Assignment

In the simulated environment there are 7 objects on a pick-up table: A red cube, a green triangle and a blue hexagon, plus 4 gold hexagons (Figure 1 and Figure 2). The gold hexagons are obstacles, and Tiago must not collide with them (or with the table) while grasping the required objects. Once an object is grasped it must be fetched and placed on the cylindrical table of the same color (see Figure 3).

The human user is represented by a ROS node (already implemented for you) which generates the sequence of delivery of the coloured objects.

The pose of the objects on the pick-up table can be obtained by exploiting the Tiago's camera and the AprilTag library thanks to the markers placed on the top of these objects. The AprilTag library identifies the markers through the IDs encoded in the QR codes. Each marker has a specific identifier. Finally, the shape of every object is known (see Figure 1).

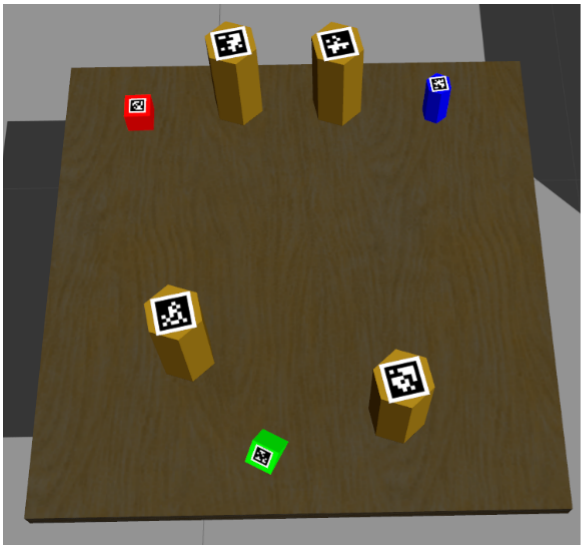


Figure 1

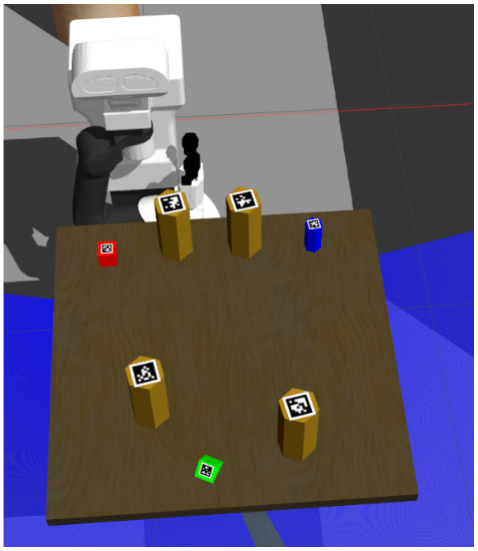


Figure 2

You have to develop a routine, exploiting the MoveIt! library, to pick and place each object (according to the received sequence) from the pick-up table (Figure 1 and 2) to its final destination in accordance with its color (i.e., the red cube on the red cylinder table) (Figure 3).

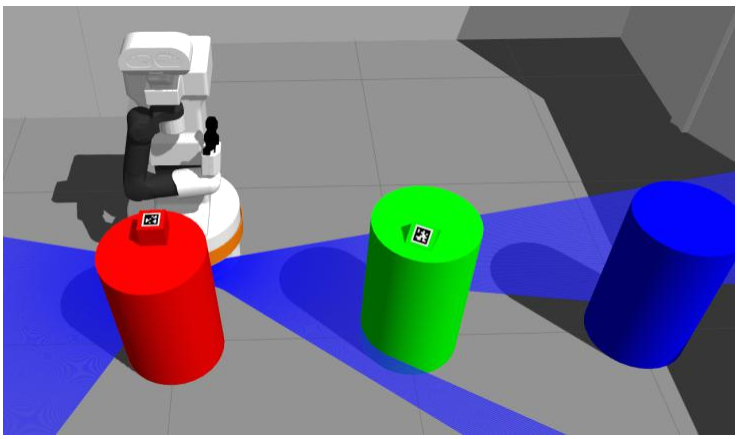


Figure 3

The whole environment is shown in Figure 4.

