Camera Pose Matters: Improving Depth Prediction by Mitigating Pose Distribution Bias

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

[Project Page] [Github] [Slides]

Abstract

Monocular depth predictors are typically trained on large-scale training sets which are naturally biased w.r.t the distribution of camera poses. As a result, trained predictors fail to make reliable depth predictions for testing examples captured under uncommon camera poses. To address this issue, we propose two novel techniques that exploit the camera pose during training and prediction. First, we introduce a simple perspective-aware data augmentation that synthesizes new training examples with more diverse views by perturbing the existing ones in a geometrically consistent manner. Second, we propose a conditional model that exploits the per-image camera pose as prior knowledge by encoding it as a part of the input. We show that jointly applying the two methods improves depth prediction on images captured under uncommon and even never-before-seen camera poses. We show that our methods improve performance when applied to a range of different predictor architectures. Lastly, we show that explicitly encoding the camera pose distribution improves the generalization performance of a synthetically trained depth predictor when evaluated on real images.

1. Introduction

Monocular depth prediction aims to estimate 3D scene geometry from a 2D image. Despite being a largely underdetermined problem, convolutional neural network (CNN) based depth predictors trained on a sufficiently large-scale dataset are able to learn the joint statistics of scene geometry and appearance, and achieve impressive performance [11, 10, 13, 16, 27, 49].

However, an important overlooked fact is that the distribution of camera poses in training sets are naturally *biased*. As a result, a learned depth predictor is unable to make reliable predictions on images captured from uncommon camera poses, as shown in Fig. 1. Importantly, camera poses of testing examples may follow a different distribution from that in the training set. This will exacerbate prediction errors on images that are captured by cameras with uncommon poses relative to the training set.

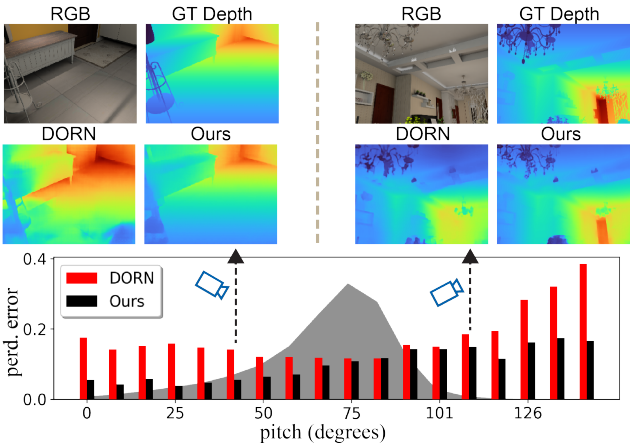

Figure 1: Contemporary monocular depth predictors, e.g., DORN [13], rely on large-scale training data which is naturally *biased* w.r.t the distribution of camera poses (e.g., pitch angle distribution shown in gray). As a result, DORN makes unreliable predictions on test images captured with uncommon poses (red bars), e.g., pitch angles $>120°$. To address this issue, we propose two novel techniques that drastically reduce prediction errors (cf. black bars) by leveraging perspective-aware data augmentation during training and known camera pose at test time. Qualitative examples with more extreme camera pitch angles (top) show that incorporating our techniques leads to notable improvements.

Contributions. To this end, we propose two novel approaches that significantly improve depth prediction under diverse test-time camera poses. First, we introduce a simple *perspective-aware data augmentation* (PDA) that synthesizes new geometrically consistent training examples with more diverse viewpoints by perturbing the camera pose of existing samples. In contrast, common data augmentation (CDA) methods such as random-crop, though widely adopted in prior work [13, 25, 52, 22, 49, 6], produce training examples where the resulting image and target depth are

inconsistent with the perspective geometry (Fig. 2). Second, we propose training a conditional depth predictor which utilizes the camera pose (e.g., acquired from IMU or other pose predictors) as a prior (CPP) when estimating depth. We propose an effective approach to encode CPP as an additional channel alongside the RGB input. We find incorporating pose using CPP yields more accurate depth predictors that generalize much better under diverse test-time camera poses.

Through extensive experiments, we show that these techniques significantly improve depth prediction on images captured from uncommon and even never-before-seen camera poses. Both techniques are general and broadly applicable to *any* network architecture. We show that incorporating them in recent state-of-the-art architectures improves their performance further. Lastly, we show that explicitly handling the biased camera pose distribution can improve the performance of a synthetically trained depth predictor when tested on real images. This highlights the importance that camera pose distribution plays in domain adaptation for 3D geometric prediction tasks.

2. Related Work

Monocular Depth Prediction and scene layout estimation have been greatly advanced since the seminal works [21, 40]. State-of-the-art approaches train increasingly sophisticated CNN-based predictors [34, 11, 27], utilize better training losses [13, 41, 49] and train on larger-scale training datasets [32, 28, 51, 54].

Surprisingly little attention has been paid to camera pose bias and out-of-distribution generalization. The recent study of [9] concluded that the learned depth predictors have a strong implicit bias causing them to perform poorly on test images when captured from differing camera poses. Our work systematically analyzes this pose bias/robustness in detail and offers technical contributions that improve generalization on test images captured under diverse camera poses.

Camera Pose Estimation plays an essential role in many traditional 3D geometry vision problems such as SLAM [2, 44, 23] and 3D reconstruction [42, 1]. Predicting the relative camera pose between a pair of frames has been widely exploited to perform self-supervised learning of monocular depth prediction [15, 56, 45]. Absolute camera pose is often represented implicitly in predictions of room layout [29, 58, 47, 46]. Closer to our work is [3] which estimates the absolute camera pose (height,pitch,roll) in order to regularize depth predictions in world coordinates.

We explore the benefit of providing the camera pose as an additional *input* to depth prediction. In practical applications, such pose information may come from other sensors (e.g., pitch from IMU) or prior knowledge (e.g., camera mounted at a known height and pitch on an autonomous vehicle). The work of [19, 12] encode camera intrinsics (e.g., focal length) as a part of the input for depth prediction, with a goal to learn a universal depth predictor that generalizes to images captured by different cameras. Similarly, we propose to encode camera extrinsic parameters (e.g., camera height) which we exploit for training better depth predictors that perform well on testing images captured with diverse camera extrinsic parameters.

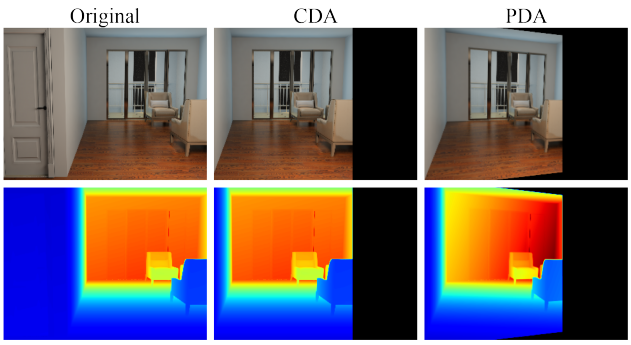

Figure 2: **Visual comparison of PDA and CDA**. From the original example (left), conventional data augmentation (CDA) synthesizes a new example (middle) by randomly cropping a sub-region. It ignores camera pose information and will simply copy the depth values w.r.t the corresponding pixels. In contrast, perspective-aware augmentation (PDA) simulates a rotation of the camera and synthesizes a new training example with geometrically consistent depth values corresponding to the new camera pose (right).

Distribution Bias. Challenges of class imbalance and bias have been a widely discussed topic in the literature on classification and recognition [17, 57, 35]. Distribution bias naturally exists in a training set, implying that some examples are underrepresented or few in number w.r.t some attributes (e.g., class labels). As a result, a trained model is unlikely to perform well on the underrepresented testing inputs. Even worse, testing examples may come from out-of-distribution data [20, 33, 30], meaning that the training set does not have similar examples. For example, in monocular depth prediction, training examples might be collected with cameras held vertical with minimal pitch and roll variations, but the testing scenario (e.g., AR/VR headset) might have potentially large variations in pitch and roll.

3. Perspective-aware Data Augmentation

Due to the biased distribution of camera poses in the training set, some examples are underrepresented w.r.t camera poses. To resolve this issue, we would like to augment the training data to cover these underrepresented poses.

