

Mapping Robot

Mapping Robot Controlled via Mobile Device

Corey Jamison

-

Joel Keeling

-

Mark Langen

Summary

Mapping classroom sized spaces via a rangefinder equipped robot controllable via a mobile application

Abstract

This project aimed to build a robot that was capable of generating a relatively detailed two-dimensional map of a flat area with a size roughly equivalent to a small classroom. The robot would communicate with a mobile application through which a user could view the map as it was constructed, as well as manually control the robot.

The resulting robot is constructed using a small two-wheeled skid-steer platform. A LIDAR range-finder is mounted onto a stepper motor to provide scanning capabilities. At its core, the robot is controlled by a Terasic DE0 Nano FPGA board with several soft cores to perform its various I/O tasks. The robot performs a full scan of its surroundings after each movement in order to adjust its position. Its mapping algorithm relies on this scan data to determine its position and adjust the map – no GPS, accelerometer, or compass data is required. The only odometry data that is used is a very rough estimate of how far the robot turned, or what direction it moved in, which can be used to disambiguate motions in very symmetrical environments. This avoids drift problems that can arise with other absolute positioning methods, and reduces the hardware demands at the cost of increasing the software complexity.

Table of Contents

Summary	1
Abstract.....	2
Table of Contents	3
Functional Requirements	5
Design and Description of Operation.....	6
Design	6
Description of Operation	7
Bill of Materials.....	8
LIDAR Lite v1 Laser Range Finder	8
SparkFun DC Motor Driver (TB6612FNG)	8
DFRobot 2WD Mobile Platform	8
SparkFun Bluetooth Mate Silver	9
2 Phase Bipolar Stepper Motor.....	9
DE0 Nano Eval Board	9
Misc.....	9
Total Cost.....	10
Available Sources	10
Datasheet	12
Robot Power Consumption Characteristics	12
Operating Conditions.....	12
Acceptable Voltage Input ranges for Components.....	12
Table of Inter-Component Signals	12
Background Research.....	16
Software Design.....	16
Test Plan.....	17
Mapping Algorithm.....	17
Bluetooth Module	18
Laser Scanning Data.....	18
Stepper motor.....	18
Chassis Wheels.....	18
Integration Testing.....	18

Results of Experiments and Characterization.....	20
Motor Acceleration / Max Speed Suitability Verification	20
Total Mass of Platform (rounded up for conservative estimate).....	20
Torque of DC Motors	20
Max Speed.....	20
Rotation of LIDAR Sensor	20
Safety	21
Voltages and Current.....	21
DE0 Nano	21
LIDAR Lite	21
DC Motor Driver	21
Sparkfun Bluetooth Mate Silver.....	21
DFRobot DC Motors	21
Stepper Motor	21
Stored Energy	21
Max Kinetic Energy.....	21
Operating Temperature	22
Laser Safety	22
Regulations and Society	22
Environmental Impact	23
Sustainability.....	23
References	24
Appendices	25
A: Quick Start Manual.....	25
Assembly.....	25
Brackets.....	25
Connections.....	25
Batteries	28
Compiled files.....	28
Flashing the DE0	28
Installing the Android application.....	28
Demoing the project	28
Powering the Robot	28

Connecting the Android application.....	29
Using the Mapping Robot.....	30
Reading the map.....	31
B: Future Work	32
C: Hardware Documentation	33
D: Source Code.....	33

Functional Requirements

The robot is capable of performing accurate scans with a resolution of 1 cm, within an accuracy of ± 2 cm, through a rotation of approximately 325° with a new measurement taken every 1.8° . The blind spot is due to a bracket that was installed to ensure that scans always begin at the same orientation. This scanning process takes approximately 3.5 seconds to complete, and occurs every time a movement is completed.

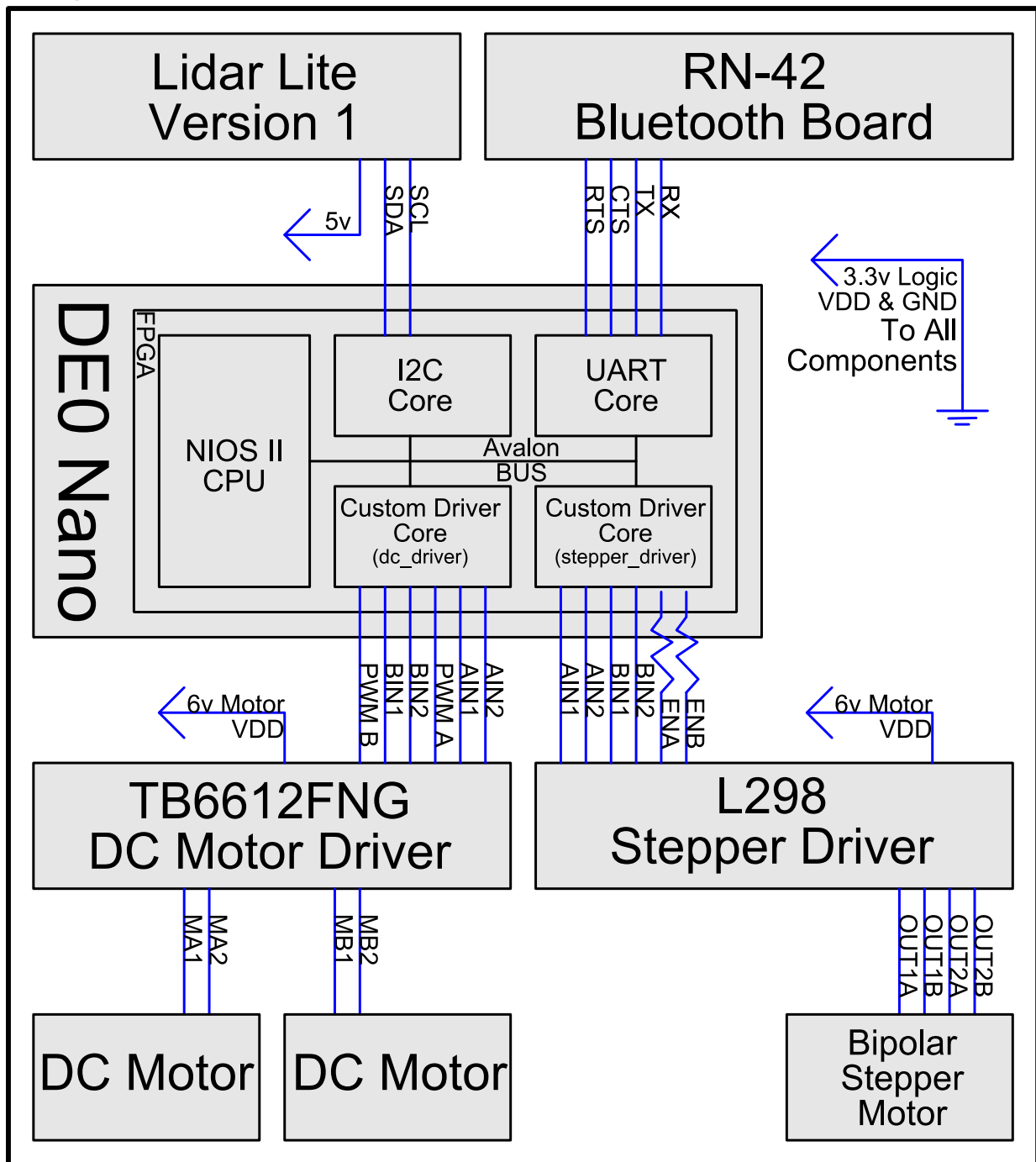
Movement is restricted to either linear motion or rotation, and after each movement a scan must be performed in order to infer the new position or orientation of the robot. Movement is controlled by the user of the Android application – the user can command the robot to drive forwards or in reverse, or rotate clockwise or counter clockwise. Each packet is sent three times by the android application to ensure its safe arrival, and while the button is held each command is repeated every half-second. A watchdog timer stops the motors if a new command is not received for one second – this prevents the robot from continuing to drive if the Bluetooth connection fails.

