Boosting Self-Supervision for Single-View Scene Completion via Knowledge Distillation

Supplementary Material

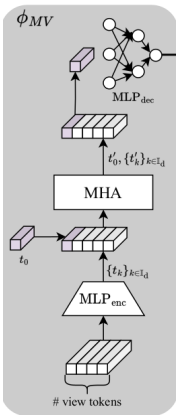


Figure 7. **Attention Layer.** We additionally investigated Multi-Head-Attention (MHA) layers to fuse geometry information from multiple images. As another method of aggregating multi-views, the Multi-Head self-At, replacing the Φ_{MV} in Fig. 2. Fig. 10 shows qualitative results with using the attention layers for multi-view aggregation.

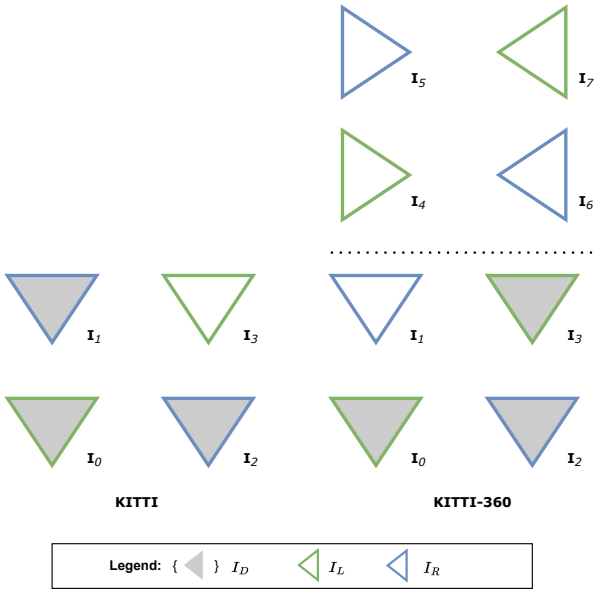


Figure 8. **Frame Arrangement in temporal steps.** The possible frames are split into either loss set I_L or render loss set I_R. Note that the first frame starting with zero index is used for an input frame as fixed. Both KITTI-360 and KITTI use stereo cameras. Depending on the experiments' setup, having fisheyes as input scenes is optional. This influence has been discussed in the ablation study of the main paper.

7. Additional Results

In the following, we present additional results of our methods and further comparisons to existing methods, such as PixelNeRF [56] and BTS. Sec. 7.1 discusses the results of the attention-based model of the ablation study more extensively. Sec. 7.2 gives more results for our ablation study, detailing the influence of the different inference setups on the performance of the multi-view model. Sec. 7.3 presents more qualitative examples of the occupancy profiles to show both the benefits of our training setup and the influence of additional frames. Sec. 7.4 gives additional qualitative results for the depth prediction task on KITTI [11].

7.1. Using attention layers

In Fig. 10, we show prediction examples coming from the attention model instead of softmax view-aggregation in Fig. 2. (See Fig. 7) While the single model seems to produce reasonable depth and occupancy prediction, adding more views leads to noisy depth predictions that get worse with each additional view. A closer inspection of the occupancy profiles shows that for all inference setups, the attention model casts thin *occupancy shadows* in the scene, which seems to degrade the quality of the depth prediction and the occupancy evaluation.

7.2. Occupancy Estimation

Fig. 9 shows the occupancy accuracy for Knowledge-Distillation Behind the Scenes (KDBTS) and BTS for different depth values. The performance increase of KDBTS mainly happens at depth values larger than 10 meters.

Tab. 5 to Tab. 9 gives the complete overview of the settings tested in the ablation study of the main paper for five different camera settings at inference time. The different setups used in these tables are illustrated in Fig. 11. It shows the general improvement of all models tested in the ablation study when providing more frames at inference time, except for the model with attention layers. Our model performs best in all settings with a few minor exceptions. We additionally show the influence of including fisheye cameras at inference time in Tab. 9.

