Fast Object Learning and Dual-arm Coordination for Cluttered Stowing, Picking, and Packing

Max Schwarz*, Christian Lenz, Germán Martín García, Seongyong Koo, Arul Selvam Periyasamy, Michael Schreiber, and Sven Behnke

***Abstract*— Robotic bin picking from cluttered bins is a challenging task. We describe our team's entry to the Amazon Robotics Challenge 2017, which required stowing items into a storage system, picking specific items, and packing cardboard boxes. Our object perception pipeline can be quickly and efficiently adapted to new items using a custom turntable capture system and the transfer learning paradigm. It produces high-quality item contours, on which grasp poses are found heuristically. A planning component coordinates manipulation actions between the two robot arms, minimizing execution time. The system has been demonstrated successfully at the Amazon Robotics Challenge 2017, where our team NimbRo Picking reached second places in both the picking task and the final stow-and-pick task. Additionally, we evaluate individual components in separate experiments.**

I. Introduction

In order to successfully approach robotic bin picking, multiple research fields ranging from computer vision, grasp planning, motion planning, and control need to be tightly coupled. Especially the case of cluttered bin picking, i.e. items of different types randomly arranged, is the focus of active research. In order to advance the state of the art, Amazon holds yearly competitions: The Amazon Picking Challenges (APC) 2015 and 2016, and the Amazon Robotics Challenge (ARC) 2017[1].

On a high level, the ARC required contestants to solve two common warehouse tasks: The stowing of newly arrived items into a storage system ("stow task"), and the retrieval and packing of specific items from storage into boxes ("pick task"). In contrast to the APC 2016, this year's competition allowed participants much more leeway with regards to the system design. In particular, the storage system itself could be built by the teams. On the other hand, the task was made more challenging by not providing all items to the teams before the competition, instead requiring participants to learn new items in short time (45 min). This forced the development of novel object perception approaches.

Our team NimbRo Picking developed a robotic system for the ARC 2017 (see Fig. 1). Contributions include:

- A method for quickly and efficiently capturing novel items with minimal human involvement,
- a highly precise semantic segmentation pipeline which can be adapted to new items on-the-fly, and
- a method for dual-arm coordination for complex picking or stowing tasks.

*All authors at AIS, University of Bonn, max.schwarz@ais.uni-bonn.de

[1]https://www.amazonrobotics.com/#/roboticschallenge/results

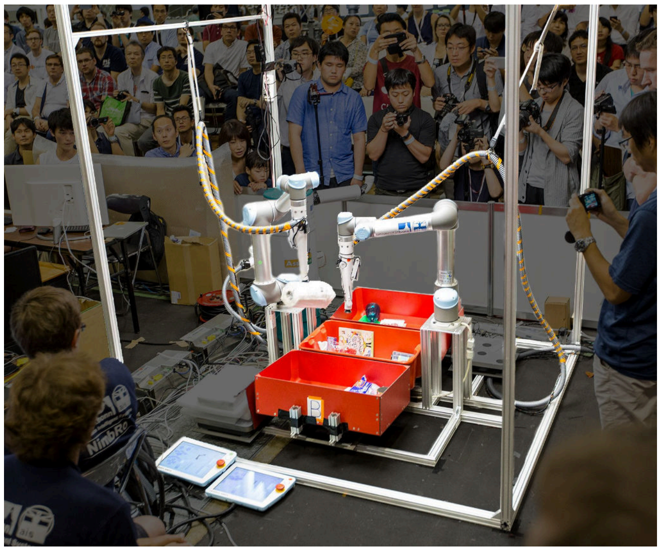
Fig. 1. Our system at the Amazon Robotics Challenge 2017 in Nagoya, performing the stow phase of the final task. Image by Amazon Robotics.

II. Related Work

The Amazon Picking Challenge 2016 resulted in the development of some very interesting systems for bin picking, serving as inspiration for our system.

Hernandez *et al.* [1] won the picking and stowing challenges. Their system consisted of an industrial arm equipped with a hybrid suction and pinch gripper. Similarly, our gripper can also apply suction and pinch grasps. However, our gripper design allows suction and pinching simultaneously, which was (to our knowledge) not possible with Team Delft's gripper. The team also used, like a number of other teams, a fixed camera setup for perception of items in the tote—allowing the perception pipeline to run while the robot is putting an item away. This convinced us to build a fixed sensor gantry for our system this year.

Matsumoto *et al.* [2] placed second in the pick task and fourth in the stow task. Their system directly trains a neural network to predict item grasp poses. We initially decided against such an approach because item grasp annotations would be expensive to obtain for new items and we were not sure whether grasp affordances could be effectively transferred from the known items.

Our own entry for the Amazon Picking Challenge 2016 [3], [4] placed second in the stow competition and third in the pick competition. In contrast to this year, our 2016

Preprint submitted to 2018 IEEE International Conference on Robotics and Automation. Received September 22, 2017.

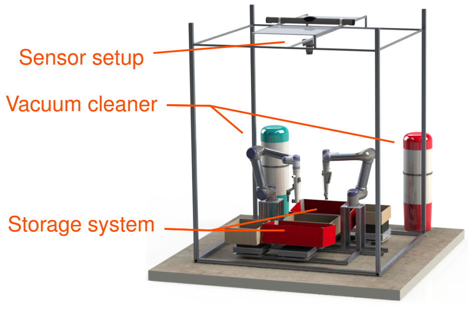

Fig. 2. CAD model of the entire system.

