**FINAL PROJECT**

# ENPM-809B : Building a Manufacturing Robot Software System


**Submitted By: Group 1**

1. Abhiram Dapke(116237024)
2. Smriti Gupta(116748612)
3. Niket Shah (116345156)
4. Prasanna Marudhu Balasubramanian (116197700)
5. Piyushkumar Bhuva(116327473)

**Submitted To:**

PROF. CRAIG SCHLENOFF
PROF. ZEID KOOTBALLY

May 6, 2020

**SPRING 2020**

## REPORT CONTENT:

## I. INTRODUCTION

### Objective:

The objective of this project was to develop an agile industrial robotic system that simulates day-to-day factory manufacturing operations. Specifically, our goal was to build a whole kit in ariac workspace that should be able to perform the following operations:

- Pick and place a part from the conveyor and bins into an AGV tray.
- Check for part availability.
- Detect a faulty part.
- Flip a part. (Pose correction)
- Quality checking.
- In Process Order Update
- Sensor Blackout handling

Robotic Operating System and Gazebo ROS is a flexible collaborative framework for robotic software development, and consists of a large collection of tools, libraries and templates. Generally speaking, creating a robust and general-purpose robotic application from scratch is challenging. The ROS community provides a platform where experts all over the world can collaborate together. Using ROS as the programming platform, research institutes, laboratories and individuals can contribute their algorithms - also known as ROS packages - to the community and build software rapidly by leveraging existing modules.

The main goal of this project was to learn kit building using ROS packages and fulfill order as per instructions from the organizers. For this, of course we have restrictions on the orientation and priority que for parts. And with the use of Robotics arms (UR10) we have to pick part (Gasket, Piston and Rod, Gear) either from the trays residing near the arms or conveyer belt which is moving as the competition starts. The architecture adapted and the procedure we followed to pick and place the parts, detect faulty products, Quality checking, In process order update are explained in detail below in this report.

## II. ARCHITECTURE:

We have seen multiple architecture in the class and among them we have chosen hybrid architecture for certain reasons. Hybrid architecture gives us flexibility for the system to navigate between 3 pillars of the system(Sense, Plan,Act). Unlike reactive paradigm, we can plan in between sense and act steps. And like reactive, we can also use reactive approach for certain tasks which do not need preplanning. It is also important to mention that hierarchical approach may not work perfectly for our case where we don't want to automate all the tasks in hand. Hence a constant coordination between sense, plan and act is admired for projects like this and that is why we have chosen hybrid architecture.

We have differentiated the whole architecture in 3 levels naming upper, intermediate and lower level respectively also called level 2,level 1 and level 0. Level 1 is called intermediate level because of which other 2 levels can interact with each other. Here is a brief explanation of every level.

**Level 2:**

As explained earlier, this level is also known as the upper level of architecture and process/competition starts and exits in this level only. Competition starts and conveyer belt starts reading and after looping through the whole architecture this level confirms whether the order is fulfilled or not. Because it only maintains status of the top level is called upper level architecture.

**Level 1:**

This level is the only level which interacts with both the layers of the architectures. This is also known as dissociative level because we dissociate order and check whether requirements for each part are meeting or not. Besides this it is checked whether part is faulty or not and provided provision if faulty part is detected. The small chunk of information is fed to the lowermost level in the architecture and if the kit is completed then we pass this information to the upper level in architecture.

