Czech Technical University in Prague
Faculty of Electrical Engineering
Department of Cybernetics

# Visual Prediction of Surface Properties in Complex Off-Road Terrain

Master Thesis

*Bc. David Kraus*

Study programme: Open Informatics
Specialisation: Computer Vision and Image Processing
Supervisor: Ing. Jan Čech, Ph.D.

Prague, May 2023

**Thesis Supervisor:**
Ing. Jan Čech, Ph.D.

# MASTER‘S THESIS ASSIGNMENT

## I. Personal and study details

Student's name: **Kraus David** Personal ID number: **483272**

Faculty / Institute: **Faculty of Electrical Engineering**

Department / Institute: **Department of Cybernetics**

Study program: **Open Informatics**

Specialisation: **Computer Vision and Image Processing**

## II. Master’s thesis details

Master’s thesis title in English:

**Visual Prediction of Surface Properties in Complex Off-Road Terrain**

Master’s thesis title in Czech:

**Vizuální predikce vlastností povrchu ve složitém off-road terénu**

Guidelines:

Previously, we studied the estimation of surface properties in front of the vehicle in terms of its roughness and friction properties [1,2]. However, the surface was assumed to be approximately planar. The problem with this approximation becomes apparent in the case of more complex off-road terrain, where non-planar effects can cause a significant roll/tilt of the vehicle or even a collision with an obstacle. Stereo vision enables the terrain in front of the vehicle to be reconstructed. Design a method that estimates the terrain map in front of the vehicle in terms of the quality of the traversability, from smooth and trouble-free passage to inadmissible passage due to an obstacle. Start from a local estimate of the surface normals, and then adjust the predictor by estimating surface properties from driving data using a sub-scale vehicle platform. Training a monocular predictor is preferred.
Make a literature survey, design a method architecture, collect the data, implement and train the model, evaluate on independent test set.

Bibliography / sources:

[1] J. Cech, T. Hanis, A. Konopisky, T. Rurtle, J. Svancar, T. Twardzik. Self-Supervised Learning of Camera-based Drivable Surface Roughness. In Proc. IEEE Intelligent Vehicles Symposium, 2021.
[2] D. Vosahlik, J. Cech, T. Hanis, A. Konopisky, T. Rurtle, J. Svancar, T. Twardzik. Self-Supervised Learning of Camera-based Drivable Surface Friction. In Proc. IEEE International Conference on Intelligent Transportation Systems, 2021.
[3] Clement Godard, Oisin Mac Aodha, Gabriel J. Brostow. Unsupervised Monocular Depth Estimation with Left-Right Consistency, In Proc. CVPR, 2017.
[4] Adam Konopisky. Surface Properties Prediction from Images Using Self-supervised Learning. Master's Thesis, Czech Technical University, FEE, 2022.

Name and workplace of master’s thesis supervisor:

**Ing. Jan Čech, Ph.D. Visual Recognition Group FEE**

Name and workplace of second master’s thesis supervisor or consultant:

Date of master’s thesis assignment: **06.02.2023** Deadline for master's thesis submission: **26.05.2023**

Assignment valid until: **22.09.2024**

| Ing. Jan Čech, Ph.D. | prof. Ing. Tomáš Svoboda, Ph.D. | prof. Mgr. Petr Páta, Ph.D. |
|---|---|---|
| Supervisor’s signature | Head of department’s signature | Dean’s signature |

## III. Assignment receipt

The student acknowledges that the master's thesis is an individual work. The student must produce his thesis without the assistance of others, with the exception of provided consultations. Within the master's thesis, the author must state the names of consultants and include a list of references.

| . | |
|---|---|
| Date of assignment receipt | Student's signature |

# Declaration

I declare that the presented work was developed independently and that I have listed all sources of information used within it in accordance with the methodical instructions for observing the ethical principles in the preparation of university theses.

Prague, May 26, 2023 .....

Bc. David Kraus

# Abstract

Visual recognition of the environment is essential for autonomous vehicle navigation through off-road terrain. This work relies on combining visual geometry estimation and scene understanding to detect off-the-plane obstacles. The network is trained to distinguish hard obstacles that should be avoided to prevent a collision from phenomena that appear off-the-plane but are soft and traversable, e.g., tall grass. The thesis focuses on the camera-only setup. The geometry is estimated either with the binocular stereo camera or with a single camera and convolutional neural network (CNN), which is trained to estimate the 3D geometry. Another CNN is trained to perform the semantic segmentation of the scene. We implemented four deep learning methods and one baseline method based on the 3D geometry only. The best-performing method was using the combination of the 3D geometry obtained from the binocular camera with supervised deep learning, and it reached an F1 score of 98.2 for pixel-wise binary classification, tested on our dataset of forest scenes.

**Keywords:** Autonomous driving, Off-road terrain, Depth map, CNN, Semantic segmentation, UNet

# Abstrakt

Vizuální rozpoznávání prostředí je nezbytné pro autonomní navigaci vozidla v terénu. Tato práce kombinuje vizuální odhad geometre a porozumění scéně za účelem detekce překážek mimo rovinu. Neuronová síť je natrénována k rozlišování pevných překážek, kterým je nutné se vyhnout, aby nedošlo ke kolizi, od překážek, které leží mimo rovinu, ale jsou průjezdné, např. vysoká tráva. Práce se zaměřuje na vozidlo vybavené pouze kamerou. Geometrie se odhaduje buď pomocí binokulární stereo kamery, nebo pomocí jediné kamery a konvoluční neuronové sítě (CNN), která je natrénována k odhadu 3D geometrie. Další CNN je natrénována k provedení sémantické segmentace scény. Byly implementovány čtyři metody hlubokého učení a jedna metoda založená pouze na 3D

geometrii. Nejlepších výsledků dosahovala metoda využívající kombinaci 3D geometrie získané z binokulární kamery se supervizovaným hlubokým učením, která dosáhla F1 skóre 98,2 pro binární klasifikaci, testovanou na našem datasetu lesních scén.

**Keywords:**

Autonomní řízení, Off-roadový terén, Hloubková mapa, CNN, Sémantická segmentace, UNet

# Acknowledgements

I would like to thank my supervisor, Ing. Jan Čech, Ph.D., for his advice, guidance, and patience.

I would also like to thank the members of the Tomi2 project, namely Doc. Ing. Tomáš Haniš, PhD., Ing. David Vošahlík, Ing. Marek Boháč and Ing. Jan Švancar.

My thanks also belong to Mgr. Irena Dušáková for lending the necessary equipment.

# Contents

# Chapter 1

# Introduction

Vehicle navigation through a terrain relies heavily on the ability to visually recognize the environment, which is achieved by combining visual geometry estimation and scene understanding.

One aspect of this recognition is the estimation of obstacles and the terrain's shape, which typically depends on binocular vision. Humans can estimate depth using binocular vision, and computers can leverage binocular stereo cameras. However, humans can also use their prior knowledge of image-to-depth mapping to compensate for the lack of binocular vision. Similarly, convolutional neural networks (CNNs) can be trained to estimate depth from a single image, as described in [1].

While visual geometry estimation is crucial for traversing terrain, it may not always be sufficient. Certain obstacles, such as high grass or fallen leaves, can be easily traversed despite their size, while others, like puddles, cannot be recognized from the scene's geometry alone. Therefore, scene understanding is needed to enable a vehicle to navigate through such terrain safely.

This thesis aims to perform semantic segmentation of camera images using various segmentation methods. The segmentation approach will categorize the scene into 'safe' and 'dangerous' classes. The 'safe' class consists of flat, traversable surfaces and soft obstacles, such as tall grass or pile of leaves. On the other hand, the 'dangerous' class consists of solid barriers, such as trees, that should be avoided. This segmentation subsequently enables the navigation of the test vehicle through the terrain. The methods utilized differ in whether they use the binocular camera and the specific deep learning techniques applied.

# Chapter 2

# Related Work

The navigation of autonomous vehicles with a camera-only setup is a well-known problem. The methods can be divided into on- or off-road navigation, depending on whether the vehicle operates on paved or off-road terrain.

A method for on-road route recommendation based on road recognition from the monocular camera is proposed by J.-M. Dai et al. [2]. Articles [3] and [4] use self-supervised learning to predict the properties of the on- or semi-off-road terrain. The self-supervision in these methods is based on the sensors mounted on the vehicle.

The off-road terrain typically requires information about the geometry. The traditional method of estimating the geometry with the camera-only setup uses the binocular stereo camera. Eppenberger et al. [5] describe a method for classifying static obstacles and tracking the dynamic object using the point cloud from the binocular camera.

Monodepth [1] is a method of estimating the depth from a single image based on deep learning. Tektonidis et al. propose using the self-supervised deep learning method of depth estimation for the navigation of autonomous vehicles [6].

Besides the camera-only setup, Light Detection And Ranging (LIDAR) is another popular environmental monitoring method. The work [7] combines the LIDAR with deep learning for off-road driveable area extraction.

The method introduced in [8] describes a NN architecture TerrainNet, which jointly predicts terrain semantics and geometry from the input images and depth maps. The geometry consists of three layers: Ground Min Elevation represents the height of the ground, Ground Max Elevation represents the height of potential obstacles lying on the

ground, and Ceiling Elevation represents the height of the potential ceiling, such as the tree branches. The vehicle is equipped with four cameras, each pointed in a different direction. This configuration allows the vehicle to capture visual information from the front, sides, and rear. The LIDAR obtains the ground truth data for training but is not used during deployment.

Schmid et al. [9] propose a self-supervised approach for traversability prediction. The self-supervision is based on the previous rides. The trajectory is projected to the camera image. The NN architecture is based on an autoencoder trained to reconstruct only the previously safely traversed terrain. This reconstruction is subsequently used to identify the traversable terrain.

Article [10] describes a multiple-camera setup for vision-based simultaneous localization and mapping (vSLAM) based on panoramic cameras.

In addition to considering the environmental factors, successful navigation also requires accounting for the physical characteristics of the vehicle. While traditional wheel-based vehicles are commonly used, more specialized vehicles like belt vehicles are frequently employed in challenging terrains. Zimmermann et al. [11] explicitly address the control of robot morphology to enhance obstacle traversal capabilities.

# Chapter 3

# Methods

This chapter presents several methods which eventually lead to the semantic segmentation of the images.

In Section 3.1, we present two approaches for generating a depth map: one using a binocular stereo camera and another using a monocular camera and deep learning.

Section 3.2 covers the conversion of the depth map into a point cloud and its subsequent processing. This process provides the geometric segmentation of the image, which is based only on the scene geometry. We use this segmentation as an input for a more sophisticated semantic segmentation. It is also used for the dataset annotation.

In Section 3.3, we introduce several approaches to semantic segmentation based on deep learning.

## 3.1 Depth map

A depth map is a digital representation of the scene's three-dimensional depth or distance information. Each pixel or point in the image is associated with a depth value representing the distance from the camera or observer to the object in the scene.

We introduce two methods of obtaining the depth map. The first is based on a binocular camera, while the second uses a single camera and deep learning.

### 3.1.1 Depth map estimation using the binocular camera

The first method for depth map estimation utilizes the binocular (stereo) camera setup, which involves capturing images from two cameras placed at a certain distance apart (a baseline), mimicking the human eyes' stereo vision. The two cameras capture images of the same scene from slightly different viewpoints, allowing for the calculation of depth information based on the disparity between corresponding points in the two images.

One of the main advantages of the binocular camera for estimating depth is that it can do so without prior knowledge. This is in contrast with the deep learning method. It can also provide more accurate depth information, as it estimates the distance from the disparity.

One significant disadvantage is the need for a calibrated binocular camera setup. Furthermore, the binocular camera has a limited range of precise depth estimation. Specifically, the depth estimation accuracy decreases as the distance between the camera and the object of interest grows. Beyond a certain distance, the stereo disparity between the left and right images becomes too small to provide reliable depth estimation. This distance depends on the baseline of the binocular camera (the distance between the two cameras). A more extended baseline increases the range, but it also intensifies the occlusions. However, in a forest environment, where the objects of interest are located within a few meters, the limited range of depth estimation is typically not an issue.

Another limitation is that it may not perform well on surfaces with low texture, such as flat or featureless surfaces. However, this limitation may also be a minor issue in the forest environment, as there are likely enough visual cues for accurate depth estimation.

### 3.1.2 Depth map estimation using the monocular depth and deep learning approach

The other approach for generating the depth map is based on a single camera and deep learning. We train the convolutional neural network (CNN) to predict the depth for each pixel in a single image.

The labels for our dataset are obtained using the binocular camera setup, which provides pixel-wise depth information for each training image. We are using 4000 images for the training and another 500 for the validation.

