Monocular depth estimation for a smart white cane

Bachelor's Thesis

Alexander Bayer

ETH Zürich

Advisors: Vaishakh Patil, Dr. Alex Liniger
Supervisor: Prof. Dr. Luc van Gool

January 13, 2022

Abstract

Visually impaired people use a white cane to navigate in the surrounding environment. However, a white cane does not provide sufficient guidance, so they must rely on their intuition to navigate. The user intuition can be improved through the design of a smart white cane. To design such a system, it is necessary to have a reliable understanding of the scene, especially scene depth. One approach would be to obtain the depth using a specialized sensor. However, depth sensors have limitations including weight, reliability, and cost. In this project, we investigate self-supervised depth estimation using a monocular camera as a reliable and cost-effective alternative. Our depth estimation method uses a VGG-16 based perceptual loss rather than directly operating on pixel values. This results in fewer artifacts in the generated disparity maps. We train this method with a newly created dataset using existing videos from the internet as well as self-collected Go-Pro videos. This new dataset contains more than one million diverse training examples consisting of indoor and outdoor scenes from different geographical regions. The training strategy above results in better performance than on the baseline self-supervised model trained on the KITTI dataset or even compared to a supervised method. We compare our model to state-of-the-art self-supervised and supervised methods on standard depth metrics. We evaluate the proposed model on depth estimation benchmarks like the Make3D and NYU depth-v2 test sets. Our approach produces quantitatively and qualitatively superior depth maps compared to existing methods.

Contents

1 Introduction

In 2012 there were 314 million blind people worldwide [6]. Currently, it is common for blind people to navigate using a white cane, and only very few blind have a guide dog. This is due to several problems that come with guide dogs, like their high cost of around CHF 65,000, their care requirements, and their limited usefulness in some situations. The amount of care and activity a dog requires is a problem especially as the average age of a blind person is relatively high. Also, guide dogs can only be used for about one hour until they become unconcentrated. In addition to that, the user and the dog both have to go through extensive training until the guide dog can remember for example the route to the nearest supermarket or train station. Despite being a lot less expensive, the white cane does not provide sufficient guidance in crowded places with a lot of people walking around and in situations with no physical obstacles. The white cane also does not warn the user about obstacles that are on the level of the head of the blind person.

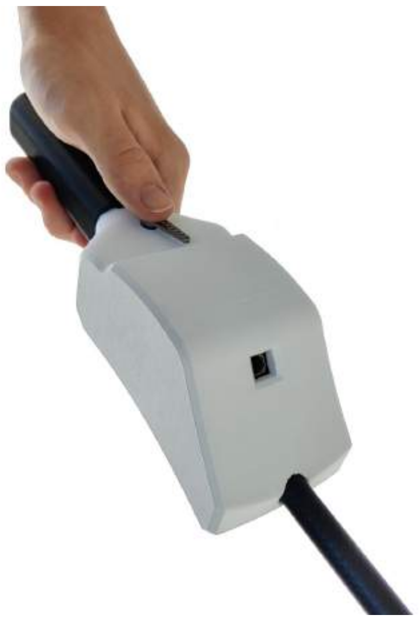

Figure 1: mono camera system with a tensor processing unit, integrated in a white cane

Therefore our goal is to provide a solution that proactively guides in complex situations by enabling the user to feel the direction that is free of obstacles. This has the potential to improve safety, as well as increase the speed at which the blind person can walk. In the past several sensor-based solutions have been developed, which use different vibration patterns to indicate if and how far away an obstacle is [7]. An example of such a system can be seen in figure 3. Another approach is to use an audio signal to inform the user about the fact that there is an obstacle close to the device. While this information is certainly useful for the blind user, the user still does not know which direction to walk in, to not collide with the obstacle. As a solution to this, we propose to use an electrically actuated pointer on the handle of a handheld device which enables the blind person to feel this direction with their thumb. In figure 1 our 3D printed prototype of such a smart white cane is depicted. The corresponding technical drawing from our European patent EP 21000319 is depicted in figure 2.

For estimating a direction that is free of obstacles, having a depth representation in addition to the color image has proven to be useful for autonomous driving, which is a closely related task.

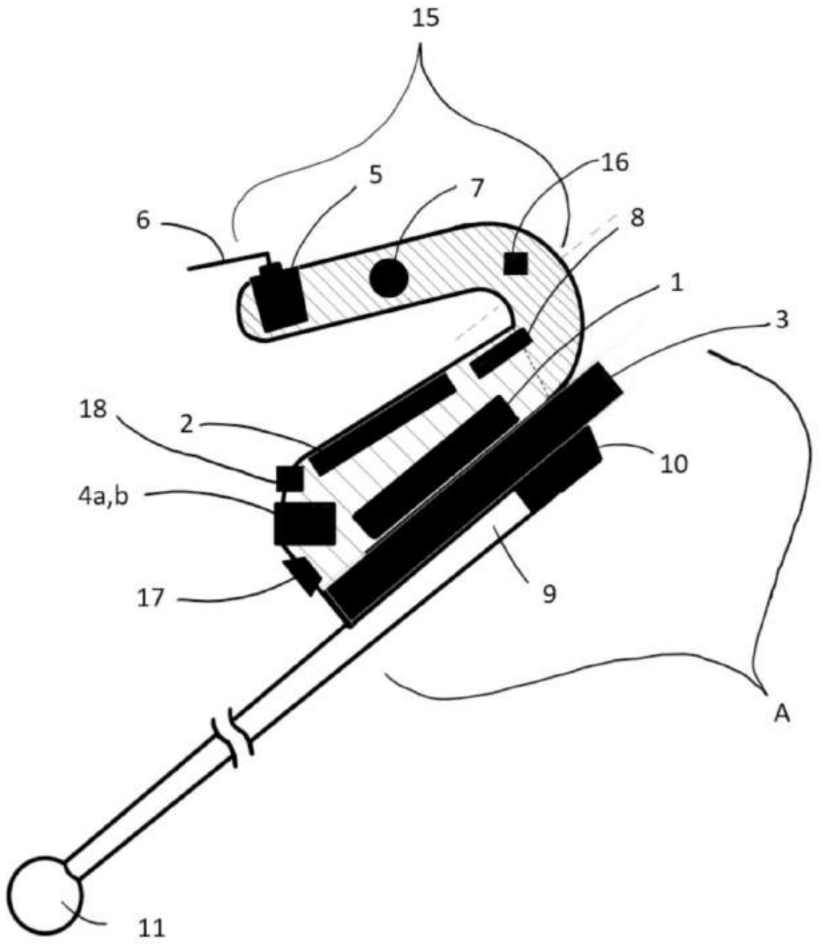

Figure 2: Patent drawing from the European patent EP21000319

Where in autonomous driving it is possible to use LIDAR scanners, in a handheld device this is not possible due to space, weight, and power consumption constraints. Even stereo vision has its problems like low texture surfaces, and relatively high computational cost to perform stereo matching. Monocular Depth Estimation appears to be the best solution for the use case, as using only a single RGB camera is lightweight and saves space. Additionally, in newer processors, it is supported to execute machine learning models on a dedicated tensor processing unit (TPU) in hardware. This not only makes the execution of a depth estimation model fast but also power efficient. Models for monocular depth estimation trained on autonomous driving datasets don't generalize very well to indoor scenes. To be able to train a depth estimation model on a large and diverse dataset with outdoor and indoor scenes, we utilize a self-supervised method to be able to use monocular video data from YouTube videos of city tours. In addition to that, we have recorded wide field-of-view stereo video using two GoPro cameras mounted on a bike (see figure 8) and indoor videos using a GoPro camera (see figure 9). It is, therefore, cost-effective to acquire data for self-supervised learning compared to acquiring ground truth data with a LIDAR scanner.

