Single Image 3D Interpreter Network

Jiajun Wu[1(✉)], Tianfan Xue[1],
Joseph J. Lim[1,2], Yuandong Tian[3], Joshua B. Tenenbaum[1],
Antonio Torralba[1], and William T. Freeman[1,4]

[1] Massachusetts Institute of Technology, Cambridge, USA
jiajunwu@mit.edu
[2] Stanford University, Stanford, USA
[3] Facebook AI Research, Menlo Park, USA
[4] Google Research, Cambridge, USA

Abstract. Understanding 3D object structure from a single image is an important but difficult task in computer vision, mostly due to the lack of 3D object annotations in real images. Previous work tackles this problem by either solving an optimization task given 2D keypoint positions, or training on synthetic data with ground truth 3D information.

In this work, we propose 3D INterpreter Network (3D-INN), an end-to-end framework which sequentially estimates 2D keypoint heatmaps and 3D object structure, trained on both real 2D-annotated images and synthetic 3D data. This is made possible mainly by two technical innovations. First, we propose a Projection Layer, which projects estimated 3D structure to 2D space, so that 3D-INN can be trained to predict 3D structural parameters supervised by 2D annotations on real images. Second, heatmaps of keypoints serve as an intermediate representation connecting real and synthetic data, enabling 3D-INN to benefit from the variation and abundance of synthetic 3D objects, without suffering from the difference between the statistics of real and synthesized images due to imperfect rendering. The network achieves state-of-the-art performance on both 2D keypoint estimation and 3D structure recovery. We also show that the recovered 3D information can be used in other vision applications, such as image retrieval.

Keywords: 3D structure · Single image 3D reconstruction · Keypoint estimation · Neural network · Synthetic data

1 Introduction

Deep networks have achieved impressive performance on 1,000-way image classification [19]. However, for any visual system to parse objects in the real world,

J. Wu and T. Xue are equal contributions.

Electronic supplementary material The online version of this chapter (doi:10.1007/978-3-319-46466-4_22) contains supplementary material, which is available to authorized users.

B. Leibe et al. (Eds.): ECCV 2016, Part VI, LNCS 9910, pp. 365–382, 2016.
DOI: 10.1007/978-3-319-46466-4_22

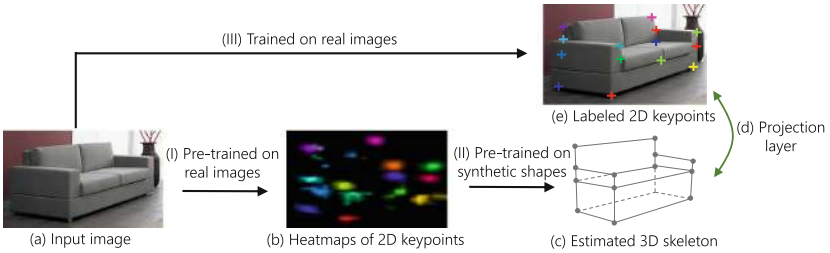


Fig. 1. An abstraction of the proposed 3D INterpreter Network (3D-INN).

it needs not only to assign category labels to objects, but also to interpret their intra-class variation. For example, for a chair, we are interested in its intrinsic properties such as its *style*, *height*, leg *length*, and seat *width*, and extrinsic properties such as its *pose*.

In this paper, we recover these object properties from a single image by estimating 3D structure. Instead of a 3D mesh or a depth map [2,9,16,18,32,40,50], we represent an object via a 3D skeleton [47], which consists of keypoints and the connections between them (Fig. 1c). Being a simple abstraction, the skeleton representation preserves the structural properties that we are interested in. In this paper, we assume one pre-defined skeleton model for each object category (*e.g.* chair, sofa, and human).

The main challenge of recovering 3D object structure from a single RGB image is the difficulty in obtaining training images with ground truth 3D geometry, as manually annotating 3D structures of objects in real images is labor-intensive and often inaccurate. Previous methods tackle this problem mostly in two ways. One is to directly recover a 3D skeleton from estimated 2D keypoint locations by minimizing its reprojection error. This method uses no training data in 3D reconstruction, thus it is not robust to noisy keypoint estimation, as shown in experiments (Sect. 4). The other is to train on synthetically rendered images of 3D objects [23,41], where complete 3D structure is available. However, the statistics of synthesized images are often different from those of real images, possibly due to lighting, occlusion, and shape details, making models trained mostly on synthetic data hard to generalize well to real images.

In this paper, we propose 3D INterpreter Network (3D-INN), an end-to-end framework for recovering 3D object skeletons, trained on both real 2D-labeled images and synthetic 3D objects. Our model has two major innovations. First, we introduce a *Projection Layer*, a simple renderer which calculates 2D keypoint projections from a 3D skeleton at the end of the network (Fig. 1d). This enables 3D-INN to predict 3D structural parameters that minimizes the error in the 2D space with labeled real images, without requiring 3D object annotations.

Second, we further observe that training with real images only under a projection layer is not enough due to the fundamental ambiguity in 2D-to-3D mapping. In other words, the algorithm might recover an unnatural 3D geometry whose projection matches the 2D image, because the projection layer only requires the 3D prediction to be plausible in 2D. We therefore incorporate synthetic

3D objects into training data, in order to encode the knowledge of "plausible shapes". To this end, our model is designed to first predict keypoint locations (Fig. 1-I) and then to regress 3D parameters (Fig. 1-II). We pre-train the former part with 2D-annotated real images and the latter part with synthetic 3D data, and then train the joint framework end-to-end with the projection layer (Fig. 1-III). We choose heatmaps of keypoints (Fig. 1b) as an intermediate representation between two components to resolve the domain adaptation issue between real and synthetic data.

