DexYCB: A Benchmark for Capturing Hand Grasping of Objects

Yu-Wei Chao¹  Wei Yang¹  Yu Xiang¹  Pavlo Molchanov¹  Ankur Handa¹  Jonathan Tremblay¹
Yashraj S. Narang¹  Karl Van Wyk¹  Umar Iqbal¹  Stan Birchfield¹  Jan Kautz¹  Dieter Fox¹,²

¹NVIDIA, ²University of Washington
{ychao,weiy,yux,pmolchanov,ahanda,jtremblay,ynarang,kvanwyk,uiqbal,sbirchfield,jkautz,dieterf}@nvidia.com

Figure 1: Two captures (left and right) from the DexYCB dataset. In each case, the top row shows color images simultaneously captured from three views, while the bottom row shows the ground-truth 3D object and hand pose rendered on the darkened captured images.

Abstract

We introduce DexYCB, a new dataset for capturing hand grasping of objects. We first compare DexYCB with a related one through cross-dataset evaluation. We then present a thorough benchmark of state-of-the-art approaches on three relevant tasks: 2D object and keypoint detection, 6D object pose estimation, and 3D hand pose estimation. Finally, we evaluate a new robotics-relevant task: generating safe robot grasps in human-to-robot object handover.¹

1. Introduction

3D object pose estimation and 3D hand pose estimation are two important yet unsolved vision problems. Traditionally, these two problems have been addressed separately, yet in many critical applications, we need both capabilities working together [36, 6, 13]. For example, in robotics, a reliable motion capture for hand manipulation of objects is crucial for both learning from human demonstration [12] and fluent and safe human-robot interaction [46].

State-of-the-art approaches for both 3D object pose [37, 20, 34, 27, 40, 26, 19] and 3D hand pose estimation [49, 23, 18, 2, 9, 14, 31] rely on deep learning and thus require large datasets with labeled hand or object poses for training. Many datasets [48, 49, 16, 17] have been introduced in both domains and have facilitated progress on these two problems in parallel. However, since they were introduced for either task separately, many of them do not contain interaction of hands and objects, i.e., static objects without humans in the scene, or bare hands without interacting with objects. In the presence of interactions, the challenge of solving the two tasks together not only doubles but multiplies, due to the motion of objects and mutual occlusions incurred by the interaction. Networks trained on either of the datasets will thus not generalize well to interaction scenarios.

Creating a dataset with accurate 3D pose of hands and objects is also challenging for the same reasons. As a result, prior works have attempted to capture accurate hand motion either with specialized gloves [10], magnetic sensors [48, 8], or marker-based mocap systems [3, 35]. While they can achieve unparalleled accuracy, the introduction of hand-attached devices may be intrusive and thus bias the naturalness of hand motion. It also changes the appearance of hands and thus may cause issues with generalization.

Due to the challenge of acquiring real 3D poses, there has been an increasing interest in using synthetic datasets to train pose estimation models. The success has been notable on object pose estimation. Using 3D scanned object models and photorealistic rendering, prior work [16, 34, 38, 5, 17] has generated synthetic scenes of objects with high fidelity in appearance. Their models trained only on synthetic data
can thus translate to real images. Nonetheless, synthesizing hand-object interactions remains challenging. One problem is to synthesize realistic grasp poses for generic objects [4]. Furthermore, synthesizing natural looking human motions is still an active research area in graphics.

In this paper, we focus on marker-less data collection of real hand interaction with objects. We take inspiration from recent work [11] and build a multi-camera setup that records interactions synchronously from multiple views. Compared to the recent work, we instrument the setup with more cameras and configure them to capture a larger workspace that allows our human subjects to interact freely with objects. In addition, our pose labeling process utilizes human annotation rather than automatic labeling. We crowdsource the annotation so that we can efficiently scale up the data labeling process. Given the setup, we construct a large-scale dataset that captures a simple yet ubiquitous task: grasping objects from a table. The dataset, DexYCB, consists of 582K RGB-D frames over 1,000 sequences of 10 subjects grasping 20 different objects from 8 views (Fig. 1).

Our contributions are threefold. First, we introduce a new dataset for capturing hand grasping of objects. We empirically demonstrate the strength of our dataset over a related one through cross-dataset evaluation. Second, we provide in-depth analysis of current approaches thoroughly on three relevant tasks: 2D object and keypoint detection, 6D object pose estimation, and 3D hand pose estimation. To the best of our knowledge, our dataset is the first that allows joint evaluation of these three tasks. Finally, we demonstrate the importance of joint hand and object pose estimation on a new robotics relevant task: generating safe robot grasps for human-to-robot object handover.

