

Move to See Better: Self-Improving Embodied Object Detection — Supplementary Materials

BMVC 2021 Submission # 615

1 Overview

The structure of this supplementary file is as follows: Section 2 provides a more detailed analysis of the method by performing ablation and comparison experiments. Section 3 includes additional visualizations to help better understand the both method’s strengths and limitations. Section 4 includes some further details on the implementation. We also include a video named “sup_vid_615.mp4” with additional 3D visualizations and urge the reviewers to refer that as well.

2 Experiments and Ablations

2.1 Weakly-Supervised Novel Object Detection

mAP@IoU	Method Name	Cushion	Nightstand	Shelf	Beanbag	Avg
0.5	SbM-ws Trained	93.62	81.25	24.38	82.18	70.35
	Limited GT Trained	87.24	79.79	16.40	88.77	68.04
0.3	SbM-ws Trained	94.23	81.25	25.61	82.18	70.81
	Limited GT Trained	87.24	79.79	16.40	88.77	68.04

Table 1: **SbM-ws labels can be used to train detectors on novel categories.** We compare the performance of the detector trained on labels produced by SbM-ws with the detector trained on ground truth labels. The results show that the detector trained on SbM-ws labels outperforms the on trained on limited ground truth labels.

Since embodied agents typically encounter novel objects while exploring, we applied our method to detect novel categories with a small number of human annotations.

We show that our method can perform pseudo-label generation when we have weak ground-truth 2D annotations, enabling us to generate high quality labels for non-COCO

instances and categories where the pre-trained detector fails. In our weakly supervised experiments, we only provide a ground truth annotation for one out of the 25 available views for each object instance. We denote this setup as SbM-ws.

We provide weak supervision for four novel non-COCO objects: Cushion, Nightstand, Shelf and Beanbag. To make a fair comparison, we compare the performance of the detector fine-tuned on pseudo-labels with the detector fine-tuned on ground truth labels for all views. The results of the experiment are shown in Table 1. We see that the detector trained on SbM pseudo-labels outperforms the detector trained on the limited ground truth data. This is because our method effectively creates more training data by propagating the weak supervision to more views.

2.2 Pre-trained Detector Quality

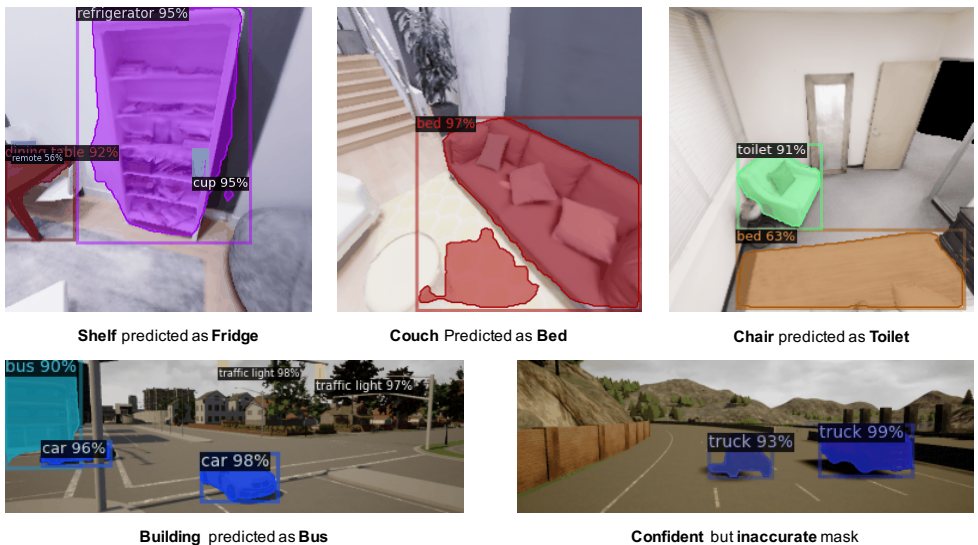


Figure 1: **Incorrect detections by pre-trained detector with high confidence.** We show three examples where the pre-trained detector incorrectly classify the object with high confidence.