Several experiments demonstrate the effectiveness of 3D-INN. First, the proposed network achieves state-of-the-art performance on various keypoint localization datasets (FLIC [35] for human bodies, CUB-200-2011 [51] for birds, and our new dataset, Keypoint-5, for furniture). We then evaluate our network on IKEA [25], a dataset with ground truth 3D object structures and viewpoints. On 3D structure estimation, 3D-INN shows its advantage over a optimization-based method [61] when keypoint estimation is imperfect. On 3D viewpoint estimation, it also performs better than the state-of-the-art [41]. We further evaluate 3D-INN, in combination with detection frameworks [11], on the popular benchmark PASCAL 3D+ [53]. Though our focus is not on pose estimation, 3D-INN achieves results comparable to the state-of-the-art [41,49]. At last, we show qualitatively that 3D-INN has wide vision applications including 3D object retrieval.

Our contributions include (1) introducing an end-to-end 3D INterpreter Network (3D-INN) with a projection layer, which can be trained to predict 3D structural parameters using only 2D-annotated images, (2) using keypoint heatmaps to connect real and synthetic worlds, strengthening the generalization ability of the network, and (3) state-of-the-art performance in 2D keypoint and 3D structure and viewpoint estimation.

2 Related Work

Single image 3D reconstruction. Previous 3D reconstruction methods mainly used object representations based on depth or meshes, or based on skeletons or pictorial structure. Depth-/mesh-based models can recover detailed 3D object structure from a single image, either by adapting existing 3D models from a database [2,3,8,15,16,36,39,40,59], or by inferring from its detected 2D silhouette [18,32,50].

In this paper, we choose to use a skeleton-based representation, exploiting the power of abstraction. The skeleton model can capture geometric changes of articulated objects [1,47,57], like a human body or the base of a swivel chair. Typically, researchers recovered a 3D skeleton from a single image by minimizing its projection error on the 2D image plane [12,22,27,33,55,62]. Recent work in this line [1,61] demonstrated state-of-the-art performance. In contrast to them, we propose to use neural networks to predict a 3D object skeleton from its 2D keypoints, which is more robust to imperfect detection results and can be jointly learned with keypoint estimators.

Our work also connects to the traditional field of vision as inverse graphics [14,21] and analysis by synthesis [5,20,52,58], as we use neural nets to decode

latent 3D structure from images, and use a projection layer for rendering. Their approaches often required supervision for the inferred representations or made over-simplified assumptions of background and occlusion in images. Our 3D-INN learns 3D representation without using 3D supervision, and generalizes to real images well.

2D keypoint estimation. Another line of related work is 2D keypoint estimation. During the past decade, researchers have made significant progress in estimating keypoints on humans [35,56] and other objects [38,51]. Recently, there have been several attempts to apply convolutional neural networks to human keypoint estimation [7,29,44,48], which all achieved significant improvement. Inspired by these work, we use 2D keypoints as our intermediate representation, and aim to recover 3D skeleton from them.

3D viewpoint estimation. 3D viewpoint estimation seeks to estimate the 3D orientation of an object from a single image [53]. Some previous methods formulated it as a classification or regression problem, and aimed to directly estimate the viewpoint from an image [10,41]. Others proposed to estimate 3D viewpoint from detected 2D keypoints or edges in the image [24,49,62]. While the main focus of our work is to estimate 3D object structure, our method can also predict its 3D viewpoint.

Training with synthetic data. Synthetic data are often used to augment the training set [30,37,40]. Su *et al*. [40] attempted to train a 3D viewpoint estimator using a combination of real and synthetic images, while Sun *et al*. [42] and Zhou *et al*. [60] also used a similar strategy for object detection and matching, respectively. Huang *et al*. [16] analyzed the invariance of convolutional neural networks using synthetic images. For image synthesis, Dosovitskiy *et al*. [9] trained a neural network to generate new images using synthetic images.

In this paper, we combine real 2D-annotated images and synthetic 3D data for training 3D-INN to recover a 3D skeleton. We use heatmaps of 2D keypoints, instead of (often imperfectly) rendered images, from synthetic 3D data, so that our algorithm has better generalization ability as the effects of imperfect rendering are minimized. Yasin *et al*. [57] also proposed to use both 2D and 3D data for training, but they uses keypoint location, instead of heatmaps, as the intermediate representation that connects 2D and 3D.

3 Methods

We design a deep convolutional network to recover 3D object structure. The input to the network is a single image with an object of interest at its center, which can be obtained by state-of-the-art object detectors [34]. The output of the network is a 3D object skeleton, including its 2D keypoint locations, 3D structural parameters, and 3D poses (see Fig. 3). In the following, we will describe our 3D skeleton representation (Sect. 3.1), network architecture (Sect. 3.2), and training strategy (Sect. 3.3).

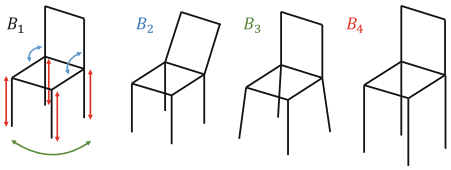


Fig. 2. A simplification of our skeleton model and base shapes for chairs

3.1 3D Skeleton Representation

As discussed in Sect. 1, we use skeletons as our 3D object representation. A skeleton consists of a set of keypoints as well as their connections. For each object category, we manually design a 3D skeleton characterizing its abstract 3D geometry.

There exist intrinsic ambiguities in recovering 3D keypoint locations from a single 2D image. We resolve this issue by assuming that objects can only have constrained deformations [47]. For example, chairs may have various leg lengths, but for a single chair, its four legs are typically of equal length. We model these constraints by formulating 3D keypoint locations as a weighted sum of a set of base shapes [18]. The first base shape is the mean shape of all objects within the category, and the rest define possible deformations and intra-class variations. Figure 2 shows an simplification of our skeleton representation for chairs: the first is the mean shape of chairs, the second controls how the back bends, and the last two are for legs. The weight for each base shape determines how strong the deformation is, and we denote these weights as the *internal parameters* of an object.

Formally, let $\mathcal{Y} \in \mathbb{R}^{3\times N}$ be a matrix of 3D coordinates of all N keypoints. Our assumption is that the 3D keypoint locations are a weighted sum of base shapes $B_k \in \mathbb{R}^{3\times N}$, or $\mathcal{Y} = \sum_{k=1}^{K} \alpha_k B_k$, where $\{\alpha_k\}$ is the set of internal parameters of this object and K is the number of base shapes.

