IEEE International Conference on Robotics and Automation (ICRA), Brisbane, Australia, May 2018.

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 picking from cluttered bins is a demanding task, for which Amazon Robotics holds challenges. The 2017 Amazon Robotics Challenge (ARC) required stowing items into a storage system, picking specific items, and packing them into boxes. In this paper, we describe the entry of team NimbRo Picking. Our deep object perception pipeline can be quickly and efficiently adapted to new items using a custom turntable capture system and transfer learning. It produces high-quality item segments, on which grasp poses are found. A planning component coordinates manipulation actions between two robot arms, minimizing execution time. The system has been demonstrated successfully at ARC, where our team reached second places in both the picking task and the final stow-and-pick task. We also evaluate individual components.**

I. Introduction

In order to successfully approach robotic bin picking, multiple research fields ranging from computer vision to grasp planning, motion planning, and execution control need to be tightly coupled. Especially the case of cluttered bin picking, i.e. the picking of randomly arranged items of different types, is the focus of active research. In order to advance the state of the art, Amazon hold annual challenges: 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, the 2017 ARC allowed participants much more leeway with regards to 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 well 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 deep semantic segmentation pipeline which can be adapted to new items on-the-fly, and
- a method for online dual-arm coordination planning and execution control for complex picking or stowing tasks.

*All authors are with the Autonomous Intelligent Systems group of University of Bonn, Germany; schwarz@ais.uni-bonn.de

[1]http://phx.corporate-ir.net/phoenix.zhtml?c=176060&p=irol-newsArticle&ID=2290376

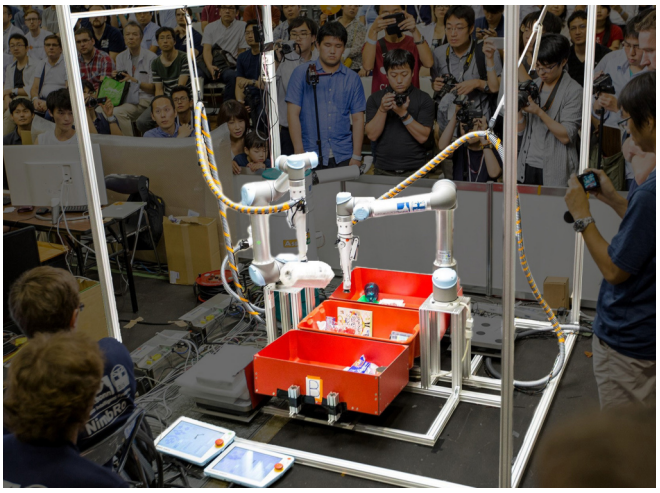

Fig. 1. Our system at the Amazon Robotics Challenge 2017 in Nagoya, performing the stow phase of the final task. Image by Amazon Robotics.

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. 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 motivated us to build a fixed sensor gantry for our system.

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. It 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.

In recent years, research on semantic segmentation advanced significantly. Large datasets allow the training of increasingly complex models (e.g. [5], [6]), but few works

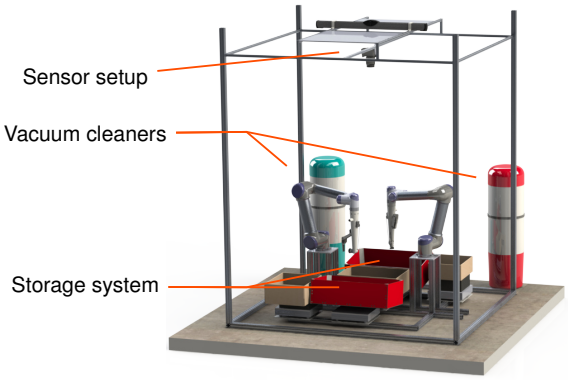

Fig. 2. CAD model of the entire system.

focus on fast item capture and training, as required for ARC.

