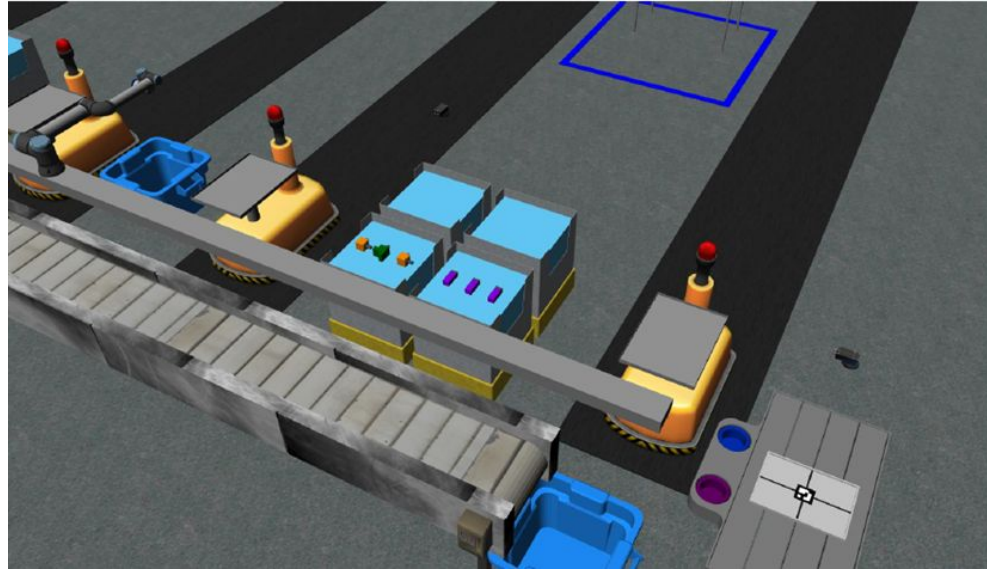


ENPM663: Building a Manufacturing Robotic Software System

RWA67



Team Members:

Shreejay Badshah

Tej Polamreddy

Ian Serbin

Arshad Shaik

Dhinesh Rajasekaran

Tim Sweeney

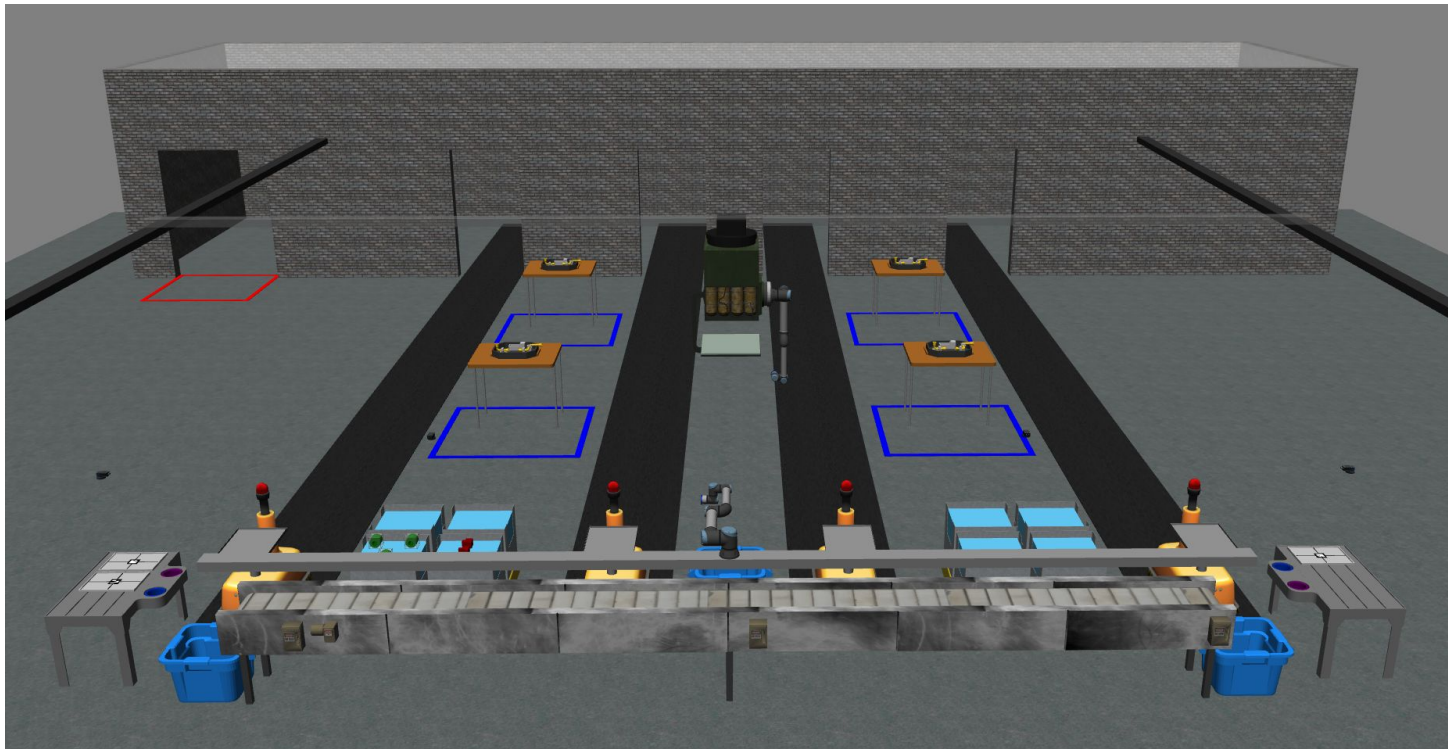
Table of Contents

- ▶ Introduction
- ▶ Problem Statement
- ▶ Code Block Diagram
- ▶ Environment Sensors
- ▶ Order Manager
- ▶ Package Contents
- ▶ Trial Files
- ▶ Result
- ▶ Challenges Faced
- ▶ How can the results be made better?
- ▶ Conclusion
- ▶ Demo

Introduction:

ARIAC is hosted by National Institute of Standards and Technology (**NIST**) to test the **agility** of the industrial robot system.

