A robust robot design for item picking

Causo, Albert; Chong, Zheng-Hao; Luxman, Ramamoorthy; Kok, Yuan Yik; Yi, Zhao; Pang, Wee-Ching; Ren, Meixuan; Teoh, Yee Seng; Jing, Wu; Tju, Hendra Suratno; Chen, I-Ming

2018

Causo, A., Chong, Z.-H., Luxman, R., Kok, Y. Y., Yi, Z., Pang, W.-C., . . . Chen, I.-M. (2018). A robust robot design for item picking. Proceedings of 2018 IEEE International Conference on Robotics and Automation (ICRA), 7421-7426. doi:10.1109/ICRA.2018.8461057

<https://hdl.handle.net/10356/142666>

<https://doi.org/10.1109/ICRA.2018.8461057>

© 2018 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works. The published version is available at:
<https://doi.org/10.1109/ICRA.2018.8461057>.

Downloaded on 26 Apr 2023 20:23:38 SGT

A Robust Robot Design for Item Picking

Albert Causo¹, Zheng-Hao Chong², Ramamoorthy Luxman³, Yuan Yik Kok⁴, Zhao Yi⁵, Wee-Ching Pang⁶, Ren Meixuan⁷, Yee Seng Teoh⁸, Wu Jing⁹, Hendra Suratno Tju¹⁰ and I-Ming Chen¹¹

Abstract—In order to build a stable and reliable system for the Amazon Robotics Challenge we went through a detailed study of the performance and system requirements based on the rules and our past experience of the challenge. The challenge was to build a robot that integrates grasping, vision, motion planning, among others, to be able to pick items from a shelf to specific order boxes. This paper presents the development process including component selection, module designs, and deployment. The resulting robot system has dual 6 degrees of freedom industrial arms mounted on fixed bases, which in turn are mounted on a calibrated table. The robot works with a custom-designed top-open extendable shelf. The vision system uses multiple stereo cameras mounted on a fixed calibrated frame. Feature-based comparison and machine-learning based matching are used to identify and determine item pose. The gripper system uses suction cup and the grasping strategy is pick from the top. Error recovery strategies were also implemented to ensure robust performance. During the competition, the robot was able to pick all target items with the shortest amount of time.

I. INTRODUCTION

Amazon Robotics holds a robotic challenge annually since 2015 in order to strengthen ties between the industry and research community, and explore solutions to the problem of automated item picking and stowing. Lessons and observations of the very first Amazon Robotics Challenge (ARC) is summarized by Correll et.al., [1]. Important factors to the success of the robot systems are stability and reliability, which could either mean being able to recover from errors such as dropped item or wrongly picked item or being tolerant to miss-calibrations or other hardware problems. Based on our own observations during the competitions, picking systems fail due to hardware issues (unconnected cables, spoiled cameras or sensors, etc), electrical (subsystems off during the run, etc), or software (undefined state, undefine input, wrong encoding, etc).

The major contributions of this paper include:

- 1) Building a mixed-mode approach to perception that combines machine learning and conventional feature-based strategy
- 2) Designing an extendable shelf in aid of perception

This work has been supported by the Singapore Agency for Science, Technology and Research (A*STAR) under National Robotics Programme - Robotics Enabling Capabilities and Technologies SERC Grant No. 162 25 00003 and the National Research Foundation - Proof-Of-Concept (NRF-POC) Project No. NRF2015NRF-POC002-011.