2. Constructing DexYCB

2.1. Hardware Setup

In order to construct the dataset, we built a multi-camera setup for capturing human hands interacting with objects. A key design choice was to enable a sizable capture space, where a human subject can freely interact and perform tasks with multiple objects. Our multi-camera setup is shown in Fig. 2. We use 8 RGB-D cameras (RealSense D415) and mount them such that collectively they can capture a tabletop workspace with minimal blind spots. The cameras are extrinsically calibrated and temporally synchronized. For data collection, we stream and record all 8 views together at 30 fps with both color and depth of resolution 640 × 480.

2.2. Data Collection and Annotation

Given the setup, we record videos of hands grasping objects. We use 20 objects from the YCB-Video dataset [44], and record multiple trials from 10 subjects. For each trial, we select a target object with 2 to 4 other objects and place them on the table. We ask the subject to start from a relaxed pose, pick up the target object, and hold it in the air. We also ask some subjects to pretend to hand over the object to someone across from them. We record for 3 seconds, which is sufficient to contain the full course of action. For each target object, we repeat the trial 5 times, each time with a random set of accompanied objects and placement. We ask the subject to perform the pick-up with the right hand in the first two trials, and with the left hand in the third and fourth trials. In the fifth trial, we randomize the choice. We rotate the target among all 20 objects. This gives us 100 trials per subject, and 1,000 trials in total for all subjects.

To acquire accurate ground-truth 3D pose for hands and objects, our approach (detailed in Sec. 2.3) relies on 2D keypoint annotations for hands and objects in each view. To ensure accuracy, we label the required keypoints in RGB sequences fully through human annotation. Our annotation tool is based on VATIC [39] for efficient annotation of videos. We set up annotation tasks on the Amazon Mechanical Turk (MTurk) and label every view in all the sequences.

For hands, we adopt 21 pre-defined hand joints as our keypoints (3 joints plus 1 tip for each finger and the wrist). We explicitly ask the annotators to label and track these joints throughout a given video sequence. The annotators are also asked to mark a keypoint as invisible in a given frame when it is occluded.

Pre-defining keypoints exhaustively for every object would be laborious and does not scale as the number of objects increases. Our approach (Sec. 2.3) explicitly addresses this issue by allowing user-defined keypoints. Specifically, given a video sequence in a particular view, we first ask the annotator to find 2 distinctive landmark points that are easily identified and trackable on a designated object, and we ask them to label and track these points throughout the sequence. We explicitly ask the annotators to find keypoints that are visible most of the time, and mark a keypoint as invisible whenever it is occluded.
2.3. Solving 3D Hand and Object Pose

To represent 3D hand pose, we use the popular MANO hand model [28]. The model represents a right or left hand with a deformable triangular mesh of 778 vertices. The mesh is parameterized by two low-dimensional embeddings \((\theta, \beta)\), where \(\theta \in \mathbb{R}^{51}\) accounts for variations in pose (i.e. articulation) and \(\beta \in \mathbb{R}^{10}\) in shape. We use the version from [14], which implements MANO as a differentiable layer in PyTorch that maps \((\theta, \beta)\) to the mesh together with the 3D positions of 21 hand joints defined in the keypoint annotation. We pre-calibrate the hand shape \(\beta\) for each subject and fix it throughout each subject’s sequences.

Since our objects from YCB-Video [44] also come with texture-mapped 3D mesh models, we use the standard 6D object pose representation [16, 17] for 3D object pose. The pose of each object is represented by a matrix \(T \in \mathbb{R}^{4 \times 4}\) composed of a 3D rotation matrix and a 3D translation vector.

To solve for hand and object pose, we formulate an optimization problem similar to [50, 11] by leveraging depth and keypoint annotations from all views and multi-view geometry. For a given sequence with \(N_H\) hands and \(N_O\) objects, we denote the overall pose at a given time frame by \(P = (P_H, P_O)\), where \(P_H = \{\theta_h\}_{h=1}^{N_H}\) and \(P_O = \{T_o\}_{o=1}^{N_O}\). We define the pose in world coordinates where we know the extrinsics of each camera. Then at each time frame, we solve the pose by minimizing the following energy function:

\[
E(P) = E_{\text{depth}}(P) + E_{\text{kpt}}(P) + E_{\text{reg}}(P).
\]

**Depth** The depth term \(E_{\text{depth}}\) measures how well the models given poses explain the observed depth data. Let \(\{d_i \in \mathbb{R}\}_{i=1}^{N_D}\) be the total point cloud merged from all views after transforming to the world coordinates, with \(N_D\) denoting the number of points. Given a pose parameter, we denote the collection of all hand and object meshes as \(M(P) = (\{M_h(\theta_h)\}, \{M_o(T_o)\})\). We define the depth term as

\[
E_{\text{depth}}(P) = \frac{1}{N_D} \sum_{i=1}^{N_D} |\text{SDF}(d_i, M(P))|^2,
\]

where \(\text{SDF}(\cdot)\) calculates the signed distance value of a 3D point from a triangular mesh in mm. While \(E_{\text{depth}}\) is differentiable, calculating \(E_{\text{depth}}\) and also the gradients is computationally expensive for large point clouds and meshes with a huge number of vertices. Therefore, we use an efficient point-parallel GPU implementation for it.

