Designing *Cartman*: A Cartesian Manipulator for the Amazon Robotics Challenge 2017

Jürgen Leitner, Douglas Morrison, Anton Milan, Norton Kelly-Boxall, Matthew McTaggart, Adam W. Tow, and Peter Corke

1 Introduction

Efficient warehouse pick and place is an increasingly important industry problem. Picking specific items out of a box of objects is also one of the canonical problems in robotics. Amazon Robotics has over the years created systems that enable the automated and efficient movement of items in a warehouse. In 2015 they initiated a competition focusing on the problem of how to perform picking and packing tasks with robotic systems.

The 2017 Amazon Robotics Challenge comprised of two tasks, stow and pick, reflecting warehouse operations for online order fulfilment. These involve transferring items between Amazon's red plastic container, a storage system (previously

This work was done by J. Leitner prior to joining LYRO Robotics.
This work was done by the author "A. Milan" prior to joining Amazon.
This work was done by the author "A. W. Tow" prior to joining Dorabot.

J. Leitner (✉)
LYRO Robotics, Brisbane, QLD, Australia
e-mail: juxi@lyro.io

D. Morrison · N. Kelly-Boxall · M. McTaggart · P. Corke
Australian Centre for Robotic Vision (ACRV), Queensland University of Technology (QUT), Brisbane, QLD, Australia

A. Milan
Amazon Research, Berlin, Germany
e-mail: antmila@amazon.com

A. W. Tow
Dorabot, Tingalpa, QLD, Australia
e-mail: adam.tow@dorabot.com

A. Causo et al. (eds.), *Advances on Robotic Item Picking*,
https://doi.org/10.1007/978-3-030-35679-8_11

an Amazon provided shelf), and a selection of standard cardboard shipping boxes. Teams were required to develop a robotic solution and were free to design their own storage system within certain limitations. The robots need to visually detect and identify objects in clutter and then successfully transfer them between locations, demanding a robust integration of manipulator, object recognition, motion planning and robotic grasping.

In this chapter we present an overview of our approach and design that led to winning the 2017 competition finals, held in Nagoya, Japan. The primary differentiating factor of our system is that we use a Cartesian manipulator, nicknamed *Cartman* (Fig. 1). We find it to greatly simplify motion planning and executing in the confines of the box shaped storage system compared to articulated robotic arms. It also enabled us to use a dual-ended end-effector comprising two distinct tools. *Cartman* stowed 14 (out of 16) and picked all 9 items in 27 min, scoring 272 points. We released four tech reports explaining the various sub-systems in more detail [17, 18, 20, 29].

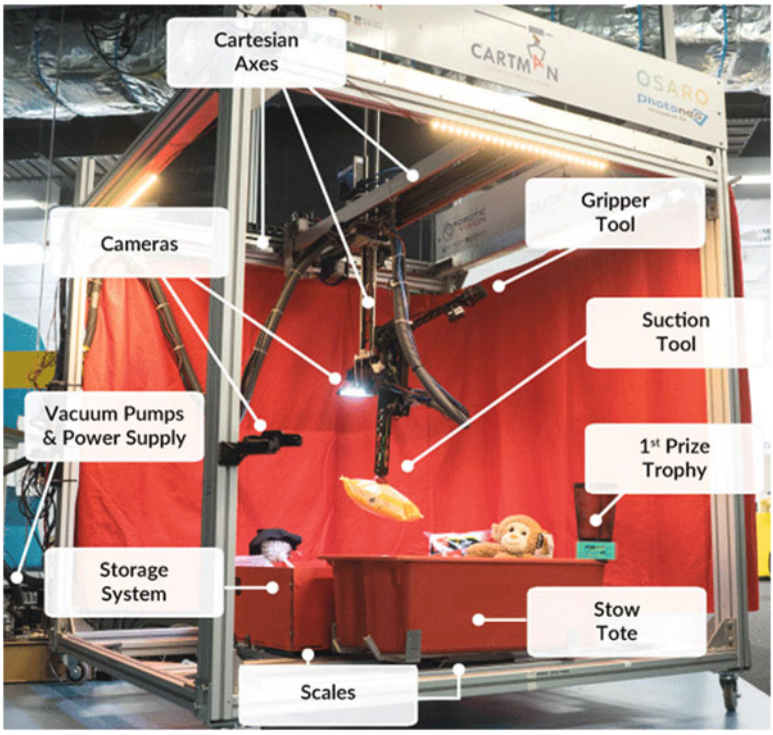

Fig. 1 *Cartman* includes three linear axes to which a wrist is attached. It holds a camera and two end-effector modalities (suction and parallel gripper) that share two revolute axes, and have an extra revolute axis each. To deal with the uncertainty we added a secondary camera, positioned on the frame to take images of picked items with a red backdrop (curtains), and scales underneath the boxes

2 Background

There is a long history of research into grasping and dexterous manipulation [28], and recent progress has largely been driven by technological developments such as stronger, more precise and more readily available collaborative robotic arms [15] and simple compliant and universal grippers [1, 19]. Robust perception is also a key challenge [4]. Over recent years, significant leaps in computer vision were seen thanks to the application of deep learning techniques and large scale datasets [5]. However, the increased performance of computer vision systems has not translated to real-world robotic improvements, highlighting deficiencies in robust perception and hand-eye coordination. Datasets and benchmarks are increasingly exploited in the robotics community to find solutions for such tasks [2, 13] yet still have some shortcomings [12]. The applied nature of competitions makes them one of the greatest drivers of progress—from self-driving cars, to humanoids, to robotic picking.

The 2017 Amazon Robotics Challenge comprised two tasks, stow and pick, analogous to warehouse assignments, which involve transferring items between Amazon's totes (a red plastic container), the team's storage system and a selection of Amazon's standard cardboard shipping boxes. Teams were required to design their own storage system within certain limitations, unlike in previous competitions where standardised shelving units were supplied. Our storage system comprises two red, wooden boxes with open tops.

In the stow task, teams are required to transfer 20 objects from a cluttered pile within a tote into their storage system within 15 min. Points are awarded based on the system's ability to successfully pick items and correctly report their final location, with penalties for dropping or damaging items, or having items protruding from the storage system.

In the pick task, 32 items were placed by hand into the team's storage system. The system was provided with an order specifying 10 items to be placed into 3 cardboard boxes within 15 min. Points were awarded for successfully transferring the ordered items into the correct boxes, with the same penalties applied for mishandled or protruding objects.

The 2017 competition introduced a new finals round, in which the top 8 teams competed. The finals consisted of a combined stow and pick task. Sixteen items were first hand-placed into the storage system by the team, followed by a vigorous rearrangement by the judges. Sixteen more items were provided in a tote and had to be stowed into the storage system by the robot. Then, the system had to perform a pick task of 10 items. The state of the robot and storage system could not be altered between stow and pick.

A major addition to the challenge compared to previous years was that not all items were known to the teams beforehand. The items for each task were provided to teams 45 min before each competition run, and consisted of 50% items selected from a set of 40 previously seen items, and 50% previously unknown items. This change introduced a major complexity for perception, as systems had to be able to

handle a large number of new items in a short period of time. This, in particular, made deep learning approaches to object recognition much more difficult.

