

Objectron: A Large Scale Dataset of Object-Centric Videos in the Wild with Pose Annotations

Adel Ahmadyan Liangkai Zhang Artsiom Ablavatski Jianing Wei Matthias Grundmann
Google Research

{ahmadyan, liangkai, artsiom, jianingwei, grundman}@google.com



Figure 1: Example videos in our dataset. Each sample is a video annotated with the object’s 3D bounding box.

Abstract

3D object detection has recently become popular due to many applications in robotics, augmented reality, autonomy, and image retrieval. We introduce the Objectron dataset to advance the state of the art in 3D object detection and foster new research and applications, such as 3D object tracking, view synthesis, and improved 3D shape representation. The dataset contains object-centric short videos with pose annotations for nine categories and includes 4 million annotated images in 14,819 annotated videos. We also propose a new evaluation metric, 3D Intersection over Union, for 3D object detection. We demonstrate the usefulness of our dataset in 3D object detection and novel view synthesis tasks by providing baseline models trained on this dataset. Our dataset and evaluation source code are available online at [Github.com/google-research-datasets/Objectron](https://github.com/google-research-datasets/Objectron).

1. Introduction

The state of the art in machine learning has achieved exceptional accuracy on many computer vision tasks solely by training models on images. Building upon these successes and advancing 3D object understanding has great potential to power a wider range of applications, such as augmented reality, robotics, autonomy, and image retrieval. Yet, understanding objects in 3D remains a challenging task due to the lack of large real-world datasets compared to 2D tasks (e.g., ImageNet [8], COCO [22], and Open Images [20]). To empower the research community for continued advancement in 3D object understanding, there is a strong need for the release of object-centric video datasets, which capture more of the 3D structure of an object, while matching the data format used for many down-stream vision tasks (i.e., video or camera streams), to aid in the training and benchmarking of machine learning models. The object-centric approach is consistent with how our brains perceive new objects too. For

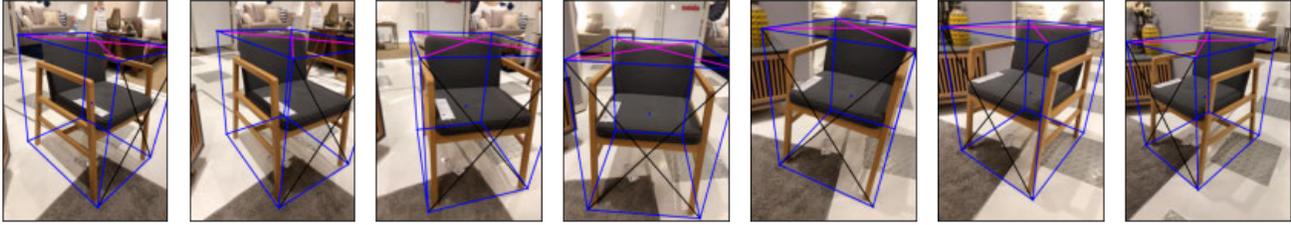


Figure 2: Our dataset consists of object-centric videos, which capture different views of the same objects from different angles.

example, when a child wants to learn the shape of a chair, they'll walk around and look at the chair from different angles to pick up information. In other words, "We must also move in order to perceive" [12][37].

We present the Objectron dataset, a collection of short, object-centric video clips capturing a larger set of common objects from different angles. Each video clip is accompanied by augmented reality (AR) session metadata that includes camera poses, sparse point-clouds, and surface planes. The data also contains manually annotated 3D bounding boxes for each object, which describe the object's position, orientation, and dimensions. The dataset consists of 14, 819 annotated video clips and 4M annotated images collected from a geo-diverse sample (covering ten countries across five continents). Figure 2 shows an example video in our dataset.

There are several advantages of using Objectron dataset over existing works:

- Videos contain multiple views of the same object, enabling many applications well beyond 3D object detection. This includes multi-view geometric understanding, view synthesis, 3D shape reconstruction, etc.
- The 3D bounding box is present in the entire video and is temporally consistent, thus enabling 3D tracking applications.
- Our dataset is collected in the wild to provide better generalization for real-world scenarios in contrast to datasets that are collected in a controlled environment [13][4].
- Each instance's translation and size are stored in metric scale, thanks to accurate on-device AR tracking and provides sparse point clouds in 3D, enabling sparse depth estimation techniques. The images are calibrated and the camera parameters are provided, enabling the recovery of the object's true scale.
- Our annotations are dense and continuous, unlike some of the previous work [30] where viewpoints have been discretized to fit into bins.

- Each object category contains hundreds of instances, collected from different locations across different countries in different lighting conditions.

We have conducted experiments for 3D object detection and novel view synthesis tasks using our dataset. We proposed a novel method to compute the precise 3D IoU of oriented 3D bounding boxes. These experiments can be used as baselines for future research.

2. Previous Work

In this section, we review several commonly used datasets for 3D object detection and compare them with Objectron.

