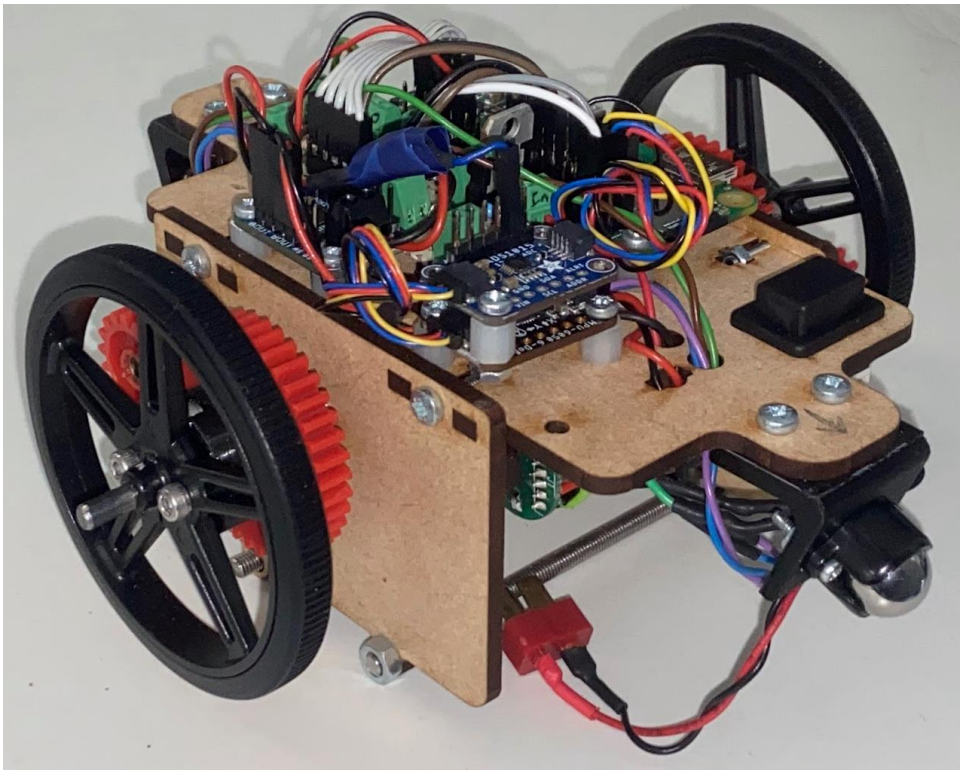
Mechatronics 2: Integration and Project Reflection

Rolling Robot Mechanics and Electronics

Harry Mills



Summary

This report details the design process of a rolling robot mechanism to go inside a plastic sphere. Specifically, this report deals with how this subsystem was integrated with other subsystems in order to achieve automated control with positional feedback to drive the rolling robot to specified locations. An individual reflection on the work undertaken in this project is also included.

Ultimately, the rolling robot subsystem functioned fully successfully and interfaced with the relevant other subsystems correctly. However, the overall integrated system did not work as intended due to other subsystems not individually functioning.

Contents

Summary i
1. Comments on previous submissions 1
1.1. Subsystem Requirements Table and Test log 1
2. Integration Activities 2
2.1. ArUco positioning and detection 2
2.1.1. ArUco height and size 2
2.1.2. Backlighting ArUco 2
2.1.3. Green ArUco 3
2.2. Motor Control Integration 4
2.3. Robot Electro-mechanical integration 4
2.3.1. Chassis-Electronics Integration 4
2.3.2. Robot to Sphere interface 6
2.3.3. Raspberry Pi to Power/Drivetrain electronics integration 6
2.3.4. Further chassis concept iterations 7
3. Personal Reflection 7
Aims 7
How did I approach the aims? 7
How well did I achieve the aims? 8
What would I do differently? 8
References 10
Appendix A 11

1. Comments on previous submissions

A few key components of the previously submitted block diagram have changed. An updated block diagram is shown in Figure 1. The main changes from the original block diagram are:

- **Omission of a steering system**. Research into spherical robot design showed that steering and drive could be achieved with just 2 wheels.
- **Specified on-board sensors**. An IMU and an ADC are included in the final design.