The CNN is trained on this labeled dataset using the mean absolute logarithm error (MALE) loss function [12]. We use this loss function because the same absolute error's impact grows with the proximity to the camera. For example, the difference between 9 and 10 meters is less significant than between 1 and 2 meters for our purpose. Using the MALE makes the error more proportional to the distance.

$$MALE(x, y) = \frac{1}{T}\sum_{t=1}^{T}|\log(x_t) - \log(y_t)| = \frac{1}{T}\sum_{t=1}^{T}|\log\left(\frac{x_t}{y_t}\right)|$$

Figure 3.1 compares the signed error and the signed logarithm error on the outputs of the trained CNN.

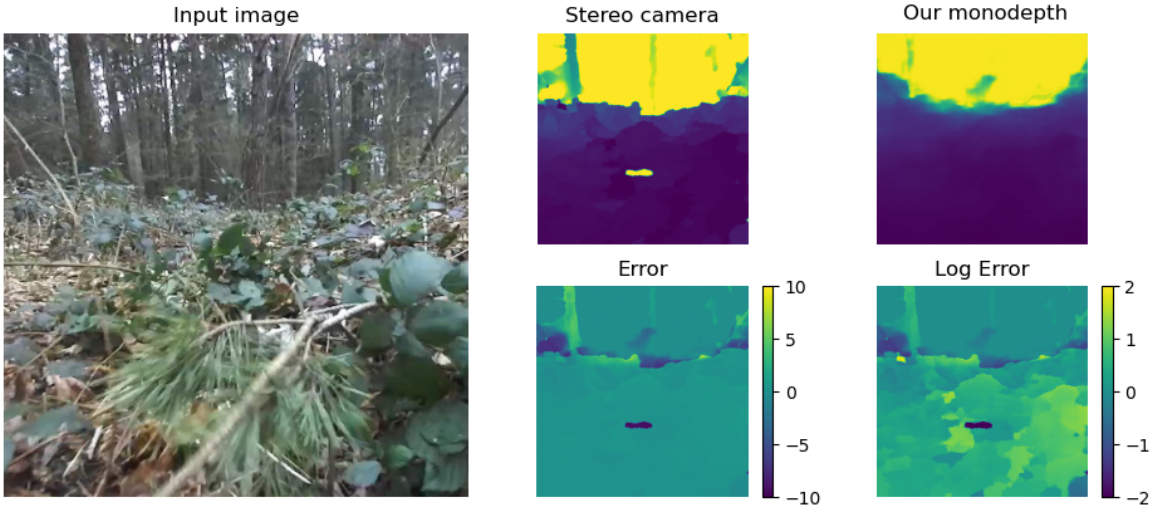



Figure 3.1: Comparison of the signed error and the signed logarithm error on the outputs of the trained CNN.

During training, we use data augmentation techniques such as random horizontal flips and random cropping to increase the size of the training set and improve the model's robustness to variations in the input data. Moreover, we set an upper limit of 10 meters for the distance in the labels, which is adequate for our objectives and removes the erroneous areas of the depth map, like the sky.

We use a U-Net architecture [13] for depth map generation. The U-Net architecture is a type of (CNN) that is commonly used for image segmentation tasks. It consists of an encoder that down-samples the input image to a low-resolution feature map and a decoder that up-samples the feature map to the output resolution. The encoder and decoder are connected by skip connections, enabling the decoder to incorporate fine-grained details

from the input image.

Once the U-Net model is trained, we can estimate depth maps for new forest scenes. The model takes as input a single RGB image and outputs a corresponding depth map.

### Evaluation of the monocular depth CNN

We focus on the signed error between the ground truth and the predicted depth map. The following two figures show the input image, the ground truth, the prediction from the CNN, and the difference between the two depth maps.

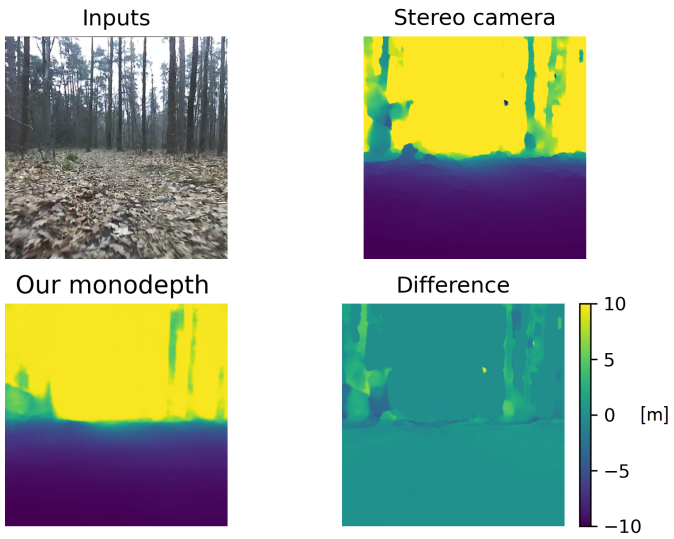



Figure 3.2: Comparison of the ground truth depth map and the depth map predicted with the CNN

Figure 3.2 captures a relatively simple scene with flat ground and distant obstacles. The ground also contains many visual cues, enabling a precise depth estimation by the binocular camera.

The comparison of the results indicates that the deviation is relatively insignificant in the ground region and the distant areas, as the ground truth is capped at 10 meters for training.

However, errors are noticeable in distant obstacles such as trees, which are challenging to estimate as the estimation of their distance is solely based on the width in the monocular image, which could be due to their distance from the camera or actual size.

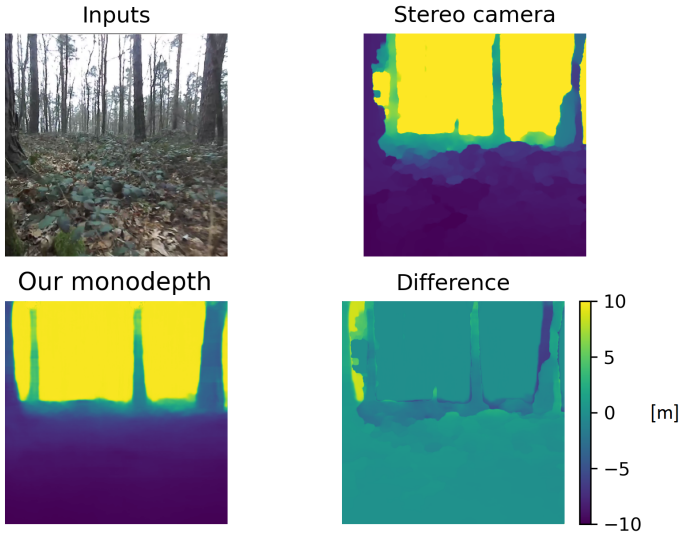
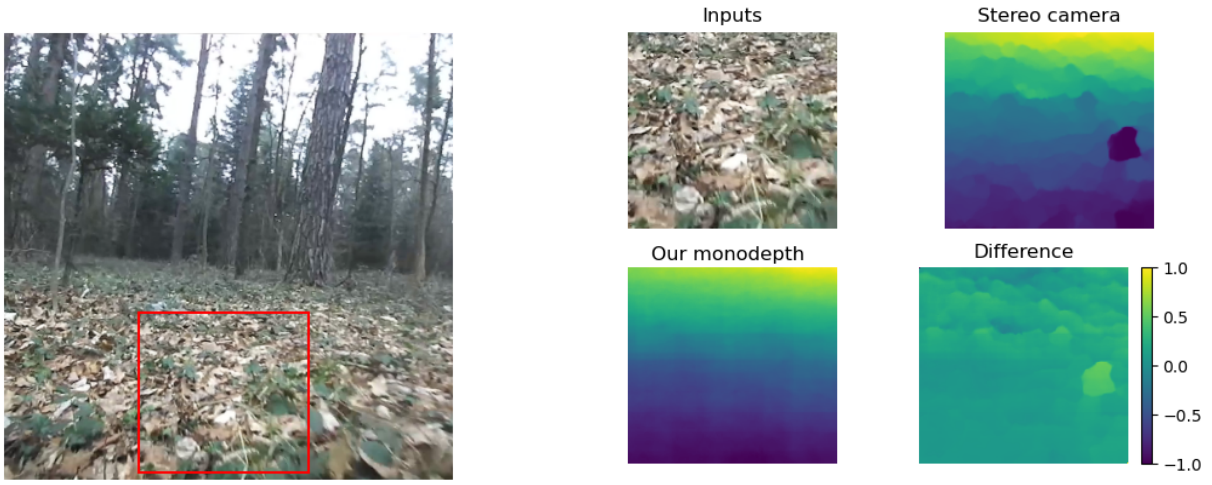



Figure 3.3: Comparison of the ground truth depth map and the depth map predicted with the CNN

Figure 3.3 captures a more complex scene with many occlusions. Notably, the CNN occasionally outperforms the stereo approach, as evident, for example, from the tree on the left. The occlusion at the side of the tree results in an imprecise estimation of the depth. However, this occlusion does not pose a problem for the CNN approach since it relies solely on the monocular image.




(a) The original image (b) The detail

Figure 3.4: Detail of the ground truth depth map and the depth map predicted with the CNN. The ground truth and the predicted depth map are logarithmically scaled for better visibility.

Figure 3.4 captures a detail of the ground. The logarithmic scaling of the two depth maps is applied to make the details more visible. The comparison is presented in meters to show the real difference. The CNN encounters difficulty capturing fine details, although these are usually not crucial for navigation purposes.

The plots in figure 3.5 demonstrate the model's performance quantitatively. Note that the range of the distance is 0.5 - 10 m. Obstacles at closer distances are usually occluded, resulting in inaccurate stereo camera estimations. Additionally, such objects are not generally an issue for navigation purposes.

Also, the last bin on the graph, which corresponds to 10 meters or more, was excluded. This is because the maximum distance is capped at 10 meters, which results in an artificially lower mean absolute error value on this bin. At the same time, the pixel count would be significantly larger than in other bins.

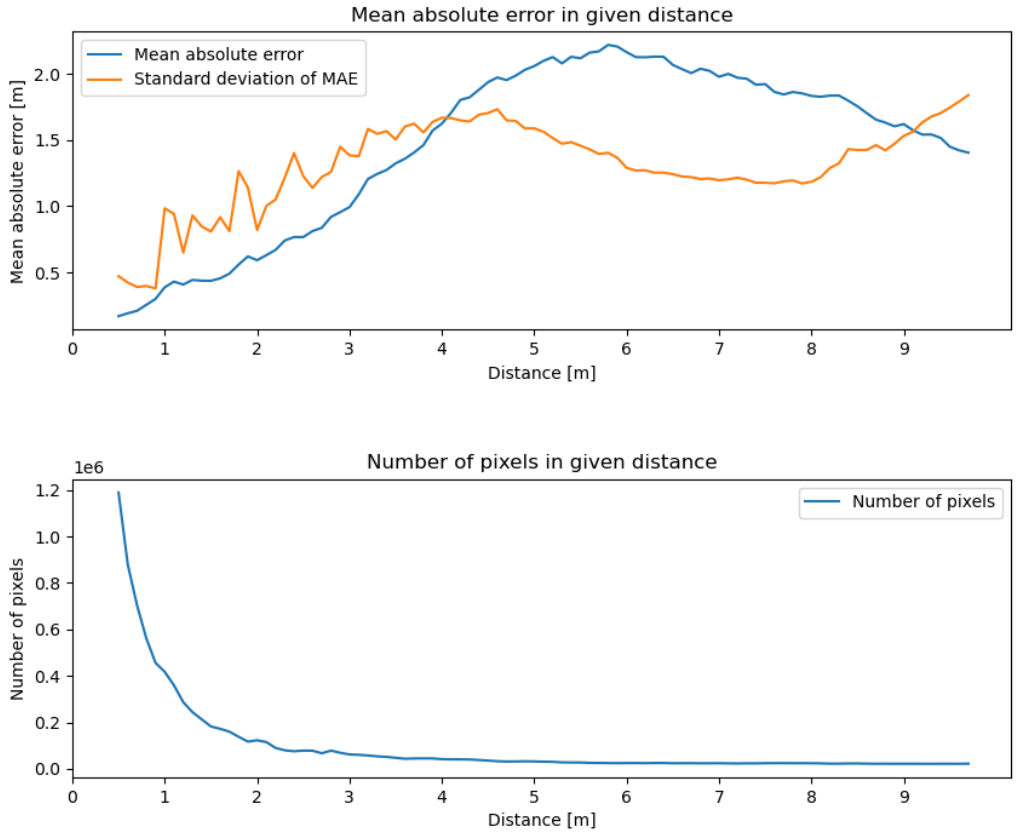



Figure 3.5: The upper plot displays the average deviation from the ground truth relative to distance, while the lower plot shows the average number of pixels at specific distances. The distance between the bins is 10 cm.

## 3.2 Point cloud

A point cloud is a set of $N$ points $(x_i, y_i, z_i)_{i=1}^{N}$ in 3D space that represent the surface of an object or a scene. It is typically used for storing and processing 3D information.

