Minimal Transfer Learning for Monocular Depth Estimation

Luigi Pizza [*1] **Aryan Sood** [*1] **Alessandro Tazza** [*1] **Federico Villa** [*1]

Abstract

In recent years, transformer-based architectures have significantly advanced computer vision tasks, outperforming traditional models and exhibiting strong transfer learning capabilities. In this work, we leverage pre-trained transformer variants, originally trained on diverse tasks, to address Monocular Depth Estimation, which involves the prediction of per-pixel distance to the camera from a single RGB image. We demonstrate that effective depth estimation can be achieved through transfer learning with minimal fine-tuning. Our final model outperforms a strong baseline on the public test set of the ETHZ CIL Monocular Depth Estimation 2025 competition.

1. Introduction

Monocular Depth Estimation (MDE) is a fundamental task in computer vision, aiming to predict the depth value of each pixel from a single RGB image. Accurate depth perception is critical for a wide range of applications including 3D modeling (Deng et al., 2024), manufacturing (Mahjourian & Nguyen, 2025), autonomous driving (Wang et al., 2020), and robotics (Dong et al., 2021). MDE is needed whenever the 3D structure of a scene is required, but no direct LIDAR or stereo camera measurements are available.

Recovering depth from a single 2D image is inherently a geometrically ill-posed problem, as it requires inferring missing spatial information lost during the projection from 3D to 2D. This challenge can only be resolved by exploiting semantic cues and spatial regularities in images, which transformers (Vaswani et al., 2017) have been shown to excel at, far surpassing more traditional convolution-based architectures such as CNNs and U-Net (Ronneberger et al., 2015).

In this work, we explore the use of pre-trained transformer variants, which were originally trained on different upstream tasks, for monocular depth estimation via transfer learning (Zhuang et al., 2020). We show that with minimal fine-tuning, these models achieve performance that surpasses strong baselines. Our experiments on the ETHZ CIL Monocular Depth Estimation 2025 competition dataset highlight the practical effectiveness and flexibility of transformer-based approaches for dense prediction problems. In particular, we introduce a modified version of the SegFormer model, a hierarchical transformer architecture originally designed for semantic segmentation tasks (Xie et al., 2021). Our variant retains SegFormer's efficient hierarchical transformer encoder but replaces its segmentation-focused decoder with a depth regression head. This adaptation enables the model to predict dense depth maps and yields superior performance compared to established CNN-based architectures, while maintaining high computational efficiency.

2. Models and Methods

2.1. Motivation

The Transformer architectures has become a central component in modern deep learning, particularly within computer vision. Their success is largely attributed to their ability to capture both global context and fine-grained spatial relationships in images. Recent work, such as the Dense Prediction Transformer (DPT) (Ranftl et al., 2021), has demonstrated that a shared vision encoder can effectively support dense prediction tasks—including semantic segmentation and depth estimation—using only a lightweight decoder. This suggests that transformer-based encoders learn semantically rich internal representations that generalize well across diverse downstream tasks.

Our experiments indicate that SegFormer, despite its relatively compact architecture, learns high-quality intermediate features that exhibit representational properties similar to those of significantly larger models; furthermore, we show that minimal fine-tuning is enough to achieve good results.

2.2. Dataset and Data Preprocessing

Our models were trained on a dataset comprising 23,971 RGB training images and 650 test images, all at a resolution of 426×560 pixels.

During preliminary experiments, we noticed a significant

[*]Equal contribution [1]ETH Zürich. Correspondence to: Luigi Pizza <lpizza@ethz.ch>, Aryan Sood <arsood@ethz.ch>, Alessandro Tazza <atazza@ethz.ch>, Federico Villa <fvilla@ethz.ch>.

Computational Intelligence Lab FS2025, ETH Zuerich

discrepancy between validation performance and final results obtained from the Kaggle submission system. This inconsistency was traced to the presence of similar images in the dataset, likely captured from identical scenes with only minor changes in camera orientation or position (see appendix A Figure 4 for some examples). As a result, random data splitting introduced strong correlations between the training and validation sets, leading to overoptimistic validation scores.