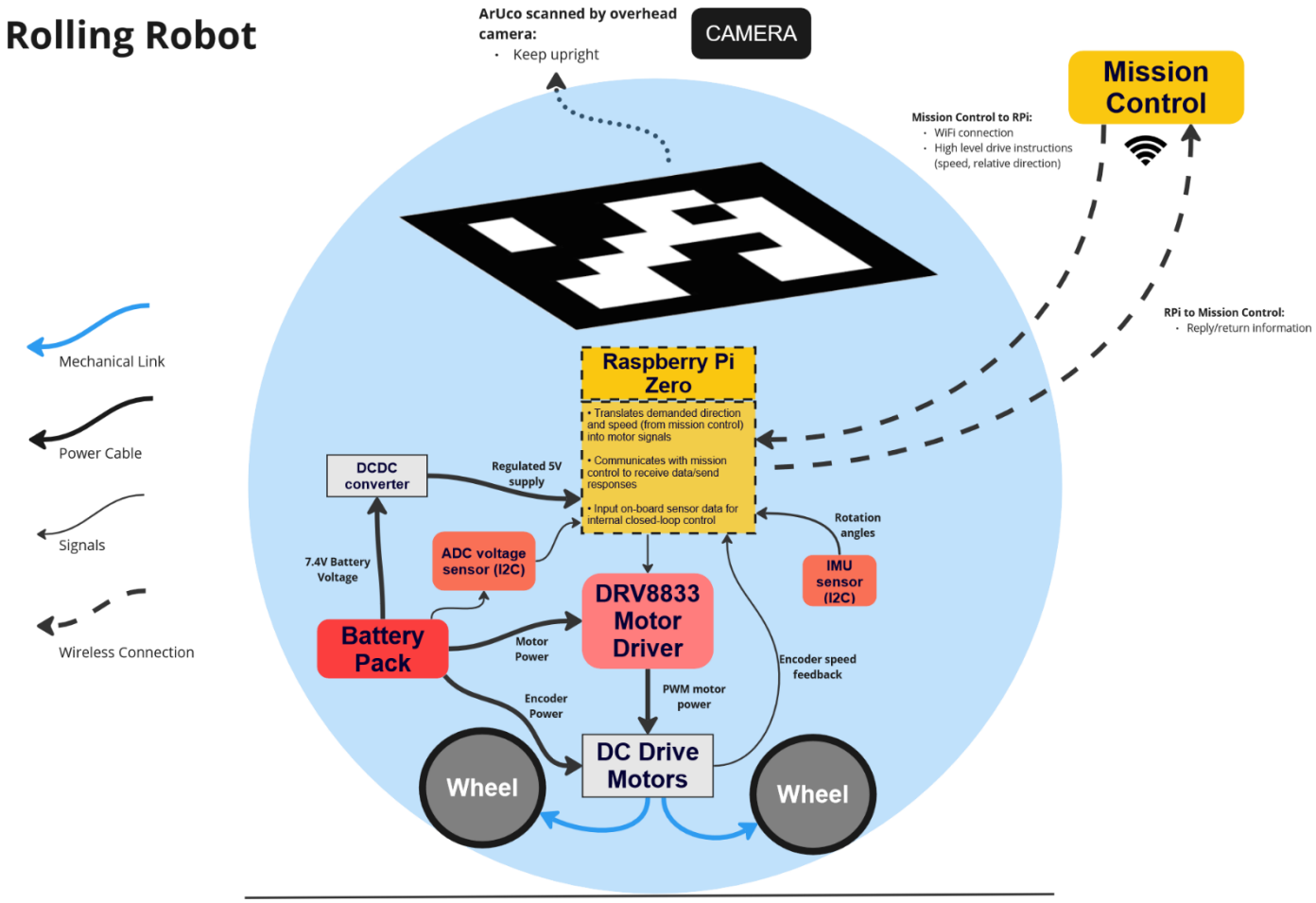


Figure 1 Rolling Robot subsystem block diagram

1.1. Subsystem Requirements Table and Test log

Several requirements were partially/not tested at the point of submission of the logbook. These were tested and updated. See Appendix A for details of the updated requirements and justifications.

2. Integration Activities

After demonstrating that the rolling robot could operate wirelessly under open-loop (SR-01) and closed-loop (SR-02) control using a Simulink model via WiFi hotspot, the next stage was to integrate with the relevant other subsystems to meet the overall system requirements (I-01 - 03, and D-01 – 05).

2.1. ArUco positioning and detection

Placing the ArUco code inside the ball so it could be seen by the camera system was assumed to be a trivial task, however upon testing with the whole system the glare from ceiling lights obscured part of the marker and made it difficult to read. In order to combat this several different options were considered.

2.1.1. ArUco height and size

In order to minimise the difficulty of detecting the ArUco it was placed as closed to the centre of the sphere as possible so it could be as big as possible. This was limited by the components and wiring on the robot, so the ArUco was placed 30mm above the chassis. At this point, the width of the sphere is 19cm, giving a maximum ArUco side length of 13.5cm (Figure 2).

