

# Autonomous Dishwasher Loading from Cluttered Trays using Pre-trained Deep Neural Networks

---

**Isobel Voysey**  
Bio-Inspired Robotics Laboratory  
Department of Engineering  
University of Cambridge, UK  
iv256@cam.ac.uk

**Thomas George Thuruthel**  
Bio-Inspired Robotics Laboratory  
Department of Engineering  
University of Cambridge, UK  
tg444@cam.ac.uk

**Fumiya Iida**  
Bio-Inspired Robotics Laboratory  
Department of Engineering  
University of Cambridge, UK  
fi224@cam.ac.uk

## Abstract

Autonomous dishwasher loading is a benchmark problem in robotics that highlights the challenges of robotic perception, planning and manipulation in an unstructured environment. Current approaches resort to a specialized solution, however, these technologies are not viable in a domestic setting. Learning-based solutions seem promising for a general purpose solutions, however, they require large amounts of catered data, to be applied in real-world scenarios. This paper presents a novel learning-based solution without a training phase using pre-trained object detection networks. By developing a perception, planning and manipulation framework around an off-the-shelf object detection network, we are able to develop robust pick-and-place solutions that are easy to develop and general purpose requiring only a RGB feedback and a pinch gripper. Analysis of a real-world canteen tray data is first performed and used for developing our in-lab experimental setup. Our results obtained from real-world scenarios indicate that such approaches are highly desirable for plug-and-play domestic applications with limited calibration. All the associated data and code of this work is shared in a public repository.

## 1 Introduction

The process of autonomously loading a dishwasher is a benchmark problem for domestic robots (Dillmann, 2004; Vischer, 1992). This task is of particular interest as it introduces the challenges of an unstructured environment in perception and manipulation, while constraining the complexity of the problem with a well-defined task and a limited set of objects. The task involves challenges in design, perception, planning and control. We restrict our study to the perception, planning and control problem by using a general purpose 6 Degree of Freedom (DoF) industrial manipulator with a simple pinch gripper (See Figure 1).

The first step in developing an autonomous dishwasher loading robot is to detect and estimate the grasp point of objects in a tray. The level of accuracy required in the detection process depends on the object manipulation system. With specialized cleaning devices, the output required from the perception process can

be minimal (CambridgeConsultants, 2019) to being absent (Dishcraft, 2019). For general purpose service robots, the perception problem is challenging because of the variabilities among a class of object, visual occlusion and sensory limitations. Next, we briefly enumerate the works on learning-based approaches for perception, planning and control to examine the state-of-the art and their limitations.

The objective of the perception system is to estimate grasp points required for the given grasping mechanism. Traditionally, this involved the process of segmenting and obtaining a full 3D model of the object and calculating the grasping locations based on force balancing (Bicchi and Kumar, 2000; Miller et al., 2003; Weisz and Allen, 2012). Although, such methods can provide optimal grasping strategies, they are practically difficult to implement due to their heavy sensory and computational requirements. Learning-based approaches are promising due to their ability to cut out mid-level computational processes. Saxena et. al. showed that it is possible to estimate 3D grasp points using multiple 2D visuals (Saxena et al., 2008a). Their work was based on the insight that man-made objects have certain visual features that correspond to their appropriate grasping location. Given labelled images with the 'correct' grasp points, the mapping from the image to grasp location can be learned in a supervised manner. However, such an approach requires a large amount of manually labelled data, that is subject to human interpretation of the best grasp location. Moreover, they are still susceptible to occlusions and visual noise. Successive works showed that incorporating depth information significantly improved the performance of the system. as expected (Saxena et al., 2008b). Other related works include automatic approach vector generation for grasping using learning by demonstration (Ekvall and Kragic, 2007).

Deep learning based approaches have demonstrated their superiority in visual processing tasks (Hinton and Salakhutdinov, 2006). With the relative ease of generating samples for learning and the ability to share data set among different problems and domains make them highly desirable for our problem. Deep learning based approaches have shown superior performance in object detection and classification problems with high generalizability (Zhao et al., 2019; Girshick et al., 2014). Lenz et. al. proposed a deep learning framework for generating object grasp locations from RGB-D images for the first time (Lenz et al., 2015). Their work showed that deep learning based methods are highly desirable for grasp detection with excellent generalizability. Additionally, their deep network directly provided the grasp location and orientation without additional processing or hand-tuning and worked independently of the gripping hardware. However, the process still required numerous manually labelled data set for training their network, which could also incur human biases on the best grasp location.

Once the target objects are identified and localized, the next process is to develop the planning strategy, which involves the sequence of objects to be picked and the trajectory to follow while doing so. The general version of the problem is NP-hard and is a vast research topic (Wilfong, 1991). However, the process can be simplified without strict constraints on a collision-free plan. Planning strategies can vary from push-grasping (Lynch and Mason, 1996; Dogar and Srinivasa, ) to multi-arm motion planning (Koga and Latombe, 1994; Smith et al., 2012). It is however clear that physics based motion planning strategies still remain a challenging robotic task (Yu et al., 2016). Deep reinforcement learning approaches have therefore looked promising for this problem (Finn and Levine, 2017; Levine et al., 2016). However, such methods require a large number of real-world trials or accurate simulation models (Andrychowicz et al., 2020) to be developed. This work focuses mainly on the task of grasp identification, and thus we will consider only objects that can be picked up without performing complex manipulation. We use a simple pinch-gripper for grasping and the motion planning does not penalize object collisions explicitly. Collisions can lead to losing grip and perception errors, which will be reflected in the clearance performance. The next paragraph presents the past research works focusing specifically on the dishwasher loading problem.

Even though dishwasher loading is a common domestic application ripe for automation, there have been very few works that have attempted to solve it entirely. Earliest works have tried to use supervised learning techniques along with a rigid body simulation to decide the location to pick and place. However, the demonstrations were limited to simulations and requires the availability of rigid body models for each object type. Other works focus on handling occlusion and noisy sensors while performing the task of dishwasher loading (Pajarinen and Kyrki, 2017). They model multi-object manipulation of crowded occluded objects

as a partially observable Markov decision process in order to improve planning process. This study was, however, limited to a single object and ignored the complexities of object detection. Object placement after picking is another complex problem that is involved in the dishwasher loading problem (Jiang et al., 2012b; Jiang et al., 2012a). The dishwasher loading problem has also been used to develop grasp point identification, as discussed in the previous section (Saxena et al., 2008a).

Although learning-based approaches seem the most promising for solving our specific problem, all of these techniques require a data acquisition and learning phase which is heavily time consuming. Moreover, these methods are either limited by the need for a simulation environment, subjective data labelling, high sensory requirements, etc. This limits the generalizability of these approaches and their large scale deployment in domestic environments. The aim of this study is to investigate a general purpose learning-based solution for the dishwasher loading problem without a time consuming learning/calibration phase and the need for complex physics models and rich sensory data. The core idea of this work is to use commonly available pre-trained neural networks for objection detection and develop algorithms to extract relevant information from the output of these networks. The immense progress in image detection deep networks can be attributed to the developments in computational devices and the collective scientific effort to organize large labelled databases. The Imagenet and COCO data sets are some of the most commonly used ones (Deng et al., 2009; Lin et al., 2014). For this work we will be using the COCO data set. These databases have led to the development of several deep-learning architectures for image detection and segmentation (Ren et al., 2015; He et al., 2016; Girshick, 2015; Russakovsky et al., 2015; Krizhevsky et al., 2012; He et al., 2017). Image detection involves classification and localization of objects using bounding boxes and image segmentation involves classification and localization with edge detection. As objection detection has no labelling ambiguity compared to grasp localization, they are desirable to avoid human biases. In this work, we use pre-trained deep networks trained on the COCO data set for image detection, specifically a unified deep learning architecture for image classification and detection called YOLO (You Only Look Once) (Redmon et al., 2016a). There are few reasons for using this network trained on the COCO data set. First, the COCO data set contains thousands of labelled images of objects typically found in a dishwasher loading problem (with some exceptions such as plates shown later). Hence, deep networks trained on this database will be robust to visual noise and be highly generalizable. Second, the YOLO architecture is very compact for the accuracy it provides. Hence, training the network on additional images or testing the pre-trained network is fast. Image segmentation networks are not considered as the available databases are currently limited to very few objects. Note that pre-trained models like YOLO does not provide object pose information and hence has to be estimated by further processing techniques as demonstrated in this work.

## 1.1 Contributions

This work presents an integrated object detection, grasping and manipulation framework build around pre-trained deep neural networks for the dishwasher loading problem. We use a commonly available object detection network called YOLO, trained on the COCO dataset. As our learning-based approach is derived from pre-trained models and existing databases, there is no training required for setting up, while we maintain the generalizability of deep neural networks and their robustness to visual noise. The viability of using YOLO for object detection and grasping has already been proposed in an earlier work (Tian et al., 2019). We extend this to a thorough framework that extracts the relevant information for object localization, pose estimation and motion planning. The grasp point estimation and planning algorithm is independent of the object class geometry, thereby requiring no parameter tuning for grasping and placing new unseen objects. We incorporate a simple force feedback to the system to remove the requirements for depth information and hence, requiring only a RGB camera. We run benchmark tests on our visual perception system from real-world canteen data and develop a test protocol based on this data. The complete control framework is then tested on an in-lab kitchen setup. To the best of our knowledge, this is the first research work that has extensively tested a complete framework for dishwasher loading on real-world scenarios. Additionally, we demonstrate the importance of interactive perception in manipulation tasks through a simple feedback strategy that can drastically improve the performance of our system (Bohg et al., 2017; Katz and Brock, 2008). The results and data set obtained through our study can also serve as a tool for future researchers

and for developing novel deep networks.