7.3. Occupancy profiles

We present additional visualizations of our models and the baselines BTS [52] and PixelNeRF [56] in Fig. 13 and

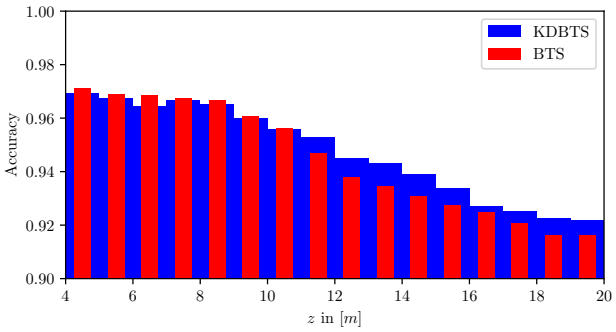


Figure 9. **Occupancy Accuracy At Different Depths.** KDBTS shows on par performance for occupancy estimation to BTS for depth values (z) smaller than 10 m. For depths larger than 10 m, KDBTS starts to outperform BTS significantly.

Fig. 14 in the monocular occupancy prediction setting. The examples show overall improvements in our scene reconstruction. Our models produce cleaner edges and show a better holistic scene understanding by reconstructing house facades as straight lines, removing some of the bulges in BTS. Apart from the overall improvements of our approach compared to BTS, they also demonstrate some of the limitations of the static scene assumption taken in our model. Dynamic objects in the scene can lead to conflicting information in the scene reconstruction of our MVBTS model, which also affects KDBTS. This results in either drawn-out shadows of dynamic objects (see the first example of Fig. 13) or our models removing the dynamic object entirely (see the first example of Fig. 14).

7.4. Depth Prediction

We evaluate the depth predictions generated by our models compared to established baselines such as BTS [52] and PixelNeRF [56], specifically focusing on monocular input.

Figures 13 and 14 visually present the depth predictions obtained from our models and the aforementioned baselines. Our models demonstrate performance on par with BTS in terms of overall accuracy. However, a notable difference lies in the prediction of car windows, where our models tend to exhibit fewer holes compared to BTS. Conversely, our models may show slightly increased blur at the edges of reconstructed objects.

To further analyze the performance differences, we examine the error distribution depicted in Fig. 12. While the majority of errors appear similar between our model (KDBTS) and BTS, KDBTS tends to exhibit more large errors, indicating the violation of static scene assumption.

In Fig. 15, we provide a qualitative comparison of the depth predictions generated by KDBTS and BTS. Although both methods perform comparably overall, KDBTS shows a tendency to perform worse, particularly in scenarios involving dynamic objects. This discrepancy is attributed to violations of the static scene assumption, where the prediction of moving cars may vanish due to conflicting information from multiple time steps (as illustrated in Fig. 14). These inconsistencies in temporal information impact the reconstruction quality, affecting both depth and occupancy estimates.

8. Implementation Details

In the following, we detail the implementation details of our method, including the network architecture and training hyperparameters. For additional details, such as more information about the rendering process and the positional encoding, we refer the reader to [52] and its supplementary material.

8.1. Network Architecture

For implementation reasons, our network consists of a backbone encoder-decoder network and two decoder networks for both the single-view and multi-view settings, respectively.

Backbone. For the backbone, we follow Monodepth2 [13] and BTS [52] such that the reported results stem from the different training setups. It is comprised of a ResNet50 network [19] with an adjustable channel size of 64. As with BTS [52], there is no feature reduction in the upconvolutions of the network.

Single-View Head. The single-view decoding network follows [52] exactly and is comprised of a layer-connected network with ReLU activation functions and residual connections. The input dimension of the MLP is 103 (64 feature channel size + 39 positional encoding size) and the network has a hidden dimension of 64.

Multi-View Head. The multi-view decoding network consists of two MLPs with the same architecture as the single-view decoding network. MLP_1, acting as a feature reduction network, has an input dimension of 103, a hidden dimension of 128, and an output size of 17 (1 confidence value + 16 feature channel size). MLP_2 has an input dimension of 16, and a hidden dimension of 16. The softmax layer in between the MLPs uses no temperature scaling.

Ablation Study Network. For our ablation study, we also test networks where the MLP dimensions are set to the following for the large network:

- MLP_1 (input: 103, hidden: 256, output: 33)
- MLP_2 (input: 32, hidden: 32, output: 1)

and to

- MLP_1 (input: 103, hidden: 128, output: 2)
- no MLP_2

for the small model. The small network does not do feature fusion but rather directly fuses the different frames' density prediction.

