A1: Mechatronic Design and Evaluation - Triangle & Mini-Triangle

ECEN430: Charlene Leong

Introduction

A series of medium-sized robots of varying designs developed for the purposes of the National Autonomous Robotics Competition (NI-ARC) in previous years have been left in a state of disarray. The primary task of the robot is to be able to map an indoor office environment effectively using simultaneous localisation and mapping (SLAM) techniques. This is intended to be performed with a mixture of proprioceptive (internal) and exterioceptive (external) sensors powered by an Intel NUC, a powerful four-by-four inch mini PC, in conjunction with the now standard suite of robotics middleware, Robot Operating System (ROS). Map information is to be gathered using a 2D LIDAR range scanner and an Intel RealSense R200 RGB-D camera. Hence, the aim of this assignment is to assess the viability of integrating with the intended localisation, mapping and navigation stack by evaluating the compatibility of each robot's hardware in performing different SLAM techniques.

The resulting list of **hard**/*soft* requirements include -

- **Mobility** - Capable of moving smoothly on an indoor environment of even terrain
- **Mountability** - Capable of supporting a NUC, RealSense Camera and LIDAR sensor
- **Exterioceptive sensors** - Capable of processing external sensor feedback
- **Proprioceptive sensors** - Capable of processing internal sensor feedback
- *ROS compatibility* - ROS packages available for Arduino, Motor Drivers, IMU, Mouse sensor
- *Ease of assembly* - Number of parts still required
- *Low Cost* - Time and effort required to get robot going which corresponds to monetary cost
- *Modularity* - Capable of swapping out hardware/software components

The SMART objective is to build a map of an indoor environment of even terrain using 3D SLAM techniques with an Intel NUC, Hokuyo URG-04LX-UG01 LIDAR and an Intel RealSense RGB-D camera mounted on a robotic platform.
The core metrics of success are -

LIDAR

- Capable of obtaining distance measurements ranging from 60-4000 mm with a max error of +/-2.5%
- Capable of scanning an angle of 240 degrees with an angular resolution of at least 0.36 degrees
- Min resolution of scan to be able to detect obstacles of min 30 mm at most 2000 mm away

RealSense

- Capable of obtaining distance measurements from 500 mm to 350 mm
- Capable of building 3D map with min error of +/-4 cm per meter moved
- Capable of maintaining odometry lock when 1m in max 5 seconds

Actuators

- Capable of travelling forwards and backwards, left and right
- Capable of remote communication via SSH
- Can drive 1m in max 5 seconds
- Can move in a straight line with max odometry error of +/-2cm per meter moved
- Can turn left and right with max odometry error of +/-2° per 45° moved

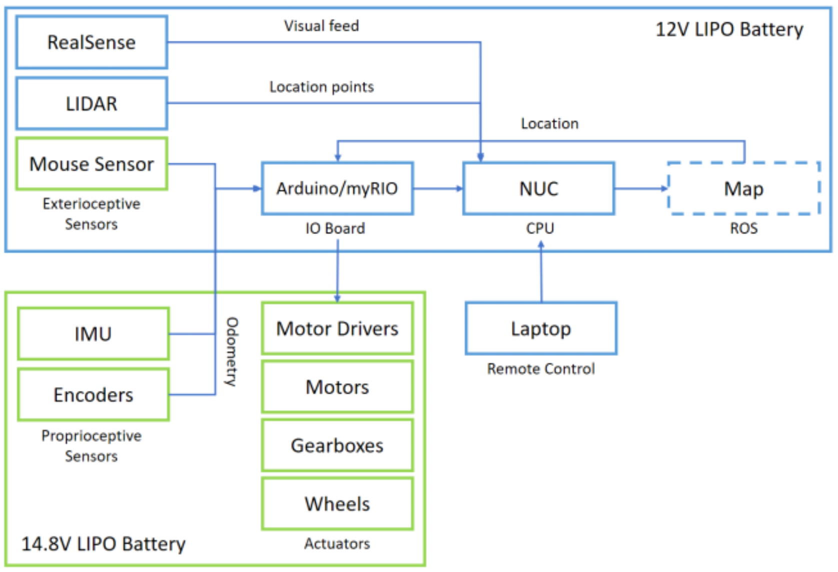
These metrics are to aid in future assessment but will not be tested thoroughly of current due to a lack of batteries to power the NUC and motors remotely on the robots.

