# Perception and decision making for intelligent robots

Oscar Hjertqvist, Martin Lamm, Hasan Sharineh, Yaochen Song, Joel Trollheden

***Abstract*—**

## I. Introduction

Autonomous Robots are already an essential part in many personal assistance and manufacturing applications, and the usage of Autonomous Robots in social indoor environments such as offices or homes are expected to increase in the coming years [1]. In order to enable Autonomous Robots to make informed decisions, perform actions autonomously, and maneuver in complex and dynamic environments, the robots require rigorous perception systems to perceive its surroundings. Some examples of commonly used sensors for autonomous navigation are: Absolution positioning sensors (GPS/GNSS), visual sensors (Cameras), range sensors (Radar/Lidar/Ultrasonic sensors), and relative positioning sensors (IMU/INS) [2].

The aim of this article is to investigate how different sensors can be used for autonomous navigation in indoor environments, and implement the perception and decision making system on a physical robot. The testbed used in this research was a TIAGo 101 robot from PAL Robotics[3], which is further described in section III-B. The research was limited to three areas of investigation:

- The development of a localization and mapping algorithm that allows the TIAGo robot to map its surroundings and to identify its location in a map.
- The implementation of an off-the-shelf perception system that allows for the TIAGo to identify objects ahead of it. For example people, chairs, trashcans and other objects that might show up in a indoor environment.
- Implementation of a basic navigation and decision making algorithm that can use the information from the sensors, the map and its current location to take decisions such as stop, move, slow down, and speed up.

## II. Related work

Autonomous Vehicles, Object detection, Path planing? [4]
When it comes to autonomous navigation an interesting research angle is autonomous navigation solely based on vision. In the literature there are a plethora of different implementations ranging from SLAM methods focusing on utilizing point clouds to neural network based solutions [5].

## III. Method

The following section describes the method, including the simulation environment, the robot, and the tools used in the study. The section also provides a detailed description of how the object detection, decision making, and localization was implemented on the robot.

### A. Simulation environment

During the project Gazebo [6] was used to test the different software packages. In Gazebo there are multiple different scenarios or virtual worlds that can be utilized for testing different things, such as an office environment or a space with many different rooms which would be suitable for testing localization. The two scenarios mainly used in this project were the `simple_office_with_people` and `small_office`. In figure 1 the former of the two can be viewed.

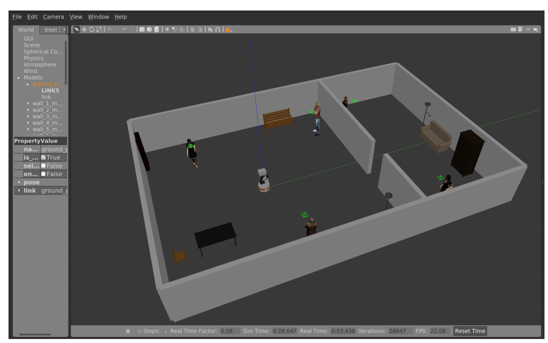

Fig. 1. The environment simple_office_with_people in Gazebo

The reason for using simulation instead of only implementing it on the real robot is due to safety concerns, both for the humans around the robot as well as the robot itself. Furthermore testing is easier and more efficient in simulation when compared to the real robot as the environment is completely known, can easily be reset and manipulated.

### B. TIAGo robot

The perception and decision making system was implemented on a TIAGO 101 robot from PAL robotics, visualized in figure 2. TIAGo is a multi-functional mobile research robot with several different sensors, and actuators. The TIAGo robot utilizes ROS for managing its sensors, actuators and the communication between its different parts. The available sensors on the TIAGo robot are the following [3]:

- A laser range-finder is mounted to look forward.
- Sonars in the back of the base.
- IMU in the base.
- Motors current feedback in arm/wheels

- Stereo microphones in the torso.
- Force/torque sensors in the wrist.
- RGB-D camera in the head.

However, as this article is focused on object detection, mapping and localization as well as decision-making, only the laser range-finder, odometry data from the wheels and RGB-D camera were used and considered in this study.

