International Journal of Robotics Research (IJRR), vol. 37, number 4-5, pp. 437-451, 2018.

RGB-D Object Detection and Semantic Segmentation for Autonomous Manipulation in Clutter

Journal Title
XX(X):1–17

DOI: 10.1177/ToBeAssigned
www.sagepub.com/
SAGE

Max Schwarz[1], Anton Milan[2], Arul Selvam Periyasamy[1], and Sven Behnke[1]

Abstract

Autonomous robotic manipulation in clutter is challenging. A large variety of objects must be perceived in complex scenes, where they are partially occluded and embedded among many distractors, often in restricted spaces. To tackle these challenges, we developed a deep-learning approach that combines object detection and semantic segmentation. The manipulation scenes are captured with RGB-D cameras, for which we developed a depth fusion method. Employing pretrained features makes learning from small annotated robotic data sets possible. We evaluate our approach on two challenging data sets: one captured for the Amazon Picking Challenge 2016, where our team NimbRo came in second in the Stowing and third in the Picking task, and one captured in disaster-response scenarios. The experiments show that object detection and semantic segmentation complement each other and can be combined to yield reliable object perception.

Keywords

Deep learning, object perception, RGB-D camera, transfer learning, object detection, semantic segmentation

1 Introduction

Robots are increasingly deployed in unstructured and cluttered domains, including households and disaster scenarios. To perform complex tasks autonomously, reliable perception of the environment is crucial. Different tasks may require different levels of cognition. In some cases, it may be sufficient to classify certain structures and objects as obstacles in order to avoid them. In others, a more fine-grained recognition is necessary, for instance to determine whether a specific object is present in the scene. For more sophisticated interaction, such as grasping and manipulating real-world objects, a more precise scene understanding including object detection and pixel-wise semantic segmentation is essential.

Over the past few years, research in all these domains has shown remarkable progress. This success is largely due to the rapid development of deep learning techniques that allow for end-to-end learning from examples, without the need for designing handcrafted features or introducing complex priors. Somewhat surprisingly, there are not many working examples to date that employ deep models in real-time robotic systems. In this paper, we first demonstrate the application of deep learning methods to the task of bin-picking for warehouse automation. This particular problem setting has unique properties: While the surrounding environment is usually very structured—boxes, pallets and shelves—the sheer number and diversity of objects that need to be recognized and manipulated as well as their chaotic arrangement and spatial restrictions pose daring challenges to overcome.

In addition to bin-picking, we also validate our approach in disaster-response scenarios. Contrary to bin-picking, this setting is much less structured, with highly varying and cluttered backgrounds. These scenes may include many unknown objects.

Despite their remarkable success, deep learning methods exhibit at least one major limitation. Due to a large number of parameters, they typically require vast amounts of hand-labeled training data to learn the features required for solving the task. To overcome this limitation, we follow the transfer learning approach (cf. e.g. Girshick et al. (2014); Pinheiro and Collobert (2015); Lin et al. (2017)), where

[1]University of Bonn
[2]University of Adelaide

Corresponding author:
Max Schwarz, University of Bonn, Computer Science Institute VI, Friedrich-Ebert-Allee 144, 53113 Bonn, Germany
Email: max.schwarz@uni-bonn.de

Figure 1. Picking objects from the APC shelf.

the initial layers are pretrained on large collections of related, already available images. These early layers are usually responsible for extracting meaningful features like edges, corners, or other simple structures that are often present in natural images. Then, merely the few final layers that perform the high-level task of detection or segmentation need to be adjusted—or finetuned—to the task at hand. This strategy enables us to exploit the power of deep neural networks for robotic applications with only little amounts of additional training data.

To summarize, our main contributions are as follows.

1. We develop two deep-learning based object perception methods that employ transfer learning to learn from few annotated examples (Section 3).
2. Both deep-learning techniques are integrated into a real-life robotic system (Section 4).
3. We present a simple, yet effective technique to fuse three separate sources of depth information, thereby improving the overall accuracy of depth measurements (Section 4.2, Figure 9). The resulting depth measurements are integrated into both perception methods.
4. Experimental results of the proposed methods are presented on two scenarios, bin-picking (Section 4) and disaster response (Section 5), showing their validity and generality.

Finally, we discuss lessons learned (Section 6) from our preparation and the competition at the Amazon Picking challenge.

2 Related Work

Even though early ideas of deep neural networks date back several decades (Ivakhnenko and Lapa 1966; LeCun et al. 1989; Behnke 2003), it was not until recently that they gained immense popularity in numerous applications including machine translation (Sutskever et al. 2014), speech recognition (Graves et al. 2013), and computer vision (Krizhevsky et al. 2012). Powered by the parallel computing architectures present in modern GPUs as well as the availability of labeled data (Russakovsky et al. 2015; Krishna et al. 2017; Song et al. 2015), deep learning methods now easily outperform traditional approaches in tasks like image classification (Krizhevsky et al. 2012), object detection (Johnson et al. 2016; Liu et al. 2016), or semantic segmentation (Chen et al. 2015; Lin et al. 2017). The core of such methods typically constitutes a multi-layer architecture, where each layer consists of a series of convolutional operations, passed through a non-linear activation function, such as e.g. rectified linear unit (ReLU), followed by spatial maximum pooling (Scherer et al. 2010) to produce local deformation invariance. Perhaps one of the most popular of such networks is the architecture proposed by Krizhevsky et al. (2012), often referred to as AlexNet. Although initially employed for image classification, it has been since adapted to various other tasks including image captioning (Karpathy and Li 2015), pedestrian detection (Girshick 2015), and semantic segmentation (Long et al. 2015). The original AlexNet consists of five convolutional and three fully-connected layers. Other prominent deep networks include the 16-layer VGG (Simonyan and Zisserman 2014), Inception with 22 layers (Szegedy et al. 2015), and more recently the so-called Deep Residual Networks with skip connections and up to 152 layers (He et al. 2016). Other works in the area of semantic segmentation include Badrinarayanan et al. (2015), who use a multi-stage encoder-decoder architecture that first uses maximum pooling to reduce spatial resolution and later upsample the segmentation results using the indices of the local pooling maxima. For object detection, one line of work considers the task as region proposal followed by classification and scoring. Girshick et al. (2014) process external region proposals using RoI pooling to reshape intermediate CNN feature maps to a fixed size. To increase performance, all regions may be processed in a single forward pass (Girshick 2015). Finally, region proposal networks that regress from anchors to regions of interest are integrated into the Faster R-CNN detection framework (Ren et al. 2015).

Our approach is mainly based on two methods: the OverFeat feature extractor introduced by Sermanet et al. (2013), as well as the DenseCap region detection and captioning approach of Johnson et al. (2016). In particular, we adopt both approaches to detect

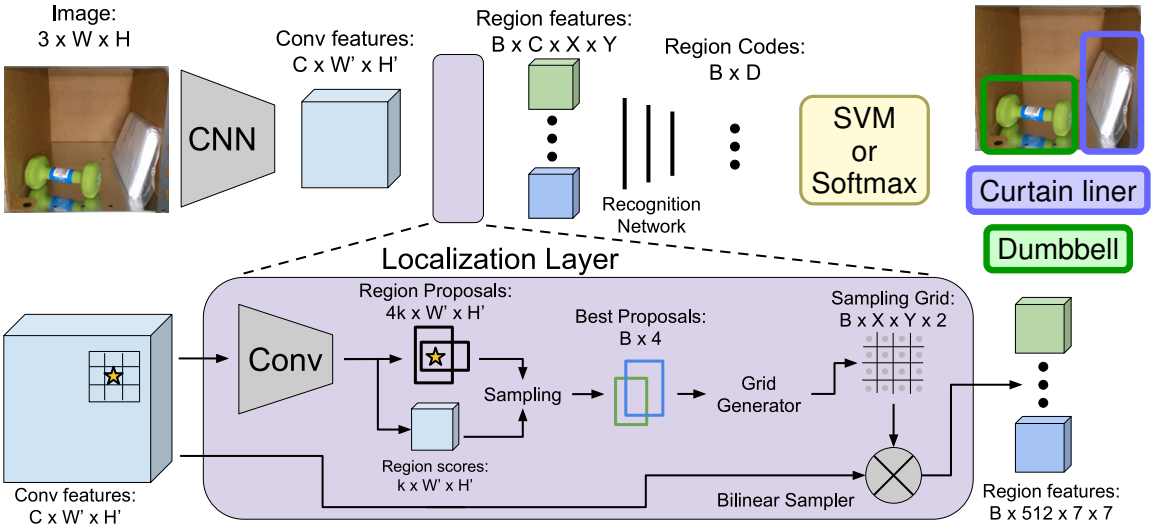


Figure 2. Architecture of the RGB object detection pipeline. Adapted from Johnson et al. (2016). The input size WxH is flexible, for the APC situation we used 720x405. VGG-16 produces C=512 feature maps after the last convolution. B=1000 bounding boxes are extracted using non-maximum suppression.

and segment objects for robotic perception and manipulation in new settings, namely bin picking and disaster-response tasks. To that end, we rely on transfer learning, a method for adapting pre-trained models for a specific task. Note that this training approach is not entirely new and in fact has been followed in several of the above works. For example, R-CNN (Girshick et al. 2014) starts from ImageNet features and fine-tunes a CNN for object detection. Pinheiro and Collobert (2015) exploit large amounts of image-labeled training data to train a CNN for a more fine-grained task of semantic segmentation. Schwarz et al. (2015) used pretrained features and depth preprocessing to recognize RGB-D objects and to estimate their pose. In this work, we apply transfer learning to robotic perception and manipulation scenarios. Recently, combinations of object detection and semantic segmentation have been introduced, such as Mask R-CNN (He et al. 2017), which predicts local segmentation masks for each detected object. In contrast, our work combines results after separate object detection and semantic segmentation, allowing independent training. Nevertheless, combining both approaches in a single network is both elegant and effective.

