# Camera Pose Matters: Improving Depth Prediction by Mitigating Pose Distribution Bias
## –*Supplementary Material*–

Yunhan Zhao[1] Shu Kong[2] Charless Fowlkes[1]
[1]UC Irvine [2]Carnegie Mellon University
{yunhaz5, fowlkes}@ics.uci.edu shuk@andrew.cmu.edu

## Outline

*In the supplementary document, we provide additional experimental results to further support our findings, as well as details of our experiments and more visualizations. Below is the outline.*

- **Section 1: Handling infinite values in CPP encoding.** In the main paper, we apply the inverse tangent operation to deal with infinite values in the CPP encoded maps. We study an alternative based on a simple clipping operation.
- **Section 2: Hyperparameter analysis in CPP encoding.** Our CPP encoding method has a hyperparameter $C$ which is the distance between the upper and lower planes (i.e., a ceiling and ground plane). We study how $C$ affects depth prediction performance.
- **Section 3: Quantitative results of blind predictions.** We show the quantitative results of blind prediction to better understand how CPP helps capture prior knowledge for depth prediction.
- **Section 4: Further study of PDA augmentation scales.** We provide a thorough study between the depth predictor performance and PDA augmentation scales.
- **Section 5: Further study of camera height and rotation in CPP encoding.** We compare the performance of CPP on InteriorNet (with nearly fixed roll) by either encoding the true pitch or camera height while fixing the other one.
- **Section 6: CPP Encoding with Predicted Poses.** Considering the scenario where test time camera poses are not always available, we show experimental results of CPP encoding with predicted poses during evaluations.
- **Section 7: Additional details in experiments.** We present more experimental details such as RGB and depth preprocessing steps, training the camera pose prediction models, our evaluation protocol, and the ScanNet camera pose distribution.
- **More Visual Results.** We include more depth prediction visualizations of different methods in Fig. 7 and 8.

Table 1: Comparisons of different encoding methods evaluated on InteriorNet test-sets. CPP applies an inverse tangent transform $\tan^{-1}$ in encoding the camera poses. In contrast, CPP-Clip replaces the $\tan^{-1}$ function with a clipping operation while keeping every other step the same as CPP encoding. Both CPP and CPP-Clip perform better than Vanilla model, demonstrating the effectiveness of our CPP method. Clearly, using the inverse tangent operator is better than clipping.

| Models | *Natural-Test-Set* | | *Uniform-Test-Set* | |
|---|---|---|---|---|
| | ↓ better Abs$^r$/Sq$^r$/RMS-log | ↑ better $\delta^1$ / $\delta^2$ | ↓ better Abs$^r$/Sq$^r$/RMS-log | ↑ better $\delta^1$ / $\delta^2$ |
| InteriorNet | | | | |
| Vanilla | .154 / .148 / .229 | .803 / .945 | .183 / .146 / .250 | .724 / .926 |
| + CPP-Clip | .109 / .124 / .204 | .871 / .956 | .111 / .096 / .189 | .867 / .959 |
| + CPP | **.108 / .120 / .199** | **.872 / .958** | **.106 / .088 / .183** | **.876 / .961** |
| ScanNet | | | | |
| Vanilla | .125 / .068 / .186 | .837 / .962 | .177 / .121 / .265 | .711 / .928 |
| + CPP-Clip | .110 / .064 / .179 | .869 / .964 | .157 / .108 / .248 | .758 / .936 |
| + CPP | **.108 / .060 / .171** | **.871 / .965** | **.154 / .106 / .239** | **.781 / .943** |

## 1. Handling Infinite Values in CPP Encoding

At the last step in computing CPP encoded maps $\mathbf{M}_{CPP}$, we apply the inverse tangent operator to eliminate the infinity values (happens when the ray shooting from camera is parallel to the ground plane) and maps the values of $\mathbf{M}$ (i.e. $\mathbf{M}_{CPP} = \tan^{-1}(\mathbf{M})$) to the range $[\tan^{-1}(\min\{h, C-h\}), \frac{\pi}{2}]$. However, the inverse tangent operator is not the only choice. We provide an ablation study that replaces the $\tan^{-1}(\cdot)$ with a clipping operation.

