Assignment 1

DEADLINE: 23rd December at 23:59 (CET)

Implement a routine that lets Tiago navigate inside the environment, see Figure 1. The environment comprises two rooms with movable obstacles (i.e. you can move them in Gazebo) and a narrow space. Tiago has to navigate from point *Starting Pose* to *Pose_B* (Figure 1).

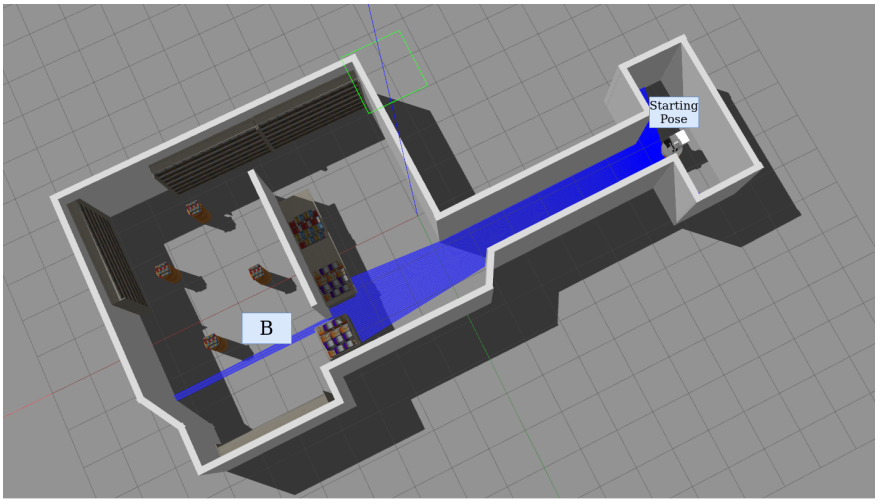

Figure 1

The *Pose_B* must be inserted by the user via the command line, i.e., it is an input from the command line for your code. Once the robot reaches *Pose_B*, it has to detect the obstacles, and print on the screen the position of the movable obstacles, e.g. the cylindrical tables, visible from that pose. (Figure 2). The static obstacles that are part of the map, e.g. the walls, or the shelves, must not be detected as obstacles.

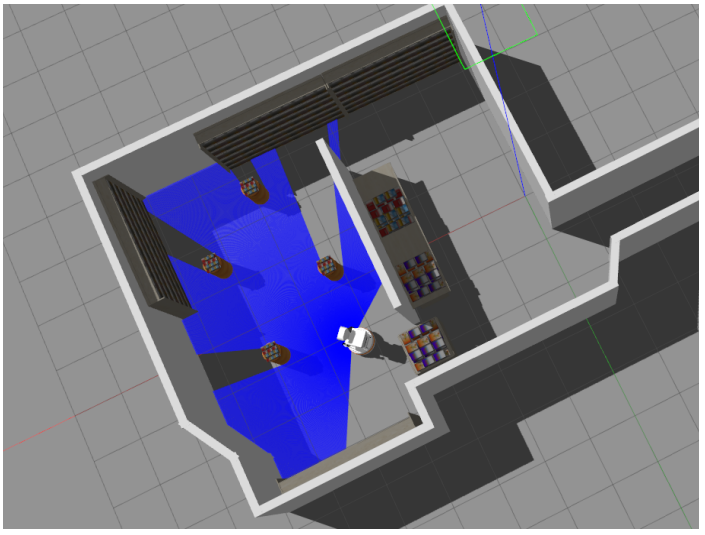

Figure 2

Suggestion: use the laser range finder mounted on the robot (topic /scan).

For the assignment, **you have to create a package** which contains the code to implement.