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 online coordination of independent manipulation in a shared workspace is under-researched. 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 objectives: 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, in the finals no team completely stowed and picked all required items.

A. Arms and Grippers

Our experience from APC 2016 year told us that suction is a very powerful tool for bin picking—we could manipulate all items using suction at APC 2016, but had to develop special grasp motions for specific items. Since for ARC 2017 half of the items in the competition were unknown beforehand, we wanted to be prepared for items requiring mechanical grasping. To this end, we developed a hybrid gripper, similar to other top APC 2016 teams.

To address our second design goal, speed, we developed 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.

Our robot system consists of two 6 DoF Universal Robots UR5 arm. Each arm is equipped with an end effector with

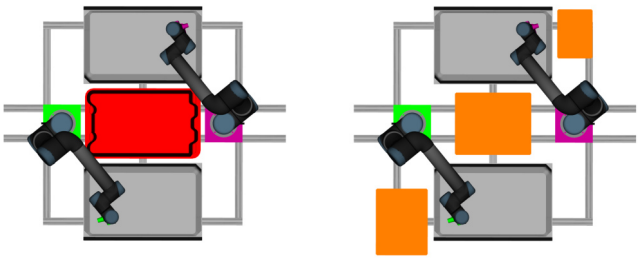

Fig. 3. System setup for both tasks. Storage system bins are depicted in gray. Left: Configuration with tote (red) for the stow task. Right: Cardboard boxes (orange) for the pick task.

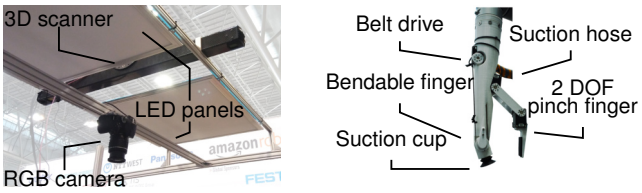

Fig. 4. Left: Gantry setup. Right: 3 DoF suction gripper.

a bendable suction finger and a 2-DoF second finger (see Fig. 4). The high air flow needed for imperfectly sealed suction grasps is generated by two powerful vacuum cleaners, one for each arm. The vacuum can be released through an actuated bleed valve.

B. Storage System

The Amazon-provided red tote contains the items to be stored. It is easily accessible from the top for perception and manipulation. Our storage system meets the maximum allowed volume and area constraints. To match the perception and manipulation situation to the tote, we divided the available volume into two similarly shaped bins and made then red as well. 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 placed 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. An industrial scale (1 g absolute precision) is mounted under each of the five possible pick and place locations for detecting contact and for confirming picks by checking the item weight.

C. Gantry Sensors

For workspace perception, 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[2]. Both sensors are mounted on a gantry approx. 2 m above the storage system and tote. This configuration allows for observing the entire workspace without moving the sensors. Two LED panels provide diffuse lighting to reduce the influence of uncontrolled outside light.

[2] http://photoneo.com/product-detail/phoxi-3d-scanner-xl

Fig. 5. Turntable capture and automatic segmentation. Left: Input image. Right: Extracted segments in standing and lying configuration.

IV. Object Perception

A. Item 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 (see Section VI-B). 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 item views using an automated turntable setup (see Fig. 5), as used for many RGB-D object datasets [12]. The turntable is equipped with a Nikon D3400 camera and LED panels (as on the gantry) and an Intel RealSense SR300 RGB-D sensor. It captures twenty views from all sides in 10 s. A typical item can be scanned in three different resting poses in about a minute (including manual repositioning), resulting in sixty views.

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 tuned using a graphical user interface. The background color is exchangeable, but we did not need to do so during the ARC 2017 competition—even objects with red parts could be reliably extracted from red background after manual tuning of the extraction thresholds. Note that a red background creates similar effects on transparent objects as if they were placed in the red tote or storage system.