¹Corresponding author – acauso@ntu.edu.sg

The authors are from the Robotics Research Centre, School of Mechanical and Aerospace Engineering, Nanyang Technological University, Singapore, {²zhchong, ³rluxman, ⁵zhao.yi, ⁶weeching, ⁸ysteoh, ⁹wujing, ¹⁰hstju, ¹¹michen}@ntu.edu.sg, {⁴yk004, ⁷mren001}@e.ntu.edu.sg

- 3) Improving upon existing feature-based matching strategy and machine learning method using sliding windows

In this paper, we discuss our approach to designing and building a robotic picking system by presenting the different modules and the requirements or problem that had to be solved for each one. We present the requirements based on rules of the Challenge and the design decisions taken based on these requirements in Section II. The rest of the robot system will be detailed in Sections III (Hardware), IV (Software), V (Motion Planning), VI (Perception), VII (End Effector). Our system's performance during the Challenge is discussed in Section VIII.

II. AMAZON ROBOTICS CHALLENGE 2017

A. Different from Past Challenges

The Amazon Robotics Challenge (ARC) is a competition held by Amazon Robotics annually for the past 3 years with the aim of exploring robotic solutions to the item picking problem. The competition held in 2017 was different from the previous two challenges on a few major points:

- The Amazon shelf was not used, instead, participants were free to design their own shelves as long as it conformed to the volumetric requirements set by Amazon;
- The picked items must be placed in specific order boxes, unlike before where all picked items were dropped in a single red tote;
- The number of training items was increased to 40;
- Half of the items used in every task were new items, which were given only 30 minutes prior to the beginning of the task.

B. The Picking Task

The rules¹ for the picking task is summarized below:

- The items to be used (or "competition set"), composed of 16 training items ("training set") and 16 new items, will be given to the team 30 minutes before the run. The team can do whatever they wish with the items, except to damage or modify them.
- After 30 minutes, the competition has to be returned to the judges.
- The judges will then hand over the item one by one to the team, which can then be positioned in the shelf by hand by the team.

¹<https://www.amazonrobotics.com/#/roboticschallenge/rules>

Fig. 1. The system set-up.

- Afterwards, the team will receive a thumbdrive from the judges containing the order file and the item location file.
- The judges will do some last minute rearrangement of the items in the shelf.
- Picking can now proceed.

Each run lasts for only 15 minutes. A team can call a reset, for whatever reason, as long as items have not been picked yet. Once a reset is called, the team has five minutes before they have to restart the run with the remaining time minus a one minute penalty.

C. Scoring the Run

Points are given each target item picked and placed into the correct order box (10 points). Bonus points are given when the item is new (10 points), or if each box can be closed without moving any items (10 points), or 1 point for every 5 seconds or fraction thereof remaining on the clock when the task is completed.

Points are subtracted when items are not in the shelf or the order box (for target items only) or grasped by the robot when time runs out (-15 points). A penalty of 5 points is given for items dropped into a box from a height of more than 15 cm, for items protruding more than 2 cm out of the shelf, and for minor damage to an item or box. A 20-point penalty is given for major damage to an item or a box.

III. HARDWARE SET-UP

A. System Set Up

The robot system comprises of two lightweight robot manipulators, three stereo cameras and two custom-built robotic grippers. The robots are mounted on a table that covers most of the working space for packing. The power supplies, hardware controller and pneumatic system are placed and stored under the table. A camera-stand hangs above the table and carries two ZED cameras mounted above the two storage bins while the other ZED camera is mounted directly over the packing area of the table. Electric magnets and stretch bands are used to keep the order boxes in place and open

Fig. 2. The top-open drawer-style shelf. Closed state (above) and open state (below).

during the picking process. Built-in lasers help keep the table and the camera frame aligned and calibrated. Fig.1 shows the design of the robot system set up.

With a dual robot design, we could double the working speed of the whole system, when motion is planned properly. We chose UR5 (Universal Robots) so that the whole system could be broken apart and be lightweight enough to check in flights. The dimensions of the table is just enough to cover the work space of two UR5. The dimensions of the camera frame or the tables are less than 2 meters, to fit airline requirements for checked items.

