

ObjectNet3D: A Large Scale Database for 3D Object Recognition

Yu Xiang^(✉), Wonhui Kim, Wei Chen, Jingwei Ji, Christopher Choy,
Hao Su, Roozbeh Mottaghi, Leonidas Guibas, and Silvio Savarese

Stanford University, Stanford, USA
yuxiang@cs.stanford.edu

Abstract. We contribute a large scale database for 3D object recognition, named ObjectNet3D, that consists of 100 categories, 90,127 images, 201,888 objects in these images and 44,147 3D shapes. Objects in the 2D images in our database are aligned with the 3D shapes, and the alignment provides both accurate 3D pose annotation and the closest 3D shape annotation for each 2D object. Consequently, our database is useful for recognizing the 3D pose and 3D shape of objects from 2D images. We also provide baseline experiments on four tasks: region proposal generation, 2D object detection, joint 2D detection and 3D object pose estimation, and image-based 3D shape retrieval, which can serve as baselines for future research using our database. Our database is available online at <http://cvgl.stanford.edu/projects/objectnet3d>.

Keywords: Database construction · 3D object recognition

1 Introduction

Recognizing 3D properties of objects from 2D images, such as 3D location, 3D pose and 3D shape, is a central problem in computer vision that has wide applications in different scenarios including robotics, autonomous driving and augmented reality. In recent years, remarkable progress has been achieved on 3D object recognition (e.g. [9, 15, 24, 33, 39, 42]), as the field has benefited from the introduction of several important databases that provide 3D annotations to 2D objects. For example, the NYU Depth dataset [29] associates depth to 2D images; the KITTI dataset for autonomous driving [10] aligns 2D images with 3D point clouds, and the PASCAL3D+ dataset [40] aligns 2D objects in images with 3D CAD models. With the provided 3D information, supervised learning techniques can be applied to recognize 3D properties of objects. In addition, these datasets serve as benchmarks for comparing different approaches.

However, the existing databases with 3D annotations are limited in scale, either in the number of object categories or in the number of images. At least,

Electronic supplementary material The online version of this chapter (doi:10.1007/978-3-319-46484-8_10) contains supplementary material, which is available to authorized users.

they are not comparable to large scale 2D image databases such as ImageNet [1] or Microsoft COCO [21]. After witnessing the progress on image classification, 2D object detection and segmentation with the advance of such large scale 2D image databases, we believe that a large scale database with 3D annotations would significantly benefit 3D object recognition.

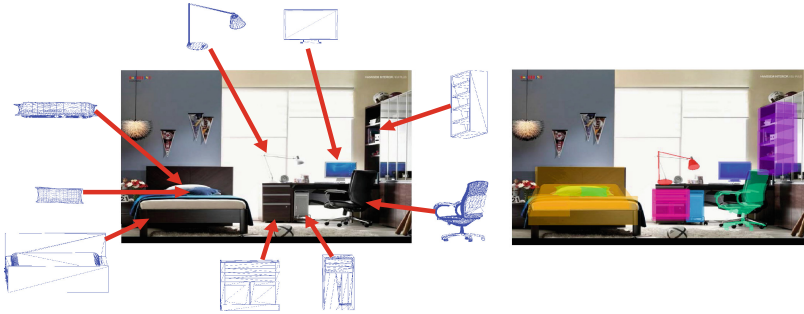


Fig. 1. An example image in our database with 2D objects aligned with 3D shapes. The alignment enables us to project each 3D shape to the image where its projection overlaps with the 2D object as shown in the image on the right

In this work, we contribute a large scale database for 3D object recognition, named ObjectNet3D, that consists of 100 object categories, 90,127 2D images, 201,888 objects in these images and 44,147 3D shapes. The images in our database are collected from the ImageNet repository [1], while the 3D shapes are from the ShapeNet repository [4]. In addition to 2D bounding box annotations for objects of interest, a key aspect of our new database is that each object in an image is aligned with a 3D shape. The alignment enables us to project the 3D shape to the image, where the projection of the 3D shape matches the corresponding 2D object in the image (see an example in Fig. 1). The alignment between 2D and 3D provides 3D annotations to objects in 2D images, i.e., the 3D pose annotation and the closest 3D shape annotation for a 2D object. As a result, our database is useful for 3D object recognition from 2D images. Projection of the 3D shape also produces segmentation boundaries of the object. In addition, we can render an arbitrary number of synthetic images from the 3D shapes. The rendered images can also be used to help 3D object recognition in real images as demonstrated by [34].

The task of aligning a 2D object with a 3D shape is non-trivial. First, we need to select a 3D shape that is similar to the given 2D object from hundreds or thousands of 3D shapes. It is not feasible to ask annotators to go through all the 3D shapes one by one. Second, orienting the pose of the 3D shape for alignment is error-prone, so it is not easy to control the quality of the alignment. (i) To facilitate the selection of 3D shapes, we have applied the deep metric learning method [31] to learn a feature embedding of the 3D shapes using their rendered images. Given a 2D object in an image, our method is able to retrieve top ranked

similar 3D shapes using the learned embedding. Then we ask annotators to select the best one among the top K returned 3D shapes. (ii) To guarantee the quality of the alignment, we have designed an annotation tool to align the 3D shape with the 2D object. Our annotation interface allows annotators to interactively find a set of camera parameters for each object that produce good alignment.

We have conducted baseline experiments on the following tasks using our database: object proposal generation, 2D object detection, joint 2D detection and 3D object pose estimation, and image-based 3D shape retrieval. These experiments can serve as baselines for future research.

2 Related Work

We review representative datasets related to 3D object recognition.

