Active Embodied Vision - Towards Self-Supervised Never Ending Learners

UNDERGRADUATE THESIS

Submitted in partial fulfillment of the requirements of
BITS F421T Thesis

By

Ayush JAIN
ID No. 2017A7TS0093P

Under the supervision of:

Prof. Katerina FRAGKIADAKI
&
Prof. Pratik NARANG

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE PILANI, PILANI CAMPUS
December 2020

Declaration of Authorship

I, Ayush JAIN, declare that this Undergraduate Thesis titled, 'Active Embodied Vision - Towards Self-Supervised Never Ending Learners' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: _____

Date: _____

Certificate

This is to certify that the thesis entitled, "*Active Embodied Vision - Towards Self-Supervised Never Ending Learners*" and submitted by Ayush JAIN ID No. 2017A7TS0093P in partial fulfillment of the requirements of BITS F421T Thesis embodies the work done by him under my supervision.

Supervisor
Prof. Katerina FRAGKIADAKI
Asst. Professor,
Carnegie Mellon University
Date:

Co-Supervisor
Prof. Pratik NARANG
Asst. Professor,
BITS-Pilani Pilani Campus
Date:

"There's three things you can do. You can just give up and quit, you can stick your head in the sand, or you can dig up and solve the problem."

Ayush Jain

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE PILANI, PILANI CAMPUS

Abstract

Bachelor of Engineering (Hons.) Computer Science

Active Embodied Vision - Towards Self-Supervised Never Ending Learners

by Ayush JAIN

Humans learn to better understand the world by moving around their environment to get more informative viewpoints of the scene. Most methods for 2D visual recognition tasks such as object detection and segmentation treat images of the same scene as individual samples and do not exploit object permanence in multiple views. Generalization to novel scenes and views thus requires additional training with lots of human annotations. In this paper, we propose a self-supervised framework to improve an object detector in unseen scenarios by moving an agent around in a 3D environment and aggregating multi-view RGB-D information. We unproject confident 2D object detections from the pre-trained detector and perform unsupervised 3D segmentation on the point cloud. The segmented 3D objects are then re-projected to all other views to obtain pseudo-labels for fine-tuning. Experiments on both indoor and outdoor datasets show that (1) our framework performs high quality 3D segmentation from raw RGB-D data and a pre-trained 2D detector; (2) fine-tuning with self supervision improves the 2D detector significantly where an unseen RGB image is given as input at test time; (3) training a 3D detector with self supervision outperforms a comparable self-supervised method by a large margin.

Acknowledgements

I would like to thank Prof. Katerina Fragkiadaki for giving me an opportunity to work at the Machine Learning Department of Carnegie Mellon University. Prof. Katerina has been a great mentor and guide throughout my thesis. She has always been very responsive and approachable, with frequent project discussions. Her ideas and leadership throughout the project played a great role in shaping this research work as well as my research interests. I am also grateful to Adam Harley (PhD. Student, CMU) for inviting me to work on my undergraduate thesis. I have benefited immensely from their continuous guidance and illuminating discussions. Adam has been a great support throughout my thesis by being a friendly, compassionate and approachable person. I am immensely grateful to him for being a person I can be comfortable being vulnerable with, and being a source of light every-time I hit a roadblock. He is a role-model for me on how to be a leader and a good person. I would not have succeeded in completing this thesis successfully without his mentorship.

This thesis would not have been possible without the awesome collaborators I had the opportunity to work with: Gabriel Sarch (Ph.D. Student at CMU), and Zhaoyuan Andy Fang (M.S.R Student at CMU). We had a lot of project-related discussions as well as fun during my thesis. I got to learn a lot from them. I am also grateful to Fish Tung (Ph.D Student at CMU), who proof-read my thesis and provided extremely helpful suggestions to improve it. I am also thankful to Mihir Prabhudesai (Research Assistant, CMU) and Shamit Lal (MSCV Student, CMU) for mentoring me during the initial part of my thesis. I am also grateful to Prof. Christopher Atketson (Professor, CMU) for the insightful discussions during the lab meetings.

I would also like to express my gratitude towards the Department of Computer Science and Information Systems at my university, Birla Institute of Technology and Science (BITS) Pilani, for allowing me to conduct computer vision research and gain international exposure by pursuing my thesis at CMU. In particular, I would like to extend my special thanks to Prof. Pratik Narang for his support as the co-supervisor of this research endeavour. I would also like to thank Sanwar Lal for his administrative support.

I am also thankful to Laura D. Winter for her administrative support, and to SCS Help Staff for offering technical assistance.

Lastly, I thank my family for their unwavering support.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Human beings live in a 3D world. We perceive 3D objects all around us and learn to reason about them. For example, we might say that a pillow lies on top of the bed. We can imagine what would happen if we take away that pillow from the bed. Even our 2-year-old babies are reasoning superstars when we compare them with current AI systems.

1.2 Current Drawbacks

Computer vision has made a lot of advancement in the last decade with awesome utilities like automatic focusing in cameras, face unlock and much more. Despite their huge success, they suffer from a few major drawbacks:

- **Data Hungry**: Most of the object detectors are trained on millions of labelled images, which are costly to procure. Furthermore, it is seen that the models fail to generalise to objects not represented well in the training data. This type of setting is fairly popular in the computer vision community and is known as supervised learning. As expected, there is something called unsupervised or semi-supervised learning, where we make use of no or very few labelled training samples for learning the task. This setting is an extremely crucial step towards advancing this field further as we cannot always procure large datasets which can represent every object present in the universe. We want our systems to learn on relatively cheap and abundant unlabelled data without compromising much on the accuracy and precision for completing the desired task.
- **Inability to handle occlusions**: Imagine a robot moving around a set of your favourite lego blocks. While moving, it observes let say 2 blocks. After moving to a slightly different location, one block

gets hidden/occluded when seen from this new perspective. Through common sense, a child can reason that the block is still there, behind the other block. But many AI systems fail miserably. This problem can be referred to as "lack of object permanence" as the occluded objects are not represented well in the feature space of most Computer Vision architectures.

- **Entangled Representations**: The scene and camera motion is highly entangled - objects move with camera motion, their sizes change when camera zooms in or zooms out. This entangled representation is difficult to work with using traditional 2D CNN pipeline.

- **Embodied Camera Vision**: Traditional CNN architectures have shown to work really well with datasets like Imagenet which is composed of photographs takes by expert photographer. This is in contrast to embodied camera setting common in robotics, where the world is perceived through a camera mounted on top of a robot. The captured images are shaky and marked with lots of occlusion.

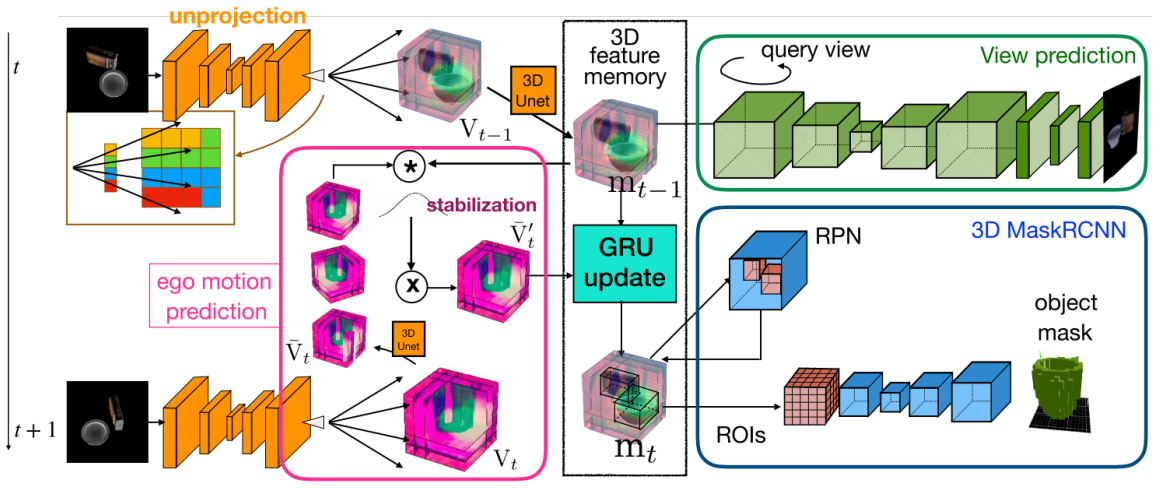

FIGURE 1.1: **GRNN Architecture**: At each time step t, an RGB image I is the input to a 2D U-net [52]. The resulting 2D deep feature maps are unprojected to 4D tensors Vt, which in turn are input to a 3D U-net [52]. The resulting 3D deep feature maps V are oriented to cancel the relative camera motion between the current viewpoint and the unprojected to 4D tensors Vt, which in turn are input to a 3D U-net [52]. The resulting 3D deep feature maps V are oriented to cancel the relative camera motion between the current viewpoint and the coordinate system of the 3D GRU memory state mt-1, as estimated by an egomotion estimation module. The resulting oriented 3D deep feature maps Vt update the 3D GRU memory state and output mt. The updated state of the GRU module is then projected from specific coordinate system of the 3D GRU memory state mt-1, as estimated by an egomotion estimation module. The resulting oriented 3D deep feature maps Vt update the 3D GRU memory state and output mt. The updated state of the GRU module is then projected from specific viewpoints and decoded into a corresponding RGB image for view prediction, or fed into a 3D MaskRCNN [71] to predict 3D object bounding boxes and object voxel occupancies.

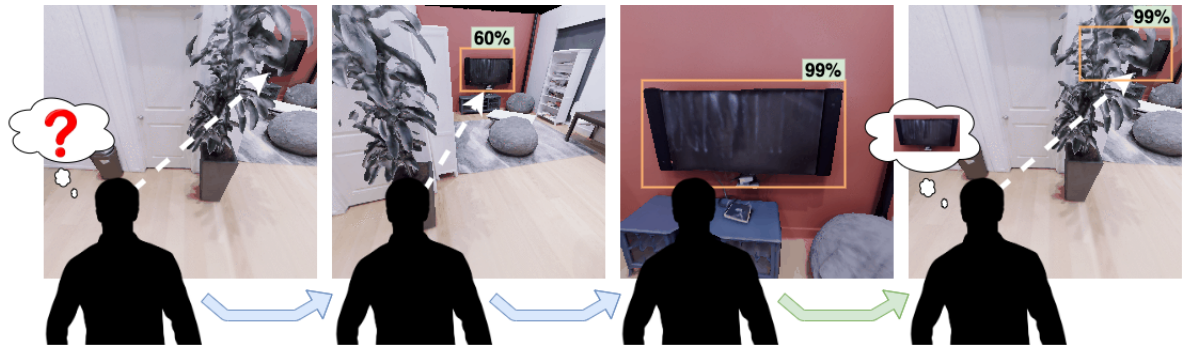

FIGURE 1.2: **Improving object recognition by moving.** An agent is viewing an object from an occluded, unfamiliar viewpoint. By moving to less occluded, more familiar viewpoints of the object (blue arrow), the agent can use the familiar viewpoints to self-supervise the previously unfamiliar viewpoints (green arrow). Subsequently, perception of objects in the previously unfamiliar views improves.

1.3 High Level Description

For tasks that require high-level reasoning, intelligent systems must be able to recognize objects despite partial occlusions or uncommon poses. The ability to perceive both the visible and the occluded regions of the environment, known as amodal perception, is especially important for understanding fundamental relationships in the scene such as depth ordering and object permanence [42]. Humans and other mammals actively move their eyes, head, and body in order to obtain less occluded and more familiar viewpoints of the objects of interest [16, 58]. Thus, in a self-supervised manner, animals are able to inform viewpoints they are less confident about by moving to better viewpoints. Amodal perception remains challenging for state-of-the-art deep learning methods, which typically build complex models to hallucinate 3D instances from 2D labels [17, 31, 74]. Do we need to hallucinate what's behind a mug, if we can simply lean over and see around it?

