Supplementary Material
Out-of-Distribution Detection for Monocular Depth Estimation

Julia Hornauer[1], Adrian Holzbock[1], and Vasileios Belagiannis[2]

[1]Ulm University, Germany , {first.last}@uni-ulm.de
[2]Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany , vasileios.belagiannis@fau.de

1. Further Implementation Details

In this section, we describe our implementation details for the related methods log-likelihood maximization (Log [6, 8]), Monte Carlo Dropout (Drop [3]) and BayesCap (BCap [10]) that we applied to the models Monodepth2 [5], Pixelformer [1] and MonoViT [12].

Log-Likelihood Maximization (Log) When using log-likelihood maximization, the model does not only predict the pixel-wise depth $\hat{\mathbf{d}}$ but also the pixel-wise variance $\mathbf{u}$, which is used as an uncertainty measure. For the supervised trained Monodepth2 [5], we optimize the model with the following loss function:

$$\mathcal{L}_{log} = \frac{|\hat{\mathbf{d}} - \mathbf{d}|}{\mathbf{u}} + \log \mathbf{u}, \tag{1}$$

where the Laplace distribution is assumed as the prior distribution over the model output [8]. In this context, $\mathbf{u}$ represents the variance of the output distribution and $\mathbf{d}$ the ground truth depth.

The supervised trained Pixelformer [1], on the other hand, is optimized with a Scale-Invariant loss:

$$\mathcal{L}_{SILog} = \alpha \sqrt{\frac{1}{n}\sum_i g_i^2 - \frac{\lambda}{n^2}(\sum_i g_i)^2}, \tag{2}$$

where $\alpha = 10$, $\lambda = 0.85$ and n is the number of pixels with available ground truth. We replace the log-scaled difference $g_i = \log \hat{d}_i - \log d_i$ and model the Laplacian distribution as in Equation 1 with

$$g_i = \frac{|\hat{d}_i - d_i|}{u_i} + \log u_i. \tag{3}$$

For Monodepth2 [5] and MonoViT [12] trained in a self-supervised manner with monocular sequences, the loss function proposed in [6] is used during optimization:

$$\mathcal{L}_{log} = \frac{\mathcal{L}_{pm}}{\mathbf{u}} + \log \mathbf{u}, \tag{4}$$

where $\mathcal{L}_{pm}$ is the photometric matching loss and $\mathbf{u}$ the predicted variance.

Monte Carlo Dropout (Drop) To obtain the empirical mean and variance over the model parameters with Monte Carlo Dropout, we use 8 forward passes with activated dropout [9] and dropout probability of 0.2. In the case of Monodepth2 [5], we apply dropout after each *ConvBlock* in the depth decoder. Similarly, we use dropout after each decoder *ConvBlock* for MonoViT [12]. In the case of Pixelformer [1], we apply dropout after the blocks *Q3*, *Q2* and *Q1*.

BayesCape (BCap) For all models, we rely on the BayesCap implementations from [10]. We set $T1$ and $T2$ to 10 and 1, respectively. Furthermore, we reduce $T1$ after each epoch with exponential decay and keep $T2$ fixed. For NYU, we divide the output depth by the maximum depth of 10 meters to scale the depth values in the range of $[0, 1]$ before passing them to the BayesCap model. For KITTI, the output of the depth estimation model is already in the range of $[0, 1]$.

2. Further Visual Results

In Figure 1 to Figure 4, we show further visual examples of our image decoder. The figures show the original input image $\mathbf{x}$ (left), the reconstructed image $\hat{\mathbf{x}}$ (center), and the resulting error map $\mathbf{e}$ (right). In Figure 1 and Figure 2, visual results with Pixelformer [1] and Monodepth2 [5] trained on NYU Depth V2 [7] (NYU) as in-distribution are displayed. We use test images from the same dataset and the OOD dataset Places365 [13] (Places). In Figure 3 and Figure 4, visual results with Monodepth2 [5] and

