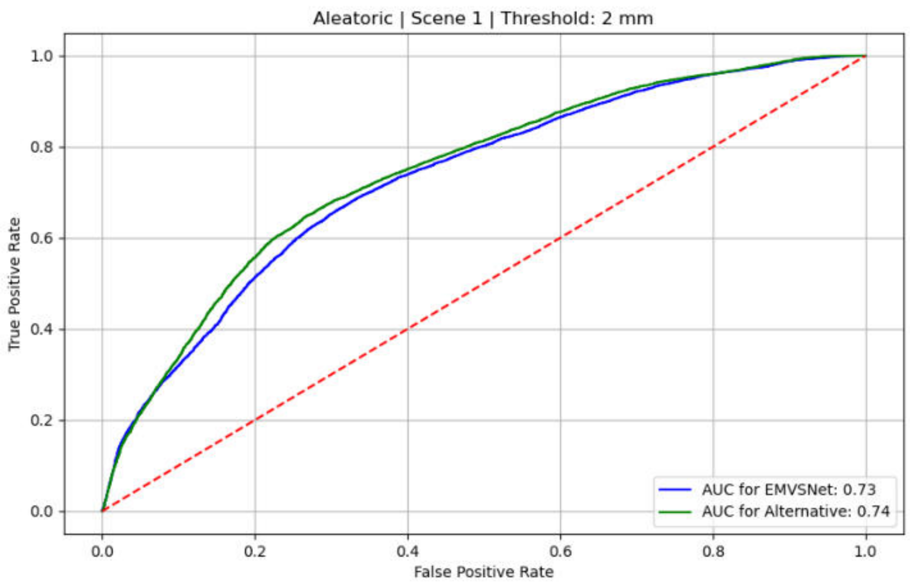


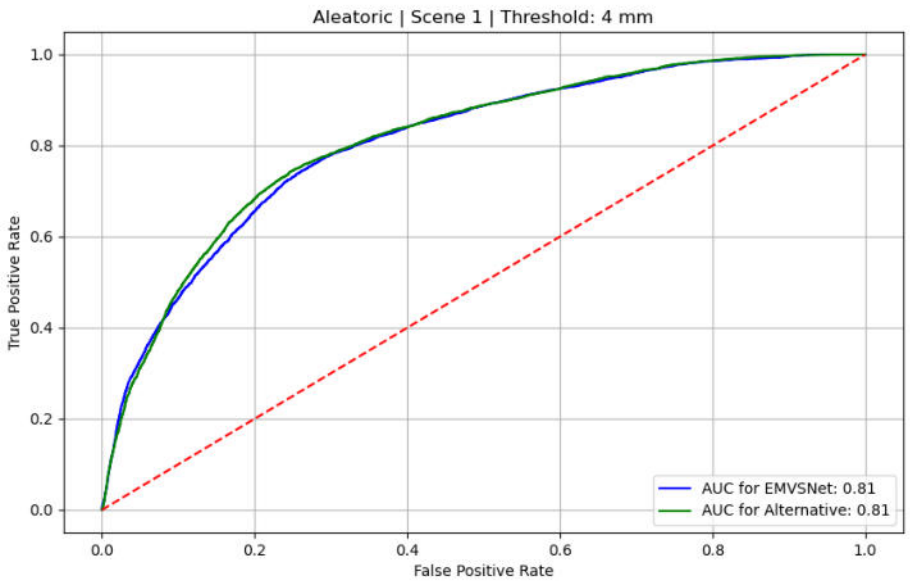




8.6 ROC curves

Aleatoric Uncertainty for Scene 1





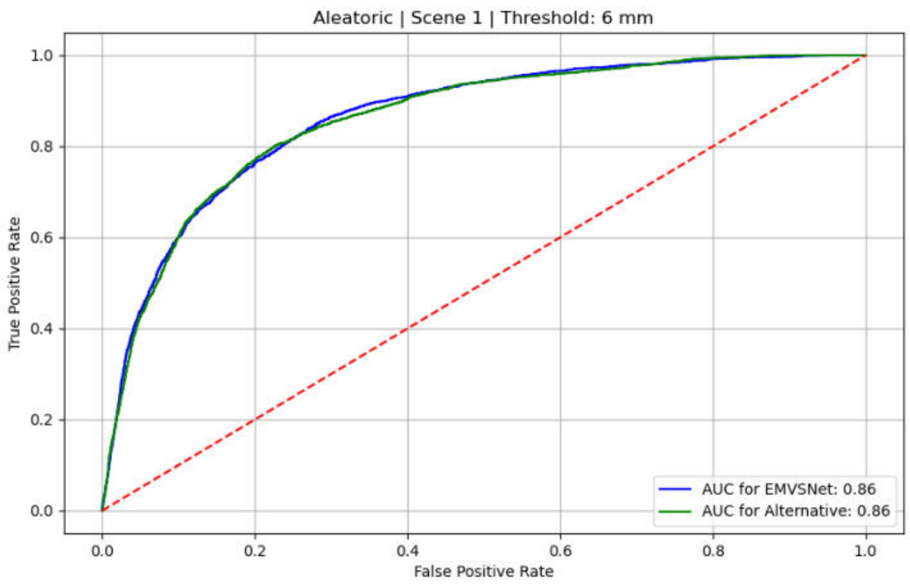
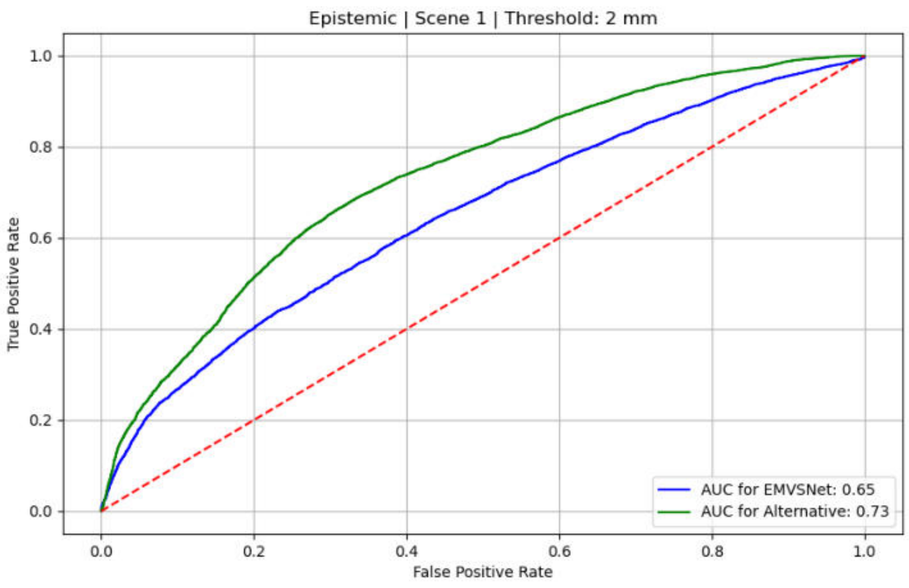
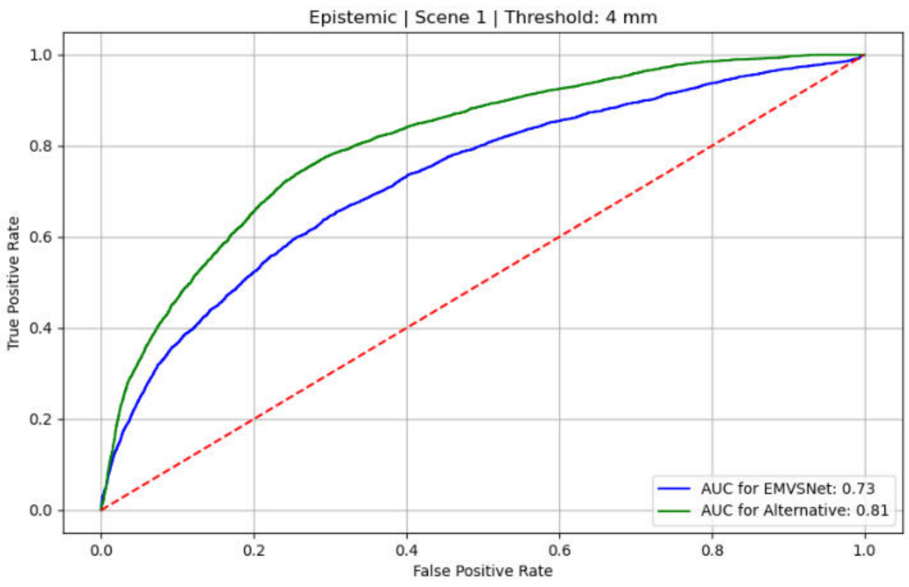


Figure 8.14: The red lines indicates an AUC of 0.5 what would mean random guessing.

Epistemic Uncertainty for Scene 1





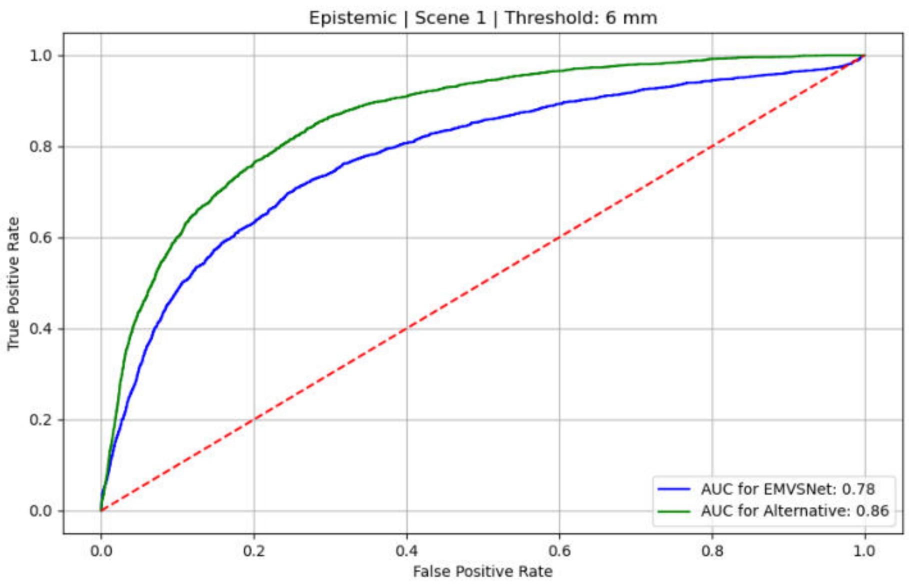
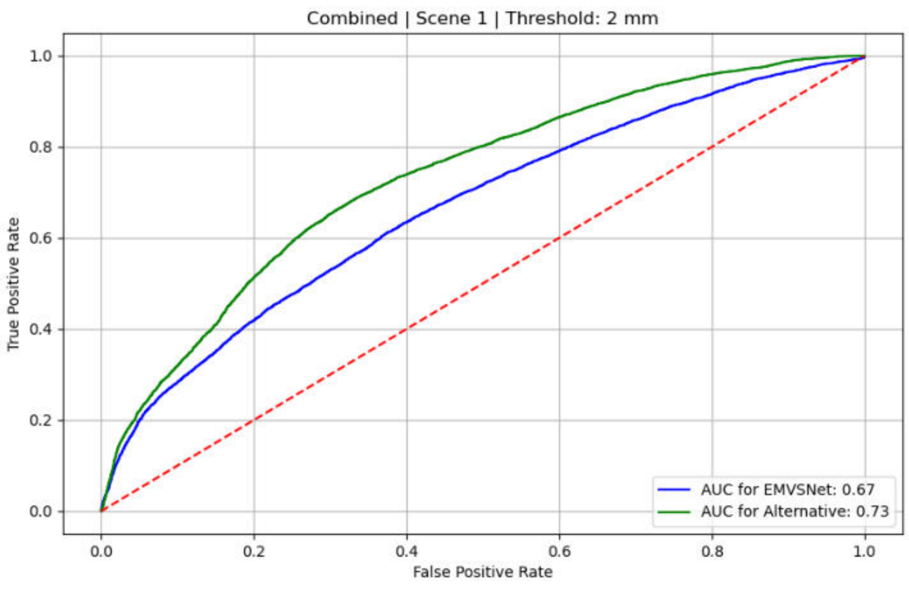
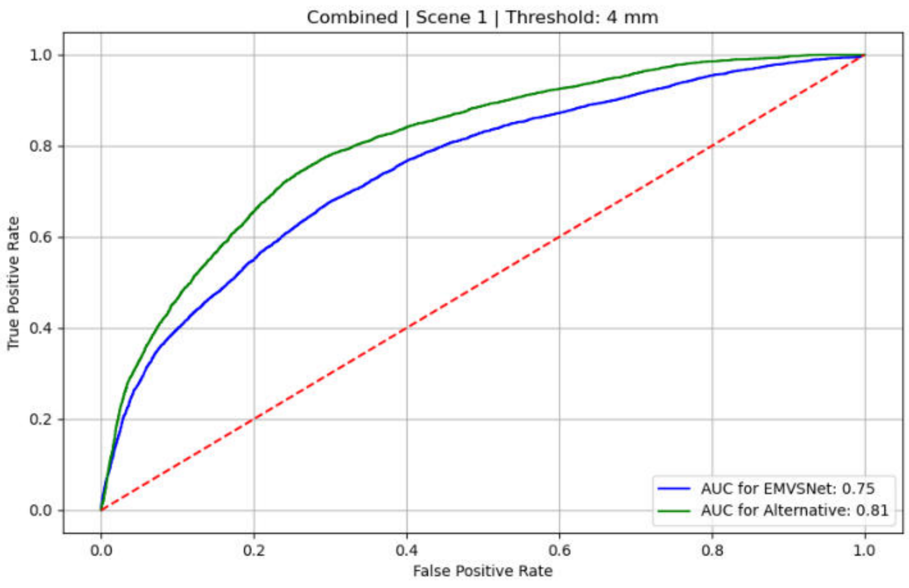


Figure 8.15: The red lines indicates an AUC of 0.5 what would mean random guessing.