### 3.2.1 Conversion of the depth map to the point cloud

In this work, we generate the point cloud from the depth map obtained using the binocular camera or the CNN-based approach. We use the knowledge of the camera's intrinsic parameters to transform each pixel from the depth map to a 3D point with respect to the camera frame.

The intrinsic parameters are the focal length and the projection center. The focal length represents the distance of the projection (image) plane from the camera center. A projection plane is a hypothetical plane in the camera that captures the 3D scene and projects it onto a 2D image. We denote the focal length as $f$. In our case, it is the same in the x and y directions. The focal length of our camera is 529 pixels or 2.096 mm. The projection center refers to the point where the optical axis intersects the projection plane. We denote it as $c_x$ in the x direction and $c_y$ in the y direction. The values are 635.8 and 365.1 pixels (2.54 and 1.46 mm), respectively.

We will assume a point in the scene $X_s = (x, y, z)$, which is projected to a point $x_i = (u, v)$ in the image. Each point in the image is associated with a depth, or a distance from the camera, denoted with $d_s$. We can use the triangle similarity to compute the coordinates $x, y, z$ from $u, v$ and $d$, which is known.

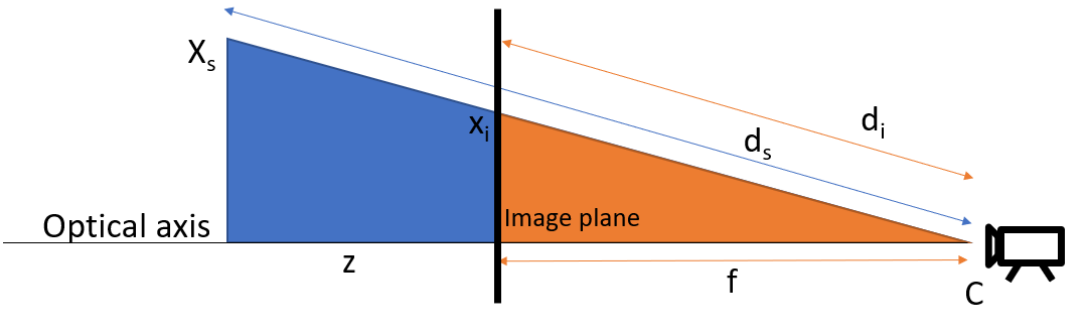



Figure 3.6: This projection displays the triangles, with the distance of the scene point to the optical axis denoted as $o$, and the distance of the image point from the optical axis as $p$.

Figure 3.6 shows these triangles. In our scenario, the camera is located at the origin of the world coordinate system, while the projection center is positioned at the center of the image coordinate system. The two coordinate systems are aligned so that the axes $x$ and $y$ align with $u$ and $v$, while the axis $z$ is parallel to the optical axis and perpendicular to the image plane. In our example, we have only considered the 2D projection of the setting (omitting the $y$ axis, or supposing that $y = 0$, so the point $X_s$ is represented by its $(x, z)$ coordinates), but the same principles hold for the 3D scene.

We can represent the distances of the points $X_s$ and $x_i$ from the optical axis as $|X_s|$ and $|x_i|$, respectively. Since the intrinsic parameters are known, we can compute the distance $|x_i|$. We denote the distance of the projection $X_i$ from the camera center $C$ as $d_i$. It can be computed as $d_i = \sqrt{f^2 + x_i^2}$. It holds that $\frac{z}{f} = \frac{d_s}{d_i}$, so $z$ can be computed as $z = \frac{f d_s}{d_i}$. It holds that $\frac{|X_s|}{z} = \frac{|x_i|}{f}$. From that, we obtain $|X_s| = \frac{z|x_i|}{f}$.

Both coordinates of the 2D point $X_s$ are now known. The procedure for the 3D point would be equivalent. This approach maps each pixel in the depth map to one point in the point cloud.

### 3.2.2 Obstacle detection from the point cloud

To detect the obstacles from the point cloud, we first estimate the ground plane and then measure the distance of the off-the-plane points from this plane.

The point cloud is always computed from a single depth map. Therefore, each three-dimensional point in the point cloud corresponds to one pixel in the depth map. The point cloud is stored in a three-dimensional array of size $W \times H \times 3$, where $W$ is the width and $H$ is the height of the depth map.

#### Ground plane estimation with RANSAC

The algorithm for estimating the ground plane is based on the Random sample consensus (RANSAC) [14]. We only utilize a subset of points from a rectangular region of interest in the lower part of the image. The reason is that the surface in this region tends to be relatively flat, while obstacles are often located further away. Furthermore, we can achieve faster processing times by limiting the processing to a smaller region of interest. In our case, the size of this region is $500 \times 250$ pixels. It is selected to be located horizontally in the middle of the image, with a vertical offset of 20 pixels from the lower border, thus

covering the area in front of the vehicle (see figure 3.7). Note that the rectangular area in the image results in the trapezoidal area in the scene.

Figure 3.7: An input image with the region of interest

The RANSAC runs in 100 iterations maximum. In each iteration, a random triplet of the points is selected. Once the plane is computed from the triplet, the RANSAC algorithm evaluates the fit quality by calculating the sum of the absolute distances of all points from the estimated plane. To account for potential noise or outliers in the point cloud data, we clip the distances to the range of 0 - 10 cm. A lower sum of clipped absolute distances indicates a better fit of the estimated plane to the points, suggesting a more accurate ground plane estimation. We will refer to this sum as the score. The estimation with the lowest score is selected as the candidate ground plane.

We also apply a validation process to avoid erroneous estimates. These errors commonly arise from significant obstacles within the region of interest. The candidate plane is compared with the previous planes for evaluation. Suppose the current score deviates from the mean of the previous scores by more than 1.5 times the standard deviation, and the angle between the candidate plane and the preceding plane exceeds 18°. In that case, the current plane is identified as an outlier, and the preceding plane is used as the current scene's ground plane.

**Elevation Map**

Once we have estimated the ground plane from the subset of points, we return to the whole point cloud. To identify obstacles, we compare the vertical distance of each point in the point cloud to the estimated ground plane. Points located above the ground plane by a significant threshold are classified as obstacles. The threshold is set to 15 cm for the point cloud obtained with the stereo camera and 30 cm for the deep learning approach. These points represent potential objects, obstacles, or structures in the scene that deviate from the expected ground surface.

Our approach does not consider holes below the estimated ground plane. The camera's field of view and the settings of our system do not capture data from obscured or occluded areas, resulting in gaps or missing points in the point cloud data below the ground plane. Furthermore, our dataset doesn't contain any holes that are difficult to traverse with the vehicle.

To facilitate further processing, we convert the point cloud back to a 2D array, which aligns with the resolution of the original depth map. In this array, each element corresponds to a point from the point cloud and contains the distance of the point from the ground plane. Points that lie on the ground plane will have a distance of 0 in the converted array. Points above the ground plane will have positive distances, indicating their height or elevation from the ground plane. Similarly, points that are below the ground plane will have negative distances.

This data structure is used for subsequent obstacle detection algorithms that rely on depth information to identify scene obstacles. We will use the term 'elevation map' as a shorthand reference.

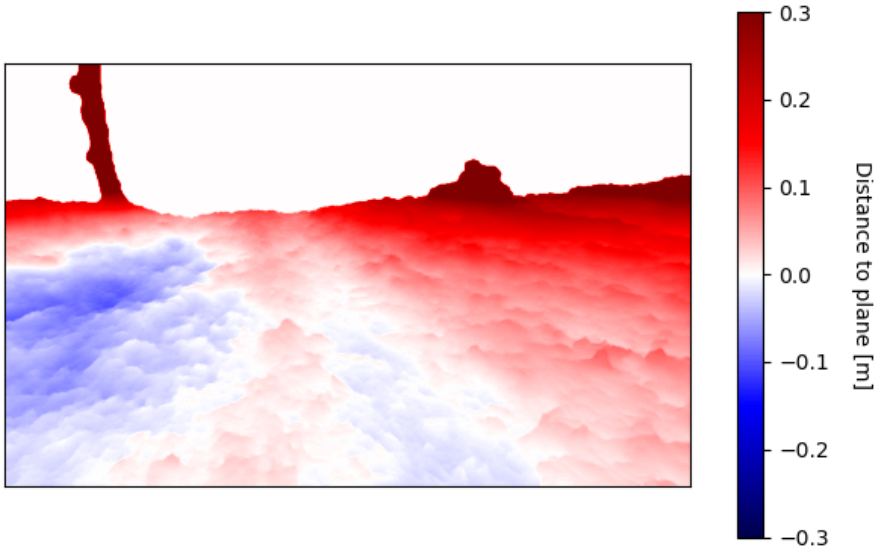



Figure 3.8: The elevation map (points further than 5 m are hidden, distances from the ground plane are clipped at ±0.3 m).

### 3.2.3 Geometric segmentation

The geometric segmentation relies solely on geometric information from the acquired point cloud data without semantic segmentation. This approach contrasts with more advanced methods that utilize semantic segmentation techniques for identifying obstacles.

We use the depth map and elevation map to construct the geometric segmentation. The depth map provides information about the distances of points in the point cloud from the camera, while the elevation map encodes the distances of points from the ground plane. We set a threshold on both maps.

In our settings, we set the threshold for the depth map to 5 meters since more distant points are irrelevant. Additionally, the accuracy of depth estimation decreases with increasing distance. Consequently, we label the points beyond this threshold as 'too distant.'

The threshold for the elevation map is set to 15 cm. In our specific settings, which involve a forest environment, the fluctuations in the elevation map are often caused by the wavy and uneven nature of the terrain rather than the presence of obstacles. This can cause fluctuations in the elevation map, even in the absence of actual obstacles. By setting a threshold of 15 centimeters for the elevation map, we account for this inherent

terrain variability and minimize false positives in obstacle detection. Points above this threshold are labeled as 'too high.'

The geometric segmentation can be constructed directly from the depth map. The depth map is estimated from the binocular camera or the CNN, which estimates the depth from a single image. The segmentation is used to simplify the manual annotation, as an input for more sophisticated semantic segmentation methods, and as the baseline method of the segmentation.

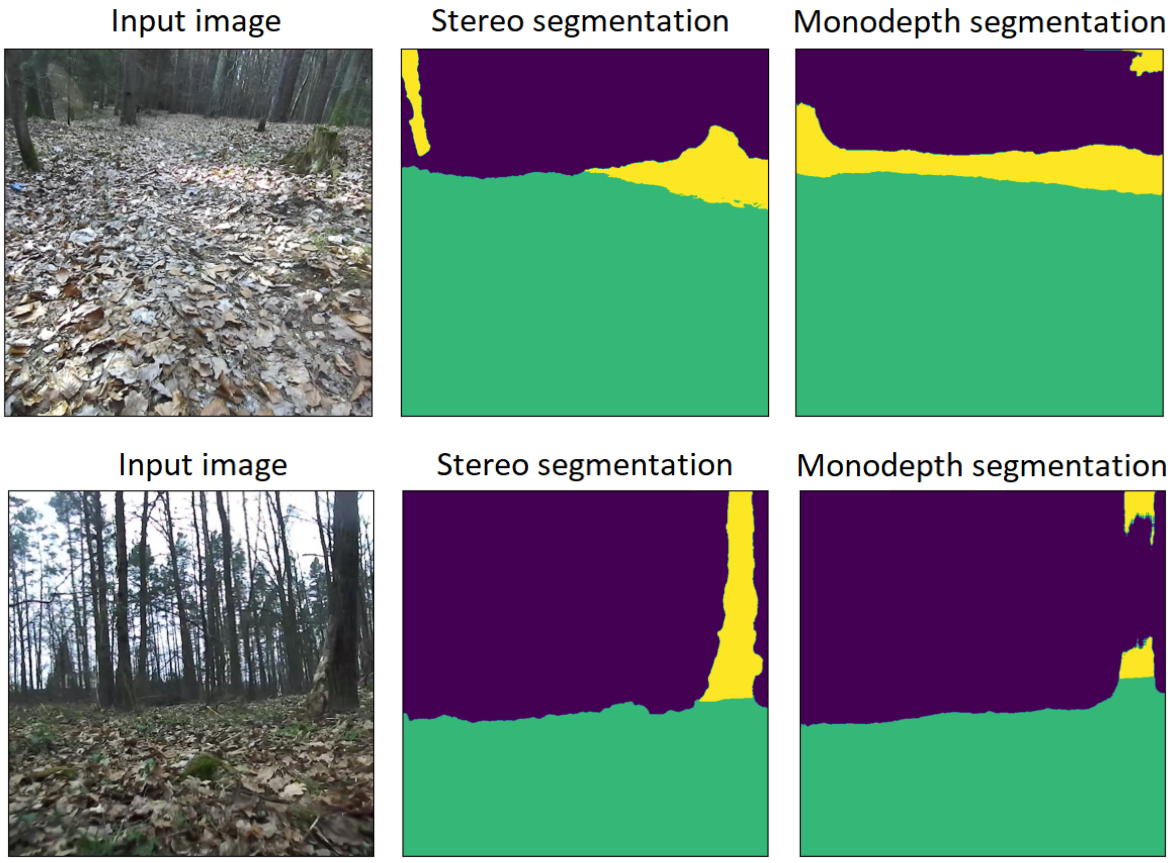



Figure 3.9: Comparison of geometric segmentation based on the stereo camera and the deep learning approach. Note that even though the upper right corner in the first image and the tree in the second image are classified incorrectly with the deep learning approach, both of these misclassified regions eventually fall into the 'dangerous' class

.

