Mitigating Perspective Distortion-induced Shape Ambiguity in Image Crops

Aditya Prakash, Arjun Gupta, and Saurabh Gupta

University of Illinois Urbana-Champaign
{adityap9,arjung2,saurabhg}@illinois.edu
https://bit.ly/AmbiguityEnc

Abstract. Objects undergo varying amounts of perspective distortion as they move across a camera's field of view. Models for predicting 3D from a single image often work with crops around the object of interest and ignore the location of the object in the camera's field of view. We note that ignoring this location information further exaggerates the inherent ambiguity in making 3D inferences from 2D images and can prevent models from even fitting to the training data. To mitigate this ambiguity, we propose Intrinsics-Aware Positional Encoding (KPE), which incorporates information about the location of crops in the image and camera intrinsics. Experiments on three popular 3D-from-a-single-image benchmarks: depth prediction on NYU, 3D object detection on KITTI & nuScenes, and predicting 3D shapes of articulated objects on ARCTIC, show the benefits of KPE.

Keywords: 3D from single image · perspective distortion · shape ambiguity

1 Introduction

Making metric predictions from a single image suffers from scale-depth ambiguity. Even when the camera intrinsics are known, it is impossible to disambiguate if Circle A shown in Fig. 1 has a radius of 0.485 cm (say) & is 2 cm away (say) or a radius of 4.85 cm and 20 cm away. Familiar size, or knowledge of the size of a reference object in the image, is a useful cue for resolving this ambiguity [10, 18].

The scale-depth ambiguity is a well-known and fundamental ambiguity. What we describe next is a subtle different ambiguity that occurs when we consider object crops from images for feeding into a neural network. Let's assume that we are using size familiarity to resolve scale-depth ambiguity, *i.e.* assume that the circles always have a radius of 0.485 cm. Given this information, it should be possible to infer how far they are in the image. This can be readily derived using similar triangles and the *location of the circle's image in the visual field*. In fact, this principle goes back many centuries and was known to Da Vinci [36].

However, if we consider crops around the object, say those output by an object detector, and try making a prediction for the circle's distance based on the appearance of the crop, this can prove to be quite difficult. Fig. 1 shows

fixed sized center crops around the center of the circle's image. It turns out, just looking at the crops while ignoring the location of the crop, it is no longer possible to predict the distance of the circle from the camera! A 0.5 cm wide image could correspond to Circle A that is 2 cm away directly in front of the camera, or Circle B that is further away (2.43 cm away) but off to the side. This simple example only exposed global pose error, but our detailed case study in Sec. 3 reveals that this ambiguity exists both in global pose and root-relative 3D shape. We call this *Perspective Distortion-induced Shape Ambiguity in Image Crops or PSAC*.

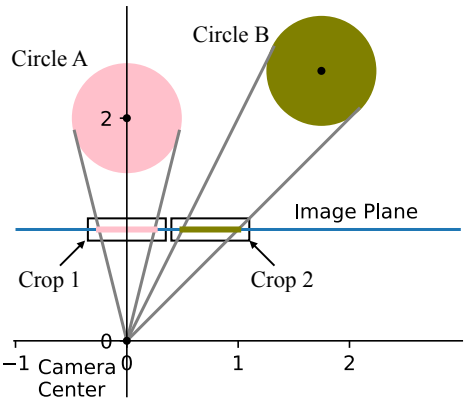


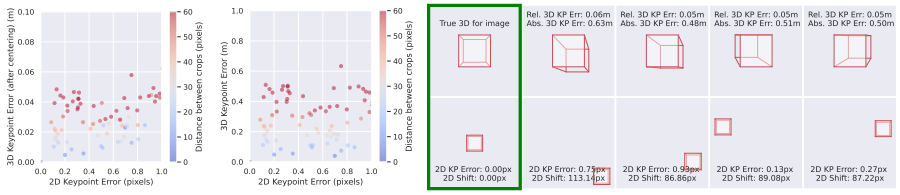
Fig. 1: Perspective Distortion-induced Shape Ambiguity. Consider two circles of the same size undergoing perspective projection under a pinhole camera. Even though they are at different distances from the camera, they appear to be the same size in the image due to perspective distortion. A model (e.g. a neural network) that predicts the distances of these circles from the camera based purely on the appearance of the image crops, *without taking into account their location in the camera's field of view*, will fail at this task. We call this the *Perspective Distortion-induced Shape Ambiguity in Image Crops* or *PSAC*. In this work, we propose an encoding to incorporate the crop location in the camera's field of view as input and show its effectiveness on metric depth prediction, 3D object detection & 3D pose estimation of articulated objects (Sec. 5).

To mitigate this ambiguity, we propose a intrinsics-aware positional encoding (KPE) that incorporates the location of the crop in the camera's field of view (Fig. 4). This allows the network to specialize its interpretation of 3D geometry based on the location in the image. We show results on three popular 3D-from-single-image benchmarks: articulated object pose estimation on ARCTIC [12], pixel-wise metric depth predictions on NYUD2 [32] and 3D object detection from monocular images on KITTI [13] and nuScenes [7]. Across these three real world benchmarks and a diagnostic setting, we find this additional information to improve metrics.

We acknowledge that similar encodings have been used in literature [11, 14]. However, their use was motivated by the need to train models across different cameras. Our contribution is to identify perspective distortion-induced shape ambiguity in image crops, even with a single camera, and show the effectiveness of encoding the location of the crop in the camera's field of view to mitigate this ambiguity.

2 Related Work

Ambiguity in 3D from a single image: Scale-depth ambiguity is well known and is commonly dealt with by only evaluating 3D predictions up to a scale [29] or



(a) Root-relative 3D (b) Absolute 3D (c) Example Ambiguous Parallelepipeds.

