Location Field Descriptors: Single Image 3D Model Retrieval in the Wild

Alexander Grabner[1] Peter M. Roth[1] Vincent Lepetit[2,1]

[1]Institute of Computer Graphics and Vision, Graz University of Technology, Austria

[2]Laboratoire Bordelais de Recherche en Informatique, University of Bordeaux, France

{alexander.grabner,pmroth,lepetit}@icg.tugraz.at

Abstract

We present Location Field Descriptors, a novel approach for single image 3D model retrieval in the wild. In contrast to previous methods that directly map 3D models and RGB images to an embedding space, we establish a common low-level representation in the form of location fields from which we compute pose invariant 3D shape descriptors. Location fields encode correspondences between 2D pixels and 3D surface coordinates and, thus, explicitly capture 3D shape and 3D pose information without appearance variations which are irrelevant for the task. This early fusion of 3D models and RGB images results in three main advantages: First, the bottleneck location field prediction acts as a regularizer during training. Second, major parts of the system benefit from training on a virtually infinite amount of synthetic data. Finally, the predicted location fields are visually interpretable and unblackbox the system. We evaluate our proposed approach on three challenging real-world datasets (Pix3D, Comp, and Stanford) with different object categories and significantly outperform the state-of-the-art by up to 20% absolute in multiple 3D retrieval metrics.

1. Introduction

3D model retrieval from a single RGB image, as shown in Fig. 1, is a challenging but important task with applications in augmented reality, robotics, 3D printing, 3D scene understanding, and 3D scene modeling. Compared to reconstruction [13, 58], retrieval provides 3D models designed by humans which are rich in detail. Due to the growing number of large-scale 3D model databases, like ShapeNet [9] or 3D Warehouse[1], efficient image-based retrieval approaches have become a fundamental requirement.

Recent works address the retrieval task by directly mapping 3D models and RGB images to a common embedding space [32, 55]. However, previous approaches have a number of limitations in practice. First, the learned mapping is highly prone to overfitting, because training data in the form of RGB images with 3D model annotations is scarce [53, 61]. Second, systems purely trained on synthetic data do not generalize to real data due to the domain gap between RGB images and RGB renderings [13, 36]. Finally, the black box characteristic of these systems makes it hard to understand why the approaches fail in certain scenarios.

Figure 1: Given a single RGB image, we retrieve a 3D model with accurate geometry for each object in the image from a previously seen or unseen 3D model database.

To overcome these limitations, we map 3D models and RGB images to a common low-level representation in the form of location fields from which we compute pose invariant 3D shape descriptors. Location fields [57, 61] are image-like representations that encode a 3D surface coordinate for each object pixel (see Fig. 2). In particular, we render location fields from 3D models and predict location fields from RGB images using a CNN. Then, instead of exhaustively comparing location fields from different viewpoints [25, 36], we compute pose invariant 3D shape descriptors in an embedding space optimized for retrieval from the location fields. Thus, we call our approach Location Field Descriptors.

Regarding 3D model retrieval, location fields have several advantages compared to other rendered representations: RGB renderings [13, 67] are subject to appearance variations which are irrelevant for the task caused by material, texture, and lighting. Texture-less gray-scale render-

[1]https://3dwarehouse.sketchup.com

ings [31, 36] are still affected by the scene lighting. Silhouettes [10] are not affected by such appearance variations but discard valuable 3D shape information. Depth [14, 69] and normal renderings [60, 66] capture 3D geometry but lose the relation to the 3D pose in the object's canonical coordinate system. In contrast, location fields explicitly present 3D shape and 3D pose information, as they establish correspondences between 2D object pixels and 3D coordinates on the object surface. Considering 3D shape, the dense 3D coordinates provide a partial reconstruction of the object geometry. Considering 3D pose, the object rotation and translation can be geometrically recovered from the 2D-3D correspondences using a P*n*P algorithm [8, 26].

The benefits of our approach are threefold: First, the intermediate location field prediction serves as a regularizing bottleneck which reduces the risk of overfitting in the case of limited training data compared to directly mapping to an embedding space. Second, major parts of the system benefit from training on a virtually infinite amount of synthetic data due to the early fusion of 3D models and RGB images. Third, the predicted location fields are visually interpretable and offer valuable insights in cases where the approach fails.

Finally, to demonstrate the benefits of our novel 3D model retrieval approach, we evaluate it on three challenging real-world datasets with different object categories: Pix3D [53] (*bed*, *chair*, *sofa*, *table*), Comp [61] (*car*), and Stanford [61] (*car*). We present quantitative as well as qualitative results and significantly outperform the state-of-the-art. To summarize, our main contributions are:

- We present the first method that uses location fields for pose invariant 3D model retrieval. Our approach is accurate, scalable, and interpretable.
- We outperform the state-of-the-art by up to 20% absolute in multiple 3D retrieval metrics given both previously seen and unseen 3D model databases.

2. Related Work

In this section, we discuss previous works in the fields of 3D coordinate regression and 3D model retrieval from a single RGB image.

2.1. Location Fields

Regressing 3D coordinates from 2D observations is a well-studied problem in computer vision [16]. While traditional approaches generate 3D point clouds from multi-view RGB images [51, 54], recent works predict unstructured 3D point clouds from a single RGB image using deep learning [12, 35, 38]. Complementary to these works, PointNet [42] and successors [29, 43] showed that even such unstructured 3D point clouds can be used to address various 3D vision tasks with deep learning.

In this work, however, we focus on predicting structured 3D point clouds in the form of location fields [57, 61]. A location field encodes a 3D surface coordinate for each object pixel. Thus, it is important to know which pixels belong to an object and which pixels belong to the background or another object [7, 8]. Recent works showed that deep learning techniques for instance segmentation [17] significantly increase the accuracy on this task [15, 26, 59]. However, until now location fields have only been used for 3D pose estimation, but not for 3D model retrieval or other tasks.

2.2. 3D Model Retrieval

A large number of previous works perform retrieval given a query 3D model [48, 49]. These methods either directly operate on 3D data, *e.g.*, in the form of voxel grids [41, 65], spherical maps [11], or point clouds [42], or process multi-view renderings of the query 3D model [5, 10, 41, 52] to compute a shape descriptor.

However, in this work, we focus on the much more challenging task of 3D model retrieval from a single RGB image [27]. One approach to address this task is to train a classifier which provides a 3D model for each fine-grained class on top of handcrafted [2] or learned [37] features extracted from an RGB image. This, in consequence, restricts the retrieval to 3D models seen during training.

The most popular strategy to overcome this limitation is to map 3D models and RGB images to a common embedding space in which retrieval is performed using distance-based matching [30]. In this case, the mapping, the embedding space, and the distance measure can be designed in a variety of ways.

Numerous works match features extracted from an RGB image against features extracted from multi-view RGB renderings to predict both shape and viewpoint. In this context, [3] uses a CNN trained for ImageNet classification [46] to extract features. [36] takes a similar approach, but additionally performs nonlinear feature adaption to overcome the domain gap between real and rendered RGB images. [22] and [25] use a CNN trained for object detection as a feature extractor. However, the CNNs used in these methods are not optimized for 3D model retrieval.

Thus, other approaches train mappings to predefined embedding spaces. [55] trains CNNs to map 3D models, RGB images, depth maps, and sketches to an embedding space based on text for cross-modal retrieval. [24] constructs a low-dimensional embedding space by performing PCA on 3D key points and maps 3D key points predicted using a CNN to that space for retrieval. [32] and [56] train a CNN to map RGB images to an embedding space computed from pairwise similarities between 3D models.

Instead of handcrafting an embedding space, an embedding space capturing 3D shape properties can be learned. [58] reconstructs voxel grids from RGB images of objects

using CNNs. The low-dimensional bottle-neck shape descriptor is also used for retrieval. [13] combines a 3D voxel encoder and an RGB image encoder with a shared 3D voxel decoder to perform reconstruction from a joint embedding. 3D model retrieval is performed by matching embeddings of voxel grids against those of RGB images.

Finally, recent approaches explicitly learn an embedding space which is optimized for 3D model retrieval. [67] uses a single CNN to map RGB images and RGB renderings to an embedding space which is optimized using a Euclidean distance-based lifted structure loss [40]. At test time, the distances between an embedding of an RGB image and embeddings of multi-view RGB renderings are averaged to compensate for the unknown object pose. [31] uses two CNNs to map RGB images and gray-scale renderings to an embedding space and optimizes a Euclidean distance-based Triplet loss [62]. Additionally, cross-view convolutions are employed to aggregate a sequence of multi-view renderings into a single descriptor to reduce the matching complexity. [14] also trains two CNNs, but maps RGB images and depth maps to a common space. In contrast to other approaches, the 3D pose of the object in the RGB image is explicitly estimated and used in the 3D model retrieval.

Compared to these approaches, we also learn an embedding space which is optimized for 3D model retrieval, but first predict location fields from RGB images and then compute pose invariant 3D shape descriptors from predicted and rendered location fields in an end-to-end trainable way.

3. Location Field Descriptors

Given a single RGB image and a 3D model database, we retrieve a 3D model for each object in the image, as shown in Fig. 3. For this purpose, we first generate location fields from 3D models and RGB images. We then compute pose invariant 3D shape descriptors from the locations fields. Finally, we match the descriptors to find the best 3D model.

3.1. Location Field Generation

The first step in our approach is to map 3D models and RGB images to a common low-level representation in the form of location fields. As illustrated in Fig. 2, a location field [57, 61] is an image-like representation that encodes a 3D surface coordinate for each object pixel. Compared to its reference RGB image, a location field has the same size and spatial resolution, but the three channels encode XYZ 3D coordinates in the canonical object coordinate system instead of RGB colors. Locations fields explicitly present 3D shape and 3D pose information, because they encode dense correspondences between 2D pixel locations and 3D surface coordinates. From these 2D-3D correspondences, the 3D pose can be geometrically recovered using a P*n*P algorithm [8, 26]. Additionally, location fields can also be interpreted as structured partial 3D point clouds.