Combined Uncertainty for Scene 1





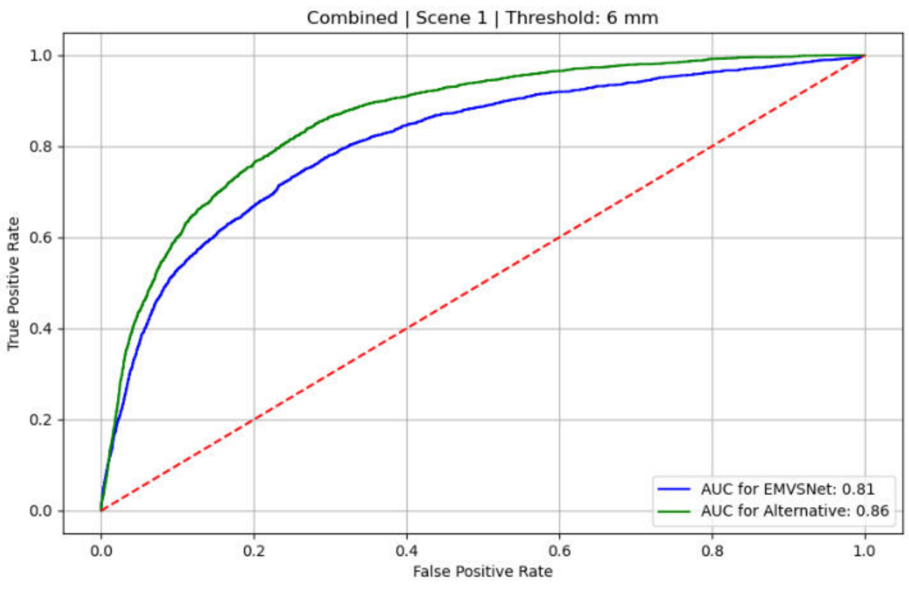
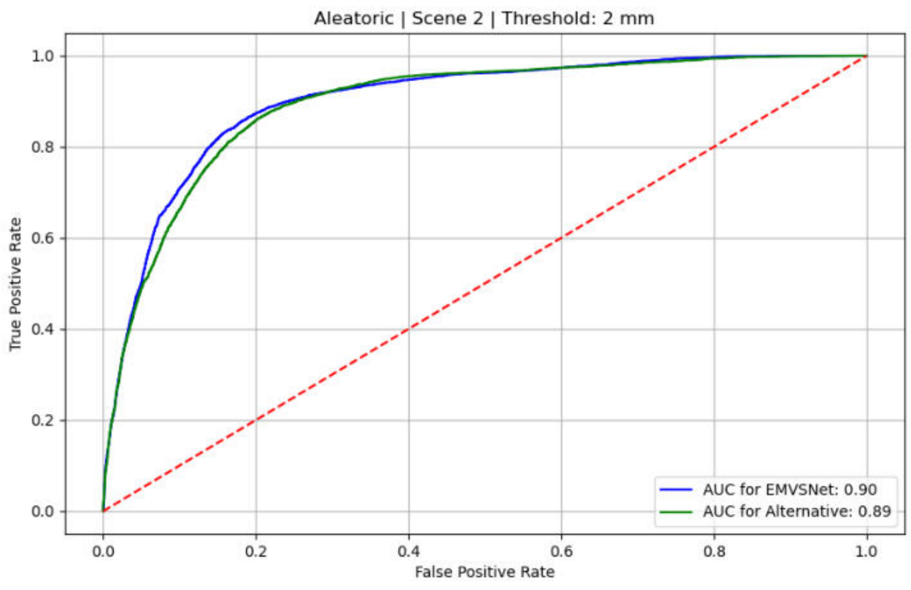
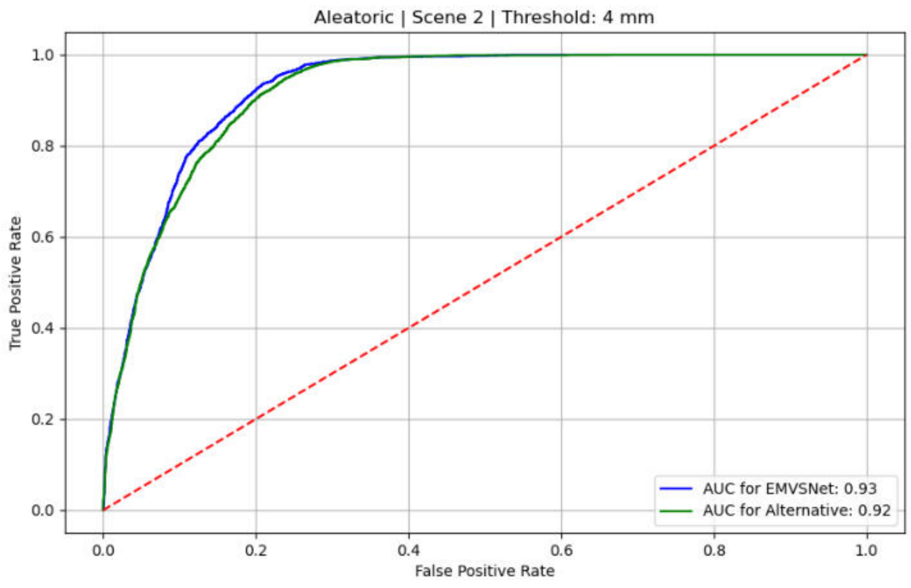


Figure 8.16: The red lines indicates an AUC of 0.5 what would mean random guessing.

Aleatoric Uncertainty for Scene 2





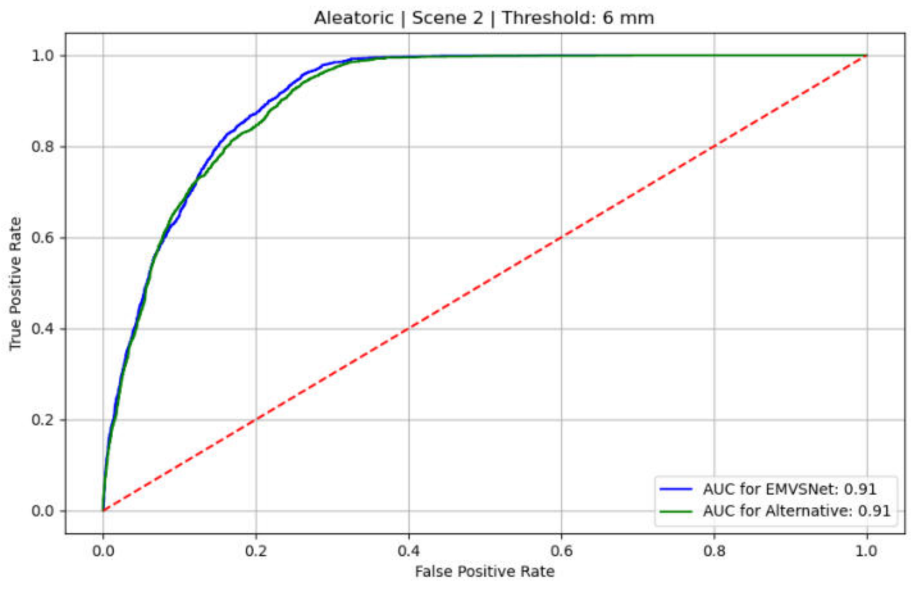
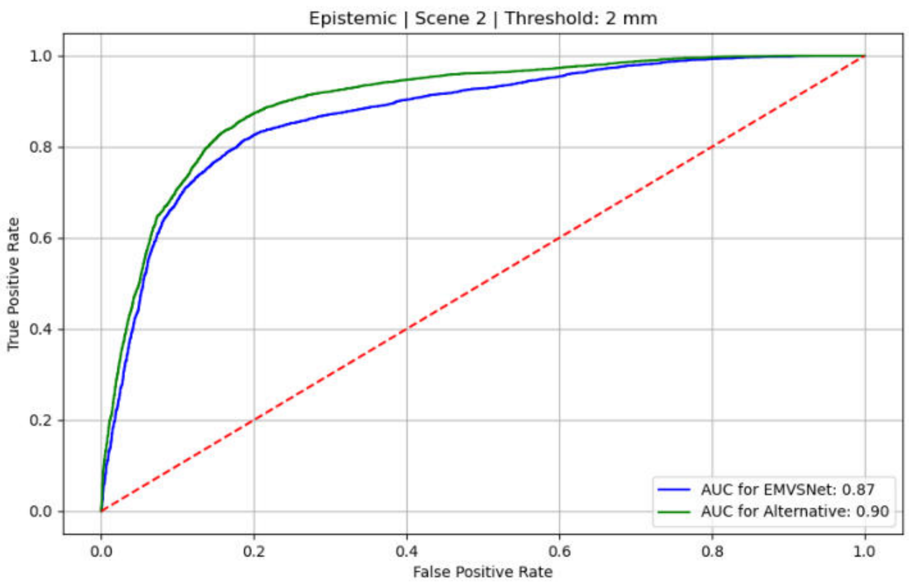
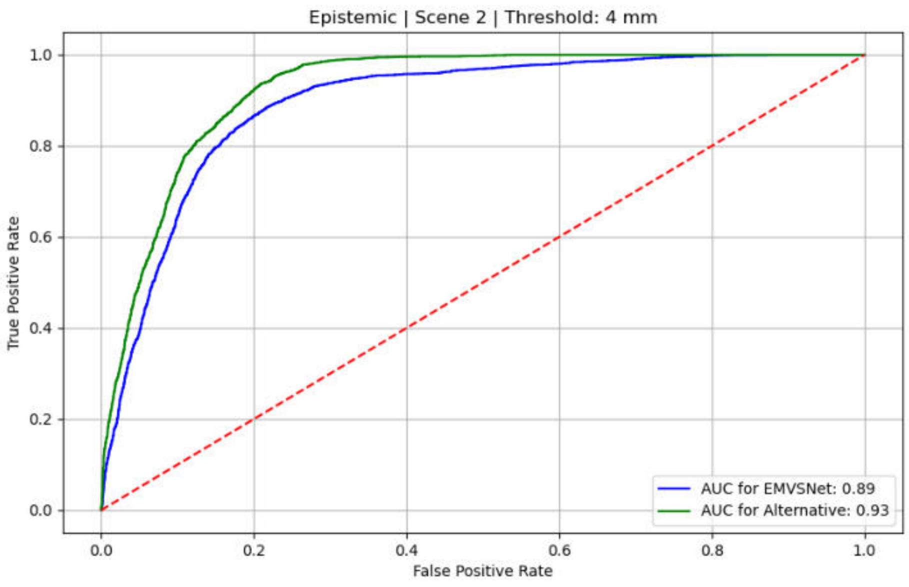


Figure 8.17: The red lines indicates an AUC of 0.5 what would mean random guessing.

Epistemic Uncertainty for Scene 2





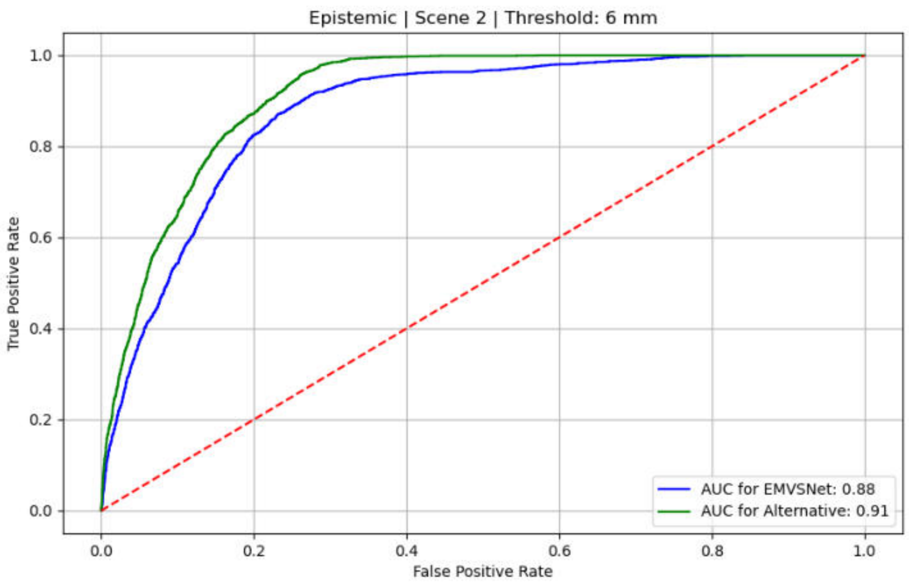
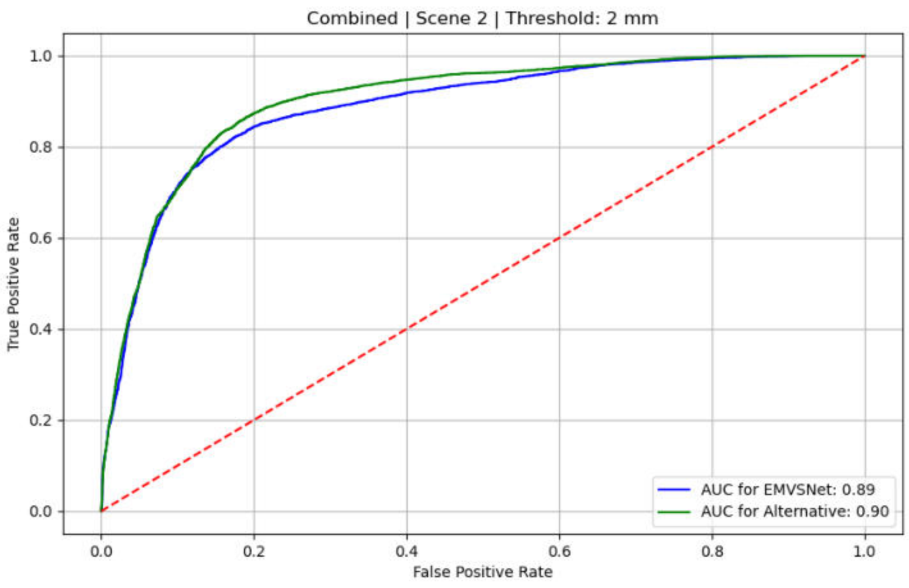
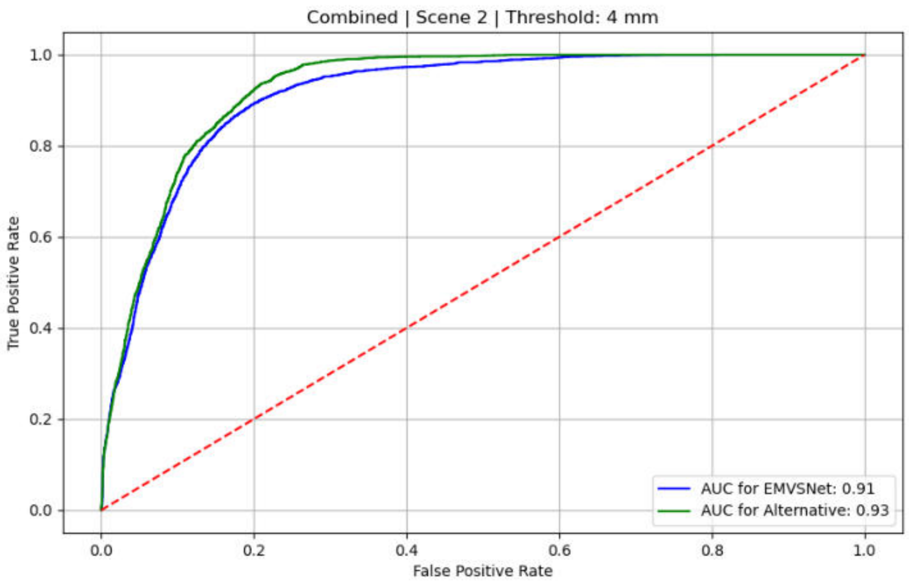


Figure 8.18: The red lines indicates an AUC of 0.5 what would mean random guessing.

