3D Pose Estimation for Fine-Grained Object Categories

Yaming Wang[1]
wym@umiacs.umd.edu
Xiao Tan[2]
tanxiao01@baidu.com
Yi Yang[2]
yangyi05@baidu.com
Xiao Liu[2]
liuxiao12@baidu.com
Errui Ding[2]
dingerrui@baidu.com
Feng Zhou[2]
zhoufeng09@baidu.com
Larry S. Davis[1]
lsd@umiacs.umd.edu

[1] University of Maryland
College Park, MD, USA
[2] Baidu, Inc.

deep-6d-pose

Abstract

Existing object pose estimation datasets are related to generic object types and there is so far no dataset for fine-grained object categories. In this work, we introduce a new large dataset to benchmark pose estimation for fine-grained objects, thanks to the availability of both 2D and 3D fine-grained data recently. Specifically, we augment two popular fine-grained recognition datasets (StanfordCars and CompCars) by finding a fine-grained 3D CAD model for each sub-category and manually annotating each object in images with 3D pose. We show that, with enough training data, a full perspective model with continuous parameters can be estimated using 2D appearance information alone. We achieve this via a framework based on Faster/Mask R-CNN. This goes beyond previous works on category-level pose estimation, which only estimate discrete/continuous viewpoint angles or recover rotation matrices often with the help of key points. Furthermore, with fine-grained 3D models available, we incorporate a novel 3D representation named as *location field* into the CNN-based pose estimation framework to further improve the performance.

1 Introduction

In the past few years, the fast-pacing progress of generic image recognition on ImageNet [10] has drawn increasing attention of research in classifying fine-grained object categories [9, 23], *e.g.* bird species [24], car makes and models [8]. However, just recognizing the object labels is still far from solving many industrial problems where we need to have a deeper

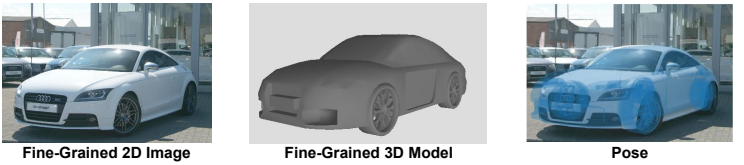

Figure 1: For an image from a fine-grained category (Left), we find its corresponding fine-grained 3D model (Middle) and annotate its pose (Right). The problem is to estimate the pose such that the projection of the 3D model align with the image as well as possible.

Dataset	# class	# image	annotation	fine-grained
3D Object [19]	10	6,675	discretized view	✗
EPFL Cars [15]	1	2,299	continuous view	✗
Pascal 3D+ [25]	12	30,899	2d-3d alignment	✗
ObjectNet3D [27]	100	90,127	2d-3d alignment	✗
StanfordCars 3D (Ours)	196	16,185	2d-3d alignment	✓
CompCars 3D (Ours)	113	5,696	2d-3d alignment	✓
Total (Ours)	309	21881	2d-3d alignment	✓

Table 1: We provide a larger-scale pose annotation than most existing datasets. Although ObjectNet3D also annotates 100 classes with more than 90,000 images, their CAD models are for generic objects, not in fine-grained details.

understanding of other attributes of the object [11]. In this work, we study the problem of estimating 3D pose for fine-grained objects from monocular images. We believe this will become an indispensable component in some broader tasks. For example, to build a vision-based car damage assessment system, an important step is to estimate the exact pose of the car so that the damaged part can be well aligned for further detailed analysis.

To address this task, collecting suitable data is of vital importance. However, large-scale as they are, recent category-level pose estimation datasets are typically designed for generic object types [25, 27] and there is so far no large-scale pose dataset for fine-grained object categories. Although datasets on generic object types could contain decent information for pose, they lack of fine-detailed matching of object shapes during annotation, since they usually use only a few universal 3D object models to match a group of objects with different shapes in one hyper class [25]. In this work, we introduce a new dataset that is able to benchmark pose estimation for fine-grained objects. Specifically, we augment two existing fine-grained recognition datasets, StanfordCars [8] and CompCars [29], with two types of useful 3D information: (*i*) for each car in the image, we manually annotate the pose parameters for a full perspective projection; (*ii*) we provide an accurate match of the computer aided design (CAD) model for each category. The resulting augmented dataset consists of more than 20,000 images for over 300 fine-grained categories.

