# **Assignment 4 (10%):**

**Path Planning and SLAM**

DUE DATE: Thursday, April 18th 2024, 11:59 PM ET

**Objectives:**

* Implement A\* path planning algorithm for autonomous navigation in known environments.
* Implement Visual SLAM techniques for real-time mapping and localization in robotic systems.

**Assignment Guidelines:**

* All students in this course must read and meet the expectations described in the [Student Academic Integrity](https://intranet.laurentian.ca/policies/2017.09.19%20-%20Policy%20and%20Procedures%20on%20Academic%20Integrity%20-%20EN.pdf).
* Assignments must follow the programming standards document published on the course website in the D2L. Marks will be taken off if standards are not followed.
* **Submit all necessary files such as .py, .world, etc. AND the associate report in PDF per group**. Name the report PDF based on your group “ID” and the assignment number, exactly as in this example for **assignment x and** **group x**: **CPSC\_5207EL\_Ax\_Gx.pdf**.
* **Do NOT zip the files** that you submit.
* You may submit the assignment multiple times, but only the most recent version will be marked.
* **If applicable:** After the due date and time, a late penalty of 2% per hour, or a portion of an hour, will be applied. After 49 hours, the penalty reaches 100%, and submissions will no longer be accepted. The date and time of the last file submitted will determine the mark for the entire assignment. **These late penalties will not be applied under the circumstances of a cyber incident.**
* We compare all submitted assignments with one another, and pursue academic dishonesty vigorously. **You must complete the Honesty Declaration in the D2L before you will be able to submit your assignment.**

1. **Introduction**

In this assignment, you will extend your previous work with ROS2, Gazebo simulation, OpenCV image processing, and the TurtleBot3 Waffle Pi to implement an autonomous navigation system with enhanced capabilities. Specifically, **you will incorporate A\* path planning for efficient navigation and SLAM for mapping the environment in real-time.**

**Requirements:**

* Completion of Assignment 3 involving the development of a vision module with OpenCV, ROS2 Humble, Gazebo simulator, TurtleBot3 simulation package, and OpenCV Python library.

1. **Task 1: A\* Path Planning on Customized Gazebo World (10%)**

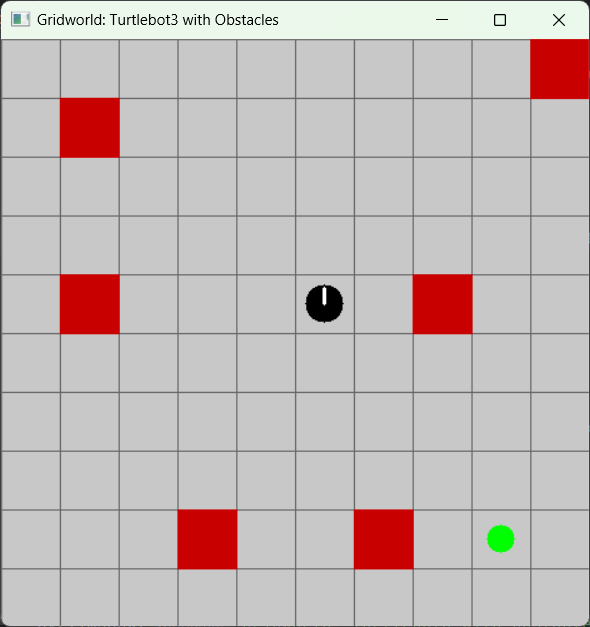
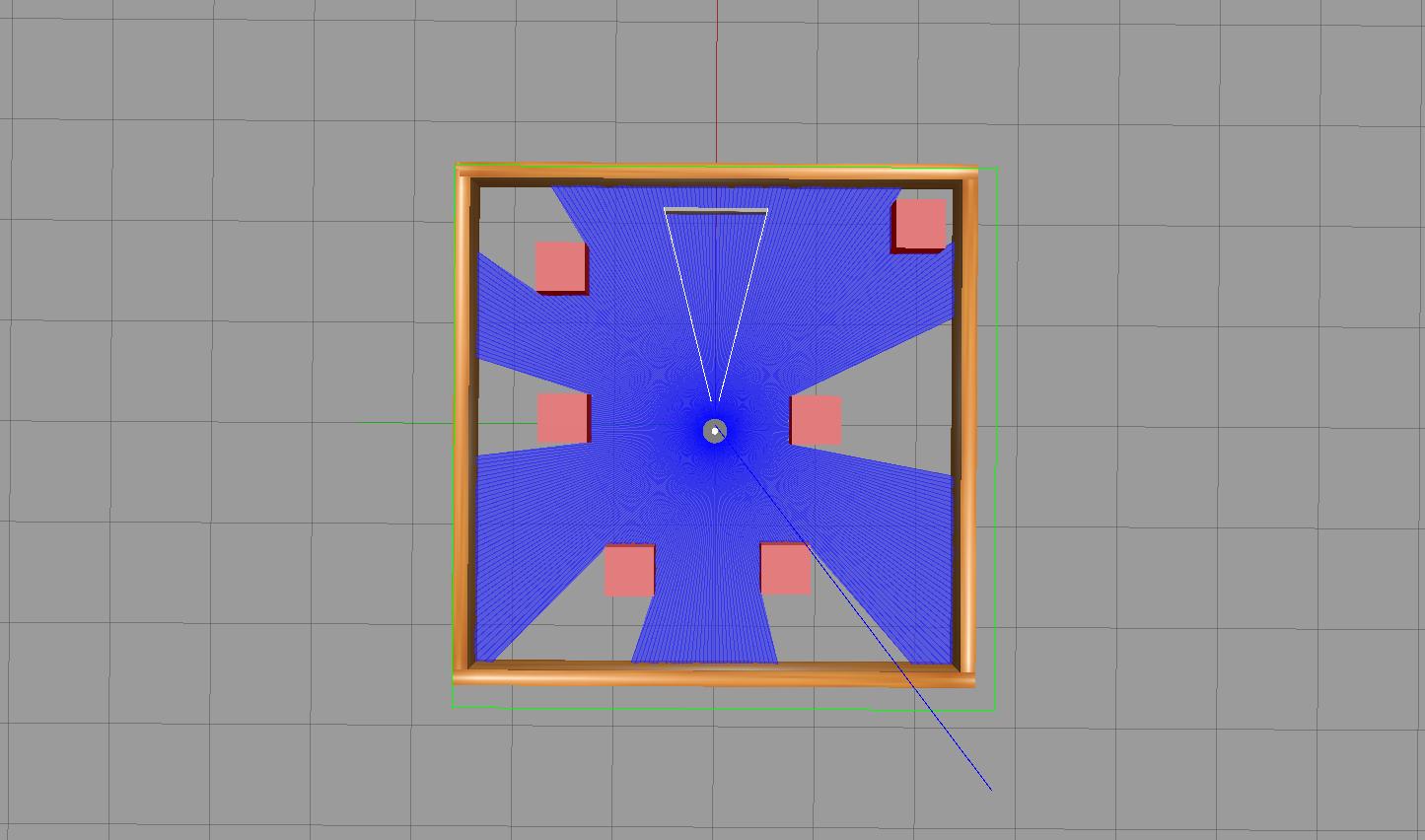