Imagine a scenario where one is attempting to determine the extent or identity of an occluded object from an unfamiliar viewpoint, such as in Figure 1.2. One can increase their certainty by simply moving to a less-occluded and more-familiar viewpoint. Then, by mapping these confident beliefs of the object back to the unfamiliar views, perception of the object from the previously unfamiliar views will improve over time and experience. In a similar manner, intelligent machine vision recognition systems can exploit simple movements and self-supervised learning to improve scene understanding, and consequently improve their performance on 2D and 3D recognition tasks.

Significant improvements have been made in the accuracy and reliability of 2D [21, 34, 36, 49, 51, 4, 12, 25, 37, 70] and 3D [32, 48, 47, 46, 63, 73] visual recognition systems. Methods trained on large static image datasets perform well in the domains they are trained on, but require large amounts of human annotation for training, and have difficulty generalizing well to novel contexts and viewpoints [7]. Recent advances in active visual learning[10, 11, 67] have focused on efficient data collection techniques, so that the detector adapts to new scenes and views after fine-tuning on the collected data. However,

these approaches require ground truth 3D segmentation of the environment or 2D human annotations of the images to train, making such methods expensive. This dependence on human annotation also makes it difficult to scale to a continual learning setting where the agent can extract knowledge from the unstructured world and adapt to new information [40, 13, 8].

1.4 Contribution

In this work, we propose a fully self-supervised method for obtaining labels to train a network to perform amodal 2D and 3D object detection and segmentation. We assume an embodied agent that moves around in its environment until it finds a high confidence detection using a pre-trained object detector. It then moves around the detected object and collects diverse posed RGB-D images. We then use the high confidence object detections to segment them in 3D space and propagate the segmentations to 2D to use as labels in all other views. Modern depth sensors, such as Lidar and stereo cameras, and pose estimation methods, such as simultaneous localization and mapping (SLAM), allow robots to represent the 3D world as a point cloud with very detailed precision [10, 55]. Thus, our label generation method is applicable to a real-world setting, and would allow the robot to obtain labels for adapting a detector to a target environment with little to no human supervision. While 2D human annotations are expensive to obtain, 3D segmentations is even more difficult to obtain in a real-world setting. Our method provides a robust way for obtaining 3D detection and segmentation labels unsupervised.

Our key contribution is a self-supervised 2D and 3D pseudo-label generation method for training an object detection and segmentation network without any human annotations needed in the entire process. On indoor and outdoor datasets, we show that fine-tuning Mask-RCNN with self-supervised labels generated by our method significantly improves the Mean Average Precision (mAP) over the pre-trained detector. When we apply our method with weak supervision, we are able to further increase performance. Additionally, we show that our self-supervised 3D detection method outperforms state-of-the-art self-supervised 3D detection methods while achieving performance comparable to fully supervised methods. We finally demonstrate that our method can iteratively improve with self-supervision, which fits our model into a continual visual learning framework, whereby the system would be able to perpetually learn to reason about the world without human intervention. We will make our code and data publicly available.

Chapter 2

Background

2.1 GRNN Architecture

Vanilla Convolutional Neural Network are well suited for dealing with translation variance. But they fail miserably, when the viewpoint of the scene is changed. This is an important disadvantage of using CNN in embodied agents where the viewpoints keeps on changing continuously. To tackle this problem, GRNN proposes specific view-training self-supervised tasks to make the features view invariant. Due to the self-supervision paradigm, the proposed model is sample efficient and easily generalisable.

In essence, the GRNN takes a scene as input at a particular time t. Then a vanilla 2D U-Net [52] extracts features from the given scene. These features are then un-projected in a reverse light rasterisation technique to 3D voxel-grid representations. The obtained 3D representations are then egomotion stabilised and passed through a 3D GRU update, to obtain a 3D feature memory. For self-supervision training, the 3D latent memory is projected to 2D RGB image from a different viewpoint. This predicted 2D RGB image is compared with ground truth 2D rotated RGB image, and the loss is backpropagated.

Detailed Description of each module and step of the GRNN model is discussed below.

Input to the model: The model takes as input a series of RGB-D image frames.

2.1.1 Important Components of the pipeline:

Unprojection Module: The 2D RGB images is passed through a pretrained 2D U-Net [52] model to extract the features from the image. From the given features, 3D voxel grid is predicted which is a 4D tensor. The voxel grid is predicted using the camera intrinsics which are assumed to be known. For every point in 3D, the corresponding point in 2D is calculated using focal length of the camera and the x, y, z coordinate of the 3D point. From the obtained x, y point in 2D, a light ray is assumed to be emanating

towards the 3D voxel grid. The voxels which the ray intersects gets filled by bi-linearly interpolated pixel value. In this way all the voxels are filled with values.

Using the depth values, assumed as input, a 3D binary depth feature map is constructed. The value of a voxel is 1, if the grid depth is equal to the input depth for that pixel. Otherwise the value is assigned 0. This depth occupancy depth, intuitively depicts the part of the object which is directly visible from the viewpoint.

The unprojected 3D voxel grid from the RGB image is multiplied by the 3D depth voxel grid, to obtain 3D representation for visible 3D region. The obtained 3D voxel grid is passed through a 3D version of U-Net [52] to obtain dense 3D features.

Egomotion and Stabilisation: Egomotion refers to the movement of the camera from one viewpoint to another. It is characterised by a rotational matrix and translation matrix. The paper assumes that the egomotion is available as input. This seems to be a reasonable assumption because egomotion sensors are widely available for mobile robots.

The model further assumes that the camera only rotates i.e. there is no translation movement. This is somewhat strong assumption because the robotic agents are generally locomotive and hence the translation is a key movement for such embodied agents. The rotation is composed of two angles: azimuth and elevation. Azimuth is basically rotation within the plane of the camera, parallel to the observed object. Elevation is the angle between normal to the plane parallel to observed scene and the camera plane.

Since the model aims to a generate a consistent latent memory, the relative elevation and azimuth needs to be estimated with respect to the first camera view. For enabling this, the model concatenates 3D features from different rotations and azimuths, sampled uniformly. The voxel grids for these different rotations and azimuths is predicted similar to the previous technique of bilinear interpolation. Using matrix inner product between the latent feature memory and the obtained feature tensor, probability distribution over different elevations and azimuths is predicted. Finally a weight sum by probability of the rotation and azimuths is used to stabilise the egomotion. Hence a feature vector parallel to the latent feature memory is predicted.

Memory Update: The egomotion stabilised feature tensors are passed through a 3D GRU update unit to update the latent memory space. Alternative techniques like averaging and LSTM could also have been experimented with.

Projection and Decoding: After latent memory is built, a new viewpoint is sampled. The latent memory is aligned to the new viewpoint using the transformation matrices. It is not trivial to project to 2D, because the model has no way to directly obtain depth from the new viewpoint. Hence, the approach proposed the used of ”d” depths sampled uniformly from center of the latent memory. Corresponding to each depth, the 2D x, y value is filled by bilinear interpolation from the 3D voxels. All these d features are concatenated

together and passed through 2D convolutions followed by the use of convLSTM decoder to map it to RGB image. The actual depth is implicitly learned by the model through training.

View Prediction Training: As the name suggests, view prediction is a self-supervised training approach i.e. it does not require use of explicit provided ground truth labels. This approach uses the latent memory and the projection and decoding procedure discussed above to generate a 2D RGB image from a new view point. A pixel wise cross entropy loss is used for training the self-supervised objective.

3D Object Detection and Segmentation: Finally, the models performs 3D Object detection and Segmentation on the 3D latent memory formed through view prediction. The proposed object detection and segmentation is supervised. The authors assume access to the 3D Bounding boxes and Segmentation Masks are provided by simulator. This is a weak assumption because the method won't have access to 3D bounding boxes and segmentations in real-world scenarios, where the access to simulator won't be there and it is common knowledge that procuring 3D bounding boxes and segmentations is tedious task. A better approach could have been to triangulate the 2D ground truth bounding boxes to 3D boxes which are vastly available in datasets like Imagenet, COCO and many others.

For every time step, the 3D latent memory is fed into 3D region proposal network which predicts the center, and other box dimensions. Further, Non-Max Suppression is applied for filtering duplicate bounding boxes. The IOU with ground truth boxes is computed for the remaining predicted boxes and the boxes with IOU above a threshhold are further used to predict bounding box adjustments as well as occupancy masks for segmentation.

2.2 Limitations of GRNN

The GRNN models makes a few assumptions which are weak and hence hampers the scalability ans applicability of the model to the real-world applications. One of the most profound limitation of GRNN is the assumption that camera motion is only rotational and not translational. Clearly, this is not an acceptable model for mobile robots. Further, restricting the translation motion of the camera restricts the models capability for dealing with occlusions. If translation is allowed, then the camera would be able to different occluded surfaces from different viewpoints. If the informations from all the viewpoints are combined, it can help in generating the 3D memory of object much more accurately.

2.3 C-N3D Architecture

This section will discuss Contrastive Neural 3D mapping (C-N3D) architecture by [23]. The goal of this model is same as GRNN, building 3D representations for common sense reasoning. It employs the use of view prediction synthesis similar to GRNN as a self-supervised task for learning view invariant

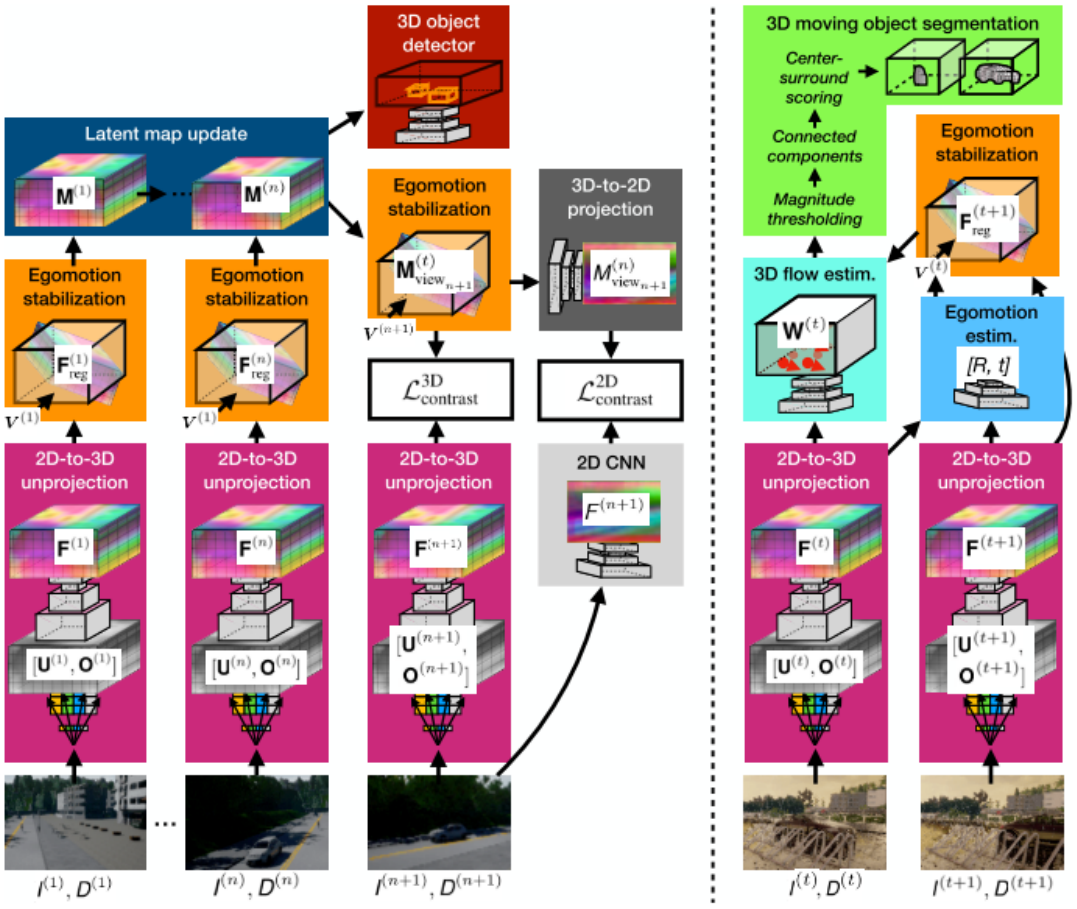

FIGURE 2.1: **C-N3D Architecture**: Contrastive predictive neural 3D mapping. Left: Learning visual feature representations by moving in static scenes. The neural 3D mapper learns to lift 2.5D video streams to egomotion-stabilized 3D feature maps of the scene by optimizing for view-contrastive prediction. Right: Learning to segment 3D moving objects by watching them move. Non-zero 3D motion in the egomotion-stabilized 3D feature space reveals independently moving objects and their 3D extent, without any human annotations.