BOP challenge [14] consists of a set of benchmark for 3D object detection and combines many of these smaller datasets into a larger one. Most of the images in the dataset are taken in a very controlled environment and feature clutter and heavy occlusion that appear in industrialized setups. T-LESS [15] features industrialized objects that lack texture or color. Rutgers APC [27] contains 14 textured objects from the Amazon picking challenge. LineMOD [13] is the most commonly used dataset for object pose estimation. Similar to LineMOD, IC-BIN dataset [9] adds a few more categories. YCB [4] dataset contains videos of objects and their poses in a controlled environment. In comparison to these datasets, Objectron has a larger scale and contains high-resolution videos of common objects in the wild.

ObjectNet3D [38] is dataset that contains 3D object poses from images. Similarly, Pascal3D+ [39] adds 3D pose annotations to the Pascal VOC and a few images from the ImageNet dataset. In comparison to Objectron, these datasets contain more categories and instances but they only include images. Furthermore, the objects are annotated with a 2D-to-3D alignment process, so their pose is annotated up to 6-DoF (instead of nine) and camera intrinsics are not available. So it is not possible to recover the object's scale from these datasets. Pix3D [32] contains pixel-level 2D-3D pose alignment. Similarly, 3DObject[30] provides discretized viewpoint annotations for 10 everyday object categories.

Another type of dataset used for 3D object detection tasks is scene datasets. In these datasets, a video of a scene is recorded with an RGBD camera or LIDAR. ScanNet [6] is

a large scale video dataset of indoor scenes with semantic annotations. The dataset contains over 1500 scenes recorded in RGBD videos. The dataset does not contain 3D pose information, however, Scan2CAD [3] annotates the original scans with ShapeNetCore [5] models to label each object's pose. Rio [35] is another dataset that contains indoor scans annotated with an object's 3D pose. In comparison Objectron videos are object-centric, and we have an order of magnitude more samples in our dataset.

Another approach is to use synthetic data. ShapeNet [5] includes CAD models for many objects and has been widely used [34][31]. However, the sample's visual quality is limited and it does not generalize well to real-world applications. Synthetic datasets offer valuable data for training and benchmarking, but the ability to generalize to the real-world is unknown. HyperSim[28] generates photo-realistic scenes with object pose annotation, but reproducibility is challenging since they have not released any of the dataset's assets.

3. Data Collection and Annotation

3.1. Object Categories

In Objectron dataset, the aim was to select meaningful categories of common objects that form a representative set of all categories that are practically relevant and technically challenging. Our goal was to capture these objects in their common environment, and in relative context whether it would be in store, indoor or outdoor environment. We also included objects of various sizes, ranging from a few centimeters (e.g. cups) to as large as chairs and bikes.

The object categories in the dataset contain both rigid, and non-rigid objects. We included non-rigid categories such as bikes and laptops specifically since we expect techniques using CAD models or strong priors will face challenges estimating the pose of these object categories. We should mention non-rigid objects remain stationary during the period of each video.

Many 3D object detection models are known to exhibit difficulties in estimating rotations of symmetric objects [21]. Symmetric objects have ambiguity in their one, two, or even three degrees of rotation. Therefore we added categories like "cups" and "bottles" specifically to test it.

It has been shown that vision models pay special attention to texts in the images [11]. Re-producing texts and labels correctly are important in generative models too. Therefore we added categories of objects with very distinct texts in their labels such as "books" and "cereal boxes". Since our dataset is collected from a geo-diverse set of countries, multiple different languages are present in the videos as well. We should report these categories, despite having relatively simple box-shaped geometry, have very different texture patterns. So our baseline experiments have difficulty estimating their poses accurately.



Figure 3: Countries where we collected data from.

Since we strive for real-time perception we included a few categories (shoes and chairs) that enable exciting applications, such as augmented reality and image retrieval.

3.2. Data Collection

Our data collection consists of the video recording when the camera moves around the object and looks at it from different angles. We also capture camera poses, point-clouds, and surface planes through an AR session (e.g. ARKit [2] or ARCore [1]). AR solutions track a set of features through the video and estimate their 3D coordinates on-device in real-time. Meanwhile, they estimate camera poses using bundle-adjustment and filtering. We emphasize all the translation and scale reported in our dataset are on the metric scale. We released both of these files (the video recording and the AR session metadata) in the dataset. We assume the standard pinhole camera model, and provide calibration, extrinsics and intrinsics matrix for every frame in the dataset.

All the videos are recorded in 1920×1080 resolution at 30fps using the back-camera of high-end phones. We only enabled data collection on a very few phone models (<5) to make sure the output quality remains consistent across our dataset. To avoid drift in the AR session, we kept the video length short at around 10sec . The collectors are instructed to avoid rapid movements to avoid blurry images. Finally, the object remains stationary in all of our videos.