B. Storage system (Shelf)

According to the rules, the storage system must have a volume of less than $95,000 \text{ cm}^3$, must occupy an area no more than $5,000 \text{ cm}^2$, and any of its dimensions must not exceed 125 cm. To prevent items from spilling out of a front-open shelf, we turned it on its back to become a top-open shelf. To get around the size limitations, we designed our shelf to extend during picking operations, similar to a drawer. Lastly, to ensure items separate from each other as the shelf extends, we put flexible ridges in the shelf's base. Fig. 2 shows the shelf in closed and open states.

The top-open extendable design also has added benefits:

- Easier motion planning since the arm's approach is always from the top, where it's not cluttered.
- Stable grasping since we use suction and the item aligns with the gravity vector once it gets lifted off the shelf.
- Extending the shelf provides larger workspace, which eliminates the need for a temporary holding location for items. It also eliminates the need to take out and return items back into the shelf, eliminating error during item transfers or spending robot time on arranging items.
- The flexible ridges help spread out the items giving enough separation space between items (i.e., less clutter) which helps the vision system.

IV. SOFTWARE ARCHITECTURE

The software architecture of our robot picker system is shown in Fig. 3. Due to the processing power required to

Fig. 3. The system architecture

run all the software modules and the need to keep the robots stable while in operation, we distributed the software among three computers.

- PC 1: Handles manipulator control, motion & grasp planning, task planning, task execution and user interface. It's also the center of the communication module between the PCs. Manipulator control has been separated from the vision modules to isolate the robot arms' real-time performance in case the perception computation becomes overloaded.
- PC 2: Connects to all camera modules and handles object recognition and pose estimation using traditional features-based method.
- PC 3: Implements the vision & perception module using machine learning method.

Our robot system integrates several sub-systems into a single, high-level autonomous capability via a state-machine implementation using the SMACH library for ROS. Each state has at least 2 status which is either "Succeeded" or "Aborted". The higher level planing module would decide based on this status. For example, a "Succeeded" status would let the system to proceed to the next step, else an "Aborted" status would trigger error message and recovery.

We created SMACH states that wrap the invocation of other nodes, including action servers and topic-based work flows and pass the result to subsequent stages.

Multiple state machines are running at the same time to cover different work flows and may interrupt one another if needed. For example, the "Picking Watcher State Machine" may stop and take over the robot control of the main "Picking State Machine" if the system is not able to finish the picking task within the time limit.

This software strategy prevents incurring penalties during a task run since priority modules could take over the system, depending on the input, time, or safety situation of the robots.

V. MOTION PLANNING

A major factor in designing our robot system is the individual and combined workspaces of the two arms. Fig 4 shows the reachability of the robot system as analyzed

Fig. 4. Reachability of the dual arm system. The red and green spots are reachable only by either arms. The blue and yellow ones are reachable by both arms. We assigned the blue for picking (location of the shelf) and the yellow for placing (location of the order boxes).

Fig. 5. Workspace planning to prevent collision. The red and blue zones means robot arms can move freely. The green zone is a shared cell which requires permission to enter if the other arm is still inside it.

using fast inverse kinematics. The red and green spots are areas where the other robot arm won't be able to reach. The yellow and blue spots show the reachable space for both robot arms. We assigned the blue for the location of the shelf (picking) and the yellow for the location of the order boxes (placing). We adjusted the position of the two arms with respect to each other to get the maximum blue and yellow area; for the yellow area, it has to be at least 3 times as large as the largest order box available.

For optimum picking performance and avoid collisions between the arms, we define 3 work cells, which are shown in Fig. 5. The red and blue work cells correspond to the left and right arm arm picking areas, where each robot could work without any constraints, except for the shelf itself. The arms are not allowed to cross-over to the other's work cell. Thus, items that fall into a specific work cell belongs to the robot in that cell.

The green work cell shown in Fig. 5 represents shared workspace. The arms have to request access permission to move within this area and need to queue if the work cell is occupied by the other arm.