To mitigate this issue, we implemented a similarity-based clustering strategy. First, image embeddings were generated using MobileNet (Howard et al., 2017). Then, pairwise cosine similarity scores were computed between all images. Pairs of images with a cosine similarity above 0.9 were connected to form a similarity graph. Using this graph, clusters of connected components were identified.

This approach is a simplified variant of the broader community detection problem, which seeks to partition a graph into groups of densely connected nodes. Since community detection is known to be NP-hard, we adopted a high similarity threshold (0.9), which empirically results in non-overlapping, non-interacting clusters.

To construct balanced and independent training and validation splits, we sorted the clusters by size and selected clusters for the validation set until approximately 20% of the training data was included. The remaining clusters formed the training set. This strategy effectively prevents data leakage by ensuring that visually similar images do not appear across both training and validation sets.

2.3. Training Loss

The metric optimized during training is the Scale-Invariant Log Loss (Eigen et al., 2014), which, given the predictions y and the targets $y^\star$, is calculated as follows:

$$\mathcal{L}(y, y^\star) = \frac{1}{n}\sum_{i=1}^{n} d_i^2 - \frac{\lambda}{n^2}\left(\sum_{i=1}^{n} d_i\right)^2$$

where $d_i = \log y_i - \log y_i^\star$ and λ is a hyperparameter.

The parameter λ controls how much scale invariance we want. To understand how λ influences the scale invariance take $y = \eta y^\star$, so assume that the model correctly predicts the depth but up to a scale η. The distance becomes $d_i = \log y_i - \log y_i^\star = \log \eta$ and the loss becomes:

$$\begin{aligned}\mathcal{L}(y, y^\star) &= \frac{1}{n}\sum_{i=1}^{n} d_i^2 - \frac{\lambda}{n^2}\left(\sum_{i=1}^{n} d_i\right)^2 \\ &= (\log \eta)^2 - \lambda(\log \eta)^2 \\ &= (1-\lambda)(\log \eta)^2\end{aligned}$$

Setting $\lambda = 1$ makes the loss completely scale-invariant, as it will be zero for every η. On the other hand, setting $\lambda = 0$ recovers the full L2 distance, making the loss fully scale aware. Typically, a value like $\lambda = 0.5$ is chosen to balance between scale invariance and the model being scale aware.

2.4. SegFormer Architecture

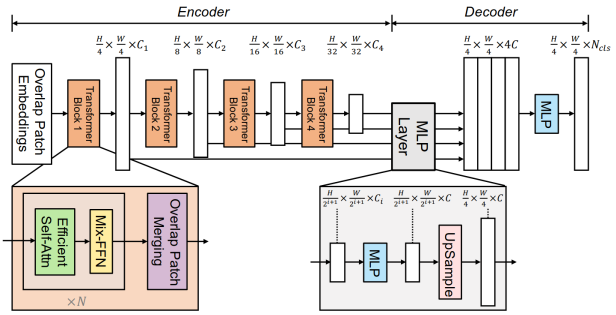

Figure 1. SegFormer model as described in (Xie et al., 2021). We adapt the model such that the output has only one channel, that is $N_{cls} = 1$.

SegFormer is a model designed for semantic segmentation introduced in (Xie et al., 2021). Semantic Segmentation is a computer vision task that involves dividing an image into multiple segments, each representing a specific object. As seen in Figure 1, the SegFormer model is composed of two parts, a hierarchical Transformer encoder and a lightweight MLP decoder. Here, we describe the main parts of the SegFormer architecture.

Hierarchical Feature Representation: Unlike ViT, which produces a single resolution feature map, each block of the SegFormer encoder generates feature maps at multiple scales. As a result, features are extracted at various levels of spatial resolution. Starting from an image of size $H \times W \times 3$, the first transformer block produces a feature map of size $\frac{H}{2} \times \frac{W}{2} \times C_1$, the second block outputs a map of size $\frac{H}{4} \times \frac{W}{4} \times C_2$, and so on, until the last block produces a feature map of size $\frac{H}{32} \times \frac{W}{32} \times C_4$. This can be beneficial for tasks like segmentation and depth estimation, as they need details and contextual information both at high and low level resolutions.