Datasets with viewpoints. In these datasets [19,22,23,28,36], objects are annotated with both bounding boxes and viewpoints. Generally speaking, most datasets with viewpoint annotations are small in scale, coarse in viewpoint discretization and simple in scene context. For example, the 3DObject dataset [28] provides viewpoint annotation for 10 everyday object classes such as car, iron and stapler, with 10 instances per category observed from varying viewpoints. The EPFL Car dataset [23] consists of 2,299 images of 20 car instances at multiple azimuth angles, with almost identical elevation. Compared to these datasets, we provide continuous viewpoint annotation to realistic images from the web.

Datasets with depths or 3D points. Datasets in which 2D images are registered with depth or 3D points are introduced. The RGB-D Object dataset [18] provides RGBD images of 300 common household objects organized in 51 categories which are captured from a turntable. The NYU depth dataset [29] contains 1,449 densely labeled pairs of aligned RGB and depth images, where objects from 894 categories are labeled. The SUN RGB-D dataset [32] has 10K RGBD images and provides 2D polygons, 3D bounding boxes with orientations and 3D room layout annotations. The KITTI dataset [10] proposed for autonomous driving registers images with 3D point clouds from a 3D laser scanner. Compared to these datasets, we align a 3D shape to each 2D object and provide 3D shape annotation to objects, which is richer information than depth or 3D points and allows us to transfer meta-data from the 3D shape back to the image.

Datasets with 2D-3D alignments. An influential work in building datasets with 2D-3D alignment is LabelMe3D [27] that estimates the 3D scene structure based on user annotations. Recently, a few datasets were introduced that align 2D objects in images with 3D shapes. The IKEA dataset [20] provides 759 images aligned with 213 3D shapes which are IKEA 3D models from Trimble 3D Warehouse [5]. [17] introduces a new dataset for 2D-3D deformable shape matching. PASCAL3D+ [40] provides 2D-3D alignments to 12 rigid categories in PASCAL VOC 2012 [8]. Although PASCAL3D+ contains 30,899 images, it is still limited in the number of object classes (12 in total) and the number of 3D shapes (79 in total), so it is insufficient to cover the variations of common object

Table 1. Comparison between representative datasets with 3D annotations

	# categories	# images	# 3D shapes	3D annotation type
3DObject [28]	10	6,675	N/A	Discretized view
EPFL Car [23]	1	2,299	N/A	Continuous view
NYU Depth [29]	894	1,449	N/A	Depth
SUN RGB-D [32]	~ 800	10,335	N/A	Depth
KITTI [10]	2	14,999	N/A	3D point
IKEA [20]	11	759	213	2D-3D alignment
PASCAL3D+ [40]	12	30,899	79	2D-3D alignment
ObjectNet3D (Ours)	100	90,127	44,161	2D-3D alignment

categories and their geometry variability. We contribute a large-scale 3D object dataset with more object categories, more 3D shapes per class and accurate image-shape correspondences. Table 1 compares our dataset to representative datasets in the literature with 3D annotations.

3 Database Construction

Our goal is to build a large scale database for 3D object recognition. We resort to images in existing image repositories and propose an approach to align 3D shapes (which are available from existing 3D shape repositories) to the objects in these images. In this way, we have successfully built the ObjectNet3D database.

3.1 Object Categories

We aim at building a database for object category recognition, so the first step is to decide what categories to work on. Since we are going to utilize 3D object shapes to provide 3D annotations to objects in 2D images, we consider only rigid object categories in this work, where we can collect a large number of 3D shapes for these categories from the web. For deformable and articulated objects such as animals, deformation and articulation of a 3D shape is required in order to align the shape with the corresponding object in a 2D image. We consider the extension to non-rigid object categories as a future work. Table 2 lists all 100 object categories in our database. These categories cover most of the rigid categories in the commonly used object recognition datasets, such as the 12 rigid categories in PASCAL VOC [8] and the 9 rigid categories in [28], but our dataset is one order of magnitude larger in terms of the number of categories.

3.2 2D Image Acquisition

After finalizing the object categories in the database, we collect images for each object category from the ImageNet database [1]. ImageNet organizes images according to the WordNet hierarchy. Each node in the hierarchy is named a

Table 2. 100 object categories in our database

Aeroplane	Camera	Eraser	Jar	Pencil	Shovel	Toothbrush
Ashtray	Can	Eyeglasses	Kettle	Piano	Sign	Train
Backpack	Cap	Fan	Key	Pillow	Skate	Trash bin
Basket	Car	Faucet	Keyboard	Plate	Skateboard	Trophy
Bed	Cellphone	Filing cabinet	Knife	Pot	Slipper	Tub
Bench	Chair	Fire extinguisher	Laptop	Printer	Sofa	Tvmonitor
Bicycle	Clock	Fish tank	Lighter	Racket	Speaker	Vending machine
Blackboard	Coffee maker	Flashlight	Mailbox	Refrigerator	Spoon	Washing machine
Boat	Comb	Fork	Microphone	Remote control	Stapler	Watch
Bookshelf	Computer	Guitar	Microwave	Rifle	Stove	Wheelchair
Bottle	Cup	Hair dryer	Motorbike	Road pole	Suitcase	
Bucket	Desk lamp	Hammer	Mouse	Satellite dish	Teapot	
Bus	Diningtable	Headphone	Paintbrush	Scissors	Telephone	
Cabinet	Dishwasher	Helmet	Pan	Screwdriver	Toaster	
Calculator	Door	Iron	Pen	Shoe	Toilet	

synset that corresponds to a noun (e.g., car, boat, chair, etc.), and each synset contains images depicting the concept. For each object category in our database, we first find its corresponding synset in ImageNet, then we download images in the synset. For can, desk lamp and trophy, we did not find the corresponding synsets in ImageNet. For fork and iron, there are a limited number of images in ImageNet. Thus, we crawl images for these 5 categories using Google Image Search. Figure 2 displays some images from different object categories in our database. For most of these images, there are salient objects from the corresponding category in the image. Some objects tend to appear together, such as teapot and cup, bed and pillow, keyboard and mouse, and so on. Some objects may appear in complex scenes, for instance, a car in a street scene or a bed in an indoor scene.