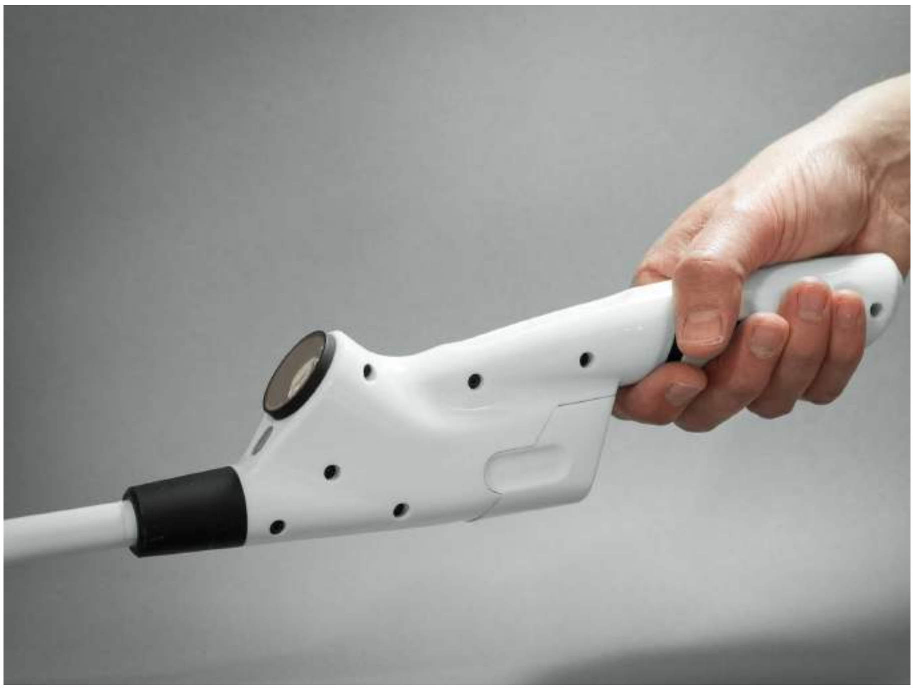
Figure 3: Electronic white cane with vibrational feedback and a LIDAR sensor

2 Related Work

2.1 Supervised Methods

To use a ground truth depth image for supervision during training is the more traditional and straightforward approach to monocular depth estimation. However here the challenge lies in acquiring a large and at the same time also diverse dataset with ground truth dense depth. We discuss some of the state-of-the-art supervised depth estimation Methods below.

2.1.1 Depth Map Prediction from a Single Image using a Multi-Scale Deep Network

Eigen et al. [1] propose a method where one network estimates the depth map at a lower resolution and another network is being used to upscale this coarse prediction to the input resolution. Additionally in their work they propose to use a scale-invariant loss to only estimate the depth of the scene up to a scale factor. The idea behind this is that the coarse-scale network extracts the information about the overall structure of the depth map from visual cues in the input image using four convolution layers and two fully connected layers. This information is then concatenated to the output of the first layer of the fine-scale network, in order for it to use this more global information.

2.1.2 Deeper depth prediction with fully convolutional residual networks

In [5], Laina et al. propose a fully convolutional residual network architecture for supervised depth estimation. It is trained in an end-to-end fashion on the NYU depth v2 dataset. As a general encoder architecture, the performance of AlexNet, VGG and ResNet is compared. As a replacement for the up-convolution operation in the decoder network a novel up-projection operation is being used. As a loss function, the BerHu loss is used, which resulted in more accurate depth maps compared to L2 loss in their experiments.

2.1.3 Deep Ordinal Regression Network for Monocular Depth Estimation

The problem of slow convergence for convolutional neural networks is addressed by Fu et al. in their work [2] by using a spacing-increasing discretization strategy. This means the interval between the discrete depth steps used, increases with increasing depth value, as the depth information contained in the input image becomes less rich as well with increasing depth.

2.1.4 Towards Robust Monocular Depth Estimation: Mixing Datasets for Zero-shot Cross-dataset Transfer

In [8], Ranftl et al. use a scale and shift-invariant loss to use different datasets for training and to evaluate on completely different datasets. This approach makes it possible to have high diversity and also allows to train on large dataset. In their work, they use five different datasets for training: DIML Indoor, MegaDepth, ReDWeb, WSVD, and 3D Movies. In total, they trained their model on more than 1.9 million images. They have found that the performance of their model to increased up to 15% when using a ResNeXt-101-WSL compared to ResNet-50. The ResNeXt-101-WSL encoder has been pretrained on 960 million images from Instagram with the hashtags under the images as a label for weakly supervising the training.

2.2 Unsupervised Methods

In contrast to supervised methods for depth estimation, it is less challenging to acquire a large and diverse dataset for self-supervised methods as 1) no ground truth depth is needed, and 2) monocular (or on some cases stereo) video sequences are needed. The supervision is generated through reprojection of pixel values using generated depth. Here we discuss some of the recent self supervised methods.

2.2.1 Unsupervised CNN for Single View Depth Estimation: Geometry to the Rescue

In [3], Garg et al. use a setup that is similar to an autoencoder, only that the network learns to encode the depth of the scene. The learned depth map can be seen as a feature representation which is then used to warp the input image in a way, such that it resembles the image of the scene from a different camera position. The quality of the reconstruction of this RGB image is then used as a measure of quality for the depth representation. In order to train this pipeline, a stereo camera setup from the KITTI dataset is used.

2.2.2 Unsupervised Learning of Depth and Ego-Motion from Video

A method that requires only monocular video sequences for training is proposed by Zhou et al. [9]. As a supervisory signal, the estimated depth is used to generate a new view of the scene, which can then be compared to the actual next frame of the video. To make this possible, also the relative change in pose needs to be estimated. For this a separate pose network is used. This network takes two frames as input and predicts a relative pose between them.

2.2.3 Digging Into Self-Supervised Monocular Depth Estimation

In [4], Godard et al. extends the method [9]. They propose to use a minimum reprojection loss to avoid occluded regions of the image. In addition to that, they calculate the loss at four different resolutions(scales) to reduce visual artifacts. As there would be a large loss for moving objects in the image, they use a learned mask to exclude such regions from the calculation of the loss. The principle used in monodepth2 is to estimate a depth and a pose translation and rotation for a pair of images. Then the pose and depth are used to reproject the image from the estimated viewpoint of the second image. The loss is then computed by comparing the reconstructed image to the real image that was taken at this position using structural similarity and L1 loss.

3 Method

Even though for the monocular training no stereo setup is used, it still relies on images taken from different points in the scene. Normally the relative position of those points is known, as the baseline of a stereo camera setup is fixed and known. When using video sequences to train however this relative change of the camera poses between two frames needs to be estimated. For this purpose, either a separate pose network is used to estimate the pose change (translation and rotation), or in some cases, the encoder network for depth estimation is also used for this purpose. Once the depth decoder network has computed a hypothesis for the depth, this estimated depth is then used to reproject the RGB image information to the estimated new pose of the camera in the next frame. This way a structural similarity loss or perceptual loss can be computed between the reprojected image and the real image of the next frame. This allows us to get an estimate for how good the depth estimate was. So even though in this setup the images are taken from positions that lay in front of each other, it is still useful to first make clear how depth information about the scene can be extracted using a stereo camera setup.