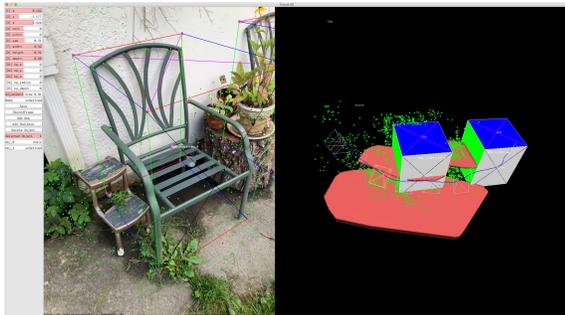
By using mobile phones for data collection we were able to quickly launch a data collection campaign across many different countries. The dataset is geo-diverse and covers 10 countries over five continents. Figure 3 shows which countries are represented in the dataset. The samples are distributed uniformly in each region.

3.3. Data Annotation

Efficient and accurate data annotation is the key to building large-scale datasets. Annotating 3D bounding boxes for each image is time-consuming and expensive. Instead, we annotate 3D objects in a video clip and populate them to all frames in the clip, scaling up the annotation process, and

reducing the per image annotation cost. The user interfaces for the annotation tool is shown in Figure 4a. The input to our annotation tool is a video sequence of stationary objects. It covers different viewing angles of objects.

Next, we show the 3D world map to the annotator side-by-side with the images from the video sequence (Figure 4a). The annotator draws a 3D bounding box in the 3D world map, and our tool projects the 3D bounding box over all the frames given pre-computed camera poses from the AR sessions (such as ARKit or ARCore). The annotator looks at the projected bounding box and makes necessary adjustments (position, orientation, and the scale of a 3D bounding box) so the projected bounding box looks consistent across different frames. At the end of the process, the user saves the 3D bounding box and the annotation.



(a) Our annotation tool



Figure 4: Data annotation. The annotated 3D box is verified at multiple views, and then populated to all images in the sequence.

The benefits of our approach are 1) by annotating a video once, we get annotated images for all frames in the video sequence; 2) by using AR, we can get accurate metric sizes for bounding boxes.

3.4. Annotation Variance

The accuracy of our annotation hinges on two factors: 1) the amount of drift in the estimated camera pose throughout the captured video, and 2) the accuracy of the raters annotating the 3D bounding box. We compared the relative positional drift in our camera pose against an offline refined camera pose (obtained by an offline bundle adjustment) and

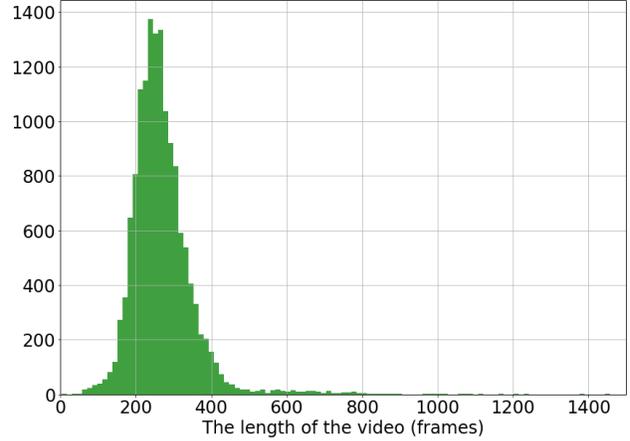


Figure 5: Distribution of the video length in our dataset. Majority of the videos are 10 seconds long (300 frames) and the longest video is 2022 frames long.

observed an average of 0.69% drift in length. To further reduce the drift, we usually capture shorter sequences below 10sec and further instructed our annotators to reject any sequences that with drift and jitter artifacts. Figure 5 shows the distribution of the length of the videos in our dataset. To evaluate the accuracy of the rater, we asked eight annotators to re-annotate same sequences. A few samples are shown in Figure 6. Overall for the chairs, the standard deviation for the chair orientation, translation, and scale was 4.6° , $1cm$, and $4cm$, respectively which demonstrates insignificant variance of the annotation results between different raters.

4. Objectron Dataset

In this section, we describe the details of our Objectron dataset and provide some statistics. The dataset contains videos in 9 categories: bikes, books, bottles, cameras, cereal boxes, chairs, cups, laptops, and shoes. Some of these objects are non-rigid (e.g. bikes or laptops) and all of them remain stationary during the video recording. In each video, the camera moves around the object, capturing it from different angles. In total there are 17,095 object instances that appear in $4M$ annotated images from 14819 annotated videos (not counting the unreleased evaluation set for future competitions). The dataset is collected from ten different countries. This is important for categories that contain texts and labels, such as books, cereal boxes and bottles. So our samples contain different languages as well as different local environments. Each category is divided to train and test sets. Table 1 shows detailed per-category statistics of our dataset.

Each sample contains the high-resolution image, along with the camera pose, point-cloud (from tracking), and planar surfaces in the environment. The data also contains manually annotated 3D bounding boxes for each object, which



Figure 6: The overlay of the 3D bounding boxes annotated by different annotators shows the annotations from different raters are very close.

