Aalborg Universitet

A Collaborative Robot Cell for Random Bin-picking based on Deep Learning Policies and a Multi-gripper Switching Strategy

Olesen, Albert Sonne; Gergaly, Benedek Benjamin; Ryberg, Emil Albin; Thomsen, Mads Riis; Chrysostomou, Dimitrios

Published in:
Procedia Manufacturing

DOI (link to publication from Publisher):
10.1016/j.promfg.2020.10.002

Publication date:
2020

Document Version
Publisher's PDF, also known as Version of record

Citation for published version (APA):
Olesen, A. S., Gergaly, B. B., Ryberg, E. A., Thomsen, M. R., & Chrysostomou, D. (2020). A Collaborative Robot Cell for Random Bin-picking based on Deep Learning Policies and a Multi-gripper Switching Strategy. *Procedia Manufacturing*, *51*, 3-10. https://doi.org/10.1016/j.promfg.2020.10.002

Available online at www.sciencedirect.com

ScienceDirect

Procedia Manufacturing 51 (2020) 3–10

Procedia MANUFACTURING

www.elsevier.com/locate/procedia

30th International Conference on Flexible Automation and Intelligent Manufacturing (FAIM2021)
15-18 June 2021, Athens, Greece.

A Collaborative Robot Cell for Random Bin-picking based on Deep Learning Policies and a Multi-gripper Switching Strategy

Albert S. Olesen[a], Benedek B. Gergaly[a], Emil A. Ryberg[a], Mads R. Thomsen[a], Dimitrios Chrysostomou[b]

[a]Dept. of Materials and Production, Aalborg University, Fibigerstraede 16, Aalborg East, DK-9220, Denmark
[b]Robotics & Automation Group, Dept. of Materials and Production, Aalborg University, Fibigerstraede 16, Aalborg East, DK-9220, Denmark

Abstract

This paper presents the details of a collaborative robot cell assembled with off-the-shelf components designed for random bin-picking and robotic assembly applications. The proposed work investigates the benefits of combining an advanced RGB-D vision system and deep learning policies with a collaborative robot for the assembly of a mobile phone. An optimised version of YOLO is used to detect the arbitrarily placed components of the mobile phone on the working space. In order to overcome the challenges of grasping the various components of the mobile phone, a multi-gripper switching strategy is implemented using suction and multiple fingertips. Finally, the preliminary experiments performed with the proposed robot cell demonstrate that the increased learning capabilities of the robot achieve high performance in identifying the respective components of the mobile phone, grasping them accurately and performing the final assembly successfully.

Keywords: Random Bin-picking; Deep Learning; Collaborative robot; Multi-gripper; Industry 4.0

1. Introduction

Since the introduction of robots in the manufacturing domain and the evolution of automated production systems there has been a desire to fully automate and introduce intelligence to the majority of manufacturing procedures. The fourth industrial revolution has further increased this desire for intelligence and learning capabilities in collaborative robots. A major proof of intelligence in collaborative robots and one of the hardest problems to solve in automated production is the accurate detection and robust manipulation of random industrial parts. Especially in cases where the parts are diverse and multiple grasping techniques are required the problem of bin-picking becomes even more challenging to tackle. In this paper, a collaborative robot cell is built, based on off-the-shelf components, to perform random table-picking and assembly of dummy mobile phones directly from a box full of the components. The final scope of this system is to demonstrate how an intelligent bin-picking system can be designed and deployed in a collaborative robot cell based on deep learning policies to identify the components and a multi-gripper design to accommodate the grasping of the diverse parts of the phone. This project is based around the FESTO Cyber-Physical (CP) Factory [1] and its mockup mobile phones which are illustrated in Figures 1 and 2. The FESTO CP Factory is an educational production-line setup which is used to demonstrate various manufacturing processes and encapsulate an Industry 4.0 environment. It consists of several different modules which can be re-arranged as desired. In this case the FESTO CP Factory is set up as seen in Figure 1 and is responsible for manufacturing mockup mobile phones [3]. The facility receives an order from a Manufacturing Execution System (MES) and the personalised customer order in the system proceeds with picking a blue, black or white bottom cover depending on given instructions and places it onto a pallet. This pallet is then moved through the production line either on a conveyor belt or with the means of a mobile robot [4] where different assembly operations are carried out. In a confined robot cell containing a KUKA KR 6 R700

* Corresponding author. Tel.: +45 21 18 19 18
E-mail address: dimi@mp.aau.dk (Dimitrios Chrysostomou).

2351-9789 © 2020 The Authors. Published by Elsevier Ltd.

10.1016/j.promfg.2020.10.002

Fig. 1: FESTO Cyber-physical Factory.

Fig. 2: The parts of the mock up mobile phone. [2]

sixx industrial manipulator, a PCB, and two fuses are added onto the back cover. Later in the production line, a black top cover is added and snapped together with the bottom cover to complete the assembly. The full assembly process is illustrated in Figure 4.

