Probabilistic Multi-Class Segmentation for the Amazon Picking Challenge

Rico Jonschkowski Clemens Eppner* Sebastian Höfer* Roberto Martín-Martín* Oliver Brock

***Abstract*— We present a method for multi-class segmentation from RGB-D data in a realistic warehouse picking setting. The method computes pixel-wise probabilities and combines them to find a coherent object segmentation. It reliably segments objects in cluttered scenarios, even when objects are translucent, reflective, highly deformable, have fuzzy surfaces, or consist of loosely coupled components. The robust performance results from the exploitation of problem structure inherent to the warehouse setting. The proposed method proved its capabilities as part of our winning entry to the 2015 Amazon Picking Challenge. We present a detailed experimental analysis of the contribution of different information sources, compare our method to standard segmentation techniques, and assess possible extensions that further enhance the algorithm's capabilities. We release our software and data sets as open source.**

I. Introduction

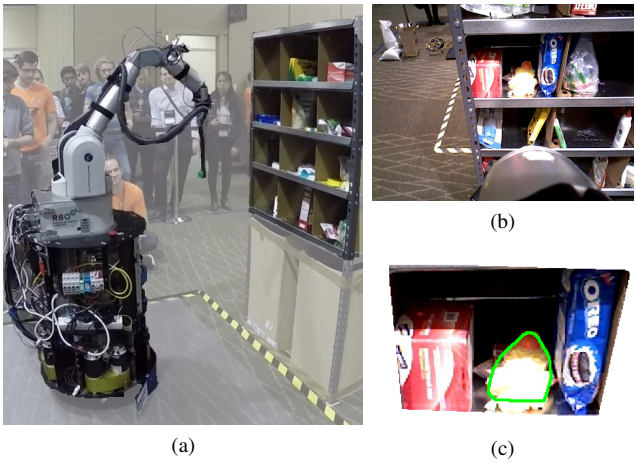
The multi-class segmentation approach we present in this paper is a key component of our winning entry [1] to the 2015 Amazon Picking Challenge (APC) [2]. This warehouse logistics challenge required robots to fulfill an order by autonomously recognizing and picking twelve objects from the bins of a warehouse shelf (see Fig. 1). Each bin contained between one and four objects, selected from a set of 25 known objects. To localize the target object, our system performs object segmentation and classification on an RGB-D image (see Fig. 1c). During the APC, our method segmented and identified all of the twelve objects correctly (Fig. 3), enabling the robot to successfully pick ten of them, outperforming all 25 other teams. A detailed description of the complete system can be found in our systems paper [1].

Prior to the competition, our tests of off-the-shelf libraries for object recognition, segmentation, and pose estimation revealed substantial shortcomings in the APC setting. These findings are confirmed in a poll of all APC teams after the competition: the teams considered perception to be the most difficult aspect of the APC [3]. This difficulty stands in contrast to the availability of excellent open source libraries, such as PCL and OpenCV, and occurred for a seemingly simple perception problem. More detailed tests of LINEMOD, a standard object detection and pose estimation algorithm, showed only 32% accuracy when applied to the APC setting [4]. Even after tailoring the method to the APC setting, it only achieves 60% accuracy.

We gratefully acknowledge the funding provided by the Alexander von Humboldt foundation and the Federal Ministry of Education and Research (BMBF), the European Commission (SOMA project, H2020-ICT-645599) and the German Research Foundation (Exploration Challenge, BR 2248/3-1).

All authors are with the Robotics and Biology Laboratory, Technische Universität Berlin, Germany

* Authors contributed equally to this work.



(a) (b) (c)

Fig. 1. Our trial in the Amazon Picking Challenge: (a) the robot looks for an object in the shelf; it observes the scene from an RGB-D camera that is mounted on its forearm; (b) view from the RGB-D camera; (c) result of the segmentation of the target object (*duck toy*)

If strong research teams from academia and industry cannot leverage the achievements of decades of computer vision research in the context of the APC, we must question our assumptions. Can we achieve generic and task-agnostic perception for robotics? In our APC solution, we deviated from the trajectory of solving perception problems in their most general form. Instead, we were successful with a simple perception method that exploits knowledge about the task at hand. Our system probabilistically combines a number of simple color and depth features, designed to take advantage of the characteristics of the warehouse setup.

Of course, the resulting perception system is task-specific and tailored to the APC. However, we propose that a path towards robust and more general robot perception could follow a series of such task-specific systems of increasing generality. Only by taking this path, we can identify crucial components for general robot perception. Towards this goal, we perform an extensive post-hoc evaluation of our method which provides some surprising findings, e.g. how simple image processing can outperform probabilistic inference. We believe that these findings provide valuable insight not only for building robots for warehouse scenarios, but also for more robust robot perceptual systems.