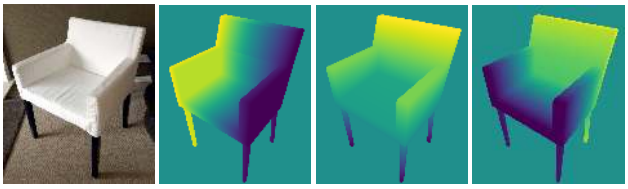


Figure 2: An image of an object and its location field. Location fields encode a 3D surface coordinate in the canonical object coordinate system for each object pixel. We show the three channels which correspond to the X, Y, and Z values of the 3D coordinates in separate images.

Location fields can be directly rendered from 3D models. For this purpose, we rasterize 3D meshes using OpenGL and implement a custom fragment shader which linearly interpolates per-vertex 3D coordinates along the triangles of a 3D mesh. Because the interpolated values describe 3D coordinates in the canonical object coordinate system, the relation to the inherent object orientation is preserved.

In order to generate location fields from RGB images, we need to detect objects in 2D and predict a location field of each object. For this purpose, we introduce a Location Field CNN (see Fig. 3) which extends the generalized Faster/Mask R-CNN framework [17, 45]. This generic multi-task framework includes a 2D object detection pipeline to perform per-image and per-object computations. In this way, we address multiple different tasks using a single end-to-end trainable network.

In the context of the generalized Faster/Mask R-CNN framework, each output branch provides a task-specific subnetwork with different structure and functionality. We introduce a dedicated output branch for estimating location fields alongside the existing object detection branches [61]. Similar to the mask branch [17], the location field branch performs region-based per-object computations: For each detected object, an associated spatial region of interest in the feature maps is aligned to a fixed size feature representation with a low spatial but high channel resolution using linear interpolation, *e.g.*, $14 \times 14 \times 256$. These aligned features serve as a shared input to the classification, mask and location field branches. Each branch is evaluated N times per image, where N is the number of detected objects.

Our location field branch uses a fully convolutional subnetwork to predict a tensor of 3D points at a resolution of $56 \times 56 \times 3$ from the shared aligned features. We also modify the mask branch to predict 2D masks at the same spatial resolution and use the predicted masks to threshold the tensor of 3D points to get low-resolution location fields. We experimentally found this approach to generate significantly higher accuracy location fields compared to directly regressing low-resolution location fields, which tends

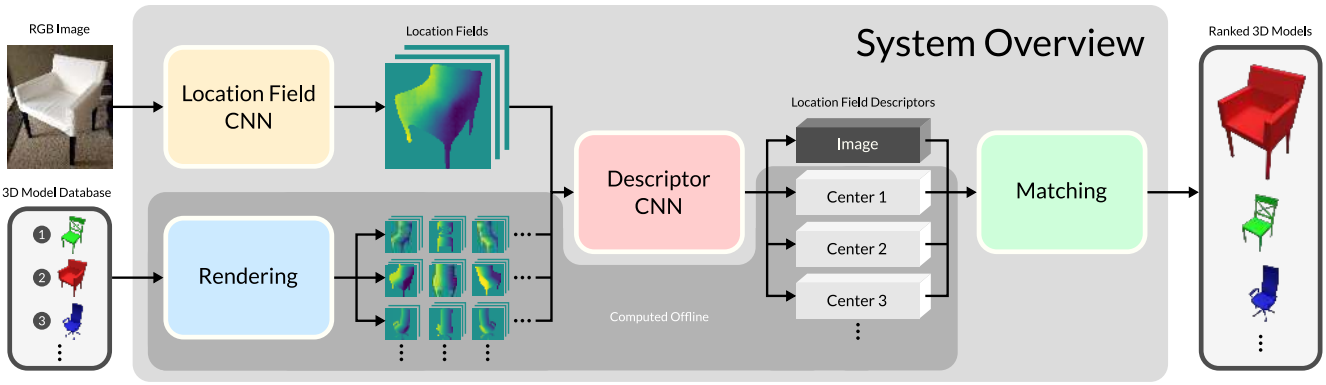


Figure 3: Overview of our approach. Given a single RGB image and a 3D model database, we use CNNs to predict a location field and a pose invariant location field descriptor for each object in the image. For each 3D model in the database, we learn a single center descriptor from multi-view location fields offline during training. Finally, we match location field descriptors predicted from the image against offline computed center descriptors to retrieve a ranked list of the 3D models in the database.

to predict over-smoothed 3D coordinates around the object silhouette. During training, we optimize our predicted location fields using the Huber loss [23].

The resulting low-resolution location fields can be upscaled and padded to obtain high-resolution location fields with the same spatial resolution as the input image. This is especially helpful in cases where our approach fails. A visual overlay of the input image and the predicted location fields is intuitively interpretable and offers valuable insights to why the system fails.

However, we compute pose invariant descriptors from the low-resolution location fields, because upscaling does not provide additional information but increases the computational workload. Moreover, the predicted low-resolution location fields are tightly localized crops, which reduces the complexity of the descriptor computation.

3.2. 3D Shape Descriptors

Instead of exhaustively comparing each predicted location field to multiple rendered location fields from different viewpoints [25, 36], we map location fields to pose invariant 3D shape descriptors in an embedding space. We refer to descriptors in this space as Location Field Descriptors.

For this purpose, we introduce a Descriptor CNN (see Fig. 3) which utilizes a dense connection pattern [21]. Similar to ResNets [18, 19], DenseNets [21] introduce skip-connections in the computational graph, but concatenate feature maps instead of adding them. The dense connection pattern encourages feature reuse throughout the network and leads to compact but expressive models. This architecture is well-suited for computing descriptors from location fields, because they already provide a high level of abstraction. Location fields are not affected by task irrelevant appearance variations caused by color, material, texture or lighting. The object is already segmented from the background and occlusions in the predicted location fields are resolved by the 2D mask used for thresholding the tensor of 3D points. In fact, even the raw 3D coordinates provide useful matching attributes, for example by aligning query and test point clouds using ICP [6]. Thus, extensive feature reuse within the Descriptor CNN is rational.

In order to learn an embedding space which is optimized for 3D model retrieval, we need to address two requirements. First, the embedding space has to be discriminative in terms of 3D models. Second, the computed descriptors have to be invariant to the 3D pose of the object in the location field. We jointly address both requirements by learning a representative center descriptor for each 3D model, as shown in Fig. 3. For this purpose, we train the Descriptor CNN to map location fields of a 3D model from different viewpoints close to its corresponding center descriptor. At the same time, we make sure that all center descriptors are discriminatively distributed in the embedding space. Thus, during training, we penalize the distances between location field descriptors and center descriptors in a way that each location field descriptor and its corresponding center descriptor are pulled closer together, while all center descriptors are pulled further apart. This approach resembles a nonlinear discriminant analysis [47], in which the intra-class variance is minimized, while the inter-class variance is maximized to train more discriminative embeddings.

In particular, we build on the ideas of Center loss [63] and Triplet-Center loss [20] to optimize our embedding space. The Center loss

$$L_C = \sum_{i=1}^{N} D(f_i, c_{y_i}) \tag{1}$$

minimizes the distance $D(f_i, c_{y_i})$ between a location field descriptor f_i and its corresponding center descriptor c_{y_i}. In

this case, y_i is the index of the corresponding 3D model and N denotes the number of samples. For the distance function $D(\cdot)$, we use the Huber distance [23]. In contrast, the Triplet-Center loss

$$L_{\text{TC}} = \sum_{i=1}^{N} \max\left(0\,, D(f_i, c_{y_i}) + m - \min_{j \neq y_i} D(f_i, c_j)\right) \quad (2)$$

enforces the same distance $D(f_i, c_{y_i})$ to be smaller than the distance between a location field descriptor and its closest non-corresponding center descriptor $\min_{j \neq y_i} D(f_i, c_j)$ by at least the margin m.

As a consequence, the Center loss only minimizes intra-class variance, while the Triplet-Center loss aims at both minimizing intra-class variance and maximizing inter-class variance. In many cases, however, the Triplet-Center loss fails to achieve these goals. Instead, it learns degenerated clusterings, because the optimization criterion does not guarantee the desired properties [28]. Thus, we employ a combination of Center loss and Triplet-Center loss in our Descriptor loss

$$L_{\text{D}} = L_{\text{softmax}} + \alpha L_{\text{C}} + \beta L_{\text{TC}} \quad (3)$$

to achieve both low intra-class variance and high inter-class variance [28, 64]. In practice, these losses are combined with a softmax loss L_{softmax} to learn more discriminative embeddings than classification alone [20, 63]. The parameters α and β control the impact of the different loss terms.

We want to emphasize that the center descriptors are not fixed, but learned during training. In fact, the center descriptors are trainable weights of the Descriptor CNN in our implementation. Also, even though we optimize a triplet criterion, we do not require location field triplets as training input. Only a single location field and its corresponding 3D model index y_i are needed. The center descriptors required for the Triplet loss are sampled within the Descriptor CNN. Additionally, hard triplet mining [50] is less important, because we always sample the closest non-corresponding center descriptor and also employ Center and softmax losses.

We jointly train the Descriptor CNN on predicted and rendered location fields. This is a major advantage compared to previous approaches that directly map to an embedding space, because training data in the form of RGB images with 3D model annotations is limited. In contrast, we benefit from training on a virtually infinite amount of synthetic data. Additionally, the intermediate location field prediction serves as a regularizing bottleneck and reduces the risk of overfitting, because regressing location fields is more difficult than computing embeddings.

Since there is a domain gap between predicted and rendered location fields, we perform Feature Mapping [44]. We use a residual block [18, 19] to map location field descriptors from the predicted to the rendered domain. Thus, the training input either consists of pairs of corresponding predicted and rendered location fields or single location fields, and a 3D model index y_i in both cases. In the case of pairs, we compute an additional Feature Mapping loss between corresponding feature-mapped predicted and rendered location field descriptor using the Huber distance [23].