Combined Uncertainty for Scene 2





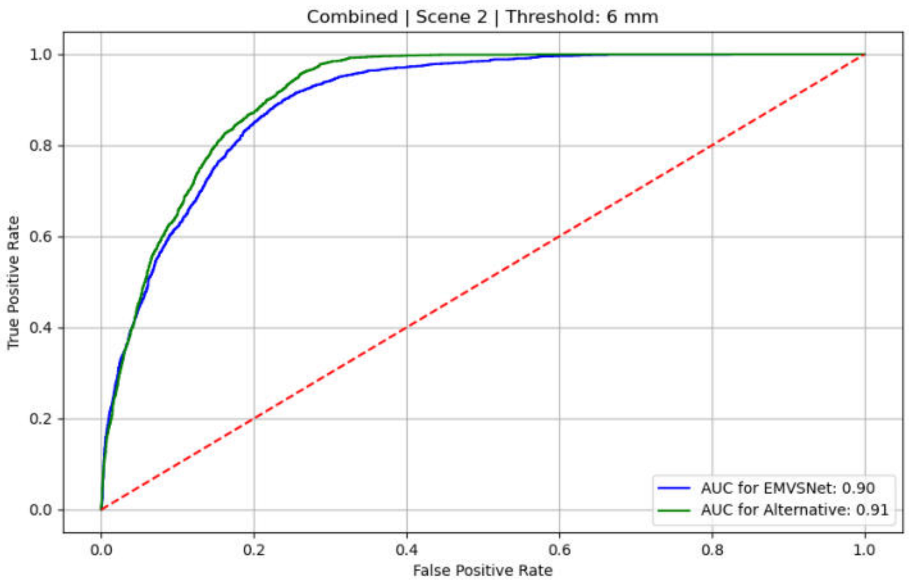
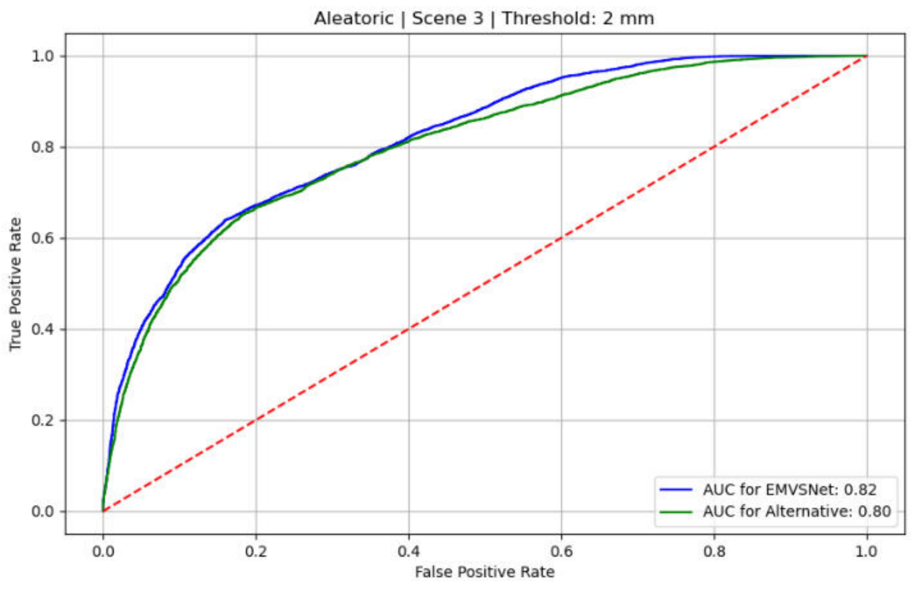
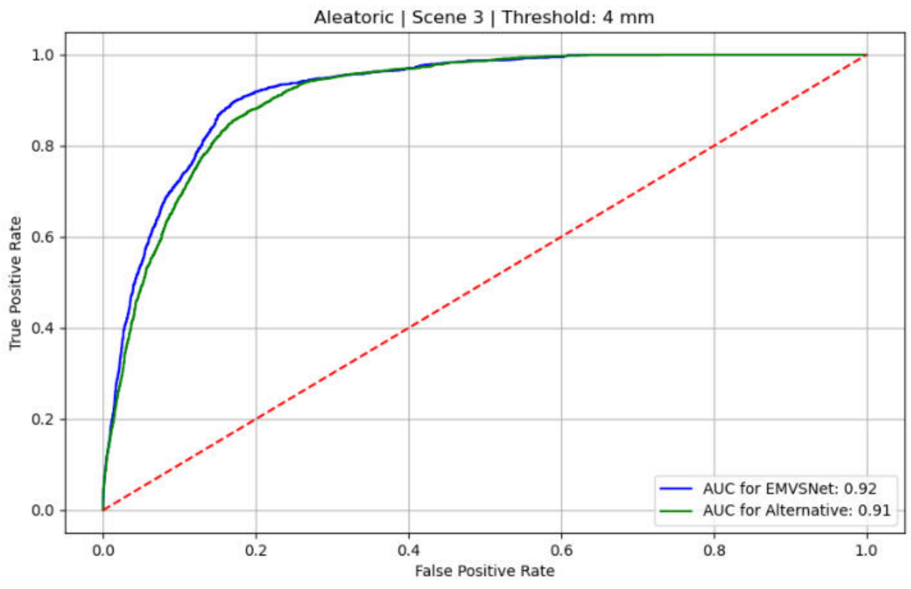


Figure 8.19: The red lines indicates an AUC of 0.5 what would mean random guessing.

Aleatoric Uncertainty for Scene 3





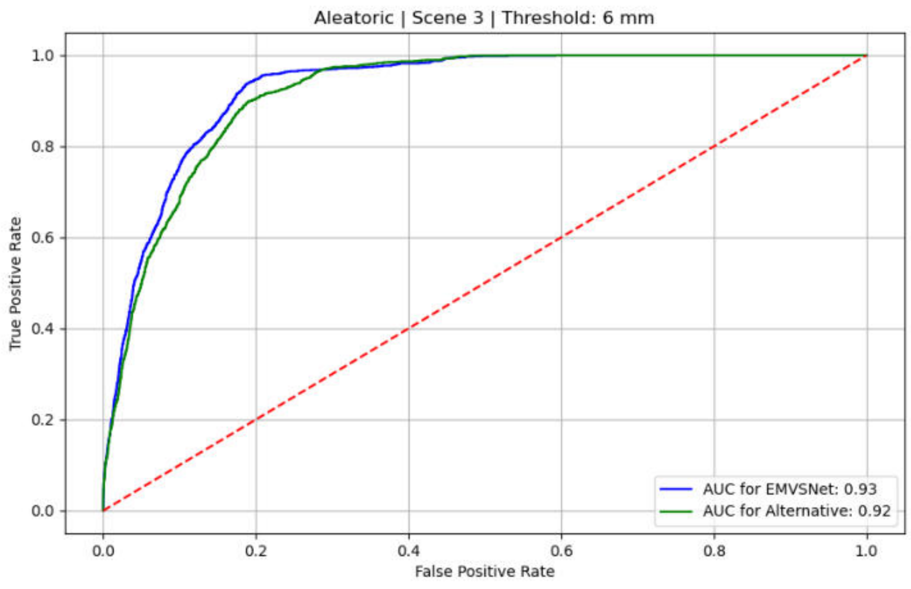
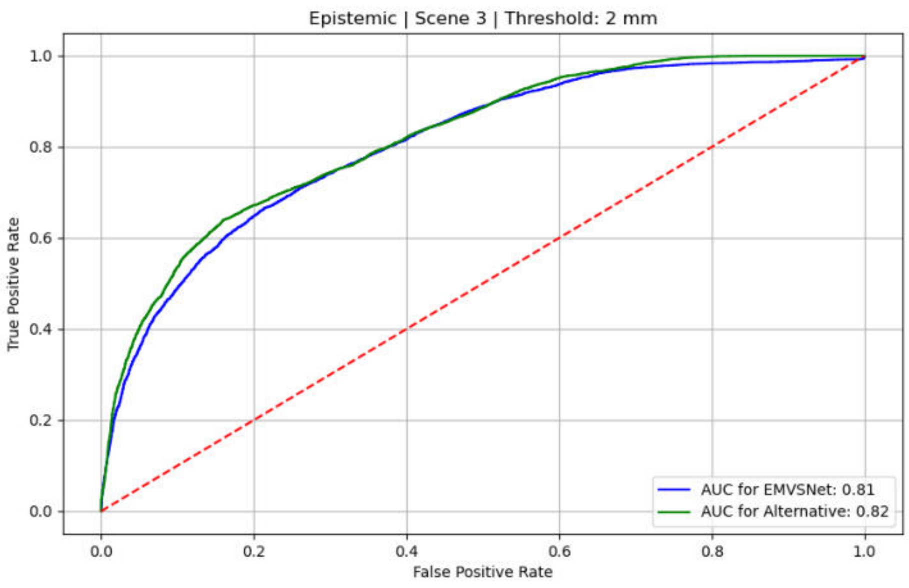
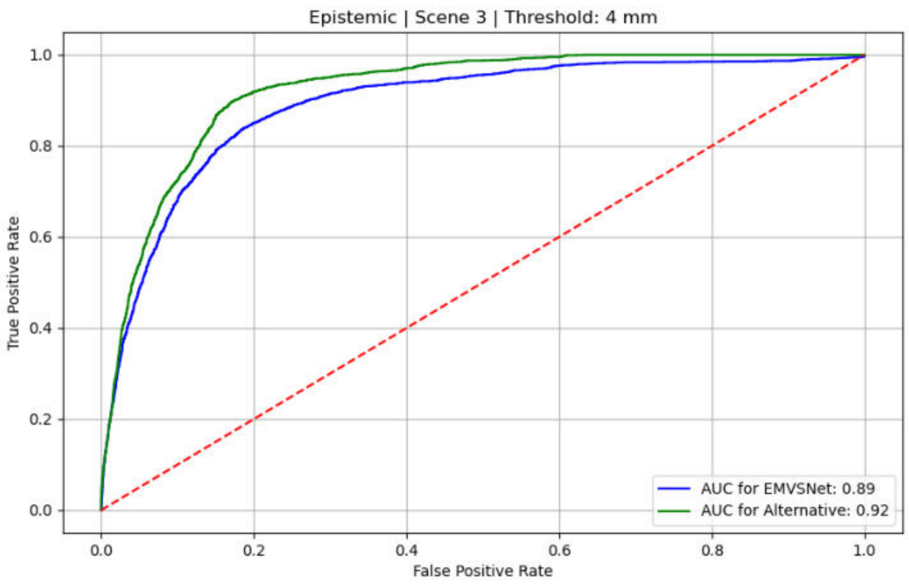


Figure 8.20: The red lines indicates an AUC of 0.5 what would mean random guessing.

Epistemic Uncertainty for Scene 3





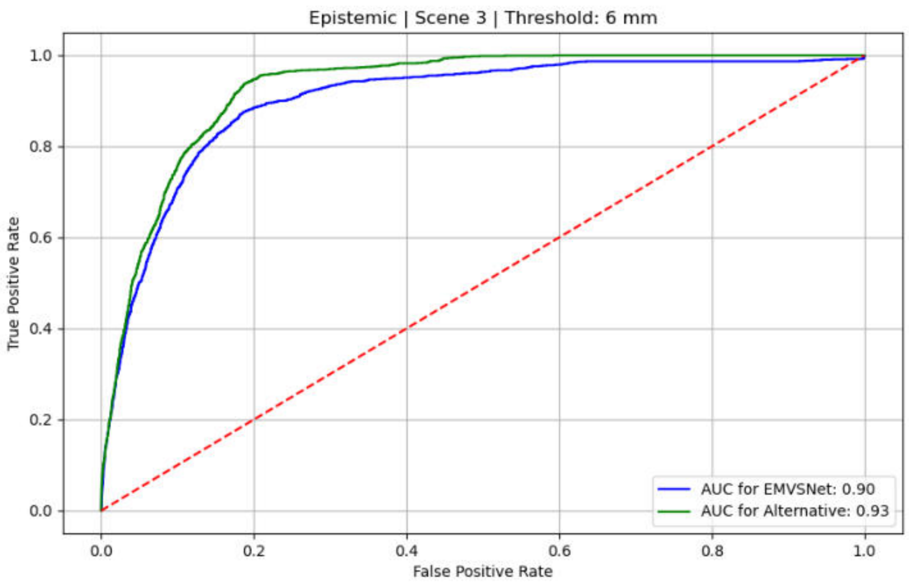
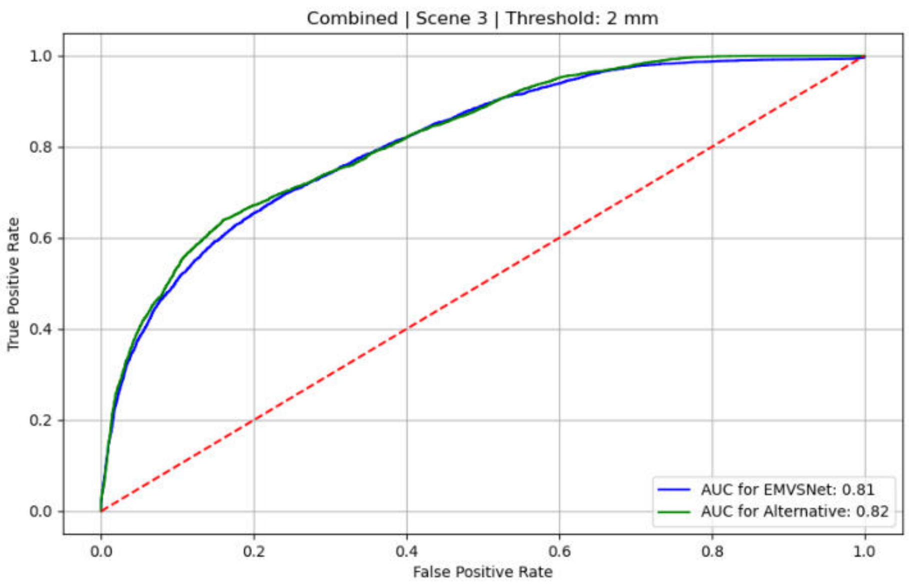
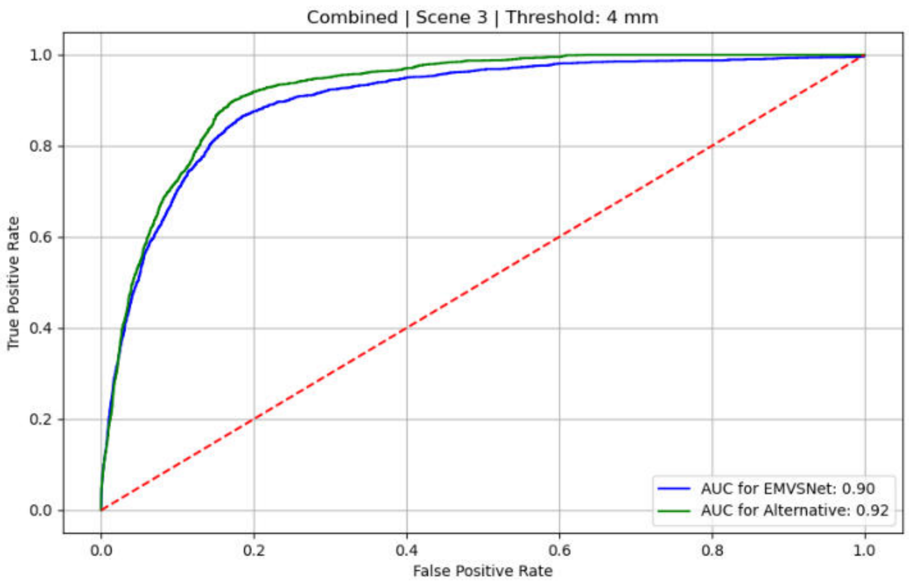


Figure 8.21: The red lines indicates an AUC of 0.5 what would mean random guessing.