While the FESTO CP Factory can produce the mockup mobile phones in a structured matter, errors can still happen. A system for disassembly of these phones has been previously built [2], to identify these errors and any missing components. The proposed system in this paper, is built as the next step in an Industry 4.0 production environment built for circular economy and recycling purposes. At first, the FESTO CP Factory executes the personalised order from the customer and delivers a fully assemble phone. In case of a report from a customer that there is an error on the phone, the dual-arm robot [2] can disassemble the phone and deliver the parts in an unstructured way inside a bin. Finally the current system can refurbish the phone, assemble it again and introduce it back to the production by picking the parts which are randomly placed on a table.

2. Related research

There has been a great amount of research and experimentation done in the field of bin-picking of industrial parts [5]. Systems such as the one developed by Tuan-Tang Le and Chyi-Yeu Lin [6] achieve more than 99% success-rate, by utilising semantic segmentation and depth data around the segmented objects to compute 3D poses. However, they only use suction grasping which limits the system's capabilities in terms of interacting with different shapes and sizes.

A different, more versatile system is the one developed for the 2017 Amazon Robotics Challenge by Andy Zeng et al. [7]. This system uses a multi-gripper which allows for more dexterous grasping capabilities. Furthermore, it is a quite fast system having a processing speed of 0.05 x n seconds, where n is the number of possible grasp angles. The speed of the system is, however, penalised by the lower accuracy of approximately 89%. The system uses deep learning to find grasp-success probability maps for 4 different grasping options (suction down, suction side, grasp down, flush grasp).

Other systems also suggested using CAD-based pose-estimation such as the ones developed by Kai-Tai Song et al. [8] and Yu-Kai Chen et al. [9] who achieved an error of 1.06 mm (X), 1.48 mm (Y), 0.72 mm (Z), 4.58° (Rotation) and 0.998 mm (X), 1.036 mm (Y), 0.912 mm (Z), 2.126° (Rotation) respectively. D. Morrison et al. [10] proposed a multi-gripper system which achieved a grasp success rate of 72% using point-cloud computed surface-normals for suction and grasped parallel to the principal-axis of the RGB-segmented objects. Their system required 7 training images for a novel object.

The proposed prototype has a few distinct differences compared to other related research. One significant aspect is the dynamic camera calibration using fiducial ArUco markers, which is not common in Bin Picking applications where the robots are usually manually calibrated to their working stations. Paired with no requirement for a depth camera, this enables a cheap and flexible camera setup, which is fast to setup, lightweight in terms of complexity and computational power and can be moved as the user desires - as long as all four ArUco markers remain inside the camera's field of view. At the same time, this prototype can be trained quickly to handle multiple industrial parts at once with the multi-gripper design and it demonstrates how an effective low-cost alternative can be used and implemented in today's factories.

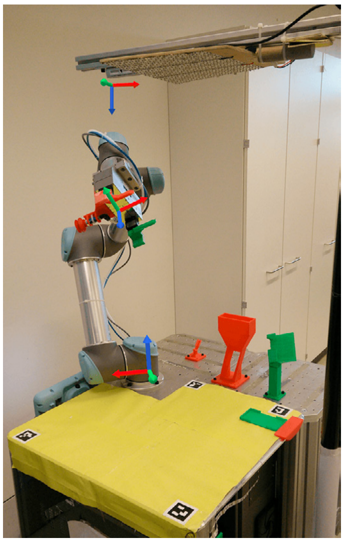


Fig. 3: Setup of the collaborative robot cell with coordinate frames attached to the robot, camera and gripper.