Overlapped Patch Merging: To actually go from a feature map of size $\frac{H}{2^{i+1}} \times \frac{W}{2^{i+1}} \times C_i$ to a feature map of size $\frac{H}{2^{i+2}} \times \frac{W}{2^{i+2}} \times C_{i+1}$, standard convolutional filters are applied. Here, K, S, and P refer to the kernel size, stride, and padding used in the convolution operation. Unlike ViT, using convolution (with $K > S$) allows overlapping between patches, which helps to capture the local context among neighboring patches. This is especially important in tasks like segmentation and depth estimation, where each patch should be aware of its immediate surroundings.

MLP Decoder: The SegFormer's decoder takes the multi-scale features from the encoder as seen in Figure 1 and

produces the final segmentation mask.

For a more detailed explanation of the architecture, particularly the self-attention part, we refer directly to (Xie et al., 2021).

Building upon the SegFormer's original architecture, our model adapts the architecture originally designed for segmentation to the Monocular Depth Estimation problem.

Our model initializes the SegFormer encoder with weights pre-trained on a segmentation dataset. We modify the SegFormer decoder so that instead of predicting a class for each pixel from a large set of categories, it instead outputs one single channel, representing the estimated depth of each pixel in the image. After the decoder, our model upsamples the images by bilinear interpolation to the desired resolution of 426×560.

This adaptation is based on the principle of *transfer learning*: a method that consists of adapting a model originally developed for a task to a related task.

Our main contribution has been to adapt SegFormer to solve our depth estimation task. We argue that features learned during semantic segmentation are highly transferable because both tasks rely on understanding spatial structure. In segmentation, the model learns to recognize object boundaries, shapes, and relative spatial arrangements, capabilities that are important to understand the 3D structure of a scene. These features capture how objects are laid out in space, and can provide strong priors about scene geometry that can be later used by the decoder to actually learn to output a depth map.

3. Experiments

3.1. Evaluations

Our models were evaluated on the Kaggle competition "*ETHZ CIL Monocular Depth Estimation 2025*", which established two baseline points: the easy baseline had a threshold (Score 4) at a SI-RMSE loss of 0.31972 and the hard baseline (Score 6) at a SI-RMSE loss of 0.14660. Note that a lower score is better.

3.2. CNN U-Net model

Our initial experiments used the canonical U-Net architecture following the original implementation in (Ronneberger et al., 2015). The model obtained a SI-RMSE score of 0.27101 on the validation set.

Similarly, we adapted a UNet++ architecture, a model originally developed for image segmentation in (Zhou et al., 2018), to be used for monocular depth estimation. The model achieved a SI-RMSE validation loss of 0.36963.

3.3. CNN pretrained ResNet encoder model U-Net decoder

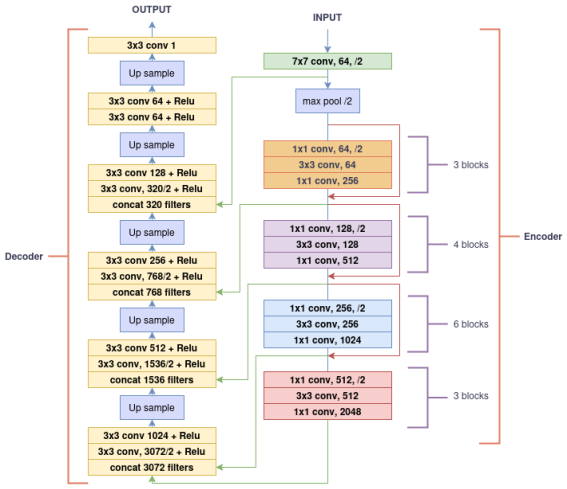

Figure 2. Our resnet based model with skip connection between encoder and decoder. Between each block and sub-block of the encoder there is a Relu activation function.

Based on the findings of our initial experiments, a hybrid architecture was adopted. As described in Figure 2, we used a pretrained ResNet encoder and a decoder composed of a U-Net decoder with skip connections.

The ResNet encoder is based on the model introduced by (He et al., 2015) and utilize ImageNet-pretrained ResNet50 weights.

The ResNet-based model received a public score on the Kaggle competition of 0.16063, showing a significant improvement compared to the U-Net models, achieving almost the required SI-RMSE to beat the hard baseline (0.14660).

