Robotic Simulation Home Assignment solution - 7.7.23

Ifat rosenberg
e-mail:ifatrosenberg@gmail.com

Simulation description

The simulation implements a four wheel skid driving control robot that receives a geographic point command ([latitude longitude] ; wgs84) from the GUI prompt, than it plans a path including the distance[meter] and the bearing angle[deg] from the current pose of the robot, to the requested geographic point and drives towards it with a distance error<1[meter].

The robot is able to receive a new setpoint over the course of driving to a current setpoint command, in this case the robot will stop the path to the current command after decelerating, calculate the path to the updated target and send a command to the wheels.

The 4 wheeled robot has a controlled driving system that converts distance[meter] and bearing[deg] commands to torque commands [rad/sec] with acceleration and deceleration logic.

The robot receives its full odometry from a simulated IMU that serves as the Robot's main sensor and feedback to the control.

The SERVO control of the robot from the torque command to the actual movement of the robot wheels is implemented in a GAZEBO plug (hinge joints velocity controllers).

The navigation of the robot includes an IMU that is also implemented from a GAZEBO plugin.

Recorded video

The attached recorded video demonstrates the initialization and running of the simulation.

Source code

The source code can be found in the gitHub repository **tractor_sim_ws_repository** in the following link:https://github.com/ifatro/tractor_sim_ws_repository.git

The source code includes:

1. gazebo_tractor_sim_pkg
2. ros_tractor_sim_pkg
3. CMakeLists.txt
4. README.md

In order to run the simulation all 1-4 need to be downloaded under a Workspace/src folder in with the workspace sourced to the bashrc.

Implementation

The simulation is implemented using the following 5 ROS nodes, each node has a different functionality and connections to other nodes:

Node	class file	Brief description of functionality
GUI	main.python	The node receives the robot and target locations (wgs84) and draws them on a geographic map.
robot_path_planning_node 10[Hz]	robot_path_planning.cpp	The node calculates the robot bearing angle and distance to the required target.
robot_navigation_node 100[Hz]	robot_navigation.cpp	The node receives the IMU measurements and extracts the robot full odometry (angular & linear).
robot_control_node 30[Hz]	robot_control.cpp	The node receives the required robot’s path from the path planner node and the robot’s odometry from the robot’s navigation node and calculates the commands to the controller wheels.
sim_ground_truth_node* 10[Hz]	robot_sim_ground_truth.cpp	The node receives the robot ground truth odometry and calculates the robot's true location to be drawn in the GUI app.

*The sim_ground_truth_node is not connected to the robot’s nodes (navigation, control, path planning) and serves only to present the robot’s true location on the GUI geographic map.

The nodes are communicating using ROS publishers and subscribers detailed in the following table

Publishers and subscribers of each node:

Node	Publishers	Subscribers
GUI	• `target_location`	• `robot_location`
robot_path_planning_node	• `robot_target_path`	• `robot_estimated_location` • `in_turn`
robot_navigation_node	• `robot_estimated_location` • `robot_odometry` • `robot_bearing_imu`	• `imu`
robot_control_node	• `left_wheel_2_hinge_joint_velocity_controller/command` • `left_wheel_hinge_joint_velocity_controller/command` • `right_wheel_2_hinge_joint_velocity_controller/command` • `right_wheel_hinge_joint_velocity_controller/command` • `curr_dist` • `cmd_dist` • `time_from_start_driving_ahead` • `in_turn`	• `robot_target_path` • `robot_odometry` • `robot_bearing_imu`
sim_ground_truth_node	• `robot_location`	• `ground_truth/state`

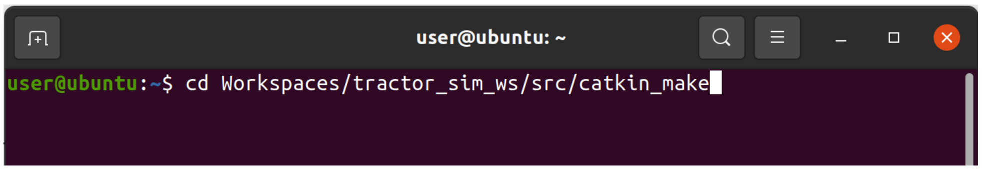
Simulation compilation

The simulation is implemented in a workspace called tractor_sim_ws.
This package includes the gazebo package to launch the GAZEBO simulation, and the ROS package to launch the GUI and the ROS code. both packages are located under the src/ folder:

The packages are called:

- gazebo_tractor_sim_pkg - launchs the GAZEBO simulation
- ros_tractor_sim_pkg - launches the GUI and ROS system.

The compilation command for both packages is:



```
user@ubuntu:~$ cd Workspaces/tractor_sim_ws/src/catkin_make
```

For the compilation there may be additional packages needed to be installed.