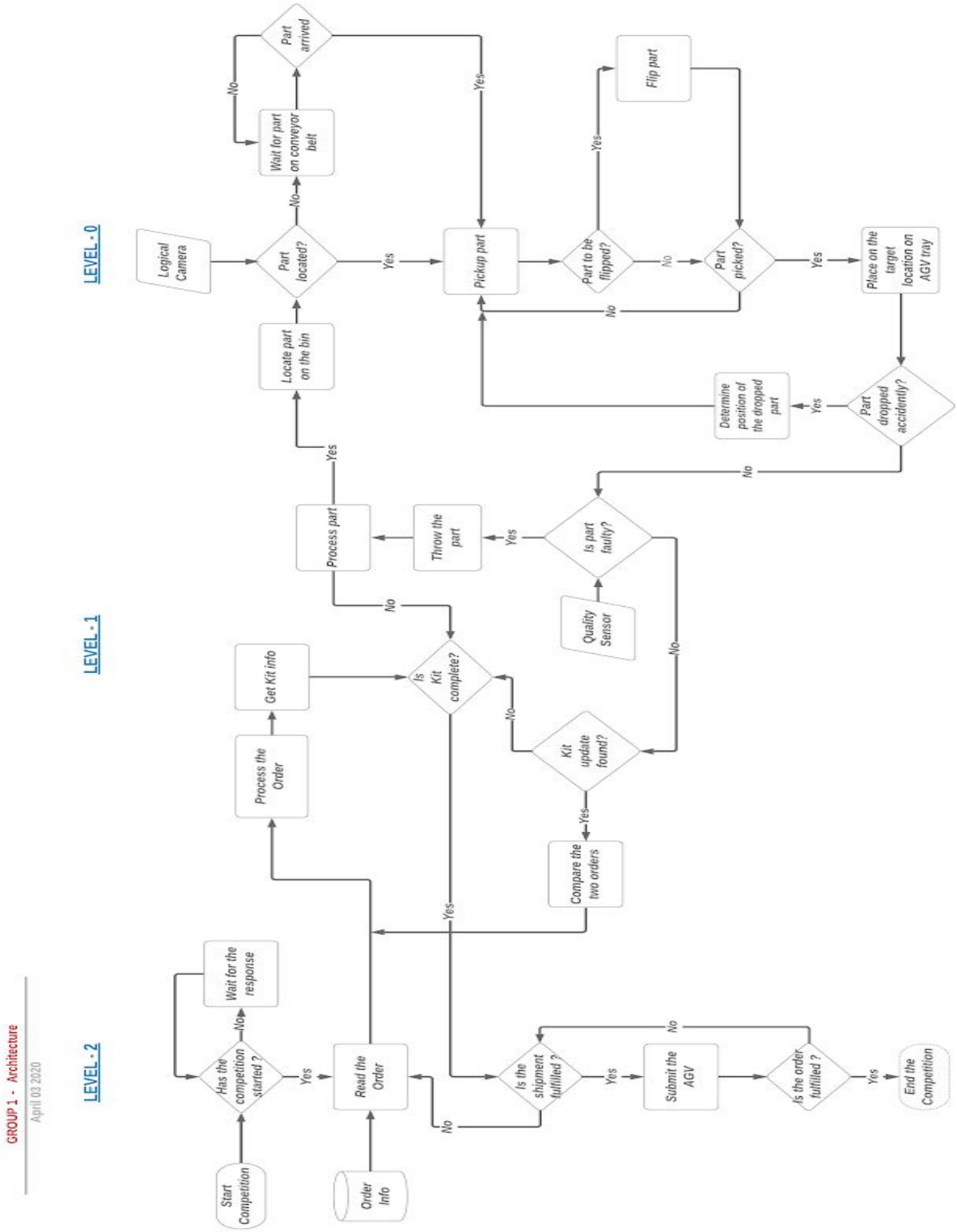
**Level 0:**

This level of architecture is called lower level in architecture because this is where most of the hardware and actual work is done. The logical sensor gives an idea whether we located the part or not and if yes then we pick the part or we wait for the conveyor belt to roll out and then we pick the part using the robotics arm. Part is then placed on AGV with appropriate orientations to meet order requirements. Subjectively, this is the most difficult stage to execute because the complexities involved with planning in pick and place operations and interacting with the hardware itself.

We have made this conscious choice because this particular architecture gives flexibility to move between sense, plan and act. On the other hand, it is also most modular and robust in terms of operation reliability. Because of this architecture we are not bound to follow stringent rules of other architectures. Thus due to these reasons we made a choice upon hybrid architecture.

The architecture of our project can be found here in the Drive link.

**ARCHITECTURE:**:



## III. PART HANDLING

### 2.1) Pick and Place

To pick up and place a part from the conveyor belt, the following tasks were performed. We created a ROS package that was able to store order, shipment and product data. It had the readings from the sensors/cameras placed over the conveyor belt to capture moving products.

We used two approaches to pick a part from the bins. The difference between the two is the goal position of the end effector we provide to the planner. The two are:

- Cartesian Coordinates
- Joint Configuration

Initially, we worked with cartesian coordinates as the target position for the planner and we observed inconsistency in the motion of the arms. The reason being, to achieve a particular goal position, there can be several different joint configurations the arm can achieve. So we decided to take a different approach and provide the planner the arm configuration we want. This brought about a considerable amount of consistency in the behavior of the arms. The one issue is that, since we retrieve all the part positions in cartesian, it becomes difficult to determine the arm configuration to pick a part and hence we use a combination of the above two approaches.