Figure 1

Create a customized version of the turtlebot3\_dqn\_stage1.world (470 cm x 470 cm) by adding multiple red-colored cube obstacles as shown in Figure 1 (left). Add at least six red-colored cube obstacles at various locations within the world. Each cube should have a side length of 47 cm. As shown in Figure 1 (right), the robot's direction is set to 90 degrees, and the white line indicating the direction is drawn accordingly, pointing upwards. You can download the [sample](https://drive.google.com/file/d/1wfI84nClhutOnIc5j4GMpxeSjQ3KEXRw/view?usp=sharing) code to generate the gridworld.

*Note: a small position deviation for the obstacles between the simulation and the gridworld are acceptable (e.g. obstacle at the top right corner in Figure 1).*

Implement an A\* path planning algorithm in Python (based on the above gridworld sample) that generates a shortest collision-free path (a set of motions) from the robot's current position to a goal position (green circle as shown in Figure 1 (right)) while avoiding obstacles. Integrate this algorithm into your autonomous navigation program from Assignment 3.

*Note: for simplicity, you can fix the rotational motion of the robot to 90 degrees.*

Create a custom launch file to initialize your custom Gazebo world and the autonomous navigation system. Your launch file should start the Gazebo simulation with custom\_turtlebot3\_world.world and incorporate the execution of your A\* path planning-based autonomous navigation program (or you can run the Python script as Assignment 3).

1. **Task 2: Visual SLAM Implementation (10%)**

Develop a Visual SLAM (Simultaneous Localization and Mapping) system using ROS2 and OpenCV that enables the TurtleBot3 to create a map of its environment in real-time (No Lidar data should be utilised in the implementation). Follow these requirements:

* Subscribe to the TurtleBot3's camera topic to receive images from its environment (Ref: Assignment 3).
* Implement image processing techniques using OpenCV to detect obstacles and walls (Ref: Assignment 3). Consider using color detection to identify red cubes.
* Use the detected obstacles to update a grid-based map of the environment. Implement SLAM techniques such as occupancy grid mapping to create and update the map (as in Task 1) - discretized probabilistic mapping.
* *Note: Assume new obstacles are added into Gazebo, and the robot will update the map accordingly. The robot's position on the gridworld should be updated from cell to cell.*
* Display the generated map in a separate window (as in Task 1).

1. **Reporting (5%):**

* Write a 2-3 pages report in Arial, 11-point font, single-spaced, detailing your implementation and results for tasks 3 and 4.
* Explanation of the approach taken to customize the Gazebo world and implement A\* path planning for autonomous navigation.
* Description of the VSLAM implementation, including the image processing techniques used for obstacle detection and map generation.
* Discussion of any challenges encountered during the implementation and how they were overcome.
* **Video Demonstration**: Record a 1-minute video showing your TurtleBot3 autonomously navigating the custom world. The video should clearly demonstrate the robot avoiding obstacles and walls. Upload the video to a private YouTube link and include the link in your submission.

**Submission Guidelines:**

1. Submit your source code files, including the ROS2 package containing your autonomous navigation program (python) and **any modified or new scripts for the custom Gazebo world**.
2. Submit your report as a **PDF** file, ensure that all figures, and tables are properly labeled.
3. Your assignment should be self-contained, meaning a person should be able to understand your process and results just by reading your report and going through your code.

**Evaluation Criteria:**

1. A\* Path Planning: Effectiveness and efficiency of the A\* algorithm in generating collision-free paths for autonomous navigation.
2. SLAM Implementation: Accuracy and completeness of the generated map using SLAM techniques.
3. Documentation: Clarity and completeness of the report, including explanations of the implemented algorithms and techniques.
4. **Note: A few best completions of the assignments will be selected to extend their works for a conference submission.**

**Note on Group Contributions and Grading:**

If any group member believes that another member of their group deserves a lower grade due to their contribution level, they are encouraged to address this concern. To formalize this, the group can include an additional section in their report detailing the situation and the proposed grade adjustment, with the consent of all group members. It's essential that all group members agree and provide their consent for any proposed grade changes.

It's always best to communicate openly within your group and seek collaborative solutions. However, if discrepancies in contributions are significant and consensus is achieved, this mechanism ensures fairness in grading.

**Best of luck! Remember, the process and learning are as important as the final results.**