Your code must be implemented using the following structure:

- The user can input the pose of *Pose_B* by command line;
- Your code **must** be implemented with an Action Client/Server structure:
    - The action client receives the input from the user;
    - The action client calls the action server that executes all the tasks;
    - The action server sends the final list of positions of the obstacles as result to the action client;
    - The action client also implements callbacks to the feedback of the action server, and prints the task's current status in the terminal. The feedback **must** reflect the current status of the robot, e.g.: *the robot is stopped, the robot is moving, the robot started the detection of the obstacles, the detection is finished, etc.*

Extra points

Typically, the ROS Navigation Stack is inefficient in narrow passages. Usually, the programmer has to implement an ad hoc routine to directly process the sensor data (e.g., laser and/or sonar data) and generate velocity commands for the robot, i.e. the programmer creates a **"motion control law"** for the robot. In this assignment, to get extra points, we ask you to implement your control law to navigate through the narrow corridor using the laser data. The routine has to use the laser and send the velocity commands to the topic */cmd_vel* to move the robot without calling the *move_base stack*.

N.B.: Please, specify in the submission in the report that you implemented the extra point part of the assignment.

Instruction to start the homework

Local ROS Installation

Who has a local ROS installation (e.g. Ubuntu or Virtual Machine) is required to install the Tiago packages. Please, follow these instructions.

Install the Tiago simulation workspace:

```
> sudo apt-get update
> sudo apt-get install git wget ipython3 python3-rosinstall
  ros-noetic-desktop-full python3-catkin-tools python3-rosdep
  python-is-python3 ros-noetic-actionlib-tools
  ros-noetic-moveit-commander
> sudo apt install -y ros-noetic-four-wheel-steering-msgs
  ros-noetic-urdf-geometry-parser ros-noetic-ddynamic-reconfigure
```

```
    ros-noetic-people-msgs ros-noetic-twist-mux ros-noetic-map-server
    ros-noetic-amcl ros-noetic-robot-pose-ekf
    ros-noetic-moveit-planners-ompl
    ros-noetic-moveit-simple-controller-manager ros-noetic-global-planner
    python3-future ros-noetic-teb-local-planner ros-noetic-hokuyo3d
    ros-noetic-urg-node ros-noetic-sick-scan
>   mkdir ~/tiago_public_ws && cd ~/tiago_public_ws
>   wget
    https://raw.githubusercontent.com/pal-robotics/tiago_tutorials/noetic-
    devel/tiago_public-noetic.rosinstall
>   source /opt/ros/noetic/setup.bash
>   rosinstall src /opt/ros/noetic tiago_public-noetic.rosinstall
>   catkin build
>   source src/setup.bash
>   rosdep install -y --from-paths src --ignore-src --rosdistro noetic
    --skip-keys "urdf_test omni_drive_controller orocos_kdl pal_filters
    libgazebo9-dev pal_usb_utils speed_limit_node camera_calibration_files
    pal_moveit_plugins pal_startup_msgs pal_local_joint_control
    pal_pcl_points_throttle_and_filter current_limit_controller
    hokuyo_node dynamixel_cpp pal_moveit_capabilities pal_pcl
    dynamic_footprint gravity_compensation_controller pal-orbbec-openni2
    pal_loc_measure pal_map_manager ydlidar_ros_driver"
>   cd ..
```

Create a workspace to develop your packages

```
>   mkdir catkin_ws  && cd catkin_ws
>   source ~/tiago_public_ws/devel/setup.bash
>   mkdir src && cd src && catkin_init_workspace
>   cd .. && catkin build
```

Now you are ready to install the environment for the assignment inside `catkin_ws`

- Clone your repository into your workspace, it will initially contain all the environment and launch files necessary to start the development. You **must** create your package into your group repository and deliver this one as a whole:
  ```
  > git clone
  https://github.com/PieroSimonet/tiago_iaslab_simulation.git
  ```

- To remap to a new repository it can be helpful remove the old git blob:
  ```
  > rm -rf tiago_iaslab_simulation/.git
  ```

- Build the package:
  ```
  > cd ~/catkin_ws
  > catkin build
  > source ~/catkin_ws/devel/setup.bash
  ```

- Start the simulation (cmd console 1):
  ```
  > roslaunch tiago_iaslab_simulation start_simulation.launch
  world_name:=robotics_library
  ```

- Navigation stack (cmd console 2):

```
> roslaunch tiago_iaslab_simulation navigation.launch
```

N.B. depending on your GPU setup, you may face some problems in the laser simulation. Anyway, you can use VLAB to correctly simulate laser sensors.

VLAB

In VLAB, the tiago environment is already installed inside the container.

