Fast Object Segmentation Learning with Kernel-based Methods for Robotics

Federico Ceola¹², Elisa Maiettini¹, Giulia Pasquale¹, Lorenzo Rosasco²³ and Lorenzo Natale¹

Abstract—Object segmentation is a key component in the visual system of a robot that performs tasks like grasping and object manipulation, especially in presence of occlusions. Like many other computer vision tasks, the adoption of deep architectures has made available algorithms that perform this task with remarkable performance. However, adoption of such algorithms in robotics is hampered by the fact that training requires large amount of computing time and it cannot be performed on-line.
In this work, we propose a novel architecture for object segmentation, that overcomes this problem and provides comparable performance in a fraction of the time required by the state-of-the-art methods. Our approach is based on a pre-trained Mask R-CNN, in which various layers have been replaced with a set of classifiers and regressors that are re-trained for a new task. We employ an efficient Kernel-based method that allows for fast training on large scale problems. Our approach is validated on the YCB-Video dataset which is widely adopted in the computer vision and robotics community, demonstrating that we can achieve and even surpass performance of the state-of-the-art, with a significant reduction (∼6×) of the training time. The code to reproduce the experiments is publicly available on GitHub.

I. INTRODUCTION

Object segmentation is a key task in computer vision and robotics. A precise localization of the object on the image plane is key to solve problems like pose estimation [1], [2], refinement [4], [5] and grasping or manipulation [6]. Like many other computer vision tasks (e.g., image classification and object detection), for decades this was mostly addressed with feature extraction (e.g., sparse coding [7], Fisher vectors [8]) followed by Kernel methods [9] or structured prediction approaches [10], [11]. In recent years, instead, end-to-end training of deep architectures became mainstream since it showed to provide superior performance [12], [13], [14]. However, deep learning is known to require large training data to optimize complex architectures with expensive and long training procedures. While the problem of training data has recently been alleviated by synthetic image generation [15], [16], the computational burden remains. Lately, the literature of one-shot and few-shot object segmentation in video [17] presented methods based on the idea of fine-tuning only part of the network on-line [18], [19]. This approach can learn to segment novel objects in shorter time than standard end-to-end approaches, while keeping performance. However, the procedure of fine-tuning a deep network in general does not guarantee generalization properties, since it involves a non convex optimization and several hyper-parameters, while it can be still prohibitive for robotic settings which require learning to happen very fast (e.g., in seconds or minutes).

Recently, it was shown that pre-trained deep features can be efficiently re-used also to train Kernel-based methods for tasks like image classification [20] object detection and even segmentation [21], [22]. The advantage of Kernel-based methods, among others, is that the number of hyper-parameters to tune is smaller (e.g. in [23], one regularization parameter, one Kernel parameter) and convergence and generalization properties are guaranteed [23]. This approach was recently adopted to achieve high performance and remarkably faster training times in interactive object learning applications for object recognition [24] and detection [25] in robotics. By relying on the above pipelines, a robot can observe a short image sequence of the objects of interest and seamlessly train to detect them right afterwards.

In this paper, we build on this prior work and present a method to re-use general-purpose deep features (e.g., pre-trained on COCO [26]) to train Kernel methods for the dense task of image segmentation. As evidenced from prior work [25], [22], [21], the main challenge lies in the difficulty of training a non-linear classifier over the large training sets that characterize object detection and segmentation problems (where each sample is a region or a single pixel in an image).
To this end, we feed deep features to FALKON [23], a recently proposed Nyström-based Kernel method optimized for large-scale problems. We instantiate one per-pixel FALKON classifier for every class and train it on pixel-wise features extracted from the second-last layer of the network. We first solve the detection task by applying the same approach, which was presented in [25]. Then, we feed the dense features extracted from the detected regions to

1 Federico Ceola, Elisa Maiettini, Giulia Pasquale and Lorenzo Natale are with Humanoid Sensing and Perception, Istituto Italiano di Tecnologia, Genoa, Italy name.surname@iit.it
2 Lorenzo Rosasco and Federico Ceola are with Laboratory for Computational and Statistical Learning and with Dipartimento di Informatica, Bioingegneria, Robotica e Ingegneria dei Sistemi, University of Genoa, Genoa, Italy
3 Lorenzo Rosasco is also with Istituto Italiano di Tecnologia and Massachusetts Institute of Technology, Cambridge, MA lrosasco@mit.edu
https://github.com/robotology/online-detection
https://youtu.be/HdmDYIL48H4
https://youtu.be/eT-2v6-xoSa
FALKON classifiers to perform figure-ground segmentation within each region. To tackle the well-known problem of dataset (positive-negative) imbalance which characterizes the task of object detection, we use the bootstrapping technique from [25]. Instead, to manage the great quantity of data in the figure-ground segmentation task, we apply data subsampling in our training procedure.