Fig. 2. Overview of TIAGo robot, Source: [3]

### C. Object detection, segmentation & depth estimation

The objects were detected and tracked by utilizing YOLOv8 in the form of a ready-to-use ROS package [7]. YOLOv8 (You Only Look Once) is a computer vision model that is trained to recognize objects and then do segmentation and/or tracking among other functions [8].

The given tracker node was adapted to not only do object detection, but instance segmentation and depth estimation of the objects as well. Figure 3 visualizes the implemented object detection node, and its subscribed and published topics.

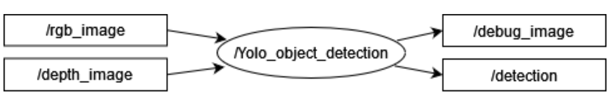


Fig. 3. Object detection node (oval) and its subscribed and published topics (squares)

From the RGB image data the object classes, the bounding boxes and the object masks were estimated using the YOLOv8n-seg model [7]. An example of a detected object and mask is visualized in figure 4. The distance to the different objects were then derived by using the depth image and calculating the average depth of all the registered pixels in the corresponding object mask. Since the depth image weren't able to register all pixels and contained nan-values, shown as blacked out parts in figure 5, the average depth of each object was calculated as:

$$d = \frac{1}{N_{\mathcal{S}}} \sum_{x_{ij} \in \mathcal{S}} d_{ij}, \qquad (1)$$

where $d_{ij}$ is the depth of each pixel $x_{ij}$, $\mathcal{S}$ is the set of real depth value pixels in the object mask, and $N_{\mathcal{S}}$ is the number of elements in $\mathcal{S}$. The detected object classes and positions (bounding box and depth) were then published as an array to be used in the decision-making.

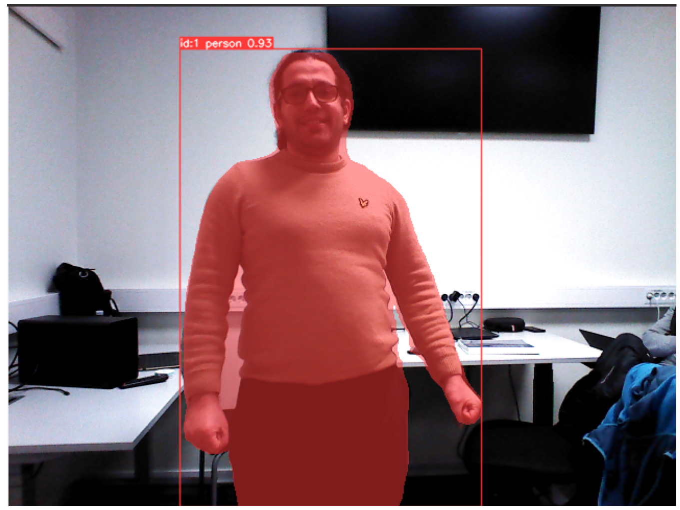


Fig. 4. Object detection and segmentation of a person from Yolov8 model.

Fig. 5. Example of a depth image, given by the RGBD-camera. Pixels with unregistered depth are blacked out in the image.

From 5 looking at the blacked-out are in the image and comparing to 4. It is noticeable that the depth sensor could not register the depth of reflective objects, such as the TV, and object edges.

### D. Mapping & Localization

Along with the position of the objects, the state of the agent is also necessary to make informed decisions [4]. A mapping and localization method was therefore implemented for the TIAGo robot. Using the TIAGo robot's laser range finder sensor at the base of the robot, together with the odometry from the motor feedback sensors, a map of the area was created utilizing the Gmapping ROS package [9].

Gmapping is a highly efficient SLAM (Simultaneous Localization and Mapping) algorithm that utilizes a Rao-Blackwellized particle filter to learn grid maps from laser range data [10]. SLAM minimizes the error of mapping and localization by considering both operations together [11]. Gmapping has good capacity for robustness, which enables it to generate accurate maps in various environments[12]. By manually navigating a room, the SLAM algorithm created a map of the room simultaneously as it kept track of the position of the robot. For this research project, a map of a lab and corridor environment was created.

The map was later saved and used in the pose estimation, path planning and the autonomous navigation of the robot. The localization and pose estimation was done using the AMCL ROS package [13]. AMCL (Adaptive Monte Carlo Localization) is a probabilistic localization algorithm for a vehicle moving in a two-dimensional occupation grid map [14]. It uses a particle filter to find the pose of the robot by comparing its current estimated position, using the range-finder and odometry data, against a map. This means that initially the robot is uncertain of its location and has to be driven- or circled around for the particles to be updated.