Specifically, we set a threshold $\tau$ that represents the prior knowledge of the distance from camera to the furthest point in the scene. Mathematically, for each point $[u, v]$ in the new CPP clipping encoded map $\mathbf{M}^{Clip}_{CPP} \in \mathbb{R}^{\mathbb{H}\times\mathbb{W}}$, we compute the pseudo depth value:

$$\mathbf{M}^{Clip}_{CPP}[u,v] = \begin{cases} M[u,v] & M[u,v] < \tau \\ \tau & \text{otherwise} \end{cases}$$

We set $\tau = 20.0$ in this experiment. After clipping, we linearly rescale the encoded map to the range of [-1.0, 1.0]

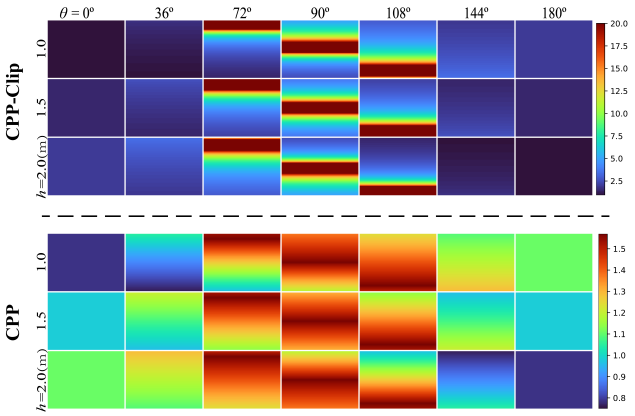


Figure 1: Visual comparisons of encoded maps of CPP and CPP-Clip with different pitch $\theta$ and camera height $h$. We set the threshold $\tau$=20 for CPP-Clip. Encoded maps computed by CPP-Clip have the "red stripe" when the pitch is around 90° while CPP encoded maps have more smooth transitions when capturing the horizon.

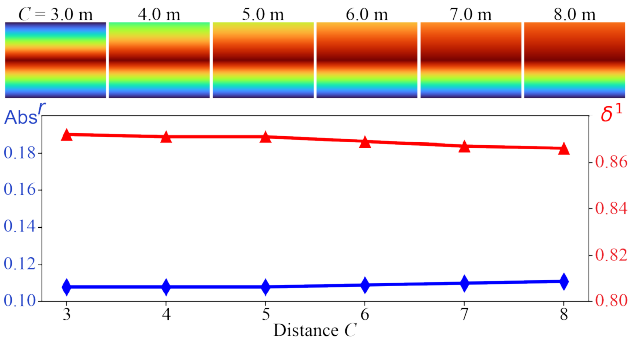


Figure 2: **Uppeer:** visualizations of CPP encoding with different hyper-parameter $C$ (top); **bottom:** depth prediction performance as a function of hyper-parameter $C$. We train depth predictors on InteriorNet *Natural* train-set and test on its *Natural* test-set. From visual inspection, changing the parameter $C$ only affects the part of CPP encoded maps where pixels are above the horizon. As shown by the performance curve, our proposed CPP encoding is very robust w.r.t different values of $C$.

to match the statistics of RGB images. We find this yields better performance than directly using $\mathbf{M}_{CPP}^{Clip}$. We visually compare some encoded maps in Fig. 1, where we see the clipping method introduces artificial stripes. Probably due to this, CPP-Clip does not perform as well as CPP that adopts inverse tangent transform, as shown in Table 1.

## 2. Hyperparameter Analysis in CPP Encoding

CPP encoding assumes that the camera moves in an empty indoor scene with an infinite floor and ceiling and the distance between two planes in the up direction is described by the parameter $C$. This distance is set to $C = 3$ meter in all experiments in the main paper. To verify the performance change w.r.t the distance $C$, we conduct experiments on InteriorNet *Natural* train-set with various distance

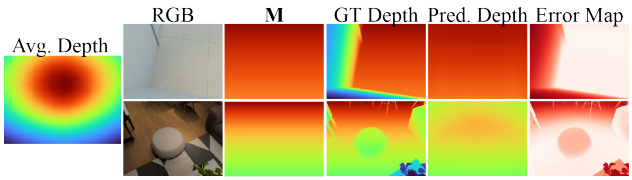


Figure 3: Illustration of how camera pose provides a strong depth prior through "blind depth prediction". Specifically, over the InteriorNet *Natural* train-set, we train a depth predictor solely on the CPP encoded maps $\mathbf{M}$ *without* RGB as input. For visual comparison, we compute an averaged depth map (shown left). We visualize depth predictions on two random examples. All the depth maps are visualized with the same colormap range. Perhaps not surprisingly, $\mathbf{M}$ presents nearly the true depth in floor areas, suggesting that camera pose alone does provide strong prior depth information for these scenes.