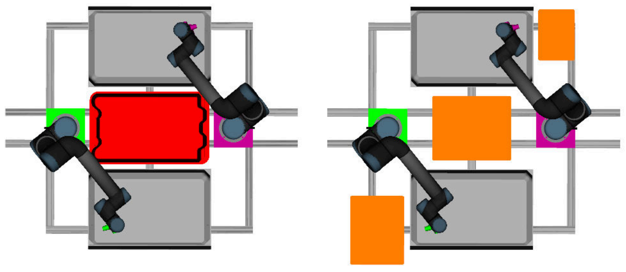

Fig. 3. System setup for both tasks. Storage system bins are depicted in gray. Left: Configuration with tote (red) for the stow task. Right: Cardboard boxes (orange) for the pick task.

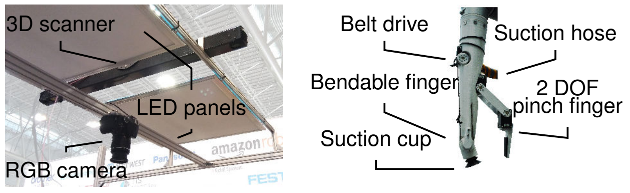

Fig. 4. Left: Gantry setup. Right: 3 DOF suction gripper.

system used a single UR10 arm, could only use suction for manipulation, and required manual annotation of entire tote or shelf scenes for training the object perception pipeline.

Autonomous driving has given a large impulse into the field of semantic segmentation. Large datasets allow the training of increasingly complex models (e.g. [5], [6]), but few works focus on fast training from few examples, as required in this application.

Dual-arm manipulation has been investigated for a long time, mostly inspired by the human physiology. Smith *et al.* [7] survey different approaches and introduce the useful distinction of goal-coordinated manipulation (two arms working towards a shared goal without direct physical interaction) and bimanual manipulation (two arms manipulating the same item). In this scheme, our system falls into the former category. Since most works focus on the bimanual manipulation case [8], [9] or consider sequential manipulation of one item with two arms [10], our case of independent manipulation in a shared workspace is quite interesting. Other works focus on collision free multi robot manipulation planned offline [11]. This is not sufficient in our case since the arm trajectories and timings are not fully known in advance.

III. Mechatronic Design

Our system design was driven by three design goals: Task completion, speed, and simplicity (in this order). It was important to focus on task completion first, since any time bonus would only be awarded if the task was complete. We figured that it would be likely that only few, if any, teams would complete the entire task—indeed, at the competition no team was able to fully complete the final task.

A. Arms and Grippers

Our experience from last year told us that suction is a very powerful tool for bin picking—we could manipulate all items using suction at APC 2016. Nevertheless, it was clear that this time mechanical grasping would be required as well.

To address our second design goal, speed, we decided to go for a dual-arm system. In particular the pick task lends itself to parallelization—three cardboard boxes have to be filled with specific items, which can be done mostly independently as long as multiple target items are visible, i.e. not occluded by other items.

Consequently, our robot system consists of two Universal Robotics UR5 arms with 6-DOF. Each arm is equipped with an endeffector having a bendable suction finger and a 2-DOF second finger (see Fig. 4). The suction is generated by two powerful vacuum cleaners, one for each arm. The suction power can be controlled with an actuated bleed valve.

B. Storage System

Our storage system meets the maximum allowed volume and area constraints. The rules specify a maximum of ten bins. We chose two bins, since then the bin size is similar to the tote size, equalizing the perception situations in both containers. For the same reason, our storage system is painted red, in the same color as the tote.

During the design of our storage system, we tried to make it easily accessible for manipulation and perception. Both parts of the storage system are reachable by both arms (see Fig. 3) and are tilted by approx. 5° towards the center of the robot system to increase the visibility of items located close to the inner walls of the bins. The tote (stow task) or one of the cardboard boxes (pick task) is located between the two storage system bins. One remaining cardboard box for the pick task is placed next to each arm and is only accessible by this arm. We placed an industrial scale under each of the five possible pick and place locations for measuring the weight of the picked item and detecting contact between the arms and the bins, cardboard boxes, and the tote.

C. Gantry Sensors

Our robot system is equipped with a 24 MPixel photo camera (Nikon D3400) and a 3.2 MPixel Photoneo PhoXi$^{®}$ 3D-Scanner XL (see Fig. 4). The 3D scanner offers sub-millimeter absolute accuracy on well-measurable surfaces. Both sensors are mounted on a gantry approx. 2 m above the storage system and tote. This configuration allows us to

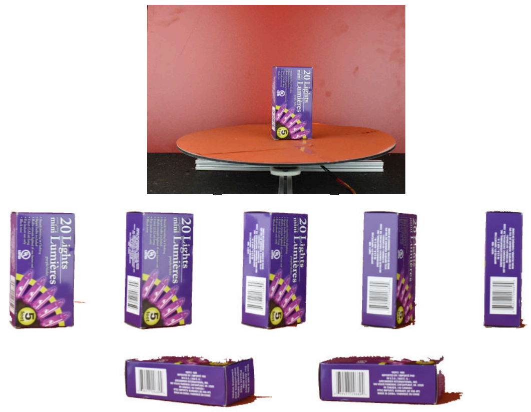
Fig. 5. Turntable capture and automatic segmentation. Top: Input image. Middle/Bottom: Extracted segments in standing and lying configuration.