### *E. Navigation & Decision Making*

This section still under development

By using the previous steps then navigation and decision making could also be implemented. This was done by utilizing the package `tiago_2dnav` to execute navigation to a specific location and using two nodes created by the group.

The nodes, object_detection_node and decision_making_node, allow for an object in the form of a human to be detected, along with the distance to the human. The decision making node listens to the previous node and when a human is detected within a certain preset distance then it will tell the TIAGo robot to stop.

## IV. Results

In figure 6 the created map of the lab room as well as the surrounding corridor can be observed. Note the rather convoluted lab room with several objects of some kind in blue or pink can be seen. These objects are chairs, tables or people who are standing in the lab room and they are picked up by the laser range finder.

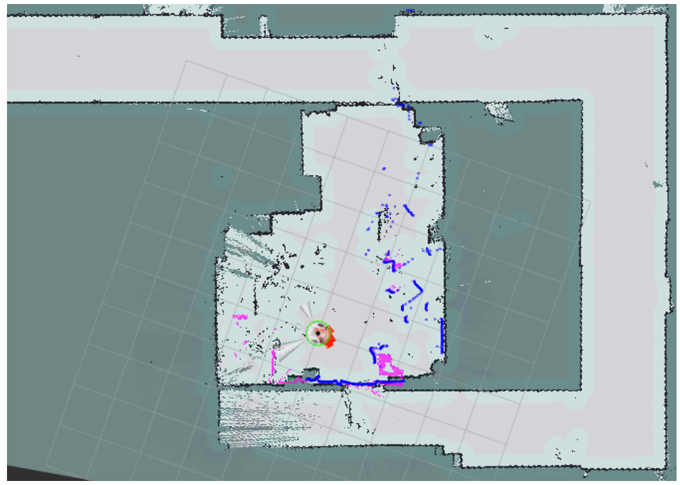

Fig. 6. The map that was created around the lab room and the surrounding corridors, along with the current location of the TIAGo robot.

Figure 7 shows the robot in a corridor with a door opened in front of it. The purple lines represent what the laser range finder is seeing at the moment while the black lines is the saved map. The short red lines on the TIAGo robot are the particle filter from the AMCL algorithm that is used to localize the robot on the map.

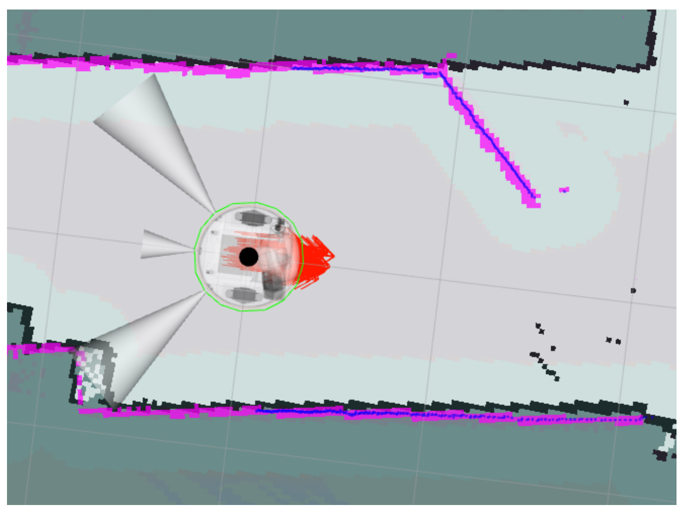

Fig. 7. The TIAGo robot localizign itself on the map in a corridor seeing a open door in front of it.

We are also planning to show graphs of the velocity profile and distance to objects as the TIAGo robot is navigating throughout the map and a human walks in front of it.

## V. Discussion

As could be seen in figure 6, tables, chairs and human legs were not that clear in the map. This could pose an issue as the TIAGo robot's laser range finder is close to the ground and it may not see certain objects well. Therefore the navigation was implemented in the corridor instead of inside of the lab. This could be resolved by also using the RGBD camera to augment the map with all the objects that the laser range finder couldn't see quite as well. Though this was not deemed necessary as for the project goal it did not require perfect navigation in a cluttered lab room.