The data from the LIDAR sensor is transmitted as soon as it is available to the android application over Bluetooth. Each packet is sent three times and verified with a checksum at the receiver to avoid packet loss or corruption. Additionally, the robot sends packets indicating when a scan is beginning and when a scan is finished. Upon scan completion, the Android application uses the data received since the scan began to perform an iteration of the mapping algorithm, adjusting the robot's position in the virtual map and then refining the map using the aforementioned data.

The android application user interface allows the user to select from a list of paired Bluetooth devices and, upon selecting a device, displays the current map in progress with four directional keys that the user can hold to drive the robot. The map displays the robot's current position, the latest set of LIDAR data, and the line segments created by the mapping algorithm, with the newest line segments highlighted in red.

Design and Description of Operation

Design



The hardware of the robot is broken up into four main sections – LIDAR measurements, Bluetooth communications, the stepper motor (for rotating the LIDAR), and the two DC motors (for moving the robot). These four sections each interface with a soft core in the FPGA. Their operation is coordinated by a NIOS II soft CPU core. The CPU runs a simple C application broken into several tasks, each responsible for a part of the functionality.

The LIDAR communicates with the built-in Altera I2C core. In order to communicate with the unit, a code representing POLL is written to the device's registers, and after a delay of 20ms, the distance (in cm) can be read from the registers. The bytes returned are then concatenated together to produce the resulting integer distance measurement.

Bluetooth communications also use a built-in Altera core, in this case the UART core. The RN-42 Bluetooth chip emulates a serial port and as such it is sufficient to simply attach the UART Tx line to the Bluetooth Rx, and vice versa. Once this configuration is complete, the Bluetooth chip will fire an interrupt when a new byte has been received, and writing a byte to its register will cause it to be transmitted immediately.

Two additional features are implemented on top of this byte transmission capability. First, bytes are concatenated into packets by looking for START_PACKET and END_PACKET bytes. Any bytes in between those bytes are combined into a byte array to be processed. Additionally, the first two bytes in each packet store its ID and length, and each packet is sent three times. In the android application, the length is used for a simple checksum, and the first packet with each ID is passed along to be used – subsequent packets with a given ID, and packets that fail the checksum are discarded.

The stepper motor is controlled by a custom VHDL core that interfaces with the L298 stepper driver chip. This driver core stores a desired position internally alongside the current position and adjusts the stepper's position towards the desired position at a configurable speed, allowing a delay between steps to allow the LIDAR a chance to scan. The DC motors are also controlled by a custom VHDL driver core that communicates with the TB6612FNG chip.

Description of Operation

While operating, the robot maintains a simple state machine to ensure correct operation. There are three primary states:

- 1) Idle
- 2) Moving
- 3) Scanning

In the idle state, the robot is able to receive commands from the android application. Upon receiving a command, one of two things will occur. When an android device initially connects, it sends a packets that causes the robot to perform a scan immediately. Otherwise, a command will tell the robot to move.

While moving, the robot ignores all commands except the stop command. This is to ensure that the movement of the robot matches what is expected by the algorithm. Upon receiving a stop command (the software does not distinguish between a stop sent by the android application, and a stop caused by the watchdog timer), the robot will begin a scan.

While scanning, the robot rejects all commands until the scan is complete. This is inevitable as the full set of data is necessary for mapping. Once the scan is complete, this is communicated to the android application and the robot returns to the idle state.

In the Android application, the data from the robot is collected during a scan, and once the scan is completed the mapping algorithm will perform localization using the new data, then update the map, and finally update the display with that new map and position of the robot in it. The android application waits for user input, and sends commands every half-second when a button is held in order to maintain the requested movement.

Bill of Materials

LIDAR Lite v1 Laser Range Finder

Description: Laser Range Finder

Cost: \$84.95 USD

Specifications: 3.3V I2C interface, Max Rep Rate: 100 Hz, Range: 1- 40m, 5V <100mA current draw

Supplier Link: <http://www.zagrosrobotics.com/shop/item.aspx?itemid=955>

Datasheet: <https://cdn.sparkfun.com/datasheets/Sensors/Proximity/LIDAR-Lite-Laser-Datasheet.pdf>

SparkFun DC Motor Driver (TB6612FNG)

Description: DC Motor driver capable of driving 2 DC motors. Controlled using digital input signals.

Cost: \$8.95 USD

Specifications: Motor Voltage: 4.5 - 13.5V, Up to 1 A for each motor, 3.3V logic for input signals

Supplier Link: <https://www.sparkfun.com/products/9457>

Datasheet: <https://www.sparkfun.com/datasheets/Robotics/TB6612FNG.pdf>

DFRobot 2WD Mobile Platform

Description: 2-Wheeled robot chassis. Includes two DC motors and 5 slot battery pack

Cost: \$49.30 CAD

Specifications: Two 6V DC motors included, Motor stall current 470mA, total weight: 400g

Supplier Link: <http://www.robotshop.com/ca/en/dfrobot-2wd-mobile-platform-arduino.html>

Datasheet: <http://www.robotshop.com/media/files/pdf/datsheet-rob0005.pdf>

Motor Link: <http://www.robotshop.com/en/dfrobot-6v-180-rpm-micro-dc-geared-motor-with-back-shaft.html>

SparkFun Bluetooth Mate Silver

Description: Module that contains a Bluetooth IC. Communication to the module is done using UART.

Cost: \$24.95 USD

Specifications: Max current draw of 50mA, Bluetooth 2.1/2.0/1.2/1.1, UART interface

Supplier Link: <https://www.sparkfun.com/products/12576>

Datasheet: <http://cdn.sparkfun.com/datasheets/Wireless/Bluetooth/Bluetooth-RN-42-DS.pdf>

2 Phase Bipolar Stepper Motor

Description: Standard stepper motor.

Cost: \$20.95 USD

Specifications: 12VDC, Max: 480mA, weight: 230g, holding torque 0.6 Kg-cm

Datasheet: <http://www.ece.ualberta.ca/~cmpe401/docs/105881.pdf>

DE0 Nano Eval Board

Description: Evaluation board for FPGA unit

Cost: \$120.33 CAD

Specifications: Altera Cyclone IV FPGA, GPIO headers, external power 3.6-5.7V

DataSheet: www.terasic.com.tw/cgi-bin/page/archive_download.pl?Language=English&No=593&FID=75023fa36c9bf8639384f942e65a46f3

Misc

40-pin Ribbon Cable

Cost: \$10.00 CAD

6-slot battery pack

Cost: \$6.00 CAD

Perf Board

Cost: \$7.50 CAD

Additional Small Screws

Unit Cost: \$0.02

Quantity: 19

Total Cost: \$0.38 CAD

Additional Small Nuts

Unit Cost: \$0.02

Quantity: 19

Total Cost: \$0.38 CAD

Additional Large Stepper Screws

Unit Cost: \$0.10

Quantity: 2

Total Cost: \$0.20 CAD

Meccano LIDAR Mount

Cost: \$5.00 CAD

Connectors

Unit Cost: \$0.30
Quantity: 4
Total Cost: \$1.20 CAD

Headers

Unit Cost: \$0.04/pin
Quantity: 92pins
Total Cost: \$3.68 CAD

Crimps

Unit Cost: \$0.05
Quantity: 12
Total Cost: \$0.60 CAD

STM L-298 H-Bridge

Cost: \$6.17CAD

Resistors

Unit Cost: \$0.15
Quantity: 2
Total Cost: \$0.30 CAD

Diodes

Unit Cost: \$0.30
Quantity: 8
Total Cost: \$2.40 CAD

Slider

Cost: \$1.25 CAD

Total Cost

Total USD: \$139.80 USD

Total CAD: \$214.69 CAD

Using the current USD to CAD conversion rate, total cost: \$397.41 CAD

Available Sources

The project uses Altera SOPC Builder auto generated code for the NIOS II Soft-Core that will be programmed into the DE0 Nano board.