The arm 1 picks the belt part and waits for the arm 2 to also pick up the part and once the arm 2 also picks up the part then the arm 1 moves to the AGV tray 1 and drop the part on the desired drop pose.After the first part is dropped then the arm 2 moves to the AGV tray 2 and tries to drop the part.

The belt part is dropped before placing it on the expected drop pose. The drop_pose() function is called wherein the current pose of the part and expected pose of the part are compared. The arm 2 moves to the dropped pose and it picks it up to drop on the expected part drop pose. Once the belt parts are handled, then the other parts in the order list are executed.

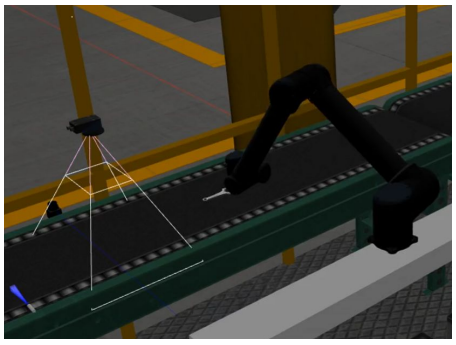


Figure 1: Picking Belt Parts

Figure 2: Both arms picked up the belt parts

### 2.2) Building a whole kit

The aim of this project was to build a whole kit using an order having two shipments, each consisting of 5 products. There were two AGVs involved in this task. We had a sensor blackout of 50 seconds but it didn't affect our operation.
The shipments consisted of the following parts that were fulfilled in the kit:

1. gasket_part
2. piston_rod_part
3. gear_part
4. pulley_part
5. Disk_part

There were two faulty parts namely, a gasket_part_3 and a gear_part_3. We were successfully able to pick up the parts from the conveyor, handle them by using both the robot arms and deliver the products on the respective AGV. One of the parts needed to be flipped during this operation and the faulty parts were detected and discarded(i.e. Dropped on the ground).

Figure 3: Completed Shipment 0 on AGV tray 1

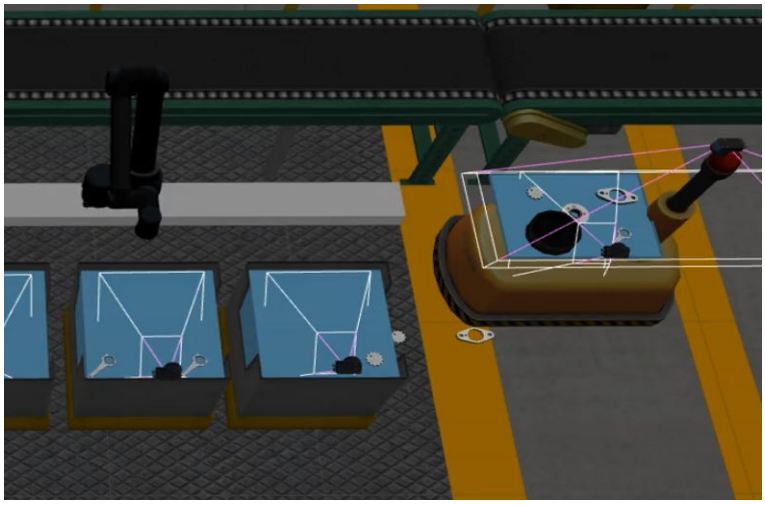


Figure 4: Completed Shipment 1 on AGV tray 2

### 2.3) Part Flipping

In order to meet order requirements, we also have to consider orientation of the different parts and in order to do so we have to flip the parts as mentioned in the order update.

To flip a part, let's assume a disk part, we first pick the part slightly from the bin by using the robot arm and place it vertically on the bin. The next step is to slightly push the disk from the opposite side of the disk so that when it falls on the ground, it is flipped. Now, as the task of flipping is accomplished, we can simply pick up the part by using the robot arm and place it wherever desired.

