EgoObjects: A Large-Scale Egocentric Dataset for Fine-Grained Object Understanding
– Supplementary Materials –

Chenchen Zhu, Fanyi Xiao, Andrés Alvarado, Yasmine Babaei, Jiabo Hu
Hichem El-Mohri, Sean Chang, Roshan Sumbaly, Zhicheng Yan
Meta AI
{chenchenz, fanyix, zyan3}@meta.com

video ID	distance	camera motion	background	lighting
01	near	horizontal	simple	bright
02	medium	horizontal	simple	bright
03	near	horizontal	simple	dim
04	medium	horizontal	busy	bright
05	far	horizontal	busy	bright
06	medium	vertical	busy	bright
07	medium	combined	busy	bright
08	near	horizontal	busy	dim
09	medium	horizontal	busy	dim
10	far	horizontal	busy	dim

Table S1: *Different settings of video condition variables for 10 videos of each main object.*

loss weight			val		
index	positive	negative	AP	AP50	AP75
✗	✓	✗	12.9	25.3	11.7
✗	✓	✓	$13.6_{+0.7}$	$27.2_{+1.9}$	$12.1_{+0.4}$
✓	✓	✗	$17.4_{+4.5}$	$31.4_{+6.1}$	$17.1_{+5.4}$
✓	✓	✓	$18.7_{+5.8}$	$35.0_{+9.7}$	$17.7_{+6.0}$

Table S2: ***Ablation study on the training losses of TA-IDet.*** *We train TA-IDet with different combinations of losses and report the performance on the EgoObjects* `val` *split. Backbone is ResNet-50.*

A. Additional details for data collection

Given 4 variables including object distance, camera motion, background complexity, and lighting, participants are instructed to collect 10 videos of each main object according to 10 predefined configurations, which is presented in Table S1. Figure S1 shows representative frames of these 10 videos featuring one main object - pressure cooker.

Moreover, we attach the full hierarchical structure of the object category taxonomy in `Object Taxonomy.pdf`, a 7-layer tree structure with a total of 638 leaf nodes.

B. Additional details for data annotation

We demonstrate the annotation tools for our multi-stage annotation process in Figure S2. Both tools have video player as the back end. We enqueue all 10 videos of the same main object into a single job to ensure label consistency.

For stage 1 (category discovery) and stage 2 (exhaustive instance labeling), we use a per-frame object bounding box interface (Figure S2a with some panels on the right for additional information. Annotators are asked to label a track for each identified instance across the frames by drawing bounding boxes and assigning them consistent category label and instance ID. Annotators can consult the "main object category" panel and the "statistics" panel for the number of annotated categories and number of instances for each category in the current frame. Annotators can also quickly jump to other frames of the track of an instance by navigating through the "Objects" panel.

For stage 3 (negative category verification), we use a per-frame attribute classification interface (Figure S2b). On the right panel there is a list of categories with check boxes. Annotators are asked to check for each category whether there is any instance of that category in the image. Categories not ticked are verified negative categories for the image.

Figure S1 shows representative annotations of both the main object and the surrounding secondary objects.

C. Ablation studies on TA-IDet

At model training, we can compute three losses: positive loss, index loss, and negative loss. The positive loss is the instance detection loss in the positive image, which contains the same object instance in a different view as in the index image. The index loss is the instance detection loss in the index image. The supervision signal is the model should be able to localize and recognize the same object instance in the original index image. The negative loss refers to the