B. Semantic Segmentation Architecture

In our previous APC 2016 entry [3], we demonstrated gains from enhancing semantic segmentation with results from object detection, which produces less spatial detail in its bounding box outputs, but has a better notion of "objectness" and detects entire object instances—which helps to eliminate spurious segmentation results. For ARC 2017, we decided to go with a pure semantic segmentation pipeline. This decision was motivated by i) the small gain obtained by the hybrid pipeline and ii) 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-precise segmentation instead of just bounding box-based object detection is a big 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 [13], extracted after each of the four ResNet blocks. Since the features become 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 [14].

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.

C. Cluttered Scene Synthesis & Fast Training

As mentioned above, a key requirement for our system 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 a turntable without occlusions, we generate new synthetic scenes for training (see Fig. 6).

This scene generation is done 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. The new items are allowed to occlude each other to simulate complex arrangements. The scene generation part runs purely on CPU and is multithreaded to achieve maximum performance. As the scene generation is faster than CNN training, we can generate a new scene for every training iteration—ensuring that the model does not overfit to specific arrangements.

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[3]. Using one scene generation pipeline per GPU card, we can obtain 100% GPU utilization. During ARC 2017, the network was trained on four Titan X (Pascal) cards.

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 [15]. We pretrain the network on the set of 40 known objects, and then finetune during the competition for the new objects. After every

[3] https://github.com/NVIDIA/nccl

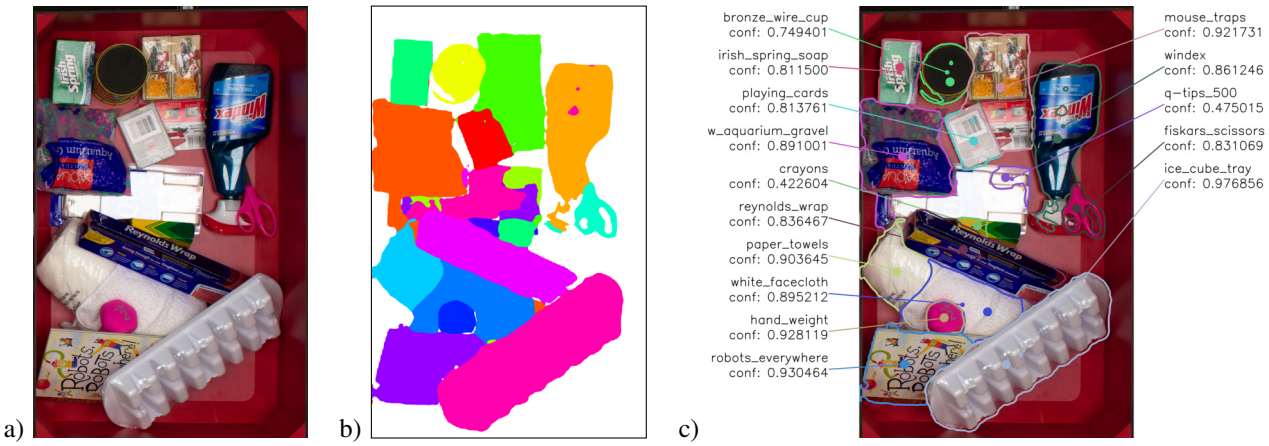

Fig. 7. Object perception example from the picking phase of our finals run at ARC 2017. The original model trained during the run was used. a) RGB image captured by the Nikon camera. b) Segmentation output. c) Processed item contours with average confidences, polygon center of mass (small points), and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

epoch, the filesystem is scanned for new turntable captures and, if required, the classification layers are adapted to a new number of classes by extending the weight tensors with random near-zero values.

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 [16]. For fast computation, we use an approximation algorithm[4].