The purpose of ARIAC is to make industrial robots **more autonomous**, **productive** with **minimum human input**.

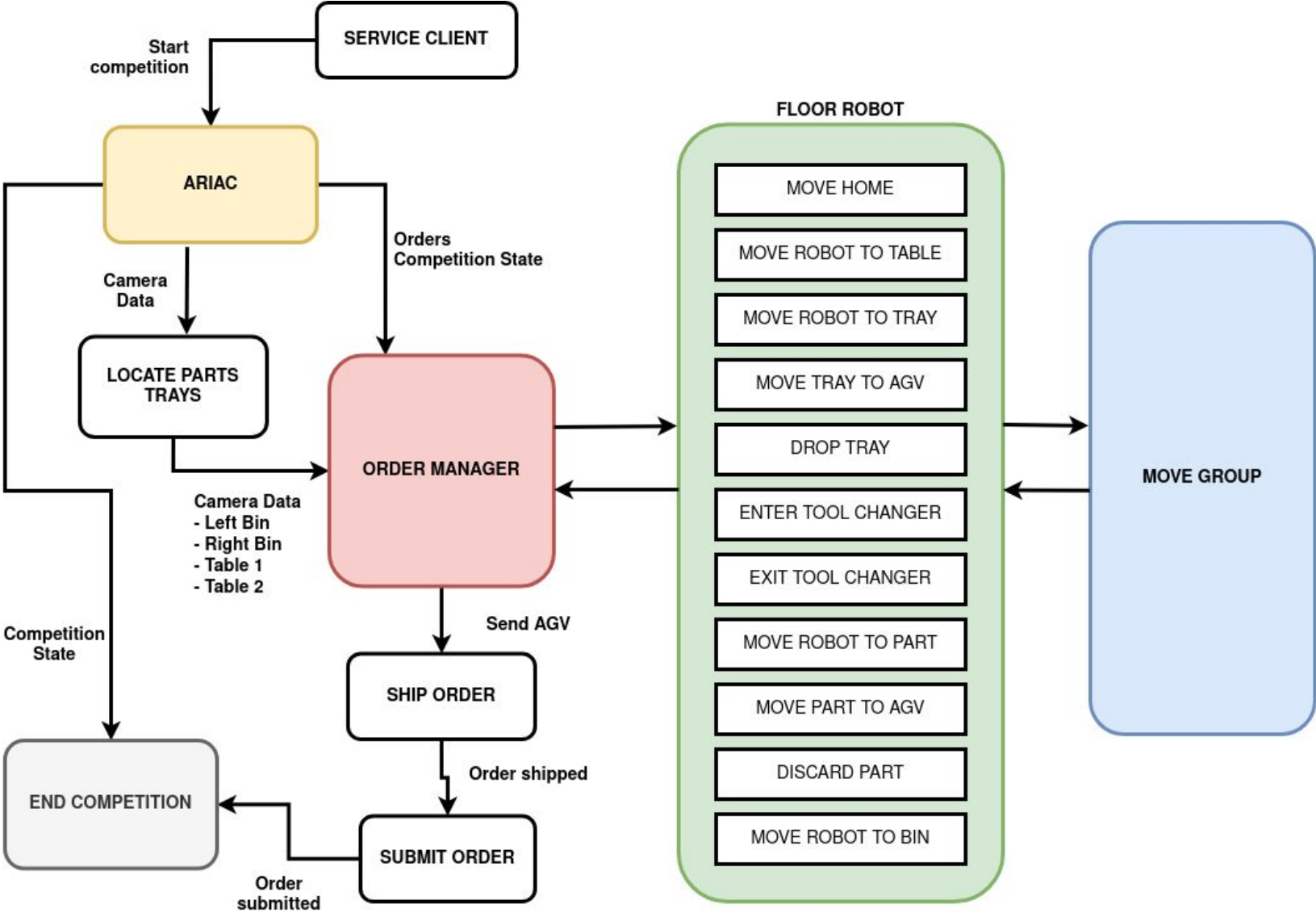


Problem Statement:

For this course the problem was to test the agility of an industrial floor robot to complete kitting tasks with the challenges of:

- ▶ Faulty part
 - ▶ Parts placed in certain quadrants for the first time are deemed faulty. They must be properly disposed of in the discard bin to avoid penalty
- ▶ Insufficient part
 - ▶ There are not enough parts to fulfill an order. The order must be complete to its fullest and still submitted.

Code Block Diagram:

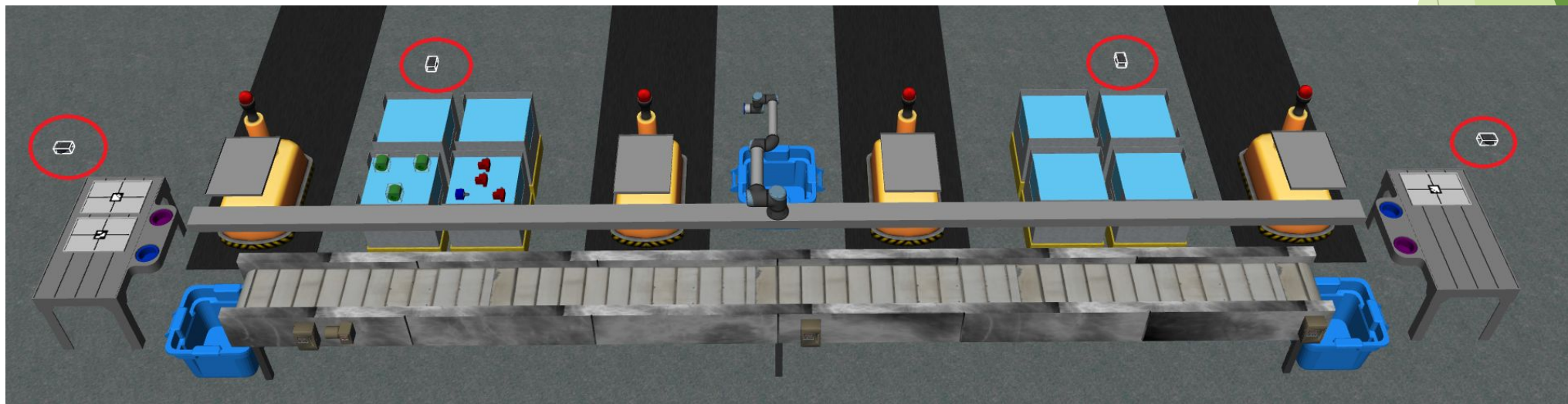


Environment Sensors:

- ▶ Provide control system with information necessary to build world model
- ▶ 4x Advanced Logical Sensors
 - ▶ Provide tray id and pose
 - ▶ Provide part pose, color, and type
- ▶ AGV Tray Sensors
 - ▶ Located on each AGV
 - ▶ Provide quality information of tray and parts

sensors.yaml

```
sensors:  
  left_bins_camera:  
    type: advanced_logical_camera  
    pose:  
      xyz: [-2.286, -2.96, 1.8]  
      rpy: [pi, pi/2, 0]  
  
  right_bins_camera:  
    type: advanced_logical_camera  
    pose:  
      xyz: [-2.286, 2.96, 1.8]  
      rpy: [pi, pi/2, 0]  
  
  kts1_camera:  
    type: advanced_logical_camera  
    pose:  
      xyz: [-1.3, -5.8, 1.8]  
      rpy: [pi, pi/2, pi/2]  
  
  kts2_camera:  
    type: advanced_logical_camera  
    pose:  
      xyz: [-1.3, 5.8, 1.8]  
      rpy: [pi, pi/2, -pi/2]
```

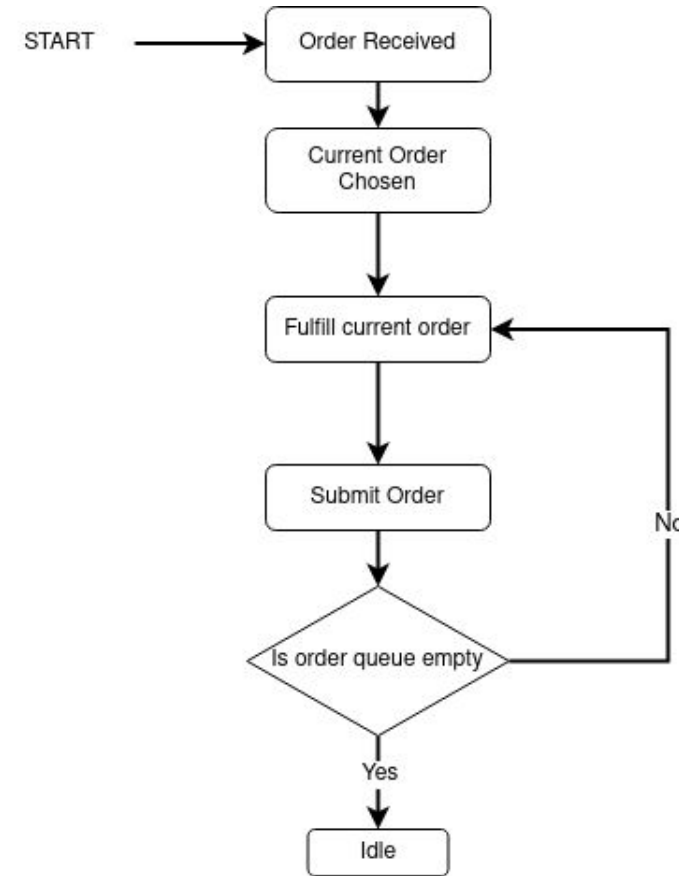
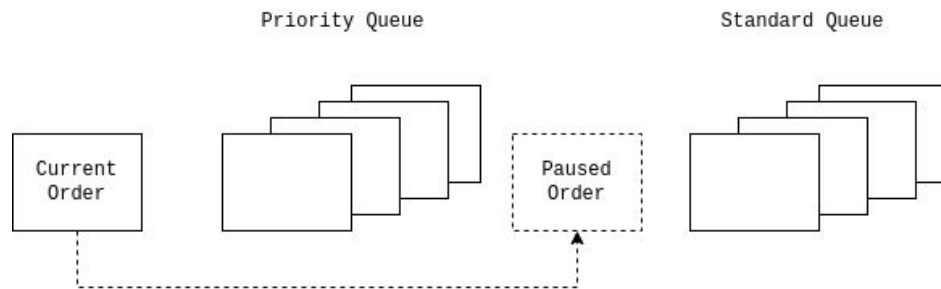


Order Manager - In Depth:

The Order Manager node is the heart of the competition package. It is the

Subscriptions

- `ariac/competition_state`
Begins fulfilling orders
- `ariac/orders`
Adds orders to the queues
- `/left_bin_camera_data`
`/right_bin_camera_data`
`/kitting_tray1_camera_data`
`/kitting_tray2_camera_data`
Records the poses of parts and trays. These have been obtained by `locate_parts_trays`, translated to world pose, then republished



Order Manager - In Depth (cont.):