To our best knowledge, this is the first work for fine-grained object pose estimation. Given the built dataset with high-quality pose annotations, we show that the pose parameters can be predicted from a single 2D image with only appearance information. Compared to most previous works [16, 22, 30], our method does not require the intermediate prediction of 2D/3D key points. In addition, we assume a full perspective model, which is a more challenging setting than previous works of estimating discrete/continuous viewpoint angles (azimuth) [3] or recovering the rotation matrices only [13]. Our expected goal is that by projecting the fine-grained 3D model according to the regressed pose estimation, the projection can align well with the object in the 2D image. To tackle this problem, we integrate pose

estimation into the Faster/Mask R-CNN framework [6, 18] by sharing information between the detection and pose estimation branches. However, a simple extension leads to inaccurate prediction result. Therefore, we introduce *3D location field*, a novel dense 3D representation that maps each pixel to the 3D location on the model surface. This representation provides powerful supervision for the CNNs to efficiently capture the 3D shape of objects. Additionally, it requires no rendering such that there is no domain gap between real-world annotated data and synthetic data. Using large amount of synthetic location fields for pre-training, we overcome the problem of data shortage as well as the domain gap caused by rendering.

Our contribution is three-fold. First, we collect a new large 3D pose dataset for fine-grained objects with a better match to the fine-detailed shapes of objects. Second, we propose a system based on Faster/Mask R-CNN that estimates a full perspective model parameters on our dataset. Third, we introduce *location field*, a new 3D representation that efficiently encodes the object 3D shapes for deep network usage.

2 Related Work

Dataset. Earlier object pose datasets are limited not only in their dataset scales but also in the types of annotation they covered. Table 1 provides a quantitative comparison between our dataset and previous ones. For example, 3D Object [19] dataset only provides viewpoint annotation for 10 object classes with 10 instances for each class. EPFL Car dataset [15] consists of 2,299 images of 20 car instances captured at multiple azimuth angles; moreover, the other parameters including elevation and distance are kept almost the same for all the instances in order to simplify the problem [15]. Pascal 3D+ [25] is perhaps the first large-scale 3D pose dataset for generic object categories, with 30,899 images from 12 different classes of the Pascal VOC dataset [2]. Recently, ObjectNet3D dataset [27] further extends the scale to 90,127 images of 100 categories. Both Pascal 3D+ and ObjectNe3D datasets assume a camera model with 6 parameters to annotate. However, different images in one hyper class (*i.e.*, cars) are usually matched with a few coarse 3D CAD models, thereby the projection error might be large due to the lack of accurate CAD models in some cases. Being aware of these problems, we therefore project fine-grained CAD models to match with images. In addition, our datasets surpass most of previous ones in both scales of images and classes.

Pose Estimation. Despite the fact that continuous pose parameters are available for dataset such as Pascal 3D+, a majority of previous works [8, 17, 21, 22, 25] still casts the pose estimation problem as a multi-class classification of discrete viewpoint angles, which can be further refined as shown in [5, 28]. There are very few works except [13, 16] that directly regresses the continuous pose parameters. Although [16] estimates a weak-perspective model for object categories and is able to lay the 3D models onto 2D images for visualization, its quantitative evaluation is still limited to 3D rotations. In contrast, we tackle a more challenging problem that estimates the full perspective matrices from a single image. Our new dataset allows us to quantitatively evaluate the estimated perspective projection. Based on this, we design a new efficient CNN framework as well as a new 3D representation that further improves the pose estimation accuracy.

