Matching RGB Images to CAD Models for Object Pose Estimation

Georgios Georgakis¹, Srikrishna Karanam², Ziyang Wu², and Jana Košecká¹
¹Department of Computer Science, George Mason University, Fairfax VA
²Siemens Corporate Technology, Princeton NJ

ggeorgak@gmu.edu, {first.last}@siemens.com, kosecka@cs.gmu.edu

Abstract

We propose a novel method for 3D object pose estimation in RGB images, which does not require pose annotations of objects in images in the training stage. We tackle the pose estimation problem by learning how to establish correspondences between RGB images and rendered depth images of CAD models. During training, our approach only requires textureless CAD models and aligned RGB-D frames of a subset of object instances, without explicitly requiring pose annotations for the RGB images. We employ a deep quadruplet convolutional neural network for joint learning of suitable keypoints and their associated descriptors in pairs of rendered depth images which can be matched across modalities with aligned RGB-D views. During testing, keypoints are extracted from a query RGB image and matched to keypoints extracted from rendered depth images, followed by establishing 2D-3D correspondences. The object’s pose is then estimated using the RANSAC and PnP algorithms. We conduct experiments on the recently introduced Pix3D [33] dataset and demonstrate the efficacy of our proposed approach in object pose estimation as well as generalization to object instances not seen during training.

1. Introduction

Estimating the 3D object pose of objects is an important capability enabling robots’ interaction with real environments and objects as well as augmented reality applications. While several approaches to this problem assume RGB-D data, most mobile and wearable cameras are not paired with a depth sensor, prompting recent research focus on the RGB domain. Furthermore, even though several methods have shown promising results on 3D object pose estimation with real RGB images, they either require accurate 3D annotations or 3D object models with realistic textures in the training stage. Currently available datasets are not large enough to capture real world diversity, limiting the potential of these methods in generalizing to a variety of applications. In addition, capturing real RGB data and manual pose annotation is an arduous procedure.

The problem of object pose estimation is an inherently 3D problem; it is the shape of the object which gives away its pose regardless of its appearance. Instead of attempting to learn an intrinsic decomposition of images [14], we focus on finding the association of parts of objects depicted in RGB images with their counterparts in 3D depth images. Ideally, we would like to learn this association in order to
establish correspondences between a query RGB image and a rendered depth image from a CAD model, without requiring any existing 3D annotations. This, however, requires us to address the problem of the large appearance gap between these two modalities.

In this paper, we propose a new framework for estimating the 3D pose of objects in RGB images, using only 3D textureless CAD models of objects instances. The easily available CAD models can generate a large number of synthetically rendered depth images from multiple viewpoints. In order to address the aforementioned problems, we define a quadruplet convolutional neural network to jointly learn keypoints and their associated descriptors for robust matching between different modalities and changes in viewpoint. The general idea is to learn the keypoint locations using a pair of rendered depth images from a CAD model from two different poses, followed by learning how to match keypoints across modalities and changes in viewpoint. These are used to establish 2D-3D correspondences, followed by a RANSAC and PnP algorithm for pose estimation.

At test time, given a query RGB image, we extract keypoints and their representations and match them with a database of keypoints and their associated descriptors extracted from rendered depth images. These are used to establish 2D-3D correspondences, followed by a RANSAC and PnP algorithm for pose estimation.

To summarize, the key contributions of this work are the following:

- We present a novel framework for 3D object pose estimation which uses textureless CAD models and aligned RGB-D frames in the training stage, without explicitly requiring 3D pose annotations for the RGB images.
- We present an end-to-end learning approach for keypoint selection optimized for the relative pose estimation objective, and transfer of keypoint predictions and their representations from rendered depth to RGB images.
- We demonstrate the generalization capability of our method to new (unseen during training) instances of the same object category.

2. Related Work

There is a large body of work on 3D object pose estimation. Here, we review existing methods based on the type and the amount of used training data and its modalities.