Bin picking is one of the classical problems in robotics and has been investigated by many research groups in the last three decades, e.g. (Buchholz et al. 2014; Nieuwenhuisen et al. 2013; Pretto et al. 2013; Domae et al. 2014; Drost et al. 2010; Berner et al. 2013; Martinez et al. 2015; Holz et al. 2015; Kaipa et al. 2016; Harada et al. 2016). In these works, often simplifying conditions are exploited, e.g. known parts of one type being in the bin, parts with holes that are easy to grasp by sticking fingers inside, flat parts, parts composed of geometric primitives, well textured parts, or ferrous parts that can be grasped with a magnetic gripper.

During the Amazon Picking Challenge (APC) 2015, various approaches to a more general shelf-picking problem have been proposed and evaluated. Correll et al. (2016) aggregate lessons learned during the APC 2015 and present a general overview and statistics of the approaches. For example, 36 % of all teams (seven of the top ten teams) used suction for manipulating the objects.

Eppner et al. (2016) describe their winning system for APC 2015. Mechanically, the robot consists of a mobile base and a 7-DOF arm to reach all shelf bins comfortably. In contrast, our system uses a larger arm and can thus operate without a mobile base (see Section 4.1). The endeffector of Eppner et al. (2016) is designed as a fixed suction gripper, which can execute top and side picks. Front picks are, however, not possible. The object perception system is described in detail by Jonschkowski et al. (2016). A single RGB-D camera captures the scene. Six hand-crafted features are extracted for each pixel, including color and geometry-based features. The features are then used in a histogram backprojection scheme to estimate the posterior probability for a particular object class. The target segment is found by searching for the pixel with the maximum probability. After fitting a

3D bounding box, top or side grasps are selected heuristically. The team performed very well at APC 2015 and reached 148 out of 190 points.

Yu et al. (2016) reached second place with Team MIT in the APC 2015. Their system uses a stationary industrial arm and a hybrid suction/gripping endeffector. The industrial arm provides high accuracy and also high speed. Similar to our approach, an Intel RealSense sensor mounted on the wrist is used for capturing views of the bin scenes (together with two base-mounted Kinect2 sensors). A depth-only GPU-based instance registration approach is used to determine object poses. Team MIT achieved 88 points in the competition.

In contrast to the first edition, the 2016 Amazon Picking Challenge introduced more difficult objects (e.g. the heavy 3 lb dumbbell), increased the difficulty in the arrangements and the number of items per bin, and introduced the new stowing task. Leitner et al. (2017) introduce a physical picking benchmark to ease reproducibility in warehouse automation research and present their own approach as a baseline.

Hernandez et al. (2016) reached first place in both the picking and the stowing task in the APC 2016. Their system consists of a large industrial 7-DOF arm mounted on a horizontal rail, resulting in eight degrees of freedom in the arm and base. The gripper is a complex custom design, allowing both suction and pinch grasps. Like our design described in Section 4.1, the suction cup can be bent to facilitate top, side, and frontal grasps. Object perception is based on RGB-D measurements from an Ensenso 3D camera. The authors report problems with reflections and noise, and therefore built in heuristics to reject false registrations. In contrast, we added a second RGB-D camera to be able to filter out false measurements (see Section 4.2). Similarly to our architecture, object detection is carried out using an approach based on Faster R-CNN (Ren et al. 2015). After detection, object poses are estimated using a point cloud registration method. For grasp planning, primitive shapes are fitted to find grasp candidate spots. Candidates are then filtered using reachability measures. For deformable objects, a simple measurement-based heuristic is used, similar to our approach.

Zeng et al. (2017) describe the approach of Team MIT-Princeton, who performed successfully in the APC 2016 reaching 3rd and 4th place. Like Hernandez et al. (2016), a high-precision industrial robot arm with a combination of a two-finger gripper and a suction cup was used for object manipulation. The perception pipeline included segmentation and 6D pose estimation of objects based on a multi-view point cloud. Segmentation is addressed using a VGG-type CNN (Simonyan and Zisserman 2014) in each of the 15-18 RGB-D frames, which are then combined to produce a semantic 3D point cloud. Pose estimation is then performed on the dense point cloud by aligning 3D-scanned objects using a modified ICP algorithm. Note that over 130,000 training images were used for their system, which is about three orders of magnitude more than in our approach.

3 Methods

For perceiving objects in the vicinity of a robot, we developed two independent methods. The first one solves the object detection problem, i.e. outputs bounding boxes and object classes for each detection. The second one performs semantic segmentation, which provides a pixel-wise object category labeling.

Since training data and time are limited, it is crucial not to train from scratch. Instead, both methods leverage convolutional neural networks (CNNs) pre-trained on large image classification datasets and merely adapt the networks to work in the specific domain.

3.1 Object Detection

We extend an object detection approach based on the DenseCap network (Johnson et al. 2016). DenseCap approaches the problem of dense captioning, i.e. providing detailed textual descriptions of interesting regions (bounding boxes) in the input image. Figure 2 shows the general architecture of the DenseCap network. In a nutshell, the network first extracts CNN features and samples a fixed number of proposals (1000 in our case) using an objectness score within a region proposal network. Since the sampled regions can be of arbitrary size and shape, the intermediate CNN feature maps are interpolated to fit a fixed size for each proposal. The proposals are then classified using a recognition network. In contrast to other recent works on object detection (e.g. Faster R-CNN, Ren et al. 2015), the DenseCap architecture can be trained end-to-end without the need for approximations, since its bilinear interpolation layer is fully differentiable. The underlying VGG-16 CNN (Simonyan and Zisserman 2014) used for feature extraction was pretrained on the ImageNet (Russakovsky et al. 2015) dataset. Afterwards, the entire pipeline was trained end-to-end on the Visual Genome dataset (Krishna et al. 2017).

Obviously, textual descriptions of image regions are not relevant for bin-picking scenarios. However, these captions are generated from an intermediate feature vector representation, which is highly descriptive and

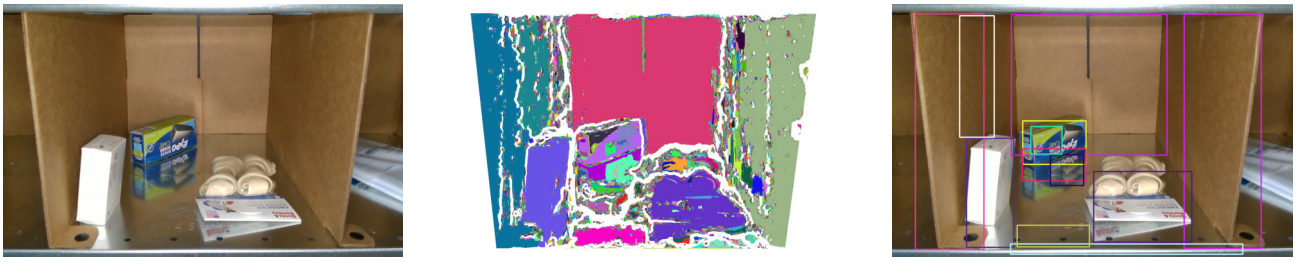


Figure 3. RGB-D based additional region proposals. Left: RGB frame. Center: Regions labeled using the connected components algorithm. Right: Extracted bounding box proposals.

therefore should not be simply ignored. To exploit the power of this feature representation, we use the network up until the captioning module for feature extraction, and replace the language processing part by a classification component that uses a linear support vector machine (SVM). As an alternative, we investigate a soft-max classifier layer, which allows us to fine-tune the network during training.

3.1.1 Linear SVM. In the first case, we remove the language generation model and replace it with a linear SVM for classification. We also introduce two primitive features based on depth: The predicted bounding box is projected into 3D using the depth value of its center. The metric area and size are then concatenated to the CNN features. Since linear SVMs can be trained very efficiently, the training can happen just-in-time before actual perception, exploiting the fact that the set of possible objects in the bin is known. Note that restricting the set of classes to the ones present in the current scene also has the side-effect that training time and memory usage are constant with respect to the set of all objects present in a warehouse. This allows us to potentially scale this approach to an arbitrarily large number of object categories.