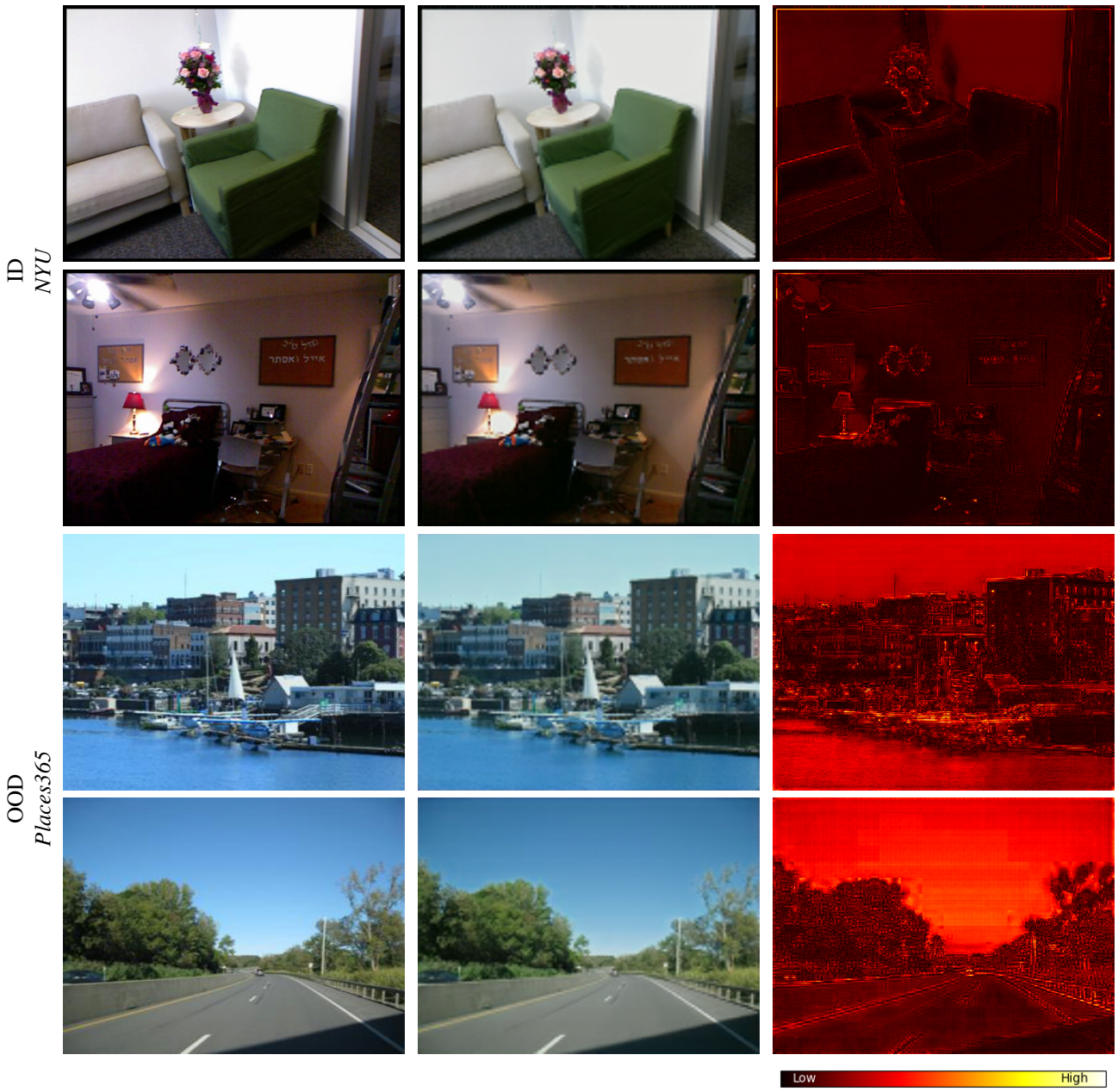

Figure 1: Pixelformer [1] is trained on the in-distribution dataset NYU Depth V2 [7] (NYU). We show test images $\mathbf{x}$ from the in-distribution (ID) dataset NYU and the out-of-distribution (OOD) dataset Places365 [13] on the left. The corresponding reconstructed images $\hat{\mathbf{x}}$ from our image decoder and resulting error maps $\mathbf{e}$ are shown in the middle and right, respectively.

MonoViT [12] trained on KITTI [4] as in-distribution are illustrated. We demonstrate images from the same dataset and the OOD datasets Places365 [13] (Places), India Driving [11] (India), and virtual KITTI [2] (vKITTI). While the error maps of the in-distribution test images show low errors, the images of the different OOD datasets yield high errors.