8.2. Training Configuration

Hyperparameters For training on both KITTI [11] and KITTI-360 [32], we use the same set of hyperparameters.

We use a batch size of 8 during training. We use the patched-based sampling strategy of [52] and sample 32 random patches of size 8×8, giving us 2048 rays in total per batch. The loss weights are set to $\lambda_{SSIM} = 0.85$, $\lambda_{\text{L1}} = 0.15$ following [13, 52] and $\lambda_{\text{EAS}} = 10^{-3}$ following the code implementation of [52]. We use an ADAM optimizer with a learning rate of $\lambda = 10^{-4}$ for the first 120,000 steps and $\lambda = 10^{-5}$ for the rest. We apply the same color augmentation and random flips as [52]. For our knowledge distillation, we train with a constant learning rate of $\lambda = 10^{-4}$.

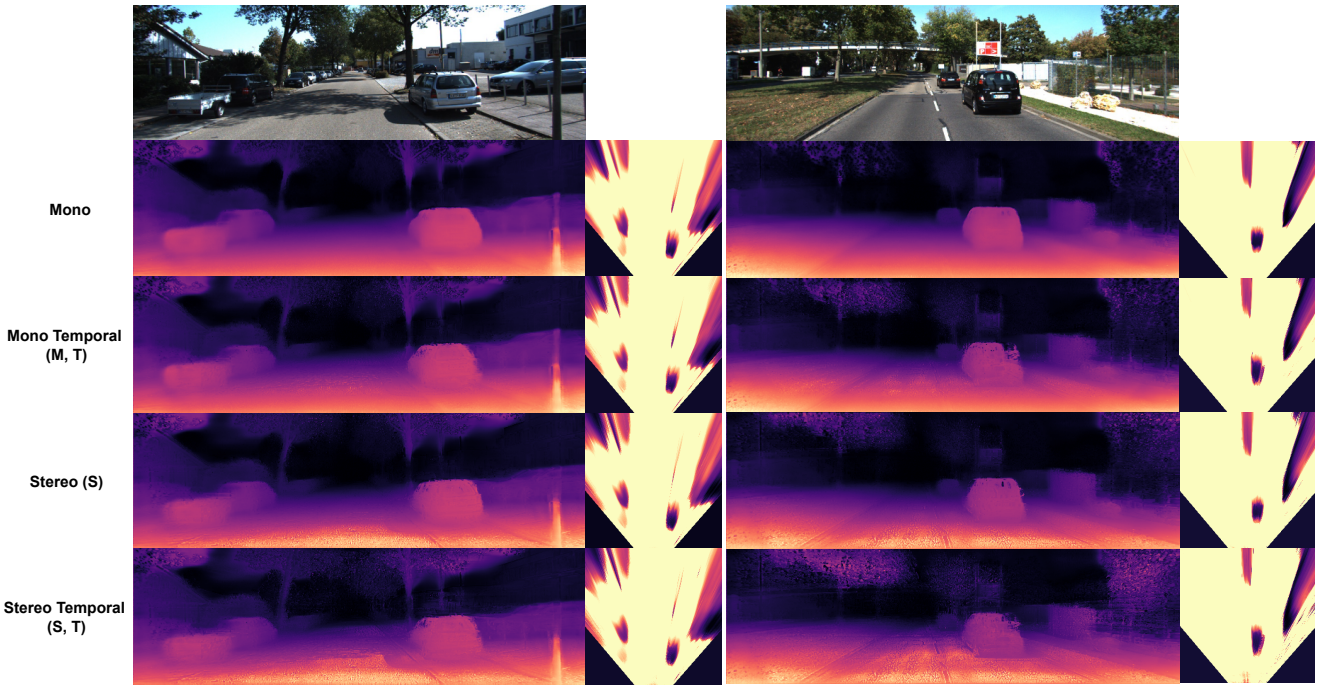


Figure 10. **Attention Layer Qualitative Results.** Comparison in both depth- and occupancy estimation. The camera frustum is set up in x=[-9, 9]m, y=[0, 0.75]m, and z=[3,21]m. The monocular occupancy prediction produces reasonable results for both the expected ray termination depth and the occupancy profiles. Adding more frames to the prediction leads to increased noise in the predictions. A closer inspection of the occupancy profiles shows that the attention model produces long and thin *occupancy shadows* along rays cast from the camera. The occupancy predictions seem to be quite sensitive to changes in the features coming from the pixel-aligned feature map.