Combined Uncertainty for Scene 3





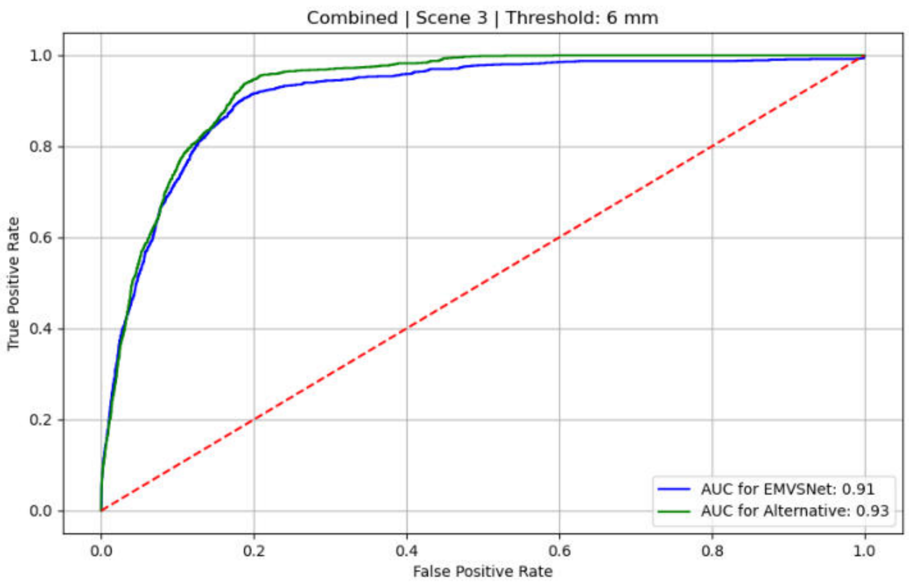


Figure 8.22: The red lines indicates an AUC of 0.5 what would mean random guessing.

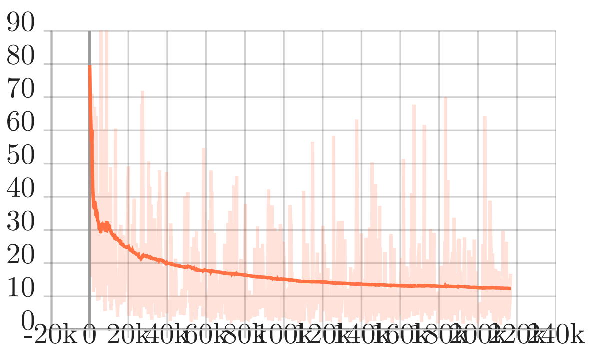
8.7 Frames of scene 32

(a) Source frame of scene 32.

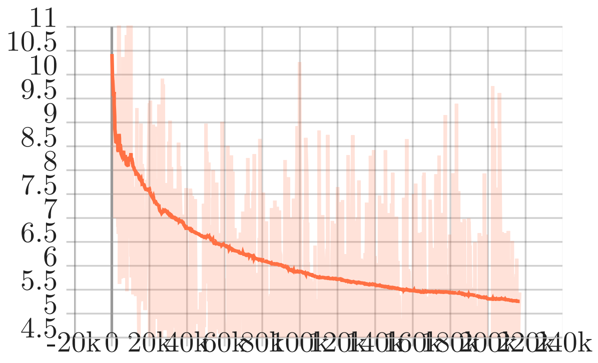
(b) Reference frame of scene 32.

(c) Source frame of scene 32.

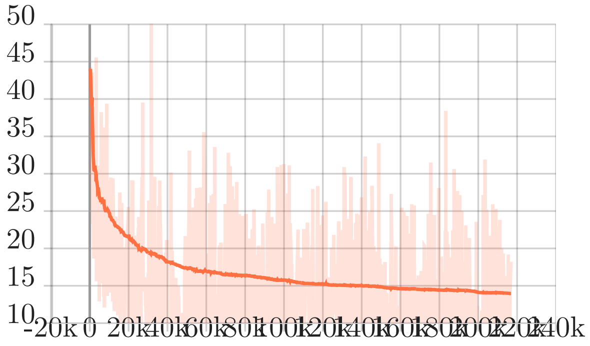
8.8 Graphs: EMVSNet Train



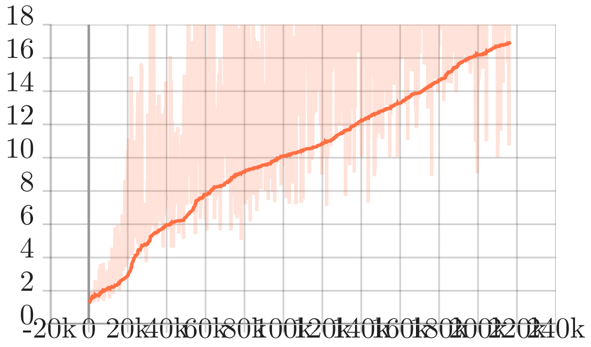
(a) Graph for absolute error.



(b) Loss in testing.

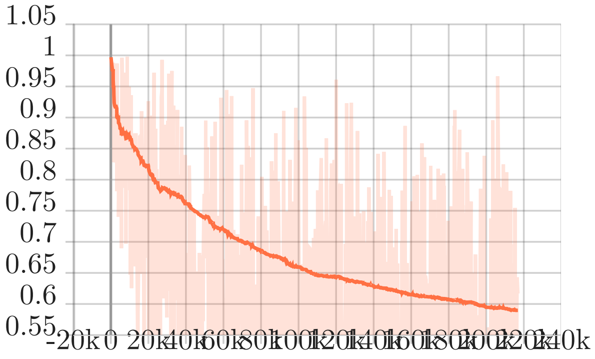


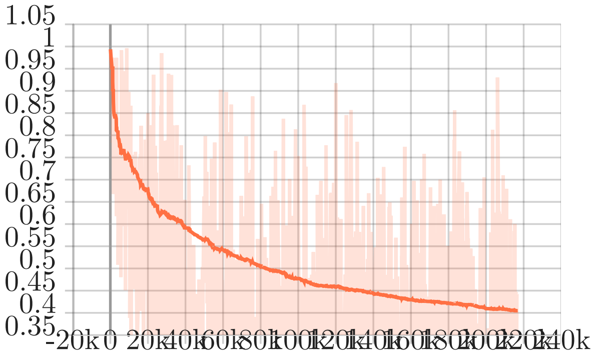
(c) Aleatoric uncertainty.

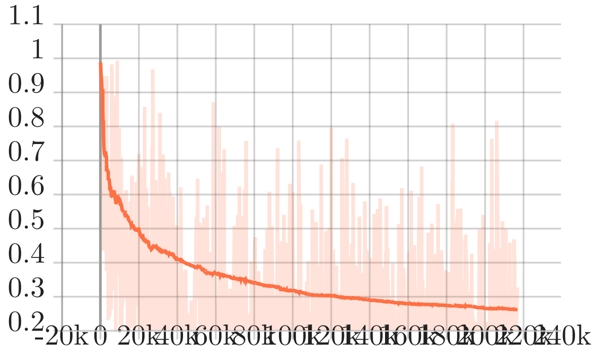


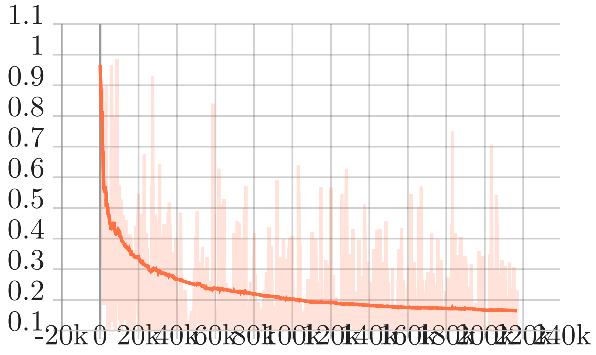
(d) Epistemic uncertainty.

Figure 8.24: All measurements given in millimeter.


(a) 2 cm error for testing data.


(b) 4 cm error for testing data.


(c) 8 cm error for testing data.


(d) 16 cm error for testing data.

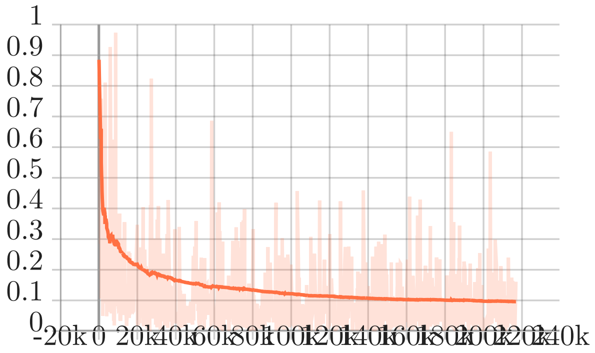
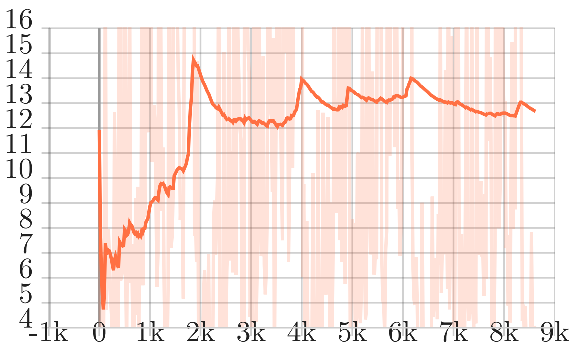
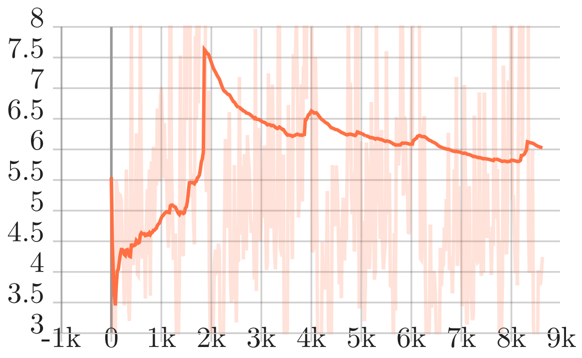

(e) 32 cm error for testing data.

Figure 8.25: The plots display the number of depth predictions that fall outside the specified error range, with all measurements provided in decimal scale.

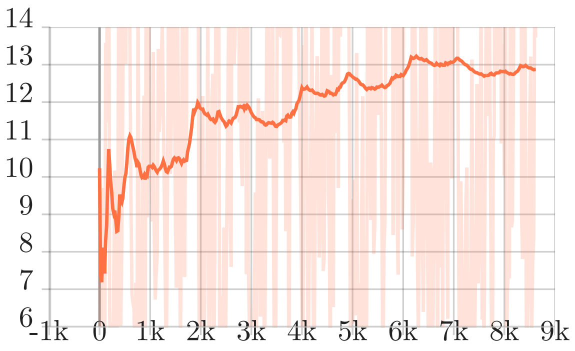
8.9 Graphs: EMVSNet Test



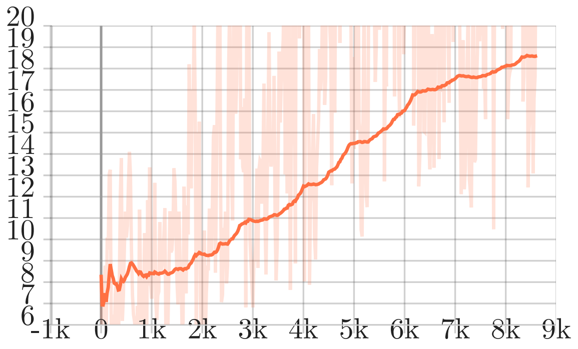
(a) Graph for absolute error.



(b) Loss in testing.

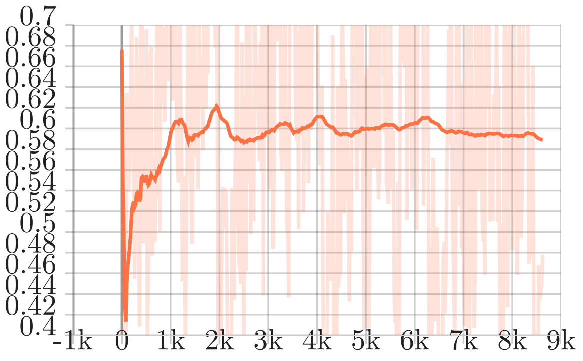


(c) Aleatoric uncertainty.

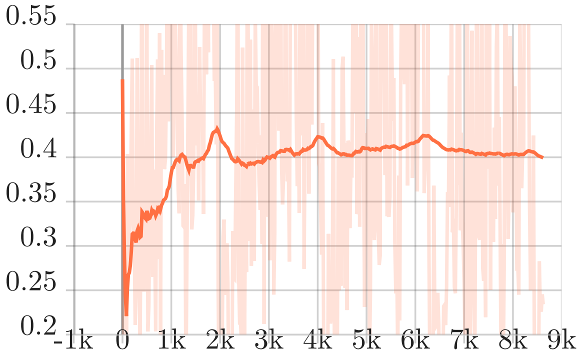


(d) Epistemic uncertainty.

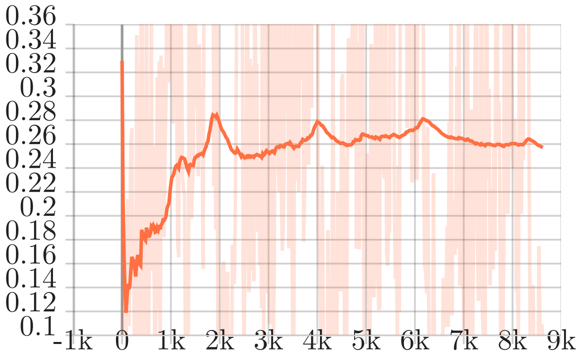
(e) All measurements given in millimeter.