**Keypoint** The keypoint term \(E_{\text{kpt}}\) measures the reprojection error of the keypoints on the models with the annotated keypoints, and can be decomposed by hand and object:

\[
E_{\text{kpt}}(P) = E_{\text{kpt}}(P_H) + E_{\text{kpt}}(P_O).
\]

For hands, let \(J_{h,j}\) be the 3D position of joint \(j\) of hand \(h\) in the world coordinates, \(\hat{p}_{h,j}\) be the annotation of the same joint in the image coordinates of view \(c\), and \(\gamma_{h,j}\) be its visibility indicator. The energy term is defined as

\[
E_{\text{kpt}}(P_H) = \frac{1}{N_C} \sum_{c=1}^{N_C} \sum_{h=1}^{N_H} \sum_{j=1}^{N_J} \gamma_{h,j} ||\text{proj}(J_{h,j}) - \hat{p}_{h,j}||^2,
\]

where \(\text{proj}(\cdot)\) returns the projection of a 3D point onto the image plane of view \(c\), and \(N_C = 8\) and \(N_J = 21\).

For objects, recall that we did not pre-define keypoints for annotation, but rather asked annotators to select distinctive points to track. Here, we assume an accurate initial pose is given at the first frame where an object’s keypoint is labeled visible. We then map the selected keypoint to a vertex on the object’s 3D model by back-projecting the keypoint’s position onto the object’s visible surface. We fix that mapping afterwards. Let \(K_{o,k}\) be the 3D position of the selected keypoint \(k\) of object \(o\) in view \(c\) in world coordinates. Similar to Eq. (4), with \(N_K = 2\), the energy term is

\[
E_{\text{kpt}}(P_O) = \frac{1}{N_K} \sum_{c=1}^{N_C} \sum_{o=1}^{N_O} \sum_{k=1}^{N_K} \gamma_{o,k} ||\text{proj}(K_{o,k}) - \hat{p}_{o,k}||^2,
\]

To ensure an accurate initial pose for keypoint mapping, we initialize the pose in each time frame with the solved pose from the last time frame. We initialize the pose in the first frame by running PoseCNN [44] on each view and select an accurate pose for each object manually.

**Regularization** Following [24, 13], we add an \(\ell_2\) regularization to the low-dimensional pose embedding of MANO to avoid irregular articulation of hands:

\[
E_{\text{reg}}(P) = \frac{1}{N_H} \sum_{h=1}^{N_H} ||\theta_h||^2.
\]

To minimize Eq. (1), we use the Adam optimizer with a learning rate of 0.01. For each time frame, we initialize the pose \(P\) with the solved pose from the last time frame and run the optimizer for 100 iterations.

3. Related Datasets

3.1. 6D Object Pose

Most datasets address instance-level 6D object pose estimation, where 3D models are given a priori. The recent BOP challenge [16, 17] has curated a decent line-up of these datasets which the participants have to evaluate on. Yet objects are mostly static in these datasets without human interactions. A recent dataset [41] was introduced for category-level 6D pose estimation, but the scenes are also static.

3.2. 3D Hand Pose

We present a summary of related 3D hand pose datasets in Tab. 1. Some address pose estimation with depth only,
e.g., BigHand2.2M [48]. These datasets can be large but lack color images and capture only bare hands without interactions. Some others address hand-hand interactions, e.g., Mueller et al. [24] and InterHand2.6M [22]. While not focused on interaction with objects, their datasets address another challenging scenario and is orthogonal to our work. Below we review datasets with hand-object interactions.

**Synthetic** Synthetic data has been increasingly used for hand pose estimation [25, 49, 23, 14]. A common downside is the gap to real images on appearance. To bridge this gap, GANerated Hands [23] was introduced by translating synthetic images to real via GANs. Nonetheless, other challenges remain. One is to synthesize realistic grasp pose for objects. ObMan [14] was introduced to address this challenge using heuristic metrics. Yet besides pose, it is also challenging to synthesize realistic motion. Consequently, these datasets only offer static images but not videos. Our dataset captures real videos with real grasp pose and motion. Furthermore, our motion data from real can help bootstrapping synthetic data generation.

