Leveraging Geometry for Shape Estimation from a Single RGB Image

Florian Langer
fml35@cam.ac.uk
Ignas Budvytis
ib255@cam.ac.uk
Roberto Cipolla
rc10001@cam.ac.uk

Department of Engineering
University of Cambridge
Cambridge, UK

Abstract

Predicting 3D shapes and poses of static objects from a single RGB image is an important research area in modern computer vision. Its applications range from augmented reality to robotics and digital content creation. Typically this task is performed through direct object shape and pose predictions [12, 26, 33] which is inaccurate. A promising research direction [13, 15, 18] ensures meaningful shape predictions by retrieving CAD models from large scale databases [2, 30] and aligning them to the objects observed in the image. However, existing work [18] does not take the object geometry into account, leading to inaccurate object pose predictions, especially for unseen objects. In this work we demonstrate how cross-domain keypoint matches from an RGB image to a rendered CAD model allow for more precise object pose predictions compared to ones obtained through direct predictions. We further show that keypoint matches can not only be used to estimate the pose of an object, but also to modify the shape of the object itself. This is important as the accuracy that can be achieved with object retrieval alone is inherently limited to the available CAD models. Allowing shape adaptation bridges the gap between the retrieved CAD model and the observed shape. We demonstrate our approach on the challenging Pix3D [30] dataset. The proposed geometric shape prediction improves the AP^{mesh} [12] over the state-of-the-art [18] from 33.2 to 37.8 on seen objects and from 8.2 to 17.1 on unseen objects. Furthermore, we demonstrate more accurate shape predictions without closely matching CAD models when following the proposed shape adaptation.

1 Introduction

The past few years have seen rapid advances in object recognition in RGB images [14, 17, 23]. However, for many applications reasoning not in the 2-dimensional image but in the 3-dimensional world is crucial. One important task is 3D shape estimation from a single image, with applications ranging from robotics to augmented reality and digital content creation. Current research on this task can be categorised into two different streams: *direct shape prediction* and *shape estimation via object retrieval*.

For direct object shape predictions different representations are used ranging from voxels [6], point clouds [10, 34], meshes [12, 26, 27, 33], packed spheres [10], binary space partitioning [5], convex polytopes [8], signed distance fields [28] to other implicit representations [25].

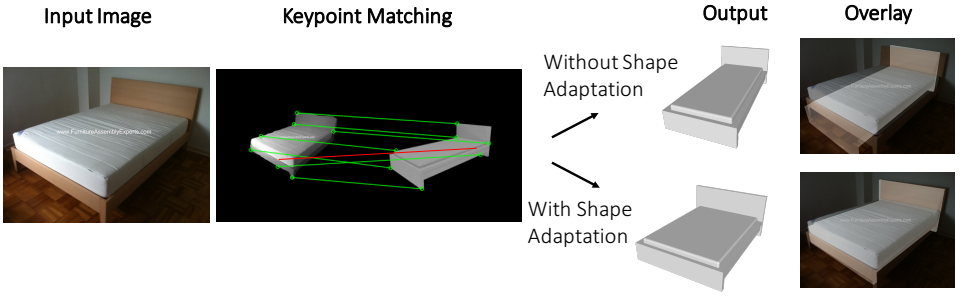


Figure 1: **Example result of our approach.** Given an input image we retrieve a CAD model rendering and perform key-point matching with the masked input image. Without shape adaptation the retrieved CAD model prediction is limited by the availability of similar CAD models in the database. When shape adaptation is performed target object shapes and their poses can be predicted precisely.

Most of these approaches suffer either from a lack of precision [6, 8, 10, 12, 26, 27, 33, 34] or lack applicability to a wide range of object classes [5, 25, 28].

Regardless of the representation chosen, directly predicting the shape and pose of an object from a single image is a difficult learning task. It is difficult as crucial information about the parts of an object without direct line of sight to the camera is absent at train and test time. The absence of complete shape information leads to an inherent ambiguity, as multiple consistent shape predictions may be possible; this ambiguity prevents the networks from learning effectively, causing them to perform class-averaged object shape predictions [31].

A natural way to deal with the ill-posed nature of the problem lies in retrieving CAD models from existing large-scale databases [2]. [20] and IM2CAD [15] show how CAD models and real images can be effectively encoded into a joint embedding space. At test time given a target RGB image multiple candidate CAD models can be retrieved. In order to estimate the object pose [18] regress rotation parameters and use the ground truth z-coordinate for predicting object translation (see Section 2). While in this manner a neural network is able to store and interpolate between poses for objects that were seen during training, it fails to generalise to unseen objects. We avoid this problem by using a deep network to perform the simpler task of matching keypoints between the real image and the retrieved CAD model render [11], and using these matches as constraints to calculate the object pose analytically.

More importantly we show that we can use keypoint matches to modify the shape of the retrieved CAD model to better fit the object observed (see Figure 1). We modify CAD model shapes by stretching them along the normal vector of 3D-planes. Stretching can be seen as a local operation which in contrast to a global scaling operation can modify proportions of objects within a single dimension (e.g. adjusting the height of the sitting area of a chair as shown in Figure 3). Currently our approach uses stretching along the three principal object axis. However, in the future predicting additional stretch planes and limiting stretch extents to variable 3D-boxes will allow for even more fine-grained shape adaptations.

We evaluate our approach on the Pix3D [30] dataset. When combining object retrieval with a geometric pose prediction we outperform existing work [12, 18] on splits containing both seen and unseen CAD models at train time (see Table 1). We evaluate the proposed object adaptation on a range of adaptation experiments. Here we observe that dynamic fitting improves the shape predictions when no access to correct CAD models is given at test time and retrieved models have to be adapted (see Figure 6).