To perform retrieval from a previously unseen 3D model database, we generate center descriptors without retraining. For each unseen 3D model, we render 100 location fields under different 3D poses, compute their embeddings using the Descriptor CNN and average them to obtain a new center descriptor. Alternatively, we can retrain the Descriptor CNN by incorporating the new 3D models as additional rendered location fields. In any case, the center descriptors are computed offline which results in fast inference.

During inference, we only need to process RGB images using our CNNs, since the center descriptors have already been computed offline. For each RGB image, we evaluate the Location Field CNN once and the Descriptor CNN N times to compute location field descriptors, where N is the number of detected objects. We then match each computed location field descriptors against all center descriptors and generate a ranked list of 3D models based on the Euclidean distance between the descriptors, as shown in Fig. 3.

Finally, the entire system, *i.e.*, the Location Field CNN and the Descriptor CNN, is end-to-end trainable. The system loss is a combination of our Location Field loss, our Descriptor loss, our Feature Mapping loss, and the Detection losses of the generalized Faster/Mask R-CNN framework.

4. Experimental Results

To demonstrate the benefits of Location Field Descriptors, we evaluate our approach on three challenging real-world datasets with different object categories: Pix3D [53] (*bed*, *chair*, *sofa*, *table*), Comp [61] (*car*), and Stanford [61] (*car*). Details on the datasets, the implementation, and the evaluation are provided in the **supplementary material**.

In particular, we provide quantitative results for 3D model retrieval from seen and unseen databases in comparison to the state-of-the-art in Sec. 4.1, present qualitative results of our approach in Sec. 4.2, and perform an ablation study in Sec. 4.3. For our quantitative evaluation, we use the following well-established metrics:

Detection. We report the detection accuracy $Acc_{D_{0.5}}$ which gives the percentage of objects for which the intersection over union between the ground truth 2D bounding box and the predicted 2D bounding box is larger than 50% [68]. This metric is an upper bound for other Acc metrics since we do not make blind predictions.

Retrieval Accuracy. We evaluate the retrieval accuracies $Acc_{\textit{Top-1}}$ and $Acc_{\textit{Top-10}}$ which give the percentage of objects for which the ground truth 3D model equals the top ranked

Method	Dataset	Category	$Acc_{D_{0.5}}$	seen 3D models				unseen 3D models	
				$Acc_{Top\text{-}1}$	$Acc_{Top\text{-}10}$	d_{HAU}	d_{IOU}	d_{HAU}	d_{IOU}
[3]	Pix3D	bed	99.0%	19.4%	46.6%	0.0821	0.3397	0.0960	0.2487
[14]				35.1%	83.2%	0.0385	0.5598	0.0577	0.3013
Ours				**64.4%**	**89.0%**	**0.0152**	**0.8074**	**0.0448**	**0.3490**
[3]	Pix3D	chair	91.5%	17.3%	49.1%	0.0559	0.3027	0.0843	0.1334
[14]				41.3%	73.9%	0.0305	0.5469	0.0502	0.1965
Ours				**58.1%**	**81.8%**	**0.0170**	**0.7169**	**0.0375**	**0.2843**
[3]	Pix3D	sofa	96.9%	21.7%	52.2%	0.0503	0.3824	0.0590	0.3493
[14]				44.1%	89.8%	0.0197	0.7762	0.0294	0.6178
Ours				**67.0%**	**94.4%**	**0.0075**	**0.9028**	**0.0178**	**0.7472**
[3]	Pix3D	table	91.2%	12.0%	34.2%	0.1003	0.1715	0.1239	0.1047
[14]				33.9%	66.1%	0.0607	0.4500	0.0753	0.1730
Ours				**53.3%**	**80.1%**	**0.0288**	**0.6383**	**0.0482**	**0.2573**
[3]	Pix3D	*mean*	94.6%	17.6%	45.5%	0.0722	0.2991	0.0908	0.2090
[14]				38.6%	78.3%	0.0374	0.5832	0.0531	0.3222
Ours				**60.7%**	**86.3%**	**0.0171**	**0.7663**	**0.0370**	**0.4095**
[3]	Comp	car	99.9%	2.4%	18.2%	0.0207	0.7224	0.0271	0.6344
[14]				10.2%	36.9%	0.0158	0.7805	0.0194	0.7230
Ours				**20.5%**	**58.0%**	**0.0133**	**0.8142**	**0.0165**	**0.7707**
[3]	Stanford	car	99.6%	3.7%	20.1%	0.0198	0.7169	0.0242	0.6526
[14]				11.3%	42.2%	0.0153	0.7721	0.0183	0.7201
Ours				**29.5%**	**69.4%**	**0.0110**	**0.8352**	**0.0150**	**0.7744**

Table 1: Experimental results on the Pix3D, Comp, and Stanford datasets. We provide results for 3D model retrieval from both seen (in training dataset) and unseen (ShapeNet) 3D model databases given unseen test images. We significantly outperform the state-of-the-art in all metrics and datasets. A detailed discussion of the reported numbers is presented in Sec. 4.1.

3D model (*Top-1*) [14], or is in the top ten ranked 3D models (*Top-10*) [67]. These metrics can only be provided if the ground truth 3D model is in the retrieval database.

Hausdorff Distance. We compute a modified Hausdorff distance [1, 4]

$$d_{\text{H}} = \frac{1}{|\mathcal{X}| + |\mathcal{Y}|} \Big(\sum_{\boldsymbol{x} \in \mathcal{X}} \min_{\boldsymbol{y} \in \mathcal{Y}} D(\boldsymbol{x}, \boldsymbol{y}) + \sum_{\boldsymbol{y} \in \mathcal{Y}} \min_{\boldsymbol{x} \in \mathcal{X}} D(\boldsymbol{y}, \boldsymbol{x}) \Big) \tag{4}$$

between the ground truth 3D model $\mathcal{X}$ and the retrieved 3D model $\mathcal{Y}$. For each vertex $\boldsymbol{x} \in \mathcal{X}$ and $\boldsymbol{y} \in \mathcal{Y}$, we calculate the Euclidean distance $D(\cdot)$ to the closest vertex from the other 3D model and compute the mean over both sets. Before computing d_{H}, we regularly resample each 3D model. We report the mean modified Hausdorff distance for all detected objects (d_{HAU}). Since all 3D models are consistently aligned, the score is in the interval $[0, \sqrt{2}]$ (lower is better).

3D Intersection Over Union. We compute the 3D intersection over union between a voxelization of the ground truth 3D model and a voxelization of the retrieved 3D model [56]. For this purpose, we voxelize 3D models using `binvox` [39] with a resolution of $128 \times 128 \times 128$. We report the mean 3D IOU for all detected objects (d_{IOU}). The score is in the interval $[0, 1]$ (higher is better).

4.1. Comparison to the State-of-the-Art

We are the first to present results for 3D model retrieval on Pix3D, Comp, and Stanford. For this purpose, we compare our approach to a baseline method [3] and a state-of-the-art method [14]. Since [3] and [14] assume that objects are already detected in 2D, we use the detections given by our approach for a fair comparison. The results are summarized in Table 1, where we significantly outperform the state-of-the-art in all metrics and datasets.

First of all, we correctly detect 95% of all objects in the images on average ($Acc_{D_{0.5}}$), since object detection is tightly integrated into our approach. In fact, our Location Field CNN is initialized with weights trained for instance segmentation [17] on COCO [34] and all evaluated categories are present in COCO.

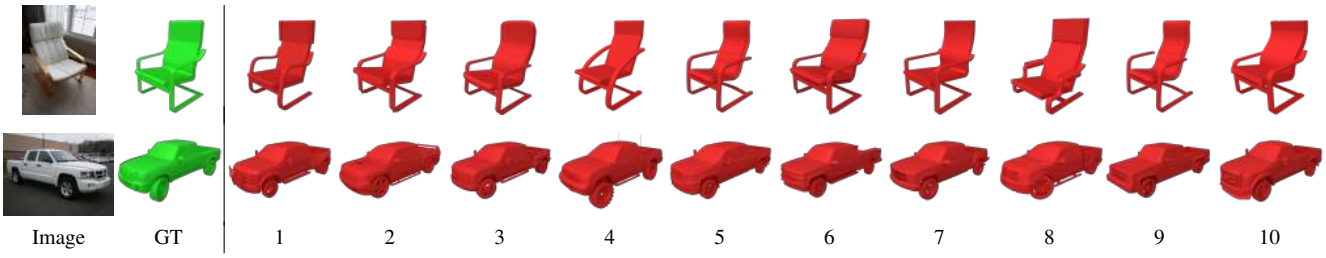


Figure 4: Qualitative results for 3D model retrieval from ShapeNet. From left to right, we show the input image, the ground truth 3D model and the top ten ranked 3D models. The overall 3D shape of the retrieved models is consistent and accurate.

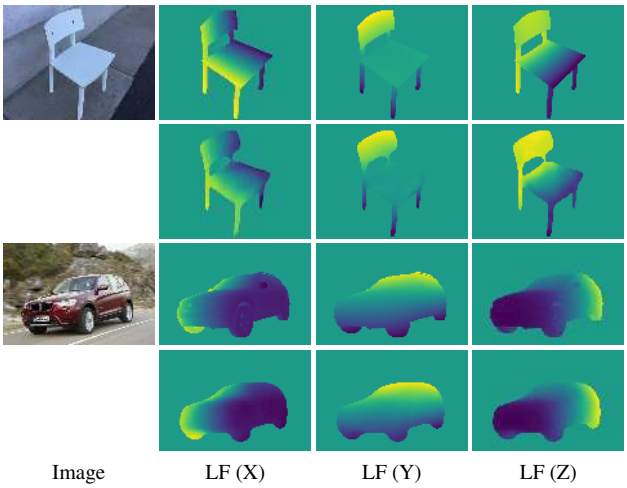


Figure 5: Qualitative examples of our predicted location fields. For each example image, the top row shows the ground truth and the bottom row shows our prediction. The overall 3D shape is recovered well, but fine-grained details like the side mirrors of the car are missed.