For most lightweight items, the pole of inaccessibility suffices, which reduces the risk of missed grasps. For heavy items, it is more important to grasp close to the center of mass. To this end, we also check the 2D polygon center of mass and compute its distance d_m to the contour. If $\frac{d_m}{d_p} > \tau$, we prefer to grasp at the center of mass. We use a threshold $\tau = 0.8$ for lightweight and $\tau = 0.4$ for heavy items (weight>800 g). 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], [17]. 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 point the second finger towards the bin center, to avoid collisions. We add Gaussian noise on translation ($\sigma = 1.5$ cm) and rotation ($\sigma = 60°$), in order to obtain different grasp poses for each manipulation attempt.

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

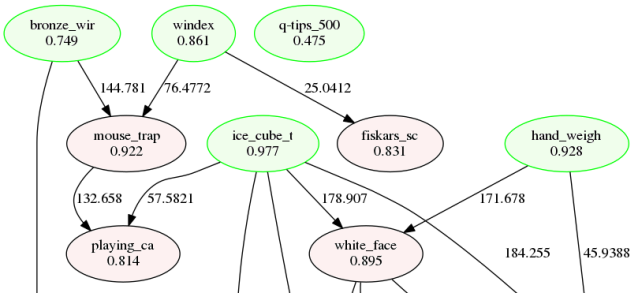

Fig. 8. Clutter graph for the scene in Fig. 7. The bottom half is cut off, leaving only the items on top of the pile. Vertices contain the class name and detection confidence. Green vertices have no predecessor. Edges are labelled with the point count (predecessor higher than successor).

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 that would need to be removed to get access to the item of interest. For this reason, we generate a directed graph that we call *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 (as evidence for the occlusion) 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, the graph may still contain cycles, which would block certain items from ever being removed. We resolve this situation by deleting the set of edges with minimum point count sum (i.e. minimum evidence) that makes the graph acyclic. This is called the minimum feedback arc set and is NP-hard [18], but for our small graphs we can quickly compute a brute-force solution. This method both reliably

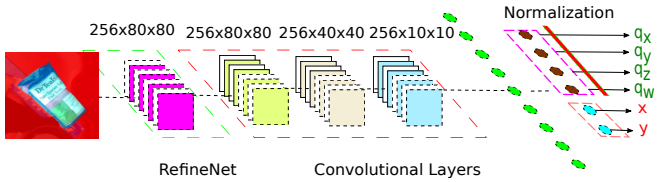

Fig. 9. Pose estimation network architecture.

removes cycles caused by measuring errors and forces an acyclic solution in cases where the occlusion is actually cyclic—allowing the system to try and extract an item from the cycle, since no other action is possible. The result is a directed acyclic graph containing the occlusion information.

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. To make this possible, we developed a 6D pose estimation module, which would allow specifying grasps relative to an item frame. Our method does not compute a fused 3D item model, which can be difficult to obtain for transparent objects. Instead, we train an additional CNN on predicting the pose from individual views.

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 origin. The network consists of the RefineNet backbone as in the semantic segmentation network, followed by three convolution layers and two fully connected layers. For M item classes, the network predicts $6M$ values—one quaternion and translation offset per item class. In this way, the predictor is conditioned on the object class, which is inferred by the segmentation network.

During training, the object 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.

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 [19]).

B. Inverse Kinematics

For keyframes defined in Cartesian space, we use a selectively damped least square (SDLS) solver, 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 roughly 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].

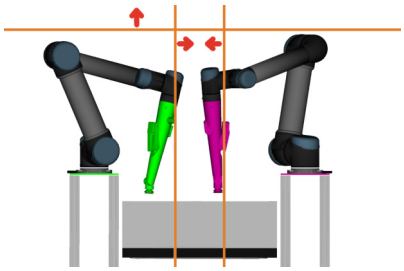

Fig. 10. Cost function planes for the IK solver. The planes affect the wrist of the robot. The vertical plane keeps the endeffector vertical, as long as the horizontal planes are not active (purple robot). The horizontal planes keep the wrist away from the robot base to prevent self-collisions.