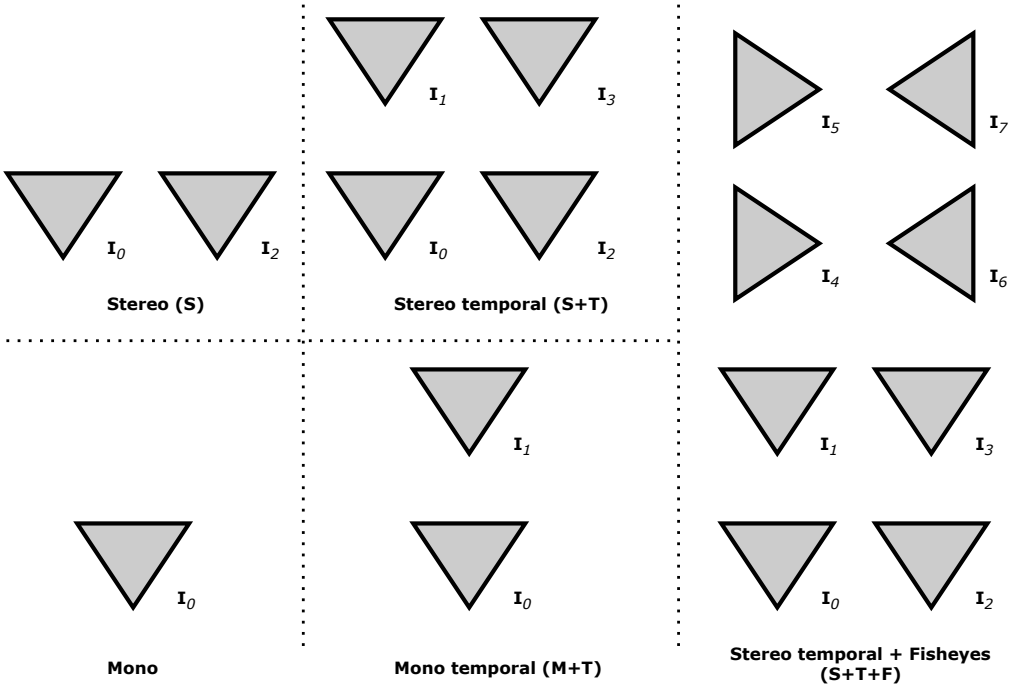


Figure 11. **Frame Arrangement in Inference.** Illustration of the settings used at inference time in Tab. 5, Tab. 6, Tab. 7, Tab. 8, and Tab. 9.

Inference	dropout	MLPs	attn. layers	encode fisheye	O_{acc} ↑	O_{prec} ↑	O_{rec} ↑	IE_{acc} ↑	IE_{prec} ↑	IE_{rec} ↑
(*Mono*)	0.5	middle	✓	✗	93.98%	57.41%	80.47%	74.43%	48.40%	44.47%
(*Mono*)	0.0	middle	✗	✓	93.81%	57.07%	82.60%	73.03%	50.55%	39.54%
(*Mono*)	0.2	middle	✗	✓	94.37%	59.03%	80.91%	76.24%	51.06%	46.28%
(*Mono*)	0.5	middle	✗	✓	94.71%	60.31%	83.35%	77.89%	52.96%	46.02%
(*Mono*)	0.8	middle	✗	✓	94.78%	60.09%	85.38%	77.76%	53.25%	43.43%
(*Mono*)	0.5	small	✗	✗	94.51%	59.62%	85.18%	76.88%	52.80%	40.80%
(*Mono*)	0.5	large	✗	✗	94.16%	58.56%	80.96%	76.04%	50.78%	46.69%
(*Mono*)	0.5	middle	✗	✗	94.76%	60.83%	84.51%	78.00%	53.69%	44.04%

Table 5. **Ablation Studies in Monocular Inference.** Evaluation of all models in the ablations study in the monocular setting (see Fig. 11 for more details of the frame arrangements). As for the setting in the main paper, our final architecture performs the best with a few exceptions. The model with more dropout performs slightly better in the occupancy estimation task. This is likely due to having fewer frames available during training. Additionally, some of the methods that also encoded fisheye cameras during training as well perform better when it comes to the IE_{rec} but worse at IE_{prec}. They produce fewer false negatives, but more false positives, meaning less space is predicted as being empty.