The LIDAR sensor has a relatively simple open source Arduino Driver available on Github, which demonstrates basic communication with the LIDAR over I2C lines. We wrote a custom driver based on that open source driver.

To communicate between our processor and LIDAR unit we used I2C communication, which requires a soft-core. Altera provided demo projects that provided us with basic I2C software drivers.

Altera provides a UART core with an Avalon interface that can communicate to Nios II processor cores easily [13]. We used this core for interfacing between our processor and Bluetooth module.

Some open-source SLAM packages exist, but they were not used in the project. Most of those packages have very heavy RAM and / or CPU / GPU speed requirements, which would totally rule out running them on the robot itself, and cause issues even running them on a mobile app.

Datasheet

- performance, user perspective block diagrams, operating conditions
- expand on IO signals to produce a datasheet of the complete system
- provide calculated and measured power in various modes of operation (peak, idle, standby)
- Record V and I off bench power supply
- describe how each is calculated and measured

Robot Power Consumption Characteristics

	Idle without Bluetooth	Idle with Bluetooth	While Scanning	While Moving
DE0 (4.5V)	380mA	365mA	380mA	365mA
Motors (6V)	180mA	180mA	270mA	700mA

Operating Conditions

- Keep robot dry
- Robot is for indoor use only, and runs on flat surfaces only.
- The robot works best when it is at least 30cm away from the nearest obstacle (It will often work fine even with some very close obstacles, however performance will degrade as more are added)

Acceptable Voltage Input ranges for Components

DE0 Nano: 3.6 – 5.7V

DC Motor: 5 – 6V (nominal), 3 – 7.5V (operational)

Stepper Motor: 5 – 20V

Table of Inter-Component Signals

Type of Signal	Signal Name	Power Supplied	Origin	Pin #	Destination	Pin #
Serial	UART_RX	N/A	BT Module	13	DE0 GPIO	0
Serial	UART_TX	N/A	BT Module	14	DE0 GPIO	1
Serial	UART_RTS	N/A	BT Module	15	DE0 GPIO	Not in use
Serial	UART_CTS	N/A	BT Module	16	DE0 GPIO	Not in use
Wireless	BT Wireless 1.1/1.2/2.0/2.1	3.3V 12-50mA	BT Module	N/A	Mobile Device (BT)	N/A
Digital	Motor A IN1	N/A	DE0 GPIO	4	Motor Driver Board	21
Digital	Motor A IN2	N/A	DE0 GPIO	5	Motor Driver Board	22
Digital	Motor B IN1	N/A	DE0 GPIO	6	Motor Driver Board	17
Digital	Motor B IN2	N/A	DE0 GPIO	7	Motor Driver Board	16
PWM	Motor A PWM	N/A	DE0 GPIO	2	Motor Driver Board	23
PWM	Motor B PWM	N/A	DE0 GPIO	3	Motor Driver Board	15
Digital	Motor STBY	N/A	DE0 GPIO	8	Motor Driver Board	19
DC	Motor A O1	2.7 V 1.2 A	Motor Driver Board	1+2	Wheel Motor A	N/A
DC	Motor A O2	2.7 V 1.2 A	Motor Driver Board	5+6	Wheel Motor A	N/A
DC	Motor B O1	2.7 V 1.2 A	Motor Driver Board	11+12	Wheel Motor B	N/A
DC	Motor B O2	2.7 V 1.2 A	Motor Driver Board	7+8	Wheel Motor B	N/A
I2C	LIDAR SCL	N/A	LIDAR Sensor	4	DE0 GPIO	15

I2C	LIDAR SDA	N/A	LIDAR Sensor	5	DE0 GPIO	16
AC	OUT1A	6V	Stepper Driver	16	Stepper Motor	N/A
AC	OUT1B	6V	Stepper Driver	21	Stepper Motor	N/A
AC	OUT2A	6V	Stepper Driver	9	Stepper Motor	N/A
AC	OUT2B	6V	Stepper Driver	4	Stepper Motor	N/A
Digital	Stepper AIN1	N/A	DE0 GPIO	11	Stepper Driver	AIN1
Digital	Stepper AIN2	N/A	DE0 GPIO	12	Stepper Driver	AIN2
Digital	Stepper BIN1	N/A	DE0 GPIO	13	Stepper Driver	BIN1
Digital	Stepper BIN2	N/A	DE0 GPIO	14	Stepper Driver	BIN2
Digital	Stepper ENA	N/A	DE0 GPIO	9	Stepper Driver	ENA
Digital	Stepper ENB	N/A	DE0 GPIO	10	Stepper Driver	ENB