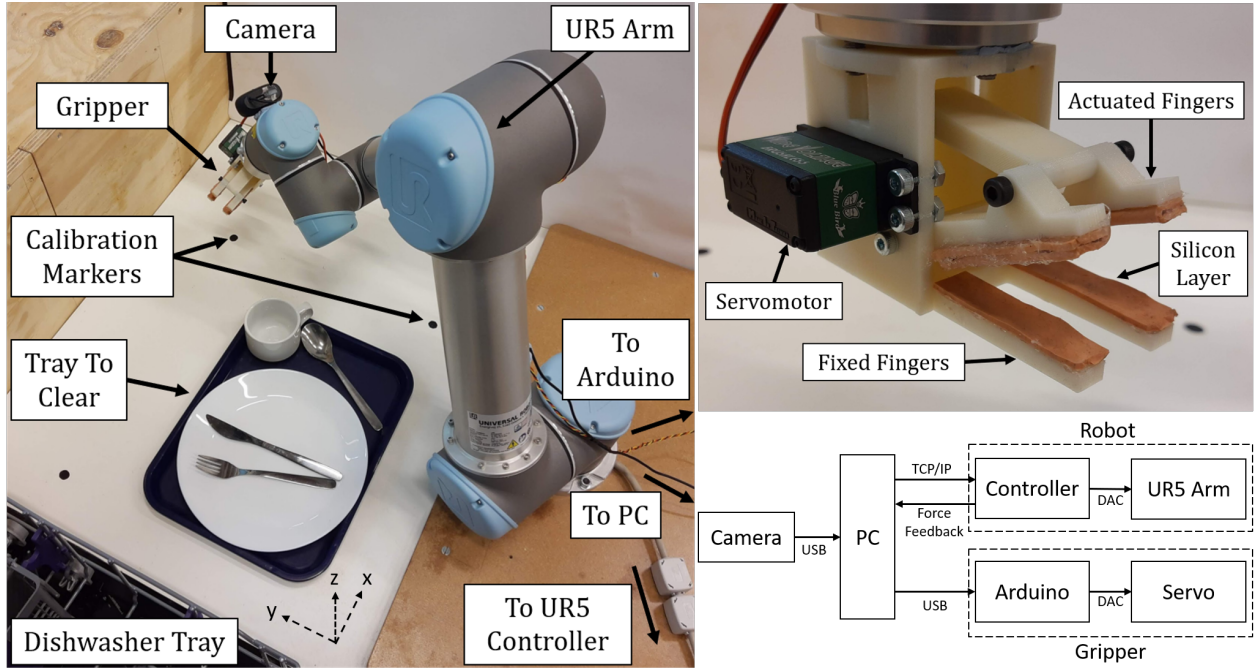


Figure 1: Experimental setup and flow of communication.

## 2 Experimental Method

This section presents the experimental setup and the algorithms developed for the dishwasher loading problem. First, we introduce the experimental setup to provide an insight about the problem and the hardware constraints. Further sections describe in detail the algorithms developed for the dishwasher loading problem built around our experimental setup (See figure 2).

### 2.1 Experimental Setup

The setup we used for performing our tests is shown in Figure 1. It comprises of a pinch-gripper mounted on a 6 degree of freedom (DoF) UR5 arm, which is mounted on a movable platform. The gripper is actuated by a single servo motor. A silicon padding is added to the gripper to improve its grasping ability. An RGB camera is mounted on last link of the UR5 robot arm to provide information about the scene. The camera is mounted directly on the arm for two reasons. First, this allows us to easily transfer the whole system with minimal calibration. Second, it provides an extra control authority over the scene detection. We use a 720p logitech c270 camera for obtaining the RGB images. The pinch-gripper is controlled by commands from the PC through an Arduino. The UR5 arm is controlled from the PC through the UR5 controller through a TCP/IP connection. Commercially available kitchen utensils are used as our test objects. The tray with the items and the dishwasher tray are place on a standard kitchen top. Three calibration markers are added to the table for obtaining the camera-to-robot coordinate transformation. All the computation is done on an Intel(R) Core(TM) i7-8750H CPU @2.20 Ghz with 16 Gb RAM. A GPU is not used for running the deep neural network. All the codes are developed in the python programming language and are available on the open repository [bitbucket.org/cambirllab/isobel\\_4th\\_yr/](https://bitbucket.org/cambirllab/isobel_4th_yr/).



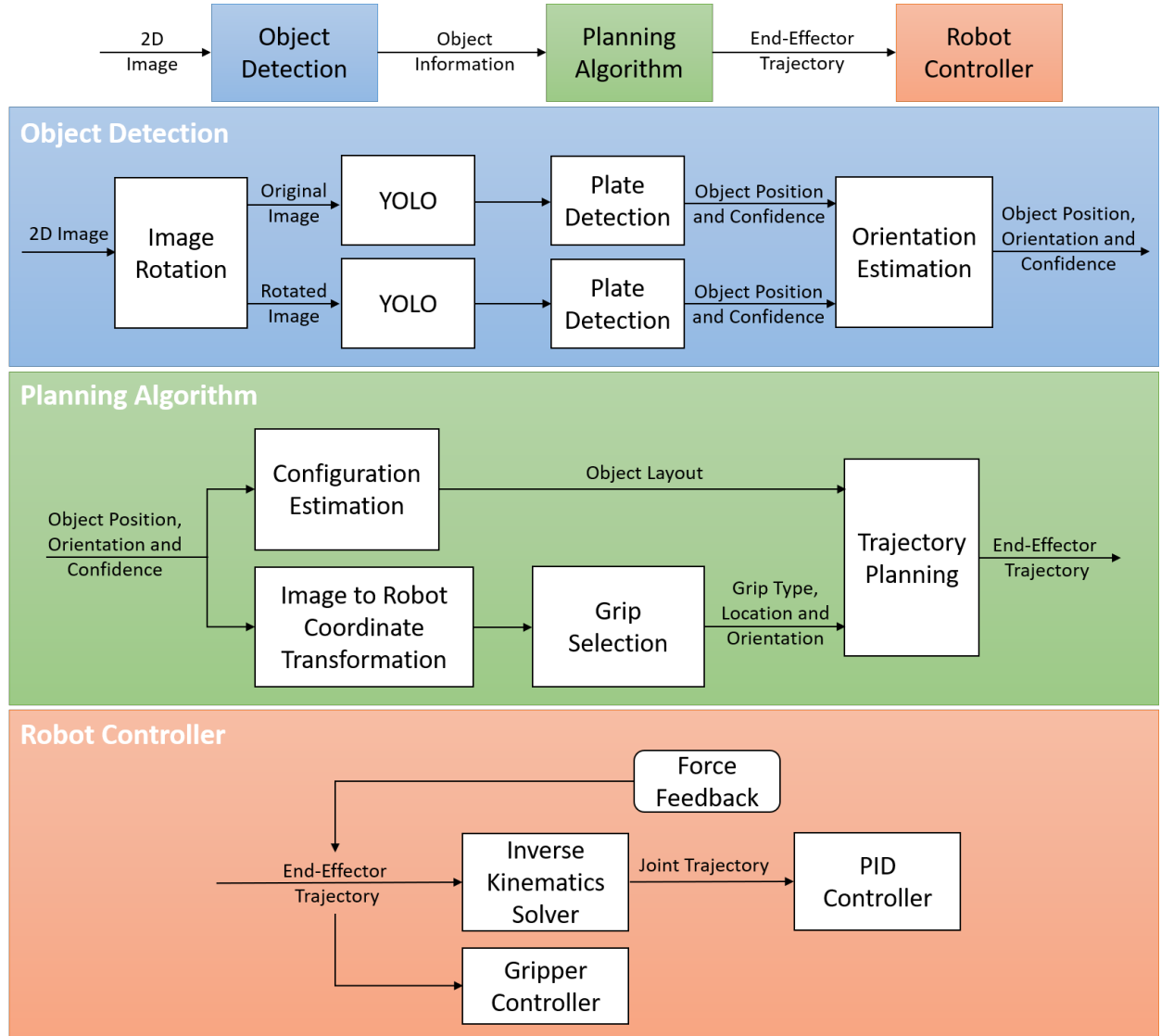


Figure 2: Overview of the complete process.

## 2.2 Object Detection

This section presents the object detection module upon which our planning algorithm and robot controller is built on (Figure 2). The object detection process takes in a single image from the RGB camera and outputs all the detected object, their relative pose, size and confidence in their estimation. This information is then used for deciding the picking strategy. The object detection process starts with the capture of single image from a known location in the robot workspace, creating a rotated duplicate and feeding it to the YOLO network for object detection. This process is performed to infer the orientation of each object as explained in section 2.2.2. The original image is rotated in a way to keep dimensions of image the same, i.e. the corners of the image are cut off rather than resizing image to keep corners in frame. The subsequent subsection provides a brief overview of the YOLO network.

### 2.2.1 YOLO

You Only Look Once (YOLO) is an object detection algorithm where object detection is framed as a regression problem to spatially separated bounding boxes and associated class probabilities. This makes it possible to use a single neural network to predict bounding boxes and class probabilities directly from the full images in one evaluation making the process fast and easy to train (Redmon et al., 2016b). Other algorithms have to go through the image twice or separate localisation and classification into two stages. Only going through the image once means that YOLO is very fast, with the latest version (YOLOv3) achieving speeds of 34fps (Redmon and Farhadi, 2018). YOLOv3 downsamples the input image to a dimension of 416x416x3 and has 106 layers. Hence, for more accurate detection, it is necessary to perform YOLO detection on segments of the image separately.