2 Related Work

Related literature on shape estimation for static objects from single images can be categorised into direct shape prediction methods and retrieval based methods.

Direct Prediction Methods. Current direct prediction methods differ greatly in the choice of object representations that are used. In the *voxel* representation [6] the 3D world is discretised into cubes and object shapes are encoded as binary occupancies of these cubes. This allows the shape prediction task to be formulated as a binary classification task which is usually simpler to learn compared to a regression task. However, the voxel representation suffers from an inherent trade off between accuracy and storage space due to the cubic scaling. *Meshes* [12] alleviate the storage-accuracy trade-off by encapsulating information about the 2D object surface rather than its 3D volume. An object is represented as a set of vertices which are interconnected to form (usually triangular) mesh faces. The difficulties associated with meshes are twofold; predicting the correct object topology and predicting precise vertex positions. Early work [33] simply deformed an original ellipsoid mesh and was therefore not able to predict shapes with complex topologies. In contrast [12] first predict a rough shape estimate in the voxel representation. This serves as the initialisation for the topology of the shape and is subsequently refined as a mesh. Recently Topology Modification Networks [27] aim to directly predict varying object topologies. Other works decompose objects into a set of *convex polytopes* [8] or assemble 3D shapes from *geometric primitives* [32]. Besides these explicit object encodings other implicit representations exist for defining the object boundary. [5] approximate an object by predicting a set of *3D planes* and the corresponding side on which the object lies on. Given a 3D point *signed distance fields* [28] predict the distance of the point from the object surface and whether it resides within or outside of the object. *Neural radiance fields* [25] encode information not just about the object shape but also its texture. While these implicit representations are promising research directions, they require vast amounts of training and have not been shown yet to work on a large number of different shapes in realistic settings.

Object Retrieval. Almost all existing work on object retrieval constructs an embedding space of CAD model renderings. At test time a real image is embedded into this space and its nearest neighbours are retrieved. While early work [1] construct and map into this space based on the Histogram-of-Gradient [7] (HoG) descriptor, [20] learn real image embeddings using a convolutional neural network. Instead of constructing the embedding space from HoG-descriptors [15] learns a *joint embedding space* between CAD model renderings and real images. In order to align retrieved CAD models with the objects in the image [15] follows a *Render-and-Compare* approach in which they optimise for an object pose by iteratively comparing the re-rendered CAD model embedding to the original image embedding and update the pose accordingly. While [15] can successfully predict rough room layouts and retrieve similar CAD models to the ones observed, their approach is significantly less accurate in terms of object retrieval and pose prediction than ours. The work most similar to ours is Mask2CAD [18]. In contrast to our method [18] estimates the object pose by regressing the rotation parameters as well as the offset between the reprojected object center and the 2D bounding box centers. Using the ground truth z-coordinate [18] can estimate the object pose. Unlike our system [18] is limited to pure shape retrieval as it can not perform shape adaptations. Patch2CAD [19] is similar to [18] but instead of retrieving a single CAD model based on the entire image, [19] retrieves many CAD models for different image regions and perform majority voting to obtain the final prediction. [13] learn an embedding space and estimate poses using a Location Field Descriptor containing the estimated 3D object coordi-

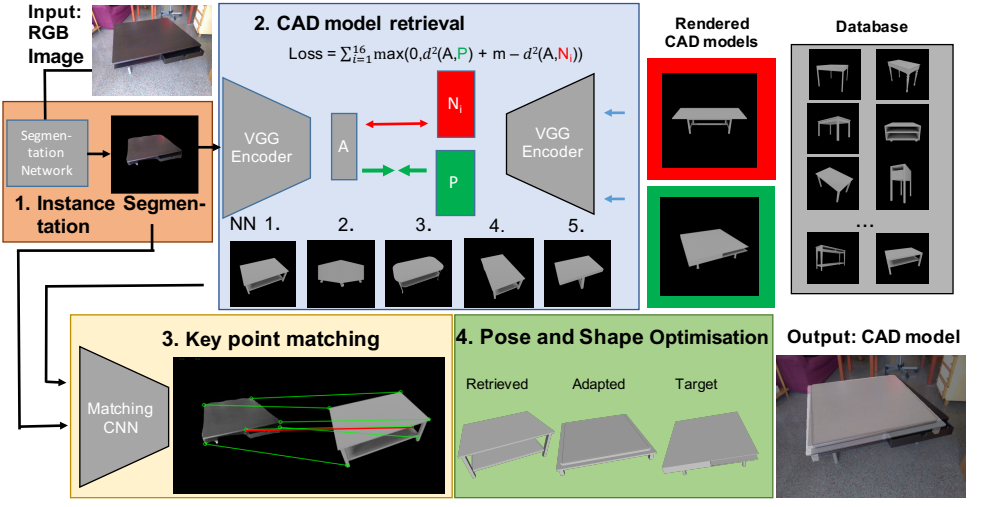


Figure 2: **Method:** Given an RGB image we perform object detection and instance segmentation (step 1). We then retrieve a set (e.g. 10) of the nearest neighbour CAD model renderings (step 2) and perform keypoint matching between the retrieved image and the rendered image using SuperPoint [9] (step 3). The keypoint matches are subsequently used to jointly optimise over the shape and pose of the object (step 4).

nate for every pixel. However, their method struggles to predict correct location fields from which the pose can be accurately computed and do not work for occluded objects. Recently [24] have extended [18] to temporally combine predictions from individual video frames into more accurate CAD model and pose predictions. We differ from all existing work on object retrieval in the usage of keypoint matches for precise pose and shape estimation.