3 *Cartman*: System Overview and Design

Most pick-and-place robots require at least a manipulator, a gripper, and a perception system. The design of these components and the overall robot system itself, must consider a range of constraints specific to the given application. These constraints will often include: the items the robot must manipulate, the environment in which the robot will operate, required operating speed, cost or return on investment, overall system reliability, grasp success rate, human-safe operation, etc. Herein we describe our final system design (both hardware and software) and how it met the constraints of the competition. We also discuss our design process, of which, we argue, contributed largely to our success in the competition.

Teams almost exclusively competed with articulated robotic arms [4], yet the task of picking items from a storage system is mostly a linear problem, which can easily be broken down into straight line motions. We had previously, in the 2016 challenge, competed with a Baxter robot [13], and encountered difficulties in planning movements of its 7-DoF arm within the limited confines of the shelf. Linear movement of an articulated arm requires control of multiple joints, which if not perfectly executed may result in sections of the arm colliding with the environment.

Our 2017 challenge robot, *Cartman*, in contrast, greatly simplifies the task of motion planning within a storage system due to with its Cartesian design and ability to move linearly along all three axes. We find considerable benefits of using such a design in warehouse pick-and-place tasks:

- *Workspace*: Cartesian manipulators have a compact rectangular workspace, compared to a circular of arms. An advantage particularly for the ARC constraints.
- *Simplicity*: Motion planning with Cartesian robots is simple, fast, and unlikely to fail even in proximity to shelving.
- *Predictability*: The simple, straight-line motions mean that failed planning attempts and erratic behaviour are less likely.
- *Reachability*: Cartesian design results in improved reachability (see Fig. 3) within the workspace and lowered chance of collision compared to arms.

The software developed for *Cartman* is utilising ROS (Robot Operating System) [23], which provides a framework for integrating the various sensors, processing sub-systems, and controllers. As per the regulation of the challenge, it is fully open-source and freely available for download.[1]

[1] https://github.com/warehouse-picking-automation-challenges/team_acrv_2017.

Storage System In 2017 the teams were allowed to design their own "storage system", replacing the previously used Kiva shelf provided by Amazon. The constraints were a total bounding-box volume of 5000 cm^3 (roughly the volume of two Amazon totes) and between 2 and 10 internal compartments. This major difference to the previous competitions opened a new design space for the teams. Our design consists of two red, wooden boxes approximately matching the dimensions of the Amazon totes, and opts for a horizontal (top-down picking) design instead of a vertical shelf-like design, which the large majority of teams concluded to be the more reliable approach. The similarity in colour to the totes allows the same perception system to be used for both the tote and the storage system.

4 Mechanical Design

As mentioned, *Cartman* employs a Cartesian design as depicted in Fig. 1. Articulated manipulators are common for many robotics applications due to their versatility and large workspace with respect to their mechanical footprint. They though have singularities and discontinuities for certain end-effector configurations [3, 9]. There are ways of reducing but not eliminating these drawbacks on motion planning [6, 21]. Using a Cartesian manipulator with a wrist joint to work in a Cartesian workspace eliminates almost all singularities and discontinuities (see Fig. 3 for a comparison of *Cartman* and other robot arms). On the other hand the disadvantage with a Cartesian manipulator is the requirement for a larger mechanical footprint in ratio to the overall workspace of the manipulator. This is due to the fact that the linear motion requires some form of support, usually a rail, along the entire length of the axis. This is less of a problem in warehouses and other conveyor-belt type operations, where the operation space can be completely enclosed by the robot.

4.1 Specifications

The entire manipulator system is mounted on frame of aluminium extrusions and rails for the three axes (Fig. 2). Apart from the three linear axes it consists of a wrist joint controlling roll, pitch, and yaw of a multi-modal end-effector also developed for the challenge. The following specifications were set out before designing the manipulator for the Amazon Robotics Challenge:

- A reachable workspace of 1.2 m × 1.2 m × 1.0 m
- A top linear velocity of 1 m/s under load along the three linear axes ($X/Y/Z$).
- A top angular velocity of 1rad/s under load along the angular axes (roll/pitch/yaw).
- A payload capacity of 2 kg.

- Six DoF at the end-effector, given by three linear axes forming the Cartesian gantry and a three-axis wrist.
- Ability to be easily de/constructed to simplify transport of the robot overseas to the event.

4.2 Mechanical Components

The manipulator frame is mounted on a stand of standard aluminium extrusions as seen in Fig. 2. The **frame** consists of laser cut and folded 1.2 mm sheet aluminium for housing the rails, providing a great trade-off between stability and weight. The reduced weight was important as the robot was to be transported overseas for the competition. The sheet aluminium formed the main outer frame housing the X-axis belt system and transmission rod.

The linear **rails** used for the X- and Y-axes are TBR20 and TBR15 precision type profile rail, respectively. Smaller rails were used in the Y-axis to reduce the overall weight of the system. The Y-axis consists of two 10 mm round rails. The downside to using 10 mm rails, however, is that when the Z-axis is extended a

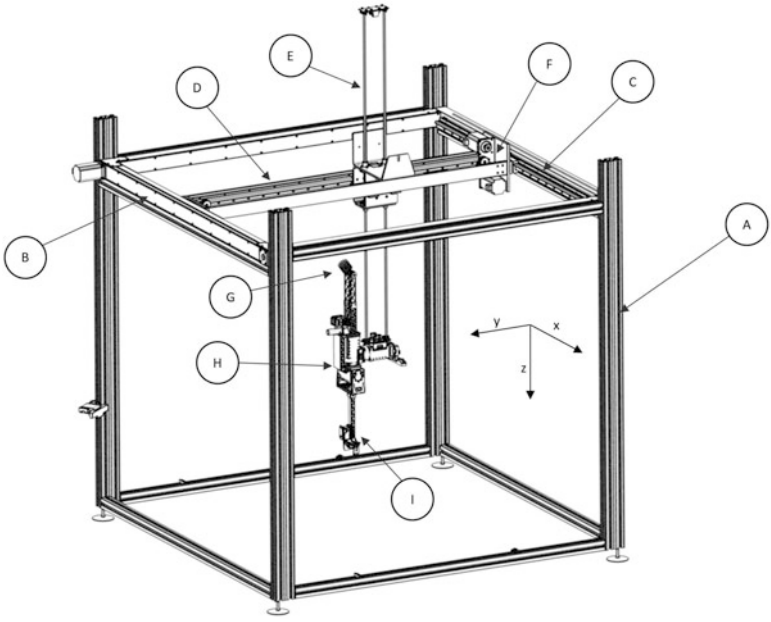

Fig. 2 Isometric view of the entire manipulator. Key components have been labelled and are as follows: (**a**) Aluminium T-slot stand, (**b**) Manipulator Aluminium frame, (**c**) X-axis TBR20 profile rails, (**d**) Y-axis TBR15 profile rails, (**e**) Z-axis 10 mm round rails, (**f**) Y–Z motor-carriage, (**g**) Suction gripper, (**h**) Wrist, (**i**) Parallel plate gripper

pendulum effect is created inducing oscillations at the end-effector due to the rail's deflection. Although deflection and oscillations are present, steady state accuracy is still achieved with ease once settled. We considered this trade-off during the design process. Additionally, the oscillations are minimised by raising the Z-axis when performing large translational movements.