This report aims to provide an broad assessment of all six available robots as well as provide an in-depth commentary for Mini-Triangle and Triangle -

1. **Mini-Triangle** - a medium-sized three-wheeled omnidirectional robot designed to interface with the myRIO
2. **Triangle** - a larger three-wheeled omnidirectional robot designed to interface with the myRIO
3. Michelangelo - a small four-wheeled mecanum robot with a steel frame designed to interface with the Arduino Uno
4. Bean MK1 - a small frame with no wheels, only four stepper motors designed to interface with the Arduino Uno
5. PWNBot - a very large four-wheeled mecanum robot with positional motor controllers, battery monitor or intelligent power supply designed to interface with the Arduino Mega
6. Bolt - a medium sized four-wheeled non-holonomic racing robot designed to travel very fast in a straight line

System Design

The following high level system block diagram describing the overall design where components in blue are components that are definite and components in green are those that vary with each robot platform.



Fig[1] High Level System Block Diagram for robot platform performing indoor SLAM

Robot Assessment Requirements Summary

The following table summarises **hard** and *soft* requirements of each robot with needed or missing components underlined.

	Mobility	**Mountability**	**Exterioceptive**	**Proprioceptive**
Robot 1 (Mini Triangle)	Holonomic - omniwheels	LIDAR mount existing, broken RIO mount, space for batteries existing, needs raised platform to mount NUC & RealSense	Mouse sensor	Encoders IMU
Robot 2 (Triangle)	Holonomic - omniwheels	2 LIDAR mounts existing, has RIO mounted below base frame, needs raised mounts for NUC, RealSense & batteries	Mouse sensor	Encoders IMU
Robot 3 (Michelangelo)	Holonomic - mecanum wheels	Has flatspace for NUC and batteries (on top?) but needs supports, needs mounts for LIDAR and RealSense	None	Needs encoders and IMU
Robot 4 (Bean MK1)	Needs wheels	Needs raised platform to mount NUC & batteries, needs raised mounts for LIDAR and RealSense	Mouse sensor	Needs encoders and IMU
Robot 5 (PWN Bot)	Holonomic - mecanum wheels	Has space for NUC and batteries, needs mounts for LIDAR & RealSense	Mouse sensor	Encoders Needs IMU
Robot 6 (Bolt)	Non-holonomic - racing tyres	Needs raised platform for NUC and batteries, needs mounts for LIDAR and RealSense	None	Needs encoders and IMU

	ROS compatibility	*Ease of assembly*	*Low Cost - effort + time*	*Modularity*
Robot 1 (Mini Triangle)	Available for MPU-6050 IMU None for ESCON Maxon motor controllers	Base is assembled, needs mounts for NUC, RealSense and batteries	Need to construct mounts and get Arduino	Rio to Arduino+NUC Custom power board
Robot 2 (Triangle)	Available for MPU-6050 IMU None for ESCON Maxon motor controllers	Base is assembled, needs mounts for NUC, RealSense and batteries	Has a myRIO but can use Arduino, need to construct mounts	Rio to Arduino+NUC
Robot 3 (Michelangelo)	Available for L6470 Stepper Driver and Arduino Uno (rosserial)	Base is assembled, needs mounts for NUC, RealSense and batteries and LIDAR	Needs encoders, IMU, platform and mounts	Arduino to NUC Metal Chassis
Robot 4 (Bean MK1)	Available for L6470 Stepper Driver and Arduino Uno (rosserial)	Needs wheels, platform for NUC batteries and mounts for RealSense and LIDAR	Needs wheels, encoders , IMU, platform and mounts	Pretty modular as missing so many parts
Robot 5 (PWN Bot)	Available for M4-ATX battery monitor, EPOS motor controller and Arduino Mega (rosserial)	Base is assembled, needs supporys for NUC and batteries , needs mounts for RealSense and LIDAR	Needs NUC and battery supports, IMU and mounts + figuring out custom board	Custom board
Robot 6 (Bolt)	None for Sky Toro speed controller	Base is assembled, needs platform for NUC batteries and mounts for RealSense and LIDAR	Needs platform for NUC and batterees, IMU and mounts	Speed controllers especially for racing tyres