Inference	dropout	MLPs	attn. layers	encode fisheye	O_{acc} ↑	O_{prec} ↑	O_{rec} ↑	IE_{acc} ↑	IE_{prec} ↑	IE_{rec} ↑
(*M+T*)	0.5	middle	✓	✗	93.79%	58.97%	77.44%	72.93%	45.81%	47.91%
(*M+T*)	0.0	middle	✗	✓	93.69%	57.36%	81.59%	73.50%	45.66%	41.07%
(*M+T*)	0.2	middle	✗	✓	94.42%	59.45%	81.76%	76.45%	51.44%	45.94%
(*M+T*)	0.5	middle	✗	✓	94.78%	60.69%	84.37%	77.93%	55.32%	45.81%
(*M+T*)	0.8	middle	✗	✓	94.90%	60.42%	86.04%	79.21%	53.55%	43.73%
(*M+T*)	0.5	small	✗	✗	94.56%	59.95%	85.98%	78.37%	55.30%	42.65%
(*M+T*)	0.5	large	✗	✗	94.30%	59.21%	81.28%	76.65%	50.74%	47.85%
(*M+T*)	0.5	middle	✗	✗	94.82%	61.02%	84.83%	78.73%	53.88%	44.81%

Table 6. **Ablation Studies in Temporal Monocular Inference.** Using one additional temporal frame shows slight improvements for all methods, with the exception of the attention layer model (see Fig. 11 for more details of the frame setup). Otherwise, the difference between the models is similar to the monocular setting. The model with a higher dropout gains additional performance improvements to our final model.

Inference	dropout	MLPs	attn. layers	encode fisheye	O_{acc} ↑	O_{prec} ↑	O_{rec} ↑	IE_{acc} ↑	IE_{prec} ↑	IE_{rec} ↑
(*S*)	0.5	middle	✓	✗	93.92%	59.26%	78.57%	73.34%	45.11%	47.37%
(*S*)	0.0	middle	✗	✓	93.41%	57.21%	80.23%	72.40%	46.38%	42.14%
(*S*)	0.2	middle	✗	✓	94.53%	59.02%	82.04%	76.65%	53.67%	45.97%
(*S*)	0.5	middle	✗	✓	94.87%	60.09%	84.58%	78.37%	54.75%	45.71%
(*S*)	0.8	middle	✗	✓	94.89%	60.07%	85.64%	77.98%	54.12%	43.22%
(*S*)	0.5	small	✗	✗	94.56%	59.34%	86.20%	77.92%	54.36%	42.44%
(*S*)	0.5	large	✗	✗	94.29%	58.86%	81.83%	76.15%	50.90%	46.15%
(*S*)	0.5	middle	✗	✗	94.84%	61.12%	85.28%	78.62%	54.02%	44.05%

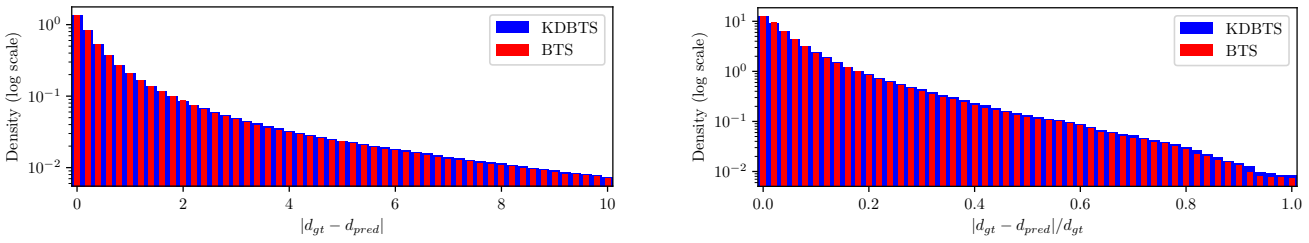
Table 7. **Ablation Studies in Stereo Inference.** Using stereo frame also shows slight improvements for all methods, with the exception of the attention layer model (see Fig. 11 for more details of the frame setup).