Resampling (RS) the training data with replacement to enlarge the prevalence of uncommon poses in the train-set is perhaps the simplest approach. However, this does not increase the diversity of the train-set because it cannot generate new training examples. Perhaps even worse, in practice, training on repeated examples from uncommon camera

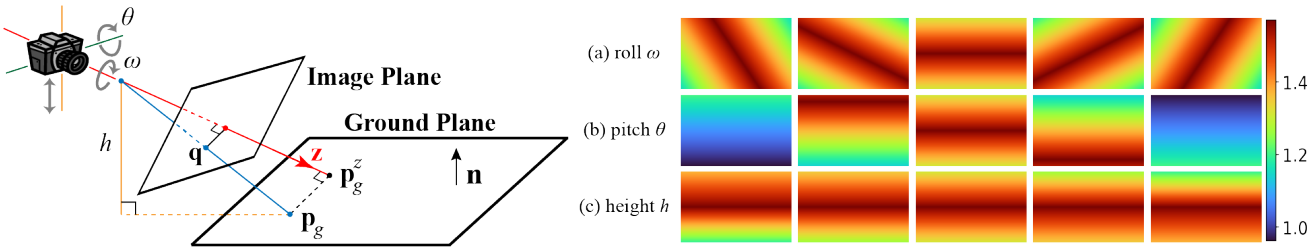

Figure 3: **Left:** We illustrate the proposed CPP which encodes camera pose (ω, θ, h) as a 2D image. Intuitively, for a spatial coordinate $\mathbf{q}$ on the image plane (i.e., the encoding map), we find its physical point $\mathbf{p}_g$ on the ground plane, along the ray cast from the camera. Then we compute the pseudo depth value as the length from the camera to $\mathbf{p}_g^z$ which is the projection of $\mathbf{p}_g$ onto the depth direction $\mathbf{z}$ (red line) using Eqn. 7. This results in an encoded CPP map for a given camera pose. **Right:** We visualize some encoded CPP maps by varying the ω, θ and h independently.

poses forces the model to weight them more (cf. overfitting), while sacrificing performance on other examples captured under common camera poses.

Conventional data augmentation (CDA), specifically random-cropping of the training examples, is widely used to enrich the training data in prior work [11, 26, 13, 25, 49].

While CDA seems to increase the diversity of the training set, its generated data could adversely affect depth prediction. Cropping a subregion from an original example naively copies depth values without considering the view-dependent nature of depth maps that depend on both the camera pose and scene geometry. Such a cropped image is equivalent to capturing another image of the same scene using a camera with an off-center principle point (Fig. 2) and can make it difficult to train depth predictors [12]. In our work, we find CDA helps only if we crop sufficiently large regions, which presumably reduces this effect.

Perspective-aware data augmentation (PDA) is our solution that augments training examples consisting of RGB images and depth maps. Given a training example, PDA first perturbs the camera pose, re-projects all pixels to a new image, and recomputes the depth values for the new image using the new camera pose and the original depth map. Despite its simplicity, to the best of our knowledge, *no prior work exploits this idea for data augmentation during the training of a monocular depth predictor*.

Given the training image $\mathbf{I}$, depth $\mathbf{D}$ and camera intrinsic matrix $\mathbf{K}$, we would like to synthesize a new image $\mathbf{I}_s$ and depth $\mathbf{D}_s$ corresponding to a perturbed viewing direction. Let $\mathbf{T}_{rel}$ be the relative rotation between the two poses. Then for any point $\mathbf{q} = [u\ v]$ on $\mathbf{I}$, we compute the corresponding point $\mathbf{q}'$ on the new image $\mathbf{I}_s$ via the homography:

$$\mathbf{q}'_h \sim \mathbf{K}\mathbf{T}_{rel} z \mathbf{K}^{-1}\mathbf{q}_h, \tag{1}$$

where z is the corresponding depth value of the point $\mathbf{q}$ from the original depth map $\mathbf{D}$; $\mathbf{q}_h$ and $\mathbf{q}'_h$ are the homogeneous coordinates of points $\mathbf{q}$ and $\mathbf{q}'$, respectively.

Similarly, we compute the depth value for each pixel in the new depth map $\mathbf{D}_s(q')$:

$$z' = \mathbf{v}_{proj}^{\mathrm{T}} \mathbf{T}_{rel} z \mathbf{K}^{-1}\mathbf{q}_h, \tag{2}$$

where $\mathbf{v}_{proj} = [0, 0, 1]^{\mathrm{T}}$ is a unit vector pointed along the z-axis of the camera.

While the above demonstrates the computation of per-pixel depth values, in practice, we can compute the whole depth map $\mathbf{D}_s$ efficiently by using backward-warping and standard bilinear interpolation. For larger rotations, we note that the synthesized views will have void regions on the boundary, as shown Fig. 2. This does not pose a problem for depth since we simply exclude those regions from the loss during training. However, we find it helpful to pad the RGB void regions using values from the RGB images (using the "reflection" mode during warping).

We also considered applying PDA with camera translation. However, this requires forward-warping and introduces "holes" in the synthesized result [37] at disoccluion boundaries. Handling disocclusions is still an open problem and out of our scope [5, 38, 36], therefore, we choose to only augment camera rotations to avoid disocclusion artifacts and allow for efficient computation.

4. Depth Prediction with Camera Pose Prior

Depth maps are *view-dependent representations* that depend on both the scene geometry and camera pose. The camera pose inherently provides *prior knowledge* about the expected scene depth map. For example, knowing a camera is pointing down to the ground at one-meter height, we should expect a depth map of one-meter height roughly everywhere. Therefore, we are motivated to train a conditional depth predictor on camera pose as prior (CPP).

Camera pose is a six-degree-of-freedom (6DoF) vector that describes translation and rotation in 3D world coordinates. In typical terrestrial man-made scenes, we consider a global coordinate with one axis pointing upwards (as specified by gravitational acceleration) and fix the origin along that axis to be 0 at the ground plane. Since there is no unique origin along the two remaining axes, we assume that our

camera pose prior should be uniform over translations parallel to the ground plane. Similarly, there is no unique way to specify the orientation of the axes parallel to the ground plane so our prior should necessarily be uniform over rotations of the camera about the up-axis. This leaves three degrees-of-freedom (3DoF): the height of the camera above the ground plane h, the pitch (angle relative to the up-axis) of the camera θ, and any roll ω of the camera around its optical axis (Fig. 3).

A naive encoding approach. We now consider using this 3DoF camera pose as a part of the input (along with RGB) to depth predictors. To incorporate this as input into a CNN, inspired by the literature [12, 50], we convert 3DoF camera poses into 2D maps, which are concatenated with the RGB image as a whole input to learn the depth predictor. Naively, we can create three more channels of resolution $H \times W$, which copy values of roll ω, pitch θ and height h, i.e., $\mathbf{M}_\omega[:,:] = \omega$, $\mathbf{M}_\theta[:,:] = \theta$ and $\mathbf{M}_h[:,:] = h$, respectively. However, the effect of pose on the depth distribution depends strongly on the position in the image relative to the camera center so translation-equivariant convolutions cannot fully exploit this encoding (except by relying on boundary artifacts). This is supported by an experimental comparison showing this naive encoding is inferior to our proposed *CPP* encoding method, elaborated in the following.

CPP encoding encodes the pose locally by assuming that the camera is placed in an empty indoor scene with an infinite floor and ceiling. Intuitively, it encodes the camera pose by intersecting rays from the camera center with predefined ground/ceiling planes and recording the depth, see Fig. 3.

Let $\mathbf{p} = [x\ y\ z]$ be a 3D point in the global coordinate, $\mathbf{q} = [u\ v]$ be a 2D point on the image plane whose homogeneous form is $\mathbf{q}_h$, $\mathbf{n} \in \mathbb{R}^3$ denotes the normal vector of ground planes, C be the distance between two planes in the up direction.

The projection from 3D coordinates to 2D image plane is:

$$\lambda \mathbf{q}_h = \mathbf{K}\mathbf{R}^{-1}(\mathbf{p} - \mathbf{t}), \tag{3}$$

where λ is a scale factor; $\mathbf{K}$ is the intrinsic matrix; $\mathbf{R} \in SO(3)$ and $\mathbf{t} \in \mathbb{R}^3$ are rotation and translation matrices known from the camera pose, respectively. We compute the 3D point $\mathbf{p}$ where the ray shooting from the camera center through point $\mathbf{q}$ eventually intersects with planes or the horizon. However, λ is an unknown scalar and we need extra constraints to compute it. Taking ground plane as an example, we know the collections of 3D points intersecting with ground plane is $\{\mathbf{p} : \mathbf{n}^{\mathrm{T}}\mathbf{p} = 0\}$. With this new constraint, we rearrange Eqn. 3 and multiply $\mathbf{n}^{\mathrm{T}}$ on both sides of the equation:

$$\mathbf{n}^{\mathrm{T}}\mathbf{p} = \lambda \mathbf{n}^{\mathrm{T}}\mathbf{R}\mathbf{K}^{-1}\mathbf{q}_h + \mathbf{n}^{\mathrm{T}}\mathbf{t} \ \Rightarrow\ \lambda = \frac{-\mathbf{n}^{\mathrm{T}}\mathbf{t}}{\mathbf{n}^{\mathrm{T}}\mathbf{R}\mathbf{K}^{-1}\mathbf{q}_h} \tag{4}$$

Now, we plug the computed λ back to Eqn. 3, then the 3D point $\mathbf{p}_g$ that intersects the ground plane for $\mathbf{q}$ is:

$$\mathbf{p}_g = \frac{-\mathbf{n}^{\mathrm{T}}\mathbf{t}}{\mathbf{n}^{\mathrm{T}}\mathbf{R}\mathbf{K}^{-1}\mathbf{q}_h}\mathbf{R}\mathbf{K}^{-1}\mathbf{q}_h + \mathbf{t} \tag{5}$$