Once the robot has specified a target goal (i.e., a target item to pick), a motion plan will be generated which includes a sequence of robot arm trajectory points:

- Target suction pose : A final pose for the robot manip-

Fig. 6. Travel cost computation.

ulator to achieve.

- Target approach pose : A short distance pose along the final suction pose vector normal.
- Mid-point poses : A sequence of steps from the robot's current position to the target approach pose. The number of mid-point poses depends on the obstacle between the target approach pose and robot's current position. A larger obstacle to avoid increases the number of mid-point poses. A 3D occupancy mapping, Octomap, is used as a constraint to define the number of mid-points pose.

Then, the path identified in the task space has to be converted into joint space for the robot to execute. Each point along the task space path will generate a list of inverse kinematics solution. The optimal path in the joint space is found through the methods discussed below using linear programming.

$$C_i = \sum_{x=0}^n [q_x - q_y][w] \quad (1)$$

The optimal path is decided by the travel cost along all the task-space points as shown in Fig. 6. There is N layers of "nodes", where each node represents a valid pose of inverse kinematics (IK) solutions that exists within the joint limits. Each Layer represents one trajectory point in the task space. Any layer can have an arbitrary number of nodes although there must be at least one node per layer else this trajectory will become invalid. Every node in layer i connects to every node in layers $(i+1)$. The optimality of the path through joint space depends on the cost definition for moving from one pose to another. Thus the objective is to choose a sequence of nodes (one from each layer) that defines a path from layer 0 to later n such that this path has the lowest possible total cost from start to goal.

In contrast to methods using RRT planning, such as in [2], the current plan optimization is much simpler. Since the planning problem to be solved conforms to the Stagecoach Problem, an optimal path can be found efficiently using Eqn. 1.

VI. PERCEPTION

The introduction of new items 30 minutes prior to the picking task was meant to throw off any learning-based perception system. To address this, we combined traditional feature extraction and matching techniques with CNN classifier. This approach greatly reduces the false detection rate in the picking process.

h

TABLE I
GMS PARAMETERS

Parameter	value
Camera resolution	2560 x 720
Camera tilt angle	0
GMS matcher threshold factor	7.0
GMS matcher grid size	20 x 20
GMS inlier mask withScale bool	true
GMS inlier mask withRotation bool	true
KD tree search radius for normal estimation	0.03

Fig. 7. Sliding Window

A. Identification - Feature Extraction And Matching

GMS (Grid-Based Motion Statistics) [3] is a statistical formulation for partitioning true and false matches of features between images based on the number of neighboring matches. GMS provided a matcher for feature matching. Based on our experiments, this feature matching algorithm is more robust than SIFT and ASIFT; GMS consistently outperforms other real time matchers [3].

A few caveats when using it include:

- 1) The items to compare between two images should have a similar size in pixel scale;
- 2) The images to compare should have a similar size in pixel scale;
- 3) GMS does not work on featureless items such as plastic wine glass or white folder binder.

We further improved this approach by standardizing the training database with the actual run scenarios. That is, we capture training images in similar pixel scale, resolution, background, and camera perspective as that of the images obtained in real-time from cameras during actual run. The GMS parameters used are listed in Table I.

The system uses ZED camera to capture images of the shelf during actual picking time run and to build the image database. The size of the shelf picture was 540 X 440 pixels while the images of each item was around 200 X 200 pixels.

To address the disparity in size between the shelf image (the input image) and the target item image in the database (the comparison image), we implemented a sliding window strategy. Instead of comparing the whole shelf image to the item image, we compare it to a subset of the shelf image using a sliding window. The sliding window measures 270 X 220 pixels and starts at the upper left corner of the image (orange box in Fig. 7). The window moves at a rate of half

Fig. 8. Sample CNN training dataset for an item.

the shelf picture width every iteration, which creates a total of 9 sub-images. GMS is run between the sub-images and the target item image, returning the coordinates of the sub-image with the highest match points.