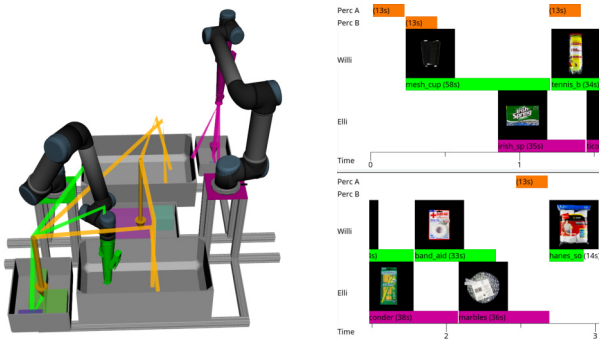

Fig. 11. Planning for the pick task. Left: Visualization of manipulation tasks. Chosen tasks are marked in green and purple. Right: Timeline of actions including perception time and arm motions.

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 according to 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.

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.

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 $\mathcal{A}$, currently possible task items $\mathcal{B}$, and items which will be picked later $\mathcal{C}$. 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 $\mathcal{A}$ have a fixed position and items from $\mathcal{B}$ have to be placed before items from $\mathcal{C}$. 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 $\mathcal{B}$.

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. 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. To enable parallel work, 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 (from the clutter graph) and, finally, the best half of these are considered as possible tasks.

Since manipulation is performed open-loop after perception, we allow only 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 to 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 higher confidence.

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.

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	

F. Grasp Execution

In both tasks (pick and stow) after each perception run, a predefined per-object probability decides which grasp type (suction or pinch) should be performed. Our 2016 system suffered from grasping wrongly identified items during the stowing task—a failure case which easily leads to incorrect internal states of the high-level control, cascading the failure and creating even more problems. Due to this experience, we wanted to eliminate false positives for ARC 2017 by double-checking perception and manipulation using a second modality. To this end, after grasping and lifting an item, the item weight is measured with the scale mounted below the container 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 again and increment the fail counter. This check also protects the system from accidentally grasping more than one item.

VI. EVALUATION

We evaluated our work on a system level at the Amazon Robotics Challenge 2017, which was held at RoboCup in Nagoya, Japan. 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 abilities of our system. In our practice slot, we successfully 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 necessary replacement and reconfiguration work

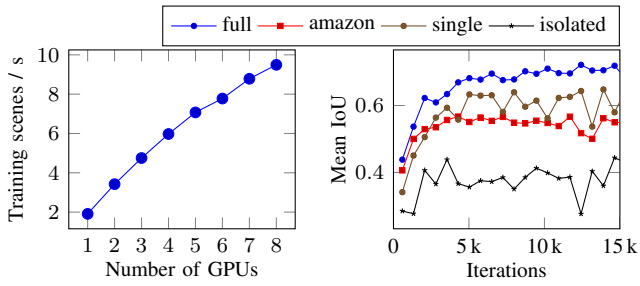

Fig. 12. Segmentation experiments. Left: Training image throughput depending on the number of GPUs. Right: Test set IoU during training.

TABLE II
ABLATION EXPERIMENTS FOR SCENE SYNTHESIS

Variant	full	amazon	single	isolated
Turntable images[1]	✓		✓	✓
Rendered objects[2]	5	5	1	1
Complex background[3]	✓	✓	✓	
Mean IoU	**0.720**	0.571	0.642	0.375

[1] Otherwise the scene generation only uses Amazon images.
[2] Number of rendered objects per synthetic scene. Note that one object means that this object is never occluded.
[3] Whether complex images (full totes / storage systems) or empty totes are used as background for synthetic scenes.

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, which 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 lowered the overall runtime and allowed for more manipulation attempts in a given time. Overall, our system performed very well and showcased all of its abilities at the ARC 2017.