Installation of required packages commands:

- ☐ sudo apt install ros-noetic-joint-state-controller
- ☐ sudo apt install ros-noetic-velocity-controllers
- ☐ sudo apt-get install gazebo11-plugin-base
- ☐ sudo apt install ros-noetic-hector-gazebo-plugins
- ☐ sudo apt-get install ros-noetic-navigation

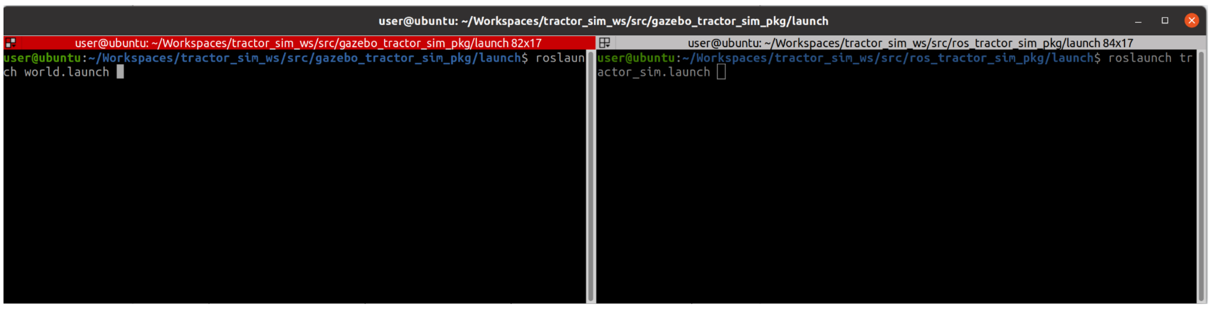
Simulation running

The simulation was developed for Ubuntu 20.04 with ROS noetic.

In order to run the simulation it is needed to launch the couple of the launch files:

1. world.launch - launches the tractor model and the world in the gazebo environment.
2. tractor_sim.launch - launches the ROS noes, parameters and GUI.

Launch commands should be written as in the following image:



user@ubuntu: ~/Workspaces/tractor_sim_ws/src/gazebo_tractor_sim_pkg/launch

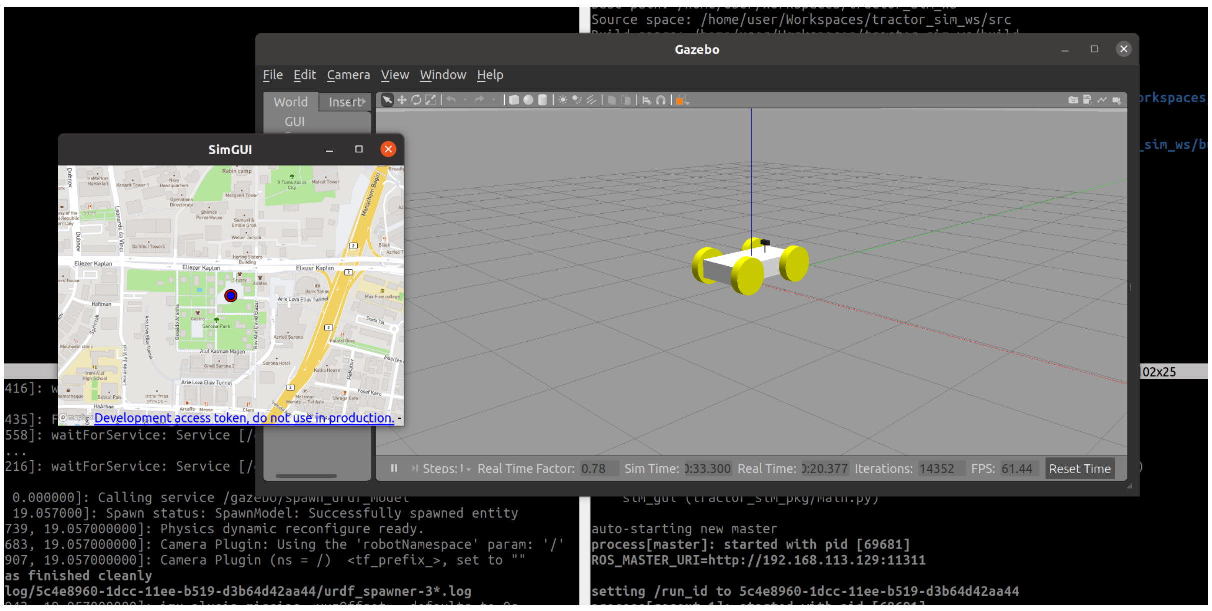
user@ubuntu: ~/Workspaces/tractor_sim_ws/src/gazebo_tractor_sim_pkg/launch 82x17

```
user@ubuntu:~/Workspaces/tractor_sim_ws/src/gazebo_tractor_sim_pkg/launch$ roslaunch world.launch
```

user@ubuntu: ~/Workspaces/tractor_sim_ws/src/ros_tractor_sim_pkg/launch 84x17

```
user@ubuntu:~/Workspaces/tractor_sim_ws/src/ros_tractor_sim_pkg/launch$ roslaunch tractor_sim.launch
```

Then both the GAZEBO and the GUI are opened and the simulation can be run.



```
416]: w
435]: F
558]: waitForService: Service [/
...
216]: waitForService: Service [/

 0.000000]: Calling service /gazebo/spawn_urdf_model
 19.057000]: Spawn status: SpawnModel: Successfully spawned entity
739, 19.057000000]: Physics dynamic reconfigure ready.
683, 19.057000000]: Camera Plugin: Using the 'robotNamespace' param: '/'
907, 19.057000000]: Camera Plugin (ns = /)  <tf_prefix_>, set to ""
as finished cleanly
log/5c4e8960-1dcc-11ee-b519-d3b64d42aa44/urdf_spawner-3*.log
```

```
Source space: /home/user/Workspaces/tractor_sim_ws/src

    sim_gui (tractor_sim_pkg/main.py)

auto-starting new master
process[master]: started with pid [69681]
ROS_MASTER_URI=http://192.168.113.129:11311

setting /run_id to 5c4e8960-1dcc-11ee-b519-d3b64d42aa44
```