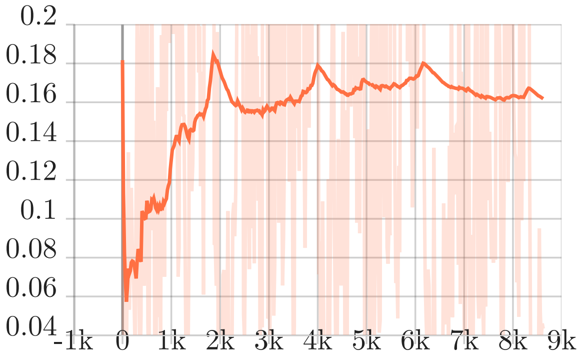
(a) 2 cm error for testing data.



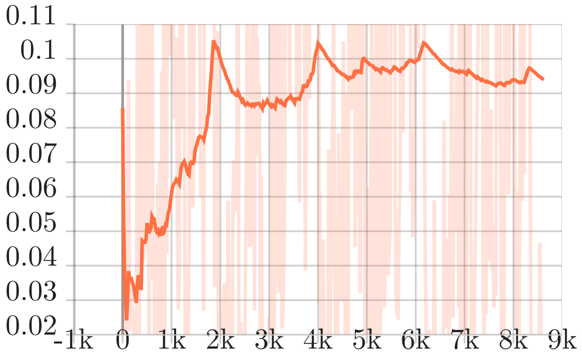
(b) 4 cm error for testing data.



(c) 8 cm error for testing data.



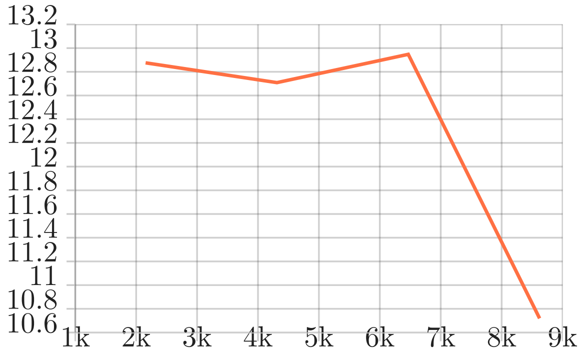
(d) 16 cm error for testing data.



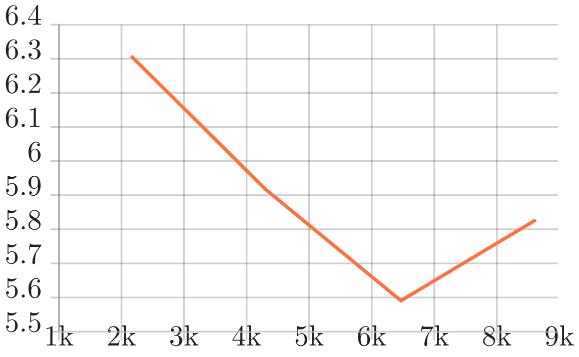
(e) 32 cm error for testing data.

Figure 8.27: The plots display the number of depth predictions that fall outside the specified error range, with all measurements provided in decimal scale.

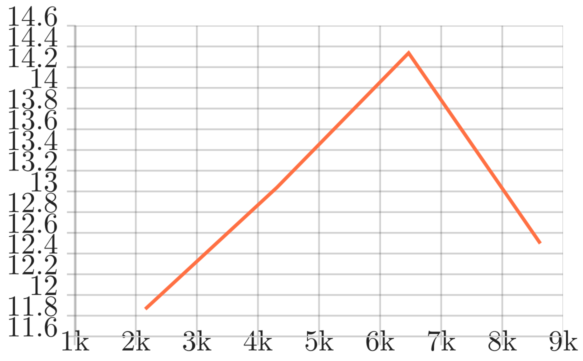
8.10 Graphs: EMVSNet Fulltest



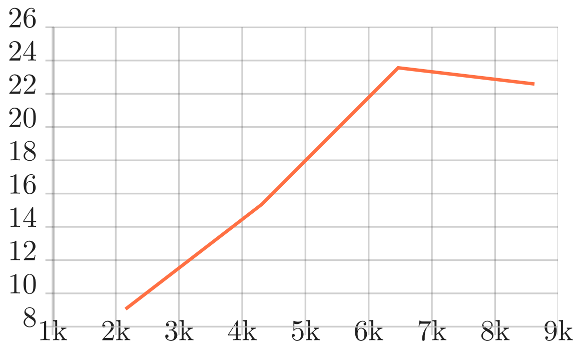
(a) Graph for absolute error.



(b) Loss in full testset.

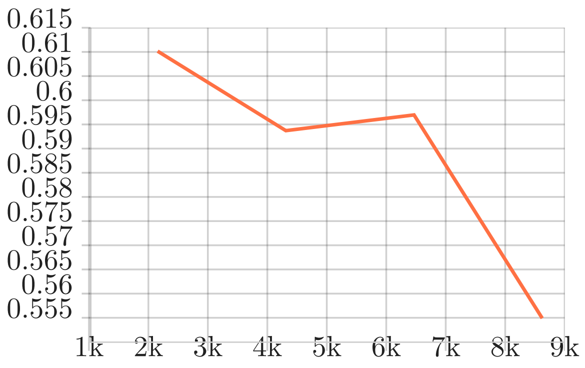


(c) Aleatoric uncertainty.

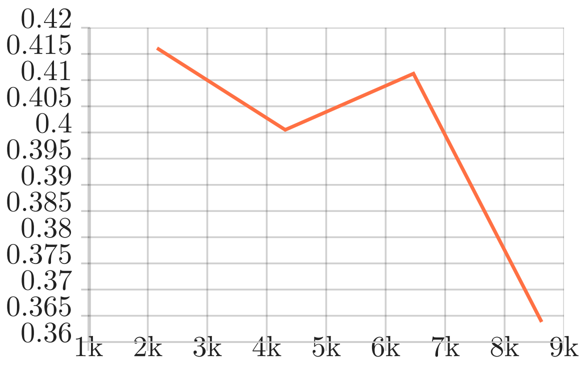


(d) Epistemic uncertainty.

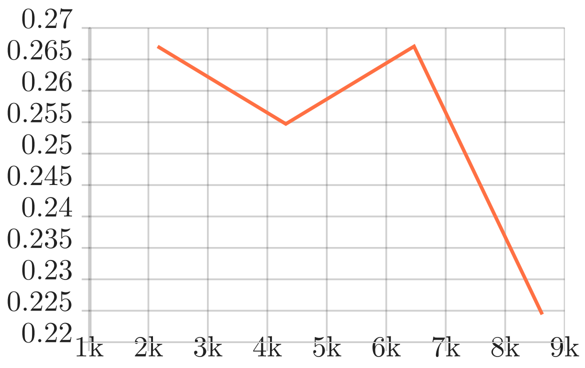
Figure 8.28: All measurements given in millimeter.



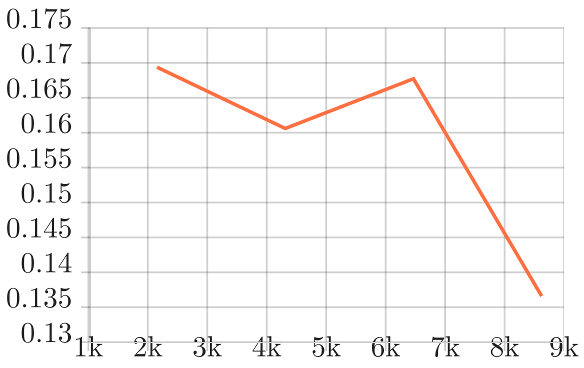
(a) 2 cm error for full testset.



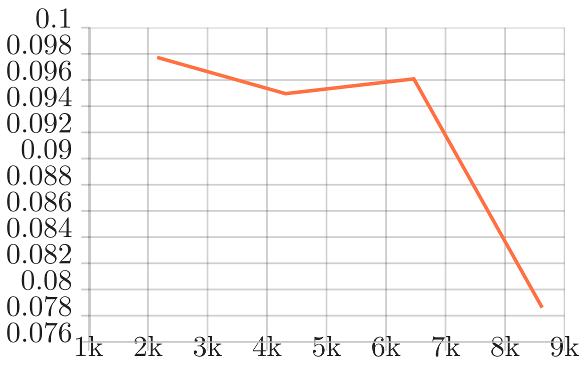
(b) 4 cm error for full testset.



(c) 8 cm error for full testset.



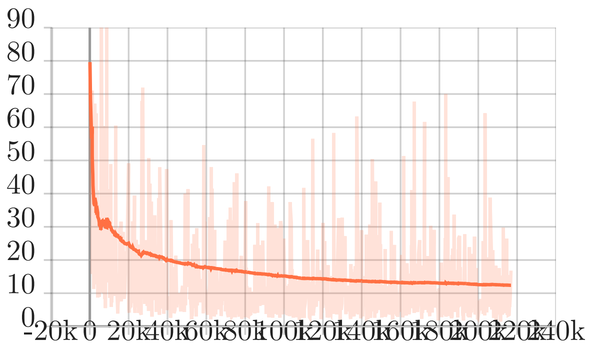
(d) 16 cm error for full testset.



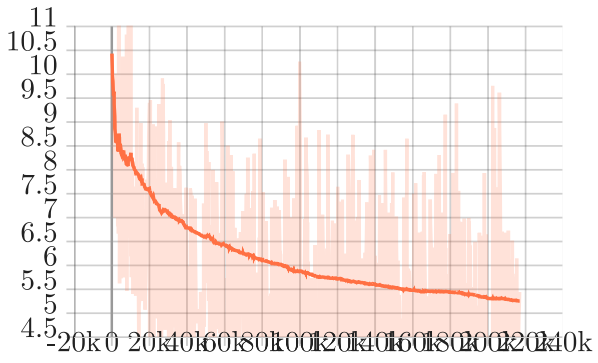
(e) 32 cm error for full testset.

Figure 8.29: The plots display the number of depth predictions that fall outside the specified error range, with all measurements provided in decimal scale.

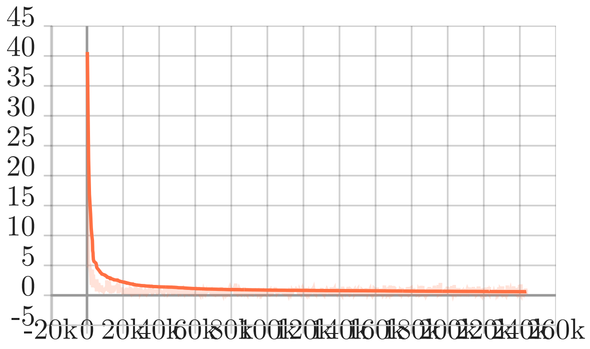
8.11 Graphs: Alternative Train



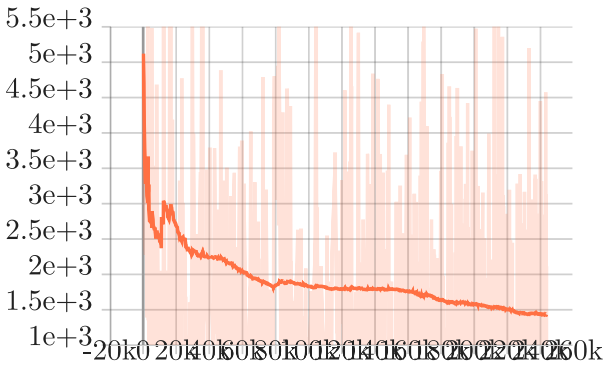
(a) Graph for absolute error.



(b) Loss in testing.

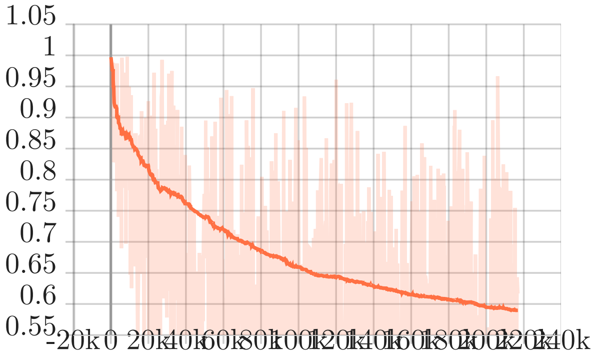


(c) Aleatoric uncertainty.

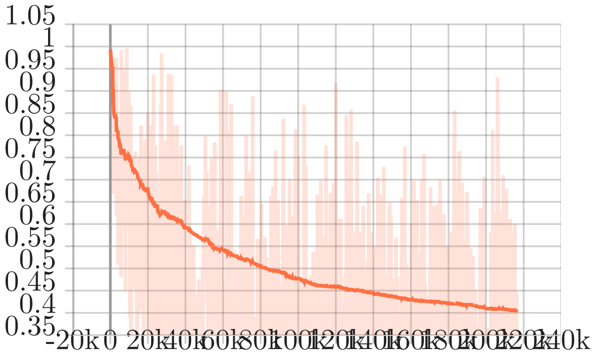


(d) Epistemic uncertainty.

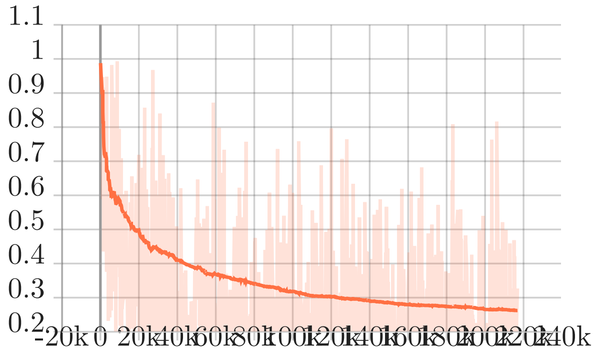
Figure 8.30: All measurements given in millimeter.



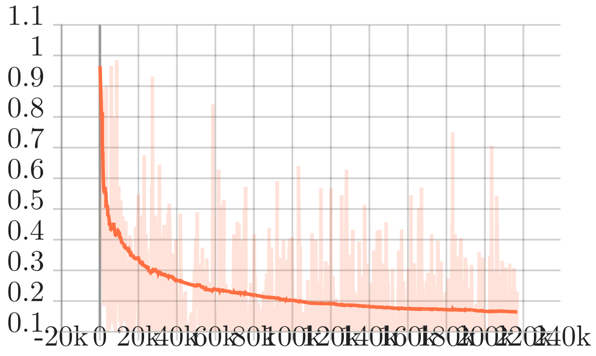
(a) 2 cm error for testing data.