- As usual, please follow the instructions to start the container:
  ```
  > start_tiago
  > source /opt/ros/noetic/setup.bash
  > source /tiago_public_ws/devel/setup.bash
  ```

- Create a new workspace (you can also use one already created):
  ```
  > mkdir -p ~/catkin_ws/src
  > cd ~/catkin_ws
  > catkin build
  > source ~/catkin_ws/devel/setup.bash
  ```

- Download the package with the simulated environment into the workspace:
  ```
  > cd ~/catkin_ws/src
  > git clone
  https://github.com/PieroSimonet/tiago_iaslab_simulation.git
  ```

  You are not supposed to edit this package. Just download and use it.

- To remap to a new repository it can be helpful remove the old git blob:
  ```
  > rm -rf tiago_iaslab_simulation/.git
  ```

- Build the package:
  ```
  > catkin build
  > source ~/catkin_ws/devel/setup.bash
  ```

- Start the simulation (cmd console 1) and wait until the start up is completed:
  ```
  > roslaunch tiago_iaslab_simulation start_simulation.launch
  world_name:=robotics_library
  ```

- Navigation stack (cmd console 2):
  ```
  > roslaunch tiago_iaslab_simulation navigation.launch
  ```

N.B. Sometimes the simulation does not properly initialize. If you experience some issues, try to kill and restart the last two commands.

How to submit your solution

To submit your solution, you **must** do the following:

1. **Setup your group repository**

a. Create your **private** group repository on Bitbucket named as "ir2324_group_XX", where XX must be substituted with your group number:

```
> cd ~/catkin_ws/src
> git init
> touch README.md
> git add README.md
> git commit -m "first commit"
> git push
```

If you are using local installation, please be sure to upload in the repository only the packages you are required to submit, for example calling:

```
> echo -e 'tiago_iaslab_simulation/' >> .gitignore
```

b. Includes all the group members and the 3 tutors in the collaborators of the repository. Here our BitBucket profiles:

- simonetto.piero@gmail.com
- alberto.bacchin.1@phd.unipd.it
- danieleva2005@gmail.com

c. Public repositories or repositories we cannot access **will not be considered for correction**.

2. Start coding

a. **Push your code (with good comments)** into your group's git repository. Code explainability will be part of the evaluation criteria.

b. We encourage you to commit the code as you write it and not in one block

c. **Add a README.md file** which contains the necessary commands to correctly execute and test your code (e.g. *roslaunch* and *rosrun* commands). Make sure the instructions are working properly. **The first lines of the README.md must contains the following informations**:

GROUP XX
Member1's name, email
Member2's name, email
Member3's name, email (if any)

d. We suggest you to test your code step by step in a new clear workspace, cloning the code from your repository in order to be sure that your code is executable by third parties.

e. **Make sure your code is compiling before submitting**. Not compiling codes will be penalized in the final grade. We will make a small compilation test a week before the deadline in order to let you know if your code compiles properly. We will post on the forum the test modalities.

3. Submission

1. **You are allowed to push the code in the repo until the submission on Moodle**. Later commits will not be considered for the evaluation.
2. To get the maximum grade, you have to submit within the 23rd December at 23:59 (CET). Later submissions will be penalized.

3. **Prepare a brief PDF report (max. 2-3 pages)** that explains your solution. The report is supposed to explain the high-level ideas, strategies or algorithms you implemented to solve the assignment. **The first lines of the report must be like**:

   GROUP XX
   Member1's name, email
   Member2's name, email
   Member3's name, email (if any)
   LINK TO THE REPOSITORY
   LINK TO ADDITIONAL MATERIAL (see point 5)

4. **Prepare a short video** (e.g., screen recording) that shows the execution of your software by the robot. This is necessary in case we cannot easily run your code.
5. **Report and video must not be placed in the repository**. We suggest you create a Google Drive folder, upload the video or any additional material and share with us the link to the folder inside the report.
6. We will open a submission to Moodle. You will be able to attach the report to your submission (be sure that it contains all the needed information such as a link to the repo, link to the video, additional info, etc.). **The name of the submission must be as follows: GroupXX_A1**. Please submit only once per group, one member can submit for everybody.