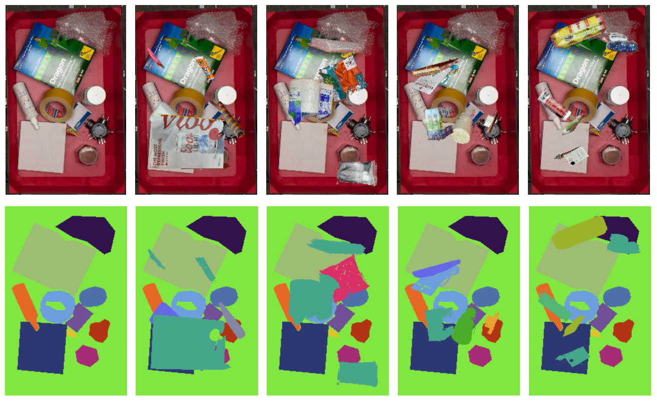
Fig. 6. Generated synthetic scenes. All scenes were generated with the same annotated background frame (left column) for easier comparison. Top row: RGB. Bottom row: Color-coded generated segmentation ground truth.

observe all important parts of the system without moving the sensors. Two LED panels provide active lighting, reducing the influence of outside lights.

D. GPU Server

For fast training of deep neural networks, our system includes a GPU server with four Nvidia Titan X (Pascal) cards. The server exports a network share for transferring new training data captured on the turntable.

IV. Object Perception

A. Data Capture & Modeling

During a competition run, our system has to quickly adapt to the provided new items. We experimented with using only the few images provided by Amazon, but obtained significantly better results using more images. The key issue is that capturing tote scenes and annotating them manually as in our 2016 system [3] would be much too time consuming.

Instead, we capture turntable frames using an automated turntable setup (see Fig. 5). The turntable is equipped with a Nikon D3400 camera (identical to the one in the gantry) and an Intel RealSense SR300 sensor. The turntable captures twenty frames per revolution, with one revolution lasting 10 s. This means a typical item can be scanned in three different resting poses on the turntable in about a minute, including manual repositioning.

Before starting the item capture, we also record a frame without the item. We then use a background subtraction scheme to automatically obtain a binary item mask. The masks are visualized and the mask generation parameters can be quickly modified to fit the particular item using a graphical user interface. While the generated masks are not perfect, they suffice for training.

B. Semantic Segmentation Architecture

In contrast to our previous APC 2016 entry [3], which combined state-of-the-art object detection and semantic segmentation approaches, we decided to go with a pure semantic segmentation pipeline for the ARC 2017. This decision was motivated by a) the small gain obtained by the hybrid pipeline and b) the fact that Amazon removed the possibility of multiple items of the same class being in the same container, making true instance segmentation unnecessary. In our experience, having pixel-precision segmentation instead of just rectangle-based object detection is a large advantage for scene analysis and grasp planning.

As a basis, we reimplemented the RefineNet architecture proposed by Lin *et al.* [5], which gave state-of-the-art results on the Cityscapes dataset. It uses intermediate features from a pretrained ResNet-101 network [12], extracted after each of the four ResNet blocks. Since the features get more abstract, but also reduce in resolution after each block, the feature maps are sequentially upsampled and merged with the next-larger map, until the end result is a both high-resolution and highly semantic feature map. The classification output is computed using a linear layer and pixel-wise SoftMax. For our purposes, we replaced the backbone network with the similar but newer ResNeXt-101 network [13].

C. Scene Synthesis & Fast Training

As mentioned above, a key requirement is the fast adaption to new items. Since the amount of training images we can capture is very limited and the item images are recorded on the turntable without occlusions, we generate new synthetic scenes for training (see Fig. 6).

This scene generation happens on-the-fly during training, so that we can immediately start training and add new turntable captures as they become available. Manually annotated dataset frames are used as background, with five new items placed randomly on top. This generates enough occlusion on the new items, while not overloading the scene. The scene generation part runs purely on CPU and is multithreaded to achieve maximum performance.

The network training itself is distributed over N GPUs. We train on N images (one image per card) and then average and synchronize the weight gradients using the NCCL library[2]. Using one scene generation pipeline per GPU card, we can obtain 100% GPU utilization.