Our contribution is threefold: (i) we describe our object segmentation method that was key to winning the APC; (ii) we present a thorough experimental evaluation beyond the challenge requirements; and (iii) we derive more general lessons of how to build perception systems for specific tasks.

II. Related Work

The perception problem in the APC is an instance of the general object detection and segmentation problem, which is actively researched in computer vision. Results of popular vision competitions such as PASCAL VOC [5] or ImageNet [6] show that solutions related to our problem currently receive significant attention and winning entries steadily reduce error metrics over the years. A common theme among those entries are sliding-window approaches using deformable parts models [7] or deep neural networks in combination with large-scale datasets [8]. They give as output bounding boxes with highly likely object locations, which contain many pixels that are not part of the object. This renders these representations difficult to use in a robotic manipulation context, where accurate, or at least conservative, shape estimates are crucial to decide on appropriate actions.

Multi-class segmentation addresses this problem by identifying for each pixel in an image to which object class it belongs. A popular approach to multi-class image segmentation are conditional random fields (CRFs, [9]). They encode local (per-pixel or region) and pairwise statistical preferences, and define an energy whose minimum corresponds to the most likely segmentation. CRFs provide a principled way to integrate different sources of information (e.g. color, texture, shape, location, and depth) and can easily incorporate smoothness constraints [10, 11]. Similar to CRFs, our approach combines different sources of information in a probabilistic fashion. A comparison between a generic CRF and our method is shown in Sec. V-B.

A more classical yet effective approach to object segmentation is histogram backprojection [12]. Given the color histogram of the target object, the method backprojects the histogram into the image by replacing each pixel color with the respective bin count of the histogram. Areas with high bin counts are then assumed to be the target object. In this paper, we extend histogram backprojection to a probabilistic version and also incorporate additional non-color features.

In the context of robotic manipulation, approaches to object detection usually aim at estimating the full 6D object pose, and therefore rely more heavily on depth data. Detection and pose estimation can be based on CAD models [13], feature point histograms [14, 15], local keypoint descriptors like SIFT and SURF [16], or edge and normal information to address textureless objects as done in LINEMOD [17]. These approaches are based on table top assumptions and do not scale well when confronted with the limited visibility and clutter imposed by the APC setup. For example, LINEMOD [17] shows already significant translational error with only two items per bin [4]. Although we do not estimate the 6D pose of objects, our results show that the information contained in the segmentation is sufficient for our system to pick successfully.

III. Problem Analysis

In order to create an multi-class segmentation method for the APC setting, we analyze the problem to identify how to address its challenges by leveraging useful problem structure.

A. Challenges in the APC Setting

No single discriminative property for all objects: The 25 APC objects were chosen to reflect the large variety present in a warehouse scenario. No single perceptual feature suffices for identification: some objects have distinctive shapes, others are deformable; some objects have distinctive colors, others have similar color histograms, or view-dependent variations; some objects have surfaces amenable to our RGB-D sensor, others are wrapped in plastic bags. We address this problem by combining a variety of features.

Limited object visibility: Camera-based perception can only obtain a partial view of an object in the shelf from a particular camera position. Nearby objects sometimes partially occlude others. We address this challenge by training on perceptual data of objects in different poses.

Uncontrolled lighting: Visual perception is sensitive to lighting conditions. At the challenge venue, the lighting was directly from above and extremely bright, relative to the ambient light. In these conditions, images were nearly saturated in bright regions and appeared black in the remaining ones. To alleviate this problem we transform the RGB image to HSV color space and include features based on depth.

Partial 3D measurements: Kinect-like sensors do not provide reliable 3D measurements for reflective or translucent materials, such as the plastic-wrapped objects or the metal shelf of the APC. We turn this problem into a source of information by using missing depth values as a feature for segmentation.

B. Useful Problem Structure in the APC Setting

Small number of known objects: Since the complete set of objects was available and known beforehand, it was possible to collect training data of these objects in different orientations and bin locations.

Few objects per bin: Since bins contained at most four known objects, we can ignore all other objects during segmentation. Our method automatically uses the most discriminative features for the particular subset of objects present in a bin.

Known shelf: Since the objects are placed in a known shelf that can be tracked by the robot, we can use shelf-related features, such as the height of a pixel in the bin or the distance to the tracked shelf model. These features help to discriminate objects of different sizes and to differentiate between objects and the shelf.

