IR Project Report – Assignment 2

(EXTRA POINT DEVELOPED)

AUTHORS
Group 9

Boscolo Simone	Casarin Marco	Polato Anna
2026826	2044727	2007061

1. Introduction to the solution

Our solution is based on a `human_client` node that requests for the ids list to `human_node` and for each object it sends a goal to the `pick_and_place` action server. The latter manages the sequence of tasks to fully complete a pick and place of an object, including the navigation by calling the `assignment_1` server and the calls to `pick` and `place` action servers. Additionally, some services have been implemented: `get_poses` to get the waypoints, `transform` to transform poses between reference frames and `planning_scene` to update the collision objects. These services are used by the other nodes when needed. Overall, the philosophy behind our architecture is to use actions whenever the robot or any of its parts are moving (to handle feedback) while services for instantaneous operations, such as detection.

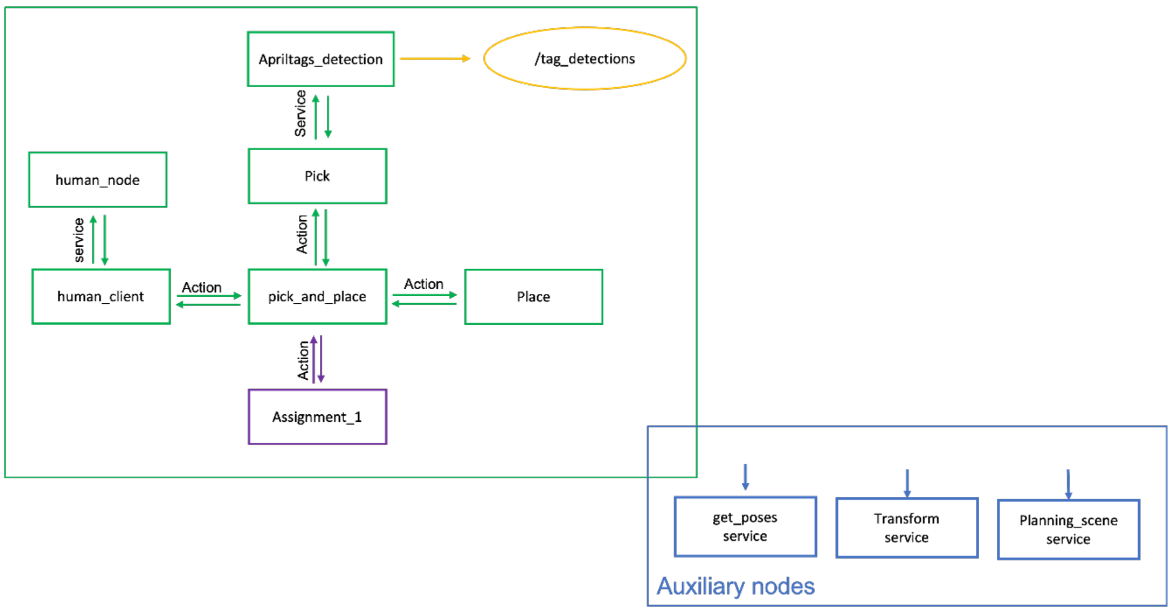

Figure 1: framework of the project. A total of eight new nodes were developed to implement a fetch and delivery behavior for Tiago Robot.

The Use of Waypoints and Fixed Poses

Several times the Robot needs to rely on the use of fixed waypoints in order to both manipulate objects and navigate without colliding obstacles. To make the code more flexible and adaptable to different scenarios, we preferred to build a service server (`get_poses`) that, when required, stores the needed poses from a text file passed as input from the user. The figures below illustrate all the waypoints and fixed poses chosen for the projects.