3D representations. Though the basic idea of C-N3D is similar to GRNN, the specific approach is quite different.

The proposed model taken in input 2.5D video streams as input, and unprojects them into 3D representations. The camera is assumed to have 6 degree of freedoms, which means it can perform both rotation and translation. Further the objects in the scene can move too. The proposed model can perform self-supervised 3D object detectors and unsupervised learning of 3D moving object detectors.

Detailed Description of each module and step of the C-N3D model is discussed below:

Input to the model: The model assumes access to the 2.5D input video stream and the 3D depth map. Further, the model have access to its pose at train time. Using these inputs, the models learns to imagine the 3D representation of the scene and the corresponding egomotion.

2.3.1 Neural 3D Mapping

Unprojection: The unprojection module is similar to GRNN discussed above. It takes 2D RGB-D image as input, reconstructs 3D voxel grid using the camera intrinsics and bilinear interpolation and finally pass it through a 3D U-NET [52] feature extractor for obtaining a 3D representation of the scene.

Egomotion Estimation: The egomotion estimation pipeline of C-N3D is entirely different from GRNN mainly because the camera has more degrees of freedom including translation which the relatively simpler pipeline of GRNN egomotion estimation could not handle. They use a 3D coarse-to-fine alignment search, where they first estimate transformations at coarse scale and then at fine scale which is similar to PWC-Net [57]. The training is performed in a supervised manner where they assume access to annotated egomotion between different frames.

Latent Map Update: Unlike the use of GRU in GRNN, they simply take a running average of egomotion stabilized feature tensors for obtaining a view-invariant 3D feature memory. Since the translation motion is allowed, this model is far more suited for handling occlusion as it can have access to different frames, combining which the entire object can be imagined.

Projection: This module projects the 3D latent feature memory to 2D given a viewpoint. This architecture is similar in all sense to GRNN, but unlike GRNN they project to 2D features map instead of 2D RGB image. This is a nice change because its easier for neural networks to argue if two images are similar in feature space, than though cross pixxel RGB comparison.

3D Object Detection: The model uses a 3D adaptation of Faster-RCNN model for performing 3D object detection on latent memory. This module is also supervised and requires ground truth annotated 3D object boxes.

2.3.2 Contrastive Predictive Training

Input to the module: RGB images, point clouds, camera poses

Output: Feature projection of unseen input

They employ two kinds of representations:

1. **Top Down:** Here they assume that they have access to camera pose for the unseen image, and it needs to find the corresponding 3D and 2D representations from that viewpoint. They do this through the Neural 3D mapping procedure discussed above.

2. **Bottom Up:** Here they assume that they have access to RGB image and Point clouds in the new viewpoint, but not the viewpoint itself. From this information they need to find corresponding to 2D and 3D representations, which is just a simple feature extractor

Contrastive Training: Intuitively we know that the 2D and 3D representations from top down and bottom down approach should be close together. This means that the contrastive loss for between both the representations should be close to 0. Here they employ a small threshhold alpha for cutting off the contrastive loss. Through this training, the representations become view invariant and 3D meaningful.

2.3.3 Unsupervised 3D moving object detection

For performing the moving object detection, they perform clustering of 3D estimation flow vectors. This approach is in contrast with GRNN as the GRNN is unable to work for moving camera or objects. The module maps 2D frames to egomotion stabilised feature maps. Since the features have been accounted for camera motion, the only motion which needs to be encountered is that of moving objects. We will discuss the corresponding steps below:

Estimating 3D Imagination Flow: This module is a 3D version of the popular PWC-NET optical flow model [57]. This module is trained entirely through self-supervised learning.

1. **Synthetic Rigid Transformations:** Here, the authors employ random rotations and translations to the scene, and the model needs to recover the corresponding 3D optical field. This is a clever approach which do not require any external supervision as the rotations and translations are known by design.

2. **Unsupervised 3D temporal feature matching:** After obtaining the optical flow from above, the module takes in the features of the next time step and using the estimated flow, tries to warp it back to features at the time step before. The reconstruction loss is minimized as optimization task for training the model.

2.4 Limitations of N-C3D

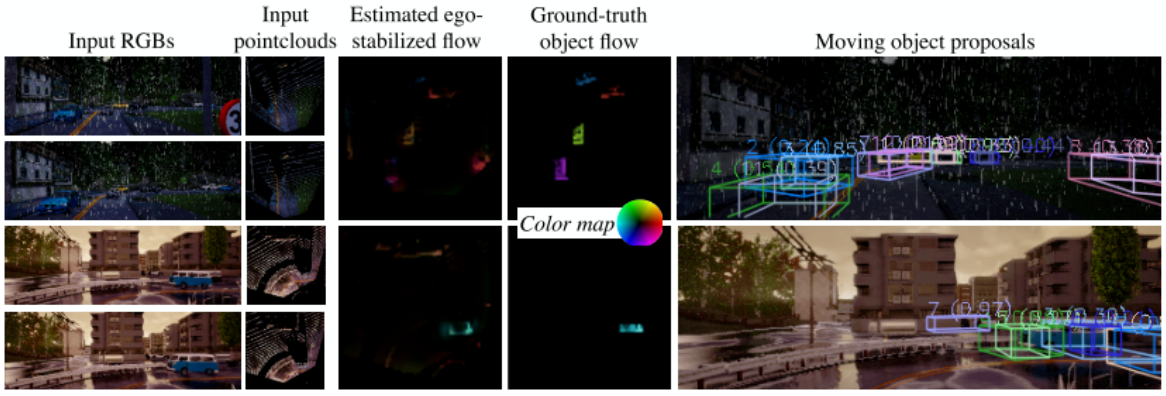

FIGURE 2.2: **3D feature flow and object proposals, in dynamic scenes.** Given the input frames on the left, our model estimates dense egomotion-stabilized 3D flow fields, and converts these into object proposals. We visualize colorized pointclouds and flow fields in a top-down (bird's eye) view.

While the model works with fewer assumptions about the world scene, it employs complex and multiple supervision tasks which are difficult and less stable to train simultaneously. Further since the ego-motion module is supervised, hence it requires ground truth egomotions which are difficult to obtain. Further, the 3D voxel representations are highly memory intensive, hence they need to work with very low 3D resolution regime. Hence the model losses a lot of important information which makes working with complex tasks much more difficult.

2.5 Comparative Study

In this section, we will compare the GRNN and N-C3D models.

2.5.1 Similarities

1. **Supervision:** Both GRNN and C-N3D tries to learn 3D representation of the surrounding scenes using as few annotations as possible.

2. **3D representations:** The main intuition behind both these models is this that the 3D representations are much more realistic and helpful in downstream common sense tasks like affordability reasoning : It's easier to infer whether a ball can fir into a cup or not in 3D, than in 2D. Also the 3D representations do not suffer from occlusion which is a major challenge in 2D world.

2.5.2 Differences

1. **Use-cases:** The GRNN model is a very restricted model, where the camera can only rotate and the objects in the scene do not move. In contrast C-N3D model have 6-DOF movement of camera as well as the objects in the scene can move too. This is a more realistic and real-world scenario as compared to assumptions of GRNN. 2. **Decoder**: The N-C3D model decodes the 3D representation to 2D features instead of 2D RGB image which GRNN does. This is better approach because it is easier to establish similarity at feature level than at pixxel level. 3. **Model complexity**: The GRNN model is relatively simpler than N-C3D model. N-C3D model is a lot less stable to train than the GRNN model.

Hence to conclude, if the assumptions of GRNN make sense in the environment one choses to work in, go with GRNN as it would be lot less cumbersome to train. But if the model needs to generalise to real-world scenario and you have the required ground truth annotations, go for N-C3D.

2.6 Results

In this section, we will discuss some of the main results of both these networks. There are lot of illustrations for the results, we direct the interested reader to the respective papers ([59], [23]).

1. **View Prediction**: The GRNN model is able to perform view prediction much more effectively than the previous models by Neural-Renderer[18]. Both the models are trained on two objects in the scene. When tested on 4 object scene, GRNN is able to generalise easily, while the Neural-Renderer model fails to see more than 2 objects. This shows the generalisability capability of the model.

2. **Scene Arithmetics**: The GRNN model is able to perform common sense arithmetics like scene A - B + C, which means removing objects occurring in B from A and adding objects in C. The [18] baseline fails to perform this task comprehensively,

3. **3D Object detection and segmentation**: The authors compare their model against the 2D object detection models. Their 3D model outperforms the 2D variant by significant margin.

4. **Unsupervised 3D moving object detection**: The visual results for C-N3D model is shown in Figure-3. C-N3D model is able to achieve excellent results on moving object detection in an unsupervised way when compared to completely supervised approaches.

An interesting application of GRNN approach is the work by [45]. They use language together with GRNN module to perform many interesting common-sense reasoning tasks.

In essence they map scenes to 3D representations using the GRNN module. Using this representation and parse-tree of instructions, they can perform following interesting tasks:

1. **Affordability Reasoning**: The model can argue which configurations are affordable and which are not. For example, ”object A to the left of B, B to the left of C, C to the left of A” is not plausible. The model can infer that using its 3D representations. Given the parse tree, the model first generates the objects, for instance A, B and C and tries to put these objects as prescribed by the statement. If there is no possible way to arrange that, the model can infer that the situation itself is not affordable.

2. **Referential Expression Detection**: Given the instruction, ”the blue sphere behind yellow cube’, the model is able to correctly identify and segment the blue sphere, referred to by the instruction.

3. **Instruction Following**: Given an object placement instruction like ”place the cup behind the bowl”, it is able to follow the instruction. First it generates the representations of cup and bowl using a generative model. Then, in its 3D representation, the model predicts the relative position of the concerned objects.

Chapter 3

Related Work

3.1 2D Object Recognition

2D object recognition has been one of the most dominant tasks in the field of Computer Vision. With the surge of deep learning, researchers collected large scale datasets to ground the development of methods [30, 35, 72]. Deep networks now achieve extraordinary performance on visual recognition tasks, including object detection [21, 34, 36, 49, 51] and semantic segmentation [4, 12, 25, 37, 70]. However, a recent study [7] showed that state-of-the-art detectors are less likely to recognize an object under unique viewpoints correctly by testing them on a new manually generated dataset of uncommon views. In this work, we aim to improve a pre-trained Mask-RCNN [65] in new environments and viewpoints in a fully self-supervised way.