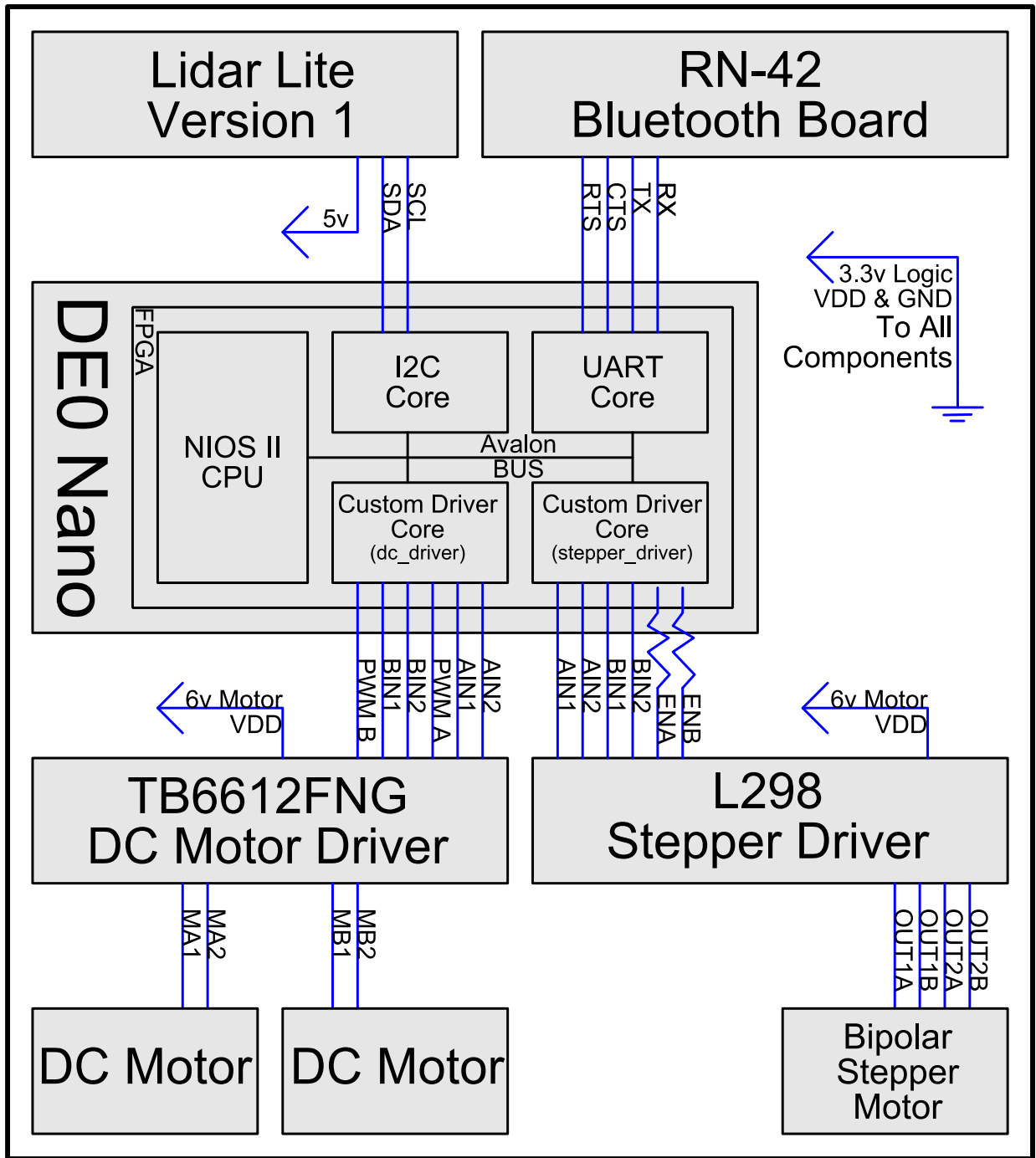


Figure: Overview of IO Signals

Background Research

The problem that we are attempting to solve is a widely studied problem in the field of robotics known as SLAM (Simultaneous Localization And Mapping). OpenSLAM.org is a platform that provides a variety of projects attempting to solve SLAM, open to the public to view and modify for research purpose. At the time of writing, OpenSLAM.org had 33 projects listed, ranging from 2D algorithms using measurements such as laser or sonar to fully 3d solutions using multiple cameras. One such algorithm is titled SSA (Sparse Surface Adjustment). It is very similar to our approach, as it uses laser scans to produce graph-based maps. Its algorithm is described in the paper “Highly Accurate Maximum Likelihood Laser Mapping by Jointly Optimizing Laser Points and Robot Poses”[14], and its source code is licensed under the LGPL v3 and available from OpenSLAM [15].

Software Design

The main software component in the project is the mapping algorithm, packaged as a java library to be used by the android application. The mapping algorithm works through standard SLAM (Simultaneous Localization and Mapping) techniques. The following is a high level overview of the algorithm (See Appendix D for a link to the full mapping library source code). It includes three phases when updating the map and location using a new set of data:

1) Feature Extraction

The input data to the mapping algorithm is a set of angles and distances, which can be immediately converted into points in the world using the current position. The first step is to extract some “features” from that set of points, which can be matched to existing features in the world to determine the current position. The algorithm for the robot used line segments as its features. The line segments are extracted by looking for groups of points which are collinear within a given slop allowance related to the variability of the LIDAR measurements. A linear regression is then done on the collinear groups of points to generate the final line segment features for a given scan.

2) Feature Matching

The features that were extracted are then matched to the features in the existing map. Given the constraint that the robot only turns or moves linearly in a single movement, the algorithm can reduce the degrees of freedom of the problem significantly. For a linear motion, segments with a similar rotation to segments in the map are compared, giving a position delta along the direction of movement. For angular motion, segments with a similar perpendicular distance to the robot are compared to give a rotation delta. These deltas are then put into a histogram with some additional weighting factors and the highest weight delta is chosen as the true delta, and applied as a change to the robot position or rotation.

3) Map Updates

Now that robot knows its new position in the world, the map can be updated using the segments. Segments which do match well with existing segments can be used to refine the estimate of those segments position and direction in the map. Segments which do not match well to any segment are used to extend the map as the robot sees new areas which were not originally in its view.

The robot itself is a “dumb terminal” for the android application. Simply taking commands, executing them, and returning back sensor data. The logic in the robot can be summed up as a three-state state machine with the following states:

1) (Initial State)

At startup the robot drives the stepper motor -360 degrees running it into the rotation stop to get it registered at the zero rotation regardless of what state it was in to begin with.

2) Idle

In the idle state, the robot waits for a command from the user. When one is received, it moves to the Moving state.

3) Moving

When entering into the moving state, the motors are started to drive the robot in the desired way. While the moving state, the robot waits for a “stop” command from the user, ignoring other commands. When moving out of the moving state, the drive motors are turned off.

4) Scanning

Entering the scanning state the robot sends a “start scan” packet to the phone. In the scanning state, the stepper motor is moved through a 300 degree sweep, taking intermittent scans every 1.8 degrees using the LIDAR, starting at the zero point that it was driven to at startup or back to after a operation. After each sensor reading a “scan datum” packet is sent to the phone. Finally, a “scan complete” packet is sent to the phone, and the robot returns to the Idle state.

Test Plan

Each module was tested individually. The interfaces between modules are largely not needed to test them, and they can be tested individually through testbeds.

Mapping Algorithm

We have developed a desktop application that can be used to generate the input that the algorithm will be used. The application provides a GUI for the user to manually control a virtual simulation of the vehicle, and displays in real-time the algorithm’s current best-guess map overlaid atop the actual map provided. The mapping algorithm will be implemented as a library,

in order to decouple the library from the I/O coming from either the simulation or the robot, and facilitate swapping it over once the robot is providing data.

Bluetooth Module

The Bluetooth module will be fairly simple. It will transmit JSON packets bi-directionally over a short distance to a single client. This can be implemented independently of the rest of the functionality, as the data being transmitted over the connection is unimportant for testing the connection. In order to test this, a very basic Android application to display text output will need to be implemented. Once a basic connection has been established and tested, we will also be testing its performance in various conditions: at distance, around/through obstacles, and sending large volumes of data.

Laser Scanning Data

The data coming from the laser scanner will be tested as well. In order to test this, we will need to integrate the laser into our system, and display its output over the console. We will need to test its accuracy, and any bias it may have (overestimating/underestimating) by comparing with an actual measurement using a tape measurement. In addition, we will need to test its angular accuracy by slowly moving targets perpendicularly to the beam and recording when the laser picks them up.

Stepper motor

The motor upon which the laser will sit will be tested independently from the laser. We will need to be able to both control this motor's rotation precisely and accurately, and read in its current rotation. In addition, we will need to test the speed at which it can be rotated while still maintaining accurate data.

Chassis Wheels

The chassis will need to be able to both drive consistently in a straight line, and rotate in place with minimal drift. This will need to be tested, both in order to properly calibrate it to minimize drift, and also to record which could be problematic sources of error. Fortunately, errors are accounted for within the algorithm, so perfect accuracy is not required.

Integration Testing

The modules will be integrated together in steps. First, the laser sensor and the stepper motor upon which it rotates will be integrated, to ensure that the stepper motor and laser can both generate accurate data at the same rate, as well as how long the motor takes to physically rotate. In addition, the wiring will have to be tested for robustness while rotating.

The next step will be integrating Bluetooth. Getting a display working on the mobile device is an incredibly important part of the demonstrability of the project. Even though the final output isn't available, displaying the measurements on the phone will be a good stress test of the Bluetooth module, and displaying them will be a test of the phone app.

Next, the software algorithm will be used to generate actual data to send to the phone app, which will allow it to display its actual output.

Finally, the chassis wheels will be integrated with the mobile app's control signals, to allow for driving the car as well as testing the mapping performance with moving data.