## 3.3 Semantic segmentation

In this section, we introduce several methods for the semantic segmentation. The first is based on geometric segmentation, and the rest is based on deep learning. The output is always a two-class segmentation, where points are categorized into either 'safe' (non-

obstacle) or 'dangerous' (obstacle) classes.

A dataset of 2000 images was used for training and validation purposes. The dataset was split in a 4-to-1 ratio, with 80% of the images used for training and 20% for validation. The images in the dataset were of size 448 pixels, and a learning rate of 0.0001 was used during training. The training process was carried out for 25 epochs. We used the Adam optimizer [15] to update the model's weights. We also implemented the random horizontal flip and random offset of the square cut-out from the side of the original rectangular image.

The methods differ in the way the ground truth annotation is obtained and the type of information provided as input to the segmentation task.

### 3.3.1 Baseline

The geometric segmentation described in section 3.2.3 is used as the baseline method. It is the only method that is not based on deep learning. The geometric segmentation produces three classes: 'safe' for points that satisfy the height and distance thresholds, 'too high' for points that exceed the height threshold, and 'too distant' for points beyond the distance threshold. In the baseline method, the 'safe' class from the geometric segmentation corresponds to the 'safe' class, while combining the 'too high' and 'too distant' classes forms the 'dangerous' class.

We will refer to this segmentation as 'The Baseline method.'

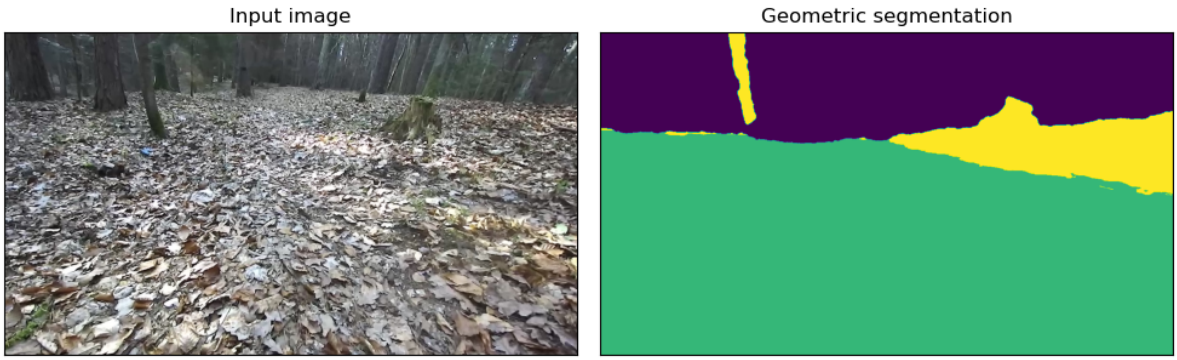



Figure 3.10: The geometric segmentation. Green: safe; yellow: too high; purple: too distant.

### 3.3.2 Supervised deep learning

In our experiments, we found that our cart can successfully overcome any obstacle below 15 cm in height, which means that we can assume that there are no false negatives in our geometric segmentation results. However, due to the wavy terrain common in our forest environment and some soft obstacles, such as high grass or a pile of fallen leaves, there may be false positives in the 'too high' class. We can omit the potential false positives in the 'too distant' class because the distant points do not pose an issue to the cart's navigation and may contain inaccuracies.

We aim to train the CNN to classify the points into two classes: 'safe' and 'dangerous.' The annotation is based on the geometric segmentation from the baseline method. In our geometric segmentation, we will assign all points classified as 'too distant' to the 'dangerous' class, as these points are irrelevant to the cart's navigation. Similarly, all points classified as 'safe' in our geometric segmentation will be assigned to the 'safe' class.

The high points in the elevation map can indicate different scenarios, ranging from real obstacles, such as trees, to variations in the terrain or soft obstacles, like grass. As a result, manual annotation is necessary to identify and label these points in the training dataset accurately.

The classes of 'safe' and 'too distant' are already distinguished by geometric segmentation. To simplify the manual annotation process, we apply a mask to the image to conceal these two classes, leaving only the 'too high' class for manual annotation.

This approach has the downside of ignoring the potential obstacles below the 15 cm threshold, for example, the holes or the puddles. However, such obstacles are not present in our dataset.

The supervised deep learning approach can be further categorized into two scenarios.

#### Supervised learning with geometric segmentation

In the first scenario, the geometric segmentation, which classifies points as 'safe,' 'too high,' or 'too distant,' is attached to the image as an additional input. This allows the model to learn from the raw image data and incorporate the geometric information obtained from the elevation map. It could improve the accuracy and robustness of the obstacle detection system. The geometric segmentation is attached to the image as a fourth channel, which results in the input size $448 \times 448 \times 4$.

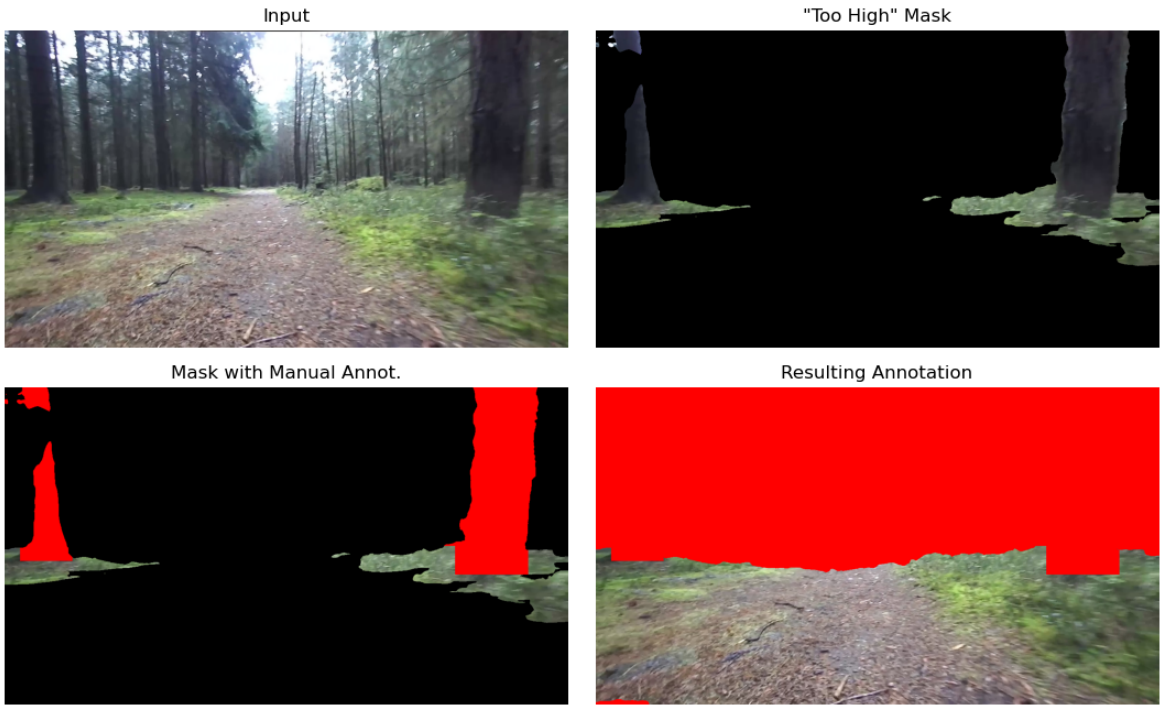



Figure 3.11: The process of manual image annotation. Note that only the masked image is being annotated, which simplifies the process.

The downside of this approach is the need for a calibrated binocular camera on the vehicle. Furthermore, the training data needs to be annotated manually.

We will refer to this scenario as the 'Geometric Segmentation Supervised' approach.

### Image-Only Supervised approach

In the second scenario, only the image is used as the input for the CNN. This approach requires geometric segmentation and manual annotation for the training, but the trained CNN works with monocular data only. Therefore, there is no need for a calibrated binocular camera on the vehicle.

The downsides of this method are the need for manual annotation and the lack of information about the scene's geometry.

We will refer to this scenario as the 'Image-Only Supervised' approach.

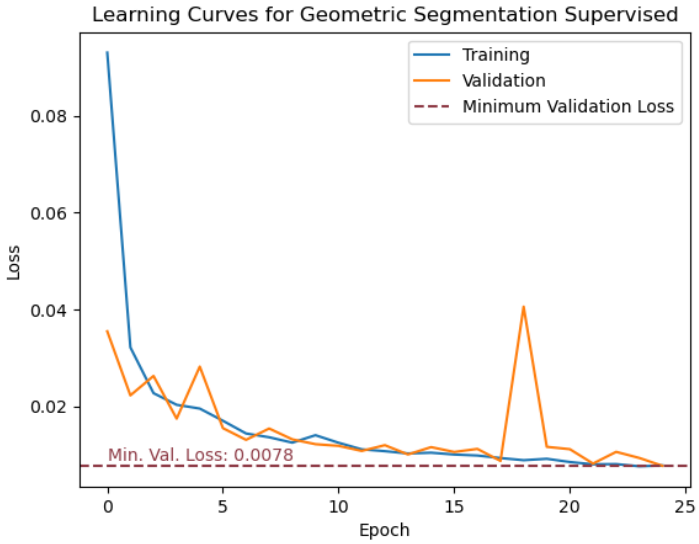



Figure 3.12: The learning curves of the model. There is an average loss on one image on the y-axis.

### 3.3.3 Self-supervised deep learning

We manually annotated ground truth labels in the previous chapter, resulting in a more precise but time-consuming process. In contrast, in this chapter, we utilize automatic annotation methods to generate labels for training data, which is more manageable.

Again, we use geometric segmentation as the ground truth label. In this experiment, the 'too high' and 'too distant' classes are assigned to the 'dangerous' class without further manual intervention.

The assumption is that the scene elements, such as trees and the ground, will likely be accurately classified into their respective categories in the geometric segmentation. While occasional misclassifications, such as parts of the ground being labeled as 'too high,' may occur, most of the ground is expected to be classified as 'safe' in the geometric segmentation, which enables the CNN to learn to classify it correctly.

This approach can be categorized into two scenarios, each with a distinct method of obtaining the depth map.

#### Self-supervision with the stereo camera

The geometric segmentation is derived from the depth map, typically acquired using a binocular camera. The depth map obtained through this method is known for its high

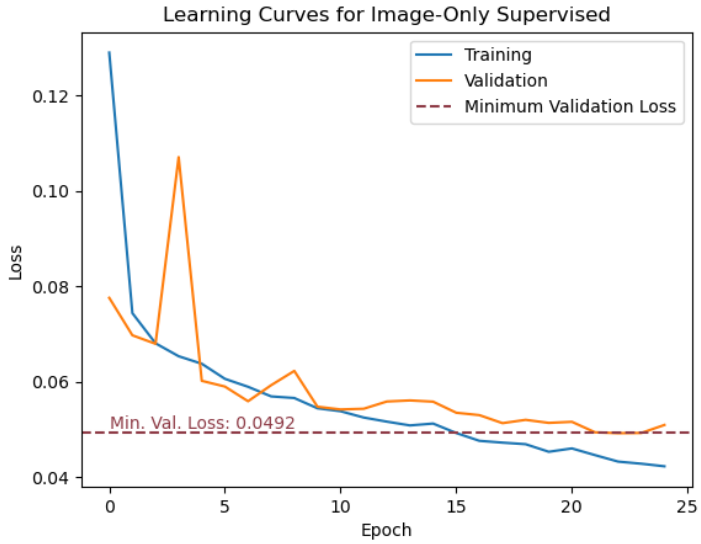



Figure 3.13: The learning curves of the model. There is an average loss on one image on the y-axis.

accuracy and can be generated in any scene, as it relies on a geometric approach. However, a drawback of this approach is the requirement for a calibrated binocular camera.

We will refer to this approach as the 'Self-Supervised Binocular Depth' method.