Fine-Grained Recognition. Fine-grained recognition refers to the task of distinguishing sub-ordinate categories [8, 23, 24]. In earlier works, 3D information is a common source to gain recognition performance improvement [14, 20, 26, 31]. As deep learning prevails and fine-grained datasets become larger [9, 12], the effect of 3D information on recognition

diminishes. Recently, [20] incorporate 3D bounding box into deep framework when images of cars are taken from a fixed camera. On the other hand, almost all existing fine-grained datasets are lack of 3D pose labels or 3D shape information [8], and pose estimation for fine-grained object categories are not well-studied. Our work fills this gap by annotating poses and matching CAD models on two existing popular fine-grained recognition datasets and performing the new task of pose estimation based on the augmented annotations.

3 Dataset

Our dataset annotation process is similar to ObjectNet3D [27]. We first select the most appropriate 3D car model from ShapeNet [1] for each category in the fine-grained image dataset. For each image, we then obtain its pose parameters by asking the annotators to align the projection of the 3D model with the image using our designed interface.

3.1 3D Models

We build two fine-grained 3D pose datasets for vehicles. Each dataset consists of two parts, *i.e.*, 2D images and 3D models. The 2D images of vehicles are collected from StanfordCars [8] and CompCars [29] respectively. Target objects in most images are non-occluded and easy to identify. In order to distinguish between fine-grained categories, we adopt a distinct model for each category. Thanks to ShapeNet [1], a large number of 3D models for fine-grained vehicles are available with make/model names in their meta data, which are used to find the corresponding 3D model given an image category name. If there is no exact match between a category name and meta data, we manually select a visually similar 3D model for that category. For StanfordCars, we annotate images for all 196 categories, where 148 categories have exact matched models. For CompCars, we only include 113 categories with matched 3D models in ShapeNet. To our best knowledge, our dataset is the very first one which employs fine-grained category aware 3D model in 3D pose estimation.

3.2 Camera Model

The world coordinate system is defined in accordance with the 3D model coordinate system. In this case, a point $\mathbf{X}$ on a 3D model is projected onto a point $\mathbf{x}$ on a 2D image:

$$\mathbf{x} = \mathcal{P}\mathbf{X}, \tag{1}$$

via a perspective projection matrix:

$$\mathcal{P} = K[R|T], \tag{2}$$

where K denotes the intrinsic parameter:

$$K = \begin{bmatrix} f & 0 & u \\ 0 & f & v \\ 0 & 0 & 1 \end{bmatrix}, \tag{3}$$

and R encodes a 3×3 rotation matrix between the world and camera coordinate systems, parameterized by three angles, i.e., elevation e, azimuth a and in-plane rotation θ. We assume that the camera is always facing towards the origin of the 3D model. Hence the translation

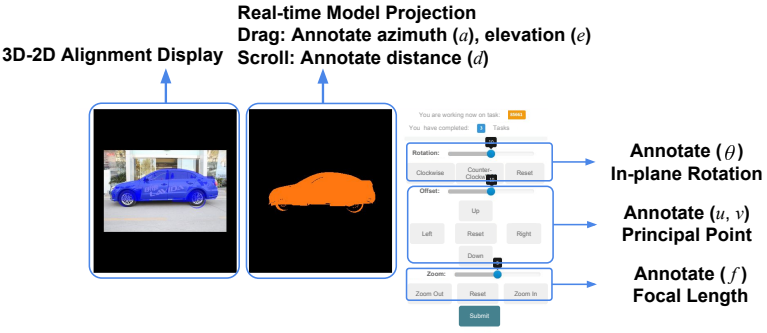

Figure 2: An overview of our annotation interface.

$T = [0,0,d]^{\mathrm{T}}$ is only defined up to the model depth d, the distance between the origins of two coordinate systems, and the principal point (u,v) is the projection of the origin of world coordinate system on the image. As a result, our model has 7 parameters in total: camera focal length f, principal point location u, v, azimuth a, elevation e, in-plane rotation θ and model depth d. Note that, since the images are collected online, even the annotated intrisic parameters (u, v and f) are approximation. Compared with previous annotations [25, 27] with 6 parameters (f fixed), our camera model considers both the camera focal length f and object depth d in a full perspective projection for finer 2D-3D alignment.