3.1 Stereo setup

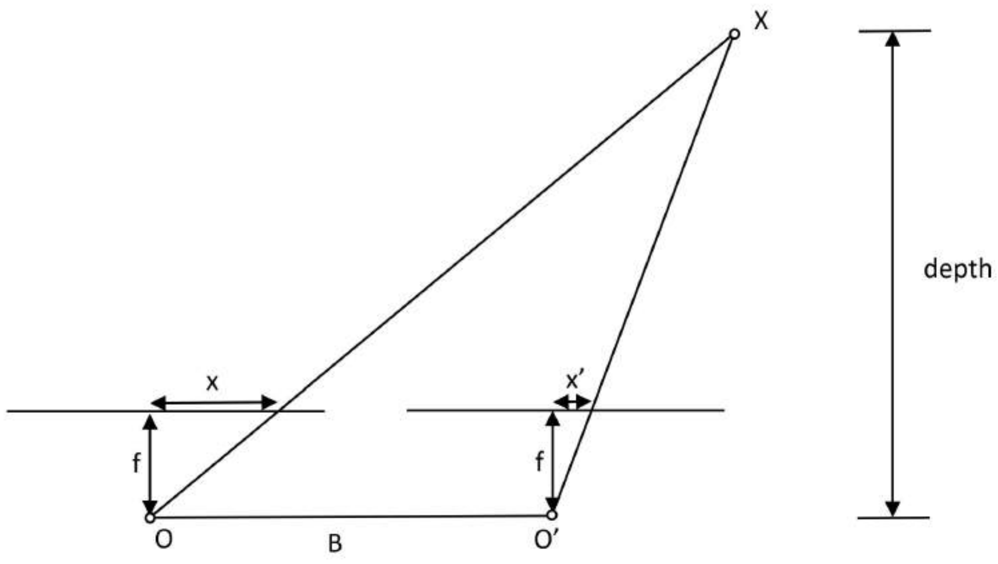

Figure 4: The principle of the stereo camera setup

In the following, the pinhole camera model will be used to elaborate a bit more on epipolar geometry and the stereo setup. Here the lens of the camera is replaced by a pinhole, such that the image gets projected onto the image plane one focal length behind the pinhole. One focal length in front of the pinhole there is the virtual image plane, which will be used to explain the stereo setup, as it is easier to explain it as the image on the virtual image plane is not mirrored. When using only one camera, several points X would be projected on the same point x on the image sensor of the camera. Therefore it is impossible to reconstruct the depth information of a point X from the image. When using two cameras however different points X are projected onto different points x' on the second camera's image sensor. All those points x' lay on the same line, called epipolar line. Because of this when performing stereo matching, the point X only needs to be searched along this epipolar line, which can speed up matching a lot. When draw a line between the two camera centers O and O' we can obtain the epipole e by intersecting it with the image plane. In this the point where all possible epilines l intersect. In most stereo camera setups e is outside of the image, as the cameras are not itself visible in the other camera's image. In order to calculate the position of the epipole, it is necessary to calibrate the stereo setup, which means determining the translation T and the rotation R between the two cameras. To get the Fundamental matrix F we additionally need to determine the camera intrinsics. This is normally done by taking an image of a chessboard pattern with both cameras at the same time. In the stereo camera setup, however the cameras are pointed in the same direction and are mounted in a fixed distance, called baseline relative to each other. As illustrated in figure 4 the point X in the scene projects to different points in the two image planes. The difference between the locations x and x' is called disparity. Using the known values for the baseline B and the focal length f it is possible to calculate the depth z of the point X using equation 1.

$$z = \frac{fB}{x - x'} \tag{1}$$

3.2 Deep learning based monocular depth estimation

If only one camera is available, for example, due to space constraints, stereo matching is not an option anymore. It is however possible to nonetheless obtain a depth map using deep learning based monocular depth estimation. There are two general approaches to monocular depth estimation: supervised methods and unsupervised methods.

3.2.1 Supervised methods

For the depth estimation model most of the time a network consisting of an encoder and a decoder is used, which are connected by skip connections. Often a ResNet encoder is used to encode the RGB image into a feature vector that is passed to the depth decoder network. For the depth decoder network also a convolutional network is used which upsamples the image in several steps. For training ground truth depth or disparity is needed to compute for example the L1 loss of the generated depth map. While this process is relatively straightforward, collecting accurate ground truth depth for large and diverse datasets is not. For this purpose, expensive and large LiDAR sensors are needed.

3.2.2 Unsupervised methods

While using the same general network architecture as supervised methods, no ground truth depth is needed for self-supervised depth estimation. The idea is to use a video sequence or stereo video as a supervisory signal. This is achieved by first generating a depth map of an RGB image using the encoder-decoder network mentioned above. In addition to that, the pose change between this image and the next image is estimated by another neural network. This relative pose is now used to reproject the RGB image to the estimated pose. The generated RGB image is compared to the real next image using L1, SSIM, or perceptual loss, as illustrated in figure 5. In the case of training with stereo video, the Pose change between the two images does not need to be estimated because the relative position of the cameras is known, as the baseline of the setup is known. Training with stereo images has the advantage that moving objects in the image do not pose a problem, as the two images are taken at the same time. In the monocular case, these objects need to be specially treated to not be included in the loss, because their position in the next frame does not only depend on the camera movement and the depth but also their movement.

4 Depth estimation for pedestrian scenes

For our experiments, we use setup from monodpeth2 as it can also be applied to video sequences other than the KITTI dataset. For monocular training, first frame is fed through a ResNet-18 encoder and a depth decoder network. Here skip-connections are used, so a U-net-like architecture is achieved. In addition to that, the current frame and the next frame are fed into a separate pose network, which estimates the relative camera pose change between three images. Then the estimated disparity map is converted to a 3D point cloud. This point cloud is used to reproject the current frame from the estimated camera position of the next frame. In an ideal case, the should resemble the next frame exactly. If the disparity map (or the pose estimate) however is not perfectly accurate, there is a difference to the actual next frame. The L1 difference and the Structural similarity loss function between the reconstructed next frame and the actual next frame can now be used as a training objective. This principle is illustrated in figure 5. In order to reduce artifacts in the generated disparity maps, a perceptual loss has been implemented. For training, a learning rate of 0.0001 has been used, which was reduced by a factor of 10 after 3 epochs. As an optimizer, the Adam optimizer has been used. In addition to the ResNet, used to estimate the pose of the images, a ResNet-18 was used as a depth encoder. The models were trained at a resolution of 384 $\times$ 288 pixels, for at most 7 epochs due to time limitations as one epoch takes about 30 to 50 hours depending on the exact model used on an Nvidia Titan Xp GPU.