3 Method

Our method consists of four steps: (i) object detection and instance segmentation, (ii) CAD model retrieval, (iii) keypoint matching and finally, (iv) pose and shape optimisation.

Object Detection and Instance Segmentation. For object detection and instance segmentation we train a Swin-Transformer [23] network on the Pix3D [30] dataset. During training we employ standard image augmentation techniques including random crops, scaling, rotations, horizontal flipping, and random brightness and contrast adjustments. We also report results on segmentation masks obtained using Mask R-CNN [14] trained by [12] as our approach is sensitive to the quality of segmentation predictions. The Swin-Transformer network provides more accurate segmentations of objects with hard edges (e.g. chairs, tables or wardrobes), while Mask R-CNN is superior on objects from categories with soft edges such as beds and sofas.

Learning a Joint Embedding Space. In order to map CAD models and masked RGB images into a joint embedding space, we first render a CAD model in regular intervals and represent it as a collection of these renderings [4]. This step is important as it bridges the domain gap from CAD models to RGB images and increases the similarity of the two inputs therefore simplifying the matching task. We use a single VGG [29] encoder for encoding both real masked RGB images and rendered inputs. This encoder is trained using a triplet-

loss [3]:

$$\mathcal{L} = \sum_{i=1}^{16} \max(0, d^2(A,P) - d^2(A,N_i) + m). \quad (1)$$

Here A is the encoding of an anchor RGB image, P is the encoding of the positive example and N_i is the encoding of a negative example. Only the rendering of the corresponding ground truth CAD model in the most similar orientation to the object is considered to be the positive example. A random subset of 16 renderings of different CAD models are used as negative examples N_i. Finally, m is the margin and $d(x,y)$ is the Euclidean distance over the 128-dimensional encodings. As in [18] we employ hard example mining during training. Only renderings of CAD models of the same category as the query image are considered for hard-negative mining. Unlike [18], no hard example mining is applied to positive examples. Doing hard positive mining forces the network to match very different views, often not sharing any features with each other, therefore necessarily leading to poor performance via over-fitting. Instead, as mentioned above, the CAD model rendering in the most similar orientation is used as a positive example. At test time we use an embedding of a masked RGB image and retrieve the nearest neighbour CAD model renderings which are then passed on for keypoint matching and joint pose and shape estimation.

Key-Point Matching. In order to precisely estimate poses, keypoint matching is performed between the masked RGB image and its retrieved CAD model rendering [11]. Finding and matching valid keypoints across different domains (RGB and rendered image) may seem initially a difficult task due to the contrast of cleanly (sharp boundaries, perfect segmentation masks, no occlusions) rendered images and varying appearances of objects in RGB images (variety of textures, lighting conditions and imperfect segmentation masks). However, we found a SuperPoint [9] keypoint detection and matching network to be well suited for this task. Its robustness stems from the fact that it was initially trained to detect corners of triangles, quadrilaterals, lines, cubes, checkerboards and stars in synthetic images which was followed by fine-tuning on real images, hence performing well on both domains. Crucially man-made furniture objects contain many of the aforementioned primitives. We used the trained off-the-shelf network implementation from [9] to avoid over-fitting as Pix3D [30] is considerably smaller than typical datasets used for keypoint detector training. SuperPoint [9] returns approximately 25 keypoints (each described with a 256-D vector) on average per real RGB image (15 on rendered image) using a default confidence threshold of 0.015. Nearest neighbour matches are found for each keypoint using the L2-distance. Cross-checking is used to eliminate one-sided matches producing on average 12 matches per RGB and rendered image pair.

Pose Estimation. The keypoint matches provide correspondences between real image pixel coordinates (2D) and 3D world coordinates in CAD model space. This allows us to formulate the pose estimation as a PnP-problem. Since the obtained matches are often noisy (only 4-5 matches out of 12 are correct on average) we estimate poses for all available quadruplets[1] of matches using the UPnP [16] algorithm. Instead of using an inlier scoring as is typically done in robust pose estimation, we select a final pose from the computed poses by approximating the Intersection-over-Union (IoU) silhouette overlap of the reprojected CAD model and the predicted segmentation mask. For this purpose we sample 1000 points from the CAD model and reproject them into the RGB image. Additionally we sample 1000 points from within the predicted segmentation mask. For each reprojected point we compute the distance

[1] On average we obtain 12 keypoint matches per image leading to $12 \times 11 \times 10 \times 9/(4 \times 3 \times 2 \times 1) = 495$ possible quadruplets and poses.

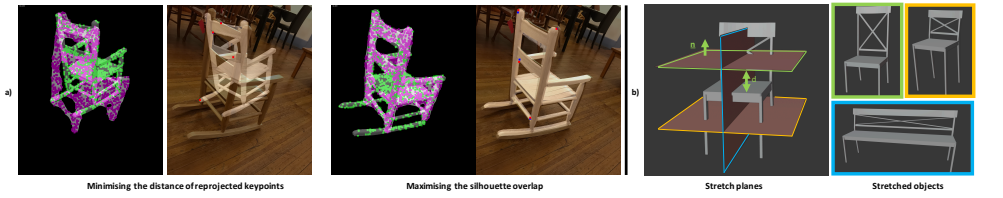


Figure 3: Left: **Pose Selection.** Green points are reprojected points from the CAD model under the current pose estimate. Purple points are sampled inside the predicted segmentation mask. Selecting a pose based on the approximated silhouette overlap yields precise poses (top) while using the minimum distance of the reprojected keypoints selects inaccurate poses (bottom). Right: **Stretching Procedure.** A CAD model can be stretched along different stretch planes to generate new shapes. The plane normal $\mathbf{n}$ and the distance d of the plane from the object center are highlighted for one of the stretch planes.

to the closest points sampled from within the predicted segmentation mask and vice versa (see Figure 3 on the left). For each object pose the average of the largest 20% of pairwise distances is computed and the pose with the smallest average is selected as the final pose. We found that selecting the final pose based on the average of all pairwise distances was less robust in pose estimation (see supplementary material).