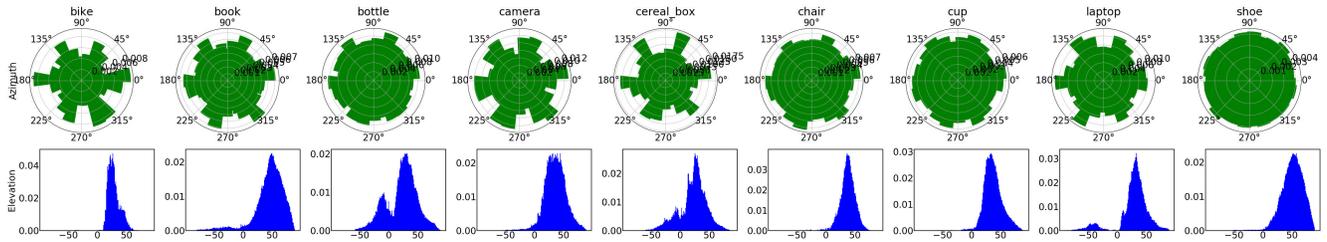


Figure 7: View-point distribution of samples per object category. The top row shows the azimuth distribution in polar graph, and the bottom row denotes the elevation distribution.

describe the object’s orientation, translation, and size relative to the camera pose. Our coordinate system follows a left-hand rule, where $+y$ axis is up. From the pose, we also compute the object’s bounding box 3D keypoints, projected 2D keypoints, as well as azimuth and elevation. Furthermore, each frame in the video contains the camera pose, and the camera’s projection and view matrices.

The annotation of the object contains its rotation, translation w.r.t. the camera center as well as the object’s scale. To get a better understanding of the viewpoint distribution, we computed the azimuth of each object instance w.r.t the camera center. Here azimuth 0 degree indicates the object is being viewed from the front. Figure 7-top shows the azimuth distribution for different object categories in our dataset. For some categories, there is a specific bias toward the front and top views.

We optimized the camera poses using a standard bundle adjustment algorithm from a Structure-from-Motion (SfM) pipeline. The bundle adjustment algorithm minimizes the reprojection error between different projections of keypoints across multiple frames to produces an up-to-scale pose trajec-

tory. We used the scale factor of the original sequence to re-scale the trajectory and bring it to metric scale. Then we computed the mean absolute positional error between the optimized and the original trajectory $\sum \|p_i - q_i\|_2 / \sum \Delta p_i \times n$ where p_i is the position of the original camera pose, q_i is the optimized camera pose, $\sum \Delta p_i$ is the length of the trajectory and n is the number of frames in the sequence. On average, we observed 0.69% mean absolute positional error between the original and optimized camera poses.

5. Baseline Experiments and Evaluations

Our dataset is released with an evaluation code to assess the performance of 3D object detection algorithms using various metrics. The evaluation code computes the average precision for multiple 2D and 3D metrics such as 3D IoU, 2D projection error, view-point error, polar and azimuth error, and rotation error.

Except for the 3D IoU, the other metrics are fairly standard and we refer the readers to our code as well as other references for further details. In this section, we explain our

class	bike	book	bottle	camera	cereal_box	chair	cup	laptop	shoe
No. videos	476	2024	1928	815	1609	1943	2204	1473	2116
No. frames	150k	576k	476k	233k	396k	488k	546k	485k	557k
No. instances	487	2084	2014	931	1638	2362	2378	1710	3500
Avg instance per video	1.03	1.02	1.04	1.14	1.01	1.21	1.07	1.16	1.65

Table 1: Per-category statistics of Objectron dataset.

Model	bike	book	bottle	camera	cereal_box	chair	cup	laptop	shoe	average
MobilePose [16]	0.3109	0.1797	0.5433	0.4483	0.5419	0.6847	0.3665	0.5225	0.4171	0.4461
Two-stage	0.6127	0.5218	0.5744	0.8016	0.6272	0.8505	0.5388	0.6735	0.6606	0.6512

(a) Average precision at 0.5 3D IoU metric for different categories (Higher is better).

Model	bike	book	bottle	camera	cereal_box	chair	cup	laptop	shoe	average
MobilePose [16]	0.1581	0.0840	0.0818	0.0773	0.0454	0.0892	0.2263	0.0736	0.0655	0.1001
Two-stage	0.0828	0.0477	0.0405	0.0449	0.0337	0.0488	0.0541	0.0291	0.0391	0.0467

(b) Mean pixel error (MPE) of 2D projection of box vertices for different categories (Lower is better).