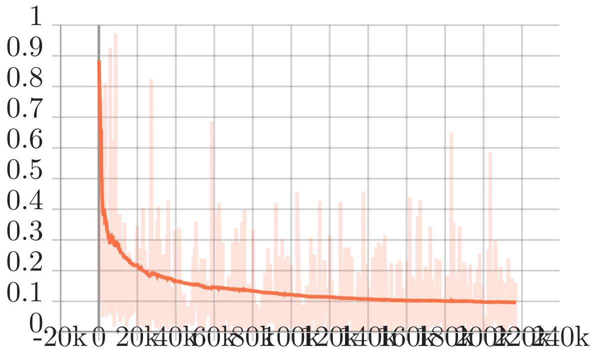
(b) 4 cm error for testing data.



(c) 8 cm error for testing data.



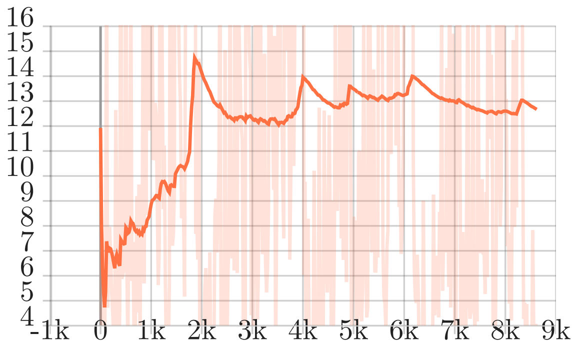
(d) 16 cm error for testing data.



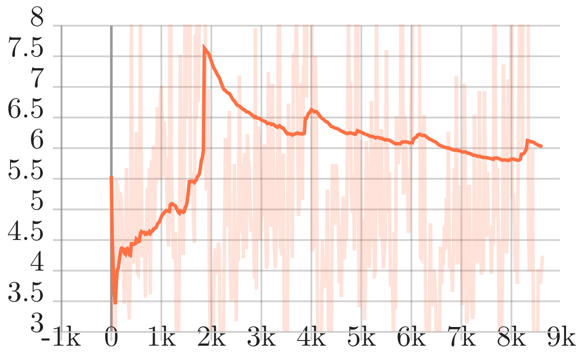
(e) 32 cm error for testing data.

Figure 8.31: The plots display the number of depth predictions that fall outside the specified error range, with all measurements provided in decimal scale.

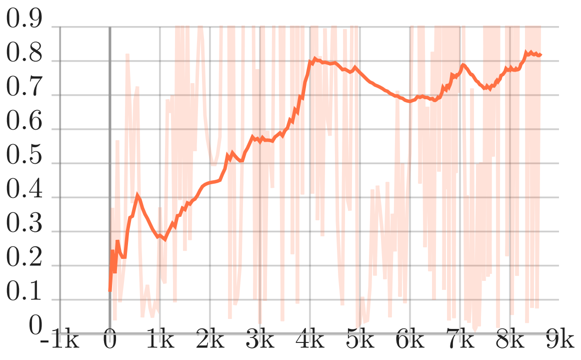
8.12 Graphs: Alternative Test



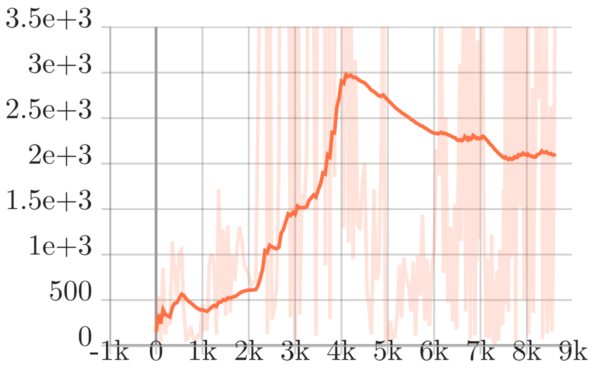
(a) Graph for absolute error.



(b) Loss in testing.

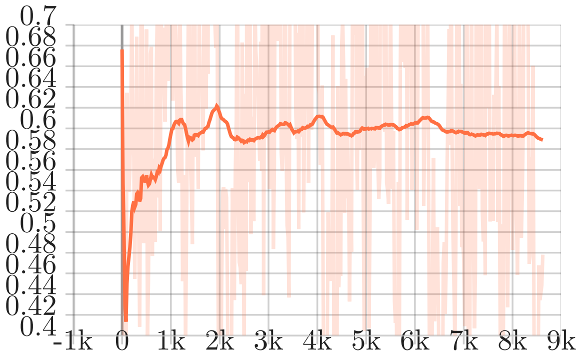


(c) Aleatoric uncertainty.

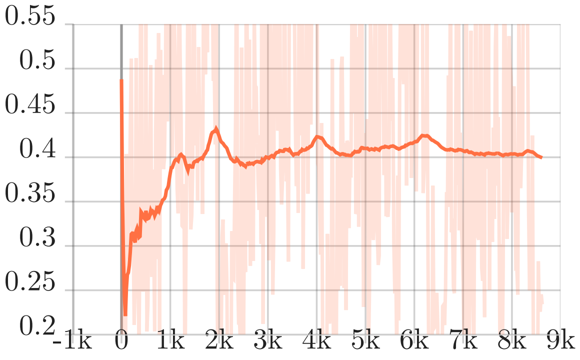


(d) Epistemic uncertainty.

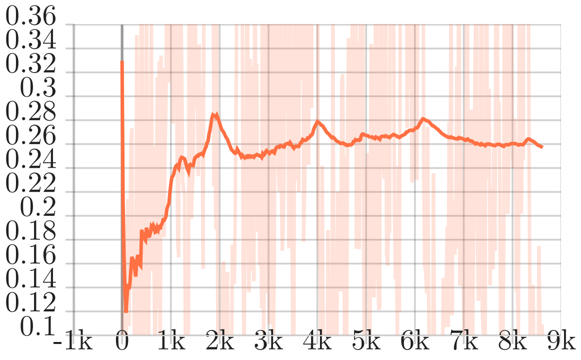
Figure 8.32: All measurements given in millimeter.



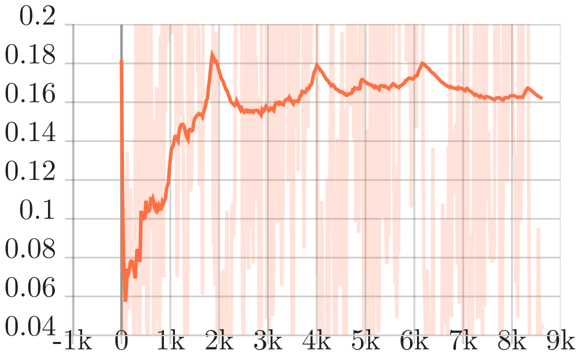
(a) 2 cm error for testing data.



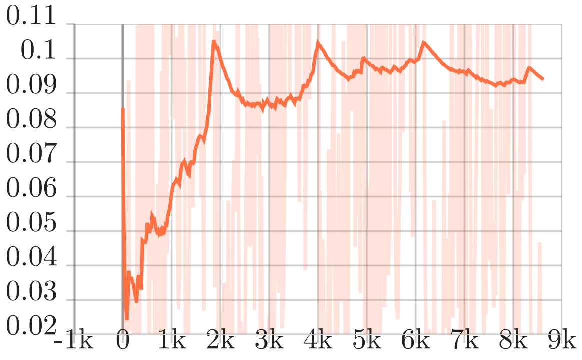
(b) 4 cm error for testing data.



(c) 8 cm error for testing data.



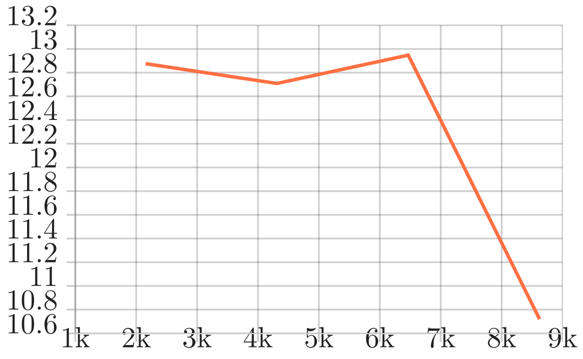
(d) 16 cm error for testing data.



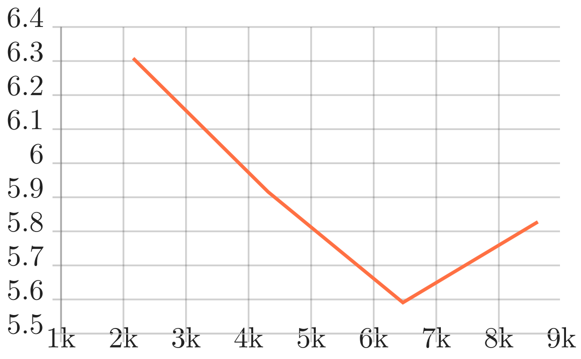
(e) 32 cm error for testing data.

Figure 8.33: The plots display the number of depth predictions that fall outside the specified error range, with all measurements provided in decimal scale.

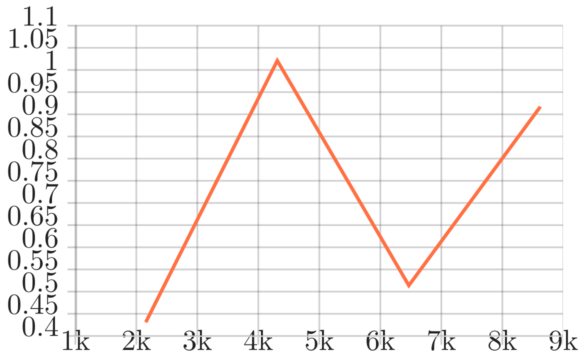
8.13 Graphs: Alternative Fulltest



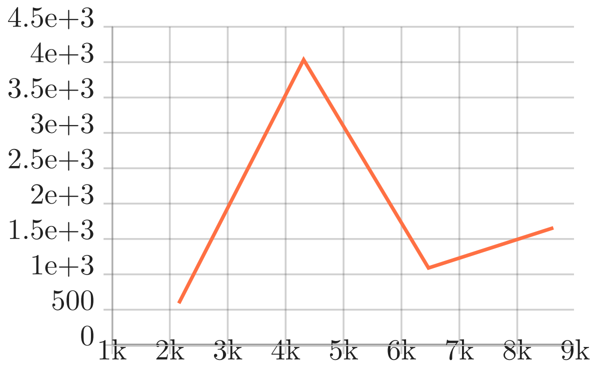
(a) Graph for absolute error.



(b) Loss in full testset.

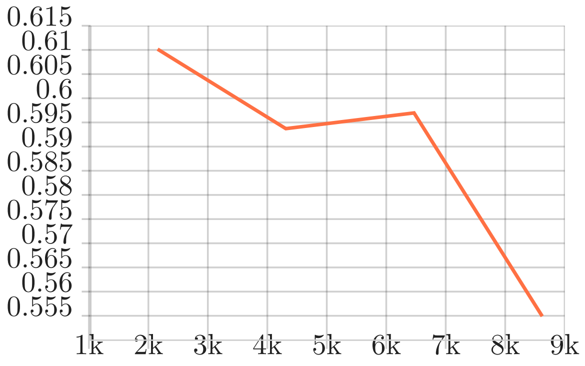


(c) Aleatoric uncertainty.

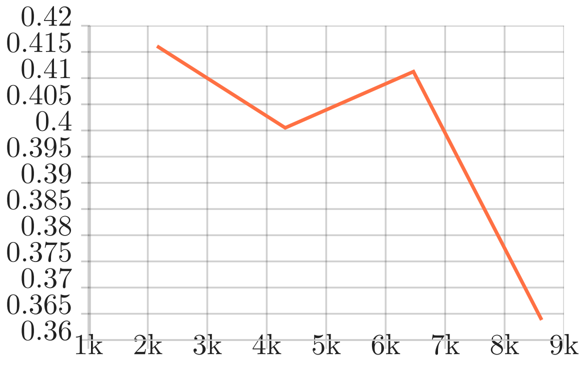


(d) Epistemic uncertainty.

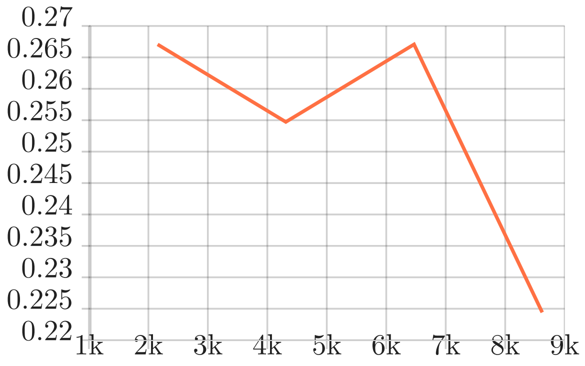
Figure 8.34: All measurements given in millimeter.



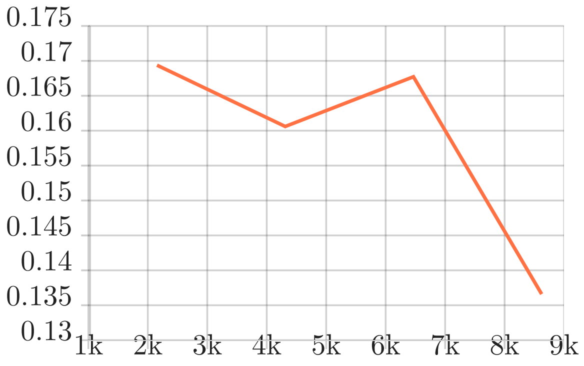
(a) 2 cm error for full testset.



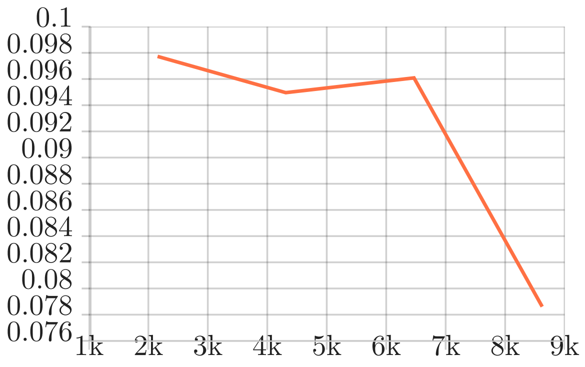
(b) 4 cm error for full testset.



(c) 8 cm error for full testset.



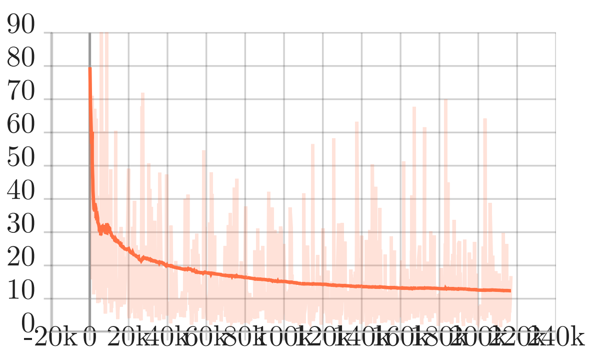
(d) 16 cm error for full testset.