The SVM is used to classify each predicted bounding box. To identify a single output, the bounding box with the maximum SVM response is selected. This ignores duplicate objects, but since the goal is to retrieve only one object at a time, this reduction is permissible.

3.1.2 Softmax Classification. For finetuning the network, we use a soft-max classification layer instead of the SVM. All layers except the initial CNN layers (see Fig. 2) are optimized for the task at hand. Contrary to SVM classification, the softmax layer predicts confidences over all object classes. In the bin-picking scenario, the bounding box with the highest score in the desired object class is produced as the final output, i.e. the object to pick.

3.1.3 Incorporating Depth. The existing object detection network does not make use of depth measurements. Here, we investigate several methods for incorporating depth into the network.

As with all architectures based on R-CNN, it is straightforward to classify bounding boxes generated from an external proposal generator. One way to include depth information is therefore to use an external RGB-D proposal generator. To this end, we augment the network-generated proposals with proposals from a connected components algorithm running on the RGB and depth frames (see Fig. 3). Two pixels are deemed connected if they do not differ more than a threshold in terms of 3D position (5 mm), normal angle (50°), saturation, and color (10). Final bounding boxes are extracted from regions with an area above a predefined threshold (10,000 pixels for 1920×1080 input).

A second possibility is to treat depth as an additional mid-level feature. For this purpose, we use the popular three-channel HHA encoding (Gupta et al. 2014), which augments depth with two geometric features (height above ground and angle to gravity). We downsample the HHA map and concatenate it to the feature maps generated by the pretrained first convolutional layers (see Fig. 4). Furthermore, we can also use the same pretrained CNN to extract higher-level features from HHA, as shown in Fig. 5.

Finally, Gupta et al. (2016) propose to use an RGB reference network to generate the training data needed for the other modality, a technique they call Cross Modal Distillation. In essence, the pretrained RGB network ϕ computes a feed-forward pass on the RGB frame I_s, generating the target feature maps $\phi(I_s)$. A back-propagation step then trains the depth network ψ to imitate these features on the corresponding depth frame I_d, minimizing the objective

$$\min_{W_d} \sum_{(I_s,I_d)\in U_{s,d}} ||\psi(I_d) - \phi(I_s)||^2, \qquad (1)$$

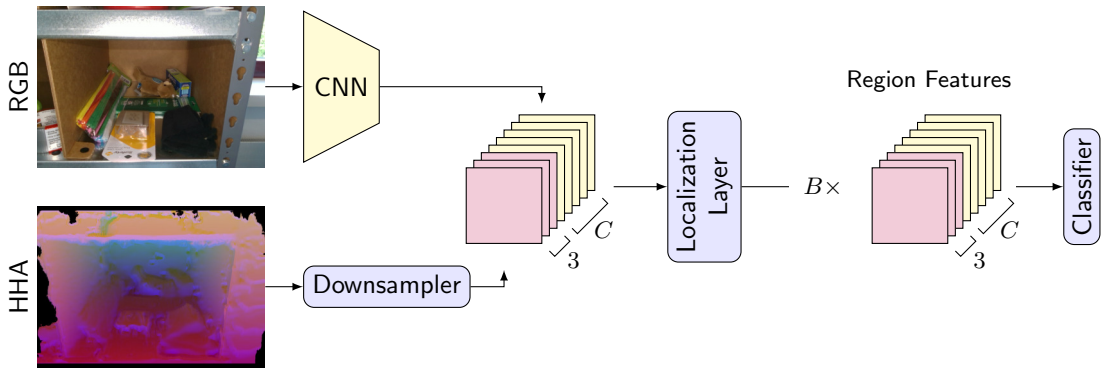


Figure 4. Detection pipeline with CNN features from RGB and downsampled HHA-encoded depth. C denotes the number of CNN feature maps after the last convolutional layer (512 for VGG-16). The internal proposal generator produces B proposals (1000).

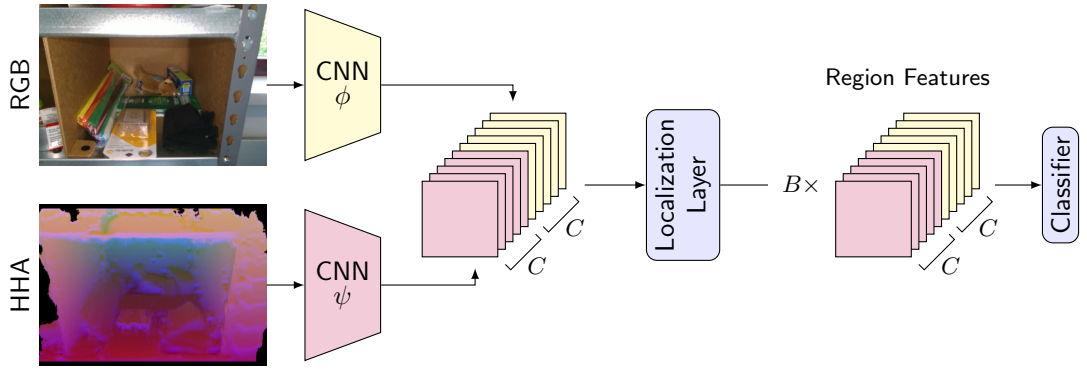


Figure 5. Detection pipeline with concatenated CNN features from RGB and HHA-encoded depth. C denotes the number of CNN feature maps after the last convolutional layer (512 for VGG-16). The internal proposal generator produces B proposals (1000). For the Cross Modal Distillation approach, CNN ψ is trained to imitate the pretrained CNN ϕ.

where W_d are the weights of the depth network, and $U_{s,d}$ is the training set of RGB and depth frame pairs. Note that no annotation is necessary on $U_{s,d}$, so any RGB-D video (ideally of the target domain) can be used to perform the supervision transfer. In our case, the (annotated) training set is used for distillation, since additional unlabeled RGB-D sequences of the target domain are not available.

After the initial Cross Modal Distillation training, the trained network can be used in place of the RGB network for depth feature extraction (see Fig. 5).

3.1.4 Implementation Details. As in the original DenseCap work, the ADAM optimizer (Kingma and Ba 2015) is used for training the network with parameters $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$. However, we adapt a custom learning rate schedule to dampen training oscillations at the end of training (annealing): The learning rate starts at $1 \cdot 10^{-5}$ and is kept constant for 15 epochs, then linearly lowered to $1 \cdot 10^{-6}$ during the next 85 epochs. At 200 epochs, the rate is lowered to $5 \cdot 10^{-7}$, and finally to $1 \cdot 10^{-7}$ at 250 epochs. To prevent overfitting, 50% dropout is used throughout the entire network. As in the original DenseCap pipeline, input images are scaled such that the longest side has 720 pixels.

3.2 Semantic Segmentation

Manipulation of real-world objects requires a precise object localization. Therefore, we also investigated pixel-level segmentation approaches in the context of robot manipulation tasks. As opposed to object detection, which only provides a rather coarse estimate of object location in terms of a bounding box, segmentation offers a much more detailed representation by classifying each pixel as one of the known categories.

Our segmentation network is based on the work of Husain et al. (2016) and consists of six convolution layers, as shown in Figure 7. Only the last three layers of the network are trained on the domain specific dataset. For RGB data, the first two convolutional layers from the OverFeat network (Sermanet et al.

(a) RGB frame (b) Water bottle (c) Glue (d) Bubble mailer (e) Gloves

Figure 6. DenseCap probability estimates for an example frame. The objectness in each pixel is approximated by rendering a Gaussian that corresponds to the bounding box center and extent.

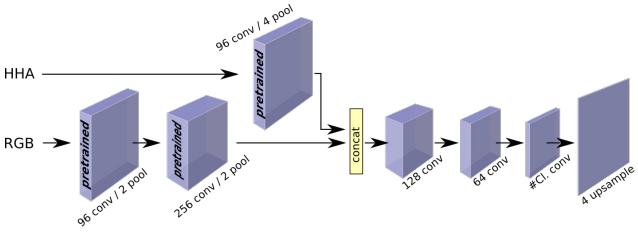


Figure 7. Our network architecture for semantic object segmentation.

2013) are applied. OverFeat was trained on the ImageNet dataset (Russakovsky et al. 2015) consisting of over one million images, each of which was labeled with one of 1000 available categories. For depth data, we also use the HHA encoding, similar to the object detection pipeline. Since the OverFeat network was trained in the RGB domain, and its higher-level features are not meaningful on HHA, we only use the first convolutional layer of the network for HHA. Still, CNN pretraining on RGB can be used on HHA, as demonstrated by Gupta et al. (2014). The extracted RGB and HHA features are then concatenated. The dimension of the final convolution layer is adjusted for each task to reflect the number of classes (#Cl.) present in the dataset.

3.2.1 Implementation Details. The network is fine-tuned using stochastic gradient descent. We follow Husain et al. (2016) and set the learning rate to 10^{-3}, use a momentum of 0.9, and a 50% dropout in the final three layers. The learning rate is set to decay over time at a rate of 10^{-4}.