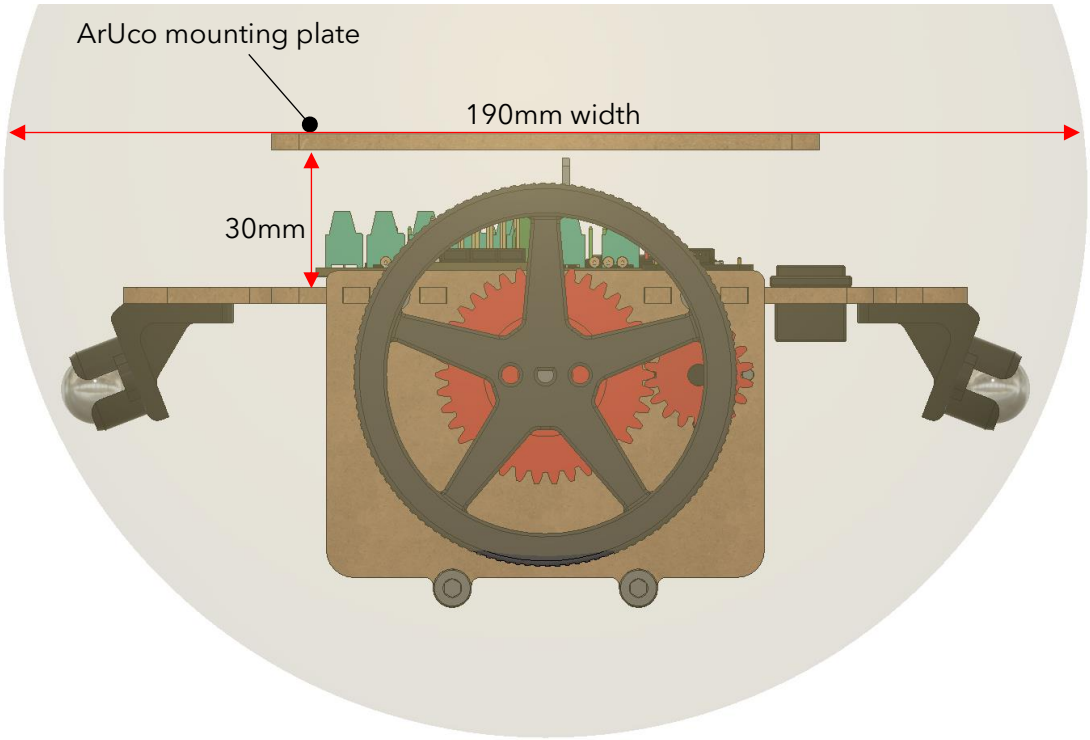


Figure 2 ArUco width calculations

2.1.2. Backlighting ArUco

A strip of LEDs was used to backlight the ArUco, as shown in Figure 3 This was also tested with a sheet of baking paper as a makeshift light diffuser. The LED operated on a 12V supply and required a separate DC/DC boost converter to operate.

Ultimately, this option was deemed not suitable. As can be seen in FIG, the LEDs were not bright enough to combat the glare and the added complexity of the LEDs and power circuitry made it non-feasible.

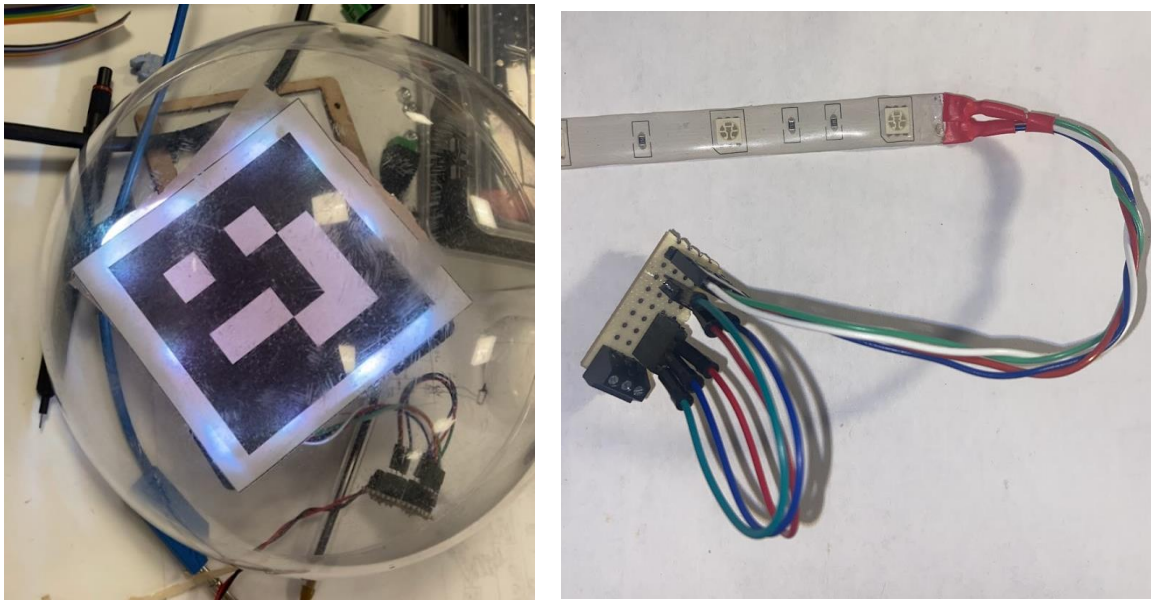


Figure 3 Backlighting ArUco test

With further development this could have been a feasible option. For example, making the black sections more opaque, using brighter and more LEDs, and dimming the room lights would certainly have made ArUco detection easier. However, a second method of improving ArUco detection was being tested simultaneously which turned out to be a more effective and easier method, so development of LED backlighting was stopped.

2.1.3. Green ArUco

This work was done in conjunction with Hena (Computer Vision)[1].

A simpler approach was to colour the white sections of the ArUco green, and then filtering the camera image to green. More detail of this is in Hena's report[1], but with certain light conditions and software image processing, this was deemed enough to effectively detect the ArUco from inside the ball.

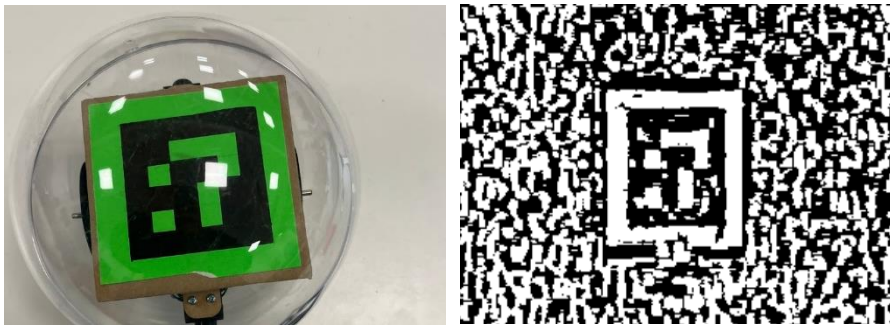


Figure 4 Left: Green ArUco code inside ball. Right: software filtered image of the green ArUco

2.2. Motor Control Integration

Demonstrating open-loop and closed-loop control of the motors was done by verifying SR-01 and 02. To interface with the rest of mission control (Maxime)[2], a demanded speed (as rpm, or fraction of full-speed) can simply be sent to the left and right motor respectively (Figure 5).