3.3 2D-3D Alignment

We annotate 3D pose information for all 2D images in our datasets through crowd-sourcing. To facilitate the annotation process, we develop an annotation tool illustrated in Figure 2. For each image during annotation, we choose the 3D model according to the fine-grained car type given beforehand. Then, we ask the annotators to adjust the 7 parameters so that the projected 3D model is aligned with the target object in 2D image. This process can be roughly summarized as follows: (1) shift the 3D model such that the center of the model (the origin of the world coordinate system) is roughly aligned with the center of the target object in the 2D image; (2) rotate the model to the same orientation as the target object in the 2D image; (3) adjust the model depth d and camera focal length f to match the size of the target object in the 2D image. Some finer adjustment might be applied after the three main steps. In this way we annotate all 7 parameters across the whole dataset. On average, each image takes approximately 60 seconds to annotate by an experienced annotator. To ensure the quality, after one round of annotation across the whole dataset, we perform quality check and let the annotators do a second round revision for unqualified examples.

4 3D Pose Estimation for Fine-Grained Object Categories

Given an input image of a fine-grained object, our task is to predict *all* the 7 parameters related to Equation (2), *i.e.*, 3D rotation $R(a,e,\theta)$, distance d, principal point (u,v) and f, such that the projected 3D model can align well with the object in the 2D image.

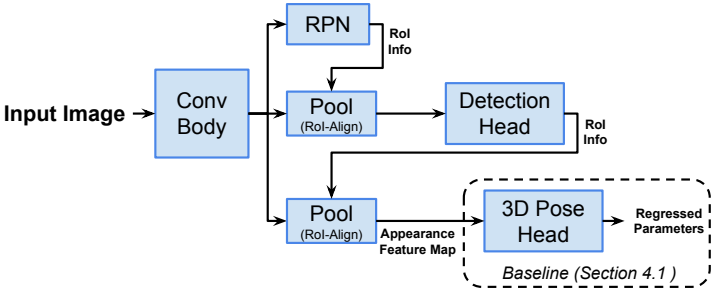

Figure 3: Our base pose estimation framework. Given an input 2D image, we adapt the Mask R-CNN framework to regress the pose parameters from the pooled appearance feature map with its area determined by the Detection module. The whole network is trained end-to-end.

4.1 Baseline Framework

Our baseline method only uses 2D appearance to regress the pose parameters. It is a modified version of Faster R-CNN [18] which was originally designed for object detection. Casting our pose estimation problem into a detection framework is motivated by the relation between the two tasks. Since we are not using key points as an attention mechanism, performing pose estimation within the region of interest (RoI) helps us get rid of unrelated image regions hence make use of 2D information more effectively. In addition, 3D pose estimation is highly related to the detection task, especially the intrinsic parameters in Equation (3).

We parametrize the 3D rotation using the *quaternion* representation, converted from the angles (a, e, θ). The principal point (u, v) is highly related to RoI center. Therefore, we regress $(\Delta u, \Delta v)$, the offset of the principal point from the RoI center. Such offset exists since the projection of the 3D object center might not necessarily be the 2D center depending on the poses. For other parameters (d and f), we regress the standard format as they are.

The modification of the network architecture is relatively straightforward. As shown in Figure 3, we add a pose estimation branch along with the existing class prediction and bounding box regression branches. Similar to the bounding box regression branch, the estimation of each group of pose parameters consists of a fully-connected (FC) layer and a smoothed L_1 loss. The centers of the RoIs are also used to adjust the regression targets at training time and generate the final predictions at test time, as discussed above. For each training image, its bounding box is figured out from the perspective projection of the corresponding 3D model. Since we have fine-grained 3D models and high-quality annotations, these bounding boxes are tight to their corresponding objects.

4.2 Improve Pose Estimation via 3D Location Field

The key difference of our dataset to previous ones is that we have fine-grained 3D models such that the projection aligns better with the image. This advantage allows us to explore the usage of dense 3D representations in addition to 2D appearance to regress the pose parameters.