IV. Multi-Class Segmentation Method

Our multi-class segmentation method (video link: `https://youtu.be/TsVUQtRNIts`) consists of the following steps: using a variety of features, we compute for each pixel and each object the probability of the pixel belonging to the object. We then propagate probabilities between nearby pixels using Gaussian smoothing, assign the most likely object label to each pixel, and select the most probable segment per object. In a last step, we make the size of the segment consistent with our expectation and greedily segment objects in sequence, eliminating already segmented objects. We will now explain all steps in detail.

A. Features

Based on the analysis from Sec. III, we describe each pixel by six features that jointly discriminate between the objects:

Color: A discrete value in the range $0-182$ based on the hue-saturation-value (HSV) color representation, which is relatively invariant to lighting conditions. We project the HSV color space to a single dimension by thresholding appropriately: we set the feature to H (ranging from $0-179$) for pixels with distinctive color (S $>$ 90 and V $>$ 20), and otherwise to 180 for white (V $>$ 200), to 181 for gray (200 $>$ V $>$ 50), and to 182 for black pixels (50 $>$ V).

Edge: A binary feature that describes whether the pixel is in the vicinity of a visual edge. We compute this feature by applying Canny edge detection to the image and dilating it with a small elliptical kernel (with a diameter of 11 pixels).

Missing 3D: A binary value representing whether a pixel contains valid depth and therefore 3D information.

Distance to shelf: A continuous value (in *mm*) that denotes the distance of a pixel to the closest point on the shelf. We estimate this value by tracking the shelf in the RGB-D image. For this we start with an estimate based on the localization and forward kinematics of the robot and refine it using the iterative closest point (ICP) method. Pixels without valid depth information are ignored.

Height (3D): A continuous value (in *mm*) that denotes the shortest distance of a pixel to the ground plane of the bin, computed similarly to the distance-to-shelf feature. Pixels without valid depth information are again ignored.

Height (2D): A continuous value (in *mm*) that describes the height of the pixel projected onto the (open) front plane of the shelf bin. This feature approximates 3D height and is only used for pixels without valid depth information.

B. Learning Phase

Given a 6D feature vector per pixel, we now explain how to learn the likelihood of the features for the APC objects.

1) Data Collection: During the preparation of the APC, a dataset was made available that included RGB-D images of all objects from multiple views together with estimated 3D models of the objects.[1] However, we found that differences in the cameras, viewing angle, and lighting conditions made it difficult to transfer models from this dataset to our robot. Moreover, in this dataset the objects are not inside the APC shelf. Therefore, the likelihood of some shelf dependent features as well as a model for the shelf itself could not be learned from this dataset.

We therefore generated a dataset which closely resembled the competition scenario. We placed the objects in the shelf in different poses and collected RGB-D images and the estimated shelf pose. Finally, we manually segmented the images until we had a sufficient number examples for each object to cover their possible poses (161 samples in total, on average 6 samples per object).

[1] http://rll.berkeley.edu/amazon_picking_challenge/

2) Computing Feature Likelihoods: Based on our dataset, we generate feature likelihoods for each object $o \in O$. For each feature f (e.g., color, height), we compute a histogram from the pixels that belong to the hand-labeled object segments and normalize this histogram to get a likelihood $P(X^{(f)}|O=o)$ over the possible values $x^{(f)} \in X^{(f)}$. To be robust against small changes in feature values, we smooth non-binary likelihood functions with a Gaussian kernel (standard deviations for the smoothing kernel: $\sigma_{\text{color}_{0-179}} = 3$, $\sigma_{\text{color}_{180-182}} = 1$, $\sigma_{\text{dist to shelf}} = 7.5mm$, $\sigma_{\text{height(3D)}} = 3mm$, $\sigma_{\text{height(2D)}} = 6mm$). For robustness to large changes in feature values, we mix the likelihoods with uniform distributions, $P(X^{(f)}|O=o) \leftarrow p_{\text{uni}_f} P_{\text{uni}}(X^{(f)}) + (1-p_{\text{uni}_f}) P(X^{(f)}|O=o)$, where we use the following parameters for the different features: ($p_{\text{uni}_{\text{color}}} = 0.2$, $p_{\text{uni}_{\text{dist to shelf}}} = 0.05$, $p_{\text{uni}_{\text{edge}}} = 0.4$, $p_{\text{uni}_{\text{miss3D}}} = 0.2$, $p_{\text{uni}_{\text{height(3D)}}} = 0.4$, $p_{\text{uni}_{\text{height(2D)}}} = 0.8$). Thus, even feature values that have never been observed for an object have non-zero probability and do not entirely rule out certain objects. The parameters σ_f and p_{uni_f} define how much we trust feature f.

C. Multi-Class Segmentation Phase

The learned feature likelihoods are the base of our multi-class segmentation phase, which is illustrated in Figure 2.