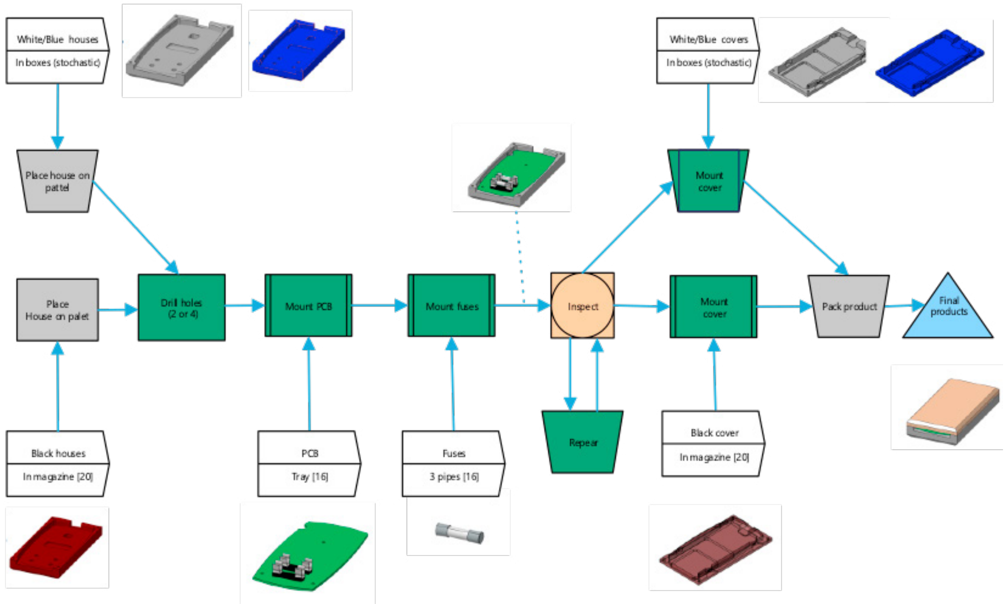


Fig. 4: Assembly flow of the mockup phone assembly process.[2]

3. System overview

The proposed collaborative robot cell consists of two tables with a UR5 robotic manipulator attached on top of one, a Schunk WSG 50-110 electronic gripper, custom alignment and assembly fixtures and an Intel RealSense D415 camera mounted above the main worktable, as can be seen in Figure 3. There is also a suction tool mounted on the side of the gripper. The main worktable is covered in yellow paper to provide a flat, reflection-free surface that has good contrast against all the parts' colours. The four ArUco markers printed on the four corners of the work table are used for camera calibration purposes. The camera pole also houses a LED panel for even lighting. A Hokuyo URG-04LX-UG01 laser scanner is mounted on the corner of the main table to to increase safety and detect if people are getting close to the cell. Finally, coloured LEDs are placed around the table to signal the status of the system when humans approach the cell or are moving away.

The prototype uses deep learning algorithms to locate the objects in 3D space in order to enable the handling of the objects using the UR5 manipulator. The individual objects are detected using the YOLOv3 object detection network, more specifically the YOLOv3-tiny version, due to the available processing power not being powerful enough for the full YOLOv3 architecture [11]. Furthermore, a second CNN is later used in the pipeline for detecting the orientation of the parts relative to the camera.

The extrinsic camera calibration are done using fiducial ArUco markers placed on the worktable [12]. This enables the camera mounting to be more agnostic to its placement relative to the worktable and the robot.

The different parts have different shape, size and material, as can be seen in Figure 2, which means that they cannot be picked up using the same end-effector attachment. To solve this problem a multi-purpose gripper and multiple fixtures for aligning the parts are designed and used in the proposed robot cell.

The system was programmed in Python, with the main libraries used being: urx for sending direct commands to the UR5 robot, OpenCV for camera calibration and a YOLO wrapper to interface with the natively running YOLO implementation.

The codebase was deployed on a Thinkpad P50 laptop (Intel i7-6820HQ, Nvidia Quadro M1000M, 16 GB RAM). Both the YOLO network and the custom-made orientation detection CNN run both on the CPU due to size. This resulted in an approximate inference time of 5 and 2 seconds respectively.

The robot and the gripper were connected to the laptop via Ethernet, the camera was connected via USB 3.0 and the Hokuyo laser scanner via serial over USB. The coloured LEDs were controlled by an Arduino Uno which received commands via serial over USB.

3.1. Object detection

To ensure that the object detection subsystem will provide accurate and robust detection rate, multiple images of all parts were acquired. Each image contained one of each of the covers, one PCB and two fuses, totaling to 7 objects appearing in each training image. An example such training image prior to labelling is depicted in Figure 5. All possible parts were placed in multiple, arbitrary locations on the worktable and in all possible orientations. It was also important to flip the parts and gather equally as many images with flipped parts as non-flipped parts. The images were then labelled using the OpenLabeling software [13]. For each image labelled using OpenLabeling, the software outputs a text file containing the coordinates of the bounding box and the object classification contained in that bounding box. This is then used by the DarkNet framework.

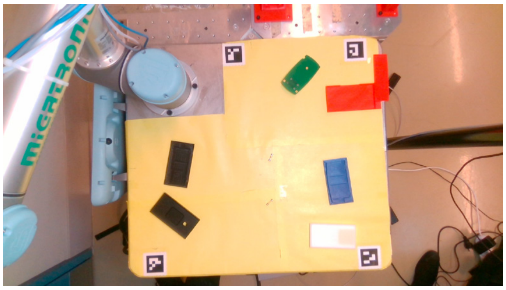


Fig. 5: An example of a training image, containing the back cover, PCB, two fuses and the 3 colored front covers.