3.3 3D Shape Acquisition

In order to provide 3D annotations to objects in 2D images, we collect 3D shapes for the object categories in our database. First, we manually select representative 3D shapes for each category from Trimble 3D Warehouse [5]. These 3D shapes are selected to cover different “subcategories”. For example, we collect 3D shapes of sedans, SUVs, vans, trucks, etc., for the car category. Figure 3(a)

**Fig. 2.** Example images in our database



Fig. 3. Examples of the 3D shapes for bench in our database. (a) 3D shapes manually selected from Trimble 3D Warehouse. (b) 3D Shapes collected from ShapeNet

shows the seven 3D shapes we collected for bench, where they represent different types of benches. These 3D shapes have been aligned according to the main axis of the category (e.g., front view of bench), with their sizes normalized to fit into a unit sphere. In addition, we have manually selected key points on each 3D shape as illustrated by the red dots in Fig. 3(a), which can be used for key point recognition in images or in 3D shapes. There are 783 3D shapes in total that are collected from Trimble 3D Warehouse in this way, which cover all the 100 categories.

Second, to increase the number of 3D shapes in our database, we download 3D shapes from the ShapeNet repository [4]. Similar to ImageNet, ShapeNet organizes 3D shapes according to the WordNet hierarchy. We use the ShapeNet-Core subset [7] since all models are single 3D objects with manually verified category annotations and alignment. ShapeNetCore covers 55 object categories, among which 42 categories overlap with the 100 categories in our database. So we download 3D shapes from these 42 categories in ShapeNetCore contributing additional 43,364 3D shapes to our database. Figure 3(b) shows some 3D shapes of benches from the ShapeNetCore repository. These 3D models are valuable since they capture more shape variations and have rich texture/material information.

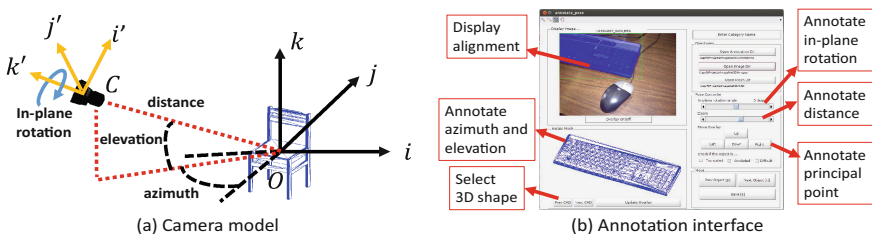


Fig. 4. Illustration of the camera model in our database (a) and the annotation interface for 2D-3D alignment (b)

3.4 Camera Model

After collecting the images and the 3D shapes, our next step is to align an object in an image with a 3D shape. We describe the camera model we use for the alignment as illustrated in Fig. 4(a).

First, the world coordinate system O is defined on the center of the 3D shape, with the three axes (i, j, k) aligned with the dominating directions of the 3D shape. Second, the camera coordinate system C is denoted by (i', j', k') , and we assume that the camera is facing the negative direction of the k' axis towards the origin of the world coordinate system. In this case, the rotation transformation between the two coordinate systems can be defined by three variables: azimuth a , elevation e and in-plane rotation θ . Let's denote the rotation matrix by $R(a, e, \theta)$. The translation vector of the camera center in the world coordinate system can be defined by azimuth, elevation and distance d as $T(a, e, d)$. R and T determine the extrinsic parameters of our camera model. Third, for the intrinsic parameters of the camera, we use a virtual focus length f that is fixed as one. We define the viewport size as $\alpha = 2000$ (i.e., unit one in the world coordinate system corresponds to 2,000 pixels in the image), and denote the principal point as (u, v) (i.e., the projection of the world coordinate origin in the image). Finally, the projection matrix M of our camera model is

$$M = \underbrace{\begin{bmatrix} \alpha f & 0 & u \\ 0 & \alpha f & v \\ 0 & 0 & 1 \end{bmatrix}}_{\text{intrinsic parameters}} \underbrace{[R(a, e, \theta) T(a, e, d)]}_{\text{extrinsic parameters}}. \quad (1)$$

By fixing the focal length f and the viewport α , we have 6 variables to be estimated: azimuth a , elevation e , in-plane rotation θ , distance d and principal point (u, v) , which define the alignment between the 3D shape and the 2D object.