Similarly, with the constraint $\{\mathbf{p} : \mathbf{n}^{\mathrm{T}}\mathbf{p} = C\}$, the 3D point $\mathbf{p}_c$ that intersects with the ceiling plane for $\mathbf{q}$ is:

$$\mathbf{p}_c = \frac{C - \mathbf{n}^{\mathrm{T}}\mathbf{t}}{\mathbf{n}^{\mathrm{T}}\mathbf{R}\mathbf{K}^{-1}\mathbf{q}_h}\mathbf{R}\mathbf{K}^{-1}\mathbf{q}_h + \mathbf{t} \tag{6}$$

Once we have the 3D intersection point $\mathbf{p}$ for each point $\mathbf{q}$ on the image plane, we compute the projection on the camera $\mathbf{z}$ direction (i.e., the depth direction):

$$z(\mathbf{p}) = \mathbf{v}_{proj}^{\mathrm{T}}\mathbf{R}^{-1}(\mathbf{p} - \mathbf{t}), \tag{7}$$

where $\mathbf{v}_{proj}$ is the projection vector that computes the projection of a 3D point along the camera depth direction. Finally, for each point $[u, v]$ in the encoding map $\mathbf{M} \in \mathbb{R}^{H \times W}$, we compute both $\mathbf{p}_g$ and $\mathbf{p}_c$ and take the maximum (positive) value as the pseudo depth value (Fig. 3):

$$\mathbf{M}[u, v] = \max\{z(\mathbf{p}_g), z(\mathbf{p}_c)\} \tag{8}$$

The encoding map $\mathbf{M}$ can have infinite values, e.g., when the ray shooting from the camera is parallel to the ground plane. To map values into a finite range, we apply the inverse tangent operator $\tan^{-1}(\cdot)$ to obtain our final encoding $\mathbf{M}_{CPP} = \tan^{-1}(\mathbf{M})$ which takes on values in the range $[\tan^{-1}(\min\{h, C - h\}), \frac{\pi}{2}]$. We visualize some CPP maps in Fig. 3-right.

To train a conditional depth prediction, we simply concatenate the CPP encoded map with the corresponding RGB as a four-channel input. This implies that our CPP encoding approach applies to *any* network architectures with a simple modification on the first convolution layer.

5. Experiments

We validate our methods through extensive experiments. Specifically, we aim to answer the following questions:[1]

- Do our methods improve depth prediction on a test-set that has a different camera pose distribution from the train-set?
- Does resampling improve depth prediction on a test-set that has a different camera pose distribution?
- Do our methods improve depth estimation on testing images captured with out-of-distribution camera poses?
- Does applying our methods to other state-of-the-art depth predictors improve their performance?
- Do our methods improve the performance of a depth predictor when training and testing on different datasets?

[1] Answers: yes, no, yes, yes, yes.

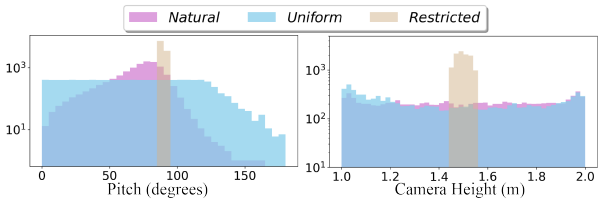

Figure 4: Distribution of camera pitch and heights for three subsets of images from InteriorNet. From the *Natural* subsect, we observe the dataset of InteriorNet does have a naturally biased distribution (esp. pitch). Please refer to the text on how we construct the three subsets.

Datasets. In our work, we use two publicly available large-scale indoor-scene datasets, InteriorNet [31] and ScanNet [8] which come with ground-truth depth and camera pose. Compared to other datasets such as [14, 7, 43], these were selected because they illustrate a much wider variety of camera poses. InteriorNet [31] consists of photo-realistic synthetic video sequences with randomly generated camera trajectories with a wide variety of pitch and height but minimal roll. ScanNet [8] contains real-world RGBD videos of millions of views from 1,513 indoor scenes collected with substantial variation in pitch, roll, and height. For each dataset, we create train/test sets by randomly splitting *scenes* into two disjoint sets with 85%/15% examples, respectively. We use a stratified sampling approach to avoid picking adjacent video sequences in the same set.

Sampling. To explore how the biased camera pose distribution affects depth learning and prediction, we sample three subsets from the test/train set (Fig. 4) which each have 10k/1k images but with different distributions of camera poses.

- *Natural* selects samples at random to reflect the natural distribution of the dataset.
- *Uniform* selects samples that simulate a uniform distribution over poses with priority: pitch→roll→height. Concretely, we quantize the range of camera pitch/roll/ height into equal-size bins and sample an approximately equal number of examples in each bin.
- *Restricted* samples images within a narrow range: camera pitch $\theta \in [85°, 95°]$ and height $h \in [1.45, 1.55]$ (meters). While InteriorNet does not have roll variations, ScanNet does: roll $\omega \in [-5°, 5°]$. We create *Restricted* to particularly study how depth predictor performs on testing images captured with out-of-distribution camera poses.

Evaluation Metrics. There are several evaluation metrics widely used in the literature [11, 13, 49], including absolute relative difference (Abs^r), squared relative difference (Sq^r), root mean squared log error (RMS-log), and accuracy with a relative error threshold of $\delta^k < 1.25^k, i = 1, 2$.

Implementation. We use a standard UNet structured model to perform most of our experimental analysis [54, 53, 55]. We also demonstrate that our techniques apply to other depth predictor architectures (Section 5.3). Unless specified, all models are trained with the L1 loss and applied random left-right flip augmentation during training.

While we initially learn the conditional depth predictor using the *true* camera pose, we also tested encoding a *predicted* camera pose in Section 5.4. For predicting camera pose, we train a pose predictor with the ResNet18 architecture [18] to directly regresses to camera pitch, roll, and height. We resize all images and depth maps to 240×320, and adjust camera intrinsic parameters (for CPP encoding) accordingly. We use PyTorch [39] to train all the models for 200 epochs on a single Titan Xp GPU. We use the Adam optimizer [24] with a learning rate 1e-3, and coefficients 0.5

Table 1: **Within & cross-distribution evaluation**. In each dataset, we train depth predictors on their *Natural* train-sets and evaluate on both *Natural* and *Uniform* test-sets. We apply different methods to a Vanilla model. All models use the same network architecture. Vanilla performs poorly in cross-distribution evaluation (cf. *Natural*-test vs. *Uniform*-test), demonstrating that the biased camera pose distribution affects the training of depth predictors. As expected, RS hurts depth prediction compared to Vanilla. In contrast, CPP and PDA show better performance; jointly applying them performs the best (i.e., "Both"). Finally, comparing alternative methods to our CPP (vs. Native) and PDA (vs. RS and CDA) shows the merits of our methods.

Models	*Natural-Test-Set*		*Uniform-Test-Set*	
	↓ better Abs^r/Sq^r/RMS-log	↑ better δ^1 / δ^2	↓ better Abs^r/Sq^r/RMS-log	↑ better δ^1 / δ^2
InteriorNet				
Vanilla	.154 / .148 / .229	.803 / .945	.183 / .146 / .250	.724 / .926
+ RS	.192 / .203 / .267	.726 / .918	.210 / .174 / .272	.661 / .906
+ CDA	.142 / .137 / .222	.825 / .950	.172 / .125 / .238	.738 / .934
+ PDA	.138 / .123 / .207	.834 / .957	.168 / .122 / .229	.757 / .942
+ Naive	.132 / .145 / .219	.835 / .944	.137 / .116 / .213	.810 / .944
+ CPP	.108 / .120 / .199	.872 / .958	.106 / .088 / .183	.876 / .961
+ Both	**.095 / .101 / .180**	**.898 / .966**	**.091 / .069 / .161**	**.903 / .973**
ScanNet				
Vanilla	.125 / .068 / .186	.837 / .962	.177 / .121 / .265	.711 / .928
+ RS	.216 / .158 / .279	.619 / .889	.218 / .168 / .300	.630 / .881
+ CDA	.116 / .062 / .179	.853 / .964	.174 / .121 / .264	.727 / .922
+ PDA	.115 / .059 / .171	.860 / .970	.166 / .110 / .248	.752 / .938
+ Naive	.120 / .069 / .184	.846 / .959	.173 / .127 / .255	.755 / .923
+ CPP	.108 / .060 / .171	.871 / .965	.154 / .106 / .239	.781 / .943
+ Both	**.102 / .052 / .160**	**.882 / .973**	**.143 / .097 / .230**	**.809 / .952**

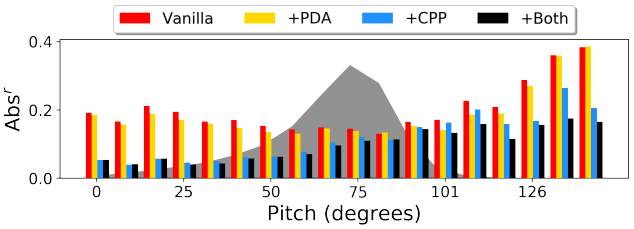

Figure 5: Breakdown analysis of depth prediction w.r.t pitch. The background shade denotes the camera pose distribution w.r.t pitch. Clearly, both PDA and CPP improve depth prediction in under-represented camera poses. Surprisingly, CPP remarkably boosts depth prediction, while applying both CPP and PDA achieves the best performance "everywhere".

