



While you're waiting for us to get started, why not checkout our website?
<https://robocupjunior.org.au>

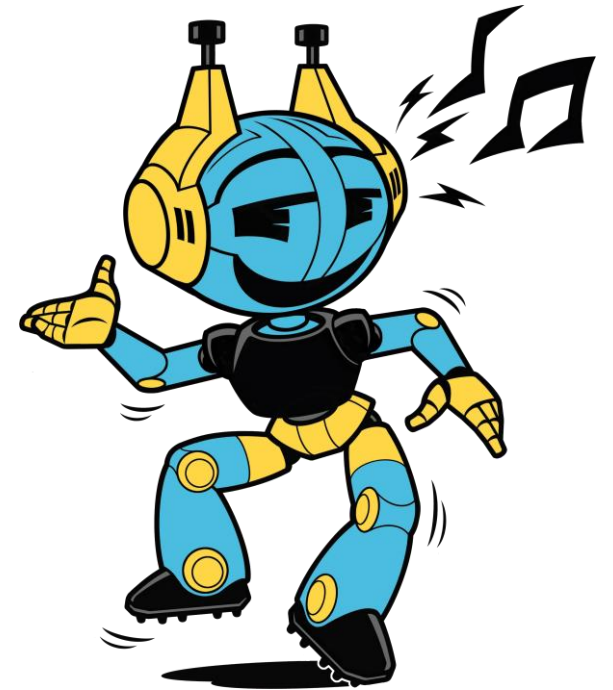
Welcome

Thank you for joining us for our 2023 Online CustomBot Workshop

Liam Whitehouse (Workshop Lead)
liam.whitehouse@robocupjunior.org.au

Contact details of section presenters are included later in the presentation

Please note this evening's workshop is being recorded and will be published online with the slides to support the continuing sharing of knowledge



Structure of Today's Online Workshop

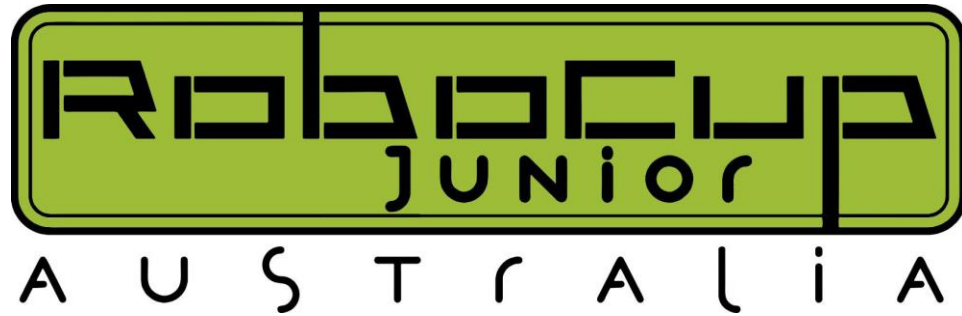
- Introduction of the team
- Introduction to CustomBots
- Taster of Vision Sensing in CustomBots
- CustomBot Starter Kits
- Split off to smaller groups:
 - Arduino in the Classroom
 - Arduino in OnStage
 - Arduino in Rescue Line
 - Arduino in Soccer
 - Raspberry Pi and Vision (Rescue Line Focus)



Introducing the Team

- Liam Whitehouse – RCJV Recue Line Coordinator ([email me](#))
- Evan Bailey – RCJA President, RCJV State Chair ([email me](#))
- Margaux Edwards – RCJA OnStage Coordinator, RCJQ State Chair ([email me](#))
- Ashley Kasper – RCJNSW Committee Member ([email me](#))
- William Plummer – RCJQ State Vice-Chair ([email me](#))
- Seb Taylor – RCJV Committee Member ([email me](#))
- Zac McWilliam – RCJA & RCJV Digital Platforms Coordinator ([email me](#))
- Tim Ronchi – RCJV Rescue Maze Coordinator ([email me](#))



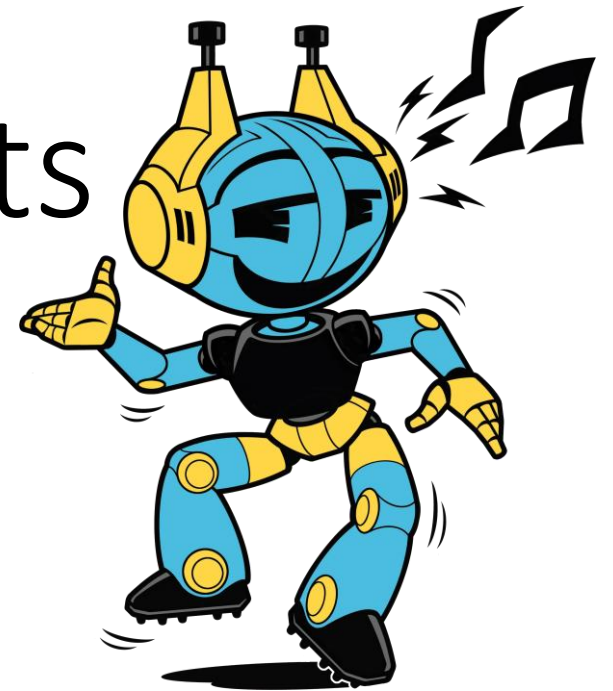


While you're waiting for us to get started, why not checkout our website?
<https://robocupjunior.org.au>

Introduction to CustomBots

Thank you for joining us for our 2023 Online Arduino Workshop

Liam Whitehouse – RCJV Rescue Line Coordinator
liam.whitehouse@robocupjunior.org.au



Interruption Policy

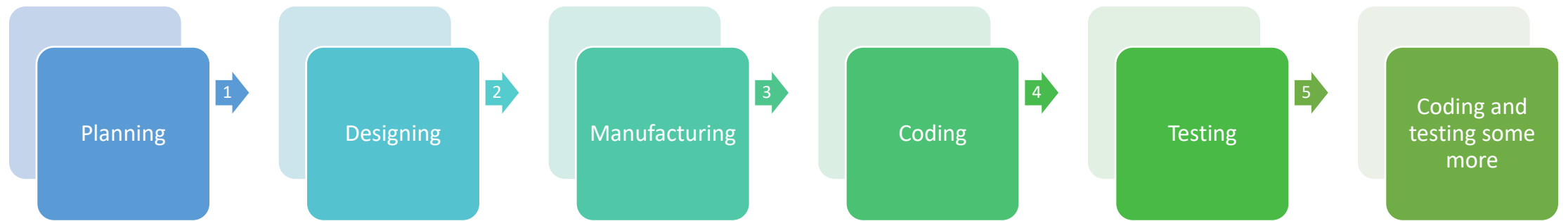
Please interrupt at any
time as often as you like*

This is a Workshop *not* a Lecture

* Yes, like that annoying kid in class today

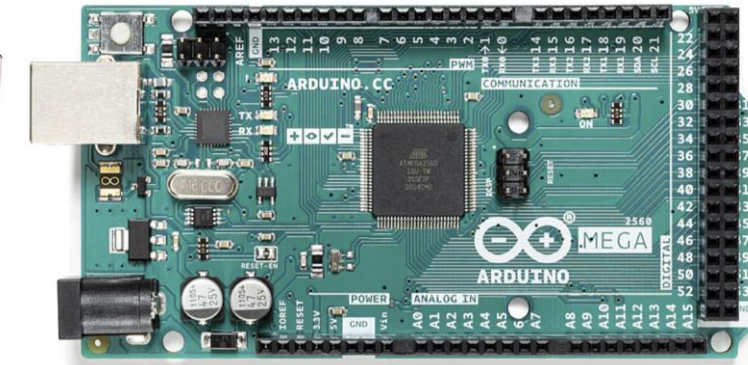
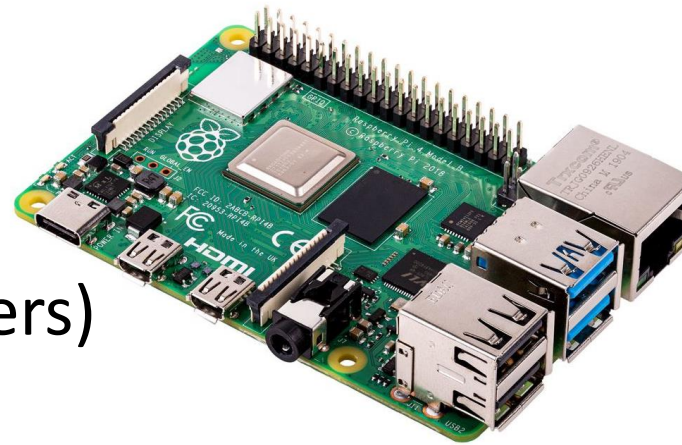


The Process



Planning – Selecting Your Processor

- Arduino
 - Uno
 - Mega (More Ports)
- Arduino Like
 - Adafruit Feather
 - Teensy
- SBCs (Single Board Computers)
 - Raspberry Pi
 - NVIDIA Jetson
 - Capable of Vision Processing, but more \$\$\$



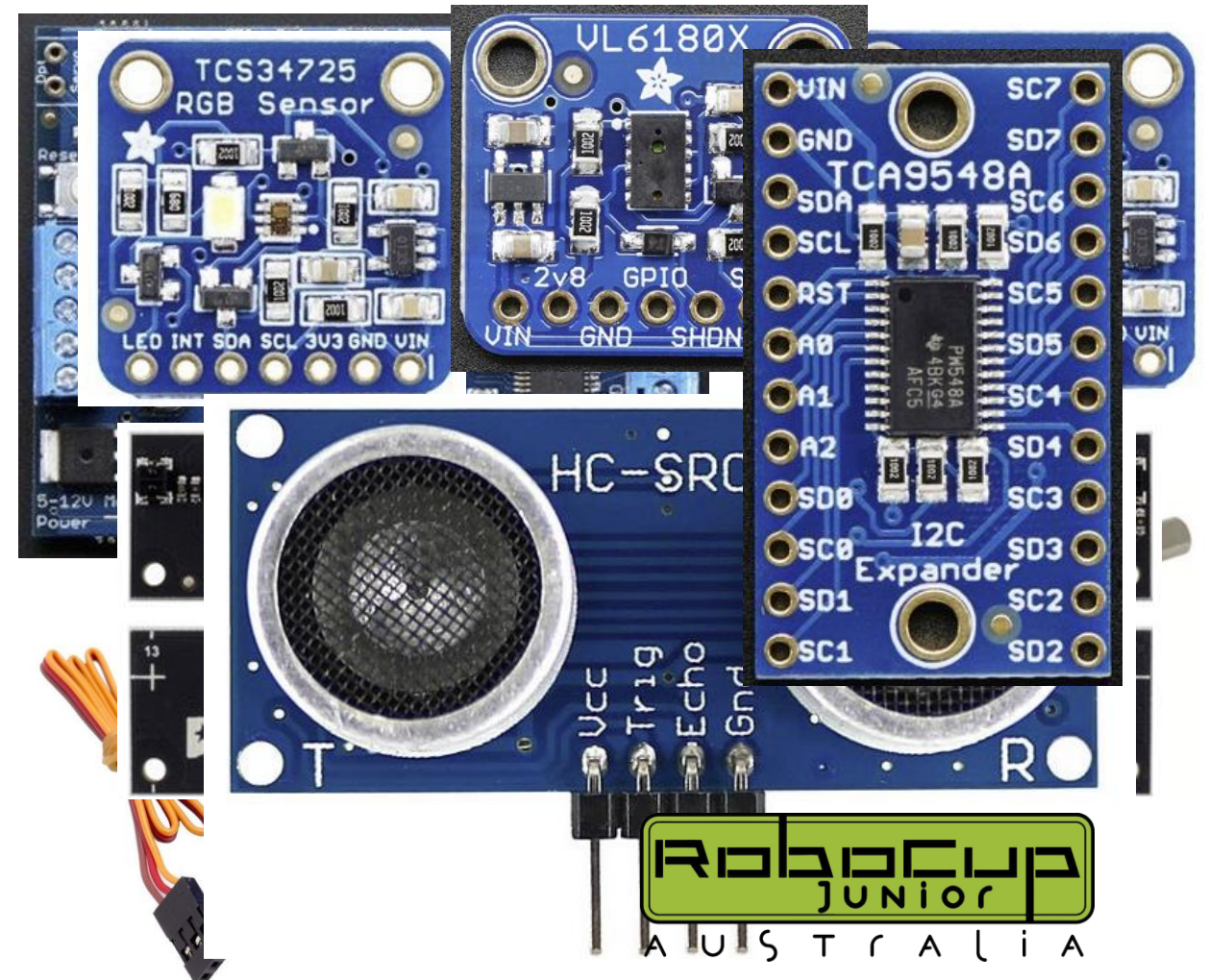
Planning – Choosing your parts

- It's not as easy as Lego
- Predominantly open source
- Many different parts that achieve the same thing (e.g. several options for colour sensors), so selection becomes difficult
- There are some good resources on the RoboCup Website to help you get started: <https://www.roboocupjunior.org.au/resources-for-a-arduino-based-rescue-robots/> (note these are not perfect, and some are no longer for sale – we are working on an updated version)

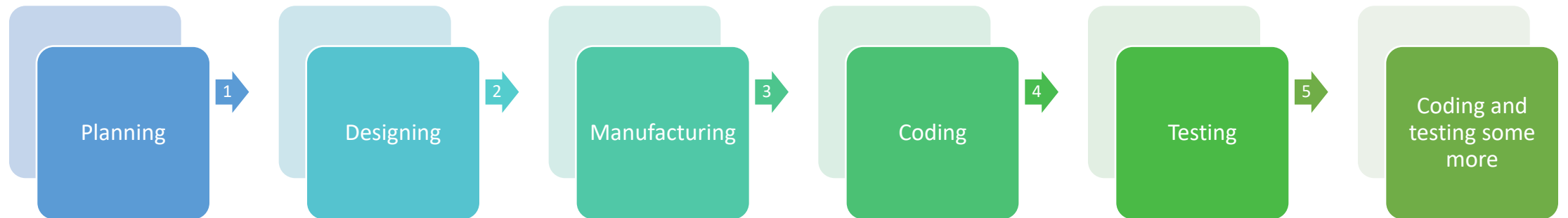


Planning – Choosing your parts

- Motor Shield/ Motors/ Servos
- Color Sensors
- Line Array
- Time of Flight (TOF)
- Ultrasonic Sensor
- Port Expander (this image is for I2C)



The Process

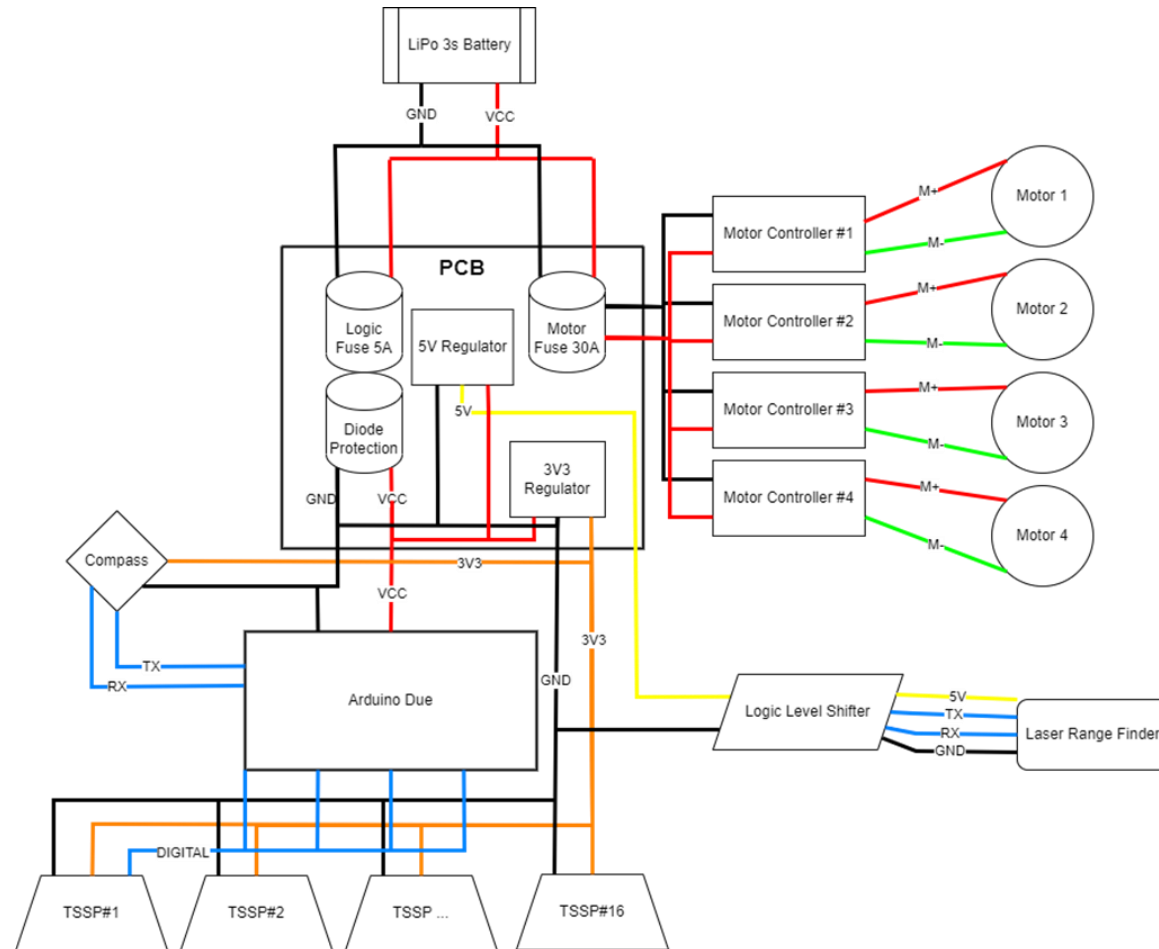


- Get the planning and designing section done quickly, because the making, coding and testing components are a lot of work, especially in the custom robot world because stuff goes wrong!

Planning – Manufacturing

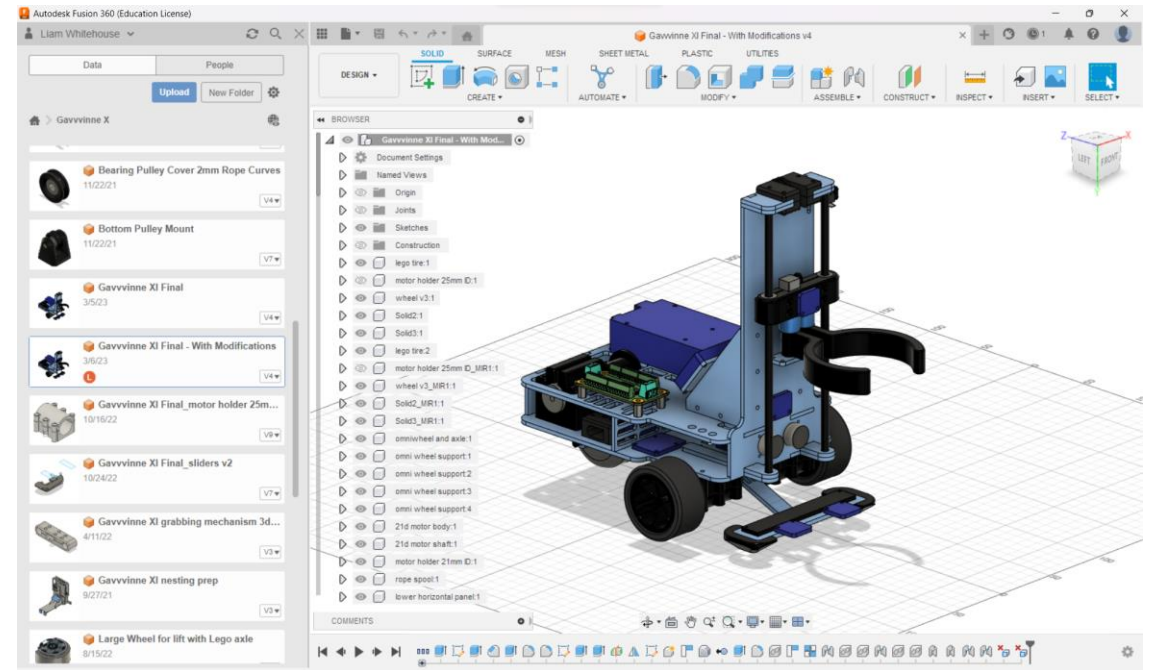
- How are you going to build your robot?
- What material would you like to use? (Wood, Plastic, Metal)
- Are you going to 3D print, Laser Cut, CNC, or a combination of them
- This is important to consider whilst also considering the next section, which is an amazing segway...

Planning – Design



Designing – Software

- Fusion 360 – My personal favorite, available free for educators & students
- TinkerCAD – Great for beginners, but can be limiting
- Adobe Illustrator
- Sketch Up
- Solid Works

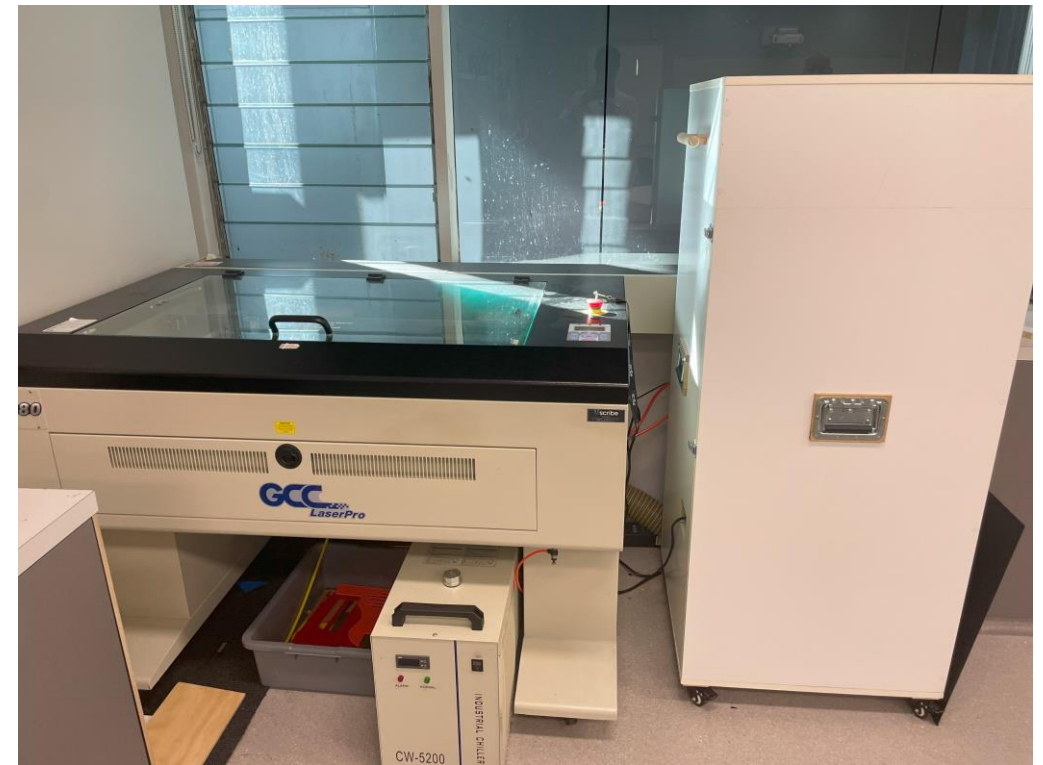


Designing

- Start to think about:
 - Materials
 - Manufacturing
 - Tools
 - Components

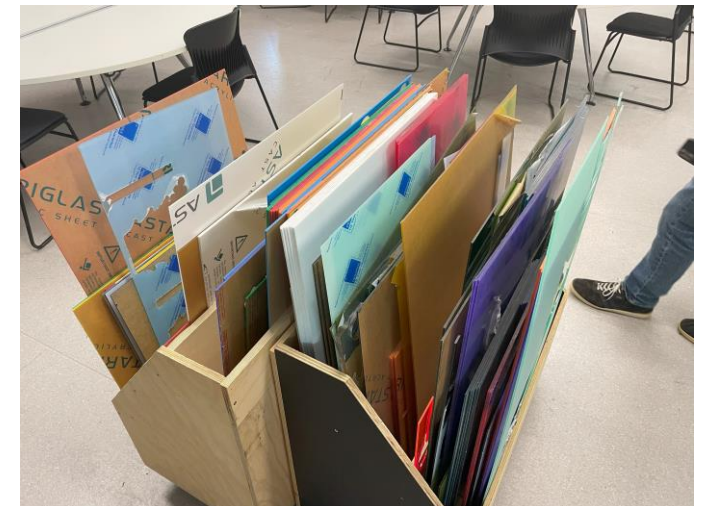
Manufacturing – Laser Cutting

- A Laser Cutter is great for building a base plates
- Very expensive (good to run alongside other programs) (i.e. for Schools and Product Design classes)
- Very precise, and fast
- This one here costed about \$15,000 to set up
- Price range of around \$2500 - \$40,000
- <https://www.teaching.com.au/catalogue/mta/mta-stem-laserbox>



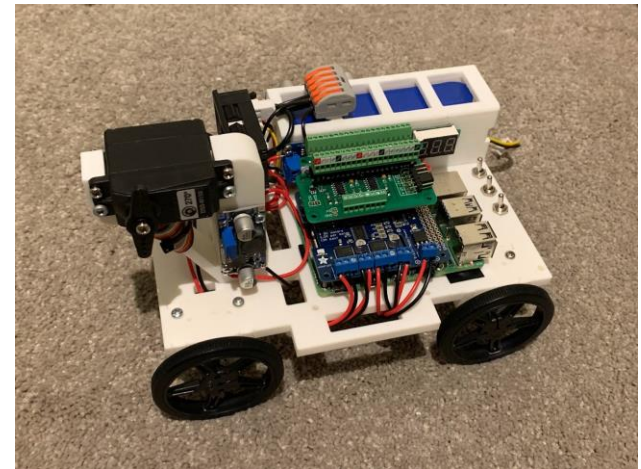
Manufacturing – Laser Cutting

- Acrylic 3mm Plastic is one of the most commonly used materials, including MDF wood.
- A4 size cost about for a sheet \$4-6, however you are only able to buy a full sheet which is about \$140
- Some materials **cannot** be cut due to toxic fumes produced or require specific filtering systems to be installed



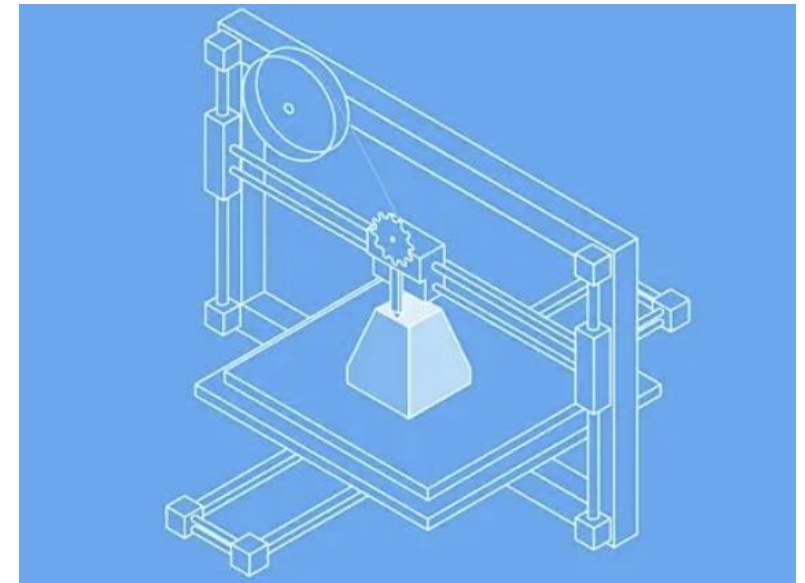
Manufacturing – 3D Printing

- Great for smaller components
- It is recommended to print separate components which connect together with screws, such as motor hubs, but there has been both success and failure in the past when printing the entire robot, in the end, it comes down to design.
- The more parts it is split into, usually the better and more flexible. Also allows for easier repairs, should something break.