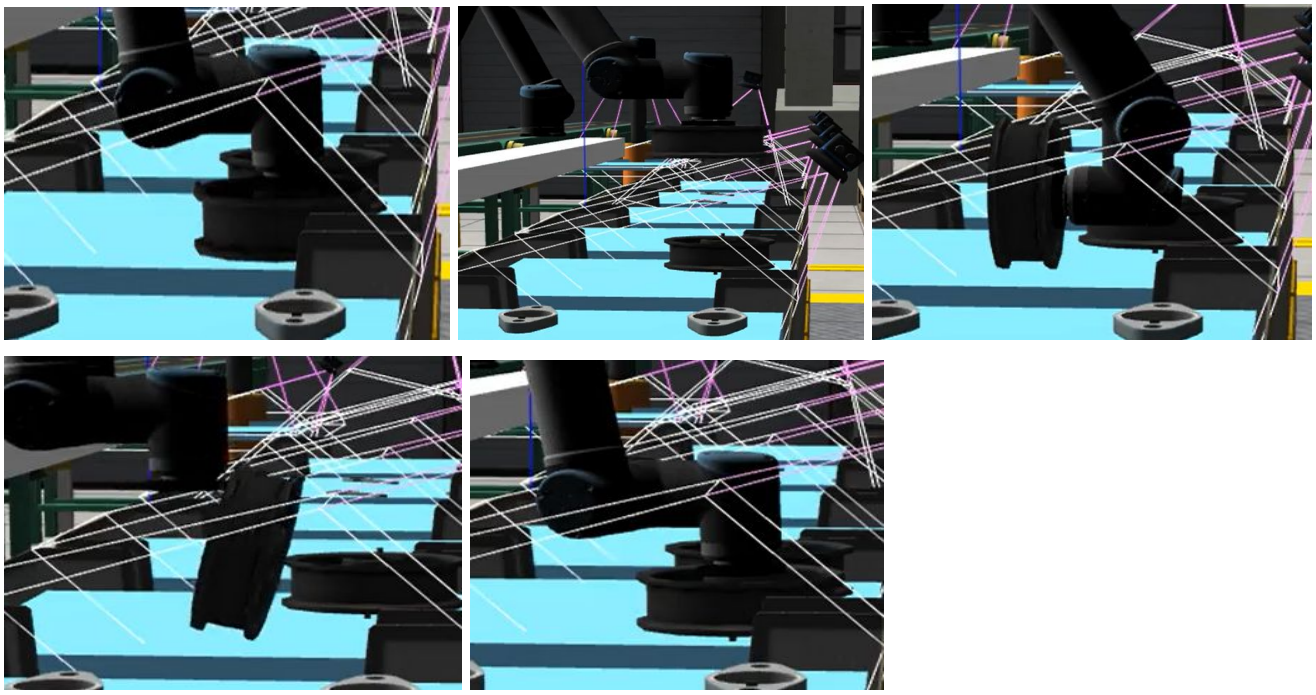


Figure 5: The above pictures show the sequence of actions we undertook to Flip a part.

### 2.4) Faulty Part Handling

While fulfilling an order, there can be times when the part is faulty. Initially, the part is picked up from the bin and placed onto the AGV as per the order. Once the pick and place operation is complete, the part in the AGV is checked by a quality sensor which is mounted on the top of the AGV to detect faulty parts. If the part turns out to be faulty, then the robot arm picks up the faulty part from the AGV again and drops it anywhere on the ground. This indicates that we no longer require that part and it has been permanently discarded.

Few of the parts in the environment were tagged faulty and our task was to make sure none of them ended up on the AGV tray. The quality sensors were placed on the AGVs and it will return the product details(part type and pose) if a part is faulty. If we find a part faulty, we discard it and replace it right away.