Both detection and segmentation methods were implemented using the Torch7 deep learning framework* and integrated into a real-time robotics system based on ROS (Quigley et al. 2009). We hope that our implementation, which is publicly available†, will lower the burden for other researchers to apply deep learning methods for robotic tasks.

3.3 Combining Detection and Segmentation

During the APC competition, we used a combination of the SVM object detection approach and the semantic segmentation. In particular, the bounding boxes predicted by the object detection network were rendered with a logistic estimate of their probability and averaged over all classes. This process produced a "probability map" that behaved similar to a pixel-wise posterior. In the next step, the detection probability map was multiplied element-wise with the class probabilities determined in semantic segmentation. A per-pixel max-probability decision then resulted in the final segmentation mask used in the rest of the pipeline.

After the APC, we replaced the hard bounding box rendering with a soft Gaussian whose mean and covariance were derived from the box location and size, respectively, cf. Fig. 6 for an illustration. This yielded better results, because such a representation matches the actual object shape more closely than an axis-aligned bounding box. The Gaussian blobs for all detections are accumulated and the resulting map P_{det} is normalized, i.e. scaled so that the maximum equals to one. To allow for detection mistakes, we introduce a weak prior that accounts for false negatives. The final combined posterior is computed as

$$P_{\text{combined}} = P_{\text{seg}}(0.1 + 0.9P_{\text{det}}), \tag{2}$$

where P_{seg} is the posterior resulting from semantic segmentation and P_{det} is the estimated posterior from object detection.

While this combination is relatively straightforward (note that the product assumes conditional independence) and the shape approximations by Gaussian masks are rather coarse, this strategy yields a consistent increase in performance nonetheless (see Section 4.4).

*http://torch.ch
†http://ais.uni-bonn.de/apc2016/

4 Application to Bin-Picking

In July 2016, in conjunction with RoboCup, Amazon held the second Amazon Picking Challenge (APC)[‡], which provided a platform for comparing state-of-the-art solutions and new developments in bin picking and stowing applications. The challenge consisted of two separate tasks, where contestants were required to pick twelve specified items out of chaotically arranged shelf boxes—and to stow twelve items from an unordered pile in a tote into the shelf. Amazon provided a set of objects from 39 categories, representing a large variety of challenging properties, including transparency (e.g. water bottle), shiny surfaces (e.g. metal or shrink wrap), deformable materials (e.g. textiles), black and white textureless surfaces which hamper reliable depth measurements, heavy objects, and mesh-like objects with holes that could not be grasped by using suction alone. Moreover, the shiny metal floors of the shelf boxes posed a considerable challenge to the perception systems, as all objects are also visible through their mirrored image. Before the run, the system was supplied with a task file that specified which objects should be picked as well as with all object locations (which shelf box they are stored in). After the run, the system was expected to output the new locations of the items. For completeness, before presenting our results we will briefly describe our mechatronic design as well as motion generation and grasp selection methods that were developed for the APC competition.

4.1 Robotic System

Our robot consists of a 6-DOF arm, a 2-DOF endeffector, a camera module, and a suction system. To limit system complexity, we chose to use a stationary manipulator. This means the manipulation workspace has to cover the entire shelf, which places constraints on the possible robotic arm solutions. In our case, we chose the UR10 arm from Universal Robotics, because it covers the workspace sufficiently, is cost-effective, lightweight, and offers safety features such as an automatic (and reversible) stop upon contact with the environment.

Attached to the arm is a custom-built endeffector (see Fig. 8). For reaching into the deep and narrow APC shelf bins, we use a linear actuator capable of 37 cm extension. On the tip of the linear extension, we mounted a rotary joint to be able to carry out both front and top grasps.

For grasping the items, we decided to employ a suction gripper. This choice was motivated by the large success of suction methods during the APC 2015 (Correll et al. 2016), and also due to the presented set of objects for the APC 2016, most of which could be manipulated easily using suction. Our suction system is designed to generate both high vacuum *and* high air flow. The former is needed to lift heavy objects, the latter for objects on which the suction cup cannot make a perfect vacuum seal.

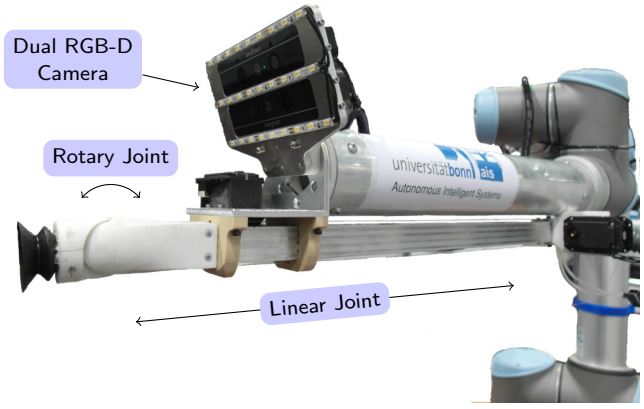
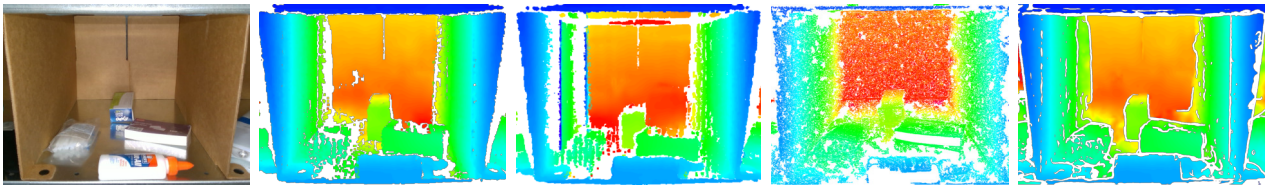


Figure 8. Our dual-camera setup on the UR10 arm used for the Amazon Picking Challenge 2016.

The suction cup has a 3 cm diameter. For most objects, it is sufficient to simply place the suction cup anywhere on the object. For simple arrangements, this suction pose could be inferred from a bounding box. For more complex arrangements with occlusions, there is an increased risk of retrieving the wrong object. Similarly, very small objects can be easily missed. For these reasons, a high-quality localization such as offered by pixel-wise semantic segmentation is required. The final suction pose is derived using two different heuristics (top- or center grasp) operating on the segmentation mask depending on the object height. A nullspace-optimizing IK solver and keyframe interpolation are used to generate motions, similar to the one described in Schwarz et al. (2017b). Further details on robot motion control and grasp generation are described by Schwarz et al. (2017a).

For control and computations, two computers are connected to the system. The first, tasked with high- and low-level control of the robot, is equipped with an Intel Core i7-4790K CPU (4 GHz). The second is used for vision processing, and contains two Intel Xeon E5-2670 v2 (2.5 GHz) and four NVIDIA Titan X GPUs. For training, all four GPUs can be used to accelerate training time. At test time, two GPUs are used in parallel for the two deep learning approaches: object detection and semantic segmentation (see Section 3).

[‡] `http://amazonpickingchallenge.org/`



(a) RGB frame (b) Upper depth (c) Lower depth (d) Stereo depth (e) Fused result

Figure 9. RGB-D fusion from two sensors. Note the corruption in the left wall in the lower depth frame, which is corrected in the fused result.

4.2 RGB-D Preprocessing

After experimenting with multiple sensors setups in the APC setting, we decided to use the Intel RealSense SR300 RGB-D sensor due to its low weight, high resolution, and short-range capabilities. However, we noticed that the depth sensor produced systematic artifacts on the walls of the shelf (cf. Figure 9(c)). The artifacts seem to depend on the viewing angle, i.e. they were present only on the right side of the image. To rectify this situation, we designed a dual sensor setup, with one of the sensors rotated 180° (see Fig. 8). To reduce the undesirable effect of external illumination, we used local LED lighting while capturing the data.

Using two separate sensors also makes a second RGB stream available. To exploit this, we also compute dense stereo disparity between the two RGB cameras using LIBELAS (Geiger et al. 2010). The three depth streams are projected into one common frame (the upper camera in our case) and are then fused using a linear combination with predefined weights α_i[§]. In particular, the stereo stream is fused using a low weight, since the depth measurements of the SR300 cameras are usually more precise (but not always available). Finally, we distrust measurements where the different depth sources disagree by introducing an additional "spread" weight w. In summary, we obtain the following equations for combining depth measurements D_i of a single pixel to a depth measurement D and weight w:

$$D = \frac{\sum \alpha_i D_i}{\sum \alpha_i}, \tag{3}$$

$$w = \exp(-(\max_i D_i - \min_i D_i)). \tag{4}$$

Pixels where $\max_i D_i - \min_i D_i > 5\,\text{cm}$ are disregarded entirely. Figure 9 shows an exemplary scene with individual raw sensor measurements as well as the fused depth map.