As mentioned in the paper, a possible limitation of our method is that in order to propagate high-quality labels, the pre-trained detector must detect objects correctly with high confidence. In novel environments and viewpoints, the pre-trained detector sometimes detects wrong objects with high confidence, as shown in Figure 1. In this experiment, we ask the following question: is the confidence score of the pre-trained detector reliable?

To investigate this question, we set different confidence score thresholds θ for the pre-trained detector and report their precision, recall, and mAP@IoU=0.5 on CARLA training set. The results are shown in Table 2. We observe that although the detector’s confidence does not serve as a perfect cue all the time, it is calibrated enough for our method’s assumption to hold. Therefore, by setting a high confidence threshold we are able to obtain high precision detections. Our 3D segmentation then propagates the high precision detections to

all views, resulting in pseudo-labels with both high precision and high recall, as shown in the last column in Table 2. We also note that recent works [2] have shown promising results in training classifiers that are well-calibrated while preserving the performance.

Setting	Precision	Recall	mAP@0.5
Pre-trained, $\theta = 0.5$	83.97	81.91	68.05
Pre-trained, $\theta = 0.6$	85.66	80.43	68.96
Pre-trained, $\theta = 0.7$	87.46	76.74	67.60
Pre-trained, $\theta = 0.8$	89.58	70.71	64.06
Pre-trained, $\theta = 0.9$	92.41	60.99	57.23
Pseudo-labels, $\theta = 0.9$	92.92	92.76	86.88

Table 2: **The pre-trained detector is overall well-calibrated.** We set different confidence score thresholds for the pre-trained detector and report their precision, recall, and mAP@IoU=0.5 on CARLA training set.

2.3 Design Choice of 3D Segmentation

In our method, we employ a simple yet effective two-stage 3D segmentation method for its fast runtime and strong performance. In this experiment, we ask the following question: will an alternative “deep” segmentation method improve the performance, without sacrificing runtime?

To investigate this question, we replace the unary classifier in our segmentation module with a PointNet [5]. For each scene, an input sample to the PointNet is a random subset of $P \in \mathbb{R}^{M_i \times 6}$ with M_i being the number of pixels with an associated depth value. We sample randomly while making sure that in each sample there exist both points from P_{fg} and P_{bg} . During training, loss is only computed for points from P_{fg} and P_{bg} . We also ablate the effectiveness of our refinement stage with CRF, reporting the performance both with and without CRF refinement. We perform this experiment on CARLA training set to evaluate the resulting pseudo-label quality. The results are shown in Table 3. We observe that PointNet improves marginally compared to the SVM alone, while being about $280\times$ slower. Adding CRF refinement stage significantly improves SVM, while only marginally improving PointNet. Therefore, we show that our two-stage 3D segmentation method is simple, fast, and effective. It is important to note that training a single PointNet on many P ’s from many episodes might yield better performance for such a deep model, but that requires many more episodes of data collection. In contrast, our method operates on each episode separately and generates high-quality pseudo-labels even in the low-data regime.

2.4 Actuation Noise

In Section 4.2 of the paper, we mentioned that our method show consistent improvement over the baseline even under actuation noise modeled by a real robot. Here, we provide the details. To simulate actuation noise as observed in a real-world robot, we apply the noise model from Chaplot *et al.* [10]. This fits a separate Gaussian Mixture Model for each

Method	mAP@0.5	mAP@0.3	runtime (s)
SVM	67.57	87.44	1.18
SVM+CRF	86.88	92.93	1.46
PointNet	69.52	84.47	332.04
PointNet+CRF	71.85	85.22	332.32

Table 3: Performance and runtime comparison with PointNet. We perform comparison between our two-stage segmentation method and PointNet, as well as ablate the effectiveness of CRF as a refinement module.

action (move forward, turn right, turn left) based on noise measurements from LoCoBot¹. To register a cleaner scene pointcloud from the noisy measurements, we refine the camera pose by optimizing a cycle consistency objective based on flow from depth.