Along with the PID motor control model (shown in test logbook), this was the extent of the integration with mission control. The robot speed and direction can be controlled by inputting 2 numbers. Further motion control and communication activities were part of Mission Control activities [2].

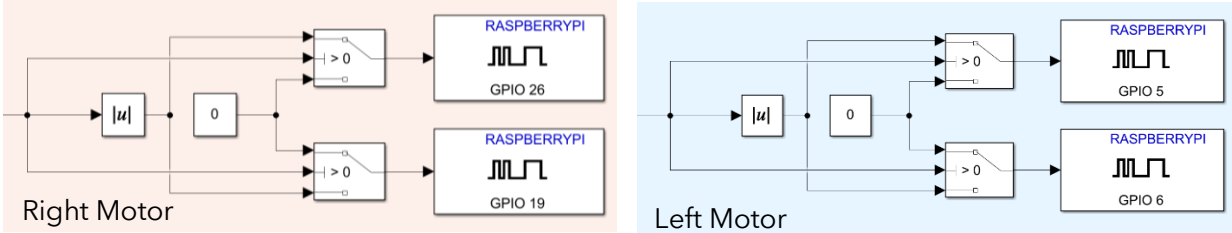


Figure 5 Raspberry Pi GPIO pins to drive the rolling robot motors. Also shown is the simulink code to switch the motors between forwards and backwards motion

2.3. Robot Electro-mechanical integration

2.3.1. Chassis-Electronics Integration

Within the rolling robot the electronics and mechanical subsystems were integrated to create a compact, tidy, and efficient system. This was achieved by laying out the electronic components efficiently and using a small custom interface board. This process involved mapping connections and testing electronics layouts on early chassis iterations. Figure 6 shows iterations of the chassis-electronics layout.

As well as optimising the layout of the electronic components, the final chassis-electronics configuration (Figure 7) also includes:

- Standoff mounting holes for each PCB module to keep everything secure
- Wire routing holes for cable management to the motors and battery
- On/off switch to save power and protect the Raspberry Pi
- Custom crimped ribbon cables for connecting key components with shorter cables to keep everything tidy.

This was done on the principle of improving the design for assembly and maintenance [4], as it made it easy to build, swap out parts and diagnose problems with the electronics.

The chassis is made of interlocking laser cut MDF panels which are quick to design, manufacture and assemble. Design for manufacturing [4] was the key aim here, as 3D printing was in high demand and takes many hours for a single part. By laser cutting as much as possible the time between iterations was greatly reduced. The interlocking panels required slots between each piece. Guidance for creating these slots can be found in [3], but ultimately it required iteration to find the best fit - which was 0.1mm oversized male fittings due to the kerf thickness of the laser cutter [6]. This created a rigid chassis to which electronics and drivetrain components could be mounted to integrate the rolling robot.

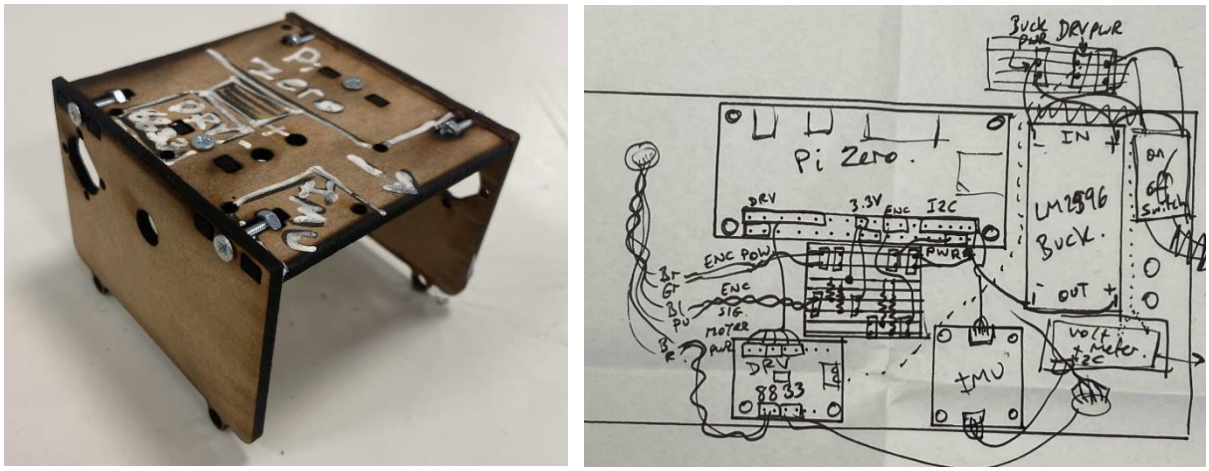


Figure 6 Early chassis iterations with electronics layout (left). Right: rough early plan for electronics layout to map out all connections

