Progress Report 1
Gazebo simulations of contract-based design for an automated valet parking system

SURF student: Tom Andersson

Mentor: Richard M. Murray

Co-mentor: Josefine Graebener

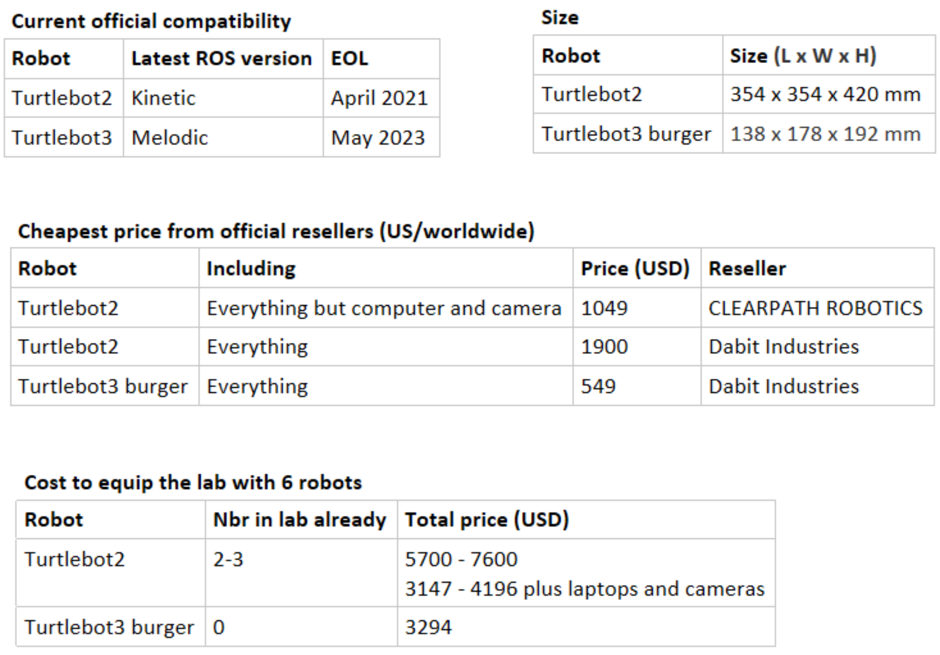
June 29, 2020

Week 1

I spent this week learning about the current state of the project by going through the code, one file at a time, by having meetings with Tung and Josefine and by sketching up a simplified flowchart of the major parts of the system. Due to the size of the flowchart I have chosen to not include it here but it can be found under documentation in the branch gazebo_ros. Note that the flowchart was made with learning rather than presentation in focus. In parallel with this I downloaded Ubuntu, ROS, Python, Git etc to my computer.

Week 2

I started this week by making a comparison between using turtlebot2 and turtlebot3 burger for the simulations. This comparison was made to optimise for a small scale test at a later point in time. The collected data can be found in figure 1. This comparison shows that turtlebot3 burger is a good choice when it comes to price, official compatibility and size (a smaller size reduces the size of the physical area needed during a small scale test). In parallel with this comparison I was learning about the Gazebo simulator buy going through tutorials and running demos with turtlebots in Gazebo.



Current official compatibility

Robot	Latest ROS version	EOL
Turtlebot2	Kinetic	April 2021
Turtlebot3	Melodic	May 2023

Size

Robot	Size (L x W x H)
Turtlebot2	354 x 354 x 420 mm
Turtlebot3 burger	138 x 178 x 192 mm

Cheapest price from official resellers (US/worldwide)

Robot	Including	Price (USD)	Reseller
Turtlebot2	Everything but computer and camera	1049	CLEARPATH ROBOTICS
Turtlebot2	Everything	1900	Dabit Industries
Turtlebot3 burger	Everything	549	Dabit Industries

Cost to equip the lab with 6 robots

Robot	Nbr in lab already	Total price (USD)
Turtlebot2	2-3	5700 - 7600 3147 - 4196 plus laptops and cameras
Turtlebot3 burger	0	3294

Figure 1: A comparison between turtlebot2 and turtlebot3 burger.

With the help from these demos i constructed a launch file for the turtlebot3, one_robot.launch, one launch file for launching multiple turtlebots at specific positions, robots.launch, and one main launch file for launching the turtlebots in an empty world, main.launch. These functions can be found under gazebo/launch in the gazebo_ros branch. Figure 2 shows 2 turtlebots launched in an empty world.