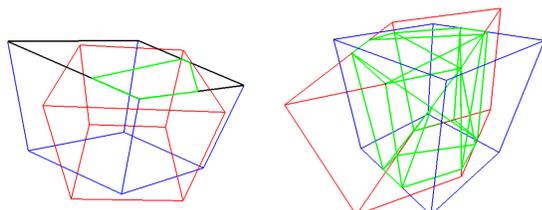
Model	bike	book	bottle	camera	cereal_box	chair	cup	laptop	shoe	average
MobilePose [16]	0.4376	0.4111	0.4413	0.5293	0.8780	0.6195	0.0893	0.6052	0.3934	0.4894
Two-stage	0.8234	0.7222	0.8003	0.8030	0.9404	0.8840	0.6444	0.8561	0.5860	0.7844

(c) Average precision at 15° Azimuth error for different categories (Higher is better).

Model	bike	book	bottle	camera	cereal_box	chair	cup	laptop	shoe	average
MobilePose [16]	0.7130	0.6289	0.6999	0.5233	0.8030	0.7053	0.6632	0.5413	0.4947	0.6414
Two-stage	0.9390	0.8616	0.8567	0.8437	0.9476	0.9272	0.8365	0.7593	0.7544	0.8584

(d) Average precision at 10° Elevation error for different categories (Higher is better).

Table 2: Evaluation of different baseline models for the Objectron dataset.



(a) Clip a polygon from two boxes using the Sutherland-Hodgman algorithm. (b) Compute intersection volume (green) using the convex hull algorithm.

Figure 8: Accurate computation of 3D IoU using polygon-clipping algorithm.

implementation of 3D IoU metric, which to the best of our knowledge, is novel.

5.1. 3D Intersection Over Union

Intersection over Union (IoU), also known as *Jaccard* index, is an evaluation metric for computer vision tasks such as object detection, segmentation, and tracking. It measures how close the prediction bounding boxes are to the ground

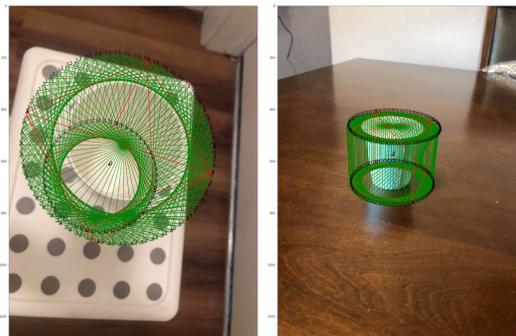


Figure 9: 3D IoU computation for symmetric objects: Rotating the bounding box along the Y axis of symmetry to maximize 3D IoU.

truth. The IoU takes two boxes as input: the predicted box (by the model) and the ground truth box and computes the intersection volume of the two boxes. The IoU metric is invariant under change in the scale, as well as any rigid transformations that belong to the SE3 group. Therefore if we change the coordinate system, or rotate, translate, and scale both boxes by the same transformation, their IoU does

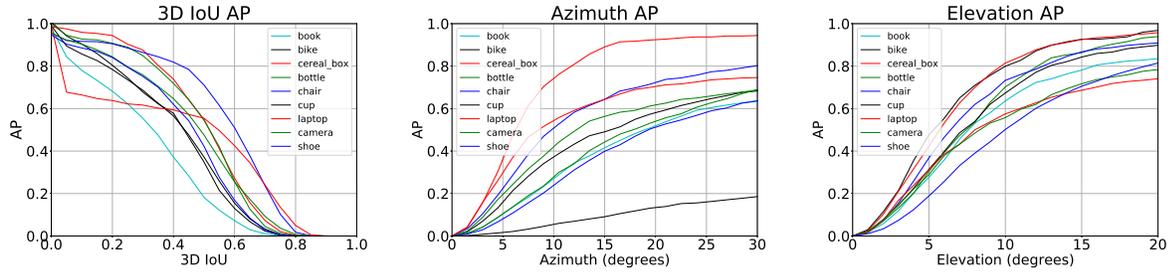


Figure 10: Evaluation of MobilePose network[16] on the Objectron dataset

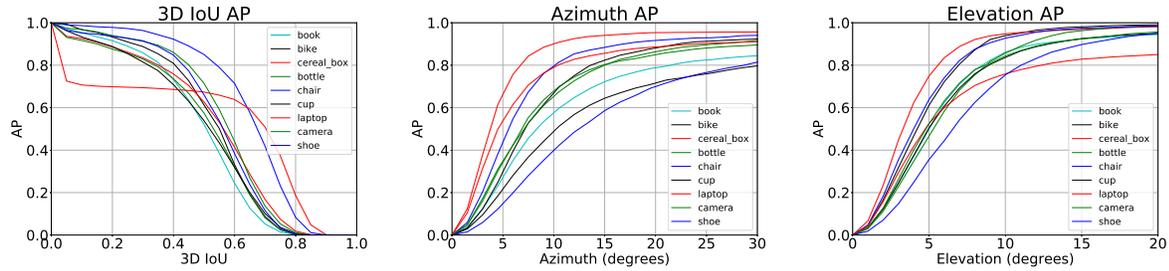


Figure 11: Evaluation of two-stage network on the Objectron dataset

not change. This property becomes useful later. The IoU is a normalized metric and its value ranges between 0 to 1, where 1 is considered the perfect score.