[2] `https://github.com/NVIDIA/nccl`

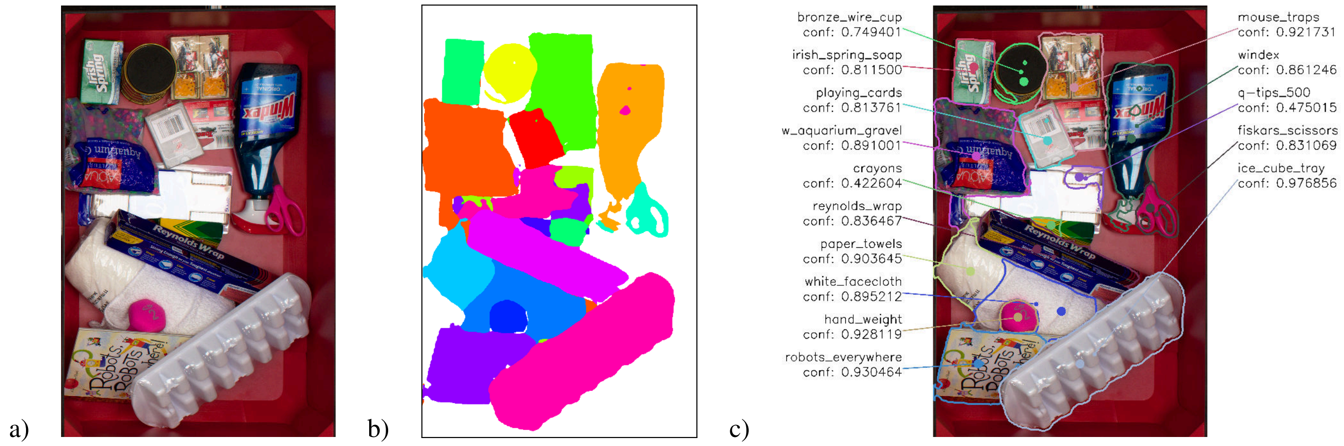
a)
b)
c)
bronze_wire_cup conf: 0.749401
irish_spring_soap conf: 0.811500
playing_cards conf: 0.813761
w_aquarium_gravel conf: 0.891001
crayons conf: 0.422604
reynolds_wrap conf: 0.836467
paper_towels conf: 0.903645
white_facecloth conf: 0.895212
hand_weight conf: 0.928119
robots_everywhere conf: 0.930464
mouse_traps conf: 0.921731
windex conf: 0.861246
q–tips_500 conf: 0.475015
fiskars_scissors conf: 0.831069
ice_cube_tray conf: 0.976856

Fig. 7. Object perception example from the picking phase of our finals run at ARC 2017. The original model trained during the run was used. a) RGB image captured by the Nikon camera. b) Segmentation output. c) Processed item contours with average confidences, polygon center of mass (small points), and suction spots (large points). Best viewed in color.

While the ResNeXt backbone network is kept fixed during training, all other RefineNet layers and the final classification layer are trained with a constant learning rate. Weight updates are computed using the Adam optimizer [14]. We pretrain the network on the set of known objects, and then finetune during the competition for the new objects. After every epoch, the filesystem is scanned for new turntable captures, the classification layers are potentially adapted to a new number of classes, and training is resumed.

D. Heuristic Grasp Selection

Since it is infeasible to manually specify grasp positions for the large number of items, especially for the new items in each competition run, we built a robust grasp pose heuristic for 2D grasp proposal. The heuristic is tuned towards suction grasps. To avoid the dangerous item boundaries, the heuristic starts with the contour predicted by the segmentation pipeline. As a first guess, it computes the point with maximum distance d_p to the item contour, the so-called pole of inaccessibility [15]. For fast computation, we use an approximation algorithm[3].

For most lightweight items, the pole of inaccessibility suffices. For heavy items, we also check the 2D polygon center of mass and compute its distance d_m to the contour. If $\frac{d_m}{d_p} > 0.5$, we prefer to grasp at the center of mass. See Fig. 7 for examples.

In order to generate a 5D suction pose (rotations around the suction axis are not considered), depth information is needed. We upsample and filter the depth map generated by the PhoXi 3D scanner by projecting it into the camera frame and running a guided upsampling filter [4], [16]. The resulting high-resolution depth map is used to estimate local surface normals. Finally, the 5D suction pose consists of the 3D grasp contact point and the local surface normal.

For pinch grasps, the rotation around the suction axis has to be determined. Here we try to point the second finger towards the bin center, to avoid collisions. We add Gaussian noise on both translation ($\sigma = 1.5\,\text{cm}$) and the rotation ($\sigma = 60°$), in order to obtain slightly different grasp poses on each manipulation attempt.

[3]https://github.com/mapbox/polylabel

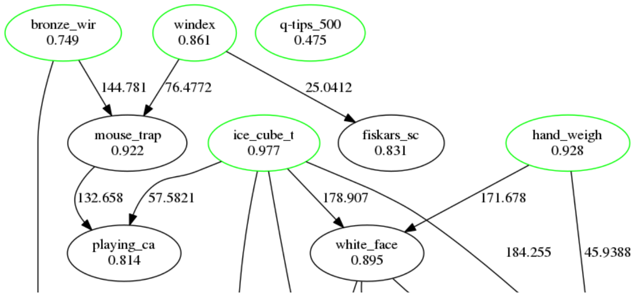

Fig. 8. Clutter graph for the scene in Fig. 7. The bottom half is cut off, leaving only the items on top of the pile. Vertices contain the class name and detection confidence. Green vertices have no predecessor. Edges are labelled with the point count (predecessor higher than successor).

E. The Clutter Graph

For high-level planning, it is quite important to estimate which items are currently graspable and which are occluded by other items, which would need to be moved first. For this reason, we generate a directed graph, which we call the clutter graph. All perceived items are vertices in this graph, with an edge from A to B indicating that A is occluding B. See Fig. 8 for an example.

The graph is initially generated by examining the item contours. Along the contour, we check the upsampled depth map for points on the outer side which are higher than the corresponding points on the inner side. These points are counted and an edge is inserted into the graph, directed from the occluding item to the item under consideration. The point count is attached to the edge.