Manufacturing – 3D Printing

- Many Options – definitely do your research!
- While resin printers seem cool – limited by cost and use toxic materials so may present OHS issues
- We would recommend a Fused Deposition Modeling/Fused Filament Fabrication (FDM/FFF) system, which work by extruding plastic filaments (most commonly ABS & PLA)
- ABS is less brittle and more tolerant to changing temperature, however requires an enclosed print chamber and heated print bed
- PLA is more forgiving to print with, but can warp (after printing) if exposed to warm temperatures
 - When starting out, PLA is a great option, and is still great for experienced 3D printers to use for more complex models too
- A wide variety of other materials are available to print including TPU, and CF Nylon. They all have their uses but are usually harder to print or are for very specific purposes. TPU is a flexible plastic and can be used to make custom tires and compliant mechanisms
- Use the filament recommend for your printer by the manufacturer, as some brands use slightly proprietary formulas that are optimised to perform best with their printers
 - Savings from cheap filament are quickly eroded by wasted filament consumed by failed prints!
 - Poor quality & unreliable printers also cause a large waste of filament, however the price of the printer does not always correlate to the quality!



Manufacturing – 3D Printing

- School Friendly Printers:

- Bambu Lab X1 Carbon

- Cost: \$2,200
 - Generally excellent, highly reliable, fast, high-quality prints, self-calibrating and self-error identifying (broken filament, peeled off first layer etc.) Enclosed build chamber for higher temperature plastics
 - Great option to get started with, easy to set up and get it printing well

- Bambu Lab P1P – Stripped down version of X1

- Cost: \$1,069
 - Generally excellent, highly reliable, fast and high-quality prints. Can be easily enclosed by user with panels if higher temperature plastics need to be printed
 - Great option to get started with, easy to set up and get it printing well

- Ultimaker

- Cost: \$6,000-7,000
 - Great if it can be mastered, but very difficult to master

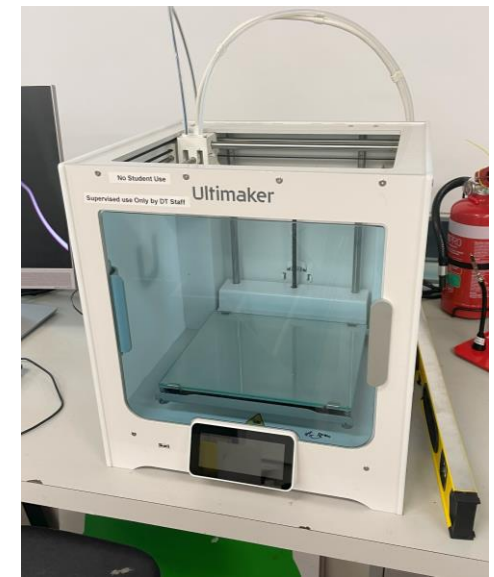
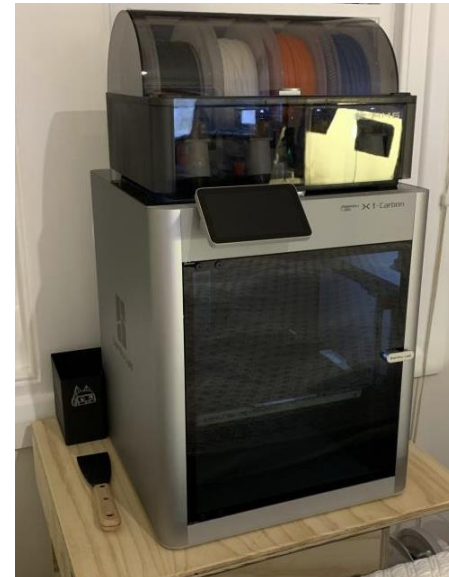
- Cheaper Hobbyist Printers:

- Ender 3 Pro | Ender 3 S1

- Cheap, little bit of work to setup
 - Cost \$300 - \$500

- Prusa MK3S+

- Mid range, High quality prints
 - Cost \$1000



Manufacturing – CNC

- CNC – Computerized Numerical Control. It is a slightly cheaper alternative to laser cutting which is less precise and requires a lot more maintenance but still not a terrible idea. (For schools I would recommend investing in a laser cutter)
- Can cut a wider range of materials
- Cost: about \$9,000
- Harder to use & slower to cut than a Laser Cutter



Manufacturing is Expensive!

Many of the machines mentioned above are pricey!

- Many local and international companies offer manufacturing services that very competitive prices
- Have a look around, there is a lot of competition in this market segment
- Local manufacturing companies and MakerSpaces love supporting RoboCup activities and make great team sponsors, especially in-kind support (machined components, technical advice etc.)


Check your local
MakerSpace



3D Printing

Our fleet of 3D printers are optimized for different uses. The fully enclosed adventure 3 & 4 printers are great for PLA or ABS, and our Mar 2 Pro prints resin with incredible resolution.

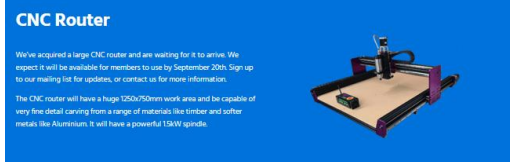
All of these printers are well maintained and very simple to use.



Laser Cutting

Our professionally maintained laser cutter allows you to create amazing custom signage, project enclosures, works of art and more. It can engrave and cut up to seven sheet materials including acrylic, MDF, plywood, foam, and more!

Our camera assisted software means you can drag and drop your drawing straight onto your workpiece. The workflow is optimized for ease of use with a camera and auto focus system built right in.



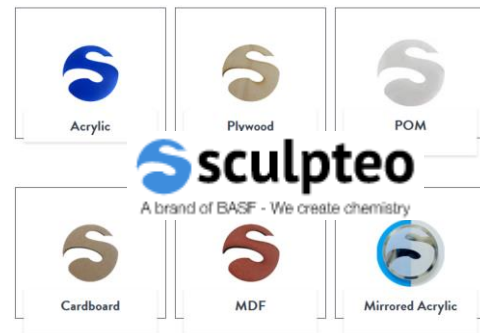
CNC Router

We've acquired a large CNC router and are waiting for it to arrive. We expect it will be available for members to use by September 20th. Sign up to our mailing list for updates, or contact us for more information.

The CNC router will have a huge 1500x750mm work area and be capable of very fine detail carving from a range of materials like timber and softer metals like Aluminium. It will have a powerful 13kW spindle.

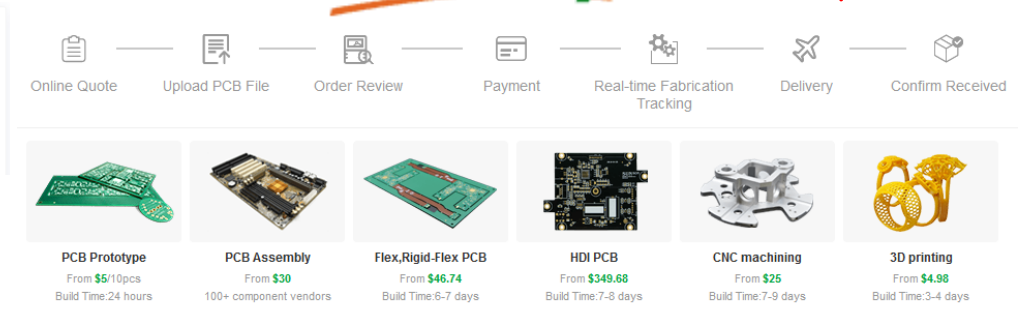


1&2 layers From \$2 /5pcs Build Time: 24 hours	4-8 layers From \$2 /5pcs Build Time: 4 days	PCB Assembly From \$8 Build Time: 24 hours	3D Printing From \$1 Build Time: 48 hours
--	--	--	---



sculpteo
A brand of BASF - We create chemistry

 Acrylic	 Plywood	 POM
 Cardboard	 MDF	 Mirrored Acrylic



Online Quote — Upload PCB File — Order Review — Payment — Real-time Fabrication Tracking — Delivery — Confirm Received

 PCB Prototype From \$5 /10pcs Build Time: 24 hours	 PCB Assembly From \$30 100+ component vendors	 Flex,Rigid-Flex PCB From \$46.74 Build Time: 6-7 days	 HDI PCB From \$349.68 Build Time: 7-8 days	 CNC machining From \$25 Build Time: 7-9 days	 3D printing From \$4.98 Build Time: 3-4 days
--	---	---	--	--	--



<https://onlinelasercutting.com.au>

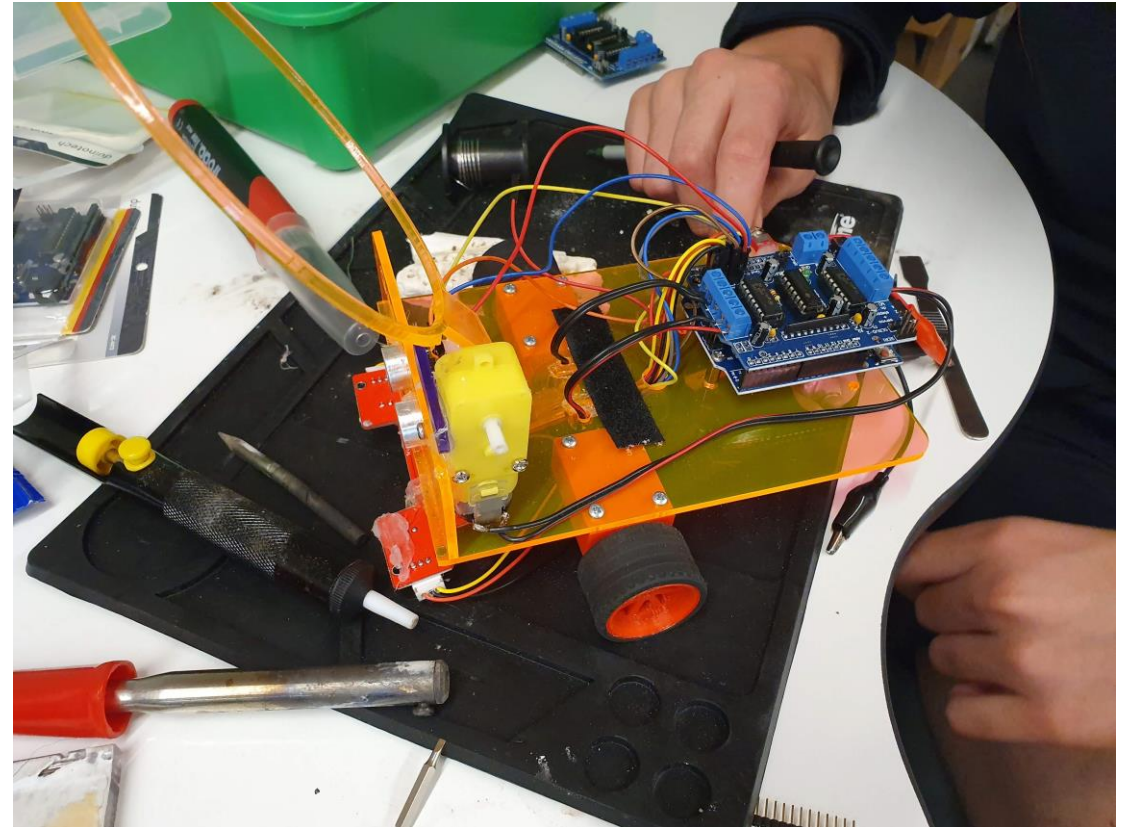
Manufacturing – Soldering Iron

- For doing Custom, this is essential!
- Allows you to make and repair robots
- Uses solder (a conductive metal) to attach wires and components to the main processor
- Look around the \$100 - \$250 price range
 - Get an iron with temperature control (300-400 degrees for most soldering)
 - Brass wool to clean the tips
 - Potentially some replacement tips



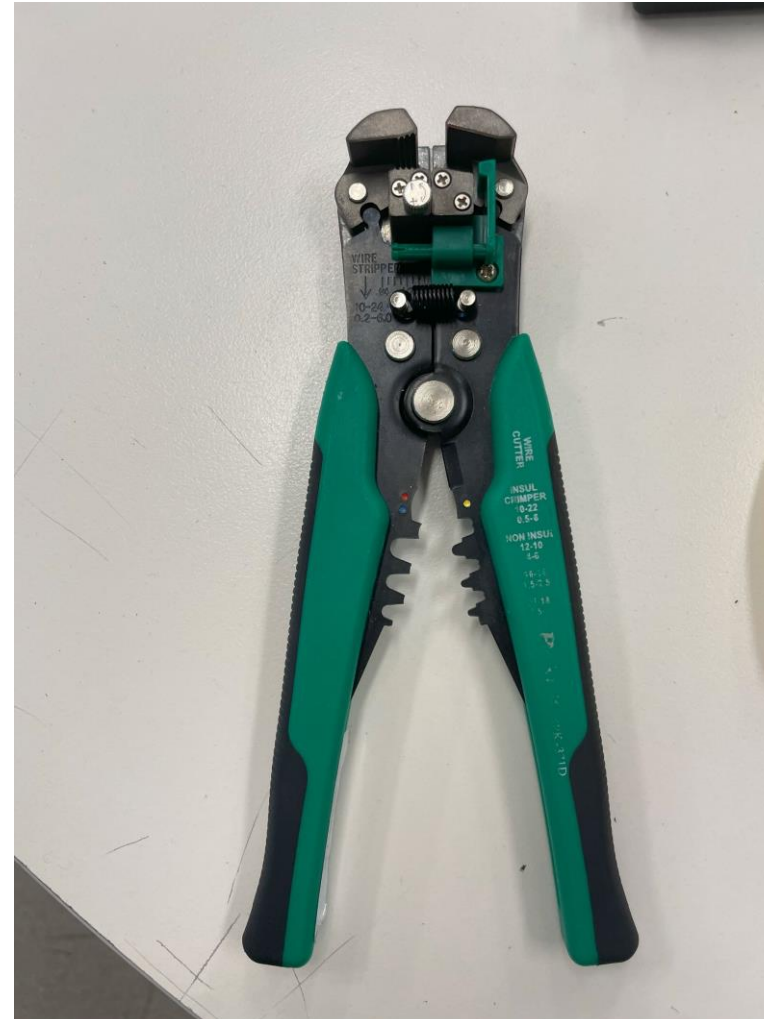
Manufacturing – Tools and Parts

- Over the next few slides, we will show some examples of what might be needed to help build a custom robot.



Manufacturing – Tools and Parts

- Wire Cutters
- Wire Strippers



Manufacturing – Tools and Parts

- You need the actual wires, we recommend getting a lot or a long wheel of it because I can guarantee you will make mistakes and need more of it.
- Recommended Wires and Sizes:
 - Use premade jumper wires in the first instance
 - Assortment of heat shrink tubing ranging from 1mm up to 4mm is useful
 - Most custom electronics use 0.1" (2.54mm) pitch crimped connectors
 - Custom length wires can be made requiring crimp on connectors, crimping tool, crimp housings and 22 to 28 gauge wire



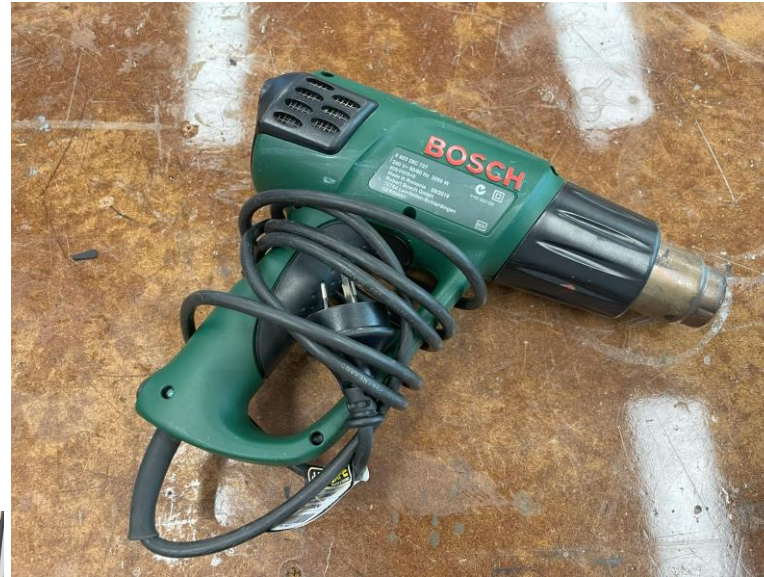
Manufacturing – Tools and Parts



- Multimeter
 - Great for identifying if wires have been correctly soldered.
 - Great for identifying why some components broke, for example if you gave it too much voltage.

Manufacturing – Tools and Parts

- Powered Drill
- Heat Gun
- Acrylic Cement
- Files



Manufacturing – Tools and Parts

- Screwdriver Sets
- Which brings me to...
- (another brilliant segway)



Manufacturing – Tools and Parts

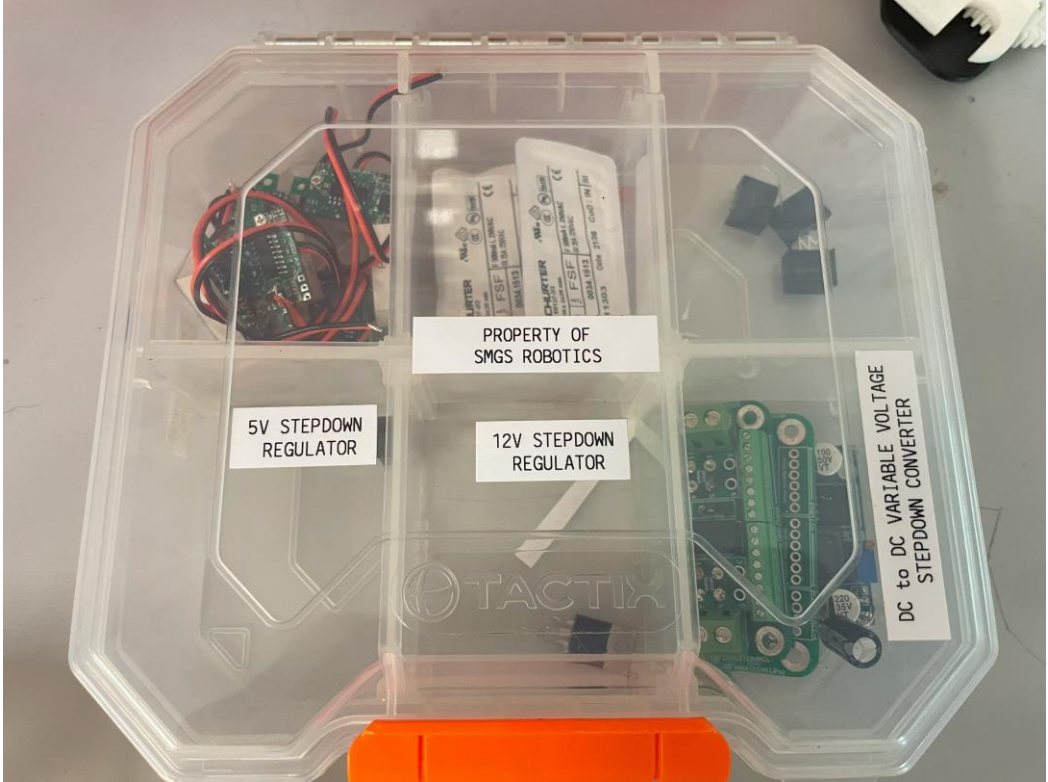
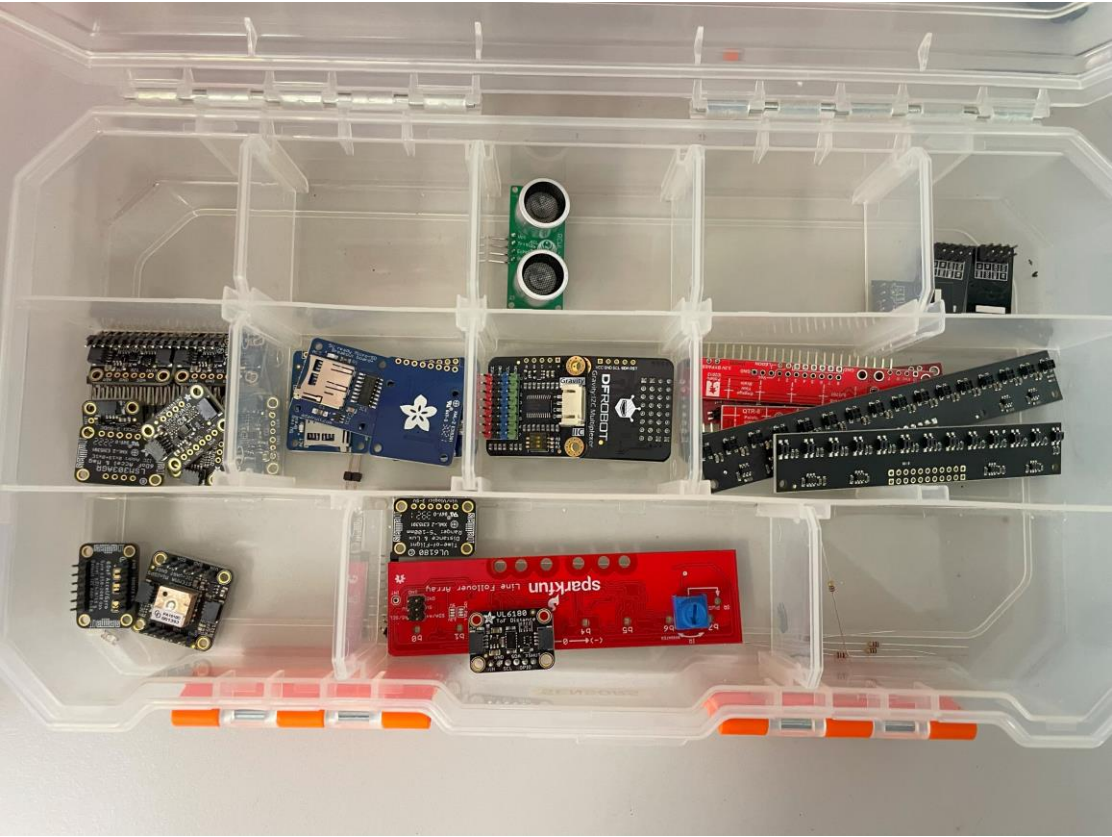
- Screws!