Fig. 2: Parallelepipeds Case Study. Figure (a) plots the root relative 3D keypoint error *vs*. 2D keypoint error of different parallelepipeds placed at different locations in the camera's field of view w.r.t. a reference cuboid shown in the green box in (a.2). Points are color coded with the distance between the parallelepipeds crops. As we let crops go farther away (the red points), we start finding parallelepipeds that have very different 3D shape but happen to project such that their 2D keypoints look the same as the reference cuboid. Figure (b) shows a similar plot but for absolute 3D keypoint error. Figure (c) shows these ambiguous 3D parallelepipeds in the top row and their renderings in the bottom row. The 1st figure in the green box is the reference w.r.t. which we measure 2D and 3D keypoint errors.

after a rigid alignment with the ground truth [21]. Where metric 3D predictions are needed, training is either done using domain specific knowledge [4], camera intrinsics as input [6] or predicted camera calibration parameters [20, 22], *e.g.* focal length [22] using appearance cues [20]. Notably, due to this ambiguity, models can not trivially be trained with scale and crop augmentations unless some adjustments are made [6, 9]. Ambiguity induced due to crops (the focus of our work) has not been analyzed in literature to the best of our knowledge. Note that this ambiguity is an artifact of *planar* perspective projection in images and doesn't exist in *spherical* perspective projection. The relationship between planar and spherical perspective have been used to simplify analysis, *e.g.* [26].

Geometric embeddings: Past works have considered the use of different encodings, based on pixel locations [8, 25], camera intrinsics [11] or extrinsics [38, 40], as input to networks to improve performance. Liu *et al.* [25] show that preserving spatial location of the pixels helps in generative modelling, object detection, and reinforcement learning. Different forms of positional encodings, *e.g.* learnable [8, 23], cartesian spatial grid [19], sinusoidal [27, 35], implicit [31, 37], help incorporate spatial inductive biases in GANs [37]. Positional encodings, *e.g.* learned [8], relative [34], or rotational [33], are also commonly use in training transformer [35]-based models. Recent works [11, 14] have also studied the effectiveness of pixel-level encodings in multi-dataset setting (with varying cameras) on depth estimation tasks. Researchers have also considered canonicalizing the inputs (captured with different cameras) to a common camera intrinsics [1, 6] to indirectly provide camera information. [15, 28, 38] employ this idea in multi-view settings. Yifan *et al.* [38] used such encodings to inject geometric inductive biases (*e.g.* the epipolar geometry constraint) into a general perception model for multi-view stereo. Guizilini *et al.* [15] extend this further and use per-pixel

embeddings to capture the image coordinate and camera intrinsics & extrinsics for multi-view depth estimation. Miyato *et al.* [28] propose a geometry-aware attention mechanism for transformers for novel view synthesis tasks.

Our use of the location of the crop in the camera's field of view as an additional input to neural networks is motivated by these past works. Orthogonal to these works, our contribution is to demonstrate the effectiveness of this encoding in mitigating the perspective distortion-induced shape ambiguity in image crops. We conduct this study across 3 representative 3D from a single image tasks, and where appropriate, show comparisons to past embeddings: Cam-Conv [11] and sinusoidal image location embeddings [35].

3 Parallelepipeds Case Study

We first analyze simple parallelepipeds to intuitively understand the perspective distortion-induced ambiguity in image crops and the effectiveness of the crop location in the camera's field of view to mitigate this ambiguity. It is important to note that this ambiguity does not always exist. If the shape and size of the object are perfectly known (say we know that it is a front-facing cube), then the location of the cube in the visual field can itself be deduced from an image crop.

However, if the geometry of the object is also unknown, then it may no longer be possible to disentangle shape & pose from a given image crop. Consider the family of 3D shapes with a square face (of width 0.2m lying in XY plane) that is extruded along an arbitrary direction, *i.e.* parallelepipeds with atleast one face being a square. Fig. 2 (c, top row) shows different parallelepipeds when placed directly in front of the camera (*i.e.* from one consistent viewing angle across the row). Note that the parallelepipeds here do not extend purely in the Z-direction (except for the leftmost reference in green box), but can extend at an oblique angle. Fig. 2 (c bottom row) shows these same parallelepipeds when placed at different locations in the visual field. These locations were mined separately for each so as to maximize the visual similarity to the leftmost reference parallelepiped. As evident, even though the parallelepipeds across the row have different shapes, they all look the same in the 2D image (with a sub-pixel reprojection error), albeit for their 2D locations.

This ambiguity only exists if we only look at the image crops locally without factoring in the absolute 2D location of the crop. Let's treat the parallelepiped in the first column (a cuboid) as reference. Fig. 2 (a) plots the root relative 3D shape error as a function of the root relative 2D keypoint error of different parallelepipeds w.r.t. the reference cuboid. For root relative error, we compute the keypoint error *after* the centers of the parallelepipeds have been aligned. Points are color coded based on the 2D distance of the image crops from the cuboid crop (blue means image crops are closer, red means they are farther away). If we limit reasoning to only close-by crops, *i.e.* the blue points, low 2D keypoint error implies low 3D shape error. However, if we look at the farther away points (the red points), image with a low relative 2D keypoint error can actually have a high 3D shape error. This is the inherent ambiguity: even if two shapes appear

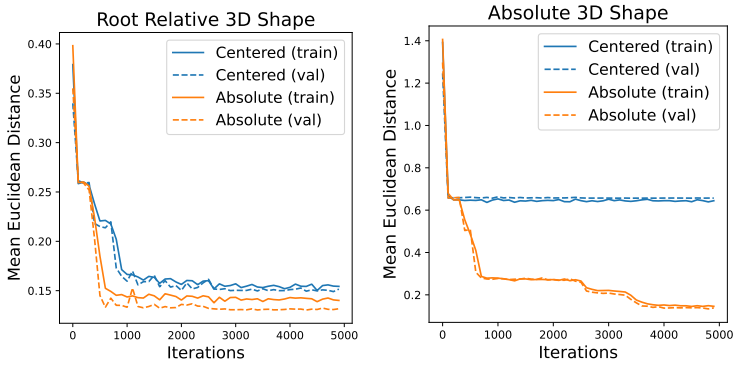


Fig. 3: Predicting root relative (left) or absolute (right) 3D shape from 2D image crops fails in the absence of information about the location of the crop in camera's field of view. Training loss saturates at a high value because of the inherent ambiguity. Adding information about the location of the crop in camera's field of view alleviates this ambiguity, leading to better metrics for both root-relative and absolute 3D prediction.

similar visually, their 3D shapes may be vastly different. Fig. 2 (b) presents the analogous plot where we measure absolute 3D error instead of root-relative error.