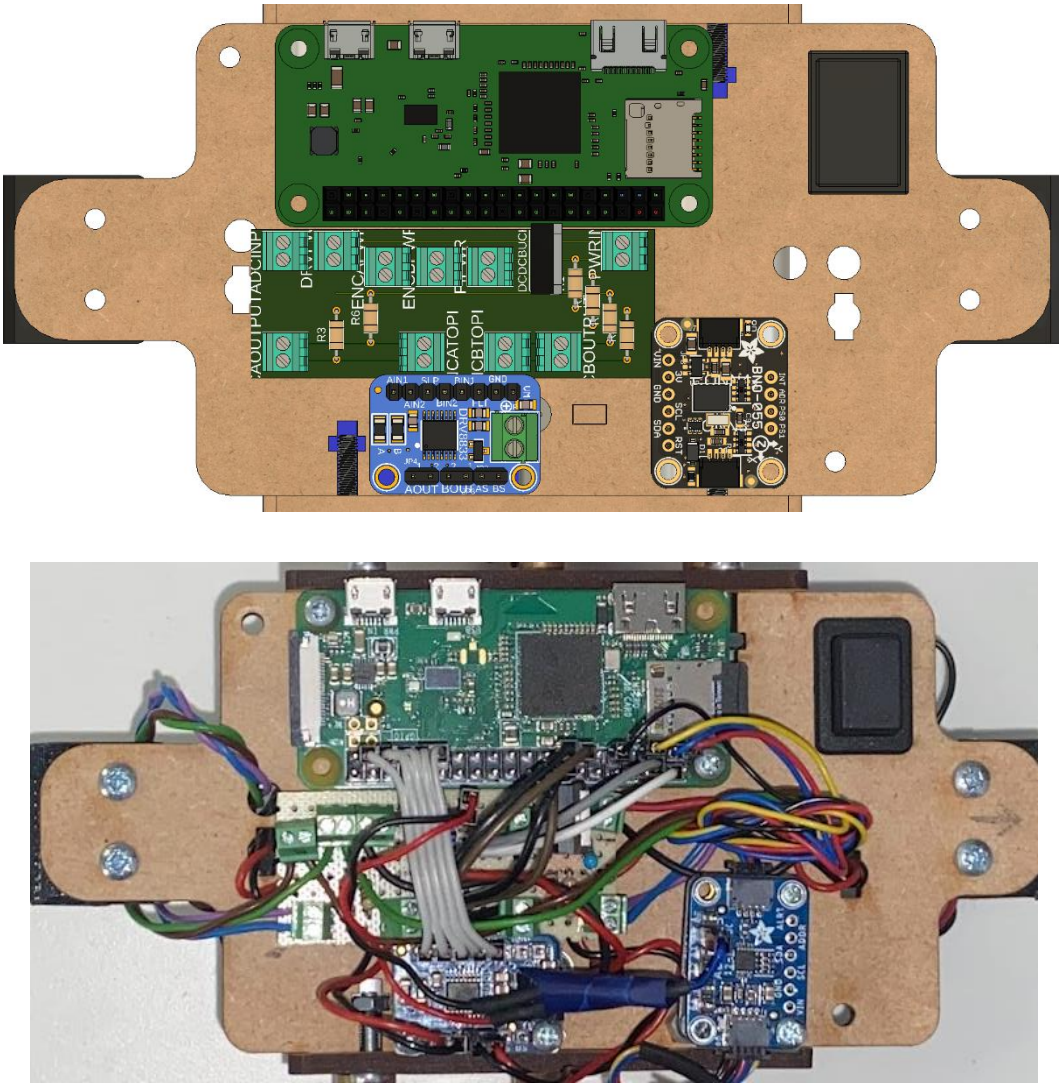


Figure 7 Top: final CAD model of the chassis with mounted electronics modules, wire routing holes and integrated on/off switch. Bottom: physical final prototype with wiring added to show use of wire routing holes and ribbon cables

2.3.2. Robot to Sphere interface

In order to ensure the robot fit within the sphere and the ball casters did not take weight off the wheels a parameterised calculator (Figure 8) was implemented to find the diameter of wheels required at certain motor heights and chassis widths. This is available at: www.desmos.com/calculator/g4a1bu32u4