3. Failure Cases

Two possible failure cases of the proposed image decoder are that in-distribution images may not be reconstructed well but OOD images are reconstructed well. This leads to the images in question being incorrectly categorized as OOD, even though they are in-distribution and vice

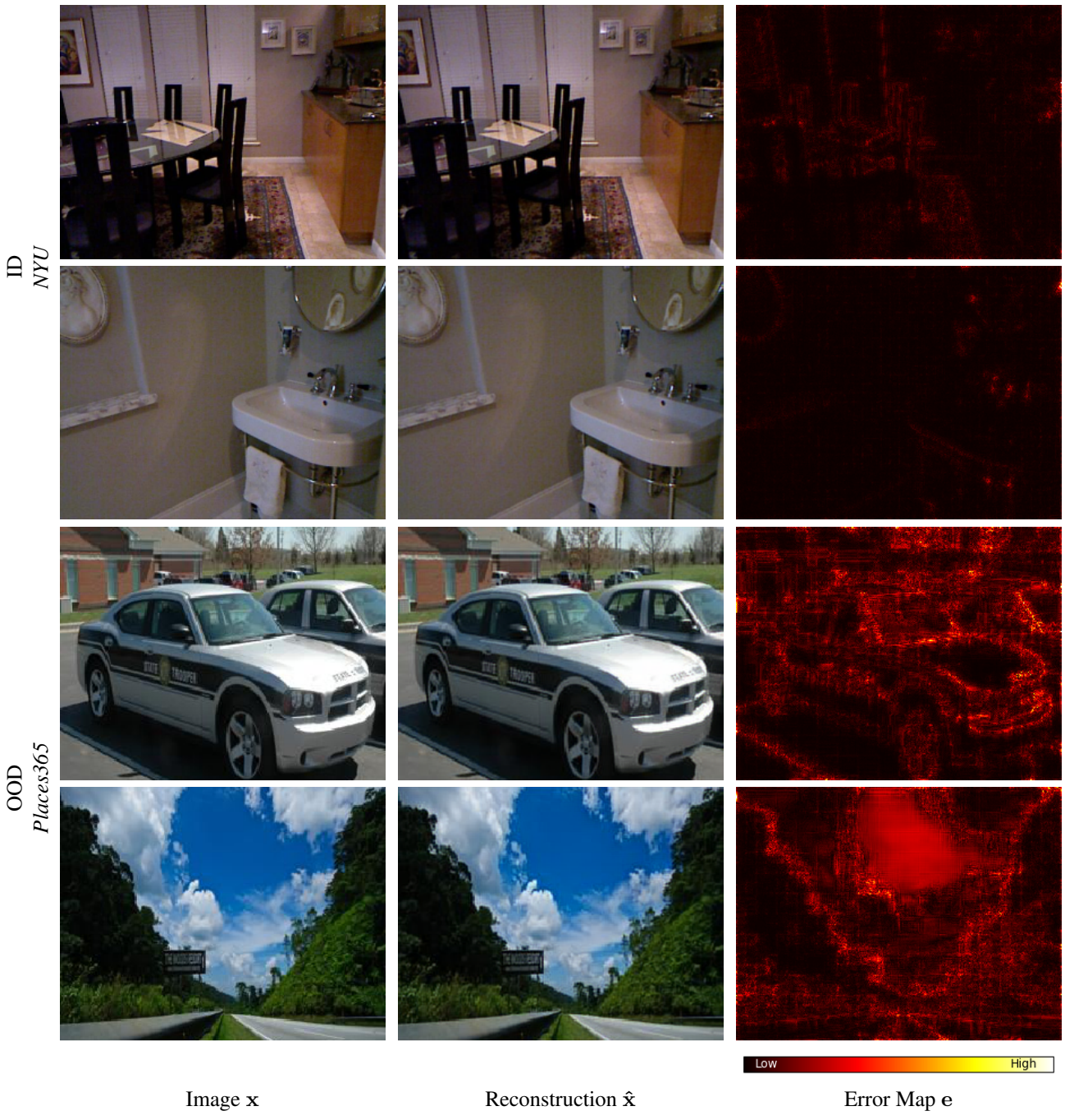

Figure 2: Monodepth2 [5] is trained on the in-distribution dataset NYU Depth V2 [7] (NYU). We show test images $\mathbf{x}$ from the in-distribution (ID) dataset NYU and the out-of-distribution (OOD) dataset Places365 [13] on the left. The corresponding reconstructed images $\hat{\mathbf{x}}$ from our image decoder and resulting error maps $\mathbf{e}$ are shown in the middle and right, respectively.