Inference	dropout	MLPs	attn. layers	encode fisheye	O_{acc} ↑	O_{prec} ↑	O_{rec} ↑	IE_{acc} ↑	IE_{prec} ↑	IE_{rec} ↑
$(S+T)$	0.5	middle	✓	✗	93.23%	60.77%	70.69%	69.83%	47.40%	58.94%
$(S+T)$	0.0	middle	✗	✓	93.44%	56.85%	79.74%	73.02%	45.54%	43.90%
$(S+T)$	0.2	middle	✗	✓	94.57%	59.85%	83.20%	77.13%	54.41%	45.64%
$(S+T)$	0.5	middle	✗	✓	94.93%	60.72%	85.47%	79.05%	55.73%	46.39%
$(S+T)$	0.8	middle	✗	✓	94.94%	60.43%	86.40%	79.41%	55.35%	44.80%
$(S+T)$	0.5	small	✗	✗	94.64%	59.64%	86.65%	78.54%	55.04%	43.70%
$(S+T)$	0.5	large	✗	✗	94.38%	59.88%	82.51%	77.24%	51.66%	46.90%
$(S+T)$	0.5	middle	✗	✗	94.91%	61.73%	85.78%	79.47%	55.08%	45.23%

Table 8. **Ablation Studies in Temporal Stereo Inference.** For convenience, we repeat the findings of the main paper here (see Fig. 11 for more details of the frame setup).

Inference	dropout	MLPs	attn. layers	encode fisheye	O_{acc} ↑	O_{prec} ↑	O_{rec} ↑	IE_{acc} ↑	IE_{prec} ↑	IE_{rec} ↑
$(S{+}T{+}F)$	0.5	middle	✓	✗	92.88%	58.51%	71.39%	69.67%	47.31%	57.28%
$(S{+}T{+}F)$	0.0	middle	✗	✓	93.22%	55.79%	78.58%	72.36%	44.49%	44.09%
$(S{+}T{+}F)$	0.2	middle	✗	✓	94.54%	58.77%	84.05%	77.19%	55.16%	44.00%
$(S{+}T{+}F)$	0.5	middle	✗	✓	94.90%	60.34%	85.31%	78.84%	56.08%	46.04%
$(S{+}T{+}F)$	0.8	middle	✗	✓	94.91%	60.23%	86.22%	79.31%	55.22%	44.50%
$(S{+}T{+}F)$	0.5	small	✗	✗	94.67%	59.17%	86.65%	78.54%	54.97%	43.45%
$(S{+}T{+}F)$	0.5	large	✗	✗	94.38%	59.22%	82.68%	77.08%	51.53%	45.90%
$(S{+}T{+}F)$	0.5	middle	✗	✗	94.89%	60.38%	85.83%	79.44%	55.30%	44.90%

Table 9. **Ablation Studies in Temporal Stereo Fisheye Inference.** Using the fisheye cameras for inference does not give large improvements for all methods. This shows that a lot of the scene information can already be captured in the pinhole frames alone (see Fig. 11 for more details of the frame setup).



(a) absolute error distribution.

(b) relative error distribution.

Figure 12. **Depth Prediction Error Distributions.** The error distributions are similar for both methods - KDBTS has slightly more large errors than BTS. Lower errors for KDBTS (blue dots) and BTS (red dots), intensity encodes magnitude. Qualitative examples (see Fig. 15) for the depth error on the KITTI test set show that KDTBS often performs worse on dynamic objects (e.g. cars, cyclists).