Further, let $\mathcal{X} \in \mathbb{R}^{2\times N}$ be the corresponding 2D coordinates. Then the relationship between the observed 2D coordinates $\mathcal{X}$ and the internal parameters $\{\alpha_k\}$ is

$$\mathcal{X} = P(R\mathcal{Y} + T) = P(R\sum_{k=1}^{K} \alpha_k B_k + T), \tag{1}$$

where $R \in \mathbb{R}^{3\times 3}$ (rotation) and $T \in \mathbb{R}^3$ (translation) are the external parameters of the camera, and P is a projective transformation. P only depends on the focal length f under the central projection we assuming.

Therefore, to recover the 3D structural information of an object in a 2D image, we only need to estimate its internal parameters ($\{\alpha_k\}$) and the external viewpoint parameters (R, T, and f). In the following section, we discuss how we design a neural network for this task, and how it can be jointly trained with real 2D images and synthetic 3D objects.

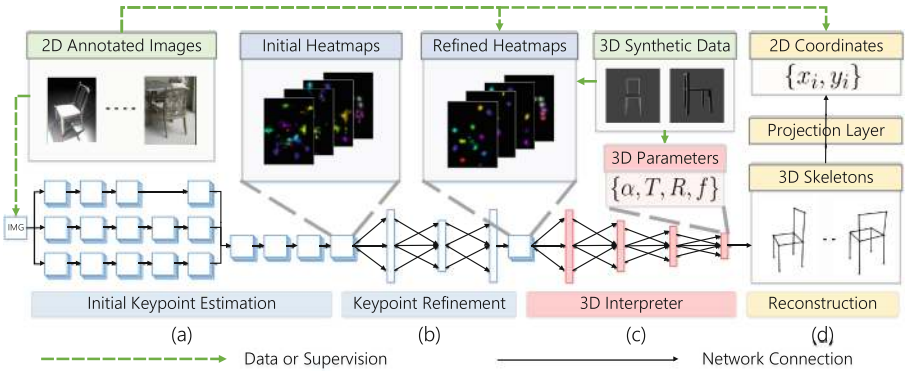


Fig. 3. 3D-INN takes a single image as input and reconstructs the detailed 3D structure of the object in the image (*e.g.*, human, chair, *etc.*). The network is trained independently for each category, and here we use chairs as an example. **(a)** Estimating 2D keypoint heatmaps with a multi-scale CNN. **(b)** Refining keypoint locations by considering the structural constraints between keypoints. This is implicitly enforced with an information bottleneck which yields cleaner heatmaps. **(c)** Recovered 3D structural and camera parameters $\{\alpha, T, R, f\}$. **(d)** The projection layer maps reconstructed 3D skeletons back to 2D keypoint coordinates. (Color figure online)

3.2 Architecture of 3D-INN

Our network consists of three components: first, a keypoint estimator, which localizes 2D keypoints of objects from 2D images by regressing to their heatmaps (Fig. 3a and b, blue part); second, a 3D interpreter, which infers internal 3D structural and viewpoint parameters from the heatmaps (Fig. 3c, red part); third, a projection layer, mapping 3D skeletons to 2D keypoint locations so that real 2D-annotated images can be used as supervision (Fig. 3d, yellow part).

Keypoint Estimation. The keypoint estimation component consists of two steps: initial estimation (Fig. 3a) and keypoint refinement (Fig. 3b).

The network architecture for initial keypoint estimation is inspired by the pipeline proposed by Tompson *et al.* [44,45]. The network takes multi-scaled images as input and estimates keypoint heatmaps. Specifically, we apply Local Contrast Normalization (LCN) on each image, and then scale it to 320×240, 160×120, and 80×60 as input to three separate scales of the network. The output is k heatmaps, each with resolution 40×30, where k is the number of keypoints of the object in the image.

At each scale, the network has three sets of 5×5 convolutional (with zero padding), ReLU, and 2×2 pooling layers, followed by a 9×9 convolutional and ReLU layer. The final outputs for the three scales are therefore images with resolution 40×30, 20×15, and 10×7, respectively. We then upsample the outputs of the last two scales to ensure they have the same resolution (40×30). The outputs from the three scales are later summed up and sent to a Batch Normalization layer and three 1×1 convolution layers, whose goal is to regress to

target heatmaps. We found that Batch Normalization is critical for convergence, while Spatial Dropout, proposed in [44], does not affect performance.

The second step of keypoint estimation is keypoint refinement, whose goal is to implicitly learn category-level structural constraints on keypoint locations after the initial keypoint localization. The motivation is to exploit the contextual and structural knowledge among keypoints (*e.g.*, arms cannot be too far from the torso). We design a mini-network which, like an auto-encoder, has information bottleneck layers, enforcing it to implicitly model the relationship among keypoints. Some previous works also use this idea and achieve better performance with lower computational cost in object detection [34] and face recognition [43].

In the keypoint refinement network, We use three fully connected layers with widths $8,192$, $4,096$, and $8,192$, respectively. After refinement, the heatmaps of keypoints are much cleaner, as shown in Fig. 5 and Sect. 4.

3D Interpreter. The goal of our 3D interpreter is to infer 3D structure and viewpoint parameters, using estimated 2D heatmaps from earlier layers. While there are many different ways of solving Eq. 1, our deep learning approach has clear advantages. First, traditional methods [13,47] that minimize the reprojection error consider only one keypoint hypothesis, and is therefore not robust to noises in keypoint detection. In contrast, our framework uses soft heatmaps of keypoint locations, as shown in Fig. 3c, which is more robust when some keypoints are invisible or incorrectly located. Further, our algorithm only requires a single forward propagation during testing, making it more efficient than the most previous optimization-base methods.