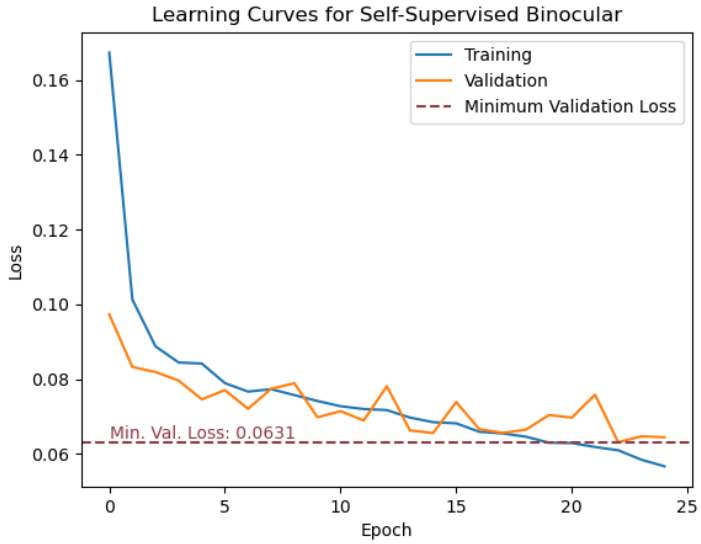



Figure 3.14: The learning curves of the model. There is an average loss on one image on the y-axis.

### Self-supervision with monocular camera

The alternative approach for obtaining the depth map is through deep learning techniques. We use our model, which predicts the depth from a single image. The main advantage of the deep learning-based approach is that it eliminates the need for a stereo camera setup. On the other hand, this approach is environment-specific, meaning the model is trained on a specific data set and may not generalize well to different environments.

We will refer to this approach as the 'Self-Supervised Monocular Depth' method.

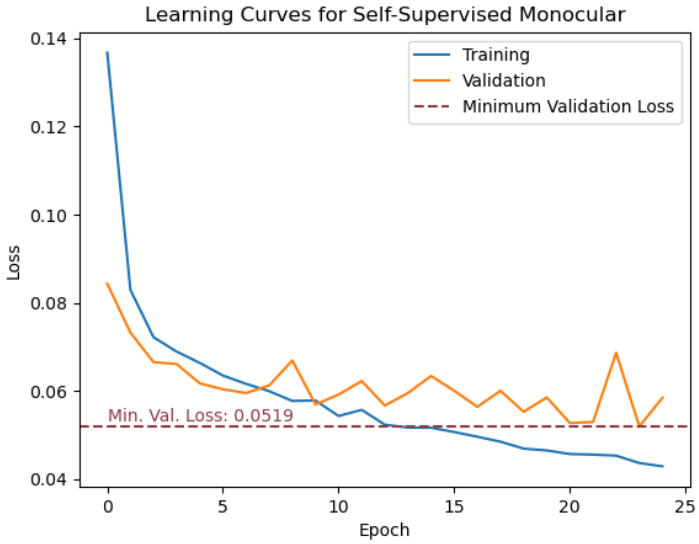



Figure 3.15: The learning curves of the model. There is an average loss on one image on the y-axis.

# Chapter 4

# Experiments and Results

This chapter presents the experimental setup and the results of the experiments of methods described in Chapter 3. The first section provides an overview of the hardware components employed. The second section focuses on the evaluation of the experiments, both qualitatively and quantitatively.

## 4.1 Hardware

This section describes the Tomi2 platform, the substitute manually controlled cart, and the stereo camera ZED 2i.

### 4.1.1 Tomi2

This thesis continues to work on research started by the Tomi2 project. The implemented methods are meant to be subsequently deployed on this platform.

The Tomi2 platform is based on the commercial Losi1:5 DBXLE platform. Its current weight is around 20 kg, and the dimensions are $844 \times 501 \times 308$ mm. It is propelled with an electric motor and modified so that each wheel has its own servo motor, enabling independent 4-wheel steering.

Multiple computing units have been added to the platform. The neural network runs on an NVIDIA Jetson Xavier computer, which contains a GPU compatible with the NVIDIA CUDA system, which is necessary for running neural networks implemented in

Pytorch.

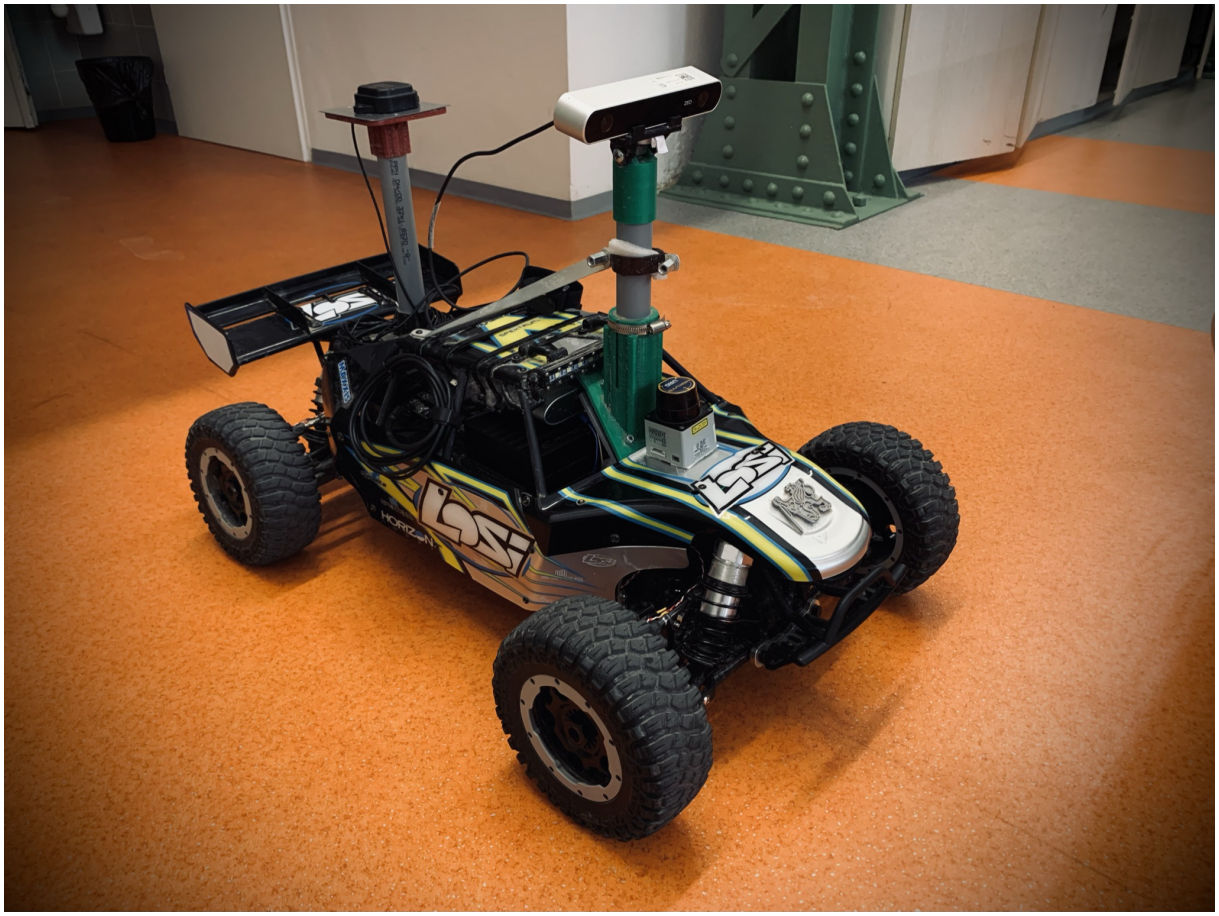


Figure 4.1: The Tomi2 platform

Apart from the computing units, the platform also contains multiple sensors, like GPS or accelerometers, and a camera. Only the data from the camera was used for this project.

### Manually controlled cart

The Tomi2 is a complex platform that is difficult to operate. For example, operating the platform in one person is impossible due to safety risks. Transporting it to the terrain suitable for our experiments is also problematic.

Therefore, an alternative, manually controlled cart was used for collecting the data. It was designed to resemble the Tomi2 platform. It weighs 12 kg, the wheels' circumference is 25 cm, and the camera is 35 cm above the ground. The camera is positioned to look straight ahead. It is connected to a laptop located off the cart.

The cart is being pulled rather than pushed for better maneuverability and to overcome

(a) The cart (b) The detail of the ZED 2i stereo camera

Figure 4.2: The Tomi 0.5 platform

larger obstacles. This results in the reversion of the camera recording. The reversion is not a significant issue because it does not affect the analysis or interpretation of the individual frames of the recording, which are the focus of interest.

### 4.1.2 ZED 2i

ZED 2i is a compact binocular camera produced by Stereolabs. It contains two separate lenses so that it can capture stereo recordings. Each of the lenses has a ratio of 16:9 and is capable of capturing video in the resolution up to $2208 \times 1242$ for each camera. The pixels are square, and their size is 0.004 mm. The baseline (the distance between the two cameras) is 12 cm.

Stereolabs provide a software development kit (SDK), which contains multiple tools for the camera. For this project, we used the depth sensing tool, which implements the algorithm for obtaining the depth map based on the disparity of correspondences.


Figure 4.3: The binocular camera ZED 2i

The recording is carried out using a Python script. Several parameters can be specified for the camera. The resolution is set to $1280 \times 720$, and the video is recorded in 60 frames per second.

The depth computation mode is set to 'ultra,' which is supposed to provide a more accurate depth estimation. On the other hand, it is computationally more expensive, which is not an issue in our offline settings.

Additionally, we are using the 'fill mode' to address blank spaces in the depth map. The blank spaces are usually caused by occlusions or a lack of texture in the image. The exact algorithm is not disclosed, but it presumably infers the distance from the neighborhood where it is known. This enables more convenient work with the depth map since we do not have to consider the empty depth map regions. On the other hand, we can not rely on the values because they might be incorrect.

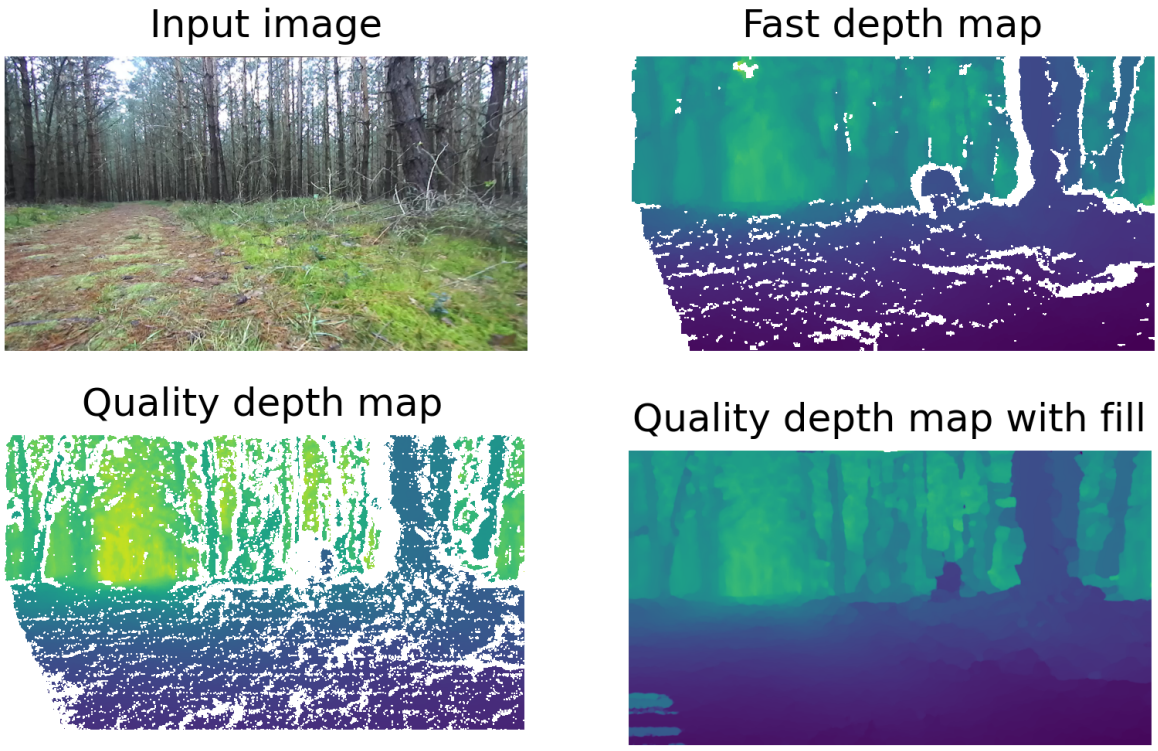



Figure 4.4: Comparison of the depth sensing modes. The depth maps are visualized with a logarithmic color scale.

## 4.2 Evaluation

Our test set consists of 150 hand-picked images. These images were mostly captured in different seasons than the training dataset, resulting in potential variations in the appearance of vegetation. We applied a threshold to the output probabilities of the neural network and compared them with the ground truth.

Figure 4.5: Samples from our test set.

Unlike those in the training data set, the images in the test data set were annotated fully manually. For instance, the entire tree trunks were labeled as 'dangerous.'