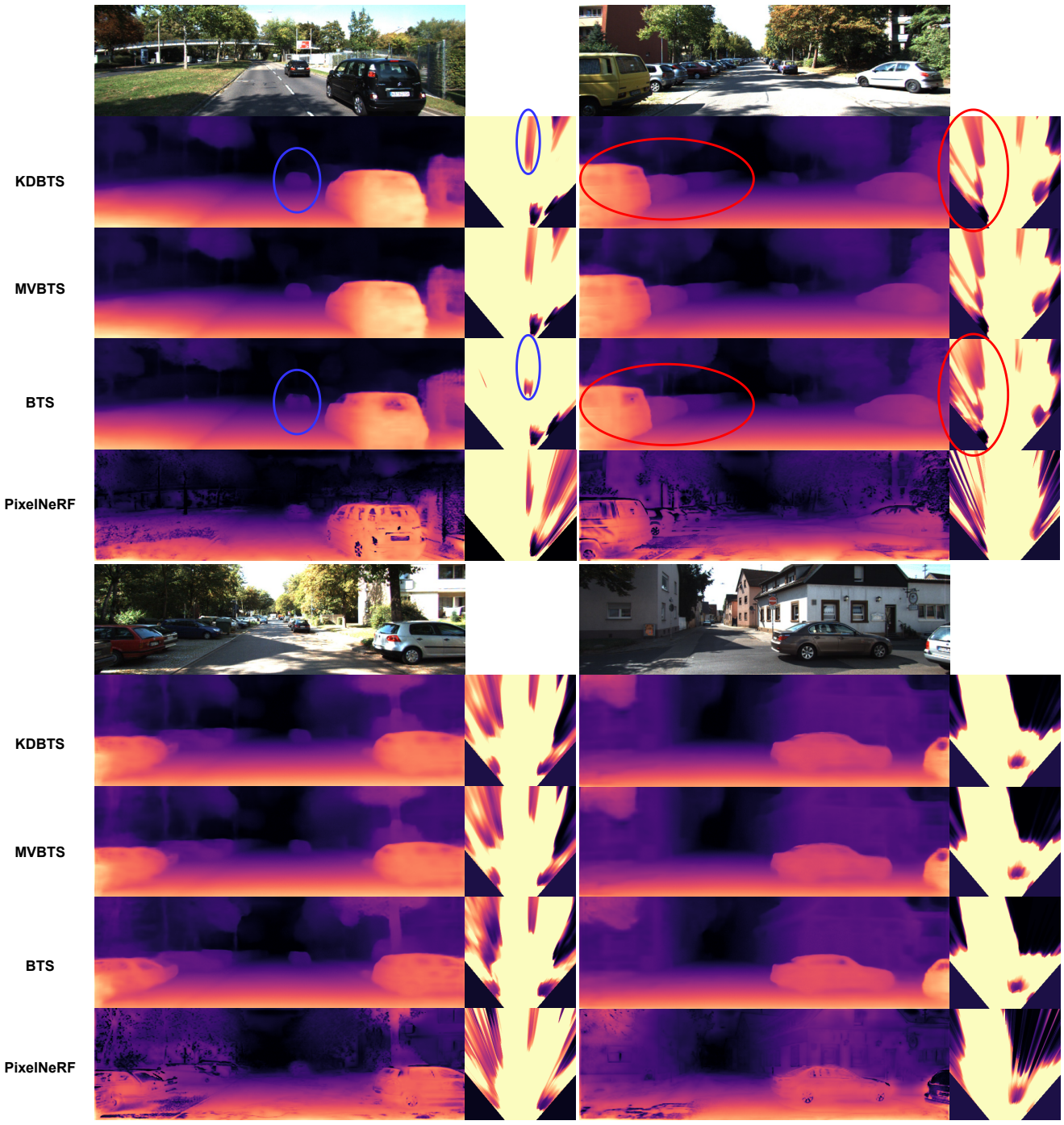


Figure 13. **Qualitative Baseline Results 1.** Baselines comparison in both depth- and occupancy estimation. The camera frustum is set up in x=[-9, 9]m, y=[0, 0.75]m, and z=[3,21]m. It shows general improvements by our methods, such as removing occupancy behind parked cars, leading to cleaner occupancy predictions (see top right example). The top left shows a failure case of our method where a moving car produces a drawn-out shadow for our methods, likely resulting from conflicting temporal information.