versa. In Figure 5, failure cases of the image decoder for Monodepth2 trained on NYU are shown. The first failure case shows an in-distribution test image from NYU where the reconstruction error in the window region is high, although it is an in-distribution image. Here, the window blinds cause strange reflections that are not normally seen in other training images. The second example shows an image from the OOD dataset Places365. In this case, the reconstruction error is low but should be high since the image is not covered by the training distribution.

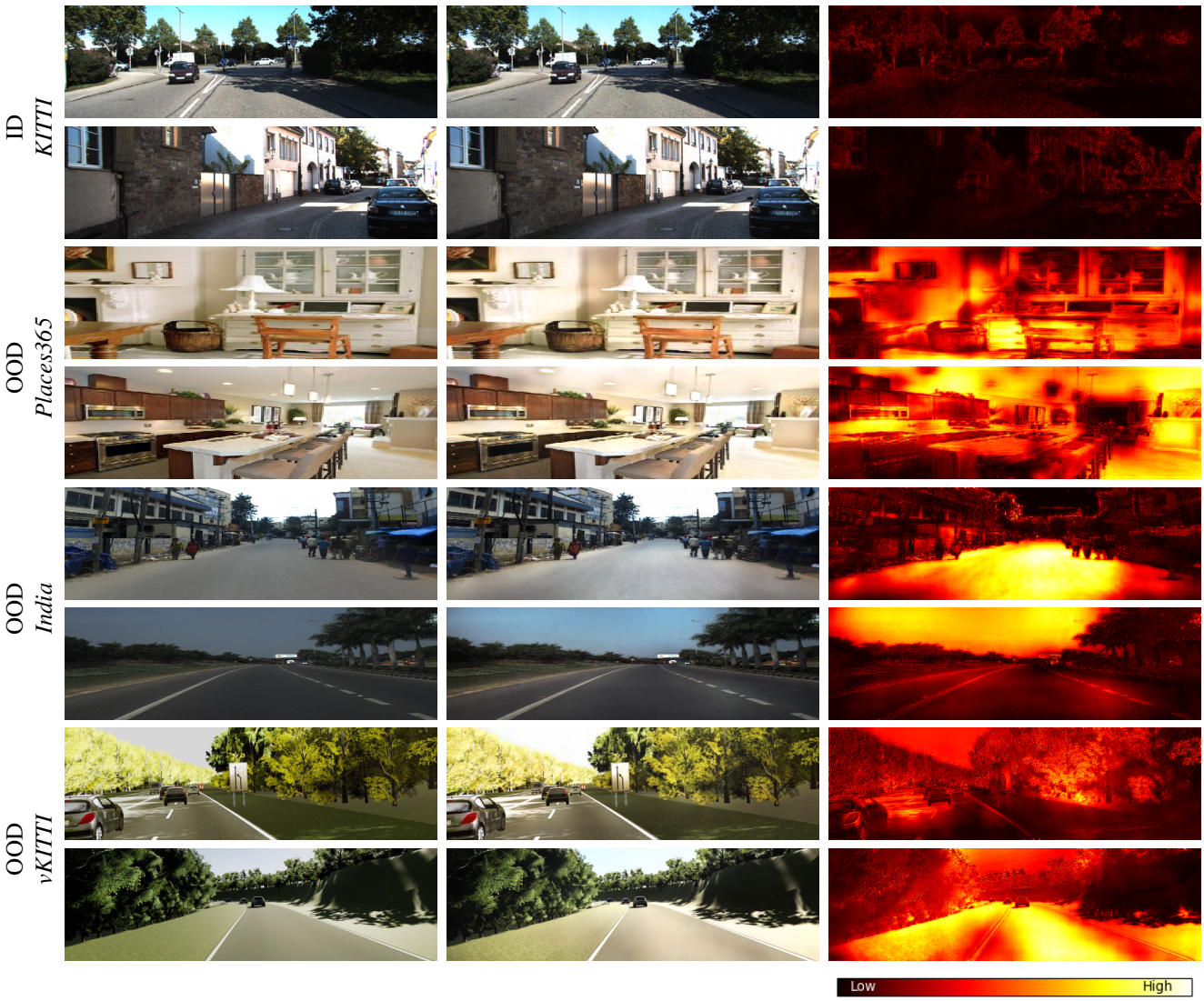

Figure 3: Monodepth2 [5] is trained on the in-distribution dataset KITTI [4]. We show test images $\mathbf{x}$ from the in-distribution (ID) dataset KITTI and the out-of-distribution (OOD) datasets Places365 [13], India Driving [11] (India) and virtual KITTI [2] (vKITI) on the left. The corresponding reconstructed images $\hat{\mathbf{x}}$ from our image decoder and resulting error maps $\mathbf{e}$ are shown in the middle and right, respectively.

Figure 6 demonstrates failure cases of the image decoder for Monodepth2 trained on the outdoor dataset KITTI. The image decoder reconstructs the blue container of the in-distribution test image in a different color scheme. This might happen as no similar object appears in the training data. The test image from the OOD database virtual KITTI (vKITTI), on the other hand, is too well reconstructed.