Results of Experiments and Characterization

Motor Acceleration / Max Speed Suitability Verification

Total Mass of Platform (rounded up for conservative estimate)

Platform: 400g

DC Driver Motors: 2x 50g

Lidar Lite & Housing: 20g

DE0 Nano: 50g

Stepper Motor for Lidar Rotation: 300g

Additional Supporting Framework: 100g

= *Total Mass: < 1kg*

Torque of DC Motors

1.92kg.cm (from datasheet) x2 motors = 3.84kg.cm

Convert to N.cm = 3.84 x 9.81 = 37.7N.cm

Wheel size: 7cm diameter - > 3.5cm radius

Acceleration: 37.7N.cm / 3.5cm / 1kg =

= *Robot Acceleration: 10.7 m/s²*

Max Speed

Max Motor RPM: 100 RPM = 1.66 rotations / s

Wheel Circumference: 0.07m * PI = 0.22m

Max Speed: 0.22m / rot * 1.66 rot / s = 0.37 m/s

= Robot Max Speed: 0.37 m/s

Time to Accelerate to Max Speed:

Acceleration Time: (0.37 m/s) / (10.7m/s²) = 0.03s

= *Time to Accelerate: 1/30 of a second*

Rotation of LIDAR Sensor

Sensor Housing Dimensions: 20 x 50 x 35mm

Sensor Mass: 20g

Moment of Inertia: $I = 20g * ((50mm)^2 + (35mm)^2) / 12 = 6200 \text{ g.mm}^2 = 6.2 \text{ kg.mm}^2$

Stepper Motor Break Torque: 25.5 N.cm = 255 N.mm = 255000 kg.mm² / s²

Resulting Angular Acceleration:

$\alpha = 255000 \text{ kg.mm}^2/\text{s}^2 / 6.2 \text{ kg.mm}^2 = 41000 \text{ rot/s}^2$

Steps on Motor: 100

Time to Accelerate One Half Step:

$\theta = 0.5 * \alpha * t^2 \rightarrow 2 * \pi / 200 = 0.5 * 41000 * t^2 \rightarrow t = 0.0012s$

Time to Accelerate up and down through one step: 2x 0.0012s = 0.0025s

= *~400 steps/s max by the LIDAR*

These are all well within our desired operating range. The robot's movement speed will allow us to map our mid-sized classroom size requirement (10x10m) in 1-3 minutes depending on how many passes it has to make over the space. The stepper motor's max rotation rate is greater than the rate that the LIDAR can sample at, so it will not bottleneck the ability of the sensor to get range measurements at a value lower than we wanted when we selected the sensor.

Safety

Voltages and Current

DE0 Nano

5.7V(max), 500mA(max) [1]

LIDAR Lite

6V(Max), <100mA(continuous operation) [7]

DC Motor Driver

15V(Max), 3.2A(peak), 1.36W(power dissipation) [6]

Note that the driver will be powered using the voltage required by the motors (in this case nominal voltage is 6V) [4]

Sparkfun Bluetooth Mate Silver

3.6V(Max), 50mA(Max) [10]

DFRobot DC Motors

7.5V(Max), 6V(Nominal), 2.8A(Stall Current) [4]

Stepper Motor

12V, 480mA [8]

Even though some of our components can handle larger voltages and currents it appears that the highest voltage used will be 12V (Stepper motor) with peak current at 2.8A (DC motors).

Stored Energy

Two Battery packs one with 4 AA batteries and the other with 3 AA batteries.

7 batteries * 2.850Ah * 1.5V / battery = 29.925Wh

Max Kinetic Energy

$E_k = 0.5mv^2$

estimated mass (m) < 1kg

max velocity = 0.35m/s max (see results of experimentation and characterization)

$E_k = 0.5 \times 1 \text{ kg} \times (0.35\text{m/s})^2 = .06\text{J}$

Which is an insignificant amount of kinetic energy.

Operating Temperature

Cyclone IV FPGA -40C to 125C [2]

LIDAR Lite 70C(Max) [7]

DC Motor Driver -20C to 85 C [6]

Bluetooth Mate Silver -40C to 85C [10]

DFRobot DC Motors -10C to 60C [5]

Stepper Motor 80C(Max Temperature Rise), -20C to 50C (Ambient Temperature) [8]

Stepper Motor Driver -20C to 85C [9]

It appears that our operating should be -10C to 60C to keep all of the components happy. We don't expect many of the components to be operating close to their limits and therefore don't expect too much heat to be generated. Of course the DE0 nano board may heat up due to the presence of the FPGA. We don't not anticipate this to be an issue however if need be a heat sink can be applied to the FPGA unit.

Laser Safety

The LIDAR Lite unit that will be used for range finding purposes is rated as a Class 1 laser[7]. This means that the laser we will be using will not provide any sort of danger to human eyes unless is is passed through a sufficiently powerful telescope or microscope. Before powering our laser range finder we will have to make sure that no one in our testing area is making use of a powerful telescope or microscope.

Protective eyewear is only required for Class 3B and Class 4 lasers so our Class 1 laser does not fall into this category[11]. It is also advised to limit the use of lasers in the oxygen-rich environments however this does not apply to the Capstone laboratory[11].

Regulations and Society

For communicating and transferring data between the robot and the Android device we will be using the Bluetooth wireless standard. With all wireless standards there exists a potential hacking danger. Fortunately Bluetooth supports authentication features that we could make use of for our project. Although it is worth noting that our robot will not be dealing with any sort of sensitive information and will not be capable of using its moving parts to cause significant damage to anything or anyone besides itself(as it has a max speed of 0.35 m/s and a mass of under a kilogram). Therefore, even if someone were to maliciously hack our Bluetooth authentication we do not believe our project would provide any real threat to society.

For range finding we will be using a laser range finder. Laser products often require a certain amount of precautions to be taken in order to be used safely. As described in the Safety section our laser range finder is rated as a Class 1 laser[7]. This means that the laser we will be using will not provide any sort of danger to human eyes unless it is passed through a sufficiently powerful telescope or microscope. Before powering on the range finder we will have to be sure neither of the above objects are present in the area.

Environmental Impact

All parts contained in the vehicle are lead-free/RoHS compliant, with the exception of the MC3479P stepper motor driver.

Sustainability

Electricity Generation in Alberta: 55% coal, 35% natural gas, 6% hydro/wind, 3% biomass [19]
Coal mined Alberta is 78% subbituminous, 22% bituminous [20]

Bituminous coal produces 2.07 lb of CO₂ per kWh (0.939 kg) [18]

Subbituminous coal produces 2.16 lb of CO₂ per kWh (0.980 kg)

Natural gas produces 1.22 lb of CO₂ per kWh (0.553 kg)

Hydro/wind produce negligible CO₂ during operation

Biomass produces roughly 50% more CO₂ per kWh than coal, so $1.5 * 0.939\text{kg} = 1.409\text{kg}$ [21]

Therefore, averaging those values proportionally, electricity in Alberta produces 0.770 kg CO₂ / kWh

The vehicle will not be run continuously – rather, it will be run in active mode for short bursts of time, on the order of ten minutes to an hour at a time, depending on the size of the room to be mapped, and then be completely turned off.

When operational, the vehicle draws the following currents:

- 360 mA at 6V from the motors (2x130 mA drive + 100 mA stepper)
- 2 mA at 3.3V from the Bluetooth chip
- 100 mA at 5V from the LIDAR sensor
- 5 mA at 5V to power the DE0 Nano board

The vehicle consumes electricity at a maximum rate of 3.3W. Given that our batteries will hold 25Wh of power, this gives us an operating time of about seven and a half hours off of each battery pack, easily enough for our purposes.