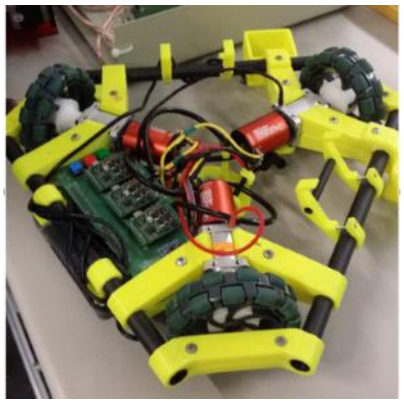
Broad Component Assessment

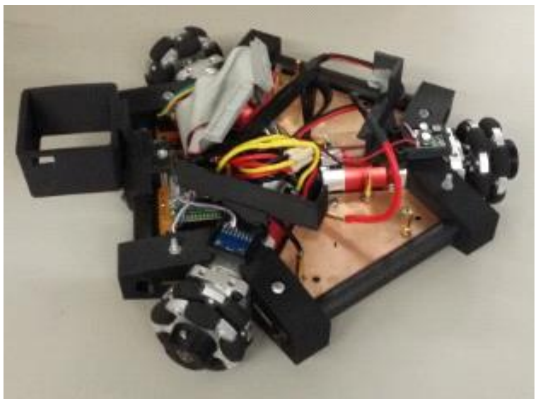
	IMU	Mouse Sensor	Motor Drivers	Motors	Gearbox	Wheels	IO Board	Additional Components
Robot 1 (Mini Triangle)	1x MPU-6050 IMU	1x Mouse sensor	3x ESCON Maxon motor drivers	3x TP 21.5 motors	3x P60k 26:1 gearboxes	3x 70mm Omni wheels		1x LIDAR mount
Robot 2 (Triangle)	1x MPU-6050 IMU	1x Mouse sensor	3x ESCON Maxon motor drivers	3x Tekin 1200kV motors	3x P60k 26:1 gearboxes	3x 60mm Omni wheels	1x myRIO	2x LIDAR mount
Robot 3 (Michelangelo)			4x Sparkfun motor drivers	4x Stepper motors		4x Mecanum wheels	1x Arduino Uno	
Robot 4 (Bean MK1)				4x Stepper motors				
Robot 5 (PWN Bot)		1x Mouse sensor	4x EC Maxon motor drivers	4x EC Maxon motors		4x Mecanum wheels	1x Arduino mega	1x battery monitor
Robot 6 (Bolt)				2x Toro motors		4x Contact RC racing tyres		2x TS 150 speed controllers

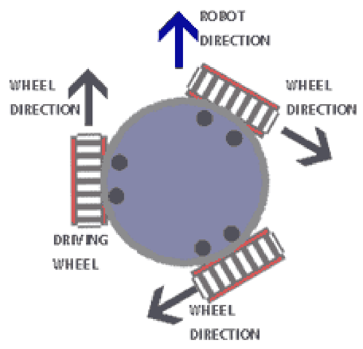
From the broad component assessment above, it is observed that Triangle is the most complete robots out of the six options and therefore will take the least effort, time and cost to get up and running in order to perform indoor SLAM. Mini-Triangle is still missing an IO Board and PWN Bot could do with an IMU but they are also not too difficult to get up and running.