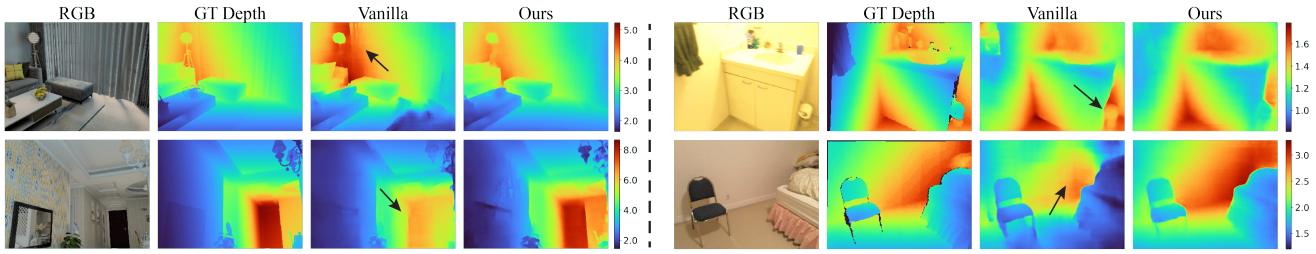

Figure 6: Qualitative comparison between Vanilla and our method (using both CPP and PDA) on random testing images from InteriorNet (left) and ScanNet (right). Depth maps for each example are shown with the same colorbar range. Notably, the images are captured under some uncommon camera angles relative to the training pose distribution (Fig. 5). The Vanilla model seems to make erroneous predictions w.r.t the overall scale of the depth. In contrast, by applying CPP and PDA, the new model ("ours") produces visually improved results.

and 0.999 for computing the running averages of gradient and its square.

For CPP encoding, we set the ceiling height $C = 3$ meters. We have studied different settings of C, but find little difference (details in the supplement). For PDA, we randomly perturb camera pose within $[-0.1, 0.1]$ radius angle jointly w.r.t pitch, roll, and yaw; we also ablate the perturbation scale in Section 5.5.

5.1. Within & Cross-Distribution Evaluation

We start with initial experiments designed to reveal how bias in camera pose statistics affects depth predictor performance (Fig. 4). The experiments also explore the design choices of our methods and validate their effectiveness. Specifically, we train depth predictors on the *Natural* train-sets, and test them on both *Natural* and *Uniform* test-sets. We apply a sequence of different modifications to a "Vanilla" baseline model, i.e., a UNet-based predictor. Table 1 lists detailed comparisons, and Fig. 6 shows qualitative comparisons.

The Vanilla model degrades notably when evaluated on a test-set that has a different pose distribution, i.e., from *Natural* to *Uniform*, showing the clear influence from the biased distribution of camera poses. In terms of data augmentation, simply resampling the training data (RS) hurts performance (cf. Vanilla and "+RS"). Moreover, our PDA outperforms CDA, demonstrating the importance of synthesizing geometric-aware training examples using the corresponding camera and the scene geometry in depth prediction. As for camera pose encoding, our CPP encoding method clearly outperforms the Naive method, Importantly, jointly applying CPP and PDA performs the best on both within-distribution (*Natural*) and cross-distribution (*Uniform*) test-sets. Finally, we breakdown the performance in Fig. 5 to analyze when (i.e., w.r.t pitch angle) PDA and CPP improve depth prediction. Generally, both of them help reduce prediction errors on testing images captured with underrepresented camera poses, while CPP yields the largest benefits. Applying both achieves the best performance "*everywhere*".

Table 2: **Out-of-distribution evaluation**. We train depth predictors on the *Restricted* train-sets and test on both *Restricted* and *Natural* test-sets of each datasets. Vanilla model performs poorly on *Natural* test-sets, clearly showing the challenge of depth prediction on images captured under novel/never-before-seen camera poses. CPP slightly improves performance, but PDA helps more. CDA also improves performance presumably because it synthesizes more training examples, but underperforms PDA. As expected, jointly applying both CPP and PDA achieves the best performance on both *Restricted* and *Natural* test-sets.

Models	*Restricted-Test-Set*		*Natural-Test-Set*	
	↓ better Abs^r/Sq^r/RMS-log	↑ better δ^1 / δ^2	↓ better Abs^r/Sq^r/RMS-log	↑ better δ^1 / δ^2
InteriorNet				
Vanilla	.150 / .234 / .296	.819 / .916	.265 / .368 / .412	.538 / .767
+ CPP	.149 / .237 / .301	.820 / .915	.264 / .350 / .397	.548 / .782
+ CDA	.139 / .228 / .286	.830 / .922	.227 / .279 / .336	.637 / .845
+ PDA	.136 / .178 / .239	.833 / .935	.237 / .266 / .295	.652 / .878
+ Both	**.131 / .170 / .237**	**.839 / .936**	**.216 / .231 / .286**	**.666 / .894**
ScanNet				
Vanilla	.177 / .131 / .259	.705 / .897	.317 / .304 / .390	.444 / .748
+ CPP	.169 / .125 / .258	.726 / .897	.301 / .283 / .384	.464 / .756
+ CDA	.163 / .121 / .259	.724 / .904	.289 / .266 / .371	.467 / .771
+ PDA	.160 / .112 / .244	.729 / .914	.283 / .251 / .353	.493 / .795
+ Both	**.155 / .108 / .228**	**.731 / .918**	**.277 / .245 / .348**	**.504 / .804**

5.2. Out-of-Distribution Evaluation

We now study how our methods help when training depth predictors on a train-set which have a rather restricted range of camera poses. Specifically, we train models on the *Restricted* train-sets of InteriorNet and ScanNet, respectively, and test the models on their *Natural* test-sets. This setup is synthetic and unlikely to be a real-world scenario, but it allows for exclusive analysis of depth predictors when tested on images captured with out-of-distribution or never-before-seen camera poses.

Table 2 lists detailed comparisons. Vanilla model performs poorly when tested on *Natural* test-set. CPP slightly improves prediction, but PDA helps a lot. CDA (i.e., random-crop as augmentation) also helps, presumably owing to more diverse training examples, but underperforms our PDA. This further confirms the importance of generating geometric-aware new training examples for the given scene and camera in depth prediction. Under expectation, applying both PDA and CPP performs the best.

Table 3: **Applicability to other predictor architectures**. In each dataset, we train state-of-the-art depth predictors (DORN [13] and VNL [49]) by optionally applying our CPP and PDA approaches. All models are trained/tested on the *Natural* train/test-sets per dataset. Clearly, both CPP and PDA boost their performance.

Models	InteriorNet		ScanNet	
	↓ better Abs^r/Sq^r/RMS-log	↑ better δ^1 / δ^2	↓ better Abs^r/Sq^r/RMS-log	↑ better δ^1 / δ^2
DORN	.128 / .126 / .218	.854 / .957	.112 / .065 / .197	.856 / .959
+ CPP	.098 / .105 / .197	.899 / .962	.108 / .062 / .191	.875 / .960
+ PDA	.115 / .112 / .207	.867 / .958	.110 / .068 / .195	.857 / .963
+ Both	**.085 / .088 / .124**	**.912 / .971**	**.093 / .049 / .153**	**.903 / .979**
VNL	.131 / .140 / .235	.853 / .954	.115 / .069 / .205	.855 / .960
+ CPP	.101 / .120 / .207	.893 / .960	.112 / .064 / .195	.871 / .961
+ PDA	.117 / .115 / .210	.861 / .959	.110 / .065 / .199	.855 / .962
+ Both	**.086 / .085 / .131**	**.909 / .970**	**.095 / .051 / .157**	**.901 / .980**

5.3. Applicability to Other Predictor Networks

CPP and PDA are general and applicable to different model architectures. We study applying them to training two well-established depth predictors: DORN [13] and VNL [49]. Compared to the Vanilla model, both predictors adopt different network architectures [48, 18] and different loss functions. They convert continuous depth values to discrete bins and model depth prediction as a classification problem. Moreover, VNL incorporates a virtual surface normal loss which provides a geometric-aware regularization during training. We implement DORN and VNL using publicly-available third-party code. We train and test all models on the *Natural* train/test-sets, in InteriorNet and ScanNet, respectively.

Table 3 lists detailed results, which are comparable to the *Natural-Test-Set* column in Table 1. Consistent with previous experiments, both CPP and PDA improve the performance further based on DORN and VNL. As our CPP and PDA improve other depth predictors as a general approach, we suggest using them in future research of monocular depth estimation.