3.2 3D Object Recognition

3D object recognition has also been explored in various forms. Some methods quantize pointclouds into 3D voxel grids [50, 73] to get structured data, but voxel-based methods become expensive as the resolution increases. PointNet [48, 47, 46] is an architecture that directly operates on unordered pointclouds for learning deep point set features applicable to object detection and segmentation. SPGN [63] extends the direct consumption of pointcloud to instance segmentation. Later works [32, 62] integrate convolution into pointclouds. All of these methods require 3D annotations. Wang et al. [61] proposed a semi-supervised method named LDLS, which performs 3D segmentation by diffusing pre-trained detectors' predictiosn from single view RGB-D images by building graph connections between 2D pixels and 3D points without requiring 3D ground truth information. In this work, we show that from our pseudo-labels we can train a 3D object detector [46] which outperforms LDLS and achieves performance comparable to fully supervised methods.

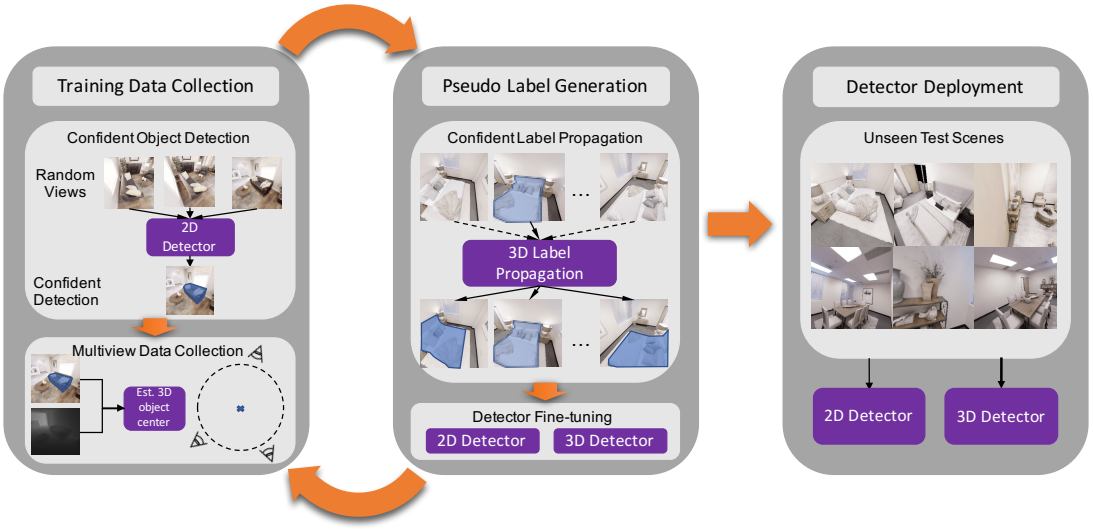

FIGURE 3.1: **Seeing by Moving (SbM).** We use confident detections of a pre-trained 2D object detector to guide self-supervised multi-view data collection and pseudo-label generation. Our 3D segmentation module uses confident detections in some views to label the object in other views, generating pseudo-labels automatically. The 2D detector fine-tuned on the pseudo-labels performs better on unseen test scenes.

3.3 Active Visual Learning

The problem of active vision [2, 5, 54] presents an agent with a large unlabelled set of images and asks the agent to select a subset for labelling, which will provide the maximal amount of information about the full dataset [53]. Psychology research also confirms active vision as a natural method used by humans to attend to relevant visual features [6, 16, 58, 69]. This has been applied to several fields including object detection [26, 27, 60, 68], instance segmentation [44], feature learning [1], and medical image analysis [29]. Another related setting is explored by Chaplot et al. [11] where instead of selecting images to label from pre-collected data, a policy is trained for efficiently acquiring unlabeled data. In this work, we propose a self-supervised technique complementary to both directions, where we select "easy" viewpoints according to the confidence of a pre-trained detector, and propagate information from these viewpoints to more challenging ones.

3.4 Embodiment

Embodied agents can move and interact with their environment through a physical apparatus. 3D simulators have been an important part of modelling embodiment in a virtual setting. Many of the environments are photo-realistic reconstructions of indoor [56, 3, 66, 9] and outdoor [15, 20] scenes, and provide 3D ground truth labels for objects. These simulated environments have been used to study tasks such as visual navigation and exploration [10, 22, 19], visual question answering [14], tracking [24], and

object recognition [11, 67]. In our work, we use a simulated embodied agent to discover objects and fixate their sensors on them to obtain object-centric data for fine-tuning a detector.

Chapter 4

Seeing By Moving

4.1 Introduction

We aim to improve a 2D or 3D object detector in novel scenes and viewpoints in a self-supervised way. Most previous methods that attempt to improve a detector [10, 11, 67] require either ground truth 3D object segmentation or human annotations of 2D images after they have been collected by the embodied agent. Some of those methods train the movement of the embodied agent to select specific viewoints for later labelling [11, 67]. However, acquiring the annotations for the collected images still remains extremely expensive.

We propose a model called Seeing by Moving (SbM). This approach removes the bottleneck of expensive annotations by making the labeling of images self-supervised. We take advantage of the classifier head in a pre-trained object detector, which has high confidence when the object is viewed unoccluded in a common pose. The confidence values of the pre-trained detector serves as a cue to help us select good views of objects in unseen scenarios. By unprojecting the high confidence 2D detections to 3D, segmenting, and re-projecting the 3D segmentations to all views, we are effectively propagating the high confidence detections from "lucky" views to "unlucky" views. Note that this label propagation can also be used on with any embodied agent that collects a series of images of the same scene. We then fine-tune the 2D / 3D object detector using generated pseudo-labels and show large improvements on both indoor and outdoor datasets. At test time, the detector is able to work on a single view. The overview of our framework is shown in Figure 3.1.

4.2 Self-Supervised Data Collection

The data collection pipeline has two stages. The first stage uses an embodied agent with a depth sensor and a pre-trained object detector to estimate the 3D locations of objects in the scene. The second stage

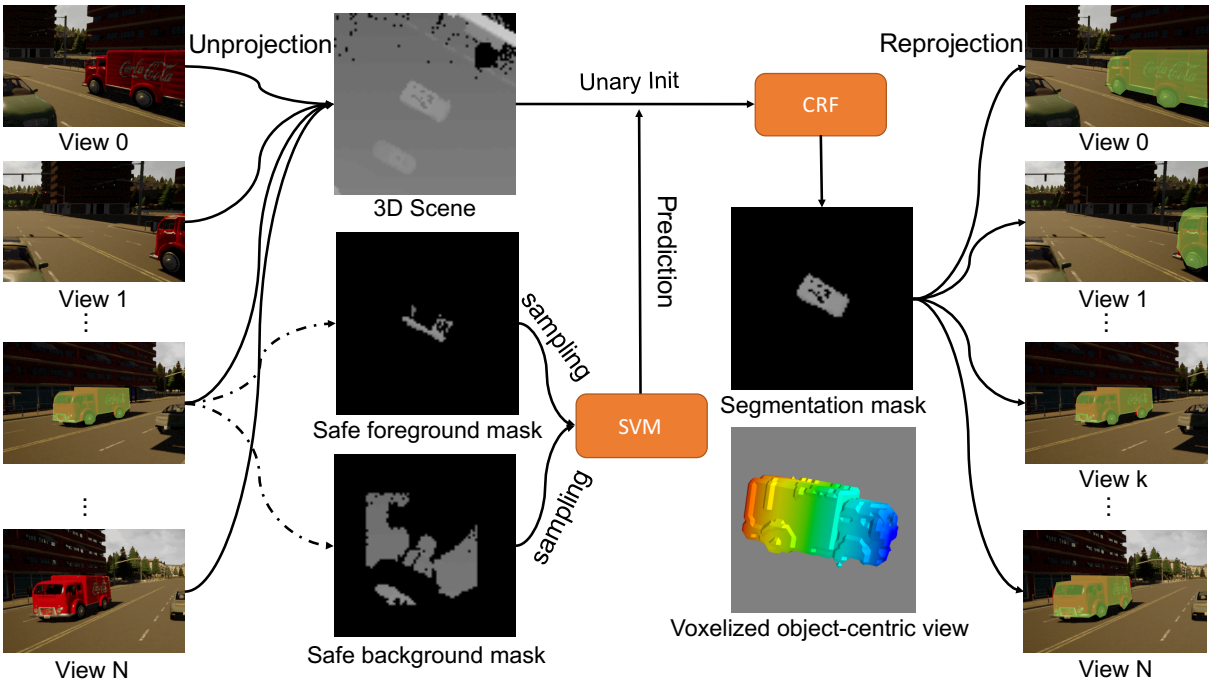

FIGURE 4.1: **Label Propagation.** All observations are unprojected into 3D space, as well as the segmentation mask from confident view k. We sample foreground and background points to train an SVM, then use these to initialize the unary potentials of the fully connected CRF model. The final 3D segmentation is then reprojected to all views to obtain pseudo-labels.

makes the agent rotate around the estimated object centroids to obtain multiview RGB-D observations. Note that any multiview RGB-D data collection method (for example [11]) would be acceptable for our label generation to work. We use a simple hand-designed fixation policy to ensure we obtain a diverse set of views of the objects, and to show that our method works with an easily programmable policy.

In the first stage of our data collection, we randomly spawn the agent in the environment and rotate the agent about its axis until there is at least one highly confident detection of an object in view. For that view, we take all high confidence detection masks, and estimate the 3D centroid of each object using depth and pose information. The centroid of the 3D mask is used to approximate the actual centroid of the object. To avoid repeated detections of the same object, we perform non-maximum suppression based on the heuristic that the 3D centroid estimate of the same object are close together.

In the second stage, we iterate through each centroid, moving the agent inside a ring around the object. This is accomplished by first uniformly sampling navigable and non-obstructed points at various radii from the object centroid. We then have the agent move along a trajectory connecting these locations, and fixate the camera on the object such that the centroid projection is centered in the frame. Using this pipeline, we obtain a set of multi-view images in which there is at least one view with a high confidence detection, which we then propagate to other views.

4.3 Self-Supervised Label Generation

Given N observations from different views for each episode obtained in our data collection, we propagate high confidence detections in some views to all frames in the episode. Taking advantage of object permanence, our self-supervised label generator segments objects in 3D space and reprojects the 3D segmentation to produce 2D pseudo-label masks. The core of the label generator is a fully connected conditional random field (CRF) with Gaussian edge potentials [28], whose unary potential is computed using an SVM classifier. A diagram of our label propagation is shown in Figure 4.1.

From the RGB observations and depth measurements, we first unproject the RGB images from all views into a common 3D reference frame using camera pose and depth. It is important to avoid generating pseudo-labels corrupted with errors, or self-supervision with these pseudo-labels may actually make the detector worse. Simply using high-confidence segmentation masks is not enough, because segmentations are likely to contain errors at the object boundaries [61]. Therefore, for each high confidence detection, we dilate its 2D object mask and take its inverse to obtain a safe 2D background mask. Then we also erode the 2D object mask to obtain a safe 2D object mask. We then unproject all object masks and background masks to the 3D reference frame. Hence, we obtain a pointcloud $S = \bigcup_{i=1}^{N} S^{(i)}$, where $S^{(i)} = \{(x, y, z)_j\}_{j=1}^{K_i}$ denotes the set of points unprojected from view i, with K_i being the number of points.