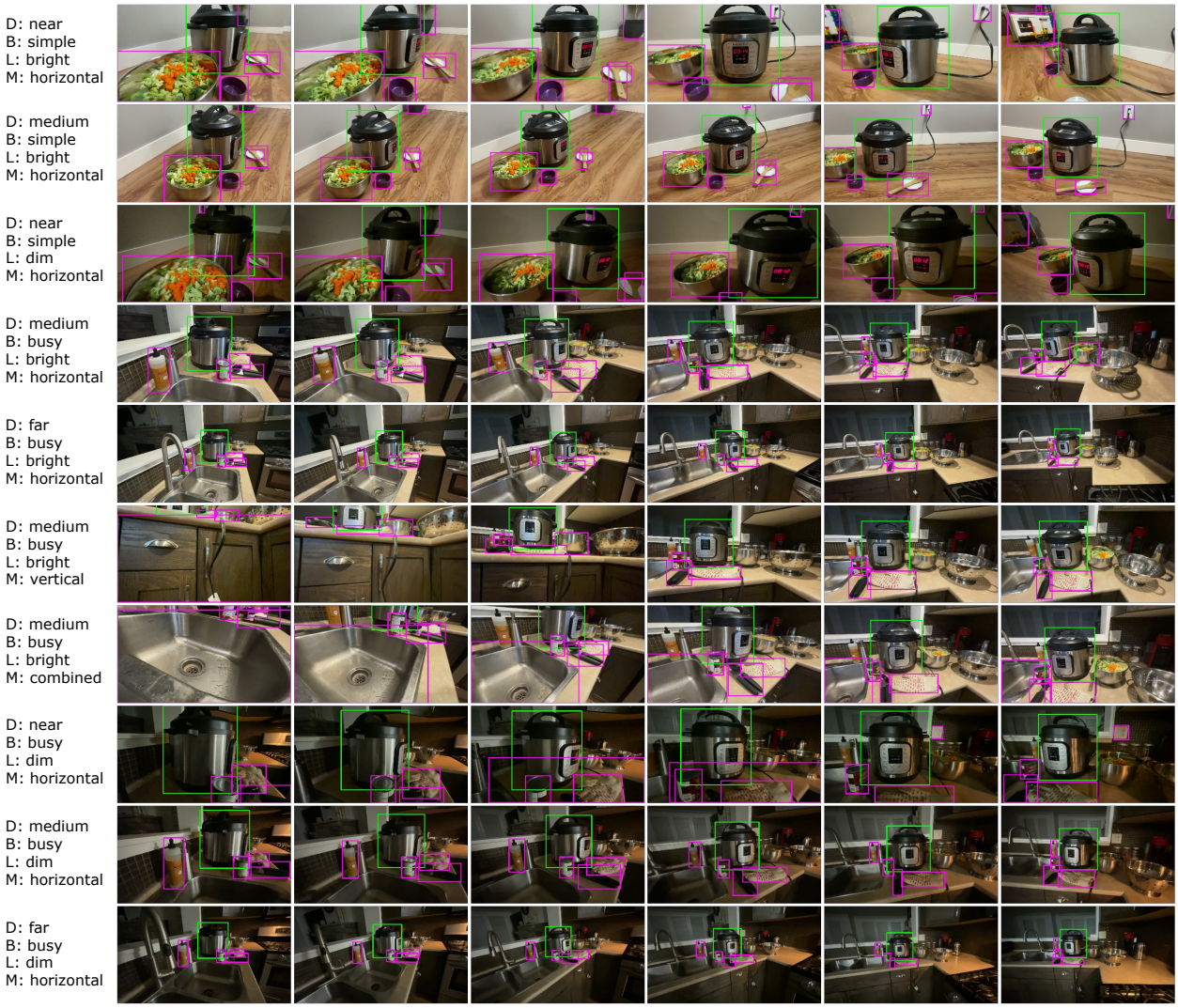


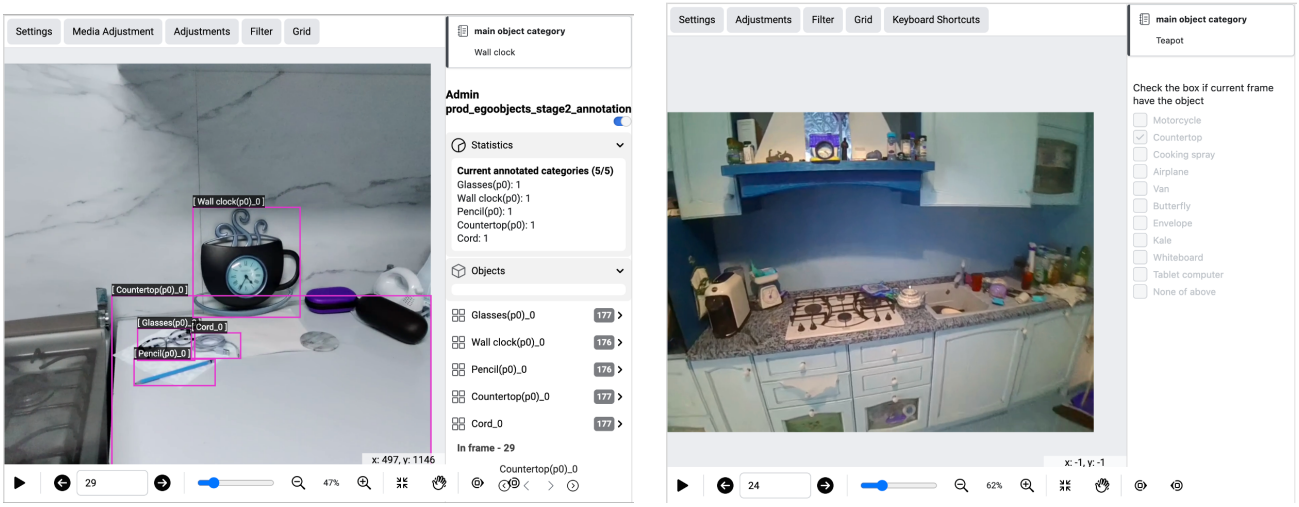
Figure S1: ***EgoObjects data visualization.*** *We show video frames of objects caputred from the first-person viewpoint under 10 conditions as in Table S1. Multiple objects are annotated with both category labels and instance IDs. In each row we visualize the annotations of both the main object (green box) and the secondary objects (pink box) in on video under one set of condition variables. For clarity, we use shorthand notations: D – Distance, B – Background, L – Lighting, M – Camera Motion. We omit category labels and instance IDs for clarity. Best viewed digitally.*

classification loss in the negative image. The negative image does not contain the same object instance in the index image. Regardless of what object bounding box is predicted by the instance localization module, the object confidence score is minimized. We study the effects of the index loss and the negative loss by setting their loss weights to zero and compare with model trained with all the losses. In Table S2, we confirm the highest accuracy is achieved when all three losses are used.