Due to the variability in the egomotion estimates obtained from this setting, we apply an additional constraint to remove pairs of frames where the optical flow obtained from the egomotion estimate is not cycle-consistent. This is a well-known “check” for optical flow – if flow does not align when computed in the forward $t \rightarrow t+1$ and the backward $t+1 \rightarrow t$ direction, it is not likely to be correct. We use this strategy to identify exceptionally poor egomotion estimates between frames to remove them. We first generate forward flow $\mathbf{w}_{t \rightarrow t+1}$ and backward flow $\mathbf{w}_{t+1 \rightarrow t}$ between each pair of views by warping the 3D point cloud from the first view to the second view using the noisy egomotion estimate, and take the delta between 2D points to be the flow. We estimate the rigid motion twice: once using the forward flow, and once using the backward flow (which delivers an estimate of the inverse transform, or backward egomotion). We then measure the inconsistency of these results, by applying the forward and backward motion to the same pointcloud, and measuring the displacement:

$$XYZ'_0 = RT_{10}^{bw} RT_{01}^{fw} (XYZ_0) \quad (1)$$

$$err = \underset{n}{average}(\|XYZ'_0 - XYZ_0\|), \quad (2)$$

where RT_{01}^{fw} denotes the rotation and translation computed from forward flow, which carries the pointcloud from timestep 0 to timestep 1, and RT_{10}^{bw} is the backward counterpart. If the average displacement across the entire pointcloud is above a threshold (set to 0.1 meters), then we treat the egomotion estimate for that image pair as “incorrect”, and remove that pair of frames from the analysis. We keep a minimum of 10 views and a maximum of 25 views (by taking 25 views with lowest error) from this process.

We show quantitative results of our method when the actuation noise is added. The pseudo-label quality is evaluated on the Replica training set and the CARLA training set, shown in Table 4 and Table 5, respectively. We observe that our method is still able to produce significantly better pseudo-labels compared to the pre-trained detector’s predictions, even under realistic noise settings. In addition, we show qualitative segmentation results in the included video.

¹<http://locobot.org>

Method	mAP@0.5	mAP@0.3
Pre-trained	21.36	26.14
SbM Labels w/ noise (ours)	23.15	33.68
SbM Labels w/o noise (ours)	26.20	38.12

Table 4: **Pseudo-label accuracy with pose noise in the Replica training set.** We show that actuation noise weakens the data collection, yet our method is still able to produce pseudo-labels that are better than the pre-trained detectors’ predictions.

Method	mAP@0.5	mAP@0.3
Pre-trained	68.05	73.09
SbM Labels w/ noise (ours)	75.75	91.44
SbM Labels w/o noise (ours)	86.88	92.93

Table 5: **Pseudo-label accuracy with pose noise on the CARLA training set.** We show that actuation noise weakens the data collection, yet our method is still able to produce pseudo-labels that are better than the pre-trained detector’s predictions.

2.5 Effect of the Number of Views

In our experiments in the main paper, we set the number of views to $N = 25$ in all cases. In this experiment, we ablate the effect of adding more views to the scene point cloud on the performance of pseudo-labels. The quality of pseudo-labels is evaluated on the CARLA training set. The results are shown in Table 6. We observe a steady increase in accuracy initially, but it starts to saturate with an increased number of images. Since our method require at least 1 confident detection out of all views, scaling the number of views initially increases the probability of finding a confident detection. After a certain number of views, confident views are captured more often than not, resulting in a diminishing return in accuracy of pseudo labels. However, we expect the performance of MaskRCNN fine-tuned with our pseudo labels to increase monotonously with increasing views simply because of increased training dataset size.

# views	2	5	10	15	20	25
mAP	64.9	75.9	84.4	85.5	87.2	86.9

Table 6: **Effect of the number of views** We report mAP@IoU=0.5 on the CARLA training set, when we use 2, 5, 10, 15, 20, 25 views in each episode. When the number of views are less than 25, we randomly sample without replacement.

2.6 Multi-view Aggregation for Segmentation

Since in an episode we have multiple views of the objects, it is often likely that multiple confident detections of the same object from different views exist. Therefore, another natural question that may arise in our setup is: does leveraging multi-view information help segmentation?