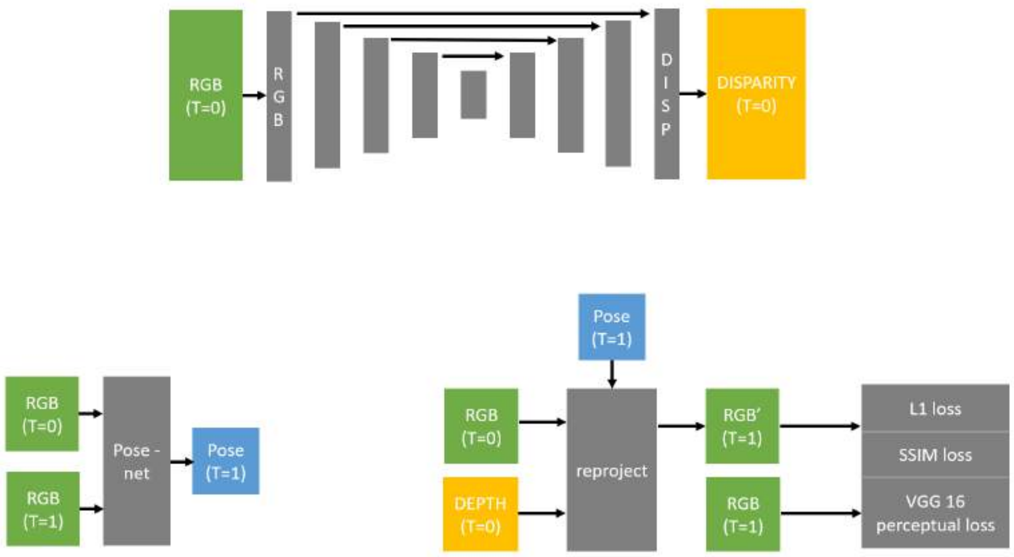

Figure 5: Illustartion of the model used in Monodepth2 for training with monocular video sequences

4.1 Loss functions

As a general loss in monodepth2, the photometric reprojection loss is used. However, to compute the difference between the generated and real image, there are several possibilities. The most straightforward method is to use the L1 loss function (equation 2).

$$L_1 = \sum_{(x,y)\in P} |I^*(x,y) - I(x,y)| \tag{2}$$

Dataset Statistics		
Type of data	Video sequence length	amount of video data
YouTube walking	4:57 h	10.3 GB
Youtube cycling	8:59 h	28.7 GB
Youtube mall	0:12 h	0.41 GB
GoPro indoor	0:33 h	51.5 GB
GoPro outdoor	5:08 h	144.8 GB
total	**19:50 h**	**235.71 GB**

Table 1: Overview over the total amount of data in the dataset and its sources

In addition to that in monodepth2 also SSIM (equation 3) loss is used to compute the structural similarity between the two images. We have however replaced the SSIM loss with a VGG-16 perceptual loss. For this perceptual loss, the RGB images to be compared, are fed into a VGG-16 network. Then the features generated by the VGG-16 network are compared with L1 loss.

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + C_1) + (2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{3}$$

4.2 Training datasets

For all experiments, the model was trained on a mixed dataset consisting of the GoPro video sequences and the video sequences that have been downloaded from YouTube. In total, 1,022,063 training images and 1,000 validation images were obtained. In figure 7, the GPS tracks of the GoPro camera recordings are shown on a map . On the other hand, figure 6 shows a histogram of the speed in km per hour which has been extracted from the GPS metadata. This could in the future be used to extract the frames not at a fixed rate from the video, but at a speed-dependent rate. The absolute time signal contained in the GPS metadata could in the future also be used to synchronize the video footage of the two GoPro cameras used in a stereo configuration (see figure 8 and figure 9) for the stereo training mode of the monodepth2 code. The camera matrix k was determined by taking a picture with the GoPro camera of a chessboard calibration pattern and applying the standard algorithm provided in OpenCV to it. This matrix is used to remove any distortion from the images taken with the GoPro before saving the individual frames. After cropping the images to the middle area the new camera matrix for the cropped images is passed to the implementation of the reprojection algorithm used during training. The cropping is necessary because removing the distortion from the images results in black areas in the image for which no image information is available. In an ideal case, it would be necessary to also provide the reprojection algorithm with the correct camera matrix for the videos downloaded from YouTube. This is however not possible because the type of camera used for these videos is unknown. Another problem was, that some videos were not recorded with a camera that does not automatically correct for lens distortion. Correcting this by hand would be possible by trying different parameters, however this would have taken more time than simply excluding the video and downloading a different one. In addition to that, in almost all videos downloaded from YouTube, there were city maps, logos, and text at the beginning and end of the videos and sometimes also in the middle of the video as well as sequences where the videographer uses the zoom function of the camera. Those events had to be manually identified as well as possible and have been manually removed from the dataset.

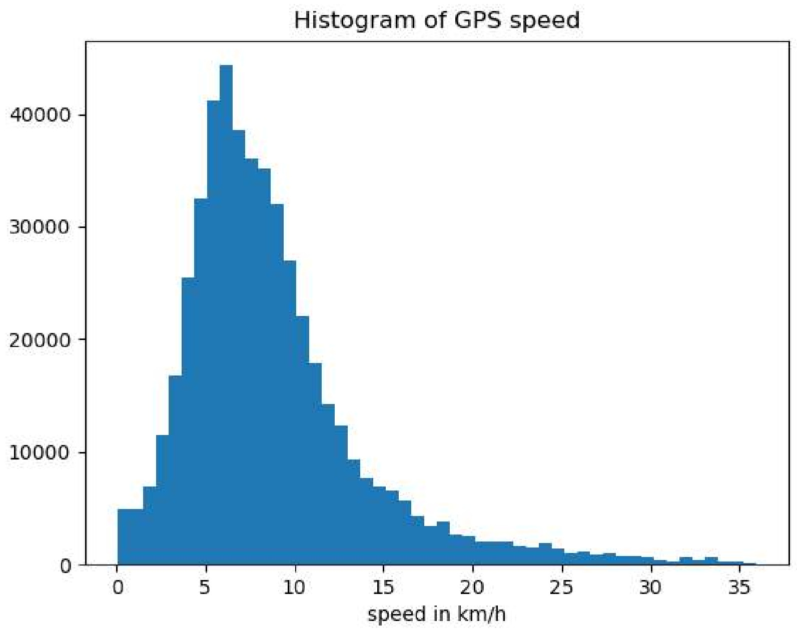

Figure 6: Histogram showing the speed in km/h in the GoPro dataset

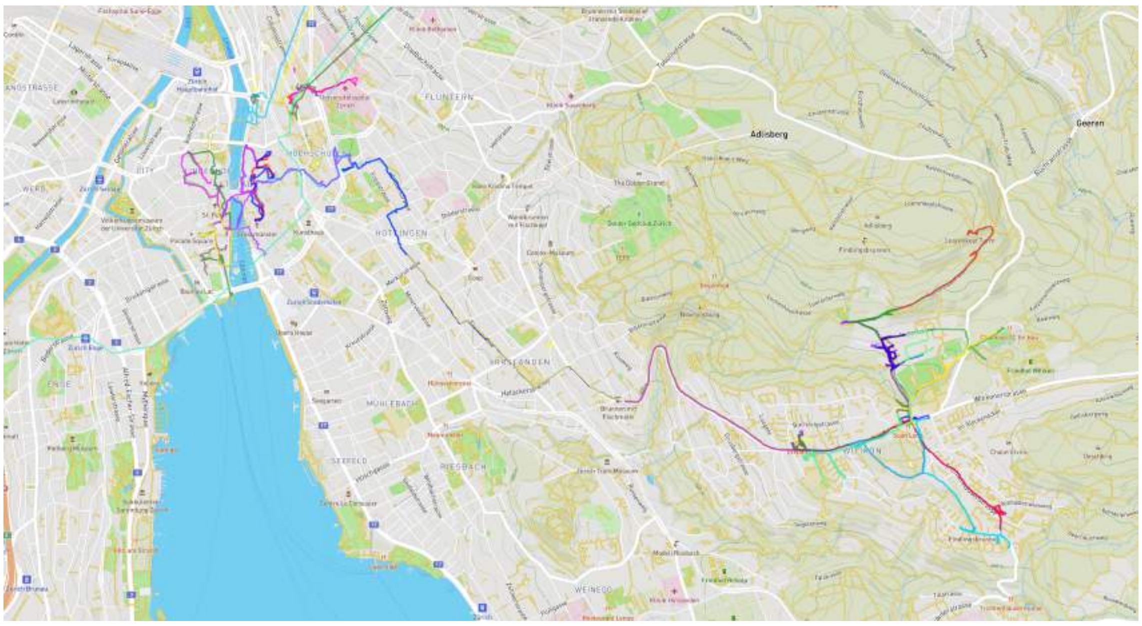

Figure 7: Overview of the GPS tracks recorded by the GoPro cameras

Dataset Statistics		
Type of data	Video sequence length	amount of video data
by night	2:17 h	5.85 GB
grocery store	0:39 h	20 GB
GoPro stereo	4:19 h	2x 117 GB
snow	1:38 h	29.9 GB