3.5 Annotation Process

We describe our annotation process to provide 3D annotations to objects in 2D images. (i) We label the 2D bounding box of every object in the images that

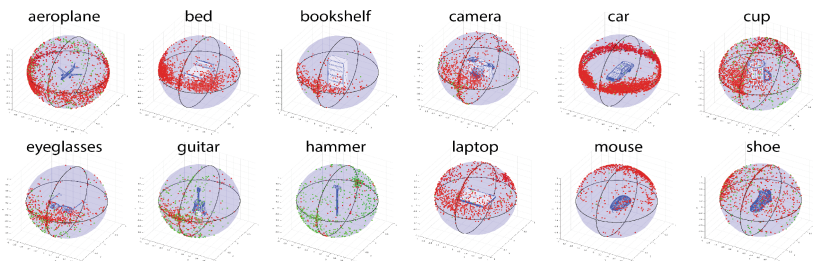


Fig. 5. Viewpoint distributions of different categories in our database. We visualize the camera position as a point on the unit sphere (red points: in-plane rotation $< 15^\circ$; green points: in-plane rotation $> 15^\circ$). See [3] for more plots (Color figure online)

belong to the 100 categories in our database. Occluded objects and truncated objects are also labeled. (ii) Given an object indicated by its bounding box, we associate it to the most similar 3D shape among those downloaded from Trimble 3D Warehouse (Fig. 3(a)). That is, an annotator is asked to select the most similar 3D shape from 7.8 3D shapes for each object on average. (iii) We align the selected 3D shape to the 2D object using the camera model described in Sect. 3.4. For the alignment, we have developed an annotation interface as illustrated in Fig. 4(b). Objects are shown one by one in the annotation tool, and annotators can modify the camera parameters via the interface to align the objects. Annotators have full control of all the 6 camera parameters using the interface: azimuth, elevation, distance, in-plane rotation and principal point. Whenever these parameters are changed, we re-project the 3D shape to the image and display the overlap, which is helpful for the annotator to find a set of camera parameters that align the 3D shape with the 2D object well. Our criterion for the alignment is maximizing the intersection over union between the projection of the 3D shape and the 2D object. Figure 4(b) shows the finished alignment for a computer keyboard. Figure 5 illustrates the viewpoint distribution of several categories in our database.

3.6 3D Shape Retrieval

In our annotation process, we have manually associated every object in the 100 categories to a 3D shape among these 3D models from Trimble 3D Warehouse, where we have around 7 or 8 3D shapes per category. For 42 object categories among the 100 categories, we have additional 3D shapes from ShapeNetCore (Fig. 3(b)). In this case, it is not feasible to manually select the most similar 3D shape among thousands of 3D shapes. So we develop a 3D shape retrieval method by learning feature embeddings with rendered images, and we use this method to retrieve the closest 3D shapes for objects in the 42 object categories.

Specifically, given a 2D object o and a set of N 3D shapes $\mathcal{S} = \{S_1, \dots, S_N\}$, our goal is to rank the N 3D shapes according to their similarity with the 2D object. We formulate it as a metric learning problem, where the task is to learn a distance metric between a 2D object and a 3D shape: $D(o, S)$. To bridge the two different domains, we use rendered images to represent a 3D shape: $S = \{s_1, \dots, s_n\}$, where s_i denotes the i th rendered image from the 3D shape S and n is the number of rendered images ($n = 100$ in our experiments). Then we define the distance between a 2D object and a 3D shape as the mean distance between the 2D object and the rendered images from the 3D shape: $D(o, S) = \frac{1}{n} \sum_{i=1}^n D(o, s_i)$. Now, the task converts to learning the distance metric $D(o, s_i)$ between 2D images, which is an active research field in the literature.

We apply the lifted structured feature embedding method [31] to learn the distance metric between images, which achieves better performance than contrastive embedding [13] and triplet embedding [38]. The training is conducted with rendered images only, where images rendered from the same 3D shape are considered to be in the same class. As a result, the learned feature embedding will group images from the same shape together. In testing, given a 2D object

o , we compute its euclidean distance $D(o, s_i)$ with each rendered image s_i in the embedding space, and average them to compute its distances with 3D shapes. In order to minimize the gap between rendered images and real test images, we add backgrounds to rendered images and vary their lighting conditions as in [34].

4 Baseline Experiments

In this section, we provide baseline experimental results on four different tasks: object proposal generating, 2D object detection, joint 2D detection and 3D pose estimation, and image-based 3D shape retrieval. We split the images in our dataset into a training/validation (trainval) set with 45,440 images, and a test set with 44,687 images. In the following experiments, training is performed on the trainval set, while testing is conducted on the test set.

4.1 Object Proposal Generation

Recent progress in object recognition can largely be attributed to the advance of Deep Neural Networks (DNNs). Region proposals are widely used in different DNN-based object recognition methods [12] as a preprocessing step to reduce the search space on images. The main idea is to generate a few hundreds or thousands of regions per image that are likely to be objects, then these regions are classified with a DNN. We first apply four different region proposal methods to our dataset and evaluate their performances: SelectiveSearch [37], EdgeBoxes [43], Multiscale Combinatorial Grouping (MCG) [6] and Region Proposal Network (RPN) [25].

We use detection recall to evaluate the region proposal performance. It is defined as the percentage of Ground Truth (GT) object boxes that are correctly covered by the region proposals, and we say a GT box is correctly covered if its Intersection over Union (IoU) with one of the region proposals is larger than some threshold. Figure 6 presents the detection recall of these four region proposal

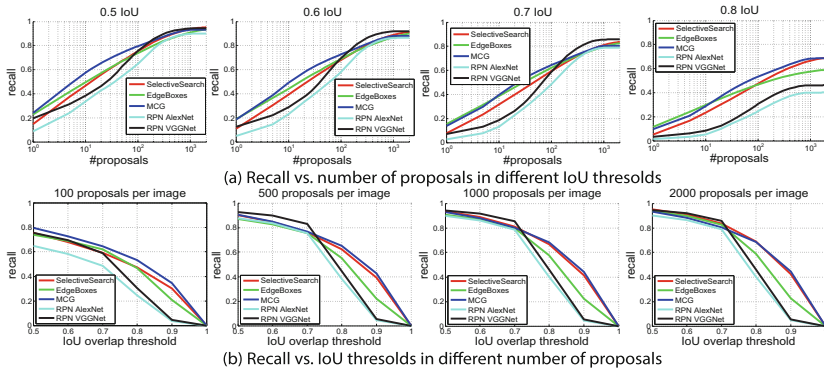


Fig. 6. Evaluation of four different object proposal method on our dataset: SelectiveSearch [37], EdgeBoxes [43], MCG [6] and RPN [25]