We now describe the segmentation of objects on pointclouds. For clarity, we focus on one high confidence detection in view k. Note that we can partition the pointcloud $S^{(k)}$ into three sets: foreground, background, and unknown border. For all other views $S^{(k')}$ where $k' \neq k$ the points are unlabelled. To segment the unknown border and the other unlabelled points, we first train an SVM on the foreground and background sets from $S^{(k)}$ using normalized RGB and position as inputs to the classifier. Note that we can also use a pre-trained deep network to obtain the feature representation, but we noticed that the simple 6 dimensional color and position feature works just as well and is much faster. To incorporate pairwise information in the labelling, we refine the SVM estimated with a fully-connected CRF. In the CRF model, each node is a point $S_j^{(i)}$ in the pointcloud and we set its unary potential as the predicted class probabilities from SVM. The pairwise potential between two points $S_j^{(i)}$ and $S_{j'}^{(i')}$ can be written as:

$$\psi_p\left(S_j^{(i)}, S_{j'}^{(i')}\right) = \mu\left(S_j^{(i)}, S_{j'}^{(i')}\right) k(\mathbf{f}_j^{(i)}, \mathbf{f}_{j'}^{(i')}) \tag{4.1}$$

where μ is the label compatibility function given by the Potts Model, and k is a contrast-sensitive potential given by the combination of an appearance kernel and a smoothness kernel, defined similarly as in [28]. The Gibbs energy is therefore the summation of the unary and pairwise potentials.

The process delivers a 3D pointcloud segmentation for each object and we use these segmentations as pseudo-labels to train a 3D object detector on RGB-D data. Visualizations of the pointcloud segmentation are shown in Figure 5.1. For visualizations of 2D pseudo-labels generated from 3D segmentations, please

see the appendix. To obtain pseudo-labels for 2D segmentation, we re-project the 3D segmentation to all views. If multiple confident input masks of the same object (from different views) are given, we keep the pixels corresponding to the most confident prediction at reprojection time. Note that as a benefit of the segmentation in 3D space, we can obtain both modal and amodal masks in 2D, by reprojecting either only points from this view or all points that belong to the object.

Chapter 5

Experiments

5.1 Datasets

5.1.1 Environments

We run our experiments in an indoor and an outdoor environment. For the indoor environment, we use the Habitat simulator [38] with the Replica dataset [56], which contains high quality and realistic reconstructions of indoor spaces. For the outdoor environment, we use the CARLA simulator [15], which renders urban driving scenes. We have an agent move around in the 3D simulator while collecting multi-view RGB-D images of objects in the scenes. Though the simulated environments have noise-free pose and depth measurements, previous work has shown that accurate estimates of pose and depth can be obtained from RGB image under noisy odometry [41]. Since pose and depth estimation is tangential to our work, we assume ground truth pose and depth. The Replica dataset consists of 18 distinct indoor scenes, such as offices, hotels, and apartments. We split the scenes into disjoint sets such that there are 15 for training, 1 for validation, and 2 for testing. In our self-supervised data collection, we capture 25 views in each episode. We have 17k images for training, 1k for validation, and 2.3k for testing. The CARLA driving scenes consist of five distinct towns. We again split them into 3 towns for training, 1 for validation, and 1 for testing. In our self-supervised data collection, we capture 25 views in each episode. We have 5.3k images for training, 1.8k for validation, and 1.8k for testing.

5.1.2 Objects

For CARLA, we randomly spawn two vehicles for each episode. Since CARLA has the same semantic label for all vehicles, we consider detection of all vehicle classes of the COCO dataset [35] during evaluation. For Replica, we keep the default layout of objects in each scene. We consider a subset of the object categories in each simulator based on the following standards: (1) the category is shared between

COCO and Replica, and 2) enough instances (more than 10) occurred in the dataset. These include chair, couch, plant, tv, fridge, bed, table and toilet for Replica. Though our method does not necessitate choosing categories shared between COCO and Replica, its essential for grounding our experimental results in comparison to fully supervised methods that require ground truth annotations for training.

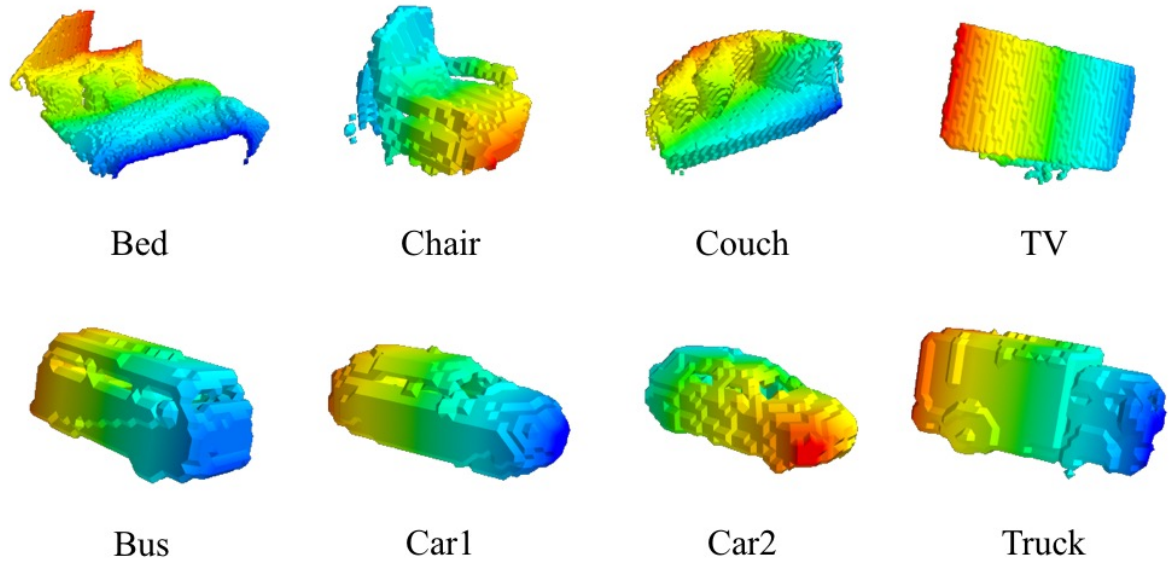

FIGURE 5.1: **Visualizations of 3D object segmentation on CARLA and Replica datasets.** We show colorized voxel visualisations of the 3D segmentations of our method, on both Replica (top) and CARLA (bottom).

5.2 Implementation Details

5.2.1 2D Object Detection

We use the Mask-RCNN model [65] with FPN [33] using ResNet-50 as the backbone. The detector is pre-trained on the COCO dataset. We finetune it on the 2D pseudo-labels of the training set and compute its mAP on a validation at IoU threshold of 50% every 5000 iterations. For comparison, we also fine-tuned the network on the same datasets but with ground truth annotations. For both datasets and both settings, we use a learning rate of 0.001 and a batch size of 2.

5.2.2 3D Object Detection

We train the frustum PointNet model [46] with PointNet [48] backbone on CARLA in a self-supervised way. 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 self-supervised 2D and 3D bounding boxes generated from SbM's 2D and 3D bounding boxes. To gauge our self-supervised frustum PointNet's performance, 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. We estimated convergence manually by estimating when the validation curve has plateaued. We test both models on a new unseen town and compare their performance.

5.3 2D Object Detection

We analyze our SbM framework for 2D object detection by asking the following questions: (1) do our pseudo-labels ouytperform the detector on which they are based? (2) does fine-tuning the object detector on these pseudo-labels improve the detector's performance on unseen scenes? Our experiments in CARLA and Replica show that the answer to both is "yes".

mAP@IoU	Method	Train	Test
0.5	Pre-trained	68.05	68.23
	SbM Labels	**75.94**	76.14
	SbM fine-tuned	-	**78.59**
	GT fine-tuned	-	93.76
0.3	Pre-trained	73.09	75.55
	SbM Labels	**85.62**	85.78
	SbM fine-tuned	-	**86.33**
	GT fine-tuned	-	94.71

TABLE 5.1: **2D object detection performance comparison on CARLA test set** Fine-tuning 2D detector on self-supervised SbM labels increases pre-trained models performance taking its performance closer to supervised fine-tuning.

5.3.1 CARLA

The performance of our method, the pre-trained detector, and the detector fine-tuned on ground truth data is shown in Table 5.1. We report mAP at IoU of 0.5 and 0.3 using the mAP implementation of Padilla et al. [43]. At training time, we investigate the setting where the embodied agent is free to move around, obtain observations, and use SbM to generate predictions for all views. We observe that pseudo-labels generated by SbM have better performance than the pre-trained detector outputs. This shows that moving and performing segmentation in the 3D space helps the detector see better, when compared to treating multi-view images as individual observations. We also finetune the maskrcnn on the SbM pseudo label dataset generated from training set. At test time, the SbM fine-tuned model is deployed

FIGURE 5.2: **Visualizations of 2D detector performance on CARLA test set.** We show paired qualitative examples of the detections of pre-trained 2D detector (left) and SbM fine-tuned 2D detector (right) on the CARLA test set. The improvements are shown in larger fonts for better visibility. The pre-trained detector misses objects and classifies the object as the wrong category, while the fine-tuned model produces accurate class predictions, bounding boxes, and semantic masks.

in unseen environments where only a single RGB image is given as input. Results show that the detector fine-tuned with SbM outperforms the pre-trained detector by a large margin. This indicates that we can improve the detector's performance in unseen environments with no additional human labels. By moving a detector around to gather and label data using information it acquired previously, the detector is able to improve itself. Figure 5.2 shows qualitative comparisons of the detections of the pre-trained detector and the detector fine-tuned by SbM pseudo-labels.

mAP@IoU	Method	Bed	Chair	Couch	Table	Plant	Fridge	Toilet	TV	Avg
0.5	Pre-trained	3.13	12.48	22.81	0.14	13.18	8.37	1.45	30.43	11.58
	SbM (ours)	**7.03**	**22.65**	**34.15**	0.14	**16.26**	**26.49**	**3.72**	**31.39**	**17.73**
	SbM-ws (ours)	29.82	46.91	51.54	29.65	52.15	32.85	42.18	52.10	42.15
0.3	pre-trained	12.79	14.06	23.22	**0.16**	35.57	8.37	2.38	30.86	15.93
	SbM (ours)	**19.39**	**33.03**	**36.74**	0.14	**38.20**	**30.97**	**7.18**	**38.57**	**25.53**
	SbM-ws (ours)	52.21	67.46	75.76	46.69	68.58	56.23	62.53	69.51	61.99

TABLE 5.2: **2D object detection performance of pre-trained Mask-RCNN vs self-supervised SbM vs weakly supervised SbM on Replica training set.** Self-Supervised SbM consistently outperform pre-trained MaskRCNN across most categories. Providing weak supervision (a single view ground truth annotation) to SbM increases its performance significantly on all categories.

mAP@IoU	Method	Bed	Chair	Couch	Table	Plant	Fridge	Toilet	TV	Avg
0.5	Pre-trained	-	11.87	**30.44**	0.76	33.07	**0.50**	-	41.60	19.71
	SbM Fine-tuned (ours)	-	**25.81**	26.33	**5.59**	**44.47**	0.00	-	**43.82**	**24.34**
	GT Fine-tuned	-	57.86	53.47	5.31	89.92	46.46	-	87.20	56.70
0.3	Pre-trained	-	15.61	**40.59**	0.76	36.86	**0.50**	-	41.60	22.65
	SbM Fine-tuned (ours)	-	**20.48**	36.99	**5.59**	**64.55**	0.00	-	**57.99**	**32.60**
	GT Fine-tuned	-	62.22	57.09	5.31	93.48	50.28	-	88.38	59.46

TABLE 5.3: **2D object detection performance of pre-trained, SbM fine-tuned (ours), and ground truth fine-tuned Mask-RCNN on Replica test set.** Training on SbM generated pseudo-labels improve the detector performance on the test set by a large margin.

