EN613 Final Project Template

# Overview

In this zip file you will find all the files you need to get started on the final project. This template includes the following two folders

1. robot\_spawner\_pkg

This package includes an example world file, model files, and ros2 launch files for launching gazebo and spawning a basic robot. The current robot model uses a differential drive plugin for steering and publishes laser scan messages from a 2D laser range finder.

1. robot\_control\_pkg

This package includes a placeholder controller node and placeholder estimator node. The placeholder controller node subscribes to the estimated state and publishes a desired velocity message, currently all zeros. The placeholder estimator node accepts the laser scan message and odometry message and publishes a current estimated state message, currently all zeros. These nodes

# How to Run

1. Create a workspace

The first step is to download the final\_project\_template.zip file and create a ROS2 workspace. Use the following commands as originally described in Assignment 1.

mkdir -p ~/final\_ws/src  
cp ~/Downloads/final\_project\_template.zip ~/final\_ws/src/

cd ~/final\_ws/src

unzip final\_project\_template.zip

cd ..  
colcon build –symlink-install

1. Launch Gazebo using the robot\_spawner\_pkg

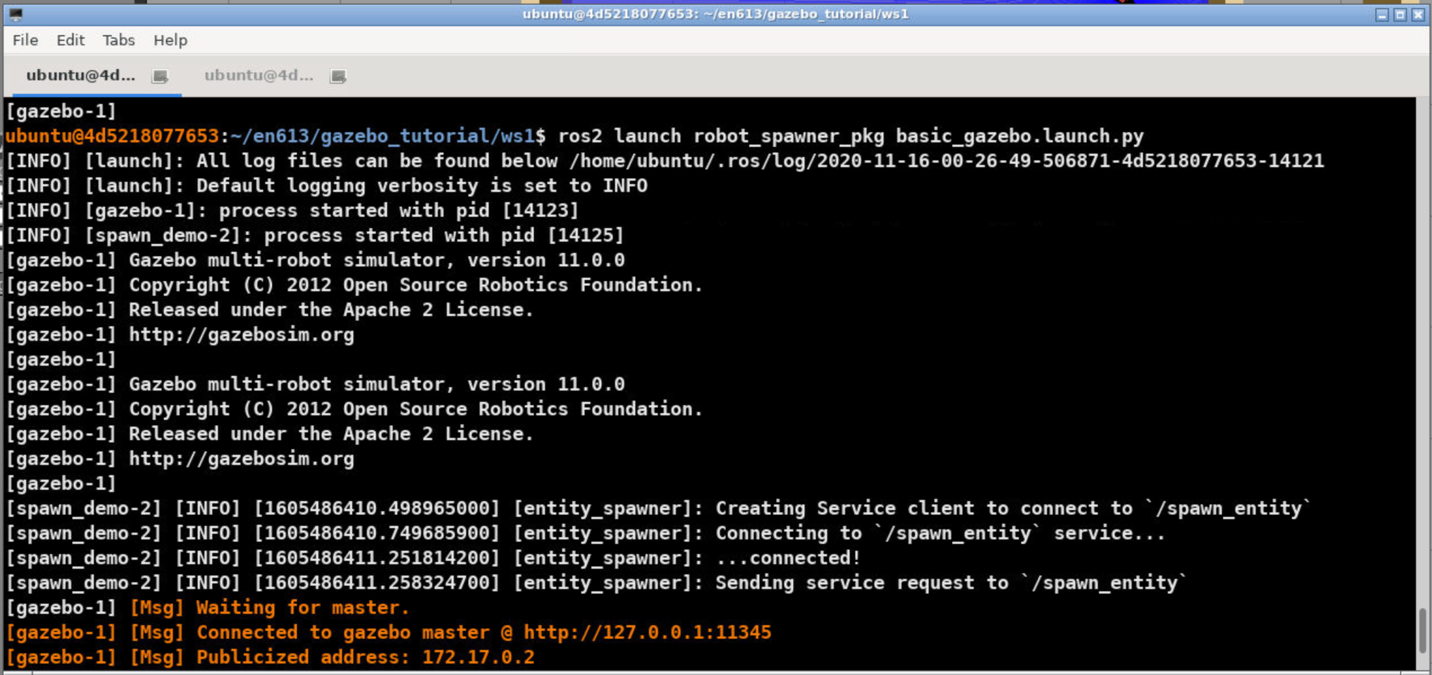
The robot\_spawner\_pkg includes a launch file for starting gazebo with the ROS2 plugins already enabled and then spawning the BasicBot inside the gazebo simulation. To test this run the following command.