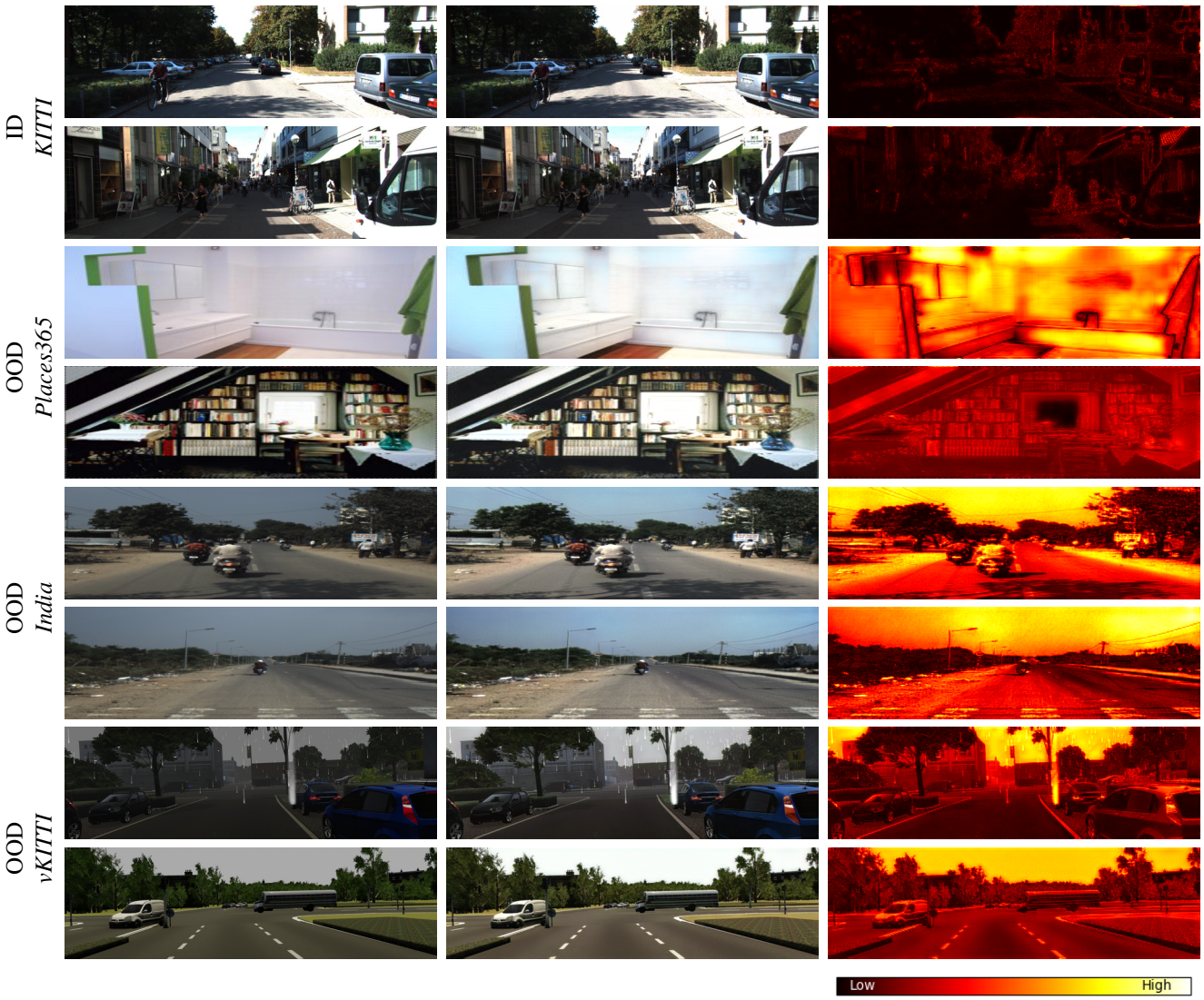

Figure 4: MonoViT [12] is trained on the in-distribution dataset KITTI [4]. We show test images $\mathbf{x}$ from the in-distribution (ID) dataset KITTI and the out-of-distribution (OOD) datasets Places365 [13], India Driving [11] (India) and virtual KITTI [2] (vKITI) on the left. The corresponding reconstructed images $\hat{\mathbf{x}}$ from our image decoder and resulting error maps $\mathbf{e}$ are shown in the middle and right, respectively.

Figure 5: Failure of the image decoder for in-distribution (ID) image from NYU [7] and out-of-distribution (OOD) image from Places365 [13] as input to Monodepth2 [5] trained on NYU [7].

Figure 6: Failure of the image decoder for in-distribution (ID) image from KITTI [4] and out-of-distribution (OOD) image from virtual KITTI (vKITTI) [2] as input to Monodepth2 [5] trained on KITTI [4].

References

[1] Ashutosh Agarwal and Chetan Arora. Attention attention everywhere: Monocular depth prediction with skip attention. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 5861–5870, January 2023. 1, 2

[2] Yohann Cabon, Naila Murray, and Martin Humenberger. Virtual kitti 2. 2020. 2, 4, 5, 6

[3] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. *Proceedings of The 33rd International Conference on Machine Learning*, 06 2015. 1

[4] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *International Journal of Robotics Research (IJRR)*, 2013. 2, 4, 5, 6

[5] Clément Godard, Oisin Mac Aodha, and Gabriel J. Brostow. Digging into self-supervised monocular depth estimation. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3827–3837, 2019. 1, 3, 4, 6