Linear motion is performed by Technic's ClearPath SD-SK-2311S **motors**. They are closed-loop brush-less motors designed to be a drop in replacement for stepper motors, eliminates the need for external encoders. They were chosen due their high performance and ease of use. Three Dynamixel Pro L54-50-500 are controlling the roll, pitch, and yaw axes. These provide the necessary operating torque to hold a 2 kg payload while under acceleration.

To actuate the prismatic joints, a belt and pulley system was used. A single motor drives the X-axis. In order to eliminate a cantilever effect on this, a transmission rod is used to transmit power from one member to another. One common design that has been observed in a lot of simple manipulator designs is each axis motor needs to carry the weight of all distal motors as well as the payload. As a result, more powerful motors are required which increases weight as well as cost. To solve this problem a differential belt system was designed. Rather than using a single motor to drive a single axis, two motors work in tandem to drive two axes (Fig. 3).

4.3 Software and Electrical Components

A single microcontroller is used to control all six axes. We employed a Teensy 3.6 with a breakout board including a logic shifter circuits for each of the ClearPath motor pins. In order to interface with the Dynamixel Pro motors, an RS485 module was added. ROS JointState message type which was processed by the Teensy. The low level firmware functions send commands to both the ClearPath

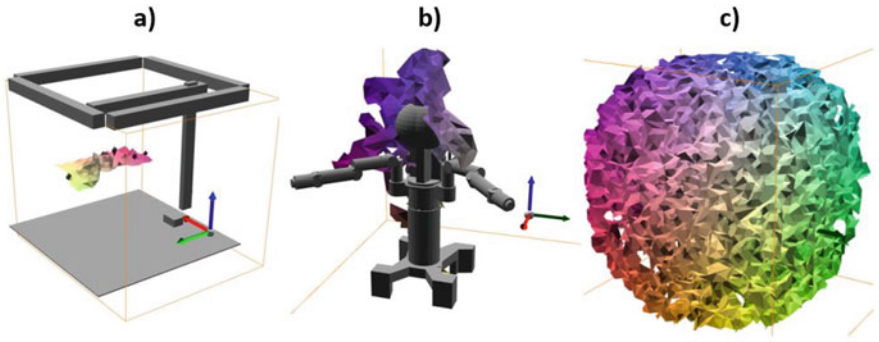

Fig. 3 Discontinuity maps of (**a**) *Cartman*'s end-effector, (**b**) a Baxter's left gripper, and (**c**) a UR-5. If the end-effector passes through these boundaries, joint velocities can accelerate to infinity. *Cartman*'s design limits these discontinuity boundaries, making planning simpler and safer

and Dynamixel Pro motors and also read any feedback that was available from the motors. As the ClearPath motors are a drop-in replacement for stepper motors and as such the open source AccelStepper library [16] is used. It provides the ability to deploy an acceleration profile per motor. The Dynamixel Pro is controlled using a slightly modified version of the OpenCR library [10]. ROS (the Robot Operating System) [24] is handling the higher-level (system) functionality. Desired end-effector poses and robot states are ROS messages published by the microcontroller and used by the MoveIt! package [27], which was the interface to the state-machine.

The complete design is open sourced and available online.[2] For a more in-depth analysis of the design of *Cartman* readers are referred to our tech report [17].

4.4 Multi-Modal End-Effector

The challenge requires teams to pick a very diverse set of items, including rigid, semi-rigid, hinged, deformable, and porous objects. We employ a hybrid end-effector design (Fig. 4) comprising vacuum suction and a parallel plate gripper. Due to the use of a Cartesian system, we do not have to combine suction and gripping into a single tool, leading to a less complex end-effector design (e.g. [7, 25]). We integrate these two distinct tools at the wrist, to be swapped by a single motor rotation. A further advantage, particularly during development, was that this design allows each tool to be developed and tested individually reducing dependencies and downtime.

The grasping system relies on a tight coupling of the physical end-effector and its relevant software components, which are discussed in Sect. 5.5.

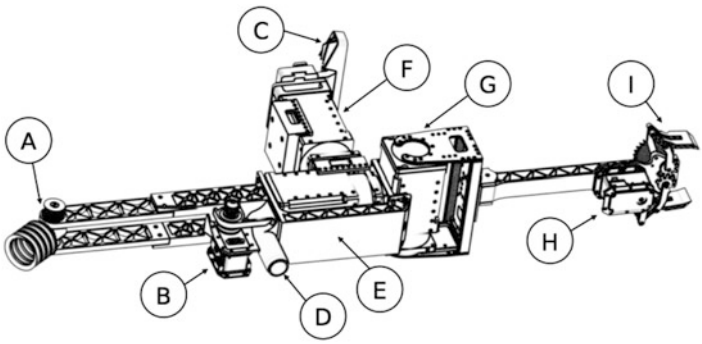

Fig. 4 End-effector Assembly. (**a**) Rotating suction cup. (**b**) Suction gripper pitch servo (drive belt not pictured). (**c**) wrist-mounted RealSense camera. (**d**) suction hose attachment. (**e**) Roll motor. (**f**) Yaw (tool-change) motor. (**g**) Gripper pitch motor. (**h**) Gripper servo. (**i**) Parallel plate gripper

[2] http://Juxi.net/projects/AmazonRoboticsChallenge/.

Vacuum Suction Tool We opted for vacuum suction as the primary grasping mechanism, based on previous year's experience [4]. It lends itself to simpler grasp planning, as only a single, rotation-invariant, grasp point is required to be detected. It has in the past shown an outstanding ability to grasp a large range of items.

The suction system consists of a 40 mm diameter multi-bellow suction cup. It can be rotated in a full semi-circle at its base, allowing it to attach to vertical surfaces inside cluttered environments. Various approaches were tested before settling on a belt-driven (by a Dynamixel RX-10 servo) implementation, which had the best trade-off in terms of overall size, rotation, and reachability. The suction gripper is our default tool, with the parallel gripper being specified for porous or deformable objects which create hardly any vacuum seal, e.g., cotton gloves or marbles.

To get a working prototype for full end-to-end system testing, the initial end-effector design was a simple, rigid suction tool with no articulation (Fig. 5a), based on our previous design [13]. Even using just the initial design, our first *Cartman* prototype was able to attach to 80% of the objects provided. The main limitations identified from the first test were: firstly, to incorporate gripping as a secondary grasping modality is required, and secondly, improving robustness, particularly in more challenging, cluttered environments. We observed drastically reduced grasp success when unable to approach the grasp at a perpendicular angle, especially encountered with objects leaning against the side of the storage system.