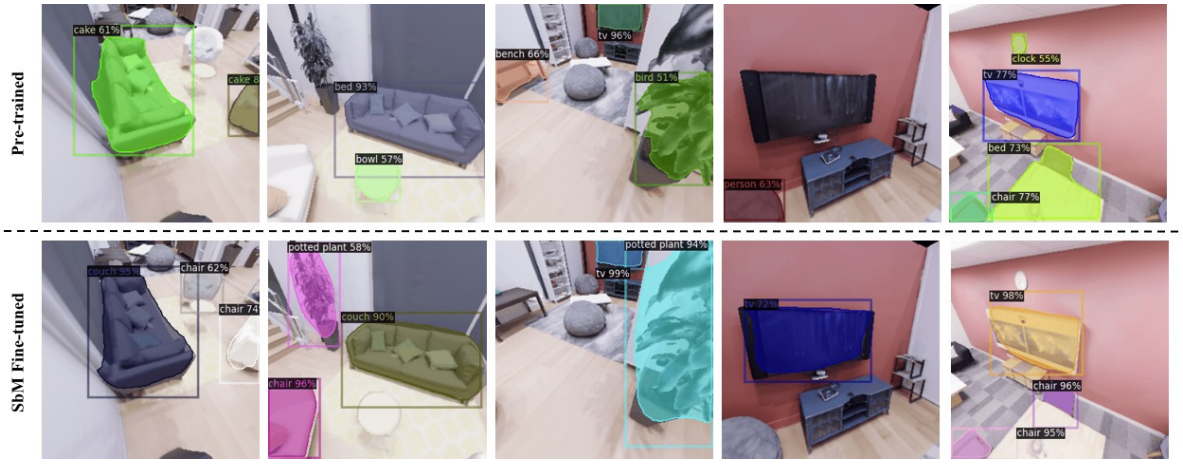

FIGURE 5.3: **Visualizations of 2D detector performance on Replica test set.** We show paired qualitative examples of the predictions of the pre-trained 2D detector (top) and the SbM fine-tuned 2D detector (ours) (bottom) on Replica test set. The pre-trained detector misses objects and classifies the object as the wrong category, while the fine-tuned model produces accurate class predictions, bounding boxes, and semantic masks.

5.3.2 Replica

The performance of our SbM label propagation on the training set is shown in Table 5.2. We observe that SbM-generated labels are more accurate than the pre-trained detector on all classes, indicating that moving around helps generate better labels. Note that the performance on "table" category is very low for both the pre-trained detector and SbM. This is because the dining table class in COCO is visually very different from the table class in Replica (see supplementary). The performance comparison of the pre-trained, SbM fine-tuned, and ground truth fine-tuned detectors on the test set is shown in Table 5.3. The SbM fine-tuned detector overall outperforms the pre-trained detector by large margins while improving performance on most categories. In Figure 5.3, we also present qualitative comparisons of the detections of the pre-trained detector and the detector fine-tuned by SbM pseudo-labels. This confirms that fine-tuning on pseudo-labels generated by moving around makes the detector robust to viewpoint change and output

accurate predictions. Surprisingly, the performance of couch decreased after fine-tuning even though the pseudo-label mAP of couch is higher than pre-trained MaskRCNN on the training set. From visualizations (see supplementary), we found that the pre-trained detector often recognizes couch as bed with high confidence, which are propagated to other views by SbM, thus corrupting the pseudo-labels. This reveals a limitation of our method, which can possibly be mitigated by using context aware detectors [39, 64] or by providing weak supervision.

mAP@IoU	Method	Bed	Chair	Couch	Table	Plant	Fridge	Toilet	TV	Avg
0.5	SbM	7.03	22.65	34.15	**0.14**	**16.26**	**26.49**	**3.72**	31.39	17.73
	SbM Fine-tuned	**9.12**	**25.10**	**42.20**	0.01	11.75	21.84	0.27	**40.47**	**18.85**
0.3	SbM	19.39	33.03	36.74	**0.14**	38.20	**30.97**	**7.18**	38.57	25.53
	SbM Fine-tuned	**36.08**	**38.83**	**54.56**	0.01	**38.74**	26.03	0.47	**57.50**	**31.53**

TABLE 5.4: **Fine-tuning with pseudo labels increases quality of generated labels.** Reported is the performance of pseudo labels obtained from using pre-trained Mask-RCNN versus using our fine-tuned Mask-RCNN in the segmentation pipeline. This demonstrates how our method can be used in continual learning setup.

5.4 Weakly-Supervised Label Propagation

As previously noted, moving around does not help much in cases where the pre-trained detector performs poorly on all views. Can we generate higher quality labels if provided weak supervision? We show that our SbM framework can also perform pseudo-label generation for the set of multi-view images when we have some ground-truth 2D annotations, enabling us to generate high quality labels for non-COCO instances, as well as for categories where the pre-trained detector fails. In our experiments, we only provide a ground truth annotation on 1 view for each object out of the 25 available views, making the label propagation procedure weakly supervised in 2D. We report the label quality in Table 5.2, where the weakly supervised SbM is denoted as SbM-ws. We observe that with one ground truth 2D annotation per object, the pseudo-label quality is better than both pre-trained MaskRCNN and self-supervised SbM by a large margin. This suggests that our SbM framework would generates better pseudo-labels with an improved pre-trained detector.

5.5 Continual Improvement

From previous experiments, we see that (1) the pre-trained detector can be improved in a self-supervised manner on collected data; (2) better detection (weak supervision) on some views improves the pseudo-label quality. Therefore, we can naturally formulate the problem in a continual learning setting: we can

deploy the fine-tuned detector again in the training environments and repeat the first and second stage of our framework to improve the detector further. For clearer comparison, we deploy the fine-tuned detector on the collected training images again and perform confident label propagation. The results are presented in Table 5.4. Comparing the pseudo-labels generated from pre-trained detector detections and the ones from fine-tuned detector detections, we see that the quality of the generated labels is further improved in this continual setting. While the overall mAP and the mAP for most categories improved upon deploying the fine-tuned detector for generating labels, interestingly, performance on some categories dropped slightly. We believe that adding more relationship constraints within environment categories is essential for scaling to full continual learning setup as proposed by Chen et al. [13] and Carlson et al. [8].

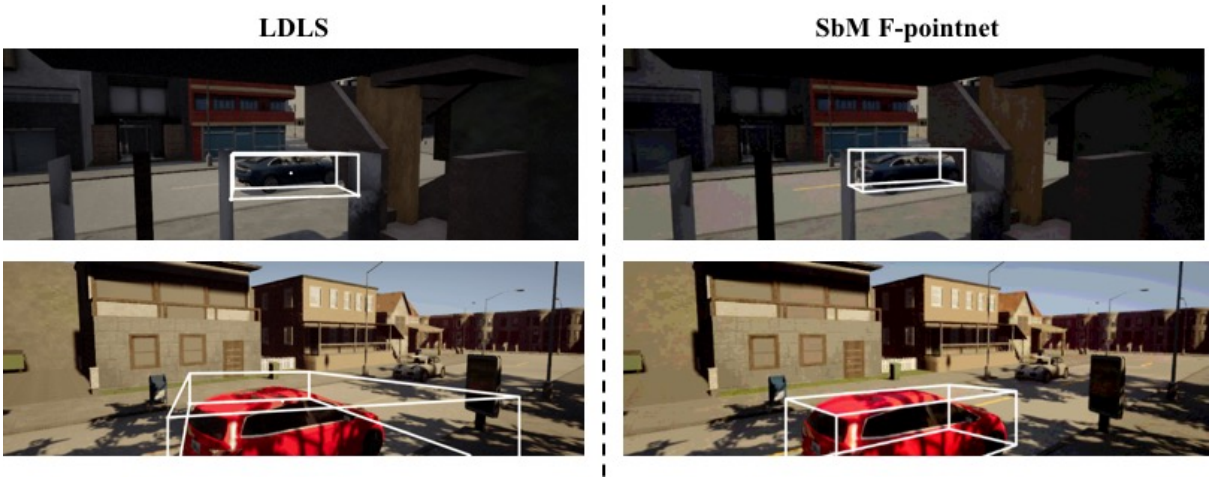

FIGURE 5.4: **Visualizations of 3D detector performance on CARLA test set.** We show paired qualitative examples of the detections of LDLS [61] (left) and SbM-trained frustum PointNet (ours) (right) on the CARLA test set. While LDLS gets a rough box estimation, the SbM-trained frustum PointNet is able to obtain bounding boxes that are much tighter and better-oriented.

5.6 3D Object Detection

Can we train a 3D object detector self-supervised without any ground truth data? To answer this question, we compare the two versions of frustum PointNet: one trained on SbM's self-supervised 3D and 2D labels (Figure 5.1), and the other trained on ground truth 3D and 2D labels. We also compare our method with the semi-supervised LDLS [61] method. The experiments are conducted in CARLA.

Table 5.5 shows the test set performance of LDLS [61], frustum PointNet trained on SbM segmentations, and frustum PointNet trained on ground truth. Our self-supervised frustum PointNet model outperforms LDLS significantly. Our model also achieves reasonably good performance as compared to the fully supervised frustum PointNet model. We show qualitative examples of the 3D detections from LDLS and SbM fine-tuned frustum PointNet in Figure 5.4. This demonstrates that the 3D segmentation labels

Method	mAP@IoU=0.25
LDLS [61]	44.03
SbM Self-Sup. F-PointNet (ours)	**64.39**
Supervised F-PointNet	85.06

TABLE 5.5: **Fine-tuning with SbM labels outperform self-supervised LDLS.** 3D object detection performance of LDLS [61], frustum PointNet trained on SbM segmentations, and GT-trained frustum PointNet on the CARLA test set.

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.

Chapter 6

Conclusion

In this work, we introduce a fully self-supervised method for obtaining labels for fine-tuning a pre-trained detector. In multiple domains, we demonstrate that our method significantly improves the performance of a pre-trained detector simply by moving around, and generalizes to test domain. We also show our method can be extended to train a 3D detector without any 3D annotations. For future work, we believe there is a lot more to explore in the area of self-supervised or weakly-supervised improvement of detectors. Our method currently assumes accurate odometry and depth as input, whereas in real-world applications these inputs are likely to be noisy. Another limitation of our method is that it assumes that the pre-trained detector makes correct predictions for at least some of the available views. Loosening these constraints is a direct avenue for future work.

Appendix A

Appendix

A.1 Apendix Overview

In Section A.2, we provide implementation details for our SbM method and data collection. In Section A.3, we provide additional visualizations of our output. In Section A.4, we discuss the limitations of self-supervised SbM in detail. In Section A.5, we investigate the effects of adding multi-view consistency constraints on the quality of SbM generated labels. Finally, In Section A.6, we discuss the weakly supervised SbM method that can address some of the limitations of fully supervised method and also work on novel categories.

A.2 Method Details

A.2.1 Point Cloud

2D-to-3D Unprojection This module converts the input RGB image $I \in \mathbb{R}^{w \times h \times 3}$ and depth map $D \in \mathbb{R}^{w \times h \times 1}$ into a 3D point cloud $\mathbf{P} \in \mathbb{R}^{x \times y \times z}$. For each image, using known intrinsic parameters of the camera $\mathbf{K} \in \mathbb{R}^{4 \times 4}$, we calculate the 3D coordinate of each pixel in the image relative to the camera. To obtain an aggregated point cloud over all viewpoints, for each of the K views in the multi-view sequence, we rotate and translate the point cloud to the reference frame, defined as the coordinate frame of the first view, using the ground truth transformation matrix $[\mathbf{R}|\mathbf{t}] \in \mathbb{R}^{4 \times 4}$. We then aggregate the transformed 3D points across all viewpoints to obtain a dense multi-view point cloud.

3D-to-2D Projection Given a point cloud in a camera coordinate frame, this module computes the visible 2D projection of the point cloud onto a target viewpoint. Before projection, we rotate and translate the point cloud to be in the coordinate frame of the target camera using ground truth transformation matrix $[\mathbf{R}|\mathbf{t}] \in \mathbb{R}^{4 \times 4}$. For each point in the 3D point cloud indexed by (i, j, k), we compute the 2D pixel location (x, y) which the point projects onto, from the current camera viewpoint:

$$[x, y] = [f \cdot i/k, f \cdot j/k], \tag{A.1}$$

where f is the focal length of the camera. If multiple points fall along the same casted ray, we take only the point which is closest to the camera as the projection.

