ROS Project : Auto-race with Turtlebot3

Group : Camille Tagilone, Walid Brahim

Module : Robotics engineering

Contents :

1 - Introduction

The M2 Robotics project would tie all the tutorials and classes both take at the university campus and the construct platform.

The topics seen within the semester are as follows :

- Python3 for robotics
- ROS basics
- Turtlebot3
- ROS perception

A a summary, the knowledge obtained from these courses with the aid of the internet, the members of the group felt comfortable with the goals and objectives of the project, although multiple obstacles were met.

Lab works aided us in asking the questions, as well as regarding the obstacles met, to progress in the right direction.

Here are the examples of the work done during the laboratory sessions :

- Camera launch and manipulation
- Camera Calibration
- Localisation and Mapping
- Lane tracking
- Obstacle avoidance

2 - Goals and Planning

Goals :

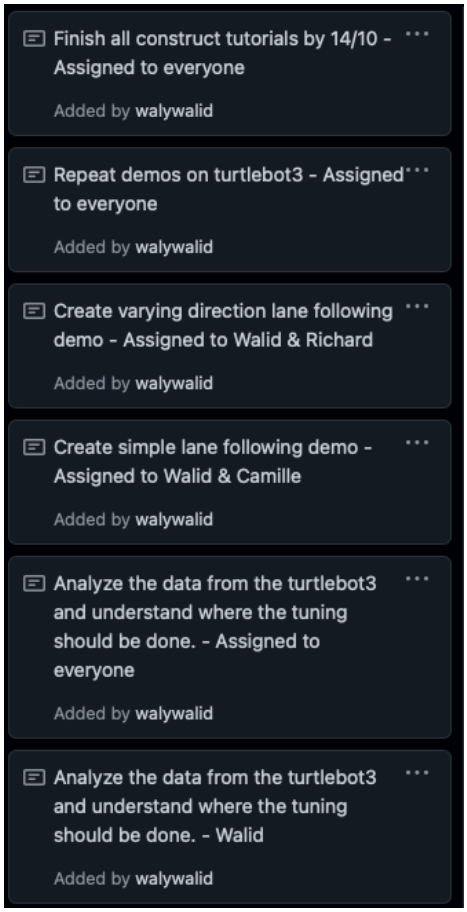
The ROS project demanded to implement all that we have seen throughout the semester to be able to configure the turtlebot3 and its sensors in order to go through the Auto-race track (seen in fig.).

The auto-race track consists of varying u-turns, sharp turns, tunnels, and traffic lights.

To program the turtlebot3 and the packages from the auto-race e-manual to go around the track, where the camera would be used to navigate and control the what the velocities should be for each wheel.

Planning :

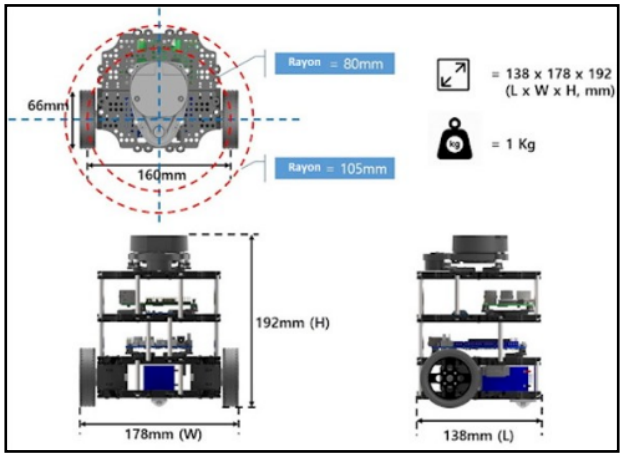
As for the planning of the tasks and assigning who should take it on, the following figure on the right shows how we divided the work throughout the semester.



3 - Turtlebot3 & ROS

Turtlebot3 components :

- 360 degree LiDAR
- Single board computer (Raspberry Pi 3)
- 2 servo-powered wheels (top speed 0.22 m/s)
- Raspi RGB camera



Connecting to :

A common way to control the turtebot3 is to remotely connect to the workstation.

This can be easily done with Ubuntu by using the SSH build-in command, but in order to enable the remote control, you need to modify the bashrc file of the workstation (often located in the Home folder).

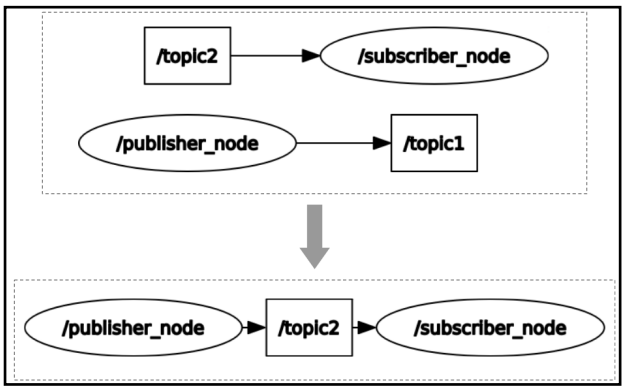
ROS nodes, topics, and messages :

Nodes :

Nodes in ROS are known to be processes that execute a task of some sort, often linked with computation. Nodes also have the ability to “speak” to one another through the main mechanism used, “messages”.

This simplifies large tasks such as working with a robot with multiple input sensors (LiDAR, RGB camera etc…) and output servo/hydraulic motors

In the upper half of fig. there are two types of nodes that can be seen, the publisher and the subscriber, which correspond to what they are receiving or communicating.



Topics :

Topics are in most cases specific information given by a node to another, if appropriate use can be seen.

The bottom half of fig. shows how the “/publisher_node” is publishing a node of a certain topic that is subscribed a “/subscriber_node”.

Messages :

Messages in ROS are defined by the data format and type of message.

Some types of messages in ROS ;
- String
- Int8/32/64
- Float64
- Bool

, and many more.

The understanding of how this mechanism works, and how to view, edit the publishing/ subscriber (as well as create new ones), and lastly edit the types of messages within a package are necessary for steps relating image processing and control theory.

Launching, creating, and editing packages :

Packages in ROS are what host all the codes, their relation to a node (or more), datasets and the configuration files.

For instance the turtlebot3 has a package named bringup, “turtlebot3_bringup”, which within its configuration has the following subscribed and published topics ;

Published Topics	Subscribed Topics
battery_state(sensor_msgs/BatteryState)	cmd_level(geometry_msgs/Twist)
cmd_vel_rc100(geometry_msgs/Twist)	motor_power (std_msgs/Bool)
diagnostics(diagnostic_msgs/DiagnosticArray)	reset(std_msgs/Empty)
etc…	sound(turtlebot3_msgs/Sound)

4 - Implementation and results

To see the implementation and results please check our GitHub repository which contains all the videos and bash files at https://github.com/walywalid/ROS_Project

5 - Improvements

The only major obstacle that we stumbled upon is that the turtlebot3 could not (at an acceptable rate) perform correct navigation at the biggest turn on the auto-race track (seen on the photos below).

The turtlebot3 seems to be always turning too early and run off the track, this may be due to the calibration not being done properly or just random error.

We attempted to rewrite the values to give preference for the outside lane (white lane)

6 - References