As discussed in Sect. 3.1, the set of 3D parameters we estimate consists of $S = \{\alpha_i, R, T, f\}$, with which we are able to recover the 3D object structure using Eq. 1. As shown in Fig. 3c, we use four fully connected layers as our 3D interpreter, with widths $2,048$, 512, 128, and $|S|$, respectively. The Spatial Transformer Network [17] also explored the idea of learning rotation parameters R with neural nets, but our network can also recover structural parameters $\{\alpha_i\}$. Note that our representation for latent parameters may also be naturally extended to other types of abstract 3D representations.

Projection Layer. The last component of the network is a projection layer (Fig. 3d). The projection layer takes estimated 3D parameters as input, and computes projected 2D keypoint coordinates $\{x_i, y_i\}$ using Eq. 1. As all operations are differentiable, the projection layer enables us to use 2D-annotated images as ground truth, and run back-propagation to update the entire network.

3.3 Training Strategy

A straightforward training strategy is to use real 2D images as input, and their 2D keypoint locations as supervision for the output of the projection layer. Unfortunately, experiments show that the network can hardly converge using this training scheme, due to the high-dimensional search space and the ambiguity in the 3D to 2D projection.

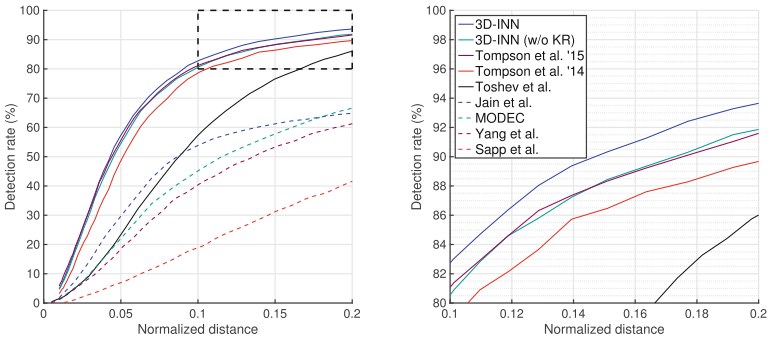


Fig. 4. PCK curves on the FLIC dataset. 3D-INN performs consistently better than other methods. Without keypoint refinement, it is comparable to Tompson *et al.* [44]. A zoomed view of the dashed rectangle is shown on the right.

We therefore adopt an alternative three-step training strategy: first, training the keypoint estimator (Fig. 3a and b) using real images with 2D keypoint heatmaps as supervision; second, training the 3D interpreter (Fig. 3c) using synthetic 3D data as there are no ground truth 3D annotations available for real images; and third, training the whole network using real 2D images with supervision on the output of the projection layer at the end.

To generate synthetic 3D objects, for each object category, we first randomly sample structural parameters $\{\alpha_i\}$ and viewpoint parameters P, R and T. Then we calculate 3D keypoint coordinates using Eq. 1. To model deformations that cannot be captured by base shapes, we add Gaussian perturbation to 3D keypoint locations of each synthetic 3D object, whose variance is 1 % of its diagonal length. Examples of synthetic 3D shapes are shown in Fig. 3c. Note that we are not rendering these synthesized objects, as we are only using heatmaps of keypoints, rather than rendered images, as training input.

4 Evaluation

We evaluate our entire framework, 3D-INN, as well as each component within. In this section, we present both qualitative and quantitative results on 2D keypoint estimation (Sect. 4.1) and 3D structure recovery (Sect. 4.2).

4.1 2D Keypoint Estimation

Data. For 2D keypoint estimation, we evaluate our algorithm on three image datasets: FLIC [35] for human bodies, CUB-200-2011 [51] for birds, and a new dataset Keypoint-5 for furniture. Specifically, FLIC is a challenging dataset containing 3,987 training images and 1,016 test images, each labeled with 10 keypoints of human bodies. The CUB-200-2011 dataset was originally proposed for fine-grained bird classification, but with labeled keypoints of bird parts. It has

5,994 images for training and 5,794 images for testing, each coming with up to 15 keypoints.

We also introduce a new dataset, Keypoint-5, which contains five categories: bed, chair, sofa, swivel chair, and table. There are 1,000 to 2,000 images in each category, where 80 % are for training and 20 % for testing. For each image, we asked three workers on Amazon Mechanical Turk to label locations of a predefined category-specific set of keypoints; we then, for each keypoint, used the median of the three responses as ground truth.

Metrics. To quantitatively evaluate the accuracy of estimated keypoints on FLIC (human body), we use the standard Percentage of Correct Keypoints (PCK) measure [35] to be consistent with previous works [35,44,45]. We use the evaluation toolkit and results of competing methods released by the Tompson *et al*. [44]. On CUB-200-2011 (bird) and the new Keypoint-5 (furniture) dataset, following the convention [26,38], we evaluate results in Percentage of Correct Parts (PCP) and Average Error (AE). PCP is defined as the percentage of keypoints localized within 1.5 times of the standard deviation of annotations. We use the evaluation code from [26] to ensure consistency. Average error is computed as the mean of the distance, bounded by 5, between a predicted keypoint location and ground truth.

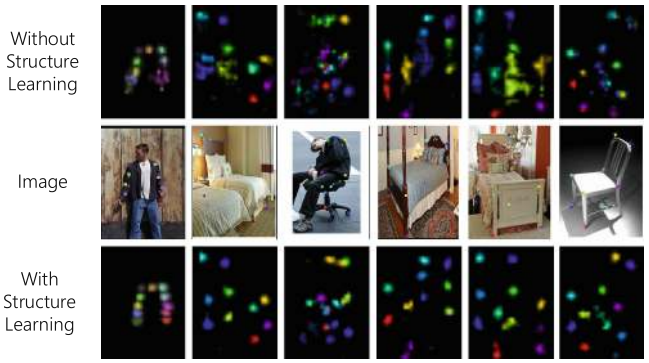


Fig. 5. 2D keypoint predictions from a single image, where each color corresponds to a keypoint. The keypoint refinement step cleans up false positives and produces more regulated predictions. (Color figure online)

Table 1. Keypoint estimation results on CUB-200-2011, measured in PCP (%) and AE. Our method is comparable to Mdshift [38] (better in AE but worse in PCP), and better than all other algorithms.