5.4. Synthetic-to-Real Generalization

Previous experiments have validated that addressing the biased camera pose distribution helps train a depth predictor that works better on another test-time distribution, or more generally another domain. Here we evaluate performance in the presence of substantial synthetic-to-real domain shift which includes both low-level shifts (e.g., local material appearance) and high-level shifts (novel objects, layouts and camera poses) [54]. Specifically, we synthetically train a depth predictor (on InteriorNet *Natural* train-set) and test it on real images (ScanNet *Natural* and *Uniform* test-sets). We also consider a more practical scenario that one does not have access to the true camera pose but instead must rely on the predicted poses. To this end, we train a camera pose predictor on ScanNet *Natural* test-set to predict camera pitch, height and roll for a given RGB image. Then, we perform CPP encoding with the predictive pose, i.e., CPP$_{pred}$

Table 4: **Mitigating distribution bias of camera poses improves synthetic-to-real domain adaptation**. We train depth predictors synthetically on InteriorNet (*Natural* train-set) and test them on real-world images from ScanNet *Natural* and *Uniform* test-sets. This is a typical setup for synthetic-to-real domain adaptation in the context of depth prediction. Interestingly, we find that CDA hurts the performance, presumably because the generated training examples by CDA do not obey the relations among camera model, scene geometry and camera pose, and hence do not necessarily help training a generalizable depth predictor. In contrast, our PDA helps synthetic-to-real generalization and applying CPP improves further. Importantly, applying CPP with predictive poses (CPP$_{pred}$) achieves a remarkable performance boost, whereas using the true camera pose in CPP performs the best.

Methods	*Natural-Test-Set*		*Uniform-Test-Set*	
	↓ better Abs^r/Sq^r/RMS-log	↑ better δ^1 / δ^2	↓ better Abs^r/Sq^r/RMS-log	↑ better δ^1 / δ^2
Vanilla	.242 / .204 / .315	.570 / .852	.305 / .259 / .364	.457 / .797
CDA	.246 / .205 / .321	.568 / .843	.316 / .288 / .391	.433 / .771
PDA	.239 / .198 / .311	.575 / .857	.298 / .252 / .359	.461 / .804
PDA+CPP$_{pred}$	.219 / .190 / .305	.586 / .866	.273 / .231 / .346	.548 / .835
PDA+CPP	**.208 / .170 / .282**	**.677 / .893**	**.245 / .228 / .315**	**.620 / .858**

Table 4 lists detailed setups and results; we summarize the salient conclusions. Vanilla achieves worse performance compared to Table 1, showing a clear domain gap between the two datasets. Applying CDA hurts the performance, presumably because the generated training data by CDA disobey the projective geometry relations between scene geometry and camera model and pose (cf. Section 3). In contrast, our PDA helps, but CPP improves even more notably. Applying CPP with the predictive pose (CPP$_{pred}$) achieves a remarkable performance boost over PDA, suggesting that using predictive poses, or more generally exploiting camera poses during training, is quite valuable in depth prediction. We analyze in the next section how the model is resilient to the errors in predicted poses. Lastly, using the true camera pose in CPP performs the best.

5.5. Further Discussion and Ablation Study

"Blind" depth prediction without RGB. To characterize the prior knowledge carried by camera poses in terms of depth prediction, we train a "blind predictor" on the InteriorNet *Natural* train-set, taking as input only the CPP encoded maps of camera poses *without* RGB images. For comparison, we compute an average depth map over the whole *Natural* train-set. We qualitatively compare results on two testing examples in Fig. 8 (quantitative results in the supplement). Visually, encoding camera pose alone using CPP reliably provides depth estimate on floor regions. This intuitively explains that using camera pose does serve as strong prior knowledge of scene depth.

Resilience to noisy camera pose. Camera pose estimates (e.g., from IMU sensors) are potentially noisy, and the predicted camera poses are undoubtedly erroneous (cf. the previous experiment using predicted poses in CPP). We study how resilient CPP is to test-time errors in the camera

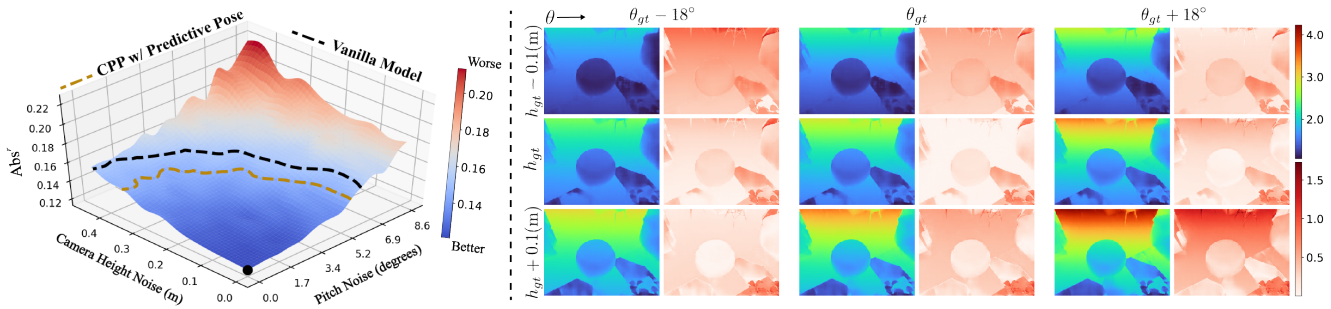

Figure 7: **Left:** We plot depth prediction error (Abs^r) w.r.t different levels of noise in camera height and pitch. We apply CPP to train/test a depth predictor (based on Vanilla) on InteriorNet *Natural* train/test-set. For a given noise level δ, the trained model makes depth predictions using a CPP map computed with a perturbed camera pose, e.g., the pitch is sampled from $\theta_{gt}+\mathbf{U}[-\delta, \delta]$. The black dot at the origin stands for the (best) performance using the true camera pose (i.e., no noises are presented in pitch and height). The dashed lines represent the average performance levels for the Vanilla and CPP with predictive poses. **Right**: We visualize depth prediction by CPP model when encoding perturbed camera pitch angles $\theta_{gt}\pm 18°$ and heights $h_{gt}\pm 0.1$(m). CPP model predicts shallower depth when both pitch and camera height decrease (i.e., camera is tilted down or translated closer to the floor). This qualitatively confirms that the camera pose prior induces a meaningful shift in the estimator. The corresponding RGB image and ground-truth depth appear in Fig. 8.

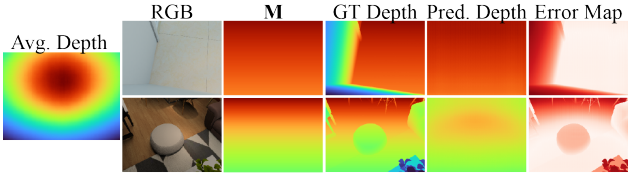

Figure 8: Camera pose alone provides a strong depth prior even for "blind" depth prediction. Specifically, over the InteriorNet *Natural* train-set, we train a depth predictor solely on the CPP encoded maps $\mathbf{M}$ *without* RGB as input. For visual comparison, we compute the average depth map (shown 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.

pose. Specifically, when testing a trained model with CPP, we randomly perturb the camera poses of all testing examples up to a pre-defined scale, and measure the overall performance as a function of prediction error w.r.t noise added to the true camera pitch θ_{gt} and height h_{gt}, as shown in Fig. 7. We find that CPP outperforms the vanilla model even with significant misspecification of the pose (e.g., height error < 0.3m, pitch error < 5 degrees).

Augmentation scales in PDA. We ablate the augmentation scale in PDA during training depth predictors, as detailed in Fig. 9. Perhaps surprisingly, applying a larger scale PDA consistently improves depth prediction until a very large perturbation (i.e., rotating at most 80°), presumably when very large void regions are introduced in the synthesized training examples (Fig. 2).

6. Conclusion

While large-scale datasets allow for end-to-end training of monocular depth predictors, we find the training sets naturally biased w.r.t distribution of camera poses. As a result, trained predictors fail to make reliable depth predictions for testing examples captured under uncommon camera poses. We mitigate this bias with two novel methods, perspective-aware data augmentation (PDA) and camera pose prior encoding (CPP). We show that applying both our methods improves depth prediction on images captured under uncommon or never-before-seen camera poses. Moreover, our methods are general and readily applicable to other depth predictors, which can perform better when trained with PDA and CPP, suggesting using them in the future research of monocular depth estimation.

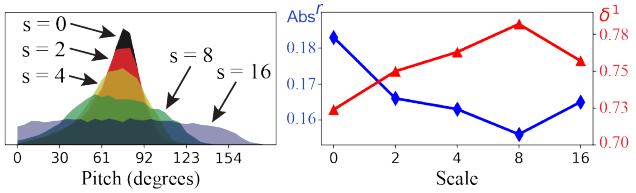

Figure 9: During training on InteriorNet *Natural* train-set, we randomly perturb camera pose to generate new training examples. We specify the scale of the perturbation $s = \{0, 2, 4, 8, 16\}$, meaning that, when $s = 2$, we randomly perturb pitch/roll/yaw angles by adding a perturbation within $[-s * 5°, s * 5°]$. **Left**: Applying more larger scale PDA "flattens" camera pose distribution of the whole training set. **Right**: We test each of the trained models on the InteriorNet *Uniform* test-set. We find applying more intense PDA consistently improves depth prediction until $s = 16$ (i.e., rotating at most 80°), presumably when very large void regions are introduced in the synthesized training examples (Fig. 2).

Acknowledgements We thank anonymous reviewers for their valuable suggestions. This research was supported by NSF grants IIS-1813785, IIS-1618806, a research gift from Qualcomm, and a hardware donation from NVIDIA.