Since the resulting fused depth map is usually sparse, we need to fill in the missing data. We follow the work of Ferstl et al. (2013), who upsample depth images guided by a high-resolution grayscale image. In contrast to many other guided upsampling approaches, this one does not assume a regular upsampling grid. Instead, any binary (or even real-valued) weight matrix can be used to specify the location of source pixels in the output domain. This makes the approach applicable to our scenario, where the mask of valid pixels has no inherent structure.

The upsampling is formulated as an optimization problem, minimizing the energy term

$$\min_{u,v} \left\{ \alpha_1 \int_{\Omega_H} |T^{\frac{1}{2}}(\Delta u - v)|\,\mathrm{d}x + \alpha_0 \int_{\Omega_H} |\Delta v|\,\mathrm{d}x + \int_{\Omega_H} w|(u - D_s)|^2\,\mathrm{d}x \right\}, \tag{5}$$

where the first two summands regularize the solution using Total Generalized Variation, and the last summand is the data term. For details about the problem formulation and the solver algorithm, we refer the reader to Ferstl et al. (2013). The guided upsampling was implemented in CUDA to achieve near real-time performance (<100 ms per image).

4.3 Overall Results

The system performed both the picking and stowing task successfully during the APC 2016.

4.3.1 Stowing task. Our system was able to stow eleven out of twelve items into the shelf.[¶] However, one of the successfully stowed items was misrecognized: Our approach falsely identified a whiteboard eraser as a toothbrush and placed it into the shelf. This meant that the system could not recognize the toothbrush as the final item remaining in the tote. This unlikely event was expected to be caught by a built-in fallback mechanism which would attempt to recognize all known objects. However, this mechanism failed because the only remaining object was thin and therefore discarded based on a size threshold. The

[§]Our experiments use $\alpha_{\text{stereo}} = 0.1$ and $\alpha_{\text{RGB-D}} = 40.0$.

[¶]Video at `https://youtu.be/B6ny9ONfdx4`

Table 1. Picking Run at APC 2016

Bin	Item	Pick	Drop	Report
A	duct tape	×	×	×
B	bunny book	✓	✓	×[2]
C	squeaky eggs	✓	×	✓
D	crayons[1]	✓	×	✓
E	coffee	✓	✓	×[2]
F	hooks	✓	×	✓
G	scissors	×	×	×
H	plush bear	✓	×	✓
I	curtain	✓	×	✓
J	tissue box	✓	×	✓
K	sippy cup	✓	×	✓
L	pencil cup	✓	✓	×[2]
	Sum	10	3	7

[1] Misrecognized, corrected on second attempt.
[2] Incorrect report, resulting in penalty.

misrecognition of the item led to the attainment of the second place in the stow task.

4.3.2 Picking task. In the picking task, our system picked ten out of twelve items.[‖] Despite the high success rate (the winning Team DELFT achieved a success pick-up rate of only nine items), only a third place was achieved as a consequence of dropping three items during picking. Note that we could correctly recognize that the objects were not picked successfully using the air velocity sensor of our robot. However, the system incorrectly deduced that the items were still in the shelf, when they actually dropped over the ledge and into the tote. Since the system was required to deliver a report on the final object locations, the resulting penalties reduced our score from 152 points to 97 points—just behind the first and second place of Team DELFT and PFN, both of which achieved 105 points.

4.4 Object Detection and Semantic Segmentation

In addition to the system-level evaluation at the APC, we evaluated our perception approaches on our own annotated dataset, which was also used for training the final APC model. The dataset contains 201 shelf frames, and 132 tote frames. The frames vary in the number of objects and location in the shelf. As far as we are aware, this number of frames is quite low in comparison to other teams (e.g. Team Princeton-MIT, Zeng et al. 2017), which highlights the effectiveness of our transfer learning approach. Figure 10 shows an exemplary scene from the dataset with object detection and segmentation results.

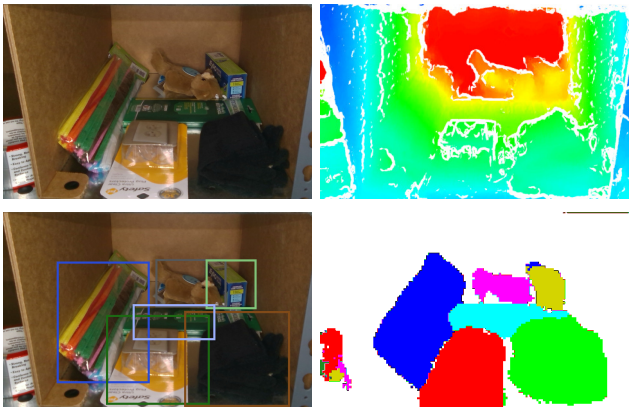


Figure 10. Object perception example. Upper row: Input RGB and depth frames. Lower row: Object detection and semantic segmentation results (colors are not correlated).

For evaluation, we define a five-fold cross validation split on the shelf and tote datasets. To ensure that the examples for each class are distributed as evenly across the splits as possible, we use Iterative Stratified Sampling (Sechidis et al. 2011).

Please note that the DenseCap pipeline has been thoroughly evaluated on the Visual Genome Dataset by Johnson et al. (2016) and the semantic segmentation network has been evaluated on the NYU Depth v2 dataset by Husain et al. (2016). Both achieved state-of-the-art results on the respective datasets.

4.4.1 Object detection. Traditionally, object detectors are evaluated using a retrieval metric like *mean Average Precision (mAP)*, which measures the quality of ranked results over the whole test set. Usually, an Intersection-over-Union (IoU) threshold of 0.5 is chosen, focusing on detection and rough localization instead of precise localization. Generally, the mAP metric as defined above places greater weight on correct detection than on precise localization. Indeed, it is much easier to achieve perfect average precision scores than perfect localization precision.

To also provide location sensitivity, one can define a metric for object detection based on pixel-level precision and recall. In this work, we consider the use case for an object detector in the context of warehouse automation: It is known that a particular object resides in a particular shelf bin, and we need to retrieve it. Here, we are only interested in the detection i with maximum confidence c_i for this object class. We

[‖] Video at `https://youtu.be/q9YiD80vwDc`

Table 2. Evaluation of object detection architectures on the shelf dataset. The mAP object detection score and the F1 localization score are shown for each architecture.

		mAP		
Input	Variant	Uninf.	Inf.	F1
RGB	SVM (plain)	–	0.288	0.685
RGB	SVM (tailor)	–	0.289	0.684
RGB	Softmax (no augmentation)	0.860	0.890	0.769
RGB	Softmax (with augmentation)	0.865	0.896	0.771
RGB-D (TGV)	HHA Features (Fig. 4)	0.865	0.898	0.776
RGB-D (TGV)	Ext. Proposals	0.870	0.898	0.775
RGB-D (TGV)	HHA CNN (Fig. 5)	0.865	0.901	0.790
RGB-D (TGV)	Distillation	**0.878**	**0.911**	**0.798**
RGB-D (single)[1]	Distillation	0.865	0.901	0.787
RGB-D (DT)[2]	Distillation	0.875	0.903	0.788

[1] Without depth fusion, only from upper camera. Filled using TGV method.
[2] Old filling method used during APC 2016 based on a color-guided smoothing filter.

measure its precision and recall as follows:

$$\text{precision} = \frac{|B \cap G|}{|B \cup G|} = \text{IoU}(B, G) \tag{6}$$

$$\text{recall} = \frac{|B \cap G|}{|G|}, \tag{7}$$

where B is the detected bounding box and G denotes the closest ground truth bounding box. Note that a complete mislocalization results in zero precision and recall. Both quantities are then combined into the final F1 score:

$$\text{F1} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}. \tag{8}$$

Table 2 shows the influence of the design choices described in Section 3.1. Both mAP (informed and uninformed case) and the custom F1 metric are shown for each architecture. As a first result we note that the softmax variant with its ability to finetune the entire network including region proposal is far superior to a fixed network paired with an SVM classifier. In particular, the SVM classifier is bad at ranking the detections across images, which is evident in the mAP metric. A calibration step (e.g. Platt scaling) could improve this behavior. Note that the F1 score, which is more relevant in the APC scenario, is closer to the rest of the methods, but still suboptimal. Training the SVM on-the-fly for just the items in the current shelf bin ("tailor" variant) makes little difference in both metrics. All remaining tests were performed with the superior softmax classifier. Data augmentation (image mirroring) slightly improves performance, so all other tests were performed with augmentation.

As expected, incorporating depth measurements results in increased performance. The external RGB-D proposal generator performs better than the naive HHA concatenation. However, reusing the RGB CNN for depth feature computation outperforms the proposal generator. Finally, training a depth CNN using Cross Modal Distillation gives the best results.

We also investigate the depth fusion method described in Section 4.2. The TGV-regularized method is superior to a simple color-guided smoothing filter. The advantage of fusing the two camera streams can be seen. Table 4 shows the final object detection results on the shelf and tote datasets.

4.4.2 Semantic Segmentation. For segmentation, pixel-level precision and recall are calculated. Resulting F1 scores are shown in Table 5. Knowledge of the set of possible objects improves the performance slightly but consistently. The chosen HHA encoding of depth is far superior to raw depth ($\sim 7\%$ increase). Finally, the combination of the finetuned object detector and the semantic segmentation yields a small but consistent increase in performance.