Method	Poselets [6]	Consensus [4]	Exemplar [26]	Mdshift [38]	3D-INN	Human
PCP (%)	27.47	48.70	59.74	**69.1**	66.7	84.72
Average error	2.87	2.13	1.80	1.39	**1.35**	1.00

Table 2. Keypoint estimation results of 3D-INN and Tompson *et al.* [44] on Keypoint-5, measured in PCP (%) and AE. 3D-INN is consistently better in both measures. We retrained the network in [44] on Keypoint-5.

	Method	Bed	Chair	Sofa	Swivel chair
PCP	3D-INN	**77.4**	**87.7**	**77.4**	**78.5**
	Tompson *et al.* [44]	76.2	85.3	76.9	69.2
AE	3D-INN	**1.16**	**0.92**	**1.14**	**1.19**
	Tompson *et al.* [44]	1.20	1.02	1.19	1.54

Results. For 2D keypoint detection, we only train the keypoint estimator in our 3D-INN (Fig. 3a and b) using the training images in each dataset. Figure 4 shows the accuracy of keypoint estimation on the FLIC dataset. On this dataset, we employ a fine-level network for post-processing, as suggested by [44]. Our method performs better than all previous methods [35,44,45,48,56] at all precisions. Moreover, the keypoint refinement step improves results significantly (about 2 % for a normalized distance ≥ 0.15), without which our framework has similar performance with [44]. Such improvement is also demonstrated in Fig. 5, where the heatmaps after refinement are far less noisy.

The accuracy of keypoint estimation on CUB-200-201 dataset is listed in Table 1. Our method is better than [26] in both metrics, and is comparable to the state-of-the-art [38]. Specifically, compared with [38], our model more precisely estimates the keypoint locations for correctly detected parts (a lower AE), but miss more parts in the detection (a lower PCP). On our Keypoint-5 dataset, our model achieves higher PCPs and lower AEs compared to the state-of-the-art [44] for all categories, as shown in Table 2. These experiments in general demonstrate the effectiveness of our model on keypoint detection.

4.2 Structural Parameter Estimation

For 3D structural parameter estimation, we evaluate 3D-INN from three different prospectives. First, we evaluate our 3D interpreter (Fig. 3c alone) against the optimization-based method [61]. Second, we test our full pipeline on the IKEA dataset [25], where ground truth 3D labels are available. We show qualitative results on three datasets: Keypoint-5, IKEA, and SUN [54] at last.

Comparing with an optimization-based method. We first compared our 3D interpreter (Fig. 3c) with the state-of-the-art optimization-based method that directly minimizing re-projection error (Eq. 1) on the synthetic data.

We first tested the effectiveness of our 3D interpreter (Fig. 3c) on synthetic data. We compare our trained 3D interpreter against the state-of-the-art method on directly minimizing re-projection error (Eq. 1). Since most optimization based methods only consider the parallel projection, we extend the one by Zhou *et al.* [61] as follows. We first uses their algorithm to get an initial guess of internal

parameters and viewpoints, and then applying a simple gradient descent method to refine it considering perspective distortion.

We generate synthetic data for this experiment, using the scheme described in Sect. 3.3. Each data point contains the 2D keypoint heatmaps of an object, and its corresponding 3D keypoint locations and viewpoint, which we would like to estimate. We also add different levels of salt-and-pepper noise to heatmaps to evaluate the robustness of both methods. We generated 30,000 training and 1,000 testing cases. Because the analytical solution only takes keypoint coordinates as input, we convert heatmaps to coordinates using argmax.

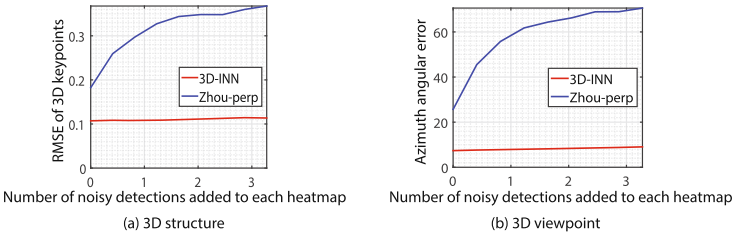


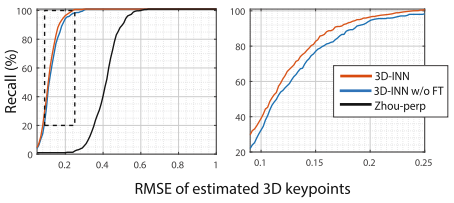
Fig. 6. Plots comparing our method against an analytic solution on synthetic heatmap. (a) The accuracy of 3D structure estimation; (b) The accuracy of 3D viewpoint estimation.

For both methods, we evaluate their performance on both 3D structure recovery and 3D viewpoint estimation. For 3D structure estimation, we compare their accuracies on 3D keypoint estimation ($\mathcal{Y}$ in Sect. 3.1); for 3D viewpoint estimation, we evaluate errors in azimuth angle, following previous work [41]. As the original algorithm by Zhou *et al.* [61] was mainly designed for the parallel projection and comparatively clean heatmaps, our 3D interpreter outperforms it in the presence of noise and perspective distortion, as shown in Fig. 6.

Evaluating the full pipeline. We now evaluate 3D-INN on estimating 3D structure and 3D viewpoint. We use the IKEA dataset [25] for evaluation, as it provides ground truth 3D mesh models and the associated viewpoints for testing images. We manually label ground truth 3D keypoint locations on provided 3D meshes, and calculate the root-mean-square error (RMSE) between estimated and ground truth 3D keypoint locations.

As IKEA only have no more than 200 images per category, we instead train 3D-INN on our Keypoint-5, as well as one million synthetic data points, using the strategy described in Sect. 3.3. Note that, first, we are only using no more than $2,000$ real images per category for training and, second, we are testing the trained model on different datasets, avoiding the possible dataset bias [46].