Wrong matching result still happens despite making the GMS more robust. Thus, to be able to reject wrong matching results, we use two parameters obtainable from the GMS: match points and average match distance of match points. We used past experiment data to establish a rejection threshold. The match could be assumed to have a greater chance of being right if:

- 1) The matched points is larger than 100;
- 2) The average distance is less than 40;
- 3) The ratio of match points and average distance is larger than 1.

B. Machine Learning (Classification Methods)

To address the identification of featureless items or new items, we took to machine learning.

Convolutional neural networks (CNNs) have recently shown tremendous progress, and proven to be effective, in classifying images [4] [5]. CNNs enable learning of rich and layered hierarchical image representations from sufficient training data. However, training a CNN classifier would amount to estimating an excessive number of parameters and it requires a very large dataset of annotated images. Training CNNs from scratch can take tremendous amount of time, which makes it difficult to apply to the Amazon Robotic Challenge because only limited training data (a few multi-perspective photos of the items) and short training time (30 minutes) are provided before a task.

To address this problem, transfer learning [6] of image representations learned with CNNs on large image datasets has been employed in our work. This involves retraining a model that has been already trained on another problem to suit our application. The benefits of transfer learning include overcoming the deficit of training data by adapting previously trained model for other categories, thereby shortening training time and being able to cope with variations in lighting,

Fig. 9. Result of the CNN classifier on identifying the item pie.plates background and viewpoint.

The implementation of the classifier includes acquiring of training data, preprocessing of the acquired data and retraining an existing model with the new data. In particular, 120 RGB images are captured for each face of an item or a total of about 300 to 600 images data can be acquired for each item, depending on the number of faces the item has. Each image is then divided into multiple image samples of 80×80 pixels. Image samples that depict an item partially or fully, as shown in Figure 8, are used as the input of the next steps.

The next process is preprocessing, which includes converting the image format, resizing the images to random sizes, changing other conditions of the images like brightness, and annotating the images for training.

Finally, an open-access trained learning model, Inception-v3 [7], is retrained within the TensorFlow [8] framework, to recognize N number of items during the competition. The trained model is made up of many layers stacked on top of each other [9]. These layers have been trained previously and provide valuable information for classification. The retraining process trains just the last layer and replaces the original output layer with one output for each label in the list of new items.

The implemented N -classes classifier is then used to identify the items in the scene, given an image captured by a camera. First, a sliding window strategy is applied to extract around 200 patches from the image. The patches are squares of width ranging from 50 to 100 pixels, depending on the size of the items, with at least 50% overlap between neighboring patches. Each patch is further rescaled to 120×120 pixels to form a valid input for the classifier.

For each patch, the classifier would produce N number of scores, which indicate the confidence level of the N trained items. The following aggregation formula is used to compute the overall score for item O_n in the image

$$\text{score}(O_n) = \frac{1}{M} \sum_{i=1}^M P(O_n | I_i)^k, \quad (2)$$

where $P(O_n | I_i)$ is the output of the classifier for the probability of item O_n on image patch I_i , M is the number of

Fig. 10. Modified suction cup sewed and glued together

patches in the image and $k \geq 1$ is a thresholding parameter. Higher values of k would retain the high-scoring patches and attenuate the contributions of low-scoring patches. The location of O_n in the image is the centroid position of the image patch at which $P(O_n | I_i)$ is the maximum. Figure 9 illustrates the output result of the classifier when identifying the item `pie_plates` in the scene.

The implemented classifier is subsequently modularized into a ROS package, with CUDA GPU acceleration. The model is trained using a Zotac ZOBX (CPU: Intel core i7-6700, quad-core 3.4 GHz, up to 4.0 GHz, GPU: NVIDIA GeForce GTX 1080 8GB). The time taken to train a 40-class classifier for identifying the known items is about 100 minutes.