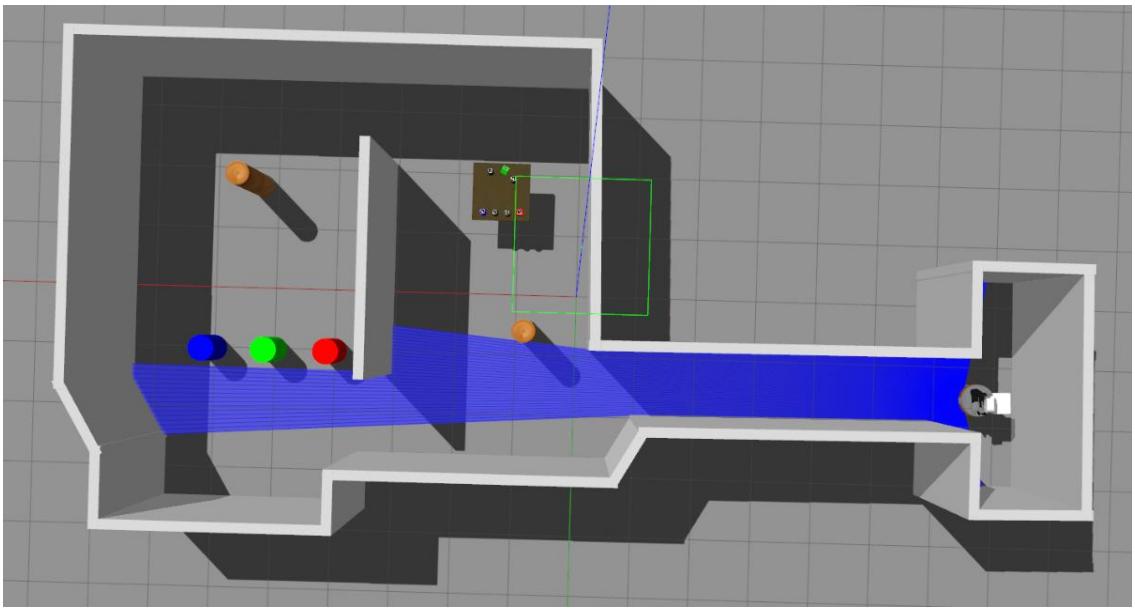
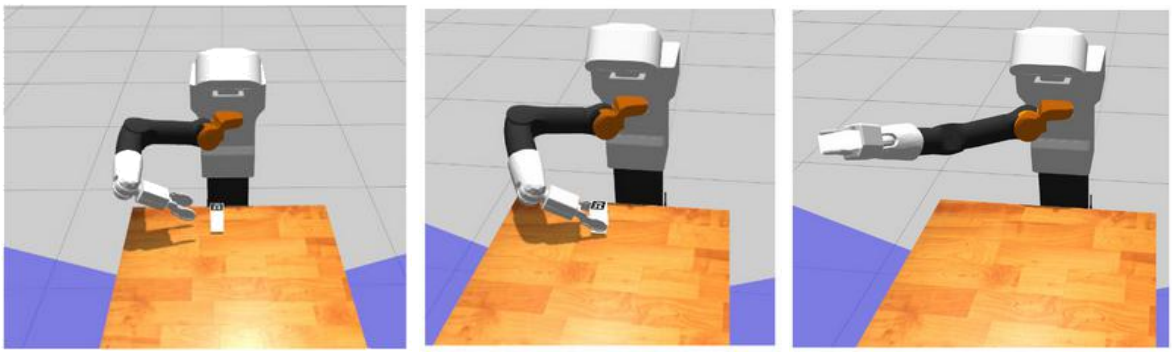


Figure 4

To navigate in the environment and to move from the pick-up table to the delivery tables, you must use the navigation module you developed in the previous assignment.

The assignment is fulfilled if, in the end, the three coloured objects are delivered on top of the corresponding coloured table in the assigned sequence.



a b c

Figure 5: Example of the pick routine.