**Marker-based** FPHA [8] captured hand interaction with objects in first person view by attaching magnetic sensors to the hands. This offers a flexible and portable solution, but the attached sensors may hinder natural hand motion and also bias the hand appearance. ContactPose [3] captured grasp poses by tracking objects with mocap markers and recovering hand pose from thermal imagery of the objects surface. While the dataset is large in scale, the thermal based approach can only capturing rigid grasp poses but not motions. GRAB [35] captured full body motion together with hand-object interactions using marker-based mocap. It provides high-fidelity capture of interactions but does not come with any visual modalities. Our dataset is marker-less, captures dynamic grasp motions, and provides RGB-D sequences in multiple views.

**Marker-less** Our DexYCB data falls in this category. The challenge is to acquire accurate 3D pose. Some datasets like Dexter+Object [32] and EgoDexter [23] rely on manual annotation and are thus limited in scale. To acquire 3D pose at scale others rely on automatic or semi-automatic approaches [30, 50]. While these datasets capture hand-object interactions, they do not offer 3D pose for objects.

Most similar to ours is the recent HO-3D dataset [11]. HO-3D also captures hands interacting with objects from multiple views and provides both 3D hand and object pose. Nonetheless, the two datasets differ both quantitatively and qualitatively. Quantitatively (Tab. 1), DexYCB captures multiple views and provides both 3D hand and object pose. Quantitatively (Tab. 1), DexYCB captures interactions with more objects (20 versus 10), from more views (8 versus 1 to 5), and is one order of magnitude larger in terms of number of frames (582K versus 78K) and number of sequences (1,000 versus 27). Qualitatively (Fig. 3), we highlight three differences. First, DexYCB captures full grasping processes (i.e. from hand approaching, opening fingers, contact, to holding the object stably) in short segments, whereas HO-3D captures longer sequences with object in hand most of the time. Among the 27 sequences from HO-3D, we found 17 with hand always rigidly attached to the object, only rotating the object from the wrist with no finger articulation. Second, DexYCB captures a full tabletop workspace with 3D pose of all the objects on the table, whereas in HO-3D one object is held close to the camera each time and only the held object is labeled with pose. Finally, regarding annotation, DexYCB leverages keypoints

<table>
<thead>
<tr>
<th>dataset</th>
<th>visual modality</th>
<th>real image</th>
<th>marker-less</th>
<th>hand-object int</th>
<th>hand-obj int</th>
<th>3D hand obj pose shape</th>
<th>resolution</th>
<th>#frames</th>
<th>#sub</th>
<th>#obj</th>
<th>#views</th>
<th>motion</th>
<th>#seq</th>
<th>dynamic grasp</th>
<th>label</th>
</tr>
</thead>
<tbody>
<tr>
<td>BigHand2.2M [48]</td>
<td>depth</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>640×480</td>
<td>2,200K</td>
<td>10</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>magnetic sensor</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SynthHands [25]</td>
<td>RGB-D</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>640×480</td>
<td>220K</td>
<td>7</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>synthetic</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rendered Hand Pose [49]</td>
<td>RGB-D</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>320×320</td>
<td>44K</td>
<td>20</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>synthetic</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GANerated Hands [23]</td>
<td>RGB</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>256×256</td>
<td>331K</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>synthetic</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ObMan [14]</td>
<td>RGB-D</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>256×256</td>
<td>154K</td>
<td>20</td>
<td>3K</td>
<td>✗</td>
<td>✗</td>
<td>synthetic</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FPDA [8]</td>
<td>RGB-D</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>1920×1080</td>
<td>105K</td>
<td>6</td>
<td>4</td>
<td>✓</td>
<td>1,175</td>
<td>✓</td>
<td>magnetic sensor</td>
<td></td>
</tr>
<tr>
<td>ContactPose [3]</td>
<td>RGB-D</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>960×540</td>
<td>2,991K</td>
<td>50</td>
<td>25</td>
<td>✓</td>
<td>2,303</td>
<td>✗</td>
<td>synthetic</td>
<td></td>
</tr>
<tr>
<td>GRAB [35]</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>1,624K</td>
<td>10</td>
<td>51</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td>1,335</td>
<td>✓</td>
<td>automatic</td>
</tr>
<tr>
<td>Mueller et al. [24]</td>
<td>RGB</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>640×480</td>
<td>80K</td>
<td>5</td>
<td>4</td>
<td>✓</td>
<td>11</td>
<td>–</td>
<td>synthetic</td>
<td></td>
</tr>
<tr>
<td>InterHand2.6M [22]</td>
<td>RGB</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>512×334</td>
<td>2,590K</td>
<td>27</td>
<td>&gt;80</td>
<td>✗</td>
<td>✗</td>
<td>–</td>
<td>automatic</td>
<td></td>
</tr>
<tr>
<td>Dexter+Object [32]</td>
<td>RGB-D</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>640×480</td>
<td>3K</td>
<td>2</td>
<td>2</td>
<td>✓</td>
<td>6</td>
<td>✓</td>
<td>automatic</td>
<td></td>
</tr>
<tr>
<td>EgoDexter [25]</td>
<td>RGB-D</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>640×480</td>
<td>3K</td>
<td>4</td>
<td>–</td>
<td>1</td>
<td>✓</td>
<td>4</td>
<td>manual</td>
<td></td>
</tr>
<tr>
<td>FreiHAND [50]</td>
<td>RGB</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>224×224</td>
<td>37K</td>
<td>32</td>
<td>27</td>
<td>✓</td>
<td>–</td>
<td>–</td>
<td>automatic</td>
<td></td>
</tr>
<tr>
<td>DexYCB (ours)</td>
<td>RGB-D</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>640×480</td>
<td>582K</td>
<td>10</td>
<td>20</td>
<td>8</td>
<td>✓</td>
<td>1,000</td>
<td>✓</td>
<td>manual</td>
</tr>
</tbody>
</table>