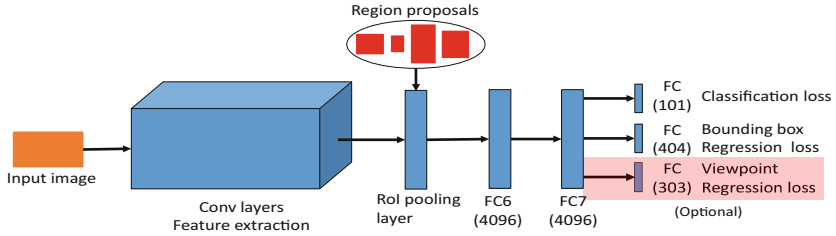


Fig. 7. Illustration of the network architecture based on Fast R-CNN [11] for object detection and pose estimation

methods on our test set according to different number of object proposals per image and different IoU thresholds. For the RPN in [25], we experiment with two network architectures, i.e., AlexNet [16] and VGGNet [30] (VGG16), where we fine-tune their pre-trained networks on ImageNet [26] with our trainval set for region proposal generation. First, we can see that, by using around 1000 proposals, all the four methods achieve recall more than 90% with 0.5 IoU threshold. The state-of-the-art region proposal methods work well on our dataset. Second, it is interesting to note that the recall of RPN drops significantly when an IoU threshold larger than 0.7 is used. This is because the RPNs are trained with 0.7 IoU threshold, i.e., all proposals with IoU larger than 0.7 are treated as positive examples equally. There is no constraint in the RPN training to ensure that proposals with larger IoU threshold are preferred. Third, RPN with VGGNet achieves the best recall with IoU threshold from 0.5 to 0.7, while MCG performs consistently well across different IoU thresholds.

4.2 2D Object Detection

We evaluate the 2D object detection performance on our dataset using the Fast R-CNN framework [11] which is considered to be a state-of-the-art object detection method. We fine-tune two CNN architectures pre-trained on the ImageNet dataset [26] for the 2D detection task: AlexNet [16] and VGGNet [30] (VGG16). Figure 7 illustrates the network architecture used in Fast R-CNN for object detection. First, an input image is fed into a sequence of convolutional layers to compute a feature map of the image. Then, given a region proposal, the RoI pooling layer extracts a feature vector for the region proposal from the feature map. The feature vector is then processed by two Fully Connected (FC) layers, i.e., FC6 and FC7 each with dimension 4096. Finally, the network terminates at two FC branches with different losses (i.e., the third branch for viewpoint estimation in Fig. 7 is not used here), one for object class classification, and the other one for bounding box regression (see [11] for more details). We have 100 categories in our dataset, so the FC layer for classification has dimension 101 with an additional dimension for background. The FC layer for bounding box regression has dimension 4×101 , i.e., for each class, it predicts the center location, width, height of the bounding box.

Table 3. Object detection evaluation in terms of mAP with the four object proposals and two CNN architectures in the Fast R-CNN framework

	SelectiveSearch [37]	EdgeBoxes [43]	MCG [6]	RPN [25]
AlexNet [16]	56.3	52.5	56.0	54.2
VGGNet [30]	67.3	64.5	67.0	67.5

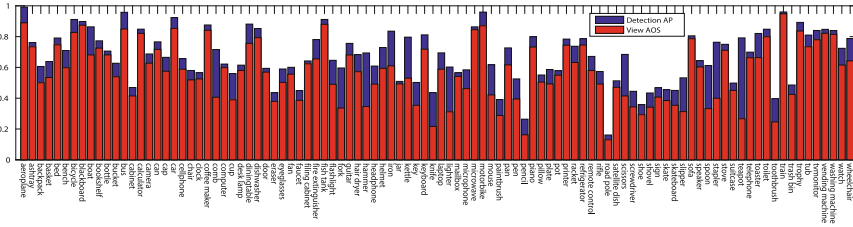


Fig. 8. Bar plot of the detection AP and viewpoint estimation AOS of the 100 categories on the test set from VGGNet with SelectiveSearch proposals

We use Average Precision (AP) to evaluate the 2D object detection performance. AP is computed as the area under the precision-recall curve for each category, where 50 % overlap threshold is used as in PASCAL VOC [8]. To evaluate the overall detection performance on the dataset, we compute mean AP (mAP) across all the categories. Table 3 presents the mAP for 2D object detection across all the 100 categories on the test set. First, we can see that VGGNet achieves significantly better mAPs compared to AlexNet. Second, among the four region proposal methods, using object proposals from SelectiveSearch and MCG achieves better detection performance than using EdgeBoxes. RPN is able to benefit from more powerful networks such as the VGGNet. Third, with VGGNet and RPN proposals, we achieve the best mAP 67.5. For reference, the best mAP on the ImageNet detection challenge 2015 (200 ImageNet categories) is 62.0 [2]. Figure 8 shows the detection AP of each category in our dataset. As we can see, some categories are relatively easy to detect such as aeroplane, motorbike and train, and some are more difficult such as cabinet, pencil and road pole. These categories either have large intra-class variability or have less discriminative features. Finally, we group all 100 categories into six super-categories: container, electronics, furniture, personal items, tools and vehicles, and analyze the detection false positives of these six groups using the diagnosing tool from [14]. Figure 9 summarizes the results. For tools and vehicles, localization error is the main source of false positives, while for the other four groups, a large portion of the detection errors is also attributed to confusion with other categories or background.

