RWTH Aachen University - LELY Solution Documentation

RWTH Team

June 29, 2022

1 Abstract

Our envisioned solution to the LELY Challenge consists of a team of asymmetrical robots with a control system implemented in ROS 1. Robot A navigates and maps its environment with use of a 2D LIDAR system. Robot B recieves a map of the environment and location data from Robot A and then uses its ultrasonic sensors to navigate its environment.

This document provides an outline of the design of the systems and the thought process behind their development.

All code, CAD files, and this document itself are available at the following GitHub repository: https://github.com/URSAREN/ERF22

Further explanatory comments are included in the source code files.

2 Robot A

2.1 Overview

Robot A is a LIDAR-capable platform. Due to the walls of the challenge being only 37cm tall, the mounting point of the LIDAR is on the bottom front of the chasis, providing a 200 degree field of view. The restricted point of view is overcome through the natural turning of the robot during navigation or through a simple and relatively quick rotation.

2.2 Sensors

A RPLIDAR A1M8-R6 LIDAR is mounted on the front of the robot. The LIDAR interfaces with the laptop directly over USB.

The A1M8-R6 was chosen due to its excellent support in the ROS ecosystem, allowing for the implementation of a straightfoward mapping stack.

2.3 Hardware

A 3D-printed mount was created for easy placement and adjustment of the LIDAR. It consists of two parts: the baseplate, which is directly connected to the sensor, and the connector plate, which is taped on the lower platform of the Juno. Both plates are connected through two screws, which form a one degree of freedom joint. This allows the horizontal adjustment of the baseplate, to gain more accurate data from the LIDAR.

Figure 1: The LIDAR sensor and 3D-printed mounting point attached to Robot A

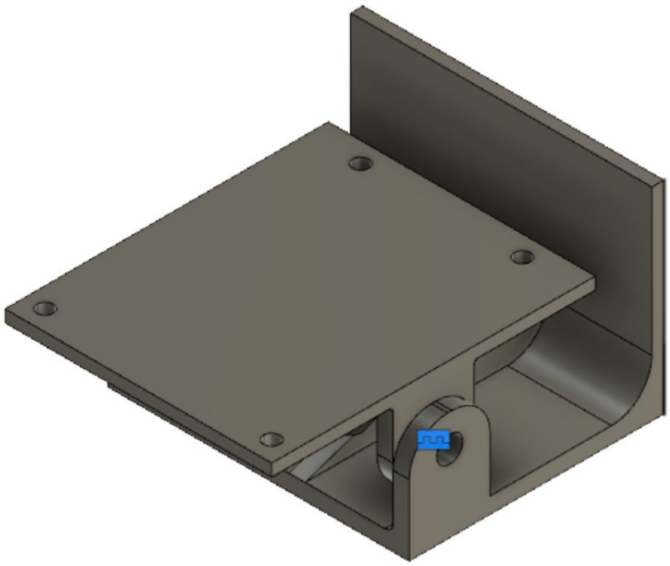

Figure 2: CAD rendering of LIDAR mount

2.4 Software

The A1M8-R6 LIDAR comes with ROS packages which allow for SLAM mapping without odometry information up to 12 meters. An ideal navigation stack for Robot A would be to use the gmapping ROS package to generate a cost map, allowing for the robot to be steered via the ROS move_base package. However, due to technical difficulties with move_base, debugging and testing of this algorithem was not possible.

Attempts at remediating this were unsuccessful within the given time frame.

3 Robot B

3.1 Overview

Robot B uses ultrasonic sensors mounted via aluminium frame to navigate its environment.

Figure 3: Robot B with mounting frame, 3D-printed connectors, and ultrasonic sensors

3.2 Sensors

Three HC-SR04 ultrasonic sensors are placed in an array infront of the robot. The sensors interface with the laptop via an Arduino Uno.

A problem faced whilst designing the sensor array was throughput limitations on the Arduino. A single HC-SR04 requires two digital pins for input-output, allowing for a maximum of three sensors per Arduino, as opposed to the originally planned five.

3.3 Hardware

The sensor mounting for Robot B was far more complicated, because it has a complete casing. For this reason, an aluminum frame existing of four edge tubes with 3D-printed connection points extends from the top mounting point to the bottom of the robot, allowing the sensors to attach to a low-enough point to read the walls of the maze.