Table 2: Overview over the type of data in the videos recorded with the GoPro cameras

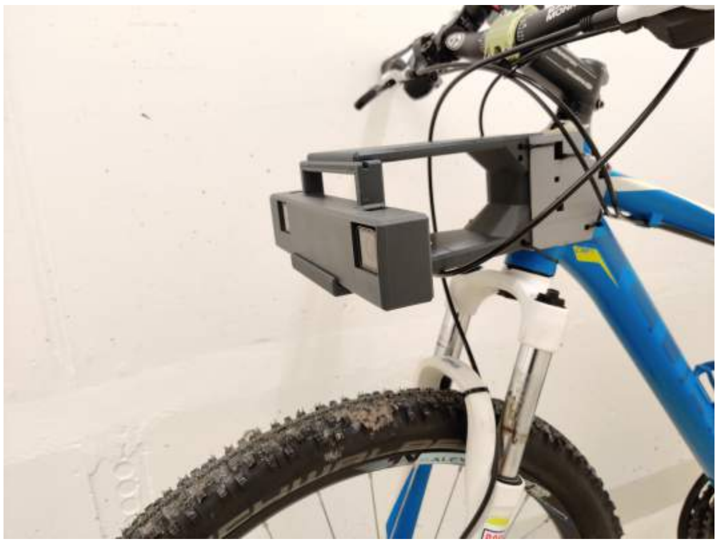

Figure 8: 3D printed GoPro stereo camera setup mounted on a bike with a baseline of 200 mm.

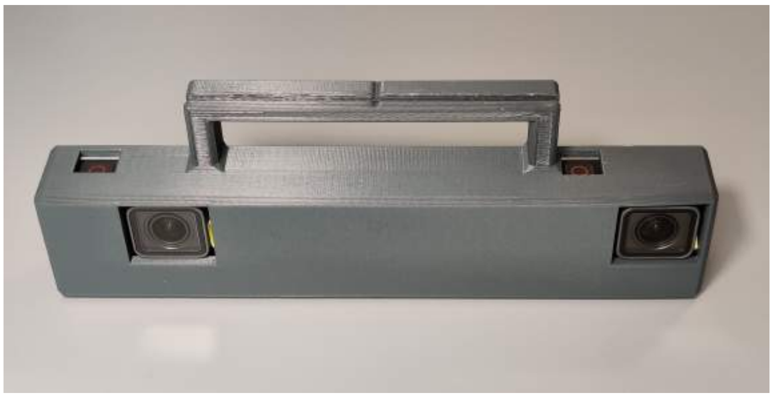

Figure 9: 3D printed handheld GoPro stereo camera setup with a baseline of 200 mm.

4.3 Test datasets

To obtain information on how the trained models perform in real-world indoor and outdoor scenarios, we test our model on different datasets than what it was trained on, so we use a similar approach to this as Ranftl et al. in their work [8]. In particular, we test the models on Make3D and NYUdepth v2. An overview of the different datasets used can be seen in figure 10.

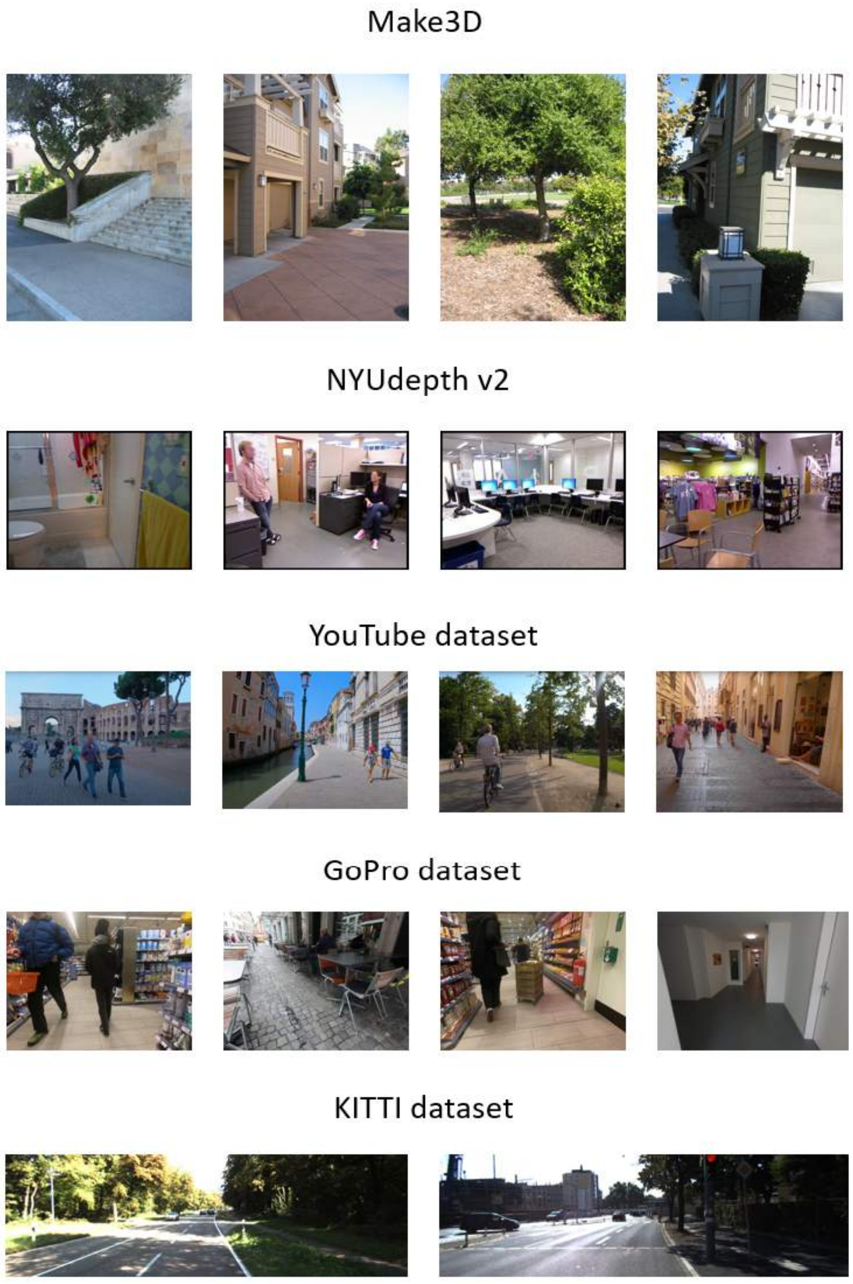

Figure 10: Sample images from the datasets used for testing (Make3D, NYUdepth v2) and Training (YouTube, GoPro)

5 Results

For testing, images from different GoPro video sequences have been used, of which no frames are in the training or validation set, as well as images from NYUdepth and Make3D.