In this experiment, we explore a variant of our method that aggregates confident detections from multiple views to obtain the 3D segmentation. However, correspondence does not come for free: we don't know how different detections from different views correspond to one another. To sidestep this limitation and combine multiple detections of the same object from different views, we use a voting mechanism. For the i -th object category, we keep a voxel grid V_i with dimensions X, Y, Z of size $X \times Y \times Z$ by voxelizing pointclouds in the reference frame. The confident detections of all instances from the i -th category are unprojected, transformed to the reference frame, voxelized, and added into V_i . To segment a particular instance for class i , we initialize the sets P_{bg} the same way as before, but initialize P_{fg} from the largest connected component in V_i that overlaps with the unprojection of that instance detection. Then, the same two-stage segmentation method is used.

We compare the pseudo-label quality of the method in our main paper with this multi-view aggregation variant on the CARLA training set, and report the results in Table 7. The results suggest that pseudo-labels from multi-view detection aggregation significantly outperform the detections of the pre-trained detector, but underperforms our original method. Furthermore, the performance drop for mAP@0.5 is larger than the drop for mAP@0.3. We conjecture that this is due to two reasons: (1) to effectively cast votes from multi-view detections, we voxelize the points after they have been transformed to the reference frame, losing detailed information about the point locations; (2) our two-stage segmentation method can already perform robust segmentation of the full 3D object from a small subset of P_{fg} and P_{bg} (from a single detection).

Method	mAP@0.5	mAP@0.3
Pre-trained	68.05	73.09
SbM Labels w/ view agg.	80.29	89.26
SbM Labels original	86.88	92.93

Table 7: **Pseudo-label accuracy with multi-view aggregation** We show that multi-view detection aggregation significantly improves over the baseline pre-trained detector, but underperforms our original method.

3 Additional Visualizations

3.1 Qualitative Results for 3D Detection

As mentioned in the main paper, we show qualitative results of the 3D detections from LDLS [10] and our SbM fine-tuned frustum PointNet in Figure 2. This demonstrates that the 3D segmentation labels produced by SbM are high quality and could be successfully used to train state of the art 3D detection models without ground truth 3D annotations.



Figure 2: **Visualizations of 3D object detection.** We show paired examples of the results of LDLS [1] (left) and SbM-trained frustum PointNet (ours, right) on the CARLA test set. While LDLS estimates the bounding box roughly, the SbM-trained frustum PointNet is able to obtain much tighter and better-oriented boxes.

3.2 Visually Unmatched Categories

As we described in Section 4.3 of the paper, categories that overlap between COCO and Replica are sometimes visually (and even semantically) very different, making it hard to obtain high confidence detections of certain objects. In Figure 3, we show that the tables in COCO and Replica differ significantly in semantic meaning and appearance. In the left image (from Replica), a detector pre-trained on COCO identifies a “dining table” with a confidence score of 86%, while it is not labelled as a table in the Replica annotations. In the right image, the bounding boxes show two tables and a couch annotated in Replica. We observe that these annotated tables are visually very different from the “dining table” class in COCO.

3.3 Pseudo-label Visualizations

We show visualizations of the 2D pseudo-label masks re-projected from the 3D segmentation for a variety of classes in the Replica dataset in Figure 4. We can see that the segmentation is complete with sharp borders between foreground and background. In addition, we show qualitative segmentation results in the included video.

3.4 Weakly-supervised Novel Object Detection

In Figure 5, we show visualizations of the 2D detector performance fine-tuned on SbM-ws pseudo-labels. The pseudo-labels are generated with weak supervision (ground truth on 1 view per episode) on novel categories (corresponding to Section 4.5 of the main paper). The detector detects the objects under diverse viewpoints.