The object detection neural network currently consists of a dataset with approximately 1000 images with a total of roughly 7000 labels. These images were captured over a period of several months, to acquire images with different lighting conditions due to season change and different interior lighting. This way the system is built in such way that can work in as many environments as possible without overfitting e.g. due to specific lighting conditions. The neural network was then trained on the Darknet framework using the pre-existing YOLOv3 weights for transfer learning, to reduce training time and reduce overfitting since the dataset is small relative to the network's trainable parameters. An output of the YOLOv3 can be seen in Figure 6. The network

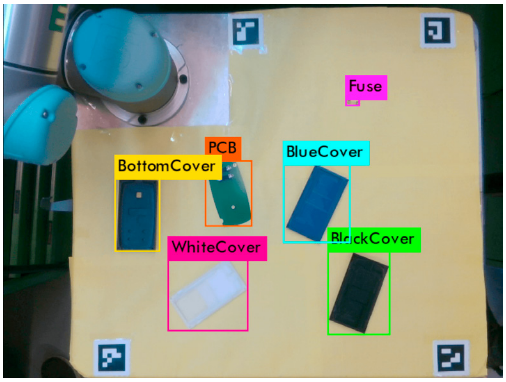


Fig. 6: An example of the output labels from the YOLOv3 network on an image from the test set.

was evaluated using the intersection over union (IoU) and mean average precision (mAP) metrics, which are common metrics for evaluating an object detection neural network [14]. The equation for computing the IoU is:

$$IoU = \frac{area(B_p \cap B_{gt})}{area(B_p \cup B_{gt})} \tag{1}$$

where B_p denotes the predicted bounding box, and $B_g t$ denotes the groundtruth bounding box. The mAP can be computed by calculating the average precision of all the classes, and then computing the mean of all these averages. The average precision can be computed by calculating the area under the precision-recall curve. This can be done by interpolating for the area under the curve.

$$AP = \sum_{r=0}^{1} (r_{n+1} - r_n)\rho_{interp}(r_{n+1}) \tag{2}$$

with

$$\rho_{interp}(r_{n+1}) = \max_{\tilde{r}:\tilde{r} \geq r_{n+1}} \rho(\tilde{r}) \tag{3}$$

where $\rho(\tilde{r})$ is the measured precision at recall $\tilde{r}$. These metrics are automatically calculated by the Darknet framework. After training the custom YOLOv3 network it performed well with an average IoU score of 85.4% and a mAP score of 98.29%. An example of IoU on the test set is depicted in Figure 7.

Fig. 7: An example of the IoU score of the PCB in one of the test images. The prediction is illustrated with a purple rectangle while the ground truth with a brown one.

3.2. *Detecting final orientation of parts*

After the object detection step the label of a part and whether the part is flipped or not (excluding fuses) is known, which means that the robot is able to flip and align the part the correct way using the alignment fixtures. To place the part in the assembly fixture correctly, the orientation of the part has to be determined. The possible configurations are with an offset of 180° from each other. The orientation is measured around the normal to the main plane of the part. To determine this orientation, a simple Convolutional Neural Network (CNN) based on the VGG-16 network was trained on a custom dataset for the mockup phones to predict whether a part is facing left or right as viewed from the camera. The network was trained on 2647 images and achieved an accuracy of 99.99% on the test-set.

3.3. Alignment fixtures

The alignment fixtures serve two purposes: To align the grasped parts of the phone in such way that they will have a fixed location in relation to the robot, and to flip the parts if it is needed. The front-cover and back-cover fixture is slightly tilted and slanted fixture that allows the part to slide into the same spot for correct alignment. The PCB fixture takes on a funnel design, where it will slide down to the correct position independent of the orientation. The fuse fixture is designed to compliment the cylinder form of the object. Models of the fixtures are illustrated in Figure 8.

3.4. Multi-gripper

A multi-gripper strategy has been implemented in order to enable more robust ways to grasp objects. It consists of 3 different methods of grasping: a large grasp for the phone covers, small and narrow grasp for the fuses and suction grasp for the PCB board. These grippers are oriented with a 90° angle offset to each other. The benefits of using a multi-gripper instead of a tool-changer is faster switching of tools and no need for taking up extra space for a tool-changer, reducing clutter in the workspace. Naturally, a sophisticated end-effector such as the one proposed also results to more singularities that had to be taken into consideration for the optimal path planning of the robot to achieve the fastest assembly procedure.

The gripper was designed through multiple iterations, since initially a different actuator for the fingers were used, the pneumatic SMC MHZ2-20D, which did not have an adequate actuation displacement, and thus it was decided to use the Schunk WSG-50-110-B electrical gripper instead. The final multi-gripper was designed for the Schunk gripper to be mounted on the wrist of the UR5. Fingers for the covers and the fuses were designed to be mounted on the actuating plates on the Schunk gripper as well. The mounting for the suction was designed separately to be mounted to the side of the Schunk gripper, on the mounting plates used between the Schunk gripper and the UR5 wrist. The gripper and suction can be seen in Figure 8d.