[6] Maria Klodt and Andrea Vedaldi. Supervising the new with the old: Learning sfm from sfm. In *European Conference on Computer Vision*, 2018. 1

[7] Pushmeet Kohli Nathan Silberman, Derek Hoiem and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *European Conference on Computer Vision*, 2012. 1, 2, 3, 6

[8] David A. Nix and Andreas S. Weigend. Estimating the mean and variance of the target probability distribution. *Proceedings of 1994 IEEE International Conference on Neural Networks (ICNN'94)*, 1:55–60 vol.1, 1994. 1

[9] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014. 1

[10] Uddeshya Upadhyay, Shyamgopal Karthik, Yanbei Chen, Massimiliano Mancini, and Zeynep Akata. Bayescap: Bayesian identity cap for calibrated uncertainty in frozen neural networks. In *European Conference on Computer Vision*, 2022. 1

[11] G. Varma, A. Subramanian, Anoop M. Namboodiri, Manmohan Chandraker, and C. V. Jawahar. Idd: A dataset for exploring problems of autonomous navigation in unconstrained environments. *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1743–1751, 2018. 2, 4, 5

[12] Chaoqiang Zhao, Youmin Zhang, Matteo Poggi, Fabio Tosi, Xianda Guo, Zheng Zhu, Guan Huang, Yang Tang, and Stefano Mattoccia. Monovit: Self-supervised monocular depth estimation with a vision transformer. In *International Conference on 3D Vision*, 2022. 1, 2, 5

[13] Bolei Zhou, Àgata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40:1452–1464, 2018. 1, 2, 3, 4, 5, 6