5.1 Evaluation metrics

As evaluation metrics, we use relative L1 and L2 difference (formula 5 and 6), Root-Mean-Square-Error RMSE (formula 6), logarithmic RMSE (RMSLE, see formula 7), and the delta-metric. In contrast to L1, L2 weights large errors over proportionally. RMSLE is included, as it is more robust to outliers in the image than RMSE. The delta-metrics (formula 8, 9, and 10) intuitively speaking count the relative number of pixel values that lie within a 25% (56% and 95% for $\delta_{1.25^2}$ and $\delta_{1.25^3}$) margin of the ground truth value.

$$L_1(relative) = \sum_{(x,y)\in P} \frac{|I^*(x,y) - I(x,y)|}{I^*(x,y)} \tag{4}$$

$$L_2(relative) = \sum_{(x,y)\in P} \frac{[I^*(x,y) - I(x,y)]^2}{I^*(x,y)} \tag{5}$$

$$RMSE = \sqrt{\frac{1}{n} \sum_{(x,y)\in P} (I^*(x,y) - I(x,y))^2} \tag{6}$$

$$RMSLE = \sqrt{\frac{1}{n} \sum_{(x,y)\in P} (log[I^*(x,y)] - log[I(x,y)])^2} \tag{7}$$

$$\delta_{1.25} = \frac{1}{n} \sum_{(x,y)\in P} \begin{cases} 1, & \text{if } \max(\frac{I(x,y)}{I^*(x,y)}, \frac{I^*(x,y)}{I(x,y)}) < 1.25 \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

$$\delta_{1.25^2} = \frac{1}{n} \sum_{(x,y)\in P} \begin{cases} 1, & \text{if } \max(\frac{I(x,y)}{I^*(x,y)}, \frac{I^*(x,y)}{I(x,y)}) < 1.25^2 \\ 0, & \text{otherwise} \end{cases} \tag{9}$$

$$\delta_{1.25^3} = \frac{1}{n} \sum_{(x,y)\in P} \begin{cases} 1, & \text{if } \max(\frac{I(x,y)}{I^*(x,y)}, \frac{I^*(x,y)}{I(x,y)}) < 1.25^3 \\ 0, & \text{otherwise} \end{cases} \tag{10}$$

5.2 Outdoor

As can be seen in the 1st, 3rd, and last row of images in figure 12, the performance for images containing persons is significantly better for our model compared to MiDaS v2. Nonetheless, in row 8 the MiDaS model performed best for the person in the image. Also in comparison with the model trained on the KITTI dataset, the performance of our model is better for persons. However in the last row of figure 12 the shadow of the person holding the camera is mistakenly recognized as a person standing close by our model, not by the other models however. In general, it can be seen that the model trained on the KITTI dataset produces a lot of artifacts in the output, especially for images that are very different compared to those contained in the KITTI dataset. Even though the disparity maps generated by the MiDaS v2 model feature a high smoothness and

Quantitative evaluation on KITTI eigen split							
Method	rel. L1	rel. L2	RMSE	RMSLE	$\delta_{1.25}$	$\delta_{1.25^2}$	$\delta_{1.25^3}$
monodepth2 trained on KITTI, pretrained on ImageNet	**0.115**	**0.902**	**4.862**	**0.193**	**0.877**	**0.959**	**0.981**
monodepth2 trained on our dataset, pretrained on ImageNet	0.169	1.465	6.924	0.269	0.747	0.909	0.959
monodepth2 trained on our dataset, pretrained on ImageNet and KITTI	0.177	1.453	7.057	0.279	0.721	0.899	0.958
monodepth2 trained on our dataset with VGG16 perceptual loss	0.169	1.475	7.182	0.279	0.739	0.901	0.955

Table 3: Quantitative evaluation on the KITTI test set: monodepth2 model trained on KITTI best on the KITTI test data

sharper edges at object boundaries than their models, they often lack fine detail, as can be seen in row 5 of figure 12. In the 6th row of images in figure 12 the model from MiDaS v2 estimated that the train track would be on the same height as the platform, whereas our model recognizes that the train track is actually on a lower level than the platform.

Similarly to the example from the train station, the model from MiDaS v2 in contrast to our model does not recognize that there is a step down to the street, but shows the street to be on the same level as the sidewalk in the 3rd row of figure 11. Also our model provides a more accurate disparity map in the 1st row of figure 11 compared to both MiDaS v2 and the monodepth2 model trained on the KITTI dataset. Even though there is an artifact in the image (encircled in green), the person in the image is most accurately recognized by our model, compared to the other models in the last row of figure 11. In the last row of figure 11 it becomes visible that using the perceptual loss for training reduced the artifacts in the image significantly in comparison to the models trained with the SSIM loss. This effect can also be observed for the image in the 8th row of figure 12.

5.3 Indoor

When comparing the indoor performance of the models, the visual artifacts of the model trained on KITTI become even more apparent in all rows of figure 13. In the 3rd row in figure 13 it can be seen that our model seems to struggle with correctly recognizing the edge of the wall and the disparity map shows a discontinuity where it should not be. In the 4th row, artifacts on the floor can be seen in the disparity map of our model, which result from the joints between the floor tiles visible in the RGB image. In the same row however, encircled in green it can be seen that the shelf it not recognized by the MiDaS v2 model, but is recognized by our model. There is also more detail in the left part of the disparity map of our model compared to the other disparity maps. In the last row of figure 13 our model produces a disparity map which contains more detailed information about the background as there is no railing visible as encircled in green in the disparity map from MiDaS v2 and also in the disparity map from monodepth2 trained on KITTI.