Chosen Viability Assessment - Triangle and Mini-Triangle

Triangle and Mini-Triangle are both three-wheeled omni-wheel holonomic robots with similar components and architecture. Major points of difference include the second LIDAR mount on Triangle, the myRIO on Triangle, and the custom power board and lack of wheel frames on Mini-Triangle. The motors, motor drivers, and gearboxes are identical. Both will require additional 3D printed mounts to be printed to mount the NUC. The way the RealSense will be mounted will also differ where on Triangle, it will be mounted on the wheel frame whereas on Triangle whereas it would be mounted on the frame between the wheels on Mini-Triangle.


Fig[2] Triangle


Fig[3] Mini- Triangle



Fig[4] Omnidirectional Drive

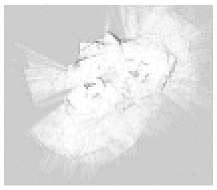
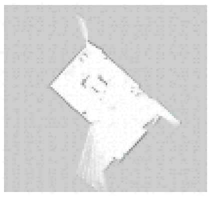
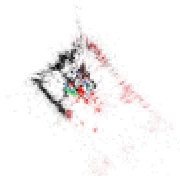
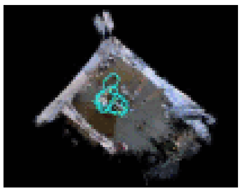
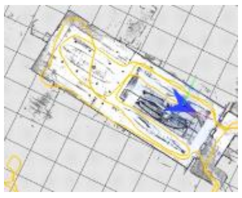
Triangle and Mini-Triangle are the most complete robots and also allow for high mobility due to their holonomic design meaning they can instantaneously change direction of motion without having to perform intermediate rotation steps. This complete 2 degrees of freedom of movement is due to the structure of the omniwheel. An omniwheel consists of a big core wheel, where along the peripheral there are many additional small wheels that have an axis perpendicular to the axis of the core wheel which allows the robot's core wheels to move parallel to its' own axis as well as normal wheel rotation. This also makes calculating the current state vector when implementing SLAM a little easier as we do not need to account for potential turning errors when tracking odometry.

However, despite these benefits, there are many disadvantages to a three-wheeled omnidirectional robot. The uncommon design of three-wheeled omnidirectional control means there are fewer libraries available for this type of kinematic control which suggests a high probability of implementing the actuator controller from scratch. A three-wheeled robot will never achieve 100% efficiency compared to a four wheel robot as no more than one wheel will ever be aligned with the direction of motion. It is also computationally cheaper to operate four wheels as there will always be a pair of wheels directly opposite of each other, meaning a single calculation is needed for a pair of wheels where one is negative, the other positive whereas a three-wheeled design requires three separate vector calculations for movement. Additionally, for an omniwheel robot to translate at a particular angle, each wheel must rotate at a particular rotational velocity and direction. Since the robot is moving at angles, it will need to do trigonometric calculations to determine these wheel speeds. However, processing these trig calculations constantly is quite computationally heavy for the Arduino and the NUC especially if the NUC is also reading data in from the LIDAR, RealSense and performing SLAM at the same time. Instead, it'd be much more efficient to implement a trigonometric lookup table which the Arduino can reference, trading accuracy for speed.

As for position control, the omniwheel works on the basis of wheel slippage, so using things like encoders for position will not work. Therefore Triangle and Mini-Triangle will need to detect its surroundings to track its motion accurately using other exterioceptive sensors such as the IMU and mouse sensor. Additionally, there are several things we need to consider when optimising the actuator controller such as angle control, motor speed, rotational control, maximum motor speed and its global angle. For the omniwheel robot to translate at a certain angle each motor needs to run at a certain speed with relation to the others where the actual speed doesn't but rather the ratio of speeds between the wheels. When rotating at some particular speed, it also must add or subtract equally form the motor speed of each motor.