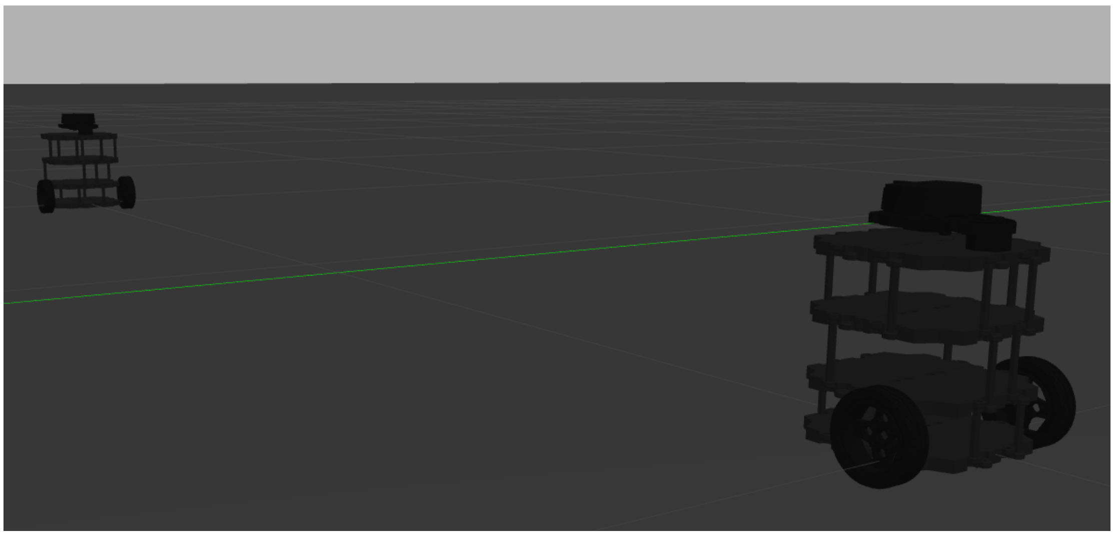


Figure 2: 2 turtlebots launched in an empty world.

The next step was to control the turtlebots through ROS. The turtlebots can be controlled by setting linear and rotational velocity. This data shall be sent in a Twist-object as a ROS-topic. To keep going the turtlebots need to continue to receive new data, if the data stops coming the turtlebot will stop. The python program robot_commander.py creates the Twist-objects and sends this data at a given frequency, currently set to 100Hz. This file subscribes to a ROS-topic of the type Float32MultiArray called robot_vel. The array sent in this topic tells a robot number, a linear velocity and a rotational velocity. As soon as a new topic is received robot_commander will arrange that data into a Twist-object and then continue to publish the new data. The file test_publisher was written to test this functionality. It asks the user to choose robot number, linear velocity and rotational velocity. Thereafter, the given data is published so that robot_commander receives it. The described ROS-structure is shown in figure 3. One can choose how many robots to simulate by changing the variable nbr_of_robots in global_vars.py within the folder variables. Unfortunately I have yet not figured out how to make a dynamic launch file so one also needs to put in change the number of robots in the robots.launch file. How to run this is described in the README file of the branch gazebo_ros.

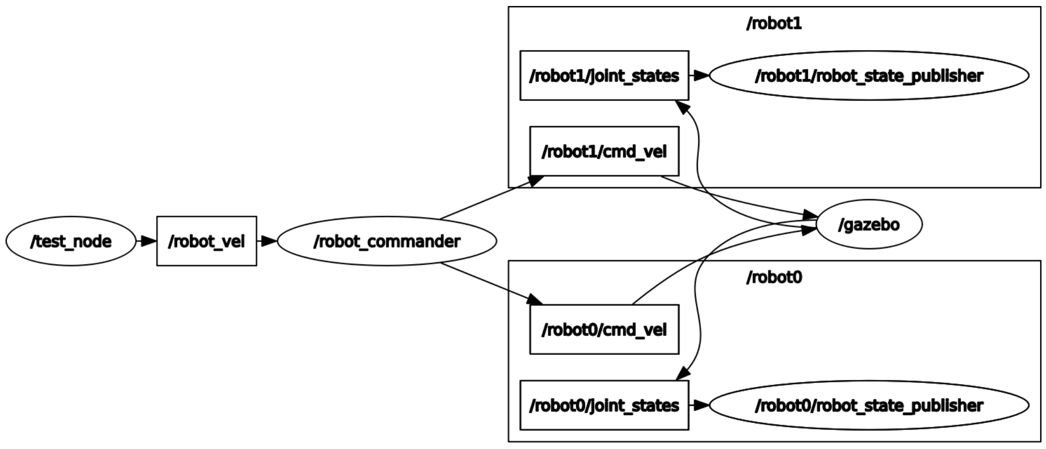


Figure 3: The current ROS structure.

Upcoming weeks

During the upcoming 2 weeks, June 29 to July 12, my goals are to implement the functionality to read the state, i.e linear velocity, rotational velocity and position, of the turtlebots in the simulations and to build up the valet parking environment in a Gazebo world.