The Order Manager node is the heart of the competition package. It contains the main logic and makes the service requests to floor_robot

```
Fullfill_order:
    order = current_order
    tray = order.tray
    agv = order.agv

    tray_pose = search_tray_inventory(tray)
    move_robot_to_table()
    change_to_tray_gripper()
    grab_tray()
    place_tray_on_agv()
    remove_tray_from_inventory()

    move_robot_to_table()
    change_to_part_gripper()
    for (part in list_of_parts):
        part_pose = search_part_inventory(part)
        If (part_pose == None):
            Continue #part not found.. Moving on
        place_part_on_tray()
        perform_quality_check()
        If (faulty):
            Move part to bin
            Add part back to list_of_parts
        Remove_part_from_inventory()
    complete_order()
```


Package Contents:

- ▶ Robot_msgs
 - ▶ Custom Service Messages

```
▼ robot_msgs
  ▼ srv
    ≡ DiscardPart.srv
    ≡ DropPart.srv
    ≡ DropTray.srv
    ≡ EnterToolChanger.srv
    ≡ ExitToolChanger.srv
    ≡ MovePartToAGV.srv
    ≡ MoveRobotToBin.srv
    ≡ MoveRobotToPart.srv
    ≡ MoveRobotToTable.srv
    ≡ MoveRobotToTray.srv
    ≡ MoveTrayToAGV.srv
  M CMakeLists.txt
  📄 package.xml
```

- ▶ Robot_msgs
 - ▶ Main Competitor Package

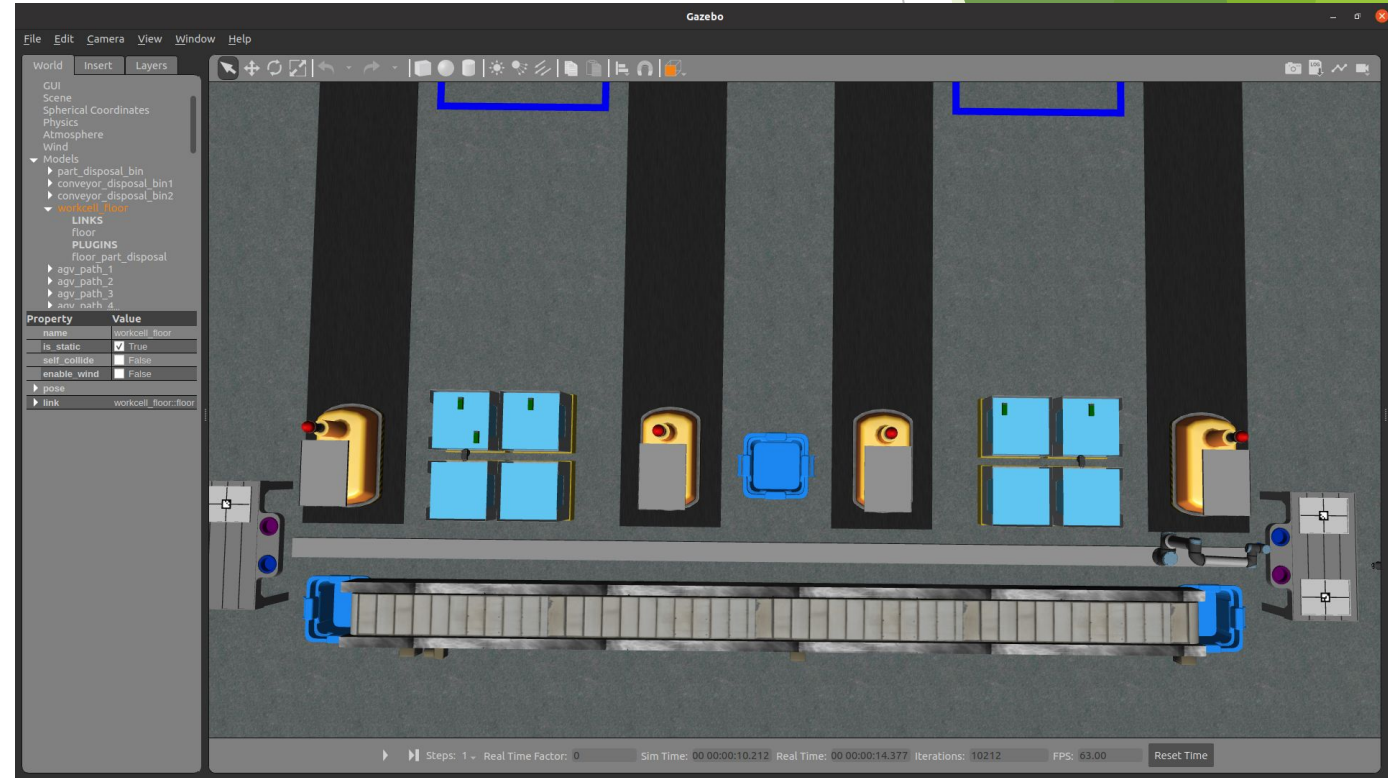
```
▼ rwa67
  ▼ config
    ! floor_robot_targets.yaml
    ! sensors.yaml
  ▼ include/rwa67
    📄 ariac_constants.hpp
    📄 ariac_tf_util.hpp
    📄 floor_robot.hpp
    📄 locate_parts_trays.hpp
    📄 service_client.hpp
    📄 shipOrder.hpp
    📄 utils.hpp
  ▼ launch
    📄 rwa67_app.launch.py
  > meshes
  ▼ nodes
    📄 end_comp_client_exe.py
    📄 order_manager.py
    📄 submit_order.py
  ▼ rwa67
    📄 __init__.py
    📄 end_comp_client.py
    📄 orders_interface.py
    📄 submit_order_interface.py
  ▼ src
    📄 ariac_tf_util.cpp
    📄 floor_robot.cpp
    📄 locate_parts_trays.cpp
    📄 main.cpp
    📄 service_client.cpp
    📄 shipOrder.cpp
  M CMakeLists.txt
  📄 package.xml
  ▼ trials
    ! rwa67_summer2023_SB_test1.yaml
    ! rwa67_summer2023_TS_test1.yaml
```