cd ~/final\_ws/

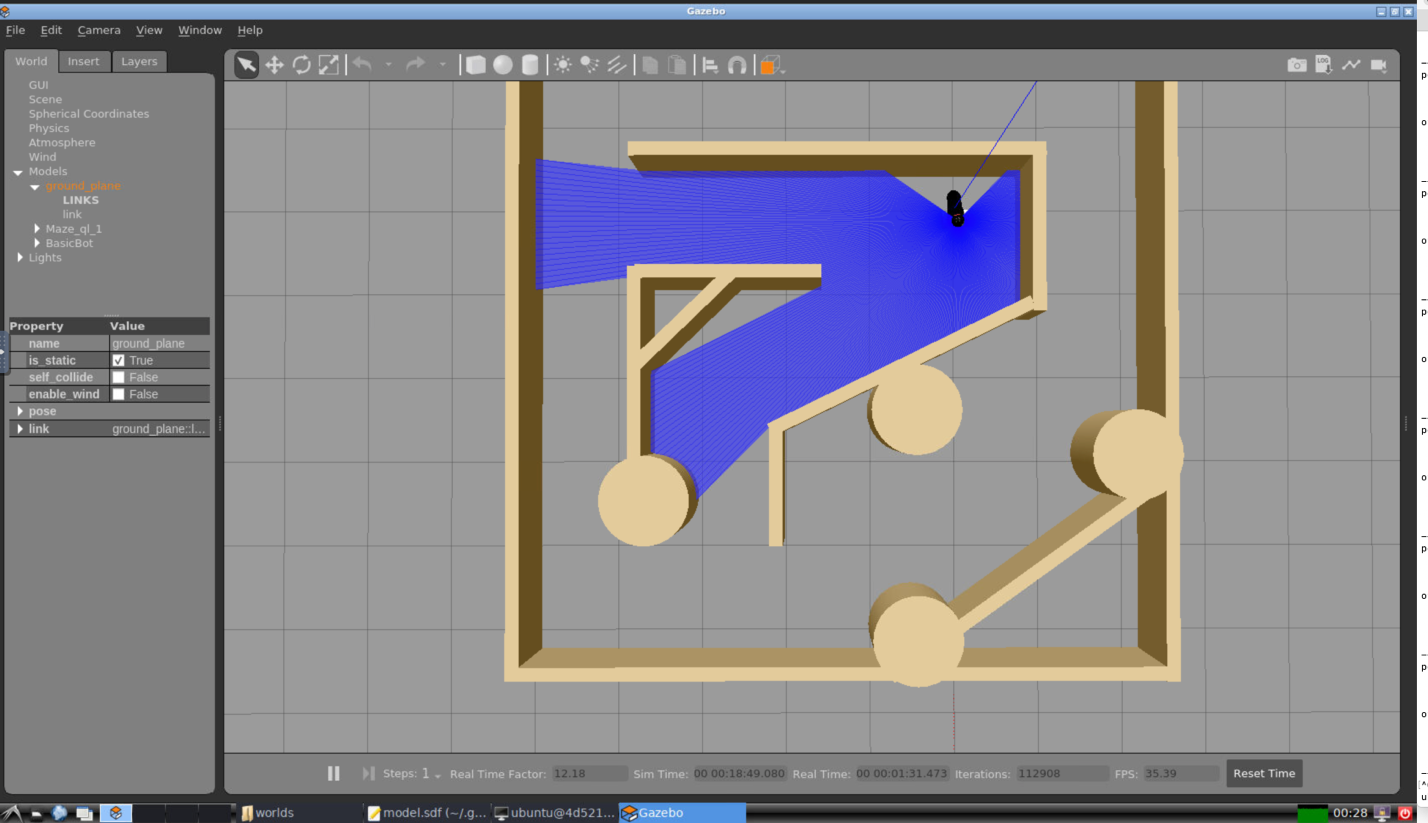
source install/setup.bash

Ros2 launch robot\_spawner\_pkg basic\_gazebo.launch.py

The following output should appear in your terminal.



Then you should see the following window pop up.