For evaluation, although 3D IoU has been used in previous work, its computation is overly simplified based on some assumptions. In [36], 3D boxes are assumed to be axis-aligned. Another approach, used by autonomous datasets (e.g. in [25] and others), is to project the 3D bounding boxes to the ground plane and then compute the intersection of the 2D projected polygons. Then the intersection volume is estimated by multiplying the area of the 2D intersecting polygon with the height of the 3D bounding boxes. Although this approach works for vehicles on the road, it has two limitations: 1) The object should sit on the same ground plane, which limits the degrees of freedom of the box from 9 to 7. The box only has freedom in yaw, and the roll and pitch are set to 0. 2) it assumes the boxes have the same height. For the Objectron datasets, these assumptions do not hold.

We propose an algorithm for computing accurate 3D IoU values for general 3D-oriented boxes. First, we compute the intersection points between the faces of the two boxes using the Sutherland-Hodgman Polygon clipping algorithm [10]. Let x denote the predicted box and y denote the annotation label. To compute the intersecting points between the boxes x and y , first transform both boxes using the inverse transformation of the box x . The transformed box x will be axis-aligned and centered around the origin while the box y is brought to the coordinate system of the box x and remains oriented. Volume remains invariant under rigid-body

transformation. We can compute the intersecting points in the new coordinate system and estimate the volume from the transformed intersection points. Using this coordinate system allows for more efficient and simpler polygon clipping against boxes since each surface is perpendicular to one of the coordinate axes.

Next, we clip each face in box y , which is a convex polygon, against the axis-aligned box x . There is a well-known polygon clipping algorithm in computer graphics [10], where each world polygon is clipped against the camera frustum to determine the rendered environment. We use a robust Sutherland-Hodgman algorithm to perform the clipping. To clip a polygon against a plane, the edges of the plane are hypothetically extended to infinity. We iterate over each edge in the polygon in clockwise order and determine whether that edge intersects with any faces in the axis-aligned box x . For each vertex in the box y , we check whether any of them are inside the box x . We add those vertices to the intersecting vertices as well. We repeat the whole process swapping the box x and y . We refer the readers to [10] for the details of the polygon clipping algorithm. Figure 8a shows an example of a polygon clipping.

The volume of the intersection is computed by the convex hull of all the clipped polygons, as shown in Figure 8b. Finally, the IoU is computed from the volume of the intersection and volume of the union of two boxes. We are releasing the evaluation metrics source code along with the dataset.

For symmetric objects, such as bottles or cups, the 3D IoU metric is not well-defined. In those instances, we rotate



Figure 12: Results of the NeRF [24] model on our dataset. 1) Input image with annotation, 2) NeRF rendering from the same view, 3) Estimated depth map from NeRF, 4) Extracted segmentation mask, and 5) Synthesized novel view.

the estimated bounding box along the symmetry axis and evaluate each rotated instance, then pick the bounding box that maximizes the IoU. Figure 9 shows an example for asymmetric cup.

5.2. Baselines for 3D object detection

We provide baseline results for 3D object detection and viewpoint estimation. We trained a state-of-the-art model [16] over our dataset for detecting 3D bounding boxes. MobilePose is a lightweight network that is designed to perform in real-time on mobile devices. We evaluate the network’s output using our evaluation code and report various metrics, such as average precision for 3D IoU, 2D pixel projection error, azimuth, and elevation¹. For each category, we trained the network separately without any pre-training or hyperparameter optimization. Table 2 shows the evaluation results. We empirically observed pre-training and hyperparameter optimization can significantly improve the baseline results. Each model was trained for 100 epochs (~12 hours) on eight V100 GPUs.

We also designed a new two-stage architecture for 3D object detection. The first stage estimates a 2D crop of the object of the size 224×224 using SSD model[23][17], followed by a second stage model using EfficientNet-Lite [33] architecture which uses the 2D crop to regress the keypoints of the 3D bounding box. We use a similar EPnP algorithm as in [16] to lift the 2D predicted keypoints to 3D. This network is very lightweight (5.2MB size) and runs at 83fps on Samsung S20 mobile GPU. Figure 11 shows the evaluation results, including the average precision plot for 3D IoU, azimuth, and elevation for the two-stage model.

For the average precision, first, the detector has to detect the 3D bounding box using the center of the box, then proceed to compute the other metrics. The experimental result shows the model is more accurate in estimating elevation than azimuth because the distribution of the elevation in our dataset (Figure 7) is biased toward 45° , but azimuth is uni-

¹The trained models and their evaluation report can be downloaded from <https://github.com/google-research-datasets/Objectron>

formly distributed. In other words, in our videos, the data collector is looking down and walking around the object to capture the video. Data augmentation techniques, such as affine transformation or cropping, can change the distribution of viewpoints in the dataset and help generalization. We should also highlight how badly the network performs on estimating the rotation of the ‘cup’, as evident by the average precision of azimuth for the cups (Figure 10 and Table 2c).