As network, we use the region-based Mask R-CNN [13], to which we basically replace every output layer with a set of FALKON classifiers or regressors. The resulting architecture is clean and flexible, allowing to easily train accurate instance segmentation on small datasets in a few minutes, but possibly also on large ones.

We benchmark our method on the YCB-Video dataset [1], a state-of-the-art benchmark for pose estimation and tracking in challenging conditions for robotics, featuring clutter and occlusion across varied objects. In fact, while pose estimation results are widespread over this latter, but also other robotics datasets (see, e.g., the BOP challenge [27]), it is less common for robotic papers to present the partial results over the segmentation task—which, however, is almost always part of their pipelines. Hence, we also hope that this work could pave the way to other works in robotics, to present not only pose estimation but also intermediate segmentation results.

The remaining of this paper is organized as follows. In Sec. II we review the state-of-the-art in the fields of instance segmentation and efficient learning strategies for visual tasks in robotics. Then in Sec. III we describe the proposed approach and in Sec. IV we report on the performed empirical validation. Finally, in Sec. V we summarize the main contributions and identify directions for future works.

II. RELATED WORK

In this section, we overview the main state-of-the-art approaches for object segmentation (Sec. II-A), then we present the literature that tackles the problem of fast adaptation of robotic vision systems to novel tasks (Sec. II-B).

A. Object instance segmentation

The task of instance segmentation [14] aims at classifying all the pixels in an image, while identifying different instances of the categories of interest. Note that, this task is different from semantic segmentation, which has the similar objective of classifying every pixel of an image but not being aware of the different instances. While the field of semantic segmentation is mainly dominated by approaches based on Fully Convolutional Networks (FCNs) [12], [28], the literature in instance segmentation can be divided in the following three groups.

Region-based approaches. Methods in this group split the segmentation task into four main stages: (i) a convolutional feature map is generated by a FCN applied to the whole image, (ii) a list of Regions of Interest (RoIs) is generated by a region proposal method, (iii) a RoI pooling layer [29] warps each RoI and extracts per-RoI features from the convolutional map and, finally, (iv) two sibling shallow Neural Networks predict detections and masks for each feature map. The main representative example of this group is Mask R-CNN [13] which extends the object detection architecture Faster R-CNN [29] by adding, in parallel to the final detection layers, a further output layer for binary mask prediction. Note that, this is in contrast to latest approaches [30], [31], [32], where the classification depends on the prediction of the masks. Mask R-CNN is devised as a monolithic architecture which trains end-to-end, via stochastic gradient descent and back-propagation, all the components for region proposals and feature extraction, object detection and segmentation. The network optimization is carried out by minimizing a “multi-task loss” [29], [13], which accounts for the contribution of classification, bounding box and mask precision (see [29], [13] for details).

Pixel-based approaches. Methods of this group predict a so-called “auxiliary” information for each pixel (like, e.g., energy levels [33]) and then a clustering algorithm groups pixels into object instances based on such information. An example of this approach is presented in [33], where the watershed transform is combined with a deep convolutional network to produce an energy map of the image where object instances are represented as energy basins. Then, they perform a cut at a single energy level to yield connected components corresponding to object instances.

Snake approaches. Methods of this group deform an initial coarse contour to the object boundary by optimizing a handcrafted or learned energy measure with respect to the contour coordinates. In [34], for instance, the presented pipeline uses a graph convolutional network to predict vertex-wise offsets for contour deformation. In [35], instead, the object’s contour is not treated as a general graph, but it leverages the cycle graph topology and uses the circular convolution for efficient feature learning on a contour.

All the aforementioned methods of the state-of-art are “monolithic” deep architectures, trained end-to-end via back-propagation and stochastic gradient descent, with long training times that are not suited for on-line robotic applications. In this work, we propose a region-based method for object segmentation that, building on previous work [25], addresses this issue, allowing for fast model learning.