TABLE III
POSE ESTIMATION ERRORS

	Translation [pix]		Angular [°]	
	train	val	train	val
Salts	2.28	3.32	1.80	3.19
Paper	2.41	3.79	1.68	3.09
Windex	2.25	3.41	1.86	2.78
Average	2.31	3.51	1.78	3.02

B. Semantic Segmentation

After the ARC, we annotated the collected images during our final run 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 5,000 to 10,000 iterations the curve saturates. Using four GPUs, as during the ARC, this occurs after approx. 15–30 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 time.

We performed an ablation study to determine the usefulness of individual scene synthesis steps (see Table II). Training on scenes with objects rendered from our own turntable images strongly outperforms using only the Amazon-provided object images. This may be due to both insufficient number of views (up to six in the Amazon data vs. 40–60 in ours) and our red turntable background, yielding realistic transparency response for objects in the red tote or storage system. Creating occlusions on the rendered objects is important. Finally, training on isolated scenes with only one object in an empty tote yields poor performance, maybe indicating that our background images with complex arrangements help regularizing the segmentation.

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 III 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.

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 full runs were performed in simulation.

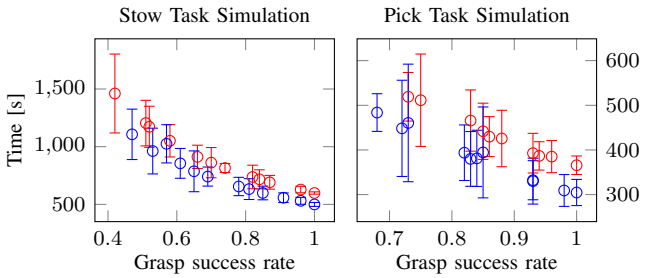

Fig. 13. Averaged run time with standard deviation (10 runs each) in simulation for stow and pick task with one (red) and two arms (blue) used.

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 13 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 & Conclusion

The ARC 2017 also allowed us to evaluate our fundamental design choices. 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 reduced the execution time further by optimizing our storage system layout. The dual-arm speed-up from factor 1.3 to 1.2 is slightly disappointing and is mostly limited due to resource conflicts, e.g. both arms wanting to place in the central box. In our design, we minimized the arms' common workspace while ensuring that storage bins and tote could be reached by both arms. However, a different placement of boxes or more global planning could potentially alleviate the 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 made 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.

Overall, we demonstrated a successful solution 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.

Acknowledgment

This research has been supported by grant BE 2556/12-1 (ALROMA) of German Research Foundation (DFG). We gratefully acknowledge travel support provided by Amazon Robotics and the PhoXi® 3D-Scanner XL, which was provided by Photoneo free of charge.

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," in *RoboCup 2016: Robot World Cup XX*, Springer, 2017.

[2] E. Matsumoto, M. Saito, A. Kume, and J. Tan, "End-to-end learning of object grasp poses in the Amazon Robotics Challenge," in *Warehouse Picking Automation Workshop (WPAW), ICRA*, 2017.

[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," *Int. Journal of Robotics Research (IJRR)*, 2017.

[5] G. Lin, A. Milan, C. Shen, and I. Reid, "RefineNet: Multi-path refinement networks for high-resolution semantic segmentation," in *Computer Vision and Pattern Recongition (CVPR)*, 2017.

[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, 2012.

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

[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)*, 2013.

[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 *Int. Conf. on Rob. and Automation (ICRA)*, 2012.

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

[12] A. Singh, J. Sha, K. S. Narayan, T. Achim, and P. Abbeel, "Bigbird: A large-scale 3D database of object instances," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2014.

[13] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *Eur. Conf. on Computer Vision (ECCV)*, 2016.

[14] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *IEEE Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017.

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

[16] 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.

[17] D. Ferstl, C. Reinbacher, R. Ranftl, M. Rüther, and H. Bischof, "Image guided depth upsampling using anisotropic total generalized variation," in *Int. Conf. on Computer Vision (ICCV)*, 2013.

[18] R. M. Karp, "Reducibility among combinatorial problems," in *Complexity of computer computations*, Springer, 1972, pp. 85–103.

[19] 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, 2017.