5.3. Baselines for Neural Radiance Field

Recently [24] has shown neural radiance field (NeRF) can learn the scene and object representation with fine details. The NeRF model learns the color and density value of each voxel in the scene and can generate novel views. We trained the NeRF [7] model on the optimized camera poses from our dataset (Figure 12). We used NeRF for two baselines: 1) Computing segmentation mask and 2) Novel view synthesis. We estimated the depth map of the image by synthesizing the same view. Then we re-scaled the depth map to metric scale using the depth of the box centroid. For each pixel, we cast a ray with the given depth to get the 3D point. Finally, we used a culling algorithm to check if the point is inside the 3D bounding box. As a result, we could compute the segmentation mask of the object for each frame in the video. Another application is to generate novel views from the same scene. Objectron dataset is slightly biased toward specific elevation views (Figure 7). We used the NeRF model to generate improbable views in the dataset.

6. Conclusion

This paper introduces the Objectron dataset: a large scale object-centric dataset of 14,819 short videos in the wild with object pose annotation. We developed an efficient and scalable data collection and annotation framework based on on-device AR libraries. By releasing this dataset, we hope to enable the research community to push the limits of 3D object geometry understanding and foster new research and applications in 3D understanding, video models, object retrieval, view synthetics, and 3D reconstruction.

References

- [1] ARcore. developers.google.com/ar. Accessed: 2020-11-16. **3**
- [2] ARKit. developer.apple.com/augmented-reality/. Accessed: 2020-11-16. **3**
- [3] Armen Avetisyan, Manuel Dahnert, Angela Dai, Manolis Savva, Angel X Chang, and Matthias Niessner. Scan2CAD: Learning CAD Model Alignment in RGB-D Scans. *Computer Vision and Pattern Recognition*, Nov. 2018. **3**
- [4] Berk Calli, Arjun Singh, James Bruce, Aaron Walsman, Kurt Konolige, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M Dollar. Yale-CMU-Berkeley dataset for robotic manipulation research. *The International Journal of Robotics Research*, 36(3):261–268, Apr. 2017. **2**
- [5] 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. *arXiv. cs.GR*, Dec. 2015. **3**
- [6] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Niessner. ScanNet: Richly-Annotated 3D Reconstructions of Indoor Scenes. *Proc. Computer Vision and Pattern Recognition*, pages 5828–5839, 2017. **2**
- [7] Boyang Deng, Jonathan T. Barron, and Pratul P. Srinivasan. JaxNeRF: an efficient JAX implementation of NeRF, 2020. **8**
- [8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255. IEEE, Apr. 2009. **1**
- [9] Andreas Doumanoglou, Rigas Kouskouridas, Sotiris Malasiotis, and Tae-Kyun Kim. Recovering 6D Object Pose and Predicting Next-Best-View in the Crowd. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3583–3592, 2016. **2**
- [10] Christer Ericson. *Real-Time Collision Detection*. CRC Press, 2004. **7**
- [11] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. *International Conference on Learned Representation*, Nov. 2019. **3**
- [12] James J Gibson. *The Ecological Approach to Visual Perception*. 1979. **2**
- [13] Stefan Hinterstoisser, Vincent Lepetit, Slobodan Ilic, Stefan Holzer, Gary Bradski, Kurt Konolige, and Nassir Navab. Model Based Training, Detection and Pose Estimation of Texture-Less 3D Objects in Heavily Cluttered Scenes. In *Asian Conference in Computer Vision (ACCV)*, pages 548–562. Springer, Berlin, Heidelberg, Berlin, Heidelberg, Nov. 2012. **2**
- [14] Tomas Hodan, Rigas Kouskouridas, Tae-Kyun Kim, Federico Tombari, Kostas Bekris, Bertram Drost, Thibault Groueix, Krzysztof Walas, Vincent Lepetit, Ales Leonardis, Carsten Steger, Frank Michel, Caner Sahin, Carsten Rother, and Jiri Matas. BOP: Benchmark for 6D Object Pose Estimation. *Computer Vision and Pattern Recognition*, (Chapter 36):589–600, Oct. 2018. **2**
- [15] Tomas Hodan, Pavel Haluza, Stepan Obdržálek, Jiri Matas, Manolis Lourakis, and Xenophon Zabulis. T-LESS: An RGB-D dataset for 6D pose estimation of texture-less objects. *IEEE Robotics and Automation Letters*. **2**
- [16] Tingbo Hou, Adel Ahmadyan, Liangkai Zhang, and Jianing Wei. MobilePose: Real-Time Pose Estimation for Unseen Objects with Weak Shape Supervision. *arXiv 2003.03522*, 2020. **6, 7, 8, 1, 3**
- [17] Jonathan Huang, Vivek Rathod, Chen Sun, Menglong Zhu, Anoop Korattikara, Alireza Fathi, Ian Fischer, Zbigniew Wojna, Yang Song, Sergio Guadarrama, and Kevin Murphy. Speed/Accuracy Trade-Offs for Modern Convolutional Object Detectors. In *Computer Vision and Pattern Recognition*, pages 7310–7311, 2017. **8**
- [18] H. Pirsiavash J. Lim and A. Torralba. Parsing ikea objects: Fine pose estimation. In *ICCV*, 2013. **2**
- [19] A. Owens J. Xiao and A. Torralba. Sun3d: A database of big spaces reconstructed using sfm and object labels. In *ICCV*, 2013. **2**
- [20] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Mallocci, Alexander Kolesnikov, Tom Duerig, and Vittorio Ferrari. The Open Images Dataset V4: Unified image classification, object detection, and visual relationship detection at scale. *International Journal of Computer Vision*, (7):1956–1981, Nov. 2020. **1**
- [21] Yann Labbé, Justin Carpentier, Mathieu Aubry, and Josef Sivic. CosyPose: Consistent multi-view multi-object 6D pose estimation. In *Proceedings of the European Conference on Computer Vision*, Aug. 2020. **3**
- [22] 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. *Computer Vision and Pattern Recognition*, May 2014. **1**
- [23] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. SSD: Single Shot MultiBox Detector. *Computer Vision and Pattern Recognition*, (Chapter 2):21–37, Dec. 2015. **8**
- [24] 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 *ECCV*, 2020. **8**
- [25] Arsalan Mousavian, Dragomir Anguelov, John Flynn, and Jana Kosecka. 3D Bounding Box Estimation Using Deep Learning and Geometry. *Computer Vision and Pattern Recognition*, Dec. 2016. **7**
- [26] Derek Hoiem P. K. Nathan Silberman and R. Fergus. Indoor segmentation and support inference from rgb-d images. In *ECCV*, 2012. **2**
- [27] Colin Rennie, Rahul Shome, Kostas E Bekris, and Alberto F De Souza. A Dataset for Improved RGBD -based Object Detection and Pose Estimation for Warehouse Pick-and-Place. *IEEE Robotics and Automation Letters*. **2**