YOLO still lags behind state-of-the-art detection systems in accuracy. While it can quickly identify objects in images it incurs error in localizing some objects, especially small ones. YOLO's limitations stem from the division of image required for processing in a single run. The image is divided into  $7 \times 7$  grid cells and non-maximal suppression is carried out for objects detected within these regions. This means a maximum of 49 objects can be detected in a single image. However, this is unlikely to be a problem for our application. For our study we use a pretrained YOLOv3 network trained on the COCO data set. Although the COCO data set has only 80 objects classes, they include almost all of the objects found in a typical canteen tray. This makes them particularly suited for our application. As plates are not labelled in the COCO data set, we use a traditional circle detection algorithm called Hough circle detection for identifying and localizing them. With progress in the manual labelling of various image data sets, we can expect this drawback to be addressed in the future. Note that for adding new object labels to a image data set and to retrain an object detection deep network, it is vital to manually label all the images with the new object for ideal training.

The regression problem tries to predict the two dimensional centre point, as well as a width and height for each viable object candidate. Typically, this is represented as a bounding box around the detected object. There are a few metrics that are typically used to evaluate the performance of such detection algorithms. They include intersection-over-union (IOU), precision, and recall (see figure 3). IOU is a measure of how closely the predicted bounding box overlaps with the ground truth bounding box. When it exceeds a chosen threshold, the prediction made is considered to be a true one. Precision is the ratio of true predictions to total predictions and recall is the ratio of true predictions to ground truth occurrences. They are given by:

$$\text{IOU} = \frac{\text{area of intersection}}{\text{area of union}}$$
$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad \text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

where TP stands for true positives, FP for false positives, and FN for false negatives. For all these metrics, a value of 1 corresponds to ideal detection. Note that the ground truth labelling is still subject to subjective errors.

### 2.2.2 Orientation Estimation

As YOLO returns a bounding box parallel to the image axes, there remains ambiguity about how the object lies within the bounding box. This is particularly relevant for long thin objects, such as pieces of cutlery, which will usually lie on one of the diagonals of the rectangle. Without highly adaptive gripper, it is necessary to resolve this ambiguity. For example, in order to pick up items of cutlery with the pinch gripper used here, it is necessary to know which diagonal to align the mouth of the gripper with.

The solution proposed here uses a rotated version of the original image, as mentioned in the beginning of the section. It runs the same object detection algorithm on it, and compares how the dimensions of the bounding box for an object differ from the initial image and the rotated image. For instance, the two possible arrangements of a piece of cutlery in a given bounding box are shown in Fig. 4. Below these cases

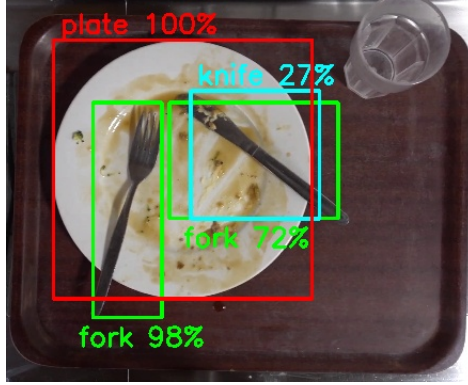


Figure 3: A visualisation of the YOLO and plate detection results on an image from the canteen data set. Object classes, confidence levels and bounding boxes are shown. The failure to detect the cup is an example of a false negative for cups. The identification of the knife as a fork is an example of a false positive result for forks.

is the expected bounding box on an image rotated anticlockwise. The dimensions change in different ways depending on the initial arrangement. In Fig. 4 this is illustrated using the detection results from a example tray captured at a canteen (see Section 3). For the cutlery lying on the diagonal from top right to bottom left, the bounding box is narrower and taller in the image rotated anticlockwise. For the cutlery lying on the diagonal from top left to bottom right, the bounding box is wider and shorter in the image rotated anticlockwise. This means it is possible to infer which diagonal the object lies on, marked on the final image. In some cases the bounding boxes may not change significantly or the changes are inconclusive or the object is not detected in both images. In these cases, the diagonal was chosen randomly with equal probability to reduce the computational time. Such cases can be solved by adjusting the viewing point of the camera or by relying on a closed-loop strategy as proposed in this paper.

Since each image likely contains multiple objects and therefore multiple bounding boxes, we must compare bounding boxes corresponding to the same object. This is not always possible purely by considering object classes as there may be multiple instances of the same object in an image, or an object might be correctly localised but misidentified as a similar object (e.g. a fork misidentified as a spoon in the rotated image). The same rotation transformation that was applied to the initial image is applied to the centroid of bounding boxes found in the initial image. This then gives the predicted location of the centroid in the rotated image. This relies on both bounding boxes being very close to the the ground truth, within a certain radius of the predicted centroid. This is reasonable as the image rotation is done in software rather than by rotating the camera physically, thereby preserving the image characteristics.

## 2.3 Planning Algorithm

Once the objects are detected, localized and the their orientation determined, the next step is to predict the best picking strategy (Figure 2). This involves deciding the order in which to pick the objects and determining the motion plant to do so. The subsequent section presents the algorithm for determining the spatial configuration of each object with respect to each other.

### 2.3.1 Configuration Estimation

Before planning specific trajectories, a plan must be made about the order in which to pick the objects. There may exist dependencies between objects, wherein object A must be picked before object B. For example, in the case of object A being stacked on top of object B or partially covers object B. As the deep neural network is trained on a large set of real-world data, even occluded objects can be correctly identified and

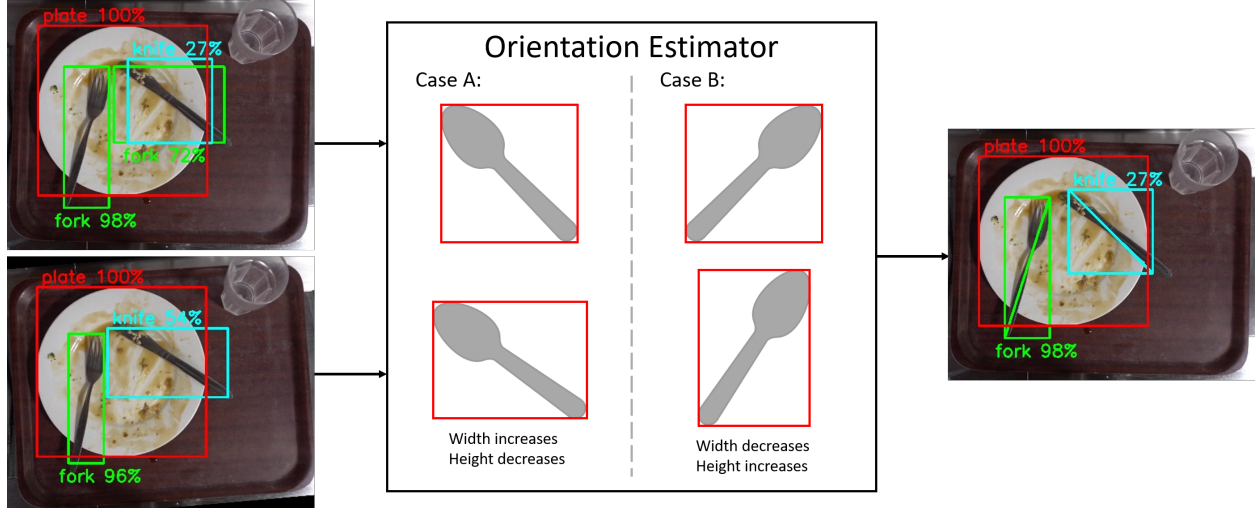


Figure 4: Example of inputs to and outputs from the orientation estimator represented on the images of the tray. The detection results for the original and rotated image are input. Diagonals predicted from comparing bounding boxes are marked on the output.

localized by the YOLO network.

The network does not explicitly provide any information about the relative configuration of the objects. Adding a depth sensor or using multiple images from different angles and triangulation could be one way to solve this problem. However, this would incur additional computational and calibration overhead along with the requirement of additional hardware. Here we propose a method of inference based on the relationship between bounding boxes of detected objects and their estimation confidences. The algorithm is based on the insight that partially occluded objects are less likely to be predicted with high confidence and that the intersection of the object bounding boxes provide some information about their relative configuration. The inference is based on what fraction of the bounding box of object A overlaps with the bounding box of object B. That is, if  $\frac{\text{Area of } A \cap B}{\text{Area of A}} > \text{threshold}$ , then A is on top of B, otherwise it would not have been detected (see figure 5). If object A was underneath object B, a large proportion of it would be occluded and object detection would be unlikely to identify the object or it would be detected with lower confidence. The algorithm in the pseudocode form is shown in Algorithm 1.

### 2.3.2 Image to Robot Coordinate Transformation

Once the relative configuration of the objects and their corresponding poses are obtained in the image coordinates, an image-to-robot coordinate transformation is performed to plan the robot and gripper motion. The transformation is done from  $(x', y', w', h')$  in the image coordinate to  $(x, y, w, h)$  in the robot coordinate space, where the variables represent the planar coordinates of the objects and their width and height. We use a simple planar perspective projection based on the a priori knowledge of three fixed points in image coordinates and robot coordinates.

The three points were marked by small black circles in the upper left, upper right, and lower right of the camera's field of view. Their coordinates in the image space were found using blob detection. Their coordinates in robot space were found using kinesthetic demonstration by moving the tool centre point of the robot to directly above each point and recording the XY coordinates of the end-effector. The Z-coordinate of the surface was found using downward movement until contact was detected from the force feedback. This also allows us to calculate the vertical distance between the camera and the worktop.

