Ambient Assisted Living for Long-term Monitoring and Interaction (ALMI)

Technical Report

12th January 2023

Abstract

The “Ambient Assisted Living for Long-term Monitoring and Interaction” (ALMI) project has focused on the domestic support of patients with mild cognitive impairment (MCI) or at early dementia (Alzheimer Disease-AD) stages. The duty of care for these patients typically resides with family members or caregivers, posing a significant burden on them. However, recent advances in science and assistive technologies open up new opportunities. The combination of already existing solutions with specified target group requirements can set new paths toward the development of future service robots, capable of providing effective assistance in the daily life of the patients. Along these lines, ALMI has researched and developed a novel proof-of-concept robotic assistant to help patients to cope with challenges in their everyday life in their natural home environment. Our demonstrator consists of an assisted-living TIAGo solution. In this solution, the TIAGo robot has used both its speech interaction and its object manipulation capabilities to help a user with mild motor and cognitive impairments in the daily activity of preparing a meal and also reacting in emergency situations such as calling for help, detecting/removing obstacles.

Contents

Use cases

The following outline the two use cases referred to throughout the remainder of this report.

A use case is a user-centred scenario, which demonstrates interactions between the robotic system, the user and the environment. The main aim is to accomplish the task and achieve a planned goal. Offering solutions should be useful and accepted by the user. Therefore, two conditions should be followed: usability and acceptability.

Use case 1a - Support in Daily life activities (food preparation)

The main goal for this use case is to support the user in activities of everyday life such as food preparation, e.g., by bringing ingredients proactively and/or on demand.

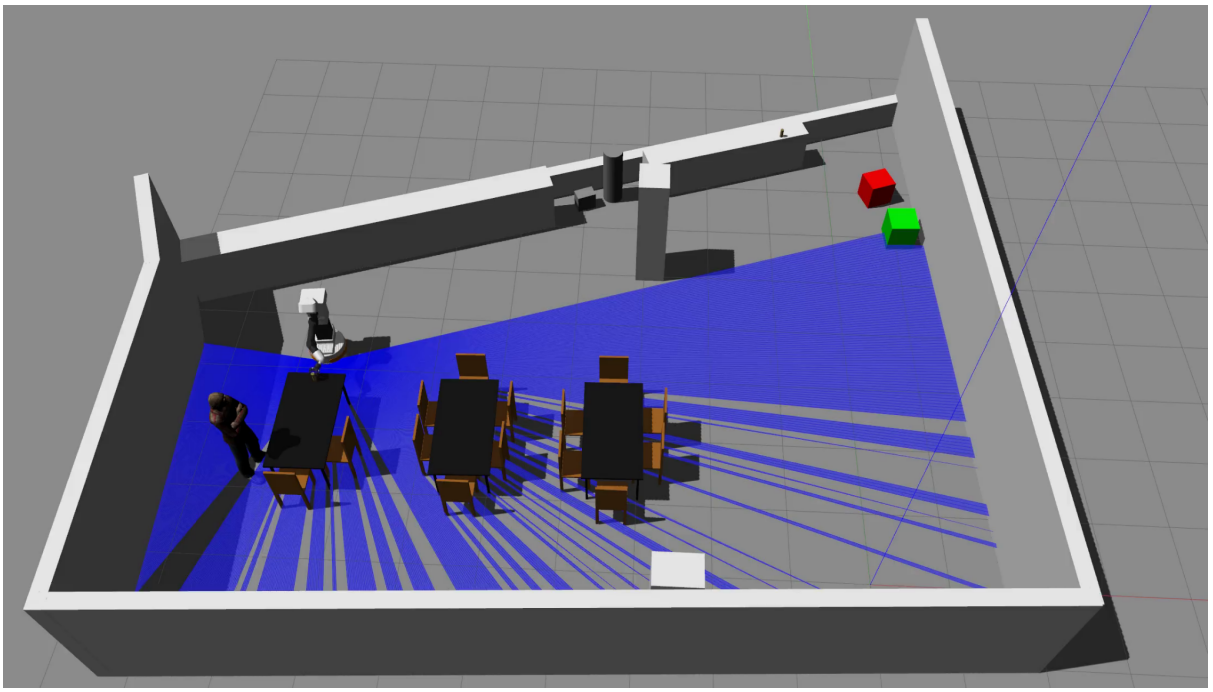

Figure 1: Human-robot collaboration in preparing a meal in a simulated environment

The robot approaches the detected person, informs them that it is lunch time and asks the person whether preparation should begin. The expected answer is “YES/NO”.

The robot then proceeds to inform the person about:

1. all the required ingredients and utensils;
2. confirms that it will take care of the tomato sauce and the spaghetti, with the user being required to bring the utensils.