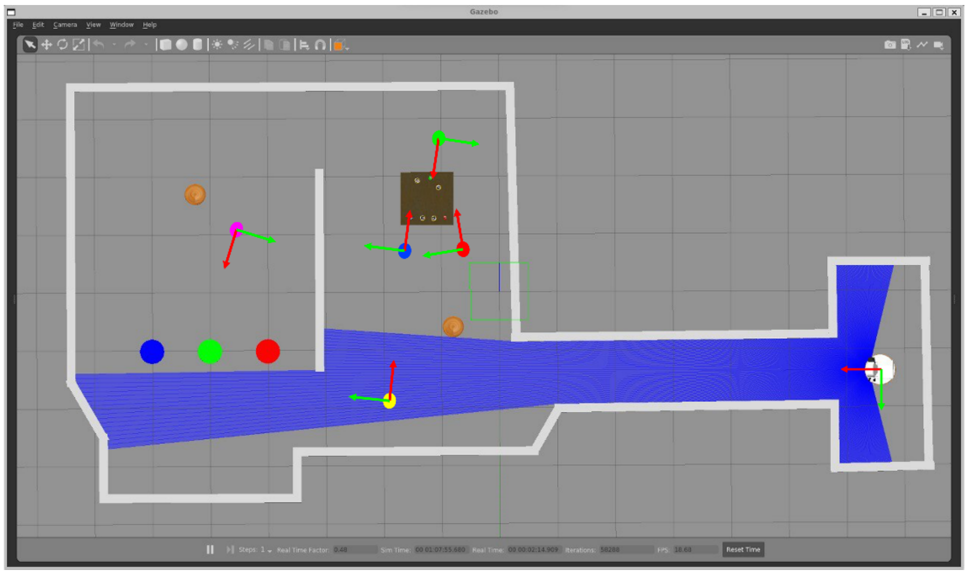
Figure 2: navigation waypoints.

In Figure 2, there are the following waypoints:

- Yellow: it avoids Tiago to get stuck between the wall and the log after the corridor;
- Purple: this waypoint avoids Tiago to detect the log at its right during the detection of target tables. It also helps Tiago to go back to the other room without passing through the cylinders (in this case the orientation is the opposite of the one shown in the figure);
- Red, Blue and Green: picking poses.

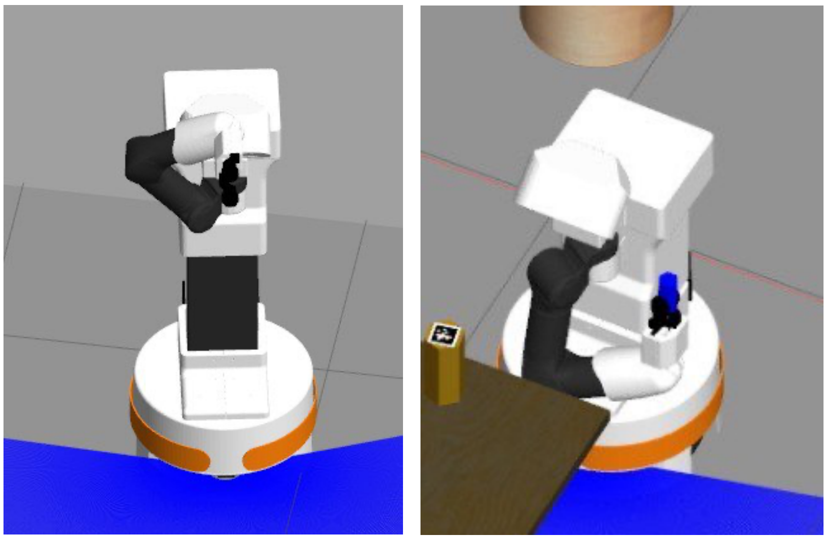
Figure 3: safe arm poses for navigation with and without object.

In figure 3, from left to right

- Arm pose for navigation without objects: the arm is ready for picking a new object.
- Arm pose for navigation with object: the arm is in a safe position to avoid collisions.

One more pose was chosen to lower Tiago's head and allow the camera to capture the table and objects as obstacles.

2. Picking routine and Planning scene

During the manipulation tasks, all the objects of the planning scene and the arms movements poses are defined with respect to the base_footprint reference frame. The planning scene is built every time a picking

or placing routine starts, by calling the service `planning_scene`. When the `pick` node calls `apriltag_detection` to detect the poses of the *apriltags* on the source table, the shapes and *apriltags* poses of the found objects are returned, so they can be added to the *planning scene*. For simplicity, it was assumed that the shape of red and green objects is a box, while all obstacles and the blue object are cylinders. Besides the objects, the coffee tables and cylindrical tables are added to the *planning scene* when required. The dimensions were all taken consulting the simulation on gazebo.
All the objects are approached from the top, as shown in Figure 4; The gripper (*gripper_grasping_frame)* is oriented with the pose of the *apriltag* but placed a few centimeters above it, pointing downwards at it. The vertical offset is computed as a function of the dimensions of the object and the gripper fingers. Then the gripper lowers along the vertical dimension so that to get close enough to grasp the object.