Additionally, we study the effects of different T^{cls} feature resolutions $S \times S$ in Table S3. We find $S = 5$ produces the best accuracy.

	val		
S	AP	AP50	AP75
1	11.3	20.4	11.3
3	18.5	34.6	17.6
5	**18.7**	**35.0**	**17.7**
7	18.3	34.3	17.4

Table S3: ***Ablation study on the T^{cls} feature resolution of TA-IDet.*** *We train TA-IDet with different T^{cls} feature resolution S and report the performance on the EgoObjects* `val` *split. Backbone is ResNet-50.*



(a) *Annotation UI for stage 1&2.* (b) *Annotation UI for stage 3.*

Figure S2: ***Annotation tools for multi-stage process***. *Stage 1&2 is done in a per-frame object bounding box interface (left). Stage 3 is done in a per-frame attribute verification interface (right).*

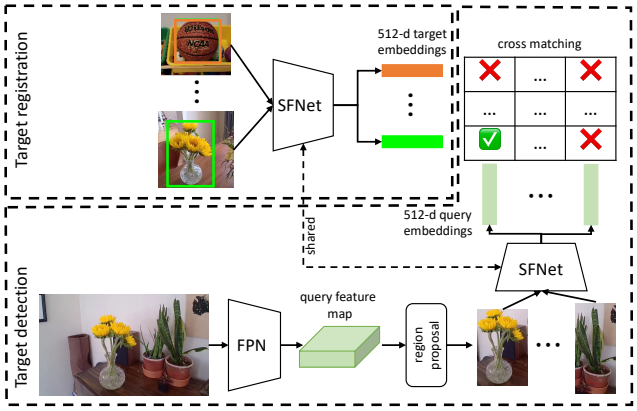


Figure S3: ***Architecture of target-agnostic instance detector. Top:*** *in target registration, target embeddings are generated from SFNet [13].* ***Bottom:*** *during target detection, the model generates region proposals and their corresponding query embeddings, which are cross-matched with target embeddings via cosine similarity.*