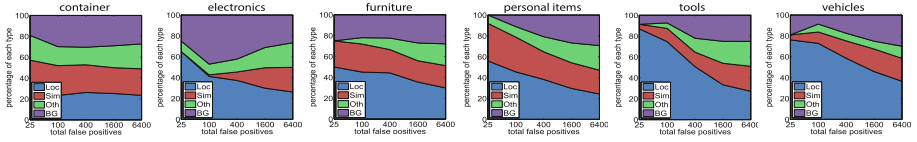


Fig. 9. Distribution of top-ranked false positive types from VGGNet with SelectiveSearch proposals: Loc - pool localization; Sim - confusion with a similar category; Oth - confusion with a dissimilar category; BG - a false positive fires on background

4.3 Joint 2D Detection and Continuous 3D Pose Estimation

In this experiment, our goal is to jointly detect objects in 2D images and estimate their continuous 3D pose. By aligning 3D shapes with 2D objects, we provide continuous 3D pose annotations to objects in our dataset. We provide a baseline model for this task by modifying the Fast R-CNN network. As illustrated in Fig. 7, we add a viewpoint regression FC branch after the FC7 layer. So the network is trained to perform three tasks jointly: classification, bounding box regression and viewpoint regression. The FC layer for viewpoint regression has dimension 3×101 , i.e., for each class, it predicts the three angles of azimuth, elevation and in-plane rotation. The smoothed L1 loss is used for viewpoint regression as for bounding box regression.

Two different metrics have been proposed for joint detection and pose estimation: Average Viewpoint Precision (AVP) in PASCAL3D+ [40] and Average Orientation Similarity (AOS) in KITTI [10]. However, they only consider the estimation error in azimuth. In order to evaluate joint detection and continuous pose estimation in three angles, i.e., azimuth, elevation and in-plane rotation, we generalize AVP and AOS, where we define the difference between the estimated pose and the ground truth pose as follows: $\Delta(R, R_{gt}) = \frac{1}{\sqrt{2}} \|\log(R^T R_{gt})\|_F$, which is the geodesic distance between the estimated rotation matrix R and the GT rotation matrix R_{gt} . In AVP, we consider an estimation to be correct if $\Delta(R, R_{gt}) < \frac{\pi}{6}$. In AOS, the cosine similarity between poses is used: $\cos(\Delta(R, R_{gt}))$ [10].

Table 4. Joint 2D detection and 3D pose estimation evaluation in terms of AOS/AVP with the four object proposals and two CNNs in the Fast R-CNN framework

	SelectiveSearch [37]	EdgeBoxes [43]	MCG [6]	RPN [25]
	AOS / AVP			
AlexNet [16]	48.1 / 37.3	44.5 / 33.6	47.8 / 37.0	46.1 / 35.4
VGGNet [30]	57.1 / 43.0	54.2 / 39.6	56.8 / 42.9	57.0 / 42.6

Table 4 presents the joint detection and pose estimation results on the test set with four region proposal methods and two CNN architectures. Figure 8 shows

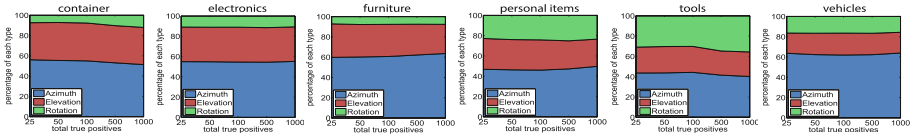


Fig. 10. Viewpoint error distribution of top-ranked true positives from VGGNet with SelectiveSearch proposals

the AOS of each category from the VGGNet with SelectiveSearch proposals. Due to the way AOS is computed, detection AP is always an upper bound of AOS. The closer AOS is to AP, the better the viewpoint estimation. By examining the gaps between AOS and AP in Fig. 8, we can figure out a few categories with poor viewpoint estimation such as comb, fork and teapot. These categories may be nearly symmetric or have large in-plane rotation angles. To understand the viewpoint error distribution in azimuth, elevation and in-plane rotation, we visualize it in Fig. 10. As we can see, azimuth error dominates for the six super-categories. For tools and personal items, such as hammer or watch, in-plane rotation error is also significant.

4.4 Image-Based 3D Shape Retrieval

In our annotation process, we propose an image-based 3D shape retrieval method by deep metric learning (Sect. 3.6). The goal is to find the most similar 3D shapes to a given 2D object. We present details of this experiment here. We learn a feature embedding for each category among the 45 categories where we have 3D shapes from ShapeNetCore. First, to generate training images, we render 100 synthetic images for each 3D shape from different viewpoints. These viewpoints are sampled from a distribution estimated with kernel density estimation using the viewpoint annotations (azimuth, elevation and in-plane rotation) in our database for that category. To mimic real images, we overlay the rendered images on randomly selected background images from the SUN database [41]. Second, we consider images rendered from the same 3D shape as in the same “class”. In this way, we are able to apply existing deep metric learning methods to learn a feature embedding for the rendered images. Specifically, we experiment with the contrastive embedding [13], the triplet embedding [38] and the lifted structured embedding [31], where we fine-tune GoogLeNet [35] pre-trained on ImageNet [26] to learn the embedding. Finally, after training the network, each rendered image is represented as a feature vector (in the embedded space) computed from the last FC layer of the GoogLeNet, while a 3D shape is represented as mean of the feature vectors of its rendered images.