Object Type	Grip Type	Grip Location	Grip Orientation
Fork	Vertical	$(x, y)$	$\pm \arctan \frac{w}{h}$
Spoon	Vertical	$(x, y)$	$\pm \arctan \frac{w}{h}$
Knife	Vertical	$(x, y)$	$\pm \arctan \frac{w}{h}$
Cup	Vertical	$(x + \frac{w}{2}, y)$	$0^\circ$
Plate	Horizontal	$(x \pm \frac{w}{2}, y)$	$\mp 90^\circ$

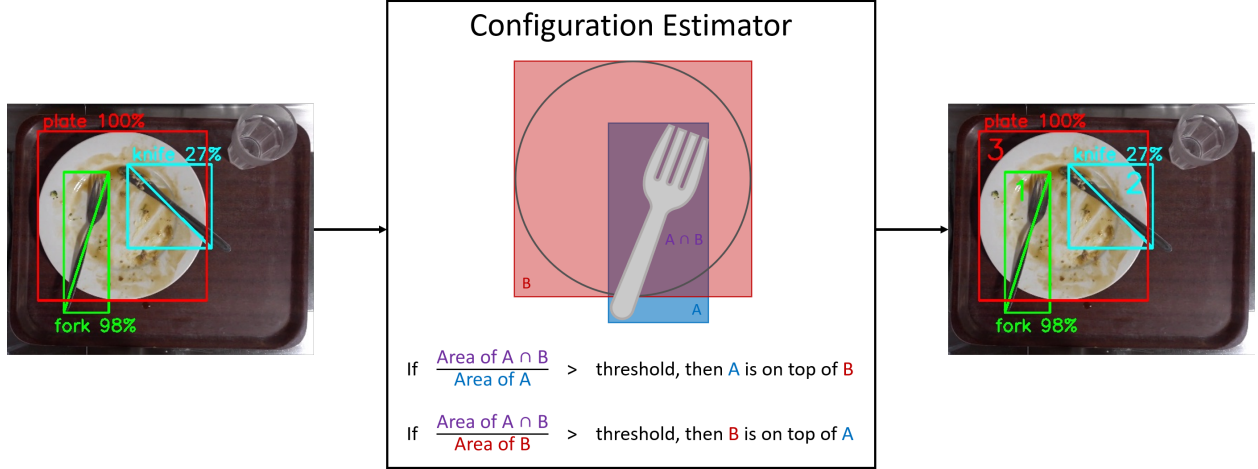


Figure 5: A representation of how the areas of bounding boxes and their intersections with other bounding boxes can be used to make inferences about configuration. The planned order for picking is marked on the output image. Full details of the algorithm are shown in Algorithm 1.

#### 2.3.4 Trajectory Planning

The trajectory planning process takes into account the configuration of objects in the scene to create an overall order to remove objects and the information from grip selection to plan specific trajectories for each object. Next we present the trajectory planner for each object class after the grip selection.

For plates the gripping location was set at the edge of the plate. The width of the bounding box returned from the object detection gave the diameter of the plate. The gripper was moved to a location so the tips of the fingers were offset from the rim of the plate. The gripper moved vertically downwards until the force sensor registered a threshold force value. As with all usages of the force sensor, the gripper was then raised by 2 mm so that surface contact with the tray was eliminated. The gripper then moved 4 cm along the plane of the worktop towards the centre of the plate (corresponding to the centroid of the bounding box). The plate was either gripped from the right or left, depending on the location relative to the centre of the image. If the centroid fell in the right half of the image the plate was picked from the left side and vice versa. This was done to avoid collision with the canteen tray. Note that other objects would have most likely been removed by the time the planner executed the removal of the plate and hence their collisions are not considered.

The diameter of the cup was taken as the smaller of the width and height of the bounding box. For cups it is necessary to also take into account errors in the image-to-robot transformation due to its elevate height. This parallax error become more noticeable if the cup lies on the outskirts of the tray, as is most often the case. To compensate for this, a manual scaling is performed to account for distance between the camera and the object imaged. This parameter is hand-tuned and could be parameterized by the cup height. However, as the cups have a higher error tolerance on the gripping success, this was not required. The cup was picked using a vertical pinch grip at the right-hand side of the rim. Like in the case of the plate, the robot arm stops when the reaction force values exceeded a certain threshold.

In order to pick cutlery, the vertical grip type was used. The angle of the piece of cutlery from the y-axis was calculated using  $\theta = \arctan \frac{w}{h}$ , where  $w$  is the width of the bounding box and  $h$  is the height of the bounding box. With the gripper jaws aligned with this angle, the picking location was taken as the centroid of the bounding box. The gripper began above this location and moved downwards until the force sensor determined contact with the surface the cutlery was assumed to rest on. Once again, after the force sensor had determined contact with the surface, the gripper was raised slightly, so that during closing of

the gripper, the surface underneath the piece of cutlery would not move due to friction between the surface and the gripper. Note that by using a unified strategy for all cutlery items, our algorithm is more robust to wrongly identified cutlery items, which is a common issue with the YOLO architecture. For instance, even if a knife was wrongly identified as a spoon or fork (which is the more likely scenario), as the grip strategy is the same for all, we can still perform a successful pick and place. Hence, by appropriately designing our planning algorithm, we can achieve better robustness to poor recall performances.

## 2.4 Robot Controller

Once the end-effector trajectory and the corresponding gripper states are calculated, these commands are sent to the UR5 controller and the arduino from the PC. The UR5 controller takes in the end-effector pose and calculates the joint angles using its in built inverse kinematics model. We use the *servoj* motion module in the UR5 controller to obtain smooth motions in the joint space. Once the motions commands are completed, a flag is sent back to the PC for obtaining the next points in the trajectory. The force feedback, which is obtained from the current intake of all the servo motors, are also sent back to the PC. Based on the force feedback and the current configuration of the arm, the next trajectory points and gripper states are modified.

# 3 Object Detection Analysis

Before we tested our control architecture on real-world scenarios, we performed a comprehensive analysis on the visual perception system. This is was done for two reasons. First, we wanted a baseline evaluation of the object identification module, as the performance of the subsequent processes depended directly on it. Second, we wanted to ground and subdivide our experimental protocol based on real-world scenarios. The next two sections describe the tests we performed and analysed to do so.

## 3.1 Experiment Protocol

To obtain real-world samples for evaluating the object detection module, we setup an image capturing setup at a typical college canteen. Images of canteen trays from the St Catharine's College at the University of Cambridge were obtained over the course of a single lunchtime. The canteen was equipped with a conveyor belt where users place their trays of dirty cutlery and crockery at the end of a meal. A camera was mounted directly above the conveyor belt and captured 1 hour 1 minutes of 1920 by 1080 video at 50 fps, from which images of tray were extracted manually, resulting in a total of 69 images of different trays.

The ground truth rectangular object bounding boxes were annotated on all images using the labelImg software (available at [github.com/tzutalin/labelImg](https://github.com/tzutalin/labelImg)). This information was saved in text files using the YOLO format. The list of classes was restricted to fork, spoon, knife, cup, bowl, plate, and cutlery. The introduction of the cutlery class was necessary due to several instances of occlusions which made it impossible to definitively determine which class of fork, spoon, or knife it belonged to. This was the case for 17 out of the 154 pieces of cutlery in the data set. Besides objects belonging to these aforementioned classes, no other objects were labelled. For occluded objects, labelling was done only on the visible section.

Every image in the data set was run through YOLO and Hough circle detection as outlined above. The YOLO network used weights supplied by the creators which had been pretrained on the COCO data set. The results of the object detection procedure were compared with the ground truth labels to assess the performance of the object detection procedures. This collection of images along with the files of corresponding ground truth labels is published along with the code in the open repository [bitbucket.org/cambirllab/isobel\\_4th\\_yr/](https://bitbucket.org/cambirllab/isobel_4th_yr/).

We also conducted an analysis of the orientation estimation performance on the canteen data set. The original 2D image and a rotated version of the same image were both run through the object detection and

the results compared to find the orientation of items of cutlery in their bounding boxes as outlined above and illustrated in Fig. 4. The orientations found through this process were displayed on the images and checked manually for correctness.

### 3.2 Detection Results

The results of the object detection module are analyzed keeping the final application in mind. The average performance metrics of the canteen dataset are shown in Fig. 6, separated into object classes based on the metrics described in section 2.2.1. Due to their large size and hence low probability to being occluded, 'plate's had a high identification and localization performance. Few times they were wrongly classified as the object class 'bowl' as shown in the confusion matrix in Figure 7. 'Cups' had the worst localization accuracy due to their poor recall performance. This means that in many cases cups were never attributed a class. Note that the precision for cups is high, which indicates that when they are identified, they were almost always correctly labelled. We observed that the poor identification result of cups occurred when the point of view was directly above the cup. Hence, a potential improvement could be to view the scene from at least two angles.

A separate class for cutlery, which includes forks, spoons and knives, is also included in Figure 6. As the gripping strategy is independent for all the items in the cutlery class, the recall rate of this class is more relevant for us. The recall rate gives a upper bound on the grasping success rate. A poor recall rate will invariable lead to a poor clearing performance. A high recall rate, on the other hand, does not ensure perfect grasps. The average IOU in combination with the grasping strategy provides an estimate of the grasp success. For instance, a poor identification of the plate geometry (low IOU), can cause the gripper to close early, leading to a weak grip on the plate and hence increase the probability of failure. Cups have higher tolerance on the IOU performance, as the vertical motion of the arm is decided by the force sensors and errors in the horizontal planes are largely accommodated by the open gripper geometry.