D. Implementation details of target-agnostic instance detector

We propose a simple baseline approach *RPN+SFNet* as the target-agnostic instance detector. It consists of a Region Proposal Network (RPN) [11] for object localization and a SFNet model [13], commonly used in metric learning, for object classification. It supports two modes, namely target registration and target detection, see Figure S3.

Target registration. To register a new target instance, we crop the instance out from the reference image and resize the crop to a fixed size (112×112). Then a 512-dimensional embedding is extracted by passing the crop through the SFNet and added to an index of target object embeddings. If several reference images per target are provided, the target embedding is averaged.

Target detection. At detection time, we feed the query image into RPN to generate several instance proposals agnostic to the targets in the index. Each object proposal is cropped from the query image and also resized to 112×112. Then we use the same SFNet as in target registration to extract 512-dimensional query embedding for each proposal. All the query embeddings are matched against all the target embeddings in the index by computing the cosine similarity. We pick the target instance in the index with the highest cosine similarity as the matched one for each proposal. We also apply a threshold to the matching score for rejecting low confident matches.

Model training. During training, the RPN is trained on the `train` split with all the object bounding boxes in a instance-agnostic way. For the SFNet, we build a classification dataset by pre-cropping all the instances from `train` images and assign them labels as the instance IDs, so that each instance has several samples of multiple views. The SFNet model is trained on this classification dataset with the SphereFace2 [13] loss, which encourages small distances between embeddings of multiple views for the same instance and large distances for embeddings of different instances. SphereFace2 is the state-of-the-art deep face recognition method. It constructs a novel binary classification framework to align the training objective with open-set verification, outperforming other metric learning losses on recognizing unseen objects. We follow the best practices of

splits	#train img	#val img	#test img	#train inst	#eval inst$_{sc}$	#eval inst$_{uc}$
Unified	79K	5.7K	29.5K	9.6K	4.1K	25
InstDet	104K	2.5K	7.8K	12.2K	1346	131

Table S4: ***Comparison between the new InstDet splits and Unified splits.*** *Eval inst$_{sc}$ and eval inst$_{uc}$ mean instances in the evaluation set with seen categories and unseen categories respectively. InstDet splits have more instances with unseen categories for better understanding the challenges in instance detection.*

[13] and adopt its hyperparameters in our model training.

E. Dataset challenges

We present several qualitative results of instance-level detection in Figure S4 and category-level detection in Figure S5. EgoObjects is challenging.

For instance-level detection, one dominant challenge is the massive number of instances. An ideal instance detector should be able to output high confidence score for each target while suppressing the confidence scores of other targets. But we observed the prediction of multiple instances with high confidence on a single object, even from the best TA-IDet-R101 model.

For category-level detection, challenges mainly come from the diverse capture conditions like far distance, cluttered background, dim lighting, extreme view angles, etc.

F. New splits for instance detection with seen/unseen categories

In the benchmark tasks of the main text (Section 4), we designed a set of data splits for benchmarking both category-level detection and instance-level detection uniformly, which maximizes the category overlap for training and testing, resulting in only 25 instances in the evaluation with unseen categories. We refer to this set of splits as **Unified** splits.

To further study the challenges in generalizing instance detection models to instances with unseen categories, we created a new set of data splits, named **InstDet** splits, to have more instances with unseen categories.