3.4. Pretrained SegFormer with fine tuning on the dataset

A significant improvement was achieved by applying transfer learning on the SegFormer model, by using pre-trained weights for the encoder, and modifying the decoder to predict depth mask starting from RGB images.

We fine-tuned the weights of two image segmentation models, both originating from the original SegFormer paper. As stated in the paper, both sets of weights were fine-tuned after the SegFormer encoder was pre-trained on the ImageNet dataset. After training, the classification head was removed and replaced with the MLP decoder. We respectively tested the weights of segformer-b4-finetuned-ade-512-512, a model fine-tuned on *ADE20k* at a resolution of 512×512, and segformer-b5-finetuned-ade-640-640, a model fine-tuned on the same dataset but at a higher resolution of 640×640. The best loss score in our experiments

was achieved using the weights of the segformer-b5 model.

The SegFormer fine-tuned model achieved a public score on the Kaggle competition of 0.12621, just after 4 epochs. Although the model could have benefitted from more fine-tuning, the main objective of our study was to show that a model trained on a segmentation task can easily be adapted for a depth estimation task.

3.5. Mask2Former model

As a last experiment, we decided to try the Mask2Former architecture which, like SegFormer, is a segmentation model. We took the Mask2Former architecture as described in (Cheng et al., 2022), upsampled our image to a resolution of 426×560 pixels and added a double convolutional layer to predict the depth mask.

The Mask2Former-based model received a public score on the Kaggle competition of 0.13343, after 4 epochs. As with SegFormer, we show that Mask2Former, a model trained on a segmentation task, can easily be adapted for a depth estimation task with minimal transfer learning.

4. Results

Model	Validation Loss	Training Loss	Kaggle Public Score
U-Net Based models			
Base U-Net	0.27101	0.32242	–
UNet++	0.36963	0.44006	–
ResNet Based models			
ResNet	0.15886	0.09157	0.16063
ResNet-Transformer	0.17421	0.11357	0.21408
Mask2Former model			
mask2former-swin	0.12976	0.097775	0.13343
SegFormer models			
mask2former-swin	0.13222	0.097775	0.13343
segformer-b4-512-512	0.14236	0.11561	0.14048
segformer-b5-640-640	0.11474	0.15951	**0.12621**

Table 1. **Depth estimation performance across different architectures:** The SegFormer based model achieved a better result compared to the other models, showing a 27.8% improvement over the ResNet based model and a 68.9% improvement over the Base U-Net and the UNet++ model.

5. Experimental Setup

All models were trained using the AdamW (Loshchilov & Hutter, 2019) optimizer, with a learning rate of 10^{-4} and a weight decay of 10^{-3}. The ResNet-based model was trained for 8 hours using a batch size of 16 on a freely available Kaggle GPU. For the SegFormer model, training was also conducted on the Kaggle GPU with a batch size of 16 and the model was trained for 4 epochs. Similarly, the Mask2Former-based model was trained with a batch size of 16 and the model was trained for 4 epochs. The Kaggle GPU used was a Tesla P100 with 16GB of VRAM.

(a) Rgb Image (b) Ground truth depth

(c) ResNet based model (d) SegFormer based model

Figure 3. **Comparison between different methods**: ResNet-based model produces a smoother estimation, but fails to capture detailed depth for every object in the scene(see for example the second chair on the right). The SegFormer-based method predicts better object depth, but produces a less smooth estimation due to the final image upsampling. For a more detailed visualization, see appendix A Figure 5

We chose Kaggle for training as the provided ETHZ student cluster GPUs could only accommodate a batch size of 2 due to VRAM limitations, resulting in a 8 times slower training process.

The RGB channels of the dataset's images were normalized to a zero-mean, unit-variance distribution using ImageNet dataset statistics. This preprocessing step is essential when using pre-trained models, as virtually most of them are pretrained on the ImageNet dataset. Omitting this step can lead to a drop in performance, since the pre-trained weights expect input to be processed in this way.

6. Conclusion

In this work, we present a method for adapting segmentation based models to predict depth from monocular RGB images, with minimal fine-tuning. A limitation of our approach is the need to upsample the images with bilinear interpolation both in the SegFormer-based model and the Mask2Former-based model, which results in less smooth depth maps and the presence of artifacts.