**Using 3D textured instance models.** Notable effort was devoted to the problem of pose estimation for object instances from images, where 3D textured instance models were available during the training stage [9, 5, 35]. Early isolated approaches led to the development of more recent benchmarks for this problem [11]. Traditional approaches of this type included template matching [9, 44], where the target pose is retrieved from the best matched model in a database, and local descriptor matching [5, 35], where hand-engineered descriptors such as SIFT [21] are used to establish 2D-3D correspondences with a 3D object model followed by the PnP algorithm for 6-DoF pose. Additionally, some works employed a patch-based dense voting scheme [4, 36, 6, 16], where a function is learned to map local representations to 3D coordinates or to pose space. However, these approaches assume that the 3D object models were created from real images and contain realistic textures. In contrast, our work uses only the textureless CAD models of object instances, i.e. relies only on their shape characteristics.

**2D-to-3D alignment with CAD models.** Other work has sought to solve 3D object pose estimation as a 2D-to-3D alignment problem by utilizing object CAD models [1, 23, 19, 13, 2]. For example, Aubry et al. [1] learned part-based exemplar classifiers from textured CAD models and applied them on real images to establish 2D-3D correspondences. In a similar fashion, Lim et al. [19] trained a patch detector from edge maps for each interest point. The work of Massa et al. [23] learned how to match viewpoint-dependent exemplar features by adapting the representations extracted from real images to their CAD model counterparts. In their attempt to bridge the gap between the two modalities, the aforementioned approaches were required to either learn a huge number of exemplar classifiers, or learn how to adapt features for each specific category and viewpoint. We avoid this problem by simply adapting the keypoint predictions and their descriptors between the two modalities.

**Pose estimation paired with object detection.** With the recent success of deep convolutional neural networks (CNN) on object recognition and detection, many works extended 3D object instance pose estimation to object categories, from an input RGB image [22, 24, 25, 41, 29, 15, 20, 17]. In Mahendran et al. [22] a 3D pose regressor was learned for each object category. In Mousavian et al. [25], a discreet-continuous formulation for the pose prediction was introduced, which first classified the orientation to a discreet set of bins and then regressed the exact angle within the bin. Xiang et al. [41] decoupled the pose estimation task into multiple components such as predicting pixel-wise object labels, estimating the object’s center and distance from the camera to recover the translation, and estimating the rotation, while Poirson et al. [29] and Kehl et al. [15] both extended the SSD [20] object detector to predict azimuth and elevation or the 6-DoF pose respectively. In Kundu et al. [17], an analysis-by-synthesis approach was introduced, in which, given predicted pose and shape, the object was rendered and compared to 2D instance segmentation anno-
tations. All of these approaches require 3D pose annotations for the RGB images during training, as opposed to our work, which only needs the CAD models of the objects.

**Keypoint based methods.** Another popular direction in the pose estimation literature is learning how to estimate keypoints, which can be used to infer the pose. These methods are usually motivated by the presence of occlusions [26, 12] and require keypoint annotations. For example, Wu et al. [39] trained a model for 2D keypoint prediction on real images and estimated the 3D wireframes of objects using a model trained on synthetic shapes. The 3D wireframe is then projected to real images labeled with 2D keypoints to enforce consistency. In Li et al. [18], the authors manually annotated 3D keypoints on textured CAD models and generated a synthetic dataset which provides multiple layers of supervision during training, while Tekin et al. [37] learned to predict the 2D image locations of the projected vertices of an object’s 3D bounding box before using the PnP algorithm for pose estimation. Furthermore, Tulsiani et al. [38] exploited the relationship between viewpoint and visible keypoints and refined an existing coarse pose estimation using keypoint predictions. Our work, rather than relying on existing keypoint annotations, optimizes the keypoint selection based on a relative pose estimation objective. Related approaches are also learning keypoints [34, 7, 42, 43], but either rely on hand-crafted detectors to collect training data [42], or do not extend to pose estimation for real RGB images [34, 7, 43].

**Synthetic data generation.** In an attempt to address the scarcity of annotated data, some approaches rely on the generation of large amounts of synthetic data for training [33, 32, 8]. A common technique is to render textured CAD models and superimpose them on real backgrounds. In order to ensure diversity in the training data, rendering parameters such as pose, shape deformations, and illumination are randomly chosen. However, training exclusively on synthetic data has shown to be detrimental to the learned representations as the underlying statistics of real RGB images are usually very different.