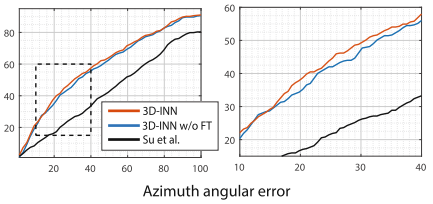
The left half of Fig. 7 shows RMSE-Recall curve of both our algorithm and the optimization-based method described above (Zhou-perp [61]). The y-axis shows the recall — the percentage of testing samples under a certain RMSE threshold. We test two versions of our algorithm: with fine-tuning (3D-INN) and without



Average recall %

Method	Bed	Sofa	Chair	Avg.
3D-INN	**88.64**	**88.03**	**87.84**	**88.03**
3D-INN w/o FT	87.19	87.10	87.08	87.10
Zhou-perp [60]	52.31	58.02	60.76	58.46

(a) Structure estimation



Average recall %

Method	Table	Sofa	Chair	Avg.
3D-INN	**55.02**	64.65	**63.46**	**60.30**
3D-INN w/o FT	52.33	**65.45**	62.01	58.90
Su *et al.* [40]	52.73	35.65	37.69	43.34

(b) Pose estimation

Fig. 7. Evaluation on the IKEA dataset [25]. (a) The accuracy of structure estimation. RMSE-Recall curved is shown in the first row, and zoomed-views of the dashed rectangular regions are shown on the right. The third row shows the average recall on all thresholds. (b) The accuracy of pose estimation.

fine-tuning (3D-INN w/o FT). Both significantly outperform the optimization-based method [61], as [61] is not designed for multiple keypoint hypothesis and perspective distortion, while our 3D-INN can deal with them. Also, finetuning improves the accuracy of keypoint estimation by about 5 % under the RMSE threshold 0.15.

Table 3. Joint object detection and viewpoint estimation on PASCAL 3D+ [53]. Following previous work, we use Average Viewpoint Precision (AVP) as our measure, which extends AP so that a true positive should have both a correct bounding box and a correct viewpoint (here we use a 4-view quantization). Both V&K [49] and our algorithm (3D-INN) use R-CNN [11] for object detection, and others use their own detection algorithm. VDPM [53] and DPM-VOC+VP [31] are trained on PASCAL VOC 2012, V&K [49] is trained on PASCAL 3D+, Su *et al.* [41] is trained on PASCAL VOC 2012, together with synthetic 3D CAD models, and 3D-INN is trained on Keypoint-5.

Category	VDPM [53]	DPM-VOC+VP [31]	Su *et al.* [41]	V&K [49]	3D-INN
Chair	6.8	6.1	15.7	**25.1**	23.1
Sofa	5.1	11.8	18.6	43.8	**45.8**

Though we focus on recovering 3D object structure, as an extension, we also evaluate 3D-INN on 3D viewpoint estimation. We compare it with the state-of-the-art viewpoint estimation algorithm by Su *et al.* [41]. The right half of Fig. 7 shows the results (recall) in azimuth angle. As shown in the table, 3D-INN outperforms Su *et al.* [41] by about 40 % (relative), measured in average

Fig. 8. Qualitative results on Keypoint-5, IKEA, and SUN databases. For each example, the first one is the input image, the second one is the reconstruct 3D skeleton using the network before fine-tuning, and third one is using the network after fine-tuning. The last column shows failure cases.

Training: beds, Test: chairs

Training: sofas, Test: chairs

Fig. 9. Qualitative results on chairs using networks trained on sofas or beds. In most cases models provide reasonable output. Mistakes are often due to the difference between the training and test sets, *e.g.*, in the third example, the model trained on beds fails to estimate chairs facing backward.

recall. This is mainly because it is not straightforward for Su *et al.* [41], mostly trained on (cropped) synthesized images, to deal with the large number of heavily occluded objects in the IKEA dataset.

Although our algorithm assumes a centered object in an input image, we can apply it, in combination with an object detection algorithm, on images where object locations are unknown. We evaluate the results of joint object detection and viewpoint estimation on PASCAL 3D+ dataset [53]. We use the standard R-CNN [11] for object detection, and our 3D-INN for viewpoint estimation. Table 3 shows that our model is comparable with the state-of-the-art [49], and ourperforms other algorithms with a significant margin. Note that all the other

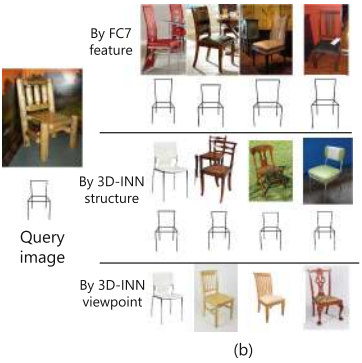


Fig. 10. Retrieval results in different feature spaces. 3D-INN helps to retrieve objects with similar 3D structures or similar viewpoints.

Fig. 11. Object graph visualization based on learned object representations: we visualize images using t-SNE [28] on predicted 3D viewpoint by 3D-INN.

algorithms are trained on either PASCAL VOC or PASCAL 3D+, while our algorithm is trained on Keypoint-5, which indicates that our learned model is not suffering much from the dataset bias problem [46].

Qualitative results on benchmarks. At last, we show qualitative results on Keypoint-5, IKEA, and the SUN database [54] in Fig. 8. When the image is clean and objects are not occluded, our algorithm can recover 3D object structure and viewpoint with high accuracy, while fine-tuning can further helps to improve the results (see chairs at row 1 column 1, and row 4 column 1). Our algorithm is also robust of partial occlusion, demonstrated by the IKEA bed at row 5 column 1. One major failure case is when the object is heavily cropped in the input image (see the last column in row 4 to 7), as the 3D object skeleton becomes hard to infer.

When 3D-INN is used in combination with detection models, it needs to deal with imperfect detection results. Here, we also evaluate 3D-INN on noisy input, specifically, on images with an object from a different but similar category. Figure 9 shows the recovered 3D structures of chairs using a model trained either on sofas or beds. In most cases 3D-INN still provides reasonable output, and the mistakes are mostly due to the difference between training and test sets, *e.g.*, the model trained on beds does not perform well on chairs facing backward, because there are almost no beds with a similar viewpoint in the training set.