The robot goes to check if there is tomato sauce and spaghetti, then goes back, brings the required cooking ingredients, and reports the status to the person. The robot explains step by step what to do, waiting for confirmation from the person in order to proceed to the next task (“OK” or similar phrase).

Use case 1b - Support in Daily life activities (food preparation)

Similar to the previous use case, but the human traversing the environment will impact the path of the robot, which demonstrates the capabilities of the robot to avoid obstacles (i.e., human or objects) and plan a different route.

Use case 2 - Emergency

The main goal of this use case is the robot's appropriate reaction to health and emergency situations, e.g., detection of an unconscious person or lack of contact/activity and taking action by calling for help.

The robot is placed in a corner of the room and periodically checks the environment looking for emergency situations (roaming the area).

When TIAGo detects a person on the floor, it goes outside of the room and looks for a person. If a person is detected, TIAGo asks for help by voice. If not, it performs an emergency call.

For more information regarding the use cases and the selection process based on importance, see deliverable D3.1 - requirements and validation scenario.

Robotic arm development

PAL Robotics has built the first prototype of the new arm including new sensors and functionalities to comply with the norms of industrial and personal care robotics. This building was successful thanks to the initial research and design of the arm to reduce the pinching points and widening the workspace area. It passed by the creation and implementation of cutting-edge electronics and actuators that allowed it to implement more advanced control like force control. Together with brakes it improved the security features of the TIAGo arm to be able to collaborate closely with humans.

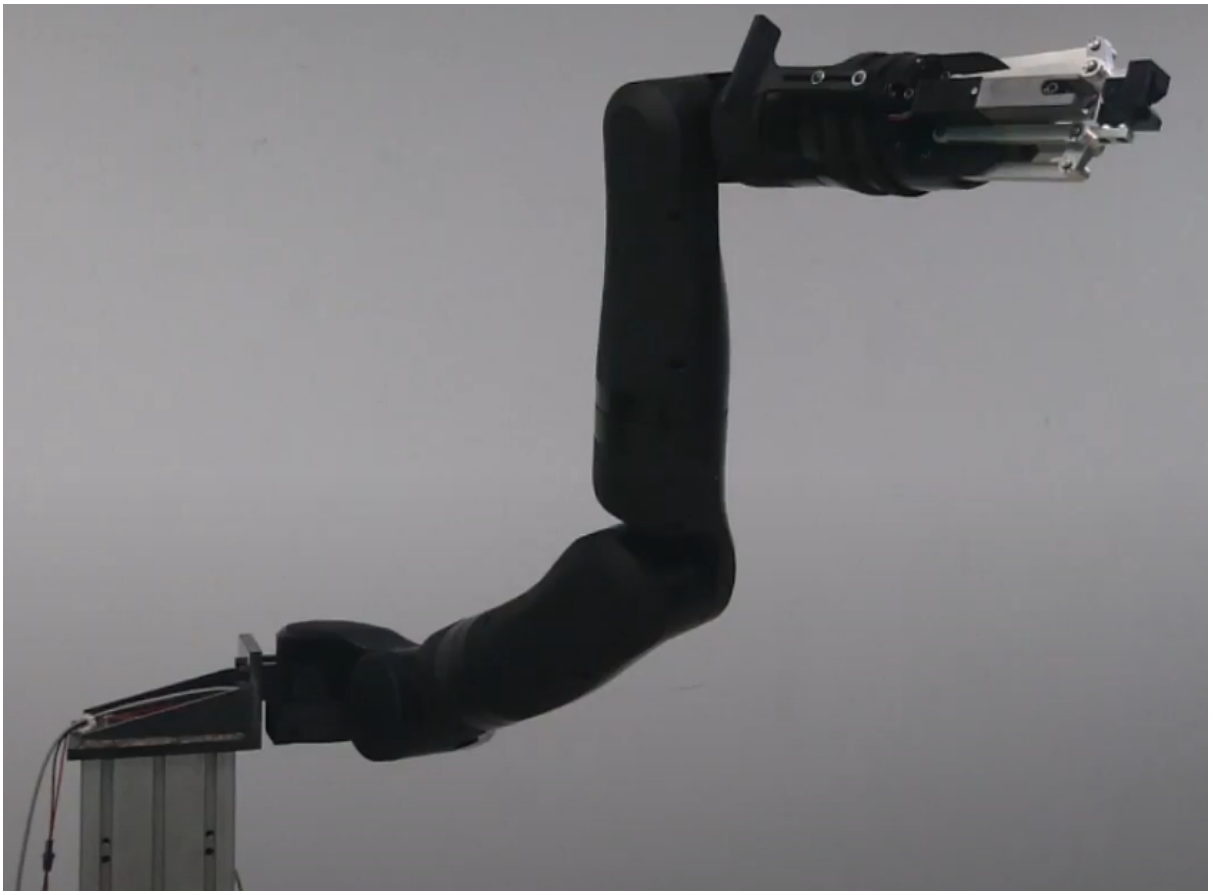

Figure 2: First prototype of the new arm

This new arm complies with what was expected in terms of security and robustness, to finalise the product more tests and calibrations of the actuators are needed in order to be able to integrate it into the TIAGo platform later in the project. The development of this arm opens many lines of research that are for the interest of PAL Robotics internally and for the potential clients that may get this feature in the project. This includes topics from very low level, such as the development of a larger range of compliant controllers, all the way up to very high level, such as applications in Human-Robot interaction and social navigation.