Let's see what happens when we train neural networks to predict 3D shape from the 2D keypoints. Given the 2D coordinates for the 8 corners, the goal is to predict the root-relative (or absolute) 3D location of the 8 corners. We consider two variants. First, we center the eight 2D corners by subtracting their mean to get centered crops. Next, we retain the absolute value of the 2D coordinates. We train a 6-layer MLP with these two inputs. Fig. 3 shows the training and validation plots for predicting root relative and absolute 3D coordinates for the 8 corners. As expected the first version that only uses the crops converges to a higher error even on the training set as it is impossible to infer the 3D shape from centered 2D corners due to the ambiguity. The second version that has access to the absolute coordinates does not suffer from this ambiguity and is able to successfully estimate the 3D shape on both training and validation data.

4 Intrinsics-Aware Positional Encoding (KPE)

We mitigate the ambiguity induced by perspective distortion (Fig. 1) through design of encodings that capture the location of the crop in the camera's field of view, referred to as Intrinsics-Aware Positional Encodings (KPE). Specifically, let's denote the principal point for the original image by (p_x, p_y) and focal length by (f_x, f_y). For any pixel (x, y), we can describe its position in the camera's field of view via $\theta_x = \tan^{-1}\left(\frac{x-p_x}{f_x}\right)$ and $\theta_y = \tan^{-1}\left(\frac{y-p_y}{f_y}\right)$, as shown in Figure 4 (a). Our proposal is to retain this information for pixels when images are cropped or resized for feeding into neural networks.

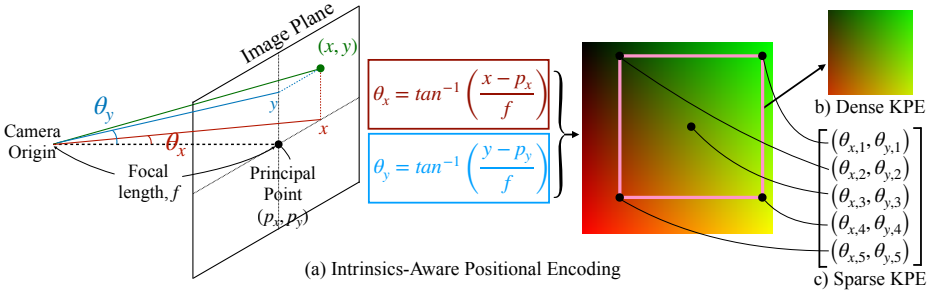


Fig. 4: Intrinsics-Aware Positional Encodings (KPE). **(a)** For each pixel in the image, we compute its position in the camera's field of view (θ_x and θ_y), or the angular distance that the pixel makes with respect to the principal point and the camera origin (as shown on the left). Note that both θ_x and θ_y are sensitive to the camera intrinsic parameters. **(b)** For a dense prediction task, we make use of a dense prediction encoding which contains the positional encoding for each pixel in the region of interest. **(c)** For other tasks, we simply represent the positions of the corners of the relevant region of interest in addition to the center point. The positional encoding can be passed into the network at the input level or concatenated to some intermediate representation, this design choice is made separately for each task.

4.1 Using KPE

KPE is quite flexible and can be used in different ways. We specifically consider two variations: Dense KPE (Figure 4 b) and Sparse KPE (Figure 4 c).

Dense KPE: We compute θ_x and θ_y for each per-pixel in a dense manner. Rather than using the raw angles, we encode them via sinusoidal encodings (4 frequencies each) [35]. The spatial feature map can be input into the network along side the raw pixels. Note that dense KPE can be computed at different resolutions and can be concatenated with a spatial feature maps at different points in the network.

Sparse KPE: We compute θ_x and θ_y for the 4 corners and the center of the crop. As for dense KPE, we encode them via sinusoids. Resulting embeddings can be concatenated and fed into MLP layers (along with flattened features for crop) or can be concatenated with spatial feature maps via broadcasting.

We experiment with sparse and dense KPE for the 3 representative tasks and find that dense KPE works best for dense tasks, while sparse KPE works best for tasks that involve summarizing the geometric properties for objects, *e.g.* predicting its 3D bounding box or 3D pose.

5 Experiments

We consider 3 challenging settings, involving the prediction of 3D geometric attributes from a single image, to show the effectiveness of KPE: 3D pose estimation of articulated objects in contact (Section 5.1), dense depth prediction(Section 5.2) and 3D object detection (Section 5.3). Since our goal is to show the utility of KPE

(not to achieve state-of-the-art results) we select some representative models from recent works on each of these tasks and incorporate KPE into their architectures. These tasks encompass sparse and dense prediction of 3D attributes, and the representative methods cover both convolutional and transformer architectures. Thus, these together comprehensively test KPE. In each of the following subsections, we describe the application (the representative method and relevant metrics), how we apply KPE, and the results we obtain. We summarize the key takeaways in Section 6.