References

[1] Sameer Agarwal, Yasutaka Furukawa, Noah Snavely, Ian Simon, Brian Curless, Steven M Seitz, and Richard Szeliski. Building rome in a day. *Communications of the ACM*, 54(10):105–112, 2011. 2

[2] Tim Bailey and Hugh Durrant-Whyte. Simultaneous localization and mapping (slam): Part ii. *IEEE robotics & automation magazine*, 2006. 2

[3] Manel Baradad and Antonio Torralba. Height and uprightness invariance for 3d prediction from a single view. In *CVPR*, 2020. 2

[4] 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. 14

[5] Pierre Buyssens, Olivier Le Meur, Maxime Daisy, David Tschumperlé, and Olivier Lézoray. Depth-guided disocclusion inpainting of synthesized rgb-d images. *IEEE Transactions on Image Processing*, 2016. 3

[6] Xiaotian Chen, Xuejin Chen, and Zheng-Jun Zha. Structure-aware residual pyramid network for monocular depth estimation. *arXiv preprint arXiv:1907.06023*, 2019. 1

[7] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Scharwächter, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset. In *CVPR Workshop on The Future of Datasets in Vision*, 2015. 5

[8] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *CVPR*, 2017. 5

[9] Tom van Dijk and Guido de Croon. How do neural networks see depth in single images? In *ICCV*, 2019. 2

[10] David Eigen and Rob Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *ICCV*, 2015. 1

[11] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. In *Advances in Neural Information Processing Systems*, 2014. 1, 2, 3, 5

[12] Jose M Facil, Benjamin Ummenhofer, Huizhong Zhou, Luis Montesano, Thomas Brox, and Javier Civera. Camconvs: camera-aware multi-scale convolutions for single-view depth. In *CVPR*, 2019. 2, 3, 4

[13] Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, and Dacheng Tao. Deep ordinal regression network for monocular depth estimation. In *CVPR*, 2018. 1, 2, 3, 5, 7

[14] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, 2012. 5

[15] Clément Godard, Oisin Mac Aodha, and Gabriel J Brostow. Unsupervised monocular depth estimation with left-right consistency. In *CVPR*, 2017. 2, 14

[16] Xiaoyang Guo, Hongsheng Li, Shuai Yi, Jimmy Ren, and Xiaogang Wang. Learning monocular depth by distilling cross-domain stereo networks. In *ECCV*, 2018. 1

[17] Haibo He and Edwardo A Garcia. Learning from imbalanced data. *IEEE Transactions on knowledge and data engineering*, 2009. 2

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 5, 7

[19] Lei He, Guanghui Wang, and Zhanyi Hu. Learning depth from single images with deep neural network embedding focal length. *IEEE Transactions on Image Processing*, 2018. 2

[20] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *arXiv preprint arXiv:1610.02136*, 2016. 2

[21] Derek Hoiem, Alexei A Efros, and Martial Hebert. Automatic photo pop-up. In *ACM SIGGRAPH 2005 Papers*, pages 577–584. 2005. 2

[22] Junjie Hu, Mete Ozay, Yan Zhang, and Takayuki Okatani. Revisiting single image depth estimation: Toward higher resolution maps with accurate object boundaries. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2019. 1

[23] Alex Kendall, Matthew Grimes, and Roberto Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *ICCV*, 2015. 2

[24] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5

[25] Shu Kong and Charless C Fowlkes. Recurrent scene parsing with perspective understanding in the loop. In *CVPR*, 2018. 1, 3

[26] Lubor Ladicky, Jianbo Shi, and Marc Pollefeys. Pulling things out of perspective. In *CVPR*, 2014. 3

[27] Iro Laina, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, and Nassir Navab. Deeper depth prediction with fully convolutional residual networks. In *2016 Fourth international conference on 3D vision (3DV)*, 2016. 1, 2

[28] Katrin Lasinger, René Ranftl, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *arXiv preprint arXiv:1907.01341*, 2019. 2

[29] Chen-Yu Lee, Vijay Badrinarayanan, Tomasz Malisiewicz, and Andrew Rabinovich. Roomnet: End-to-end room layout estimation. In *ICCV*, 2017. 2

[30] Kimin Lee, Honglak Lee, Kibok Lee, and Jinwoo Shin. Training confidence-calibrated classifiers for detecting out-of-distribution samples. *arXiv preprint arXiv:1711.09325*, 2017. 2

[31] Wenbin Li, Sajad Saeedi, John McCormac, Ronald Clark, Dimos Tzoumanikas, Qing Ye, Yuzhong Huang, Rui Tang, and Stefan Leutenegger. Interiornet: Mega-scale multi-sensor photo-realistic indoor scenes dataset. *arXiv preprint arXiv:1809.00716*, 2018. 5

[32] Zhengqi Li and Noah Snavely. Megadepth: Learning single-view depth prediction from internet photos. In *CVPR*, 2018. 2

[33] Shiyu Liang, Yixuan Li, and Rayadurgam Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. *arXiv preprint arXiv:1706.02690*, 2017. 2

[34] Fayao Liu, Chunhua Shen, Guosheng Lin, and Ian Reid. Learning depth from single monocular images using deep convolutional neural fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2015. 2

[35] Ziwei Liu, Zhongqi Miao, Xiaohang Zhan, Jiayun Wang, Boqing Gong, and Stella X Yu. Large-scale long-tailed recognition in an open world. In *CVPR*, 2019. 2

[36] Guibo Luo, Yuesheng Zhu, Zhenyu Weng, and Zhaotian Li. A disocclusion inpainting framework for depth-based view synthesis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019. 3

[37] Simon Masnou and J-M Morel. Level lines based disocclusion. In *Proceedings 1998 International Conference on Image Processing*, 1998. 3

[38] Eunbyung Park, Jimei Yang, Ersin Yumer, Duygu Ceylan, and Alexander C Berg. Transformation-grounded image generation network for novel 3d view synthesis. In *CVPR*, 2017. 3

[39] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017. 5

[40] Ashutosh Saxena, Sung H Chung, and Andrew Y Ng. 3-d depth reconstruction from a single still image. *International Journal of Computer Vision*, 76(1):53–69, 2008. 2

[41] Daeyun Shin, Zhile Ren, Erik B Sudderth, and Charless C Fowlkes. 3d scene reconstruction with multi-layer depth and epipolar transformers. In *ICCV*, 2019. 2

[42] Noah Snavely, Steven M Seitz, and Richard Szeliski. Photo tourism: exploring photo collections in 3d. In *ACM Siggraph 2006 Papers*, pages 835–846. 2006. 2

[43] Shuran Song, Fisher Yu, Andy Zeng, Angel X Chang, Manolis Savva, and Thomas Funkhouser. Semantic scene completion from a single depth image. In *CVPR*, 2017. 5

[44] Jürgen Sturm, Nikolas Engelhard, Felix Endres, Wolfram Burgard, and Daniel Cremers. A benchmark for the evaluation of rgb-d slam systems. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 573–580. IEEE, 2012. 2

[45] Benjamin Ummenhofer, Huizhong Zhou, Jonas Uhrig, Nikolaus Mayer, Eddy Ilg, Alexey Dosovitskiy, and Thomas Brox. Demon: Depth and motion network for learning monocular stereo. In *CVPR*, 2017. 2

[46] Scott Workman, Menghua Zhai, and Nathan Jacobs. Horizon lines in the wild. *arXiv preprint arXiv:1604.02129*, 2016. 2

[47] Wenqi Xian, Zhengqi Li, Matthew Fisher, Jonathan Eisenmann, Eli Shechtman, and Noah Snavely. Uprightnet: geometry-aware camera orientation estimation from single images. In *ICCV*, 2019. 2

[48] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *CVPR*, pages 1492–1500, 2017. 7

[49] Wei Yin, Yifan Liu, Chunhua Shen, and Youliang Yan. Enforcing geometric constraints of virtual normal for depth prediction. In *ICCV*, 2019. 1, 2, 3, 5, 7

[50] Yurong You, Yan Wang, Wei-Lun Chao, Divyansh Garg, Geoff Pleiss, Bharath Hariharan, Mark Campbell, and Kilian Q Weinberger. Pseudo-lidar++: Accurate depth for 3d object detection in autonomous driving. *arXiv preprint arXiv:1906.06310*, 2019. 4

[51] Yinda Zhang, Shuran Song, Ersin Yumer, Manolis Savva, Joon-Young Lee, Hailin Jin, and Thomas Funkhouser. Physically-based rendering for indoor scene understanding using convolutional neural networks. In *CVPR*, 2017. 2

[52] Zhenyu Zhang, Zhen Cui, Chunyan Xu, Yan Yan, Nicu Sebe, and Jian Yang. Pattern-affinitive propagation across depth, surface normal and semantic segmentation. In *CVPR*, 2019. 1

[53] Shanshan Zhao, Huan Fu, Mingming Gong, and Dacheng Tao. Geometry-aware symmetric domain adaptation for monocular depth estimation. In *CVPR*, 2019. 5, 14

[54] Yunhan Zhao, Shu Kong, Daeyun Shin, and Charless Fowlkes. Domain decluttering: Simplifying images to mitigate synthetic-real domain shift and improve depth estimation. In *CVPR*, 2020. 2, 5, 7

[55] Chuanxia Zheng, Tat-Jen Cham, and Jianfei Cai. T2net: Synthetic-to-realistic translation for solving single-image depth estimation tasks. In *ECCV*, 2018. 5, 14

[56] Tinghui Zhou, Matthew Brown, Noah Snavely, and David G Lowe. Unsupervised learning of depth and ego-motion from video. In *CVPR*, 2017. 2

[57] Xiangxin Zhu, Dragomir Anguelov, and Deva Ramanan. Capturing long-tail distributions of object subcategories. In *CVPR*, 2014. 2

[58] Chuhang Zou, Alex Colburn, Qi Shan, and Derek Hoiem. Layoutnet: Reconstructing the 3d room layout from a single rgb image. In *CVPR*, 2018. 2

Appendices

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 A: 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 B: 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 C: 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 D: Further study of PDA augmentation scales.** We provide a thorough study between the depth predictor performance and PDA augmentation scales.
- **Section E: 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 F: 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 G: 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. 16 and 17.