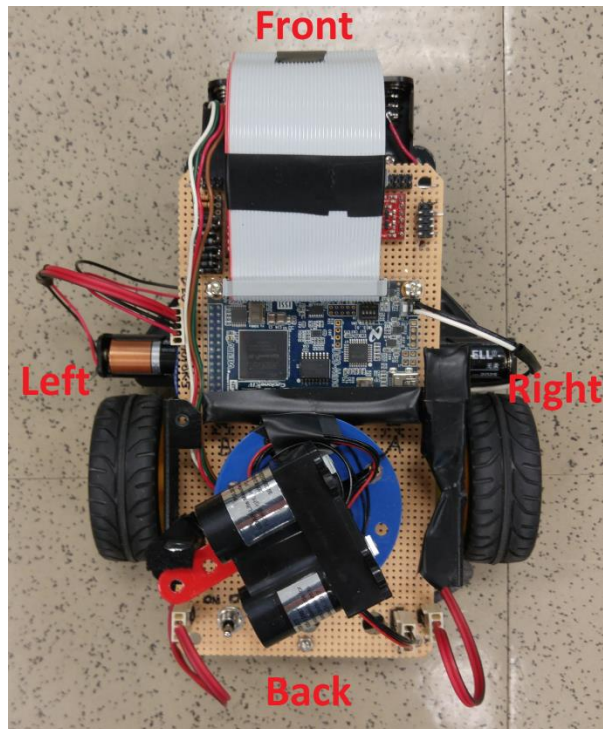
3.3W used over a period of one hour gives 3.3 Wh, which will produce 2.54 g CO₂ over a typical period of operation

References

- [1] "DE0-Nano User Manual"
http://www.terasic.com.tw/cgi-bin/page/archive_download.pl?Language=English&No=593&FID=75023fa36c9bf8639384f942e65a46f3
- [2] "Cyclone IV FPGA Device Family Overview"
https://www.altera.com/content/dam/altera-www/global/en_US/pdfs/literature/hb/cyclone-iv/cyiv-51001.pdf
- [3] "Nios II Processor: The World's Most Versatile Embedded Processor"
<https://www.altera.com/products/processors/overview.html>
- [4] "RB-Dfr-16 DFRobot 2WD Mobile Platform for Arduino"
<http://www.robotshop.com/media/files/pdf/datsheet-rob0005.pdf>
- [5] "DG02S-A130GEARMOTOR"
<http://cdn.sparkfun.com/datasheets/Robotics/DG02S.pdf>
- [6] "TB6612FNG Driver IC for Dual DC motor"
<https://www.sparkfun.com/datasheets/Robotics/TB6612FNG.pdf>
- [7] "LIDAR-Lite v1"
<https://cdn.sparkfun.com/datasheets/Sensors/Proximity/LIDAR-Lite-Laser-Datasheet.pdf>
- [8] "Unipolar Stepper Motor Specifications"
<http://www.robotshop.com/media/files/pdf/rb-soy-02-unipolar-stepper-motor-specs.pdf>
- [9] "Stepper Motor Driver Documentation Package"
<http://www.robotshop.com/content/ZIP/documentation-shd037.zip>
- [10] "RN-42/RN-42-N Data Sheet"
<http://cdn.sparkfun.com/datasheets/Wireless/Bluetooth/Bluetooth-RN-42-DS.pdf>
- [11] "Canadian Centre for Occupational Health and Safety: Lasers"
https://www.ccohs.ca/oshanswers/phys_agents/lasers.html
- [12] "Find the energy contained in standard battery sizes"
<http://www.allaboutbatteries.com/Energy-tables.html>
- [13] "Embedded Peripherals IP User Guide"
https://www.altera.com/content/dam/altera-www/global/en_US/pdfs/literature/ug/ug_embedded_ip.pdf
- [14] "Highly Accurate Maximum Likelihood Laser Mapping by Jointly Optimizing Laser Points and Robot Poses," Ruhnke, Kummerle, Grisetti, & Burgard
<http://ais.informatik.uni-freiburg.de/publications/papers/ruhnke11icra.pdf>
- [15] "SSA2D source code via svn.openslam.org"
<https://svn.openslam.org/data/svn/ssa2d/>
- [16] SDL
<https://www.libsdl.org/license.php>
- [17] Open Cores I2C
http://www.alterawiki.com/wiki/I2C_%28OpenCores%29
- [18] U.S. Energy Administration FAQ
<https://www.eia.gov/tools/faqs/faq.cfm?id=74&t=11>
- [19] Electricity Statistics, Alberta Energy
<http://www.energy.alberta.ca/electricity/682.asp>
- [20] "What is Coal?", Alberta Energy
<http://www.energy.alberta.ca/coal/645.asp>
- [21] "Carbon emissions from burning biomass for energy," Partnership For Policy Integrity
<http://www.pfpi.net/carbon-emissions>

Appendices

A: Quick Start Manual



Overview

Assembly

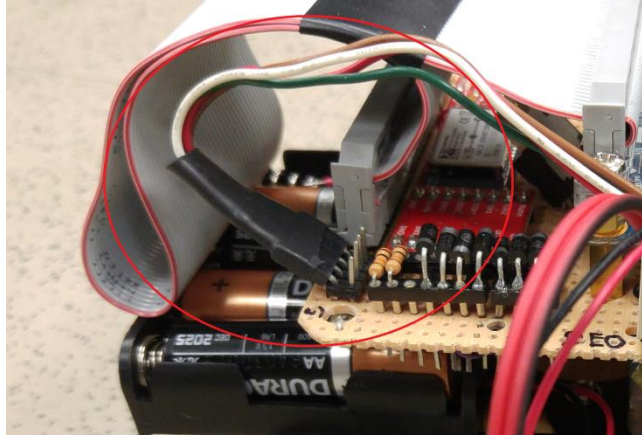
Brackets

There are three brackets that hold the perf board to chassis. One is at the back of the of the perf board in the middle. The other two are at the front past the long skinny battery pack.

Connections

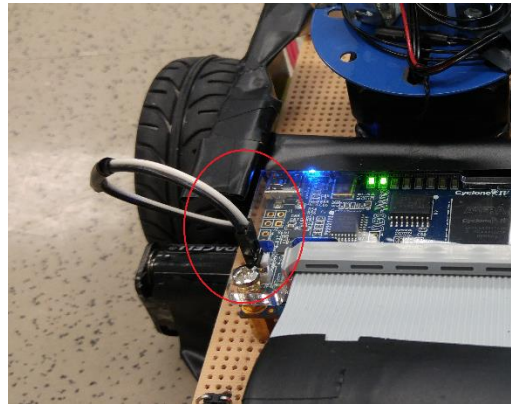
There are 7 connections that need to be made before the robot will work.

The stepper motor must be plugged into the front left header with the white wire on the most outside position.



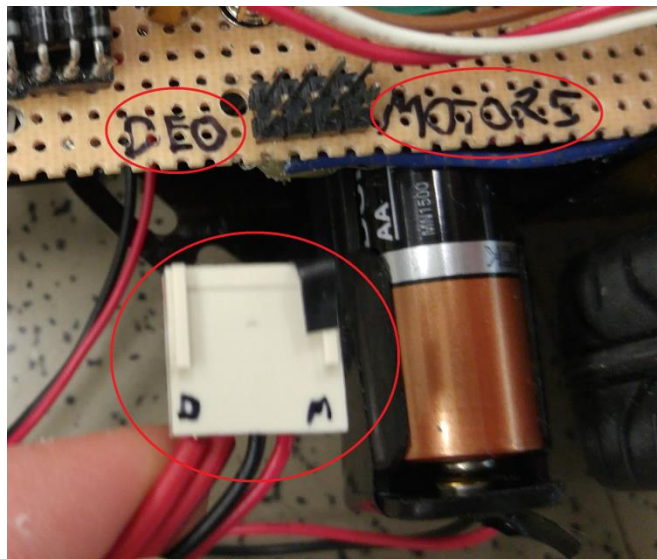
Stepper Motor Connection

The DE0 nano power cable must be connected. White is positive and black is ground.



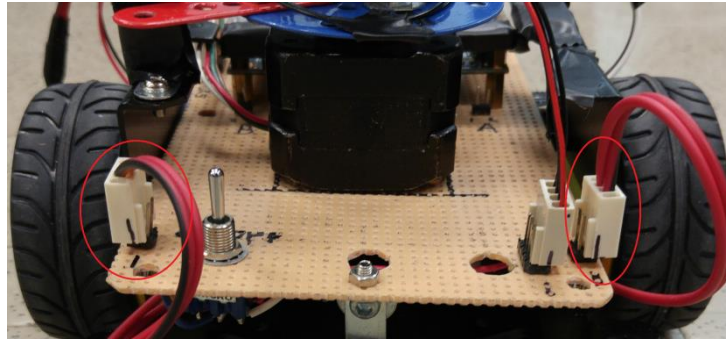
DE0 Nano Power Connection

The battery connector must be attached to the left of the DE0 nano. The perf board is labelled with a DE0 side and Motor side and the connector has a corresponding D and M written on it. Caution: the connector is a 5 slot connector but the header is a 2 x 4 pin header, be sure to attach the connector to the outside 4 pins and ensure that only the populated slots are connected.