There is an ambiguity in classifying the distant but safe parts of the scene, as they are often labeled as 'dangerous' in the training data set due to being classified as 'too far' by the geometric segmentation. However, some distant yet clearly visible parts are labeled 'safe' in the fully manually annotated test data set. This can create a discrepancy where the neural networks learn to classify these parts as 'dangerous' even though they are safe.

While misclassifying these parts as 'dangerous' does not impact the navigation since they are located far from the vehicle, it can lead to incorrect quantitative evaluation of

the results. Therefore, we introduce a new class called 'ambiguous' to address this issue. This class is designed for distant yet safe parts and is only present in the test data set. By doing this, the ambiguous parts of the scene are excluded from the evaluation, thus not affecting the accuracy assessment.

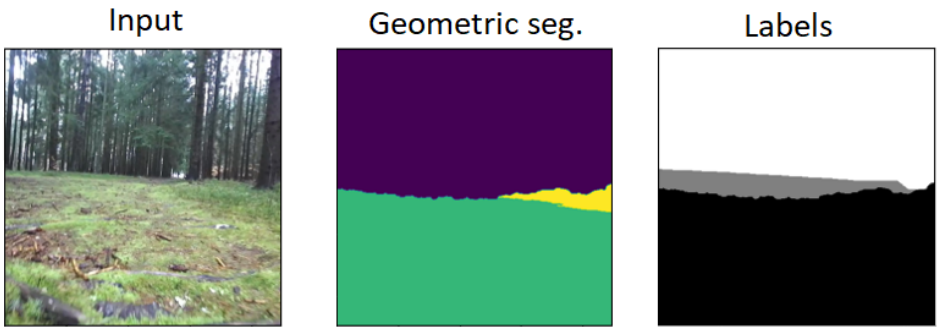



Figure 4.6: Example of the label with the 'ambiguous' class. Black is 'safe,' white is 'dangerous,' and gray is 'ambiguous.'

In the upcoming sections, we will showcase hand-picked examples from the test dataset to illustrate our methods' typical advantages and drawbacks. Following that, we will present a quantitative evaluation of these methods.

| **Method** | **Description** |
|---|---|
| Geometric Segmentation Supervised | Training data annotated manually; geometric segmentation included in the input |
| Image-Only Supervised | Training data annotated manually; only one image in the input |
| Self-Supervised Binocular | Self-supervised using the geometric segmentation from the stereo camera |
| Self-Supervised Monocular | Self-supervised using the geometric segmentation from monodepth |
| Baseline | Only the geometric segmentation; 'too high' and 'too distant' class form the 'dangerous' class |

Table 4.1: Summarization of the methods

### 4.2.1 Qualitative evaluation

We have selected some examples from the test dataset to demonstrate the strengths and weaknesses of our methods. For each example, we present the input image, the geometric segmentation, the ground truth label, and the four deep learning-based semantic segmentation outputs. Note that the baseline method is derived from geometric segmentation, so its outputs are not included among the four deep learning methods. Additionally, note that the segmentations presented in this section are displayed without applying any threshold on the softmax of the output, which allows us to demonstrate the capabilities of the CNNs better.

All the models learned to correctly classify the vegetation as 'safe,' even though it often falls into the 'too high' class in the geometric segmentation. Figure 4.7 shows a typical example of such a situation. This situation is relatively simple because the vegetation is well visible, and there is also enough flat ground to estimate the ground plane for the geometric segmentation.

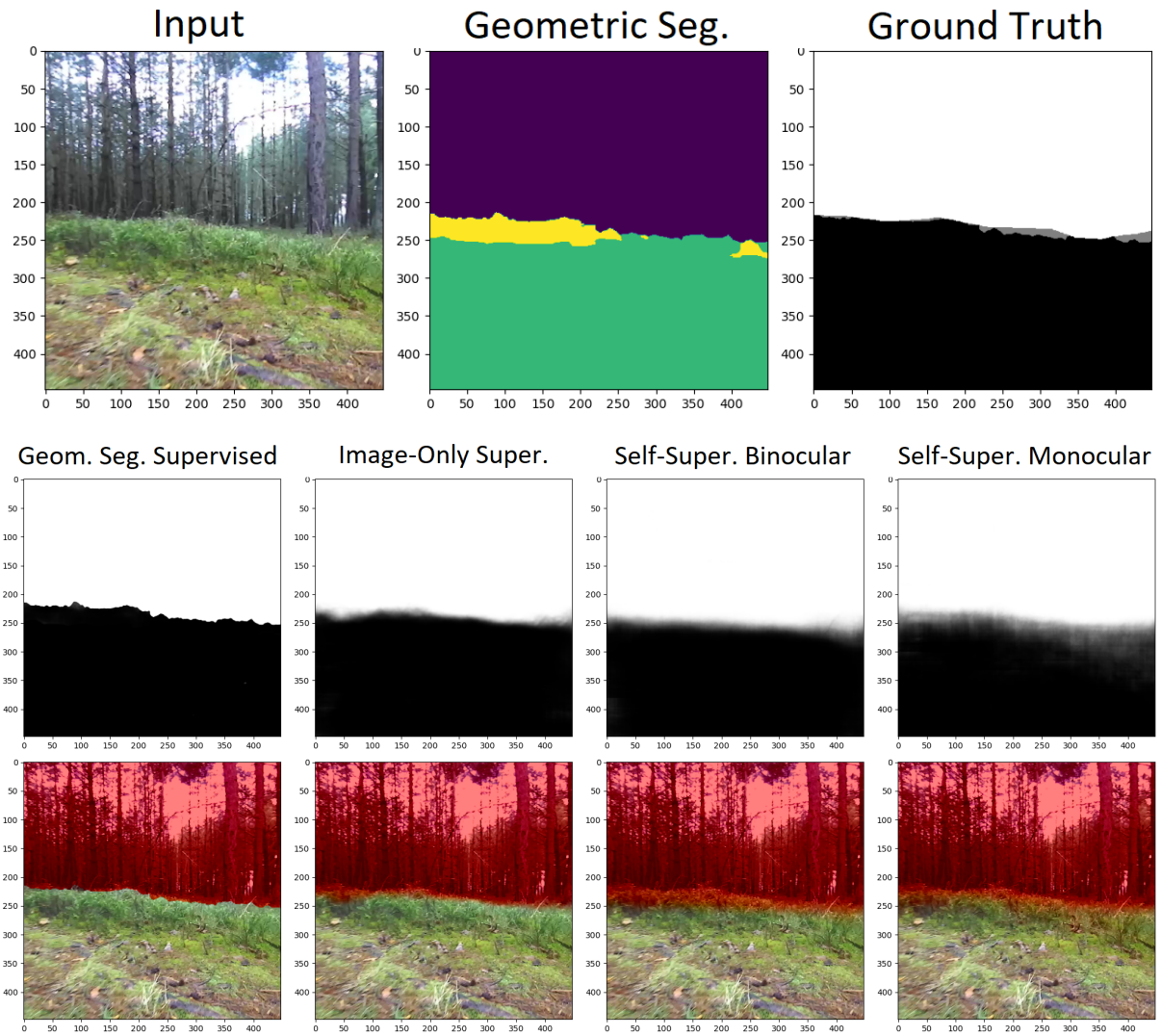



Figure 4.7: Vegetation in the 'too high' class

The situation gets more complicated when vegetation is in close proximity to the vehicle. Estimating the depth becomes more complex due to occlusions, and determining the ground plane with RANSAC is also more difficult as the ground is less visible. We implemented a validation process described in the previous chapter (3.2.2), which replaces the outlying ground plane estimates with the earlier estimations. Furthermore, CNNs can typically recognize vegetation with ease in such scenarios. Figure 4.8 show an example of such a situation.

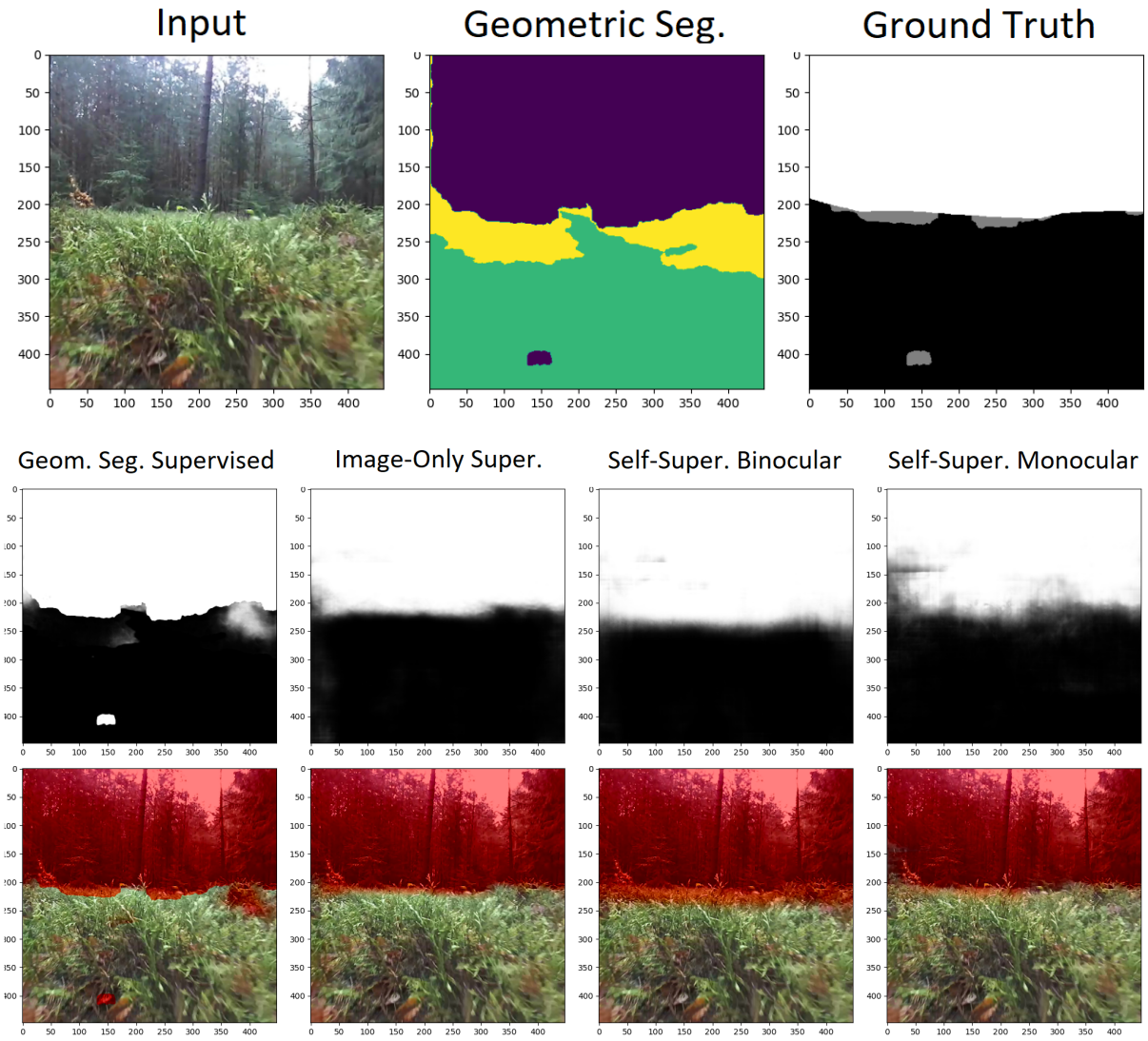



Figure 4.8: Vegetation in close proximity

Note that the region in the lower left part of the geometric segmentation is incorrectly classified as 'too far' due to an occlusion. This causes an error in the 'geometric segmentation supervised' method because it relies on geometric segmentation. In these situations, it can be outperformed by the other methods, which are usually weaker.

The trees are the most common obstacles in our dataset. All the CNNs have learned to classify the trees as 'dangerous' with no difficulties. The only difference between the methods is in the precision of the segmentation. In figure 4.9, the most precise segmentation comes from the method 'image-only supervised.'