Table 1: Comparison of DexYCB with existing 3D hand pose datasets.
Figure 3: Qualitative comparison of HO-3D [11] (top) and DexYCB (bottom). DexYCB captures full grasping processes in a tabletop workspace with 3D pose of all the on table objects.

fully labeled by humans through crowdsourcing, while HO-3D relies on a fully automatic labeling pipeline.

4. Cross-Dataset Evaluation

To further assess the merit of our DexYCB dataset, we perform cross-dataset evaluation following prior work [50, 29, 47]. We focus specifically on HO-3D [11] due to its relevance, and evaluate generalizability between HO-3D and DexYCB on single-image 3D hand pose estimation. For DexYCB, we generate a train/val/test split following our benchmark setup (“S0” in Sec. 5.1). For HO-3D, we select 6 out of its 55 training sequences for test. We use Spurr et al. [31] (winner of the HAND 2019 Challenge) as our method, and train the model separately on three training sets: the training set of HO-3D, the training set of DexYCB, and the combined training set from both datasets. Finally, we evaluate the three trained models separately on the test set of HO-3D and DexYCB.

Tab. 2 shows the results in mean per joint position error (MPJPE) reported on two different alignment methods (details in Sec. 5.4). We experiment with two different backbones: ResNet50 and HRNet32 [33]. Unsurprisingly, when training on a single dataset, the model generalizes better to the respective test set. The error increases when tested on the other dataset. When we evaluate the DexYCB trained model on HO-3D, we observe a consistent increase from 1.4× to 1.9× (e.g., for ResNet50, from 18.05 to 31.76 mm on root-relative). However, when we evaluate the combined training set and evaluate on HO-3D, we can further reduce the error from only training on HO-3D (e.g., from 18.05 to 15.79 mm). When tested on DexYCB, the error rather rises compared to training only on DexYCB (e.g., from 12.97 to 13.36 mm). We conclude that DexYCB complements HO-3D better than vice versa.

5. Benchmarking Representative Approaches

We benchmark three tasks on our DexYCB dataset: 2D object and keypoint detection, 6D object pose estimation, and 3D hand pose estimation. For each task we select representative approaches and analyze their performance.

5.1. Evaluation Setup

To evaluate different scenarios, we generate train/val/test splits in four different ways (referred to as “setup”):

- **S0 (default).** The train split contains all 10 subjects, all 8 camera views, and all 20 grasped objects. Only the sequences are not shared with the val/test split.
- **S1 (unseen subjects).** The dataset is split by subjects (train/val/test: 7/1/2).
- **S2 (unseen views).** The dataset is split by camera views (train/val/test: 6/1/1).
- **S3 (unseen grasping).** The dataset is split by grasped objects (train/val/test: 15/2/3). Objects being grasped in the test split are never being grasped in the train/val split, but may appear static on the table. This way the training set still contain examples of each object.

5.2. 2D Object and Keypoint Detection

We evaluate object and keypoint detection using the COCO evaluation protocol [21]. For object detection, we consider 20 object classes and 1 hand class. We collect ground truths by rendering a segmentation mask for each instance in each camera view. For keypoints, we consider 21 hand joints and collect the ground truths by reprojecting each 3D joint to each camera image.

We benchmark two representative approaches: Mask R-CNN (Detectron2) [15, 43] and SOLOv2 [42]. Mask R-CNN has been the de facto for object and keypoint detection and SOLOv2 is a state-of-the-art on COCO instance segmentation. For both we use a ResNet50-FPN backbone pre-trained on ImageNet and finetune on DexYCB.