Figure 11 shows the distribution of difficulty across all object classes. Both methods struggle mostly with small and/or shiny objects (tooth brush, scissors), the semantic segmentation even more so, reaching an F1 score below 0.5. We believe that both methods are affected by insufficient image resolution and annotation errors, which result in a greater effect on objects of smaller size. The object detection also struggles

Table 3. Perception runtimes.

Phase	Object detection				Segmentation
	RGB-D proposal	SVM	Softmax (RGB)	Softmax (RGB-D)	
Train	-	-	45 min	4.5 h	≈ 5 h
Test	1006 ms	3342 ms[1]	340 ms	400 ms	≈ 900 ms

[1] Includes just-in-time SVM training

Table 4. Final object detection results on the APC dataset.

	mAP		F1
Dataset	Uninformed	Informed	
Shelf	0.878	0.912	0.798
Tote	0.870	0.887	0.779

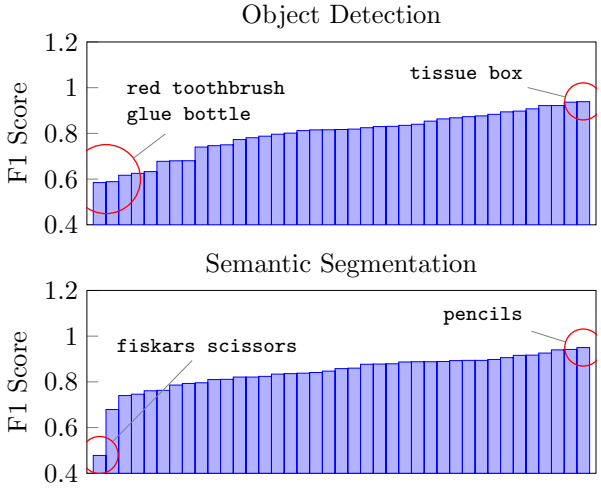


Figure 11. F1 score distribution over the objects for object detection (top) and semantic segmentation (bottom). Results are averaged over the cross validation splits. For object detection, the best RGB-D model is used.

Table 5. F1 scores for semantic segmentation.

	Shelf		Tote	
Method	Uninf.	Inf.	Uninf.	Inf.
Raw depth	0.713	0.735	-	-
HHA depth	0.780	0.813	0.817	0.839
Det+Seg[1]	0.795	0.827	0.831	0.853

[1] Object Detection + Segmentation.

with elongated shapes (e.g. toothbrush), where the approximation of the contour as a bounding box may be deficient under certain viewing angles.

We also measured the runtime of the different modules in our setup (see Table 3). The training time for the RGB-D object detection pipeline is much longer, due to high memory utilization on the GPU. This could probably be improved e.g. through precomputation of the initial pretrained layers, however training time was not taken into consideration in our application. At test time, all approaches achieve sufficient runtimes (≤ 1 s) for bin-picking applications. Note that object detection and semantic segmentation usually run in parallel.

5 Application to Disaster Response

To show the general applicability of the developed pipeline in other contexts, we also evaluate it in a second, unrelated domain. Note that the pipeline was initially designed for bin-picking perception, and is now merely adapted to this new scenario.

The dataset that was used for validating our approach was captured in the European CENTAURO** project, which aims to develop a human-robot symbiotic system for disaster response. To reduce the work load of the human operator, several perception and manipulation capabilities should be done autonomously by the robot system. For example, the robot should be able to identify commonly used tools like wrenches or drillers, correctly grasp and utilize them. Here, we apply the techniques that were used for bin-picking during the APC to the disaster response scenario. The dataset consists of 127 manually annotated RGB-D frames, captured in a cluttered mechanics workshop with six different object classes: Five mechanic tools (clamp, driller, extension box, stapler, wrench) and door handles. The dataset was captured with a Kinect version 2 camera and annotated manually with a tool developed in-house. Figure 12 shows a screenshot of the annotation tool and examples of three annotated frames. In addition to the unique setting, the dataset differs from common RGB-D datasets by offering pixel-wise labeling, highly cluttered background, and the high capture resolution of the Kinect v2 camera.

**https://www.centauro-project.eu

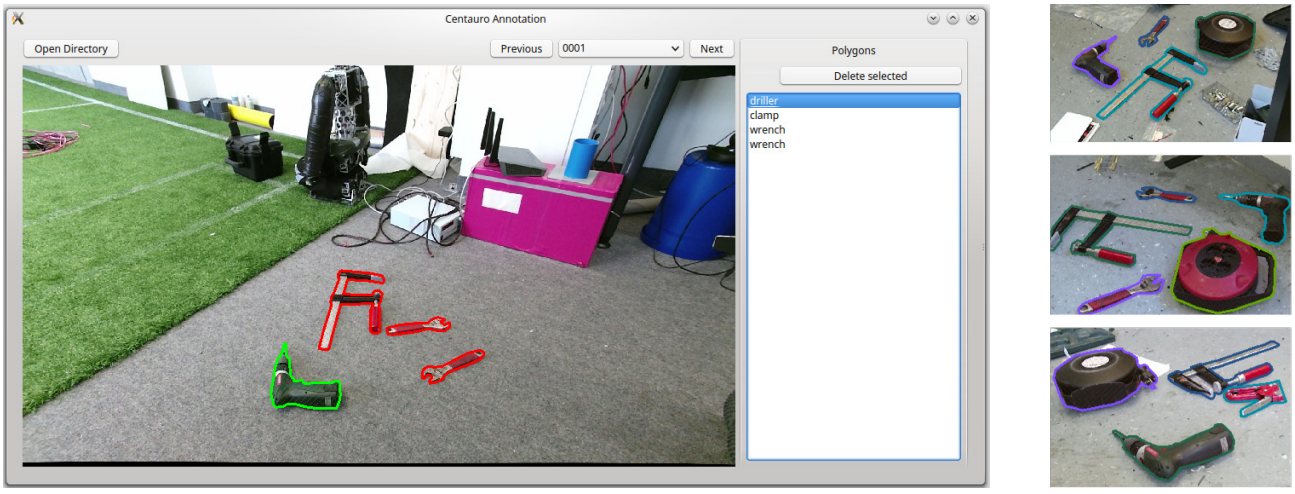


Figure 12. Our annotation tool (left) and three exemplar cropped frames from the captured dataset (right).

Table 6. Object detection results on the CENTAURO dataset.

Resolution	Clamp	Door handle	Driller	Extension	Stapler	Wrench	Mean
	AP / F1	AP / F1	AP / F1	AP / F1	AP / F1	AP / F1	AP / F1
720×507	0.881/0.783	0.522/ 0.554	0.986/0.875	1.000/0.938	0.960/0.814	0.656/0.661	0.834/0.771
1080×760	0.926/0.829	0.867/ 0.632	0.972/0.893	1.000/0.950	0.992/0.892	0.927/0.848	0.947/0.841
1470×1035	0.913/0.814	0.974/ 0.745	1.000/0.915	1.000/0.952	0.999/0.909	0.949/0.860	0.973/0.866

Table 7. Class-wise pixel classification F1 score of the semantic segmentation method in disaster-response environments.

Clamp	Door handle	Driller	Extension	Stapler	Wrench	Background	Mean
0.727	0.751	0.769	0.889	0.775	0.734	0.992	0.805

As in Section 4.4, we define a five-fold cross validation using Iterative Stratified Sampling (Sechidis et al. 2011) for evaluating object detection and semantic segmentation. Since Kinect v2 does not measure depth in the margin areas (especially left and right borders) of the 1920 × 1080 RGB image, we crop the frames to the area where depth measurements are available (1470 × 1035).

5.1 Detection and Segmentation

The object detection pipeline is identical to the one described in Section 3.1. However, since there are frames in the dataset that are either very cluttered, making it harder to robustly estimate a ground plane, or have no ground plane in the image at all (see Fig. 13, bottom), we do not use the geometric HHA features here. Instead, we train the depth CNN using Cross Modal Distillation on raw depth. The segmentation network is also very similar to the one illustrated in Fig. 7. Here, we feed raw depth (replicated to the three input channels) instead of HHA into the depth branch.

5.2 Evaluation

Object detection results for the CENTAURO dataset are shown in Table 6. With the configuration from the APC dataset, the detector shows an acceptable mAP of 83.4%. However, by default the DenseCap pipeline scales the input images such that the largest side is 720 pixels—reducing the available resolution. Since there are very small objects that occupy as little as 4% of the image width (e.g. door knobs, see Fig. 13, bottom row), higher resolutions increase the performance by a large margin (see Table 6), of course at the cost of training and prediction time. Increasing the input size to the highest available resolution (1470×1035) yields near-perfect detection performance (97.3% mAP). This increase is equally visible in the localization score, suggesting that mis-localization (and not mis-detection) is the main cause for the low mAP at lower resolution. The full resolution model takes about 10 h to train and has longer prediction times of around 1 s per image. An intermediate resolution yields 94.7% mAP with a prediction time of 550 ms.