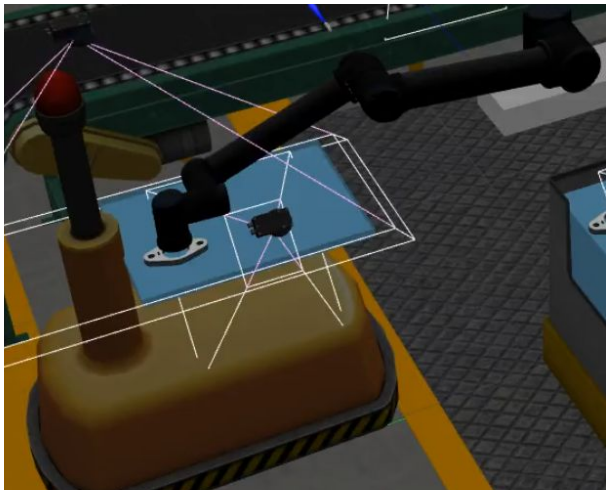


Figure 6: The Faulty product is picked from Tray

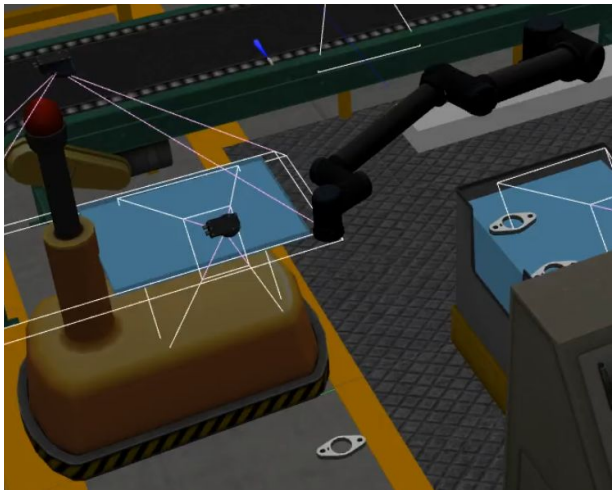


Figure 7: Faulty product is dropped on the floor

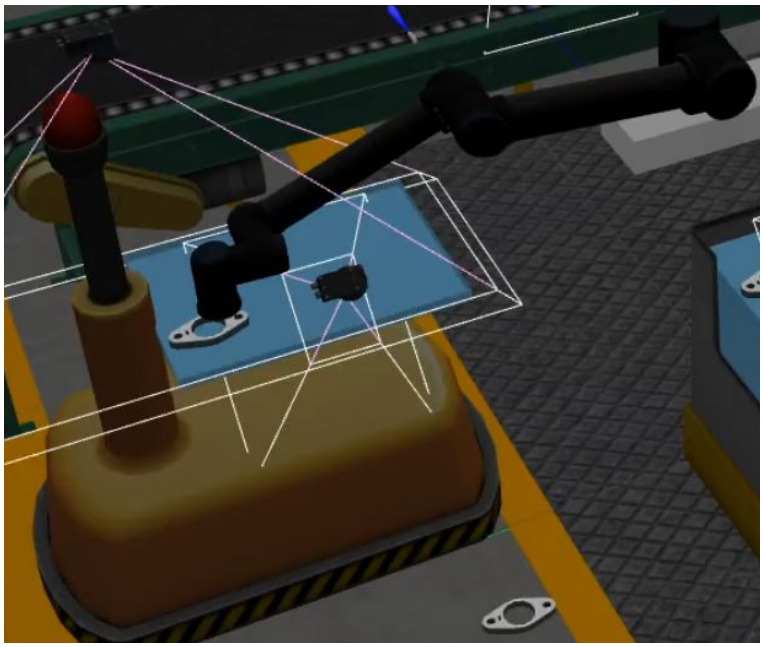


Figure 8: Faulty product is replaced with a better part in the same drop pose

### 2.5) Drop part Handling

In this challenge, the arm would accidentally drop the part at the wrong pose in the AGV tray. Our task was to correct the pose of the dropped part. We installed a camera overlooking the AGV

trays. Whenever we detect that the part is dropped by the arm before reaching the final pose, we trigger a function to determine the pose of the dropped part. This function compares with the pose of the parts from the order that is fulfilled and the parts we see from the camera and figure out the misplaced part. The arm then picks up the part from the wrong pose and places it in the correct pose.