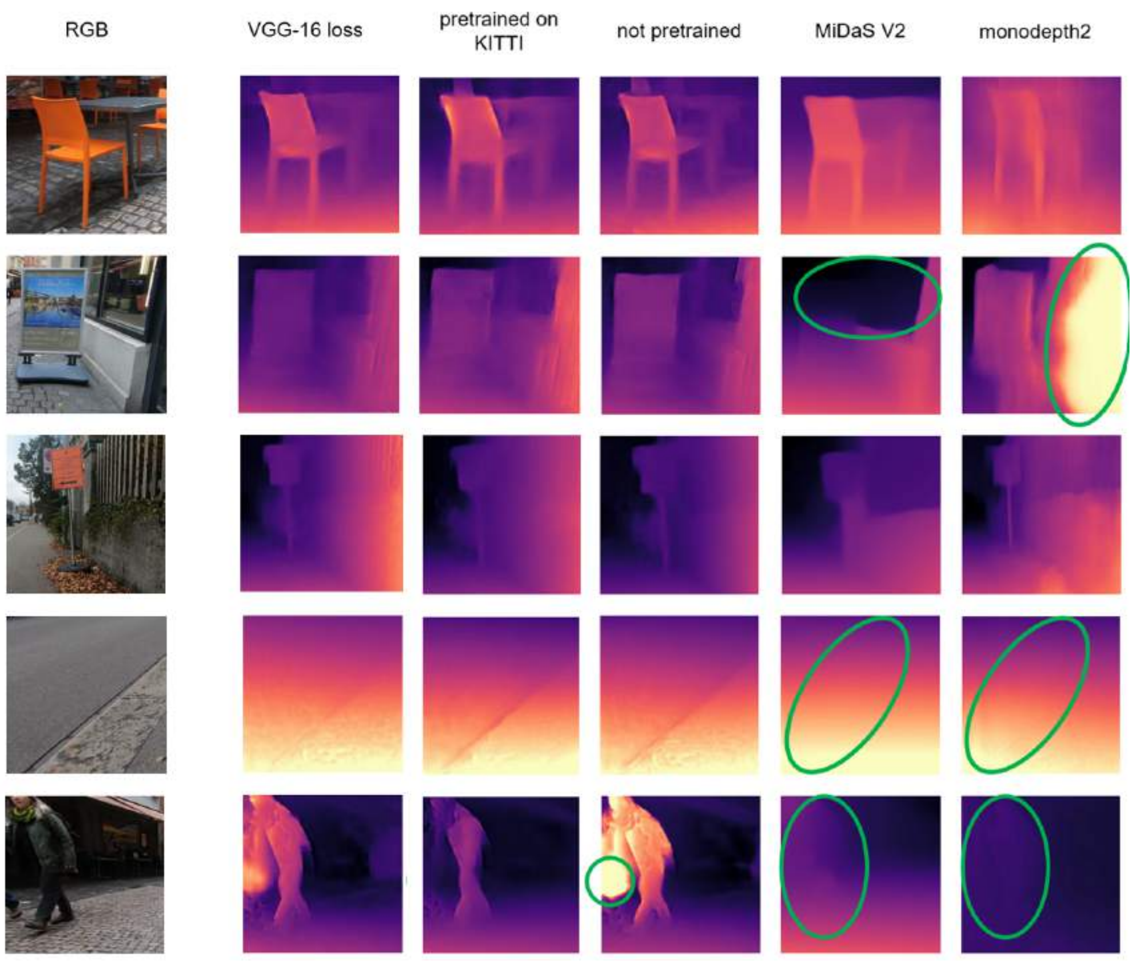

Figure 11: Zoomed in version of predicted depth of the test images.

Quantitative evaluation on NYU depth							
Method	rel. L1	rel. L2	RMSE	RMSLE	$\delta_{1.25}$	$\delta_{1.25^2}$	$\delta_{1.25^3}$
monodepth2 trained on KITTI, pretrained on ImageNet	0.407	0.244	0.321	0.433	0.437	0.719	0.866
monodepth2 trained on our dataset, pretrained on ImageNet	**0.236**	**0.043**	**0.139**	**0.289**	0.632	**0.873**	0.952
monodepth2 trained on our dataset, pretrained on ImageNet and KITTI	0.238	0.044	0.141	**0.289**	**0.628**	0.872	0.953
monodepth2 trained on our dataset with VGG16 perceptual loss	0.239	0.044	0.143	0.296	0.619	0.865	0.950
MiDaS v2	0.291	5.077	13.810	0.321	0.535	0.815	**0.965**

Table 4: Quantitative evaluation on NYU depth: monodepth2 model trained on our datasets performs best on the NYUdepth test data with a large margin to the monodepth2 model trained on KITTI

Quantitative evaluation on Make3D							
Method	rel. L1	rel. L2	RMSE	RMSLE	$\delta_{1.25}$	$\delta_{1.25^2}$	$\delta_{1.25^3}$
monodepth2 trained on KITTI, pretrained on ImageNet	0.357	0.084	0.168	0.503	0.439	0.679	0.817
monodepth2 trained on our dataset, pretrained on ImageNet	0.305	0.048	0.140	0.538	0.473	0.667	0.787
monodepth2 trained on our dataset, pretrained on ImageNet and KITTI	**0.281**	**0.042**	**0.132**	**0.481**	**0.510**	**0.715**	**0.831**
monodepth2 trained on our dataset with VGG16 perceptual loss	0.295	0.045	0.135	0.501	0.499	0.698	0.812
MiDaS v2	0.793	3.894	9.838	0.694	0.379	0.598	0.707

Table 5: Quantitative evaluation on Make3D: monodepth2 model trained on our datasets and pretrained on KITTI performs best on the Make3D test data

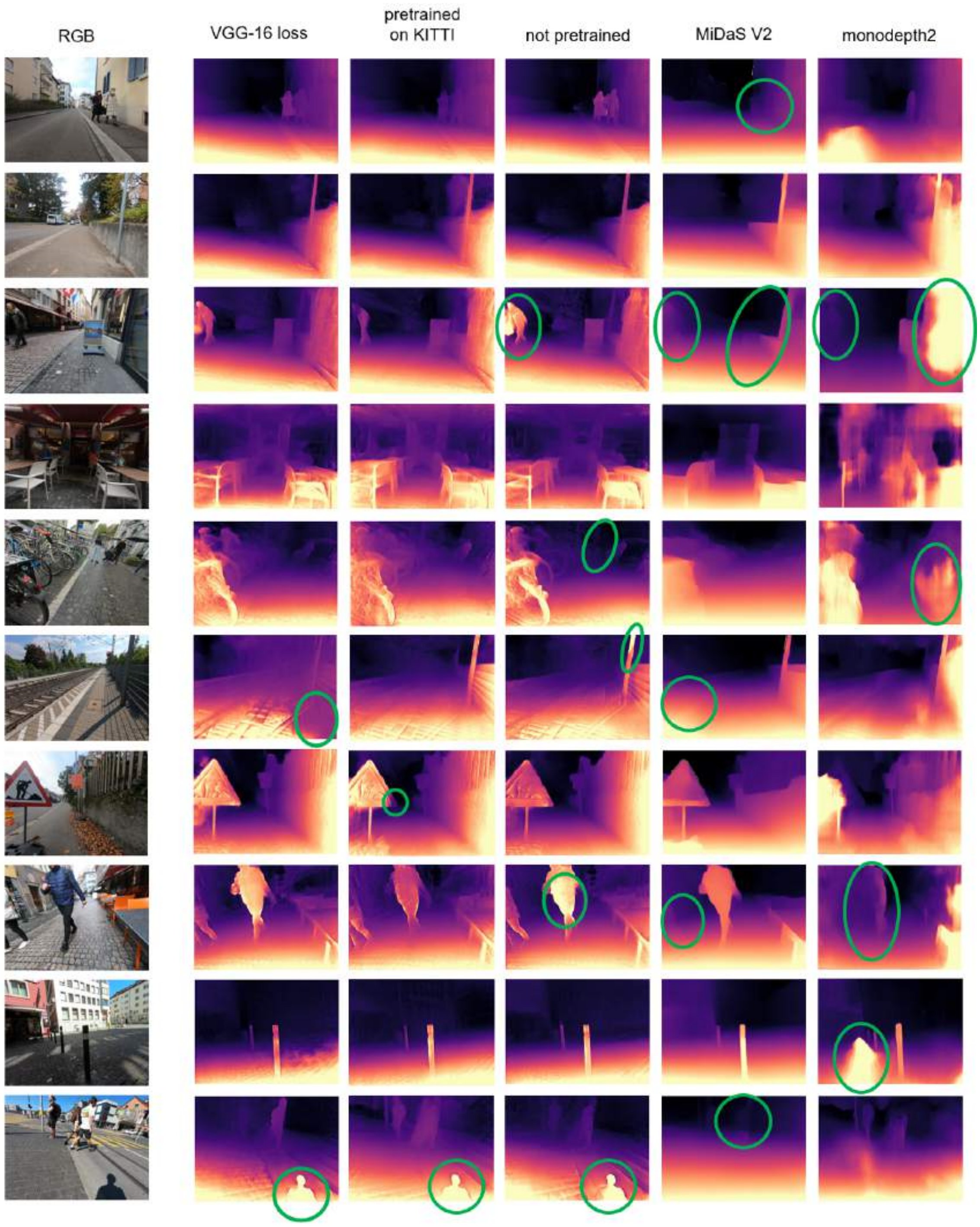

Figure 12: Qualitative comparison between different models on outdoor dataset.