Manufacturing – Tools and Parts (Storage)



Manufacturing – Tools and Parts (Storage)

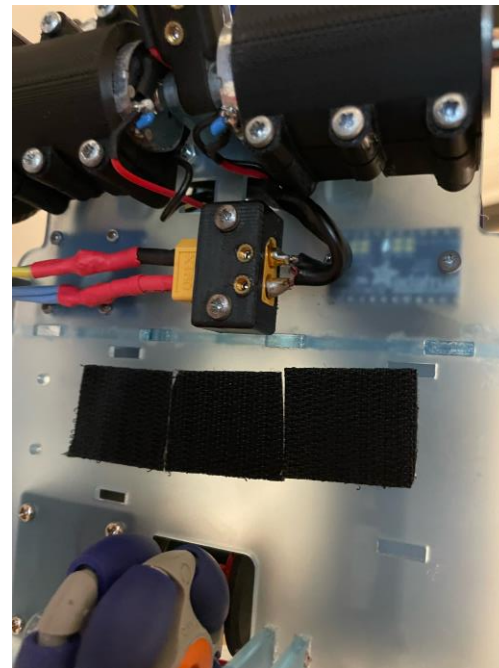
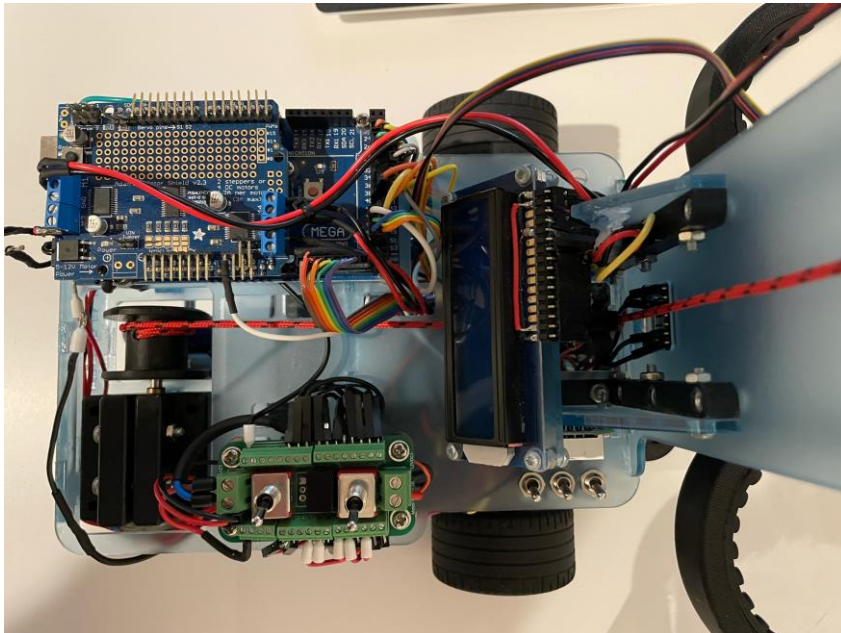


Manufacturing – Tools and Parts (Storage)



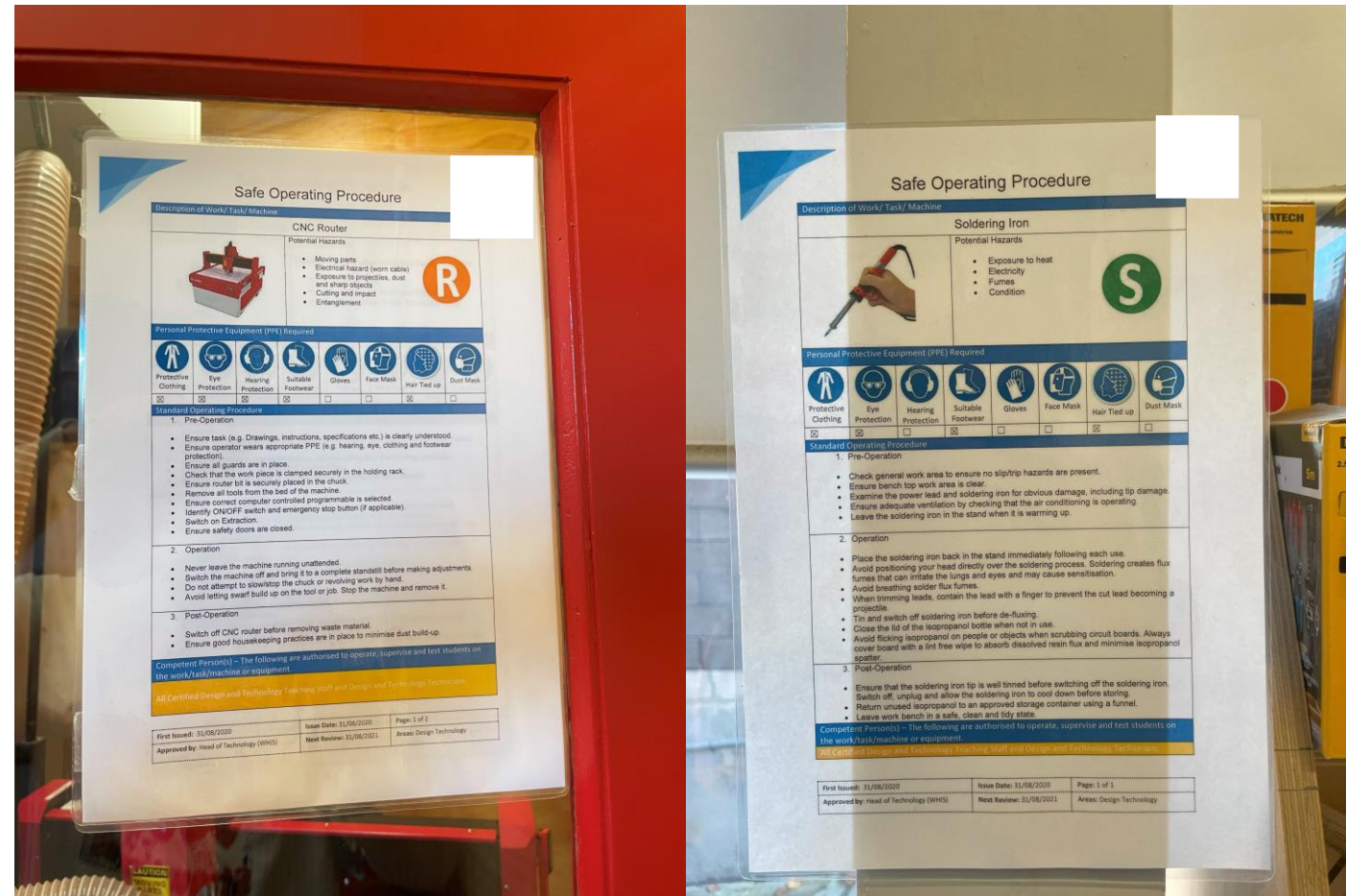
Manufacturing – Power (Caution)

- Robots need electricity to run...
- and you need to get power from a battery to the rest of the components (12V Battery, to components that only need 3.3V or 5V) - forgetting this can result in a lot of broken parts
- Power Distribution Board



Manufacturing – Safety

- As students transition into the CustomBot world the level of danger and number OHS issues that present themselves significantly increase compared to LEGO.
- Schools should consider the OHS risks associated with each of the tools they implement and Ensure students are properly trained in the use.



Manufacturing – Obligatory Battery Safety

LiPo (Lithium Polymer) Batteries are extremely powerful, compact and efficient, however caution and safety are a must.

- LiPo batteries have soft skins and when punctured can erupt violently in flames
- Careful and safe stowage is a must – LiPo Bags / Battery Boxes
- Mitigation of sharp objects near the battery whilst in the robots
- Don't leave batteries charging whilst unattended
- Think about travelling to competitions – Airlines have specific rules on how many batteries each person can carry



Manufacturing – Local Suppliers & Distributors

Many different types of parts are used in Custom Robotics, here are some suppliers with useful parts:

- Modern Teaching Aids (MTA) - General Robotics Parts
- Jaycar / Altronics – Local physical stores (bit of everything, but a slight price increase for convenience)
- Bunnings – Other larger equipment (Storage Boxes, larger tools)
- Local hobby stores – Sourcing batteries (AUS has strict battery shipping rules)

Manufacturing – International Suppliers

Many different types of parts are used in Custom Robotics, here are some suppliers with useful parts:

- Core Electronics – Robotics parts (Sensors, useful parts) (local store)
- Little Bird – Robotics parts (Sensors, basic useful parts) (local store)
- Element14 – Electrical/Mechanical/Tooling (All your equipment needs & many electrical / mechanical parts)
- SparkFun – Robotics parts (Sensors, useful parts)
- HobbyKing – LiPo Batteries
- Pololu – Motors, Regulators, Motor drivers
- RS Online - Mechanical (Fasteners, Nuts, Bolts, Mounting Standoffs)
- Mouser / Digikey – Electrical (PCB components, connectors, ICs)

Building your Makerspace?

Contact us if you have questions about recommendations on what to purchase, safety requirements etc.

Always keep a look out for Grants (Industry, State and Federal) as these can really support these programs



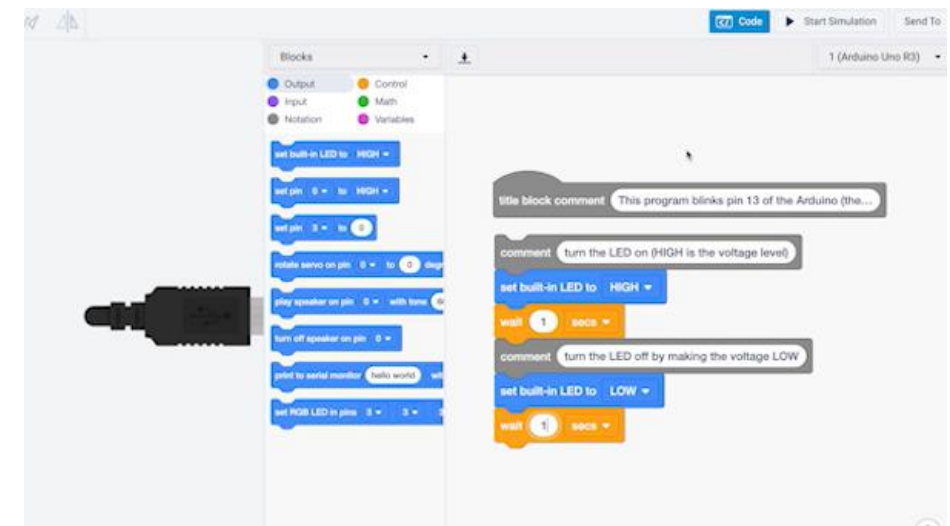
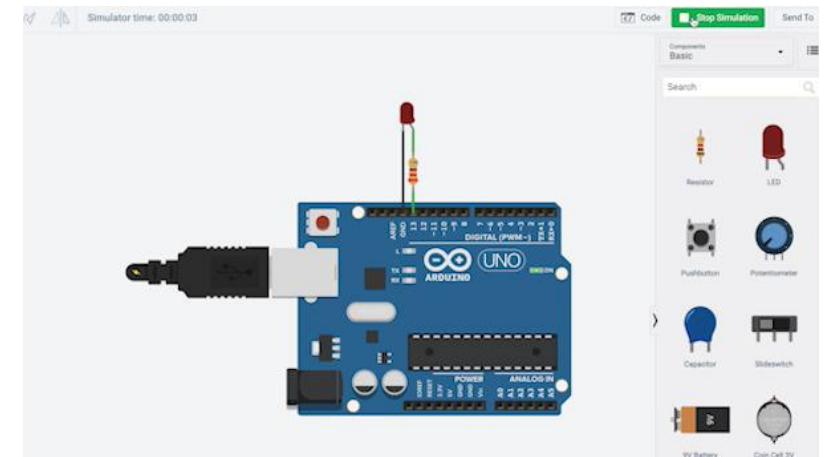
Coding – Software

Arduino

- Arduino IDE (C++)
- Tinkercad Circuits (Blockly, C++, Blocks/Code)
- PlatformIO in Visual Studio Code (C++)

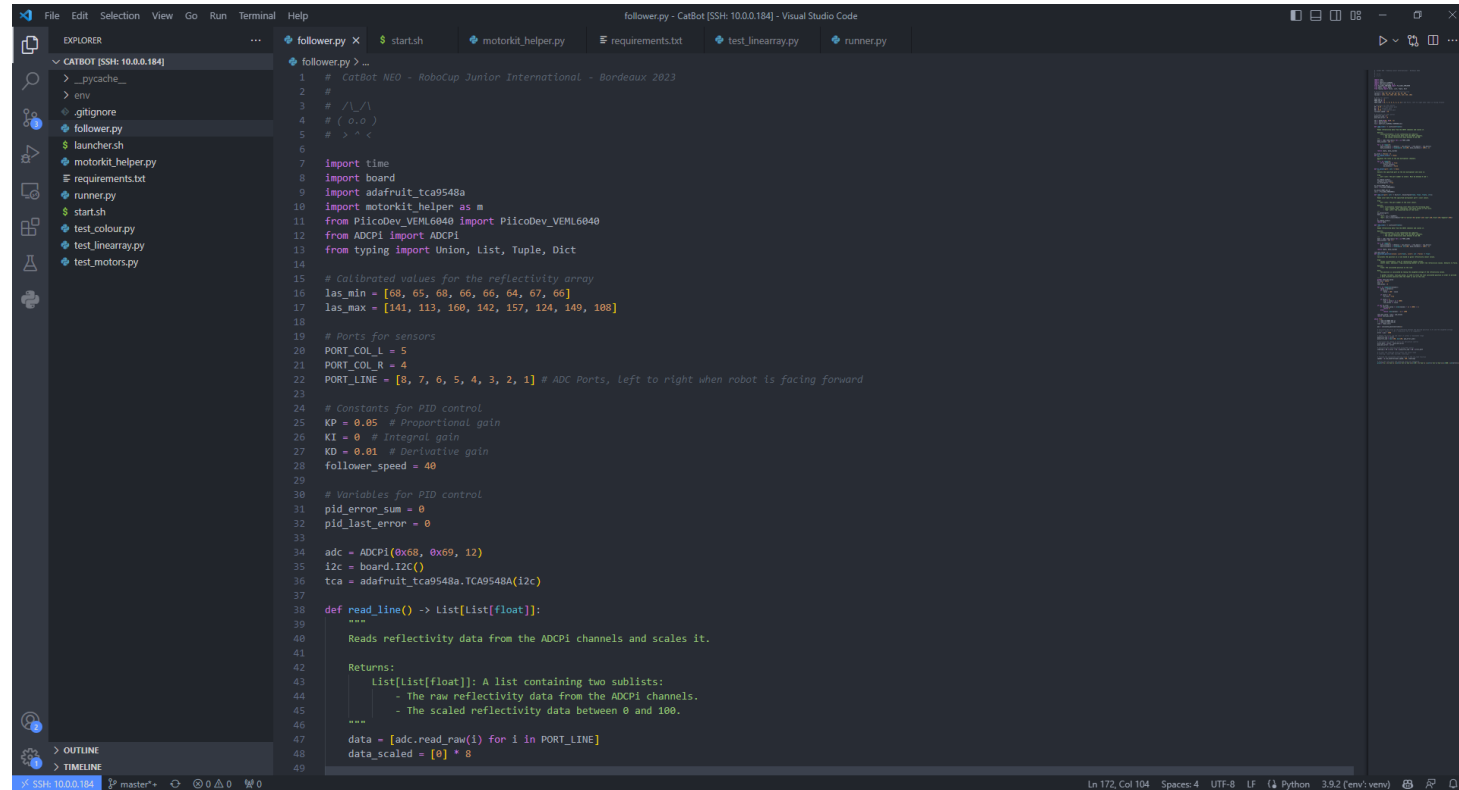
```
GawwinoX_5_4 | Arduino 1.8.19
File Edit Sketch Tools Help

GawwinoX_5_4
1 // Software to do: gridlock exit, greensone exit roopy yuma, night drive,
2 // abnormal green zones (weird shapes, multiple cans, black cans), further refine green turns
3 //
4 //
5 // Disappoint software to do: work out why void loop is taking so long to process, seeeav issue from R2B 8 regional (more testing needed, under different lighting conditions)
6 //
7 // Software to test: green turns on small tiles,
8 //
9 // Hardware to do: another set of new wheel hubs,
10 //
11 // Disappoint hardware to do: Trepotion colour sensors?
12 //
13 // Hardware to test:
14 //
15 // Libraries
16 //
17 #include <Adafruit_MotorShield.h>
18 #include <Adafruit_Gemmer.h>
19 #include <Adafruit_Protocol.h>
20 #include <Adafruit_TCS34725.h>
21 #include <Adafruit_VL6180X.h>
22 #include <NewPing.h>
23 #include <gsmemera.h> //Using library version 3.1.0
24 #include <80.h>
25 #include <801.h>
26 #include <810.h>
27 #include <Adafruit_VL53L0X.h>
28 #include <LiquidCrystal.h>
29 #include <I2C_V1.h>
30 #include <Adafruit_VL5310X.h>
31 //
32 // Motor shield
33 //
34 Adafruit_MotorShield AFMS = Adafruit_MotorShield();
35 Adafruit_DCMotor *left = AFMS.getMotor(2);
36 Adafruit_DCMotor *right = AFMS.getMotor(3);
37 Adafruit_DCMotor *lefts = AFMS.getMotor(4);
38 Adafruit_DCMotor *rights = AFMS.getMotor(5);
39 int leftSpeed = 0;
40 int grabSpeed = 0;
41 int leftSpeed = 0;
42 int rightSpeed = 0;
43 //
44 //Set Up Variables
45 //
```



Coding – Software

- Raspberry Pi
 - Code on your own computer (in python)
 - Use the command line to run the program
 - Visual Studio Code is a great program to start with, can connect through SSH for a full environment



```
1 # CatBot NEO - RoboCup Junior International - Bordeaux 2023
2 #
3 # /_/_/
4 # ( o.o )
5 # > ^ <
6
7 import time
8 import board
9 import adafruit_tca9548a
10 import PiicoDev_VEML6040 as PiicoDev
11 from PiicoDev_VEML6040 import PiicoDev_VEML6040
12 from ADCPI import ADCPI
13 from typing import Union, List, Tuple, Dict
14
15 # Calibrated values for the reflectivity array
16 las_min = [68, 65, 68, 66, 66, 64, 67, 66]
17 las_max = [141, 113, 108, 142, 157, 124, 149, 108]
18
19 # Ports for sensors
20 PORT_COL_L = 5
21 PORT_COL_R = 4
22 PORT_LINE = [8, 7, 6, 5, 4, 3, 2, 1] # ADC Ports, left to right when robot is facing forward
23
24 # Constants for PID control
25 KP = 0.05 # Proportional gain
26 KI = 0 # Integral gain
27 KD = 0.01 # Derivative gain
28 follower_speed = 40
29
30 # Variables for PID control
31 pid_error_sum = 0
32 pid_last_error = 0
33
34 adc = ADCPI(0x68, 0x69, 12)
35 I2C = board.I2C()
36 tca = adafruit_tca9548a.TCA9548A(I2C)
37
38 def read_line() -> List[List[float]]:
39     """
40     Reads reflectivity data from the ADCPI channels and scales it.
41
42     Returns:
43     List[List[float]]: A list containing two sublists:
44     - The raw reflectivity data from the ADCPI channels.
45     - The scaled reflectivity data between 0 and 100.
46     """
47     data = [adc.read_row(i) for i in PORT_LINE]
48     data_scaled = [0] * 8
49
```

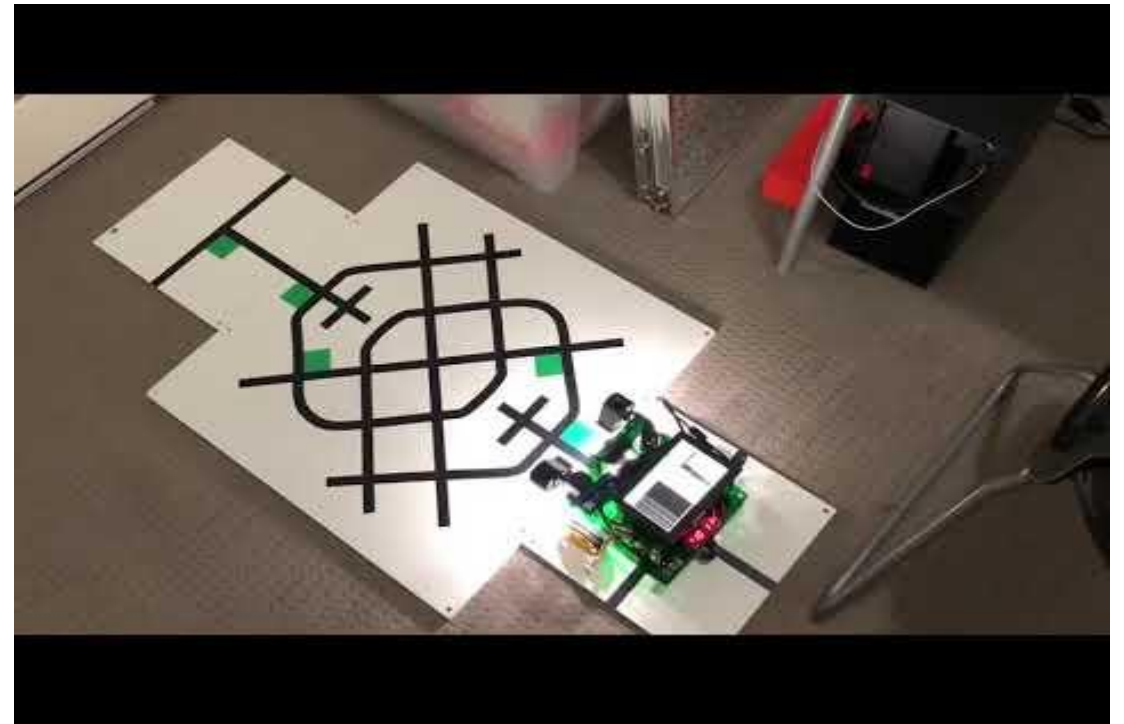
Coding – The Basics

- With every robot that we use, both Lego and custom work on the principal of recognition and reaction.
 - Recognition
 - Using sensors and if statements we determine if certain conditions are met which would trigger a response, for example if TOF has a value less than 100, recognises that there is something 100mm away from the TOF
 - Reaction
 - Based on the recognition, should the condition apply take an action. For example, stop the motors, then reverse for 2 seconds, then turn for 0.5 seconds.

Testing – the most important part

- Try your best to test as much as possible, this is the part that should take the longest
- Ideally try to test your robot in varying conditions (e.g. Light Conditions, uneven floor etc.)
- Also try to run the robot on many different combinations of courses to try and find flaws and bugs in your code (especially for Recue Line and Maze)

Computer Vision (Teaser)



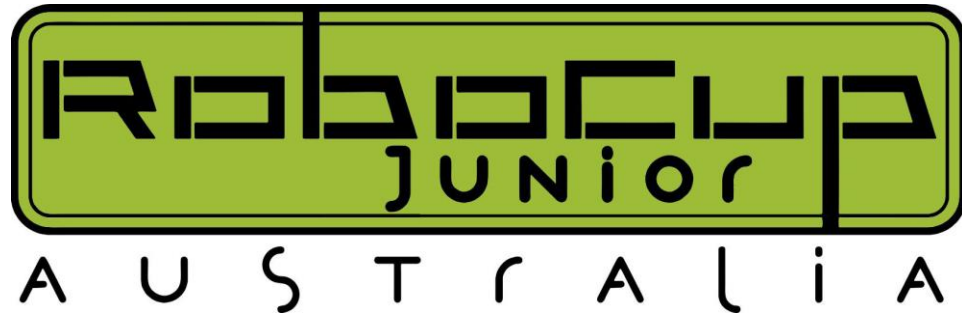
RCJA Certified CustomBot Starter Kits

- Selecting and sourcing parts is time consuming & tricky, especially for schools with difficulties purchasing from online stores
- If you don't have Laser Cutters, 3D printers or CNC routers, making robot chassis and other parts to attach sensors and motors is also tricky
- RCJA is here to help!
- We are currently deep into the R&D of some 'RCJA Certified CustomBot Starter Kits', which will contain all the parts you need to put together a CustomBot
 - Starting with Rescue Line, then expanding to other challenges based on market response
- We will also be putting together starter kits of tools & consumables needed to get started on CustomBots in your school
- We aim to have these for sale on our website within the next three months



What questions do you have?





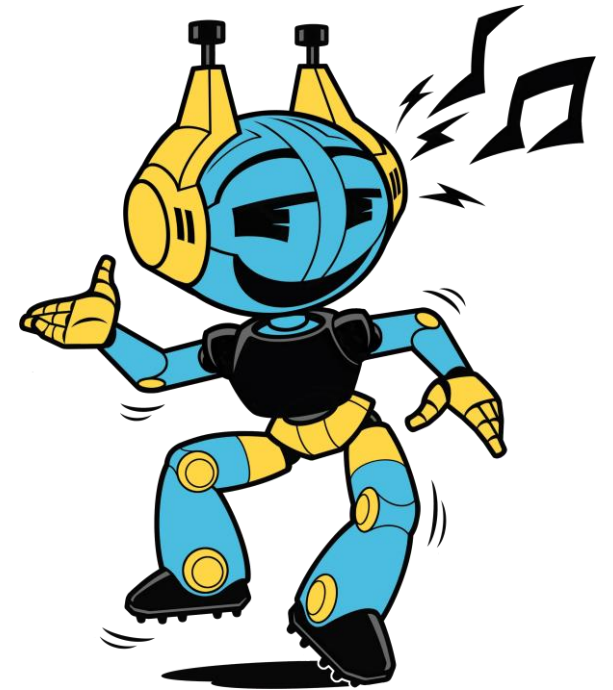
While you're waiting for us to get started, why not checkout our website?
<https://robocupjunior.org.au>

Arduino in the Classroom

Thank you for joining us for our 2023 Online Arduino Workshop

Ashley Kasper – RCJNSW Committee Member
(Curriculum and Implementation) |
ashley.kasper@robocupjunior.org.au

Tim Ronchi – RCJV Rescue Maze Coordinator (Technical)
tim.ronchi@robocupjunior.org.au



Interruption Policy

Please interrupt at any
time as often as you like*

This is a Workshop *not* a Lecture

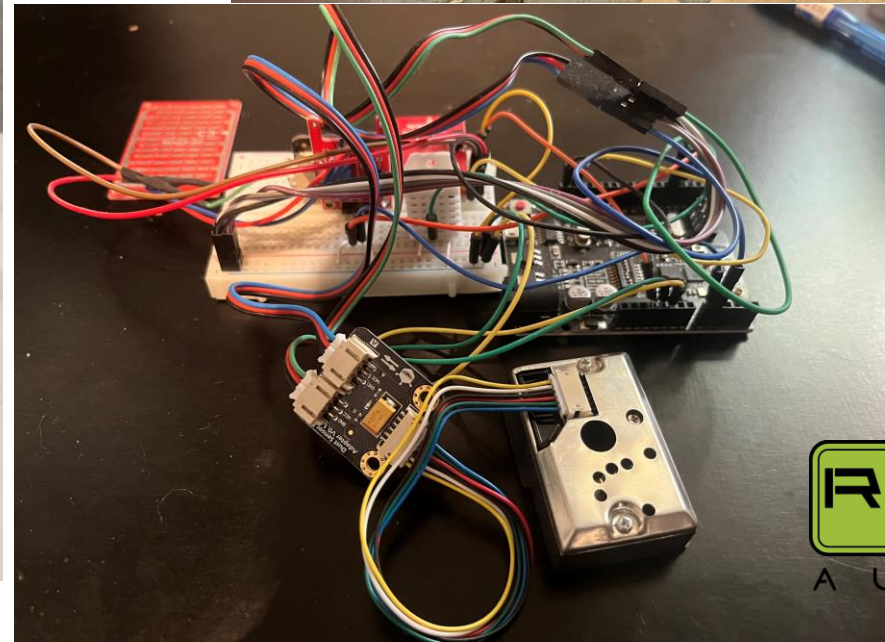
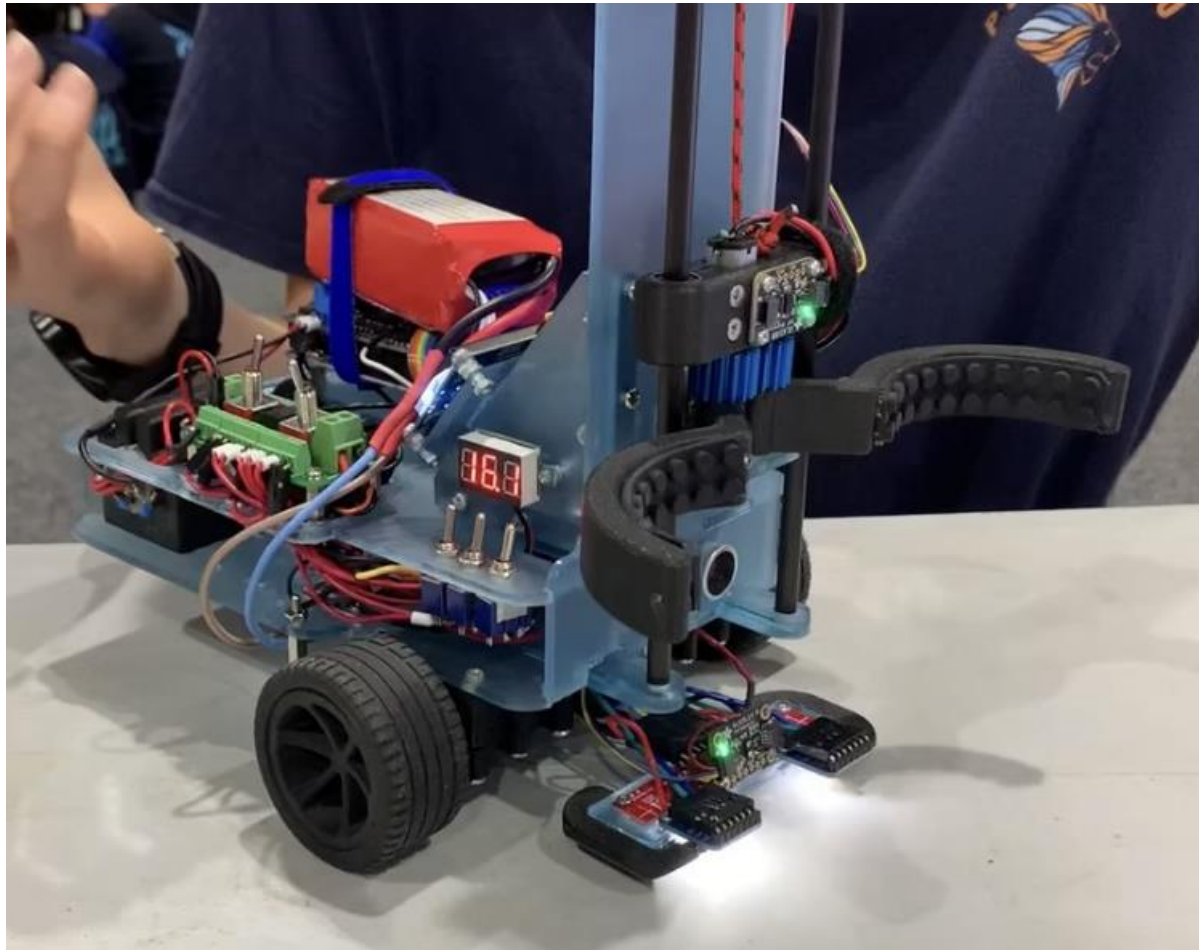
* Yes, like that annoying kid in class today



Types of Arduino

- Arduino Boards Guide: <https://www.arduino.cc/en/Guide>
 - Note: this only includes boards made by Arduino
- In the classroom, the Arduino Nano and Uno are common choices as they are cheap and powerful enough for most projects
- Computers and the Arduino IDE software require different drivers to interface with different boards. Drivers for most boards can be installed using the Arduino IDE
- Many companies make microprocessors that are Arduino replicas or modified versions of Arduino boards. Eg. Adafruit Feather
- There are also numerous other boards that use the Arduino IDE and same programming language. Eg. ESP32 (common and cheap microcontroller with built in WiFi and Bluetooth, great for IoT projects)

Versatility of Arduino



Sensors for Arduino

- Arduino boards are compatible with a massive number of sensors
 - Range from rotary encoders, to colour sensors, to temperature, to distance, to computer vision and so many more
- Some companies that make Arduino compatible electronics, such as motors, motor drivers and sensors, are Adafruit, DFRobot, Pololu & Sparkfun.