For the InstDet splits, we also divide the dataset into 4 subsets: `train`/`target`/`val`/`test`. The `train` split contains 12.2k instances with a total of 543k annotations from 104k images. The `target`, `val`, `test` splits share the remaining 1.5k instances which do not appear in the `train` images, and their categories can also be unseen during training. Among those 131 instances have *unseen* categories. In the `target` split, there is a single reference image and one annotation for each instance. The `val` and `test` splits have 2.5K and 7.8K images respectively. We compare the new InstDet splits with Unified splits in Table S4.

backbone	method	val AP	val AP50	val AP50$_{se}$	val AP50$_{un}$	test AP	test AP50	test AP50$_{se}$	test AP50$_{un}$
R50	RPN+SFNet	17.9	29.7	29.8	28.9	16.4	26.9	27.2	24.3
	TA-IDet	18.4	35.0	35.3	31.3	19.6	37.1	37.4	33.6
R101	RPN+SFNet	18.5	30.6	30.9	27.6	16.9	27.4	27.7	24.4
	TA-IDet	23.3	41.3	42.0	35.3	23.5	41.6	42.3	34.7

Table S5: ***Instance-level detection benchmarking results on the InstDet splits of EgoObjects.*** *AP50$_{se}$ and AP50$_{un}$ are computed for instances with categories seen and unseen during training. The proposed* TA-IDet *model significantly outperforms the baseline* RPN+SFNet *approach.*

Table S5 reports the results. There are clear gaps between AP$_{se}$ and AP$_{un}$ on both `val` and `test` splits, reflecting the challenges in generalizing the models to unseen categories. Models with R101 backbone tend to have larger gaps compared to models with R50 backbone, indicating the trend of overfitting to seen categories. Especially for RPN+SFNet baseline, the larger backbone does not bring as much gain on AP50$_{un}$ as the TA-IDet approach. On the other hand, TA-IDet robustly outperforms RPN+SFNet on AP50$_{un}$ in all the settings, indicating its superiority of generalizing to instances with unseen categories.

G. Discussions on CL benchmarks

G.1. CL instance detection.

The continual learning of the instance-level object detection is an under-explored area. The top submissions basically follow the similar strategy of applying continual learning techniques to a base category-level object detector and treating each instance as an individual category. For the base detector, both single-stage and two-stage detectors are applicable. The rank-1 team and the rank-3 team are using single-stage detectors (VarifocalNet [14] and FCOS [12]) while the rank-2 team is using two-stage detector (Faster R-CNN [11]). And the detectors are pretrained on either COCO [9] or LVIS [6] which are not the major factor of performance. For the network architecture, ResNet [7] and its variants [4] are still the dominating backbones. One key factor contributing to the performance is the replay buffer. All submissions observe that the use of full replay buffer is necessary for high performance. But they adopt different sampling strategies for filling the buffer. Specifically, rank-1 team samples an experience-balanced buffer, whereas rank-2 team and rank-3 team uses video-balanced buffer and instance-balanced buffer respectively. Distillation techniques are also widely used by rank-1 and rank-3 teams. However, these submissions still treat the instance detection as a close-set problem where the number of instances are fixed. This limits their adoption for real-world applications where the number of instances can scale up continuously.

Figure S4: *Visualization of ground truths (left) and predictions (right) for* ***instance-level detection****. Predictions are produced from a TA-IDet-R101 model on the* `val` *split. The labels on top of boxes contain instance IDs. Best viewed digitally.*

G.2. CL category detection.

There are several interesting observations towards the top submissions. First, when dealing with the catastrophic forgetting, both the rank-1 and rank-3 teams adopted a uniform sampling strategy to cache data from previous tasks, with the difference in that the former approach samples history data in a video-wise manner while the latter did so on a frame basis. Unlike these two teams, the rank-2 team proposes to follow a "Minority Replay" strategy in which they sort and emphasis on selecting and buffering data from rare classes. In addition to replay buffer, the rank-1 team proposes to distill information from a teacher model trained in previous tasks to the model for the current task. However, these top submissions made limited changes to the detector architecture. The rank-1 team and rank-3 team directly adopt the VarifocalNet [14] and Faster R-CNN [11], respectively. The rank-2 team proposed to train a separate detection head for each experience, leading to a multi-head Faster R-CNN. This shows the potential of designing architectures tailored to the CL tasks. But the multi-head solution is not scalable. Therefore, there is still huge room for innovative ideas in this field.