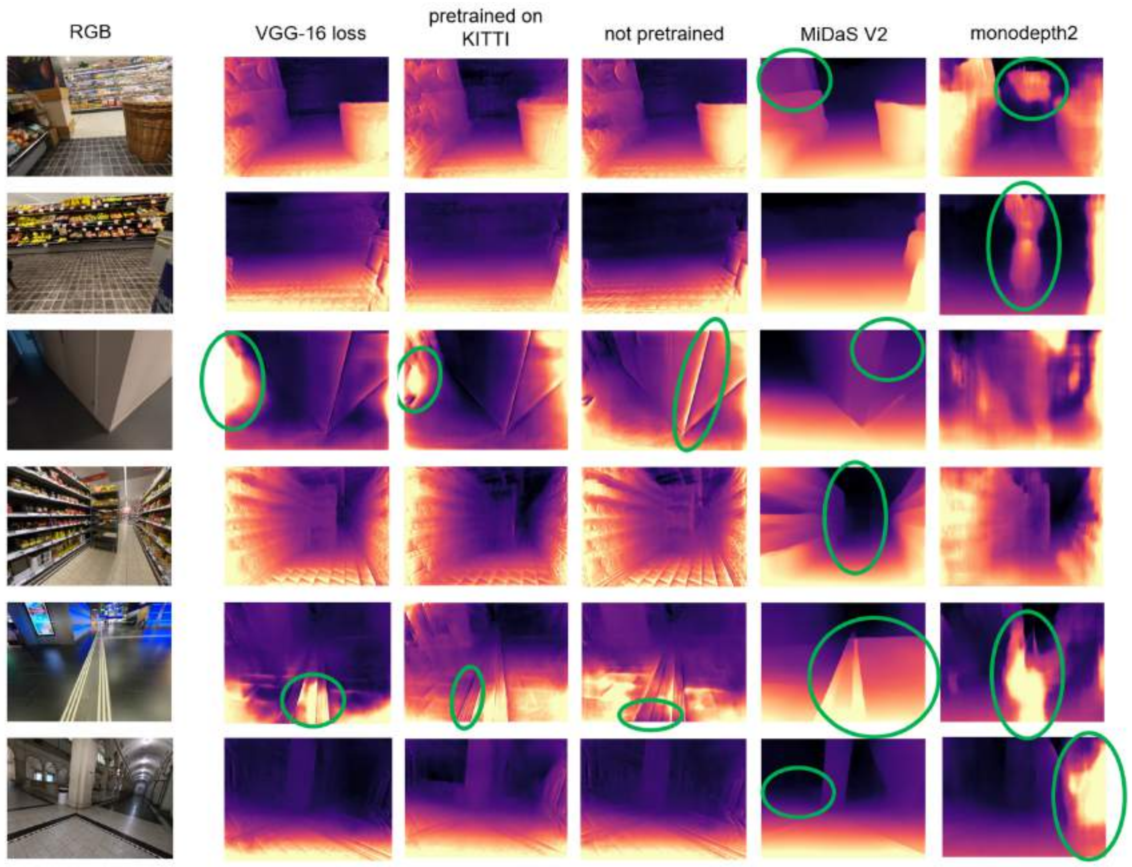

Figure 13: Qualitative comparison between different models on indoor dataset.

6 Conclusion

In this project, we trained a self supervised monocular depth estimation method for development of a smart white cane. We observed that training on a large and diverse dataset of images from city-tour YouTube videos and GoPro videos improved the accuracy of the depth estimation significantly especially for the scenes containing persons. We also noticed better indoor dataset performance in comparison with the other methods. We have found that replacing the SSIM loss used by previous methods by a perceptual loss improves the quality of the generated depth maps by reducing visual artifacts. Both outdoor and indoor scenes, are important for providing accurate guidance to a visually impaired person. A conventional white cane proves to be insufficient especially in the situations like complex indoor environments or where scene is crowed with people. The results of the such scenes can be further improved by training the model with stereo sequences. Using stereo images would help to learn the the depth of objects which move relative to the camera especially people as they move at almost the same speed as the visually impaired person. We record and train on additional stereo video sequences to address this problem. In addition, it should be also possible to extract frames at a speed-dependent rate and use the gyroscope values from the GoPro metadata to replace the optical rotation estimation to further improve the results. The predictions can be further refined by collecting and training on additional indoor data, as the large part of our dataset is outdoor. Training on a additional small amount of ground truth depth data also might improve the accuracy.

References

[1] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network, 2014.

[2] Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, and Dacheng Tao. Deep ordinal regression network for monocular depth estimation, 2018.

[3] Ravi Garg, Vijay Kumar BG, Gustavo Carneiro, and Ian Reid. Unsupervised cnn for single view depth estimation: Geometry to the rescue, 2016.

[4] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J. Brostow. Digging into self-supervised monocular depth prediction. October 2019.

[5] Iro Laina, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, and Nassir Navab. Deeper depth prediction with fully convolutional residual networks, 2016.

[6] World Health Organization. Action plan for the prevention of avoidable blindness and visual impairment 2009-2013. `https://www.who.int/blindness/ACTION_PLAN_WHA62-1-English.pdf`, 2012. [Online; accessed 18-Dec-2021].

[7] Rosalie Pyun, Yeongmi Kim, Pascal Wespe, Stefan Schneller, and Roger Gassert. *Advanced Augmented White Cane with Obstacle Height and Distance Feedback*. IEEE, Piscataway, NJ, 2013. IEEE 13th International Conference on Rehabilitation Robotics (ICORR 2013); Conference Location: Seattle, WA, USA; Conference Date: June 24-26, 2013.

[8] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer, 2020.

[9] Tinghui Zhou, Matthew Brown, Noah Snavely, and David G. Lowe. Unsupervised learning of depth and ego-motion from video, 2017.

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Declaration of originality

The signed declaration of originality is a component of every semester paper, Bachelor's thesis, Master's thesis and any other degree paper undertaken during the course of studies, including the respective electronic versions.

Lecturers may also require a declaration of originality for other written papers compiled for their courses.

I hereby confirm that I am the sole author of the written work here enclosed and that I have compiled it in my own words. Parts excepted are corrections of form and content by the supervisor.

Title of work (in block letters):

Monocular depth estimation for a smart white cane

Authored by (in block letters):
For papers written by groups the names of all authors are required.

Name(s):	**First name(s):**
Bayer	Alexander

With my signature I confirm that
- I have committed none of the forms of plagiarism described in the 'Citation etiquette' information sheet.
- I have documented all methods, data and processes truthfully.
- I have not manipulated any data.
- I have mentioned all persons who were significant facilitators of the work.

I am aware that the work may be screened electronically for plagiarism.

Place, date	**Signature(s)**
Zürich, 13.01.2022	Alexander Bayer

For papers written by groups the names of all authors are required. Their signatures collectively guarantee the entire content of the written paper.