Memory map development

The memory map is working at ROS level and at upper level with the Python Interface. It consists of several methods to add, get and update objects in the memory of the robot. The deliverable D1.2 Environment monitoring method helped the developer from York to include this tool for the demonstration to be able to initiate where the ingredients are on the map and to update it accordingly to where it is taken. This tool was updated with the python interface based on York's feedback.

The TIAGo knowledge store comprises a semantic map of the user's kitchen, with object position indicated at specific locations. This semantic map has been built using existing ROS (The Robot Operating System) Navigation Stack functionality, after mapping the user's kitchen and/or inputting the details. This semantic map has been designed in order to be the most reusable possible with custom types of objects with different attributes, to be able to store the other kitchen items such as tables, sinks as well as the food ingredients with the same tool.

In order to achieve the memory map this structure and functions have been defined :

```
/knowledge_store/add - Add a new element to the KB & dump to file
Object Type
Attribute (key / value / type pair)
-----
Object ID

/knowledge_store/update - Change a parameter on an object the KB & dump to
file
Object ID
Object Type
Attribute (key / value / type pair)
----
Full Object
Bool ok
String message

/knowledge_store/remove - Remove an object on the KB & dump to file
Object ID
----
Bool ok
String message

/knowledge_store/get - Get information on an object
Object ID # Optional
Object Name
Object type
----
Array Full objects
Bool ok
String message
```

Rosparam outline:

```
knowledge:
  place:
    - object_id: 1
      name: cupboard 1
      poi: poi_cupboard_1
      task: open_cupboard
      torso_pose: 0.54
      head_1_pose: 0.5
      haed_2_pose: 0.1
  object:
    - object_id: 2
      name: fork
      default_place: cupboard 1
      unique: false
      current_place:
        - cupbard 1
        - cupboard 2
        - table
      last_seen:
        secs: 8
        nsec: 15200000
```

As defined in this outline, thanks to the developed memory map, it is possible to create objects with different attributes. Additionally, in order to be used by users that do not have knowledge of ROS, we created a python interface to use the previously created software in ROS.

More information regarding the usage of the memory map and the python interface can be found in deliverable D1.2 - environment monitoring method.

Modelling and verification

A high-level diagram of our approach towards a) modelling of the robot and the environment, and b) verification of requirements can be seen in Figure 3 below.