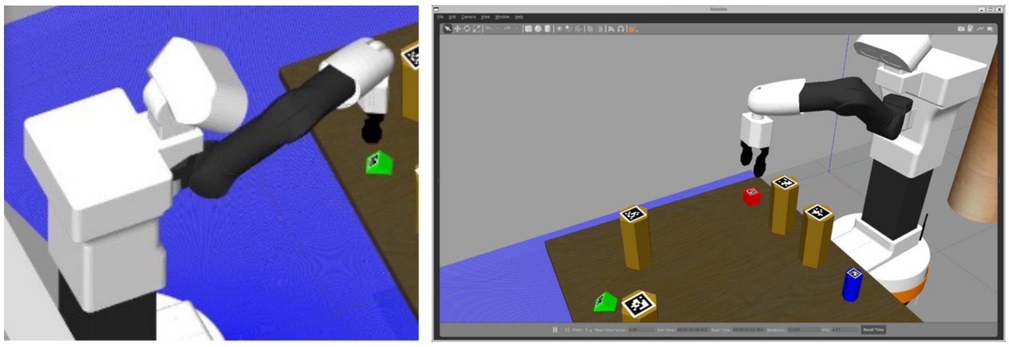

Figure 4: Example of object approach poses.

Planning scene construction

It has been noted that during the movements of the arm it might happen that grasped objects stick out the gripper and there is the risk of colliding with the obstacles in the environment, the floor and the robot itself since they are not part of its body. For this reason, the objects added to the *planning scene* are bigger than their actual size (I.e., object growing). Moreover, a wide and thin box is added to the planning scene to avoid collisions with the floor. Also, a small wall is added on the left of Tiago in the picking planning scene. The wall allows Tiago to close safely its arm, without the risk of hitting itself with the picked object. Similarly, two walls, one for each side, are used during the placing phase to avoid hitting the other tables on the side (fig.re 5). *This can be improved by adding directly to the planning scene the other tables, rather than two hard-coded walls*.

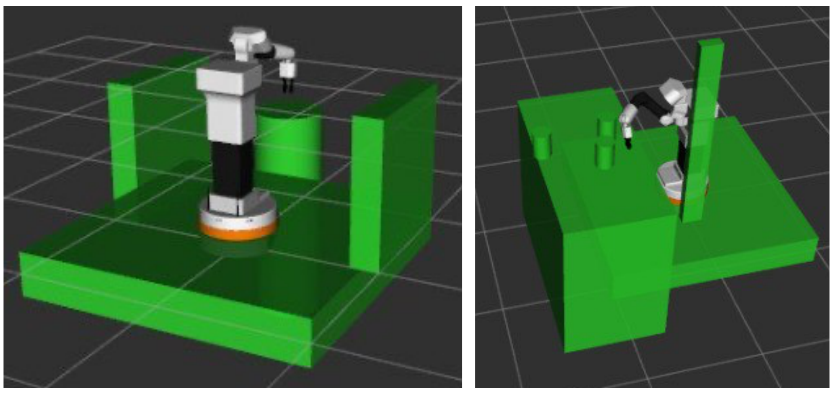

Figure 5: left – planning scene of the placing routine. Right – planning scene of the picking routine.

3. Object placing and cylinders detection [EXTRA POINT]

When Tiago is at the waypoint in front of cylindrical tables (purple point in Figure 2) it starts the detection and saves the cylindrical table poses, only the first time. After that, it moves toward the one whose color coincides with the object that it has to place. The detection is performed thanks to the routine of cylinders detection developed for the first assignment. Instead, the movement of Tiago is made by following the line that connects the center of Tiago base_link reference frame to the center of the table. Additionally, the position of the center of the table (x,y) is used as (x,y) coordinates for the placing pose, so that the object is placed on its center. A similar approach from the top (as in the pick routine) is performed and the heights are computed as a function of the table dimensions and the object sizes. Here it is assumed that all the placing tables are visible from the purple waypoint and that their order never changes.

4. Final Considerations

A large part of the work has been devoted to making the solution robust, i.e., the goal was to make sure that the robot could always fully complete the pick and place of the three objects without any collision. However, sometimes it can happen in the pick phase that the plan fails because the robot is too far from the table. After several tests, it has been noticed that (given the exact same goal pose) the navigation stack never reaches the exact same final pose; Specially the error increases/decreases when testing on a powerful VS slow machine. For this reason, it may occur that the pick planning fails.