Next, we evaluate two different retrieval setups: First, we use all 3D models from the respective dataset as a 3D model database for retrieval (*seen 3D models*). In this case, we retrieve the correct 3D model ($Acc_{Top\text{-}1}$) for more than 60% of all test samples on average on Pix3D. This is a significant improvement of more than **20% absolute** compared to the state-of-the-art. Also, the retrieval accuracy quickly raises if we consider the top ten ranked 3D models ($Acc_{Top\text{-}10}$).

In contrast, the retrieval accuracy on Comp and Stanford is significantly lower for all evaluated methods. This is due to the significantly smaller variation in the overall shape of *cars* compared to *chairs*, for example. Thus, many 3D models of *cars* have a similar appearance in multiple 3D poses and can only be discriminated by extremely fine-grained details like wheel rims or radiator grill structure. Such pixel-level information is usually discarded by CNNs.

However, by analyzing the mesh similarity between the ground truth 3D model and the top retrieved 3D model (d_{HAU} and d_{IOU}), we observe consistent high performance across all datasets and categories. To put the reported numbers in perspective, we compute the mean of the modified Hausdorff distance (0.1236) and the 3D IOU (0.0772) for all pairs of 3D models in the training datasets. These numbers represent the accuracy for picking a random 3D model. For both metrics, the mesh similarity of our retrieved 3D model is around **10 times** better compared to picking a random 3D model. Additionally, we significantly outperform the state-of-the-art by up to **50% relative** considering d_{HAU}.

Second, we perform retrieval from previously unseen 3D models from ShapeNet [9] (*unseen 3D models*). Since the correct 3D model is not in the database in this case, the achievable performance is limited. Thus, the reported numbers are slightly worse compared to retrieval from previously seen 3D models. Still, the performance is much better compared to picking a random 3D model. In fact, for some categories, *e.g.*, Stanford *cars*, our approach retrieves more accurate 3D models from an **unseen** database than the state-of-the-art from a database **seen** during training.

4.2. Qualitative Results

The quantitative performance of our approach is also reflected in our qualitative results. First, Fig. 5 shows examples of our predicted location fields. We upscale and pad the predicted location fields to match the input image resolution. The overall 3D shape is recovered well in the location fields, but fine-grained details like the side mirrors of the car are missed.

Next, Fig. 4 shows qualitative results for 3D model retrieval from ShapeNet. Considering the top ten ranked 3D models, we observe that the retrieved models have a consistent and accurate overall 3D shape and geometry.

Finally, Fig. 6 presents examples for 3D model retrieval from both seen and unseen databases. In addition, we show that location fields provide all relevant information to also compute the 3D pose of objects. For this purpose, we sample 2D-3D correspondences from the location fields and solve a P*n*P problem during inference. The projections onto the image show that both our retrieved 3D models and our computed 3D poses are highly accurate. More qualitative results are shown in the **supplementary material**.

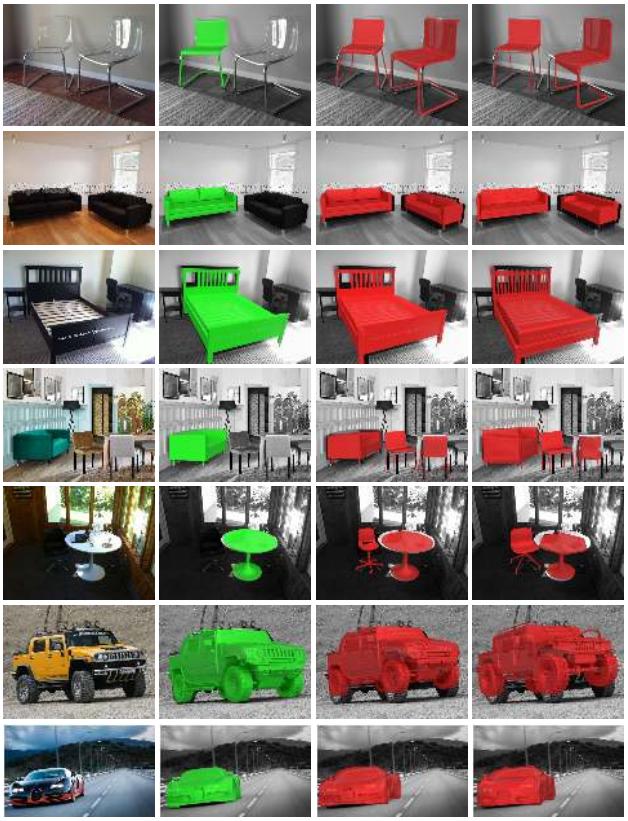


Figure 6: Qualitative results for 3D pose estimation and 3D model retrieval from both seen and unseen databases. We project the retrieved 3D model onto the image using a 3D pose computed from the predicted location field by solving a P*n*P problem. For the ground truth 3D model, we use the ground truth 3D pose. In fact, location fields provide all relevant information to jointly address both tasks.

4.3. Ablation Study

To understand which aspects of our approach are crucial for performance, we conduct an ablation study. For this purpose, we perform experiments on Pix3D, which is the most challenging dataset, because it provides multiple categories and has the largest variation in object scale and pose. We report the mean performance across all categories in Table 2.

If we train our approach without synthetic data, *i.e.*, train our Descriptor CNN purely on predicted location fields, the performance decreases significantly. Since training data is limited, we do not see location fields from many different 3D poses during training in this scenario.

Next, if we predict location fields at half of our proposed resolution ($28 \times 28 \times 3$) the performance drops significantly. In this case, fine-grained structures, *e.g.*, thin legs of a chair, cannot be recovered due to the limited spatial resolution.

Optimizing a pure softmax loss without our proposed combination of Center loss [63] and Triplet-Center loss [20] results in a small performance decrease. This shows that our proposed Descriptor loss (see Eq. 5) indeed learns more discriminative embeddings than classification alone.

Method	$Acc_{Top\text{-}1}$	d_{HAU}	d_{IOU}
Ours w/o synthetic data	55.0%	0.0219	0.7156
Ours half-res LFs	58.7%	0.0204	0.7370
Ours w/o (T)CL [20, 63]	59.9%	0.0175	0.7621
Ours w/o Mapping [44]	60.0%	0.0174	0.7630
Ours multi-view	**60.9%**	0.0173	**0.7686**
Ours	60.7%	**0.0171**	0.7663

Table 2: Ablation study of our method on the Pix3D dataset. Exploiting synthetic data in the form of rendered location fields during training and employing location fields with sufficient resolution to capture thin structures are the most important aspects for increasing performance.

Training without Feature Mapping [44] only slightly decreases performance. This is in part due to the fact that we also address the domain gap by aggressively augmenting and degenerating rendered location fields during training of our Descriptor CNN to simulate predicted location fields.

Finally, if we do not use our learned center descriptors but multi-view descriptors for matching, the performance almost remains the same. In this case, we match against 100 descriptors computed from location fields rendered under different 3D poses instead of a single center descriptor for each 3D model. This exhaustive comparison has a much higher computational complexity than our proposed approach. In fact, using center descriptors is not only significantly faster but also achieves better performance considering d_{HAU}. This experiment confirms that our approach indeed learns pose invariant 3D shape descriptors.

5. Conclusion

Learning a common embedding of 3D models and RGB images for single image 3D model retrieval is difficult due to limited training data and the domain gap between real and synthetic data. For this purpose, we map 3D models and RGB images to a common low-level representation in the form of location fields from which we compute pose invariant 3D shape descriptors. In this way, we bridge the domain gap and benefit from training on synthetic data. We evaluate our proposed approach on three challenging real-world datasets (Pix3D, Comp, and Stanford) and significantly outperform the state-of-the-art by up to 20% absolute.

Acknowledgement This work was supported by the Christian Doppler Laboratory for Semantic 3D Computer Vision, funded in part by Qualcomm Inc. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research.

References

[1] N. Aspert, D. Santa-Cruz, and T. Ebrahimi. Mesh: Measuring Errors between Surfaces using the Hausdorff Distance. In *International Conference on Multimedia and Expo*, pages 705–708, 2002.

[2] M. Aubry, D. Maturana, A. Efros, B. Russell, and J. Sivic. Seeing 3D Chairs: Exemplar Part-Based 2D-3D Alignment Using a Large Dataset of CAD Models. In *Conference on Computer Vision and Pattern Recognition*, pages 3762–3769, 2014.

[3] M. Aubry and B. Russell. Understanding Deep Features with Computer-Generated Imagery. In *Conference on Computer Vision and Pattern Recognition*, pages 2875–2883, 2015.

[4] S. Bai, X. Bai, W. Liu, and F. Roli. Neural Shape Codes for 3D Model Retrieval. *Pattern Recognition Letters*, 65(1):15–21, 2015.

[5] S. Bai, X. Bai, Z. Zhou, Z. Zhang, and L. Jan Latecki. Gift: A Real-Time and Scalable 3D Shape Search Engine. In *Conference on Computer Vision and Pattern Recognition*, pages 5023–5032, 2016.

[6] P. Besl and N. McKay. Method for Registration of 3-D Shapes. In *Sensor Fusion IV: Control Paradigms and Data Structures*, pages 586–607. International Society for Optics and Photonics, 1992.

[7] 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, 2014.

[8] E. Brachmann, F. Michel, A. Krull, M. Ying Yang, S. Gumhold, and C. Rother. Uncertainty-Driven 6D Pose Estimation of Objects and Scenes from a Single RGB Image. In *Conference on Computer Vision and Pattern Recognition*, pages 3364–3372, 2016.

[9] A. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu. Shapenet: An Information-Rich 3D Model Repository. Technical report, Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015.