Table 2: Comparison between using the average depth map (Avg) computed on the InteriorNet *Natural* train-set and the "blind predictor", which estimates depth solely from per-image CPP encoded maps *without* RGB images. We report results on InteriorNet *Natural* test-set. We find that "blind predictor" performs better than "avg depth map", implying the benefit of exploiting camera poses. We also report on two specific images on which "blind predictor" performs well compared to the average performance of Avg or Blind, as shown in Fig. 3. This further confirms that camera poses contain useful prior knowledge about scene depth.

| Models | ↓ better Abs-Rel/Sq-Rel/RMS-log | ↑ better $\delta^1$ / $\delta^2$ |
|---|---|---|
| Avg | .414 / .641 / .466 | .346 / .638 |
| Blind | **.342 / .519 / .395** | **.485 / .750** |
| Img-1 | .041 / .010 / .082 | .946 / .999 |
| Img-2 | .115 / .059 / .176 | .793 / .990 |

$C = [4, 5, 6, 7, 8]$. As shown in Fig. 2, the performance of depth predictors are very robust in terms of the parameter $C$. In other words, CPP encoding improves the depth predictor performance and reduces the distribution bias consistently, regardless of the parameter $C$.

## 3. Quantitative Results of Blind Predictions

In the main paper, we visually demonstrate that camera poses indeed contain prior knowledge of scene depth. The quantitative results of those visual examples are shown in Table 2, from which we find two key insights. First, the blind predictor achieves better performance than evaluating with average training depth maps, suggesting that camera poses alone contain the prior information about scene depth. In other words, training depth predictors with the camera pose alone are better than "random guess" from average training depth maps. Second, the blind predictor achieves promising performance on two images shown in Fig. 3 quantitatively. Together with the visualization of the prediction, we find that blind predictors make significantly more accurate depth prediction on floor regions, which con-

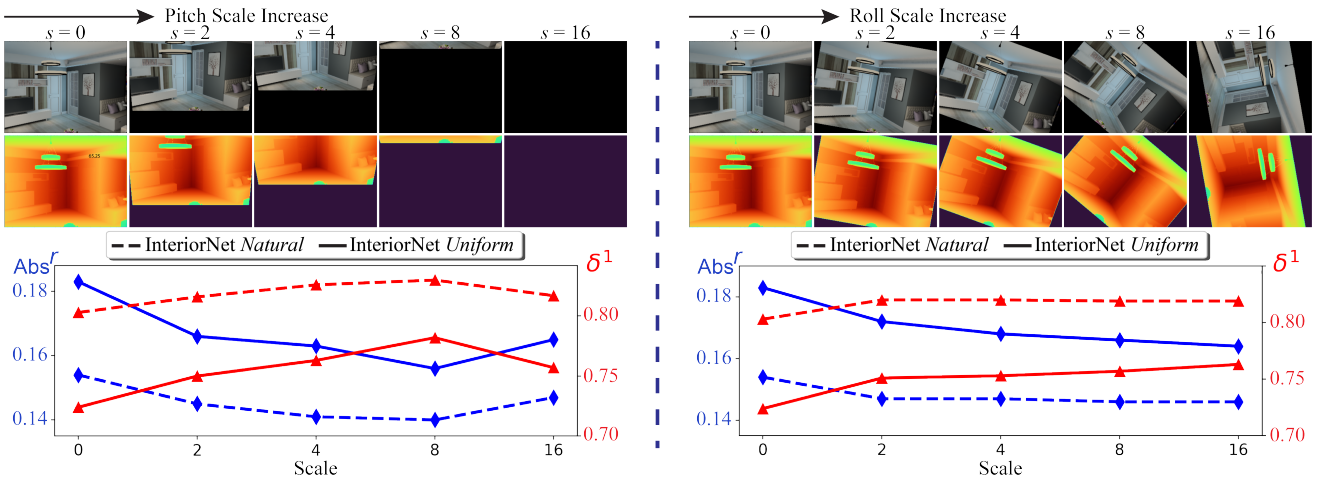


Figure 4: **Upper row:** visualizations of augmented examples using PDA with different scales. **Bottom row:** performance curves of depth predictor trained with PDA with different scales of pitch $\theta$ (left) and roll $\omega$ (right), respectively. Please refer to the main paper (Figure 9) for detailed descriptions. All models are trained on InteriorNet *Natural* train-set and evaluated on both *Natural* (dotted line) and *Uniform* (solid line) test-sets. As we increase the augmentation scale in pitch, the performance of depth predictors improves until scale $s$=16, when large void regions are introduced in the generated examples. On the other hand, increasing augmentation scales in roll lead to steady performance increments. In general, PDA consistently improves depth prediction over a Vanilla model trained without PDA.

firms that the camera pose carries the prior knowledge about scene depth, especially on floor and ceiling regions.