Approx. Lines of code = 5053

Trial Files:

The Order Manager node is the heart of the competition package:

- rwa67_summer2023.yaml
- rwa67_summer2023_SB_test1.yaml
 - 4 orders
 - Multiple Faulty Parts
 - Insufficient Parts
- rwa67_summer2023_TS_test1.yaml
 - Order contains 3 faulty parts and only 4 available



Testing: faulty parts lead to missing parts

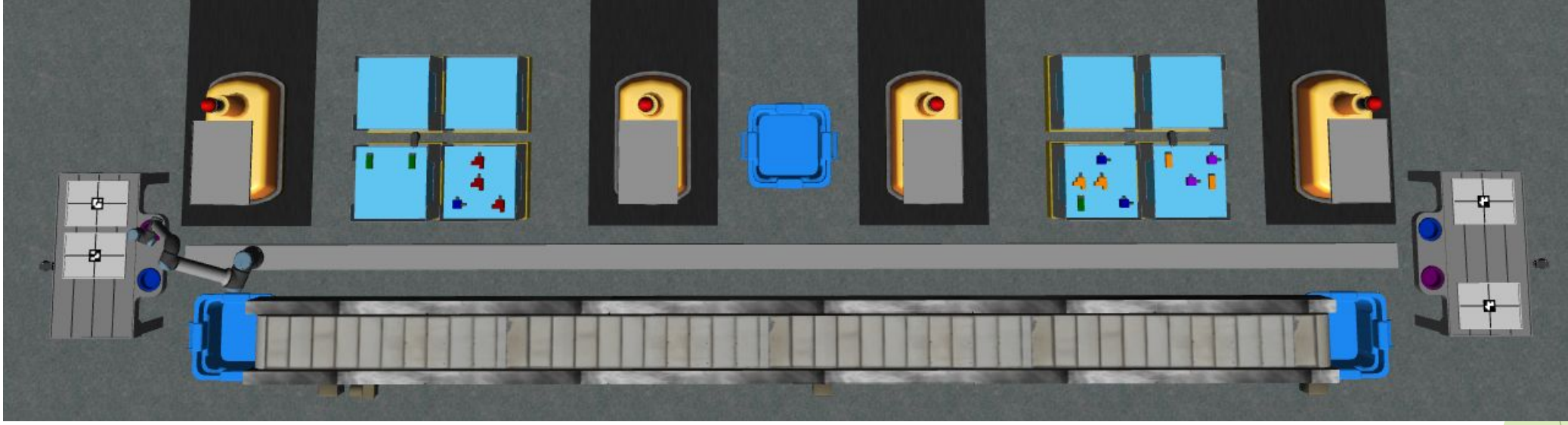
rwa67_summer2023_SB_test1.yaml

```
parts:
  bins:
    bin1:
      - type: 'battery'
        color: 'orange'
        slots: [1,6]
      - type: 'regulator'
        color: 'purple'
        slots: [3,5]
    bin2:
      - type: 'regulator'
        color: 'blue'
        slots: [2,9]
      - type: 'sensor'
        color: 'orange'
        slots: [4,5]
      - type: 'battery'
        color: 'green'
        slots: [7]
    bin5:
      - type: 'battery'
        color: 'green'
        slots: [1,3]
    bin6:
      - type: 'sensor'
        color: 'red'
        slots: [2,5,9]
      - type: 'regulator'
        color: 'blue'
        slots: [7]
```

```
orders:
  - id: 'KITTING1'
    type: 'kitting'
    announcement:
      time_condition: 0
    priority: false
    kitting_task:
      agv_number: 2
      tray_id: 3
      destination: 'warehouse'
      products:
        - type: 'battery'
          color: 'green'
          quadrant: 1
        - type: 'sensor'
          color: 'red'
          quadrant: 3
  - id: 'KITTING2'
    type: 'kitting'
    announcement:
      part_place_condition:
        agv: 2
        type: 'sensor'
        color: 'red'
    priority: false
    kitting_task:
      agv_number: 3
      tray_id: 5
      destination: 'warehouse'
      products:
        - type: 'battery'
          color: 'green'
          quadrant: 2
        - type: 'regulator'
          color: 'blue'
          quadrant: 4
```

```
  - id: 'KITTING3'
    type: 'kitting'
    announcement:
      part_place_condition:
        agv: 2
        type: 'sensor'
        color: 'red'
    priority: false
    kitting_task:
      agv_number: 4
      tray_id: 7
      destination: 'warehouse'
      products:
        - type: 'sensor'
          color: 'red'
          quadrant: 1
        - type: 'battery'
          color: 'green'
          quadrant: 2
  - id: 'KITTING4'
    type: 'kitting'
    announcement:
      part_place_condition:
        agv: 4
        type: 'sensor'
        color: 'red'
    priority: false
    kitting_task:
      agv_number: 1
      tray_id: 7
      destination: 'warehouse'
      products:
        - type: 'battery'
          color: 'orange'
          quadrant: 2
        - type: 'regulator'
          color: 'purple'
          quadrant: 4
```

```
challenges:
  - faulty_part:
      order_id: 'KITTING1'
      quadrant3: true
  - faulty_part:
      order_id: 'KITTING2'
      quadrant2: true
  - faulty_part:
      order_id: 'KITTING4'
      quadrant2: true
```

rwa67_summer2023_TS_test1.yaml

```
parts:
  bins:
    bin1:
      - type: 'battery'
        color: 'green'
        slots: [1]
    bin2:
      - type: 'battery'
        color: 'green'
        slots: [2]
    bin5:
      - type: 'battery'
        color: 'green'
        slots: [2]
    bin6:
      - type: 'battery'
        color: 'green'
        slots: [2, 9]
```