During the competition, a different set of items (the “competition set”) would be provided. Half of the items in each competition set would be from the training items set, which have been trained off-site as aforementioned and the other half would be new items. It takes about 15 minutes to train a 16-class classifier, which is also the number of new items for the picking task. The number of training images acquired during the competition was reduced for shorter retraining time. Finally, the time taken for the classifier to produce the classification result (i.e. the perception time) is around 6 seconds.

C. Mixed-Mode Perception That Combines GMS and CNN

For every target item, the system would run both GMS and CNN in parallel to identify the where the item is in the shelf. If the GMS returns a result with high confidence that is not far from the CNN result’s confidence level, we use the output of the GMS.

However, if the GMS result comes with a low confidence level, we run GMS for a few more times. If after several tries the GMS result is still of low confidence level, while the CNN returns a result with higher confidence level, then the CNN result will be used.

If both GMS and CNN return results that are of low confidence level, the system will retry for N times. Otherwise, the system will move on to find the next target item.

The output of both GMS and CNN is the centroid of the sliding window or the patch used in the matching with the highest matching score.

Algorithm 1 shows the pseudo-code implementation of how GMS and CNN are combined.

VII. END-EFFECTOR & GRASPING STRATEGY

To grasp objects with different characteristics (shape, size, weight, and porosity) a general grasping method that can handle different classes of object or an end-effector with multiple grasping mechanisms to tackle the different type of object is needed. For instance, during the Amazon Picking Challenge in 2015, we used an end-effector with multiple grasping mechanism (combination of vacuum suction and fingers) [10]. For Amazon Robotic Challenge 2017 we used a simple and yet robust suction end-effector with no DOF, to handle all the competition target items. The results shows that the end-effector can handle 99% of the items given throughout the competition.

A. Mechanism Design

Taking cue from the learnings of the 2015 Challenge [1], we observed that the use of pneumatic suction mechanisms circumvents the need for in-hand manipulation of the target object. Almost all of the training and competition items are able to be secured and transported with acceptable speed simply by lifting with suction force. For the 2017 Challenge, our end-effector design focus and effort is on creating a simple and robust pneumatic grasping mechanism not only for lifting but also in being able to compensate for the potential error with respect to the suction point of the target item location.

The initial design of the end-effector has two grasping methods, suction and finger, and 3 DOF; one on the suction cup allowing the suction cup to approach an item from top down (0 degree) and from side (variable angle, controlled with a servo motor), and the other two DOF allows the finger to position itself in front of the suction when needed and open and closed the fingers for grasping. However, after conducting experiments, we have decided to remove all the DOF and the finger gripper. More details on the decision can found at the next sub-section.

The final end-effector consist of a body attached to the robot Tool Centre Point (TCP) and a modified suction cup. The suction is powered by a Karcher vacuum cleaner T8/1 (1600W), with a pressure sensor that feedback to control the airflow using two pneumatic solenoid valves. Figure 10 illustrates the modified suction cup from Piab bag lip suction cup. The airflow is maximized by increasing the overall inner diameter. The modified suction cup is both sewed together with cotton threads and glued with epoxy adhesives.

B. Grasping Strategy

Based on our experiments with the training set, the percentage of successful grasping strategy using the different end effectors are: 98.25% for suction and 1.75% for finger gripper. The only item that is very difficult to suctioned is the black mesh cup, although its bottom part could be suctioned.

When using the suction gripper alone, 89.75% of the training items could be approached straight down from the top. Items such as books, when positioned vertically along the inner wall of the shelf cannot be suctioned directly

Algorithm 1 OBJECT IDENTIFICATION AND SUCTION POSE ESTIMATION

INPUT: Fixed ROI image from camera (RGB) I_{ROI} , database reference images ${}_T I_1^{db}, {}_T I_2^{db} \dots {}_T I_N^{db}$, where N is the total number of reference images available for target item T