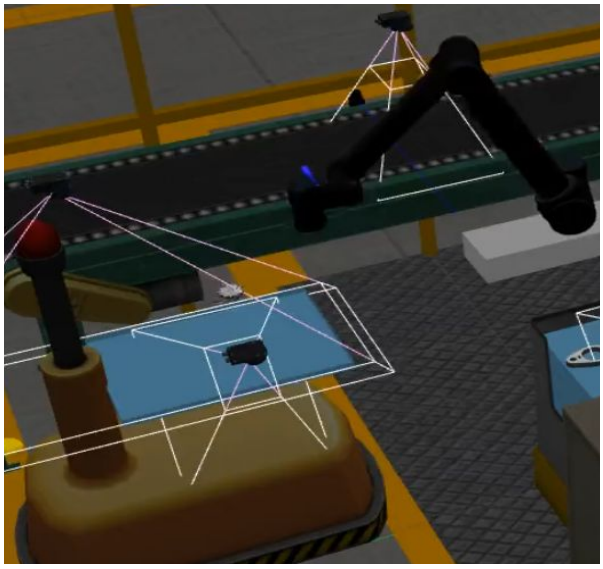


Figure 9: Gear Part dropped from Arm 1

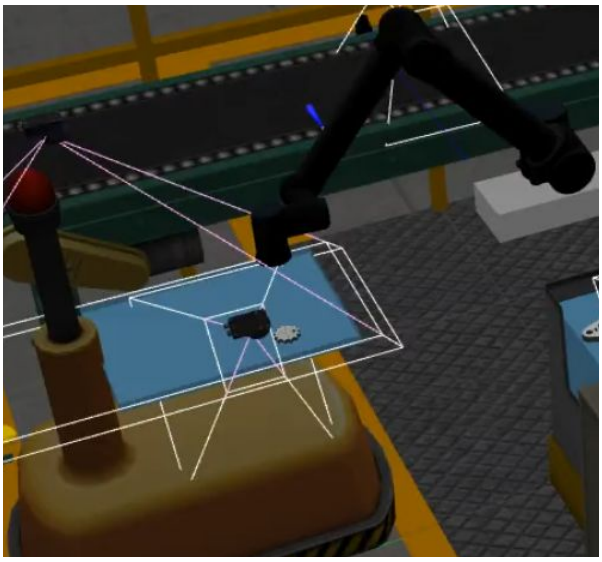


Figure 10: Dropped Gear is picked up and dropped and again on the correct drop pose

### 2.6) Different Belts Part Handling

When we start the competition, different types of parts start rolling on the conveyor belt. As soon as the part that the first robot arm is supposed to pick up arrives, it picks up the part by its end effector that is a vacuum gripper and waits for the second robot arm. After the part that is supposed to be picked up by the second robot arm arrives on the conveyor belt under the second robot arm, it picks up the part.

Once both the parts are picked up, the first robot arm uses the slider to travel near the AGV that is on its side and after reaching there, orients and places the part in the desired position on the AGV. After the arm has finished this task, the second robot arm does the same but on the right hand side AGV which is closer to that arm as they cannot cross each other.
Once both the parts are placed on the respective AGVs, both the arms come back to their home position and are ready for the next order.