5.1 Application 1: 3D Pose of Articulated Objects in Contact [12]

Task: Given an egocentric image showing an articulated object in interaction, the goal is to estimate the 3D pose of the object in contact. Specifically, the 3D pose is represented using rotation in the camera coordinate frame, the translation of the root of the object with respect to the camera center and the angle of articulation between the bottom and top components of the articulated object. The object mesh is assumed to be available. We also consider contact estimation and 4D motion reconstruction to show the versatility of KPE.
Method: We adopt the single-image ArcticNet-SF [12] architecture released with the ARCTIC benchmark [12]. The network takes a single image as input, which is processed by a ResNet50 [17] backbone to get a feature map of resolution $7 \times 7 \times 2048$. These features are then average pooled and passed to a HMR [21]-style decoder to predict the 7-dimensional pose, consisting of rotation, translation & articulated angle of the object. The network outputs rotation in axis-angle format and uses the weak perspective camera model to estimate the translation [5, 21, 24, 39]. The entire network is trained end-to-end using an L2 loss between the predictions and the ground truth for 7-dimensional pose, 3D keypoints and 2D projection of 3D keypoints.
Modifications: We modify the architecture to also take crops around the object as input, which are processed by a ResNet50 backbone [16]. Since our network requires object crops, we also train a bounding box predictor model by fine-tuning MaskRCNN [16] on the ARCTIC training set using the ground truth bounding box computed from the 2D projections of 3D object keypoints. This is required to evaluate our model on the online leaderboard for which ground truth is not available. For contact prediction, we use the hand pose from the default ArcticNet-SF decoder.
Incorporating KPE: As described in Sec. 4, we explore both sparse (captures only the center and corner pixels of the object crop) and dense (encode each pixel in the crop). We consider 3 choices for where to add KPE in the model: (1) concatenate with the input, (2) process the encoding with a MLP & add with the average pooled global features, (3) interpolate the encoding to 7×7 resolution & concatenate with $7 \times 7 \times 2048$ feature map from the last convolutional layer of ResNet50. This concatenated feature is then processed by three convolutional layers and flattened to 2048 dimensions before being passed to the decoder. We do not use batchnorm in these 3 convolutional layers.

Table 1: 3D pose estimation of articulated objects in contact – Main Result. We compare with ArcticNet-SF [12] on multiple geometric tasks: 3D pose, contact and 4D motion estimation for articulated objects on ARCTIC [12]. Our intrinsics-aware positional encoding leads to significant improvements across all metrics and compares favorably to other ways of encoding the location of the crop and past encodings [11].

Method	Object		Contact		Motion	
	AAE (°) ↓	Success (%) ↑	CDev_{ho} (mm) ↓	MRRPE_{ro} (mm) ↓	MDev_{ho} (mm) ↓	Acc (m/s^2) ↓
a) Validation Split (KPE helps w.r.t. other choices)						
ArcticNet-SF (full image) [12]	8.0	59.0	44.1	36.8	11.8	11.3
ArcticNet-SF (crop)	7.0	66.0	74.3	63.0	19.3	10.9
Full Image with Binary Mask	6.8	66.8	75.6	60.9	17.8	10.8
Cam-Conv [11]	6.3	68.8	38.7	**29.7**	10.1	9.4
Image Location (dense)	6.1	69.2	40.4	31.6	10.1	9.2
Image Location (sparse)	6.2	67.9	**39.1**	30.0	10.1	9.3
KPE (sparse)	**5.9**	**71.5**	39.4	**29.7**	**9.3**	**8.7**
b) Test Split						
ArcticNet-SF (full image) [12]	6.4	53.9	44.7	36.2	11.8	9.1
KPE (sparse)	**5.2** -19%	**64.1** +19%	**36.5** -18%	**28.3** -22%	**10.5** -11%	**7.6** -16.4%

Dataset: ARCTIC [12] is a recent dataset consisting of hands interacting with articulated objects in a free-form manner. There are 2 settings in the dataset: 1.5M frames from 8 allocentric views & 200K frames from 1 egocentric view. We show results in egocentric setting since the perspective distortion is more prominent due to objects being closer to the camera. The released dataset does not contain ground truth for test set, which is evaluated separately on the online leaderboard. We compare with ArcticNet-SF in this setting.

Metrics: We follow the evaluation protocol of the ARCTIC benchmark [12] and report the following metrics: a) **AAE**: Average Absolute Error between the ground truth degree of articulation and the prediction; b) **Success**: percentage of predicted object vertices with L2 error $< 5\%$ of the diameter of the object; c) **CDev**: Contact Deviation is average distance between the predicted in-contact hand-object vertices and ground truth; d) **MRRPE**: Mean Relative-Root Position Error between the predicted root keypoint location and the ground truth; e) **MDev**: Between consecutive frames, Motion Deviation measures the disagreement in the direction of the movement of in-contact hand-object vertices; f) **ACC**: Acceleration error is the difference in the predicted acceleration of the vertices and the ground truth.

Results: Adding KPE to ArcticNet-SF leads to consistent improvements across the 3D pose, contact and 4D motion estimation metrics across both validation (Tab. 1 a) and test sets (Tab. 1 b). Using full images as input, as done in [12], obtains worse performance due to low resolution on the object. Focusing on the object, either through masking or by inputting only a crop, helps. KPE allows focusing on the object while also retaining the location of the object in the visual field, leading to the best performance. KPE made 2 design choices: a) use of sinusoids to encode, and b) use of crop's location in the camera field of view

Table 2: 3D pose estimation of articulated objects [12] – KPE Design Choices. See Sec. 5.1 for details.