### 3. Approach

We are interested in estimating the 3D pose of objects in RGB images by matching keypoints to the object’s CAD model. Our work does not make use of pose annotations, but instead relies on CAD model renderings of different poses that are easily obtained with an off-the-shelf renderer, such as Blender [3]. These rendered depth images are used to learn keypoints and their representations optimized for the task of pose estimation. The learned representations are then transferred to the RGB domain. In summary, our work can be divided into four objectives: keypoint learning, view-invariant descriptors, modality-invariant descriptors, and modality consistent keypoints.

Specifically, each training input is provided as a quadruplet of images, consisting of a pair of rendered depth images sampled from the object’s view sphere and a pair of aligned depth and RGB images (see Figure 2). For each image, we predict a set of keypoints and their local representations, but the optimization objective for different branches differs. For the first two branches A and B, the optimization objective for different branches differs. The color coding of the CNNs signifies weight sharing.

![Figure 2: Outline of the proposed architecture depicting the four branches of the network, their inputs, and the objectives imposed during training. The color coding of the CNNs signifies weight sharing.](image-url)

**Architecture.** Our proposed architecture is a Quadruplet convolutional neural network (CNN), where each branch has a backbone CNN (e.g., VGG) to learn feature representations and a keypoint proposal network (KPN) comprised of two convolutional layers. The output feature maps from the backbone’s last convolutional layer are fed as input to the KPN, which outputs scores for a grid-based set of keypoint locations in the image, and to the region-of-interest (RoI) pooling layer, which is utilized to extract a descriptor (dim-2048) for each location. RoI pooling layer also receives a box for each keypoint location. The dimensions of the box and the density of the sampling over the 2D image are treated as hyper-parameters. The first pair of branches (A, B) of the network is trained with a triplet loss, which is applied to local features, and a relative pose loss, which is applied to the keypoint predictions. The second pair of
branches (C, D) is trained using a Euclidean loss on the local features and with a consistency loss that attempts to align their keypoint predictions. Note that branches A, B, and C share their weights, while branch D is a different network. Since branch D receives as input a different modality than the rest and we desire branches C and D to produce the same outputs, their weights during training must be independent.

In the following sections, we describe the details of the loss functions and training.

3.1. Keypoint Learning by Relative Pose Estimation

The overall idea behind learning keypoint predictions is to select keypoints which can be used for relative pose estimation between the input depth images in branches A and B. Specifically, given the two sets of keypoints, we establish correspondences in 3D space, estimate the rotation $R$ and translation $t$, and project the keypoints from depth image A to depth image B. Any misalignment (re-projection error) between the projected keypoints is used to penalize the initial keypoint selections. A pictorial representation of the relative pose objective is shown in Figure 3b.

The relative pose objective is formulated as a least squares problem, which finds the rotation $R$ and translation $t$ such that:

$$
(R, t) = \arg \min_{R \in SO(3), t \in \mathbb{R}^3} \sum_{i=1}^{n} w_i \| (R p_i + t) - q_i \|^2
$$

where $w_i = s_i^A + s_i^B$ is the weight of correspondence $i$ and $s_i^A$ and $s_i^B$ are the predicted keypoint scores, as given by KPN followed by a Softmax layer, that belong to correspondence $i$ from branches A and B respectively. Given a set of correspondences and their weights, an SVD-based closed-form solution for estimating $R$ and $t$ that depends on $w$ can be found in [31]. The idea behind this formulation is that correspondences with high re-projection error should have low weights, therefore a low predicted keypoint score, while correspondences with low re-projection error should have high weights, therefore high predicted keypoint score. With this intuition, we can formulate the relative pose loss as:

$$
L_{\text{rel-pose}} = \frac{1}{n} \sum_{i=1}^{n} w_i g(w_i)
$$

where $g(w_i) = \| (R p_i + t) - q_i \|^2$. Since our objective is to optimize the loss function with respect to estimated keypoint scores, we need to calculate the derivative of the loss with respect to the weights $w$:

$$
\frac{\partial L}{\partial w} = g(w) + w \frac{\partial g(w)}{\partial w}
$$

The estimation of $R$ includes the computation of an SVD decomposition, so in order to avoid estimating its derivative...
we instead approximate the gradients using the Taylor series first order approximation. The derivative then becomes:

$$\frac{\partial L}{\partial w} = g(w) + w \left( \frac{g(w_0) - g(w)}{w_0 - w} \right)$$  (5)

where \(w_0 = w + randu(-\alpha, \alpha)\) and \(\alpha\) is a small constant.

Our formulation allows us to penalize keypoint scores separately, by estimating the gradients for each correspondence and backpropagating them accordingly.

### 3.2. Learning Keypoint Descriptors

In order to match keypoint descriptors across viewpoints, we apply a triplet loss on local features extracted from branches A and B. This involves using the known camera poses of the rendered pairs of depth images and sampling of training keypoint triplets (anchor-positive-negative). Specifically, for a randomly selected keypoint as an anchor from the first image, we find the closest keypoint in 3D from the paired image and use it as a positive, and also select a further away point in 3D to serve as the negative. The triplet loss then optimizes the representation such that the feature distance between the anchor and the positive points is smaller than the feature distance between the anchor and the negative points plus a certain margin, and is defined as follows:

$$L_{\text{triplet}} = \frac{1}{N} \sum_i \max(0, ||f_i^a - f_i^p||^2 - ||f_i^a - f_i^n||^2 + m)$$  (6)

where \(f_i^a\), \(f_i^p\), and \(f_i^n\) are the local features for the anchor, positive, and negative correspondingly of the \(i^{th}\) triplet example and \(m\) is the margin. Traditionally, the margin hyperparameter is manually defined as a constant throughout the training procedure, however, we take advantage of the 3D information and define the margin to be equal to \(D_n - D_p\), where \(D_n\) is the 3D distance between the anchor and negative, and \(D_p\) is the 3D distance between the anchor and positive. Ideally, \(D_p\) should be 0, but practically due to the sampling of the keypoints in the image space it is usually a small number close to 0. Essentially this ensures that the learned feature distances are proportional to the 3D distances between the examples and assumes that the features and 3D coordinates are normalized to unit vectors. Note that the triplet loss only affects the backbone CNN during training and not the KPN. A pictorial representation of the triplet objective is shown in Figure 3a.

### 3.3. Cross-modality Representation Learning

Finally, we can transfer the learned features and keypoint proposals from branches (A, B) to branch D, using branch C as a bridge, similar to knowledge distillation techniques [10]. To accomplish this, network parameters in branches A, B, and C are shared, and the outputs of branches C and D are compared and penalized according to any misalignment. The core idea is to enforce both the backbone and KPN in branches C and D to generate as similar outputs as possible. This objective can be accomplished by means of two key components that are described next.

**Local Feature Alignment.** In order to align local feature representations in branches C and D (see Figure 4a), we consider the predicted keypoint features in branch C and compute each keypoint’s feature representation, \(f_i, i = 1, ..., k\). Keypoint features at corresponding spatial locations from branch D are represented as \(\hat{f}_i, i = 1, ..., k\). Formally, we optimize the following objective function:

$$L_{\text{local-l2}} = \frac{1}{k} \sum_{i=1}^{k} ||\hat{f}_i - f_i||$$  (7)

Since we want to align \(\hat{f}_i\) with \(f_i\), during backpropagation, we fix \(f_i\) as ground-truth and backpropagate gradients of \(L_{\text{local-l2}}\) only to the appropriate locations in branch D.