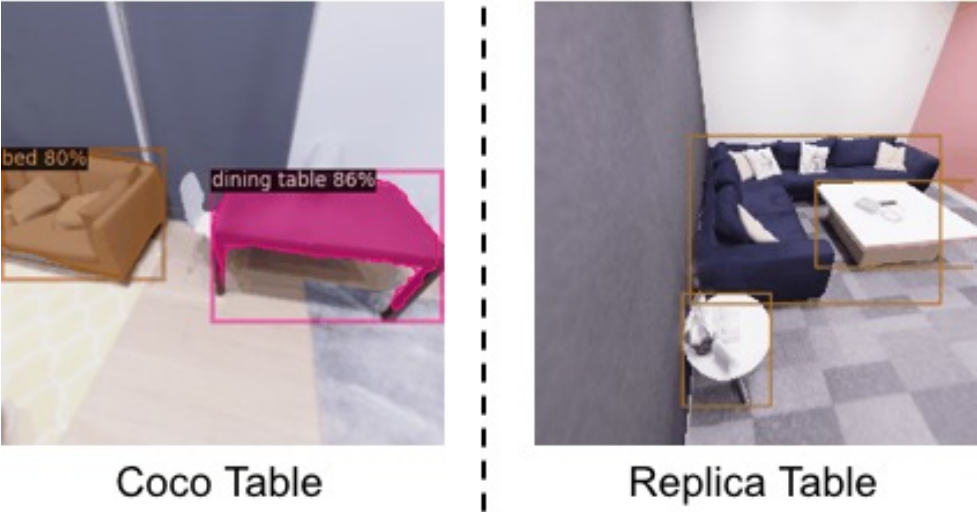


Figure 3: **Semantically and visually different tables in COCO and Replica.** We show a table (predicted as “dining table” by the pre-trained MaskRCNN) on the left, and two actual tables in Replica dataset on the right.

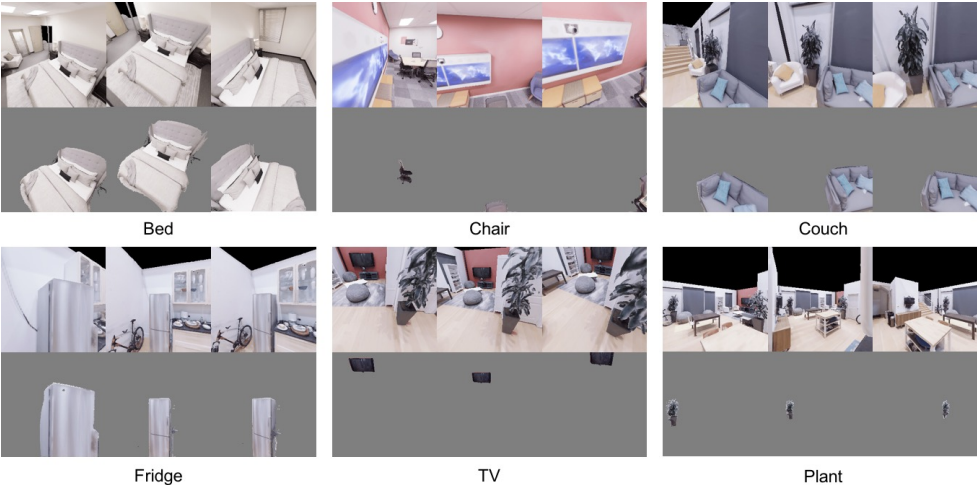


Figure 4: **Example 2D pseudo-labels reprojected from 3D segmentation.** We show examples of the rejections of 3D segmentation (which are used as 2D pseudo-labels) on a variety of objects in the Replica dataset.