B. Fast object detection methods in robotics

Fast and efficient adaptation to novel tasks is critical for robots that need to operate in unconstrained environments. Previous works in robotics analyze this problem and propose solutions for object recognition [24] and object detection [36], [25]. In both cases, the proposed pipelines rely on hybrid architectures that integrate deep Convolutional Neural Network’s (CNN) feature descriptors with efficient “shallow” Kernel-based methods (like, e.g., FALKON [23]) for respectively classification or detection. In these works, the key aspect to achieve on-line training time is to decouple the learning of the features and region proposals extractor from the optimization of the Kernel-based methods.
Moreover, in [25] an approximated bootstrapping procedure for selecting hard negative samples has been proposed, to address the well-known foreground-background imbalance problem in object detection. In this work, we extend the pipeline proposed in [25], to also perform object segmentation. Following the same principle as in [25], in our proposed approach, the training of the features and region proposals extractor is decoupled by the optimization of the detection and segmentation methods, allowing for a fast model re-adaptation.

III. METHODS

In this work, we present an instance segmentation method, which allows to learn to predict masks of previously unseen objects (referred to as TARGET-TASK) in a fraction of the time required by state-of-the-art approaches. To do this, it relies on some components of a deep learning based instance segmentation network trained once and off-line on the available data, depicting a different set of objects (referred to as FEATURE-TASK). In this section, we describe the proposed approach. Sec. [III-A] provides an overview of the pipeline, while in Sec. [III-B] the proposed on-line learning strategy for instance segmentation is described.

A. Overview of the pipeline

The pipeline proposed in this work is mainly composed of three modules, as depicted in Fig. 1:

- The Feature Extraction Module employs the first layers of the Mask R-CNN architecture, trained on the FEATURE-TASK, to extract two sets of convolutional features for a set of RoIs, one for On-line Detection and one for On-line Segmentation.
- The On-line Detection Module uses the per-region features computed by the feature extractor to predict their class and to refine their bounding box with the approach described in [25].
- The On-line Segmentation Module, instead, predicts the masks of the instances contained in the RoIs with the approach proposed in this work and described in Sec. [III-B]. As the previous module, also this module is trained on the TARGET-TASK.

The main contribution relies on the latter module. Indeed, we propose a fast and efficient learning strategy that allows to update the instance segmentation model in few minutes. Moreover, we integrate it in the on-line detection pipeline presented in [25], extending it and proposing a novel approach. This permits fast adaptation time as new data is available for both object detection and segmentation.

Learning protocol. Our learning procedure is composed of two stages. Firstly, Mask R-CNN is trained off-line on the FEATURE-TASK via backpropagation, following the training protocol proposed in [13]. Then, the on-line training of both the detector and the mask predictor on the TARGET-TASK are performed in three sub-steps. First, some of the layers of the Mask R-CNN architecture learned in the first phase on the FEATURE-TASK (specifically the backbone, the RPN and the convolutional layers following the Roi Align, depicted as green blocks in Fig. 1), are used to jointly extract the two sets of features needed to train the On-line Detection Module and the On-line Segmentation Module (represented by yellow blocks in Fig. 1). These two modules are then sequentially trained with the on-line learning strategy described in Sec. [III-B].

B. On-line learning strategy

As mentioned in Sec. [III-A], the trainings of the On-line Detection Module and of the On-line Segmentation Module are performed by employing two different sets of features computed by the Feature Extraction Module. These
are extracted in two different points of the Mask R-CNN architecture. Specifically, features necessary to train the On-line Detection Module are extracted from the set of one-dimensional tensors, which in Mask R-CNN are given as input to the last fully connected layers of the detection head. Features necessary to train the On-line Segmentation Module, instead, are contained in the convolutional feature maps produced as output by the penultimate convolutional layer of the mask head in Mask R-CNN. Once the sets of features are extracted for all the training images, the On-line Detection Module and the On-line Segmentation Module can be trained with the procedures described below.