A. 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 τ 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] to match the statistics of RGB images. We find this yields better performance than directly using $\mathbf{M}^{Clip}_{CPP}$. We visually compare some encoded maps in Fig. 10, 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 5.

Table 5: 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 δ^1 / δ^2	↓ better Abs^r/Sq^r/RMS-log	↑ better δ^1 / δ^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**

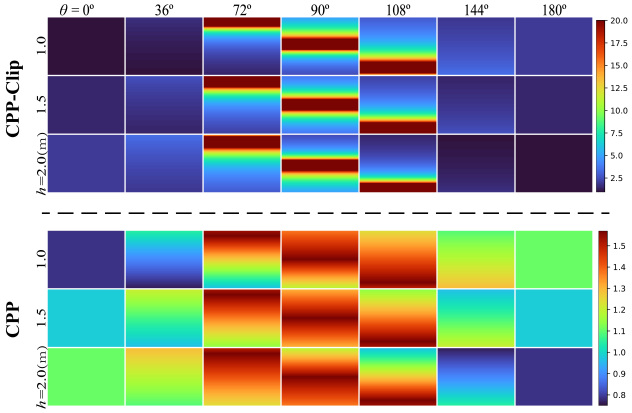

Figure 10: Visual comparisons of encoded maps of CPP and CPP-Clip with different pitch θ and camera height h. We set the threshold τ=20 for CPP-Clip. Encoded maps computed by CPP-Clip have the "red stripe" when the pitch is around 90° while CPP encoded maps have more smooth transitions when capturing the horizon.

B. 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

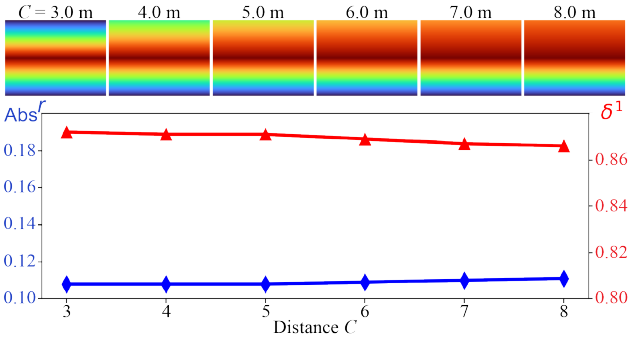

Figure 11: **Uppeer:** visualizations of CPP encoding with different hyper-parameter C (top); **bottom:** depth prediction performance as a function of hyper-parameter C. We train depth predictors on InteriorNet *Natural* train-set and test on its *Natural* test-set. From visual inspection, changing 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.

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 $C = [4, 5, 6, 7, 8]$. As shown in Fig. 11, 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.

C. 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 6, 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. 12 quantitatively. Together with the visualization of the prediction, we find that blind predictors make significantly more accurate depth prediction on floor regions, which confirms that the camera pose carries the prior knowledge about scene depth, especially on floor and ceiling regions.

D. 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. 13, the performance of

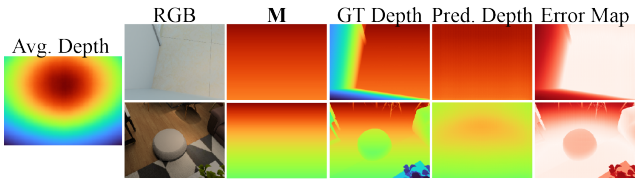

Figure 12: Illustration of how camera pose provides a strong depth prior through "blind depth prediction". Specifically, over the InteriorNet *Natural* train-set, we train a depth predictor solely on the CPP encoded maps $\mathbf{M}$ *without* RGB as input. For visual comparison, we compute an averaged 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 6: 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. 12. This further confirms that camera poses contain useful prior knowledge about scene depth.

Models	↓ better Abs-Rel/Sq-Rel/RMS-log	↑ better δ^1 / δ^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

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.

E. 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. 14, we find that encoding the true camera height (top plot) performs better than the true

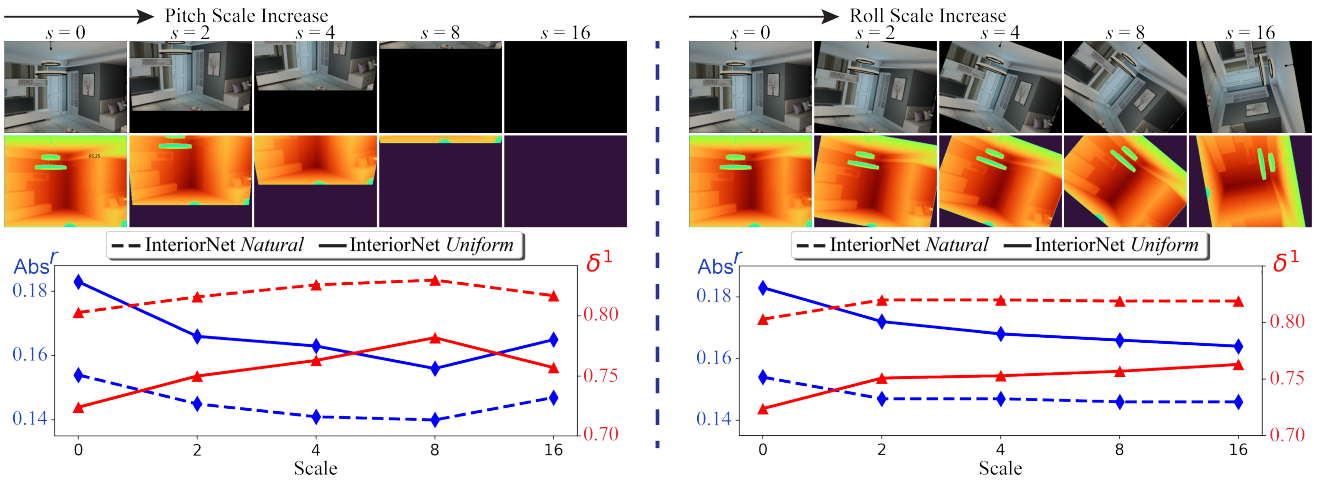

Figure 13: **Upper row:** visualizations of augmented examples using PDA with different scales. **Bottom row:** performance curves of depth predictor trained with PDA with different scales of pitch θ (left) and roll ω (right), respectively. Please refer to the main paper (Figure 9) for detailed descriptions. 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.

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).

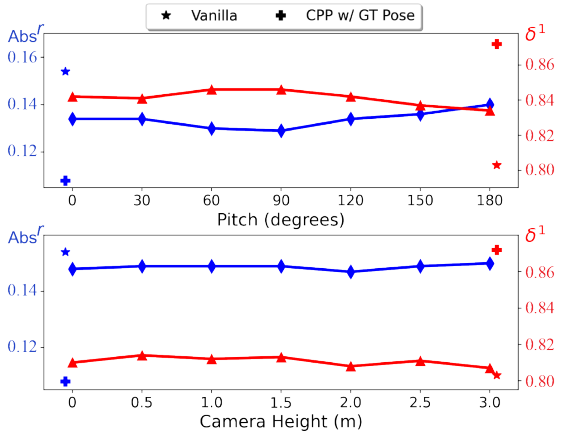

Figure 14: **Top:** CPP encoding with ground-truth camera height and fixed pitch. **Bottom:** CPP encoding with ground-truth pitch and fixed camera height. All models are trained/evaluated on InteriorNet *Natural* train/test-set. Comparing two blue or red curves across two plots, we find that encoding ground-truth height achieves better performance, suggesting height is "more important" than pitch. Moreover, encoding either ground-truth height or pitch outperforms Vanilla model.

F. 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 7). 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 7, CPP with predicted poses still outperforms Vanilla model; when jointly trained with PDA, CPP with predicted poses performs even better.