[10] D.-Y. Chen, X.-P. Tian, Y.-T. Shen, and M. Ouhyoung. On Visual Similarity Based 3D Model Retrieval. In *Computer Graphics Forum*, pages 223–232, 2003.

[11] C. Esteves, C. Allen-Blanchette, A. Makadia, and K. Daniilidis. Learning SO(3) Equivariant Representations with Spherical CNNs. In *European Conference on Computer Vision*, pages 52–68, 2018.

[12] H. Fan, H. Su, and L. Guibas. A Point Set Generation Network for 3D Object Reconstruction from a Single Image. In *Conference on Computer Vision and Pattern Recognition*, pages 605–613, 2017.

[13] R. Girdhar, D. Fouhey, M. Rodriguez, and A. Gupta. Learning a Predictable and Generative Vector Representation for Objects. In *European Conference on Computer Vision*, pages 484–499, 2016.

[14] A. Grabner, P. M. Roth, and V. Lepetit. 3D Pose Estimation and 3D Model Retrieval for Objects in the Wild. In *Conference on Computer Vision and Pattern Recognition*, pages 3022–3031, 2018.

[15] A. Grabner, P. M. Roth, and V. Lepetit. GP^2C: Geometric Projection Parameter Consensus for Joint 3D Pose and Focal Length Estimation in the Wild. In *International Conference on Computer Vision*, 2019.

[16] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2003.

[17] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask R-CNN. In *International Conference on Computer Vision*, pages 2980–2988, 2017.

[18] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. In *Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[19] K. He, X. Zhang, S. Ren, and J. Sun. Identity Mappings in Deep Residual Networks. In *European Conference on Computer Vision*, pages 630–645, 2016.

[20] X. He, Y. Zhou, Z. Zhou, S. Bai, and X. Bai. Triplet-Center Loss for Multi-View 3D Object Retrieval. In *Conference on Computer Vision and Pattern Recognition*, pages 1945–1954, 2018.

[21] G. Huang, Z. Liu, L. van der Maaten, and K. Weinberger. Densely Connected Convolutional Networks. In *Conference on Computer Vision and Pattern Recognition*, pages 2261–2269, 2017.

[22] S. Huang, S. Qi, Y. Zhu, Y. Xiao, Y. Xu, and S.-C. Zhu. Holistic 3D Scene Parsing and Reconstruction from a Single RGB Image. In *European Conference on Computer Vision*, pages 187–203, 2018.

[23] P. Huber. Robust Estimation of a Location Parameter. *The Annals of Mathematical Statistics*, 35(1):73–101, 1964.

[24] M. Hueting, P. Reddy, E. Yumer, V. Kim, N. Carr, and N. Mitra. SeeThrough: Finding Objects in Heavily Occluded Indoor Scene Images. In *International Conference on 3D Vision*, pages 267–276, 2018.

[25] H. Izadinia, Q. Shan, and S. Seitz. IM2CAD. In *Conference on Computer Vision and Pattern Recognition*, 2017.

[26] O. H. Jafari, S. K. Mustikovela, K. Pertsch, E. Brachmann, and C. Rother. iPose: Instance-Aware 6D Pose Estimation of Partly Occluded Objects. In *Asian Conference on Computer Vision*, 2018.

[27] A. Kanezaki, Y. Matsushita, and Y. Nishida. RotationNet: Joint Object Categorization and Pose Estimation using Multiviews from Unsupervised Viewpoints. In *Conference on Computer Vision and Pattern Recognition*, pages 5010–5019, 2018.

[28] M. Keller, Z. Chen, F. Maffra, P. Schmuck, and M. Chli. Learning Deep Descriptors with Scale-Aware Triplet Networks. In *Conference on Computer Vision and Pattern Recognition*, pages 2762–2770, 2018.

[29] R. Klokov and V. Lempitsky. Escape from Cells: Deep KD-Networks for the Recognition of 3D Point Cloud Models. In *International Conference on Computer Vision*, pages 863–872, 2017.

[30] B. Kulis. Metric Learning: A Survey. *Foundations and Trends in Machine Learning*, 5(4):287–364, 2013.

[31] T. Lee, Y.-L. Lin, H. Y. Chiang, M.-W. Chiu, W. Hsu, and P. Huang. Cross-Domain Image-Based 3D Shape Retrieval by View Sequence Learning. In *International Conference on 3D Vision*, pages 258–266, 2018.

[32] Y. Li, H. Su, C. R. Qi, N. Fish, D. Cohen-Or, and L. Guibas. Joint Embeddings of Shapes and Images via CNN Image Pu-

rification. *ACM Transactions on Graphics*, 34(6):234, 2015.

[33] T.-Y. Lin, P. Dollár, R. B. Girshick, K. He, B. Hariharan, and S. J. Belongie. Feature Pyramid Networks for Object Detection. In *Conference on Computer Vision and Pattern Recognition*, pages 2117–2125, 2017.

[34] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: Common Objects in Context. In *European Conference on Computer Vision*, pages 740–755, 2014.

[35] P. Mandikal, K. L. Navaneet, M. Agarwal, and R. V. Babu. 3D-LMNet: Latent Embedding Matching for Accurate and Diverse 3D Point Cloud Reconstruction from a Single Image. In *British Machine Vision Conference*, pages 55.1–55.12, 2018.

[36] F. Massa, B. Russell, and M. Aubry. Deep Exemplar 2D-3D Detection by Adapting from Real to Rendered Views. In *Conference on Computer Vision and Pattern Recognition*, pages 6024–6033, 2016.

[37] R. Mottaghi, Y. Xiang, and S. Savarese. A Coarse-To-Fine Model for 3D Pose Estimation and Sub-Category Recognition. In *Conference on Computer Vision and Pattern Recognition*, pages 418–426, 2015.

[38] K. L. Navaneet, P. Mandikal, M. Agarwal, and V. Babu. CAPNet: Continuous Approximation Projection for 3D Point Cloud Reconstruction using 2D Supervision. In *AAAI Conference on Artificial Intelligence*, 2019.

[39] F. Nooruddin and G. Turk. Simplification and Repair of Polygonal Models Using Volumetric Techniques. *IEEE Transactions on Visualization and Computer Graphics*, 9(2):191–205, 2003.

[40] H. Oh Song, Y. Xiang, S. Jegelka, and S. Savarese. Deep Metric Learning via Lifted Structured Feature Embedding. In *Conference on Computer Vision and Pattern Recognition*, pages 4004–4012, 2016.

[41] C. Qi, H. Su, M. Nießner, A. Dai, M. Yan, and L. Guibas. Volumetric and Multi-View CNNs for Object Classification on 3D Data. In *Conference on Computer Vision and Pattern Recognition*, pages 5648–5656, 2016.

[42] C. R. Qi, H. Su, K. Mo, and L. Guibas. Pointnet: Deep Learning on Point Sets for 3D Classification and Segmentation. In *Conference on Computer Vision and Pattern Recognition*, pages 652–660, 2017.

[43] C. R. Qi, L. Yi, H. Su, and L. Guibas. Pointnet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In *Advances in Neural Information Processing Systems*, pages 5099–5108, 2017.

[44] M. Rad, M. Oberweger, and V. Lepetit. Feature Mapping for Learning Fast and Accurate 3D Pose Inference from Synthetic Images. In *Conference on Computer Vision and Pattern Recognition*, pages 4663–4672, 2018.

[45] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In *Advances in Neural Information Processing Systems*, pages 91–99, 2015.

[46] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[47] C. Santa Cruz and J. Dorronsoro. A Nonlinear Discriminant Algorithm for Feature Extraction and Data Classification. *IEEE Transactions on Neural Networks*, 9(6):1370–1376, 1998.

[48] M. Savva, F. Yu, H. Su, et al. SHREC'16 Track Large-Scale 3D Shape Retrieval from ShapeNet Core55. In *Eurographics Workshop on 3D Object Retrieval*, 2016.

[49] M. Savva, F. Yu, H. Su, et al. SHREC'17 Track Large-Scale 3D Shape Retrieval from ShapeNet Core55. In *Eurographics Workshop on 3D Object Retrieval*, 2017.

[50] F. Schroff, D. Kalenichenko, and J. Philbin. FaceNet: A Unified Embedding for Face Recognition and Clustering. In *Conference on Computer Vision and Pattern Recognition*, pages 815–823, 2015.

[51] G. Stockman and L. Shapiro. *Computer Vision*. Prentice Hall PTR, 2001.

[52] H. Su, S. Maji, E. Kalogerakis, and E. Learned-Miller. Multi-View Convolutional Neural Networks for 3D Shape Recognition. In *Conference on Computer Vision and Pattern Recognition*, pages 945–953, 2015.

[53] X. Sun, J. Wu, X. Zhang, Z. Zhang, C. Zhang, T. Xue, J. Tenenbaum, and W. Freeman. Pix3D: Dataset and Methods for Single-Image 3D Shape Modeling. In *Conference on Computer Vision and Pattern Recognition*, pages 2974–2983, 2018.

[54] R. Szeliski. *Computer Vision: Algorithms and Applications*. Springer Science & Business Media, 2010.

[55] F. P. Tasse and N. Dodgson. Shape2Vec: Semantic-Based Descriptors for 3D Shapes, Sketches and Images. *ACM Transactions on Graphics*, 35(6):208, 2016.

[56] M. Tatarchenko, S. Richter, R. Ranftl, Z. Li, V. Koltun, and T. Brox. What Do Single-View 3D Reconstruction Networks Learn? In *Conference on Computer Vision and Pattern Recognition*, 2019.

[57] J. Taylor, J. Shotton, T. Sharp, and A. Fitzgibbon. The Vitruvian Manifold: Inferring Dense Correspondences for One-Shot Human Pose Estimation. In *Conference on Computer Vision and Pattern Recognition*, pages 103–110, 2012.