On-line Detection. For the training of the On-line Detection Module, we adopt the method described in [25], but we use the improved implementation of the Feature Extraction Module proposed in [37]. Indeed, such improvement allows to increase the quality of the features, due to the substitution of the first layers of Faster R-CNN [29], with the ones from Mask R-CNN [13]. As in [25], the last fully connected layers of the detection head in Mask R-CNN are substituted with $N$ FALKON [23], [38] binary classifiers ($N$ being the number of classes of the TARGET-TASK) for class prediction, and by $N \times 4$ Regularized Least Squares (RLS) regressors for the refinement [22] of the RoIs proposed by the RPN. To deal with the well-known problem of foreground-background imbalance and with the high redundancy among the negative examples in object detection problems [39], the $N$ FALKON classifiers are trained with the Minibootstrap proposed in [25]. This procedure is an adaptation of the Hard Negatives Mining strategy [22] and it allows to train the classifiers in a few seconds by visiting a random subset of negative examples.

On-line Segmentation. In Mask R-CNN [13], the mask head is a shallow Fully Convolutional Network (FCN), which for every RoI computes $N$ squared masks (one for each of the $N$ classes) of fixed size $s \times s$. In particular, the last layer of the segmentation branch (i.e., the per-pixel mask predictor) is a convolutional layer with $N$ channels, kernel size 1 and stride 1 that receives as input a convolutional feature map of size $s \times s \times f$ (where $f$ is the number of channels). Therefore, the output tensor of the segmentation head has dimension $s \times s \times N$ and the $(ij)_n$th element of such tensor represents the probability of the pixel in the location $ij$ of belonging to an instance of the $n$th class. In this work, we substitute the $N$ kernels for per-pixel classification with $N$ FALKON binary classifiers. We flatten the $s \times s \times f$ feature map into a list of $s \times s$ feature tensors of size $f$ associated to specific locations in the RoIs and use them as samples to train the FALKON classifiers. At training time, we consider the ground-truth bounding boxes as RoIs. The $(ij)_n$th element of the input feature map of size $s \times s \times f$ is considered as a positive training sample for the $n$th classifier if the $(ij)_n$th pixel of the associated ground-truth mask belongs to the foreground and the ground-truth class is $n$. Otherwise, if it is a background pixel within a ground-truth bounding box of class $n$, it is considered as a negative sample for the $n$th classifier.

To speed-up the training procedure, both the positive and the negative instances are subsampled by a fixed factor $r$. We employ FALKON to address the per-pixel classification problem since, as shown in [25] (specifically, refer to Tab. 6 in [25]), among other equivalent choices, it allows to achieve the best time-accuracy trade-off (refer to [23] for details).

Hyper-parameters. For both the detection and segmentation modules, we use FALKON with a Gaussian Kernel. The main hyper-parameters of the proposed method are:

- The standard deviation $\sigma$ of the Gaussian Kernel, the regularization parameter $\lambda$ and the number of Nyström centers $M$ of the FALKON classifiers of both the On-line Detection Module and the On-line Segmentation Module.
- The Minibootstrap parameters to train the On-line Detection Module, i.e., the number of batches $n_B$ and the size of such batches $BS$.
- The sampling factor $r$ of the training examples in the On-line Segmentation Module.

For our experiments, we rely on the criterion proposed in [25] to set the Minibootstrap’s parameters, we cross-validate the FALKON’s parameters as explained in Sec IV-A and we report on an empirical analysis of the impact of the sampling factor $r$.

IV. EXPERIMENTS

In this section, we report the experiments performed to validate the proposed approach. In Sec. IV-A we describe the experimental setup. In Sec. IV-B we provide the results obtained on the YCB-Video benchmark. Finally, in Sec. IV-C we report on ablative studies to analyze our approach.

A. Experimental setup

In this work, we consider Mask R-CNN as baseline for comparison with the proposed pipeline (Ours). We start with Mask R-CNN trained on the FEATURE-TASK and we fine-tune the output layers on the TARGET-TASK (i.e., the fully connected layers for detection and the last convolutional layer for masks prediction). We refer to this architecture and training method as Mask R-CNN (output layers). We use ResNet50 [40] as backbone of Mask R-CNN for both the Mask R-CNN (output layers) baseline and the Feature Extraction Module of the proposed approach (Ours). To set the training hyper-parameters, for both methods we perform a cross-validation on a validation set.