1) Cropping and Feature Extraction: In the first step of the multi-class segmentation phase, we crop the RGB-D image to only contain the bin with the target object. This step removes clutter and distracting objects from other bins. We then compute the 6D feature vector described in Section IV-A for each pixel in the cropped image (see Fig. 2). Note that estimating the cropping mask and some of the features (e.g. height) relies on tracking the pose of the shelf.

2) Backprojection and Bayes' Rule: In this step, we compute for each object o in the bin and every pixel i the probability that this pixel belongs to the object given its feature vector $\mathbf{X}_i$, $P(O_i = o|\mathbf{X}_i)$. This results in one *posterior image* per object (see Fig. 2). To compute them, we iterate over all features f and backproject their likelihoods $P(X^{(f)}|o)$ into the image, i.e., we replace the feature values $x_i^{(f)}$ with their likelihood $P(x_i^{(f)}|o)$, similar to [12]. Assuming conditional independence between the features, we multiply their likelihoods for each pixel: $P(\mathbf{X}|o) = P(X^{(\text{color})}|o)P(X^{(\text{height(3D)})}|o)\ldots$. Then, we apply Bayes' rule by multiplying $P(\mathbf{X}|o)$ with an object prior $P(o)$ and normalizing each pixel. We use a flat prior across all objects except the shelf, which we set to be three times as high such that the method assigns uncertain pixels to the shelf segment rather than to the wrong object.

3) Pixel-Labeling and Post-Processing: Nearby pixels often display the same object and should therefore have similar object probabilities assigned to them. To incorporate such spatial information, we smooth each object's posterior image with a Gaussian kernel ($\sigma = 4$ pixels). This step is related to locating an object by convolving its backprojected histogram with a disk [12] and to pairwise potentials in CRFs. The smoothing step evens out single pixels or small regions of much higher or much lower probability than the surrounding

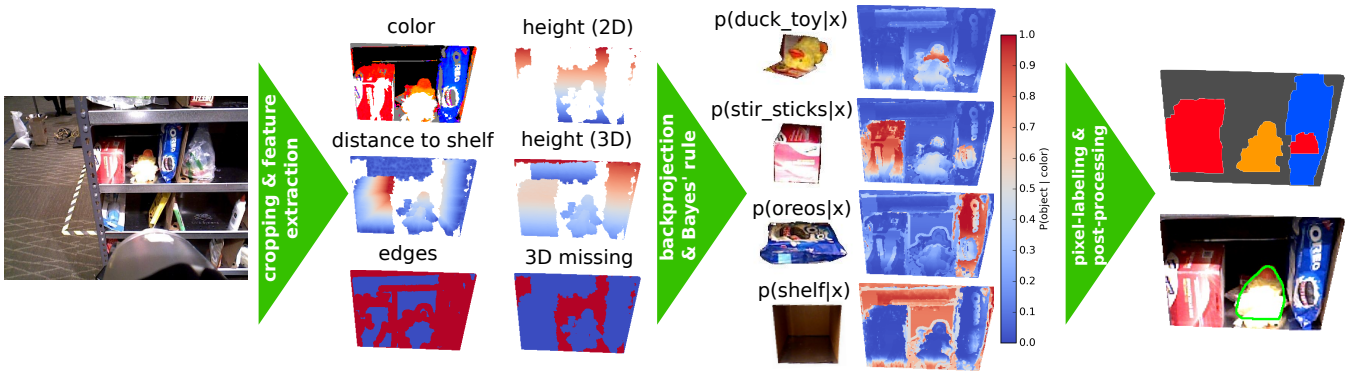


Fig. 2. Overview of the multi-class segmentation phase of our approach

area, which makes the segmentation more robust. Here we implicitly exploit that the APC objects are compact and occupy a significant area of the image.

Based on this smoothed posterior image, we label each pixel i as belonging to the object o with the highest posterior $P(o|\mathbf{X}_i)$ and extract connected regions that are assigned to the same object. In case of having multiple disconnected segments for an object, we select the one that includes the maximum in the smoothed posterior image for that object.

As a post-processing step, we make the segment convex. This step incorporates missing object parts and reflects the convexity of most APC objects. Additionally, we look at the size of the segment (number of pixels) and compare it to segment sizes for this object in our dataset. If the segment is considered too large to be correct (larger than 1.2 times its maximum size in our dataset), we reduce its posterior image (by subtracting 0.05) and reassign the object labels. We repeat this until the segment shrinks to a plausible size.

The last post-processing step is a greedy re-labeling based on the following idea: If we are confident about the segmentation of one object, we do not have to consider this object for the rest of the image. We exploit this by sequentially segmenting the objects, greedily starting with the object that we are most certain about, where we measure certainty by segment size. If the segment size is consistent with our dataset, we assume that we have found the correct segment of this object with high probability, reduce its posterior outside of the segment accordingly (by multiplying it with 0.2) and re-normalize. We proceed in the same way with the next most certain object and continue until the target object has been processed.