The confusion matrix for all the actual instances of objects is shown in Figure 7. It is color coded by each objects fraction of total occurrence. This is obtained by comparing the true labels with the predicted bounding box that best localises the object, i.e. the one with the highest IOU. The figure provides insights into potential areas of improvement in the object identification and gripping strategy. The poor recall rate for knives can be attributed to a significant portion being mislabelled as forks, which in our case would not affect the clearing performance. The poor recall rate of cups and bowls, however will significantly affect the performance of the system. One way to reduce this error is to find a uniform grasp strategy for bowls and cups. The mislabelling of bowls as spoons probably occur in cases where there is a spoon and bowl in a small portion of the image, which causes YOLO to pick one with the highest confidence. A work around would be to divide the original image into multiple segments and feed them to the network separately. Note that YOLO internally downsamples the images before passing them through the convolution layers.

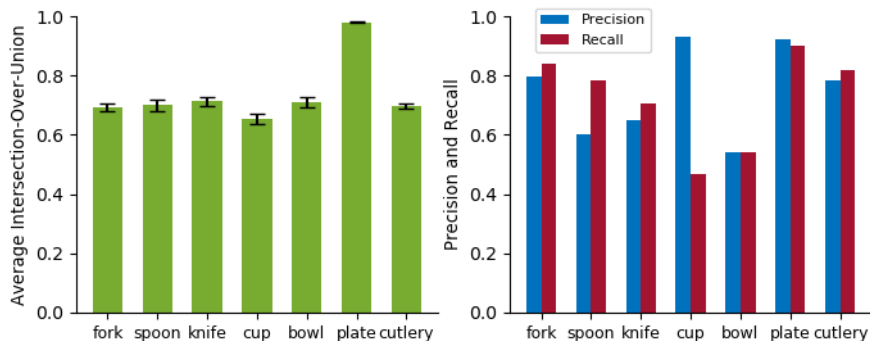


Figure 6: Performance of object detection (YOLO and Hough circle detection) on real-world data.



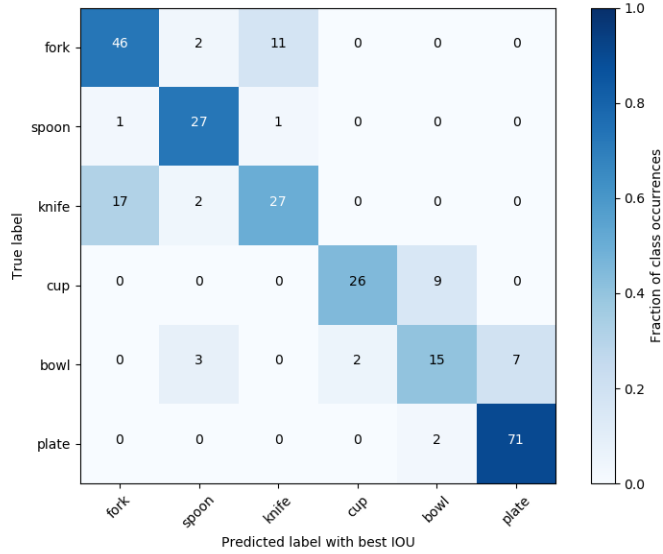


Figure 7: Confusion matrix showing classification performance where the colour coding is as a fraction of total class occurrences in the data set, shown in Table 2

A full breakdown of the items contained in the data set is shown in Table 2 along with the number identified correctly. There may appear to be a discrepancy between the values in Fig. 7 and Table 2. This is because YOLO often provides multiple bounding boxes for the same object. This is more common when the scene is more cluttered and objects are closer together. In Fig. 7 we compare the true label with the predicted bounding box that best localises the object, i.e. the one with the highest IOU. In Table 2 we count an identification as a success if the class prediction is correct and the IOU is above 0.5. The last column includes where objects were identified as similar enough objects, namely items of cutlery being identified as cutlery but not the correct subcategory. For the procedure described above this is acceptable because all cutlery is picked using the same process so it would still be possible to pick a fork misidentified as a spoon. Note that the total of fork identifications that will result in a successful pick, shown in brackets, is equal to the sum of the number of forks predicted as each of the cutlery classes, similarly for spoons and knives. This provides the counterintuitive insight that sometimes a bounding box which incorrectly identifies the object class may localise the object better (higher IOU). Overall, 61% of the objects were identified and labelled with the correct class. A further 12% were identified and labelled with an incorrect but similar enough class for the planning and picking methods described above to work.

Table 2: Breakdown of items in canteen data set. Bracketed value indicate identifications that will still result in a successful pick.

Object Type	Number in set	Number identified
Fork	63	53 (59)
Spoon	37	29 (29)
Knife	54	38 (46)
Cup	58	27
Bowl	37	20
Plate	79	71
Cutlery	171	120 (140)
All	345	238 (252)

Out of the 140 pieces of cutlery identified in the canteen images, the orientation of 95 of these was found

correctly. The orientations for 4 were incorrect and the results for the remaining 41 were inconclusive. In inconclusive cases, the orientation is selected randomly from the two cases which suggests for approximately 20 of these inconclusive cases the orientations would be selected correctly by chance. The base success rate of the orientation estimator is 67.9% and this increases to 82.1% when you consider the chance of guessing inconclusive orientations. This is a sizeable improvement on the alternative of random selection which would have a 50% success rate.

### 3.2.1 Division of Object Complexity

The results of object detection were also used to categorise the trays from the canteen data set into different levels of complexity. We hypothesize that a better performance by the object detection algorithm would indicate a less complex visual scene and hence, an easier manipulation problem. Characteristics from this analysis is then used to simulate our in-lab experimental setup and studies.

We used the average IOU across all objects in the image for deciding the complexity of the scene, which was found by comparing the detection results to the ground truth labels. In any cases where there were multiple bounding boxes for one object, the bounding box with the best IOU was used in the average. If an object was not identified at all or the class was incorrectly identified, the IOU for that object was 0. A value of 1 would be perfect performance, meaning the bounding box from object detection would overlap perfectly with the ground truth bounding box for all objects in the image.

Based on the IOU values we decided to divide each trays into three levels. Level 1 was set as images with an average IOU of above 0.6, level 2 was images with an average IOU between 0.4 and 0.6 and level 3 was images with an average IOU below 0.4. This method of separation resulted in levels 1, 2, and 3 containing 28, 28, and 13 images respectively. The images comprising each level were then analysed subjectively to pick out any common features. Images in level 1 involved objects that were often widely spaced in the image. However, this could sometimes mean cutlery positioned half on a plate, see Fig. 8a. Overall, levels of food waste were minimal. Images in level 2 overall featured an increased level of food waste. Another feature of level 2 was a variety of configurations of cutlery, sometimes these were interlocked or rested on top of each other, see the top two trays in Fig. 8b. There were also slight occlusions. Images in level 3 often involved major occlusions. These were caused by stacked plates, large amounts of food waste, or other waste, such as napkins or plastic yoghurt pots. While subjective, this analysis enabled an assessment of factors that potentially influenced object detection performance, with a view to including key features in a recreation of the environment. The values for average IOU ranged from 0.102 to 0.842 (the extreme cases are shown in Fig. ??).

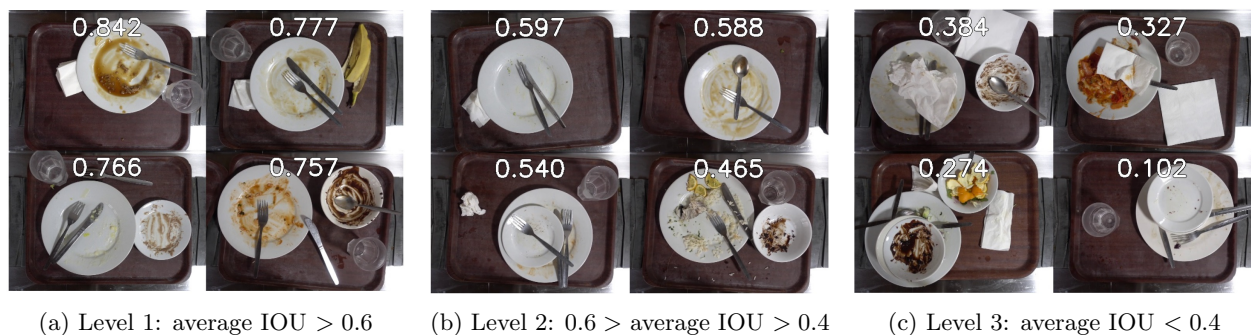


Figure 8: Four example trays for the three levels of complexity observed in the canteen data set

## 4 Experimental Results

To validate our YOLO based control architecture, we designed experiments that closely mimic a real-world canteen environment. The objective of the system is to pick items from a canteen tray and place them in a dishwasher tray. Two strategies were investigated for this task on 25 trays each. The next section details the experimental protocol and the subsequent section presents the results and analysis.

### 4.1 Experimental Protocol

The setup for the in-lab experiments is shown in Fig. 1. the tray from a household dishwasher was placed on the same level as the worktop, as is common with the pass-through dishwashers used in canteen kitchens. The tray contained five classes of objects in arbitrary configurations set manually. The objects we used were forks, spoons, knives, cups and plates. Different varieties of each classes were also used to test the generalizability of the approach.

Two strategies were evaluated for the dishwasher loading problem (Figure 9). The first strategy took visual feedback only at the start of each clearing cycle (i.e one image per tray), while the second strategy iteratively took visual feedback after every attempted pick-and-place process. The first strategy allows faster loading whereas the second strategy allowed the process to correct errors and discover new information at every cycle, at the cost of a slower cycle time. We will be referring to these strategies as the one-shot strategy and the closed-loop strategy from now on.