Resources for Arduino

- Official Arduino Resources
 - General Documentation: <https://docs.arduino.cc/>
 - Introduction to Arduino: <https://docs.arduino.cc/learn/>
 - Arduino Tutorials: <https://www.arduino.cc/en/Tutorial/HomePage>

Resources for Arduino - Electronics?

- Most name-brand manufacturers of electronics designed to interface with Arduino provide good guides and documentation to accompany them
 - Adafruit: <https://learn.adafruit.com/>
 - Pololu: <https://www.pololu.com/resources/documentation>
 - Sparkfun: <https://learn.sparkfun.com/>
- These usually include pinouts, wiring guides, recommended software libraries and example codes
- If you choose not to go with a name-brand option, if a name-brand component with the same chip can be found, you can often use the documentation and code library for it (no guarantees it will work though)

Resources for RoboCup Specific Robots

- Robocup Junior Australia Custom Electronics Resource: <https://www.robocupjunior.org.au/resources/custom-electronics/> This webpage includes:
 - Information specific to robots being built for RoboCup Junior
 - The most up to date version of this document
 - Pictures and videos of past robots for inspiration
 - Coming soon:
 - CAD models of commonly used electronic components
 - Variety of 3D printable CAD models of parts for robots



Resources for CAD?

- Fusion 360 Educational Tutorials - Product Design Online YouTube Channel: <https://www.youtube.com/@ProductDesignOnline>
 - Beginner tutorial playlist: <https://www.youtube.com/playlist?list=PLrZ2zKOtC-C4rWfapngngoe9o2-ng8ZBr>
 - CAD for 3D printing playlist: <https://www.youtube.com/playlist?list=PLrZ2zKOtC-D-tWVOhy-8pg-yrZe0ZMyV>
 - CAD for laser cutting playlist: https://www.youtube.com/playlist?list=PLrZ2zKOtC-B_HAKUEXhaHyK-2ksfFx2K

How to get started with the Arduino IDE?

- Installing boards
- Installing libraries
- How to access example sketches
- Serial monitor
- Serial plotter

How do we link digital technologies to the rest of the TAS syllabuses?

Food tech/
Agriculture

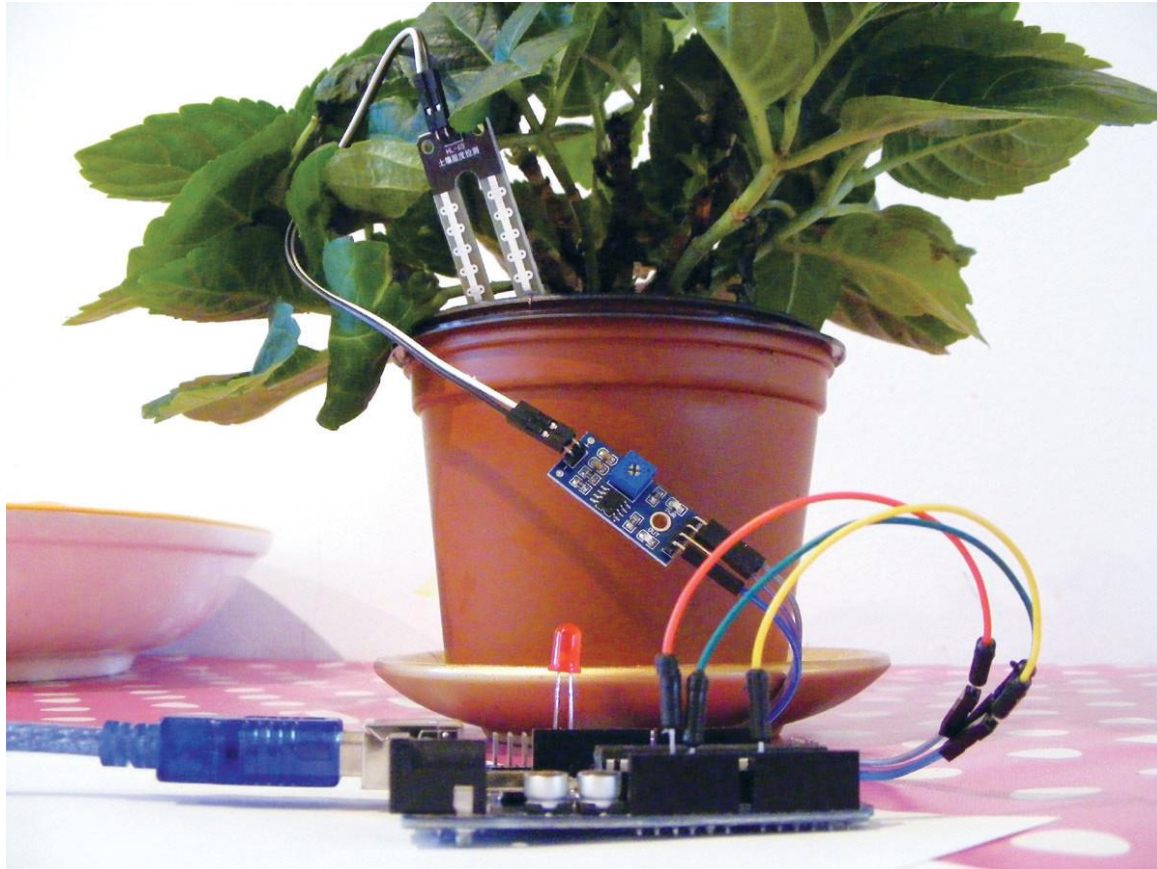
Textiles

- Wearable Technologies
- Conductive threads

Woodwork

Engineering

Applying Arduino to the Years 7+8 Syllabus



Applying Arduino to the Years 7+8 Syllabus

ARDUINO LESSON 1 - BLINK

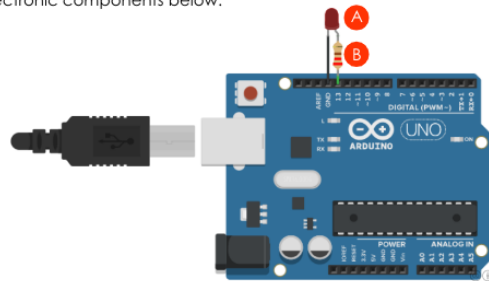
ACHIEVED

Refer to the MHS Arduino Lesson 1_Blink found on the TinkerCAD Website (<https://www.tinkercad.com/things/8LVFr3xcNI2>) for all details regarding hardware requirements, hardware connections and sketch details.

The below image illustrates the circuit schematic for the first Arduino activity.

Activity 1.1 - Circuit Schematic

Using the information on the schematic found on TinkerCAD, label each of the electronic components below.



Component A

Component B

Activity 1.2 - Sketch Simulation

Describe what occurs when you press the 'Run Simulation' button for the TinkerCAD circuit.

Activity 1.3 - Arduino Sketch

Below demonstrates the sketch for the above schematic. **Identify** each section of the sketch and **explain** each line of code within the different sections.

```
** Arduino Lesson 1: Blink**
```

```
int led = 13;
```

```
void setup() {  
  pinMode(led, OUTPUT);  
}
```

```
void loop() {  
  digitalWrite(led, HIGH);  
  delay(1000);  
  digitalWrite(led, LOW);  
  delay(1000);  
}
```

ARDUINO LESSON 1 - BLINK

ACHIEVED

Activity 1.4 - IPO Chart

Complete the IPO chart provided by correctly identifying the inputs, processes and outputs for the blink circuit.



Activity 1.5 - Flowchart

Construct a flow chart for the blink sketch demonstrated on the previous page.

Activity 1.6 - Challenge

Attempt to make the LED blink at a different rhythm. How did you change the rhythm of the LED? What line of code within the sketch did you change?

Applying Arduino to the Years 7+8 Syllabus

ARDUINO LESSON 1 - BLINK

■ ACHIEVED

Activity 1.7 - Challenge

Attempt to make the LED blink an SOS in the Morse code: short, short, short, long, long, long, short, short, short - pause. Write the sketch below to demonstrate your understanding.

Activity 1.8 - Challenge

Attempt to make two LED's flash on and off at different times. Write the sketch below to demonstrate your understanding.

ARDUINO LESSON 1 - BLINK

■ ACHIEVED

Activity 1.9 - Challenge

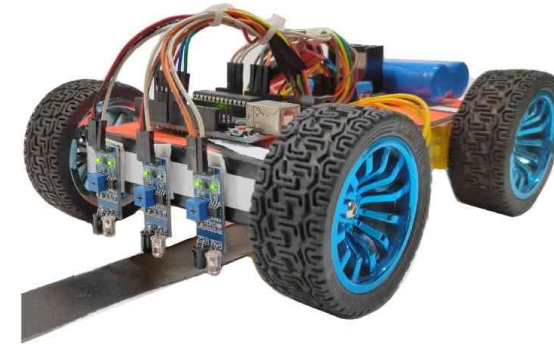
Using four LED's connected to the digital pins **9**, **10**, **11** and **12**, create a sketch to make the LED's 'chase' each other. Write the pseudocode and sketch below to demonstrate your understanding.

Step 1	Give each LED a value corresponding to the pin its connected to.
Step 2	Tell Arduino which pins are Outputs/ inputs.
Step 3	create loop
Step 4	Send High signal to LED 1 for 300 milliseconds.
Step 5	Send LOW Signal to LED 1 for 0 milliseconds.
Step 6	Repeat steps 4-5 for LED 2,3,4
Step 7	
Step 8	

Applying Arduino to Years 9+10 Syllabus

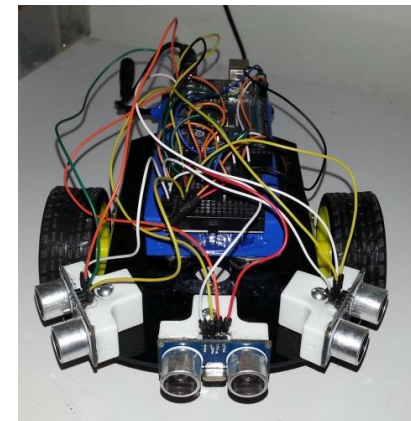
Stage 5 Computing:

https://docs.google.com/presentation/d/1094c9Y_u24OeFm9rNFKy7GT9Lr-Fk9PJGqgJsPe995w/edit?usp=sharing



Stage 5 Engineering:

https://docs.google.com/presentation/d/13JB0L2f02_ofVs_jjX-z5Wz1PUnNZTaT0R-b00N7LvA/edit?usp=sharing

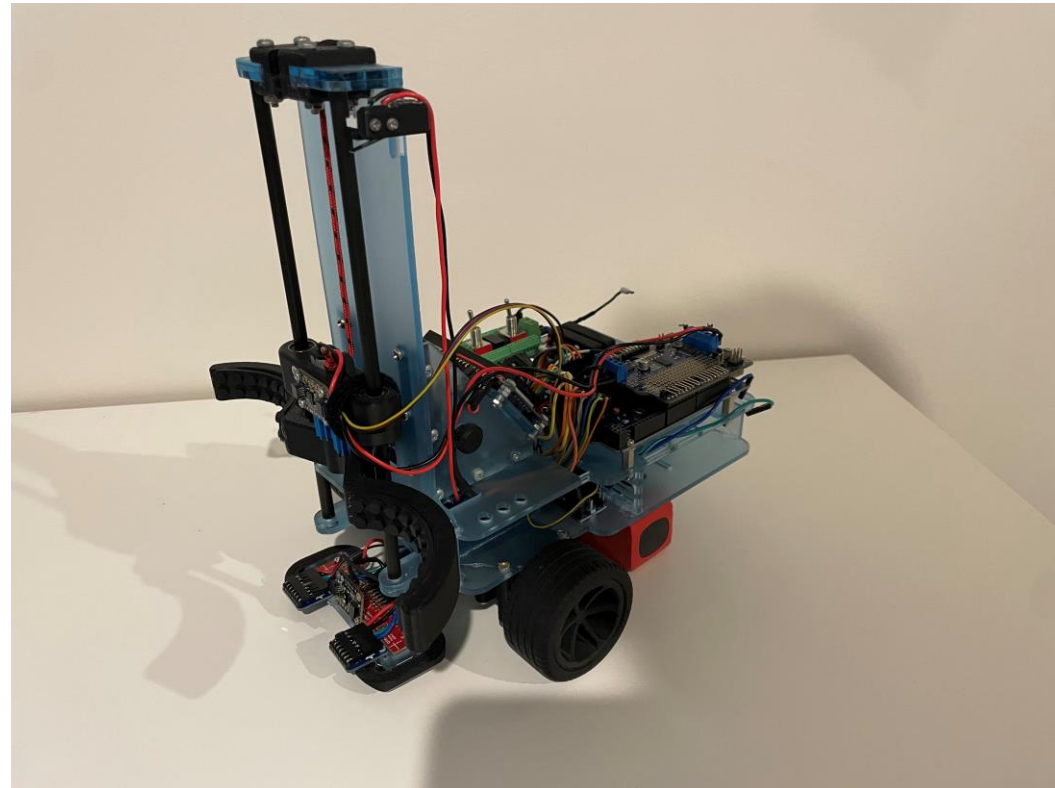
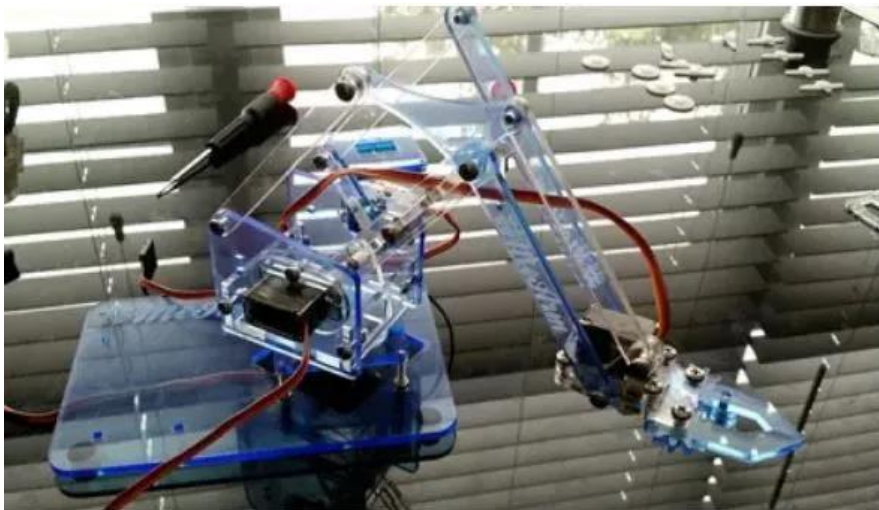


RoboCup
JUNIOR
A U S T R A L I A

Applying Arduino to Years 11+12 Syllabus

IPT/ Engineering:

<https://drive.google.com/file/d/14ieaxbj8IRmJdAD7CrLe1vTnwEuluiZU/view?usp=sharing>



What questions do you have?



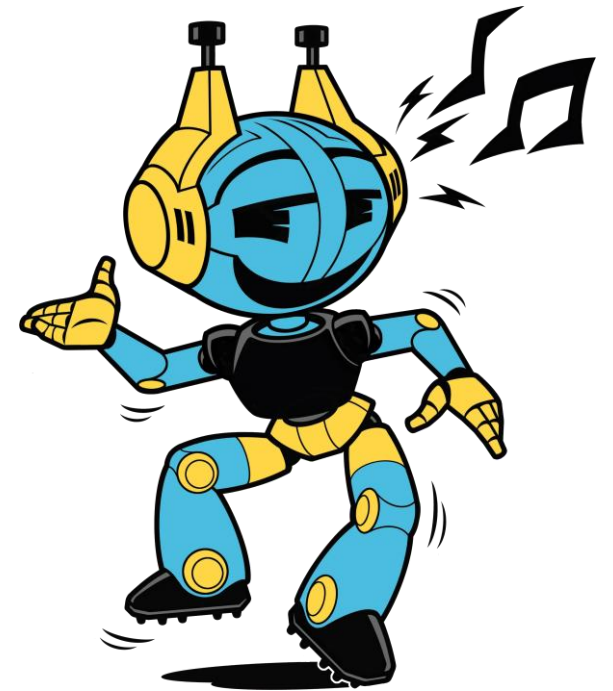


While you're waiting for us to get started, why not checkout our website?
<https://robocupjunior.org.au>

Arduino in OnStage

Thank you for joining us for our 2023 Online Arduino Workshop

Margaux Edwards – RCJA OnStage Chair
margaux.edwards@robocupjunior.org.au



What is OnStage?

OnStage is the culmination of creativity, design and integration of technologies to create a robotic performance that is entertaining for students and audiences alike

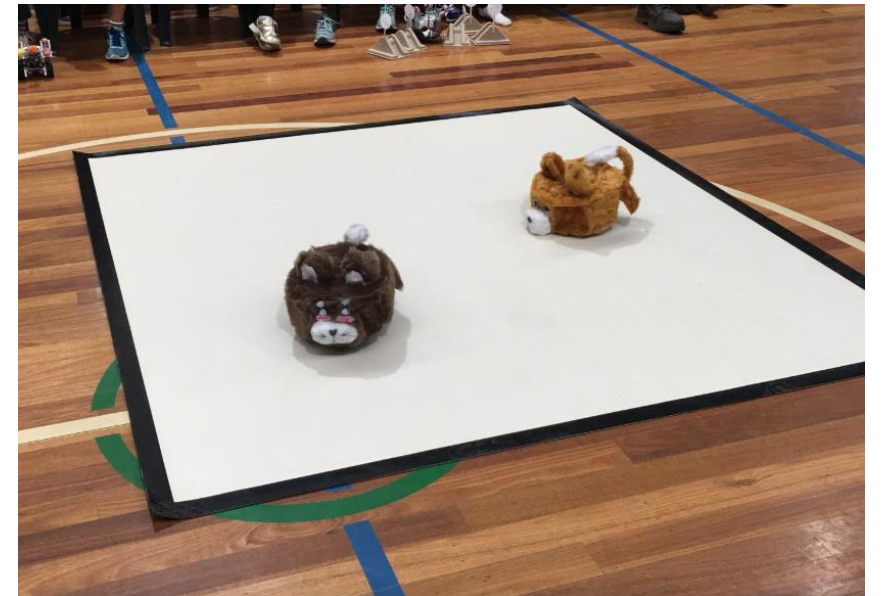
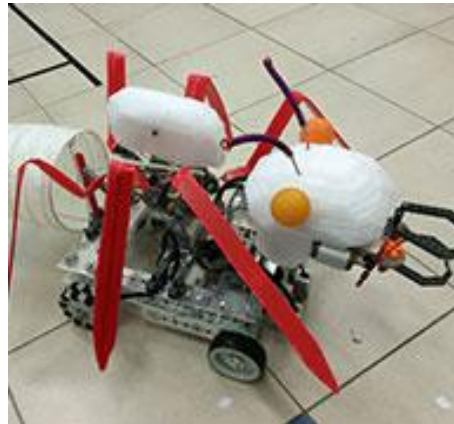


The Challenge: To create a 1-2 minute performance showcasing costumed or themed robots to a piece of music



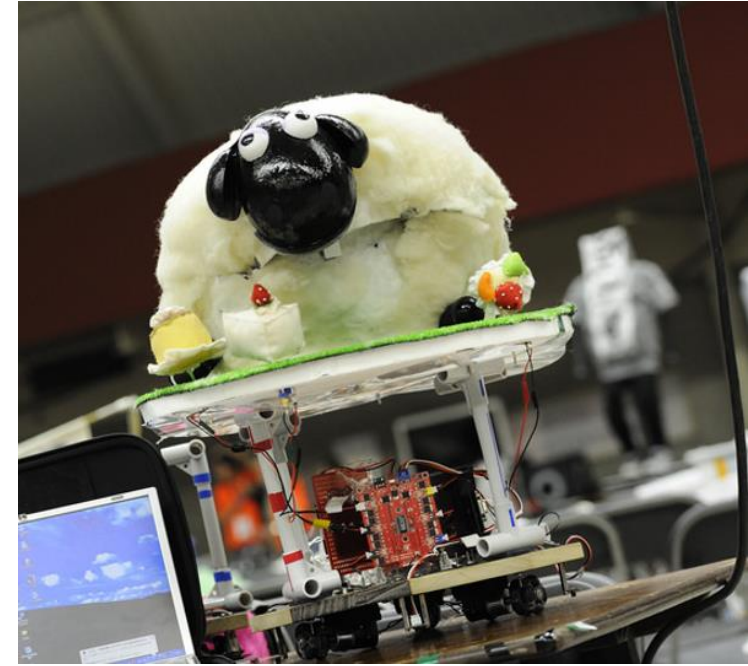
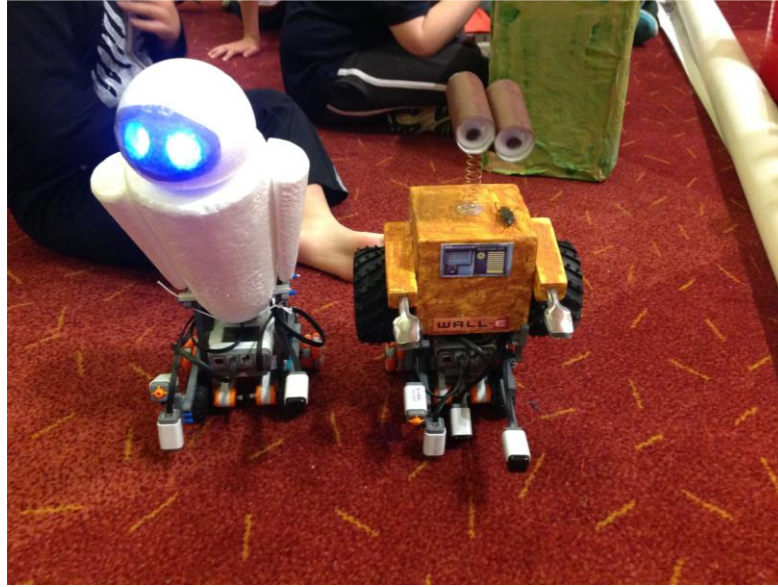
Choosing a Theme: Dance

- Dance is dictated by the music choice because the robots dance in time with the chosen piece.
- Many teams choose to create their own choreographed dance to perform, in conjunction with their robots' dance.
- *However, minimal points are given for the team's performance; remember the robot's performance is worth more!*



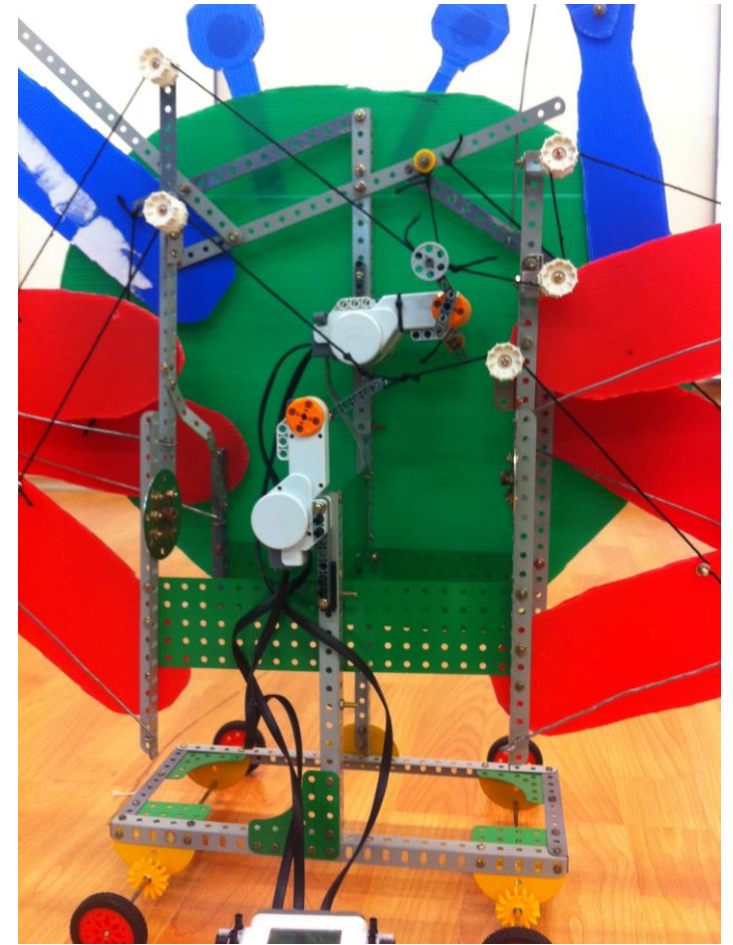
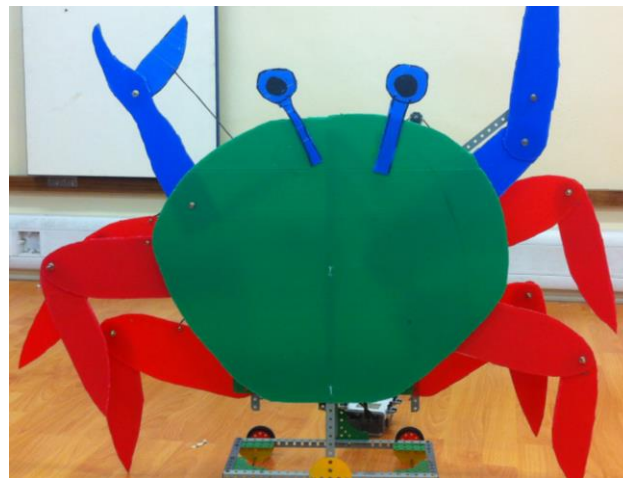
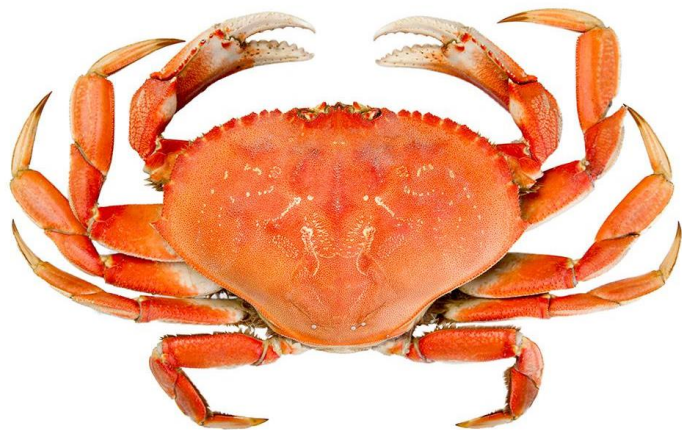
Choosing a Theme: Theatre

- A theatre performance works around a chosen theme.
- The robots interact together to portray an idea or concept through characters, music and plot.
- Previous themes have included performances from scenes in films, TV shows or popular culture.