V. Experiments

We evaluated our method on a dataset containing 346 manually segmented RGB-D images. The training set includes 161 samples (six per object) recorded in our lab in Berlin. Our test set consists of three parts: a) three runs (66 samples) recorded in our lab, b) five runs (107 samples) recorded in the challenge venue in Seattle, and c) the actual APC run (12 samples). Unless indicated otherwise, all of the following experiments use (b) as test set. Both our implementation and the dataset are publicly available.[2]

[2]https://gitlab.tubit.tu-berlin.de/rbo-lab/rbo-apc-object-segmentation.

Fig. 3. Segmentation results during the APC run; the green line outlines the segments returned by our method; all segments lie on the correct objects; mean precision: 91%, mean recall: 73%, $F_{0.5}$ score: 0.864

A. Performance Evaluations

1) Performance at Amazon Picking Challenge: Fig. 3 shows the result of applying our method in the actual competition run of the 2015 APC at ICRA in Seattle (video link: `https://youtu.be/DuFtwpxQnFI`). Based on the 3D point cloud of the estimated segment, the robot computed a bounding box of the target object, chose the side from which to pick the object, moved its end-effector towards the center of the bounding box until contact, and picked up the object with a vacuum gripper [1]. Our system outperformed the other 25 teams by successfully picking ten out of the twelve objects. The robot only failed in two cases. In bin F it accidentally picked the plush eggs instead of the spark plug due to an inaccurate picking motion. In bin K the robot could not remove the big cheezit box because it got stuck.

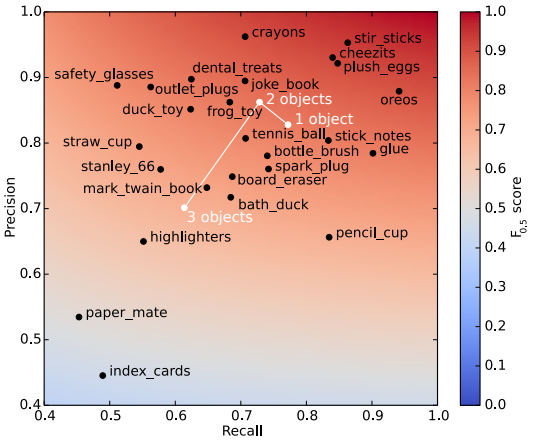


Fig. 4. Performance by object (black) and number of objects in bin (white)

(a) (b) (c)

Fig. 5. Typical failure cases (green line outlines segments found by our method); from left to right: segment detected on the wrong object, segment spanning over several objects, reflections considered as part of an object

2) Performance by Object: Fig. 4 shows the performance of our method across different objects and numbers of objects per bin. Our method had most problems with flat objects that are dominated by white color, e.g., the index cards (Fig. 3 (d) to the right), the paper mate (Fig. 3 (c) to the right), and the highlighters (Fig. 3 (e) to the right). Our method most reliably segmented large objects with distinct colors, e.g., the stir sticks (Fig. 3 (c) to the left), the cheezits (Fig. 3 (k)), and the plush eggs (Fig. 3 (f) to the right). These results are consistent with our findings in Sec. V-C.1, which show that color is in fact the most important feature.

The number of objects per bin also has a strong impact on the performance of our method (see Fig. 4) because with more objects in the bin, the features we are using like color or height become much less discriminative.

Fig. 5 shows typical failure cases: (a) part of an object with similar features is mistaken for the target object, (b) close objects with similar features lead to inaccurate object boundaries, (c) reflections are included in the target object.

3) Increasing the Number of Objects per Bin: Our method can be easily scaled to thousands of possible objects as long as there is only a small number of known objects in every bin. As Fig. 4 suggests, the performance degrades with increasing numbers of candidate objects in the bin. We explicitly augment the list of candidate objects in the bin with objects that were not present. The results in Fig. 6 show a decrease in recall whereas precision remains relatively stable.