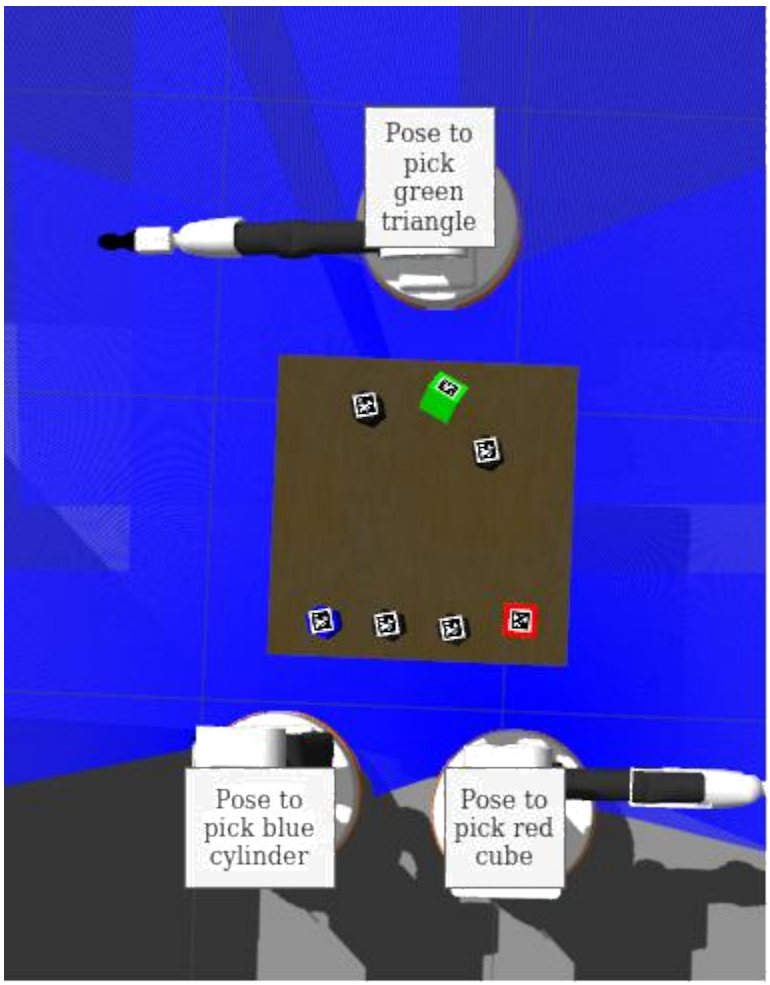


Figure 6

Extra points

Use the laser data to select the pose for the docking routine in front of the cylindrical tables during the object place phase.

In this case, you have to specify in the report that you also implemented this optional solution.

Instruction to start the homework

Follow these instructions to install the environment and start the homework:

For Local ROS Installation

- If you work on a local machine, clone the following repositories in your workspace (src folder) and compile the workspace before going on:

- Apriltag: https://github.com/AprilRobotics/apriltag.git
- Apriltag_ros: https://github.com/AprilRobotics/apriltag_ros.git
- Gazebo_ros_link_attacher: https://github.com/pal-robotics/gazebo_ros_link_attacher.git

For VLAB

- Clone the following repository Gazebo_ros_link_attacher: https://github.com/pal-robotics/gazebo_ros_link_attacher.git

```
$> cd ~/catkin_ws/src
$> git clone https://github.com/pal-robotics/gazebo_ros_link_attacher.git
$> cd .. && catkin build
$> source ~/catkin_ws/devel/setup.bash
```

For everybody

- Go inside the previous workspace
```
$> cd ~/catkin_ws
$> source ~/catkin_ws/devel/setup.bash
```

- Create a new package
```
$> cd src
$> catkin_create_package assigment2_package_name
```
- Start the simulation and MoveIt:
```
$> roslaunch tiago_iaslab_simulation start_simulation.launch world_name:=ias_lab_room_full_tables
```
- AprilTag:
```
$> roslaunch tiago_iaslab_simulation apriltag.launch
```
- Navigation stack:
```
$> roslaunch tiago_iaslab_simulation navigation.launch
```
- Human Service node:
```
$> rosrun tiago_iaslab_simulation human_node
```

NOTE: given the complexity of the pipeline, we advise you to prepare a single launch file that calls all the necessary nodes and other launch files. This will also speed up the development. Have a look at the ROS documentation if you don't know how to do it.

