Milestone 4: Navigation and Planning

(Last modified: 12 Sep 2022)

In this milestone (Week 8 and 9), you will need to implement the fruit searching module. In the arena, there will be 10 ArUco markers, 1 redapple, 1 greenapple, 1 orange, 1 mango and 1 capsicum. Since this milestone focuses on testing your navigation and planning algorithms, we will provide the true map for you to simplify the SLAM component. This map will contain:

- Locations of the 10 ArUco markers.
- Locations of 3 out of 5 fruits. The locations of the fruits to be searched are known, and the other two fruits will be considered as obstacles with unknown locations.

M4 task: Given a random list of 3 fruits contained in `search_list.txt`, your task is to autonomously navigate to the list of given fruits *in order*, while avoiding obstacles along the way. The robot should stop within 0.5m radius of the target for 3 seconds before moving onto the next target.

To achieve the M4 task, this fruit searching module can be divided into 3 components. These components build on top of each other, so we highly recommend you to complete these components in the following order:

1. Waypoint navigation
2. Path planning (known map)
3. Path planning (partially known map)

The `auto_fruit_search.py` is provided to you as a skeleton code to get you started. Please note that this skeleton code is only a general guide, you can rewrite the whole script and/or integrate it with `operate.py` if you want. You may also submit multiple scripts/variants of this code for the different marking levels you wish to attempt as described in the marking scheme.

Important notes:

- There will be a penalty for every object that the robot collides into.
- You are allowed to use the true map to simplify this milestone (you may choose not to use it), however please be aware that in future milestones you will need to generate your own map.
- The `M4_true_map.txt` and `search_list.txt` provided are just an example. They will be different during your demonstration in Week 10.

Waypoint navigation (Week 8)

As a starting point to complete the M4 task, you will need to develop a module to accept a list of waypoints (coordinates) as input to navigate the robot to the goal position. If you wish to demonstrate this component alone, you may manually enter the target waypoints. Otherwise, this module will take the output of the path planning module as input.

Key functions to complete:

1. Control the robot to drive towards a waypoint at auto_fruit_search.py line 108-124
2. Estimate the robot pose at auto_fruit_search.py line 130-136
    - We strongly recommend you to use your SLAM code from M2 to estimate the robot's pose. If you choose to do so, you will have to copy the necessary SLAM related functions and modules from M2.

General tips:

- Although it is possible to estimate the robot's pose with odometry (a much simpler implementation), we HIGHLY RECOMMEND you to correct the robot's pose after every waypoint navigation step with information from the SLAM module (M2).
- If you wish to demonstrate this component alone, you may implement one of the following to assist you with the demo:
    - *Command line waypoint input*: a command line interface to continuously take waypoints as inputs so that you can guide the robot to get to the goal manually.
    - *GUI waypoint input*: a visual-based waypoints selector (with OpenCV or Pygame), so that you can visually select/click where the robot should go instead of estimating numerical values of the goal coordinates. This will require more work.

Path planning with a known map (Week 9)

Once you have completed the waypoint navigation function, you can implement a path planning algorithm to allow the robot to generate a path (a set of waypoints) and autonomously navigate to the goal. This function should only take the true map, where the locations of all the objects (10 ArUco markers and 3 types of fruit) in the arena are stored as input. It is best to have the file path of the map as an input argument for the Python script. Once the Python script has started, you can enter no more than one command for the robot to start its task. For this component, we assume the positions of all the objects in the arena are known, so that the robot should only need to perform the path planning algorithm once. You may use any path planning algorithm you like, such as Rapidly-exploring Random Trees (RRT) or A*.

Path planning with a partially known map (Week 9)

In this component, we will introduce two types of fruit at unknown locations (unknown to the robot) in the arena. Building on top of the components described above, you will need to include:

1. A fruit detector (implemented in M3) to detect and estimate the location of the two additional fruits, whose locations are initially unknown to the robot.
2. Re-plan the robot's trajectory once an obstacle (new fruit) has been detected.

Marking schemes

We have divided M4 into 3 levels based on the 3 main components which you could attempt. The levels are:

1. Semi-auto navigation with manual inputs (60pts)
2. Autonomous navigation with a known map (20pts)
3. Autonomous navigation with a partially known map (20pts)

The following conditions applies:

- Penalty of -2pts for each fruit that the robot collides with
- Penalty of -5pts for each ArUco marker that the robot collides with
- Penalty of -5pts each time the robot goes out of the boundary (+/-1.5m from the origin, in both the x- and y-axis)

The true map and a list of 3 target fruits will be given to you in the beginning of your live demo marking, then you will have up to 10 minutes to perform the demonstration. If you choose to demonstrate Level 2 and successfully complete it, you will automatically get the points for Level 1 as well. This also applies for Level 3. Therefore, you can choose to demonstrate the level you are confident with. You may demonstrate any level as many times as you want, within the given 10-minute timeframe.

Level 1: Semi-auto navigation (60 pts)

For this level, the arena will have 10 ArUco markers, 1 redapple, 1 greenapple, 1 orange, 1 mango and 1 capsicum. The locations of all 10 markers and 5 fruits will be given to you in the true map (you should have the file name as an input argument to run the script).

You are not allowed to teleoperate the robot. You can only enter coordinate of the waypoints as input, or you may choose to do so via a GUI. You can input as many waypoints as you want to get to the targets.

You will get 20pts for each target fruit the robot can reach. The robot should be within 0.5m of the target and stop for 3s (timing does not need to be exact, as long as the robot stops) to indicate that it has found a target fruit before moving onto the next target. You will get 0pt for that target if it is not within 0.5m of the target. You should confirm with the demonstrator to check whether the robot is close enough to the target before moving on.

Level 2: Autonomous navigation with a known map (20pts)

For this level, the arena will have 10 ArUco markers and only 3 types of target fruits. The locations of all 10 markers and 3 fruits will be given to you in the true map (you should have the file name as an input argument to run the script). You are only allowed to enter a single command and the robot should perform the task autonomously.

You will get 8pts for reaching the first target fruit, and 6pts each for reaching the subsequent fruit. The robot should be within 0.5m of the target and stop for 3s (timing does not need to be exact, as long as the robot stops) to indicate that it has found a target fruit before moving onto the next target. You will get 0pt for that target if it is not within 0.5m of the target.

Level 3: Autonomous navigation with a partially known map (20pts)

For this level, the arena will have 10 ArUco markers, 1 redapple, 1 greenapple, 1 orange, 1 mango and 1 capsicum. The locations of all 10 markers and only 3 types of target fruit will be given to you in the true map (you should have the file name as an input argument to run the script). The locations of the other 2 types of fruit will not be provided and you will have to rely on your fruit detection method to avoid colliding with them. You are only allowed to enter a single command and the robot should perform the task autonomously.

You will get 8pts for reaching the first target fruit, and 6pts each for reaching the subsequent fruit. The robot should be within 0.5m of the target and stop for 3s (timing does not need to be exact, as long as the robot stops) to indicate that it has found a target fruit before moving onto the next target. You will get 0pt for that target if it is not within 0.5m of the target.