Pose and Shape Estimation. In order to perform joint pose and shape adaptation the standard PnP-formulation is not sufficient. In particular, to enable shape adaptation, we reparametrise previously fixed object coordinates x (now $x_{stretch}$) in the following way. We allow stretching of the object by an amount τ_i along the normal $\mathbf{n}_i$ of a plane P_i defined by $\mathbf{n}_i \cdot \mathbf{x}_i = d_i$. Here d_i is the distance of the plane P_i from the object center (see Figure 3 on the right). Repeating this stretching for N_{planes} orthogonal planes the stretched world coordinates become

$$\mathbf{x}_{\text{stretch}} = \mathbf{x} + \sum_{i=1}^{N_{\text{planes}}} s_i \cdot \frac{\tau_i}{2} \cdot \mathbf{n}_i \qquad \text{where } s_i = \begin{cases} 1, & \text{if } \mathbf{x} \cdot \mathbf{n}_i \geq d_i \\ 0, & \text{if } \mathbf{x} \cdot \mathbf{n}_i = d_i \\ -1, & \text{if } \mathbf{x} \cdot \mathbf{n}_i \leq d_i \end{cases}. \tag{2}$$

From the stretched world coordinates one obtains reprojected pixel $\mathbf{v} \in \mathcal{R}^2$ under the perspective camera model $(sv_x, sv_y, s) = \mathbf{K}[\mathbf{R}|\mathbf{T}][\mathbf{x}_{\text{stretch}}, 1]^T$ for camera calibration matrix $\mathbf{K} \in \mathcal{R}^{3\times3}$, rotation matrix $\mathbf{R} \in \mathcal{R}^{3\times3}$ and translation vector $\mathbf{T} \in \mathcal{R}^{3\times1}$. The rotation matrix is parameterised in terms of Euler angles $\theta \in \mathcal{R}^3$. For known camera intrinsics the objective function to be minimised is therefore $f = \sum_{j=1}^{N_{\text{matches}}} (\mathbf{u}_j - \mathbf{v}_j)^2$ with respect to $(\theta, \mathbf{T}, \tau)$ where $\mathbf{u}_j$ are pixel coordinates of the j-th match in the RGB image. L-BFGS [22] minimiser, instead of UPnP is used for the minimisation of this non-linear objective function. It is initialised with the original pose of the retrieved CAD model and no stretching $\tau = 0$. Note that in this case we sample sets of 6 matches instead of 4 to cover enough degrees of freedom for 3 deformations.

4 Experimental Setup

This section briefly describes the Pix3D [30] dataset that was used for training and evaluation, the AP^{mesh} metric we adopted for evaluation as well as the hyperparameters chosen.

Pix3D Dataset. The Pix3D [30] dataset consists of 10,069 RGB images annotated with aligned 3D CAD models (one per image). There are a total of 395 different CAD models

from 9 categories (chair, sofa, table, bed, desk, bookcase, wardrobe, tool and miscellaneous). For our experiments we consider two splits originally proposed by [12].

S1 split. The *S*1 split randomly splits the 10,069 images into 7539 train images and 2530 test images. In this split all CAD models are seen during training and the challenge is to retrieve and align the correct CAD model from images containing different scenes where (possibly occluded) objects appear with new textures under varying lighting conditions.

S2 split. Under the S2 split train and test images are split such that the CAD models that have to be retrieved at test time were unseen during training. This split is more difficult as it prohibits the embedding network to simply remember CAD models and truly tests its ability to learn meaningful embeddings.

Evaluation metric. We adopt the commonly used AP^{mesh} metric [12] for evaluating the retrieved object shapes. Following the standard COCO [21] object detection protocol of AP50-AP95 (denoted AP), we average over 10 IoU thresholds ranging from 0.50 to 0.95 in 0.05 intervals. For a given threshold the AP^{mesh} score is defined as the mean area under the per-category precision-recall curve where a shape prediction is considered a true-positive if its predicted category label is correct, it is not a duplicate detection, and its $F1^{\tau}$ is greater than the IoU threshold. For a given predicted shape the $F1^{\tau}$ score is the harmonic mean of the fraction of predicted points within τ of a ground-truth point and the fraction of ground-truth points within τ of a predicted point. We follow [12, 18] in choosing $\tau = 0.3$. For fair comparison across different object sizes we rescale all objects such that the longest edge of the ground truth model's bounding box has length 10 before computing the F1 score.

Hyperparameter settings. CAD models are rendered at 16 regularly sampled azimuthal angles spanning 360° and 4 different elevation angles between 0° and 45°. The VGG encoder is trained with a batchsize of 8 real images as each example requires 16 negative anchors and one positive anchor leading to a total of 144 images per batch. We use a learning rate of 2×10^{-6} and set the margin of the triplet-loss in Equation 1 to $m = 0.1$. We use off-the-shelf object detection and segmentation networks [14, 23] as well as keypoint matching network - SuperPoint [9].