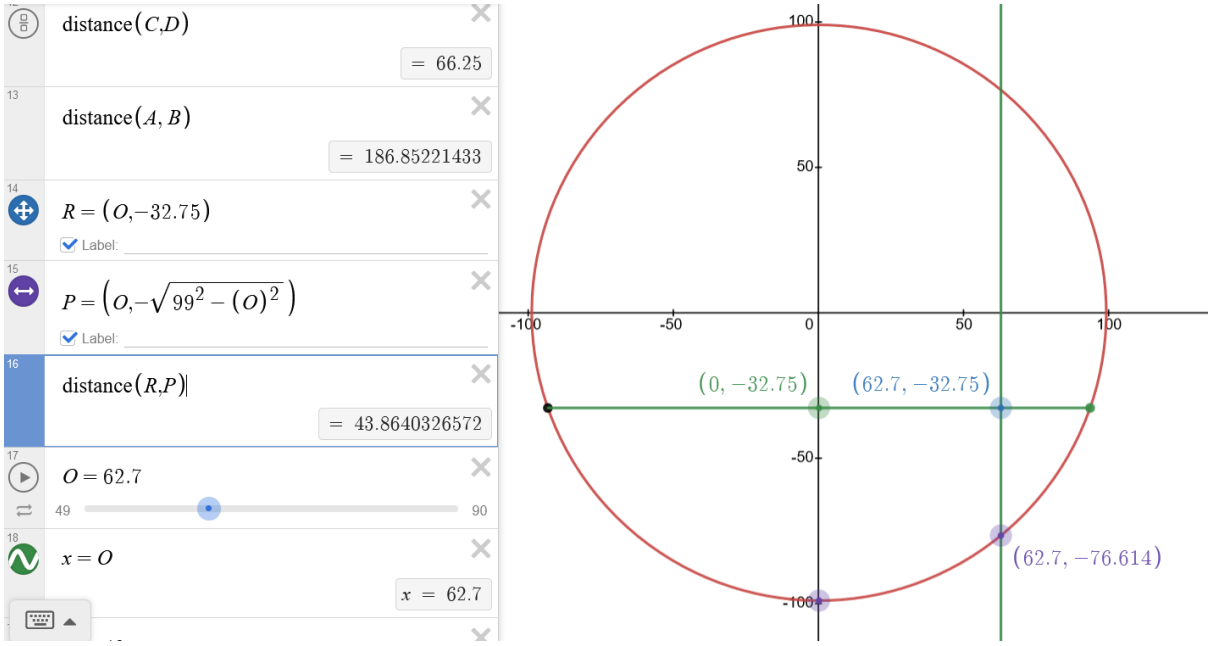


Figure 8 Desmos calculator to find wheel separation and diameter, based on chassis width and motor height (RP is wheel radius, O is chassis width, h (not shown) is motor height)

After designing the chassis in CAD, the required wheel separations and diameters were calculated in order to purchase the correct wheels.

2.3.3. Raspberry Pi to Power/Drivetrain electronics integration

To interface the Raspberry Pi with the motor drivers, encoders, and power it from the battery a small interface prototype board was constructed. This used terminal blocks to ensure solid connections and was designed to be compact to fit on the top deck of the chassis. Furthermore, the schematic and an accurate 3D model were produced in Fusion360 ECAD (Figure 9) to confirm it could be integrated into the design by fitting in between other PCB modules (Figure 7).

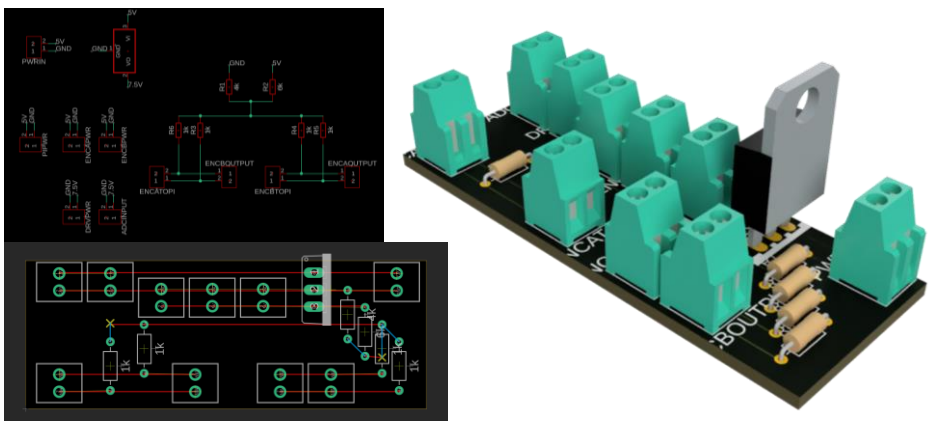


Figure 9 ECAD Schematic, layout, and 3D model of the interface board to confirm connections and fit within the final prototype.

2.3.4. Further chassis concept iterations

Another chassis iteration (Figure 10) was developed as another option in case the first design concept was not successful. This second concept swapped the position of the motors and battery, allowing the motors to sit lower down – in turn using smaller wheels as this was the approach taken in existing examples and literature [5], and was anticipated to increase the stability of the robot. This iteration also improve design for assembly [4] as earlier iterations were quite tricky to build.

Ultimately, this was not necessary as the first concept worked satisfactorily.

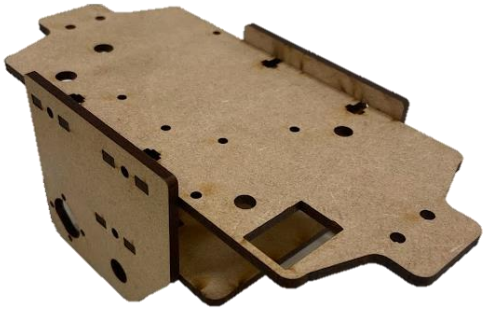


Figure 10 Additional chassis configuration to place motors below battery as a test for increasing stability

3. Personal Reflection

My role in the team was to develop the rolling robot mechanism and electronics. Ideally, I would have liked to work on the software elements of the project as this presented an opportunity to develop my skills in an area, I do not have much experience in. However, other group members were keen to do this as well, so I focused on the robot hardware design – an area where I have more experience.

Aims

Since I have experience in mechanical design and electronic prototyping, I decided my aims should be (in line with the demonstration mark scheme) to make the robot as **robust, simple, materially efficient, and reliable** as possible. Every design decision and task were done to increase these traits rather than doing something novel.

How did I approach the aims?

A timeline of my activities through the 'build' and 'integration' weeks is shown in Figure 11.