[58] S. Tulsiani, S. Gupta, D. Fouhey, A. Efros, and J. Malik. Factoring Shape, Pose, and Layout from the 2D Image of a 3D Scene. In *Conference on Computer Vision and Pattern Recognition*, pages 302–310, 2018.

[59] H. Wang, S. Sridhar, J. Huang, J. Valentin, S. Song, and L. Guibas. Normalized Object Coordinate Space for Category-Level 6D Object Pose and Size Estimation. In *Conference on Computer Vision and Pattern Recognition*, pages 2642–2651, 2019.

[60] S. Wang, J. Wu, X. Sun, W. Yuan, W. Freeman, J. Tenenbaum, and E. Adelson. 3D Shape Perception from Monocular Vision, Touch, and Shape Priors. In *International Conference on Intelligent Robots and Systems*, pages 1606–1613, 2018.

[61] Y. Wang, X. Tan, Y. Yang, X. Liu, E. Ding, F. Zhou, and L. S. Davis. 3D Pose Estimation for Fine-Grained Object Categories. In *European Conference on Computer Vision Workshops*, 2018.

[62] K. Weinberger and L. Saul. Distance Metric Learning for Large Margin Nearest Neighbor Classification. *Journal of Machine Learning Research*, 10:207–244, 2009.

[63] Y. Wen, K. Zhang, Z. Li, and Y. Qiao. A Discriminative Feature Learning Approach for Deep Face Recognition. In *European Conference on Computer Vision*, pages 499–515, 2016.

[64] P. Wohlhart and V. Lepetit. Learning Descriptors for Object Recognition and 3D Pose Estimation. In *Conference on Computer Vision and Pattern Recognition*, pages 3109–3118, 2015.

[65] J. Wu, C. Zhang, X. Zhang, Z. Zhang, W. Freeman, and J. Tenenbaum. 3D ShapeNets: A Deep Representation for Volumetric Shapes. In *Conference on Computer Vision and Pattern Recognition*, pages 1912–1920, 2015.

[66] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. Learning Shape Priors for Single-View 3D Completion and Reconstruction. In *European Conference on Computer Vision*, pages 646–662, 2018.

[67] Y. Xiang, W. Kim, W. Chen, J. Ji, C. Choy, H. Su, R. Mottaghi, L. Guibas, and S. Savarese. ObjectNet3D: A Large Scale Database for 3D Object Recognition. In *European Conference on Computer Vision*, pages 160–176, 2016.

[68] Y. Xiang, R. Mottaghi, and S. Savarese. Beyond Pascal: A Benchmark for 3D Object Detection in the Wild. In *IEEE Winter Conference on Applications of Computer Vision*, pages 75–82, 2014.

[69] X. Zhang, Z. Zhang, C. Zhang, J. Tenenbaum, W. Freeman, and J. Wu. Learning to Reconstruct Shapes from Unseen Classes. In *Advances in Neural Information Processing Systems*, pages 2257–2268, 2018.

Location Field Descriptors: Single Image 3D Model Retrieval in the Wild Supplementary Material

In the following, we provide additional details and qualitative results of our novel 3D model retrieval approach called Location Field Descriptors.

6. Datasets and Evaluation Setup

We evaluate our proposed approach for 3D model retrieval in the wild on three challenging real-world datasets with different object categories: Pix3D [53] (*bed*, *chair*, *sofa*, *table*), Comp [61] (*car*), and Stanford [61] (*car*). These datasets have only been released recently and, to the best of our knowledge, we are the first to report results for 3D model retrieval on all of them. In addition to retrieval from 3D models provided by these datasets, we also retrieve 3D models from ShapeNet [9]. Table 3 presents an overview of the object categories, the number of RGB images, and the number of 3D models in the evaluated datasets.

The Pix3D dataset provides multiple categories, however, we only train and evaluate on categories which have more than 300 non-occluded and non-truncated samples (*bed*, *chair*, *sofa*, *table*). Further, we restrict the training and evaluation to samples marked as non-occluded and non-truncated, because we do not know which objects parts are occluded nor the extent of the occlusion, and many objects are heavily truncated. For each 3D model, we randomly choose 50% of the corresponding images for training and the other 50% for testing.

The Comp and Stanford datasets only provide one category (*car*). Most images show one prominent car which is non-occluded and non-truncated. The two datasets already provide a train-test split. Thus, we use all available samples from Comp and Stanford for training and evaluation.

In contrast to these datasets which provide a large number of RGB images and a moderate number of 3D models with corresponding annotations, ShapeNet does not provide RGB images but a large number of 3D models. Due to its enormous size, ShapeNet does not only cover many different object categories but also presents a large variety in 3D model geometry. If 3D models are present in the respective dataset and in ShapeNet, we exclude them for retrieval from ShapeNet to evaluated retrieval from an entirely unseen database.

We consistently orient, scale and translate all 3D models. In particular, we rotate all 3D models to have common front facing, up, and starboard directions. Additionally, we scale and translate all 3D models to fit inside a unit cube centered at the coordinate origin $(0, 0, 0)$ while preserving the aspect-ratio of the 3D dimensions. This 3D model alignment is not only important for training our approach but also for the evaluated metrics. For example, computing the modified Hausdorff distance and the 3D IOU between two 3D models is only meaningful if they are consistently oriented, scaled and centered.

7. Implementation and Training Details

For our Location Field CNN, we use a Feature Pyramid Network [33] on top of a ResNet-101 backbone [18, 19]. For our Descriptor CNN, we use a DenseNet-50 architecture [21] with 3 dense blocks and a growth rate of 24. For our implementation, we resize and pad RGB images to a spatial resolution of $512 \times 512 \times 3$ maintaining the aspect ratio. For the location fields, we employ a resolution of $58 \times 58 \times 3$. In this configuration, the Descriptor CNN maps low-resolution location fields to a 270-dimensional embedding space.

We initialize the convolutional backbone and the detection branches of the Location Field CNN with weights trained for instance segmentation [17] on COCO [34]. The location field branch and the Descriptor CNN are trained from scratch. We train our networks for 300 epochs using a batch size of 32. The initial learning rate of $1e^{-3}$ is decreased by a factor of 5 after 150 and 250 epochs.

We employ different forms of data augmentation. For RGB images, we use mirroring, jittering of location, scale, and rotation, and independent pixel augmentations like additive noise. For rendered location fields, we additionally use different forms of blurring to simulate predicted location fields. During training of the Descriptor CNN, we further leverage synthetic data and train on predicted and rendered location fields using a ratio of $1 : 3$.

To balance the individual terms in the system loss

$$L = L_{\text{D}} + L_{\text{softmax}} + \alpha L_{\text{C}} + \beta L_{\text{TC}} + \gamma L_{\text{LF}} + \delta L_{\text{FM}}, \quad (5)$$

we assign unit weights to classification losses and non-unit

Dataset	Category	Train Images	Test Images	3D Models from Dataset	3D Models from ShapeNet
Pix3D	bed	203	191	19	254
	chair	1506	1388	221	6778
	sofa	552	540	20	3173
	table	387	351	62	8443
Comp	car	3798	1898	98	7497
Stanford	car	8144	8041	134	7497

Table 3: Overview of the evaluated datasets. The Pix3D, Comp, and Stanford datasets provide a large number of RGB images and a moderate number of 3D models with corresponding annotations. In contrast, ShapeNet does not provide RGB images but a large number of 3D models.

weights to regression losses. Thus, we combine the unmodified Detection losses (L_{D}) of the generalized Faster/Mask R-CNN framework and the Descriptor CNN softmax loss (L_{softmax}) with the weighted Center (L_{C}), Triplet-Center (L_{TC}), Location Field (L_{LF}) and Feature Mapping (L_{FM}) losses. We experimentally set $\alpha = 0.01$, $\beta = 0.1$, $\gamma = 10$, $\delta = 0.01$, and use a margin of $m = 1$. For the Huber distance [23], we set the threshold to 1.

8. Failure Cases

Fig. 8 shows failure cases of our approach. Most failure cases relate to incorrect location field predictions. For example, if the 3D pose of the object in the image is far from the 3D poses seen during training, or if multiple objects are detected as a single object in a complex occlusion scenario, we cannot predict an accurate location field. In other failure cases, we predict an accurate location field, but retrieve a 3D model from a different category due to ambiguous 3D geometries, *e.g.*, *table* instead of *chair*. While the detection branch of the generalized Faster/Mask R-CNN predicts a category for each detected object in an RGB image, we do not use this information during retrieval, because there is a category ambiguity for many objects. For example, it is unclear if a couch with sleeping functionality is a *sofa* or a *bed* (see Fig. 10, left column, last example).

9. Additional Qualitative Results

Finally, we present additional qualitative results which complement those presented in the main paper.

Fig. 9 presents further qualitative examples of our predicted location fields. We upscale and pad the predicted location fields to match the input image resolution. The overall 3D shape is recovered well in the location fields, but fine-grained details like the side mirrors of cars or thin structures like the frame ornaments of tables and beds are missed.

Fig. 7 shows additional qualitative results for 3D model retrieval from ShapeNet. Considering the top ten ranked 3D models, we observe that the retrieved models have a consistent and accurate overall 3D shape and geometry.

Figs. 10 and 11 present more qualitative results for 3D model retrieval from both seen and unseen databases. In addition, we show that our predicted location fields provide all relevant information to also compute the 3D pose of objects. For this purpose, we sample 2D-3D correspondences from the location field and solve a P*n*P problem during inference. The projections onto the image show that both our retrieved 3D models and our computed 3D poses are highly accurate. Our approach naturally handles multiple objects in a single image, however, all evaluated datasets only provide annotations for a single instance per image, as shown in Fig. 11.