After simplifying cycles of length two (edges and back edges) by reducing them to one edge with the difference in point counts, we further remove cycles by finding a set of edges of minimum point count, whose removal makes the graph acyclic. This is called the minimum feedback arc

Preprint submitted to 2018 IEEE International Conference on Robotics and Automation. Received September 22, 2017.

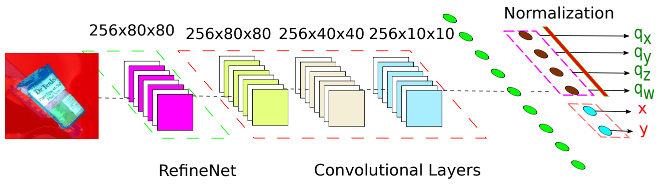

Fig. 9. Pose Estimation network architecture

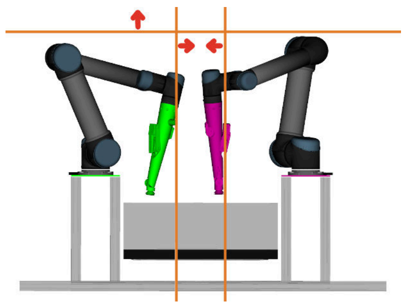

Fig. 10. Cost function planes for the IK solver. The planes affect the wrist of the robot. The vertical plane keeps the endeffector vertical, as long as the horizontal planes are not active (purple robot). The horizontal planes keep the wrist away from the robot base to prevent collisions.

set and is NP-hard, so we find a brute-force solution. The result is a directed acyclic graph containing the occlusion information, which can be easily used to read off which items are graspable.

F. Object Pose Estimation

During preparation for the ARC 2017, we anticipated more difficult items which would be graspable only at very specific grasp poses. In order to retain the needed flexibility, we decided to prepare a 6D pose estimation module, which would allow us to specify allowable grasps relative to an item frame.

This module includes the tools for arranging the turntable captures (consisting of one to four sequences of turntable revolutions with different item orientations), a 5D pose prediction network, and a post-processing part.

The architecture of the pose estimation network is shown in Fig. 9. It predicts the 3D orientation of the item relative to the camera frame in the form of an unit quaternion. A second branch predicts the 2D pixel location of the item coordinate frame. The network consists of the RefineNet backbone as in the semantic segmentation network, followed by three convolution layers, and two fully connected layers. For N item classes, the network predicts $6N$ values—one quaternion and translation offset per item class. This explicitly allows the predictor to adapt to the item class. Note that the classification decision is already made by the semantic segmentation network. During training, only the predictor corresponding to the correct item is subject to the error metric and is trained.

Using the object contour obtained from segmentation, we crop the RGB image. The size of the crop is computed based on the expected maximum size of the item in the image, which results in constant scale even if part of the object is occluded. During training, a segment captured on the turntable is placed on top of a randomly cropped storage system scene. Furthermore, the background is shifted towards red to emphasize the item currently under consideration (see Fig. 9). The output of the pose estimation network is projected to a full 6D pose using the depth map and used as an initialization for a traditional ICP-based registration method to compute the exact 6D item pose.

V. Dual-Arm Motion Generation

A. Parametrized Motion Primitives

The UR5 arms and the endeffectors are controlled with parametrized motion primitives. A motion is defined by a set of keyframes which specify the kinematic group(s) manipulated by this motion. Each keyframe either defines an endeffector pose in Cartesian space or the joint state of the kinematic group(s). The keyframes are either manually designed beforehand or generated and adapted to perception results at runtime. This motion generation has been used on other robot systems in our group before (see [3] and [17]).

B. Inverse Kinematics

For keyframes defined in Cartesian space we use a selectively damped least square (SDLS) solver is used, as in [3]. Since the arm including the suction finger has seven DOF, we can optimize secondary objectives in the null space of the Jacobian matrix. In our case, we want to keep the wrist as high as possible and thus keep the endeffector vertical in order to reduce the horizontal space needed while manipulating.

Hence, we define a horizontal plane above the robot and use the squared distance from the wrist to the plane as cost function. In the stow task, two additional vertical planes are added (see Fig. 10) to prevent the wrist getting too close to the manipulator base. For further details, we refer to [3].

C. High-Level Planning for Picking

The high-level planner for the pick task triggers the perception pipeline, processes the segmentation results and assigns manipulation tasks to the arms. The perception pipeline is started for a particular bin whenever no possible tasks are left and the bin is not occluded by an arm. Item detections are sorted regarding a per-item fail counter, the number of items occluding the target item, and the perception confidence. The two best ranked target items are marked as possible tasks for this bin. If no target items are detected or the fail counter for the best items is too large, new tasks moving non-target item out of the way are generated.

D. Placement Planning

Since the space inside the cardboard boxes is limited, our system finds optimal placement poses inside the boxes. The placement planner uses bounding box dimensions provided by Amazon for each item. It considers three disjoint sets of items per box: Already placed items (A), currently possible

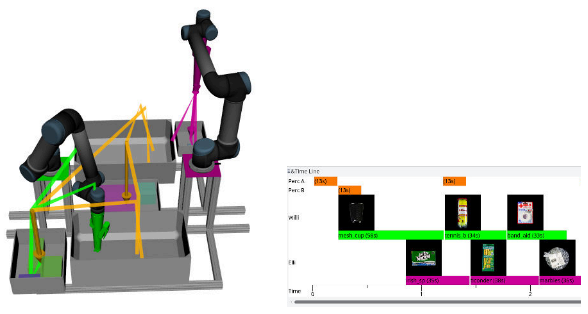

Fig. 11. Planning for the pick task. Left: Visualization of manipulation tasks. Chosen tasks are marked in green and purple. Right: Timeline of actions including perception time and arm motions.