Tab. 3 shows results for object detection in average precision (AP). First, Mask R-CNN and SOLOv2 perform similarly in mean average precision (mAP). Mask R-CNN has a slight edge on bounding box (e.g., 75.76 versus 75.13 mAP on S0) while SOLOv2 has a slight edge on segmentation (e.g., 71.56 versus 69.58 mAP on S0). This is because
Mask R-CNN predicts bounding boxes first and uses them to generate segmentations. Therefore the error can accumulate for segmentation. SOLOv2 in contrast directly predicts segmentations and uses it to generate bounding boxes. Second, the AP for hand is lower than the AP for objects (e.g., for Mask R-CNN, 71.85 for hand versus 75.76 mAP on S0 bbox). This suggests that hands are more difficult to detect than objects, possibly due to its larger variability in position. Finally, performance varies across setups. For example, mAP in S1 is lower than in S0 (e.g., for Mask R-CNN, 72.69 versus 75.76 on bbox). This can be attributed to the challenge introduced by unseen subjects. Tab. 4 shows the AP for keypoint detection with Mask R-CNN. We observe a similar trend on the ordering of mAP over different setups.

5.3. 6D Object Pose Estimation

We evaluate single-view 6D pose estimation following the BOP challenge protocol [16, 17]. The task asks for a 6D pose estimate for each object instance given the number of objects and instances in each image. The evaluation computes recall using three different pose-error functions (VSD, MSSD, MSPD), and the final score is the average of the three recall values (AR).

We first analyze the challenge brought by hand grasping by comparing static and grasped objects. We use PoseCNN (RGB) [44] as a baseline and retrain the model on DexYCB. Tab. 5 shows the AR on S0 evaluated separately on the full test set, static objects only, and grasped objects only. Without surprise the AR drops significantly when only considering the grasped objects. This recapitulates the increasing challenge of object pose estimation under hand interactions. To better focus on this regime, we evaluate only on the grasped objects in the remaining experiments.

We next evaluate on the four setups using the same baseline. Results are shown in Tab. 6 (left). On S1 (unseen subjects), we observe a drop in AR compared to S0 (e.g., 38.26 versus 41.65 on all) as in object detection. This can be attributed to the influence of interaction from unseen hands as well as different grasping styles. The column of S3 (unseen grasping) shows the AR of the three only objects being grasped in the test set (i.e., “009_gelatin_box”, “021_bleach_cleanser”, “036_wood_box”). Again, the AR is lower compared to on S0, and the drop is especially significant for smaller objects like “009_gelatin_box” (33.07 versus 46.62). Surprisingly, AR on S2 (unseen views) does not drop but rather increases slightly (e.g., 45.18 versus 41.65 on all). This suggests that our multi-camera setup has a dense enough coverage of the scene such that training on certain views can translate well to some others.

Finally, we benchmark five representative approaches: PoseCNN [44], DeepIM [20], DOPE [38], PoseBFP [5].
Table 6: 6D object pose estimation results in AR (%). Left: PoseCNN (RGB). Right: performance of representative approaches on S1.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>RGB + depth</td>
<td>RGB-D</td>
<td>RGB</td>
<td>RGB-D</td>
<td>RGB</td>
</tr>
<tr>
<td></td>
<td>S0</td>
<td>S1</td>
<td>S2</td>
<td>S3</td>
<td>S0</td>
</tr>
<tr>
<td>002_master_chef_can</td>
<td>44.53</td>
<td>60.74</td>
<td>68.40</td>
<td>19.82</td>
<td>58.04</td>
</tr>
<tr>
<td>003_cracker_box</td>
<td>57.46</td>
<td>79.43</td>
<td>84.56</td>
<td>53.50</td>
<td>67.37</td>
</tr>
<tr>
<td>004_sugar_box</td>
<td>42.45</td>
<td>51.29</td>
<td>59.34</td>
<td>32.06</td>
<td>56.55</td>
</tr>
<tr>
<td>005_tomato_soup_can</td>
<td>42.99</td>
<td>46.87</td>
<td>55.46</td>
<td>20.52</td>
<td>40.03</td>
</tr>
<tr>
<td>006_mustard_bottle</td>
<td>42.89</td>
<td>55.52</td>
<td>63.31</td>
<td>23.66</td>
<td>54.93</td>
</tr>
<tr>
<td>007_tuna_fish_can</td>
<td>39.58</td>
<td>39.18</td>
<td>46.98</td>
<td>5.86</td>
<td>36.77</td>
</tr>
<tr>
<td>008_pudding_box</td>
<td>42.60</td>
<td>52.26</td>
<td>67.18</td>
<td>14.48</td>
<td>27.24</td>
</tr>
<tr>
<td>009_gelatine_box</td>
<td>41.71</td>
<td>47.89</td>
<td>55.54</td>
<td>21.81</td>
<td>31.27</td>
</tr>
<tr>
<td>010_potted_meat_can</td>
<td>51.64</td>
<td>60.39</td>
<td>69.40</td>
<td>14.15</td>
<td>46.43</td>
</tr>
<tr>
<td>011_banana</td>
<td>42.91</td>
<td>40.21</td>
<td>48.63</td>
<td>16.15</td>
<td>39.15</td>
</tr>
<tr>
<td>019_pitcher_base</td>
<td>49.62</td>
<td>54.27</td>
<td>63.46</td>
<td>7.02</td>
<td>51.38</td>
</tr>
<tr>
<td>021_bleach_cleanser</td>
<td>42.56</td>
<td>51.46</td>
<td>64.28</td>
<td>16.70</td>
<td>53.50</td>
</tr>
<tr>
<td>024_bowl</td>
<td>57.30</td>
<td>57.05</td>
<td>60.30</td>
<td>69.40</td>
<td>74.74</td>
</tr>
<tr>
<td>025_mug</td>
<td>37.28</td>
<td>41.15</td>
<td>49.51</td>
<td>18.21</td>
<td>34.87</td>
</tr>
<tr>
<td>035_power_drill</td>
<td>41.86</td>
<td>58.10</td>
<td>64.28</td>
<td>19.60</td>
<td>62.49</td>
</tr>
<tr>
<td>036_woodblock</td>
<td>44.08</td>
<td>49.91</td>
<td>65.39</td>
<td>23.80</td>
<td>51.63</td>
</tr>
<tr>
<td>037_scissors</td>
<td>22.87</td>
<td>27.48</td>
<td>32.65</td>
<td>14.16</td>
<td>29.40</td>
</tr>
<tr>
<td>040_large_marker</td>
<td>32.12</td>
<td>33.27</td>
<td>46.63</td>
<td>-</td>
<td>29.66</td>
</tr>
<tr>
<td>052_extra_large_clamp</td>
<td>31.89</td>
<td>39.63</td>
<td>47.04</td>
<td>-</td>
<td>35.99</td>
</tr>
<tr>
<td>061_foam_brick</td>
<td>14.03</td>
<td>18.14</td>
<td>27.57</td>
<td>-</td>
<td>30.12</td>
</tr>
<tr>
<td>all</td>
<td>57.54</td>
<td>48.99</td>
<td>57.54</td>
<td>-</td>
<td>57.43</td>
</tr>
</tbody>
</table>