Research and Plan some robots!

- Has someone made a robot version before?
- What moving parts will your robot have?
 - Moving head, arms, legs?
- How does your robot detect its environment?
 - Sight, touch, distance



So, how can we use Custom
Robots in OnStage?



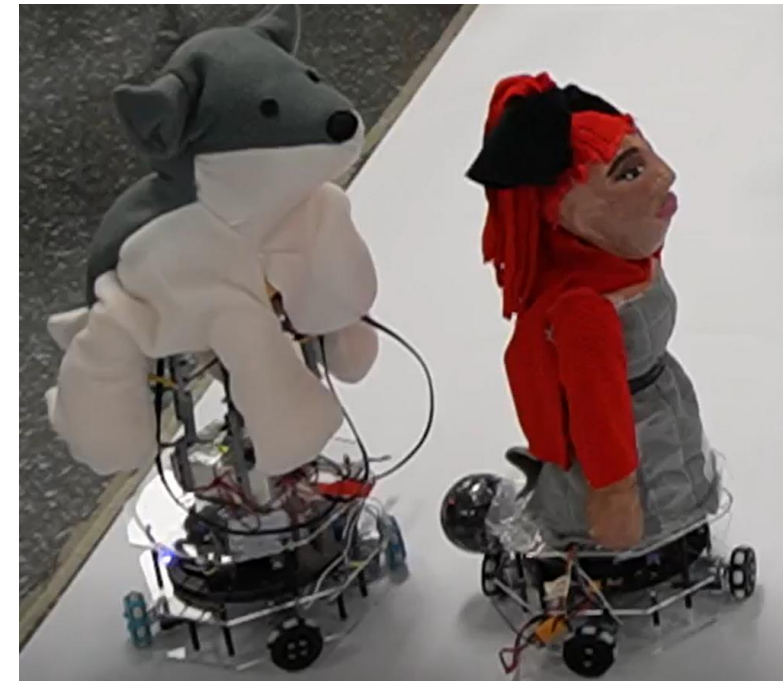
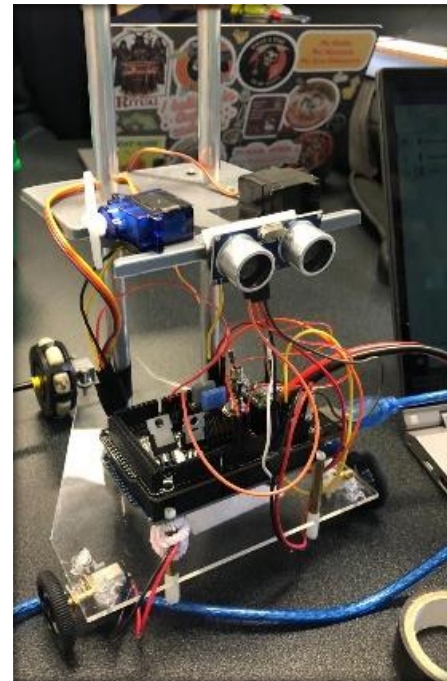


Custom robots can be designed and programmed specifically for use in RoboCup OnStage competitions. These robots can be tailored to perform specific tasks related to the performing arts, such as dancing, acting, or singing, and can be customized to fit the particular requirements of a given routine or act.

For example, a custom robot designed for a dance routine may have specialized joints and motors that allow for fluid and graceful movements, while a robot designed for a singing performance may be equipped with sensors and actuators that enable it to mimic human facial expressions and vocalizations.

Custom robots can also be used to enhance the overall experience for the audience, by incorporating advanced lighting, sound, and projection systems. These systems can be synchronized with the robot's movements to create a more immersive and engaging performance.

Overall, custom robots offer a unique opportunity to push the boundaries of what is possible in the performing arts, by incorporating cutting-edge technology and innovation into live performances.



SOFTWARE

SENSORS

- ↳ LOCALIZATION
- ↳ VISUAL RECOGNITION
- ↳ AUDIO RECOGNITION
- ↳ TACTILE FEEDBACK

ONSTAGE

ELECTRICAL

- ↳ PCBs
- ↳ SERVO MOTORS
- ↳ DC MOTORS
- ↳ LED ARRAYS

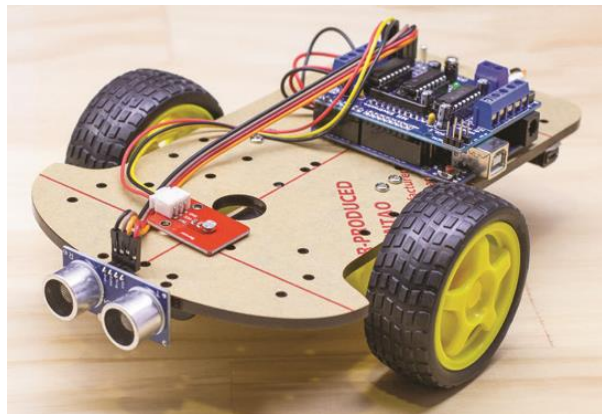
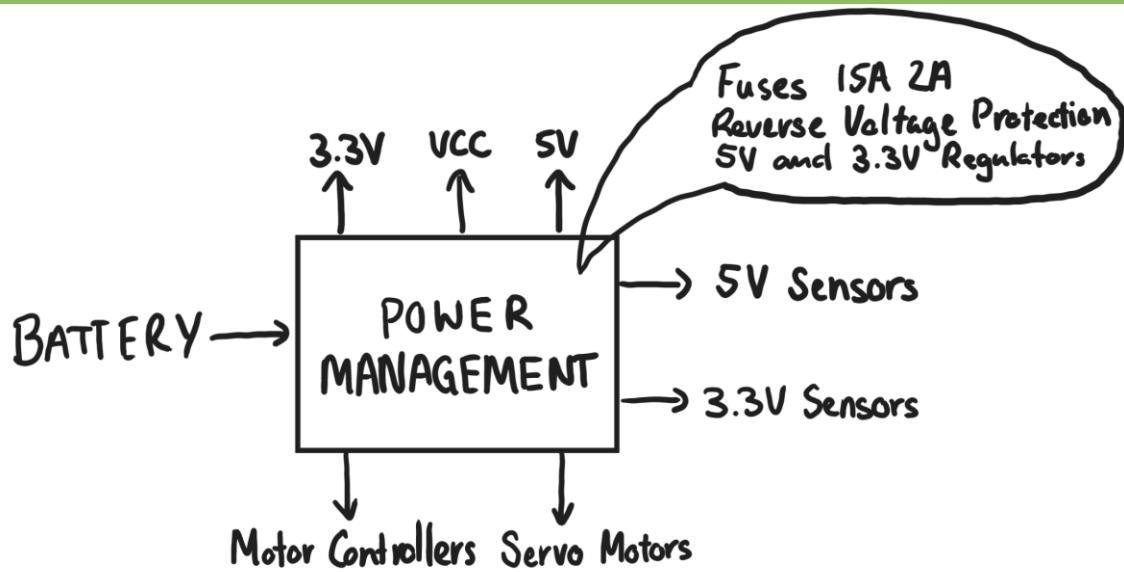
COMMUNICATION

- ↳ BLUETOOTH
- ↳ SERIAL
- ↳ ROBOT | ROBOT
- ↳ HUMAN | ROBOT

MECHANICAL

- ↳ 3D PRINTING
- ↳ LASER CUTTING
- ↳ STRUCTAL ASSEMBLY

Electrical



Motor drivers: <https://www.cytron.io/c-motor-and-motor-driver/c-motor-driver/p-3amp-4v-16v-dc-motor-driver-2-channels>

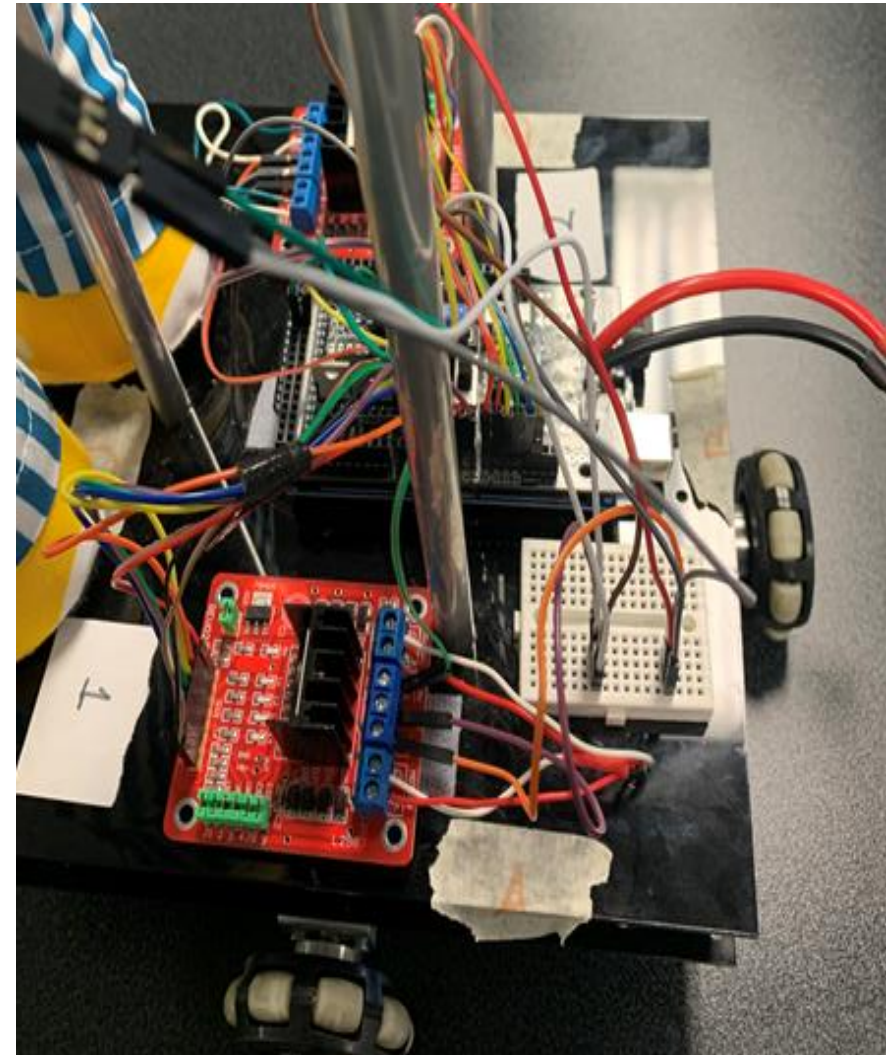


Arduino Compatible DC Voltage Regulator

CAT.NO: XC4514

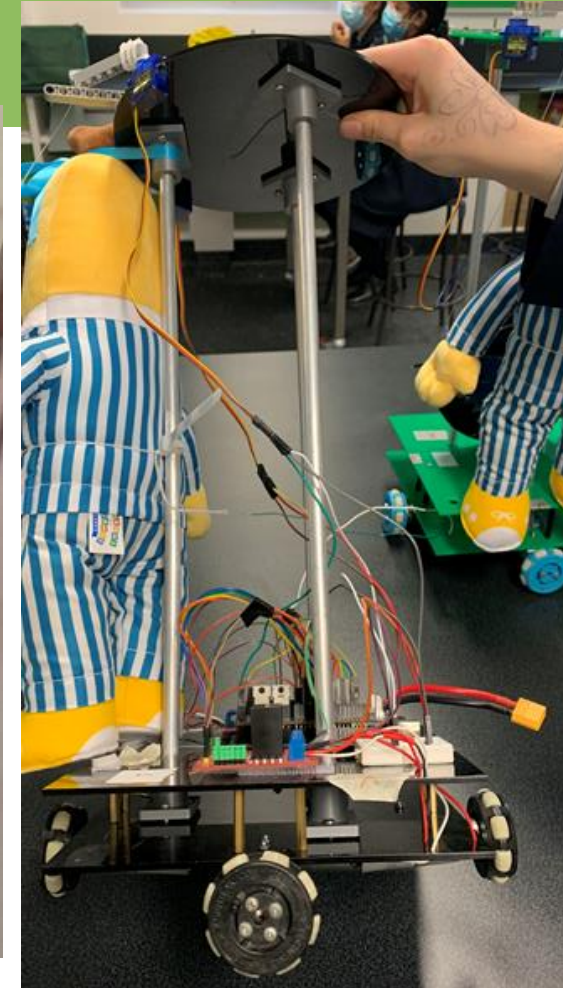
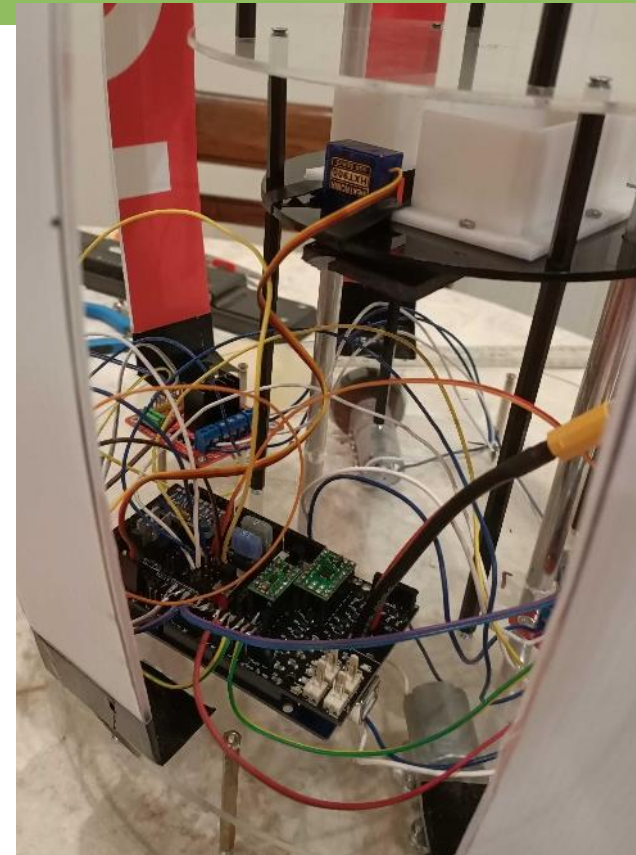
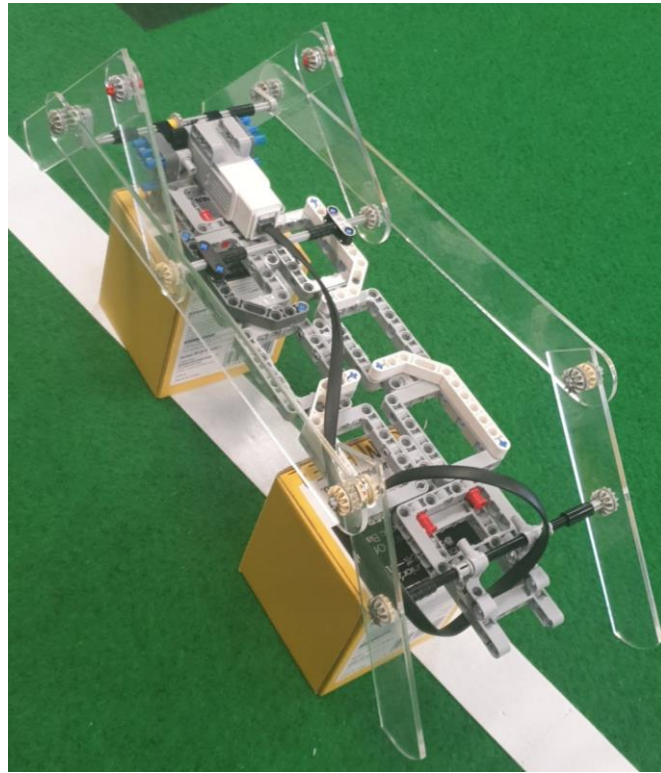
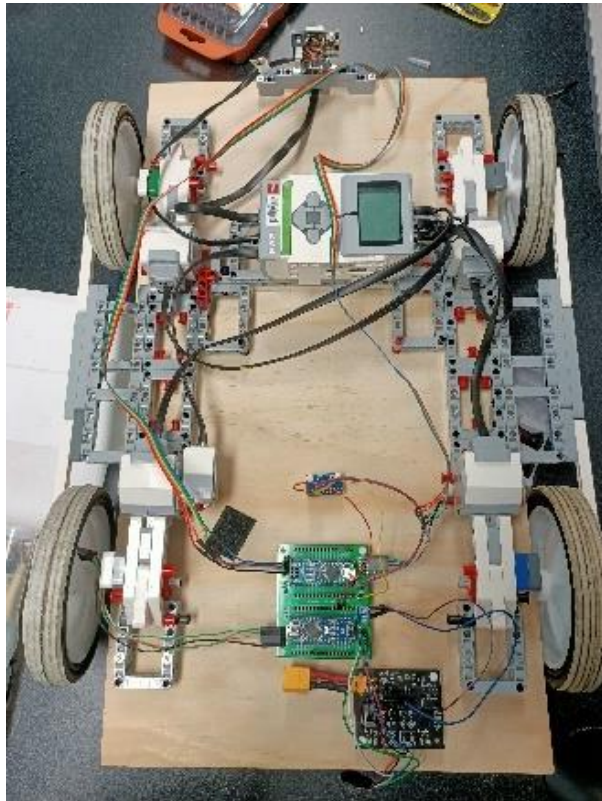
★ ADD TO WISHLIST

✉ NOTIFY ME WHEN ON SPECIAL



<https://www.jaycar.com.au/diy-dodging-robot>

Mechanical



Hybrid Design – Lego Structure with additional custom components

Custom Design – Fully Custom using laser cut, 3D printed and assembled components

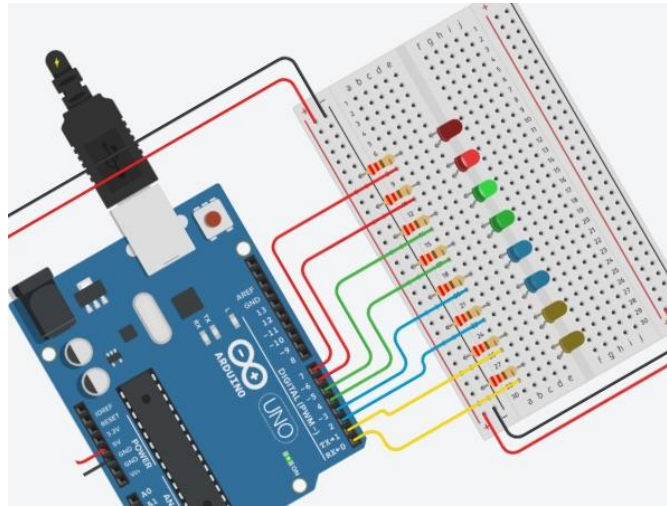
Sensors

BEGINNER SENSORS

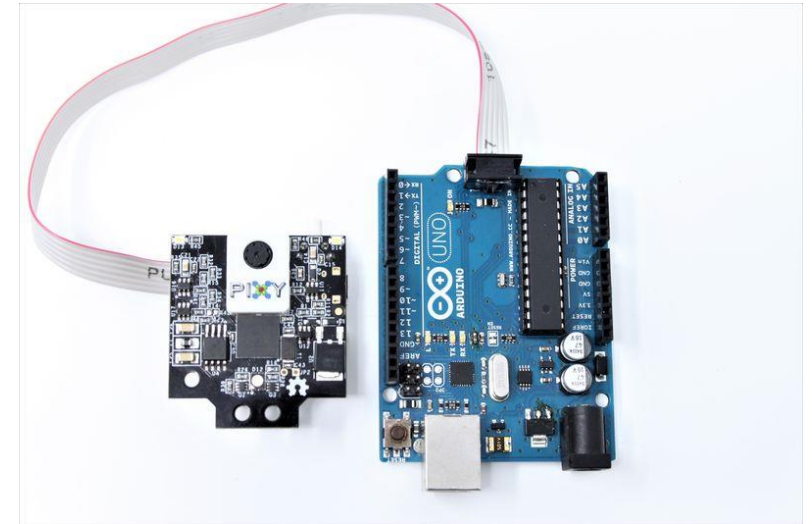
- ↳ Ultrasonic
- ↳ Light Sensor
- ↳ Colour Sensor
- ↳ Cameras (Module)
- ↳ IMU

ADVANCED SENSORS

- ↳ LIDAR
- ↳ Laser Range Finder
- ↳ Cameras



<https://www.tinkercad.com/circuits>



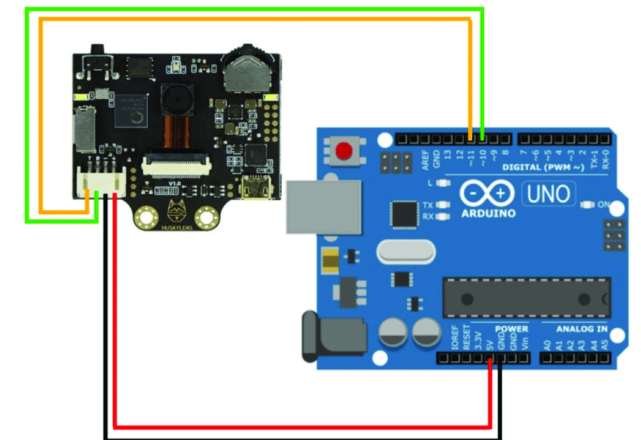
Pixylens – Arduino, EV3, Spike*



37 in 1 Sensor Kit for Arduino

CAT.NO: XC4288

With 37 different sensors and modules, this kit covers just about every input and output you can poke a soldering iron at.



HuskyLens – Arduino, EV3*, Spike*

Sensors: Computer Vision vs Visual Learning



<https://learnopencv.com/object-tracking-using-opencv-cpp-python/>

New Project

 Open an existing project from Drive.

 Open an existing project from a file.

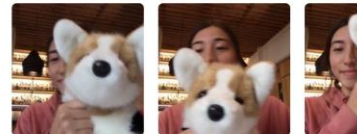
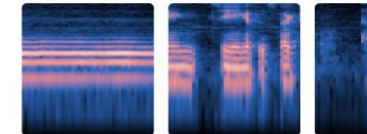


Image Project

Teach based on images, from files or your webcam.



Audio Project

Teach based on one-second-long sounds, from files or your microphone.



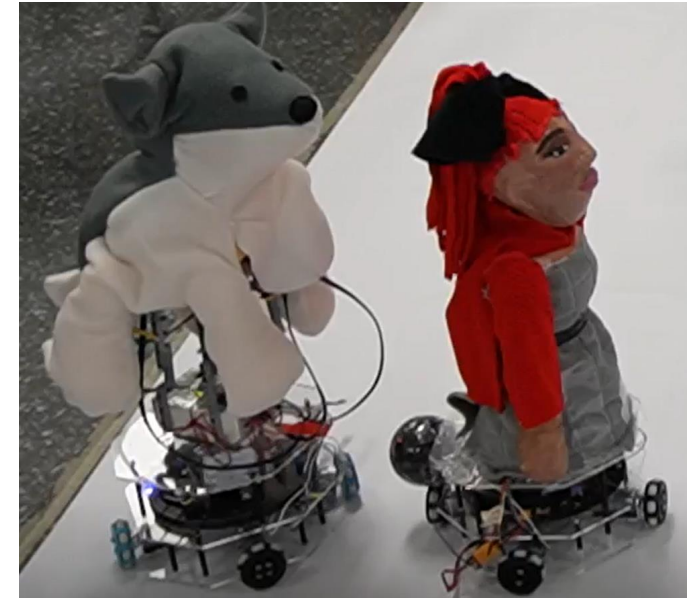
Pose Project

Teach based on images, from files or your webcam.