3.5. Programmed logic

The program can be divided into 4 phases: detection of parts, grasping of the parts, alignment of the parts and correct placement of the parts. The detection phase refers to the acquisition of an image of the worktable, and processing the image through the trained YOLO network. YOLO outputs the width and height of the bounding box which is then used to find the x and y coordinates of the centre of the object. The grasping phase refers to the movement of the robot arm above the x and y coordinates of the object it has to grasp. This is done by converting the image coordinates to world coordinates using ArUco markers. The markers are used to solve the Perspective-n-Point (PnP) problem to calculate world coordinates from image coordinates using the pin-hole camera model. The simplified pin-hole camera model is used, since $z \neq 0$ in this application:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = R \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + t \tag{4}$$

Where x, y, z is the camera's principal point, R the rotation matrix, X, Y and Z the location of the point in the world coordinate system and t is the translation vector for the camera.

Additionally:

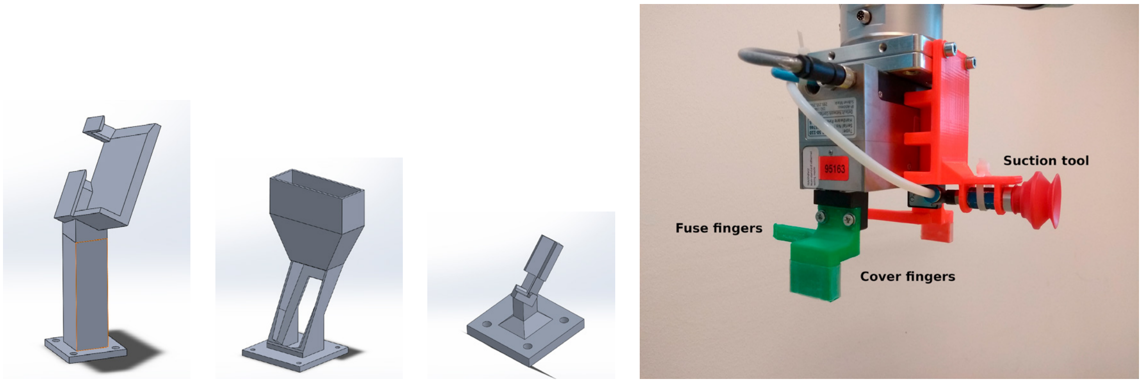
$$\begin{aligned} x' &= \frac{x}{z} \\ y' &= \frac{y}{z} \\ u &= f_x * x' + c_x \\ v &= f_y * y' + c_y \end{aligned} \tag{5}$$

Here u and v is the image coordinates and f_x, f_y, c_x and c_y are intrinsic camera parameters.

The intrinsic camera parameters are already known from the internal calibration specifications of the Intel Realsense camera. The rotation matrix, R, and the translation vector t are computed using OpenCV which utilises the ArUco markers to solve the PnP problem. Knowing these variables the world coordinates can be computed from images coordinates using equations 4 and 5.

The robot can grasp in two orientations: horizontal and vertical. These orientations are decided by comparing the width to the height from the bounding boxes, where it will grasp the largest side. The robot will then enter the alignment phase, where it will place the grasped object in the respective alignment fixtures. In the placement phase, the robot will move the aligned object up to the camera for a final inspection of orientation, where the camera will capture an image to determine if the grasped object is rotated by 180° or not. The exact orientation and position of the grasped object is then known and it can be placed in the assembled unit. A flowchart demonstrating the presented steps of the overall assembly procedure is shown in Figure 9.

Finally, a supplementary safety feature was implemented by mounting a laser scanner on one of the corner of the worktable. A safety module was then implemented which continuously checked the laser scanner data for movement. In case an object or a human worker approached the system within the one meter threshold, the system would slow down to the EU accepted TCP speed of 250 mm/s. Furthermore, if an object or a human worker approached the system within the 0.5 meter threshold the system would slow down to 0.5% of its full speed. A complete stop is effectively the same as 0.5%, however, in order to allow the system to run smoothly when people are entering and exiting the zone continuously, a complete safety stop of the system was undesired. In addition to the adjustments of the speed of the robot according to the laser scanner readings, LED strips were



(a) Cover alignment fixture (b) PCB alignment fixture (c) Fuse alignment fixture (d) The multi-gripper design equipped with small fingers for grasping fuses, larger fingers for grasping covers and a suction tool for handling the PCBs

Fig. 8: (a-c) The 3 alignment fixtures and (d) the multi-gripper design.