**Keypoint Consistency.** Enforcement of the keypoint consistency constraint requires the KPN from branch D to produce the same keypoint predictions as the KPN from branch C. It can be achieved using a cross-entropy loss, which is equivalent to a log loss with binary labels:

$$L_{\text{cross-entropy}} = -\frac{1}{n} \sum_{i=1}^{n} y_i^* \log y_i$$  (8)

where \(y_i^*\) is the ground-truth label and \(y_i\) is the prediction. This in our case becomes:

$$L_{\text{consistency}} = -\frac{1}{n} \sum_{i=1}^{n} y_i^C \log y_i^D$$  (9)

where \(y_i^C\) are the keypoint predictions from branch C, which serve as the ground-truth, and \(y_i^D\) are the keypoint predictions from branch D. This loss penalizes any misalignment between the keypoint predictions of the two branches and forces branch D to imitate the outputs of branch C. Figure 4b illustrates the inputs to the consistency loss.

**Overall objective.** Our overall training objective is the combination of the losses described above:

$$L_{\text{all}} = \lambda_1 L_{\text{triplet}} + \lambda_2 L_{\text{rel-pose}} + \lambda_3 L_{\text{local-l2}} + \lambda_4 L_{\text{consistency}}$$  (10)

where each \(\lambda\) is the weight for the corresponding loss.

### 4. Experiments

In order to validate our approach, we perform several experiments on the newly introduced Pix3D [33] dataset,
which contains 10069 images, 395 CAD models of 9 object categories, and provides precise 3D pose annotations. For each object category, we created separate train and test sets, where the latter contains only untruncated and unoccluded examples. Four main experiments are performed. First, we compare to the supervised state-of-the-art method of [25] on the chair category (sec. 4.1); second, we use all model instances for each category during training and show the pose accuracy results on Pix3D (sec. 4.2); third, we test how our model generalizes to new object instances by training only on a subset of provided instances and testing on unseen ones (sec. 4.3); and finally, data from an external dataset, such as NYUv2 [30] is used to train and test on Pix3D (sec. 4.4).

The motivation for the fourth experiment is to demonstrate that our framework can utilize RGB-D pairs from another realistic dataset, where the alignment between the RGB and the depth is provided by the sensor.

To analyze the performance on the task of pose estimation, the geodesic distance over the rotation matrices is employed: \( \Delta(R_1, R_2) = \frac{|| \log(R_1^T R_2) ||_F}{\sqrt{2}} \). The metrics used are the percentage of predictions within \( \frac{\pi}{6} \) of the ground-truth \( \text{Acc}_\pi \) and median error \( \text{MedErr} \). Additionally, we show the individual accuracy of the three Euler angles, where the distance is the smallest difference between two angles:

\[
\Delta(\theta_1, \theta_2) = \min(2\pi - ||\theta_1 - \theta_2||, ||\theta_1 - \theta_2||)\]

For the last metric we also use a threshold of \( \frac{\pi}{6} \).

**Implementation details.** For the backbone of each branch in our architecture we utilize the VGG CNN and start our training from the ImageNet pretrained weights, while KPN is trained from scratch. We train category-specific models with a base learning rate of 0.001 and set all \( \lambda \) weights to 1. In order to regularize the relative pose loss such that it predicts keypoints inside objects, we add a mask term, realized as a multinomial logistic loss. The ground-truth is a binary mask of the object in the rendered depth. This loss is only applied on branches A and B with a smaller weight of 0.25, penalizing any keypoint predictions outside the silhouette of the object.

**Training data.** All our experiments require the generation of a set of quadruplet inputs. For the first two inputs, we first sample from the view-sphere of each object and render a view every 15 degrees in azimuth and elevation for three different distances. Then, we sample rendered pairs such that their pose difference is between \( \frac{\pi}{2} \) and \( \frac{\pi}{5} \). For the last two inputs, we require a pair of aligned depth and RGB images. In order to demonstrate our approach on the Pix3D dataset, we generate these alignments using the dataset’s annotations, however, we do not use annotations during training in any other capacity. As we will show in sec. 4.4, alternatively the aligned depth and RGB images can be sampled from an existing RGB-D dataset or through hand-alignment [2]. It is important to note that for each quadruplet the selection of the first pair of inputs is agnostic to the pose of the object in the last two inputs.