A.2.2 Data Collection

Replica We discover objects using a pretrained COCO MaskRCNN detector. A detected object with a confidence threshold of 0.9 is required for the object to be used for data collection. The centroid of the object is obtained by taking the mean (x, y, z) coordinate of the 3D point cloud obtained by unprojecting

the object mask to 3D world-centric coordinates using the depth map, camera intrinsics, and agent pose. We use a radius range of 1-2 meters from the estimated object centroid for sampling points for the agent to obtain observations.

CARLA For CARLA, we spawn two vehicles so that one is randomly selected from a naturally-occurring spawning location and pose (given by the CARLA simulator), and the second is the closest naturally-occurring spawning location at least 3 meters from the first car. We spawn the agent near the vehicles and randomly sample locations for the agent to obtain observations at a radius range of 3-15 meters from the centroid of the first car.

A.3 SbM Pseudo-Label Visualizations

We show visualizations of the 2D pseudo-masks re-projected from the 3D segmentation for a variety of classes in the Replica dataset in Figure A.1. We can see that the borders of the objects are nicely segmented.

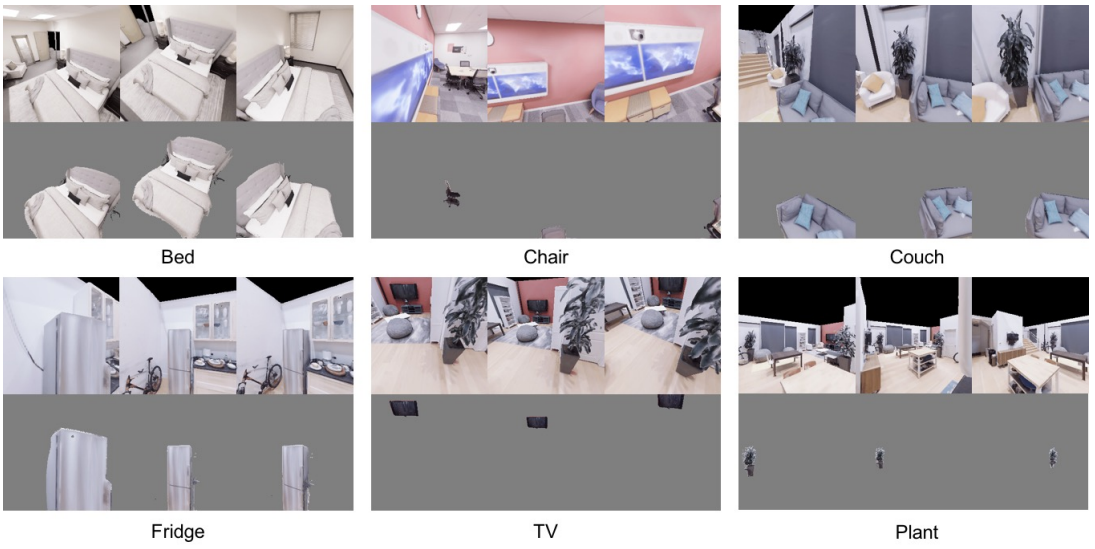

FIGURE A.1: **Example 2D pseudo-labels reprojected from 3D segmentation.** We show examples of the reprojections of 3D segmentation (which are used as 2D pseudo-labels) on a variety of objects in the Replica dataset.

A.4 Limitations of Self-Supervised SbM

Novel and Unmatched Categories One evident limitation of the method is that if objects in the new environment are not among the classes the detector is trained on, our SbM label propagation would not

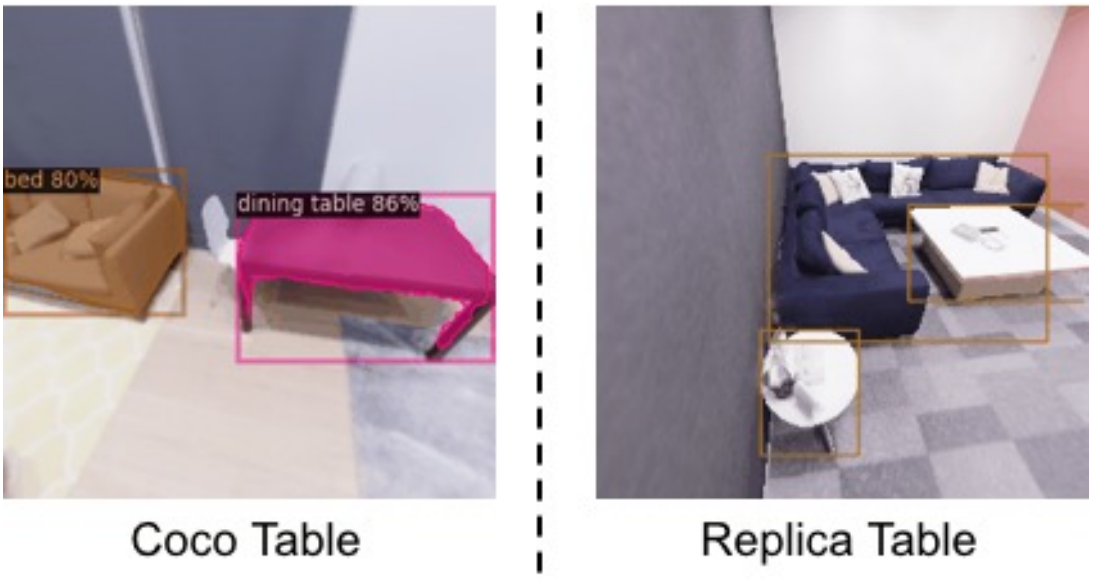

FIGURE A.2: **Semantically and visually different tables in COCO and Replica.** We show a table (predicted as "dining table" by the pre-trained MaskRCNN) on the left, and two actual tables in Replica dataset on the right.

be applicable. For example, Replica contains objects like bean bag and cushion which are not among the COCO classes, so they are never detected by the pre-trained detector. A similar case is when the categories exist in both environments, but they are semantically and visually very different. This is indicated in the paper by the poor performance of SbM on the "table" class is due to the unmatched definitions of the "table" category in COCO and Replica. In Figure A.2, we show that the tables in COCO and Replica differ significantly in semantic meaning and appearances.

Dependency of SbM on Pre-Trained Detector As briefly discussed in the paper, another limitation of the completely self-supervised version of SbM is that in order to propagate high-quality labels, the pre-trained detector must detect objects correctly with high confidence. Due to novel environments and viewpoints, the pre-trained detector sometimes detects wrong objects with high confidence, as shown in Figure A.3.

A.5 Multi-View Consistency

Inspired by previous work [11], we also examined whether adding additional constraints on the detection consistency of the semantic category predictions across views would increase the quality of our labels on the Replica dataset. We kept our pipeline the same except that we removed images from our training set which did not meet the following criteria for each detected object in the image:

mAP@IoU	Method	Bed	Chair	Couch	Table	Plant	Fridge	Toilet	TV	Avg
0.5	Pre-trained	-	11.87	30.44	0.76	33.07	**0.50**	-	41.60	19.71
	SbM w/o consistency	-	**25.81**	26.33	**5.59**	44.47	0.00	-	**43.82**	**24.34**
	SbM w/ consistency	-	4.78	**40.10**	1.08	**69.68**	0.00	-	19.20	22.47
0.3	Pre-trained	-	15.61	40.59	0.76	36.86	**0.50**	-	41.60	22.65
	SbM w/o consistency	-	**20.48**	36.99	**5.59**	64.55	0.00	-	**57.99**	**32.60**
	SbM w/ consistency	-	7.21	**46.91**	1.08	**69.68**	0.00	-	27.22	25.35

TABLE A.1: **2D object detection performance of MaskRCNN pre-trained, SbM fine-tuned, and SbM fine-tuned with a multi-view semantic consistency constraint on Replica test set.** We implemented an additional constraint to exclude views for label generation if the semantic predictions of the same object instance across multiple viewpoints was not the same. Fine-tuning MaskRCNN with the consistency constraint demonstrates worse mAP on average on the Replica dataset compared to fine-tuning without the constraint. The pretrained COCO MaskRCNN is included for reference.

FIGURE A.3: **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.

mAP@IoU	Method Name	Cushions	Nightstand	Shelf	Beanbag	Avg
0.5	SbM-ws	66.92	49.36	43.72	75.74	58.93
0.3	SbM-ws	75.68	65.16	65.82	87.90	73.64

TABLE A.2: **SbM-ws can generate high quality labels for novel categories** Reported is the performance of pseudo labels obtained from SbM-ws on the train set. The high mAP values demonstrate that SbM-ws can be effectively used to generate labels for categories not present in COCO.

1. There were atleast three other views in the mutli-view sequence with high confidence detections of the object instance from the pretrained COCO MaskRCNN.

2. All detections of the object instance predicted the same class (i.e. unique classes predicted for the object instance must not exceed one)

This enforced semantic consistency across viewpoints such that the Mask-RCNN semantic predictions of an object instance across multiple viewpoints was required to be both confident and reliable to keep the object instance for label generation. We hypothesized this consistency constraint would improve the quality of our generated labels, and subsequently the performance when we fine-tuned the detector with the obtained labels. However, the mAP of the detector fine-tuned on labels with the consistency constraint did not improve as much overall from the pretrained detector as compared to the fine-tuned detector trained on labels without the consistency constraint, as shown in Table A.1. We attribute this to a large class imbalance in object categories that were more likely to be tagged as consistent in the dataset, such as plant and couch. We did not investigate this further, and used fine-tuning without this constraint for our main results.

A.6 Weakly-Supervised SbM

FIGURE A.4: **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.

mAP@IoU	Method Name	Cushion
0.5	SbM-ws	83.74
	SbM Trained	**93.62**
	GT Trained	87.24
0.3	SbM-ws	91.58
	SbM Trained	**94.23**
	GT Trained	87.24

TABLE A.3: **SbM-ws labels can be used to train MaskRCNN on novel categories** We compare the performance of MaskRCNN trained on labels produced by SbM-ws (2nd row) vs the MaskRCNN trained on ground truth labels (3rd row) vs the performance of SbM-ws labels. The results show that MaskRCNN trained on SbM-ws labels outperforms the MaskRCNN trained on the ground truth labels.

In section 4.3 of the main paper, we described weakly supervised SbM (SbM-ws) which assumes that each object of interest has a ground truth label for one view out of the 25 recorded views. We showed that assuming one ground truth label increases the mAP performance significantly for categories contained in the COCO dataset used to train MaskRCNN. We also examined this weak supervision in the context of novel object categories not contained in the COCO dataset. Since embodied agents typically encounter a lot of new objects while exploring, it would thus be useful if those new objects could be learned with a smaller number of human annotations. In this section, we show that SbM-ws can be used to generate labels for novel objects too, which are not contained in the category set from COCO. We also show that using these labels, we can train MaskRCNN to recognize those novel objects. We use the exact same experimental settings as in Section 4.3 for data collection and label generation, except we assume that in each multi-view episode, we have a single segmentation label for a novel object. We use this ground truth label as the weak supervision for label generation.

Experiment Details For this experiment we consider four objects: Cushion, Nightstand, Shelf and Beanbag. Notice that these categories are not present in COCO and hence considered categories novel to the pre-trained detector. Using SbM-ws, we generate pseudo labels for these categories on the train set. We only included the cushion pseudo labels obtained from weakly-supervised label generation for training MaskRCNN because it was the only category occurring in more than 30% of the training set. We use a learning rate of 0.001 and a batch size of 2.

Experiment Results To gauge the quality of SbM-ws labels for novel objects, we computed the Mean Average Precision (mAP) of the propagated labels against the ground truth labels. The results are shown in Table A.2. Our results show that SbM-ws can generate highly accurate labels for other views under weak supervision.