## 4. Further Study of PDA Augmentation Scales

We provide a more detailed analysis of Vanilla depth predictor plus PDA with different scales of pitch and roll, individually. Please refer to the main paper (Figure 9) for detailed descriptions. As shown in Fig. 4, the performance of the depth predictor monotonically increase until augmenting pitch to the scale of 16. From the visual demonstrations, we believe that the performance drop is due to the introduction of large void regions. On the other hand, by rotating roll, we observe steady performance improvement, which demonstrates that PDA boosts the performance of depth predictors by generating training examples with diverse camera poses.

## 5. Further Study of Camera Height and Rotation in CPP Encoding

CPP encodes rotation (roll and pitch) and camera height, however, it is still worth exploring which DOF is more important in CPP encoding. While it is nontrivial to define "importance" as pitch/roll and height have different units and ranges, we did study the pitch and height on InteriorNet (which has a nearly fixed roll). To apply CPP, we fixed either pitch or height and only encode the other with the true value. As shown in Fig. 5, we find that encoding the true camera height (top plot) performs better than the true pitch (bottom plot), and both perform better than the vanilla model. This implies that camera height is "more important" than pitch (probably roll as well).

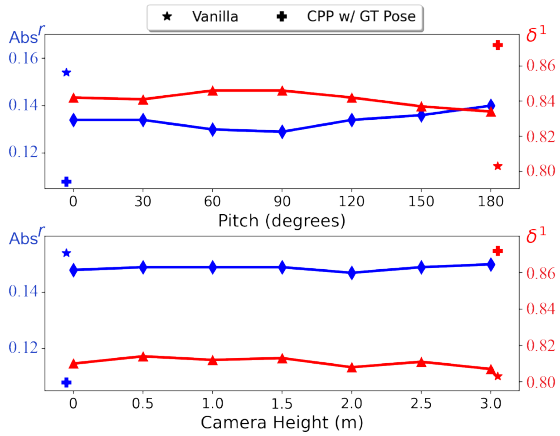


Figure 5: **Top:** CPP encoding with ground-truth camera height and fixed pitch. **Bottom:** CPP encoding with ground-truth pitch and fixed camera height. All models are trained/evaluated on InteriorNet *Natural* train/test-set. Comparing two blue or red curves across two plots, we find that encoding ground-truth height achieves better performance, suggesting height is "more important" than pitch. Moreover, encoding either ground-truth height or pitch outperforms Vanilla model.

## 6. CPP Encoding with Predicted Poses

We study CPP to encode predicted poses. Specifically, we train depth predictors with CPP using true poses on *Natural* train-sets of the two datasets (Table 3). We test models on *Natural* and *Uniform* test-sets, respectively. Note that in testing we encode the predicted poses given by a pose predictor. As shown in Table 3, CPP with predicted poses still outperforms Vanilla model; when jointly trained with PDA, CPP with predicted poses performs even better.

Table 3: **CPP Encoding with Predicted Poses**. We train depth predictors with CPP using true poses on *Natural* train-sets of the two datasets. We test models on *Natural* and *Uniform* test-sets, respectively. Note that in testing we encode predicted poses given by a pose predictor. Clearly, CPP with predicted poses still outperforms Vanilla model; when jointly trained with PDA, CPP with predicted poses performs even better. Nevertheless, encoding predicted poses underperforms encoding true poses.