Method	Object		Contact		Motion	
	AAE (°) ↓	Success (%) ↑	CDev_{ho} (mm) ↓	MRRPE_{ro} (mm) ↓	MDev_{ho} (mm) ↓	Acc (m/s^2) ↓
a) Validation Split (Sparse KPE design choices)						
KPE (sparse)	**5.9**	**71.5**	39.4	**29.7**	**9.3**	**8.7**
KPE (sparse center only)	7.2	66.8	44.5	33.1	10.9	10.2
KPE (sparse w/ glb feat)	6.2	69.8	**37.2**	30.6	9.7	9.2
b) Validation Split (Dense KPE design choices)						
KPE (dense)	**6.2**	**71.4**	**37.7**	**29.3**	**9.9**	**9.4**
KPE (dense w/ input)	7.3	68.1	48.7	38.3	13.6	10.2
c) Validation Split (Sparse vs. *Dense)*						
KPE (sparse)	**5.9**	**71.5**	**39.4**	**29.7**	**9.3**	**8.7**
KPE (dense)	6.2	71.4	37.7	29.3	9.9	9.4

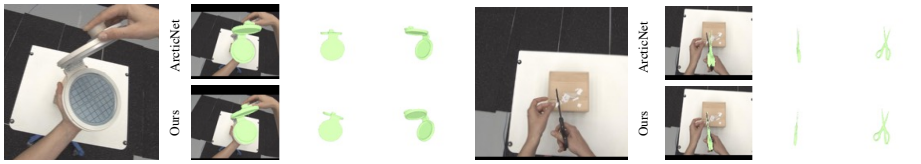


Fig. 5: 3D pose visualizations on ARCTIC. Our proposed modification of intrinsics-aware positional encoding (KPE) improves over the ArcticNet-SF [12] model by predicting better 3D poses in interaction scenarios (note the difference in the articulation angle and global pose). For each image, we show the projection of the object mesh with the predicted pose on the image and from 2 different camera views.

rather than the 2D image. Both aspects are important and KPE is better than Cam-Conv [11] that represents locations in camera's field of view but without sinusoids, and also just using sinusoidal embeddings of the 2D location in image.

We also ablate design choices for where to inject sparse and dense KPE in Table 2 b and Table 2 b. It is beneficial to inject sparse KPE at earlier layers rather than at the very end with global features. We also find that encoding both the scale and location of the crop is better than just encoding the location. For dense KPE, inputting it alongside the input is worse than using it alongside the 7×7 spatial feature map coming out from the ResNet50 backbne.

While both sparse and dense versions of KPE improve performance, the sparse one being slightly better (Table 2 c).

For each image, we visualize (in Fig. 5) the projection of the object mesh (assumed to be available) in the image using the predicted 7-dimensional pose (global rotation, camera translation & articulation angle) along with renderings of the shape from two additional views. From the visualizations, we observe that KPE leads to better angle of articulation and global rotation than ArcticNet-SF.

Table 3: Dense metric depth prediction on NYU (Main Result). KPE leads to improved depth estimation accuracy when trained and tested with cropping and scaling augmentations. KPE outperforms Cam-Conv [11], a past encoding specifically design for depth estimation. We see gains in both settings with and without pre-training of the MiDaS backbone.

Method	REL ↓	RMSE ↓	$\log_{10}$ ↓
a) No Pre-training for MiDaS backbone (KPE helps and is better than other choices)			
ZoeD-X-N	0.093	0.360	0.042
ZoeD-X-N + Cam-Conv [11]	0.098	0.382	0.042
ZoeD-X-N + KPE (dense)	**0.088** -5.4%	**0.334** -7.2%	**0.038** -9.5%
b) MiDaS backbone pre-trained on 12 datasets (KPE helps and is better than other choices)			
ZoeD-M12-N	0.088	0.337	0.040
ZoeD-M12-N + Cam-Conv [11]	0.096	0.376	0.041
ZoeD-M12-N + KPE (dense)	**0.082** -6.8%	**0.313** -7.1%	**0.035** -12.5%

Table 4: Dense metric depth prediction on NYU (KPE Design Choices). We conduct these comparisons in the setting where the MiDaS backbone is pre-trained on 12 datasets. See Section 5.2 for more details.

Method	REL ↓	RMSE ↓	$\log_{10}$ ↓
a) Dense design choices			
ZoeD-M12-N + KPE (dense, inside MiDaS backbone)	**0.082**	**0.313**	**0.035**
ZoeD-M12-N + KPE (dense, after MiDaS backbone)	0.085	0.323	0.038
b) Sparse design choices			
ZoeD-M12-N + KPE (sparse, inside MiDaS backbone)	**0.083**	**0.321**	**0.037**
ZoeD-M12-N + KPE (sparse, after MiDaS backbone)	0.087	0.331	0.038
c) Sparse vs. *Dense*			
ZoeD-M12-N + KPE (dense)	**0.082**	**0.313**	**0.035**
ZoeD-M12-N + KPE (sparse)	0.083	0.321	0.037

5.2 Application 2: Dense Metric Depth Prediction [4, 32]

Task: Given a single RGB image, the goal is to predict the distance of each pixel to the camera center along the optical axis.

Method: We adopt the ZoeDepth architecture [4]. It consists of a transformer-based MiDaS [29] backbone to estimate relative depth, which is passed to a domain specific decoder along with multi-scale image features from the backbone, to predict the metric depth for a specific dataset. The model uses a discretized representation of 3D space (into a volumetric grid, referred to as bins) to first output coarse depth (represented as the center of the bin), which is then refined by an MLP in subsequent layers (referred to as the MetricBins [4] module). The model is trained using the scale-invariant log loss (similar to [3]) with 384×512 images without any cropping and scaling augmentations during training. There are two variants of the model: a) ZoeD-X-N, trained from scratch and b) ZoeD-M12-N, initialized from a MiDAS model pre-trained on 12 datasets (also on 384×512 images).

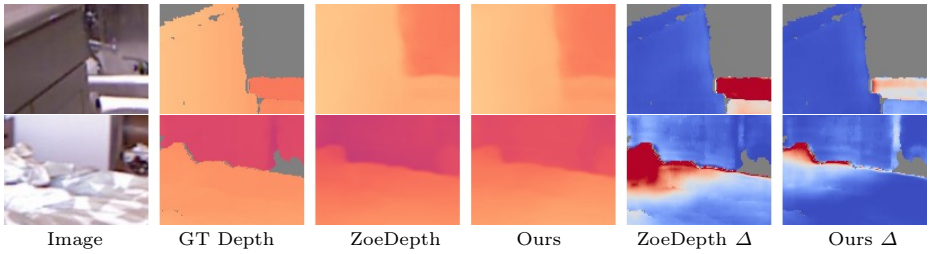


Fig. 6: Depth visualizations on NYU. We compare the depth predicted by adding KPE encoding to ZoeDepth [4] with the base ZoeDepth model. We show the depth predictions along with the squared error Δ w.r.t. to ground truth depth (ranging from dark blue: low to dark red: high, invalid regions are indicated as grey). Our model predicts better depth as evident by lower Δ (lower intensity red areas).