A possible direction for the SegFormer-based model could be to explore improved upsampling strategies or consider adding a U-Net decoder, as done in our ResNet-based model, along with skip connections from the SegFormer encoder. This ensures that the model learns to predict depth by adding differences to the input image. In this way, we can obtain a better, high-resolution output, without the need to up-sample at the end.

References

Cheng, B., Misra, I., Schwing, A. G., Kirillov, A., and Girdhar, R. Masked-attention mask transformer for universal image segmentation, 2022. URL https://arxiv.org/abs/2112.01527.

Deng, K., Liu, A., Zhu, J.-Y., and Ramanan, D. Depth-supervised nerf: Fewer views and faster training for free, 2024. URL https://arxiv.org/abs/2107.02791.

Dong, X., Garratt, M. A., Anavatti, S. G., and Abbass, H. A. Towards real-time monocular depth estimation for robotics: A survey, 2021. URL https://arxiv.org/abs/2111.08600.

Eigen, D., Puhrsch, C., and Fergus, R. Depth map prediction from a single image using a multi-scale deep network, 2014. URL https://arxiv.org/abs/1406.2283.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition, 2015. URL https://arxiv.org/abs/1512.03385.

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., and Adam, H. Mobilenets: Efficient convolutional neural networks for mobile vision applications, 2017. URL https://arxiv.org/abs/1704.04861.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization, 2019. URL https://arxiv.org/abs/1711.05101.

Mahjourian, N. and Nguyen, V. Multimodal object detection using depth and image data for manufacturing parts, 2025. URL https://arxiv.org/abs/2411.09062.

Ranftl, R., Bochkovskiy, A., and Koltun, V. Vision transformers for dense prediction, 2021. URL https://arxiv.org/abs/2103.13413.

Ronneberger, O., Fischer, P., and Brox, T. U-net: Convolutional networks for biomedical image segmentation, 2015. URL https://arxiv.org/abs/1505.04597.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention is all you need, 2017. URL https://arxiv.org/abs/1706.03762.

Wang, Y., Chao, W.-L., Garg, D., Hariharan, B., Campbell, M., and Weinberger, K. Q. Pseudo-lidar from visual depth estimation: Bridging the gap in 3d object detection for autonomous driving, 2020. URL https://arxiv.org/abs/1812.07179.

Xie, E., Wang, W., Yu, Z., Anandkumar, A., Alvarez, J. M., and Luo, P. Segformer: Simple and efficient design for semantic segmentation with transformers. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 12077–12090. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/64f1f27bf1b4ec22924fd0acb550c235-Paper.pdf.

Zhou, Z., Siddiquee, M. M. R., Tajbakhsh, N., and Liang, J. Unet++: A nested u-net architecture for medical image segmentation, 2018. URL https://arxiv.org/abs/1807.10165.

Zhuang, F., Qi, Z., Duan, K., Xi, D., Zhu, Y., Zhu, H., Xiong, H., and He, Q. A comprehensive survey on transfer learning, 2020. URL https://arxiv.org/abs/1911.02685.