Specifically, the model chosen for the Mask R-CNN (output layers) baseline has been trained for the minimum number of epochs at which the highest segmentation accuracy is achieved on the validation set. In Ours, instead, the validation set is used to select $\sigma$ and $\lambda$ of the FALKON classifiers, both in the On-line Detection Module and in the On-line Segmentation Module. To compare the performance between the two approaches, we consider three metrics: the mean Average Precision (mAP), as defined in [41] of the predicted detections (mAP bbox(%)$ and of the predicted
TABLE I: Benchmark on the YCB-VIDEO dataset. We compare the accuracy achieved by (Mask R-CNN (output layers)) (first row) and by the proposed approach (Ours) (second row).

| Method                                      | mAP50 bbox(%) | mAP50 segm(%) | mAP70 bbox(%) | mAP70 segm(%) | Train Time  |
|---------------------------------------------|----------------|---------------|---------------|---------------|-------------|
| Mask R-CNN [13] (output layers)             | 76.87          | 74.12         | 68.56         | 64.80         | 1h 57m 1s   |
| Ours                                        | 76.46          | 74.66         | 68.71         | 64.96         | 19m 32s     |

Fig. 2: Randomly chosen predictions of Ours on the YCB-Video keyframe set.

B. Benchmark on the YCB-Video dataset

We validate the proposed on-line segmentation approach by considering the 21 objects from the YCB-Video dataset as TARGET-TASK and MS COCO as FEATURE-TASK. As described in Sec. IV-A, we compare the performance obtained by our approach (Ours), with the Mask R-CNN (output layers) baseline. As in [25], to train the On-line Detection Module we set the Minibootstrap’s batch size $B_S$ to 2000, but, given the higher amount of training images with respect to the benchmark reported in [25] (Table 2), we increase the number of batches $n_B$ to 15. The number of Nyström centers $M$ of the FALKON classifiers of both the On-line Detection Module and the On-line Segmentation Module is set to 2000. The sampling factor $r$ of the segmentation training samples is set to 0.3 (refer to Sec. IV-C for an analysis of the impact of this parameter).

Results reported in Tab. I show that our approach, while being ~6 times faster than the baseline, outperforms Mask R-CNN (output layers) in the segmentation accuracy computed with both IoU thresholds, while achieving comparable performance in terms of detection mAP. Train time of Ours is composed of 18min:23s for feature extraction, 35s to train the On-line Detection Module and 34s to train the On-line Segmentation Module. This shows that, once the features are extracted, the TARGET-TASK can be learned extremely fast. We report in Fig. 2 some random images from the test set with the overlaid predicted masks.

C. Ablation studies

In this section, we provide three ablation studies of the proposed pipeline to further validate our approach. Firstly, we evaluate segmentation accuracy by decoupling such task from the detection one. Then, we analyze the impact on accuracy and training time of the sampling factor $r$ (see Sec. III-B). Finally, to compare the accuracy of our method with the one of Mask R-CNN fully trained on the TARGET-TASK, we benchmark our approach in a scenario in which FEATURE-TASK and TARGET-TASK correspond.

Segmentation of ground-truth bounding boxes. In Mask R-CNN [13] and in our pipeline, the segmentation performance depends on the detection. To separately evaluate the segmentation errors, we consider the ideal case in which the bounding boxes proposed by the detection branch correspond to the ground-truths. FEATURE-TASK

To 2000, but, given the higher amount of training images with respect to the benchmark reported in [25] (Table 2), we increase the number of batches $n_B$ to 15. The number of Nyström centers $M$ of the FALKON classifiers of both the On-line Detection Module and the On-line Segmentation Module is set to 2000. The sampling factor $r$ of the segmentation training samples is set to 0.3 (refer to Sec. IV-C for an analysis of the impact of this parameter).

Results reported in Tab. I show that our approach, while being ~6 times faster than the baseline, outperforms Mask R-CNN (output layers) in the segmentation accuracy computed with both IoU thresholds, while achieving comparable performance in terms of detection mAP. Train time of Ours is composed of 18min:23s for feature extraction, 35s to train the On-line Detection Module and 34s to train the On-line Segmentation Module. This shows that, once the features are extracted, the TARGET-TASK can be learned extremely fast. We report in Fig. 2 some random images from the test set with the overlaid predicted masks.

C. Ablation studies

In this section, we provide three ablation studies of the proposed pipeline to further validate our approach. Firstly, we evaluate segmentation accuracy by decoupling such task from the detection one. Then, we analyze the impact on accuracy and training time of the sampling factor $r$ (see Sec. III-B). Finally, to compare the accuracy of our method with the one of Mask R-CNN fully trained on the TARGET-TASK, we benchmark our approach in a scenario in which FEATURE-TASK and TARGET-TASK correspond.