<https://teachablemachine.withgoogle.com/train>

Communication

- Can be difficult to setup but there are many tutorials online:
- IR Seekers – Make your own soccer ball!
 - <https://core-electronics.com.au/guides/arduino/infrared-sensors-arduino/>
- Bluetooth Communication – Communicate between robots
 - <https://core-electronics.com.au/guides/how-to-use-bluetooth-modules/>
- Serial Communication – Communicate between controllers
 - <https://core-electronics.com.au/guides/arduino/serial-communications-arduino-uno/>
 - <https://core-electronics.com.au/guides/arduino/i2c-with-arduino/>



Arduino Compatible RFID Read and Write Module

CAT.NO: XC4506

Read and write MiFare-Type RFID cards.

• SPI Communications Protocol

• 2 Tags Included

Software

- Get simple features working first. Once you discover your robot's base functionality, a performance will come from there.
- If you're getting started, using block-to-text solutions
 - Tinkercad Circuits (Arduino, Microbits, other microcontrollers)
 - Microsoft MakeCode (Microbits)
- Use code examples & tutorials to get started
 - Lots of hobbyists release tutorials, official sellers usually link tutorials too
- Suggest using GitHub and Visual Studio Code or the Arduino app for programming

How to get started (with a little help from our AI)

Designing a custom robot for RoboCup OnStage can be a complex and challenging task, but with careful planning and preparation, it is possible to create a successful robot for competition. Here are some steps you can take to get started:

1. **Understand the rules and requirements of the competition:** Before beginning your robot design, it is important to thoroughly read and understand the rules and requirements of the RoboCup OnStage competition. This will help ensure that your robot meets all the necessary criteria for entry and can perform the required tasks.
2. **Define the purpose and goals of your robot:** Determine the specific purpose and goals of your robot in the context of the competition. Consider what type of performance or routine your robot will be required to perform, and what skills and capabilities it will need to have to succeed.
3. **Choose your robot's components:** Select the components and materials that will be used to build your robot, such as motors, sensors, actuators, and structural elements. Choose components that are appropriate for the specific tasks and requirements of your robot, and ensure that they are compatible with each other.
4. **Design the robot's mechanical and electrical systems:** Use CAD software or other design tools to create detailed schematics and 3D models of your robot's mechanical and electrical systems. Consider factors such as weight, balance, stability, and durability when designing your robot's structure.
5. **Program your robot's software:** Develop the software and programming necessary to control your robot's movements and actions. This may involve using programming languages such as C++, Python, or Java, and may require developing algorithms for image processing, speech recognition, or other advanced capabilities.
6. **Test and refine your robot:** Once your robot is built and programmed, test it thoroughly in a controlled environment to ensure that it can perform the required tasks and meet the competition criteria. Use this testing phase to identify any issues or weaknesses in your robot's design, and make any necessary refinements or adjustments.
7. **Overall, designing a custom robot for RoboCup OnStage requires a combination of technical expertise, creativity, and problem-solving skills.** By following these steps and staying focused on your goals, you can create a robot that is both innovative and competitive in the world of performing arts robotics.

OnStage Examples:

- Teams representing Australia with Custom Robotics
 - 2019 Sydney
 - 2021 Worldwide
 - 2022 Bangkok



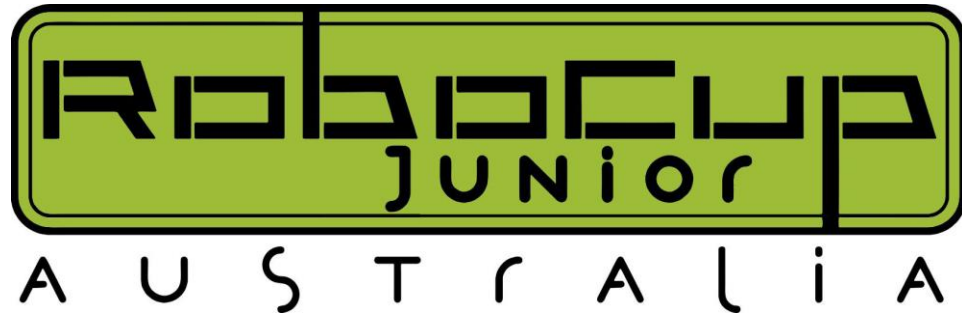
<https://www.youtube.com/@rcjonstage>



<https://youtu.be/EdDJ77uDwWQ>

What questions do you have?





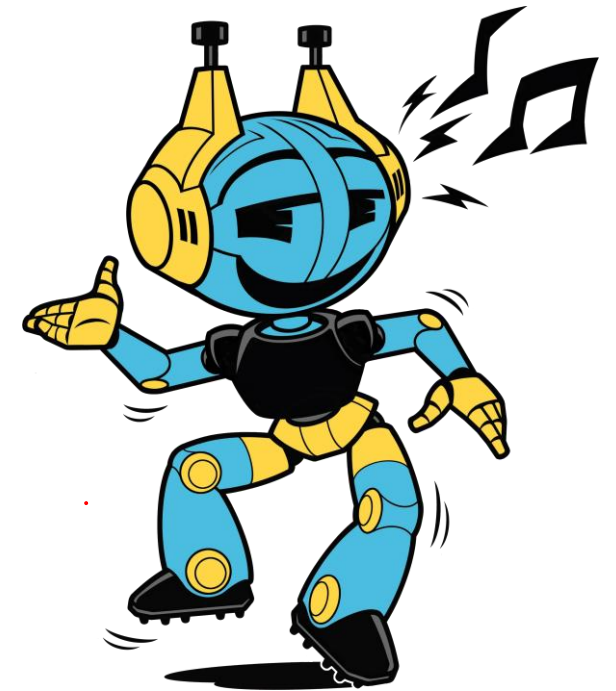
While you're waiting for us to get started, why not checkout our website?
<https://robocupjunior.org.au>

Arduino in Rescue Line

Thank you for joining us for our 2023 Online Arduino Workshop

Liam Whitehouse – RCJV Rescue Line Coordinator
liam.whitehouse@robocupjunior.org.au

Zac McWilliam – RCJA & RCJV Digital Platforms Coordinator
zac.mcwilliam@robocupjunior.org.au



Interruption Policy

Please interrupt at any
time as often as you like*

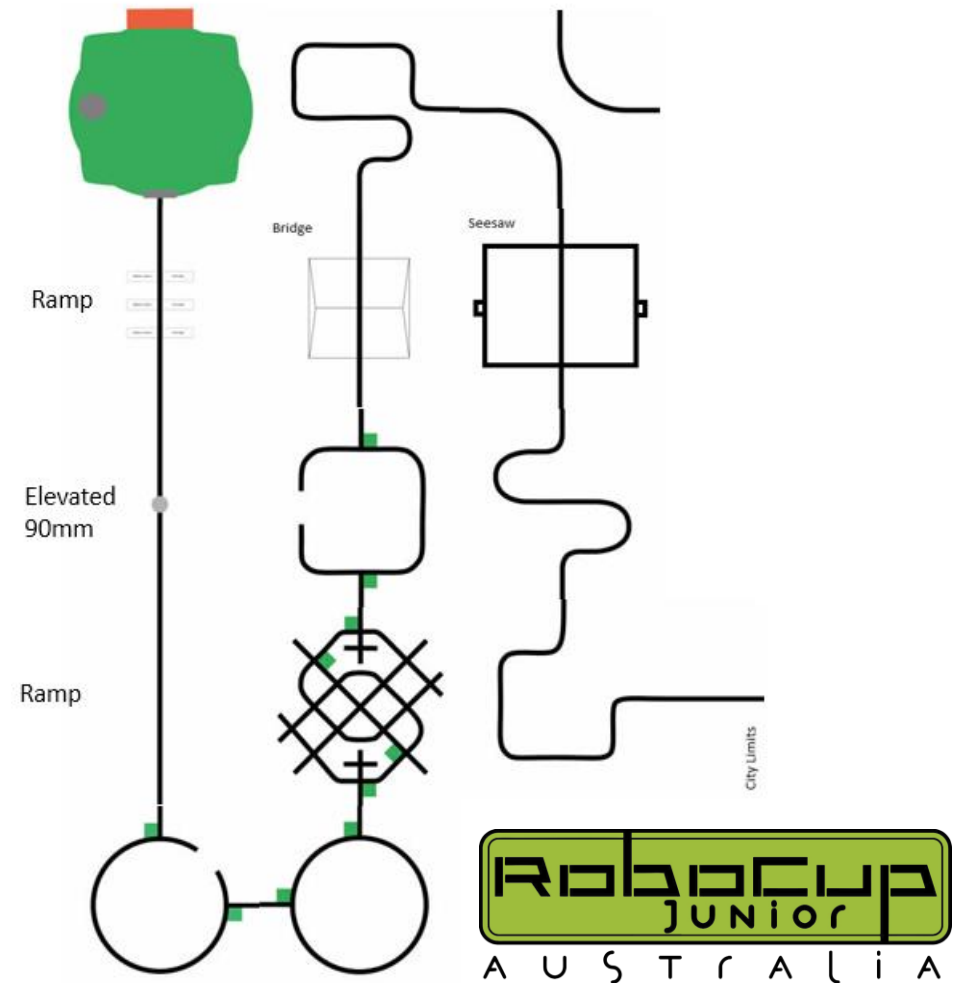
This is a Workshop *not* a Lecture

* Yes, like that annoying kid in class today



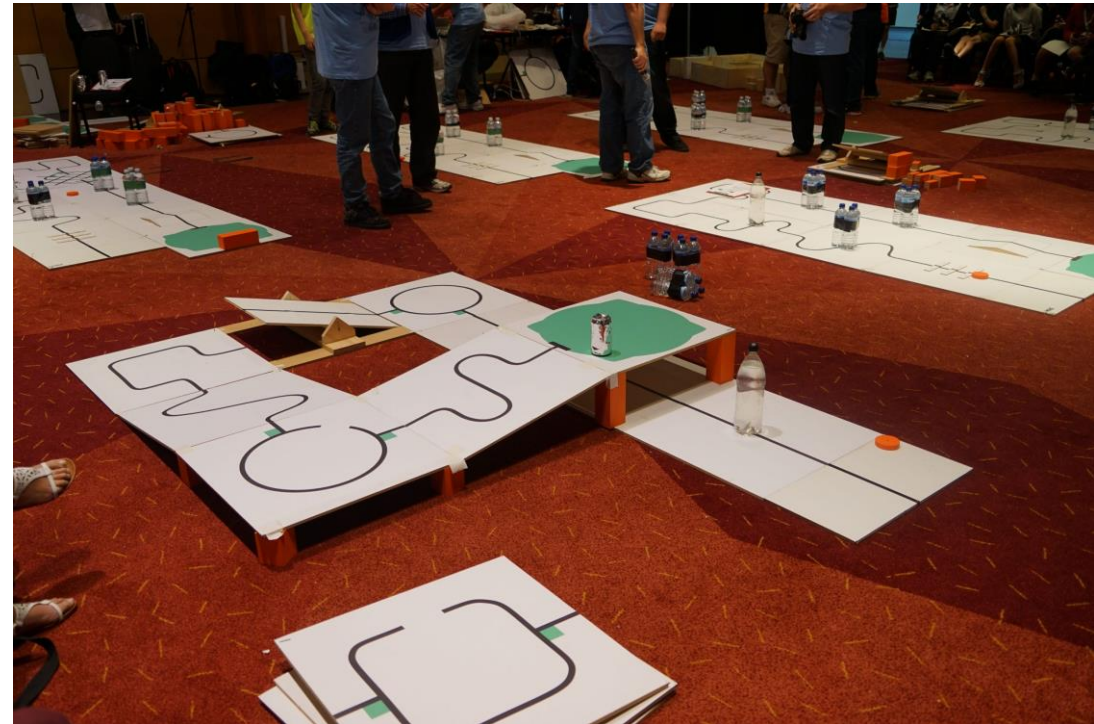
Rescue – The Challenge

- Robots compete by following a winding line on a series of tiles to a designated rescue area.
- On the way the robot could encounter obstacles, bridges and short cut opportunities that will challenge the most intrepid programmer.
- After negotiating the randomly selected path, the robot arrives at a green coloured area which indicates a chemical spill.
- While the clock is still ticking the robot must find “the victim” before pushing them out of the chemical spill to safety.



Rescue – The Divisions

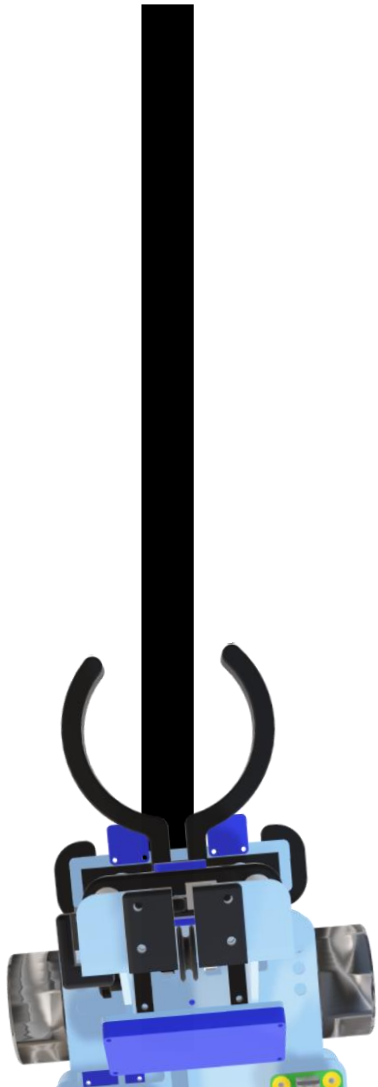
- Riley Rover Rescue
 - The Basics (Line Following, Obstacle & End Zone)
- Primary Rescue
 - Only for Primary School Students
 - Adds Green turns, Speed Bumps
- Secondary Rescue
 - Only for Secondary School Students
 - Adds requirement for control of Silver Can
- Open Rescue
 - More Complex Tiles (Gridlock), Lifting and Placing Can on Orange Block



Rescue – Sensor Recommendations

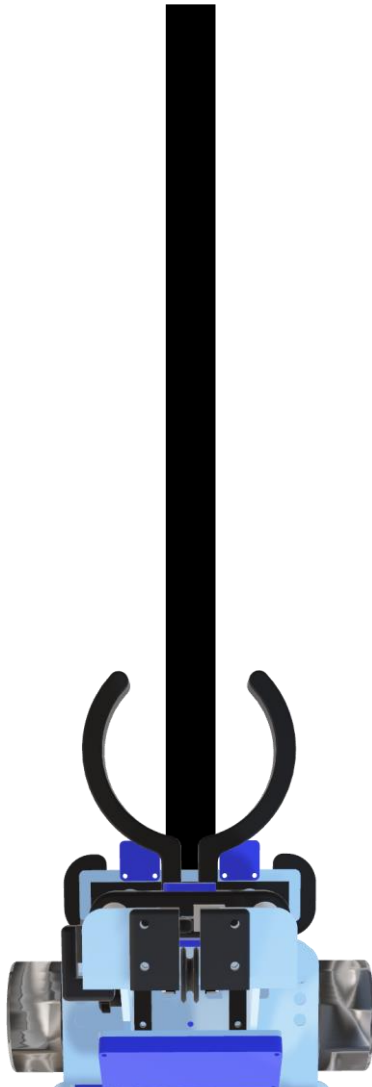
- Infrared Line Array
 - To follow the line
- Ultrasonic Sensor & Time of Flight Sensor
 - To find distances to objects (such as Can's, Obstacles, Orange Block)
- Color Sensors
 - To identify the color of the tile (Black, White, Green, Special Tile Color)

Rescue – Line Following



- Simple Method of Line Following, which is a bit jerky, but gets the job done. Uses less sensors so doesn't require many ports
- If double White
 - Go Straight
- If Left Black and Right White
 - Go Left
- If Right Black and Left White
 - Go Right

Rescue – Line Following



- PID (Proportional Integral Derivative)
- First Calculate a position value using the line array (eg 0 in the middle, -10000 at one end and 10000 at the other)
- Requires multiplication of constants to each of the components (Proportional, Integral, Derivative)
- For Arduino there are PID algorithms already out there so read their documentation for use.
- Essentially the output from the PID is used as a multiplier on the wheel speed resulting in a smoother movement.
- More complicated and needs understanding on more complicated mathematics but can be more effective.

Rescue – Design Choices

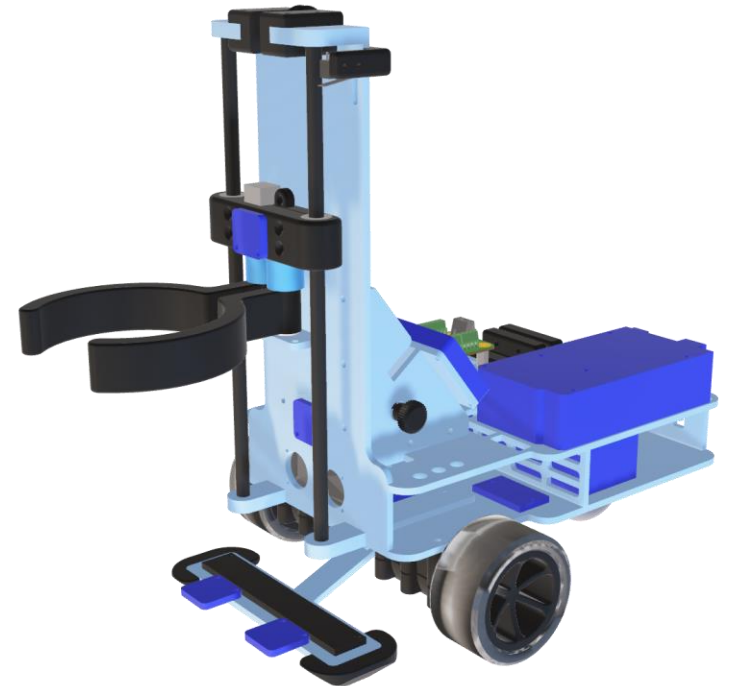
- For tiles such as speed bumps, see saw, a ramp up to another height.
- All these different circumstances result in different sensor values for the sensors that are facing down since the height is different. So maybe a floating sensor array is a good idea. But you don't need it.
- A LCD Panel is a nice thing to have can tell you what the values are of certain variables for debugging, and you can make cool visuals for when the robot turns, etc.

Rescue – Obstacle (Water Tower) + Green Turns

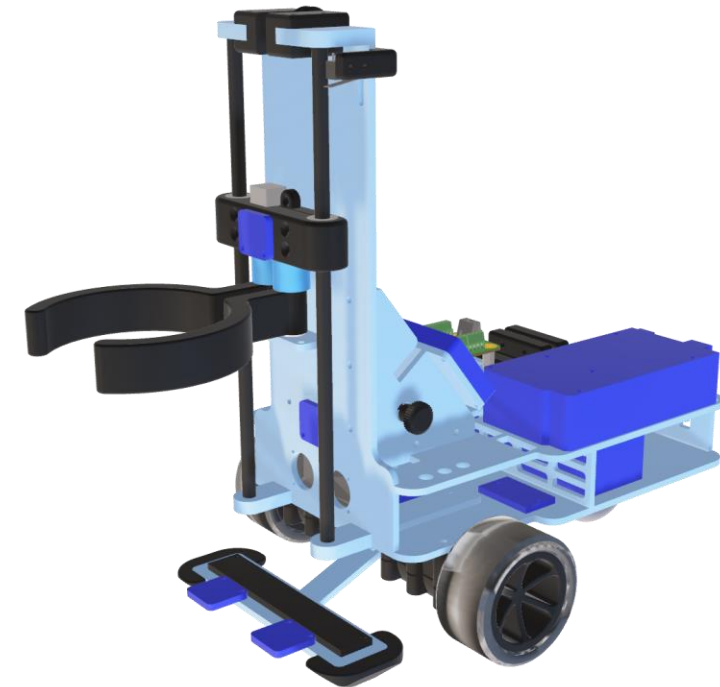
- Work Very Similar to LEGO based systems.
- Use the Color Sensors to check for Green
- Use the Ultrasonic with a combination (security check) of the TOF's to ensure you actually have an Obstacle in front of the robot.
- Micros Switches (essentially touch sensors) also work for Obstacle
- Make sure that the robot makes it back to the tile before the Obstacle Tile ends (otherwise you don't get points)

Rescue – End Zone

- Custom Bots give you significantly more flexibility in how you achieve the end zone.
- You can create custom lifting systems or use another method of your own. (There are some much better ones out there)
- Lifting is only needed for Open Rescue, but similar systems can be used for secondary in order to gain control over the Can.

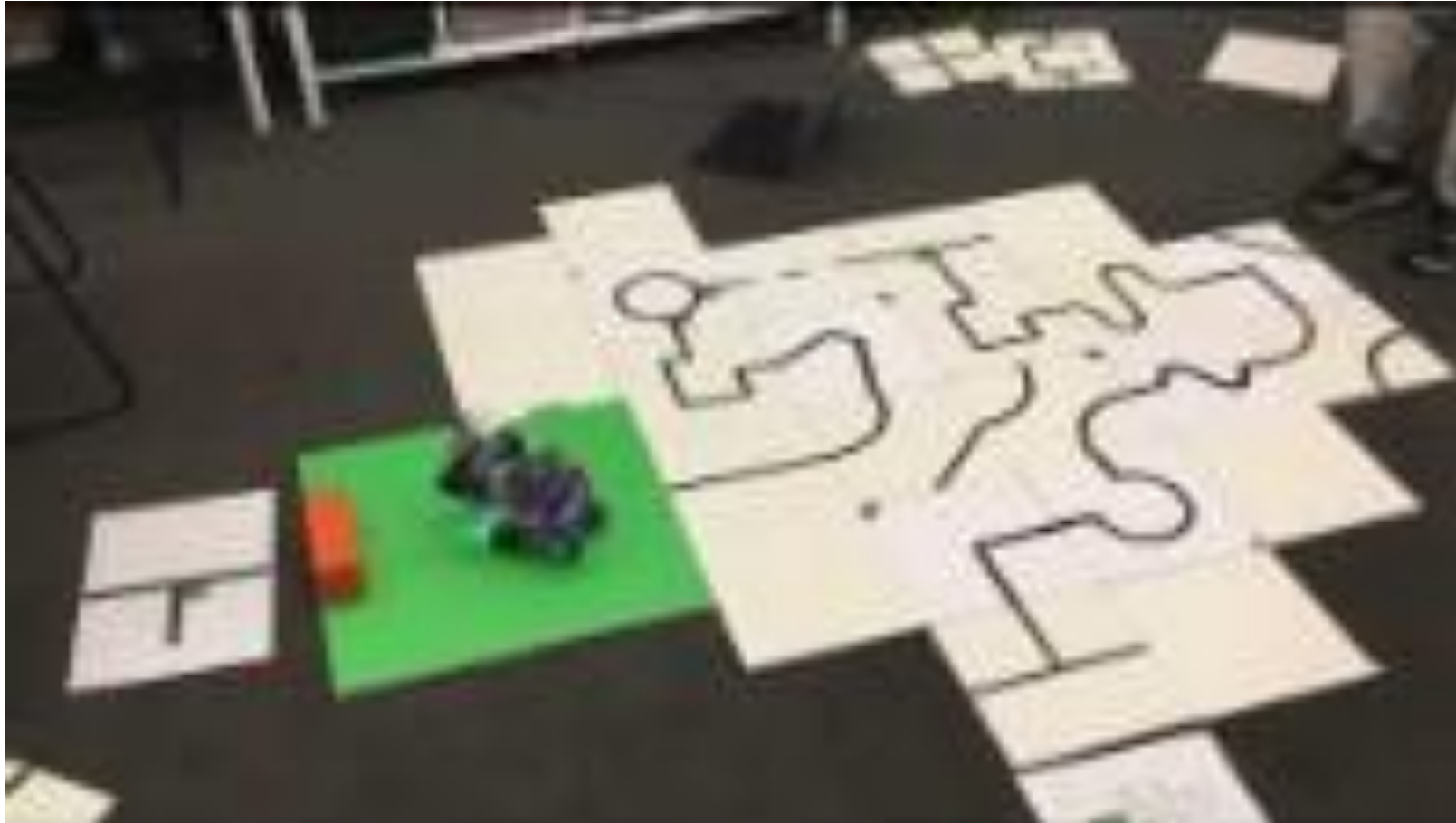


Rescue – Some Videos



RoboCup
JUNIOR
A U S T R A L I A

Rescue – Some Videos



What questions do you have?