A. Visualization

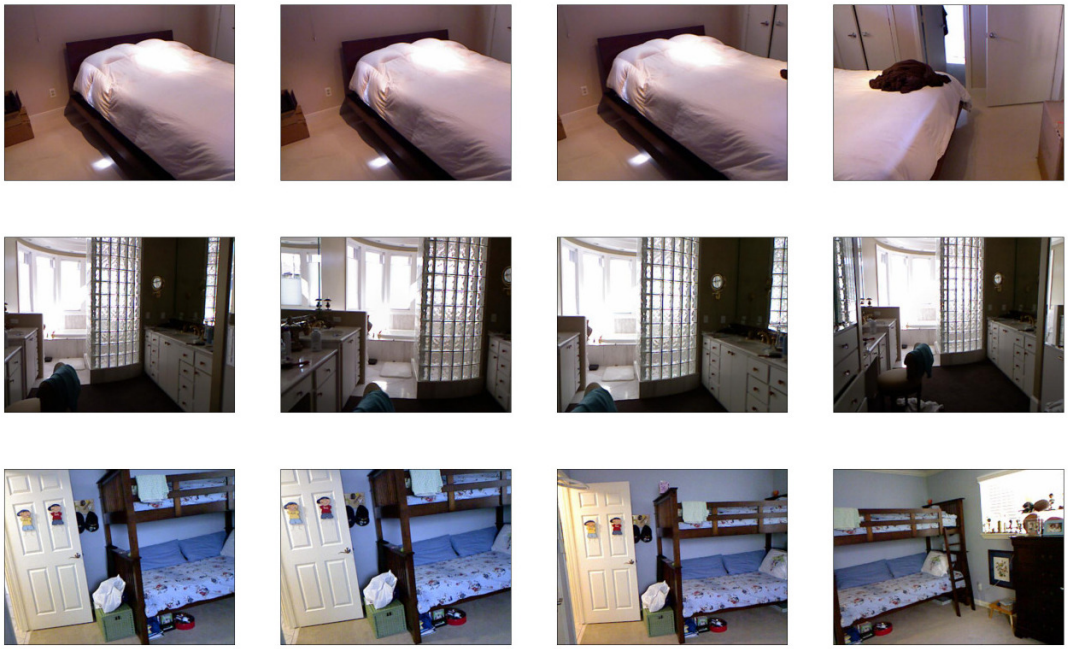

Figure 4. Some of the images clustered after applying the similarity-based clustering strategy described in section 2.2. We see that the clusters contain similar images, taken from slightly different perspectives.

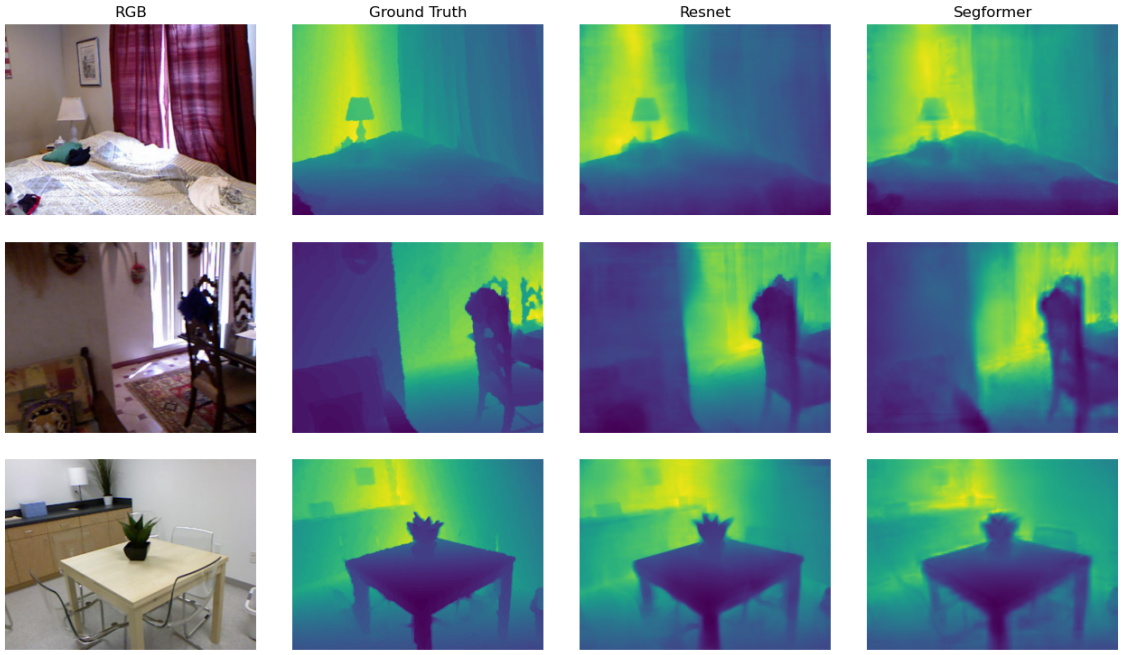

Figure 5. Comparison between the depth masks produced by ResNet, SegFormer, compared to the Ground Truth. The SegFormer model, due to its final upsampling step, produces depth masks with artifacts, less smooth compared to the ResNet-based model. On the other hand, the ResNet-based model misses some of the finer details of the objects when generating depth masks.