**Testing protocol.** For every CAD model instance used in our experiments, we first create a repository of descriptors each assigned to a 3D coordinate. To do so, 20 rendered views are sampled from the viewing sphere of each object, similarly to how the training data are generated, and keypoints are extracted from each view. Note that for this procedure, we use the trained network that corresponds to branch A of our architecture. Then we pass a query RGB image through the network of branch D, generate keypoints and their descriptors and match them to the repository of the corresponding object instance. Finally, the established correspondences are passed to RANSAC and PnP algorithm to estimate the pose of the object. For every keypoint generation step we use the keypoints with the top 100 scores during database creation and top 200 scores for the testing RGB images. For all experiments, except sec. 4.3, we have defined a test set which contains images of all category instances, with 179, 1451, and 152 images in total for \textit{bed}, \textit{chair}, and \textit{desk} category respectively.

**Baselines.** We carefully designed a set of baselines to compare the performance of our method with and bring out the benefits of the proposed approach. A naive baseline, given the unavailability of annotated RGB data, is to learn view-invariant depth representations and depth keypoints and simply use these keypoints and representations during testing. In practice, this would correspond to training our proposed approach with only the triplet and relative pose objectives. We name this approach Baseline-A since we train only the branch A. Another baseline would be to learn RGB-D modality invariant representations, i.e., similar features for RGB and depth images, which can then be used to match RGB images to depth renderings from CAD models. We implement this by training our model to align local feature representations for RGB and depth images, which is similar in spirit to and an improved version of ZDDA [27], a domain adaptation approach that maps RGB and depth modalities to the same latent space. We name this approach Baseline-ZDDA.

### Table 1: Comparison between ours and a supervised approach on the chair category. The \( \text{MedErr} \) is shown in radians.

| Metric | \( \text{Acc}_\pi \uparrow \) | \( \text{MedErr} \downarrow \) |
|--------|---------------------|---------------------|
| Mousavian et al. [25] | 23.9 | 0.9 |
| Proposed | 31.1 | 0.9 |
widely used Pascal3D+ [40] dataset. We run this method on our chair test set and report results in Table 1. The results suggest that [25] under-performs when applied on a new dataset, while our approach achieves higher $Acc_z$, even though it does not explicitly require the 3D pose annotations on the real images during training.

4.2. Category pose estimation

In this section, we report results of pose estimation experiments on the bed, chair, and desk categories. We first show, in Figure 5 (top row), qualitative keypoint prediction results on test images, where we see keypoint predictions that generally satisfy our intuition of good keypoints. We then adopt the testing protocol described above to report quantitative pose estimation results for test RGB images. Performance analysis is shown in Table 2 for the three object categories. As can be noted from the results, our proposed model generally achieves higher accuracy when compared to the baseline approaches. In particular, the improvements over Baseline-A suggests that that keypoint and
representation modality adaptation enforced in our model is critical. Furthermore, the improvements over Baseline-ZDDA suggests that simply performing modality adaptation for the RGB and depth features is not sufficient, and learning keypoints and view-invariant representations, as is done in our method, is important to achieve good performance.

4.3. Model transferability

In this section, we demonstrate the transfer capability, where the goal is to show that a model trained according to the proposed approach can generalize well to category instances not seen during training. This is key to practical usability of the approach since we cannot possibly have relevant CAD models of all instances of interest during training.

To this end, the following experimental protocol is used: during training, quadruplets are sampled from a subset of the available instances for each category, and test on RGB images corresponding to all other instances. For instance, for the bed category, we use 10 instances for training and 9 instances for testing. Similarly, for chair and desk, we use 111 and 12 instances respectively for training and the rest for testing. During testing, we use the same protocol as above. We present qualitative keypoint predictions in Figure 5 (middle row) and report quantitative performance in Table 3. As can be seen from the results, our model shows good transferability, providing (a) a similar level of detail in the predicted keypoints as before, (b) improved accuracy when compared to the baselines, and (c) absolute accuracies that are not too far from those in Table 2.