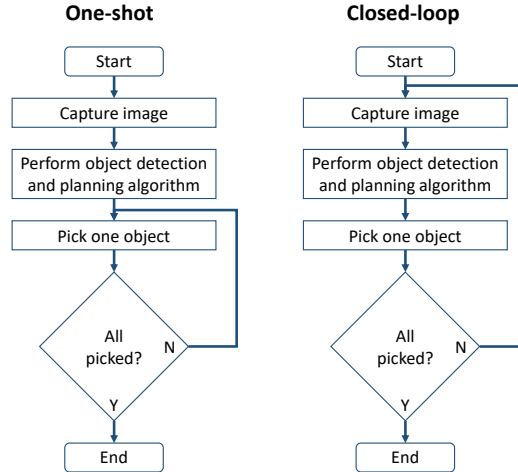


Figure 9: Flowcharts of the two strategies investigated in this study. The looping for each object is not shown in Figure 2.

For each strategy, 25 configurations of a cluttered canteen tray were artificially simulated. These 25 configurations were in turn subdivided by the three complexity levels based on the insights from the previous experiments. They were divided in the ratio 10:10:5 for complexity level 1, 2 and 3 respectively. The problem of placing objects into the dishwasher was not the focus of this work having been covered in greater detail by other researchers (Jiang et al., 2012b; Jiang et al., 2012a). Hence, we opted for a predetermined placement strategy for each item, irrespective of the current state of the dishwasher tray. There were three predetermined trajectories for placing objects in the dishwasher tray: one for cutlery, one for cups, and one for plates. These were created by moving between handcrafted waypoints found through experimentation. The next section presents the results of the one-shot control strategy.

## 4.2 Results for the one-shot strategy

The performance of the one-shot strategy is summarised in Table 3. The overall clearance success rate was around 59%. Given the fact that the whole system is based on a single 2D image, a simple pinch gripper and a 1D force feedback, this result is reasonably good. Note that the objects we have used were arbitrarily chosen. The deep neural network was never trained on these objects and the planning strategy has not been designed specifically for the particular design we used. The only exception here is the ‘plate’ class, where a hand tuned parameter is used for detection.

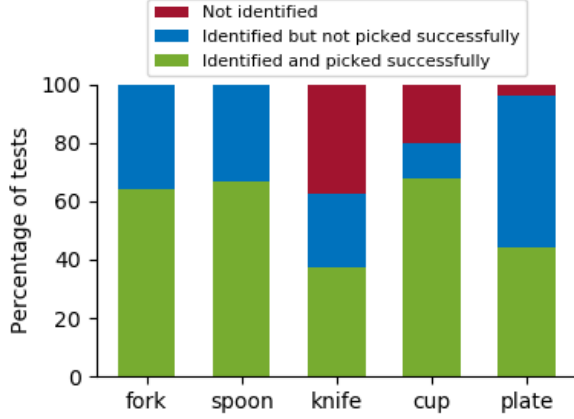
Table 3: Overall system performance in simulated kitchen conditions with the one-shot strategy.

Metric	Result	Definition
Total trays	25	Trays in experiment
Total ground-truth objects	109	25 forks, 24 spoons, 8 knives, 25 cups, 27 plates
Total objects correctly identified	92	22 forks, 23 spoons, 4 knives, 19 cups, 24 plates
Total objects successfully removed	64	16 forks, 16 spoons, 3 knives, 17 cups, 12 plates
Total picks attempted	116	Total pick-and-place actions planned from object detection results
Planning success rate	92.0 %	$\frac{\text{Total theoretically appropriate plans}}{\text{Total plans}}$
Clearing success rate	<b>58.7%</b>	$\frac{\text{Total objects removed}}{\text{Total objects}}$
Effective picking rate	55.2%	$\frac{\text{Total objects removed}}{\text{Total picks planned}}$
Total completely cleared trays	3	Trays where all objects were successfully removed
Average cycle time	159 s	Time from initial image capture to end of pick-and-place actions

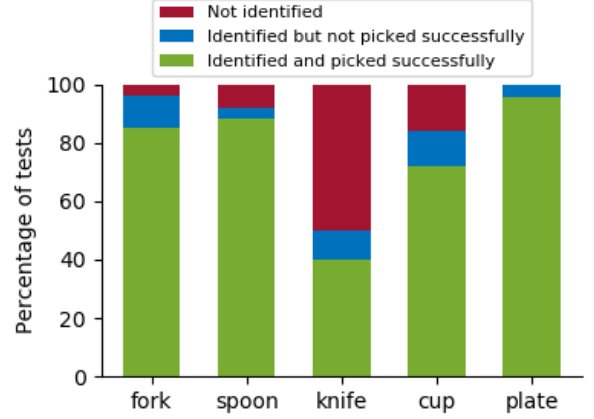
The sources of errors can be analysed from Figure 10a. The object identification module performed very well for forks, spoons and plates. Cups and knives had a lower rate of detection as observed from our analysis in section 3. Knives, in particular fared poorly. We believe this was mainly because of its shiny surface, exacerbated by our lighting condition. The object identification module did label several objects with multiple bounding boxes which resulted in 24 extra picks being planned. This increased the overall cycle time. However, there were occasions during the experiments when the object was not picked with the first attempt (due to a prediction with poor localisation but high confidence), but the object was then picked on the second attempt due to the double identifications.

The proposed planning strategy fared very well, even though it was developed on low-dimensional scene information. This shows that for real-world scenarios, the generalizability and robustness of the object detection module overpowers the required complexity of the planning phase. In other words, it is more important to have reliable identification of objects than a complex physics-based planning algorithm for successful grasps. A notable fraction of items were identified correctly but did not lead to a successful grasp. This could be because of errors in the bounding box estimation (low IOU) or displacement of items during the execution of the plan. The closed-loop strategy presented in the next section investigates the contribution of both. It is interesting to see that plates had a higher chance for failed grasps and cups had a lower chance for failed grasps. This can be attributed to the motion strategy (as mentioned in section 3) and its robustness to errors in object localization. A straightforward way to reduce the failed grasps is to increase the forces applied by the pinch-gripper so that even grasps at the edge of the plate can lead to a stable grip. Another improvement would be to include additional tactile sensors to the gripper to adjust its grip based on the tactile feedback.

All the 25 tray settings we used and their final states are shown in Figure 11. We set up three levels of complexities based on our previous insights to evaluate the performance of our algorithm in a systematic way. The first two rows in Figure 11 belong to level 1, the next two to level 2 and the final row to level 3. The average IOU for each levels were 0.80, 0.72 and 0.55 respectively. Even though the trays were set up based on our subjective insights, we were still able to recreate an environment closely matching the real-world scenarios, thus validating the experimental protocol. Level 1 had 38 objects and 25 were successfully picked (65.8%). Level 2 had 43 objects and 25 were successfully picked (58.1%). Level 3 had 28 objects and



(a) Performance breakdown of the one-shot strategy.



(b) Performance breakdown of the closed-loop strategy.

Figure 10: Comparison of the two control strategies proposed in this study. The closed-loop strategy can compensate for failed grasps to a large extent.



(a) Trays at the start of experiment



(b) Trays at the end of experiment

Figure 11: State of the trays before and after the experiment with the one-shot strategy.

14 were successfully picked (50.0%). It can be concluded that the clearance rate is directly related to the average IOU values in the one-shot strategy. Hence, increasing the IOU metrics is one way to improve the clearance performance. However, improving the average IOU requires more complex learning architectures, higher quality images and precisely labelled data. Therefore, a better strategy is to use the information gained/modified after each picking cycle to iteratively modify the planning strategy as proposed in the next section.

### 4.3 Results for the closed-loop strategy

The performance of the closed-loop strategy is summarised in Table 4. The overall clearance success rate was around 81%, much higher than the one-shot strategy. This, however, comes with an almost 100% increase in the cycle time. With the incorporation of the feedback, we can now achieve a clearing success rate which is as good as the planning success rate. Unsuccessful plans are the cases in which the wrong order of items are picked, which would most likely result in unsuccessful grasps. Some of these plans also leads to unintentional removal of items, which are not counted as a successful clearance since they were not placed

into the dishwasher.

Table 4: Overall system performance in simulated kitchen conditions with the closed-loop strategy.

Metric	Result	Definition
Total trays	25	Trays in experiment
Total ground-truth objects	111	27 forks, 25 spoons, 10 knives, 25 cups, 24 plates
Total objects correctly identified	98	26 forks, 23 spoons, 5 knives, 21 cups, 23 plates
Total objects successfully removed	90	23 forks, 22 spoons, 4 knives, 18 cups, 23 plates
Total picks planned	134	Total pick-and-place actions planned from object detection results
Planning success rate	84.0%	$\frac{\text{Total theoretically appropriate plans}}{\text{Total plans}}$
Clearing success rate	<b>81.1%</b>	$\frac{\text{Total objects removed}}{\text{Total objects}}$
Effective picking rate	67.2%	$\frac{\text{Total objects removed}}{\text{Total picks planned}}$
Total completely cleared trays	12	Trays where all objects were successfully removed
Average cycle time	296 s	Time from initial image capture to end of pick-and-place actions

The sources of errors for the closed-loop strategy can be analysed from Figure 10a. With the addition of the feedback, the number of objects that were identified, but not successfully grasped decreases dramatically. This is primarily because the system can now adjust its plan to any changes in the scene. Additionally, now the system has the ability to obtain new information once occluding objects are removed. Knives and cups are still poorly identified, reducing the overall performance of the system.

All the 25 tray settings we used and their final states are shown in Fig. 12. Like the previous experiment, the first two rows in Figure 12 belong to level 1, the next two to level 2 and the final row to level 3. The average IOU for each levels were 0.765, 0.629 and 0.501, respectively, indicating that set of trays were slightly more complex than the previous set. Level 1 had 44 objects and 39 were successfully picked (88.6%). Level 2 had 43 objects and 37 were successfully picked (86.0%). Level 3 had 24 objects and 14 were successfully picked (58.3%). For the closed-loop strategy we see that clearance rate does not depend much on the average IOU for levels 1 and 2. This is probably because of the improved detection after every pick-and-place cycle. For level 3 this improvement is not present probably because of the simulated food waste that is never removed. One of the interesting observation, particularly to the closed-loop strategy is the relation between the average IOU and the clearing success rate as shown in Figure 13. The mapping is almost logarithmic with the intrinsic noise of the process. This is very important because by just estimating the average IOU values of each plate, a reasonable estimate of the clearing rate can be found, without actually running the tests. A further study would be to investigate how this pattern shifts with better gripping mechanisms and higher quality object detection networks.