5 Applications

Our inferred latent parameters, as a compact and informative representation of objects in images, have wide applications. In this section, we demonstrate representative ones including image retrieval and object graph construction.

Image Retrieval. Using estimated 3D structural and viewpoint information, we can retrieve images based on their 3D configurations. Figure 10 shows image retrieval results using FC7 features from AlexNet [19] and using the 3D structure and viewpoint learned by 3D-INN. Our retrieval database includes all testing images of chairs and sofas in Keypoint-5. In each row, we sort the best matches of the query image, measured by Euclidean distance in a specific feature space. We retrieve images in two ways: *by structure* uses estimated internal structural parameters ($\{\alpha_i\}$ in Eq. 1), and *by viewpoint* uses estimated external viewpoint parameters (R in Eq. 1).

Object Graph. Similar to the retrieval task, we visualize all test images for chairs in Keypoint-5 in Fig. 11, using t-SNE [28] on estimated 3D viewpoints. Note the smooth transition from the chairs facing left to those facing right.

6 Conclusion

In this paper, we introduced 3D INterpreter Network (3D-INN), which recovers the 2D keypoint and 3D structure of a (possibly deformable) object given a single image. To achieve this goal, we used 3D skeletons as an abstract 3D representation, incorporated a projection layer to the network for learning 3D parameters from 2D labels, and employed keypoint heatmaps to connect real and synthetic data. Empirically, we showed that 3D-INN performs well on both 2D keypoint estimation and 3D structure and viewpoint recovery, comparable to or better than the state-of-the-arts. Further, various applications demonstrated the potential of the skeleton representation learned by 3D-INN.

Acknowledgement. This work is supported by NSF Robust Intelligence 1212849 and NSF Big Data 1447476 to W.F., NSF Robust Intelligence 1524817 to A.T., ONR MURI N00014-16-1-2007 to J.B.T., Shell Research, and the Center for Brain, Minds and Machines (NSF STC award CCF-1231216). The authors would like to thank Nvidia for GPU donations. Part of this work was done during Jiajun Wu's internship at Facebook AI Research.

References

1. Akhter, I., Black, M.J.: Pose-conditioned joint angle limits for 3d human pose reconstruction. In: CVPR (2015)
2. Aubry, M., Maturana, D., Efros, A., Russell, B., Sivic, J.: Seeing 3d chairs: exemplar part-based 2d–3d alignment using a large dataset of cad models. In: CVPR (2014)
3. Bansal, A., Russell, B.: Marr revisited: 2d–3d alignment via surface normal prediction. In: CVPR (2016)
4. Belhumeur, P.N., Jacobs, D.W., Kriegman, D.J., Kumar, N.: Localizing parts of faces using a consensus of exemplars. IEEE TPAMI **35**(12), 2930–2940 (2013)
5. Bever, T.G., Poeppel, D.: Analysis by synthesis: a (re-) emerging program of research for language and vision. Biolinguistics **4**(2–3), 174–200 (2010)

6. Bourdev, L., Maji, S., Brox, T., Malik, J.: Detecting people using mutually consistent poselet activations. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010. LNCS, vol. 6316, pp. 168–181. Springer, Heidelberg (2010). doi:10.1007/978-3-642-15567-3_13
7. Carreira, J., Agrawal, P., Fragkiadaki, K., Malik, J.: Human pose estimation with iterative error feedback. In: CVPR (2016)
8. Choy, C.B., Xu, D., Gwak, J., Chen, K., Savarese, S.: 3d–r2n2: a unified approach for single and multi-view 3D object reconstruction. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2006, Part VIII. LNCS, vol. 9912, pp. 1–17. Springer, Heidelberg (2016)
9. Dosovitskiy, A., Tobias Springenberg, J., Brox, T.: Learning to generate chairs with convolutional neural networks. In: CVPR (2015)
10. Fidler, S., Dickinson, S.J., Urtasun, R.: 3d object detection and viewpoint estimation with a deformable 3d cuboid model. In: NIPS (2012)
11. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: CVPR (2014)
12. Hejrati, M., Ramanan, D.: Analysis by synthesis: 3d object recognition by object reconstruction. In: CVPR (2014)
13. Hejrati, M., Ramanan, D.: Analyzing 3d objects in cluttered images. In: NIPS (2012)
14. Hinton, G.E., Ghahramani, Z.: Generative models for discovering sparse distributed representations. Philos. Trans. R. Soc. London B: Biol. Sci. **352**(1358), 1177–1190 (1997)
15. Hu, W., Zhu, S.C.: Learning 3d object templates by quantizing geometry and appearance spaces. IEEE TPAMI **37**(6), 1190–1205 (2015)
16. Huang, Q., Wang, H., Koltun, V.: Single-view reconstruction via joint analysis of image and shape collections. ACM SIGGRAPH **34**(4), 87 (2015)
17. Jaderberg, M., Simonyan, K., Zisserman, A., Kavukcuoglu, K.: Spatial transformer networks. In: NIPS (2015)
18. Kar, A., Tulsiani, S., Carreira, J., Malik, J.: Category-specific object reconstruction from a single image. In: CVPR (2015)
19. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: NIPS (2012)
20. Kulkarni, T.D., Kohli, P., Tenenbaum, J.B., Mansinghka, V.: Picture: a probabilistic programming language for scene perception. In: CVPR (2015)
21. Kulkarni, T.D., Whitney, W.F., Kohli, P., Tenenbaum, J.B.: Deep convolutional inverse graphics network. In: NIPS (2015)
22. Leclerc, Y.G., Fischler, M.A.: An optimization-based approach to the interpretation of single line drawings as 3d wire frames. IJCV **9**(2), 113–136 (1992)
23. Li, Y., Su, H., Qi, C.R., Fish, N., Cohen-Or, D., Guibas, L.J.: Joint embeddings of shapes and images via cnn image purification. ACM SIGGRAPH Asia **34**(6), 234 (2015)
24. Lim, J.J., Khosla, A., Torralba, A.: FPM: fine pose parts-based model with 3D CAD models. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8694, pp. 478–493. Springer, Heidelberg (2014). doi:10.1007/978-3-319-10599-4_31
25. Lim, J.J., Pirsiavash, H., Torralba, A.: Parsing ikea objects: fine pose estimation. In: ICCV (2013)
26. Liu, J., Belhumeur, P.N.: Bird part localization using exemplar-based models with enforced pose and subcategory consistency. In: ICCV (2013)