5 Experimental Results

This section showcases our experimental results. We compare against Mesh-RCNN [12] and Pixel2Mesh [33] (specifically the reimplementation by [12] which outperforms the original [33]) as well as Mask2CAD [18] and Patch2CAD [19]. Section 5.1 shows our results on the Pix3D[30] S1 and S2 split when access to the correct CAD models is provided at test time. In Section 5.2 we evaluate the proposed stretching on modified versions of the original CAD models as well as when available CAD models have to be adapted to match entirely different ones.

5.1 Geometry-Based Shape and Pose Predictions are very Precise

Table 1 shows the AP^{mesh} we obtain on the S1 and S2 splits of Pix3D, originally proposed by Mesh R-CNN [12]. Note that for a fair comparison to Mask2CAD [18], Patch2CAD [19], Mesh R-CNN [12] and Pixel2Mesh [33] we use the ground truth z-coordinate for the final pose. While our approach does not require the use of the ground truth z-coordinate (unlike [18, 19]) it improves our performance as the low F1 score threshold is very sensitive even to small displacements in the z direction arising from slight inaccuracies in the keypoint matches.

Seen Objects. Results on the S1 split show that particularly on seen objects, CAD model

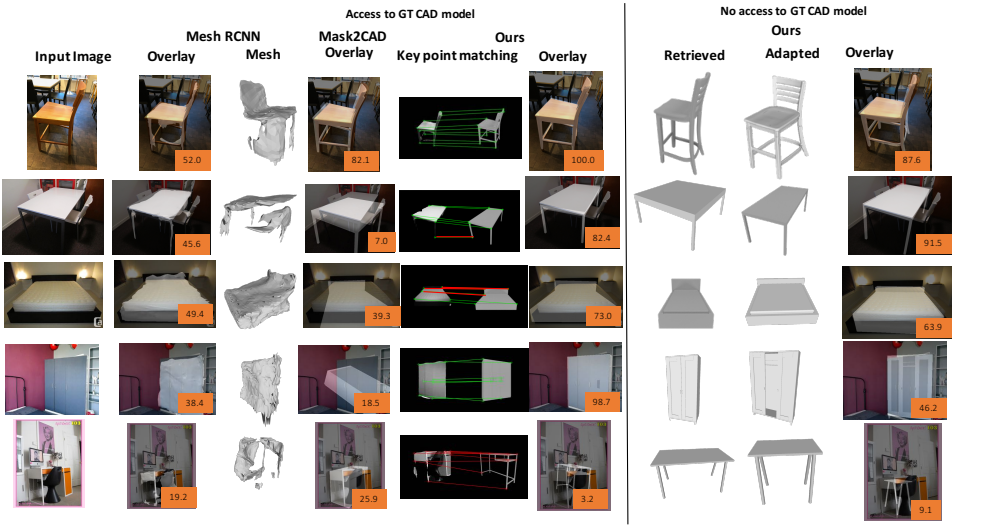


Figure 4: **Qualitative comparison for predictions on the S2 split:** The left side shows results when access to the correct CAD model is given at test time, but which were unseen at train time. The right side shows the case when no access to correct CAD models is given and retrieved CAD models have to be adapted dynamically. Overlaid numbers are the F1 score. In general the comparison shows that a geometric approach allows for very precise pose estimation whereas the direct prediction method of Mask2CAD [18] is less precise. Note that the qualitative results are from our reimplementation of [18] as neither the code nor the predictions are publicly available. In comparison to CAD model retrieval direct mesh predictions [12] are very imprecise, often failing to predict the correct topology and performing particularly poorly on the backside of objects. Row 4 shows the sensitivity of the $F1^{0.3}$ score. Despite an appropriate object retrieval and very good shape adaptation, slight imprecision in the alignments lead to a low F1 score. Finally row 5 shows a failure case of ours where poor segmentation leads to a wrong shape retrieval and correspondingly false keypoint matches resulting in a bad final pose and shape.

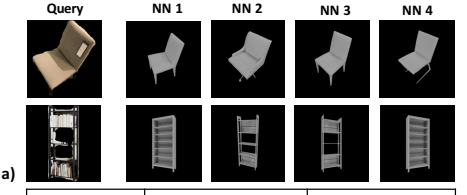
retrieval is more precise than direct predictions. For the high AP^{mesh} regions (i.e. AP and AP75) retrieval-based methods outperform direct prediction methods by a large margin. On the AP50 score Mesh-RCNN [12] performs similar to the retrieval-based methods as the AP50 score only evaluates if at least 50% of a sampled object is predicted correctly and Mesh-RCNN [12] is generally able to predict the object side in direct sight of the camera. In terms of the category average we outperform the best competitor [18] (37.8 vs 33.2). Mask2CAD [18] performs well on large planar objects such as bookcases or wardrobes, whereas we have very strong performance on high fidelity objects, such as chairs, allowing for numerous keypoint matches. When using ground truth masks compared to predicted masks we observe a very large performance gain (37.8 vs 33.2). This observation is crucial as it shows the potential of our approach with improved segmentation masks. While competing approaches will also benefit from better segmentation, having accurate instance masks for the pose prediction is not as an integral part in their pipeline as it is for our keypoint matching and does therefore not benefit them as much.