| Models | *Natural-Test-Set* | | *Uniform-Test-Set* | |
|---|---|---|---|---|
| | ↓ better Abs$^r$/Sq$^r$/RMS-log | ↑ better $\delta^1$ / $\delta^2$ | ↓ better Abs$^r$/Sq$^r$/RMS-log | ↑ better $\delta^1$ / $\delta^2$ |
| InteriorNet | | | | |
| Vanilla | .154 / .148 / .229 | .803 / .945 | .183 / .146 / .250 | .724 / .926 |
| + CPP$_{pred}$ | .142 / .132 / .212 | .825 / .951 | .164 / .121 / .228 | .756 / .946 |
| + CPP | .108 / .120 / .199 | .872 / .958 | .106 / .088 / .183 | .876 / .961 |
| + Both$_{pred}$ | .135 / .127 / .205 | .849 / .955 | .148 / .114 / .213 | .780 / .952 |
| + Both | **.095 / .101 / .180** | **.898 / .966** | **.091 / .069 / .161** | **.903 / .973** |
| ScanNet | | | | |
| Vanilla | .125 / .068 / .186 | .837 / .962 | .177 / .121 / .265 | .711 / .928 |
| + CPP$_{pred}$ | .116 / .065 / .180 | .852 / .964 | .169 / .117 / .255 | .731 / .931 |
| + CPP | .108 / .060 / .171 | .871 / .965 | .154 / .106 / .239 | .781 / .943 |
| + Both$_{pred}$ | .111 / .061 / .173 | .866 / .965 | .159 / .111 / .247 | .773 / .938 |
| + Both | **.102 / .052 / .160** | **.882 / .973** | **.143 / .097 / .230** | **.809 / .952** |

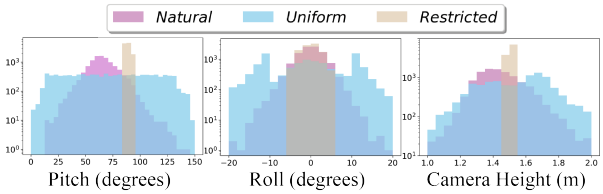


Figure 6: Distribution of pitch, roll and camera height for three subsets of images from ScanNet. From the *Natural* subset, we observe the ScanNet dataset also has a naturally biased distribution in both pitch, roll and camera height. Please refer to Section 5 in the main paper on how we construct these three subsets.

# 7. Additional Details in Experiments

## 7.1. Image and Depth Preprocessing

All input RGB images are first normalized to the range of $[-1.0, 1.0]$ and then resized to $240 \times 320$ before feeding into CNNs. Note that resizing images to $240 \times 320$ does not change their original aspect ratios. For better training, as a preprocessing step on the depth [1, 2], we apply the following operation to rescale depth maps $y$ to get a normalized map $y'$:

$$y' = (\frac{y - E_{min}}{E_{max} - E_{min}} - 0.5) * 2.0, \quad (1)$$

where $E_{min} = 1.0$ and $E_{max} = 10.0$ are the minimum and maximum evaluation values, respective. The above operation is a map from $[1.0, 10.0]$ to $[-1.0, 1.0]$. In the literature, it is reported the model can be trained better in this scale range [4, 3]. We only compute the loss for pixels that have depth values between 1.0 and 10.0 meters. We evaluate the depth prediction on the original depth scale. To do so, we apply an inverse operation of Eq. 1 to the predicted depth maps. Moreover, we also only evaluate the depth that lies in [1, 10] meters.

## 7.2. Pose Prediction Network

When camera poses are not available during testing, we train a camera pose predictor that predicts camera pitch $\theta$, roll $\omega$ and height $h$ for CPP encoding (i.e., the CPP$_{pred}$ model). We build the pose predictor over ResNet18 structure with a new top layer that outputs a 3-dim vector to regress pitch, roll, and camera height. During training, we load the ImageNet pretrained weights and finetune the weights for pose predictions with L1 loss.

## 7.3. Evaluation Protocol

The depth evaluation range in this work is from 1.0m to 10.0m for both InteriorNet and ScanNet. For each method, we save a checkpoint every 10 epochs and select the checkpoint that produces the smallest average L1 loss on the validation set to report the performance.

## 7.4. ScanNet Camera Pose Distribution

The camera pose distribution of subsets in ScanNet is shown in Fig. 6. While it is hard to sample a subset with exactly uniform distribution w.r.t to all attributes (i.e., pitch, roll, and camera height), we sample the *Uniform* subset with the priority of pitch, roll, height from high to low. As these subsets differ a lot in terms of camera pose distribution, they serve our study w.r.t camera distribution bias.