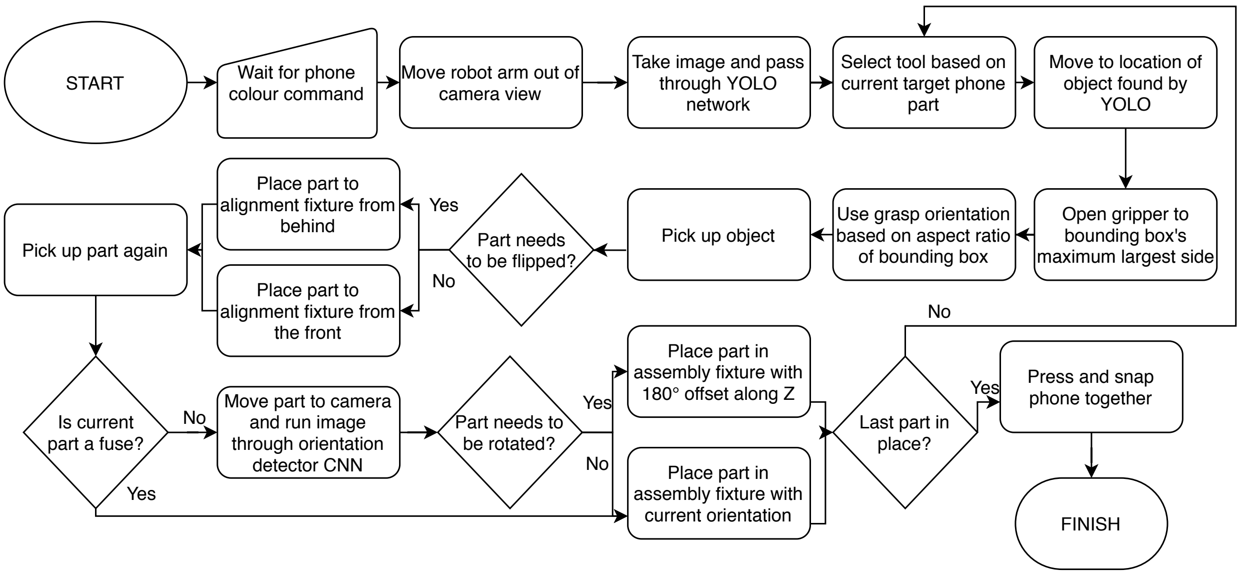


Fig. 9: Flowchart of the implemented procedure

also added to the system to provide indications of what mode the system was operating in:

- Green indicates that no humans are detected and the system is running full speed.
- Yellow indicates that one or more humans/obstacles have been detected within the 0.5 meter and 1 meter range, and the system is running with 250 mm/s.
- Red indicates that one ore more humans/obstacles have been detected within the 0 and 0.5 meter range, and the system is running at 0.5% speed.

4. Testing results

The prototype was tested through six different tests, five of which focused on separate aspects of the system and a final test assessing the overall performance.

The first test measured the performance of the object detection module, namely if it can predict the correct object with a correct bounding box. The test was performed 20 times with an overall success rate of 89.17%. More detailed results can be seen in Table 1. The second test focused on the system's ability to pick things up from the table after recognising them. Out of the 20 trials for each part, the system was 100% successful. As all five of sub-tests, this test was also isolated, meaning it does not rely on the previous steps to be successful. This explains

	Back cover	PCB	Fuse	Front cover			All parts
				Black	Blue	White	
Correct prediction	70%	95%	100%	90%	100%	90%	90.83%
Correct bounding box	100%	100%	90%	100%	100%	100%	98.33%
Everything successful	70%	95%	90%	90%	100%	90%	89.17%

Table 1: Summarised results from object detection performance test. The test was carried out 20 times and the percentages displayed in the table correspond to the average success rate for that part. The columns represent the different phone parts and the rows show the various aspects that were tested.

the high performance score of 96% in this test, even though the object detection test only reached 89.17%.

The third test measured the ability of the system to align the parts using the alignment fixtures after they have been picked up. To align the covers correctly and the PCB the object orientation must also be known, so this test also measures the performance of the CNN responsible for detecting the orientation of the parts. Out of 10 runs the system achieved an overall success rate of 96%. More detailed results can be seen in Table 2. The fourth

	B. cover	PCB	Fuse	Front cover			All parts
				Black	Blue	White	
Orientation recognition	100%	80%	N/A	100%	100%	100%	96%
Orientation alignment	100%	80%	N/A	100%	100%	100%	96%
Placement	100%	100%	100%	100%	100%	100%	100%
Order	100%	100%	100%	100%	100%	100%	100%
Everything successful	100%	80%	100%	100%	100%	100%	96%

Table 2: Summarised results from alignment test. The test was carried out 10 times and the percentages displayed in the table correspond to the average success rate for that part. The columns represent the different phone parts and the rows show the different aspects that were tested. Orientation recognition means that the CNN did output the correct orientation, orientation alignment means that the system was able to align the phone part to the correct orientation, placement means that the system placed the phone part correctly in the alignment fixture and finally, order means that the system aligned the parts in the correct order.

test focused on the performance of the system in the assembly of the phone after the parts have been aligned. Out of 5 runs, the only error was one of the fuses being misplaced by 1-2mm, which pushed the overall performance down to 95%.

The safety of the system was tested as well. More specifically, the test involved the evaluation of the capability of the robot to slow down and then stop as people or objects get closer to the laser scanner mounted on the robot platform. Out of 5 runs the system slowed down and stopped at the respective threshold distances 100% of the time.