Table 7: **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 δ^1 / δ^2	↓ better Abs^r/Sq^r/RMS-log	↑ better δ^1 / δ^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**

G. Additional Details in Experiments

G.1. Image and Depth Preprocessing

All input RGB images are first normalized to the range of $[-1.0, 1.0]$ and then resized to 240×320 before feeding into CNNs. Note that resizing images to 240×320 does not change their original aspect ratios. For better training,

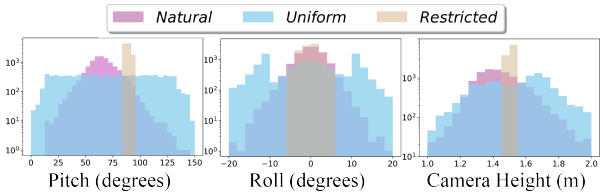

Figure 15: Distribution of pitch, roll and camera height for three subsets of images from ScanNet. From the *Natural* subset, we observe the ScanNet dataset also has a naturally biased distribution in both pitch, roll and camera height. Please refer to Section 5 in the main paper on how we construct these three subsets.

as a preprocessing step on the depth [4, 15], we apply the following operation to rescale depth maps y to get a normalized map y':

$$y' = (\frac{y - E_{min}}{E_{max} - E_{min}} - 0.5) * 2.0, \tag{9}$$

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 [55, 53]. 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. 9 to the predicted depth maps. Moreover, we also only evaluate the depth that lies in [1, 10] meters.

G.2. Pose Prediction Network

When camera poses are not available during testing, we train a camera pose predictor that predicts camera pitch θ, roll ω 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.

G.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.

G.4. ScanNet Camera Pose Distribution

The camera pose distribution of subsets in ScanNet is shown in Fig. 15. 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.

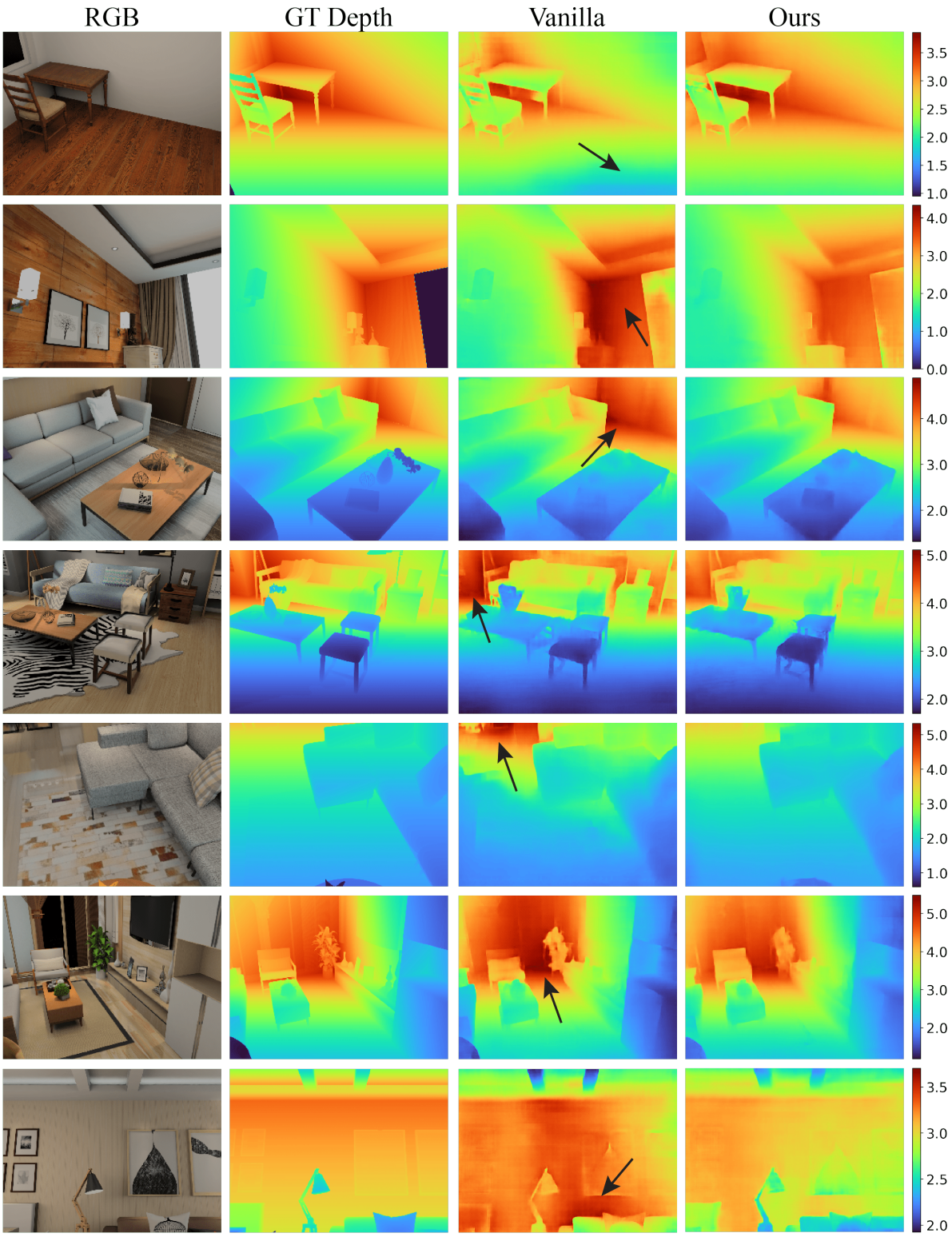

Figure 16: Depth predictions of Vanilla and our model (jointly applied CPP and PDA) on InteriorNet test-set. From these images captured under various camera poses, our model predicts better depth than Vanilla model in terms of the overall scale.

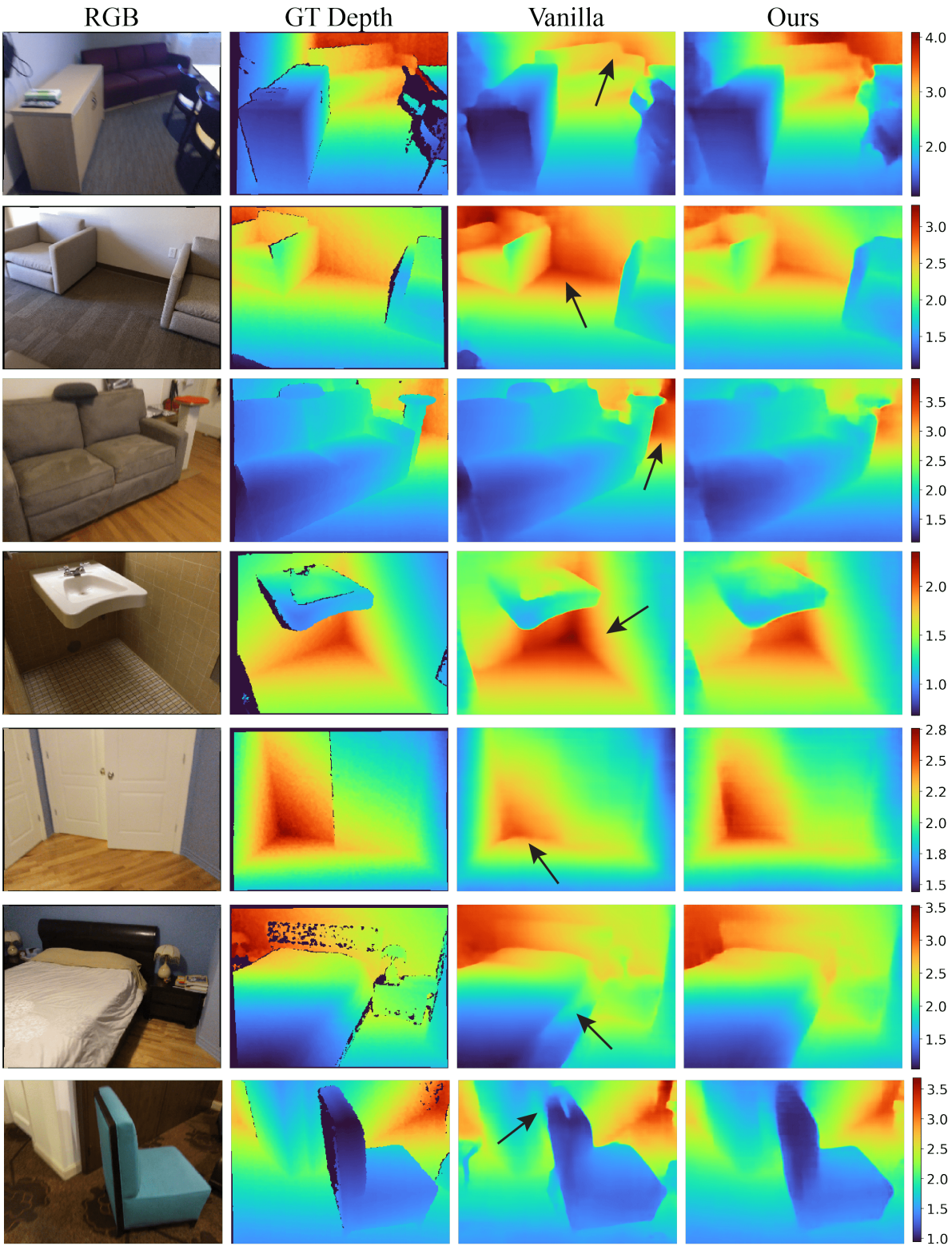

Figure 17: Depth predictions of Vanilla and our model (jointly applied CPP and PDA) on ScanNet test-set. From these images captured under various camera poses, our model predicts better depth than Vanilla model in terms of the overall scale.