OUTPUT: 6D pose Vector, $\vec{P}(X, Y, Z, \alpha, \beta, \gamma)$ representing the suction point and suction normal

```
1: procedure APPLY SLIDING WINDOW ON  $I_{ROI}$ 
2:    $rows \leftarrow$  no of rows in  $I_{ROI}$ 
3:    $cols \leftarrow$  no of cols in  $I_{ROI}$ 
4:    $D \leftarrow$  Average diagonal size of  ${}_T I_1^{db} \dots {}_T I_N^{db}$ 
5:    $n \leftarrow 0$ , no of sub images
6:   for  $r_{idx} \leftarrow 1$  to  $rows - D$  do
7:     for  $c_{idx} \leftarrow 1$  to  $cols - D$  do
8:       sub roi image,  $i_n = Rect(r_{idx}, c_{idx}, r_{idx} + D, c_{idx} + D)$ 
9:        $c_{idx} = c_{idx} + (cols - D)/2/(cols/D - 1)$ 
10:       $n = n+1$ 
11:       $r_{idx} = r_{idx} + (rows - D)/2/(rows/D - 1)$ 
11: procedure FEATURE EXTRACTION AND MATCHING
12:   for  $ref\_img \leftarrow 1$ ,  $ref\_img = ref\_img + 1$  to N do
13:     for  $sub\_img \leftarrow 1$ ,  $sub\_img = sub\_img + 1$  to n do
14:       Detect ORB feature points, K1 in  $i_{sub\_img}$ 
15:       calculate the descriptors  $-D_1$  of K1
16:       Detect ORB feature points, K2 in  $i_{ref\_img}$ 
17:       calculate the descriptors  $-D_2$  of K2  ${}_T I_{ref\_img}^{db} - D_2$ 
18:       GMS matching between  $D_1$  and  $D_2$ 
19:   GMS Prediction:
       $avg\_dist =$  average match distance
       $S =$  no of matches
       $key\_points =$  match points
      } highest no of matches (3)

20: procedure CNN CLASSIFICATION
21:    $M \leftarrow$  trained model
22:   for  $sub\_img \leftarrow 1$  to n do
23:     CNN classification on image  $i_{sub\_img}$ 
24:      $P \leftarrow$  Probability of target item in image  $i_{sub\_img}$ 
25:   CNN prediction:  $Rect_{cnn} \leftarrow$  highest probable sub image
26: procedure TARGET ITEM DETECTION
27:   Generic ambiguity removal
28:   if  $S < 15$  then return false
29:   if  $S < 100$  and  $avg\_dist > 45$  then return false
30:   if  $S < 100$  and  $P_{highest} < 0.5$  then return false
31:   Specific ambiguity removal
32:    $\delta \leftarrow$  Prediction inconsistency.
33:    $\delta = centroid(Rect_{cnn}) - centroid(bounding\_rect(key\_points))$ 
34:   if ( $\delta > D$ ) then return false
35: procedure SUCTION POSE ESTIMATION
36:   Suction point:
37:   if ( $S > 100$ ) then return  $centroid(bounding\_rect(key\_points))$ 
38:   if ( $\delta < D$  & no generic ambiguity) then return  $centroid(bounding\_rect(key\_points))$ 
39:   if (generic ambiguity != NULL &  $P_{highest} > 0.8$ ) then return  $centroid(Rect_{cnn})$ 
40:   Normal vector  $\leftarrow$  surface normal at the suction point from PCL.
```

Fig. 11. Our robotic picker during the actual competition.

from the top; the gripper has to come from the side (by rotating the wrist of the robot arm) or a 1 DOF mechanism must let the suction cup tilt 90 deg as it approaches the item from the side. Based on these early experiment results, we decided to simply power-up the airflow to enhance the picking capability to 99%. Thus, we were able to meet our objective by using an industrial vacuum cleaner, a modified the suction cup that maximized airflow, and sensors that monitor and manipulate airflow.