# References


[1] Jiawang Bian, Zhichao Li, Naiyan Wang, Huangying Zhan, Chunhua Shen, Ming-Ming Cheng, and Ian Reid. Unsupervised scale-consistent depth and ego-motion learning from monocular video. In *Advances in Neural Information Processing Systems*, 2019. 4

[2] Clément Godard, Oisin Mac Aodha, and Gabriel J Brostow. Unsupervised monocular depth estimation with left-right consistency. In *CVPR*, 2017. 4

[3] Shanshan Zhao, Huan Fu, Mingming Gong, and Dacheng Tao. Geometry-aware symmetric domain adaptation for monocular depth estimation. In *CVPR*, 2019. 4

[4] Chuanxia Zheng, Tat-Jen Cham, and Jianfei Cai. T2net: Synthetic-to-realistic translation for solving single-image depth estimation tasks. In *ECCV*, 2018. 4

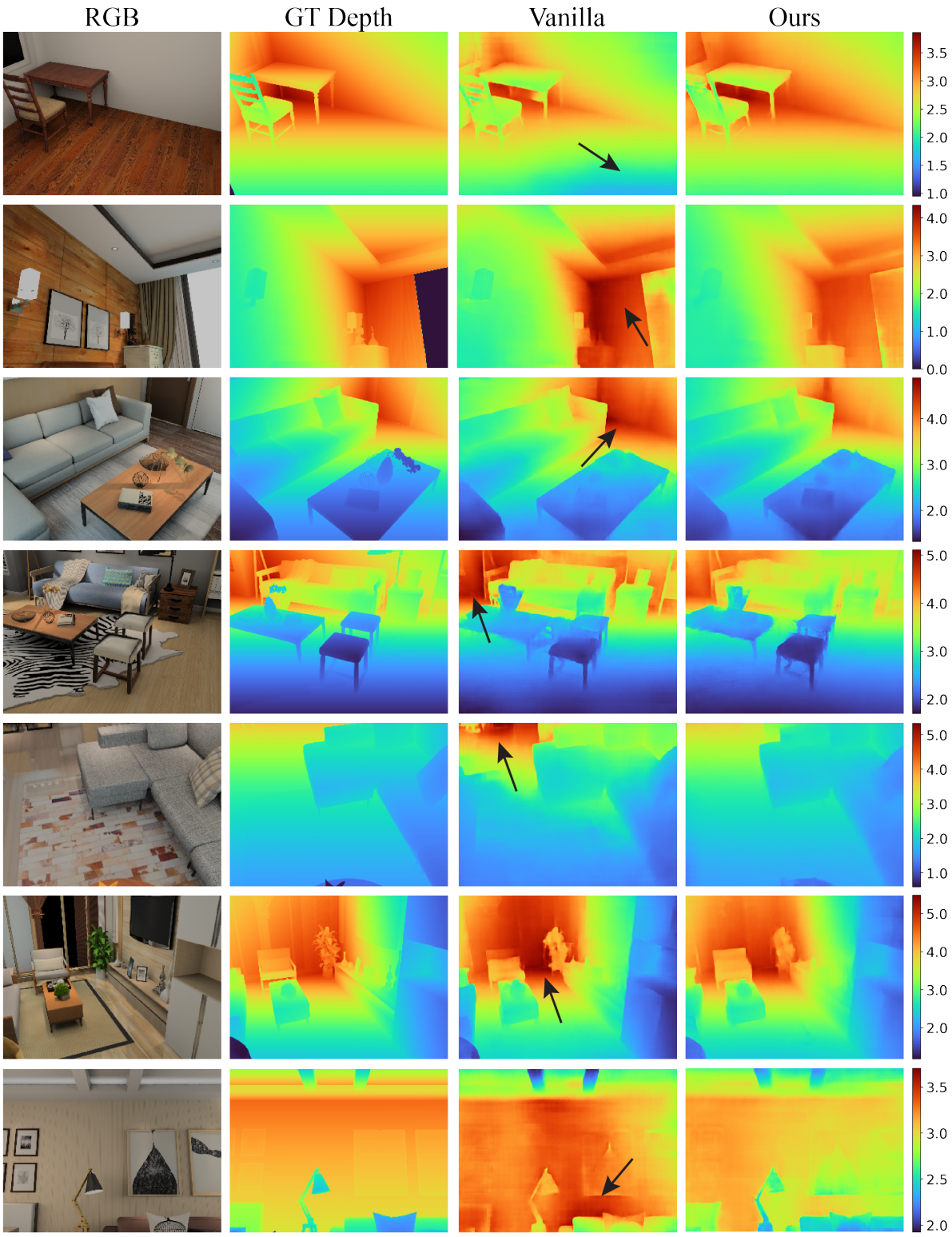


Figure 7: Depth predictions of Vanilla and our model (jointly applied CPP and PDA) on InteriorNet test-set. From these images captured under various camera poses, our model predicts better depth than Vanilla model in terms of the overall scale.