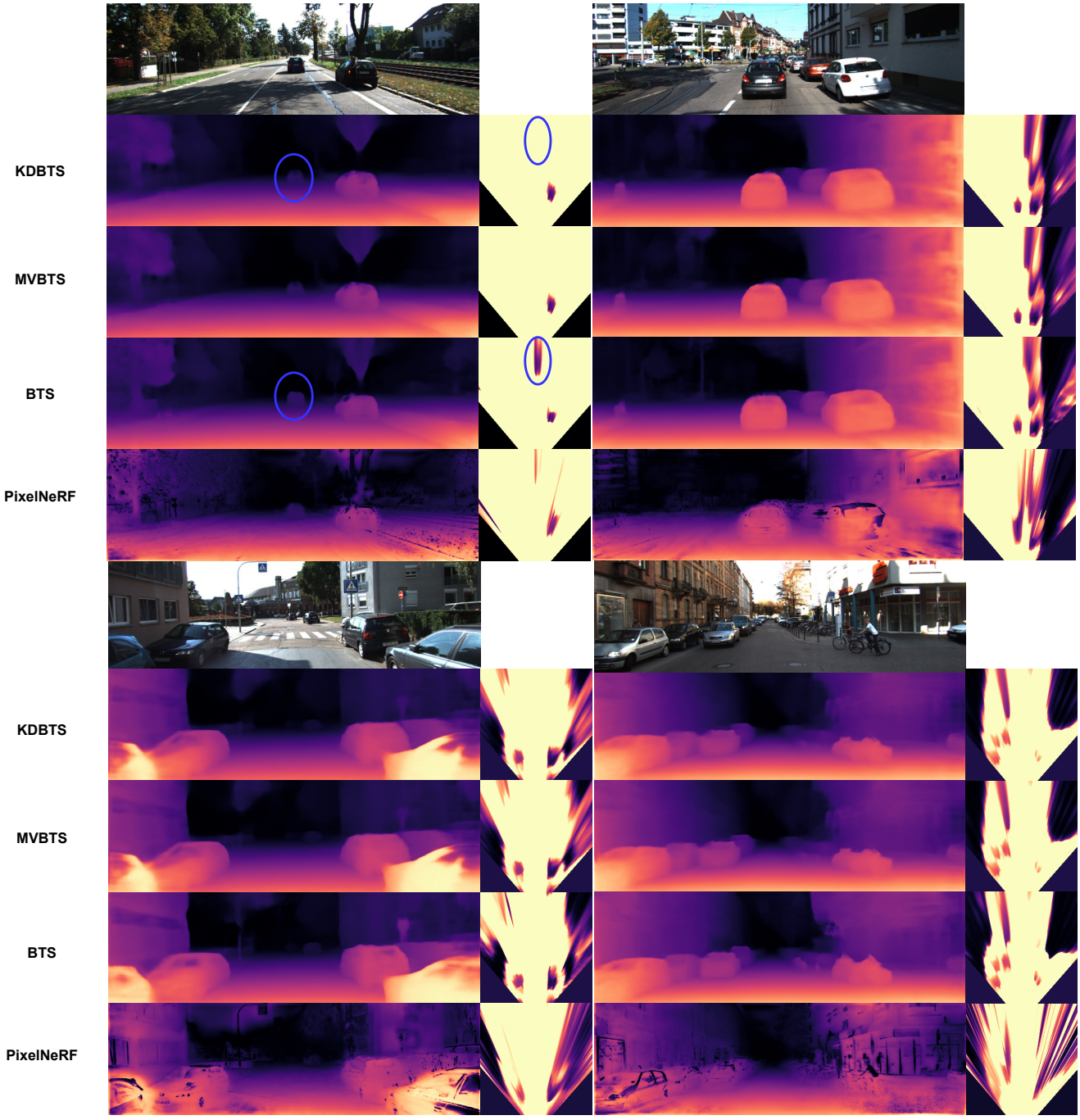


Figure 14. **Qualitative Baseline Results 2.** Baselines comparison in both depth- and occupancy estimation. The camera frustum is set up in x=[-9, 9]m, y=[0, 0.75]m, and z=[3,21]m. Our method shows general improvements, such as removing holes in car windows (see lower examples) or predicting the house facades to be in a straight line (see lower right example). It also shows a failure case (top left) where our methods remove a moving car from the scene, likely due to conflicting temporal information.

Figure 15. **Depth error comparison between BTS and KDBTS.** KDBTS exhibits slightly more large errors compared to BTS. Qualitative examples (bottom) demonstrate depth error on the KITTI test set, with lower errors depicted by KDBTS (blue dots) and BTS (red dots), with intensity representing magnitude. Each test image presents the projected scene from the LiDAR ground truth point cloud. The projected LiDAR point cloud is used to calculate the distance error between the prediction and its ground truth. Color differentiation indicates lesser distance errors between KDBTS (blue dots) and BTS (red dots). In KDBTS, the model's reconstruction is affected by moving objects, resulting in larger errors typically observed on dynamic objects (e.g., cars, cyclists). Consequently, red dots are depicted for dynamic objects, signifying lower errors for BTS.