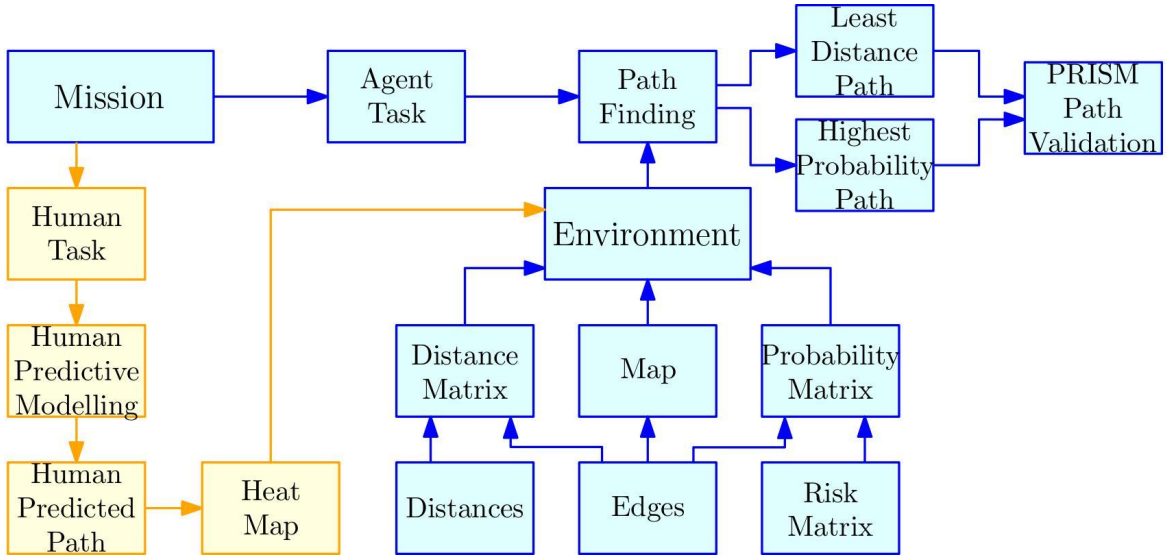

Figure 3: Generalised workflow for dynamic path planning

The approach comprises three stages: entity modelling, environment modelling, and path planning. The first stage encompasses all modelling associated with the robot and human. The sections highlighted in yellow are unique solely to the robot entity for performing human predictive modelling and are not active in the human entity. The second stage involves the modelling of the environment as a graph, with nodes representing the locations within the environment accessible to both the human and the robot, and edges annotated with a risk of moving from one node to another. The path planning algorithm invoked during the third, and final stage, returns two solutions: a path of least distance and a path with the highest probability of success. Whilst the path with the highest probability of success is a good measure of path safety, the path finding algorithm does not systematically determine the probability of success, and therefore an interface enabling the invocation of the probabilistic model checker PRISM[1] has been developed. This interface creates a series of actions based on an applied path, generating a unique Markov decision process model encoded in the PRISM modelling language. The model checker is then called at run-time to evaluate the applied path and to systematically determine the success probability of traversing the path. Finally, the path with the highest probability of success is selected by the planner and followed by the robot.

Figure 4 illustrates our path-finding framework in an example where the agent navigates through the environment to reach a specified location. The predicted path taken by the human causes a spatial conflict with the highest probability path taken by the agent as both entities at some point in time will be located at node 11. This impacts safety as the agent may be aligned on a trajectory in close proximity to the human or cause distress by blocking

[1] www.prismmodelchecker.org

their path. However, by using the workflow outlined in Figure 3, the predicted path of the human is interpreted by the agent's representation of the environment in the form of a heat map. This heat map updates the agent's risk matrix, artificially increasing the risk associated with edges of commonality and decreasing the probability of successfully traversing the edge due to the presence of the human. Path finding is then performed again for the agent based on this updated interpretation of the environment.

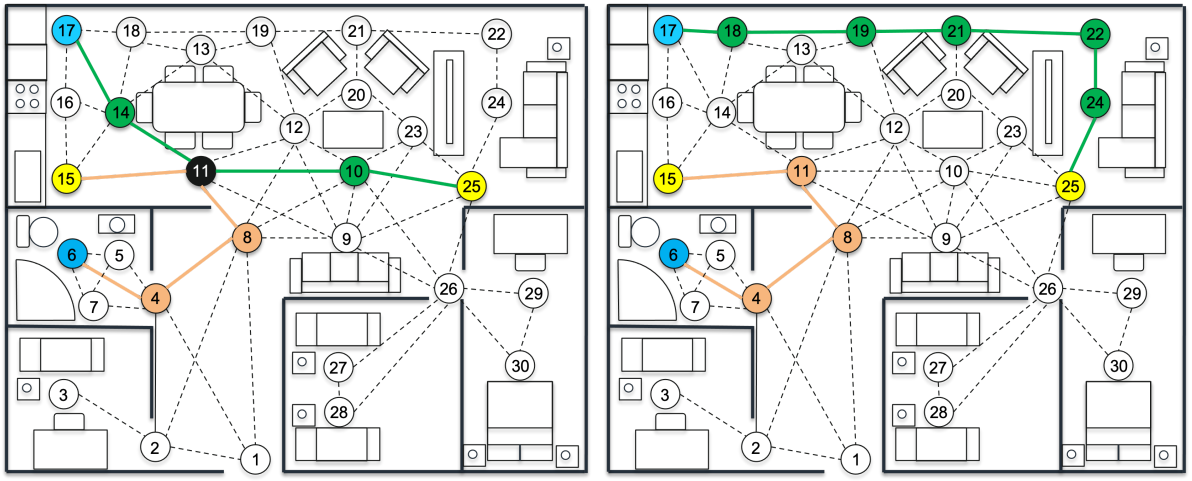

Figure 4: Conflicted agent and human paths / Updated path for the agent

Since the predicted path for the human was spatially conflicted with the original highest probability of success path for the agent (left figure), the obtained success probability no longer holds true due to the increased risk from the human presence (97% to 40% drop). Therefore, when performing path finding for the agent using the updated environment representation, a different solution is returned corresponding to the highest probability of success. This can be seen in the figure on the right where an updated path of highest probability guides the agent in avoiding the cluttered central area taken by the least distance path, with approximately 96% probability of success.

Further details

A detailed description of the approach summarised in this section is available in the following paper:

Hamilton, J., Stefanakos, I., Calinescu, R., Cámara, J. (2022). Towards Adaptive Planning of Assistive-care Robot Tasks. In *Proceedings of the 4th International Workshop on Formal Methods for Autonomous Systems (FMAS)*. 175-183

Integration

The integration inside the TIAGo platform has been achieved by translating the environment from the Gazebo simulator. Based on the environment's depiction (Figure 1), we created a representative map using our generalised approach for modelling environments as graphs with edges annotated with levels of risk.