Segmentation of ground-truth bounding boxes. In Mask R-CNN [13] and in our pipeline, the segmentation performance depends on the detection. To separately evaluate the segmentation errors, we consider the ideal case in which the bounding boxes proposed by the detection branch correspond to the ground-truths. FEATURE-TASK
Fig. 3: Segmentation mAPs and training time for increasing values of the sampling factor $r$.

| Method                  | mAP50 bbox(%) | mAP50 segm(%) | mAP70 bbox(%) | mAP70 segm(%) |
|------------------------|---------------|---------------|---------------|---------------|
| Mask R-CNN [13] (full) | 89.76         | 91.29         | 86.05         | 80.41         |
| Ours                   | 90.98         | 92.28         | 86.90         | 80.17         |

TABLE II: Ablation study with YCB-VIDEO dataset used as both FEATURE-TASK and TARGET-TASK. We compare the results of the full training of Mask R-CNN (Mask R-CNN (full)) (first row) with the proposed approach (Ours) (second row) considering as feature extractor the weights learned with the training in the first row.

Training samples vs. training time and mAP. Since the time required to train the On-line Segmentation Module is critical in robotic applications, we provide an analysis of how the choice of the sampling factor $r$ affects both the training time and the segmentation accuracy of such module. As it can be observed in Fig. 3a, the sampling factor $r$ can be pushed to the extreme while preserving the segmentation accuracy in terms of both the considered IoU thresholds.

YCB-Video for both FEATURE-TASK and TARGET-TASK. The goal of this work is to quickly learn to segment previously unseen objects. To this aim, in Sec. IV-B we showed the efficiency of the proposed approach when the TARGET-TASK is different from the FEATURE-TASK. Nevertheless, to further validate our pipeline, we evaluate the performance of our method when the two tasks correspond. In particular, we set them by considering the 21 object identification task in the YCB-Video dataset (the TARGET-TASK defined in Sec. IV-A). In this case, we train, as baseline, the entire Mask R-CNN network (starting from the weights pre-trained on the COCO dataset) on the TARGET-TASK (Mask R-CNN (full)). Then, we use such weights to extract the features needed to train the On-line Detection Module and the On-line Segmentation Module with the approach described in Sec. III-B (Ours).

V. CONCLUSIONS

Fast learning and efficient adaptation are fundamental for robots that need to quickly adjust their vision systems to ever-changing environments. In this perspective, this work presents a system for instance segmentation that extends and improves previous on-line learning approaches for object detection [25]. The proposed method combines Mask R-CNN for feature extraction and two sets of FALKON classifiers to efficiently perform object detection and segmentation. The resulting pipeline allows for fast adaptation to novel tasks in a fraction of the time of the deep learning based counter parts, representing a step toward the implementation of more adaptive robotic vision systems.

In Sec. IV-C, we showed that we can achieve high per-pixel prediction accuracy with few training samples. We believe that this can be pushed to the extreme by integrating few-shots learning techniques in the proposed pipeline. Moreover, given the importance of instance segmentation for other more complex tasks, such as 6D pose estimation and object grasping, the proposed approach may represent a starting point towards the on-line adaptation of such tasks.
[40] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 770–778, 2016.

[41] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. International Journal of Computer Vision, 88(2):303–338, June 2010.

[42] Giulia Pasquale, Tanis Mar, Carlo Ciliberto, Lorenzo Rosasco, and Lorenzo Natale. Enabling depth-driven visual attention on the icub humanoid robot: Instructions for use and new perspectives. Frontiers in Robotics and AI, 3:35, 2016.

[43] E. Maiettini, G. Pasquale, L. Rosasco, and L. Natale. Interactive data collection for deep learning object detectors on humanoid robots. In 2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids), pages 862–868, Nov 2017.

[44] Christopher Xie, Yu Xiang, Arsalan Mousavian, and Dieter Fox. The best of both modes: Separately leveraging rgb and depth for unseen object instance segmentation. In Conference on robot learning, pages 1369–1378. PMLR, 2020.

[45] Markus Suchi, Timothy Patten, David Fischinger, and Markus Vincze. Easylabel: a semi-automatic pixel-wise object annotation tool for creating robotic rgb-d datasets. In 2019 International Conference on Robotics and Automation (ICRA), pages 6678–6684. IEEE, 2019.