To overcome this articulation of the suction tool was added. At first the entire tool arm was moved in an attempt to keep the physical footprint to a minimum (Fig. 5c). However, this still required a clear approach trajectory. An extra degree of freedom was therefore added to the suction cup. In the final design an extra motor is added, as well as the "arm" extended to 240 mm. The belt-driven design pivots the suction cup closer to the end-point (Figs. 5d), keeping the footprint to a minimum. This

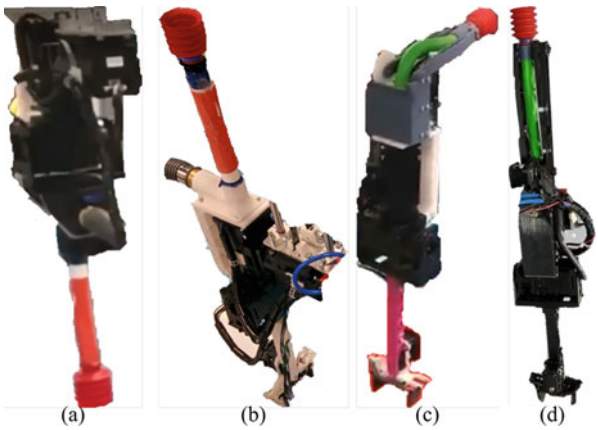

Fig. 5 Four major stages of the end-effector design, as discussed in Sect. 4.4. (**a**) Static suction tool. (**b**) Addition of gripper. (**c**) Extra degree of freedom on suction tool. (**d**) Final design

design allows for 6 degrees of control at the suction end-point, reducing the issues observed in earlier versions.

Parallel Gripper The subset of known Amazon items that could not be picked with our suction system were predominantly those which were porous, or too thin for a suction cup to be attached. We opted to use a parallel gripper as the second grasping modality. A survey of commercially available and open-source parallel grippers did not yield a promising solution for Amazon's challenge tasks, with the options either too large or could not easily be modified.

We opted therefore, to design a custom gripper purpose-built for the challenge with integration to our Cartesian manipulator in mind. By limiting our design parameters to the set of objects which could not be grasped via suction, we created a highly customised solution without any unnecessary overhead, limiting the design space for our gripper. The final design of the parallel jaw gripper uses a single actuating servo and has a stroke of 70 mm, and a maximum grasp width of 60 mm. A 190 mm extension arm connects the wrist to the gripper to ensure it can reach within the Amazon tote.

Our gripper plates feature slightly angled tips (after trialling various designs) to scoop thin objects, such as the scissors, and create a more powerful pinch at the tip for picking deformable objects, such as the sponges or gloves. In addition, high-friction tape add a small amount of compliance to the gripper, specifically helping with rigid objects, while increasing the overall grasp success rate.

For a more in-depth analysis of the end-effector design readers are referred to our tech report [29].

5 Perception System

Manipulating individual items in a large warehouse is a complex task for robots. It is challenging both from a mechanical but also from a computer vision viewpoint. Even though the environment can be controlled to a certain degree, e.g., the storage system can be custom designed to facilitate recognition, the sheer number of items to be handled poses non-trivial challenges. In addition, the items are often placed in narrow bins to save space, thus partial or even full occlusion must be addressed from both the perception and manipulation side.

5.1 Perception Pipeline

The perception system needs to deliver two key functions: (1) detection, identification, and segmentation of the correct item, and (2) the generation of a set of possible grasp points for the object (Fig. 6). The two tasks of the challenge required to detect a variable number of items in the bins, with lighting and placement creating

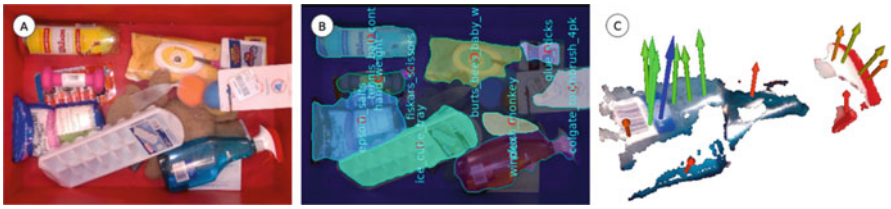

Fig. 6 The perception pipeline, showing (**a**) RGB image of items in our storage system, (**b**) output of semantic segmentation, and (**c**) segmented point cloud for the bottle, with grasp points calculated (higher ranked grasp points shown as longer and green and the best candidate in blue). Even though the semi-transparent bottle results in inaccurate depth information, our system still finds good grasp points on the centre of the bottle, with points on the neck and head being weighted lower

appearance changes throughout the competition. The biggest change, however, to previous edition was in the replacement of 50% of the training objects by new ones. These new items were presented to the participants only 45 min prior to the start of each task. These conditions require a perception system that is very robust and not fully over-fitted to the training set, yet also models that can be quickly adapted to new categories.

In our perception pipeline we perform the two mentioned key functions sequentially. We first perform an object detection, by using a fully supervised semantic segmentation solution based on deep neural networks, followed by running a grasp synthesis technique on the extracted segment. Second-placed Team NimbRo uses a similar perception pipeline, comprising a RefineNet-based architecture which is fine-tuned on synthetic cluttered scenes containing captured images of the unseen items before each competition run [25].

5.2 Perception Hardware

An Intel RealSense SR300 RGB-D camera is employed on the wrist as our main sensor. This allows to move the camera actively in the workspace, a feature we exploited by implementing a multi-viewpoint active perception approach described in Sect. 5.4.1. While the camera is light, has a small form factor and provides depth images, a drawback is the infrared projection used to determine each pixel's distance from the sensor. It is unable to produce accurate depth information on black or reflective items. We address this issue with the introduction of alternative grasp synthesis techniques for these items (Sect. 5.5). Furthermore, a second RealSense camera on the robot's frame allows for additional classification of a picked item if required.

In addition, scales are placed underneath the boxes and storage system to measure the weight of the items. These added additional functionalities to perform object identification and error detection. In terms of object classification they provide an

addition parameter to verify the correct item is grasped, that it was not dropped, and to detect when the end-effector has come into contact with an item.

A pressure switch is also included on the vacuum line to detect when a suction seal is made, and to detect dropped items.

5.3 Semantic Segmentation

To perform object identification and detection a state-of-the art semantic segmentation algorithm was implemented. It is common practice to fine-tune existing neural network models to specific tasks [8, 22, 26]. This is rather simple in cases where all categories are known and defined beforehand, it is not such a common strategy for tasks where the amount of available training data is very limited, as in the case of the Amazon Robotics Challenge.

After testing various approaches we settled on a deep neural network architecture, RefineNet [14] is able to adapt to new categories using only very few training examples. We use *RefineNet* for pixel-wise classification, and a custom vision-based (as opposed to model-based) grasp selection approach. We argue this to be a more robust and scalable solution in the context of picking in cluttered warehouse environments than those based on fitting models or 3D shape primitives, due to issues with semi-rigid, deformable, or occluded items. The semantic segmentation provides the link between a raw image of the scene and grasp synthesis for a specific object.