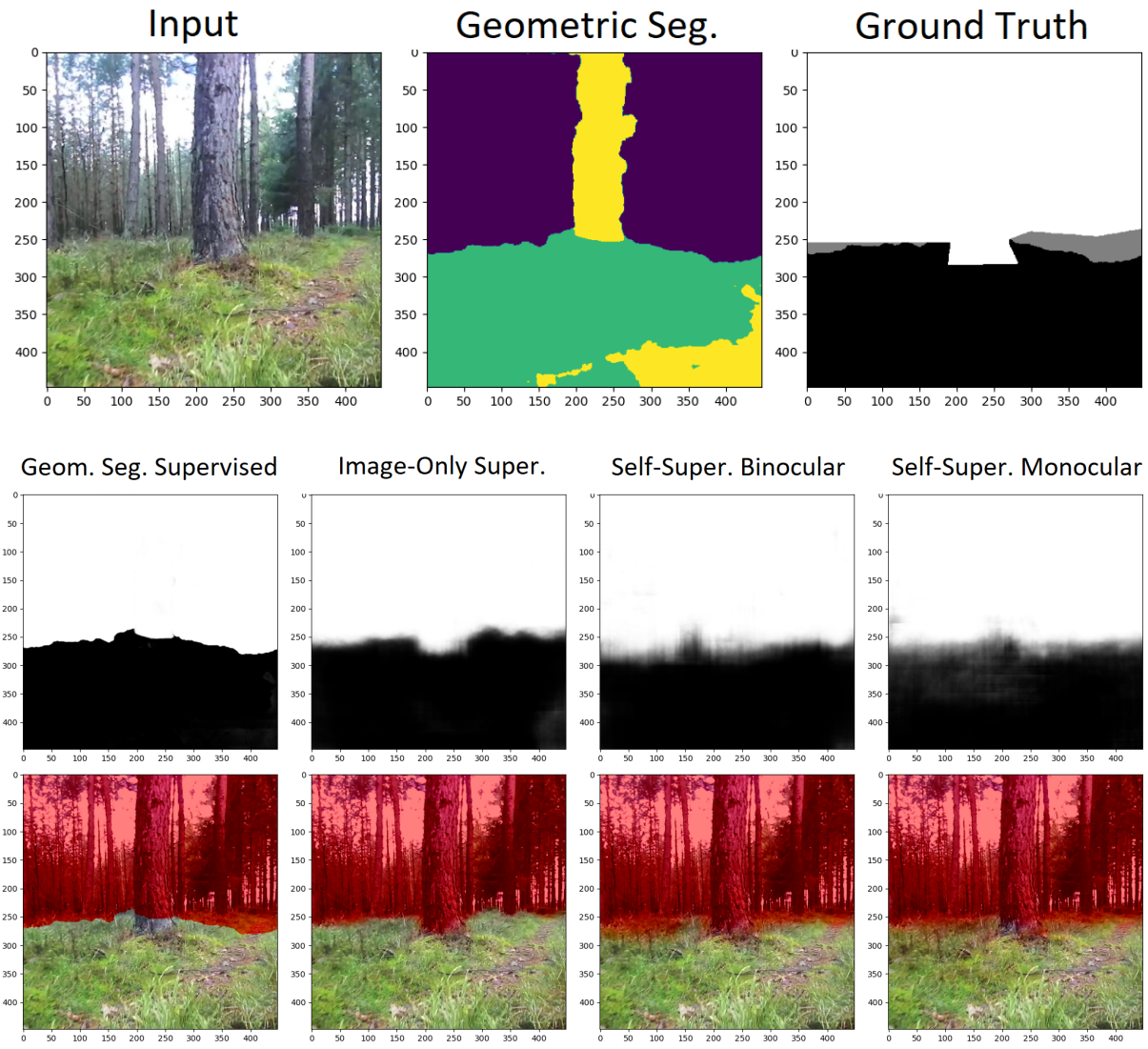



Figure 4.9: A correctly classified obstacle

The wooden block depicted in Figure 4.10 is a more challenging obstacle because it is situated within the grass, which has a comparable height. Due to its relatively low height, only the top part of the block is classified as 'too high' using geometric segmentation. This inaccuracy also applies to similar obstacles present in the training set. It is even possible that obstacles of a similar nature in the training set, which fall below the 15 cm threshold, might be classified as 'safe' by the geometric segmentation. This discrepancy leads to an imperfect annotation in the training set, which makes such obstacles more difficult to classify correctly.

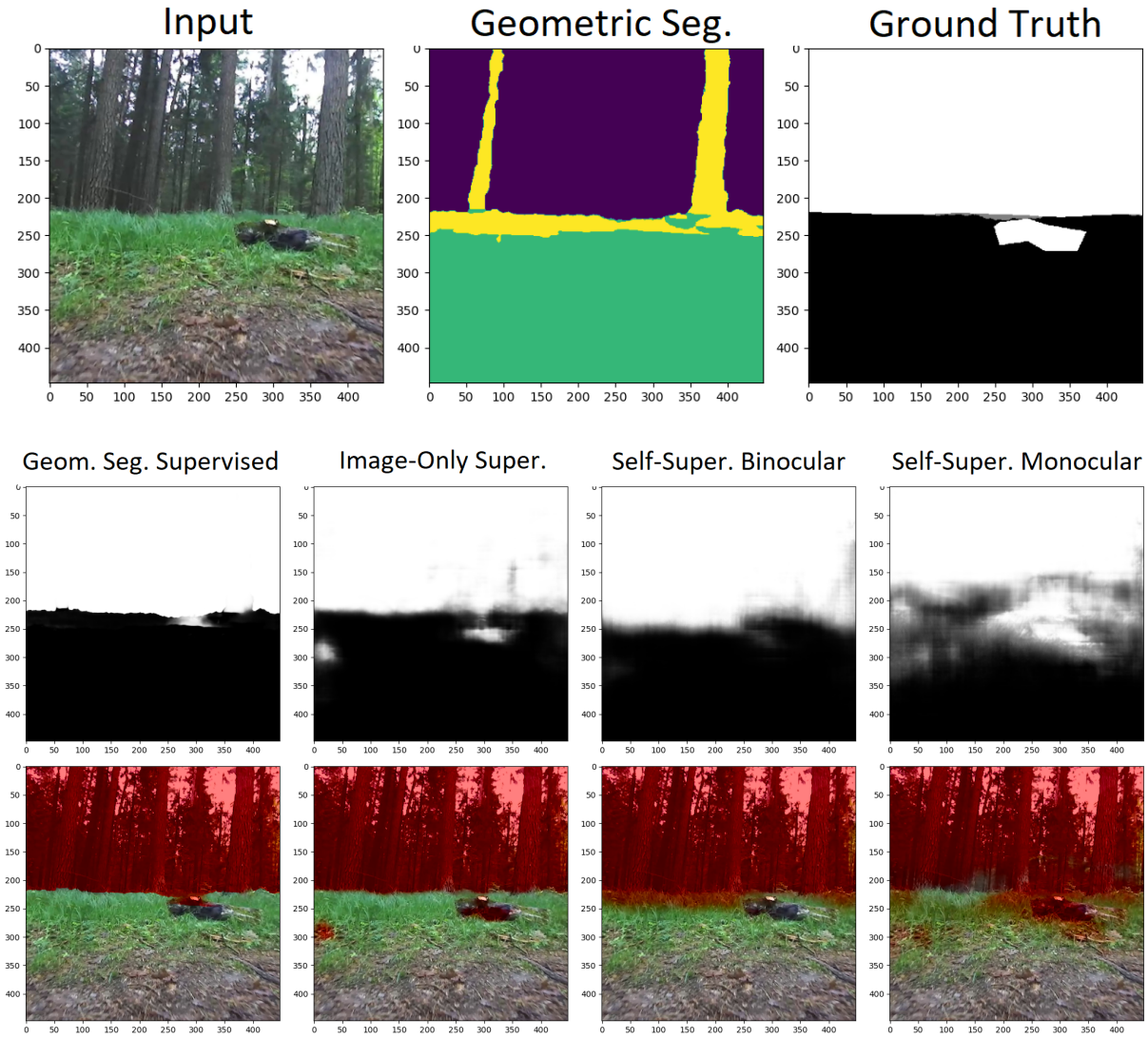



Figure 4.10: A low obstacle surrounded with grass

The 'geometric segmentation supervised' method accurately classifies the top of the obstacle but is misled by the geometric segmentation when it comes to the bottom part. On the other hand, the 'image-only supervised' method demonstrates superior performance in this scenario. However, the self-supervised methods encounter difficulties accurately classifying such obstacles, potentially due to imperfect annotation in the training set.

The fallen tree in figure 4.11 is another challenging obstacle. Due to the absence of such obstacles in the training set, it is more difficult to classify the fallen tree as 'dangerous' while identifying the flat ground beneath and beyond it as safe.

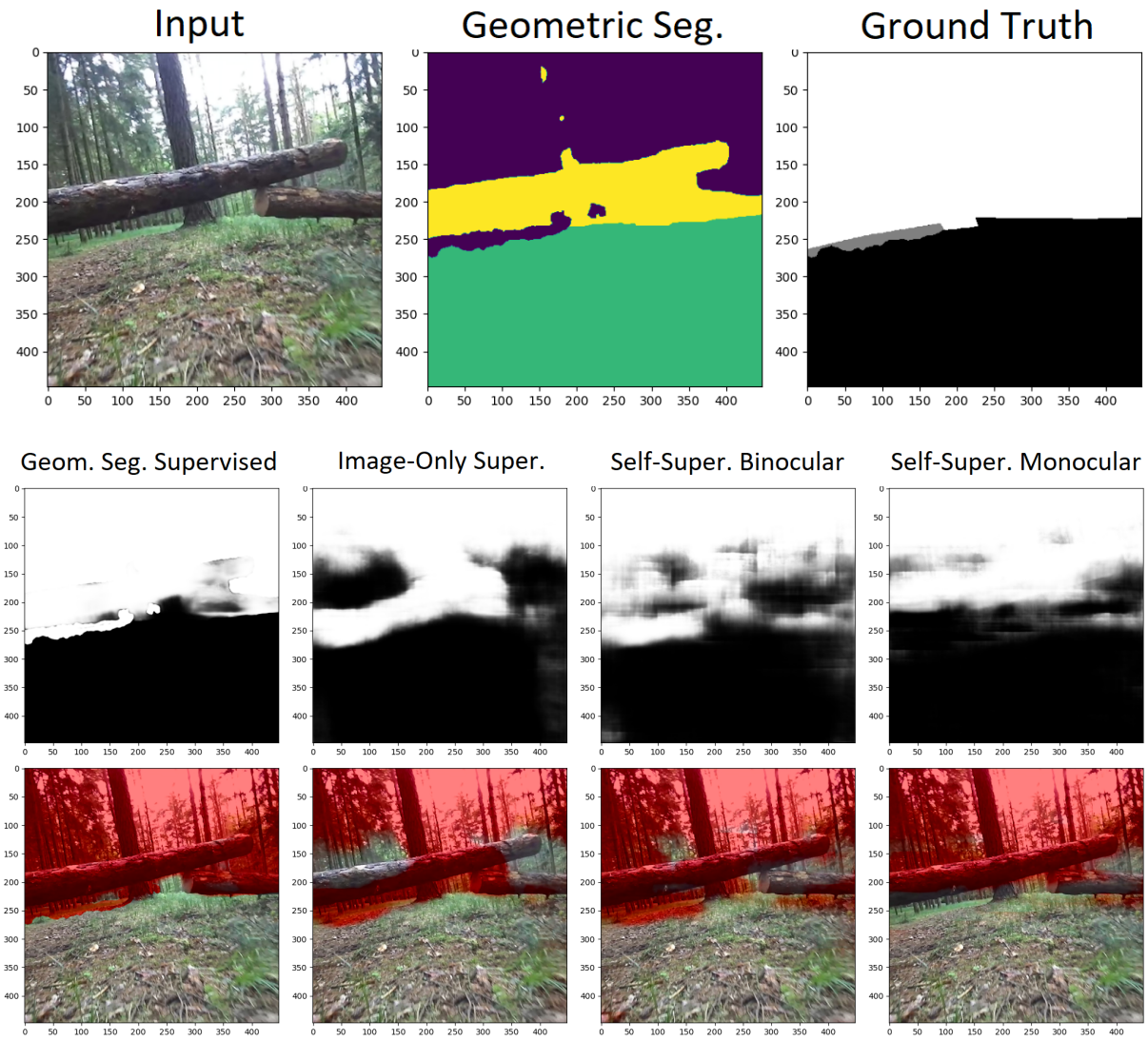



Figure 4.11: An uncommon obstacle

The availability of geometric segmentation makes the 'geometric segmentation supervised' method more robust for uncommon scenes. Also, the other methods mostly classify the trunk correctly as 'dangerous' and the area beneath and beyond as 'safe.' Most errors come from the false negative areas above the trunk.

