

# Section 7: Frontier Exploration [CGOE]

## Overview

The goals for this section:

1. Run Frontier Exploration on the TurtleBot.
2. Use the image classifier from Section 6 to perform Frontier Exploration until a stop sign is found.
3. This section uses the heading controller, AStar planner, image classifier, and frontier exploration code that you have been developing throughout the quarter. It is a final test of your autonomy stack! Congratulations on making it this far 😊

## Setup

**Task 1.1 — Cleanup.** Remove directory `~/autonomy_ws` if it exists. This may be leftover from a previous section.

**Task 1.2 — Directories.** Create the base directory structure for your group's ROS workspace, `~/autonomy_ws/src`.

**Task 1.3 — Clone and Branch.** Clone your group's GitHub repo to the `src/` directory, and create a new branch called `s7_inperson`.

**Task 1.4 — Copy homework code.** We will be adding the code you wrote in HW4 to your autonomy stack. Copy your frontier exploration node file to the folder `~/autonomy_ws/src/<group_repo>/autonomy_repo/scripts` and the launch file to the folder `~/autonomy_ws/src/<group_repo>/autonomy_repo/launch` on the AA274a laptop.

**Task 1.5 — Build.** Update the autonomy repo `CMakeLists.txt` file, and make sure your node is executable. Navigate back to the root of your workspace, `~/autonomy_ws/`, and build your ROS workspace with `colcon build`.

# Frontier Exploration (Simulation)

**Task 2.1** — [Launch simulator](#). Firstly, we will test out your HW4 code before you can test it out on the actual robot. Start your TurtleBot simulator by running:

```
Shell
ros2 launch asl_tb3_sim root.launch.py
```

**Task 2.2** - Launch frontier exploration. In a new terminal, source your workspace, and then launch your frontier exploration:.

```
Shell
source ~/autonomy_ws/install/setup.bash ros2 launch autonomy_repo
<your_frontier_exploration_launch_file>
```

- **Checkpoint — Call a CA to show that your TurtleBot in Gazebo performs Frontier Exploration.**

## Frontier Exploration + Object Detection

We will now integrate frontier exploration with object detection. We will update a copy of the frontier exploration script and launch file for this purpose.

**Task 3.1** - Create a copy of your frontier exploration script in the `scripts/` directory. In this node, create a subscriber with a callback function that listens to the `/detector_bool` topic and finishes the exploration when a stop sign is detected.

**Task 3.2** - Create a copy of your frontier exploration launch file in the `launch/` directory. Add the node you created in Task 4.1 into your launch file.

**Task 3.3** - Update the autonomy repo `CMakeLists.txt` file and build the `~/autonomy_ws` workspace.

**Task 3.4** - We would be using the same ROS bag file from last section. You can find it [here](#). Download the folder and move it to `~/autonomy_ws/src/autonomy_repo`, if you haven't already.

**Task 3.5** - Run `root.launch.py`, and then run the launch file you created in Task 4.2:

None

```
ros2 launch autonomy_repo <launch_file>
```

**Task 3.6** - In the middle of your exploration, run the bag file by:

None

```
# Run this from ~/autonomy_ws/src/autonomy_repo
```

```
ros2 bag play section6_v2/section6_v2_0.db3
```

The robot should stop for a few seconds, then continue to finish its exploration after it detects a stop sign.

- **Checkpoint — Call a CA to show that your TurtleBot performs Frontier Exploration until it views a stop sign.**

## Clean Up

Clean up the workstation.

1. Push your code (`add`, `commit`, and `push`) to your `s5_inperson` branch!
2. Create a PR either from the GitHub web page or by running `gh pr create` in your terminal.
3. Verify that you've pushed correctly by checking if the code you wrote is in the repository on [GitHub.com](https://github.com)
4. Delete the `~/autonomy_ws/`.
5. Log out of GitHub on the workstation.
6. Close all terminals and windows!
7. Shutdown laptop

- **Checkpoint — Call a CA over to sign you out!**