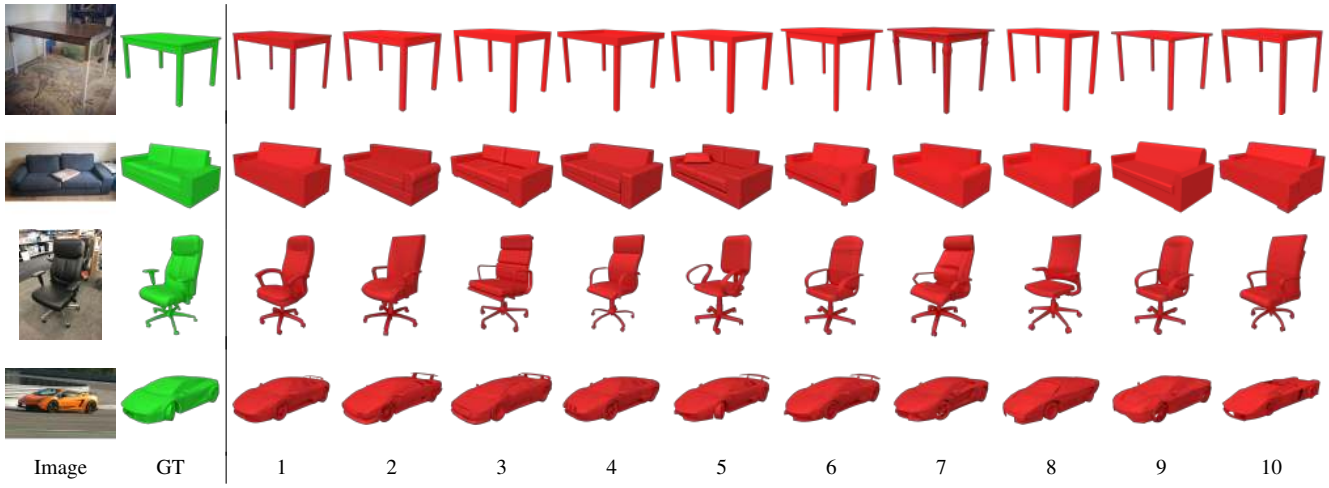


Figure 7: Additional qualitative results for 3D model retrieval from ShapeNet. From left to right, we show the input image, the ground truth 3D model and the top ten ranked 3D models. The overall 3D shape of the retrieved models is consistent and accurate.

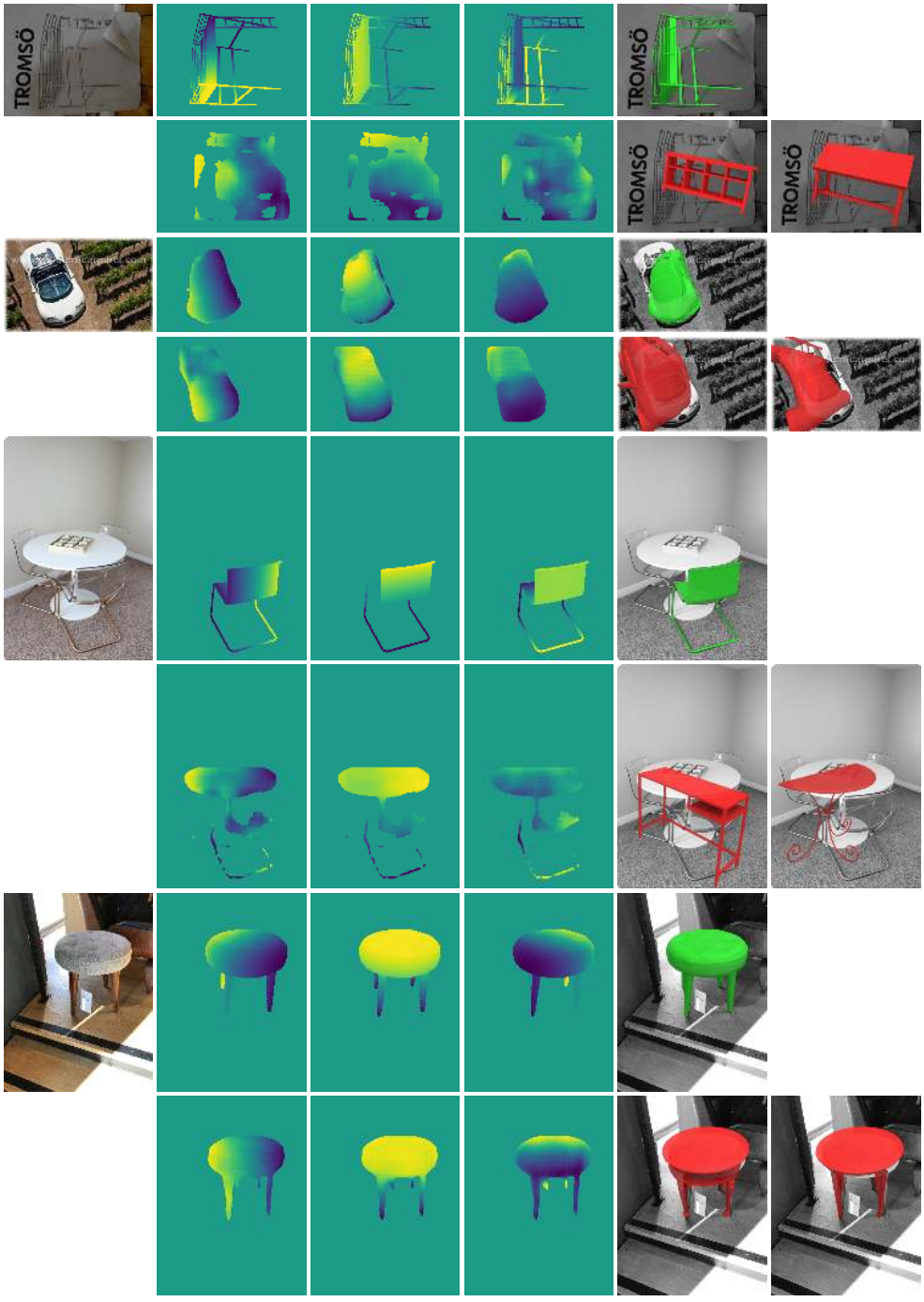


Figure 8: Failure cases of our approach. For each example image, the top row shows the ground truth and the bottom row shows our prediction. Most failure cases relate to incorrect location field predictions. If the 3D pose of the object in the image is far from the 3D poses seen during training (first and second example), we cannot predict an accurate location field. The location field prediction can also fail in complex occlusion scenarios if multiple objects are detected as a single object (third example). In other cases, we predict an accurate location field, but retrieve a 3D model from a different category due to ambiguous 3D geometries, *e.g.*, *table* instead of *chair*.

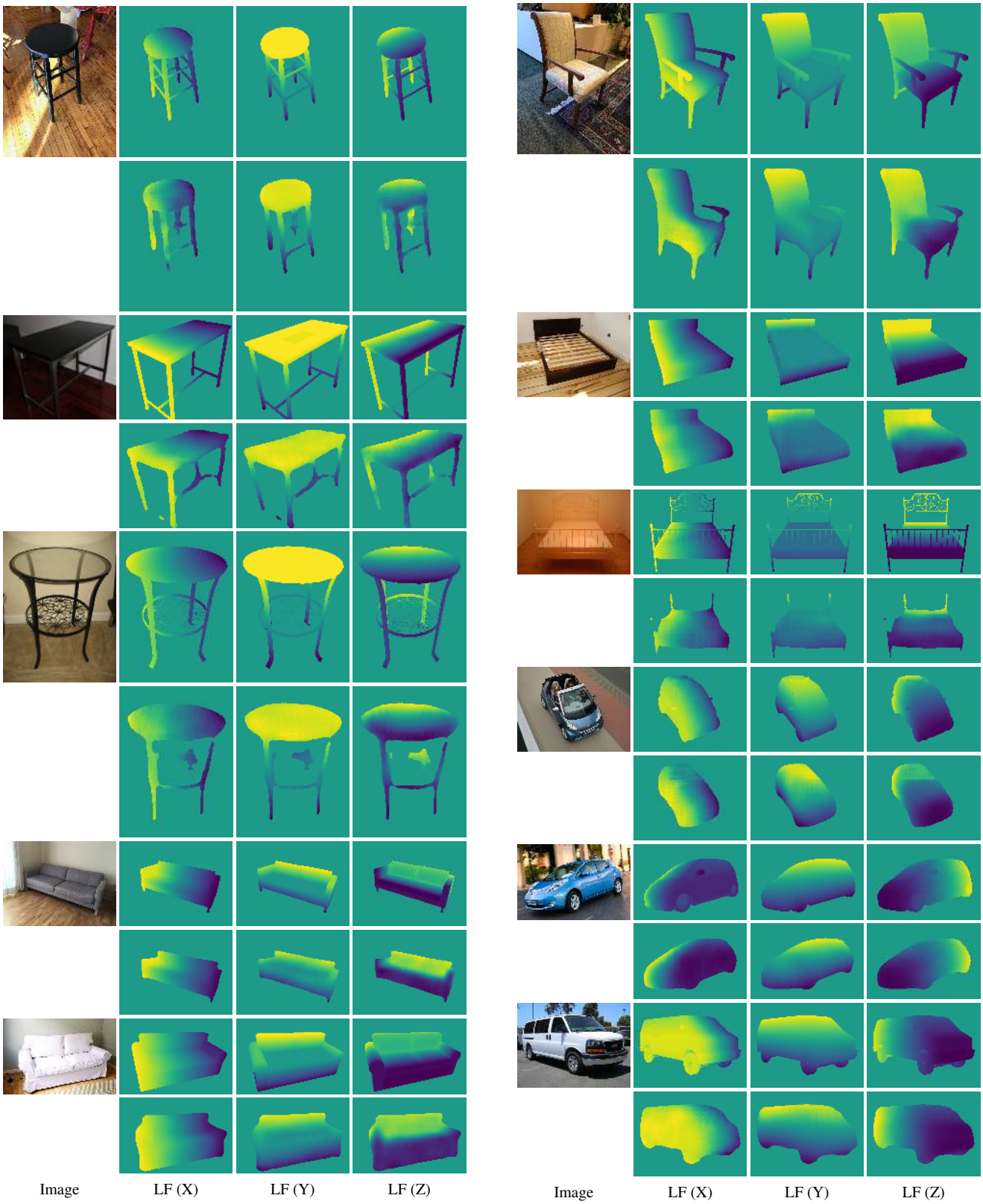


Figure 9: Additional qualitative examples of our predicted location fields. For each example image, the top row shows the ground truth and the bottom row shows our prediction. The overall 3D shape is recovered well, but fine-grained details like the side mirrors of cars or thin structures like the frame ornaments of tables and beds are missed.