Given an object in an image and its 3D model, our representation, named as *3D location field*, maps every foreground pixel to its corresponding location on the surface of the 3D model, *i.e.*, $f(x, y) = (X, Y, Z)$. The resulting field has the same size as the image and has three channels containing the X, Y and Z coordinates respectively. A sample image with

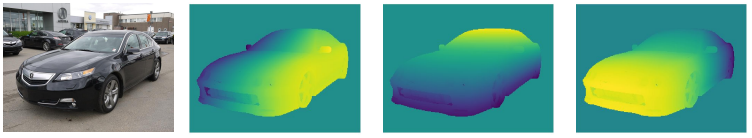

Figure 4: A sample image and its corresponding 3D location field. The location field is a 3-dimensional tensor with the same size as an image. The last channel encodes the 3d locations of a pixel on the visible surface of the 3D model.

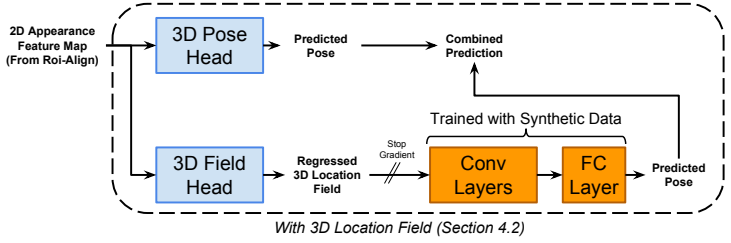

Figure 5: Our improved network architecture of using 3D location field to help pose estimation. The block in the dash-line box is to replace the corresponding base network in Figure 3. The key difference is that we add a 3D Field branch that also estimates the pose parameters.

corresponding 3D location field can be seen in Figure 4. The 3D location field is a dense representation of 3D information which can be directly used as network input.

We explore the usage of 3D location field to improve pose estimation based on Mask R-CNN. We would still expect only 2D image input at test time, therefore we regress 3D location field and use the regressed field for pose estimation. Based on the framework in Figure 3, we add a branch to regress 3D location field (instead of regressing binary masks in Mask R-CNN). The regressed location fields are fed into a CNN consisting of additional convolutional layers followed by layers to regress the pose parameters. The regressions from 2D appearance (as part of Figure 3) and 3D location field are later combined to produce the final pose parameters. Figure 5 shows the detailed network structure.

We train the pose regression from location fields using a large amount of synthetic data. The synthetic location fields are generated from the 3D models with various pre-defined poses. The location field is a very suitable representation for synthetic data augmentation due to the following reasons: (*i*) the field only encodes 3D location information without any rendering of 3D models and naturally avoids the domain gap between synthetic data and photo-realistic data; (*ii*) the field is invariant to color, texture and scale of the images.

5 Experiments

5.1 Evaluation Metrics

For each test sample, we introduce two metrics to comprehensively evaluate object poses.

Following [16, 22], the first metric, **Rotation Error**, focuses on the quality of viewpoint estimation only. Given the predicted and ground truth rotation matrices $\{R, R_{\text{gt}}\}$, the difference between the two measured by geodesic distance is $e_{\text{R}} = \frac{1}{\sqrt{2}} \| \log(R^{\text{T}} R_{\text{gt}}) \|_{\text{F}}$.

The second metric evaluates the overall quality of perspective projection. Our evaluation metric is based on **Average Distance of Model Points** in [7], which measures the

Method	Median e_{R}	Mean e_{R}	$Acc_{\frac{\pi}{6}}$	Median $\tilde{e}_{\mathrm{ADD}}$	Mean $\tilde{e}_{\mathrm{ADD}}$	$Acc_{\mathrm{th}=0.1}$
Baseline	6.68	9.89	96.59	0.0888	0.1087	60.04
w./ Field	**5.68**	**7.67**	**98.73**	**0.0834**	**0.0977**	**66.07**

Table 2: Experimental results on StanfordCars 3D dataset. The two rows show the baseline results (Section 4.1) and the results with 3D location field (Section 4.2), respectively. The rotation error e_{R} is measured in degree (°). The accuracy ($Acc_{\frac{\pi}{6}}$ and $Acc_{\mathrm{th}=0.1}$) is measured in percentage (%). Please see Section 5.1 for details about evaluation metrics.