Unseen Objects. We show qualitative comparisons on the S2 split for [18] and [12] in Figure

S1	AP	AP 50	AP 75	bed	book-case	chair	desk	misc	sofa	table	tool	ward-robe
Pixel2Mesh+	9.9	39.7	2.3	11.4	13.3	6.1	8.2	4.9	15.3	10.6	3.7	15.8
Mesh R-CNN	17.2	51.2	7.4	20.0	10.1	17.6	21.0	24.5	30.0	11.0	6.5	14.3
Mask2CAD	33.2	54.9	30.8	**39.4**	**42.4**	19.6	**31.6**	15.9	**55.8**	29.2	4.2	**60.3**
Patch2CAD	30.9	51.7	28.2	-	-	-	-	-	-	-	-	-
Ours (Mask R-CNN) Top 1	30.9	46.2	30.0	32.0	15.3	30.6	25.6	37.0	47.3	27.1	21.0	42.5
Ours (Mask R-CNN) Top 10	37.1	55.4	35.3	32.3	25.4	37.8	29.5	48.9	53.7	**37.7**	29.7	39.0
Ours (Swin) Top 1	31.1	49.6	29.4	20.6	19.6	30.2	23.6	37.4	41.7	24.7	43.3	38.9
Ours (Swin) Top 10	**37.8**	**56.1**	**36.8**	24.3	25.0	**40.8**	28.4	**51.4**	50.0	36.9	**39.6**	44.3
Ours (GT) Top 1	57.6	72.1	55.9	60.1	49.6	54.5	64.7	68.7	62.7	63.1	41.7	53.0
Ours (GT) Top 10	71.5	82.5	71.7	66.9	53.7	63.8	85.1	86.5	69.3	78.2	71.0	69.0

S2	AP	AP 50	AP 75	bed	book-case	chair	desk	misc	sofa	table	tool	ward-robe
Pixel2Mesh+	4.7	21.0	0.5	10.8	6.1	5.0	1.3	0.0	13.2	2.9	2.4	0.5
Mesh R-CNN	7.5	29.0	1.2	12.7	**17.3**	8.0	3.7	0.0	16.6	7.0	1.1	0.8
Mask2CAD	8.2	20.7	4.8	16.9	2.2	4.5	2.7	0.1	37.8	3.6	0.9	5.3
Patch2CAD	-	-	-	-	-	-	-	-	-	-	-	-
Ours (Mask R-CNN) Top 1	7.9	17.1	5.8	9.4	1.5	19.2	1.6	3.4	27.0	1.7	6.3	1.1
Ours (Mask R-CNN) Top 10	16.6	31.0	**14.6**	**25.3**	2.6	41.1	**6.2**	4.5	38.5	6.2	**16.8**	8.4
Ours (Swin) Top 1	6.9	15.8	4.4	7.0	2.6	18.9	0.4	1.3	25.2	3.5	1.4	1.5
Ours (Swin) Top 10	**17.1**	**32.2**	13.8	24.1	7.1	**44.5**	4.1	**5.9**	**40.0**	**8.3**	9.9	**11.9**
Ours (GT) Top 1	43.7	59.7	41.3	58.8	38.0	74.0	40.8	43.6	42.1	44.9	40.2	10.5
Ours (GT) Top 10	70.7	83.7	70.4	74.7	56.1	94.1	83.4	75.6	69.2	76.9	56.2	50.3

Table 1: **Quantitative results on the S1 and S2 split** for seen and unseen objects on the Pix3D dataset. Brackets indicate the segmentation masks that were used. Bold numbers are the maximum values excluding experiments that were run with ground truth (GT) masks.



	Predicted Masks		GT Masks	
	Top 1	Top 10	Top 1	Top 10
S1	65 %	85 %	87 %	98 %
S1 stretch XYZ	44 %	69 %	70 %	90%
S2	24 %	52 %	62 %	91 %
S2 stretch XYZ	18 %	48 %	49 %	80 %

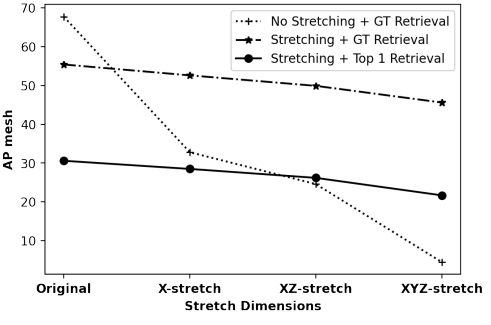


Figure 5: a) **Retrieval accuracy** for selected CAD model splits. When considering the top 10 nearest neighbours the retrieval network is able to return completely unseen CAD models in over 50% of cases. Note that different renderings of the same CAD model are considered as different nearest neighbours. b) **Ablation experiments** on the proposed object stretching with ground truth masks. We plot the average AP^{mesh} score as a function of increasing shape deformations of S2 CAD models. On the left no deformations were performed while on the right objects were stretched along the *x*,*y* and *z* direction. With increasing deformation simple object retrieval quickly becomes inaccurate, while the proposed stretching is able to maintain a high accuracy.

4. Quantitatively, we outperform all competitors on all metrics. We perform significantly better than [18] (17.1 compared to 8.2 on the class average) because of the geometric pose predictions. [18] struggles to predict poses of unseen objects whereas our pose prediction relies on analytically computing poses from correspondences which generalises a lot better to unseen shapes. Note that Mask2CAD [18] performs well on sofas, not because it is able to retrieve an unseen sofa but because for every sofa in the S2 split there is a very good fitting sofa among the seen sofas, allowing to simply retrieve that for a good performance (see the supplementary material).