Modifications: We train and test on 3:4 aspect ratio image crops that are rescaled to 384×512 before being fed into the network. We do not change the architecture or the loss function.

Incorporating KPE: We experiment with both sparse and dense KPE. We explore two design choices for where to add KPE: (1) with the relative depth & $12 \times 16 \times 256$ image features predicted by the MiDaS model, before passing it to the MetricBins module, (2) with the downsampled $24 \times 32 \times 1024$ feature map before being processed by the BeiT [2] module in MiDaS. We interpolate KPE to a resolution of 12×16 for the former and 24×32 for the latter, in case of the dense variant. For the sparse variant, we broadcast it to the required resolution.

Dataset: We use NYU Depth v2 [32], a popular indoor scene dataset, containing over 450+ unique scenes. We consider the same splits as [4]: 23K images for training, 654 images for testing and compare against ZoeDepth [4].

Metrics: Following [4], we compute 3 metrics between the predicted metric depth ($\hat{d}_i$) and the ground truth (d_i): (1) **REL**: absolute relative error, $\frac{1}{M}\Sigma_{i=1}^{M}|d_i - \hat{d}_i|/d_i$, (2) **RMSE**: root mean squared error, $[\frac{1}{M}\Sigma_{i=1}^{M}|d_i - \hat{d}_i|^2]^{\frac{1}{2}}$, (3) $\log_{10}$: average $\log_{10}$ error, $\frac{1}{M}\Sigma_{i=1}^{M}|\log_{10} d_i - \log_{10} \hat{d}_i|$.

Results: Across both settings (pre-training and no pre-training), KPE leads tom improved performance over not using it, leading to relative improvements of 5.4% – 12.5% across the different metrics and settings (Table 3 a and b). We also compare to Cam-Conv [11], a past encoding that has been used for single image depth estimation. While Cam-Conv also encodes the location of pixels in the camera's field of view, it doesn't encode them using sinusoidal embeddings that are used in KPE, and leads to worse performance.

In terms of design choices, we see more gains when adding KPE to the downsampled image features before it is passed to the BeiT [2] module in the MiDaS backbone than after the MiDaS backbone (Table 4 a and b). Early infusion allows the transformer to incorporate information about the crop location in the camera's field of view in the multi-scale feature output. From Tab. 4 c, we find

Table 5: 3D Object Detection (Main Results). We add KPE to the Cube R-CNN [6] model and compare the AP scores at different IoUs. We see consistent improvements across all metrics. KPE helps and is better than other encodings. Green numbers denote relative improvements over not using any encodings.

Method	AP^{25}_{3D} ↑	AP^{50}_{3D} ↑	AP^{75}_{3D} ↑	AP_{3D} ↑
Cube R-CNN	70.17	43.83	14.23	54.59
Cube R-CNN + Image location (dense)	71.97	45.03	15.16	56.86
Cube R-CNN + Image location (sparse)	72.08	45.93	15.32	56.30
Cube R-CNN + Cam-Conv [11]	72.62	46.51	14.59	56.82
Cube R-CNN + KPE (sparse)	**73.10** +4.2%	**47.51** +8.4%	**16.58** +16%	**57.54** +5.4%

Table 6: 3D Object Detection (KPE Design Choices). See Section 5.3 for details.

Method	AP^{25}_{3D} ↑	AP^{50}_{3D} ↑	AP^{75}_{3D} ↑	AP_{3D} ↑
Sparse vs. *Dense KPE*				
Cube R-CNN + KPE (dense)	72.74	46.75	15.23	56.78
Cube R-CNN + KPE (sparse)	**73.10**	**47.51**	**16.58**	**57.54**

the dense variant of the encoding to work better than the sparse variant. This is likely due to the dense prediction nature of this task.

In Fig. 6, we visualize the predicted depth along with the squared error Δ w.r.t. to ground truth (ranging from dark blue: low to dark red: high). Our model predicts better depth as evident by lower Δ (lower intensity red areas). More visualizations are provided in supplementary.

5.3 Application 3: 3D Object Detection on KITTI, nuScenes [6, 7, 13]

Task: The goal is to predict 3D bounding boxes for objects, given a single image.
Method: We use the recent Cube R-CNN [6] model since it provides a generic framework for 3D detection, without any domain or object specific strategies. It consists of a convolutional backbone to extract features from an image, followed by a region proposal network [30] to predict regions of interest (RoIs) (represented as 2D box proposals) used for detecting objects. The features from each detected object are passed to a decoder to predict a 3D bounding box. The model is trained using 3D ground truth supervision for the 3D bounding box on multiple datasets jointly. Since each of these datasets are collected using different cameras, the authors propose to augment camera information with the ground truth depth used for training. This is referred to as virtual depth [6] and is orthogonal to our proposed KPE encoding. Please refer to [6] for more details.
Modifications: We use the same training protocol as [6].
Incorporating KPE: The region proposal network in the backbone predicts 2D box proposals which are used for detecting objects. The positive box proposals with detected objects are then passed to the cube head to regress the 3D bounding box. We incorporate KPE along with the features of the 2D box proposals before feeding to the cube head for 3D regression. Similar to Sec. 5.1 and Sec. 5.2, we