The final full system integration test, examined the overall performance of the prototype. The test was performed 30 times, 10 times each for the different front cover colours. The results of the full integration can be seen in Table 3. It is important to note that the data was recorded in such a way, that if a step failed during the process, the run was stopped and the next steps in the assembly procedure were considered a failure. This delivers an accurate number of the overall performance, but skews the performance of the individual aspects in favour of the earlier steps in the pipeline.

	B. cover	PCB	Fuse	Front cover			All parts
				Black	Blue	White	
Correct classification	93%	100%	90%	100%	100%	90%	96%
Correct bounding box	100%	100%	83%	100%	100%	90%	96%
Successful pickup	97%	97%	88%	70%	100%	90%	90%
Successful flip	90%	80%	N/A	70%	100%	90%	85%
Successful orientation	97%	80%	N/A	70%	100%	90%	87%
Correct assembly	97%	70%	73%	50%	100%	80%	78%
Everything successful	90%	63.3%	70%	55.6%	100%	72.7%	47%

Table 3: Summarised results of from the final tests. The table illustrates the averaged results from the 30 tests done. Correct classification means the system correctly classified the part as well as if it were flipped or not. Correct bounding box means if the bounding box was accurate enough. Successful pick describes if the part was successfully picked up. Successful flip is whether the part was flipped accordingly. Successful orientation is whether the system recognised the correct orientation. Correct assembly placement is whether the part was placed correctly in the assembly fixture.

5. Discussion

While the success rate of the final assembly is not ideal, the initial approach revealed multiple sources of potential errors in the system. Some of the encountered errors include, among others, the PCB getting stuck in the alignment funnel, the fixtures getting misaligned due to vibrations and the suction not turning off (likely due to a lost network packet). Naturally, encountered errors were also stemming from object detection, but as the first test indicates, that part of the prototype performs at a reasonable level and is not a major source for the errors. The aforementioned small errors had a more severe impact on the full system integration test, since the prototype has to complete about 20 discrete steps in succession, so there is a higher chance of encountering one of these placement errors.The system also currently lacks error handling capabilities, meaning even small, detectable errors cause the whole test to fail.

In terms of implementing the proposed system in a real manufacturing setting, there are additional limitations. Initially, a company would have to gather training data on their own parts and train the YOLO-network and the orientation CNN on those parts. Furthermore, in more low-level aspects of the system it will necessary to manufacture tailored assembly and alignment fixtures for the specific parts. Additionally, the working surface of the robot should be enhanced with ArUco markers and calibrate them accordingly. Finally, the proposed approach handles the assembly of mock-up phones without performing actual manufacturing processes. In the production of a real product, these manufacturing processes would have to be added, raising the complexity of the system and subsequently, a more advanced assembly procedure would have to be performed.

6. Future work

The work of this project lays the foundation of a future project where 3D information will be included in the process pipeline, in order to do layered bin-picking, as opposed to table-picking which is examined in this paper. To accomplish this, pose estimation in 3D space will need to be incorporated, which might be done using semantic segmentation instead of object detection, while also incorporate the depth information in the learning algorithm or as a post-processing step before the final pose is estimated.

Currently the prototype only picks up the objects by orienting the gripper in 90° offset relative to the XY-plane in the world coordinate system (the worktable plane). This works well in most cases since the objects are placed flat on the worktable. Future improvements will also include an implementation of error handling techniques, making the setup resistant to vibrations and the inclusion of more training data for the YOLOv3 network. To enable 3D bin-picking with objects that can have random orientation in 3D space, a new method for calculating a reliable grasp will need to be implemented. A way to do this could be to train a learning algorithm to predict the best grasps, for example similar to Dex-Net, after the pose of the object has been estimated [15].

Finally the current safety measures using the laser scanner does not cover the full work area and does not have enough redundancy, which will also be improved in the next project to further develop the safety of the system.

7. Conclusion

This paper presents a Table-Picking solution able to pick up mockup phone parts and assemble them into a mockup phone using an UR5 collaborative robot manipulator. The system uses a version of YOLOv3 in order to detect and locate the different parts and another CNN to fine tune the orientation of the part in order to assemble the phone. The paper presents a design for a multi-purpose gripper which enables picking up larger objects, such as phone covers, smaller objects, such as fuses and non-porous flat surfaces using suction. This enables the robot to quickly change tool, by just reorienting the end-effector in order to pick up the desired object.

The prototype presented in this paper achieves a high success rate in the individual subsystems but a number of failed assembly attempts reduced the overall success rate of the system to 47% for the performed tests. While this number seems very low, it is caused by the fact that even small errors on one part in the assembly pipeline, will most likely not enable the rest of the phone to be assembled. Since not all parts had equal amount of performance, the performance of the system as a whole was determined by the lowest denominator. As described in the discussion improvements for better success rate for the individual parts can be improved to reach a potential of 99+% success rate.

The prototype successfully demonstrates how the different components can be integrated in order to perform a table-picking task, even though the overall success rate is worse than desired. Using this prototype as the foundation, an addition of the proposed improvements from Section 6 and with better error checking and handling a robust bin-picking solution can be developed.