B. Comparison to CRF

We compared our method against a widely used generic approach to multi-class segmentation: a CRF based on RGB-D input that uses a learned classifier to estimate the pixel-wise probabilities and a predefined pairwise probability

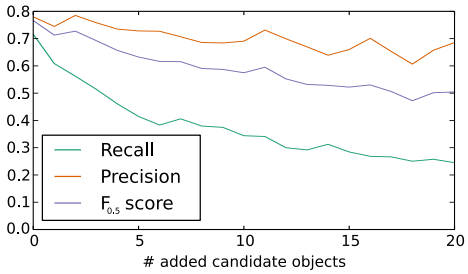


Fig. 6. Performance trend when the assumption about known objects is reduced (increasing the candidate objects); recall decreases quickly but precision stays relatively stable

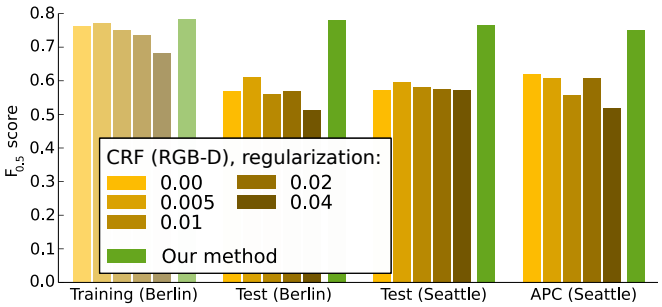


Fig. 7. Comparison to a generic conditional random field (CRF) with different amounts of regularization; the unary potentials are obtained from a random forest

(0.99) for neighbouring pixels to have the same label. We used a random forest [18] as the classifier and regularized it by setting a minimum size of leafs to a fraction of the total number of samples (see Fig. 7).

This experiment shows that the baseline works for the data that it was trained on, but does not generalize well to unseen data. Generic regularization does not solve this problem. Compare this to the performance of our method which stays almost constant when going from training to test set and even when going to data collected in Seattle under very different lighting conditions.

C. Variants of our Method

The next experiments evaluate the contribution of the different parts of our algorithm to the final result.

1) Features: In this experiment we evaluate the importance of each feature to the overall performance of the algorithm (see Fig. 8). First, we evaluated the performance of our algorithm *using only one* of the features. We can observe that none of the features alone reaches a performance comparable to using all of them together. However, color is a powerful feature by itself, because it is sufficient to discriminate well between many objects. Using only the distance to shelf or the height (3D) also obtains relatively good scores because these features discriminate very well between objects and the shelf, which already solves single object bins. The features edge, missing 3D, and height (2D) alone were not able to produce any segmentation. In the second experiment, we *deactivated one* of the features and measured the performance drop. Again color is the most crucial feature. We observe that the missing 3D feature, which could not produce any segmentation when used alone,

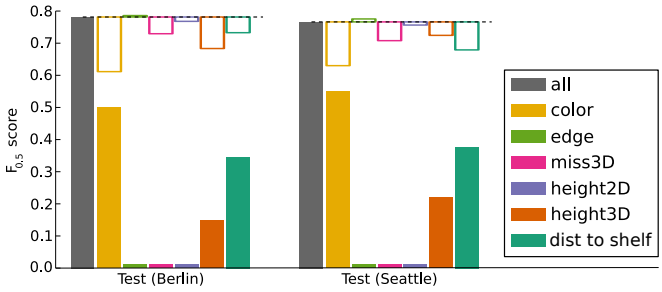


Fig. 8. Contribution of different features to the overall performance; the gray bar indicates our performance using all features; the filled bars show the performance based on a single feature alone; the empty bars show the change in performance if the corresponding feature is removed and the remaining features are used in the model; in both cases, the size of the bars correspond to the usefulness of the features

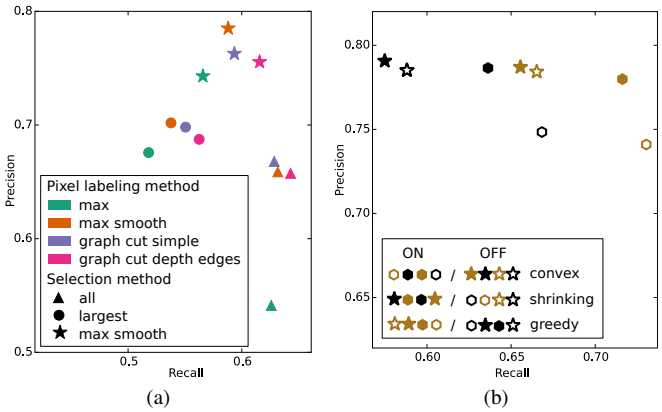


Fig. 9. Combinations of different methods for (a) pixel labeling and selection and (b) re-labeling and post-processing.

contributes substantially to the performance. Interestingly, the height (2D) features seems unnecessary and the edge feature even hurts performance.

2) Pixel Labeling and Selection: For this experiment, we divide the multi-class segmentation process into two steps: pixel labeling and selection. The pixel labeling step assigns an object label to each pixel, which creates image regions of the same label, possibly disconnected. The second step selects one of these regions as the final segment.