5.3.1 Fast Data Collection and Quick Item Learning

Due to the time-critical nature of learning the unseen items during the competition, we developed a semi-automated data collection procedure which allows us to collect images of each unseen item in 7 unique poses, create a labelled dataset and begin fine-tuning of the network within approximately 7 min. Our procedure is as follows:

- Position the wrist-mounted camera above the Amazon tote in the same pose as it would be in during a task.
- From an ordered list, place one unseen item in each half of the tote.
- Maintaining left/right positioning within the tote, change the orientation and position of each item 7 times and capture an RGB image for each pose.
- Using a pre-trained background model, automatically segment each item to generated a labelled dataset. Each automatically generated training image is manually checked and corrected where necessary.
- The new dataset is automatically merged with the existing dataset of known items.
- The RefineNet network is fine-tuned on the combined dataset until shortly before the beginning of the official run approximately 30–35 min. later.

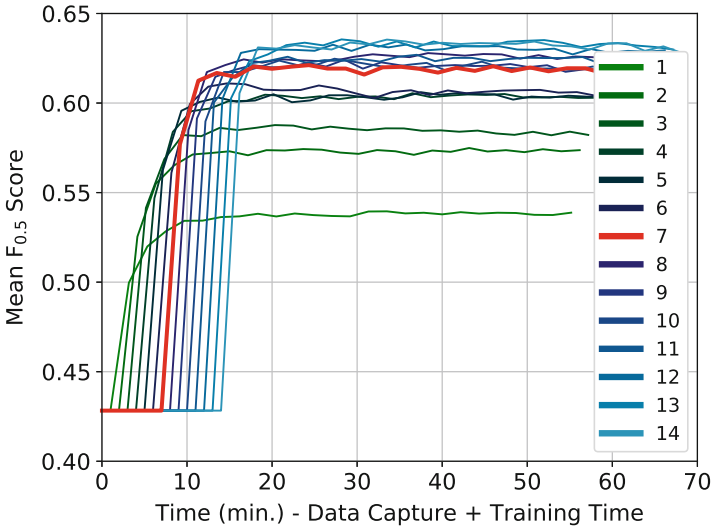

Fig. 7 Mean $F_{0.5}$ score of fine-tuned RefineNet when trained on varying numbers of images for each unseen item. The time includes data capture and training time. For the competition, we used 7 images per unseen item, trained for 13 epochs

Our data collection procedure is a trade-off between time spent on capturing data and available training time. Figure 7 shows the relative network performance for different splits between number of images captured per unseen item and training time. The performance scores are calculated on a set of 67 images representative of those encountered in competition conditions, with a mixture of 1–20 seen and unseen items in cluttered scenes. For the competition, we opted to use 7 images per unseen item, which allows enough time to repeat the data capture procedure if required.

The selection of the metric to be optimised is quite important. We use the $F_{0.5}$ metric to evaluate the performance of our perception system as it penalises false positives more than false negatives, unlike other commonly used metrics such as IOU or F_1 which penalise false positives and false negatives equally. We argue that the $F_{0.5}$ metric is more applicable for the application of robotic grasping as false positives (predicted labels outside of their item/on other items) are more likely to result in failed grasps than false negatives (labels missing parts of the item) (Fig. 8).

As a compromise to cluttered scenes, we opted to capture the images of each new item without clutter, but with as many other commonalities to the final environment as possible. To achieve this, each item was placed in the actual tote or storage container with the camera mounted on the robot's wrist at the same height above the scene as during a run. Each item was manually cycled through a number of positions and orientations to capture some of the variations the network would need to handle.

Fig. 8 An example illustrating the importance of different measures for grasping applications. Top: The object is undersegmented and the robot may pick a wrong item if a pick is executed in the segmented region. Bottom: only half of the entire object is segmented correctly. Yet $F1$ and IoU scores are very similar scores as in the above example. The $F_{0.5}$ score is higher than in the above example. We argue the $F_{0.5}$ is therefore more informative and more suitable for creating correct grasp points and successfully manipulate the objects. We argue that the $F_{0.5}$ measure is more informative. Note that precision would also be indicative of success in this example, but should not be used in isolation because it loses information about recall

To speed up the annotation of these images, we employ the same RefineNet architecture as outlined above, but trained on only two classes for binary foreground/background segmentation. After each image is captured, we perform a foreground and background separation. We parallelise this approach by placing two items in the tote at any time. Labels are automatically assigned based on the assumption that the items in the scene match those read out by a human operator with an item list.

During the data capture process, another human operator visually verifies each segment and class label, while manually correcting any flawed samples before adding them to the training set. After a few practice runs, a team of four members are able to capture 7 images of 16 items in approximately 4 min. Three more minutes are required to finalise the manual check and correction procedure.

An in-depth analysis of our RefineNet quick-training approach and a comparison with an alternative deep metric learning approach are provided in [18].

5.3.2 Implementation and Training Details

Training is performed in two steps. Our base *RefineNet* model was initialised with pre-trained ResNet-101 ImageNet weights and initially trained for 200 epochs on a labelled dataset of approximately 200 images of cluttered scenes containing the 40 known items in the Amazon tote or our storage system and an additional background class. Note that the final softmax layer contains 16 (or 10 for the stow task)

placeholder entries for unseen categories. Upon collecting the data as described above, we fine-tune the model using *all* available training data for 20 epochs within the available time frame using four NVIDIA GTX 1080Ti GPUs for training. Batch size 1 and learning rate $1e^{-4}$ is used for the initial fine-tuning stage, batch size 32 and learning rate $1e^{-5}$ is used for the final fine-tuning stage. It is important to note that we also exploit the available information about the presence or absence of items in the scene. The final prediction is taken as argmax not over all 57 classes, but only over the set of categories that are assumed to be present.

For a more in-depth analysis and comparison of the object detection system readers are referred to our tech report [18].

5.4 Active Perception

For the stow task, where all items have to be picked, we prefer picking non-occluded items at the top of the tote, which counteracts the tendency for our perception network to exhibit lower precision in more cluttered scenes. However, for the pick task, where only a subset of items have to be picked, it is likely that not all wanted items are easily visible within the storage system. In this task, we use two active perception methods to increase the chances of finding all of the wanted items.

5.4.1 Multi-View

For each storage compartment, there are three possible camera poses; one top view capturing the entire storage compartment, and two close-up views covering half of the storage compartment each. If no wanted objects are visible from the top view, the two close-up views are used, leveraging the adjusted camera viewpoint to increase the chances of finding any partially obscured items, and reducing the effective level of clutter thereby improving the performance of our perception system on the scene.

5.4.2 Item Reclassification

To be sure of a grasped object's identity, our system requires consensus from two sensors. The first is by the primary visual classification. Secondly, when a grasped item is lifted, the weight different measured by the scales is used to confirm the object's identity. If the measured weight does not match, one of the two reclassification methods is used. If the measured weight matches only one item in the source container, then the item is immediately reclassified based on weight and the task continues uninterrupted. Alternatively, if there are multiple item candidates based on weight, the item is held in view of the side-mounted camera to perform a second visual classification of the item. If the item is successfully classified as one of the candidates, it is reclassified and the task continues. If no suitable classification is given, the item is replaced and the next attempt begins.