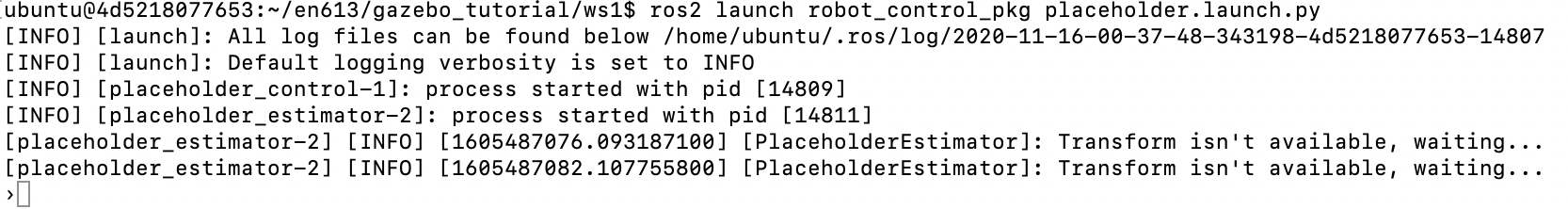


1. Start the placeholder controller and estimator

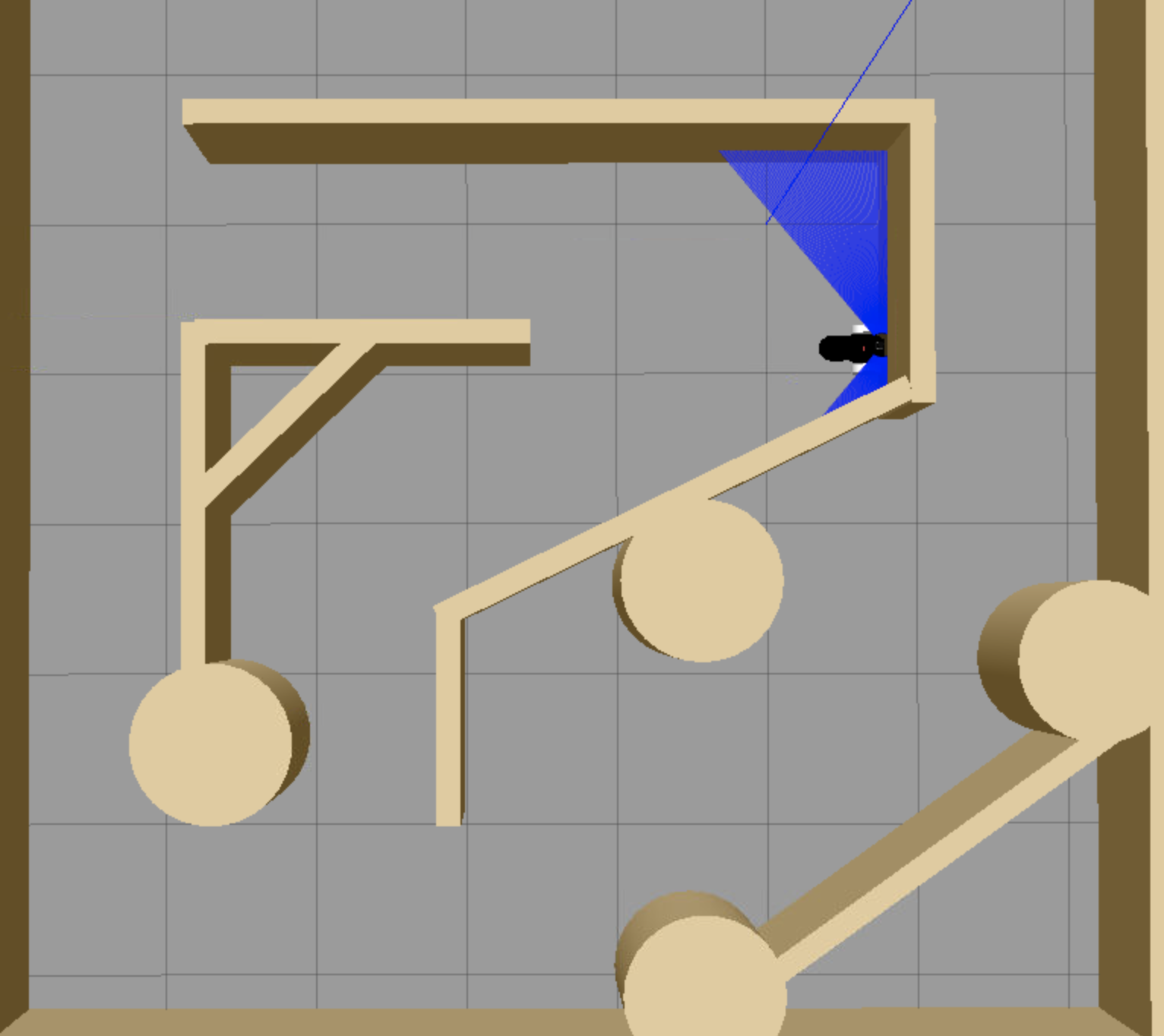
Now start the placeholder controller and estimator nodes. These nodes do not provide any particular functionality but provide examples of how to subscribe to the output of the gazebo simulation and publish commands to control the robot. Use the following command to launch these placeholder nodes.

ros2 launch robot\_control\_pkg placeholder.launch.py

You should see the following output from the terminal.