How to submit your solution

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

1. Setup your group repository

Use the same repository you set up for the Assignment 1. The repository must contain only the packages you developed for the assignments. Please, **use different packages for Assignment 1 and 2**.

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)
```

You can create a new README.md inside the Assignment 2's package or update the previous README.md, just **clearly distinguish instructions to run Assignment 2 nodes**.
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.

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 it within the 31th January 2023 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_A2**. Please submit only once per group, one member can submit for everybody.

Additional Information

APRILTAG: We provide an example of the AprilTag marker (Figure 7) and the set of IDs and corresponding frame_ids (Table 1).

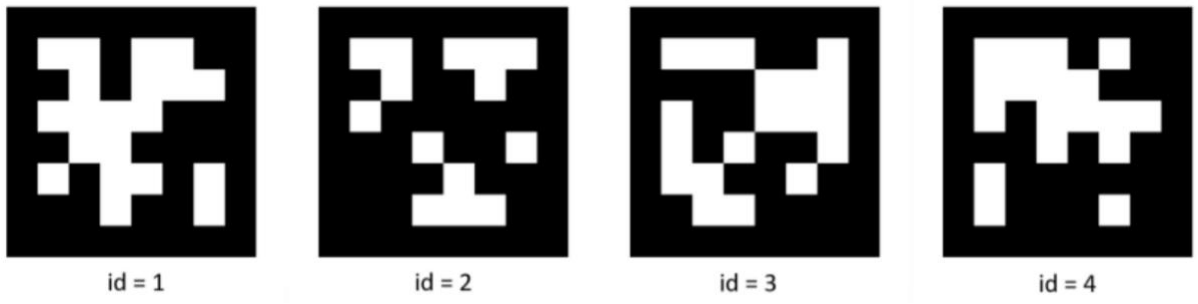


Figure 7

ID	FRAME_ID
1	blue_hexagon
2	green_triangle
3	red_cube
4	gold_obs_0
5	gold_obs_1
6	gold_obs_2
7	gold_obs_3

Table 1

HEAD ACTION: To detect the objects on the table you have to move the head of Tiago. See these two tutorials:

- http://wiki.ros.org/Robots/TIAGo/Tutorials/motions/head_action

- http://wiki.ros.org/Robots/TIAGo/Tutorials/trajectory_controller

Also, rqt_joint_trajectory_controller can be helpful to find the best position of the head. Run the following command:

```
$> rosrun rqt_joint_trajectory_controller
rqt_joint_trajectory_controller
```

GRIPPER OPEN/CLOSE: To open/close the gripper you can use the gripper controller interface. See http://wiki.ros.org/Robots/TIAGo/Tutorials/trajectory_controller

RVIZ: In RVIZ you can test MoveIt using the MotionPlanning (Figure 8)

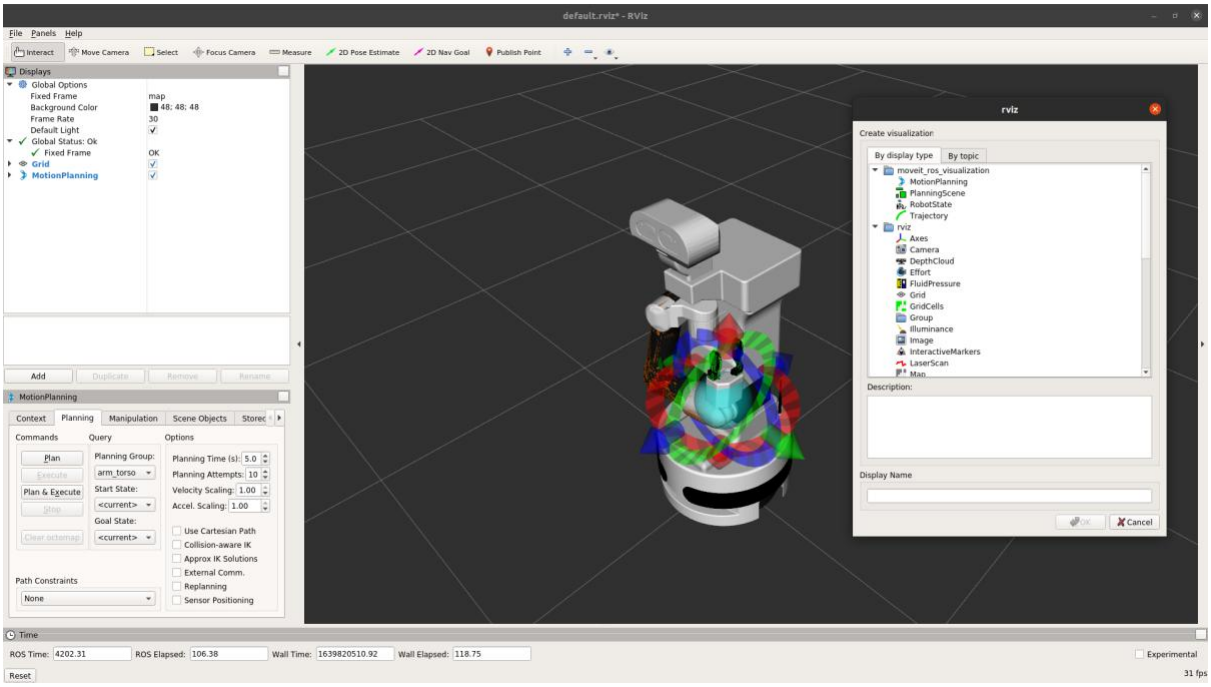


Figure 8

TF: Using rqt you can see the TF Tree (from terminal digit rqt, then in the GUI search plugins-visualization-TF Tree) (Figure 9)

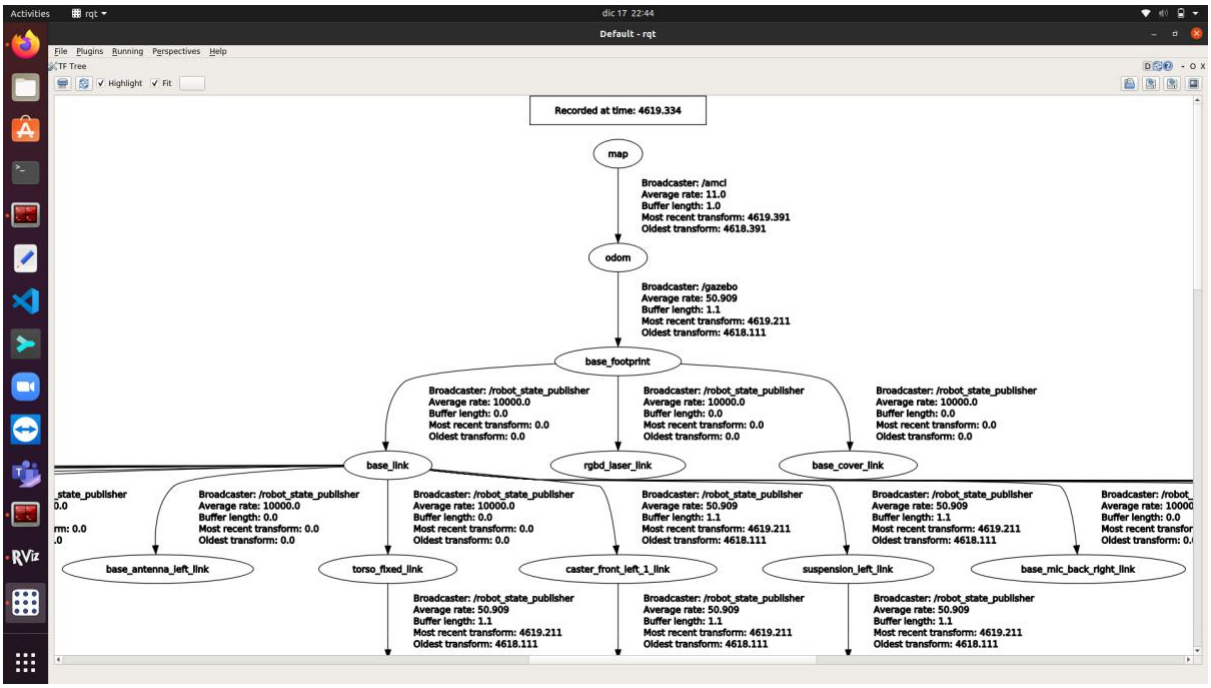


Figure 9