TABLE I

TIMINGS AND SUCCESS RATES FROM ARC 2017

		Individual Challenges			Final Challenge		
		#	Time [s]	Stddev	#	Time [s]	Stddev
Stow	Vision	not comparable			19	11.1	0.0
	Stows				14	29.8	5.4
	Fails				12	14.0	6.9
	Sum				45	13:17 min	
	Runtime				45	10:33 min	
Pick	Vision	13	12.0	0.9	32	13.1	1.3
	Picks	10	38.3	7.6	8	39.1	10.0
	Moves	4	34.5	9.0	10	30.3	3.0
	Fails	5	20.4	3.2	22	28.9	10.9
	Sum	32	12:59 min		72	27:52 min	
	Runtime	32	8:56 min		72	19:22 min	

task items (B), and items which will be picked later (C). The planner finds a brute-force solution in the form of a 3D stacking of the item bounding boxes, under the constraint that items from A have a fixed position and items from B have to be placed before items from C. The bounding boxes may be rotated by $90°$ around the vertical axis. The solution with minimum total stacking height is then used to determine the target poses for each item from B.

Objects of oblong shape are always placed such that the height dimension is the smallest dimension. This may necessitate a rotating motion during placement, since the items are always grasped roughly from above. If required, we place an additional constraint on the grasp pose which ensures that the items are grasped in a way that allows the later rotation using our single DOF on the suction finger.

E. Task Allocation

Whenever an arm is free, we assign the best marked task considering collision avoidance with the other arm. A task consists of a set of waypoints of endeffector poses starting with the current arm pose, grasp pose, place pose, home pose of the arm and some intermediate waypoints (see Fig. 11).

Since we assume the last link of the arm to be always vertical, we only consider the 2D endeffector pose for collision checking. Hence, all waypoints are projected into 2D. Next we compute the shortest Euclidean distance for each line segment defined by two consecutive waypoints of one task to all line segments of the other task. If the minimum of all these distances is larger than a threshold, the tasks can be executed in parallel. Since the number of possible tasks is limited, we can test all possible task combinations as long as an arm is free. If multiple collision-free tasks exist, we prefer tasks which can only executed by the free arm (i.e. the place location is in one of the corner boxes). We delete reached waypoints from current tasks to allow the second arm to start on new tasks as soon as possible.

Following each perception run, a predefined per-object probability decides which grasp type (suction or pinch) should be used. After grasping and lifting an item, the item weight is measured with the scale mounted below the storage system and compared with the expected item weight. If the weight difference is under 5 g or 10% of the item weight, we accept the item and place it. Otherwise, we drop it and increment the fail counter otherwise.

F. High-Level Planning for Stowing

In the stow task, the single pick location (the tote) limits the possibility to parallelize the manipulation work, since precise weight change measurements require sequential picking actions in the tote. Therefore, we assign each bin of the storage system to one arm as its associated stow location. This at least allows us to place an item with the first arm while grasping the next item with the second.

Since we have to stow all of the given items, we start with objects where we are confident that they are lying on top. Thus, the item detections are first sorted by the confidence reported by the perception pipeline. Next the best $\frac{3}{4}$ detections are sorted by the total number of items lying on top and finally the best half of these are considered as possible tasks.

Since manipulation is performed open-loop after perception, we allow a maximum of two manipulation attempts before the scene is measured again. We try to find a pair of items containing one of the two best detection results, respecting a minimum distance between the two items. This prevent the first manipulation attempt affecting the second item. If such a pair exists, we assign the items randomly to the arms, otherwise we stow only the item with the best perception confidence.

After grasping an item, the weight check is performed. In contrast to the pick task, no collision avoidance at the high-level planning is needed since each arm has its dedicated workspace and access to the tote is granted sequentially.

VI. Evaluation

We evaluated our work on a system level at the Amazon Robotics Challenge 2017. We augment this evaluation with separate experiments for the object perception pipeline and the dual-arm planner.

A. Amazon Robotics Challenge 2017

At the ARC 2017, our team had four chances to demonstrate the system's abilities. In our practice slot, we success-

Preprint submitted to 2018 IEEE International Conference on Robotics and Automation. Received September 22, 2017.

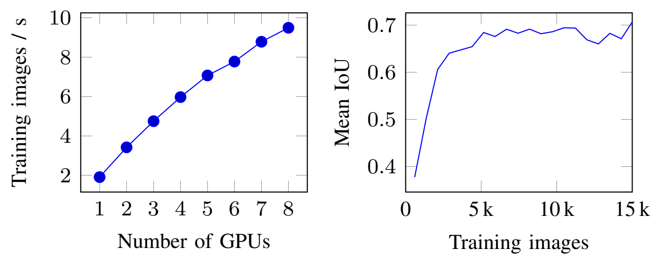

Fig. 12. Semantic Segmentation experiments. Left: Training image throughput depending on the number of GPUs. Right: Test set IoU during training.

fully attempted the pick task and obtained a score of 150 points, the maximum of all practice scores.

