ECED 3901

Laboratory 3: Design Task 1

Dead-Reckon a Square

Date Handed Out: January 14, 2026
Code, Simulation, and Live Demonstration Due: January 21, 2026

Desired Outcomes:
Design task 1 will use ROS2 to simulate and experimentally demonstrate the robot dead-reckoning a Square. Your team's simulation, code, and robot performing the task will be graded by a TA.

Prerequisites:

- Completion of Laboratories 1 and 2. You will need your git repository and robot.

Requirements and Constraints:

1) Code must be your team's own work as relates to the challenge.
2) Create a ROS2 node that subscribes to /odom and publishes to /cmd_vel.
3) Prove your program works in the Gazebo simulation environiment and show validation on your own robot.
    - One simulation demo is required for the team.
    - Experimental evidence of the robot performing the task is required.
4) If time is limited, only 3 trial attempts are permitted for demonstration.
5) Git version control must be used; it will be evaluated when code is demonstrated.
6) Most team members should make commits to the repository.
7) The robot must make a 1.0 m square using /cmd_vel and /odom topics.
8) The shape should be completed counterclockwise from the point/origin.
9) Robot shall not travel faster than 0.3 m/s, nor rotate faster than 1.0 rad/s.
10) Points will be awared for accuracy of the shape and the pose at origin upon return.
11) Close-loop control and reading encoder data is required, as opposed to timed open-loop movements.

Suggested first steps:

1. Start your Robot, and open a terminal.

2. Change directory to your team's ROS2 workspace and package.

```
$ cd ~/ros2_ws/src/eced3901
```

3. Create a README.md file, that you will populate with information about your project.

```
$ gedit README.md
```

   In the file, put a few tid-bits of useful information. Authors, team, course, university, date and purpose. You may also want to list the programs directory within the package. "ctrl + o", enter, "ctrl + x" to save and exit.

4. Add the readme file and make sure your repository is all setup before developing.

```
# Get the status, untracked files, etc.
$ git status

# Stage wanted files, here all of them
$ git add .

# Commit
$ git commit -m "Added a README.md to keep track of nodes and programs."

# Check status
$ git status
```

5. Inspect the starting template file **eced3901_dt1.cpp** that was preloaded in your package *src* folder. This is where you will implement your code for DT1.

```
$ gedit src/eced3901_dt1.cpp
```

6. Inspect the **package.xml** file and **CMakeLists.txt** files. They have been modified to ensure the program is compiled and linked. Later, you will add new programs and dependices, so it's worth your time to understand what is going on.

```
$ gedit package.xml
$ gedit CMakeLists.txt
```

7. Compile the code. Every time you change eced3901_dt1.cpp, you need to recompile the code into an executable.

```
$ cd ~/ros2_ws

$ colcon build
```

8. Start Gazebo with our ECED3901 BOT in an empty world. This calls a premade launch file for you, which we will investigate more in future labs.

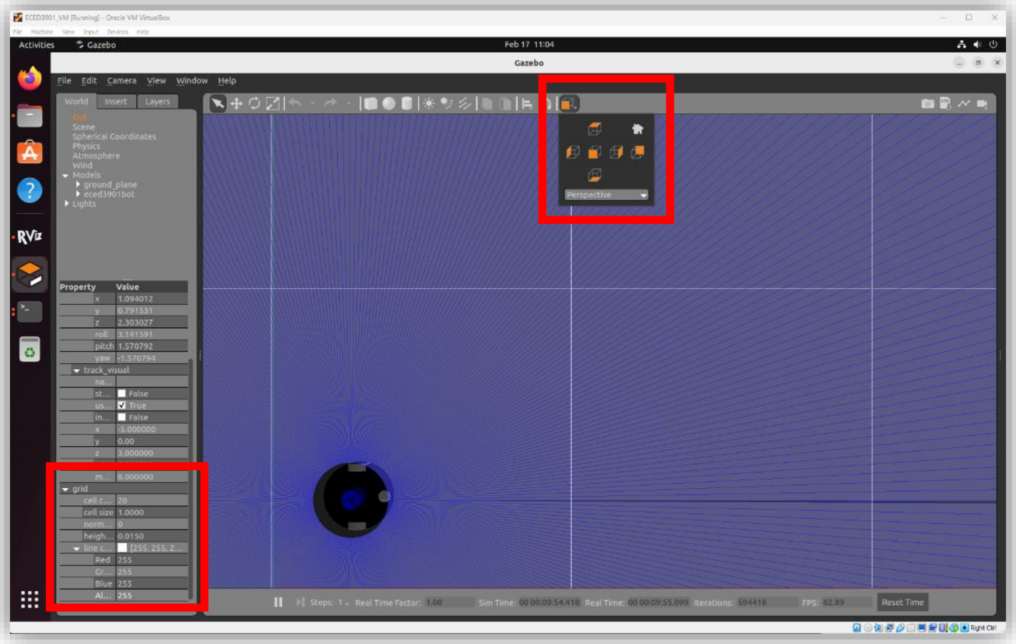
```
$ ros2 launch eced3901 lab3.launch.py
```

9. Run the DT1 node/code another terminal. It should **not** work correctly!

```
$ ros2 run eced3901 dt1
```

10. Design and test your code to fulfill the requirements above and be ready for your field trial next week! Remember to track your changes with git commits and comments!.

Tip: You can change the Camera View from top-down in Gazebo. You can also change the grid size and color for evaluating the performance visually.



Property	Value
x	1.094012
y	0.791531
z	2.303027
roll	3.141591
pitch	1.570792
yaw	-1.570794
track_visual	
na...	
st...	☐ False
us...	✓ True
in...	☐ False
x	-5.000000
y	0.00
z	3.000000
m...	8.000000
grid	
cell c...	20
cell size	1.0000
norm...	0
heigh...	0.0150
line c...	[255, 255, 2...
Red	255
Gr...	255
Blue	255
Al...	255