5.2 Shape and Pose Estimation using Stretchable CAD Models

For realistic settings a given object will not have a perfect match among the available CAD models and a retrieved CAD model will require adaptation. We evaluate our adaptation approach on three different settings: *stretched S1 models*, *stretched S2 models* and predicting *S2 test models with S2 train models* (see Figure 6). For the first two experiments the databases of S1 and S2 models are modified by applying random stretching in the *x*, *y* and *z* direction with planes passing through the center of the object. Stretch factors τ_i in each direction are ob-

a)

			AP	bed	book case	chair	desk	misc	sofa	table	tool	ward robe
1	S1 stretched	Ours (Swin) no stretching	4.4	8.7	3.1	4.5	2.6	1.0	13.8	3.3	0.7	1.9
		Ours (Swin) with stretching	15.5	16.8	12.7	16.4	15.4	8.5	29.3	17.6	5.1	17.4
2	S2 stretched	Ours (Swin) no stretching	8.5	22.8	0.9	5.8	0.2	0.0	34.7	3.5	2.0	6.4
		Ours (Swin) with stretching	9.3	21.0	7.7	17.6	2.3	1.1	24.1	5.0	0.5	4.4
3	S2 train models to S2 test models	Mask2CAD	6.5	14.0	2.2	3.2	0.2	0.0	35.4	1.2	0.6	1.6
		Ours (Swin) no stretching	6.4	13.4	0.2	4.7	0.1	0.0	29.8	1.0	7.9	0.0
		Ours (Swin) with stretching	6.5	18.6	2.1	4.6	0.9	0.0	25.3	2.9	1.7	2.2
		Ours (GT) no stretching	7.2	12.0	4.5	4.7	0.0	0.0	37.0	2.3	3.9	0.0
		Ours (GT) with stretching	11.6	24.9	11.3	4.6	3.3	4.7	35.8	11.0	2.4	6.7

b)

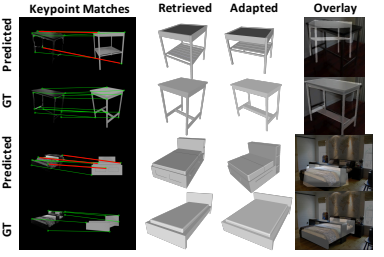


Figure 6: a) **Quantitative results on Pix3D** when no access to correct models is provided at test time. For the first two main rows (containing two sub-rows each) CAD models in the database are randomly stretched along all 3 principal directions and our method has to recover the original shape. For the third main row S2 test CAD models have to be estimated when the retrieval network has no access to the correct models and differing S2 train CAD models have to be adapted. b) **Visual comparison** of shape prediction with predicted (row one and three) and ground truth (row two and four) segmentation masks.

tained by multiplying a random number from a uniform distribution on the interval [-0.2, 0.3] by the side length of the bounding box of the object in the relevant direction. Experiments on *stretched S1 models* demonstrate that shape adaptation substantially improves the shape predictions (15.5 vs. 4.4, see Figure 6). For *stretched S2 models* we can observe significant improvements for certain classes such as bookcases, chairs or tables. However, the overall accuracy gain is smaller (9.3 vs. 8.5) and some classes, most notably sofas, show worse performance. As explained above very accurate shape models already exist for sofa objects in the non-stretched part of the database. Hence, stretching from sometimes poor segmentation masks and corresponding keypoint matches can deteriorate the estimated shapes. To further investigate the performance of the system when enabling shape deformations, we perturb the models in the database with progressively larger perturbations, and compare the predictions generated when stretching is allowed and disallowed (see Figure 5 b). As CAD model perturbations are increased, the proposed shape adaptation method can maintain a high accuracy while estimating the poses and of retrieved objects without using shape adaptation quickly becomes inaccurate. Finally, when estimating shapes *on S2 test images while only allowing to retrieve from the S2 train model set* (see Figure 6) we observe that similar to the experiments on the *stretched S2 models* we show improvements only for some classes (e.g. beds). This is due to the poor segmentation quality which prevents the matching network from successfully establishing a sufficient number of correspondences. When ground truth masks are used instead, significant improvements for almost all categories are observed.

6 Conclusion

In this work we propose to leverage geometric constraints for precisely estimating 3D object shapes from retrieved CAD models. We demonstrate that our approach is more accurate than direct pose prediction [17, 33] and the image retrieval-based approaches [18, 19]. Further, we show that by allowing object stretching we can modify retrieved CAD models to better fit the observed shapes. We believe that adapting retrieved CAD models is a promising avenue for future research as it combines the reliability of object retrieval with the expressiveness of generative approaches.

Acknowledgments

This work was supported by the Engineering and Physical Sciences Research Council [RG105266].

References

[1] Mathieu Aubry, Daniel Maturana, Alexei Efros, Bryan Russell, and Josef Sivic. Seeing 3D chairs: exemplar part-based 2D-3D alignment using a large dataset of CAD models. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, Colombus, Ohio, US, June 2014.

[2] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015.

[3] Gal Chechik, Varun Sharma, Uri Shalit, and Samy Bengio. Large Scale Online Learning of Image Similarity Through Ranking. *Journal of Machine Learning Research*, 11 (36):1109–1135, 2010.

[4] Ding-Yun Chen, Xiao-Pei Tian, Yu-Te Shen, and Ming Ouhyoung. On Visual Similarity Based 3D Model Retrieval. *Computer Graphics Forum*, 22(3):223–232, 2003.

[5] Zhiqin Chen, Andrea Tagliasacchi, and Hao Zhang. BSP-Net: Generating Compact Meshes via Binary Space Partitioning. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, Virtual, June 2020.

[6] Christopher B. Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3D-R2N2: A Unified Approach for Single and Multi-view 3D Object Reconstruction. In *Proc. 14th European Conference on Computer Vision*, Amsterdam, NL, October 2016.