Then after several seconds the robot should begin moving slowly forward until it collides with a wall. As shown below.

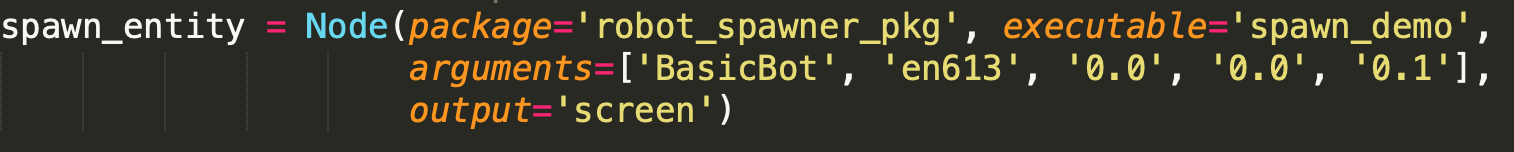


# Robot\_spawner\_pkg

This package includes all of the model files and launch files necessary to start gazebo and spawn the basic robot model. This section will briefly address the important content of this package.

## Launch

This directory contains the **basic\_gazebo.launch.py** file which is used to start gazebo and execute the **spawn\_demo.py** file. You can change location that the robot is spawned at by editing the arguments that are passed to the **spawn\_entity** object at line 49.

****

The first two arguments passed to the nodes are the name of the robot and the namespace which will be prepended to all published ros topics that are associated with the robot. The last three numbers are the x,y, and z positions of the robot’s spawning location.

## Models

This directory includes the models for both the robot [basic\_robot] and the maze object [Maze\_ql\_1]. In these directory you can find a **model.config** file that provides a high level description of what the folder contains and a **model.sdf** that actually defines the model itself. It is recommended that you experiment with the basic\_robot/model.sdf file to make it more efficient and adjust it to your tastes.

## Robot\_spawn\_pkg

This folder contains the spawn\_demo.py file which is the executable for actually spawning a robot in gazebo. This file is hardcoded to spawn the model in models/basic\_robot/ but accepts the following command line arguments.

1. Robot name
2. Robot namespace
3. X position
4. Y position
5. Z position

## Worlds

This folder contains the **maze.world** file. Another type of SDF file that is read by gazebo at launch and used to set all the environmental parameters and spawn entities.

# Robot\_control\_pkg

The robot\_control\_pkg contains example files for creating a controller node and estimator node. These files do not do anything currently except drive the robot very slowly forward. It is up to you to populate this package with functional components that will allow it to drive through a maze.

## Launch

This folder contains the placeholder.launch.py file that will launch both the placeholder controller and placeholder estimator nodes.

## Robot\_control\_pkg

This file contains source code for the placeholder controller and estimator.

### placeholder\_estimator

Currently all this node does is subscribe to the odom and scan messages published by gazebo. Then publishes the transform between odom and the robot’s chassis frame as an estimated state. It publishes this state every time it receives an odom message.

### placeholder\_controller.

Currently all this node does is subscribe to the estimated state message and publishes a twist message to the cmd\_vel topic. To make this function properly it will also need to either subscribe to a goal pose message or receive a goal location as a command line argument.

# Initial Steps

The files in this package should provide you with a good start towards building your own robotic control system. Here are a few things you can do to start working with these packages.

## Fix the sensor on the basic robot

Currently the laser sensor on the robot is partially obscured by the robot body. Causing some of the laser scans to always return 0.0. Try to stop this from occurring. Either by editing the sensor to either reduce the field of vision or altering its position on the robot.

## Update the controller to accept a goal

Currently the controller doesn’t have any knowledge of where to go. Try writing a new node that publishes the goal position and add it to the launch file. Then update the controller code to subscribe to this topic. Finally see if you can cause the vehicle to drive in a straight line towards this goal.