We further investigate if we can use SbM-ws labels for training the MaskRCNN on novel categories. We also trained Mask-RCNN on the ground truth views that the weak supervision method assumed in Section 4.3 of the main paper. The results of the experiment are shown in Table A.3 and qualitative examples are shown in Figure A.4. We see that the detector trained on SbM pseudo-labels outperforms the detector trained on the limited ground truth data.

Bibliography

[1] Pulkit Agrawal, Joao Carreira, and Jitendra Malik. "Learning to See by Moving". In: *ICCV*. 2015.

[2] John Aloimonos, Isaac Weiss, and Amit Bandyopadhyay. "Active vision". In: *IJCV* 1.4 (1988), pp. 333–356. DOI: 10.1007/BF00133571. URL: https://doi.org/10.1007/BF00133571.

[3] Phil Ammirato et al. "A dataset for developing and benchmarking active vision". In: *ICRA*. 2017.

[4] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. "SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.12 (2017), pp. 2481–2495. DOI: 10.1109/TPAMI.2016.2644615.

[5] Dana H. Ballard. "Animate Vision". In: *Artif. Intell.* 48.1 (Feb. 1991), 57–86. ISSN: 0004-3702. DOI: 10.1016/0004-3702(91)90080-4. URL: https://doi.org/10.1016/0004-3702(91)90080-4.

[6] Sven Bambach et al. "Toddler-Inspired Visual Object Learning". In: *NeurIPS*. 2018.

[7] Andrei Barbu et al. "ObjectNet: A large-scale bias-controlled dataset for pushing the limits of object recognition models". In: *NeurIPS*. 2019.

[8] Andrew Carlson et al. "Toward an architecture for never-ending language learning." In: *Aaai*. Vol. 5. Atlanta. 2010.

[9] Angel Chang et al. "Matterport3D: Learning from RGB-D Data in Indoor Environments". In: *3DV*. 2017.

[10] Devendra Singh Chaplot et al. "Learning To Explore Using Active Neural SLAM". In: *ICLR*. 2020.

[11] Devendra Singh Chaplot et al. "Semantic Curiosity for Active Visual Learning". In: *ECCV*. 2020.

[12] Liang-Chieh Chen et al. "Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation". In: *ECCV*. 2018.

[13] Xinlei Chen, Abhinav Shrivastava, and Abhinav Gupta. "Neil: Extracting visual knowledge from web data". In: *ICCV*. 2013.

[14] Abhishek Das et al. "Embodied question answering". In: *CVPR Workshops*. 2018.

[15] Alexey Dosovitskiy et al. “CARLA: An Open Urban Driving Simulator”. In: *Proceedings of the 1st Annual Conference on Robot Learning*. 2017, pp. 1–16.

[16] Shimon Edelman and Heinrich H Bülthoff. “Orientation dependence in the recognition of familiar and novel views of three-dimensional objects”. In: *Vision research* 32.12 (1992), pp. 2385–2400.

[17] Kiana Ehsani, Roozbeh Mottaghi, and Ali Farhadi. “Segan: Segmenting and generating the invisible”. In: *CVPR*. 2018.

[18] SM Ali Eslami et al. “Neural scene representation and rendering”. In: *Science* 360.6394 (2018), pp. 1204–1210.

[19] Kuan Fang et al. “Scene memory transformer for embodied agents in long-horizon tasks”. In: *CVPR*. 2019.

[20] Andreas Geiger et al. “Vision meets robotics: The kitti dataset”. In: *The International Journal of Robotics Research* 32.11 (2013), pp. 1231–1237.

[21] Ross Girshick. “Fast R-CNN”. In: *ICCV*. 2015.

[22] Saurabh Gupta et al. “Cognitive mapping and planning for visual navigation”. In: *CVPR*. 2017.

[23] Adam W Harley et al. “Learning from Unlabelled Videos Using Contrastive Predictive Neural 3D Mapping”. In: *arXiv preprint arXiv:1906.03764* (2019).

[24] Adam W. Harley et al. “Tracking Emerges by Looking Around Static Scenes, with Neural 3D Mapping”. In: *ECCV*. Lecture Notes in Computer Science. 2020.

[25] Kaiming He et al. “Mask R-CNN”. In: *ICCV*. 2017.

[26] Dinesh Jayaraman and Kristen Grauman. “End-to-End Policy Learning for Active Visual Categorization”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 41.7 (2019), pp. 1601–1614. DOI: 10.1109/TPAMI.2018.2840991.

[27] Edward Johns, S. Leutenegger, and A. Davison. “Pairwise Decomposition of Image Sequences for Active Multi-view Recognition”. In: *CVPR*. 2016.

[28] Philipp Krähenbühl and Vladlen Koltun. “Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials”. In: *NeurIPS*. 2011. ISBN: 9781618395993.

[29] Weicheng Kuo et al. “Cost-Sensitive Active Learning for Intracranial Hemorrhage Detection”. In: *MICCAI*. 2018.

[30] Alina Kuznetsova et al. “The Open Images Dataset V4: Unified image classification, object detection, and visual relationship detection at scale”. In: *arXiv:1811.00982* (2018).

[31] Ke Li and Jitendra Malik. “Amodal instance segmentation”. In: *ECCV*. 2016.

[32] Yangyan Li et al. “PointCNN: Convolution on χ-Transformed Points”. In: *NeurIPS*. 2018.

[33] Tsung-Yi Lin et al. “Feature Pyramid Networks for Object Detection”. In: *CVPR*. 2017.

[34] Tsung-Yi Lin et al. “Focal Loss for Dense Object Detection”. In: *ICCV*. 2017.

[35] Tsung-Yi Lin et al. “Microsoft COCO: Common Objects in Context”. In: *ECCV*. Ed. by David Fleet et al. 2014.

[36] Wei Liu et al. “SSD: Single Shot MultiBox Detector”. In: *ECCV*. 2016.

[37] Jonathan Long, Evan Shelhamer, and Trevor Darrell. “Fully convolutional networks for semantic segmentation”. In: *CVPR*. 2015.

[38] Manolis Savva* et al. “Habitat: A Platform for Embodied AI Research”. In: *ICCV*. 2019.

[39] Kenneth Marino, Ruslan Salakhutdinov, and Abhinav Gupta. “The More You Know: Using Knowledge Graphs for Image Classification”. In: *CVPR*. 2017.

[40] Tom Mitchell et al. “Never-ending learning”. In: *Communications of the ACM* 61.5 (2018), pp. 103–115.

[41] Raul Mur-Artal and Juan D Tardós. “Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras”. In: *IEEE Transactions on Robotics* 33.5 (2017), pp. 1255–1262.

[42] Bence Nanay. “The importance of amodal completion in everyday perception”. In: *i-Perception* 9.4 (2018), p. 2041669518788887.

[43] R. Padilla, S. L. Netto, and E. A. B. da Silva. “A Survey on Performance Metrics for Object-Detection Algorithms”. In: *2020 International Conference on Systems, Signals and Image Processing (IWSSIP)*. 2020, pp. 237–242.

[44] Deepak Pathak et al. “Learning Instance Segmentation by Interaction”. In: *CVPR Workshop*. 2018.

[45] Mihir Prabhudesai et al. “Embodied language grounding with implicit 3D visual feature representations”. In: *arXiv preprint arXiv:1910.01210* (2019).

[46] Charles R. Qi et al. “Frustum PointNets for 3D Object Detection from RGB-D Data”. In: *CVPR*. 2018.

[47] Charles R. Qi et al. “PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space”. In: *NeurIPS*. 2017.

[48] Charles R. Qi et al. “PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation”. In: *CVPR* (2017).

[49] Joseph Redmon et al. “You Only Look Once: Unified, Real-Time Object Detection”. In: *CVPR*. 2016.

[50] Mengye Ren et al. “SBNet: Sparse Blocks Network for Fast Inference”. In: *CVPR* (2018).

[51] Shaoqing Ren et al. “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”. In: *NeurIPS*. Cambridge, MA, USA: MIT Press, 2015.

[52] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. “U-net: Convolutional networks for biomedical image segmentation”. In: *International Conference on Medical image computing and computer-assisted intervention*. Springer. 2015, pp. 234–241.

[53] Ozan Sener and Silvio Savarese. "Active Learning for Convolutional Neural Networks: A Core-Set Approach". In: *ICLR*. 2018.

[54] Burr Settles. *Active Learning Literature Survey*. Computer Sciences Technical Report 1648. University of Wisconsin–Madison, 2009. URL: http://axon.cs.byu.edu/~martinez/classes/778/Papers/settles.activelearning.pdf.

[55] Nikolai Smolyanskiy, Alexey Kamenev, and Stan Birchfield. "On the importance of stereo for accurate depth estimation: An efficient semi-supervised deep neural network approach". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 2018, pp. 1007–1015.

[56] Julian Straub et al. "The Replica Dataset: A Digital Replica of Indoor Spaces". In: *arXiv preprint arXiv:1906.05797* (2019).

[57] Deqing Sun et al. "Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 8934–8943.

[58] Michael J. Tarr. "Rotating objects to recognize them: A case study on the role of viewpoint dependency in the recognition of three-dimensional objects". In: *Psychonomic Bulletin & Review* 2.1 (1995), pp. 55–82.

[59] Hsiao-Yu Fish Tung, Ricson Cheng, and Katerina Fragkiadaki. "Learning spatial common sense with geometry-aware recurrent networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 2595–2603.

[60] Sudheendra Vijayanarasimhan and Kristen Grauman. "Large-scale live active learning: Training object detectors with crawled data and crowds". In: *CVPR*. 2011.

[61] Brian H. Wang et al. "LDLS: 3-D Object Segmentation Through Label Diffusion From 2-D Images". In: *IEEE Robotics and Automation Letters* 4.3 (2019), pp. 2902–2909. DOI: 10.1109/LRA.2019.2922582.

[62] Shenlong Wang et al. "Deep Parametric Continuous Convolutional Neural Networks". In: *CVPR*. 2018.

[63] Weiyue Wang et al. "SGPN: Similarity Group Proposal Network for 3D Point Cloud Instance Segmentation". In: *CVPR*. 2018.

[64] Xiaolong Wang, Yufei Ye, and Abhinav Gupta. "Zero-shot Recognition via Semantic Embeddings and Knowledge Graphs". In: *CVPR*. 2018.

[65] Yuxin Wu et al. *Detectron2*. https://github.com/facebookresearch/detectron2. 2019.

[66] Fei Xia et al. "Gibson env: Real-world perception for embodied agents". In: *CVPR*. 2018, pp. 9068–9079.

[67] Jianwei Yang et al. “Embodied Amodal Recognition: Learning to Move to Perceive Objects”. In: *ICCV*. 2019.

[68] Jianwei Yang et al. “Visual Curiosity: Learning to Ask Questions to Learn Visual Recognition”. In: *CoRL*. 2018.

[69] Scott Cheng-Hsin Yang, Mate Lengyel, and Daniel M Wolpert. “Active sensing in the categorization of visual patterns”. In: *Elife* 5 (2016), e12215.

[70] Fisher Yu and Vladlen Koltun. “Multi-Scale Context Aggregation by Dilated Convolutions”. In: *ICLR*. 2016.

[71] Yang Yu et al. “Fruit detection for strawberry harvesting robot in non-structural environment based on Mask-RCNN”. In: *Computers and Electronics in Agriculture* 163 (2019), p. 104846.

[72] Bolei Zhou et al. “Places: A 10 million Image Database for Scene Recognition”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2017).

[73] Yin Zhou and Oncel Tuzel. “VoxelNet: End-to-End Learning for Point Cloud Based 3D Object Detection”. In: *CVPR*. 2018.

[74] Yan Zhu et al. “Semantic amodal segmentation”. In: *CVPR*. 2017.