We now compare different methods for these two steps. First, we compare four variants of the pixel labeling step. *(i) Max*: assigning to each pixel the label of the most likely object. *(ii) Max smooth*: the same as max, but smoothing the probability image first. *(iii) Simple graph cut*: formulating the labeling in terms of energy minimization: pixel probabilities are turned into potentials and connected to their four neighbors with a high probability (0.99) of having the same label. Then, we apply graph-cut to find the optimal partition of the graph into labeled segments. *(iv) Graph cut using depth edges*: similar to simple graph cut, but we adapt the pairwise probabilities depending on depth differences: very high (0.997) for neighbors with similar depth, high (0.95) if there is no depth information, and low (0.5) for neighbors that cross depth edges. Second, we compare three variants for the selection step: (i) a naive approach of selecting *all* segments, (ii) selecting the *largest* segment, and (iii) selecting the segment that includes the *maximum in the smoothed probability image*.

The results show that, surprisingly, the selection method has a much larger impact on the performance than the pixel labeling method (see Fig. 9a). The simple *Max* pixel labeling is defeated by all other methods, especially in combination with the select-*all* method. All other pixel labeling methods are comparable. Selecting all segments naturally leads to the highest recall but lowest precision. Selecting the largest segment (which is common practice) trades recall for precision. The *Max smooth* selection, however, does this more efficiently. The intuition behind this difference is the following: in most cases the largest segment is also the segment that includes the maximum probability. In these cases both method perform equally well. But in some cases they are not the same and then it seems to be much better to trust in the most probable point instead of the size of the segment. The best combination of methods is using *Max smooth* for segmentation and selection, which is the combination we used in the APC.

3) Re-labeling and Post-processing: Our method includes three re-labeling and post-processing steps: making the segment convex, shrinking it if it is too large, and greedily segmenting the image starting with the largest segment. In this experiment, we tested all combinations of these steps. The results (Fig. 9b) show that greedy re-labeling (golden color) substantially increases the recall irrespective of which other steps are used. Convexity (hexagon symbol) and shrinking (filled symbol), work best together: Making the segment convex increases the recall and shrinking it if necessary improves precision. The best performance is achieved when all three steps are combined.

4) Random Forest for Pixel Probability Estimation: In this experiment we extend our method to use a random forest classifier [18] to estimate the posterior images instead of using likelihood backprojection and Bayes' rule. For the given set of objects in the bin, we trained a random forest to discriminate just these objects and then use the probability estimates from the random forest as posterior image in our method. Apart from this, we retained all other steps and used the same features, except that the color feature is replaced by the original HSV values to allow the random forest to deviate from the thresholds we had set manually.

Our hypothesis was that this hybrid method would suffer from overfitting and not reach the performance of our original method. Surprisingly, the random forest classifier improved our method (see Fig. 10). On the one hand, this classifier did introduce a higher variance in performance between training set and test set compared to using likelihood backprojection. This is because it is less restricted, e.g., it does not assume feature independence and can represent highly complex functions. The likelihood estimation, on the other hand, includes many manually tuned parameters, e.g., parameters for smoothing the feature likelihoods or weights for mixing them with uniform distributions, which introduces a bias. Contrary to our hypothesis, the bias we introduced is larger than the variance of the random forest classifier.

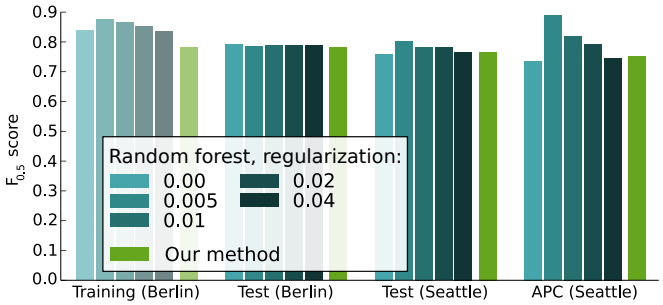


Fig. 10. Performance with incorporating a random forest classifier into our method