VIII. DISCUSSION AND CONCLUSION

During the actual picking run at the Challenge, our robot system was able to pick all 10 target items correctly in around 8 minutes. Aside from earning all the positive points for picking the right items, we also earned a lot of points from the extra minutes left after completing the task. The target items were a DVD, Windex, stainless bowl, pink rag, deodorant, black notebook, flashlight, cable ties, 3M hooks, and black gloves. Only one box couldn't be closed without moving an item (the Windex bottle was protruding out of the box). In total, the team were able to get a score of 257. Fig. 11 shows our system as the robots pick the target items.

To summarize, we designed our robotic picker according to the requirements set by the Challenge organizers and more:

- **FAST** - We used two arm system to effectively double the system throughput
- **PERCEPTIVE** - We used a mixed-mode approach that combines traditional feature-based and artificial intelligence to deal with new items
- **SENSITIVE** - The use of machine learning enabled the system to recognize items that are featureless or easily affected by other camera parameters such as shadow, view, etc
- **QUICK LEARNER** - We used a transfer learning approach to retrain the CNN for the new items within the short period of time given

- **SMOOTH** - The computations have been distributed across several computers to ensure smooth and safe operation of the robot arm
- **EXPANSIVE** - The extendable shelf gives more workspace for the arms and helps in providing uncluttered view for the cameras
- **RESILIENT** - The software architecture and the state machine allows for easy inclusion of error recovery and interrupt strategies
- **SIMPLE** - The gripper system was downgraded from a 2 DOF with tongs to a simple suction gripper as it could actually pick up 99% of items based on our experiments
- **LIGHTWEIGHT** - The whole set-up conforms to the dimension and weight requirements of airlines so that the system could be dismantled and hand-carried if needed. This also informed our choice of robot arm (UR5) instead of the long reach but heavier UR10.
- **TIGHT** - The tables, camera frames, robot arms, and shelves are manually calibrated with respect to each other, which gives lessens position error as the arms move

In order to build a stable and robust system, the requirements set by the Challenge has to be analyzed and tested carefully. Moreover, a few trade-offs were necessary so that all efforts and resource could be focused on what really works and delivers.

REFERENCES

- [1] N. Correll, K.E. Bekris, D. Berenson, O. Brock, A. Causo, K. Hauser, K. Okada, A. Rodriguez, J.M. Romano and P.R. Wurman, Analysis and observations from the first amazon picking challenge, IEEE Transactions on Automation Science and Engineering, 2016, no. 99, pp. 1-17.
- [2] F.S.Hillier and G.J.Lieberman, "Introduction to Operations Research", Holden-Day, 1967.
- [3] GMS: Grid-based Motion Statistics for Fast, Ultra-robust Feature Correspondence. JiaWang Bian and Wen-Yan Lin and Yasuyuki Matsushita and Sai-Kit Yeung and Tan Dat Nguyen and Ming-Ming Cheng. IEEE Conference on Computer Vision and Pattern Recognition 2017.
- [4] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." In Advances in neural information processing systems, pp. 1097-1105. 2012.
- [5] Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv:1409.1556 (2014).
- [6] Pan, Sinno Jialin, and Qiang Yang. "A survey on transfer learning." IEEE Transactions on knowledge and data engineering 22, no. 10 (2010): 1345-1359.
- [7] Szegedy, Christian, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. "Rethinking the inception architecture for computer vision." In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2818-2826. 2016.
- [8] Abadi, Martn, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado et al. "Tensorflow: Large-scale machine learning on heterogeneous distributed systems." arXiv preprint arXiv:1603.04467 (2016).
- [9] Szegedy, Christian, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. "Going deeper with convolutions." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1-9. 2015.
- [10] H. Zhu and Y. Y. Kok and A. Causo and K. J. Chee and Y. Zou and S. O. K. Al-Jufry and C. Liang and I. M. Chen and C. C. Cheah and K. H. Low., "Strategy-based robotic item picking from shelves," 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Daejeon, 2016, pp. 2263-2270.