Figure 11: Picking Disk part from Belt

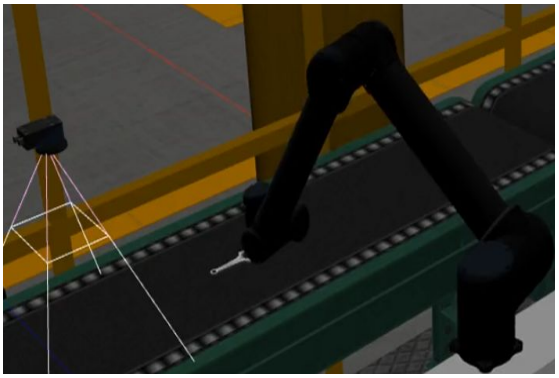


Figure 12: Picking Piston Rod part from Belt

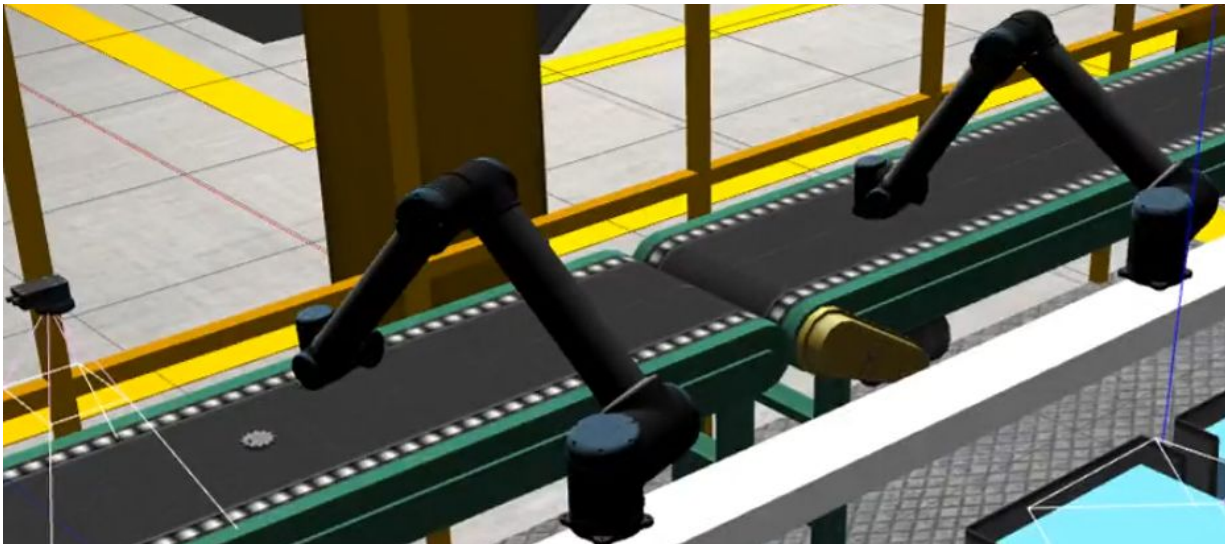


Figure 13: Picking Gear part from Belt

### 2.7) In Process Order Update

This challenge involved updating the current order by making some modifications like changing the pose of the current parts, discarding some parts, and adding some new based on the new order list. So after dropping a part on the AGV, we check for any updates on the order. If an order update is found, we compare the updated order to the older order and check for the modifications and perform them.

The flow that we followed while modifying the order is:

- We first put all the existing parts and their positions in a queue.
- We pop one product and check if it is required or not. If not, we discard it.
- If it is required, we check the new pose of the product, if unchanged we move on.
- If it is changed, we check if the new pose is occupied or not. If not, we pick up the part from that location and place.

- If occupied, we place the part at a temporary pose and push that to the queue. And move on to the next product and repeat the above.
- Once the queue is empty, we fulfill the products that were not present in the previous order.

## IV. TECHNICAL IMPLEMENTATION

### 3.1) Sensor setup (Logical and Quality):

There are totally nine Logical cameras that we used in this project to identify and locate the parts in the order. Each storage bin is mounted with a Logical Camera at the top and totally there are six cameras over the bins. There is one camera mounted above the Conveyor Belt to detect the presence of the type of the part on the moving conveyor belt.

Two more logical cameras are mounted with a skew angle on the top of the AGV tray to detect the type of parts that are dropped by the robot arms on the tray. It also helps in locating those parts on the tray and helps in analyzing whether the parts that are dropped at the AGV tray are correctly dropped at the required drop pose by the arms or it is dropped suddenly on the tray before the arm drops it in the correct pose.

### 3.2) Order handling :

The Order list is read and checked for the parts that are not available on the bin and whichever part is not available on the bin will be picked first from the conveyor using both the arms based on the number of shipments. The parts that are picked from the conveyor are dropped exactly on the correct drop pose at the AGV tray.

The parts that are available on the bin will be picked and dropped according to their sequence in the order list.The shipment 1 is completed first and then the AGV-1 is submitted for the delivery and it is followed by the pick and place of parts in the shipment 2 at AGV-2 tray

### 3.3) Robot Controlling :

The robot's home position is initially set to face the conveyor to pick up the parts from the conveyor first. After the pick and place of belt parts, robots are set to a new home position facing the bin. The robot arms will pick and place the parts according to the order sequence. The robots movements are programmed to coordinate in such a way that it won't clash against each other. The mod point on the rail is chosen for exchanging the parts picked by each arm to drop in the opposite AGV tray.

### 3.4) Part Exchange :

The part exchange function is implemented to pick the parts by an arm and drop them on the opposite AGV tray. The parts picked by the arms are initially dropped in the exchange pose on

the middle of the rail between two arms. Once the parts are dropped on the exchange pose, then the other arm approaches to that position and picks up the part. The picked up parts are taken towards opposite AGV and dropped on their respective drop pose in the AGV tray.

The Project Video of our Group1 can be seen here in this drive link: Video Link

## V. CHALLENGES FACED:

- In our initial approach, the arm moved around a lot and hit a lot of objects and always followed a random trajectory. To avoid this, we started giving joint states as goal position to the arm.

- The arms were not able to turn around to pick the parts due to MoveIt Planner limitations especially when they are facing in different directions from where the part to be picked is located.
  Hence we set the home position for the robot arms to be in a position just above the conveyor when it is going to pick the belt parts and have set just above the storage bin when it is going to pick parts from bins.

- When handling the shipment order parts, we processed the conveyor belt parts first in order to not miss the moving parts in the belt as we can't pick them up once they all ran off in the belt.

- While performing part exchange there were chances for both the robots to collide with each other, hence we chose an exchange pose and followed steps of dropping the part by one arm on the exchange pose position and picking up by the other arm from the same exchange pose position.

- When dropping the parts on the AGV tray, there are some parts that drop beforehand.
  We were not able to drop them in their expected droppose in the first attempt.
  Hence we picked them again and dropped them in their respective poses in the tray with the help of the position value obtained from the logical camera mounted on the tray.