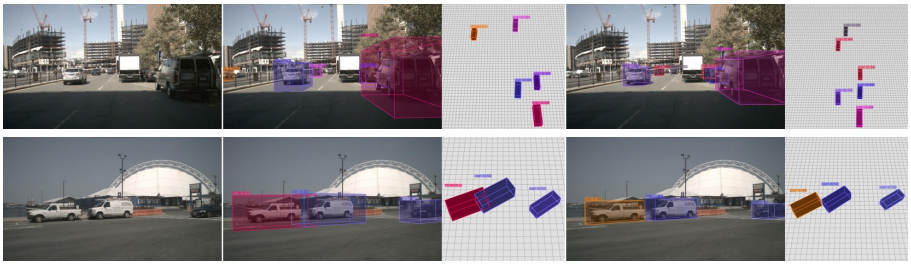


Fig. 7: 3D Object Detection on nuScenes. We show the 3D bounding box predictions on Cars category for Cube R-CNN [6] and our model (Cube R-CNN + KPE), both in image space and in top-down view. Our model performs better, as evident by fewer collisions (*i.e.* intersections between car bounding boxes) and missed detections. More visualizations on KITTI and nuScenes in supplementary.

explore both the sparse and the dense variant of our encoding. For the sparse variant, we broadcast the features to match the feature resolution whereas for the dense variant, we interpolate the features to the desired feature resolution.

Dataset: We experiment on two commonly used datasets in 3D detection: KITTI [13] and nuScenes [7]. These datasets capture urban outdoor scenes with a wide field of view, a setting in which perspective distortion is more prominent. We use the same splits as [6]: (1) KITTI: 2.8K training images & 329 validation images, (2) nuScenes: 18K training & 1.1K validation images, and focus on the Cars category. We consider the multi dataset setting where a single model is trained and tested on KITTI and nuScenes. Since KPE contains camera information, we hypothesize that it should be effective in joint dataset training as well. In the joint training on KITTI and nuScenes, we compare with the Cube-RCNN variant that does not use virtual depth to disentangle the impact of virtual depth in multi-camera setting.

Metrics: Following prior work [6], we utilize 3D average-precision (AP_{3D}) as our metric to evaluate the 3D object detection models. The predicted 3D cuboid is matched to the ground truth 3D cuboid by computing a 3D intersection-over-union (IoU). We also report the AP_{3D} over a range of different IoU thresholds.

Results: We observe consistent improvements in AP at different IoU thresholds Tab. 5 with 4.2% – 16% relative improvement over not using any encoding.

Table 6 compares KPE's encoding of box location in the camera field of view to just encoding the box location in the 2D image. We find KPE's encoding of the box's location in the camera's field of view to be more effective. KPE is also more effective than Cam-Conv [11]. Sparse KPE works better than dense KPE.

We also visualize the 3D bounding box predictions on Cars category for Cube R-CNN [6] & our model (Cube R-CNN + KPE) in Fig. 7, both in image space & in top-down view. Our model performs better, as evident by fewer collisions (*i.e.* intersections between car bounding boxes) and missed detections. More visualizations on KITTI & nuScenes are in supplementary.

6 Discussion

Our experiments provide several insights on three representative 3D from a single image tasks: 3D pose estimation of articulated objects in contact, dense metric depth prediction, 3D object detection, spanning dense & sparse predictions and convolutional & transformer architectures. Specifically, across all settings, we found the use of KPE, sparse or dense, to improve performance. For applications involving sparse output (such as 3D pose for objects), sparse KPE worked better; while for applications involving dense output (such as depth prediction), dense KPE worked better. KPE is quite flexible and can be injected in the network at different layers. The best location to inject KPE is problem and method dependent. KPE has two important aspects: (a) use of sinusoids to encode, and (b) use of crop's location in the camera field of view rather than the 2D image. Both aspects lead to KPE being better than Cam-Conv [11] that represents locations in camera's field of view but without sinusoids, and uses sinusoidal encoding of the 2D location in image. At the same time, KPE outperforms task specific prior encodings: Cam-Conv [11] for depth prediction and 2D image location for 3D object detection. Lastly ablations on the task of 3D object pose estimation (Sec. 5.1) reveal that encoding both scale and location of the crop is better than just encoding the location and KPE is more effective than other ways of denoting the location of the object, *e.g.* via a masked image.

Limitations: Our analysis only considers geometry of projection. It will be interesting to assess if shading or other cues help alleviate the ambiguity. KPE requires camera intrinsics which are typically present in image metadata but may not always be available. It would be useful to investigate if KPE could be used with predicted intrinsics [20] in such cases. Jointly estimating instrinsics using appearance cues [20] in the image crop (*e.g.* by adding a loss on the intrinsics) could also help with 3D predictions. Other future directions include analyzing the impact of KPE on shallower networks, smaller receptive fields, predicting normals or full 3D shape from a single image.

7 Conclusion

We explore the ambiguity induced by perspective distortion in image crops. We provide an intuitive understanding of this ambiguity using parallelepipeds and propose an intrinsics-aware positional encoding (KPE) that incorporates the location of the image crop in the field of the view of the camera. Experimental evaluation across three 3D from a single image tasks (3D pose estimation of articulated objects in contact, dense metric depth prediction and 3D object detection) reveals the effectiveness of KPE.

Acknowledgements: We thank Matthew Chang, Shaowei Liu, Anand Bhattad and Kashyap Chitta for feedback on the draft, and David Forsyth for useful discussion. This material is based upon work supported by an NSF CAREER Award (IIS2143873), NASA (80NSSC21K1030), an NVIDIA Academic Hardware Grant, and the NCSA Delta System (supported by NSF OCI 2005572 and the State of Illinois).