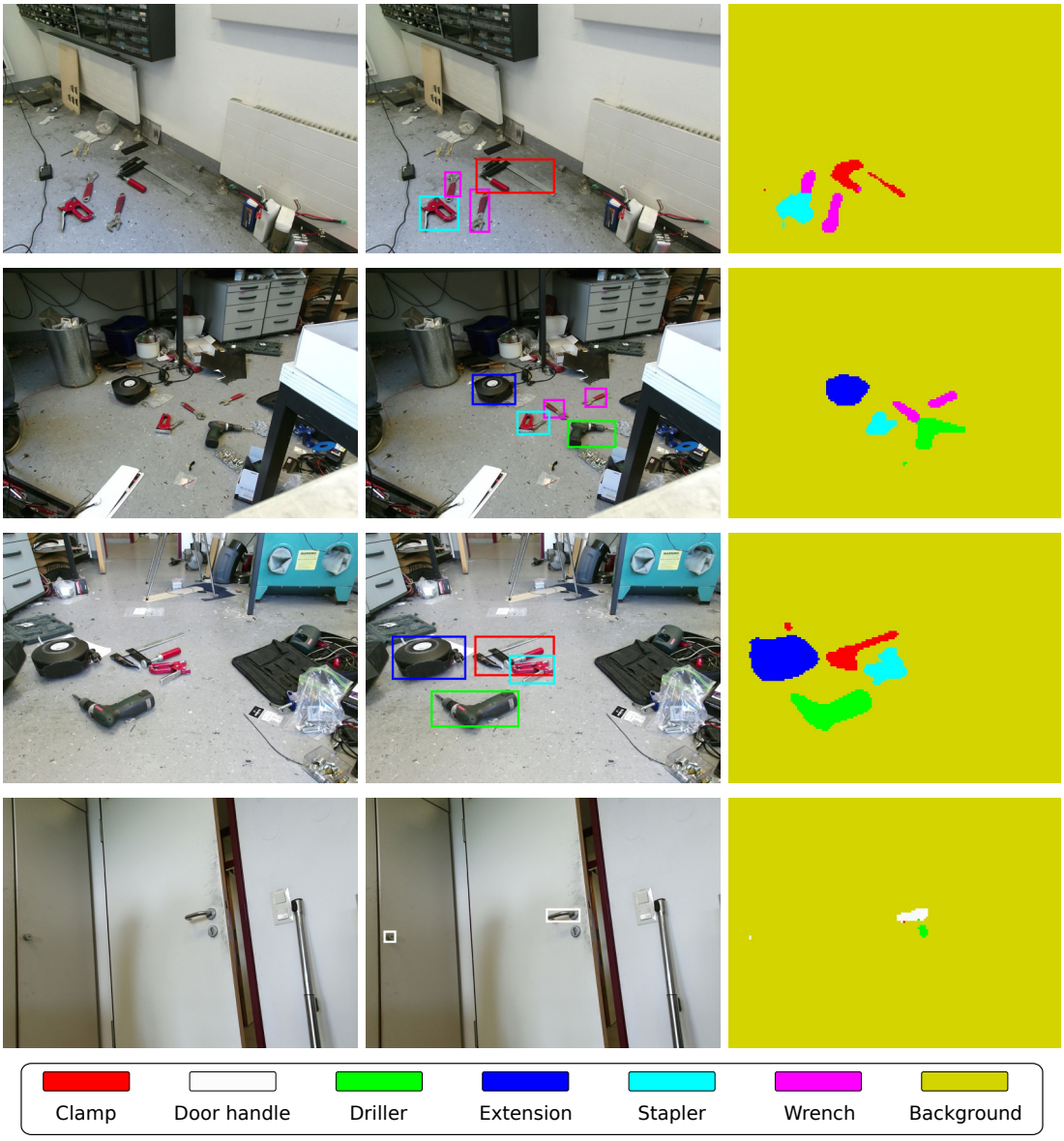


Figure 13. Example scenes (left) and results of object detection (center) and semantic segmentation (right) on the disaster response dataset. For object detection, the highest resolution model is used.

The segmentation results are presented in Table 7. Here the segmentation achieves good results, which is notable in the presence of highly cluttered background with many other objects that are visually rather similar. Finally, Figure 13 shows an example result of our detection and segmentation approaches on the CENTAURO dataset. Surprisingly, the highly cluttered background does not affect the overall performance. Rather, the main difficulty with this dataset is the small object size w.r.t. to the image size, in contrast to the APC data.

6 Lessons Learned

Designing the system, participating in the APC 2016, and finally the experiments on disaster scenes was a valuable learning experience for us. We summarize and discuss some of the points related to the perception system here.

First of all, our transfer learning approach has shown to work effectively in real-life robotic applications, requiring only few annotated images. We expect that further work (which is necessary for the ARC 2017) will reduce the number of required images even further, paving the way for one-shot or few-shot learning.

The main insight here is that these problems can be approached successfully with deep learning techniques, even if the amount of training data is low. Furthermore, our experiments showed that it is beneficial to finetune pretrained architectures rather than relying on classical CNN+SVM combination in the considered scenarios, despite the limited training data.

One valuable lesson from the disaster response scenario is that input resolution is an important parameter for the performance of detection approaches. Sadly, this is currently a highly domain-specific parameter, requiring careful adaptation to the task at hand—and the processing time available. More generally, state-of-the-art deep learning techniques still require significant amounts of manual tuning to adapt the models to the target domain. Despite the replacement of handcrafted features with learned ones, hyperparameters have to be chosen carefully to guarantee success. Finally, tricks such as the HHA encoding may boost the performance in certain settings, but are not generally applicable to all domains.

7 Conclusion

We presented a successful adaptation of two different deep-learning-based image understanding methods to robotic perception. Our experiments showed that by exploiting transfer learning, such approaches can be applied to real-world manipulation tasks without excessive need for annotating training images. We demonstrated their performance in two different settings. One is a bin-picking scenario, carried out in the context of the Amazon Picking Challenge (APC) 2016, where the number of categories, narrow working spaces, as well as shiny and textureless surfaces pose major challenges. The APC 2016 was our very first attempt to apply deep-learning techniques in a live robotic system. Our team's success underlines the effectiveness of such methods in practice. We believe that the reduction of training data is a key factor in such scenarios. This will become even more crucial in future editions of the APC, where the number of categories will increase and some categories may not even be known during the training phase.

The second application scenario is disaster response. Here, the main challenges for perception include severe clutter and unstructured background. Nevertheless, we showed that similar ideas can be transferred to this setting. To validate this claim, we collected and annotated a domain-specific dataset and observed encouraging performance in both detection and segmentation tasks.

In future, we plan to bring both tasks closer together by integrating them into a single network architecture. This would allow for end-to-end training of both components simultaneously. Finally, we make the entire code base used for the APC as well as the collected and annotated data publicly available[††]. We hope this will encourage other researchers to apply deep models in live robotic systems.

References

Badrinarayanan V, Kendall A and Cipolla R (2015) SegNet: A deep convolutional encoder-decoder architecture for image segmentation. *arXiv:1511.00561* .

Behnke S (2003) *Hierarchical Neural Networks for Image Interpretation*, *Lecture Notes in Computer Science*, volume 2766. Springer.

Berner A, Li J, Holz D, Stückler J, Behnke S and Klein R (2013) Combining contour and shape primitives for object detection and pose estimation of prefabricated parts. In: *IEEE International Conference on Image Processing (ICIP)*. pp. 3326–3330.

Buchholz D, Kubus D, Weidauer I, Scholz A and Wahl FM (2014) Combining visual and inertial features for efficient grasping and bin-picking. In: *IEEE International Conference on Robotics and Automation (ICRA)*. pp. 875–882.

Chen LC, Papandreou G, Kokkinos I, Murphy K and Yuille AL (2015) Semantic image segmentation with deep convolutional nets and fully connected CRFs. In: *International Conference on Learning Representations (ICLR)*.

Correll N, Bekris KE, Berenson D, Brock O, Causo A, Hauser K, Okada K, Rodriguez A, Romano JM and Wurman PR (2016) Analysis and observations from the first amazon picking challenge. *IEEE Transactions on Automation Science and Engineering* .

Domae Y, Okuda H, Taguchi Y, Sumi K and Hirai T (2014) Fast graspability evaluation on single depth maps for bin picking with general grippers. In: *IEEE International Conference on Robotics and Automation (ICRA)*. pp. 1997–2004.

Drost B, Ulrich M, Navab N and Ilic S (2010) Model globally, match locally: Efficient and robust 3D object recognition. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. pp. 998–1005.

Eppner C, Höfer S, Jonschkowski R, Martín-Martín R, Sieverling A, Wall V and Brock O (2016) Lessons from the Amazon Picking Challenge: Four aspects of building robotic systems. In: *Robotics: Science and Systems (RSS)*.