For the mechanical elements my main approach was to have as much as possible laser cut. This made designing parts for the chassis slightly more difficult than 3D printing, but it made the rate of iterations much faster as I could have a new chassis in under an hour. The modular laser cut design also allowed changes to individual components which could be swapped in when changes were needed.

For the electronics, a small interface board with terminal blocks (instead of pin headers) was used to reduce the likelihood of loose connections. This was placed centrally on the chassis to easily interface with the motors, motor driver, and raspberry pi and reduce messy cables trailing around the robot.

How well did I achieve the aims?

I believe I achieved my aims very successfully. The robot was compact, simple, reliable, and materially efficient. Very few problems occurred with the robot electro-mechanical system during testing and all features functioned as intended. When iterations were required, these were completed rapidly due to the modular laser cut design.

What would I do differently?

I was very happy with how the design and development of my subsystem went. I would not do much differently in that respect as I believe I worked efficiently and produced a fully functioning refined end-product.

In retrospect, the robot design may not have needed to be as refined, especially since the system ultimately did not function as intended during the demonstration. A more pragmatic approach might have been to retain the simpler chassis and electronics configuration developed earlier in the integration phase, which could have allowed more time to support other areas of the project that were behind schedule.

Another potential improvement would have been to allocate work packages differently from the outset—for example, consolidating mechanical and electronic responsibilities under fewer team members. This could have freed others to focus more fully on complex tasks such as motion control, which turned out to require more development effort than initially anticipated.

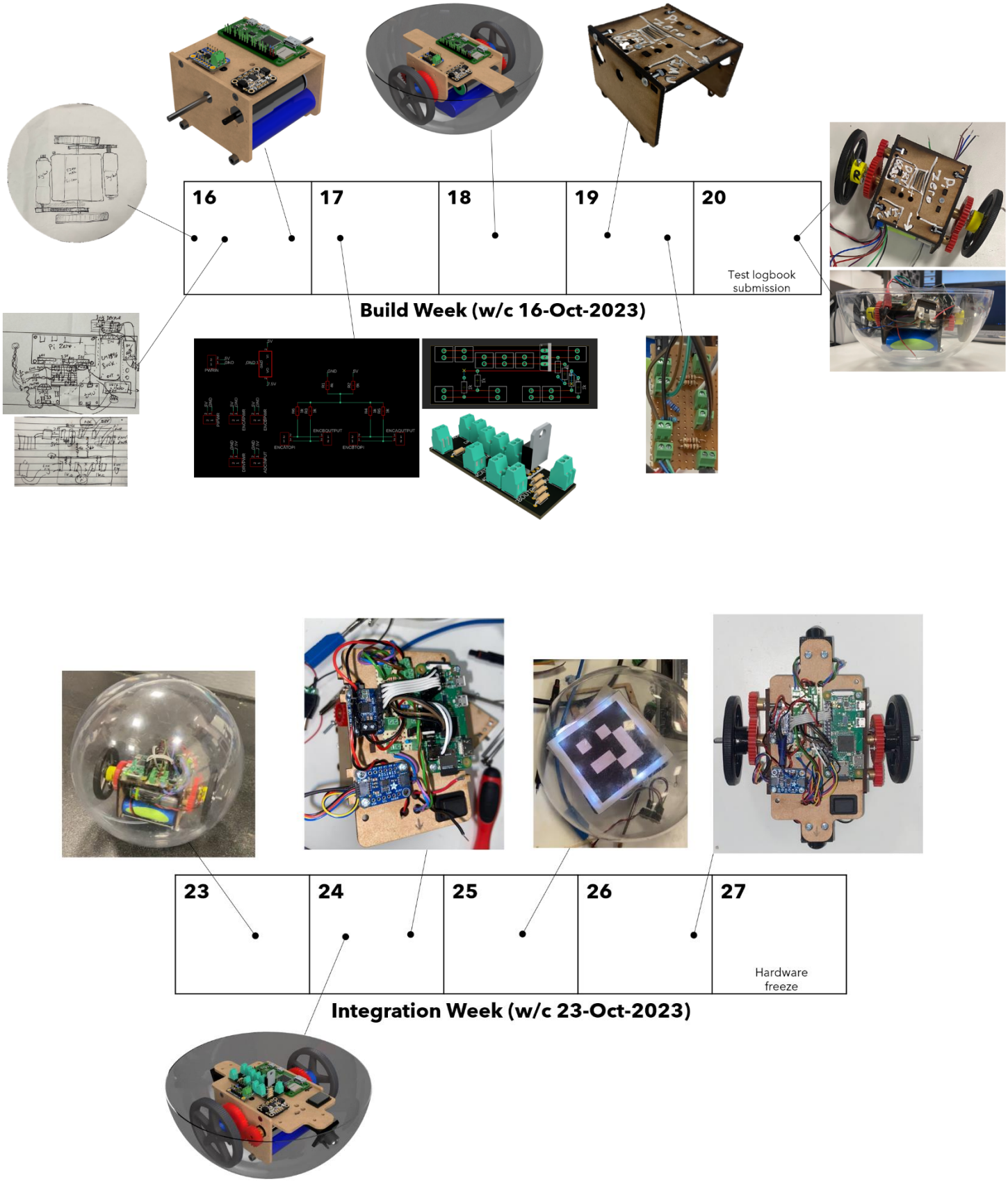


Figure 11 Design and development activities throughout the build and integration weeks