References

1. Antequera, M.L., Gargallo, P., Hofinger, M., Bulo, S.R., Kuang, Y., Kontschieder, P.: Mapillary planet-scale depth dataset. In: Proceedings of the European Conference on Computer Vision (ECCV) (2020)
2. Bao, H., Dong, L., Piao, S., Wei, F.: BEiT: BERT pre-training of image transformers. In: Proceedings of the International Conference on Learning Representations (ICLR) (2022), https://openreview.net/forum?id=p-BhZSz59o4
3. Bhat, S.F., Alhashim, I., Wonka, P.: Localbins: Improving depth estimation by learning local distributions. In: Avidan, S., Brostow, G.J., Cissé, M., Farinella, G.M., Hassner, T. (eds.) Proceedings of the European Conference on Computer Vision (ECCV) (2022)
4. Bhat, S.F., Birkl, R., Wofk, D., Wonka, P., Müller, M.: Zoedepth: Zero-shot transfer by combining relative and metric depth. arXiv (2023)
5. Boukhayma, A., de Bem, R.A., Torr, P.H.S.: 3d hand shape and pose from images in the wild. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019)
6. Brazil, G., Kumar, A., Straub, J., Ravi, N., Johnson, J., Gkioxari, G.: Omni3d: A large benchmark and model for 3d object detection in the wild. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 13154–13164 (2023)
7. Caesar, H., Bankiti, V., Lang, A.H., Vora, S., Liong, V.E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., Beijbom, O.: nuscenes: A multimodal dataset for autonomous driving. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2020)
8. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. In: Proceedings of the International Conference on Learning Representations (ICLR) (2021)
9. Eigen, D., Puhrsch, C., Fergus, R.: Depth map prediction from a single image using a multi-scale deep network. Advances in Neural Information Processing Systems (NeurIPS) **27** (2014)
10. Epstein, W.: The known-size-apparent-distance hypothesis. The American journal of psychology **74**(3), 333–346 (1961)
11. Facil, J.M., Ummenhofer, B., Zhou, H., Montesano, L., Brox, T., Civera, J.: Cam-convs: Camera-aware multi-scale convolutions for single-view depth. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 11826–11835 (2019)
12. Fan, Z., Taheri, O., Tzionas, D., Kocabas, M., Kaufmann, M., Black, M.J., Hilliges, O.: ARCTIC: A dataset for dexterous bimanual hand-object manipulation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2023)
13. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? the kitti vision benchmark suite. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2012)
14. Guizilini, V., Vasiljevic, I., Chen, D., Ambruș, R., Gaidon, A.: Towards zero-shot scale-aware monocular depth estimation. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV) (2023)
15. Guizilini, V., Vasiljevic, I., Fang, J., Ambru, R., Shakhnarovich, G., Walter, M.R., Gaidon, A.: Depth field networks for generalizable multi-view scene representation. In: Proceedings of the European Conference on Computer Vision (ECCV) (2022)

16. He, K., Gkioxari, G., Dollár, P., Girshick, R.B.: Mask R-CNN. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV) (2017)
17. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016)
18. Ittelson, W.H.: Size as a cue to distance: Static localization. The American journal of psychology **64**(1), 54–67 (1951)
19. Jaderberg, M., Simonyan, K., Zisserman, A., Kavukcuoglu, K.: Spatial transformer networks. In: Advances in Neural Information Processing Systems (NeurIPS) (2015)
20. Jin, L., Zhang, J., Hold-Geoffroy, Y., Wang, O., Blackburn-Matzen, K., Sticha, M., Fouhey, D.F.: Perspective fields for single image camera calibration. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 17307–17316 (2023)
21. Kanazawa, A., Black, M.J., Jacobs, D.W., Malik, J.: End-to-end recovery of human shape and pose. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018)
22. Kar, A., Tulsiani, S., Carreira, J., Malik, J.: Amodal completion and size constancy in natural scenes. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV). pp. 127–135 (2015)
23. Karras, T., Laine, S., Aila, T.: A style-based generator architecture for generative adversarial networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019)
24. Kocabas, M., Huang, C.P., Hilliges, O., Black, M.J.: PARE: part attention regressor for 3d human body estimation. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV) (2021)
25. Liu, R., Lehman, J., Molino, P., Petroski Such, F., Frank, E., Sergeev, A., Yosinski, J.: An intriguing failing of convolutional neural networks and the coordconv solution. Advances in Neural Information Processing Systems (NeurIPS) **31** (2018)
26. Malik, J., Rosenholtz, R.: Computing local surface orientation and shape from texture for curved surfaces. International Journal of Computer Vision (IJCV) (1997)
27. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. In: Proceedings of the European Conference on Computer Vision (ECCV) (2020)
28. Miyato, T., Jaeger, B., Welling, M., Geiger, A.: GTA: A geometry-aware attention mechanism for multi-view transformers. arXiv (2023)
29. Ranftl, R., Lasinger, K., Hafner, D., Schindler, K., Koltun, V.: Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) **44**(3) (2022)
30. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. In: Advances in Neural Information Processing Systems (NeurIPS) (2015)
31. Shaham, T.R., Dekel, T., Michaeli, T.: Singan: Learning a generative model from a single natural image. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV) (2019)
32. Silberman, N., Hoiem, D., Kohli, P., Fergus, R.: Indoor segmentation and support inference from rgbd images. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 746–760. Springer (2012)
33. Su, J., Lu, Y., Pan, S., Wen, B., Liu, Y.: Roformer: Enhanced transformer with rotary position embedding. arXiv (2021)

34. Sun, Y., Dong, L., Patra, B., Ma, S., Huang, S., Benhaim, A., Chaudhary, V., Song, X., Wei, F.: A length-extrapolatable transformer. In: Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL) (2022)
35. Vaswani, A., Shazeer, N.M., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. Advances in Neural Information Processing Systems (NeurIPS) (2017)
36. Vinci, L.D.: A treatise on painting (1632), http://gutenberg.org/ebooks/46915
37. Xu, R., Wang, X., Chen, K., Zhou, B., Loy, C.C.: Positional encoding as spatial inductive bias in gans. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2021)
38. Yifan, W., Doersch, C., Arandjelović, R., Carreira, J., Zisserman, A.: Input-level inductive biases for 3d reconstruction. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2022)
39. Zhang, X., Li, Q., Mo, H., Zhang, W., Zheng, W.: End-to-end hand mesh recovery from a monocular rgb image. In: ICCV (2019)
40. Zhao, Y., Kong, S., Fowlkes, C.: Camera pose matters: Improving depth prediction by mitigating pose distribution bias. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2021)