#### 4.4 Comparison and Discussions

As expected, the closed-loop strategy showed greater success in object removal than the one-shot strategy, but at the cost of an increase in cycle time. Optimisation of cycle time was not the main objective here, but it would be possible to decrease cycle time. For example, the movement speed of the robot arm could be increased. The time taken to run YOLO on average was 11.1 seconds which is a significant contribution to cycle time when the process is run multiple times. However, by running the YOLO network on a GPU is possible to get real-time detection of objects, as it is most well-known for (Redmon and Farhadi, 2018).

A benefit of the closed-loop strategy is the use of interactions with the environment to reveal more information about the scene. Consider the test shown in Fig. 14. Images 3 and 4 in the set show an unsuccessful attempt to pick up the plate due to its central location in the tray - the gripper contacted the raised lip of the tray rather than the surface of the tray. However, this failed attempt moved the plate further to the left of the tray, along with the napkin it was in contact with. This began to reveal a spoon that was previously totally obscured from view. The plate was then successfully removed from its shifted location, the spoon



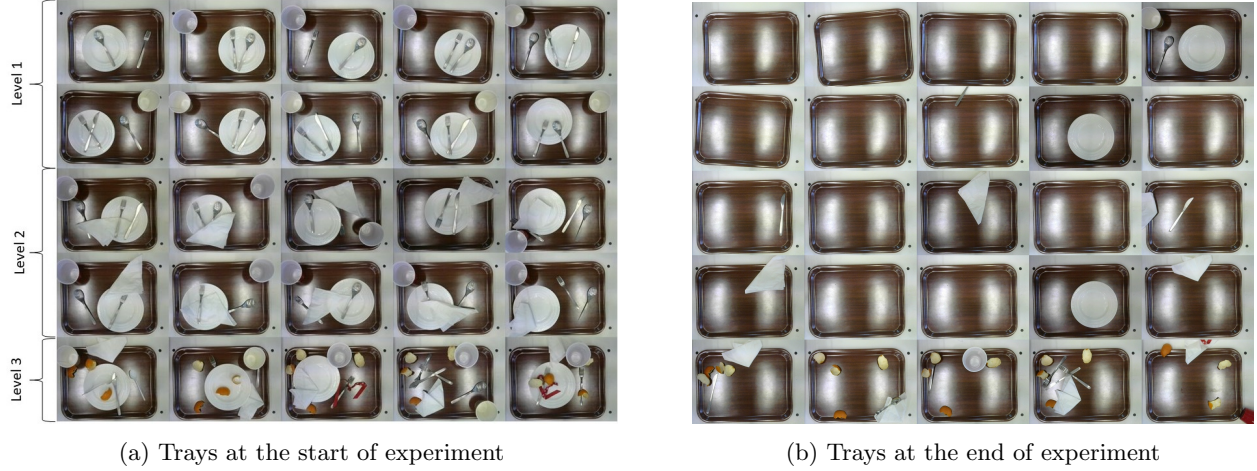


Figure 12: State of the trays before and after the experiment with the closed-loop strategy.

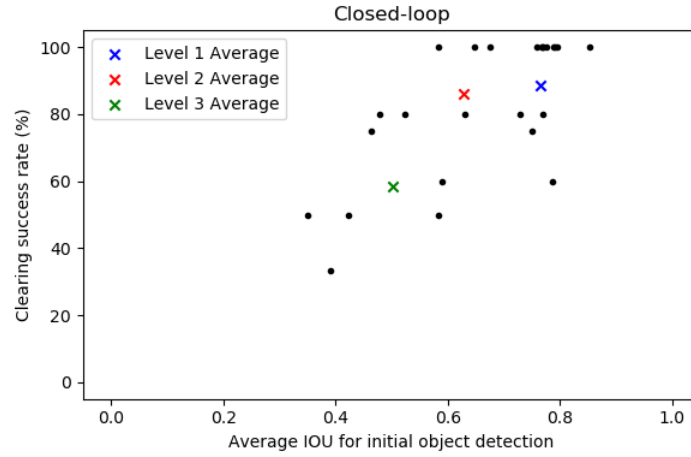


Figure 13: Relation between the average IOU for each tray and the clearing success rate for the closed-loop strategy.

totally revealed and then successfully removed as well. This shows that by incorporating simple feedback and leveraging the effects of action on perceptions, there is no need to do complex multi-angle view of system or create a 3D point cloud to discover all objects present in the scene initially. Instead, the requirements of the task - removal of objects from a scene - facilitates interactions with the environment that can reveal further information about the contents of the scene.

Failures could occur at several stages in the procedure and were caused by different elements of the system either in isolation or in combination. One common method of failure was when a plate was picked with a piece of cutlery still on top of it (typically the knife). As the plate was lifted up and rotated to be placed into the dishwasher tray, the piece of cutlery slipped off the plate. An example from the closed-loop strategy is shown in Fig. 15. The movement of the plate meant it tended to land on the upper edge of the tray, sometimes sliding further out of view. Examples of this can be seen in Fig. 11b and Fig. 12b where it appears the tray has been completely cleared but closer inspection of the top portion of the image shows the fallen piece of cutlery. This failure case was seen with both the one-shot strategy and the closed-loop strategy. In both cases the cause of failure may appear to be a failure in planning but this was not the case. The failure either stemmed from a failure in object detection or, restricted to the one-shot strategy, a failure in manipulation. In the one-shot cases, the object may have been identified successfully at the start of the

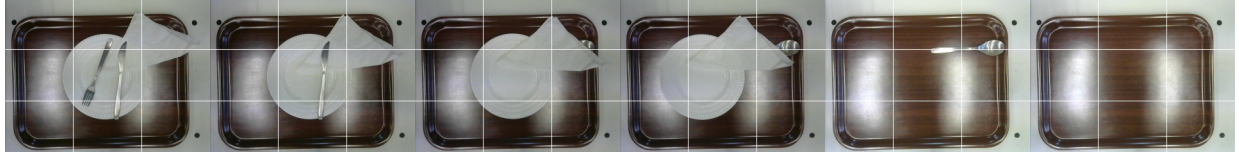


Figure 14: An example of the closed-loop strategy. This example shows plate being moved from unsuccessful picking position to successful one and a spoon being revealed. All the frames used for motion planning are shown. Grid lines are provided to show the displacement of objects in between frames.

run but interactions with the tray caused displacement of the object. The picking trajectory planned from the initial image attempts to pick the object from its original position and fails. An example of this is shown in Fig. 16 where contact with the plate causes the knife on top to slide and the subsequent pick does not grasp the knife correctly. Since the one-shot strategy assumes perfect success at picking, the plate is then picked and the cutlery slides off. It is possible to recover from this failure using the closed-loop strategy but not with the one-shot strategy.



Figure 15: Example from the closed-loop strategy showing a failure in planning caused by failure in identification followed by a failure in manipulation. Firstly, the knife was not identified during object detection so is tipped off when the plate is picked. Secondly, the napkin wrapped between the gripper and plate causes the plate to slip out of the grip during movement towards dishwasher.

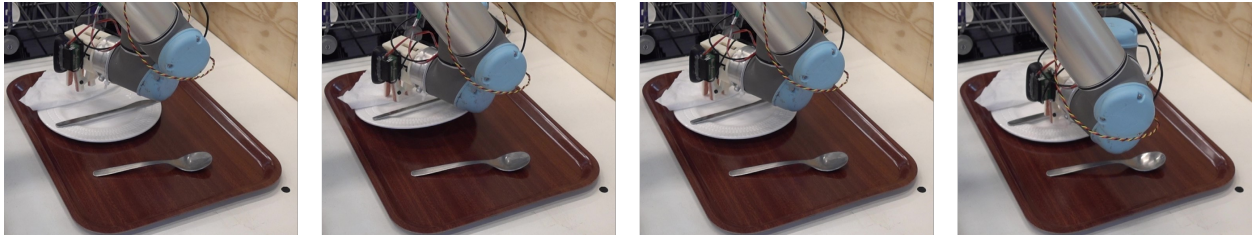


Figure 16: Example from the one-shot strategy showing disturbance of scene causing picking failure. A downward force towards the edge of plate causes it to tip and the knife to move location. The preplanned pick is then in wrong place.

One of the drawbacks of the closed-loop strategy was that the process could get stuck in a loop trying to pick an unpickable object. This happens when an object is identified with high certainty but the particular geometry makes it unsuitable for the pinch-gripper or in a pose unanticipated when planning gripping point (e.g. upturned cups, upright spoon in a cup). In some cases, the attempted pick caused sufficient changes to the tray configuration that meant subsequent picks were successful. However, in other cases little to nothing changed and subsequent picks followed the same pattern, causing a loop. To avoid this we set a cap on the maximum number of attempts for each tray. One of the aspects we ignored in this work, is the problem of collision avoidance. Although, collision avoidance can be implemented with the low-dimensional information from the YOLO network, the problem becomes computational expensive as the number of items increase.



With the addition of solid food waste, the identification problem becomes nontrivial too. The addition of solid food waste and napkins also makes manipulation more complicated since it can become trapped between the gripper and the object, reducing grip strength, as seen in Fig. 15. The manipulation was also subject to arbitrary restrictions. For example, in the case of cups, the inverted and on-the-side states were never considered. This is because such cases were either not graspable with the gripper or additional three dimensional information was required for the configuration estimation module.