Each edge is assigned a risk metric, with risk measured as the probability of failing to traverse the edge successfully. In this example, high-risk edges (depicted using orange/red lines for medium/high risks in Figure 5 below) occur in cluttered spaces areas where foot traffic is very likely. Similarly, low-risk edges (depicted as green lines) are associated with uncluttered areas where there is plenty of space to navigate for both robot and human.

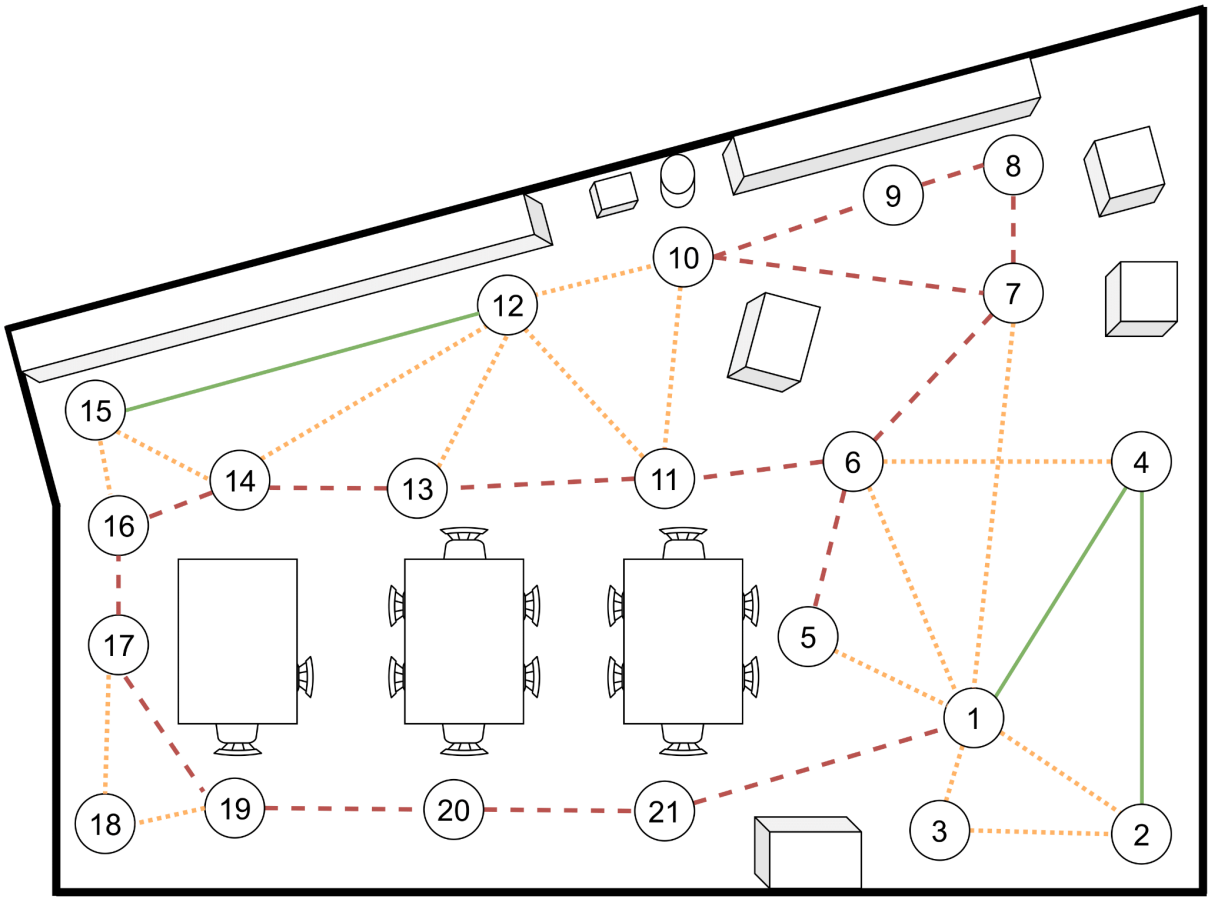

Figure 5: Risk severity colour coded edge connections

Using the above graph in combination with our developed framework we were able to proceed with the verification process. Providing the graph as input to our path-finding framework, we obtained the optimal paths for the robot to take in the environment. The integration was achieved automatically by first obtaining the result from our framework and then using the functions `goToPoseOnMap()` and `pick()/place()` to issue a chain of commands to the robot to proceed in traversing into the environment according to the specified sequence of states, and pick/place objects in the designated areas, respectively.

The arm was successfully integrated on a platform that mimics the TIAGo platform thanks to the previously extensive work made on the arm including the electronics and controllers. Because of the advanced components and obstacles faced during the integration, the arm has not been fully integrated in the actual TIAGo platform with all the functionalities needed for the demonstration. Nevertheless, effort will continue to fully integrate the arm on the actual TIAGo platform and a gripper will be developed from scratch to match the Ethercat communication system and be able to interact with the environment safely.