VI. Discussion: Path to Generality

Our evaluation shows how much the various components of our perception system for the APC contribute to the overall performance. We argue that these findings force us to question certain assumptions made by generic state-of-the-art robotics perception systems: (i) Machine learning approaches can replace large parts of the hand tuned-parameters and even learn from very limited amounts of data, but only if used correctly. This insight is reflected by the fact that using random forests with our features and post-processing outperforms the hand-tuned pixel probability estimation component (Sec. V-C.4), whereas applying it to RGB-D data directly does not (Sec. V-B). This finding is backed up by decades of applied machine learning research, but it still has not become common practice in all areas of robot perception. (ii) We must tightly connect segmentation and classification of objects, and explictly reason about the environment, not only about the target object. This finding is clearly supported by our analysis of the influence of known objects (Sec. V-A.3) and the contribution of the distance-to-shelf feature (Sec. V-C.1). However, it stands in contrast to many state-of-the art approaches to robotic perception that solve subproblems fully independently, and do not use contextual information. (iii) Missing information is an important source of information, as indicated by the contribution of the missing-3D-points feature (Sec. V-C.1). Although computing and using this information is cheap, few methods in robot perception exploit it. (iv) Our results show that simple visual post-processing can outperform complex reasoning, as exemplified by the fact that Gaussian smoothing on probability images is as effective as optimizing pairwise potentials based on depth-edges (Sec. V-C.3). We believe that this finding has practical implications: the theoretically best method does not necessarily give large – if any – improvement over approximate heuristics. Thus, the decision of choosing a method for a particular problem should be supported by empirical data, and not only be based on theoretical soundness.

VII. Conclusion

We presented and evaluated our approach to multi-class segmentation in the APC setting. This work simplified perception by leveraging information about the specific scenario, e.g. by using tailored features from multiple modalities, restricting the possible results based on the scenario conditions, and by probabilistically integrating many different sources of information. While the resulting method is tailored to one specific domain, we discussed how our approach reveals insights for building more general robot perception systems. We hope that this paper helps to build successful object perception systems for warehouse scenarios, and, more importantly, triggers research towards object recognition that leverages structural constraints that are common across a broader range of robotic tasks.

References

[1] C. Eppner, S. Höfer, R. Jonschkowski, R. Martín-Martín, A. Sieverling, V. Wall, and O. Brock, "Lessons from the Amazon Picking Challenge: Four Aspects of Building Robotic Systems," in *Proceedings of Robotics: Science and Systems*, 2016.

[2] Amazon Inc., "Amazon Picking Challenge - 2015," http://amazonpickingchallenge.org/, May 2015.

[3] N. Correll, K. E. Bekris, D. Berenson, O. Brock, A. Causo, K. Hauser, K. Okada, A. Rodriguez, J. M. Romano, and P. R. Wurman, "Lessons from the Amazon Picking Challenge," *ArXiv e-prints*, Jan. 2016.

[4] C. Rennie, R. Shome, K. E. Bekris, and A. F. De Souza, "A Dataset for Improved RGBD-based Object Detection and Pose Estimation for Warehouse Pick-and-Place," *ArXiv e-prints*, Sep. 2015.

[5] M. Everingham, S. A. Eslami, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes challenge: A retrospective," *International Journal of Computer Vision*, vol. 111, no. 1, pp. 98–136, 2014, 00046.

[6] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision*, pp. 1–42, Apr. 2015, 00215.

[7] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 32, no. 9, pp. 1627–1645, 2010, 03227.

[8] C. Szegedy, A. Toshev, and D. Erhan, "Deep neural networks for object detection," in *Advances in Neural Information Processing Systems*, 2013, pp. 2553–2561, 00072.

[9] J. D. Lafferty, A. McCallum, and F. C. N. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," pp. 282–289, 2001.

[10] J. Shotton, J. Winn, C. Rother, and A. Criminisi, "Textonboost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation," in *Computer VisionECCV 2006*, 2006, pp. 1–15, 00829.

[11] A. C. Müller and S. Behnke, "Learning depth-sensitive conditional random fields for semantic segmentation of rgb-d images," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE, 2014, pp. 6232–6237, 00011.

[12] M. J. Swain and D. H. Ballard, "Color indexing," *International journal of computer vision*, vol. 7, no. 1, pp. 11–32, 1991.

[13] U. Klank, D. Pangercic, R. B. Rusu, and M. Beetz, "Real-time cad model matching for mobile manipulation and grasping," in *Humanoid Robots, 2009. Humanoids 2009. 9th IEEE-RAS International Conference on*. IEEE, 2009, pp. 290–296, 00043.

[14] R. B. Rusu, N. Blodow, and B. Michael, "Fast Point Feature Histograms (FPFH) for 3d Registration," in *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, May 2009, pp. 3212–3217, 00491.

[15] R. B. Rusu, G. Bradski, R. Thibaux, and J. Hsu, "Fast 3d recognition and pose using the viewpoint feature histogram," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*. IEEE, 2010, pp. 2155–2162.

[16] A. Collet, M. Martinez, and S. S. Srinivasa, "The MOPED framework: Object recognition and pose estimation for manipulation," *The International Journal of Robotics Research*, vol. 30, no. 10, pp. 1284 –1306, 2011.

[17] S. Hinterstoisser, C. Cagniart, S. Ilic, P. Sturm, N. Navab, P. Fua, and V. Lepetit, "Gradient response maps for real-time detection of texture-less objects," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2012.

[18] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.