### 4.2.2 Quantitative evaluation

To evaluate the performance of our methods, we conducted an accuracy assessment. It involved comparing the outputs generated by the neural network (or solely by the geometric segmentation in the case of the baseline method) with the manually annotated ground truth labels. We counted the number of true positive (TP), false positive (FP),

true negative (TN), and false negative (FN) pixels to calculate the performance metrics, such as precision, recall, and F1 score. A true positive was defined as a pixel correctly classified as belonging to the 'dangerous' class. In contrast, a false positive was a pixel classified as belonging to this class but was, in fact, part of the 'safe' class. A true negative was a pixel correctly classified as belonging to the 'safe' class, and a false negative was a pixel classified as belonging to the 'safe' class but was actually part of the 'dangerous' class. By comparing the number of true and false positives and negatives, we were able to determine the accuracy of the model in identifying dangerous and too high areas in the scene.

| | | **Actual** | |
|---|---|---|---|
| | | **Danger.** | **Safe** |
| **Pred.** | **Danger.** | **TP** | **FP** |
| | **Safe** | **FN** | **TN** |

Precision is defined as the ratio of true positives to the total number of predicted positives. The recall is defined as the ratio of true positives to the total number of actual positives in the ground truth. The F1-score is the harmonic mean of precision and recall and balances the two measures.

$$Precision = \frac{TP}{TP + FP} \tag{4.1}$$

$$Recall = \frac{TP}{TP + FN} \tag{4.2}$$

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{4.3}$$

Precision measures the proportion of the predicted positives that are actually positive, while recall measures the proportion of actual positives that the model correctly identifies. The F1 score provides a way to evaluate the system's overall performance by balancing precision and recall. A high F1 score indicates that the system has both high precision and recall and performs well overall.

| Experiment | TP rate | TN rate | FP rate | FN rate | Prec. | Rec. | F1 |
|---|---|---|---|---|---|---|---|
| Geometric Segmentation Supervised | 51.9 | 46.2 | 0.2 | 1.7 | 99.6 | 97.0 | 98.2 |
| Image-Only Supervised | 48.0 | 45.8 | 0.6 | 5.6 | 98.7 | 90.3 | 93.4 |
| Self-Supervised Binocular | 49.2 | 44.6 | 1.8 | 4.3 | 96.5 | 92.3 | 93.8 |
| Self-Supervised Monocular | 49.5 | 44.6 | 1.8 | 4.0 | 96.6 | 92.7 | 94.0 |
| Baseline | 52.7 | 43.8 | 2.6 | 0.9 | 95.3 | 98.4 | 96.7 |

Figure 4.12: The table with quantitative results. The values are in percentages.

Figure 4.12 shows the quantitative evaluation. The 'Geometric Segmentation Supervised' method has the highest F1 score, which balances the precision and recall scores. The 'Baseline' method has the best precision but, at the same time, the worst recall. It means that it relatively rarely overlooks an obstacle, but it also often misclassifies 'safe' obstacles, such as the high grass, as 'dangerous.' This is in accordance with the expectation. The self-supervised methods perform almost equally, which signifies that the stereo camera is not beneficial for self-supervision once we have trained the mono-depth CNN.

### 4.2.3 Limitations

The main limitation of the presented methods is in the annotation of the training data. The self-supervised approach is effortless, but the resulting annotation contains inaccuracies, as discussed in Section 3.3.3.

We also presented a method for manual annotation, described in Section 3.3.2. This method represents a compromise between effortless self-supervision and slow, fully manual annotation. It is based on geometric segmentation so that only the 'too high' class is annotated manually, while the whole 'safe' class remains unchanged.

We assume that obstacles below 15 cm threshold are traversable. However, some

obstacles, such as tree stumps, are slightly higher than the threshold. Only the topmost parts of these obstacles are classified as 'too high' in the geometric segmentation and can be subsequently manually annotated as 'dangerous.' The CNN is then trained to also classify only the up-most part as 'dangerous.' Figure 4.13 shows an example of this situation.

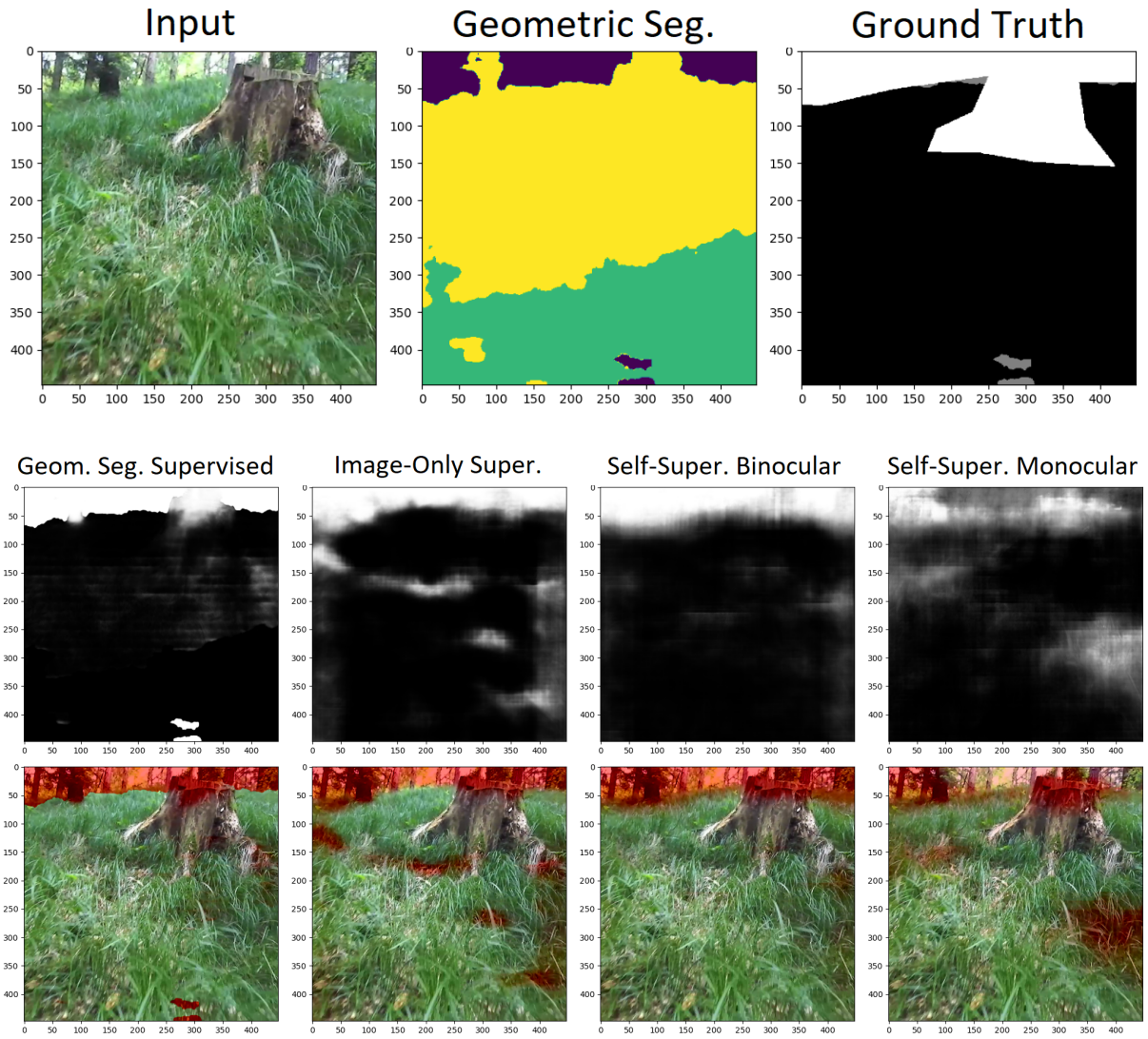



Figure 4.13: Only the tip of the tree stump is classified as dangerous.

# Chapter 5

# Conclussion

We implemented several methods to predict the properties of the off-road terrain. We combined the terrain's geometry with semantic segmentation to provide a reliable prediction.

In the first part of this work, we described several algorithms based on the depth map. The depth map was estimated either from the binocular camera or with the deep learning approach. Both methods provide adequate results for further processing.

The depth maps are subsequently converted into elevation maps by estimating the ground plane using the RANSAC algorithm. The elevation maps are used for obstacle detection. The algorithm works reliably as long as the terrain in front of the vehicle is mostly flat to estimate the ground plane. The depth map is combined with the elevation map to estimate the 'geometric segmentation.' This segmentation is based solely on geometry and is used as a component of semantic segmentation.

The semantic segmentation process relies on deep learning. Multiple methods have been implemented, varying in their utilization of the binocular camera and use of supervised or self-supervised learning. The resulting segments are classified into 'safe' and 'dangerous' classes based on whether the vehicle can safely traverse the terrain.

The performance of the supervised methods generally surpassed that of the self-supervised methods, although the self-supervised approaches still provided satisfactory results. Notably, the self-supervised method utilizing the binocular camera exhibited comparable performance to the monocular-based self-supervise approach. The best results were achieved by combining the manually annotated training data with the geometric segmentation.

One potential future enhancement could involve extending the dataset. The appearance of the off-road terrain changes across seasons but also in response to different weather conditions. Furthermore, the forest environment contains a wide array of diverse obstacles and other elements. To comprehensively capture this variety, a larger dataset is necessary.

# Bibliography


[1] C. Godard, O. Mac Aodha, and G. J. Brostow, "Unsupervised monocular depth estimation with left-right consistency", in *CVPR*, 2017.

[2] J.-M. Dai, T.-A. J. Liu, and H.-Y. Lin, "Road surface detection and recognition for route recommendation", in *2017 IEEE Intelligent Vehicles Symposium (IV)*, 2017, pp. 121–126. DOI: 10.1109/IVS.2017.7995708.

[3] J. Cech, T. Hanis, A. Kononisky, T. Rurtle, J. Svancar, and T. Twardzik, "Self-supervised learning of camera-based drivable surface roughness", in *2021 IEEE Intelligent Vehicles Symposium (IV)*, 2021, pp. 1319–1325. DOI: 10.1109/IV48863.2021.9575288.

[4] D. Vosahlik, J. Cech, T. Hanis, A. Konopisky, T. Rurtle, J. Svancar, and T. Twardzik, "Self-supervised learning of camera-based drivable surface friction", in *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, 2021, pp. 2773–2780. DOI: 10.1109/ITSC48978.2021.9564894.

[5] T. Eppenberger, G. Cesari, M. Dymczyk, R. Siegwart, and R. Dubé, *Leveraging stereo-camera data for real-time dynamic obstacle detection and tracking*, 2020. arXiv: 2007.10743 [cs.RO].

[6] M. Tektonidis and D. Monnin, "Monocular depth estimation for vision-based vehicles based on a self-supervised learning method", in *Autonomous Systems: Sensors, Processing, and Security for Vehicles and Infrastructure 2020*, M. C. Dudzik and S. M. Jameson, Eds., International Society for Optics and Photonics, vol. 11415, SPIE, 2020, p. 114150C. DOI: 10.1117/12.2558478. [Online]. Available: https://doi.org/10.1117/12.2558478.

[7] B. Gao, A. Xu, Y. Pan, X. Zhao, W. Yao, and H. Zhao, "Off-road drivable area extraction using 3d lidar data", in *2019 IEEE Intelligent Vehicles Symposium (IV)*, 2019, pp. 1505–1511. DOI: 10.1109/IVS.2019.8814143.

[8] X. Meng, N. Hatch, A. Lambert, A. Li, N. Wagener, M. Schmittle, J. Lee, W. Yuan, Z. Chen, S. Deng, G. Okopal, D. Fox, B. Boots, and A. Shaban, *Terrainnet: Visual modeling of complex terrain for high-speed, off-road navigation*, 2023. arXiv: 2303.15771 [cs.RO].

[9] R. Schmid, D. Atha, F. Schöller, S. Dey, S. Fakoorian, K. Otsu, B. Ridge, M. Bjelonic, L. Wellhausen, M. Hutter, and A.-a. Agha-mohammadi, "Self-supervised traversability prediction by learning to reconstruct safe terrain", in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022, pp. 12 419–12 425. DOI: 10.1109/IROS47612.2022.9981368.

[10] Y. Yang, D. Tang, D. Wang, W. Song, J. Wang, and M. Fu, "Multi-camera visual slam for off-road navigation", *Robotics and Autonomous Systems*, vol. 128, p. 103 505, 2020, ISSN: 0921-8890. DOI: https://doi.org/10.1016/j.robot.2020.103505. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0921889019308711.

[11] K. Zimmermann, P. Zuzanek, M. Reinstein, and V. Hlavac, "Adaptive traversability of unknown complex terrain with obstacles for mobile robots", in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 5177–5182. DOI: 10.1109/ICRA.2014.6907619.

[12] C. McDonald, *Mean absolute log error (male): A better relative performance metric*, 2023. [Online]. Available: https://towardsdatascience.com/mean-absolute-log-error-male-a-better-relative-performance-metric-a8fd17bc5f75.

[13] O. Ronneberger, P. Fischer, and T. Brox, *U-net: Convolutional networks for biomedical image segmentation*, 2015. arXiv: 1505.04597 [cs.CV].

[14] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography", *Commun. ACM*, vol. 24, no. 6, 381–395, 1981, ISSN: 0001-0782. DOI: 10.1145/358669.358692. [Online]. Available: https://doi.org/10.1145/358669.358692.

[15] D. P. Kingma and J. Ba, *Adam: A method for stochastic optimization*, 2017. arXiv: 1412.6980 [cs.LG].