Method	Median e_{R}	Mean e_{R}	$Acc_{\frac{\pi}{6}}$	Median $\tilde{e}_{\mathrm{ADD}}$	Mean $\tilde{e}_{\mathrm{ADD}}$	$Acc_{\mathrm{th}=0.1}$
Baseline	8.09	13.02	93.62	0.1275	0.1580	32.52
w./ Field	6.14	8.98	98.00	0.1141	0.1408	40.15
FT Baseline	5.51	8.69	96.84	0.0878	0.1123	58.58
FT w./ Field	**4.74**	**7.45**	**98.31**	**0.0836**	**0.1047**	**64.01**

Table 3: Experimental results on CompCars 3D dataset. The last two rows show results fine-tuned (FT) from a StanfordCars 3D pre-trained model.

averaged distance between predicted projected points and their corresponding ground truth projections. Concretely, given one test result $\{\mathcal{P}, \mathcal{P}_{\mathrm{gt}}, \mathcal{M}\}$, where its predicted pose is $\mathcal{P}$, its ground truth pose $\mathcal{P}_{\mathrm{gt}}$ and corresponding 3D model $\mathcal{M}$, the metric is defined as

$$e_{\mathrm{ADD}}(\mathcal{P}, \mathcal{P}_{\mathrm{gt}}; \mathcal{M}) = \underset{\mathbf{X} \in \mathcal{M}}{\mathrm{avg}} \left\| \mathcal{P}\mathbf{X} - \mathcal{P}_{\mathrm{gt}}\mathbf{X} \right\|_2 \tag{4}$$

According to [7], this is the most widely-used error function to evaluate a projection matrix. The unit of the above distance is the number of pixels. To make the metric scale-invariant, we normalize it using the diameter of the 2D bounding box. We denote the normalized distance as $\tilde{e}_{\mathrm{ADD}}$. It is worth mentioning again that the 3D models are only used when computing the evaluation metrics. During test time, only a single 2D image is fed into the network to predict the pose $\mathcal{P}$.

To measure the performance over the whole test set, we compute the mean and median of e_R and $\tilde{e}_{\mathrm{ADD}}$ over all test samples. Also, by setting thresholds on the two metrics, we can get an accuracy number. For e_R, following [16, 22], we set the threshold to be $\frac{\pi}{6}$. For $\tilde{e}_{\mathrm{ADD}}$, the common threshold is 0.1, which means that the prediction with average projection error less than 10% of the 2D diameter is considered correct.

5.2 Experimental Settings

Data Split. For StanfordCars 3D, since we have annotated all the images, we follow the standard train/test split provided by the original dataset [8] with 8144 training examples and 8041 testing examples. For CompCars 3D, we randomly sample $2/3$ of our annotated data as training set and the rest $1/3$ as testing set, resulting in 3798 training and 1898 test examples.
Baseline Implementation. Our implementation is based on the Detectron package [4], which includes Faster/Mask R-CNN implementations. The convolutional body (*i.e.*, the "backbone" in [6]) used for the baseline is ResNet-50. For fair comparison, the convolutional body is initialized from ImageNet pre-trained model, and other layers are randomly initialized (*i.e.*, we are not using COCO pre-trained detectors). Following the setting of Mask R-CNN, the whole network is trained end-to-end. At test time, we adopt a cascaded strategy, where the 3D pose branch is applied only to the highest scoring box prediction.

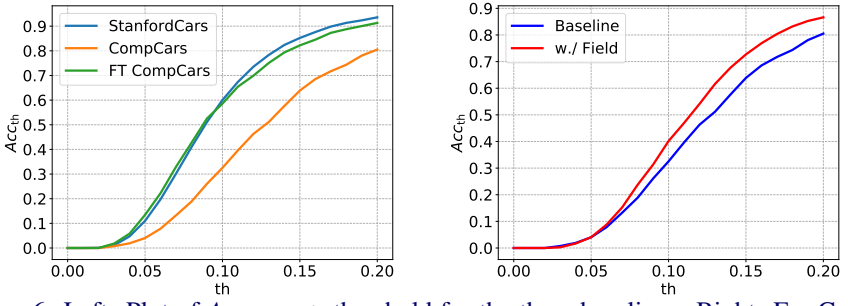

Figure 6: Left: Plot of Acc_{th} *w.r.t.* threshold for the three baselines. Right: For CompCars 3D, compare the result using location field to the baseline curve.

