DTU Electrical Engineering
Department of Electrical Engineering

Group 13 – Final Assignment 31388-22

Szilárd Czibere-s212450, Eduard Fiedler- s210134, Karol Gabor-s213030, Mar Alba Mella-s205737, Giovanni Zaniboni- s217035

Navigation

To navigate to guidemarks, the available graph planning plugin was used. First, walls of the course were added, and the localizer initialized, such that the localizer plugin could be called every 1 second. This obtained a relatively updated transformation of the robot's position into world coordinates, that are used in the *drivew* command. Next, 34 points and their connections for the graph planner were set such that a route to the guidemarks was established (Fig. 1). In order to know where to go for each guidemark, an array is set containing the position at which the guidemark can be read from. This, in combination with an array containing the actual position of the guidemarks, allows a route to be planned to where a given guidemark can be seen and then faced to read its information.

1. The robot knows its initial position and the first guidemark it should drive to
2. The route is calculated from its current position to where the desired guidemark can be seen
3. The current robot pose is compared with the second graph planner point in the queue. If there is only 1 point in the queue, the robot faces that point.

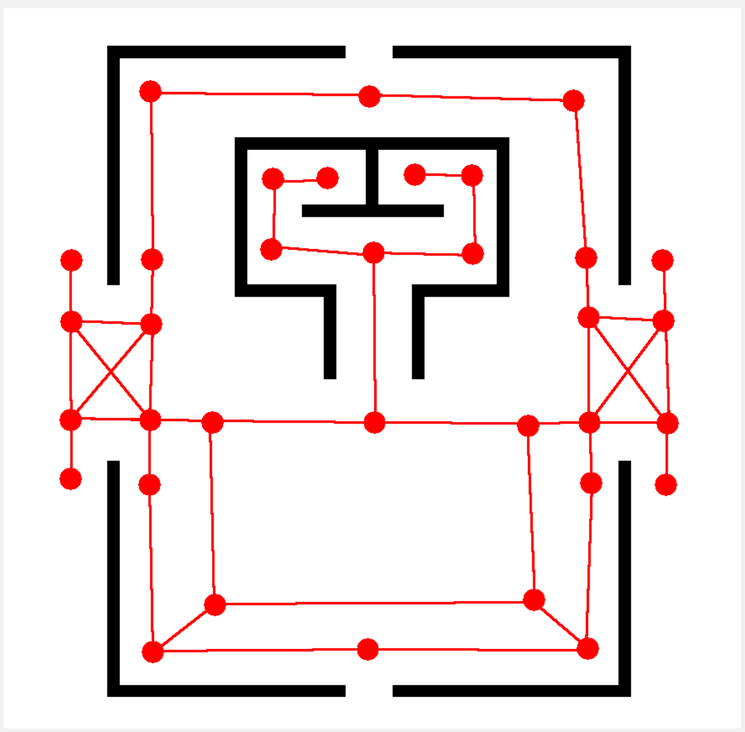

Figure 1. Map showing the points and connections of the graph planner

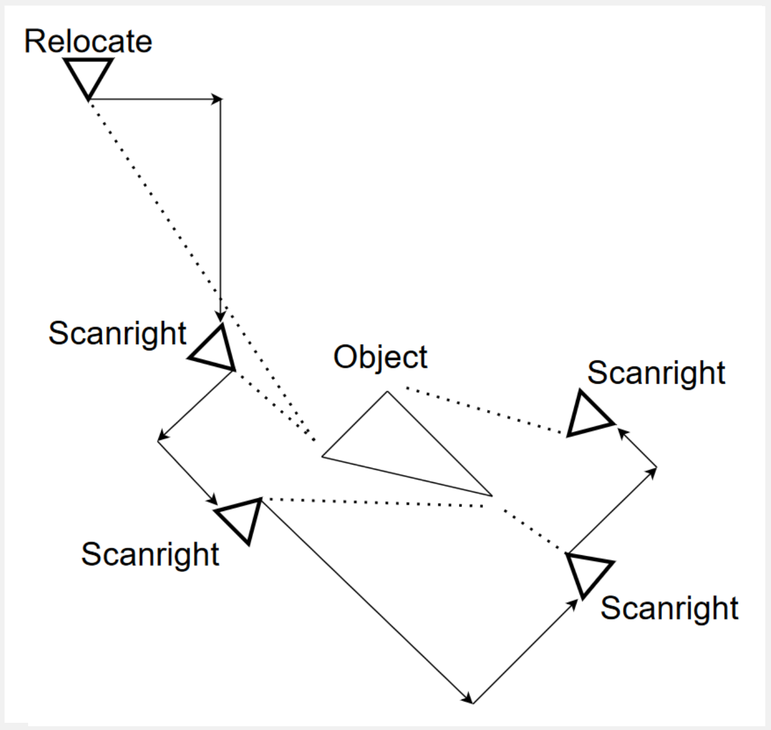

Figure 2. Object Detection Procedure

4. *drivew* commands are used to navigate through graph planner points until the
desired position is reached.
5. The current pose is compared to the position of the desired guidemark such that the robot turns towards the guidemark.
6. The ID contained in the guidemark is read and used as the next desired guidemark.
7. Repeat 2-6. until the read ID is greater than 20 as this indicates that guidemark 13 or 14 have been read.