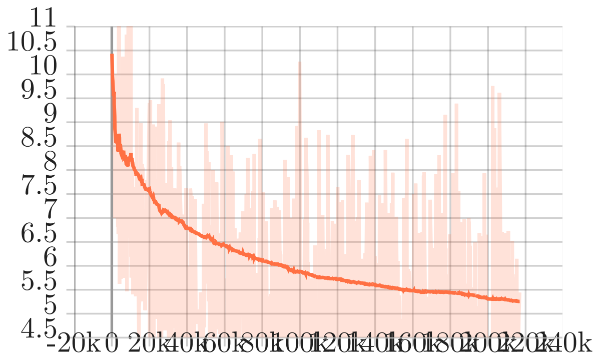
(e) 32 cm error for full testset.

Figure 8.35: The plots display the number of depth predictions that fall outside the specified error range, with all measurements provided in decimal scale.

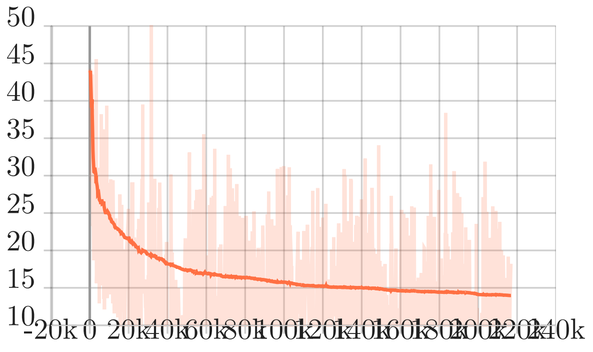
8.14 Graphs: MCD Training



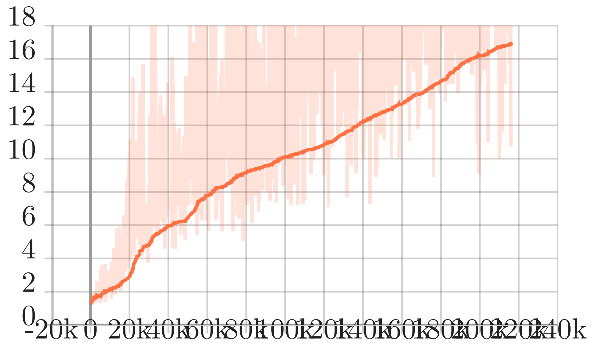
(a) Graph for absolute error.



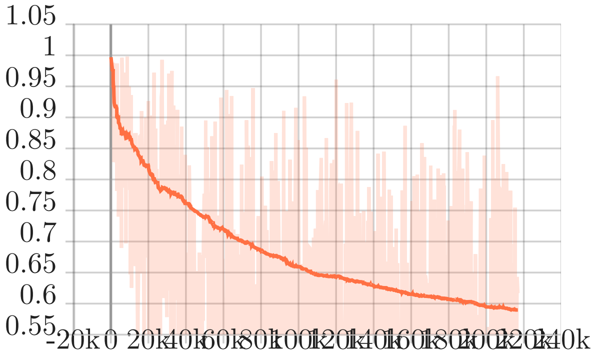
(b) Loss in training.



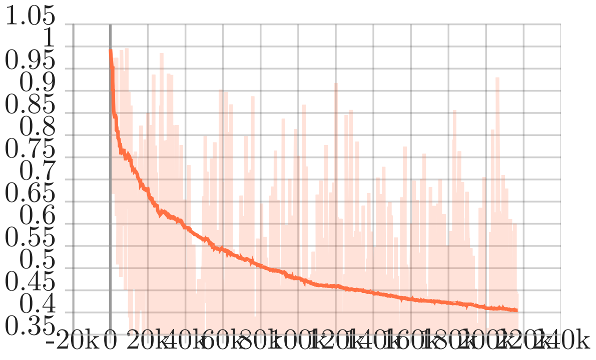
(c) Aleatoric uncertainty.



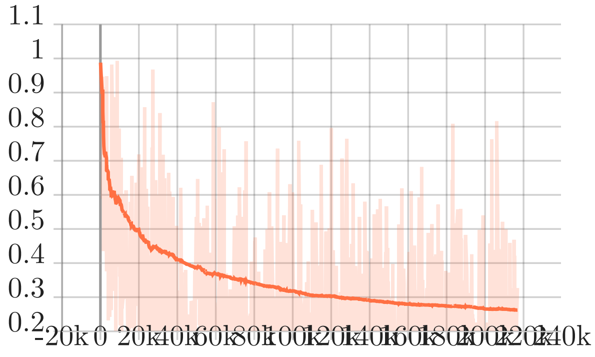
(d) Epistemic uncertainty.



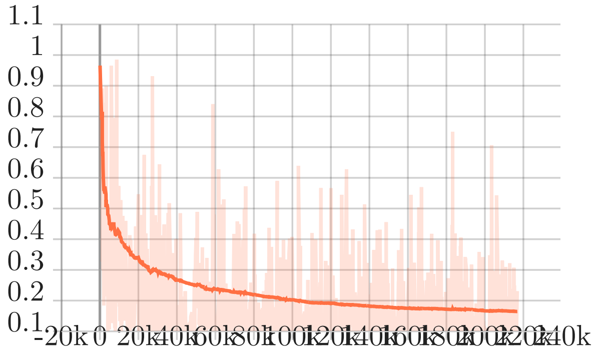
(a) 2 cm error for training data.



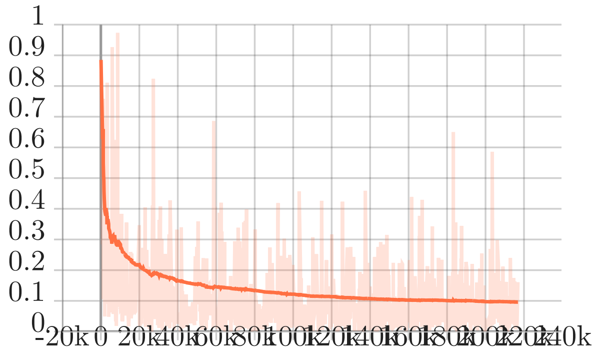
(b) 4 cm error for training data.



(c) 8 cm error for training data.



(d) 16 cm error for training data.



(e) 32 cm error for training data.

8.15 Overview: Networks

Network architectures: Multi-view Stereo

Title	Short	Year	Type	Code	Aleatoric uncertainty	Epistemic uncertainty	DTU Overall	TnT Average	Features
Uncertainty awareness with adaptive propagation for multi-view stereo	**AP-UCSNet**	2023	MVS	Yes	Yes	Yes	0,323	54,93	• Uncertainty awareness for depth hypothesis
GeoMVSNet: Learning Multi-View Stereo With Geometry Perception	**GeoMVSNet**	2023	MVS	Yes	Yes	No	0,295	65,89	
Prior depth-based multi-view stereo network for online 3D model reconstruction	**-**	2023	MVS	No	Yes	No	0,319	-	• Bayesian probability volume
Vis-MVSNet: Visibility-Aware Multi-view Stereo Network	**Vis-MVSNet**	2022	MVS	Yes	Yes	No	0,365	60,03	• Uncertainty estimation due to RobustMVS • Detailed network architecture provided
Generalized Binary Search Network for Highly-Efficient Multi-View Stereo	**GBi-Net**	2022	MVS	Yes	Yes	No	0,289	61,42	• Depth hypothesis as binary search problem
MVSFormer: Multi-View Stereo by Learning Robust Image Features and Temperature-based Depth	**MVSFormer**	2022	MVS	Yes	Yes	No	0,289	66,37	• Transformer • Self attention • Regression and classification confidence based
Multi-View Stereo Network with Attention Thin Volume	**-**	2022	MVS	No	Yes	No	0,331	55,97	• Feature-wise loss function • Thin volume attention • Attention mechanism • Low memory consumption
Uncertainty Guided Multi-View Stereo Network for Depth Estimation	**UGNet**	2022	MVS	No	Yes	No	0,332	63,12	• Uncertainty guidance • Uncertainty awareness for depth hypothesis • Detailed network layout provided
TransMVSNet: Global Context-aware Multi-view Stereo Network with Transformers	**TransMVSNet**	2022	MVS	Yes	Yes	No	0,305	63,52	• Feature-matching transformer • Adaptive receptive field module • Transformer architecture
Digging into Uncertainty in Self-supervised Multi-view Stereo	**U-MVS**	2021	MVS	Yes	Yes	Yes	0,354	57,15	• Monte Carlo dropout gives epistemic uncertainty • Does exclude points with low probability from learning
AA-RMVSNet: Adaptive Aggregation Recurrent Multi-view Stereo Network	**AA-RMVSNet**	2021	MVS	Yes	No	No	0,357	61,51	• Attention mechanism
Deep Stereo using Adaptive Thin Volume Representation with Uncertainty Awareness	**UCS-Net**	2020	MVS	Yes	Yes	No	0,344	54,83	• The depth hypothesis of each stage adapts to the uncertainties of previous per-pixel depth prediction
Deepc-mvs: Deep confidence prediction for multi-view stereo reconstruction	**Deepc-MVS**	2020	MVS	No	Yes	No	-	59,79	
Confidence-based large-scale dense multi-view stereo	**CLD-MVS**	2020	MVS	No	Yes	No	0,383	-	• Uses confidence driven interpolation
MVSNet: Depth Inference for Unstructured Multi-view Stereo	**MVSNet**	2018	MVS	Yes	No	No	0,462	43,48	• Trainable end-to-end
Depth and motion network for learning monocular stereo	**DeMoN**	2017	MVS	Yes	No	No	-	-	• Novel loss function to combine photometric consistency with a smoothness constraint
Structure-from-Motion Revisited	**COLMAP**	2016	MVS	Yes	No	No	0,532	-	
Uncertainty-Aware Deep Multi-View Photometric Stereo	**-**	2022	MVS + Stereo	No	Yes	Yes	-	-	• Monte Carlo dropout gives epistemic uncertainty •
Uncertainty Estimation for Stereo Matching Based on Evidential Deep Learning	**-**	2021	Stereo	Yes	Yes	Yes	-	-	• Normal Inverse-Gamma distribution

8.16 Overview: Datasets

Datasets: 3D Reconstruction from Multi-view Stereo

Dataset	Year	Title	Scale and environment	Special features	3D Capturing method	Facts
Skoltech3D	2023	Skoltech3D: Multi-Sensor Large-Scale Dataset for Multi-View 3D Reconstruction	• Small laboratory objects	• Different lighting setups • Various 3D capturing methods	• Smartphones • Intel RealSense • Microsoft Kinect • Industrial cameras • Structured-light scanner	• 107 scenes • 1.4 million images • 100 viewing directions • 14 lighting conditions
Giga-MVS	2022	GigaMVS: A Benchmark for Ultra-Large-Scale Gigapixel-Level 3D Reconstruction	• Ultra large scale outdoor	• Gigapixel resolution • Large urban scenes • Labeled semantics	• LiDAR	• 13 scenes • 2,9M gigapixel images
UrbanScene3D	2022	Capturing, Reconstructing, and Simulating: the UrbanScene3D Dataset	• Large scale outdoor	• Synthetic CAD models	• High precision LiDAR	• 16 scenes: • 10 synthetic and 6 real • 128k images
BlendedMVG	2020	BlendedMVS: A Large-scale Dataset for Generalized Multi-view Stereo Networks	• Small to large scale objects and laboratory objects	• Upgrade from BlendedMVS	• 3D reconstruction pipeline	• 502 scenes • 110k images
Tanks and Temples	2017	Tanks and Temples: Benchmarking Large-Scale Scene Reconstruction	• Large objects and mid-scale indoor and outdoor sites	• Large scale objects	• LiDAR	• 21 scenes • 148k images
ScanNet	2017	ScanNet: Richly-annotated 3D Reconstructions of Indoor Scenes	• Room scale indoor	• Includes semantic and instance level annotations	• Intel RealSense • Microsoft Kinect	• 1513 scenes • 2,5M images
ETH3D	2017	A Multi-View Stereo Benchmark with High-Resolution Images and Multi-Camera Videos	• Large scale indoor and outdoor	• Mixed indoor and outdoor scenes • Challenging scenarios	• LiDAR	• 24 scenes • 11k images
DTU	2016	Large-scale data for multiple-view stereopsis	• Laboratory objects	• Standard Benchmark in MVS • High resolution images • Structured setup • Various lighting conditions	• Structured Light Reconstruction	• 80 scenes • 27k images
Middlebury MVS	2006	A comparison and evaluation of multi-view stereo reconstruction algorithms	• Laboratory objects	• First widely used MVS dataset	• 3D reconstruction pipeline	• 2 scenes - temple and dino • 312 and 363 camera positions

Bibliography

[1] Dobko, Maria. CVPR 2023 summary, Jul 2023. URL https://medium.com/@dobko_m/cvpr-2023-summary-ad271d383404.

[2] Murat Sensoy, Lance Kaplan, and Melih Kandemir. Evidential deep learning to quantify classification uncertainty. *Advances in neural information processing systems*, 31, 2018.

[3] Luis Fernandez, Viviana Avila, and Luiz Gonçalves. A generic approach for error estimation of depth data from 3D sensors. 2017.

[4] Aqeel Anwar. What are Intrinsic and Extrinsic Camera Parameters in Computer Vision?, Oct 2023. Medium.

[5] Janne Heikkila and Olli Silvén. A four-step camera calibration procedure with implicit image correction. In *Proceedings of IEEE computer society conference on computer vision and pattern recognition*, pages 1106–1112. IEEE, 1997.

[6] Zhengyou Zhang. A flexible new technique for camera calibration. *IEEE Transactions on pattern analysis and machine intelligence*, 22(11):1330–1334, 2000.

[7] Stachniss, Cyrill. Camera Calibration: Zhang's Method. URL https://www.ipb.uni-bonn.de/html/teaching/photo12-2021/2021-pho1-22-Zhang-calibration.pptx.pdf.

[8] Dimo Chotrov, Z Uzunova, Y Yordanov, and S Maleshkov. Mixed-reality spatial configuration with a zed mini stereoscopic camera. In *Conf. Techno. Edu. Smart World*, volume 11, 2018.

[9] Emergent Garden. Why neural networks are able to learn almost everything., Jul 2023. URL https://www.youtube.com/watch?v=0QczhVg5HaI.

[10] Alexander Amini. MIT 6.S191 - Evidential Deep Learning. Lecture presented in Introduction to deep learning at Massachusetts Institute of Technology, 2023.

[11] Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? *Advances in neural information processing systems*, 30, 2017.

[12] Moloud Abdar, Farhad Pourpanah, Sadiq Hussain, Dana Rezazadegan, Li Liu, Mohammad Ghavamzadeh, Paul Fieguth, Xiaochun Cao, Abbas Khosravi, U Rajendra Acharya, et al. A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Information fusion*, 76:243–297, 2021.

[13] User: Jun94. Bayesian DL: 4 Types of Uncertainty. https://medium.com/jun94-devpblog/bayesian-dl-4-types-of-uncertainty-d4be46cb2dbc, 2023.