5.5 Grasp Synthesis

The function of the grasp synthesis sub-system is to provide a final pose at which the robot should position its end-effector to successfully grasp the desired item. Our grasp synthesis assumes that the desired object is segmented and will rank possible grasp points on the point cloud segment (Fig. 6) using a hierarchy of approaches. For our vision-based grasp synthesis system, the material properties of an item are of particular importance as they affect the quality of the depth information provided by our camera. To handle a variety of cases, three different grasp synthesis strategies were developed: *surface-normals*, *centroid*, and *RGB-centroid*. If one method fails to generate valid grasp points for an item, the next method is automatically used at the expense of precision. For items where the visual information is known to be unreliable, item meta-data can be provided to weigh specific methods more.

We again use vacuum suction as the primary mode to pick an item, as such we base our grasping pipeline on previous work [11]. The heuristic based ranking is taking into account geometric and visual information, such as distance from edges and the local surface curvature. Some additional end-effector specifics are taken into account as well during the ranking process, e.g., angle to vertical and distance to the workspace edges. This design helps to ensure sensible suction points are ranked high and can be reached with our robot configuration.

If the quality of the point cloud is not allowing for valid grasp points to be detected, as is common with reflective and transparent objects, we approximate the centroid of the available depth points and select it as the grasp point. This method relies on the compliance of the end-effector hardware and the design of both grasping mechanisms which were designed specifically to handle the possible range of objects in the most robust way.

In the worst case, if no valid points are detected in the point cloud, which is most common with black objects, we approximate the centre of the object using its position in the RGB image and estimate a grasp point using the camera's intrinsic parameters. This relies on the design of the end-effector to handle uncertainty in the grasp pose, as well as the mentioned extra sensing modalities of suction detection and weight sensors for collisions.

To grasp an object with the parallel gripper a similar approach including most of the same heuristics was developed. Most importantly, the extra information about the orientation of the object is taken into account to place the gripper in the correct orientation. As a result, the parallel-gripper grasping pipeline requires only one extra processing step compared to the suction system, a simple principle component analysis of the item's RGB segment and align the gripper accordingly.

One aspect of the grasping system which sets *Cartman* apart from other participants is the ability to execute multiple grasp attempts sequentially. For suction grasps, the 3 best, spatially diverse grasp points are stored, and if suction fails on one the next is automatically tried without having to perform another visual processing step, increasing the chances of succeeding in the grasp and increasing overall system speed.

6 Design Philosophy

Our design was driven by the constraints set by the challenge, such as the limits of the workspace, objects to be handled, and the overall task description. Our approach was to test the whole system in challenge-analogue conditions to validate the progress of the overall system, not sub-systems. This integrated design approach, as we are defining it, is one that favours testing of the entire system end-to-end, using the performance of the system as a whole to drive sub-system design. This is contrary to the more commonly used modular system's design approach which favours the design and testing of sub-systems in isolation, which are then, much like a puzzle, fitted together to create the wanted system.

6.1 End-to-End Testing

The frequent end-to-end system testing and the achieved mock scores allowed us to identify the bottlenecks and focus design efforts towards areas that would lead to the largest gain in competition performance. The holistic view of the system enabled design updates to include both software and hardware, as opposed to focusing on the performance of individual sub-systems.

To facilitate rapid hardware testing and integration, rapid prototyping and manufacturing techniques, such as 3D printing and motors with modular mounting options, were employed. This allows for cheap and flexible exploration of new designs as well as the ability to easily produce spare parts in case of failure.

For example, our hierarchical grasp detection algorithm was designed to work with varying levels of visual information. This fully custom design is the result of our integrated design methodology, in which both hardware and software were developed in parallel, with an emphasis on end-to-end system testing throughout the design process, leading to an approach that covered most use cases robustly.

6.2 Modularity

While focus was put on the whole system, we still designed our software and hardware with modularity in mind, making it possible for sub-systems to be developed independently, and easily integrated into the system without requiring changes to higher-level systems. In the case of software, sub-systems largely conform to pre-defined ROS message types throughout development, an example being the perception system which saw major iterations before a final solution was found. Similarly, changes to the manipulator or Cartesian gantry system can be made and easily integrated into the robot's model, leaving higher-level logic unaffected.

6.3 Rapid Iteration

Iterative design is core to the development of our system. Tasks were broken into weekly sprints, and individual design teams were expected to deliver solutions that could be integrated and tested on the system, a process facilitated by our modular design practices. This process results in a higher overall integrated system up-time and allowed the team to focus on testing and evaluating the complete system, and to rapidly improve the design at a system or sub-system level as required.

By designing a fully custom solution, we overcame a major disadvantage reported by teams in previous challenges of being locked into the functionality of off-the-shelf components [4]. Our design comprises many commonly available parts such as the frame and rails for the Cartesian gantry system, simple machined parts, and a plethora of 3D printed parts. As such, many aspects of our design are able to be integrated, tested, and re-designed within hours or days.

7 System Performance

An exhaustive testing of the whole system was performed throughout the development phase as mentioned above. In particular, we were interested in simulating scenarios we would expect to see during the challenge in Japan. This system level testing led to a large focus on robustness and error recovery in developing high-level logic for *Cartman*. Herein we present the results during the Amazon Robotics Challenge finals, as well, as a long-term test of the robot during a robotics event on campus.

7.1 Amazon Robotics Challenge Finals

The finals task of the Amazon Robotics Challenge is chosen as a benchmark for comparison. While all teams received a different set of items, similar object classes were chosen by the Amazon team to be of equal difficulty.

Table 1 compares *Cartman*'s performance in the finals task to the other teams' systems, recorded by matching video footage of the competition with score-sheets. Due to the wide range of strategies used by different teams and the complex environment of the challenge, it is difficult to directly compare the performance of different systems as a whole in the context of the challenge except by points. However, to highlight the strengths and weaknesses of different systems, three metrics for the different aspects of system performance are presented.

Grasp Success Rate The success rate of the system in executing a grasp attempt, regardless of the item or the action performed with it. We counted success as lifting an object, which may be for picking, item classification, or relocating moves.

Table 1 Speed and accuracy of all systems during finals task and our long-term test (ACRV-LT)

Team	Grasp success rate	Avg. time	Error rate	Final score
Applied robotics	50% (3/6)	101 s	0% (0/2)	20
IFL PiRo	78% (18/23)	59 s	50% (7/14)	30
NAIST-Panasonic	49% (21/43)	35 s	33% (5/15)	90
MIT-Princeton	66% (43/65)	25 s	0% (0/15)	115
IITK-TCS	79% (19/24)	40 s	15% (3/20)	170
Nanyang	53% (23/43)	32 s	4% (1/25)	225
NimbRo picking	58% (33/57)	29 s	0% (0/22)	235
ACRV	63% (33/52)	30 s	4% (1/23)	272
ACRV-LT	72% (622/863)	30 s	N/A	N/A

Average Time per Attempt The time from finishing an action with one item to finishing an action with the next, averaged over all actions. It takes into account perception, motion planning, grasp execution, and movement.

Error Rate We define the error rate of the system as the ratio of number of penalties (incorrect item labels, incorrect items picked, dropped/damaged items, etc.) incurred to the total number of items stowed and picked during the round. It is indicative of the overall system accuracy.