References

1. Mistry, Hena (2023) *Mechatronics 2: Computer Vision System,* University of Bath

2. Hofmann, Maxime (2023) *Mechatronics 2: Mission control,* University of Bath

3. FARO. *GD&T in Precision Engineering: Using slots in Precision Location*. Available at: https://www.faro.com/en/Resource-Library/Article/gd-t-in-precision-engineering-using-slots-in-precision-location [accessed 03-Nov-2023]

4. Huang, G.Q. (2012) in *Design for X: Concurrent engineering imperatives*. Springer-Science+Business Media, B.V.

5. H. V. Alizadeh (2017). *Spherical Mobile Robot design*. Online, Available at: https://cim.mcgill.ca/~hva/Spherical_Robot/ [accessed 03-Nov-2023]

6. Cutlasercut.com (2023). *What is laser kerf?.* Online, available at: https://cutlasercut.com/drawing-resources/expert-tips/laser-kerf [accessed: 03-Nov-2023]

Appendix A

- **SR-01 Subsystem control:** Partially tested. In order to demonstrate complete function of the robot mechanics and electronics, basic open loop control from a laptop hotspot was implemented to show that the robot could move forward, backwards, and turn to the left and right.
  It was also intended that this was run with a game controller for increased control and testing purposes, but this proved too difficult to implement within the timeframe and simply using sliders in the Simulink model sufficed.

- **SR-05 Sticky tyres:** Pass. Firstly, the rubber wheels purchased were more securely directly fixed to the gear system so they could not slip relative to the motor. Also it was confirmed via observation during testing that the wheels would not slip against the inside of the sphere by driving the robot at full speed, checking the angle of the robot, and then statically rotating the robot to confirm wheel slip occurred at a greater angle.

- **SR-10 Battery voltage:** Pass. Implemented with an ADC module (ADS1015) which feeds battery voltage to the Raspberry Pi over I2C.

- **SR-13 ArUco position:** Pass (Amended). The ArUco code was mounted unobstructed to the top of the robot mechanism inside the sphere. Keeping the ArUco upright was not an issue as this was dealt with by controlling the accelerations of the robot and keeping a low centre of gravity. Dealing with glare from the sphere was a bigger issue that is covered later in this report.

- **SR-15 Friction:** Pass. Covered by SR-05 and also visual observations during testing at maximum possible acceleration confirmed the sphere did not slip against the floor.

- **SR-16 Oscillations:** Moved. Using the IMU feedback to dampen oscillations was passed to Maxime as mission control.

- **SR-17 Turning radius:** Pass. Robot can spin on the spot so turning radius = ball radius (100mm).

No.	Requirement	Target	Must/ Wish	Test Method
		Motor Assembly		
SR-01	Sub-system control	Integrated Sub-system behaves as expected under RC control	Must	Basic RC control direction and speed can be sent to the motor controller (pi) to drive the sphere robot to specific locations
SR-02	Motor Control	Closed-loop control of motor speed is implemented with PI control	Must	Under **no load**, encoder output speed is graphed and compared against demanded to confirm appropriate PI response
SR-03	Motor Drivers	PWM signal from the motor controller drives each motor via the motor driver	Must	PWM signal can be varied to adjust motor output speed and direction (open-loop)
SR-04	Drive Wheels	Motors are mechanically linked to drive wheels	Must	Motors drive the wheels without putting excessive strain on motor bearings (i.e: flexible/soft coupling, or belt driven)
SR-05	Drive Tyres	Drive wheels have sticky 'tyres'	Must	Under normal drive conditions observe that wheels do not spin against the inside of the shell
SR-06	Motor Feedback	Motor speed/position is fed back to the Raspberry Pi to enable closed-loop control	Must	Test encoder output. Compare demanded speed to actual speed by averaging over 20 rotations
		Power Assembly		
SR-07	Power Converter	Regulated 5V power supplied to Raspberry pi	Must	DC-DC Buck converter regulates 7.4V battery voltage to 5V supply for Pi (check with multimeter)
SR-08	Battery Charging	Batteries can be charged without removal	Wish	Design allows access to battery cable without major disassembly of robot
SR-09	Battery Maintenance	Batteries can be removed or swapped easily	Wish	Design incorporates easy access to batteries for removal
SR-10	Battery Voltage	Feedback given to user of battery voltage state	Wish	Indicator LED or display indicates when battery voltage is low
SR-11	On/off switch	Robot includes convenient on/off switch to save power	Wish	Accessible on/off switch included in design, no dismantling required
		Mechanics		
SR-12	Low Centre of Gravity	Heavy components are placed low down to increase stability	Must	Robot self-rights when upside down (with power off)
SR-13	ArUco positioning	ArUco is placed flat on top of the mechanism and robot stays upright	Must	Confirm Vision system can always see Robot ArUco when inside the test area
SR-14	Size	Fits inside a 198mm (internal diameter) plastic ball with tolerance	Must	Analyse CAD model, allow for ±2mm of internal diameter variation
SR-15	Friction	Drive wheels/outer shell do not spin against the ground	Must	Observe whether the outer shell spins against the floor. Reduce max demanded acceleration or increase robot weight if necessary.
		Control		
SR-16	Oscillation damping	Robot (and ArUco) stays upright via active control mechanism	Wish	Closed-loop feedback with an IMU is used to actively adjust robot position to keep upright and damp oscillations
SR-17	Turning radius	Robot can manoeuvre withing the test area (2m x 2m)	Must	Turning radius <0.5 m