For more accurate sensor data, the ultrasonic sensors should be at the same height as the LIDAR. To implement this, the perpendicular tube is adjustable in height through a clamping mechanism in the connector to the horizontal tube.

To mount all the ultrasonic sensors in a flexible configuration, the mount consists of four different components. Through them, it is possible to adjust the angle and the level of the sensors.

Additionally, they can be easily installed in other configurations or removed for testing reasons.

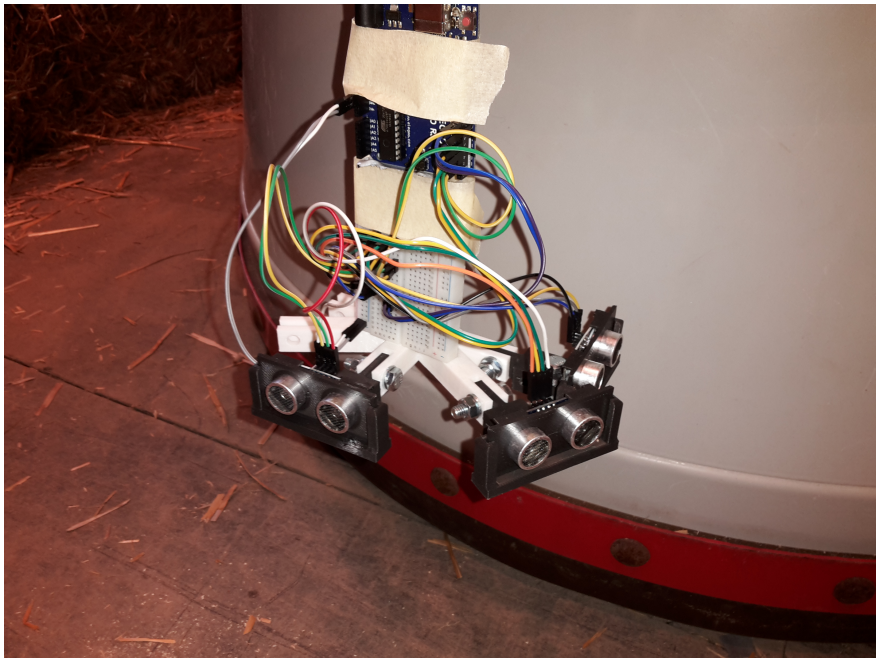

Figure 4: Robot B with mounting frame, 3D-printed connectors, and ultrasonic sensors

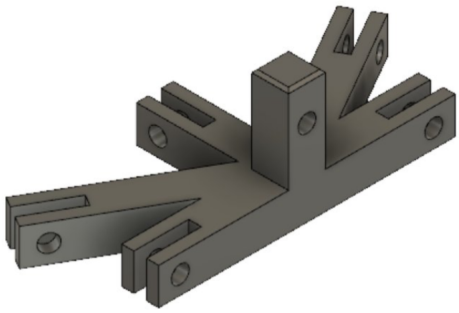

Figure 5: Central mount for connection points

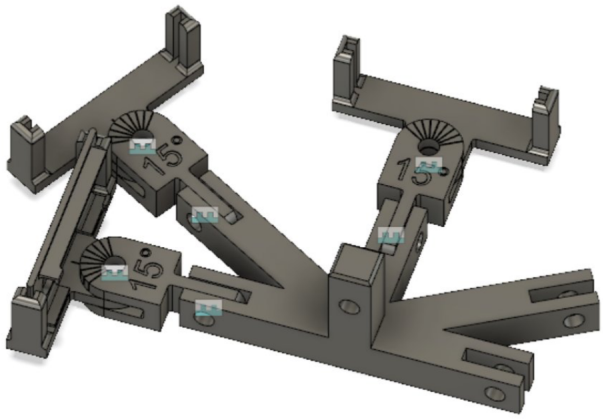

Figure 6: Central mount with connections for ultrasonic sensors

3.4 Software

The ideal algorithem for Robot B is similar to the ideal case of Robot A. After being having recieved the LIDAR map and its own position and orientation from Robot A, Robot B uses a wall following and corner counting algorithem to navigate.

This was not realised due to the previous mentioned troubles whilst developing the code for Robot A.