Z. Kootbally/C. Schlenoff
University of Maryland
College Park, MD

Final Project: Getting Ready

ENPM809B : Fall 2020
Due **Wednesday, December 2, 2020**

Contents

Introduction

The final project will be about running your program with an unseen YAML file which consists of a combination of agility challenges. The goal of this course is to develop a robust and agile robotic system for pick and place. Therefore, your program should be capable of handling any trials, including the one used for the final project.

Practice with Existing Trial Files

15 trial files, used in ARIAC 2020 finals, have been uploaded on Canvas. The best way to test the robustness of your program is to test it with all 15 trial files. Each trial file comes with a description in the form of a PDF file.

Todo List

☐ Robot can pick up each part type from each bin. The YAML snippet below shows the different possible part types in the bins where `N` can take any value in [1–16] and `COLOR` can take any value in {`red`, `green`, `blue`}.

```
models_over_bins:
  binN:
    models:
      disk_part_COLOR:
        xyz_start: [0.1, 0.1, 0.0]
        xyz_end: [0.5, 0.5, 0.0]
        rpy: [0, 0, 0]
        num_models_x: 3
        num_models_y: 3
  binN:
    models:
      piston_rod_part_COLOR:
        xyz_start: [0.1, 0.1, 0.0]
        xyz_end: [0.5, 0.5, 0.0]
        rpy: [0, 0, 'pi/4']
        num_models_x: 3
        num_models_y: 3
   binN:
    models:
      pulley_part_COLOR:
       xyz_start: [0.15, 0.15, 0.0]
       xyz_end: [0.45, 0.45, 0.0]
       rpy: [0, 0, 0]
       num_models_x: 2
       num_models_y: 1
   binN:
    models:
      gear_part_COLOR:
       xyz_start: [0.1, 0.1, 0.0]
       xyz_end: [0.5, 0.5, 0.0]
       rpy: [0, 0, 0]
       num_models_x: 3
       num_models_y: 3
   binN:
    models:
      gasket_part_COLOR:
       xyz_start: [0.1, 0.1, 0.0]
       xyz_end: [0.5, 0.5, 0.0]
       rpy: [0, 0, 'pi/4']
       num_models_x: 3
       num_models_y: 3
```

☐ Robot can pick up each part type from shelves 1, 2, 5, 8, and 11. In the final YAML file, parts will be spawned in a single row. An example is shown in Figure 1.

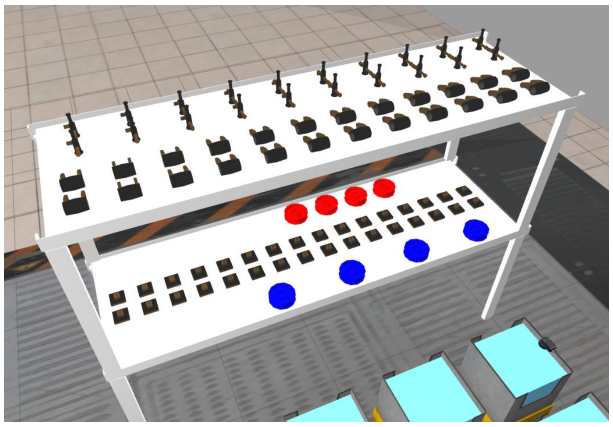

Figure 1: Parts spawn in a single on each side of a shelf.

The YAML snippet corresponding to Figure 1 is shown below.

```
models_over_shelves:
  shelf1:
    models:
      pulley_part_red:
        xyz_start: [1, 0.45, 0.0]
        xyz_end: [0., 0.45, 0.0]
        rpy: [0, 0, 0]
        num_models_x: 4
        num_models_y: 1
      pulley_part_blue:
        xyz_start: [1.5, -0.45, 0.0]
        xyz_end: [-0.5, -0.45, 0.0]
        rpy: [0, 0, 0]
        num_models_x: 4
        num_models_y: 1
```

- ☐ Robot can re-grasp parts from anywhere in the tray during the faulty gripper challenge.
- ☐ System can handle mixed parts on the conveyor belt. Do not assume you will only have one type of parts on the conveyor belt. The belt speed will be the same as in previous assignments.
- ☐ Sensor blackout will have a different time duration in the final trial file.
- ☐ Moving obstacles will be spawned in different aisles in the final trial file. The idle time and the velocity of moving obstacles will be the same as in previous assignments. Parts will be spawned on a different back row shelf (5, 8, or 11).
- ☐ Configuration of shelves will be different from RWA-5.
- ☐ Fix wrong part poses in the kit tray.