H. Video samples

We provide a group of 10 videos featuring the main object "soccer ball" in folder `samples`.

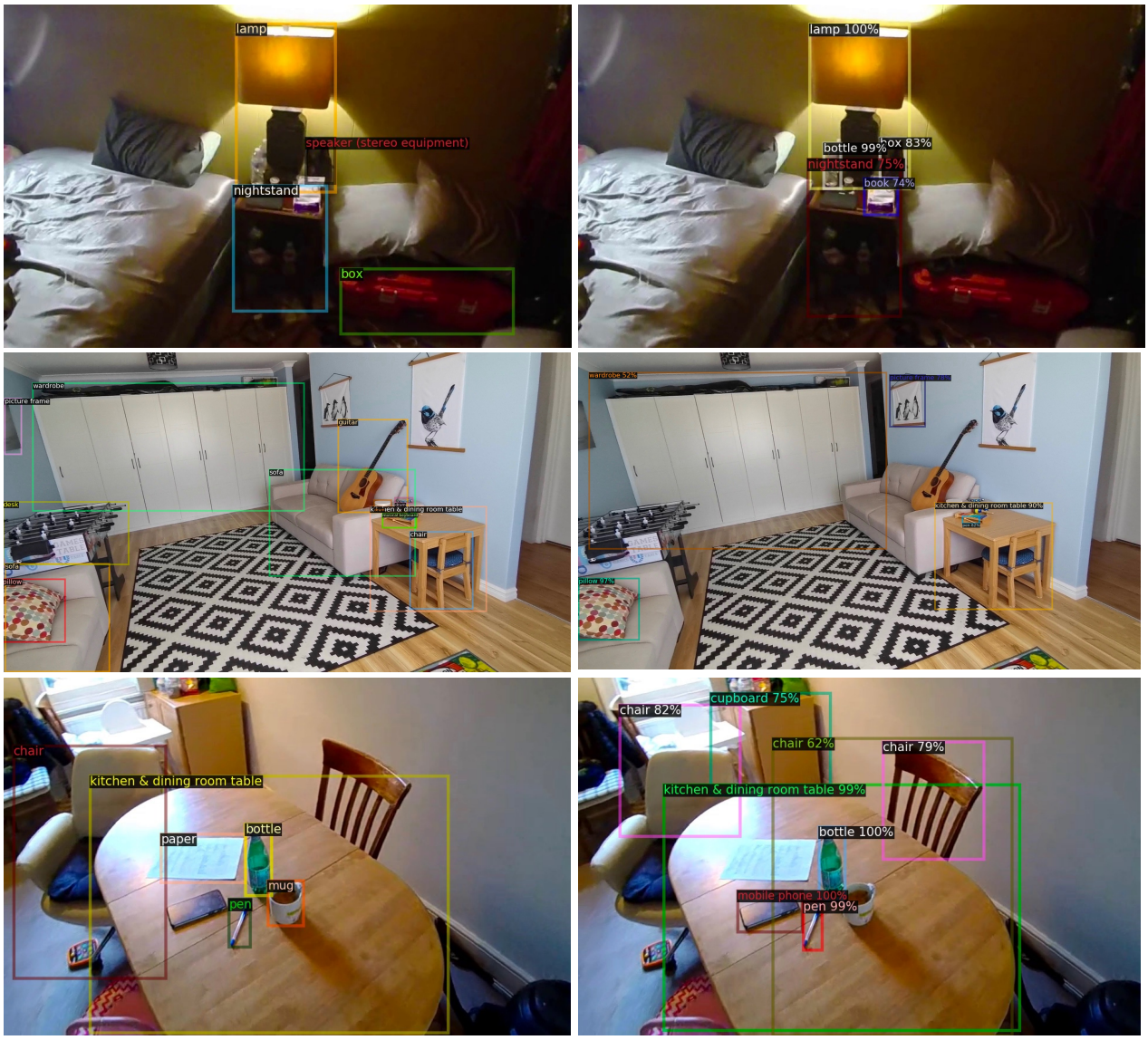


Figure S5: *Visualization of ground truths (left) and predictions (right) for* ***category-level detection****. Predictions are produced from a FasterRCNN-R101 model on the* `val` *split. Best viewed digitally.*