While having a high grasp success rate, low execution time and low error rate are all desirable aspects of an autonomous robot, Table 1 shows that no one metric is a good indicator of success. Figure 9 shows the points accumulated by each team throughout the finals task of the competition, including any penalties incurred, and highlights some of the key differentiating aspects of the teams. Performance is most consistent between teams during the stow task, with the top five teams stowing roughly the same number of items at a similar, constant rate. The main separation in points is due to the pick task. For each team, the rate of acquiring points decreases throughout the pick task, as the difficulty of remaining items and the chances of items being occluded increase, causing many teams to abort their attempt early. It is here that we attribute our design approach and our system's overall robustness to our win. During the finals round, our system relied on item reclassification, misclassification detection, failed grasp detection, and ultimately our active perception approach to uncover the final item in the pick task, making us the only team to complete the pick task by picking all available order items.

7.2 Long-Term Testing

To test the overall performance of the system, *Cartman* was run for a full day, performing a continuous finals-style task, where the pick phase was used to replace all items from the storage system into the tote. 17 items were used, consisting

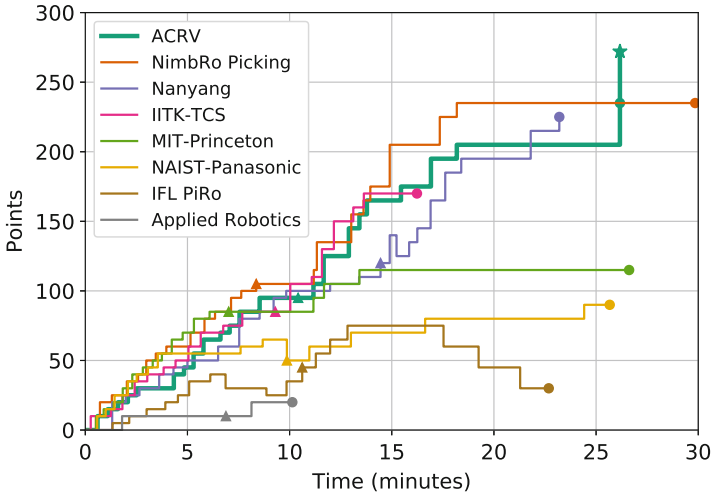

Fig. 9 Points accumulated by each team throughout their finals run, recorded by matching video footage of the competition with score-sheets. Triangles indicate the transition from stow to pick and circles indicate the end of a run. Stars indicate bonus points for completing the pick task with time remaining

of 13 items from the Amazon item set and 4 unseen items from the ACRV picking benchmark set [13]. The 4 unseen items were a soft, stuffed animal `plush_monkey`, a reflective, metallic pet's food bowl `pets_bowl`, a scrubbing brush `utility_brush`, and colourful squeaky pets toys `squeaky_balls`, which were chosen for their similarity to unseen items provided by Amazon during the competition. The 17 items were chosen to provide a range of difficulties as well as cover the spectrum of object classes that were available both physically (rigid, semi-rigid, deformable, and hinged) and visually (opaque, partially transparent, transparent, reflective, and IR-absorbing), ensuring that the full range of *Cartman*'s perception and grasping abilities were tested. Nine of the objects were acquired using suction and 8 by the gripper. Arguably the hardest item in the Amazon set, the mesh cup, was not included as *Cartman* was unable to grasp this item when it was on its side.

In 7.2 h of running time, *Cartman* completed 19 stow and 18 pick tasks, during which 863 grasping attempts were performed, 622 of which were successful (72% success rate, ACRV-LT in Table 1). Throughout the experiment, 10 items were incorrectly classified without automatic correction, requiring manual intervention to allow the system to complete a task and continue with the next. On one occasion the system had to be reset to correct a skipped drive belt.

The overall grasping success rates per item are shown in Fig. 10. Grasp attempt failures were classified as *failed grasp*, where the item was not successfully sucked or gripped, *dropped item*, where the object was successfully sucked or gripped but then dropped before reaching its destination, *weight mismatch*, where an item

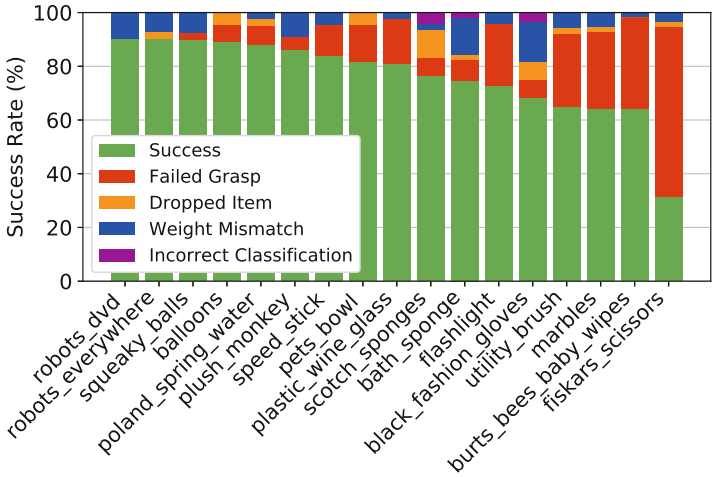

Fig. 10 Grasp success rates for the 17 items using during long-term testing

was grasped but its weight did not match that of the target object, or *incorrect reclassification*, where an object was successfully grasped but was incorrectly reclassified as a different item based on its weight.

The 168 *failed grasp* attempts can be further categorised by their primary cause, either *perception*, where the failure was caused by an incorrect or incomplete segmentation/identification of the object (27.6%), *physical occlusion*, where the object was physically occluded by another, resulting in a failed grasp attempt (10.3%), *unreachable* if the object was in a pose which was physically unobtainable by our gripper such as a small object resting in the corner of the storage system or tote (7.5%), or *grasp pose failure* if the object was correctly identified and physically obtainable and the grasp failed anyway (54.6%). Forty percentage of all failed grasps were on the challenging `fiskars_scissors` item, indicating that our manipulator or grasp point detection could be improved for this item.

7.3 Error Detection and Recovery

Like with any autonomous system designed to operate in the real world, things always can and eventually will go wrong. As such, the robot's sensors are monitored throughout the task to detect and automatically correct for failures, such as failed grasps, dropped items, and incorrectly classified or re-classified items. During the long-term testing described in Sect. 7, only 4% (10 out of 241) of failures were not corrected, requiring manual intervention for the robot to finish a task. These failures were caused by grasping an incorrect item that happened to have the same weight

as the wanted item, picking two items together and reclassifying by weight as a different item, or dropping the item outside the workspace.

Cartman's actions are heavily dependent on having an accurate internal state of the task, including the locations of all items. As an extra layer of redundancy, towards the end of a task, the system begins to visually double-check the location of items. If an item is consistently detected in the wrong location, the system attempts to correct its internal state based on visual classification and a record of any previous item classifications. This is possible due to the false negative rate of our classifier being lower in uncluttered scenes encountered at the end of a run.