Table 7: 3D hand pose estimation results in MPJPE (mm). Numbers in parentheses are AUC values.

<table>
<thead>
<tr>
<th>Method</th>
<th>ABSOLUTE</th>
<th>ROOT-RELATIVE</th>
<th>PROCRUSTES</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>S0</td>
<td>S1</td>
<td>S2</td>
</tr>
<tr>
<td>[31] + ResNet50</td>
<td>53.92</td>
<td>17.71</td>
<td>7.12</td>
</tr>
<tr>
<td>+ HRNet32</td>
<td>52.26</td>
<td>17.34</td>
<td>6.83</td>
</tr>
<tr>
<td>+ A2J [45]</td>
<td>27.53</td>
<td>23.93</td>
<td>12.07</td>
</tr>
<tr>
<td>+ ResNet50</td>
<td>70.23</td>
<td>22.71</td>
<td>8.43</td>
</tr>
<tr>
<td>+ HRNet32</td>
<td>70.10</td>
<td>22.66</td>
<td>7.98</td>
</tr>
<tr>
<td>+ A2J [45]</td>
<td>29.09</td>
<td>25.57</td>
<td>12.95</td>
</tr>
<tr>
<td>+ ResNet50</td>
<td>83.46</td>
<td>23.15</td>
<td>8.11</td>
</tr>
<tr>
<td>+ HRNet32</td>
<td>80.63</td>
<td>25.49</td>
<td>8.21</td>
</tr>
<tr>
<td>+ A2J [45]</td>
<td>23.44</td>
<td>27.65</td>
<td>13.42</td>
</tr>
<tr>
<td>+ ResNet50</td>
<td>58.69</td>
<td>19.41</td>
<td>7.56</td>
</tr>
<tr>
<td>+ HRNet32</td>
<td>55.39</td>
<td>18.44</td>
<td>7.06</td>
</tr>
<tr>
<td>+ A2J [45]</td>
<td>30.99</td>
<td>24.92</td>
<td>12.15</td>
</tr>
</tbody>
</table>

5.4. 3D Hand Pose Estimation

The task is to estimate the 3D position of 21 hand joints from a single image. We evaluate with two metrics: mean per joint position error (MPJPE) in mm and percentage of correct keypoints (PCK). Besides computing error in absolute 3D position, we also report errors after aligning the predictions with ground truths in post-processing [49, 50, 11]. We consider two alignment methods: root-relative and Procrustes. The former removes ambiguity in translation by replacing the root (wrist) location with ground truths. The latter removes ambiguity in translation, rotation, and scale, thus focusing specifically on articulation. For PCK we re-latter removes ambiguity in translation, rotation, and scale, replacing the root (wrist) location with ground truths. The