Figure 7: Visualizations of predicted poses for test examples. For each dataset, we show five examples of successful predictions and two of the failure cases, separated by the solid black line in the figure.

3D Location Field. In Section 4.2, incorporating 3D location fields involves two steps – field regression and pose regression from fields. Field regression is trained together with detection and baseline pose estimation in an end-to-end fashion, similar to Mask R-CNN. The ground truth training fields are generated from the annotations (3D models and poses). The second step, pose regression from fields is trained using the synthetic data generated from the pool of matched 3D models in a dataset (38102/14017 synthetic samples for StanfordCars&CompCars 3D). We only regress the quaternion using the location fields.

5.3 Results and Analysis

The quantitative results for StanfordCars 3D and CompCars 3D are shown in Table 2 and Table 3 respectively. The changes of Acc_{th} *w.r.t* the threshold for the datasets are shown in Figure 6. For CompCars 3D dataset, besides ImageNet initialization we also report the result finetuned from a StanfordCars 3D pretrained model, since the number of training samples in StanfordCars is relatively larger.

As can be seen in Table 2 and 3, our baseline performs very well on estimating the rotation matrix for both datasets, with Median e_R less than 10 degrees and $Acc_{\frac{\pi}{6}}$ around 95%. While recovering the full perspective model is a much more challenging task, Table 2 shows that promising performance can be achieved with enough properly annotated training samples. For StanfordCars 3D, Median $\tilde{e}_{\text{ADD}}$ (the median of the average projection error) is less than 10% of the diameter of the 2D bounding box. When the training set is limited, from the first and the third row of Table 3, we can see the effectiveness of transfer learning from a larger dataset. Regarding the effectiveness of the 3D location field, we can observe

consistent performance gain across all datasets. The main reasons are two-fold: (*i*) this 3D representation enables the usage of large amounts of synthetic training data with no domain gap; (*ii*) our field regression adapted from Mask R-CNN works well such that even the pose prediction based on the regressed field can help a lot at test time.

We visualize the predicted poses in Figure 7. As shown on the left part of Figure 7, our method is able to handle poses of various orientations, scales and locations of the projection. On the right part of Figure 7, failure cases exist in our predictions, indicating there are still potential rooms for improvement, especially for the estimation of scale, cases with large perspective distortion and some uncommon poses with few training samples.

6 Conclusion

We study the problem of pose estimation for fine-grained object categories. We annotate two popular fine-grained recognition datasets with fine-grained 3D shapes and poses. We propose an approach to estimate the full perspective parameters from a single image. We further propose 3D location field as a dense 3D representation to facilitate pose estimation. Experiments on our datasets suggest that this is an interesting problem in future.

References

[1] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. ShapeNet: An information-rich 3D model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[2] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.

[3] Amir Ghodrati, Marco Pedersoli, and Tinne Tuytelaars. Is 2D information enough for viewpoint estimation? In *BMVC*, 2014.

[4] Ross Girshick, Ilija Radosavovic, Georgia Gkioxari, Piotr Dollár, and Kaiming He. Detectron. https://github.com/facebookresearch/detectron, 2018.

[5] Kota Hara, Raviteja Vemulapalli, and Rama Chellappa. Designing deep convolutional neural networks for continuous object orientation estimation. *arXiv preprint arXiv:1702.01499*, 2017.

[6] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *ICCV*, 2017.

[7] Tomas Hodan, Jiri Matas, and Stepán Obdrzálek. On evaluation of 6d object pose estimation. In *Computer Vision - ECCV 2016 Workshops - Amsterdam, The Netherlands, October 8-10 and 15-16, 2016, Proceedings, Part III*, pages 606–619, 2016.

[8] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3D object representations for fine-grained categorization. In *ICCV Workshops on 3D Representation and Recognition*, 2013.