## 5 Conclusion

This paper presents an autonomous dishwasher loading robot using pre-trained deep neural networks. The key idea of the work is to use the power of deep neural networks pre-trained on large commonly available dataset. By conforming the perception and planning systems to the pre-trained networks, we are able to develop a control framework that requires no training phase, robust to visual noise and generalizable to novel object designs and requires only a single 2D image feedback. The grasp point estimation and planning algorithm is independent of the object class geometry, thereby requiring no parameter tuning for grasping and placing new unseen objects. We incorporate a simple force feedback to the system to remove the requirements for depth information. For the closed-loop strategy proposed here, we achieve a clearing rate of 81%. Looking at the state-of-the-art in object grasping, our method performs comparatively well (Table 5). Unlike the referenced works, our method does not require a training phase, where task-specific samples have to be obtained and trained. Hence, this approach is highly appealing for fast plug-and-play applications, where the user does not require high domain-specific knowledge or training hardware. The pre-trained network is trained on the most commonly available form of object detection data. Our analysis and experiments are performed in real-world scenarios. A general purpose industrial arm and simple pinch-gripper hardware is used for the task, making the system suitable for other tasks, unlike current commercial solutions. With the constant progress in labelling and available of object detection databases, we can expect such an approach to be easily extendable to other common objects.

Table 5: State-of the-art performance comparison

	Clearing Rate (%)	Training Samples	Sensory Feedback
(Lenz et al., 2015)	84	1035	RGB-D
(Pinto and Gupta, 2016)	73	50,000	RGB
(Mahler et al., 2017)	94	6,700,000	Point Cloud
(Levine et al., 2018)	77	900,000	RGB
(Kalashnikov et al., 2018)	96	580,000	RGB
<b>This work</b>	81	0	RGB

It is evident that the performance of our method is dependent on the complexity of the visual scene. For the tray complexities of level 1 and 2, our method performs with a clearing rate of 89% and 86%, respectively. Although this performance is comparable with the state-of-the-art, for commercial applications, it is vital to have near perfect clearing rate. One workaround is to use the platform as a collaborative robot, where the robotic clearing could be used in parallel with human workers in a high through-put restaurant, for example. In such a case, the user can facilitate the perception pipeline by clearing food waste and occlusions and the platform will aid in clearing and placing the items quickly and efficiently. As we use a general purpose manipulator, such a system can be easily adapted to perform other tasks in a kitchen.

The simplicity of our robotics platform and sensory system largely facilitated in reducing the complexity of our approach. However, this would always limit the complexity of manipulation skills achievable. For manipulation of complex object geometries, fragile items and food waste, it might be necessary to develop gripping mechanisms with additional degrees of freedom and tactile sensing abilities. Tactile sensing will also be important to detect slips and evaluate the strength of the grip. Similarly, addition of depth informa-

tion will significantly improve the performance of our approach without additional computational overhead. Another scope for improvement is on the use of object segmentation networks instead of object detection networks. Object segmentation networks provide the object class and its boundaries. This higher dimensional information has the potential to improve the performance of the orientation and configuration estimation module. Another interesting future work is to investigate how we can leverage the ability to change the camera point-of-view to improve the detection performance. Along similar lines, strategies for obtaining new information by acting on the environment in cases where the object of concern is completely occluded is another problem to pursue.

## Acknowledgments

This work was supported by BEKO PLC and Symphony Kitchens.

## References

- Andrychowicz, O. M., Baker, B., Chociej, M., Jozefowicz, R., McGrew, B., Pachocki, J., Petron, A., Plappert, M., Powell, G., Ray, A., et al. (2020). Learning dexterous in-hand manipulation. *The International Journal of Robotics Research*, 39(1):3–20.
- Bicchi, A. and Kumar, V. (2000). Robotic grasping and contact: A review. In *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, volume 1, pages 348–353. IEEE.
- Bohg, J., Hausman, K., Sankaran, B., Brock, O., Kragic, D., Schaal, S., and Sukhatme, G. S. (2017). Interactive perception: Leveraging action in perception and perception in action. *IEEE Transactions on Robotics*, 33(6):1273–1291.
- CambridgeConsultants (2019). Turbo clean: Tackling the most unloved job in the commercial kitchen. <http://cambridgeconsultants.com/turboclean>.
- Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*.
- Dillmann, R. (2004). Teaching and learning of robot tasks via observation of human performance. *Robotics and Autonomous Systems*, 47(2-3):109–116.
- Dishcraft (2019). <https://dishcraft.com>.
- Dogar, M. and Srinivasa, S. A framework for push-grasping in clutter. *Robotics: Science and systems VII*, 1.
- Ekvall, S. and Kragic, D. (2007). Learning and evaluation of the approach vector for automatic grasp generation and planning. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pages 4715–4720. IEEE.
- Finn, C. and Levine, S. (2017). Deep visual foresight for planning robot motion. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2786–2793. IEEE.
- Girshick, R. (2015). Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448.
- Girshick, R., Donahue, J., Darrell, T., and Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587.
- He, K., Gkioxari, G., Dollár, P., and Girshick, R. (2017). Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969.

- He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.
- Hinton, G. E. and Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507.
- Jiang, Y., Lim, M., Zheng, C., and Saxena, A. (2012a). Learning to place new objects in a scene. *The International Journal of Robotics Research*, 31(9):1021–1043.
- Jiang, Y., Zheng, C., Lim, M., and Saxena, A. (2012b). Learning to place new objects. In *2012 IEEE International Conference on Robotics and Automation*, pages 3088–3095. IEEE.
- Kalashnikov, D., Irpan, A., Pastor, P., Ibarz, J., Herzog, A., Jang, E., Quillen, D., Holly, E., Kalakrishnan, M., Vanhoucke, V., et al. (2018). Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation. *arXiv preprint arXiv:1806.10293*.
- Katz, D. and Brock, O. (2008). Manipulating articulated objects with interactive perception. In *2008 IEEE International Conference on Robotics and Automation*, pages 272–277. IEEE.
- Koga, Y. and Latombe, J.-C. (1994). On multi-arm manipulation planning. In *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, pages 945–952. IEEE.
- Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.
- Lenz, I., Lee, H., and Saxena, A. (2015). Deep learning for detecting robotic grasps. *The International Journal of Robotics Research*, 34(4-5):705–724.
- Levine, S., Finn, C., Darrell, T., and Abbeel, P. (2016). End-to-end training of deep visuomotor policies. *The Journal of Machine Learning Research*, 17(1):1334–1373.
- Levine, S., Pastor, P., Krizhevsky, A., Ibarz, J., and Quillen, D. (2018). Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *The International Journal of Robotics Research*, 37(4-5):421–436.
- Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer.
- Lynch, K. M. and Mason, M. T. (1996). Stable pushing: Mechanics, controllability, and planning. *The international journal of robotics research*, 15(6):533–556.
- Mahler, J., Liang, J., Niyaz, S., Laskey, M., Doan, R., Liu, X., Ojea, J. A., and Goldberg, K. (2017). Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics. *arXiv preprint arXiv:1703.09312*.
- Miller, A. T., Knoop, S., Christensen, H. I., and Allen, P. K. (2003). Automatic grasp planning using shape primitives. In *2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422)*, volume 2, pages 1824–1829. IEEE.
- Pajarinen, J. and Kyrki, V. (2017). Robotic manipulation of multiple objects as a pomdp. *Artificial Intelligence*, 247:213–228.
- Pinto, L. and Gupta, A. (2016). Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours. In *2016 IEEE international conference on robotics and automation (ICRA)*, pages 3406–3413. IEEE.
- Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016a). You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788.

- Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016b). You only look once: Unified, real-time object detection. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Redmon, J. and Farhadi, A. (2018). YOLOv3: An incremental improvement. *arXiv*.
- Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99.
- Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al. (2015). Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252.
- Saxena, A., Driemeyer, J., and Ng, A. Y. (2008a). Robotic grasping of novel objects using vision. *The International Journal of Robotics Research*, 27(2):157–173.
- Saxena, A., Wong, L. L., and Ng, A. Y. (2008b). Learning grasp strategies with partial shape information. In *AAAI*, volume 3, pages 1491–1494.
- Smith, C., Karayiannidis, Y., Nalpantidis, L., Gratal, X., Qi, P., Dimarogonas, D. V., and Kragic, D. (2012). Dual arm manipulation—a survey. *Robotics and Autonomous systems*, 60(10):1340–1353.
- Tian, L., Thalmann, N. M., Thalmann, D., Fang, Z., and Zheng, J. (2019). Object grasping of humanoid robot based on yolo. In *Computer Graphics International Conference*, pages 476–482. Springer.
- Vischer, D. (1992). Cooperating robot with visual and tactile skills. In *Proceedings 1992 IEEE International Conference on Robotics and Automation*, pages 2018–2025. IEEE.
- Weisz, J. and Allen, P. K. (2012). Pose error robust grasping from contact wrench space metrics. In *2012 IEEE international conference on robotics and automation*, pages 557–562. IEEE.
- Wilfong, G. (1991). Motion planning in the presence of movable obstacles. *Annals of Mathematics and Artificial Intelligence*, 3(1):131–150.
- Yu, K.-T., Bauza, M., Fazeli, N., and Rodriguez, A. (2016). More than a million ways to be pushed. a high-fidelity experimental dataset of planar pushing. In *2016 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 30–37. IEEE.
- Zhao, Z.-Q., Zheng, P., Xu, S.-t., and Wu, X. (2019). Object detection with deep learning: A review. *IEEE transactions on neural networks and learning systems*, 30(11):3212–3232.