Figure 6: TIAGo robot equipped with the new robotic arm

The developed memory map was successfully integrated and tested with the required level of integration. Thanks to its python interface it was integrated further in the demo process to update the localization of the object once picked and placed.

Further details

A detailed description of the integration summarised in this section is available in the deliverable D3.2 - system tested preliminary version of demonstrator.

Demonstrator

The goal of the demonstrator strand was to provide a digital-twin for the testing and demonstration of the other work-packages. To achieve this, we created an identical representation of the environment that the TIAGo robot operates in the PAL robotics facility, using the Gazebo simulator, as seen in Figure 1.

Following the environment's successful creation, we focused on the integration of the approach inside the Gazebo platform (see section Integration for further details), which allowed us to execute the developed path planning framework both in the simulated environment and in the real robot.

Use cases 1a and 1b can be seen in the video 1a and video 1b, respectively. These were recorded in the Gazebo simulation and showcase the applicability of our approach in assisting the robot to select the most appropriate path for traversing the environment, while at the same time ensuring that the safety of the human located in the environment is maximised. The recording of use case 2 took place only in the real world, and will become available with the release of a new video containing all case studies recorded in the PAL robotics facility.

Figures 7 (use case 1a) and 8 (use case 1b) below demonstrate the capability of our path planning framework in combination with the functionalities of the TIAGo robot in issuing commands for picking and placing objects in designated locations, and avoiding close contact with the human to prevent harm, respectively.