Unfortunately, the evening before our official stow run we experienced a short in the power supply wiring, damaging our control computer and a few microcontrollers beyond repair. The repair did not leave us any time for full system tests before our stow run. Consequently, due to a series of operator mistakes caused by the new configuration, our system operated with wrong item weights during the stow task and discarded nearly all grasped items.

We were able to fix these remaining issues for the pick task, where our team scored 245 points, which led to a second place in the pick competition, behind Team Nanyang with 257 points. The third placed team achieved 160 points.

Our system also performed very well in the final task, which combined the stow and pick tasks sequentially. In the stow phase, we were able to stow fourteen out of sixteen items. The remaining two items could not be picked because a bug resulted in an unfortunate gripper orientation, was prevented from execution by collision checks. In the picking phase, we picked eight out of ten target items. One of the target items had not been stowed in the stow phase, so it was impossible to pick. The other one was a cup made out of thin and sparse wires, making it both very difficult to perceive and to grasp. The system succeeded once in grasping it, but discarded it due to an imprecise weight measurement. We scored 235 points in the final task, which placed us second behind the winning Team ACRV with 272 points and in front of Team Nanyang (225 points).

Table I shows a summary of the successes and failures per task and recorded times for perception and manipulation actions. Generally, having two arms for manipulation lowers the overall runtime and allows more manipulation attempts in a given time. Overall, our system performed very well and showcased all of its abilities at the ARC 2017.

B. Semantic Segmentation

After the ARC, we annotated the collected images during our final run (pick phase) with ground truth segments to be able to quantitatively judge segmentation performance. We then recreated the segmentation training run from our final.

To investigate the scalability of our training pipeline, we ran 10 training epochs (with one epoch defined by 140 background images) on a varying number of Nvidia Titan Xp cards. Figure 12 shows that our pipeline scales nicely to up to eight GPUs (and possibly more).

Figure 12 also shows a typical test result curve recorded during training. We measure the intersection over union (IoU) separately for each class and then average over the classes present in the test set. One can see that after 5000 to 10000 images the curve saturates. Using four GPUs, as during the ARC, this occurs after approx. 15 to 30 min. Note that during a real training run, the system starts training with the images provided by Amazon and turntable captures are added over a period of 20 min, extending the needed training time.

TABLE II
POSE ESTIMATION ERRORS

	Translation [pixel]		Quaternion [norm, $\cdot 10^{-2}$]		Angular [degrees]	
	train	val	train	val	train	val
Epsom salts	2.28	3.32	1.63	3.83	1.80	3.19
Toilet paper	2.41	3.79	1.94	4.75	1.68	3.09
Yellow windex	2.25	3.41	2.04	3.23	1.86	2.78
Average	2.31	3.51	1.87	3.93	1.78	3.02

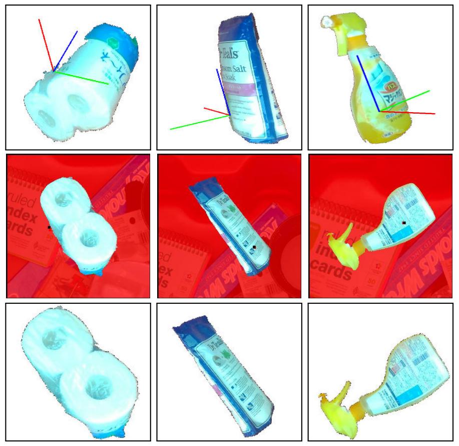

Fig. 13. Pose Estimation experiments. Top row: Item coordinate systems. Middle row: RGB input image. Black and red squares depict ground truth and predicted origin of the item, respectively. Bottom row: Dataset image (train and eval) closest to the predicted orientation. Ideally, this should be the image from the evaluation set matching the predicted orientation.

C. Pose Estimation

During the ARC 2017, pose estimation was not necessary. Our grasp heuristic was able to find good suction or pinch grasps on all of the encountered items. Nevertheless, we evaluated the pose estimation network by training it on three different items. Table II shows quantitative results of these experiments. Our pose estimator is able to predict the translation of the item origin within a few pixels and the orientation within a few degrees. Figure 13 shows example frames from this experiment.

D. Dual-Arm Experiments

We also investigated the speedup of our system achieved by using a second arm. For both tasks (pick and stow) several

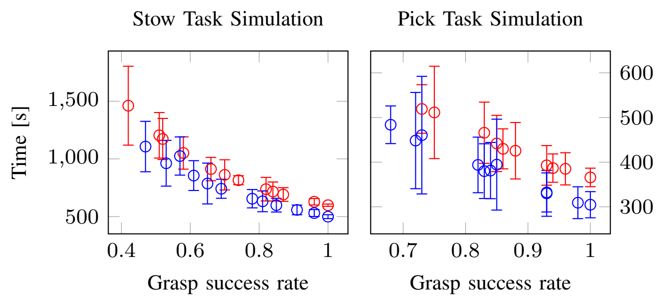

Fig. 14. Averaged run time with standard deviation (10 runs each) in simulation for stow and pick task with one (red) and two arms (blue) used.

full runs were performed in simulation.

The perception pipeline was not simulated, instead the planner was supplied with the item poses after a certain time—11 s for stow and 13 s for pick, since this was the average perception time needed in the ARC final. Object poses were generated by uniformly sampling positions inside the storage bin and the tote with fixed orientation.