I. Predict 3D geometry on EgoObjects videos

In this section, to demonstrate the potential of creating a 3D dataset from EgoObjects using a fully/semi-automatic pipeline, we leverage the latest development in neural scene representations to predict 3D geometry labels on EgoObjects videos [10, 2]. We predict both the 3D point clouds and meshes on several example videos from EgoObjects (see videos under `/3d` directory). To create the inputs for NeRF training, we sample video frames at 1 FPS and use NeRFacto [1] to train one model for each video. We render outputs at 15 FPS to generate all videos. Taking `coffeecup.mp4` as an example, from left to right we present the rendered RGB frames, the estimated depth map, the predicted point clouds, and finally the predicted meshes (sample frames are presented in Fig. S6). Meshes are constructed from point clouds using Poisson surface reconstruction [8]. Given the 2D bounding box annotations and the predicted 3D point clouds and meshes, we can easily intersect the two to produce 3D geometry for specific objects, which could potentially be used to train 3D bounding box detectors (*e.g.*, CubeRCNN [3]) and 3D mesh predictors (*e.g.*, MeshRCNN [5]).

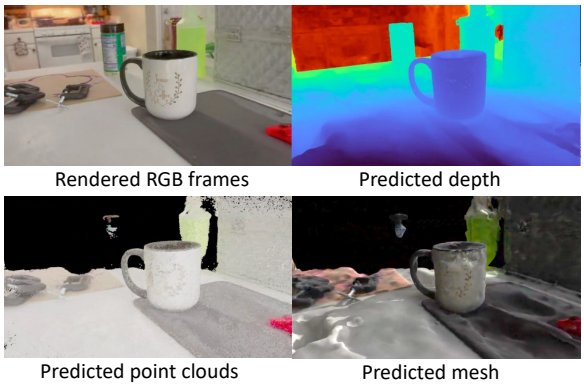


Figure S6: ***Predict 3D geometry on EgoObjects videos***. *We show sample frames from the video of* `3d/coffeecup.mp4`.

References

[1] Nerfacto: the defacto training method used in nerfstudio. https://docs.nerf.studio/en/latest/nerfology/methods/nerfacto.html, 2022. 6

[2] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *CVPR*, 2022. 6

[3] Garrick Brazil, Julian Straub, Nikhila Ravi, Justin Johnson, and Georgia Gkioxari. Omni3D: A large benchmark and model for 3D object detection in the wild. *arXiv:2207.10660*, 2022. 6

[4] Shang-Hua Gao, Ming-Ming Cheng, Kai Zhao, Xin-Yu Zhang, Ming-Hsuan Yang, and Philip Torr. Res2net: A new multi-scale backbone architecture. *IEEE transactions on pattern analysis and machine intelligence*, 43(2):652–662, 2019. 4

[5] Georgia Gkioxari, Jitendra Malik, and Justin Johnson. Mesh R-CNN. In *ICCV*, 2019. 6

[6] Agrim Gupta, Piotr Dollar, and Ross Girshick. Lvis: A dataset for large vocabulary instance segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5356–5364, 2019. 4

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 4

[8] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, 2006. 6

[9] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 4

[10] 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. *Communications of the ACM*, 2021. 6

[11] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015. 3, 4, 5

[12] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. FCOS: Fully convolutional one-stage object detection. In *ICCV*, 2019. 4

[13] Yandong Wen, Weiyang Liu, Adrian Weller, Bhiksha Raj, and Rita Singh. Sphereface2: Binary classification is all you need for deep face recognition. *arXiv preprint arXiv:2108.01513*, 2021. 3, 4

[14] Haoyang Zhang, Ying Wang, Feras Dayoub, and Niko Sunderhauf. Varifocalnet: An iou-aware dense object detector. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8514–8523, 2021. 4, 5