- [28] Mike Roberts and Nathan Paczan. Hypersim: A Photorealistic Synthetic Dataset for Holistic Indoor Scene Understanding. *arXiv 2003.03522*, 2020. 3
- [29] S. P. Lichtenberg S. Song and J. Xiao. Sun rgb-d: A rgb-d scene understanding benchmark suite. In *CVPR*, 2015. 2
- [30] Silvio Savarese and Fei-Fei Li. 3D generic object categorization, localization and pose estimation. In *IEEE Workshop on Applications of Computer Vision*, 2007. 2
- [31] Srinath Sridhar, Davis Rempe, Julien Valentin, Bouaziz Sofien, and Leonidas J Guibas. Multiview Aggregation for Learning Category-Specific Shape Reconstruction. *Conference on Neural Information Processing Systems*, pages 2348–2359, 2019. 3
- [32] 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. *Computer Vision and Pattern Recognition*, Apr. 2018. 2
- [33] Mingxing Tan and Quoc V Le. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. *Computer Vision and Pattern Recognition*, May 2019. 8
- [34] Shubham Tulsiani, Or Litany, Charles R Qi, He Wang, and Leonidas J Guibas. Object-Centric Multi-View Aggregation. July 2020. 3
- [35] Johanna Wald, Armen Avetisyan, Nassir Navab, Federico Tombari, and Matthias Niessner. RIO: 3D Object Instance Re-Localization in Changing Indoor Environments. *Proceedings IEEE International Conference on Computer Vision*, pages 7658–7667, 2019. 3, 2
- [36] H Wang, S Sridhar, J Huang, J Valentin, S Song arXiv preprint arXiv, and 2019. Normalized Object Coordinate Space for Category-Level 6D Object Pose and Size Estimation. In *The IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 7
- [37] Fei Xia, Amir R Zamir, Zhi-Yang He, Alexander Sax, Jitendra Malik, and Silvio Savarese. Gibson Env: Real-world perception for embodied agents. In *Computer Vision and Pattern Recognition*, 2018. 2
- [38] Yu Xiang, Wonhui Kim, Wei Chen, Jingwei Ji, Christopher Choy, Hao Su, Roozbeh Mottaghi, Leonidas Guibas, and Silvio Savarese. ObjectNet3D: A Large Scale Database for 3D Object Recognition. In *European Conference on Computer Vision*, pages 160–176, Cham, Oct. 2016. Springer, Cham. 2
- [39] Y Xiang, R Mottaghi, and S Savarese. Beyond PASCAL: A benchmark for 3D object detection in the wild. In *IEEE Workshop on Applications of Computer Vision*, Mar. 2014. 2