References

[1] Festo cyper-physical (cp) factory modules (accessed 22-01-2020).
URL www.festo-didactic.com/int-en/learning-systems/learning-factories,cim-fms-systems/cp-factory/

[2] J. F. Buhl, R. Grønhøj, J. K. Jørgensen, G. Mateus, D. Pinto, J. K. Sørensen, S. Bøgh, D. Chrysostomou, A dual-arm collaborative robot system for the smart factories of the future, Procedia Manufacturing 38 (2019) 333–340 (2019).
URL https://doi.org/10.1016/j.promfg.2020.01.043

[3] S. T. Mortensen, D. Chrysostomou, O. Madsen, A novel framework for virtual recommissioning in reconfigurable manufacturing systems, in: 2017 22nd IEEE Int. Conf. on Emerging Technologies and Factory Automation (ETFA), IEEE, 2017, pp. 1–4 (2017).
URL https://doi.org/10.1109/ETFA.2017.8247744

[4] R. E. Andersen, E. B. Hansen, D. Cerny, S. Madsen, B. Pulendralingam, S. Bøgh, D. Chrysostomou, Integration of a skill-based collaborative mobile robot in a smart cyber-physical environment, Procedia Manufacturing 11 (2017) 114–123 (2017).
URL https://doi.org/10.1016/j.promfg.2017.07.209

[5] M. Alonso, A. Izaguirre, M. Graña, Current research trends in robot grasping and bin picking, in: The 13th Int. Conf. on Soft Computing Models in Industrial and Environmental Applications, Springer, 2018, pp. 367–376 (2018).
URL https://doi.org/10.1007/978-3-319-94120-2_35

[6] T.-T. Le, C.-Y. Lin, Bin-picking for planar objects based on a deep learning network: A case study of usb packs, Sensors 19 (16) (2019) 3602 (2019).
URL https://doi.org/10.3390/s19163602

[7] A. Zeng, S. Song, K.-T. Yu, E. Donlon, F. R. Hogan, M. Bauza, D. Ma, O. Taylor, M. Liu, E. Romo, et al., Robotic pick-and-place of novel objects in clutter with multi-affordance grasping and cross-domain image matching, in: 2018 IEEE Int. Conf. on Robotics and Automation (ICRA), IEEE, 2018, pp. 1–8 (2018).
URL https://doi.org/10.1109/ICRA.2018.8461044

[8] K.-T. Song, C.-H. Wu, S.-Y. Jiang, Cad-based pose estimation design for random bin picking using a rgb-d camera, Journal of Intelligent & Robotic Systems 87 (3-4) (2017) 455–470 (2017).
URL https://doi.org/10.1007/s10846-017-0501-1

[9] Y.-K. Chen, G.-J. Sun, H.-Y. Lin, S.-L. Chen, Random bin picking with multi-view image acquisition and cad-based pose estimation, in: 2018 IEEE Int. Conf. on Systems, Man, and Cybernetics (SMC), IEEE, 2018, pp. 2218–2223 (2018).
URL https://doi.org/10.1109/SMC.2018.00381

[10] D. Morrison, A. W. Tow, M. Mctaggart, R. Smith, N. Kelly-Boxall, S. Wade-Mccue, J. Erskine, R. Grinover, A. Gurman, T. Hunn, et al., Cartman: The low-cost cartesian manipulator that won the amazon robotics challenge, in: 2018 IEEE Int. Conf. on Robotics and Automation (ICRA), IEEE, 2018, pp. 7757–7764 (2018).
URL https://doi.org/10.1109/ICRA.2018.8463191

[11] J. Redmon, A. Farhadi, Yolov3: An incremental improvement, arXiv preprint arXiv:1804.02767 (2018).

[12] F. J. Romero-Ramirez, R. Muñoz-Salinas, R. Medina-Carnicer, Speeded up detection of squared fiducial markers, Image and vision Computing 76 (2018) 38–47 (2018).
URL https://doi.org/10.1016/j.imavis.2018.05.004

[13] J. Cartucho, Openlabelling (accessed 16-01-2020).
URL https://github.com/Cartucho/OpenLabeling

[14] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, A. Zisserman, The pascal visual object classes (voc) challenge, Int. Journal of computer vision 88 (2) (2010) 303–338 (2010).
URL https://doi.org/10.1007/s11263-009-0275-4

[15] J. Mahler, F. T. Pokorny, B. Hou, M. Roderick, M. Laskey, M. Aubry, K. Kohlhoff, T. Kröger, J. Kuffner, K. Goldberg, Dex-net 1.0: A cloud-based network of 3d objects for robust grasp planning using a multi-armed bandit model with correlated rewards, in: 2016 IEEE Int. Conf. on Robotics and Automation (ICRA), IEEE, 2016, pp. 1957–1964 (2016).
URL https://doi.org/10.1109/ICRA.2016.7487342