Furthermore the localization as seen in figure 7 operates well as it can see new changes in the world such as an open door. The localization still has the same concern regarding table legs and chairs but in the corridor the localization performs as expected and required.

The decision making is rather simple as stopping if the TIAGo robot sees a human within a certain distance was the most appropriate decision to implement. This could be further developed by increasing either the complexity or the number of decisions possible.

Currently, the distance to an object is being calculated as the mean pixel depth for every pixel in the detected object. This might be a problem if the object has a large depth from the perspective of the camera, as it would lead to a faulty estimation of the object distance. An improvement could be to do statistical analysis on the depth data, and filter out the outlier.

## VI. Conclusion

Creating a map of the environment around the lab room and corridor as well as localizing the TIAGo robot inside of this map was successful. Furthermore the object detection could also detect a human and obtain the distance between the human and the TIAGo robot if the human could be viewed as rather flat. This allowed for the decision making to make an decision that if the human was close enough that it should stop.

More time could be used for further developing the decision making to be able to handle more complex scenarios, e.g finding new trajectory to goal, instead of only stopping the robot if it sees a human.

Overall the project has seen moderate success though with room for future work to improve some parts.

## References


[1] M. Misaros, O.-P. Stan, I.-C. Donca, and L.-C. Miclea, "Autonomous robots for services—state of the art, challenges, and research areas," *Sensors*, vol. 23, no. 23, p. 4962, 2023. [Online]. Available: https://doi.org/10.3390/s23104962.

[2] S. Nahavandi, R. Alizadehsani, D. Nahavandi, *et al.*, "A comprehensive review on autonomous navigation," *arXiv preprint arXiv:2212.12808*, vol. 1, no. 1, p. 48, 2022.

[3] "Tiago robot." Last accessed on 2023-12-07, Pal Robotics. (), [Online]. Available: https://pal-robotics.com/robots/tiago/.

[4] I. Dianov, K. Ramirez-Amaro, and G. Cheng, "Generating compact models for traffic scenarios to estimate driver behavior using semantic reasoning,"

[5] Y. D. V. Yasuda, L. E. G. Martins, and F. A. M. Cappabianco, "Autonomous visual navigation for mobile robots: A systematic literature review," *ACM Computing Surveys*, vol. 53, no. 1, 2020. DOI: 10.1145/3368961. [Online]. Available: https://doi.org/10.1145/3368961.

[6] "Gazebo simulator." Last accessed on 2023-12-11. (), [Online]. Available: https://gazebosim.org/home.

[7] Alpaca-zip. "Ultralytics_ros." Last accessed on 2023-12-07. (), [Online]. Available: https://github.com/Alpaca-zip/ultralytics_ros.

[8] "Yolov8 documentation." Last accessed on 2023-12-07, Ultralytics. (), [Online]. Available: https://docs.ultralytics.com/.

[9] J. Pages. "Create a map with gmapping." Last accessed on 2023-12-07. (), [Online]. Available: http://wiki.ros.org/Robots/TIAGo/Tutorials/Navigation/Mapping.

[10] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with rao-blackwellized particle filters," *IEEE Transactions on Robotics*, vol. 23, pp. 34–46, 2007.

[11] C. M. Revanth, D. Saravanakumar, R. Jegadeeshwaran, and G. Sakthivel, "Simultaneous localization and mapping of mobile robot using gmapping algorithm," in *2020 IEEE International Symposium on Smart Electronic Systems (iSES)*, 2020, pp. 56–60.

[12] S. K. NT *et al.*, "Mobile robot terrain mapping for path planning using karto slam and gmapping technique," in *2022 IEEE Global Conference on Computing, Power and Communication Technologies (GlobConPT)*, IEEE, 2022.

[13] J. Pages. "Localization and path planning." Last accessed on 2023-12-07. (), [Online]. Available: http://wiki.ros.org/Robots/TIAGo/Tutorials/Navigation/Localization.

[14] D. Fox, W. Burgard, F. Dellaert, and S. Thrun, "Monte carlo localization: Efficient position estimation for mobile robots," *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, 1999.