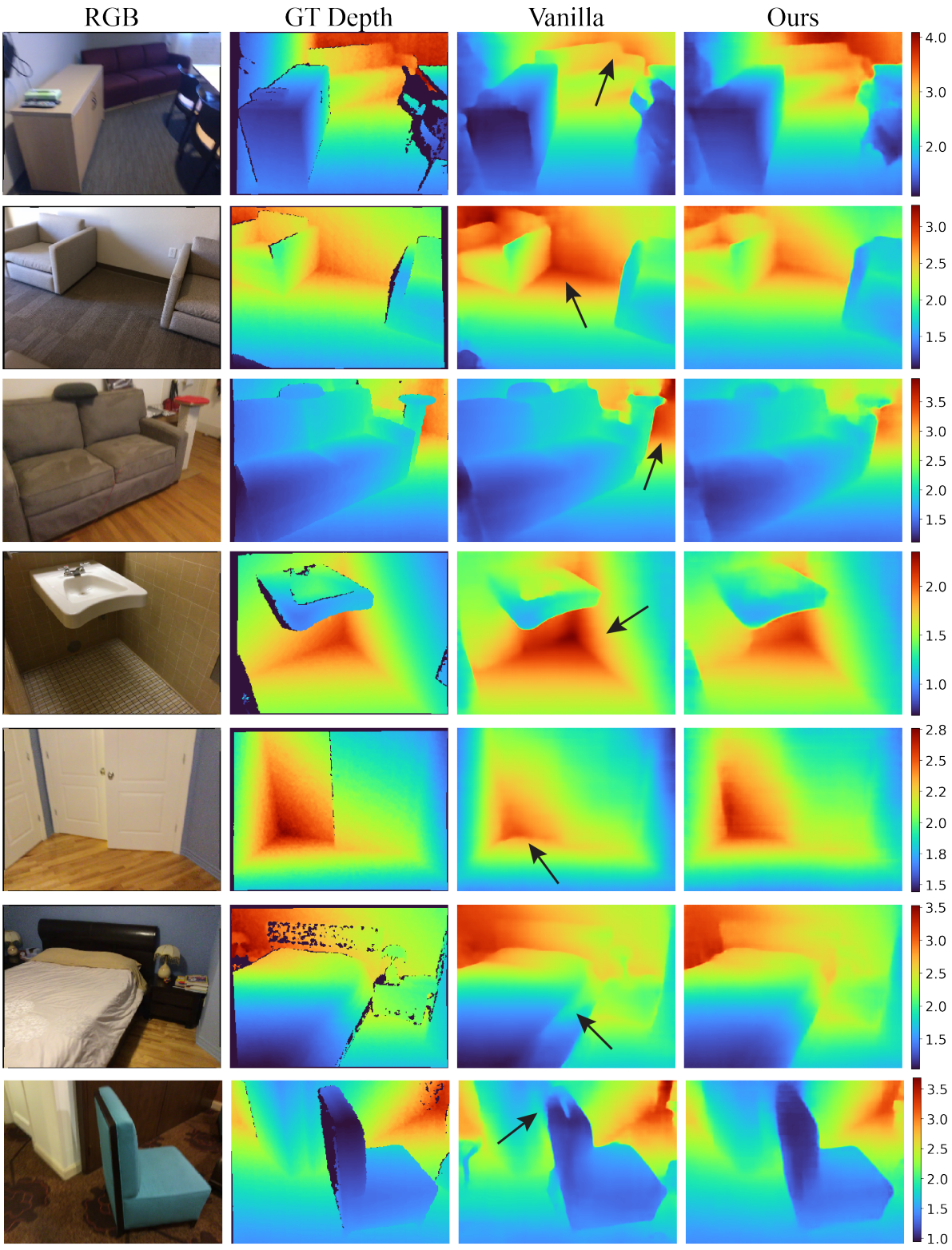


Figure 8: Depth predictions of Vanilla and our model (jointly applied CPP and PDA) on ScanNet test-set. From these images captured under various camera poses, our model predicts better depth than Vanilla model in terms of the overall scale.