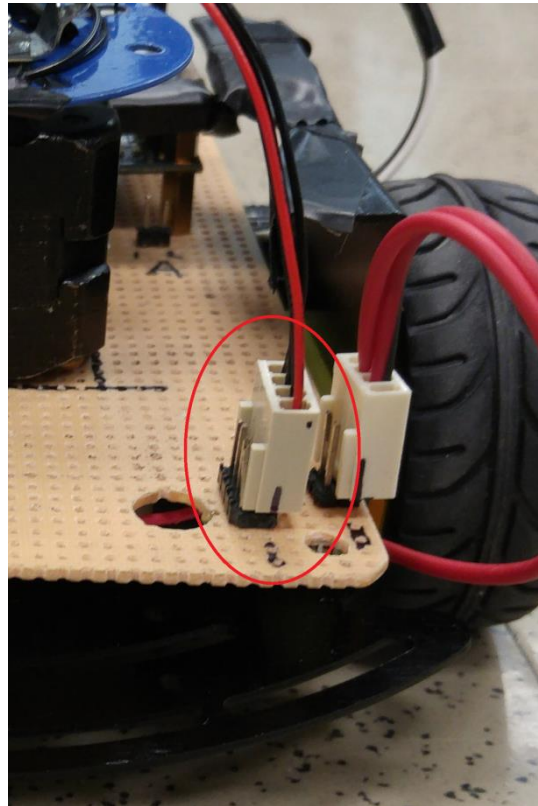
Batteries Connection

The two DC motor connectors must be connected in order for the robot to move around. These connectors are at the back of the robot. The headers are both 2 x 2 pin header. Connect to the most outside pins on both sides (they should be mirrors of each other). There are lines drawn on the perf board and the connectors, these lines should line up. Note: that the connectors are 4 slot connectors but only 2 of the slots are populated. Be sure to connect the populated slots to the headers.



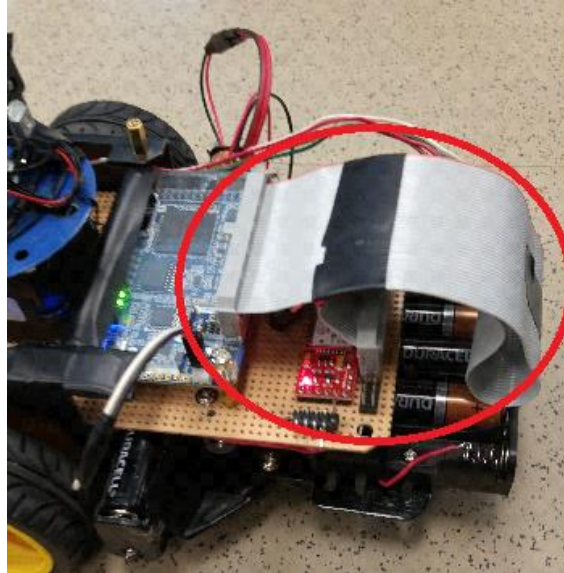
DC Motor Connections

The LIDAR also needs to be connected. The LIDAR header can be found on the inside of the right DC motor header. The LIDAR header is a 2 x 4 header, plug the connector into the pins that are not bent. There are lines on the perf board and on the connector that should line up.



LIDAR Connection

The last required connection is the 40-pin ribbon cable that is plugged into the DE0 nano GPIO 1 header on one end and the perf board on the other. Use the image below to get the correct orientation.



GPIO Connection

Batteries

There are two battery packs. There is a battery pack at the front of the robot that contains 4 batteries, this pack is used for powering the two DC motors and the stepper motor. The batteries in this pack can be replaced after removing the tape that holds the front of the ribbon cable down. There is also a battery pack underneath the perf board right in front of the wheels, it is a skinny and long pack that contains 3 batteries. This pack is used for powering the DE0 nano. These batteries can be replaced if you disconnect the battery connector and slide the battery pack out the left side of the robot (may have to lift the perf board slightly for the battery pack to slide).

Compiled files

Flashing the DE0

Installing the Android application

Demoing the project

Powering the Robot

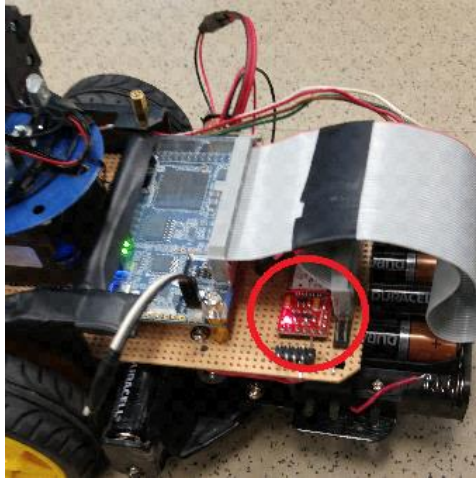
To turn on the robot a three position mechanical switch is used. The switch can be found at the back of the robot beside the left DC motor header. The robot will be ON if the switch is pointing to the left. The robot will be OFF if the switch is in the neutral position or if it is pointing to the right.



ON – OFF – OFF

Note that DC motors will move for a second when the robot is powered up. Also note that the stepper motor initializes by rotating the LIDAR mount into the guide. After initializing ensure the red arm of the LIDAR mount is against the guide.

The robot should remain stationary until an android device connects to it over Bluetooth.

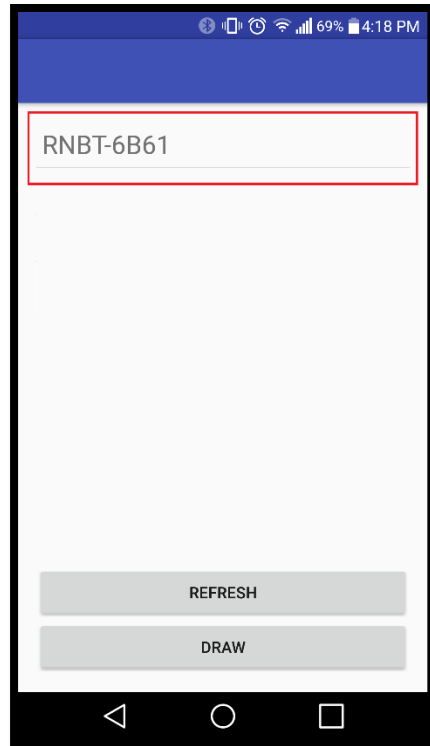


This red LED will flash while the robot waits for a BT connection

Connecting the Android application

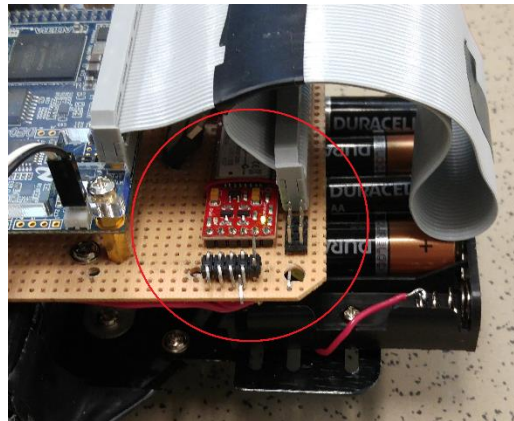
To connect the Android application first ensure that the Bluetooth of the Android device is on. Then start the application.

At this screen click on the device named RNBT-6B61. Note that you may have to try connecting to the device a couple of times before it works.