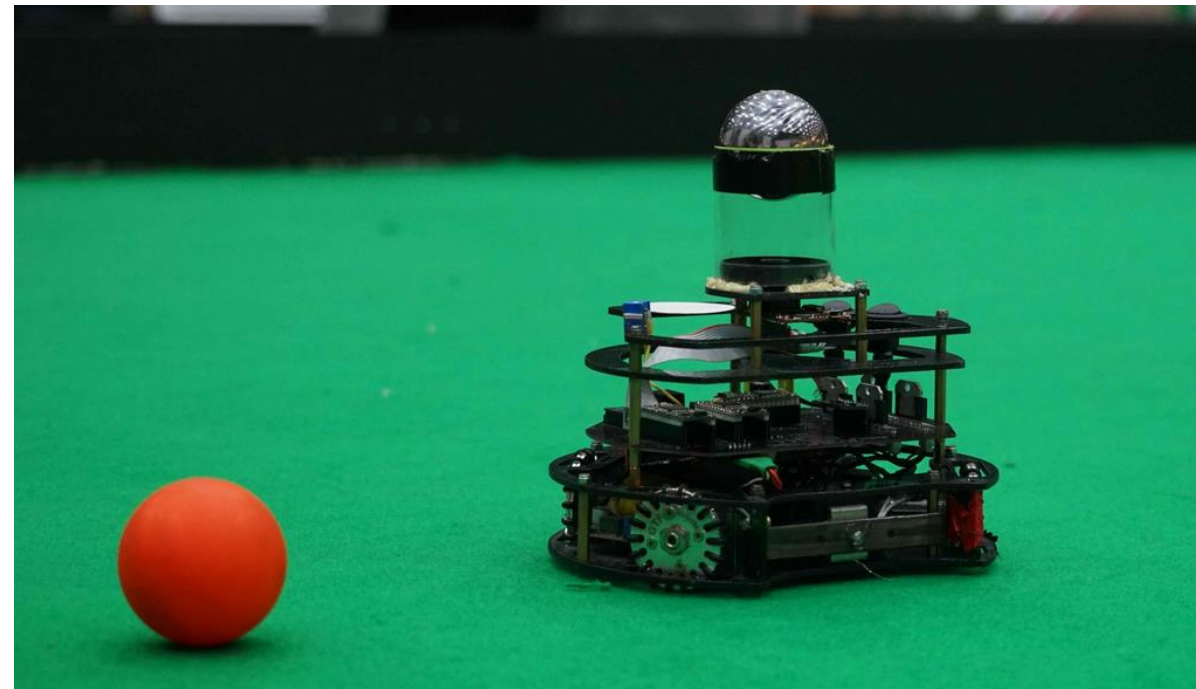




While you're waiting for us to get started, why not checkout our website?
<https://robocupjunior.org.au>

Arduino in Soccer

William Plummer – RCJQ State Chair
william.plummer@robocupjunior.org.au



Soccer League Overview

The RoboCup Junior Soccer League in Australia is divided into 3 leagues:

- Standard (Lego Only)
[Beginner – Years 5-10]
- Lightweight (Infrared Ball, 1.1 Kg weight limit)
[Intermediate – Years 8+]
- Open (Orange passive ball, 2.5 Kg weight limit)
[Advanced – Years 10+]
- 2 vs. 2 format



Soccer Custom Overview



- In Standard, many robots utilise differential drive (2-wheel tank steer)
- Omni-drive system is far easier to teach and understand for teachers and students alike
- 4 Motor / 4 Wheel Omni-drive can operate with just base vectors, acting as a steppingstone for students
- One Robot at a time when building!

Soccer – Subsystem Synergy

The Omni drive system integration with 360 sensors is one of the most beautiful synergies developed throughout robotics.

- This incredibly complex problem contains:
 - Orbital mechanics
 - 360 Data tracking
 - Sensor fusion
 - Vector dynamics
 - Environment localisation and mapping

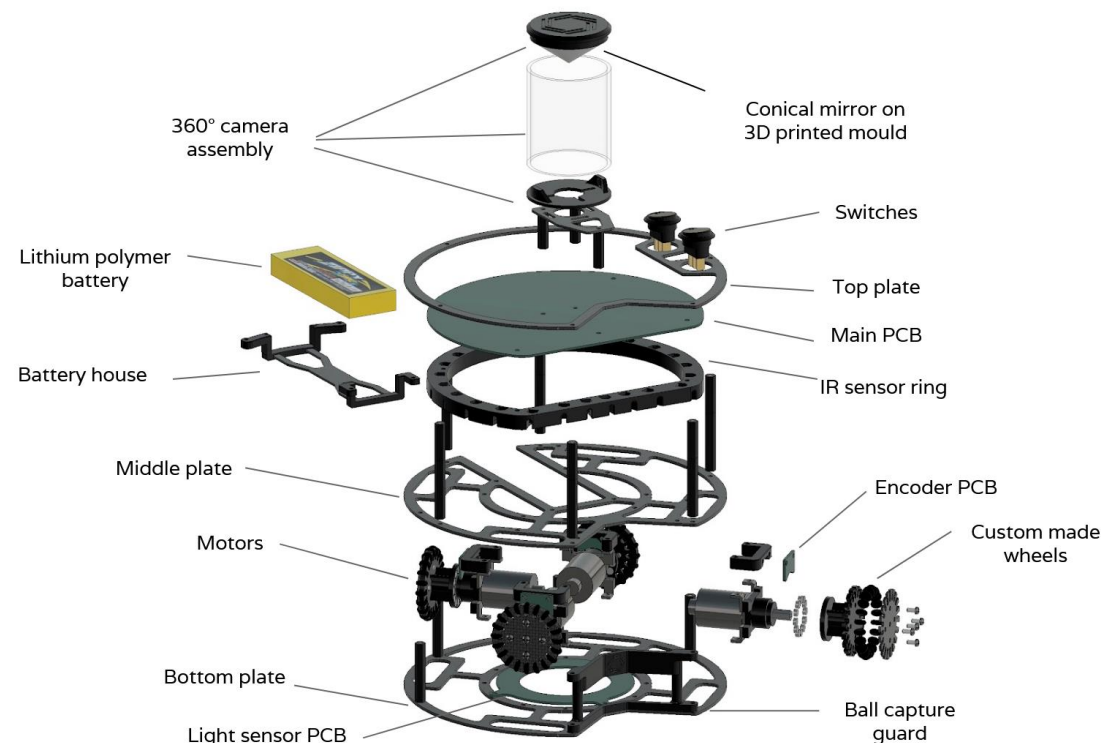
This has been so heavily refined over 20 years, that the code can be written in < 20 lines of code (Normal length too!)



Soccer – Hardware I

Recommended 3 Layer Topology:

- Top Layer
 - Power switch, user debug (Screens / LEDs), Handle (Zip ties work fine!), Ultrasonic sensors
- Middle Layer
 - Arduino processor, Motor controllers, IR sensor ring, Camera, Power regulation
- Bottom Layer
 - Motors, Battery, Light sensors
- Layers made with laser cut / wooden plates
- Layers spaced by standoffs (Brass or Nylon), or 3D printed spacers



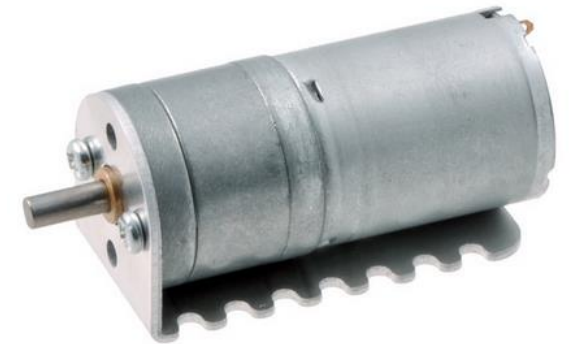
Soccer – Hardware II

- Motors

- Lightweight recommended >5 Watt Motors
- Open recommended >8 Watt Motors
- Any speed over ~500 RPM is useless, Torque is better
- <https://www.pololu.com/product/3205>
- <https://www.pololu.com/product/2676>

- Battery

- Recommended 3s LiPos for ~12V VCC system (Not Li-Ion 18650 cells)
- Most high-power parts accept 12V +/- 1 Volt
- Sizing battery capacity needed (C ratings)
- 4s 14.8V battery potentially for open, more difficult to integrate
- >2500mAh is recommend per 10-minute game (Finals are 10-minute halves)



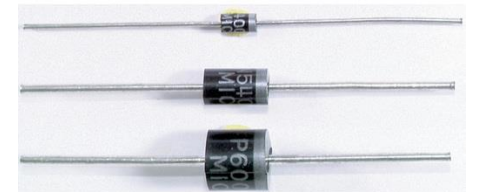
Soccer – Hardware III

- Wheels
 - Standard omni-wheels are perfectly sufficient and don't need to be complex
 - These can be mounted using Pololu shaft couplings
 - <https://www.pololu.com/product/108>
- Don't make your PCB load bearing! (I made the mistake)
- 3D printed TSSP housings to direct light
- Include a handle to pick-up your robot
 - Zip ties usually are the best here
- Fusion360 is the recommendation for 3D modelling software



Soccer – Electrical I

- Highly recommend running a 12V robot – 3 cell lipo which ranges from 12.1V - 10V when empty
- Motor drivers
 - <https://www.cytron.io/c-motor-and-motor-driver/c-motor-driver/p-3amp-4v-16v-dc-motor-driver-2-channels>
- Fuses (Please)
 - Cheap (<\$10) - Literally saves you robot from catching fire
 - Blade Mini from Jaycar / Supercheap Auto / Bunnings / Repco are perfect
- Reverse polarity protection (Literally \$1)
 - SCHOTTKY DIODE, not any diode
- Don't use the inbuilt 5V / 3V3 regulators on Arduinos



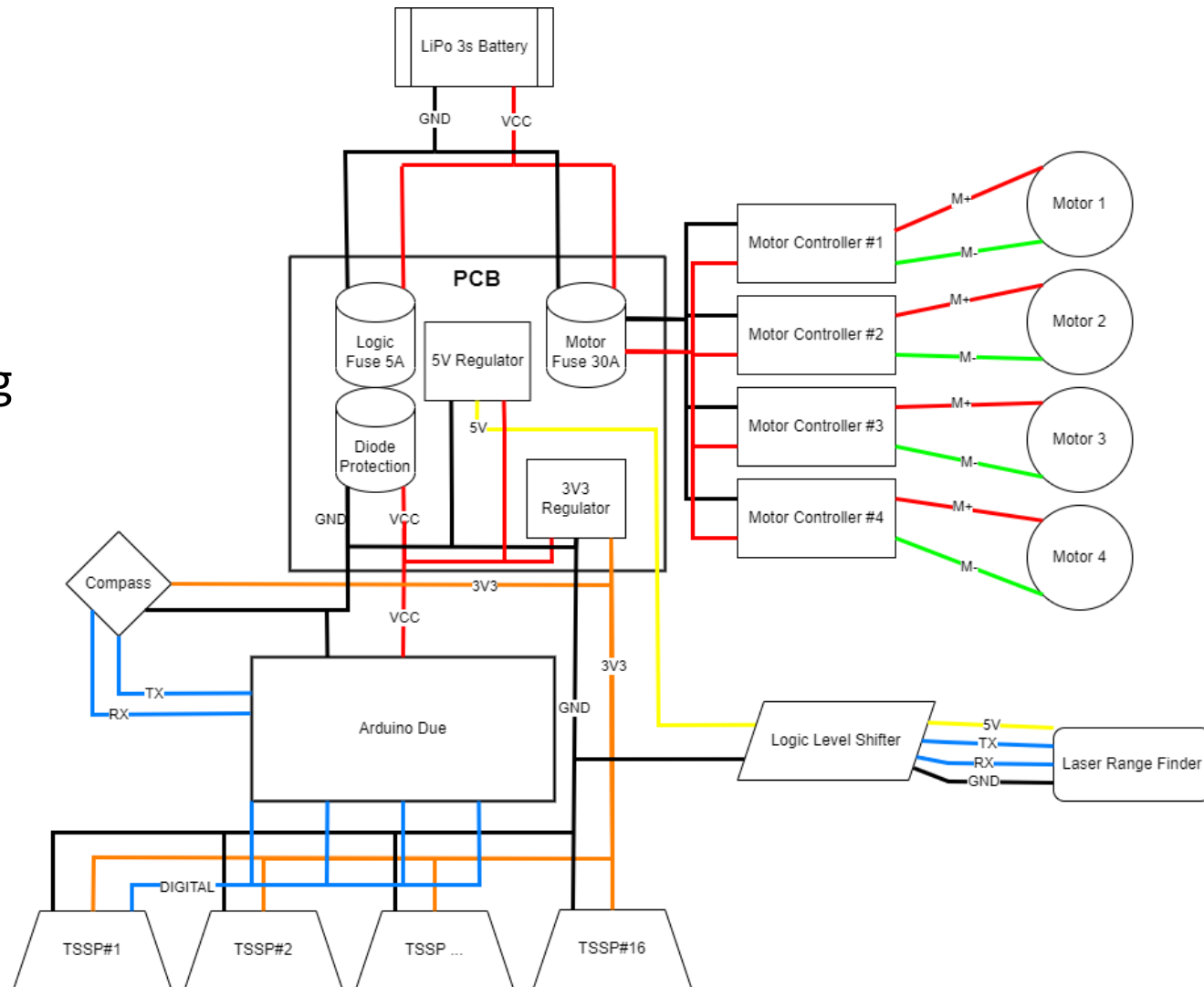
Soccer – Electrical II

- Plan your Electrical system before anything (Draw.io is amazing)
- You don't need PCBs!
- <https://app.diagrams.net/>
 - Wiring diagrams make both designing and diagnosing 100x easier
 - Do one for each PCB and the overall wiring for the system
- 5V vs. 3V3 systems & Logic Levels

Either: KiCad

Autodesk Eagle

Autodesk Fusion360 PCB



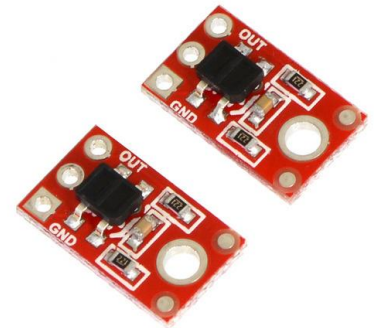
Soccer – Sensors I

- TSSPs <https://au.mouser.com/datasheet/2/427/tssp40-1767011.pdf>
 - DigitalRead in Arduino – HIGH when no signal, LOW with detection
 - Higher number of LOW readings = higher strength (which can find distance)
 - Typically, 8 – 16 per soccer robot in a ring
- Ultrasonics / LRF
 - <https://www.sparkfun.com/products/15569>
 - <https://core-electronics.com.au/tf-mini-lidar-tof-laser-range-sensor.html>
 - Ultrasonic are cheap and simple, low accuracy (not a huge issue in soccer)
 - LRF are \$\$\$ but have high accuracy
- IMU / Compass
 - Tell the robot what way is opposing goals
 - <https://core-electronics.com.au/mpu-6050-module-3-axis-gyroscope-accelerometer.html>



Soccer – Sensors II

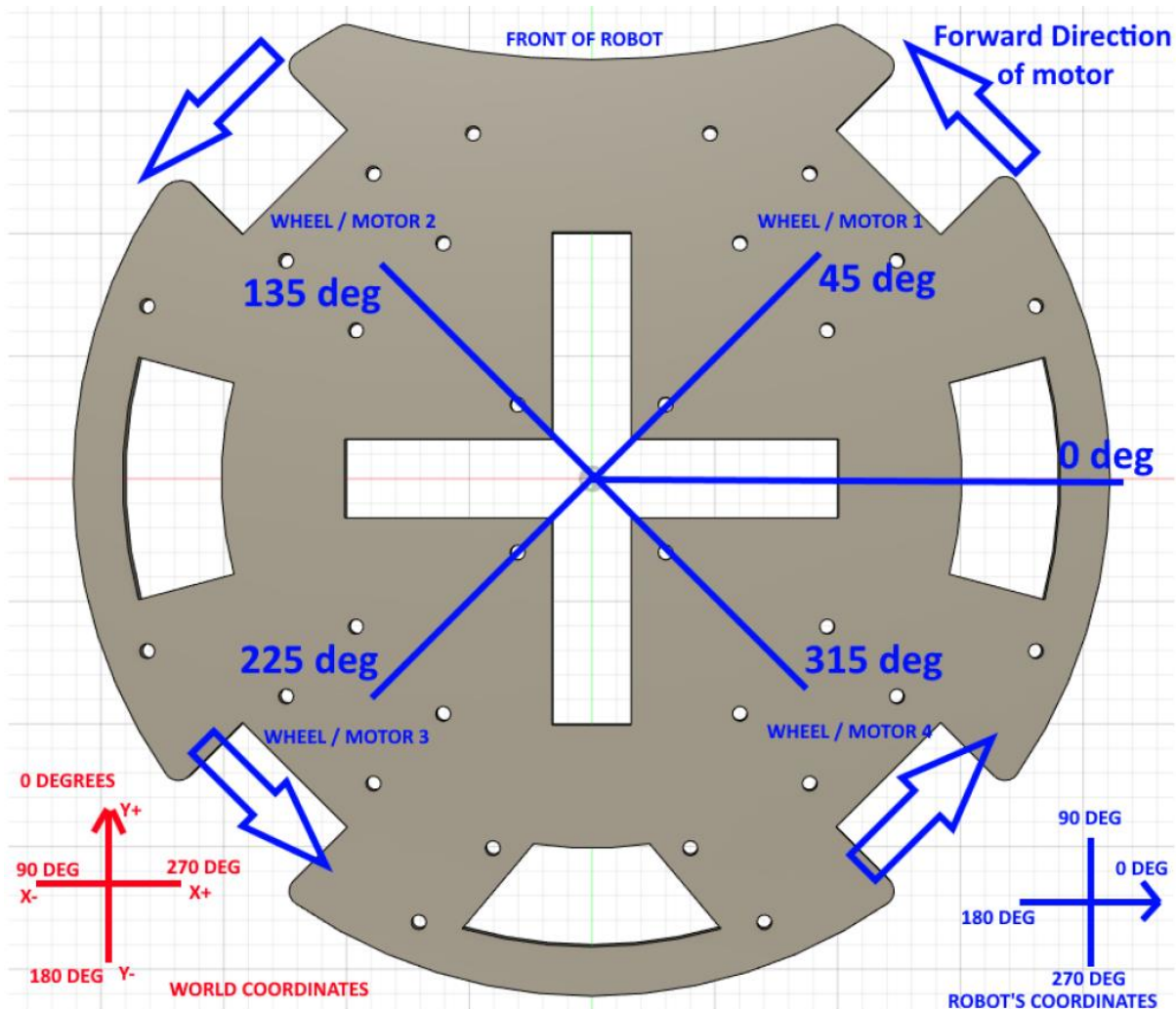
- Light Sensors: <https://arduinogetstarted.com/tutorials/arduino-light-sensor>
 - I recommend red LEDs as they have high contrast to green field
- Cameras – Pixy 2 / OpenMV
 - Both have built in processors and libraries to do most of the work for you
 - <https://pixycam.com/pixy2/>
 - <https://openmv.io/products/openmv-cam-h7>
- Not allowed parts: IR emitters between 920nm - 960nm
 - Pololu Reflected Light sensors
 - Some LRF (Laser range finders)
 - This info is usually in the datasheet



Soccer – Software I

- Highly recommend using C / C++ based IDEs (Arduino, etc.)
- Raspberry Pi / Micropython have too much overhead to process information fast enough for soccer
 - Exception of Open league for camera work
- Arduino Giga / Due is an excellent processor for speed, I/O and ease of use
 - <https://store-usa.arduino.cc/collections/boards/products/giga-r1-wifi>
 - <https://store-usa.arduino.cc/collections/boards/products/arduino-due>

Soccer – Software II



This is where the magic happens

- STRONGLY recommend 4 motor drive for 1st time
- Motors are evenly spaced 90 degrees apart, 45 degrees offset
- Forward for each motor is counterclockwise (Robot would spin anticlockwise too)

Soccer – Software III

Motor $x = \text{abs}(\text{cosine}(\text{motor angle} - \text{desired direction})) \times 255 \times \text{desired speed}$

Example:

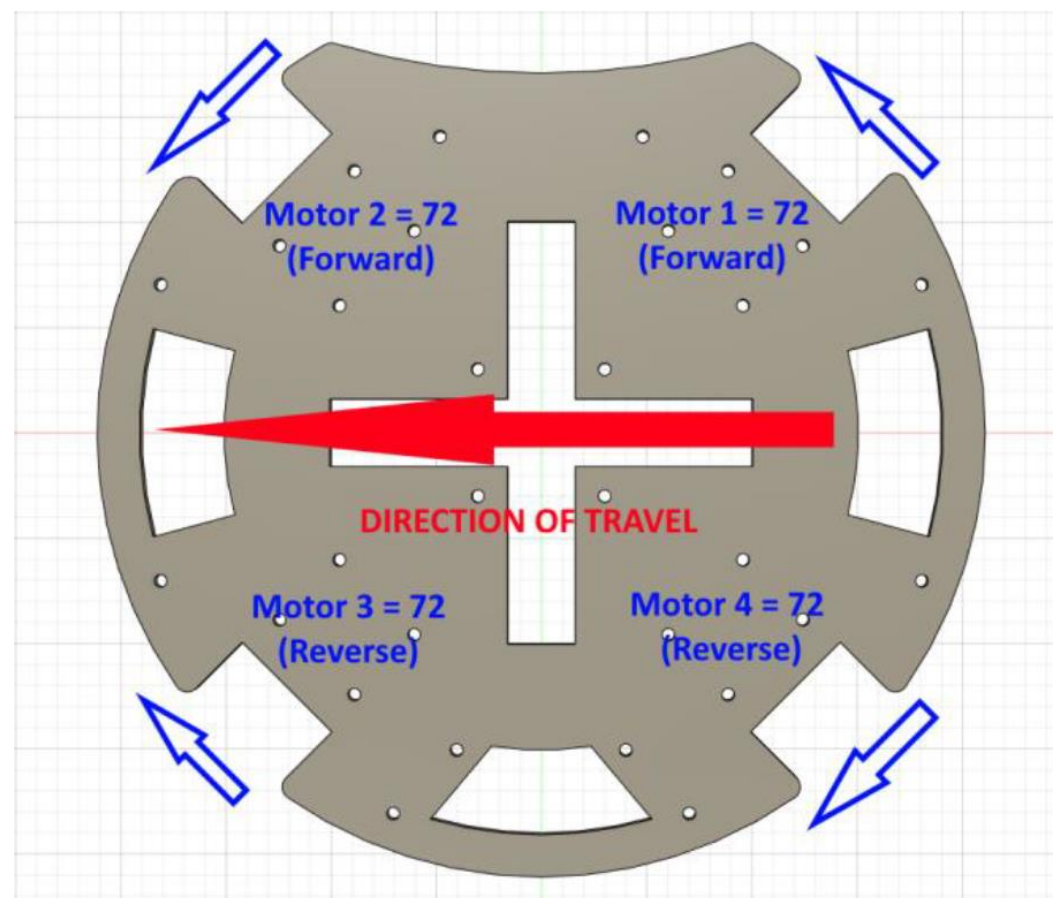
Motor 1 = $\text{abs}(\text{cos}(45 - 90)) \times 255 \times 0.40$

Motor 2 = $\text{abs}(\text{cos}(135 - 90)) \times 255 \times 0.40$

Motor 3 = $\text{abs}(\text{cos}(225 - 90)) \times 255 \times 0.40$

Motor 4 = $\text{abs}(\text{cos}(315 - 90)) \times 255 \times 0.40$

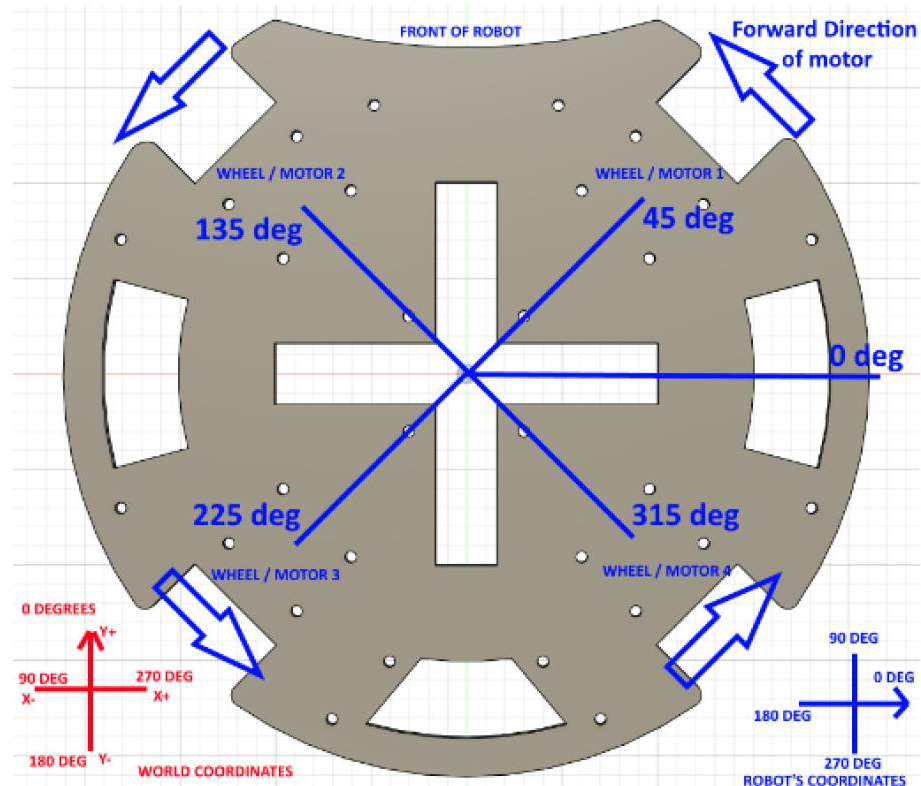
- Desmos or your preferred calculator here



Soccer – Software IIII

Motor $x = \text{abs}(\text{cosine}(\text{motor angle} - \text{desired direction})) \times 255 \times \text{desired speed}$

Adding Compass Correction



- This is where you push the limits
- Most improvements come from optimising your sub-functions that feed this main line of code

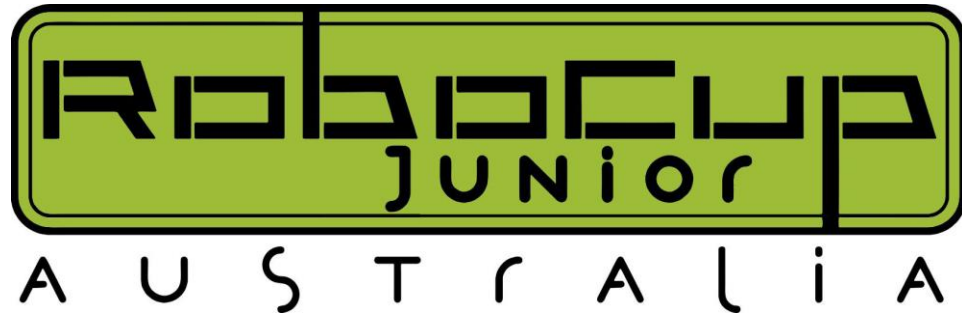
What questions do you have?