- The in-order process update was difficult to implement since we had to efficiently take care of the already processed parts and the newer order. If the part on the tray has to be placed on a new position, and that position is already occupied, we have to devise our algorithm in such a way that it is taken care of without a lot of time-delay.

## VI. INDIVIDUAL CONTRIBUTION:

**Piyush Bhuva**

I have worked on choosing the architecture and deciding the workflow of the information. Splitting up the main task into smaller more achievable tasks was part of my job. I tried to design a lower level of the architecture and besides contributed to project report and presentation till final touch. For the initial parts of the project I placed sensors like a logical camera, added on top of the conveyor belt, took note of their position and orientation and edited yml files for sensors which helped to read parts coming on the conveyor belt.I helped streamlining the code for pick and place operation.

**Niket Shah**

I worked on the implementation of arm motion planning, picking parts from the bins and placing them on the AGV tray, drop part challenge, in-order process update, part exchange between two arms, faulty part detection and discard part and flipping the part. I also worked on creation of the architecture and the flow of the system.

**Prasanna Marudhu**

I worked on initializing the sensors in our project environment and did code implementation for reading the parts in the shipment orders and processing pick and place operation for each of the parts in the shipment order and implemented the program for picking up the conveyor belt parts and storage bin parts with the help of the sensors mounted on the top of bins, trays and belt. Also generalized the whole program for performing the same operation, if either the belt parts or bin the parts are different from sample yaml file and optimized the whole program and tested with different scenarios. I also contributed to the creation of the architecture of our system.

**Smriti Gupta**

Worked on building architecture and the system flow, building pipelines for each task and implementing them in code and documented the process simultaneously .

**Abhiram Dapke**

I contributed to the streamlining of the code for placing sensors and cameras and reading their inputs, pick and place operation and kit building. I helped with documenting the architecture and adding essential components to the report and powerpoint presentation.

## VII. FUTURE IMPROVEMENTS

Several things can be improved in the system that we implemented, some of the ideas are as below:

- The system performed slower than expected, so changes can be made there. First one would be to try and implement different planners instead of just RRTConnect.

- We can make the two arms work simultaneously, for that we believe multithreading has to be implemented and the two arms are supposed to run on different threads.

- For the above implementation, we would need a higher level planner to dictate and prioritize tasks to the two arms. This can be done by creating a planner using PDDL.

- Currently, the arms wait for the parts to arrive on the conveyor belt and then pick them up and place them and move on to finishing other tasks. But that waiting time can be utilized by optimizing the process of conveyor belt pick up and completing the rest of the order while waiting for the parts on the belt.

## VIII. CONCLUSION

In this project, our aim was to control the robot arms and the Ariac environment in such a way that we were able to accomplish the tasks to fulfill the orders and build the kits with the help of the two robot arms and we were able to accomplish it.

The first task was to put sensors and cameras on the top of the required bins and on the conveyor belt and make sure they are working and displaying values on the screen. Secondly, we had to pick and place parts in the order from the moving conveyor belt into the AGV.

After that,we had to build the whole kit consisting of orders and update the orders timely so that the system was capable of anticipating and updating an order and also detect and discard the faulty products using the quality sensors.

These parts were checked after they were placed onto the AGV. If any part turned out to be faulty, that part was picked up again from the AGV with the same robot arm and dropped on the ground i.e. it was discarded.

Consequently, we had to flip a disk part by using a single robot arm in the bin. All these tasks were performed effectively and the kits were delivered in the AGV tray.

## IX. FEEDBACK REFERENCES

This course was a perfect blend of knowledge and hands-on training. Initially, all of us were relatively new to the ROS and the ARIAC environment. This course bolstered our ROS concepts and we gained an understanding of the basic and advanced methods used to manipulate robots in simulated environments.

We learned how product manufacturing takes place in a real industry and how parts are transferred in between conveyors, how they are picked up from the bins, how a faulty part is checked and discarded if required and how to flip the parts. Prof.Kootbally and Prof.Craig both are very knowledgeable and have great experience working with the Ariac environment.

Moreover, writing each and every piece of code to implement it in a virtual environment was challenging and intrigued us the most. Overall, we would love to recommend this course to anyone who is interested in learning ROS core concepts and understanding the implementation of robotic arms in the real industrial world.

## X. REFERENCES


1. Lecture Slides
2. C++ Programming book - https://books.goalkicker.com/CPlusPlusBook/
3. Object Oriented Programming concepts - https://beginnersbook.com/2017/08/cpp-oops-concepts/
4. ARIAC Website Guidelines