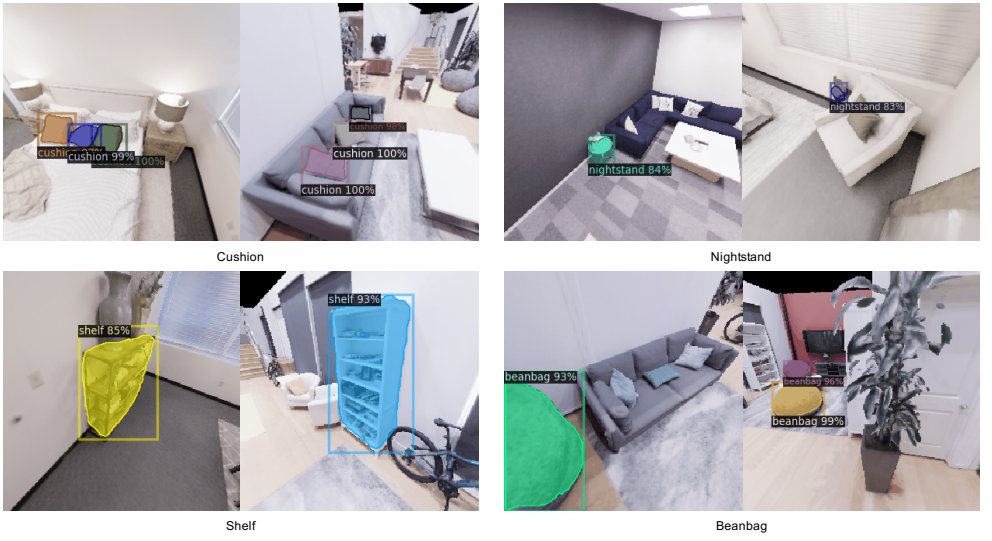


Figure 5: **Visualizations of the detection results of the detector trained with SbM-ws pseudo-labels** From visualizations of detection results on the test set, we can see that the detector supervised by SbM-ws pseudo-labels generates robust predictions.

4 Implementation Details

4.1 2D-to-3D unprojection

For the i -th view, a 2D pixel coordinate (u, v) with depth z is unprojected and transformed to its coordinate $(X, Y, Z)^T$ in the reference frame: $(X, Y, Z, 1) = \mathbf{G}_i^{-1} \left(z \frac{u - c_x}{f_x}, z \frac{v - c_y}{f_y}, z, 1 \right)^T$ where (f_x, f_y) and (c_x, c_y) are the focal lengths and center of the camera model and $\mathbf{G}_i \in SE(3)$ is the camera pose for view i relative to the reference view.

4.2 2D Object Detection

For the pre-trained 2D detector, we use a Mask-RCNN [6] with FPN [9] using ResNet-50 as the backbone, pre-trained on the COCO dataset. We fine-tune it on the 2D pseudo-labels from the training set for 100k iterations. To compare, we also fine-tune the detector on the same images but with ground truth labels. In both settings, we use a learning rate of 0.001 and a batch size of 12. For selecting the best model, we compute its mAP on a validation set at IoU threshold of 50% every 5000 iterations. We use the Mask-RCNN implementation from Detectron2 [8], keeping all other hyperparameters as default.

4.3 3D Object Detection

We use the frustum PointNet model [1] with PointNet [5] backbone on CARLA. The original frustum PointNet model uses ground truth 2D bounding boxes and camera pose to define a 3D frustum search space and then performs 3D segmentation on it using a PointNet-based

architecture, and uses ground truth 3D boxes for supervision. In our experiment, we use the 2D and 3D bounding boxes generated from our self-supervised 3D segmentation. To compare, we also train the same network using ground truth 2D and 3D bounding boxes. For both settings, we train it using a learning rate of 0.001 and a batch size of 32 until convergence. For selecting the best model, we use the validation loss curve. We test both models on a new unseen town and compare their performance.

4.4 Data Collection Details

For CARLA, we use a radius of 3.0-15.0 meters from the target object 3D centroid for sampling goal locations for navigation. For Replica, we use a radius of 0.5-3.0 meters. We estimate the 3D centroid of the target detection by unprojecting the median depth value of the masked depth image. We obtain 25 views in each episode (sample 25 goal locations).

References

- [1] Devendra Singh Chaplot, Dhiraj Gandhi, Saurabh Gupta, Abhinav Gupta, and Ruslan Salakhutdinov. Learning to explore using active neural slam. In *ICLR*, 2020.
- [2] Will Grathwohl, Kuan-Chieh Wang, Jörn-Henrik Jacobsen, David Duvenaud, Mohammad Norouzi, and Kevin Swersky. Your classifier is secretly an energy based model and you should treat it like one. In *ICLR*, 2020.
- [3] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, 2017.
- [4] Tsung-Yi Lin, Piotr Dollar, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *CVPR*, 2017.
- [5] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. *CVPR*, 2017.
- [6] Charles R. Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J Guibas. Frustum pointnets for 3d object detection from rgb-d data. In *CVPR*, 2018.
- [7] Brian H. Wang, Wei-Lun Chao, Yan Wang, Bharath Hariharan, Kilian Q. Weinberger, and Mark Campbell. Ldls: 3-d object segmentation through label diffusion from 2-d images. *IEEE Robotics and Automation Letters*, 4(3):2902–2909, July 2019. doi: 10.1109/LRA.2019.2922582.
- [8] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. <https://github.com/facebookresearch/detectron2>, 2019.