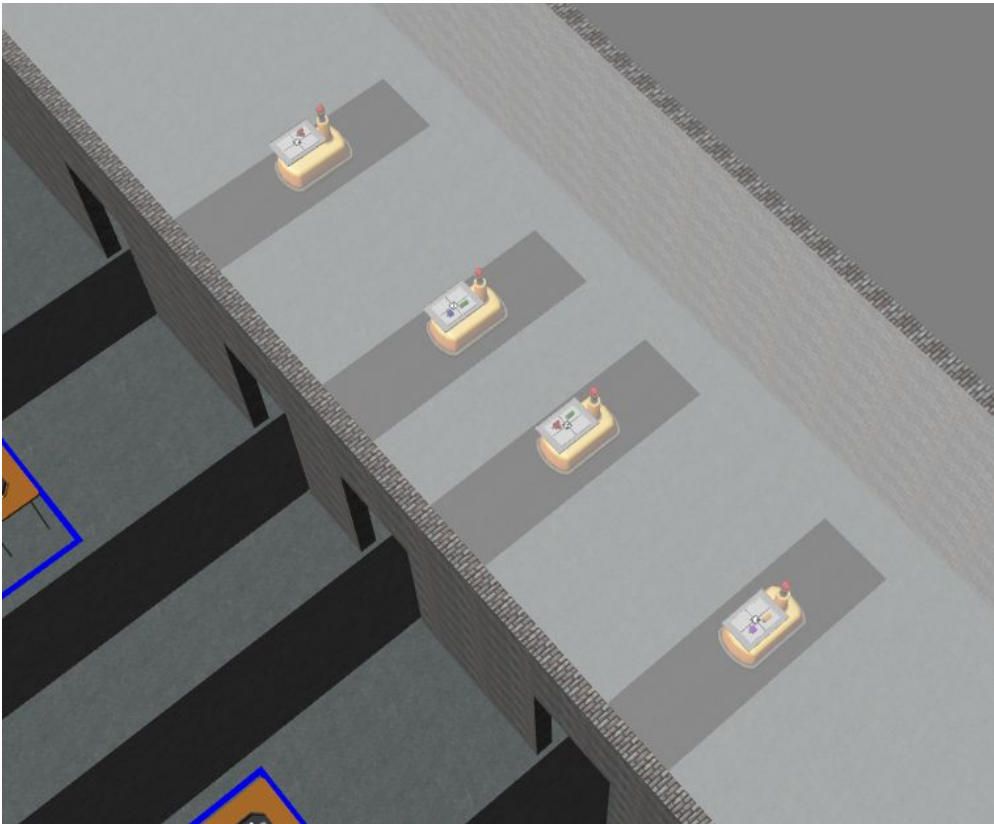
```
orders:
  - id: 'KITTING1'
    type: 'kitting'
    announcement:
      time_condition: 0
    priority: false
    kitting_task:
      agv_number: 2
      tray_id: 5
      destination: 'warehouse'
      products:
        - type: 'battery'
          color: 'green'
          quadrant: 1
        - type: 'battery'
          color: 'green'
          quadrant: 2
        - type: 'battery'
          color: 'green'
          quadrant: 3
        - type: 'battery'
          color: 'green'
          quadrant: 4
```

```
challenges:
  - faulty_part:
      order_id: 'KITTING1'
      quadrant1: true
      quadrant2: true
      quadrant3: true
```

Result:

rwa67_summer2023_SB_test1.yaml:

- completion time: 162 sec
- Score: 39

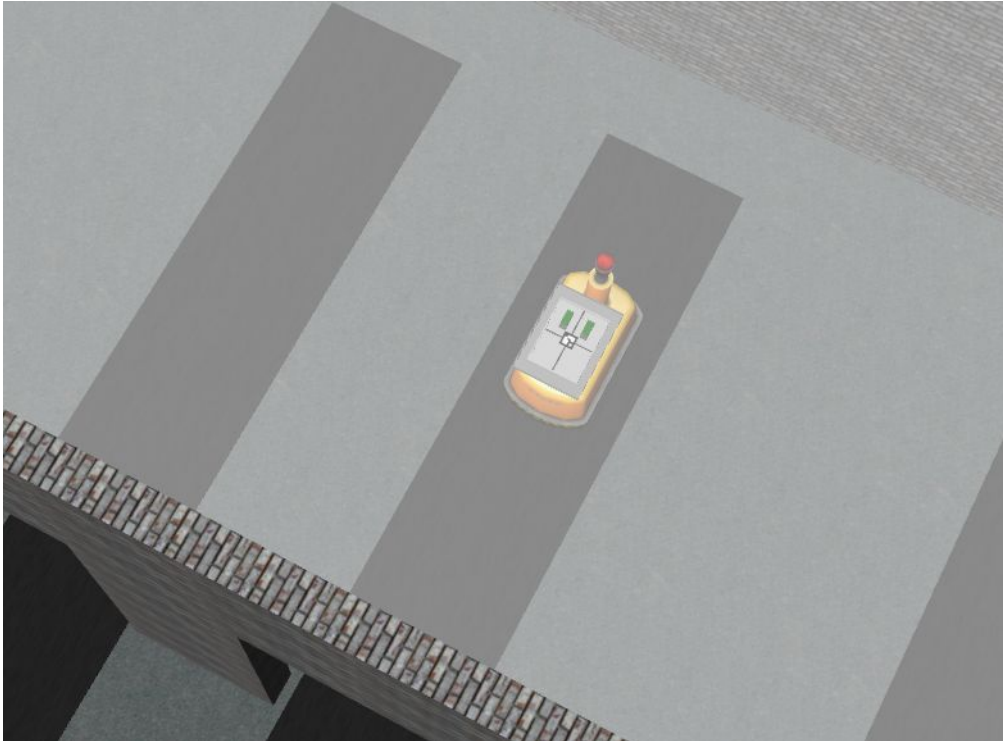


```
[gzserver-2] =====  
[gzserver-2] END OF TRIAL  
[gzserver-2] =====  
[gzserver-2] Trial file: rwa67_summer2023_SB_test1.yaml  
[gzserver-2] Trial time limit: -1.000000  
[gzserver-2] Trial completion time: 161.585000  
[gzserver-2] Trial score: 39  
[gzserver-2] =====
```