Different Types of SLAM

Different SLAM algorithms are suited for different types of sensor input and require different types of odometry which should be taking into consideration when determining the viability of each robot. For example, Hector SLAM designed for LIDAR doesn't require odometry but performs relatively poorly in environments with complex features. R-TAB Mapping, a visual SLAM technique, uses visual odometry and is capable of providing a detailed 3D map of its surroundings but is computationally expensive and prone to error in dynamic environments. This is due its sensitivity to ambient light and the fact that there is no proprioceptic odometry feedback in the algorithm. Therefore, it is important to consider the compatibility of different types of SLAM with each robot platform in order to assess the viability of implementing indoor SLAM with each robot platform. A few popular SLAM algorithms have been identified suitable for future testing as informed by an experimental evaluation of ROS compatible SLAM algorithms for RGB-D sensors [1]. The following table shows a brief SLAM output comparison on the public freiburg2-pioneer-slam2 dataset [2].

2D Hector-SLAM	2D Gmapping	3D ORB-SLAM2	3D RTAB-Map	2D/3D Google Cartographer
				

Unfortunately, due to the 12 hour time constraint, these SLAM techniques were not tested using the Intel RealSense but it would be useful to test these options and compare their performance upon implementation.

Final Performance Evaluation and Recommended Actions

After a broad requirements analysis of all the robots on Page 2, a broad component analysis on Page 3 as well as a further in-depth analysis of Triangle and Mini-Triangle, it can be concluded that out of the three most complete robots, Triangle, Mini-Triangle and PWNBot, the difficulties in implementing omniwheel drive effectively rule out Triangle and Mini-Triangle as the quickest to get up and running. This is in the context of evaluating the robot's' compatibility with ROS and the NUC in implementing SLAM.

Taking into consideration that Triangle and Mini-Triangle were originally designed to be interfaced with a myRIO for the purpose for NI-ARC competitions, the existing actuator control and associated libraries have been written in LabVIEW for the myRIO. There is a certain learning curve as LabVIEW is a block programming language (non-standard language). In terms of ease of implementation, there is a myRIO ROS package available which means we should not rule out using the myRIO as an IO board as a possibility but we should take into consideration the extra power requirements the myRIO will need vs using an Arduino. This would mean an additional LIPO on board on top of the 12V LIPO for the NUC and 14.8V LIPO required for the motors according to their datasheets. Using the Arduino as an IO board is considerably easier considering the increased familiarity with C and programming in the Arduino IDE. Additionally, there are unfortunately several disadvantages to omniwheel drive as discussed previous. Omniwheels have poor efficiency because not all wheels rotate in the direction of robot movement and there are high losses from friction too. Position control using encoders is also not possible due to high slip as well as the high computational loss by calculating angles with trigonometry.

PWNBot on the other hand, a large four-wheeled mecanum holonomic robot, has several advantages to its design in being the most viable research platform in implementing indoor SLAM most quickly. This is due predominantly due to the fact that it is much more ROS compatible than Triangle and Mini-Triangle with pre-existing ROS packages for the Arduino Mega, positional EPOS motor controllers from Maxon Motors as well as the M4-ATX battery monitor or intelligent power supply. However, there is a custom interface board which requires some further analysis. A four-wheeled robot is also much more efficient than a three wheeled omnidirectional robot all wheels contribute to the motion of the robot. PWNBot also has enough space in its chassis to fit NUC and batteries comfortably which also reduces the amount of effort of implementation vs having to 3D print a mounted platform for Triangle and Mini-Triangle to house the NUC. The RealSense can also be mounted higher on PWNBot due to its larger chassis which increases its field of view potentially increasing SLAM accuracy.

Taking all of these factors into consideration, it can be concluded that the most viable research platform to get up and running and implementing indoor SLAM with the least time and effort would be PWNBot.

[1] Experimental evaluation of ROS compatible SLAM algorithms for RGB-D sensors - https://ieeexplore.ieee.org/document/8215331/
[2] freiburg2-pioneer-slam2 Dataset Download - https://vision.in.tum.de/data/datasets/rgbd-dataset/download