4.4. Framework flexibility

While the results reported above use RGB-D pairs from the Pix3D dataset to train our model, in principle, our approach can be used in conjunction with other datasets that provide aligned RGB-D pairs as well. Such capability will naturally make it easier to train models with our framework, leading to improved framework flexibility. To demonstrate this aspect, we train our model as before, but now for input to branches C and D, we use aligned RGB-D pairs from the NYUv2 [30] dataset. Since these pairs contain noisy depth images from a real depth sensor, we synthetically apply realistic noise on the clean rendered depth images, used for branches A and B, using DepthSynth [28]. This ensures that branches A, B, and C still receive the same modality as input. A quadruplet training input for this experiment is shown in Figure 7. Note that as before, we do not use any pose annotations for the RGB images as part of training our model. Figure 5 (bottom row), shows some keypoint prediction results on test data from Pix3D. In Table 4, we report quantitative azimuth, elevation, and in-plane rotation accuracy values for the bed category. As can be noted from the results, while the numbers are not as high as those in Table 2, which is expected, they are not substantially different from the previous experiments. In fact, in some cases (e.g. azimuth for chair), the numbers are higher than Baseline-ZDDA in Table 2, where we use Pix3D aligned data for training. These results, along with those in the previous section, clearly show the potential and promise of our approach in learning generalizable models for estimating object poses in RGB images.

| Metric | Azimuth (radians) | Elevation | In-plane |
|--------|-------------------|-----------|----------|
| Bed    | 54.7              | 47.5      | 36.3     |
| Chair  | 36.3              | 47.4      | 27.6     |
| Desk   | 50.0              | 48.0      | 29.0     |

Table 4: Azimuth, elevation, and in-plane accuracy results when trained with the NYUv2 dataset. Details for this experiment can be found in section 4.4.

5. Conclusions

We proposed a new framework for 3D object pose estimation in RGB images, which does not require either textured CAD models or 3D pose annotations for RGB images during training. We achieve this by means of a novel end-to-end learning pipeline that guides our model to discover keypoints in the depth modality optimized for relative pose estimation as well as transfer keypoints and representations to
the RGB modality. Our results from a suite of experiments on the Pix3D dataset demonstrate its potential in learning generalizable models for pose estimation.

References

[1] M. Aubry, D. Maturana, A. A. Efros, B. C. Russell, and J. Sivic. Seeing 3d chairs: exemplar part-based 2d-3d alignment using a large dataset of cad models. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 3762–3769, 2014. 2

[2] A. Bansal, B. Russell, and A. Gupta. Marr revisited: 2d-3d alignment via surface normal prediction. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 5965–5974, 2016. 2, 6

[3] Blender. https://www.blender.org/. 3

[4] E. Brachmann, A. Krull, F. Michel, S. Gumhold, J. Shotton, and C. Rother. Learning 6d object pose estimation using 3d object coordinates. In European conference on computer vision, pages 536–551. Springer, 2014. 2

[5] A. Collet, M. Martinez, and S. S. Srinivasa. The moped framework: Object recognition and pose estimation for manipulation. The International Journal of Robotics Research, 30(10):1284–1306, 2011. 2

[6] A. Doumanoglou, R. Kouskouridas, S. Malassiotis, and T.-K. Kim. Recovering 6d object pose and predicting next-best-view in the crowd. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 3583–3592, 2016. 2

[7] G. Georgakis, S. Karanam, Z. Wu, J. Ernst, and J. Košecká. End-to-end learning of keypoint detector and descriptor for pose invariant 3d matching. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 1965–1973, 2018. 3

[8] S. Gupta, P. Arbeláez, R. Girshick, and J. Malik. Inferring 3d object pose in rgb-d images. arXiv preprint arXiv:1502.04652, 2015. 3

[9] S. Hinterstoisser, S. Holzer, C. Cagniart, S. Ilic, Konolige, N. Navab, and V. Lepetit. Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes. In Computer Vision (ICCV), 2011 IEEE International Conference on, pages 858–865. IEEE, 2011. 2

[10] G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531, 2015. 5

[11] T. Hodan, F. Michel, E. Brachmann, W. Kehl, A. G. Buch, D. Kraft, B. Drost, J. Vidal, S. Ihrke, X. Zabulis, S. Caner, M. Fabian, T. Federico, K. Tae-Kyun, M. Jiri, and R. Carsten. Bop: Benchmark for 6d object pose estimation. In ECCV, 2018. 2