Result (cont.):

rwa67_summer2023_TS_test1.yaml:

- completion time: 97 sec
- Score: 9



```
[gzserver-2] =====  
[gzserver-2] END OF TRIAL  
[gzserver-2] =====  
[gzserver-2] Trial file: rwa67_summer2023_TS_test1.yaml  
[gzserver-2] Trial time limit: -1.000000  
[gzserver-2] Trial completion time: 96.396000  
[gzserver-2] Trial score: 9  
[gzserver-2] =====
```

Challenges Faced:

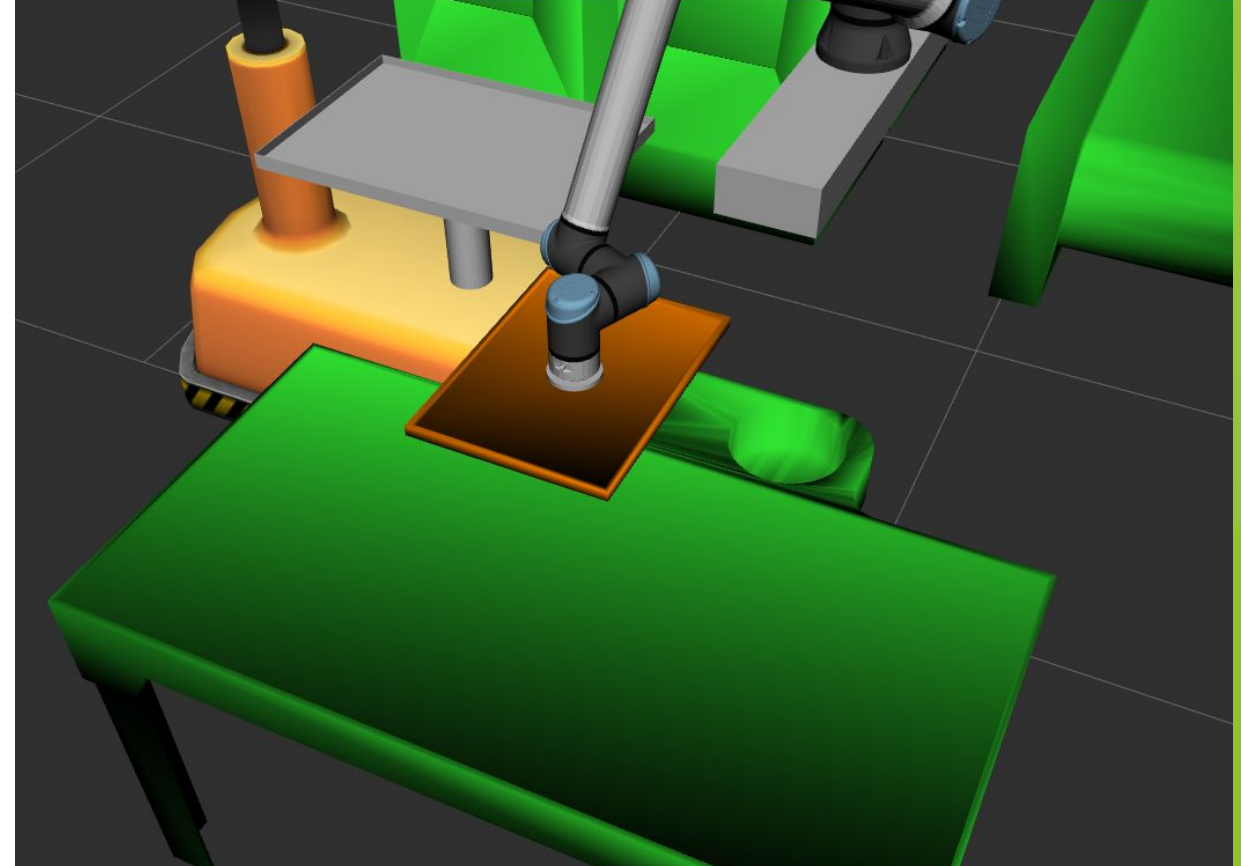
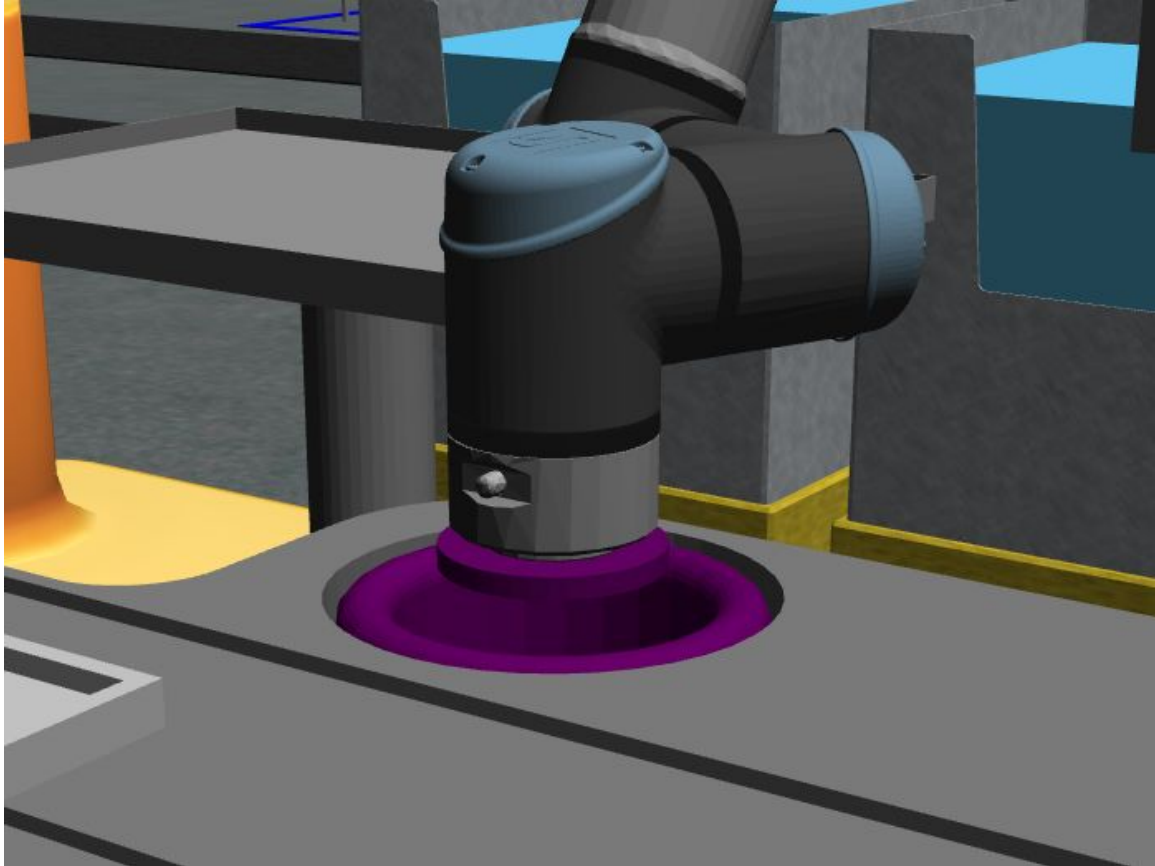
Numerous obstacles were faced during this project such as:

- ▶ Building a robust control architecture to handle challenges in orders.
- ▶ Parsing camera data to manager the order providing by the competition.
- ▶ Writing servers and using appropriate services to perform Kitting task in Gazebo environment.
- ▶ Breaking down services for a smoother simulation.
- ▶ Establishing appropriate communication between servers (written in C++) and clients (written in Python).

Challenges Faced (cont.):

- ▶ Fine tuning parameters in server callback methods for the floor robot to perform pick and place task for various parts and tray.
- ▶ Ensuring parts were properly added to planning scene before attempting to interact with them.
- ▶ Adjusting waypoints and targets to avoid collision with objects not in planning scene (objects still in bin).
- ▶ What to snack on at midnight when floor robot goes rogue, dancing on the rail.

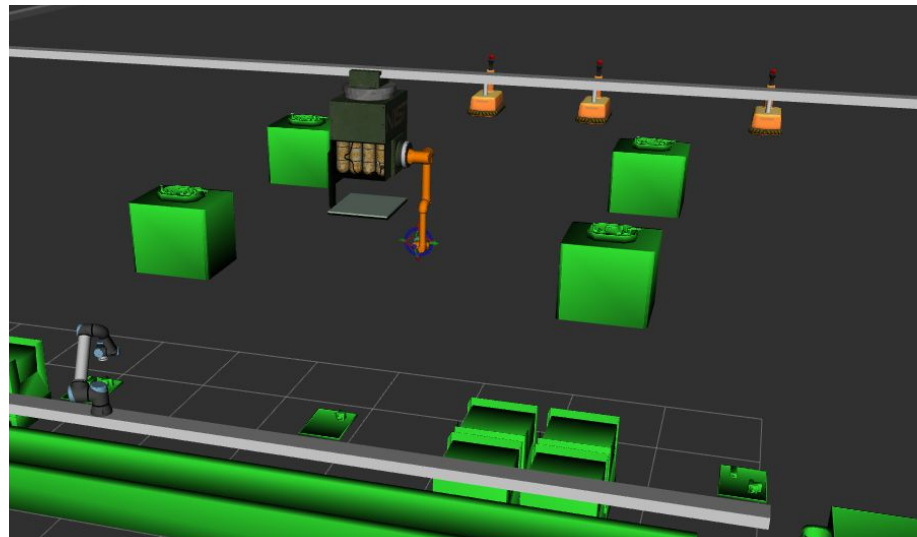
Challenges Faced (cont.):



Robot would not enter gripper changer because the planning scene still had the tray attached.

How can the results be made better?

- Tray and Part inventories are determined at start. Ideally, part and tray locations should be continuously updated (bumped parts, dropped parts)
- Integrate ceiling robot to reach all parts in bins
- Attach trays/parts to AGVs in planning view. Current implementation would cause collisions when using the AGV for a second time.
- Finish Priority order logic



Conclusion

- ▶ The 'rwa67' accomplishes its objectives:
 - ▶ Perform motion planning to perform up to orders
 - ▶ 1 per AGV
 - ▶ Cater for insufficient parts & faulty parts

Demo:

Build the packages:

```
cd ~/ariac_ws/  
colcon build --packages-select robot_msgs  
colcon build --packages-select rwa67
```

Launch:

```
ros2 launch rwa67 rwa67_app.launch.py
```