We consider two alignment methods: root-relative and Pro-crustes. We benchmark one RGB and one depth based approach. For RGB, we select a supervised version of Spurr et al. [31] which won the HANDS 2019 Challenge [1]. We experiment with the original ResNet50 backbone as well as a stronger HRNet32 [33] backbone, both initialized with ImageNet pre-trained models and finetuned on DexYCB. For depth, we select A2J [45] and retrain the model on DexYCB. Tab. 7 shows the results. For [31], we can see that estimating absolute 3D position solely from RGB is difficult (e.g., 53.92 mm absolute MPJPE with ResNet50 on S0). The stronger HRNet32 backbone reduces the errors over ResNet50 but only marginally (e.g., 52.26 versus 53.92 mm absolute MPJPE on S0). Similar to in object pose, the errors on S1 (unseen subjects) increase from S0 (e.g., 70.10 versus 70.22 mm absolute MPJPE with ResNet50 on S0). Finally, CosyPose achieves the best overall AR among all the RGB-based methods (57.43 AR on all).
crease pronouncedly and even surpass the other setups (e.g., 80.63 mm absolute MPJPE for HRNet32). This is because unlike objects, the subjects are always facing the same direction from a situated position in this setup. This further constrains the possible hand poses observed in each view, making cross view generalization more challenging. Unsurprisingly, A2J [45] outperforms [31] on absolute MPJPE due to the depth input, but falls short on estimating articulation as shown by the errors after alignment (e.g., 12.07 vs. 6.83 mm Procrustes MPJPE for HRNet32 on S0).

6. Safe Human-to-Robot Object Handover

Task  Given an RGB-D image with a person holding an object, the goal is to generate a diverse set of robot grasps to take over the object without pinching the person’s hand (we refer to as “safe” handovers). The diversity of grasps is important since not all the grasps are kinematically feasible for execution. We assume a parallel-jaw Franka Panda gripper and represent each grasp as a point in $SE(3)$.

Evaluation  We first sample 100 grasps for each YCB object using farthest point sampling from a diverse set of grasps pre-generated for that object in [7]. This ensures a dense coverage of the pose space (Fig. 5). For each image, we transform these grasps from the object frame to camera frame using the ground-truth object pose, and remove those collided with the ground-truth object and hand mesh. This generates a reference set of successful grasps $\mathcal{R}$.

Given a set of predicted grasps $\chi$, we evaluate its diversity by computing the coverage [7] of $\mathcal{R}$, defined by the percentage of grasps in $\mathcal{R}$ having at least one matched grasp in $\chi$ that is neither collided with the object nor the hand. Specifically, two grasps $g, h$ are considered matched if $|g_t - h_t| < \sigma_t$ and $\arccos(|(g_q, h_q)|) < \sigma_q$, where $g_t$ is the translation and $g_q$ is the orientation in quaternion. We use $\sigma_t = 0.05$ m and $\sigma_q = 15^\circ$.

One could potentially hit a high coverage by sampling grasps exhaustively. Therefore we also compute precision, defined as the percentage of grasps in $\chi$ that have at least one matched successful grasp in $\mathcal{R}$.

Baseline  We experiment with a simple baseline that only requires hand segmentation and 6D object pose. Similar to constructing $\mathcal{R}$, we transform the 100 grasps to the camera frame but using the estimated object pose, then remove those that are collided with the hand point cloud obtained by the hand segmentation and the depth image. Specifically, a grasp is collided if the distance of a pair of points from the gripper point cloud and the hand point cloud is less than a threshold $\epsilon$. The gripper point cloud is obtained from a set of pre-sampled points on the gripper surface. We use the hand segmentation results from Mask R-CNN (Sec. 5.2).

Results  We evaluate grasps generated with different object pose methods at different thresholds $\epsilon \in [0, 0.07 \text{ m}]$ and show the precision-coverage curves on S1 in Fig. 4. We see that better object pose estimation leads to better grasp generation. Fig. 5 shows qualitative examples of the predicted grasps. We see that most of the failure grasps (red and gray) are due to inaccurate object pose. Some are hand-colliding grasps caused by a miss detected hand when the hand is partially occluded by the object (e.g., “003_cracker_box”). This can be potentially addressed by model based approaches that directly predict the full hand shape.

7. Conclusions

We have introduced DexYCB for capturing hand grasping of objects. We have shown its merits, presented a thorough benchmark of current approaches on three relevant tasks, and evaluated a new robotics-relevant task. We envision our dataset will drive progress on these crucial fronts.
References


[9] Liucao Ge, Zhou Ren, Yuncheng Li, Zehao Xue, Yingying Wang, Jianfei Cai, and Junsong Yuan. 3D hand shape and pose estimation from a single RGB image. In CVPR, 2019. 1