We averaged the time needed for solving the task with different grasp success probability values over ten runs each. For the one arm pick task experiments the unreachable box was symmetrically placed next to the used arm.

Figure 14 shows the results. If only one arm is used, the system needs on average 1.2 to 1.3 times longer to complete the task. The large configuration space of grasp success rate, item locations, requested order, and order of detection result in a high standard deviation, nevertheless the trend is clearly observable.

VII. Lessons Learned

Overall, several design choices were validated at the ARC 2017 or have been proven suboptimal. Our strong focus on the object perception pipeline and efficient execution of the tasks, as opposed to more complicated mechanical solutions, was very successful. We also learned that even such dynamic tasks requiring fast adaption to new items are within reach of current mainstream deep learning approaches, if one can parallelize the training and makes proper use of pretraining.

In retrospect, we could have minimized the execution time further by optimizing our storage system layout. The dual-arm speed-up from factor 1.3 to 1 is slightly disappointing and is mostly limited due to resource conflicts, e.g. both arms wanting to place in the central box. A different placement of boxes or more global planning could alleviate these conflicts.

As always with robotics competitions, proper full-scale testing is important, both for the system as well as the operators. On the operator side, we had mistakes during our stow slot. On the system side, we noticed precision problems with our scales quite late in the competition, which might have cost us the first place in the finals.

VIII. Conclusion

In this paper, we presented our system designed for the ARC 2017. Our object perception pipeline is able to be quickly adapted to new items, to produce precise item contours, infer grasp poses, and predict 6D item poses. We demonstrated how to quickly plan and coordinate two arms operating in the same workspace. Our very good results at the ARC 2017 and our quantitative experiments show the effectiveness of our approach.

In future work, we will investigate replacing heuristics in our system with trainable modules to further robustify the item perception pipeline.

References

[1] C. Hernandez, M. Bharatheesha, W. Ko, H. Gaiser, J. Tan, K. van Deurzen, M. de Vries, B. Van Mil, J. van Egmond, R. Burger, M. Morariu, J. Ju, X. Gerrmann, R. Ensing, J. V. Frankenhuyzen, and M. Wisse, "Team Delft's robot winner of the amazon picking challenge 2016," *ArXiv:1610.05514*, 2016.

[2] E. Matsumoto, M. Saito, A. Kume, and J. Tan, "End-to-end learning of object grasp poses in the amazon robotics challenge,"

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

[4] M. Schwarz, A. Milan, A. S. Periyasamy, and S. Behnke, "RGB-D object detection and semantic segmentation for autonomous manipulation in clutter," *Internation Journal of Robotics Research (IJRR)*, 2017.

[5] G. Lin, A. Milan, C. Shen, and I. Reid, "Refinenet: Multi-path refinement networks with identity mappings for high-resolution semantic segmentation," *ArXiv:1611.06612*, 2016.

[6] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *ArXiv preprint arXiv:1606.00915*, 2016.

[7] C. Smith, Y. Karayiannidis, L. Nalpantidis, X. Gratal, P. Qi, D. V. Dimarogonas, and D. Kragic, "Dual arm manipulation—A survey," *Robotics and Autonomous systems*, vol. 60, no. 10, pp. 1340–1353, 2012.

[8] C. Bersch, B. Pitzer, and S. Kammel, "Bimanual robotic cloth manipulation for laundry folding," in *International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2011, pp. 1413–1419.

[9] P. Hebert, N. Hudson, J. Ma, and J. W. Burdick, "Dual arm estimation for coordinated bimanual manipulation," in *IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2013, pp. 120–125.

[10] K. Harada, T. Foissotte, T. Tsuji, K. Nagata, N. Yamanobe, A. Nakamura, and Y. Kawai, "Pick and place planning for dual-arm manipulators," in *International Conference on Robotics and Automation (ICRA)*, IEEE, 2012, pp. 2281–2286.

[11] S. Akella and S. Hutchinson, "Coordinating the motions of multiple robots with specified trajectories," in *International Conference on Robotics and Automation (ICRA)*, IEEE, vol. 1, 2002, pp. 624–631.

[12] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *European Conference on Computer Vision (ECCV)*, Springer, 2016, pp. 630–645.

[13] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," *ArXiv:1611.05431*, 2016.

[14] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *ArXiv:1412.6980*, 2014.

[15] D. Garcia-Castellanos and U. Lombardo, "Poles of inaccessibility: A calculation algorithm for the remotest places on earth," *Scottish Geographical Journal*, vol. 123, no. 3, pp. 227–233, 2007.

[16] D. Ferstl, C. Reinbacher, R. Ranftl, M. Rüther, and H. Bischof, "Image guided depth upsampling using anisotropic total generalized variation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 993–1000.

[17] M. Schwarz, T. Rodehutskors, D. Droeschel, M. Beul, M. Schreiber, N. Araslanov, I. Ivanov, C. Lenz, J. Razlaw, S. Schüller, D. Schwarz, A. Topalidou-Kyniazopoulou, and S. Behnke, "NimbRo Rescue: Solving disaster-response tasks through mobile manipulation robot Momaro," *Journal of Field Robotics (JFR)*, vol. 34, no. 2, pp. 400–425, 2017.

Preprint submitted to 2018 IEEE International Conference on Robotics and Automation. Received September 22, 2017.