Agility Challenges

Below is a summary of agility challenges you may face in the final trial file.

- Faulty part: There are faulty parts in the workcell. Each time you place a part in the tray you need to query the quality control sensor above that AGV to see if a part is faulty. If it is faulty, discard the part immediately and get a new one. You should have enough parts in the workcell to complete the order(s).
- Faulty gripper: One or multiple parts may drop earlier on the tray while the robot is trying to place the part(s). Therefore, after each **PlacePart** action, use the logical camera located above that tray to check the pose of the part. If the part is not located where it is supposed to be then re-grasp it and re-position it. In this challenge parts will always drop in a tray and not on the floor.
- Flipped part: If parts need to be flipped they will be of type pulley.
- Sensor blackout: During the simulation, an action done by the robot will trigger the sensor blackout challenge. For n simulation seconds, all sensors will stop publishing on their respective Topics. Ideally, your robot should keep working during the blackout.
- New order: Your robot will start building a kit per the first order (**order_0**) requirements. During this process, a new order (**order_1**) will be published on **/ariac/orders**. Your control system needs to stop or finish the current action and must start working on the new order. The new order must be completed and shipped as fast as possible. Once the new order is done, your control system needs to go back to completing **order_0**.
- Human presence: This agility challenge consists of 1 or 2 persons moving on the factory floor. Parts required for the orders will spawn on one or multiple back row shelves (`#5`, `#8`, or `#11`) and on the conveyor belt and/or in bins.

Before Live Demo

- Make sure you are using at least Gazebo 9.14. As mentioned by Pradeep (thanks Pradeep!) in one the discussions. Earlier versions of Gazebo may be buggy.
- Remove development mode. In the launch file you have the following snippet:

```
<node name="ariac_sim" pkg="osrf_gear" type="gear.py"
    args="--development-mode
      $(arg verbose_args)
      $(arg state_logging_args)
      $(arg gui_args)
      $(arg fill_demo_shipment_args)
      --visualize-sensor-views
      -f $(find osrf_gear)/config/sample.yaml
      $(find osrf_gear)/config/sample_user_config.yaml
      " required="true" output="screen" />
```

  - You need to remove the line --development-mode for this assignment, which will automatically run your code in competition mode. The main difference between development and competition modes is the deactivation of some services and topics in competition mode (see Cheats). Competition mode also requires that you install a custom version of Gazebo in your workspace (see Installation). You probably have this package already installed.

    -10 pts for working in development mode.

During Live Demo

- We will provide the final trial file at the beginning of next class. You will then modify your launch file to use this trial file. Next, you will run your program.
- Once you start your program, it is forbidden to interact with it through command lines. Everything should be performed through your program. For

instance, your program should be able to:

  - Start the competition.
  - Read orders.
  - Read sensor data.
  - Do pick and place.
  - Handle all the challenges.
  - Deliver the AGVs.
  - End the competition.

- You have 500 simulation seconds to complete the final trial. You should subscribe to the Topic /clock to get the simulation time. The process time is computed either when you end the competition or when you reach the time limit. If you are approaching the time limit, submit the kit (even if it is not complete) and end the competition.

- Zip and upload your package on Canvas before the deadline.

After Live Demo

- Rename your package as final-group# before compressing it and uploading it to Canvas. Make sure you can run it after you changed the name.

- You have to include a readme.txt which contains instructions on how to run your package. Points will be removed if we cannot build and run your package.

  -5 pts for not providing readme.txt.

- Provide Doxygen documentation of your code. We will inspect your submitted code and a documentation is needed. Include a Doxyfile with your package submission. See `https://www.doxygen.nl/manual/index.html`

  -15 pts for not providing documentation.

- Upload your package (including Doxyfile) by the due date and time. No extension is allowed.