8 Conclusions

Herein we presented *Cartman*, our winning entry for the Amazon Robotics Challenge. We attribute the success of our design to two main factors. In particular, we describe the key components of our robotic vision system, in particular:

- A 6-DoF Cartesian manipulator, featuring independent sucker, and gripper end-effectors.
- A semantic segmentation perception system capable of learning to identify new items with few example images and little training time.
- A multi-level grasp synthesis system capable of working under varying visual conditions.
- A design methodology focused on system-level integration and testing to help optimise competition performance.

Firstly, our method of integrated design, whereby the robotic system was tested and developed as a whole from the beginning. Secondly, redundancy in design. This meant that limitations of individual sub-systems could be overcome by design choices or improvements in other parts of the system.

Acknowledgements This research was supported by the Australian Research Council Centre of Excellence for Robotic Vision (ACRV) (project number CE140100016). The participation of Team ACRV at the ARC was supported by Amazon Robotics LLC.

References

1. Brown, E., Rodenberg, N., Amend, J., et al.: Universal robotic gripper based on the jamming of granular material. Proc. Natl. Acad. Sci. **107**(44), 18809–18814 (2010)
2. Calli, B., Walsman, A., Singh, A., Srinivasa, S., Abbeel, P., Dollar, A.M.: Benchmarking in manipulation research: using the Yale-CMU-Berkeley object and model set. IEEE Robot. Autom. Mag. **22**(3), 36–52 (2015). https://doi.org/10.1109/MRA.2015.2448951
3. Corke, P.I.: Robotics, Vision and Control: Fundamental Algorithms in MATLAB. Springer, Berlin (2011)

4. Correll, N., Bekris, K.E., Berenson, D., et al.: Analysis and observations from the first Amazon picking challenge. IEEE Trans. Autom. Sci. Eng. **15**, 172–188 (2018)
5. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: ImageNet: a large-scale hierarchical image database. In: Computer Vision and Pattern Recognition (CVPR) (2009)
6. Hauser, K.: Continuous pseudoinversion of a multivariate function: application to global redundancy resolution. In: Workshop on the Algorithmic Foundations of Robotics (2017)
7. Hernandez, C., Bharatheesha, M., Ko, W., et al.: Team Delft's robot winner of the Amazon Picking Challenge 2016. In: Robot World Cup, pp. 613–624. Springer, Berlin (2016)
8. Husain, F., Schulz, H., Dellen, B., Torras, C., Behnke, S.: Combining semantic and geometric features for object class segmentation of indoor scenes. IEEE RA-L **2**(1), 49–55 (2016). https://doi.org/10.1109/LRA.2016.2532927
9. Innocenti, C., Parenti-Castelli, V.: Singularity-free evolution from one configuration to another in serial and fully-parallel manipulators. J. Mech. Des. **120**(1), 73–79 (1998). https://doi.org/10.1115/1.2826679
10. Jung, R.L.W.: Robotis OpenCR. https://github.com/ROBOTIS-GIT/OpenCR/tree/master/arduino/opencr_arduino/opencr/libraries/DynamixelSDK
11. Lehnert, C., English, A., Mccool, C., et al.: Autonomous sweet pepper harvesting for protected cropping systems. IEEE Robot. Autom. Lett. **2**(2), 872–879 (2017)
12. Leitner, J., Dansereau, D.G., Shirazi, S., Corke, P.: The need for more dynamic and active datasets. In: CVPR Workshop on the Future of Datasets in Computer Vision. IEEE, Piscataway (2015)
13. Leitner, J., Tow, A.W., Sünderhauf, N., et al.: The ACRV picking benchmark: a robotic shelf picking benchmark to foster reproducible research. In: IEEE International Conference on Robotics and Automation (ICRA) (2017)
14. Lin, G., Milan, A., Shen, C., Reid, I.: Refinenet: multi-path refinement networks for high-resolution semantic segmentation. In: IEEE Conference on Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
15. Matthias, B., Kock, S., Jerregard, H., Källman, M., Lundberg, I.: Safety of collaborative industrial robots: certification possibilities for a collaborative assembly robot concept. In: 2011 IEEE International Symposium on Assembly and Manufacturing (ISAM) (2011)
16. McCauley, M.: AccelStepper: AccelStepper library for Arduino (2010). http://www.airspayce.com/mikem/arduino/AccelStepper/
17. McTaggart, M., Morrison, D., Tow, A.W., et al.: Mechanical design of a Cartesian manipulator for warehouse pick and place. Tech. Rep. ACRV-TR-2017-02, arXiv:1710.00967, Australian Centre for Robotic Vision (2017)
18. Milan, A., Pham, T., Vijay, K., et al.: Semantic segmentation from limited training data. In: Proceedings IEEE International Conference on Robotics and Automation (ICRA) (2018)
19. Monkman, G.J., Hesse, S., Steinmann, R., Schunk, H.: Robot Grippers. Wiley, Hoboken (2007)
20. Morrison, D., Tow, A.W., et al.: Cartman: the low-cost Cartesian Manipulator that won the Amazon Robotics Challenge. Tech. Rep. ACRV-TR-2017-01, Australian Centre for Robotic Vision (2017)
21. Nakamura, Y., Hanafusa, H.: Inverse kinematic solutions with singularity robustness for robot manipulator control. J. Dyn. Syst. Meas. Control. **108**, 163–171 (1986). https://asmedigitalcollection.asme.org/dynamicsystems/article-abstract/108/3/163/425826/Inverse-Kinematic-Solutions-With-Singularity?redirectedFrom=fulltext
22. Pinheiro, P.O., Collobert, R.: From image-level to pixel-level labeling with convolutional networks. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2015)
23. Quigley, M., Conley, K., Gerkey, B., et al.: ROS: an open-source robot operating system. In: Workshop on Open Source Software of the IEEE International Conference on Robotics and Automation (ICRA) (2009)
24. Quigley, M., Gerkey, B., Conley, K., Faust, J., Foote, T., Leibs, J., Berger, E., Wheeler, R., Ng, A.: ROS: an open-source robot operating system. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Workshop on Open Source Robotics, Kobe, Japan (2009)

25. Schwarz, M., Lenz, C., Garcia, G.M., et al.: Fast object learning and dual-arm coordination for cluttered stowing, picking, and packing. In: IEEE International Conference on Robotics and Automation (ICRA) (2018)
26. Schwarz, M., Milan, A., Lenz, C., Munoz, A., Periyasamy, A.S., Schreiber, M., Schüller, S., Behnke, S.: NimbRo picking: versatile part handling for warehouse automation. In: IEEE International Conference on Robotics and Automation (ICRA) (2017)
27. Sucan, I.A., Chitta, S.: MoveIt! http://moveit.ros.org/
28. Ulbrich, S., Kappler, D., Asfour, T., et al.: The OpenGRASP benchmarking suite: an environment for the comparative analysis of grasping and dexterous manipulation. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2011)
29. Wade-McCue, S., Kelly-Boxall, N., McTaggart, M., et al.: Design of a multi-modal end-effector and grasping system: how integrated design helped win the Amazon robotics challenge. Tech. Rep. ACRV-TR-2017-03, arXiv:1710.01439, Australian Centre for Robotic Vision (2017)