To evaluate the learned embedding, we conduct the following image retrieval experiment with rendered images, where we have ground truth 3D shape assignments: given a rendered image, the goal is to retrieve images rendered from the same 3D shape as the input image, which is exactly the task that the network is trained for. For each category, we use 50% of the 3D shapes for training, and test

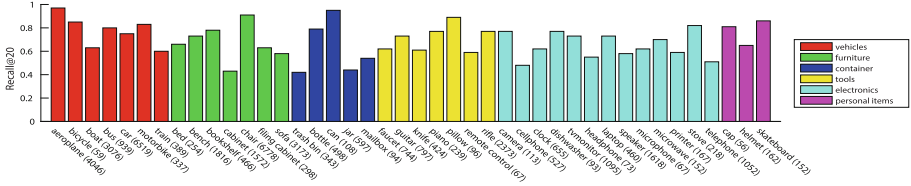


Fig. 11. Recall@20 from our user study for 42 categories that have 3D shapes from ShapeNetCore. The number of 3D shapes for each category is shown in the brackets

on the other 50 %. We compute Recall@K to evaluate the retrieval performance, which is computed as the percentage of testing images which have at least one correctly retrieved image among the top K retrieval results.

Table 5 shows the Recall@K on rendered image retrieval, where we compare the contrastive embedding, the triplet embedding and the lifted structured embedding. As we can see from the table, lifted structured embedding significantly outperforms the other two due to its ability to utilize every pairwise relationship between training examples in a batch [31]. Thus, we utilize the lifted structured embedding to retrieve 3D shapes for real images in our database. The goal is to provide the top K ranked 3D shapes for each 2D object, then ask annotators to select the most similar 3D shape among the K returned ones, since it is not feasible to ask an annotator to select the most similar shape

Table 5. Comparison between three feature embedding methods on rendered images

	Recall@1	Recall@2	Recall@4	Recall@8	Recall@16	Recall@32
Contrastive [13]	60.7±19.9	69.0±18.7	76.3±16.8	82.6±14.3	87.7±11.5	91.8±8.7
Triplet [38]	82.3±11.9	87.2±9.5	91.0±7.4	93.9±5.5	95.9±3.9	97.4±2.7
LiftedStruct [31]	91.1±4.5	94.1±3.2	96.1±2.3	97.4±1.7	98.3±1.3	98.9±0.9

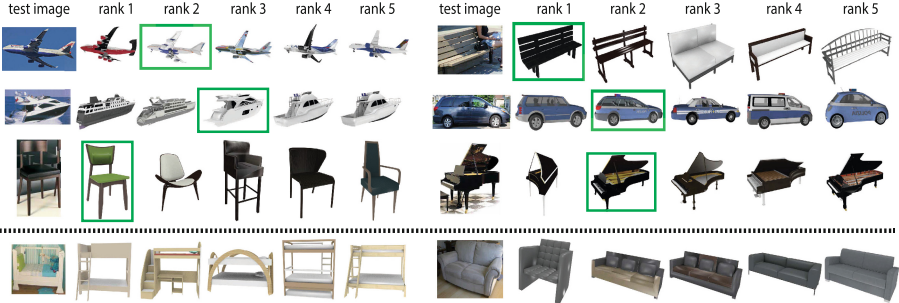


Fig. 12. Example of 3D shape retrieval. Green boxes are the selected shapes. The last row shows two examples where we cannot find a similar shape among the top 5 ones (Color figure online)

among hundreds or even thousands of 3D shapes. In this way, we are able to select a close 3D shape from ShapeNetCore for each 2D object in our dataset. Figure 12 shows some 3D shape retrieval examples using our learned lifted structured embeddings.

We have also conducted a user study to test the performance of our retrieval method on real images. We randomly sample 100 objects from each category, and ask three annotators to decide if there is a similar 3D shape among the top 20 retrieved 3D shapes for each object. Then we compute Recall@20 for each category based on the annotators' judgement. Figure 11 shows the results from the user study. The mean Recall@20 is 69.2%. The method works well on categories with a number of 3D shapes large enough to cover the shape variations (e.g. aeroplane, chair and car). On categories with fewer 3D shapes, the task becomes more challenging, especially when the category has large intra-class variability.

5 Conclusions

In this work, we have successfully built a large scale database with 2D images and 3D shapes for 100 object categories. We provide 3D annotations to objects in our database by aligning a closest 3D shape to a 2D object. As a result, our database can be used to benchmark different object recognition tasks including 2D object detection, 3D object pose estimation and image-based 3D shape retrieval. We have provided baseline experiments on these tasks, and demonstrated the usefulness of our database.

Acknowledgments. We acknowledge the support of NSF grants IIS-1528025 and DMS-1546206, a Google Focused Research award, and grant SPO # 124316 and 1191689-1-UDAWF from the Stanford AI Lab-Toyota Center for Artificial Intelligence Research.

References

1. <http://image-net.org/>
2. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C.: Imagenet large scale visual recognition challenge. *Int. J. Comput. Vis.* **115**(3), 211–252 (2015). <http://image-net.org/challenges/ilsrvrc+mscoco2015>
3. Objectnet3d. <http://cvgl.stanford.edu/projects/objectnet3d>
4. Shapenet. <http://shapenet.cs.stanford.edu/>
5. Trimble 3D warehouse. <http://3dwarehouse.sketchup.com>
6. Arbeláez, P., Pont-Tuset, J., Barron, J., Marques, F., Malik, J.: Multiscale combinatorial grouping. In: CVPR, pp. 328–335 (2014)
7. Chang, A.X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., Xiao, J., Yi, L., Yu, F.: ShapeNet: an information-rich 3D model repository. Technical report [cs.GR] (2015). [arXiv:1512.03012](https://arxiv.org/abs/1512.03012)

8. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. <http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html>
9. Fidler, S., Dickinson, S., Urtasun, R.: 3D object detection and viewpoint estimation with a deformable 3D cuboid model. In: NIPS, pp. 611–619 (2012)
10. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? the kitti vision benchmark suite. In: CVPR, pp. 3354–3361 (2012)
11. Girshick, R.: Fast R-CNN. In: ICCV, pp. 1440–1448 (2015)
12. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: CVPR, pp. 580–587 (2014)
13. Hadsell, R., Chopra, S., LeCun, Y.: Dimensionality reduction by learning an invariant mapping. In: CVPR, vol. 2, pp. 1735–1742 (2006)
14. Hoiem, D., Chodpathumwan, Y., Dai, Q.: Diagnosing error in object detectors. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012, Part III. LNCS, vol. 7574, pp. 340–353. Springer, Heidelberg (2012)
15. Kar, A., Tulsiani, S., Carreira, J., Malik, J.: Category-specific object reconstruction from a single image. In: CVPR, pp. 1966–1974 (2015)
16. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: NIPS, pp. 1097–1105 (2012)
17. Löhner, Z., Rodola, E., Schmidt, F.R., Bronstein, M.M., Cremers, D.: Efficient globally optimal 2D-to-3D deformable shape matching. In: CVPR (2016)
18. Lai, K., Bo, L., Ren, X., Fox, D.: A large-scale hierarchical multi-view RGB-D object dataset. In: ICRA, pp. 1817–1824 (2011)
19. Leibe, B., Schiele, B.: Analyzing appearance and contour based methods for object categorization. In: CVPR, vol. 2, pp. II–409 (2003)
20. Lim, J.J., Pirsivash, H., Torralba, A.: Parsing IKEA objects: fine pose estimation. In: ICCV, pp. 2992–2999 (2013)
21. Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft COCO: common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014, Part V. LNCS, vol. 8693, pp. 740–755. Springer, Heidelberg (2014)
22. Lopez-Sastre, R.J., Redondo-Cabrera, C., Gil-Jimenez, P., Maldonado-Bascon, S.: ICARO: image collection of annotated real-world objects. <http://agamenon.tsc.uah.es/Personales/rlopez/data/icaro> (2010)
23. Ozuyisal, M., Lepetit, V., Fua, P.: Pose estimation for category specific multiview object localization. In: CVPR, pp. 778–785 (2009)
24. Pepik, B., Stark, M., Gehler, P., Schiele, B.: Teaching 3D geometry to deformable part models. In: CVPR, pp. 3362–3369. IEEE (2012)
25. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. In: NIPS, pp. 91–99 (2015)
26. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L.: ImageNet large scale visual recognition challenge. *Int. J. Comput. Vis. IJCV* **115**(3), 1–42 (2015)
27. Russell, B.C., Torralba, A.: Building a database of 3D scenes from user annotations. In: CVPR, pp. 2711–2718 (2009)
28. Savarese, S., Fei-Fei, L.: 3D generic object categorization, localization and pose estimation. In: ICCV, pp. 1–8 (2007)
29. Silberman, N., Hoiem, D., Kohli, P., Fergus, R.: Indoor segmentation and support inference from RGBD images. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012, Part V. LNCS, vol. 7576, pp. 746–760. Springer, Heidelberg (2012)

30. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint (2014). [arXiv:1409.1556](https://arxiv.org/abs/1409.1556)
31. Song, H.O., Xiang, Y., Jegelka, S., Savarese, S.: Deep metric learning via lifted structured feature embedding. In: CVPR (2016)
32. Song, S., Lichtenberg, S.P., Xiao, J.: Sun RGB-D: a RGB-D scene understanding benchmark suite. In: CVPR, pp. 567–576 (2015)
33. Su, H., Huang, Q., Mitra, N.J., Li, Y., Guibas, L.J.: Estimating image depth using shape collections. *ACM Trans. Graph.* **33**(4), 37 (2014)
34. Su, H., Qi, C.R., Li, Y., Guibas, L.J.: Render for CNN: viewpoint estimation in images using CNNs trained with rendered 3D model views. In: ICCV, pp. 2686–2694 (2015)
35. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: CVPR, pp. 1–9 (2015)
36. Thomas, A., Ferrari, V., Leibe, B., Tuytelaars, T., Schiele, B., Gool, L.V.: Towards multi-view object class detection. In: CVPR, pp. 1589–1596 (2006)
37. Uijlings, J.R., van de Sande, K.E., Gevers, T., Smeulders, A.W.: Selective search for object recognition. *Int. J. Comput. Vis. (IJCV)* **104**(2), 154–171 (2013)
38. Weinberger, K.Q., Saul, L.K.: Distance metric learning for large margin nearest neighbor classification. *J. Mach. Learn. Res.* **10**, 207–244 (2009)
39. Xiang, Y., Choi, W., Lin, Y., Savarese, S.: Data-driven 3D voxel patterns for object category recognition. In: CVPR, pp. 1903–1911 (2015)
40. Xiang, Y., Mottaghi, R., Savarese, S.: Beyond pascal: a benchmark for 3D object detection in the wild. In: WACV, pp. 75–82 (2014)
41. Xiao, J., Hays, J., Ehinger, K.A., Oliva, A., Torralba, A.: Sun database: large-scale scene recognition from abbey to zoo. In: CVPR, pp. 3485–3492 (2010)
42. Zia, M.Z., Stark, M., Schindler, K.: Explicit occlusion modeling for 3D object class representations. In: CVPR, pp. 3326–3333 (2013)
43. Zitnick, C.L., Dollár, P.: Edge boxes: locating object proposals from edges. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014, Part V. LNCS, vol. 8693, pp. 391–405. Springer, Heidelberg (2014)