[††] `http://ais.uni-bonn.de/apc2016/`

Ferstl D, Reinbacher C, Ranftl R, Rüther M and Bischof H (2013) Image guided depth upsampling using anisotropic total generalized variation. In: *IEEE International Conference on Computer Vision (ICCV)*. pp. 993–1000.

Geiger A, Roser M and Urtasun R (2010) Efficient large-scale stereo matching. In: *Asian Conference on Computer Vision (ACCV)*.

Girshick R (2015) Fast R-CNN. In: *IEEE International Conference on Computer Vision (ICCV)*.

Girshick R, Donahue J, Darrell T and Malik J (2014) Rich feature hierarchies for accurate object detection and semantic segmentation. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. pp. 580–587.

Graves A, Mohamed Ar and Hinton GE (2013) Speech recognition with deep recurrent neural networks. In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. pp. 6645–6649.

Gupta S, Girshick R, Arbelaez P and Malik J (2014) Learning rich features from RGB-D images for object detection and segmentation. In: *European Conference on Computer Vision (ECCV)*.

Gupta S, Hoffman J and Malik J (2016) Cross modal distillation for supervision transfer. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. pp. 2827–2836.

Harada K, Wan W, Tsuji T, Kikuchi K, Nagata K and Onda H (2016) Iterative visual recognition for learning based randomized bin-picking. *arXiv:1608.00334* .

He K, Gkioxari G, Dollár P and Girshick R (2017) Mask R-CNN. *arXiv:1703.06870* .

He K, Zhang X, Ren S and Sun J (2016) Deep residual learning for image recognition. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Hernandez C, Bharatheesha M, Ko W, Gaiser H, Tan J, van Deurzen K, de Vries M, Mil BV, van Egmond J, Burger R, Morariu M, Ju J, Gerrmann X, Ensing R, van Frankenhuyzen J and Wisse M (2016) Team Delft's robot winner of the Amazon Picking Challenge 2016. *arXiv:1610.05514* .

Holz D, Topalidou-Kyniazopoulou A, Stückler J and Behnke S (2015) Real-time object detection, localization and verification for fast robotic depalletizing. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. pp. 1459–1466.

Husain F, Schulz H, Dellen B, Torras C and Behnke S (2016) Combining semantic and geometric features for object class segmentation of indoor scenes. *IEEE Robotics and Automation Letters* 2(1): 49–55.

Ivakhnenko AG and Lapa G Valentin (1966) *Cybernetic Predicting Devices*. CCM Information Corp.

Johnson J, Karpathy A and Fei-Fei L (2016) DenseCap: Fully convolutional localization networks for dense captioning. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Jonschkowski R, Eppner C, Höfer S, Martín-Martín R and Brock O (2016) Probabilistic multi-class segmentation for the amazon picking challenge. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Kaipa KN, Kankanhalli-Nagendra AS, Kumbla NB, Shriyam S, Thevendria-Karthic SS, Marvel JA and Gupta SK (2016) Addressing perception uncertainty induced failure modes in robotic bin-picking. *Robotics and Computer-Integrated Manufacturing* 42: 17–38.

Karpathy A and Li FF (2015) Deep visual-semantic alignments for generating image descriptions. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Kingma D and Ba J (2015) Adam: A method for stochastic optimization. In: *International Conference on Learning Representations (ICLR)*.

Krishna R, Zhu Y, Groth O, Johnson J, Hata K, Kravitz J, Chen S, Kalantidis Y, Li L, Shamma DA, Bernstein MS and Li F (2017) Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International Journal of Computer Vision (IJCV)* 123(1): 32–73.

Krizhevsky A, Sutskever I and Hinton GE (2012) ImageNet classification with deep convolutional neural networks. In: Bartlett P, Pereira F, Burges C, Bottou L and Weinberger K (eds.) *Advances in Neural Information Processing Systems (NIPS)*. pp. 1097–1105.

LeCun Y, Jackel L, Boser B, Denker J, Graf H, Guyon I, Henderson D, Howard R and Hubbard W (1989) Handwritten digit recognition: applications of neural network chips and automatic learning. *IEEE Communications Magazine* 27(11): 41–46.

Leitner J, Tow AW, Dean JE, Sünderhauf N, Durham JW, Cooper M, Eich M, Lehnert CF, Mangels R, McCool C, Kujala P, Nicholson L, Pham T, Sergeant J, Zhang F, Upcroft B and Corke PI (2017) The ACRV picking benchmark (APB): A robotic shelf picking benchmark to foster reproducible research. In: *IEEE International Conference on Robotics and Automation (ICRA)*.

Lin G, Milan A, Shen C and Reid I (2017) RefineNet: Multi-path refinement networks for high-resolution semantic segmentation. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Liu W, Anguelov D, Erhan D, Szegedy C, Reed S, Fu CY and Berg AC (2016) SSD: Single shot multibox detector. In: *European Conference on Computer Vision (ECCV)*. pp. 21–37.

Long J, Shelhamer E and Darrell T (2015) Fully convolutional networks for semantic segmentation. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Martinez C, Boca R, Zhang B, Chen H and Nidamarthi S (2015) Automated bin picking system for randomly located industrial parts. In: *IEEE International Conference on Technologies for Practical Robot Applications (TePRA)*.

Nieuwenhuisen M, Droeschel D, Holz D, Stückler J, Berner A, Li J, Klein R and Behnke S (2013) Mobile bin picking with an anthropomorphic service robot. In: *IEEE International Conference on Robotics and Automation (ICRA)*. pp. 2327–2334.

Pinheiro PO and Collobert R (2015) From image-level to pixel-level labeling with convolutional networks. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Pretto A, Tonello S and Menegatti E (2013) Flexible 3D localization of planar objects for industrial bin-picking with monocamera vision system. In: *IEEE International Conference on Automation Science and Engineering (CASE)*. pp. 168–175.

Quigley M, Conley K, Gerkey B, Faust J, Foote T, Leibs J, Wheeler R and Ng AY (2009) ROS: an open-source robot operating system. In: *ICRA workshop on open source software*, volume 3. p. 5.

Ren S, He K, Girshick R and Sun J (2015) Faster R-CNN: Towards real-time object detection with region proposal networks. In: *Advances in Neural Information Processing Systems (NIPS)*. pp. 91–99.

Russakovsky O, Deng J, Su H, Krause J, Satheesh S, Ma S, Huang Z, Karpathy A, Khosla A, Bernstein M, Berg AC and Fei-Fei L (2015) ImageNet large scale visual recognition challenge. *International Journal of Computer Vision* 115(3): 211–252.

Scherer D, Müller A and Behnke S (2010) Evaluation of pooling operations in convolutional architectures for object recognition. In: *International Conference on Artificial Neural Networks (ICANN)*. pp. 92–101.

Schwarz M, Milan A, Lenz C, Munoz A, Periyasamy AS, Schreiber M, Schüller S and Behnke S (2017a) NimbRo Picking: Versatile part handling for warehouse automation. In: *IEEE International Conference on Robotics and Automation (ICRA)*.

Schwarz M, Rodehutskors T, Droeschel D, Beul M, Schreiber M, Araslanov N, Ivanov I, Lenz C, Razlaw J, Schüller S, Schwarz D, Topalidou-Kyniazopoulou A and Behnke S (2017b) NimbRo Rescue: Solving disaster-response tasks through mobile manipulation robot Momaro. *Journal of Field Robotics* 34(2): 400–425.

Schwarz M, Schulz H and Behnke S (2015) RGB-D object recognition and pose estimation based on pre-trained convolutional neural network features. In: *IEEE International Conference on Robotics and Automation (ICRA)*. pp. 1329–1335.

Sechidis K, Tsoumakas G and Vlahavas I (2011) On the stratification of multi-label data. In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, pp. 145–158.

Sermanet P, Eigen D, Zhang X, Mathieu M, Fergus R and LeCun Y (2013) OverFeat: Integrated recognition, localization and detection using convolutional networks. *CoRR* abs/1312.6229.

Simonyan K and Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. *CoRR* abs/1409.1556.

Song S, Lichtenberg SP and Xiao J (2015) SUN RGB-D: A RGB-D scene understanding benchmark suite. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. pp. 567–576.

Sutskever I, Vinyals O and Le QV (2014) Sequence to sequence learning with neural networks. In: *Advances in Neural Information Processing Systems (NIPS)*. pp. 3104–3112.

Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V and Rabinovich A (2015) Going deeper with convolutions. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pp. 1–9.

Yu KT, Fazeli N, Chavan-Dafle N, Taylor O, Donlon E, Lankenau GD and Rodriguez A (2016) A summary of team MIT's approach to the Amazon Picking Challenge 2015. *arXiv:1604.03639* .

Zeng A, Yu KT, Song S, Suo D, Walker Jr E, Rodriguez A and Xiao J (2017) Multi-view self-supervised deep learning for 6D pose estimation in the Amazon Picking Challenge. In: *IEEE International Conference on Robotics and Automation (ICRA)*.