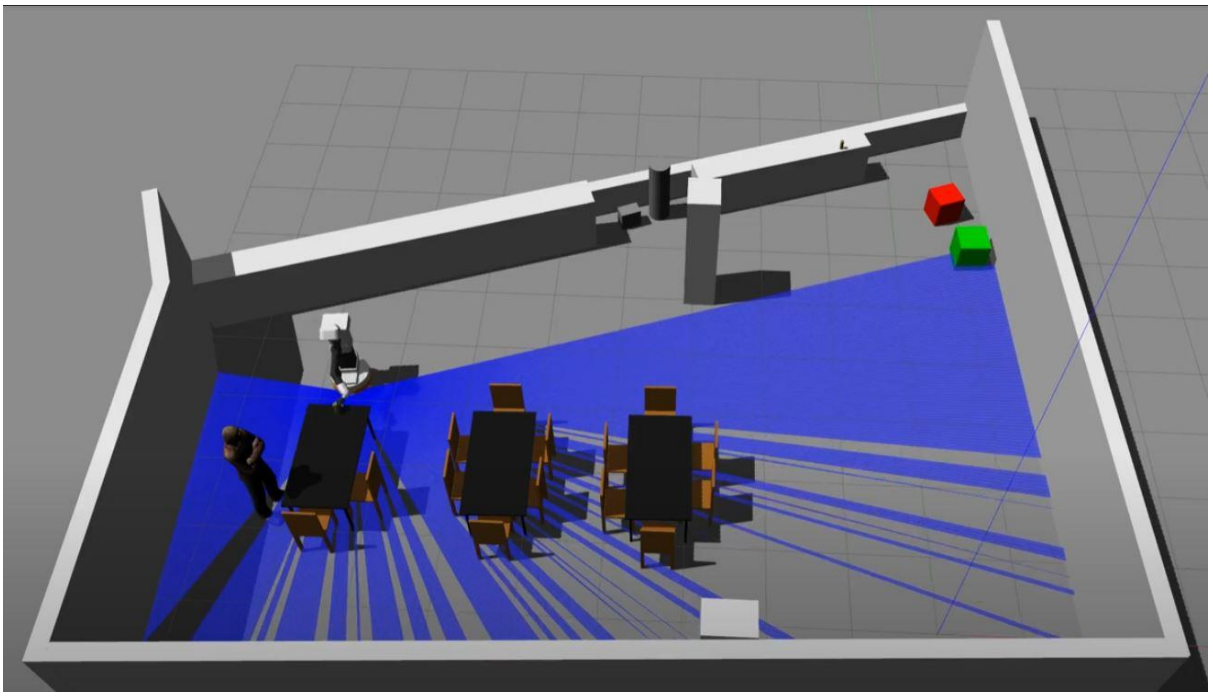

Figure 7: TIAGo robot places an object into the table in front of the human

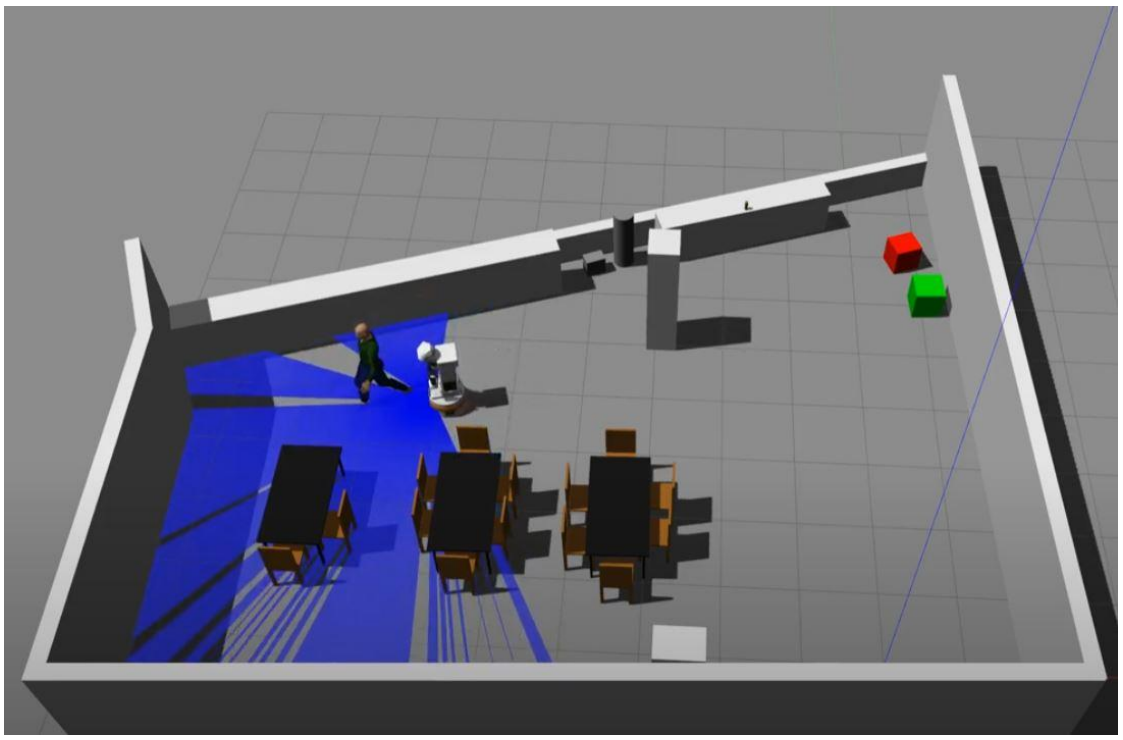

Figure 8: TIAGo robot performs rerouting to avoid collision with the human

The code that was used during the simulation, was also launched on the real robot with some minor adjustments for some of the nodes' positions to allow for better object grasping and placement (Figure 9) performed by the TIAGo robot. Additionally, the inference server with the human detection functionalities (Figure 10), the speech recognition and the fall detection (Figure 11) from the OpenDR project was active on the real robot but not supported in the simulation. Relevant adjustments have been made in the code to take the above mentioned features in account and adapt the robot's behaviour.