[7] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 886–893 vol. 1, 2005.

[8] Boyang Deng, Kyle Genova, Soroosh Yazdani, Sofien Bouaziz, Geoffrey Hinton, and Andrea Tagliasacchi. CvxNet: Learnable Convex Decomposition. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, Virtual, June 2020.

[9] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. SuperPoint: Self-Supervised Interest Point Detection and Description. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, Salt Lake City, Utah, USA, June 2018.

[10] Haoqiang Fan, Hao Su, and Leonidas Guibas. A Point Set Generation Network for 3D Object Reconstruction from a Single Image. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, Honolulu, USA, July 2017.

[11] Georgios Georgakis, Srikrishna Karanam, Ziyan Wu, and Jana Kosecka. Learning Local RGB-to-CAD Correspondences for Object Pose Estimation. In *Proc. IEEE Int. Conf. on Computer Vision*, Seoul, Korea, October 2019.

[12] Georgia Gkioxari, Jitendra Malik, and Justin Johnson. Mesh R-CNN. In *Proc. IEEE Int. Conf. on Computer Vision*, Seoul, Korea, October 2019.

[13] Alexander Grabner, Peter M. Roth, and Vincent Lepetit. Location Field Descriptors: Single Image 3D Model Retrieval in the Wild. In *2019 International Conference on 3D Vision (3DV)*, pages 583–593, 2019.

[14] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *Proc. IEEE Int. Conf. on Computer Vision*, Venice, Italy, October 2017.

[15] Hamid Izadinia, Qi Shan, and Steven M. Seitz. IM2CAD. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, Honolulu, USA, July 2017.

[16] Laurent Kneip, Hongdong Li, and Yongduek Seo. UPnP: An Optimal O(n) Solution to the Absolute Pose Problem with Universal Applicability. In *Proc. 13th European Conference on Computer Vision*, Zurich, Switzerland, September 2014.

[17] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. *Commun. ACM*, 60(6):84–90, May 2017.

[18] Weicheng Kuo, Anelia Angelova, Tsung-Yi Lin, and Angela Dai. Mask2CAD: 3D Shape Prediction by Learning to Segment and Retrieve. In *Proc. 16th European Conference on Computer Vision*, Glasgow, UK, August 2020.

[19] Weicheng Kuo, Anelia Angelova, Tsung-yi Lin, and Angela Dai. Patch2CAD: Patchwise Embedding Learning for In-the-Wild Shape Retrieval from a Single Image. In *Proc. IEEE Int. Conf. on Computer Vision*, Montreal (Virtual), October 2021.

[20] Yangyan Li, Hao Su, Charles Ruizhongtai Qi, Noa Fish, Daniel Cohen-Or, and Leonidas J. Guibas. Joint Embeddings of Shapes and Images via CNN Image Purification. *ACM Trans. Graph.*, 34(6), October 2015.

[21] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft COCO: Common Objects in Context, 2015.

[22] Dong C. Liu and Jorge Nocedal. On the Limited Memory BFGS Method for Large Scale Optimization. *Mathematical Programming*, 45:503–528, 1989.

[23] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin Transformer: Hierarchical Vision Transformer using Shifted Windows. In *Proc. IEEE Int. Conf. on Computer Vision*, Montreal (Virtual), October 2021.

[24] Kevis-Kokitsi Maninis, Stefan Popov, Matthias Nießner, and Vittorio Ferrari. Vid2cad: Cad model alignment using multi-view constraints from videos. *arXiv*, 2020.

[25] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. In *Proc. 16th European Conference on Computer Vision*, Glasgow, UK (Virtual), August 2020.

[26] Yinyu Nie, Xiaoguang Han, Shihui Guo, Yujian Zheng, Jian Chang, and Jian Jun Zhang. Total3DUnderstanding: Joint Layout, Object Pose and Mesh Reconstruction for Indoor Scenes from a Single Image. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, Virtual, June 2020.

[27] Junyi Pan, Xiaoguang Han, Weikai Chen, Jiapeng Tang, and Kui Jia. Deep Mesh Reconstruction from Single RGB Images via Topology Modification Networks. In *Proc. IEEE Int. Conf. on Computer Vision*, Seoul, Korea, October 2019.

[28] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, Long Beach, CA, USA, June 2019.

[29] Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *3rd International Conference on Learning Representations*, San Diego, CA, USA, May 2015.

[30] Xingyuan Sun, Jiajun Wu, Xiuming Zhang, Zhoutong Zhang, Chengkai Zhang, Tianfan Xue, Joshua B Tenenbaum, and William T Freeman. Pix3D: Dataset and Methods for Single-Image 3D Shape Modeling. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, Salt Lake City, Utah, USA, June 2018.

[31] Maxim Tatarchenko, Stephan R. Richter, René Ranftl, Zhuwen Li, Vladlen Koltun, and Thomas Brox. What Do Single-view 3D Reconstruction Networks Learn? In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, Long Beach, CA, USA, June 2019.

[32] Shubham Tulsiani, Hao Su, Leonidas J. Guibas, Alexei A. Efros, and Jitendra Malik. Learning Shape Abstractions by Assembling Volumetric Primitives. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2018.

[33] Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang. Pixel2Mesh: Generating 3D Mesh Models from Single RGB Images. In *Proc. 15th European Conference on Computer Vision*, Munich, Germany, September 2018.

[34] Wei Zeng, Sezer Karaoglu, and Theo Gevers. Inferring Point Clouds from Single Monocular Images by Depth Intermediation. *arXiv*, 2020.