8. Navigate to the position where the object detection begins.

Object Detection

The object detection starts with the known robot pose at the top of the area in which the object might be placed in. From there, the relocate plugin is called to find a position that is 0.7 units away from the right most detected point on the object, facing perpendicular to the found side (see Fig. 2). Then, the right most point of the side the robot is facing is scanned using a scanright plugin and converted into world coordinates. From this position, the robot relocates to face this scanned position at a ninety-degree angle 0.5 units away. The right most point is scanned from there, and the process is repeated until 4 data points are gathered. This method ensures that the gathered data is most likely comprised of corners of the shape and is ordered in a sequence.

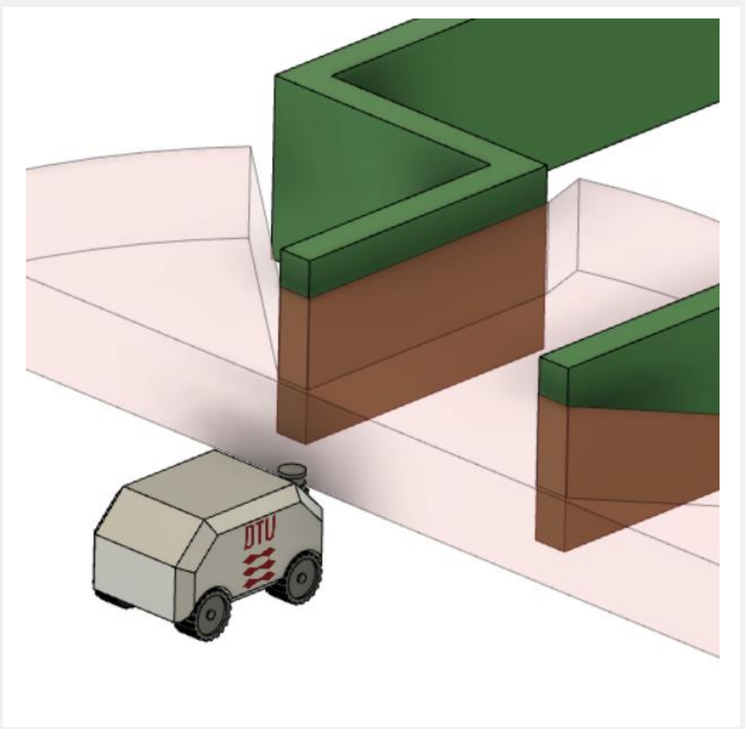

Figure 3. Robot entering the maze.

Next, the data is processed, as in the case of a triangle, one of the data points will be redundant. This is done by taking the mean of data points that ended up close to each other. Afterwards, the internal angles and side lengths are calculated allowing the classification of the object as well as its orientation and position (for the triangles). In the case of the rectangles, the position is determined by taking the mean of the data points. Generally, this method gives an accuracy in the position and orientation of +/- 0.1. When relocating around the object, the robot will not exceed y = 2.3, as this increases the likelihood of running into the maze walls. Instead, the robot will take an alternate path that will allow it to scan the object uninhibited.

After identifying the object, the robot will turn away from the object and drive until it is outside the box in which the object may be placed in. Then, it uses the graph planner points to navigate to the starting position.