Figure 9: TIAGo robot performs grasping in the PAL kitchen

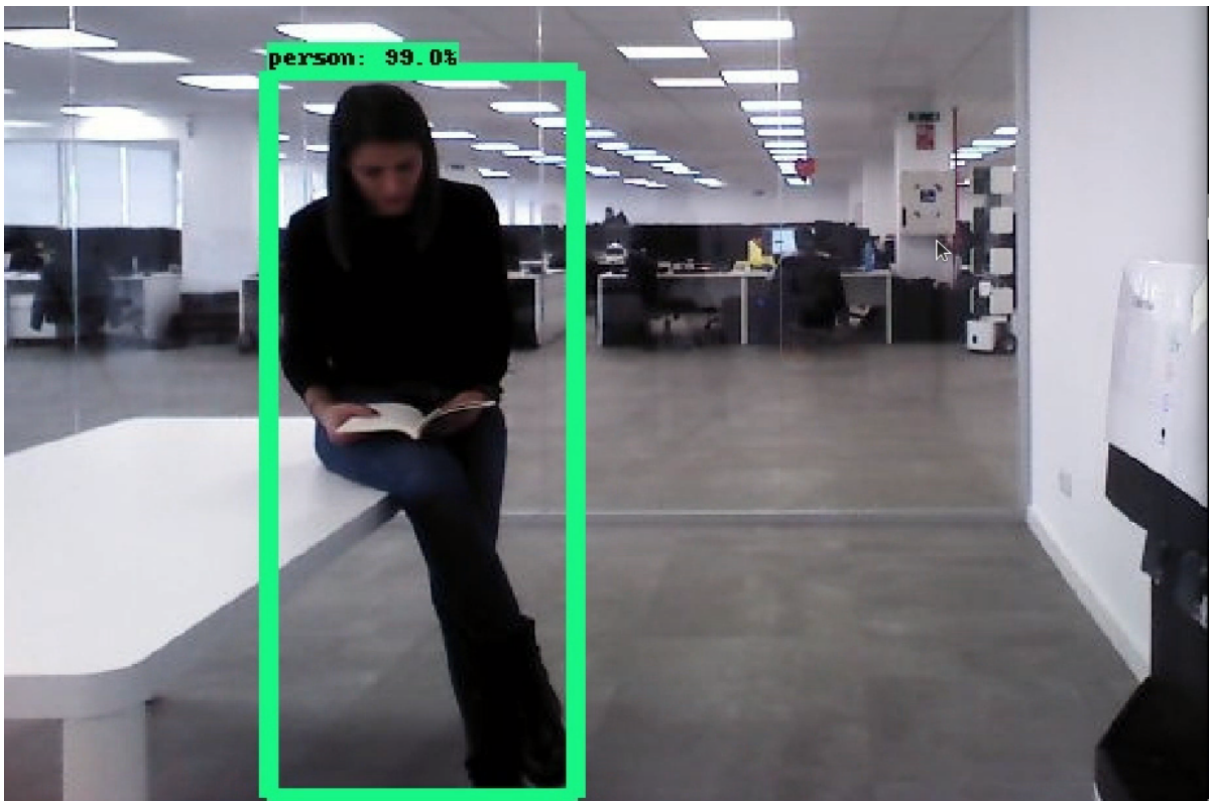

Figure 10: TIAGo robot performs human recognition

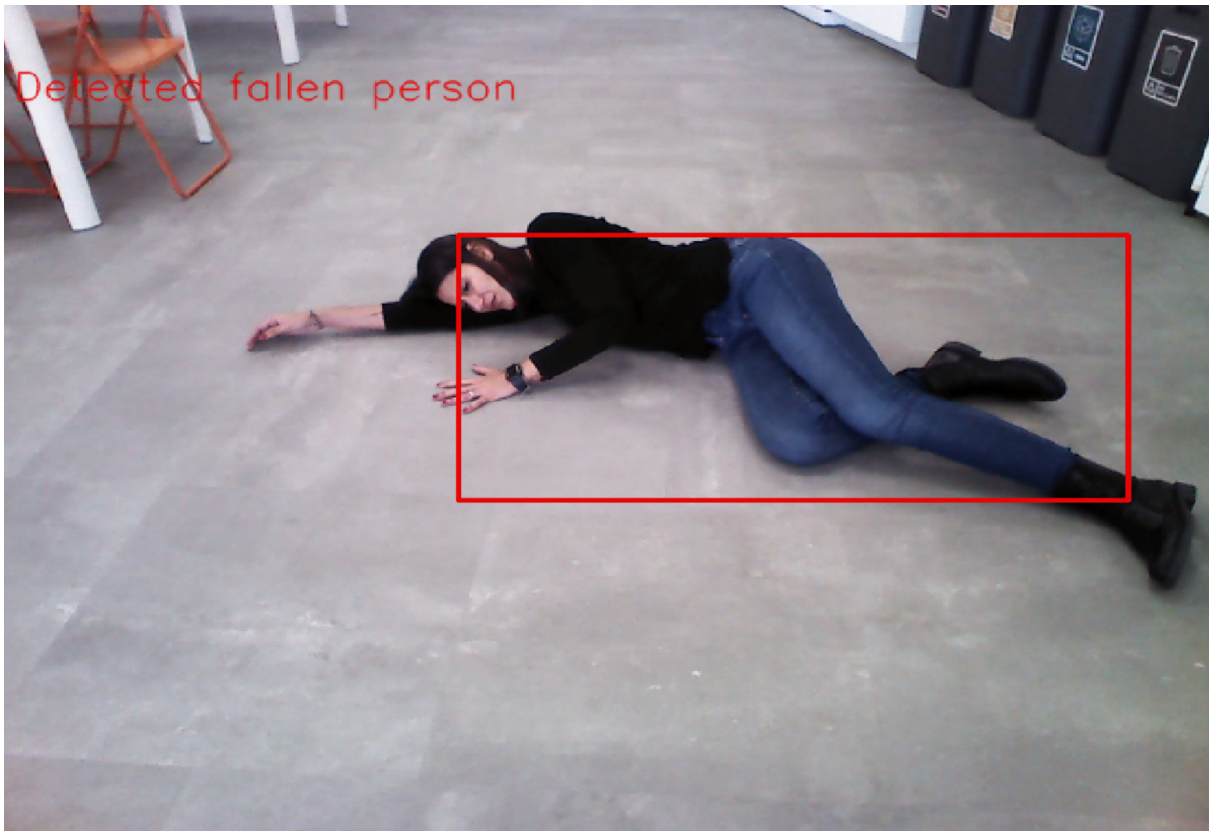

Figure 11: TIAGo robot performs fall detection from OpenDR