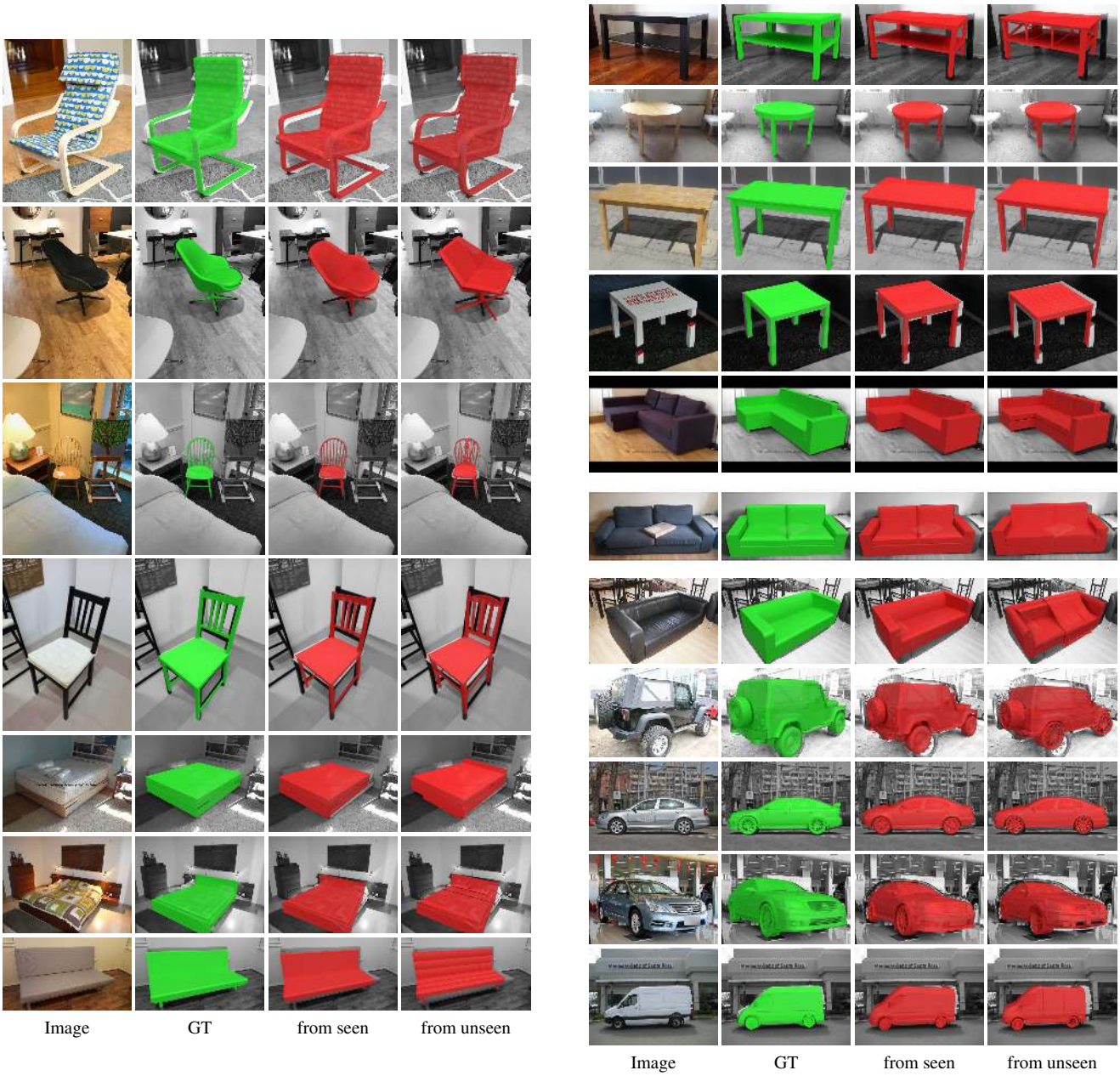
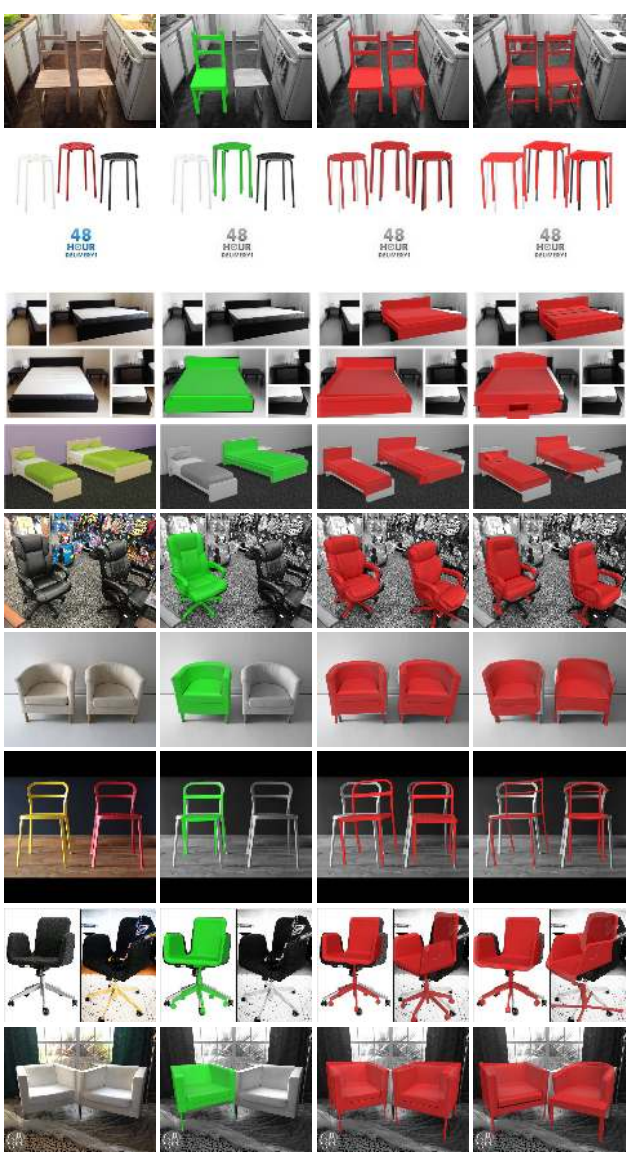


Figure 10: Additional qualitative results for 3D pose estimation and 3D model retrieval from both seen and unseen databases. We project the retrieved 3D model onto the image using a 3D pose computed from the predicted location field by solving a P*n*P problem. For the ground truth 3D model, we use the ground truth 3D pose. In fact, location fields provide all relevant information to jointly address both tasks.



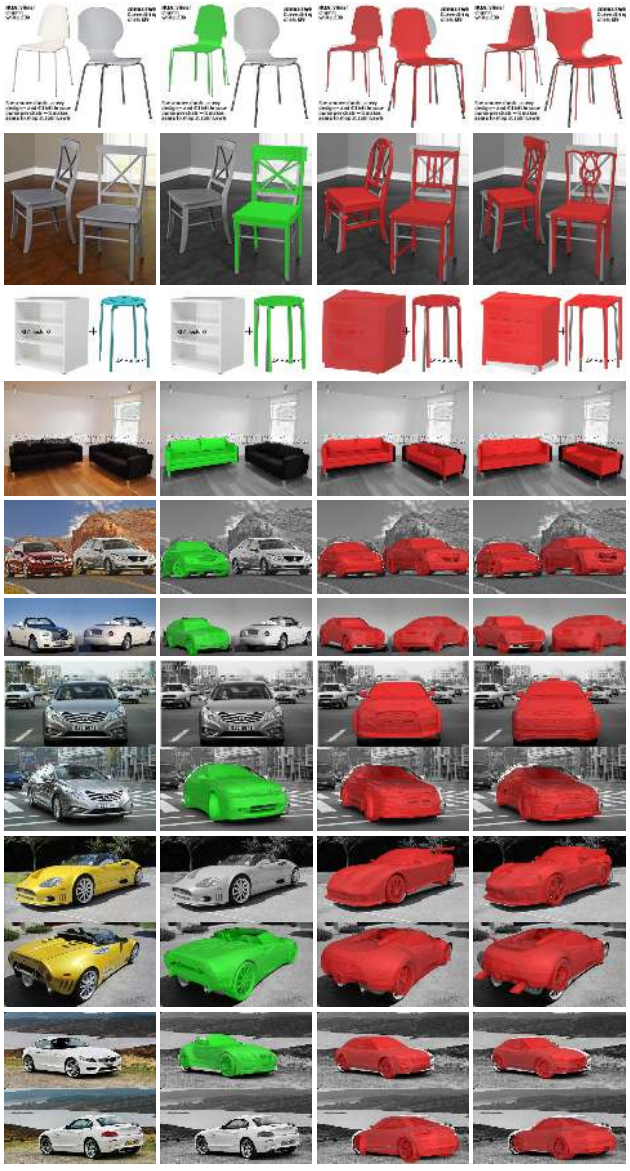


Figure 11: Additional qualitative results for 3D pose estimation and 3D model retrieval from both seen and unseen databases. We project the retrieved 3D model onto the image using a 3D pose computed from the predicted location field by solving a P*n*P problem. For the ground truth 3D model, we use the ground truth 3D pose. In fact, location fields provide all relevant information to jointly address both tasks.