27. Lowe, D.G.: Three-dimensional object recognition from single two-dimensional images. Artif. Intell. **31**(3), 355–395 (1987). Elsevier
28. Van der Maaten, L., Hinton, G.: Visualizing data using t-sne. JMLR **9**(11), 2579–2605 (2008)
29. Newell, A., Yang, K., Deng, J.: Stacked hourglass networks for human pose estimation. arXiv preprint arXiv:1603.06937 (2016)
30. Peng, X., Sun, B., Ali, K., Saenko, K.: Exploring invariances in deep convolutional neural networks using synthetic images. CoRR, abs/1412.7122 2 (2014)
31. Pepik, B., Stark, M., Gehler, P., Schiele, B.: Teaching 3d geometry to deformable part models. In: CVPR (2012)
32. Prasad, M., Fitzgibbon, A., Zisserman, A., Van Gool, L.: Finding nemo: deformable object class modelling using curve matching. In: CVPR (2010)
33. Ramakrishna, V., Kanade, T., Sheikh, Y.: Reconstructing 3D human pose from 2D image landmarks. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012. LNCS, vol. 7575, pp. 573–586. Springer, Heidelberg (2012). doi:10.1007/978-3-642-33765-9_41
34. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. In: NIPS (2015)
35. Sapp, B., Taskar, B.: Modec: multimodal decomposable models for human pose estimation. In: CVPR (2013)
36. Satkin, S., Lin, J., Hebert, M.: Data-driven scene understanding from 3D models. In: BMVC (2012)
37. Shakhnarovich, G., Viola, P., Darrell, T.: Fast pose estimation with parameter-sensitive hashing. In: ICCV (2003)
38. Shih, K.J., Mallya, A., Singh, S., Hoiem, D.: Part localization using multi-proposal consensus for fine-grained categorization. In: BMVC (2015)
39. Shrivastava, A., Gupta, A.: Building part-based object detectors via 3d geometry. In: ICCV, pp. 1745–1752 (2013)
40. Su, H., Huang, Q., Mitra, N.J., Li, Y., Guibas, L.: Estimating image depth using shape collections. ACM TOG **33**(4), 37 (2014)
41. Su, H., Qi, C.R., Li, Y., Guibas, L.: Render for cnn: viewpoint estimation in images using cnns trained with rendered 3d model views. In: ICCV (2015)
42. Sun, B., Saenko, K.: From virtual to reality: fast adaptation of virtual object detectors to real domains. In: BMVC (2014)
43. Taigman, Y., Yang, M., Ranzato, M., Wolf, L.: Web-scale training for face identification. In: CVPR (2015)
44. Tompson, J., Goroshin, R., Jain, A., LeCun, Y., Bregler, C.: Efficient object localization using convolutional networks. In: CVPR (2015)
45. Tompson, J.J., Jain, A., LeCun, Y., Bregler, C.: Joint training of a convolutional network and a graphical model for human pose estimation. In: NIPS (2014)
46. Torralba, A., Efros, A.A.: Unbiased look at dataset bias. In: CVPR (2011)
47. Torresani, L., Hertzmann, A., Bregler, C.: Learning non-rigid 3d shape from 2d motion. In: NIPS (2003)
48. Toshev, A., Szegedy, C.: Deeppose: human pose estimation via deep neural networks. In: CVPR, pp. 1653–1660 (2014)
49. Tulsiani, S., Malik, J.: Viewpoints and keypoints. In: CVPR (2015)
50. Vicente, S., Carreira, J., Agapito, L., Batista, J.: Reconstructing pascal voc. In: CVPR (2014)
51. Wah, C., Branson, S., Welinder, P., Perona, P., Belongie, S.: The Caltech-UCSD Birds-200-2011 Dataset. Technical report. CNS-TR-2011-001, California Institute of Technology (2011)

52. Wu, J., Yildirim, I., Lim, J.J., Freeman, B., Tenenbaum, J.: Galileo: perceiving physical object properties by integrating a physics engine with deep learning. In: NIPS (2015)
53. Xiang, Y., Mottaghi, R., Savarese, S.: Beyond pascal: a benchmark for 3d object detection in the wild. In: WACV (2014)
54. Xiao, J., Hays, J., Ehinger, K., Oliva, A., Torralba, A.: Sun database: large-scale scene recognition from abbey to zoo. In: CVPR (2010)
55. Xue, T., Liu, J., Tang, X.: Example-based 3d object reconstruction from line drawings. In: CVPR (2012)
56. Yang, Y., Ramanan, D.: Articulated pose estimation with flexible mixtures-of-parts. In: CVPR (2011)
57. Yasin, H., Iqbal, U., Krüger, B., Weber, A., Gall, J.: A dual-source approach for 3d pose estimation from a single image. In: CVPR (2016)
58. Yuille, A., Kersten, D.: Vision as bayesian inference: analysis by synthesis? Trends Cogn. Sci. **10**(7), 301–308 (2006)
59. Zeng, A., Song, S., Nießner, M., Fisher, M., Xiao, J.: 3dmatch: learning the matching of local 3d geometry in range scans. arXiv preprint arXiv:1603.08182 (2016)
60. Zhou, T., Krähenbühl, P., Aubry, M., Huang, Q., Efros, A.A.: Learning dense correspondence via 3d-guided cycle consistency. In: CVPR (2016)
61. Zhou, X., Leonardos, S., Hu, X., Daniilidis, K.: 3d shape reconstruction from 2d landmarks: a convex formulation. In: CVPR (2015)
62. Zia, M.Z., Stark, M., Schiele, B., Schindler, K.: Detailed 3d representations for object recognition and modeling. IEEE TPAMI **35**(11), 2608–2623 (2013)