Application after start-up

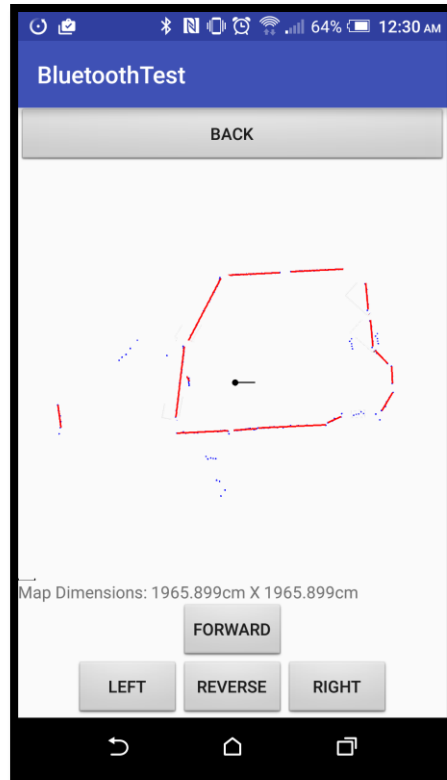
As soon as the Android device connects the Bluetooth model LED will turn green and the robot will perform its initial scan.



Robot connected over BT

Using the Mapping Robot

Once the initial scan is complete the robot will wait for movement commands to be sent from the Android application. Use the "Forward", "Left", "Right" and "Reverse" buttons on the Android application to control the robot. Note that after every single command the robot will perform another scan.



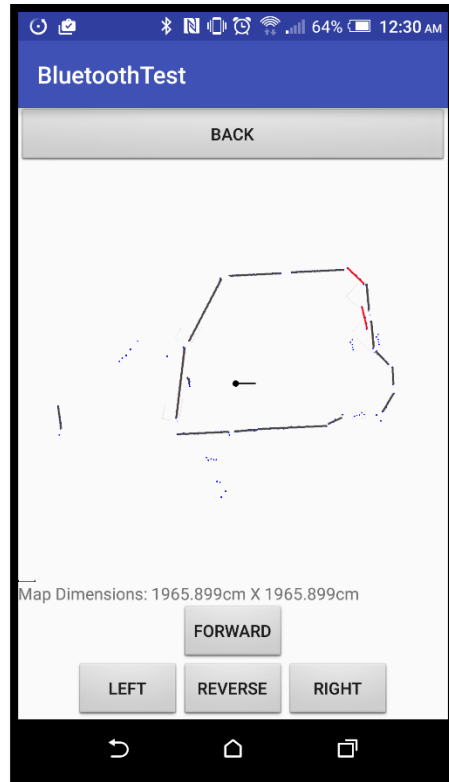
Application after a connection is made

For the best map results:

- Start by doing a couple of rotations (before going forward or backwards)
- Try to avoid rotations that are >180 degrees
- If forwards and backwards movements are not being performed by the robot in straight lines try to send those commands in short bursts between rotations instead of as long movements

Reading the map

The blue dots that are displayed after each scan represent the most recent data sent from the robot. The red lines represent new segments have just been added to the map using the blue dots. The gray lines represent map segments that were created from a previous scans.



Example application screenshot

B: Future Work

If time allows, the following features may also be added:

- 1) Semi-autonomous mode. In this mode, rather than directly controlling the vehicle, the user of the mobile device can provide higher-level commands for the device, such as entering waypoints throughout the room to travel to or entering a series of moves.
- 2) Autonomous mode. In this mode, the device will function completely autonomously, determining from its current surroundings where to travel next and performing a full sweep of the room without any user input (besides enabling this mode). In this mode the mobile application acts as a read-only display of the room model. Note that the device must be able to do all calculations to build a map of the room using only the DE0 Nano board in this mode, that is, without any processing being offloaded to the mobile app.

In the obstacle avoidance mode, the robot drives aimlessly around the world simply avoiding contact with obstacles. In the fully autonomous mode, the robot uses its existing data to plan and drive along a route that quickly and efficiently expands or completes its map of the space.

C: Hardware Documentation

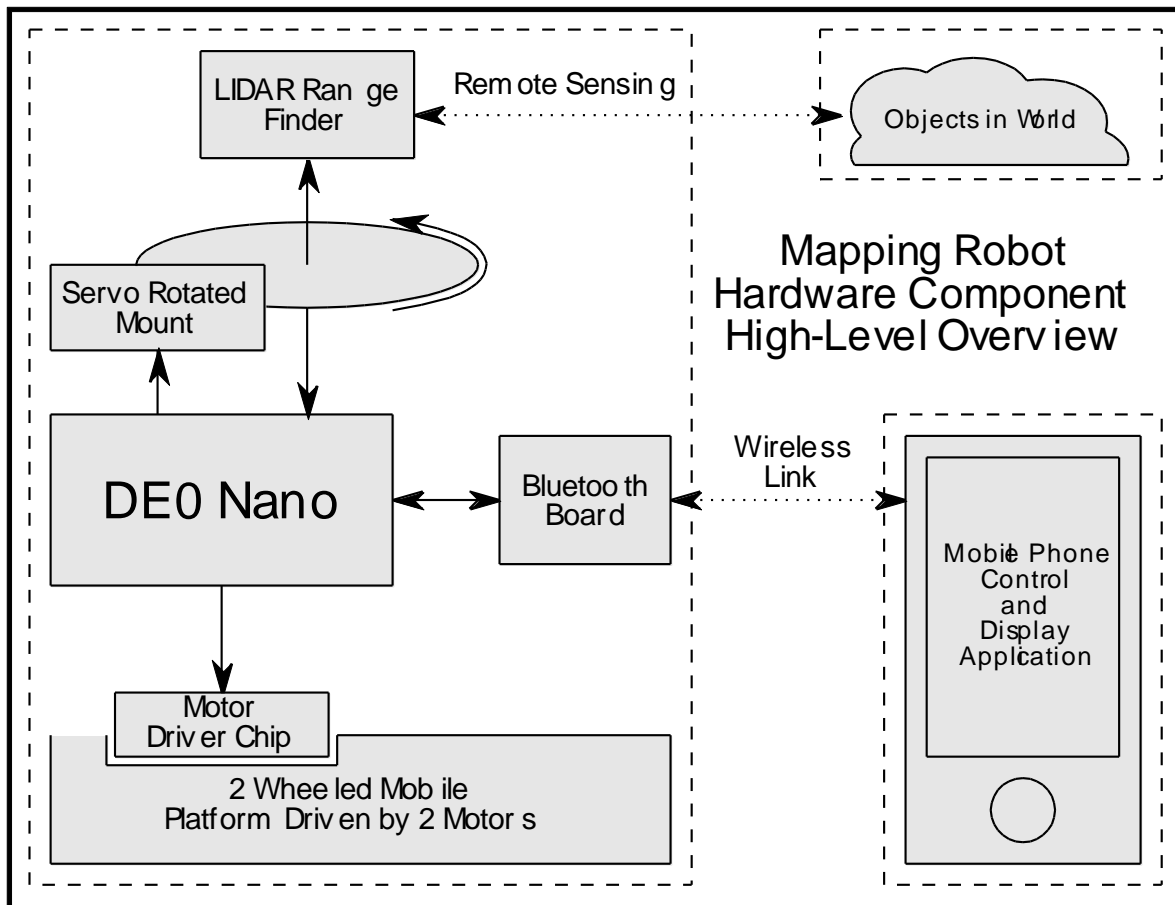


Figure: Hardware Component Overview

D: Source Code

Code for the mapping algorithm library including a testbed can be found here:

<https://github.com/ECE492winter2016g3/algorithm-simulation>

Code for the mobile app including the mapping algorithm can be found here:

<https://github.com/ECE492winter2016g3/android-application>

Code for the Quartus project including Qsys components and DE0 software can be found here: <https://github.com/ECE492winter2016g3/de0-project>