Useful Resources:

Past competitors Websites / GitHub repositories

<https://www.robocupjunior.org.au/past-competitors/>



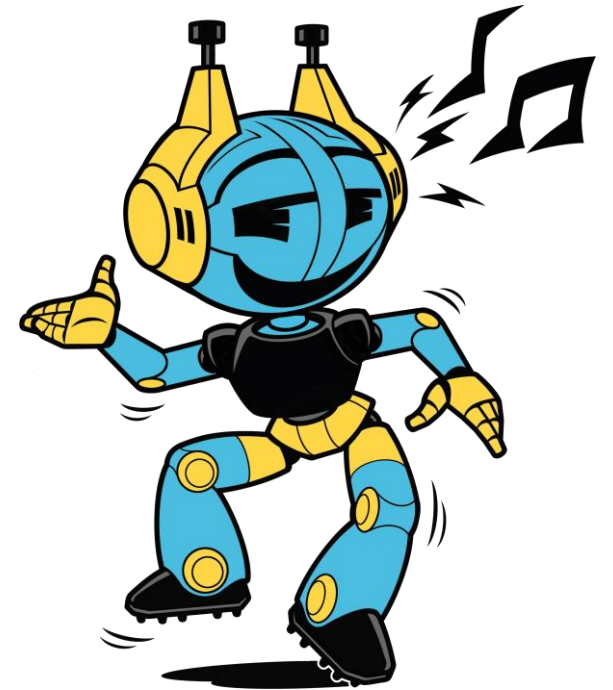


While you're waiting for us to get started, why not checkout our website?
<https://robocupjunior.org.au>

Raspberry Pi & Vision

Thank you for joining us for our 2023 Online Arduino Workshop

Seb Taylor – RCJV Committee Member
seb.taylor@robocupjunior.org.au



Interruption Policy

Please interrupt at any
time as often as you like*

This is a Workshop *not* a Lecture

* Yes, like that annoying kid in class today



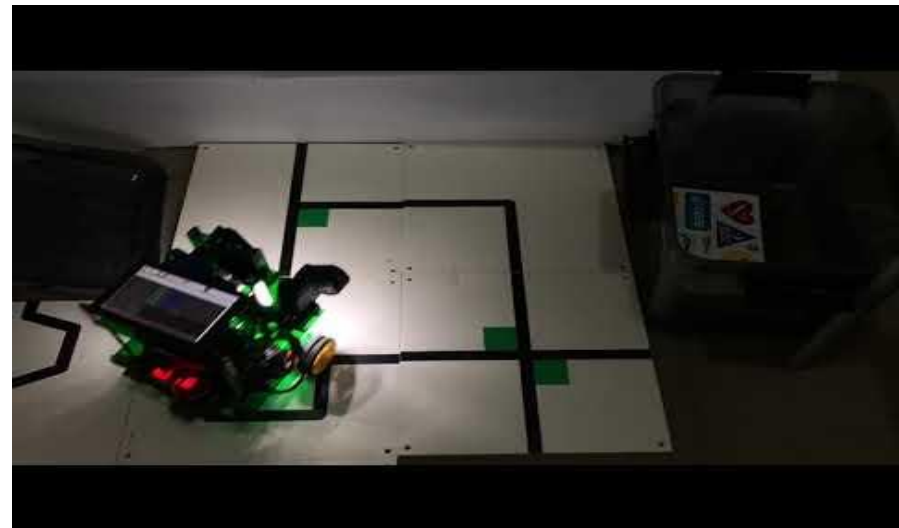
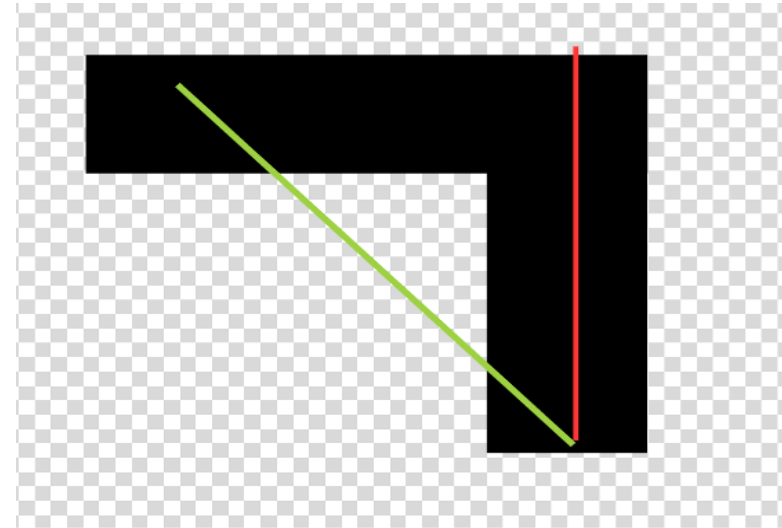
So, what's so good about a camera bot?

- Well like many things the answer is not just “camera bots are always the best not matter what, so I have put some pros and cons

Positives	Negatives
It can be much faster than other bots	It takes a lot more work
It's a new area and it is fun to discover new ideas	It's a new area so not a lot of documentation
A lot of possibilities to innovate	Most likely to be more expensive than other bots
Really rewarding when it works	Very processor intensive
Gives a lot of knowledge and experience	

So why is it so fast

- Well for a couple of reasons I think camera bots are generally faster than Arduino bots, but it can depend, there are slow camera bots and fast Arduino bots
- The first reason is that camera robots can anticipate the curvature of the line and can lead to an overall faster and more accurate line following
- Also, if you are really advanced you can do dynamic green turns, which means instead of turning a fixed distance you can use the camera to inform the robot how much to turn



When should I consider a camera bot

- Well, I do not think there is a set criteria for when someone should or shouldn't do a camera bot and it really varies from person to person, but I will say the things that helped me
- Being proficient in programming and have a general knowledge of python or C++
- Having a partner who knows hardware and can navigate software like fusion or vCarve (Although not needed is very helpful)

The Brains (SBC)

- The brains of the camera bot is arguably the most important piece of entire robot as it can determine what you can and cannot do
- The brains of the robot will do all the image processing, motors, and other sensors!
- They are also the most expensive! So, keep in mind what you want to accomplish and the budget you want to keep

Brain option 1: The Raspberry Pi

- The Raspberry Pi is probably known by most of you as it is the go-to SBC (single board chip) by many people and robotics enthusiasts
- This is the brains that I suggest most of you to use for many reasons



Pros	Cons
Incredibly good documentation and will work out of the box	Due to the current chip shortage raspberry pis are few and far between (and if you can find them, they are super expensive!)
A Raspberry Pi has a good enough CPU to deal with any image processing you can throw at it!	Not very good at very processor heavy things like artificial intelligence
Very good camera support (pi camera)	Only has one camera slot! (unless you get a CM4)
Uses a variety of coding languages, like python and C++, so no need to learn a random language	

Brain option 2: NVIDIA Jetson

- The NVIDIA Jetson is an SBC that is not as well-known as the Raspberry Pi but that does not make it bad!
- The NVIDIA Jetson is more focused on advanced camera bots (so I would not suggest it for simpler camera bots)



Pros	Cons
Very easy to buy it! The chip shortage has not affected the availability of this chip	The NVIDIA Jetson does not have as much documentation as the Pi and as thus problems can be harder to debug
The CPU is faster and better than the raspberry pi, also has better cooling	Can be very power hungry, and drain power from batteries incredibly quickly
With the AI acceleration this chip focuses on being an advanced robot	About 100\$ more than the Raspberry Pi
Also uses a variety of programming languages like the Pi	Uses linux OS Debian so a little bit harder to navigate than Raspbian
Has 2 camera slots in comparison to the raspberry pi's 2	

Brain option 3: Rock Pi

- The Rock Pi is a nice middle ground for camera robots due to its fast CPU and AI capabilities
- While it is not as fast as the Jetson for AI capabilities it still has the ability for 6TOPS, which is more than enough for most AI models



Pros	Cons
Very easy to buy it! The chip shortage has not affected the availability of this chip	Uses an incredibly weird OS that is android based
The CPU is faster than a Pi's but not as fast as a Jetson	Not very well documented (even less so than the Jetson)
Has an NPU capable of 6 TOPS (faster than the Pi but not as fast as the Jetson)	Cheaper than the Pi
Also uses a variety of programming languages like the Pi	
Has 2 camera slots in comparison to the Raspberry Pi's 2	However these camera slots are not the normal Pi camera slots

The Cameras

- Well now we are at another important section of the robot, the cameras, another choice that needs to be made with multiple options
- Like the brains it really does depend on what you plan to do with your bot and the level of the competition you want to go into
- For example, if someone is making a riley, or secondary camera bot they may only need one camera, but open robots, like Jankpot, is currently using two

Line following camera

- Unlike the brains, there is not tons of options with the line following cameras, this is because the line following camera needs to have high FPS in order to deal with sudden line changes
- In order to get this high FPS a CSI-2 camera must be used, not an USB camera (I can go into further detail about why USB cameras are capped at around 30FPS on the Pi while CSI-2 cameras have around 200FPS, if people want technical details)
- The best CSI-2 camera is made by raspberry pi with their recent Pi Camera 3 being an incredibly cheap and great product to use, this is personally what my robot uses (although it uses the pi camera 2 as 3 was not out yet)
- While you can get more expensive CSI-2 cameras, like global shutter ones, they are not needed and only really increase the price of the components in line following

Camera option 1: PiCamera 3

- The PiCamera 3 is the latest iteration of the PiCamera line and is probably the best pick to choose for a line following camera
- I really suggest for you to use this camera



Pros	Cons
Quite cheap for cameras with it being around 45\$ AUD	Only 1 can be used at a time with the standard Raspberry Pi 4
Has a very fast top framerate of around 120FPS@low resolution	Can be very power hungry, and drain power from batteries incredibly quickly
Has multiple versions with wide lens and NOIR (ir cut options)	About 100\$ more than the Raspberry Pi
Very good documentation and APIs	Does not come with global shutter and can lead to rolling effects, although not an issue on the HQ Picamera

Camera option 2: USB Cameras

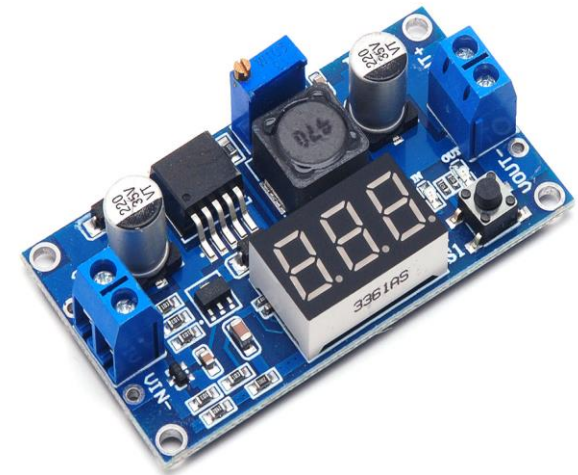
- The USB Cameras, while good I really do not prefer them over the PiCamera 3 for line following



Pros	Cons
Incredibly cheap can be around 20\$	Only 1 can be used at a time with the standard Raspberry Pi 4
You can use multiple (up to 4) at one time with the raspberry PI	Can be very power hungry, and drain power from batteries incredibly quickly
Has an incredible number of variants from many stores with global shutter	LIMITED AT 30FPS ON A PI DUE TO NO GPU ACCELERATION UNLIKE PICAMERA
Very good documentation and APIs	Usually not documented as well

Power

- Camera bots require exponentially more power than other custom robots, while Arduino robots may only need one battery, my current robot is using 3, 3 cell batteries
- This is because there are a lot of components that require power, the component that takes up the most power is the Pi by far though, so I suggest a 3-cell battery with a bit of capacity (2200mah should be adequate)
- The next battery is for motors and servos generally, as we have found that hooking the raspberry Pi and the motors to the same power source can lead to temporary drops in current causing the Pi to crash, so it is more reliable to separate the power sources
- Another thing to consider is the lighting power, while this can be hooked up to the motors there has been some issues with flickering lights with the current issues
- Keep in mind that Raspberry Pi takes 5V@2.5A although I do suggest if you are overclocking to do 5.25V@2.5A (also if you are overclocking make sure to use a cooler!)



The Software

- So now we can get started on some of the common libraries that people are going to use, I am going to write this mostly about the raspberry pi 4 and their operating system as that is the system, I have spent the most time on
- The main library that I suggest that you use for your robots is OpenCV, which is an incredibly powerful image manipulation software that can be used in multiple languages, though mainly python and C++.
- If you are using a raspberry pi, I really do suggest using the latest updates with 64bit as it can lead to a performance boost of around 25% compared to 32bit

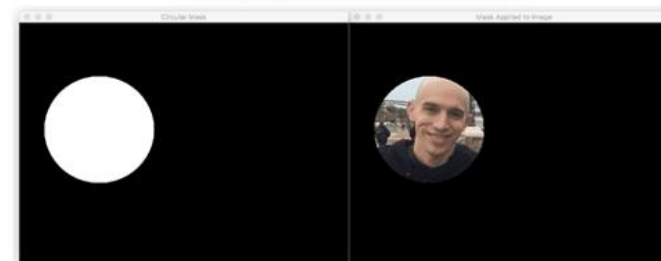
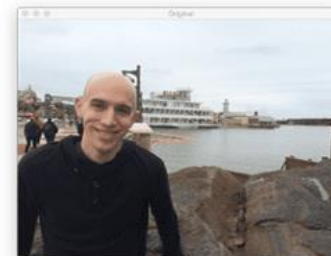


Libcamera

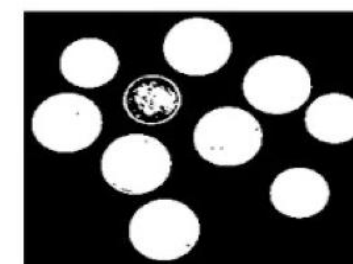
- This is only really applicable to people using the raspberry pi on the latest software
- Libcamera is the newest version for the raspberry pi to interface with the pi cameras that are sold, this is preferred over the methods that most outdated tutorials will show the use of picamera library.
- However, the picamera library has not been updated for over 2 years and has been deprecated in favor of libcamera, and as thus breaking issues with picamera will not be supported or fixed, but in libcamera it will

Important OpenCV concepts to learn

- So now we are really going to get into the meat and bones of computer vision by talking about important concepts when using OpenCV
- The first concept is thresholding, this is pretty much trying to extract features of an image depending on the pixel value, and it returns a binary image (black or white)
- This binary image can be used as an image mask (using bitwise operations) and extract only the bits you want
- You can also detect contours with the binary images, which are the white blobs with openCV



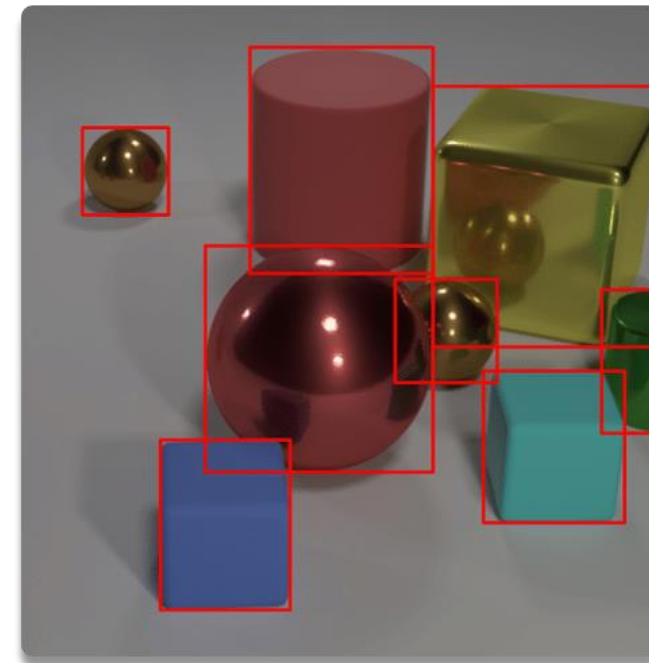
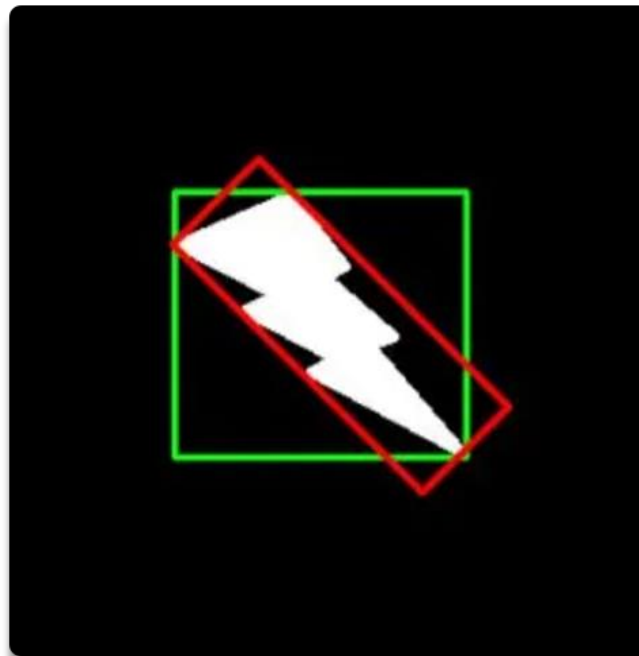
Input Image



threshold output

Important OpenCV concepts to learn

- The next important concept to learn is bounding boxes, which is a box that covers your entire contour, this is useful for finding the upper and lower bounds and the middle of your contour
- However, a bounding box is not rotated! And is always straight up, if you want to determine angles you would want to use `minAreaRect`, which can return an angle value!



How to find the line?

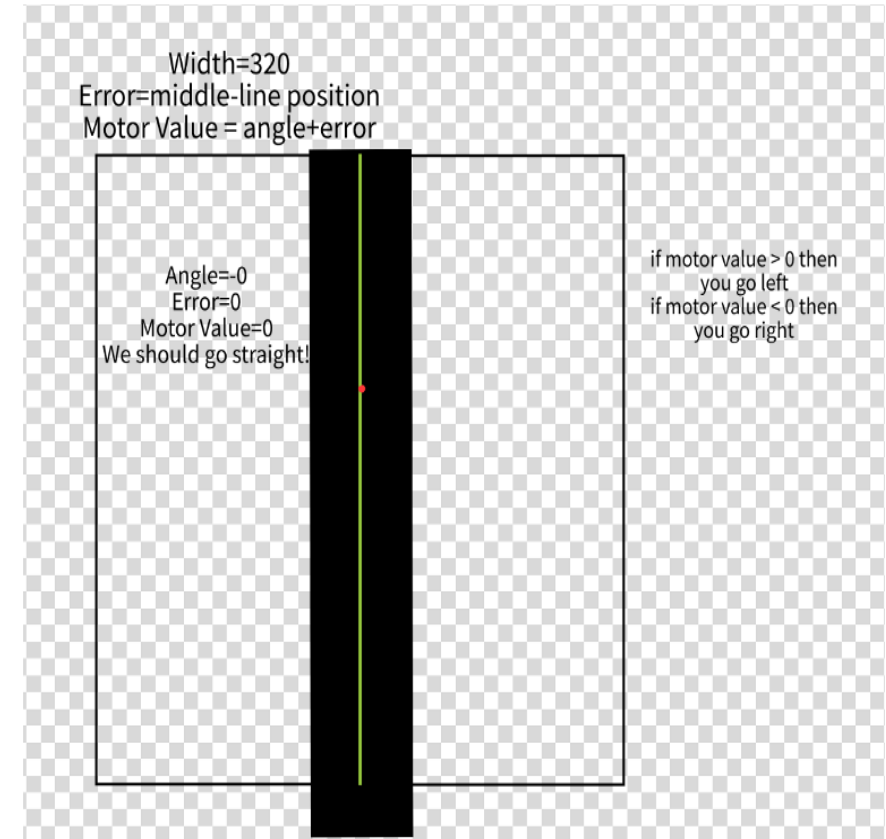
- Well now we need to actually find the line and OpenCV helps quite a bit and has very useful functions for the finding of black lines

```
img0_gray = cv2.cvtColor(img0, cv2.COLOR_RGB2GRAY)
img0_binary = cv2.threshold(img0_gray, 127, 255, cv2.THRESH_BINARY)[1]
img0_binary = cv2.dilate(img0_binary, np.ones((5,5),np.uint8), iterations=2)
img0_binary = cv2.bitwise_or(img0_binary, cv2.bitwise_not(img0_green))
img0_binary_not = cv2.bitwise_not(img0_binary)
black_contours, black_hierarchy = cv2.findContours(img0_binary_not, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
```

- And then our black contours variable contains “contours” (sets of pixels that make up the lines, and then you are going to want to do filtering by determining the distance from the last black line etc.

Line following: Basic

- So now we want to get the basic problem for line following solved and I will give some theory and talking points about how to do this
- The underlying function of linefollowing of camera bots is to get the robot to get the line straight in the eyes of our camera
- In order to get this line straight we need to give our motors a value to turn, but how do we get that one value that describes the line and its position
- Our one value is two values added together, these two values are called the error and the angle
- The error is how far the line is towards the middle of the screen, called the setpoint, and the further away the line is from the middle of the screen the greater we want to turn towards that direction
- The angle is the direction that the overall line is going and the greater that the line is turning the greater we are going to turn
- Then to calculate the motor values maths should be done to determine that 0 motor value is 100% motor speed on both motors and the max value of the motor value, in this case max Angle + max Error which is 250 (90^*+160) should be 100% right motor and 0% left motor



More line examples:

Width=320
Error=middle-line position
Motor Value = angle+error

Angle=-0
Error=-70
Motor Value=-70
We should go right!

if motor value >
you go left
if motor value <
you go right

Width=320
Error=middle-line position
Motor Value = angle+error

Angle=-15°
Error=-20
Motor Value=-35
We should go right!

if motor value >
you go left
if motor value <
you go right

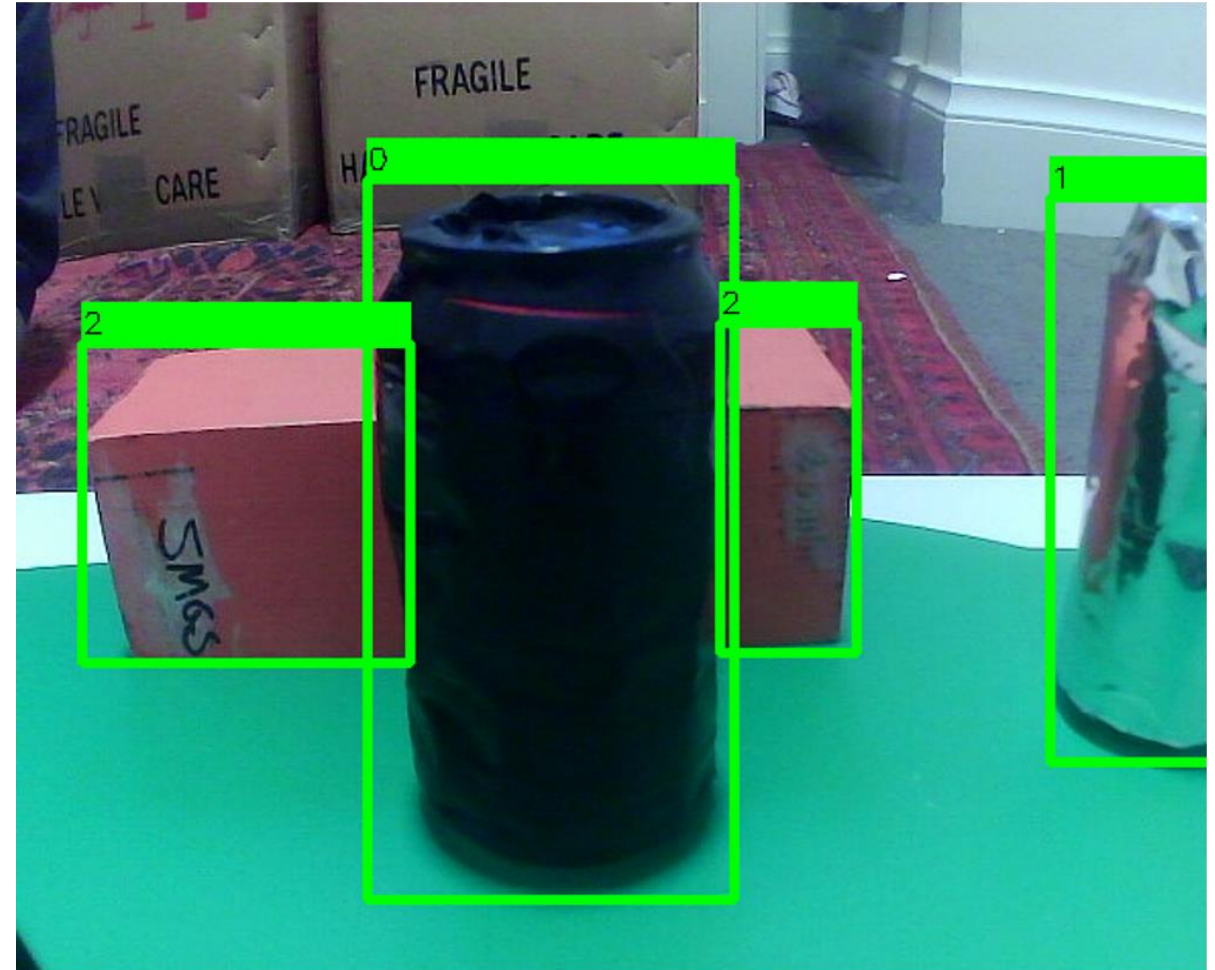
Line following: Greenturns

- Now when you are looking for green turns it may be tempting to just to use RGB and just look for a green threshold but there is a better way!
- I suggest when looking for a specific color that you choose the HSV color space over the RGB (which it is quite easy to convert an image in OpenCV) for many reasons
- Firstly, HSV can be less sensitive to lighting conditions if you only choose the hue option, and lighting issues can vary widely from event to event, so it is a very big deal for the algorithms to be robust!
- Then you can get the green using HSV thresholding, and then find the green contours and then find where the contour is in relation to the line (by finding the bounding box) to determine if it is left or right, then viola! You can do simple greenturns



Artificial Intelligence

- One of the main key points of interests with computer vision is the application of it towards artificial intelligence and object detection
- There are some things to consider what you want to apply artificial intelligence towards, is it detecting the block? Cans? Obstacle? The silver right before green zone?
- However, it is also important to consider whether there is an easier more reliable way that does not involve artificial intelligence
- This is because artificial intelligence is very intensive on the processor, and if you are only using a Pi, you will get very slow inference



How to decrease the inference time?

- Well, there are 2 ways to increase the inference time, you can optimise the model (by reducing accuracy or using a small model) or you can buy USB accelerators
- Training and optimising models can be slow but the results can be worth it, I suggest using software such as Open VINO or google coral which are both excellent open-source projects that you can use to optimise your models

USB Accelerators

- There are 3 choices for USB accelerators for the Pi that I have found, that would work for fast inference
- The first USB accelerator is the one that I am currently using which is Intel NCS2, which works well with open VINO and can run quite fast (4TOPS max), however it is EOL and not supported anymore
- However, there is an open-source alternative by OpenNCC for the NCS2 that is intended to be a drop-in replacement, although it is extremely expensive at about 200\$AUD
- Another USB accelerator that I would suggest is the google coral USB, which is cheaper than the NCS2 and can do int8 optimisation



What questions do you have?