[14] Vikram Mullachery, Aniruddh Khera, and Amir Husain. Bayesian neural networks. *arXiv preprint arXiv:1801.07710*, 2018.

[15] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *International Conference on Machine Learning*, pages 1050–1059. PMLR, 2016.

[16] Rahul Rahaman et al. Uncertainty quantification and deep ensembles. *Advances in neural information processing systems*, 34:20063–20075, 2021.

[17] Glenn Shafer. *A mathematical theory of evidence*, volume 42. Princeton university press, 1976.

[18] Arthur P Dempster. Upper and lower probabilities induced by a multivalued mapping. In *Classic works of the Dempster-Shafer theory of belief functions*, pages 57–72. Springer, 2008.

[19] Dirichlet Distribution, 2023. URL https://upload.wikimedia.org/wikipedia/commons/7/74/Dirichlet.pdf.

[20] Normal-Inverse Gamma Distribution, 2023. URL https://en.wikipedia.org/wiki/Normal-inverse-gamma_distribution#/media/File:Normal-inverse-gamma.svg.

[21] Alexander Amini, Wilko Schwarting, Ava Soleimany, and Daniela Rus. Deep evidential regression. *Advances in neural information processing systems*, 33:14927–14937, 2020.

[22] Nis Meinert, Jakob Gawlikowski, and Alexander Lavin. The unreasonable effectiveness of deep evidential regression. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 9134–9142, 2023.

[23] Alex Graves and Others. Long Short Term Memory. Wikipedia, https://de.wikipedia.org/wiki/Long_short-term_memory#/media/Datei:Long_Short_Term_Memory.png.

[24] SOTA for 3D Reconstruction on DTU (as of March 18, 2024). https://paperswithcode.com/sota/3d-reconstruction-on-dtu, March 2024.

[25] Xin Yang, Yang Gao, Hongcheng Luo, Chunyuan Liao, and Kwang-Ting Cheng. Bayesian denet: Monocular depth prediction and frame-wise fusion with synchronized uncertainty. *IEEE Transactions on Multimedia*, 21(11):2701–2713, 2019.

[26] Hansheng Chen, Yuyao Huang, Wei Tian, Zhong Gao, and Lu Xiong. Monorun: Monocular 3d object detection by reconstruction and uncertainty propagation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10379–10388, 2021.

[27] Matteo Poggi, Filippo Aleotti, Fabio Tosi, and Stefano Mattoccia. On the uncertainty of self-supervised monocular depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3227–3237, 2020.

[28] Julia Hornauer and Vasileios Belagiannis. Gradient-based uncertainty for monocular depth estimation. In *European Conference on Computer Vision*, pages 613–630. Springer, 2022.

[29] Guotai Wang, Wenqi Li, Michael Aertsen, Jan Deprest, Sébastien Ourselin, and Tom Vercauteren. Aleatoric uncertainty estimation with test-time augmentation for medical image segmentation with convolutional neural networks. *Neurocomputing*, 338:34–45, 2019.

[30] Max Mehltretter. Uncertainty Estimation for End-To-End Learned Dense Stereo Matching via Probabilistic Deep Learning, 2020.

[31] Chen Wang, Xiang Wang, Jiawei Zhang, Liang Zhang, Xiao Bai, Xin Ning, Jun Zhou, and Edwin Hancock. Uncertainty estimation for stereo matching based on evidential deep learning. *Pattern Recognition*, 124:108498, 2022.

[32] Charles R Dyer. Volumetric scene reconstruction from multiple views. In *Foundations of image understanding*, pages 469–489. Springer, 2001.

[33] Diego Nehab, Szymon Rusinkiewicz, James Davis, and Ravi Ramamoorthi. Efficiently combining positions and normals for precise 3D geometry. *ACM transactions on graphics (TOG)*, 24(3):536–543, 2005.

[34] Robust Multiview Stereopsis. Accurate, Dense, and Robust Multiview Stereopsis. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, 32(8), 2010.

[35] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4104–4113, 2016.

[36] Chunge Bai, Ruijie Fu, and Xiang Gao. Colmap-PCD: An Open-source Tool for Fine Image-to-point cloud Registration. *arXiv preprint arXiv:2310.05504*, 2023.

[37] Benjamin Ummenhofer, Huizhong Zhou, Jonas Uhrig, Nikolaus Mayer, Eddy Ilg, Alexey Dosovitskiy, and Thomas Brox. Demon: Depth and motion network for learning monocular stereo. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5038–5047, 2017.

[38] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. Mvsnet: Depth inference for unstructured multi-view stereo. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 767–783, 2018.

[39] Likang Wang, Yue Gong, Xinjun Ma, Qirui Wang, Kaixuan Zhou, and Lei Chen. Is-mvsnet: importance sampling-based mvsnet. In *European Conference on Computer Vision*, pages 668–683. Springer, 2022.

[40] Yao Yao, Zixin Luo, Shiwei Li, Tianwei Shen, Tian Fang, and Long Quan. Recurrent mvsnet for high-resolution multi-view stereo depth inference. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5525–5534, 2019.

[41] Zhaoxin Li, Wangmeng Zuo, Zhaoqi Wang, and Lei Zhang. Confidence-based large-scale dense multi-view stereo. *IEEE Transactions on Image Processing*, 29:7176–7191, 2020.

[42] Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B Goldman. PatchMatch: A randomized correspondence algorithm for structural image editing. *ACM Trans. Graph.*, 28(3):24, 2009.

[43] Andreas Kuhn, Christian Sormann, Mattia Rossi, Oliver Erdler, and Friedrich Fraundorfer. Deepc-mvs: Deep confidence prediction for multi-view stereo reconstruction. In *2020 International Conference on 3D Vision (3DV)*, pages 404–413. IEEE, 2020.

[44] Shuo Cheng, Zexiang Xu, Shilin Zhu, Zhuwen Li, Li Erran Li, Ravi Ramamoorthi, and Hao Su. Deep stereo using adaptive thin volume representation with uncertainty awareness. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2524–2534, 2020.

[45] Hongbin Xu, Zhipeng Zhou, Yali Wang, Wenxiong Kang, Baigui Sun, Hao Li, and Yu Qiao. Digging into uncertainty in self-supervised multi-view stereo. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6078–6087, 2021.

[46] Zizhuang Wei, Qingtian Zhu, Chen Min, Yisong Chen, and Guoping Wang. Aa-rmvsnet: Adaptive aggregation recurrent multi-view stereo network. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6187–6196, 2021.

[47] Gary Marcus. The next decade in AI: four steps towards robust artificial intelligence. *arXiv preprint arXiv:2002.06177*, 2020.

[48] Yikang Ding, Wentao Yuan, Qingtian Zhu, Haotian Zhang, Xiangyue Liu, Yuanjiang Wang, and Xiao Liu. Transmvsnet: Global context-aware multi-view stereo network with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8585–8594, 2022.

[49] Chenjie Cao, Xinlin Ren, and Yanwei Fu. MVSFormer: Multi-View Stereo by Learning Robust Image Features and Temperature-based Depth. *Transactions on Machine Learning Research*, 2022.

[50] Wanjuan Su, Qingshan Xu, and Wenbing Tao. Uncertainty guided multi-view stereo network for depth estimation. *IEEE Transactions on Circuits and Systems for Video Technology*, 32(11):7796–7808, 2022.

[51] Zihang Wan, Chao Xu, Jing Hu, Jian Xiao, Zhaopeng Meng, and Jitai Chen. Multi-View Stereo Network with attention thin volume. In *Pacific Rim International Conference on Artificial Intelligence*, pages 410–423. Springer, 2022.

[52] Zhenxing Mi, Chang Di, and Dan Xu. Generalized binary search network for highly-efficient multi-view stereo. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12991–13000, 2022.

[53] Jingyang Zhang, Shiwei Li, Zixin Luo, Tian Fang, and Yao Yao. Vis-mvsnet: Visibility-aware multi-view stereo network. *International Journal of Computer Vision*, 131(1):199–214, 2023.

[54] Soohwan Song, Khang Giang Truong, Daekyum Kim, and Sungho Jo. Prior depth-based multi-view stereo network for online 3D model reconstruction. *Pattern Recognition*, 136: 109198, 2023.

[55] Zhe Zhang, Rui Peng, Yuxi Hu, and Ronggang Wang. GeoMVSNet: Learning Multi-View Stereo With Geometry Perception. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21508–21518, 2023.

[56] Jinguang Chen, Zonghua Yu, Lili Ma, and Kaibing Zhang. Uncertainty awareness with adaptive propagation for multi-view stereo. *Applied Intelligence*, 53(21):26230–26239, 2023.

[57] Ziwei Liao and Steven L Waslander. Multi-view 3D Object Reconstruction and Uncertainty Modelling with Neural Shape Prior. *arXiv preprint arXiv:2306.11739*, 2023.

[58] Dennis Thomas Ulmer. A survey on evidential deep learning for single-pass uncertainty estimation. 2021.

[59] Murat Sensoy, Lance Kaplan, Federico Cerutti, and Maryam Saleki. Uncertainty-aware deep classifiers using generative models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 5620–5627, 2020.

[60] Andrey Malinin and Mark Gales. Predictive uncertainty estimation via prior networks. *Advances in neural information processing systems*, 31, 2018.

[61] Xujiang Zhao, Feng Chen, Shu Hu, and Jin-Hee Cho. Uncertainty aware semi-supervised learning on graph data. *Advances in Neural Information Processing Systems*, 33:12827–12836, 2020.

[62] Jay Nandy, Wynne Hsu, and Mong Li Lee. Towards maximizing the representation gap between in-domain & out-of-distribution examples. *Advances in neural information processing systems*, 33:9239–9250, 2020.

[63] Andrey Malinin, Sergey Chervontsev, Ivan Provilkov, and Mark Gales. Regression prior networks. *arXiv preprint arXiv:2006.11590*, 2020.

[64] Frederik Boe Hüttel, Filipe Rodrigues, and Francisco Câmara Pereira. Deep Evidential Learning for Bayesian Quantile Regression. *arXiv preprint arXiv:2308.10650*, 2023.

[65] Wentao Bao, Qi Yu, and Yu Kong. Evidential deep learning for open set action recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13349–13358, 2021.

[66] Deep Shankar Pandey and Qi Yu. Multidimensional belief quantification for label-efficient meta-learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14391–14400, 2022.

[67] Patrick Hemmer, Niklas Kühl, and Jakob Schöffer. Deal: Deep evidential active learning for image classification. *Deep Learning Applications, Volume 3*, pages 171–192, 2022.

[68] Ava P Soleimany, Alexander Amini, Samuel Goldman, Daniela Rus, Sangeeta N Bhatia, and Connor W Coley. Evidential deep learning for guided molecular property prediction and discovery. *ACS central science*, 7(8):1356–1367, 2021.

[69] Jieming Lou, Weide Liu, Zhuo Chen, Fayao Liu, and Jun Cheng. Elfnet: Evidential local-global fusion for stereo matching. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 17784–17793, 2023.

[70] Eduardo Aguilar, Bogdan Raducanu, Petia Radeva, and Joost Van de Weijer. Continual Evidential Deep Learning for Out-of-Distribution Detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3444–3454, 2023.

[71] Senqi Cao and Zhongfei Zhang. Deep hybrid models for out-of-distribution detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4733–4743, 2022.

[72] Nis Meinert and Alexander Lavin. Multivariate deep evidential regression. *arXiv preprint arXiv:2104.06135*, 2021.

[73] Akihito Nagahama. Learning and predicting the unknown class using evidential deep learning. *Scientific Reports*, 13(1):14904, 2023.

[74] Danruo Deng, Guangyong Chen, Yang Yu, Furui Liu, and Pheng-Ann Heng. Uncertainty estimation by fisher information-based evidential deep learning. *arXiv preprint arXiv:2303.02045*, 2023.

[75] Steven M Seitz, Brian Curless, James Diebel, Daniel Scharstein, and Richard Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 1, pages 519–528. IEEE, 2006.

[76] Henrik Aanæs, Rasmus Ramsbøl Jensen, George Vogiatzis, Engin Tola, and Anders Bjorholm Dahl. Large-scale data for multiple-view stereopsis. *International Journal of Computer Vision*, 120:153–168, 2016.

[77] Thomas Schops, Johannes L Schonberger, Silvano Galliani, Torsten Sattler, Konrad Schindler, Marc Pollefeys, and Andreas Geiger. 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*, pages 3260–3269, 2017.

[78] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5828–5839, 2017.

[79] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics (ToG)*, 36 (4):1–13, 2017.

[80] Yao Yao, Zixin Luo, Shiwei Li, Jingyang Zhang, Yufan Ren, Lei Zhou, Tian Fang, and Long Quan. BlendedMVS: A Large-scale Dataset for Generalized Multi-view Stereo Networks. *Computer Vision and Pattern Recognition (CVPR)*, 2020.

[81] Liqiang Lin, Yilin Liu, Yue Hu, Xingguang Yan, Ke Xie, and Hui Huang. Capturing, reconstructing, and simulating: the urbanscene3d dataset. In *European Conference on Computer Vision*, pages 93–109. Springer, 2022.

[82] Jianing Zhang, Jinzhi Zhang, Shi Mao, Mengqi Ji, Guangyu Wang, Zequn Chen, Tian Zhang, Xiaoyun Yuan, Qionghai Dai, and Lu Fang. GigaMVS: a benchmark for ultra-large-scale gigapixel-level 3D reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(11):7534–7550, 2021.

[83] Oleg Voynov, Gleb Bobrovskikh, Pavel Karpyshev, Saveliy Galochkin, Andrei-Timotei Ardelean, Arseniy Bozhenko, Ekaterina Karmanova, Pavel Kopanev, Yaroslav Labutin-Rymsho, Ruslan Rakhimov, et al. Multi-sensor large-scale dataset for multi-view 3D reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21392–21403, 2023.

[84] Huan Ma, Zongbo Han, Changqing Zhang, Huazhu Fu, Joey Tianyi Zhou, and Qinghua Hu. Trustworthy multimodal regression with mixture of normal-inverse gamma distributions. *Advances in Neural Information Processing Systems*, 34:6881–6893, 2021.

[85] Xue Li, Wei Shen, and Denis Charles. TEDL: A Two-stage Evidential Deep Learning Method for Classification Uncertainty Quantification. *arXiv preprint arXiv:2209.05522*, 2022.

[86] Michael Goesele, Brian Curless, and Steven M Seitz. Multi-view stereo revisited. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 2402–2409. IEEE, 2006.

Declaration of Authorship

I declare that this thesis is the result of independent research conducted by me under the guidance of my supervisor. It does not contain the results of any other scientific research that has been published or written by any other individuals or groups, except for those already cited in the thesis. Furthermore, I state that this work in the same or a similar form has not been submitted to an examination authority.

Hille, 14. August 2024 [illegible]

Place, Date Signature