[12] M. Hueting, P. Reddy, V. Kim, N. Carr, E. Yumer, and N. Mitra. Seethrough: finding objects in heavily occluded indoor scene images. CoRR abs/1710.10473, 2017. 3

[13] H. Izadinia, Q. Shan, and S. M. Seitz. Im2cad. In Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on, pages 2422–2431. IEEE, 2017. 2

[14] M. Janner, J. Wu, T. D. Kulkarni, I. Yildirim, and J. Tenenbaum. Self-supervised intrinsic image decomposition. In Advances in Neural Information Processing Systems, pages 5936–5946, 2017. 1

[15] W. Kehl, F. Manhardt, F. Tombari, S. Ilic, and N. Navab. Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again. In Proceedings of the International Conference on Computer Vision (ICCV 2017), Venice, Italy, pages 22–29, 2017. 2

[16] W. Kehl, F. Milletari, F. Tombari, S. Ilic, and N. Navab. Deep learning of local rgb-d patches for 3d object detection and 6d pose estimation. In European Conference on Computer Vision, pages 205–220. Springer, 2016. 2

[17] A. Kundu, Y. Li, and J. M. Rehg. 3d-rcnn: Instance-level 3d object reconstruction via render-and-compare. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017. 3

[18] C. Li, M. Z. Zia, Q.-H. Tran, X. Yu, G. D. Hager, and M. Chandraker. Deep supervision with shape concepts for occlusion-aware 3d object parsing. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018. 2

[19] J. J. Lim, H. Pirsiavash, and A. Torralba. Parsing ikea objects: Fine pose estimation. In Proceedings of the IEEE International Conference on Computer Vision, pages 2992–2999, 2013. 2

[20] W. Liu, D. Anguelov, D. Erhan, C.-Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector. In European conference on computer vision, pages 21–37. Springer, 2016. 2

[21] D. G. Lowe. Distinctive image features from scale-invariant keypoints. IJCV, 60(2):91–110, 2004. 2

[22] S. Mahendran, H. Ali, and R. Vidal. 3d pose regression using convolutional neural networks. In IEEE International Conference on Computer Vision, volume 1, page 4, 2017. 2

[23] F. Massa, B. C. Russell, and M. Aubry. Deep exemplar 2d-3d detection by adapting from real to rendered views. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 6024–6033, 2016. 2

[24] R. Mottaghi, Y. Xiang, and S. Savarese. A coarse-to-fine model for 3d pose estimation and sub-category recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 418–426, 2015. 2

[25] A. Mousavian, D. Anguelov, J. Flynn, and J. Košecká. 3d bounding box estimation using deep learning and geometry. In Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on, pages 5632–5640. IEEE, 2017. 2, 6

[26] G. Pavlakos, X. Zhou, A. Chan, K. G. Derpanis, and K. Daniilidis. 6-dof object pose from semantic keypoints. In Robotics and Automation (ICRA), 2017 IEEE International Conference on, pages 2011–2018. IEEE, 2017. 3

[27] K.-C. Peng, Z. Wu, and J. Ernst. Zero-shot deep domain adaptation. In European Conference on Computer Vision, pages 793–810. Springer, 2018. 6

[28] B. Planche, Z. Wu, K. Ma, S. Sun, S. Kluckner, O. Lehmann, T. Chen, A. Hutter, S. Zakharov, H. Kosch, et al. Depthsynth: Real-time realistic synthetic data generation from cad models for 2.5 d recognition. In 3D Vision (3DV), 2017 International Conference on, pages 1–10. IEEE, 2017. 8
In Figures 8 and 9, we show additional illustrations of rendering the poses estimated by our proposed method on several RGB images from our test set of the Pix3D dataset. These results correspond to the experiment described in Section 4.2 of the main text.
Figure 8: Additional illustrations of rendered estimated poses on test RGB images from the Pix3D dataset for the category pose estimation experiments.

Figure 9: Additional illustrations of rendered estimated poses on test RGB images from the Pix3D dataset for the category pose estimation experiments.