[9] Jonathan Krause, Benjamin Sapp, Andrew Howard, Howard Zhou, Alexander Toshev, Tom Duerig, James Philbin, and Li Fei-Fei. The unreasonable effectiveness of noisy data for fine-grained recognition. In *European Conference on Computer Vision*, pages 301–320. Springer, 2016.

[10] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet classification with deep convolutional neural networks. In *NIPS*, 2012.

[11] Joseph J Lim, Hamed Pirsiavash, and Antonio Torralba. Parsing IKEA objects: Fine pose estimation. In *ICCV*, 2013.

[12] Tsung-Yu Lin, Aruni RoyChowdhury, and Subhransu Maji. Bilinear CNN models for fine-grained visual recognition. In *ICCV*, 2015.

[13] Siddharth Mahendran, Haider Ali, and René Vidal. 3D pose regression using convolutional neural networks. In *ICCV*, volume 1, page 4, 2017.

[14] Roozbeh Mottaghi, Yu Xiang, and Silvio Savarese. A coarse-to-fine model for 3D pose estimation and sub-category recognition. In *CVPR*, 2015.

[15] Mustafa Ozuysal, Vincent Lepetit, and Pascal Fua. Pose estimation for category specific multiview object localization. In *CVPR*, 2009.

[16] Georgios Pavlakos, Xiaowei Zhou, Aaron Chan, Konstantinos G Derpanis, and Kostas Daniilidis. 6-DOF object pose from semantic keypoints. In *ICRA*, pages 2011–2018, 2017.

[17] Bojan Pepik, Michael Stark, Peter Gehler, and Bernt Schiele. Teaching 3D geometry to deformable part models. In *CVPR*, 2012.

[18] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. In *NIPS*, 2015.

[19] Silvio Savarese and Li Fei-Fei. 3D generic object categorization, localization and pose estimation. In *ICCV*, 2007.

[20] Jakub Sochor, Adam Herout, and Jiri Havel. BoxCars: 3D boxes as cnn input for improved fine-grained vehicle recognition. In *CVPR*, 2016.

[21] Hao Su, Charles R Qi, Yangyan Li, and Leonidas J Guibas. Render for CNN: Viewpoint estimation in images using CNNs trained with rendered 3D model views. In *ICCV*, 2015.

[22] Shubham Tulsiani and Jitendra Malik. Viewpoints and keypoints. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1510–1519, 2015.

[23] Grant Van Horn, Oisin Mac Aodha, Yang Song, Alex Shepard, Hartwig Adam, Pietro Perona, and Serge Belongie. The inaturalist challenge 2017 dataset. *arXiv preprint arXiv:1707.06642*, 2017.

[24] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical report, 2011.

[25] Yu Xiang, Roozbeh Mottaghi, and Silvio Savarese. Beyond PASCAL: A benchmark for 3D object detection in the wild. In *WACV*, 2014.

[26] Yu Xiang, Wongun Choi, Yuanqing Lin, and Silvio Savarese. Data-driven 3d voxel patterns for object category recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1903–1911, 2015.

[27] Yu Xiang, Wonhui Kim, Wei Chen, Jingwei Ji, Christopher Choy, Hao Su, Roozbeh Mottaghi, Leonidas Guibas, and Silvio Savarese. ObjectNet3D: A large scale database for 3D object recognition. In *ECCV*, 2016.

[28] Linjie Yang, Jianzhuang Liu, and Xiaoou Tang. Object detection and viewpoint estimation with auto-masking neural network. In *ECCV*, 2014.

[29] Linjie Yang, Ping Luo, Chen Change Loy, and Xiaoou Tang. A large-scale car dataset for fine-grained categorization and verification. In *CVPR*, 2015.

[30] Xiaowei Zhou, Spyridon Leonardos, Xiaoyan Hu, Kostas Daniilidis, et al. 3D shape estimation from 2D landmarks: A convex relaxation approach. In *CVPR*, 2015.

[31] M Zeeshan Zia, Michael Stark, Bernt Schiele, and Konrad Schindler. Detailed 3D representations for object recognition and modeling. *PAMI*, 35(11):2608–2623, 2013.