Assignment 1

The turtlebot

DEADLINE:

- **Winter Session**: November 30th 2025, 23.59 CET

For this first assignment, you will use the environment set up for Exercise 4. Follow these instructions if you did not do it before:

1. Initialize a dedicated workspace
   “’bash
   ```
   mkdir ws_$(group_number)_assignments
   cd ws_$(group_number)_assignments
   mkdir src
   cd src
   git clone https://github.com/PieroSimonet/ir_2526.git
   cd ..
   colcon build
   source install/setup.bash
   ```
   “”
2. Create a dedicated package
   “”
   ```
   cd ~/ws_$(group_number)_assignments/src
   ros2 pkg create --build-type ament_cmake --license Apache-2.0 $(group_number)_assignament_1
   ```
   “”

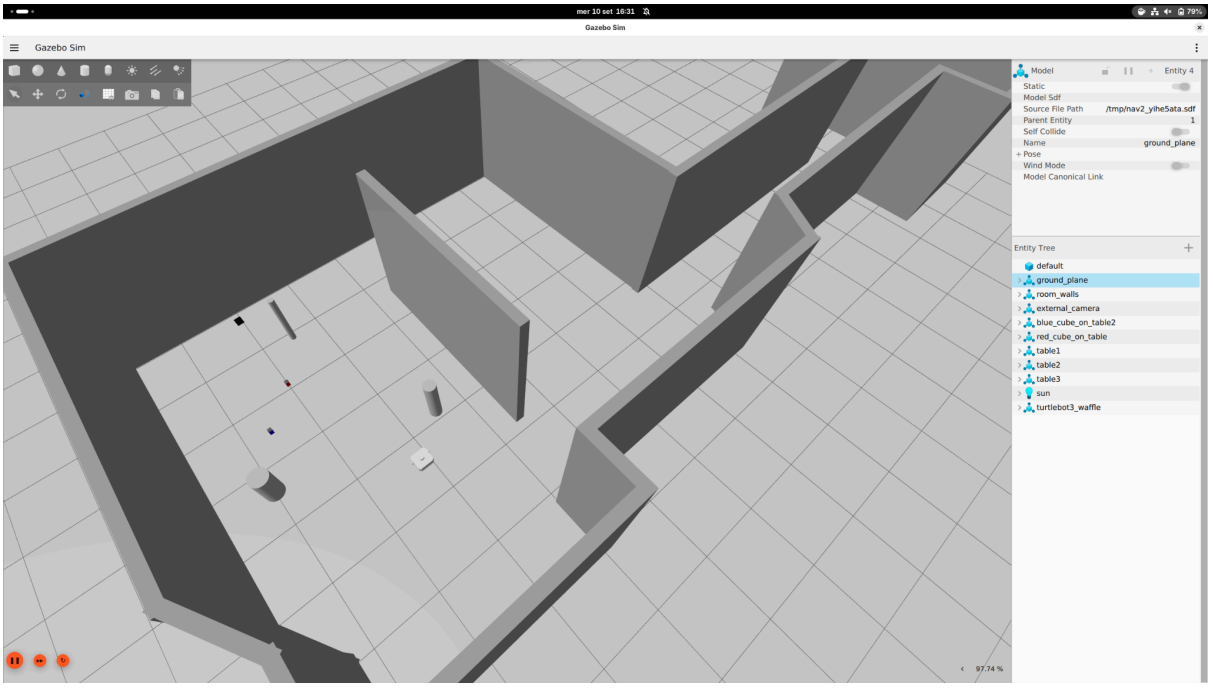
3. **ONLY MODIFY THE FILES IN YOUR PACKAGES, DO NOT EDIT OURS IN ir_2526 !!!!!!! (You will fail the exams otherwise)**

4. You will need to create a git repository for each assignment.

5. Launch the simulation-initialization of the assignment with:
   “’bash
   ```
   ros2 launch ir_launch assignment_1.launch.py
   ```
   “”
6. Tip: Write your own launch file that automatically calls the simulation-initialization launch file.

Assignment

For assignment 1, we ask you to develop a pipeline for the well-known **turtlebot 3**.



At the launch of the simulation, the turtlebot will spawn at the laslab entrance. The goal is to make the robot **move** to a position in the lab between two **AprilTags** visible from a service camera. In that position, the robot will be able to **detect** three cylindrical tables placed somewhere in the room.

The pipeline must then include:

- detection of the apriltag
- navigation to the apriltags (find a good position between the two and reach it without hitting them)
- detection of tables using any available sensor
- return the position of tables relative to odom reference frame

Remember to initialize navigation as shown in class.

You must write your launcher for the whole project. It must include a launch file for apriltag detection node otherwise you won't be able to see them, an example at ~/ir_2526/apriltag_ros/launch/camera_36h11.launch.yml

- THE APRILTAG SIZE IS: 0.050x0.050 [m]

The modularity of the project, the kind of communication between nodes, and the methods used for detection and any extra navigation strategy are all up to you and will be carefully evaluated

Extra Points (+3):

Stop the navigation to goal when the turtlebot enters the corridor and implement your own navigation method sending velocity commands to the robot and detecting the walls with the lidar. Once at the end of it, resume the navigation to the goal.

How to submit (READ ME!)

With your group, prepare the following:

- A **video** of the simulation of the assignment
    - The resolution should be enough to understand what is happening (at least 1080p)
    - From the video, we should understand that you have been able to complete the requests
    - You do not need to do a super complicated setup; a proper screen recording is enough
    - Include any tool/output that may help us understand your solution quickly.

- A **report** of your work (as discussed in class):
    MANDATORY
    - PDF format, max 3 pages, and **include all the names of the participants of the group and relative email addresses**
    - **link to the git repository**
    - **link to video** and any additional material (google drive, remember to share it with the tutors)

    - use the advice given in class to write an interesting report.
    - If you use any third-party material (e.g., algorithms or other resources), make sure to cite the source.
    - The use of content generated by artificial intelligence (AI) in an article (including but not limited to text, figures, images, and code) shall be disclosed in the acknowledgments section. The AI system used shall be identified, and specific sections of the article that use AI-generated content shall be identified and accompanied by a brief explanation regarding the level at which the AI system was used to generate the content.

- A git repository with your code:
    - Only submit the package that contains the solution for the assignment
    - We will use the commit history to track how you split the work
    - DO NOT COMMIT ALL THE CODE AT ONCE. Track your work!
    - DO NOT COMMIT OUR CODE
    - Recall the account and the name of the group participant
    - **A readme** on how to launch your solution
    - Create a private repository on GitHub and add the tutors' accounts:

- https://github.com/PieroSimonet
- https://github.com/AnnaPlt
- https://github.com/MattiaToffanin
- https://github.com/Curtaz

One member of the group submits **only the report of the project** in the moodle dedicated area.

Evaluation:

Code: grade/30
ROS structures, modularity, coherence with the assignment, difficulty of methods implemented, use of external libraries/algorithms, accuracy of the whole solution.

Report: grade/30
Accuracy, fluency, captivating writing, use of tables, figures, diagrams.

Video: grade/30
Presence of visual outputs useful for understanding your solution. Output of sensors, point of view of cameras, planning scenes, the terminal, the feedbacks, ... Coherence between code and video.

final grade = (code + report + video)/3 + extra_points