

Acknowledgements

I would like to thank my supervisor Michael Bass for his continued guidance and support in regards to writing, formatting, and publication of this report as well as the insights he provided me with during development.

I would also like to thank Dr Chriss Bates and Mick Marriott for their understanding and support in regards to robotics. I would not have thought of this project without their insight. Their continued encouragement, support, and aid was invaluable.

Finally, I extend my deepest gratitude to my parents, Courtney and Bob Moncavage, for always supporting me in my endeavors and constantly serving as inspiration.

Abstract

The aim of this project is to develop a prototype for a quadrupedal robot capable of supporting its own weight and to develop a movement cycle utilizing servo motors. This paper discusses the steps involved in developing a prototype, the research undergone into designing a concept chassis and leg mechanics as well as the process undergone for wiring and programming all components in addition to the construction of the chassis itself. Additionally covering the overall success and failure of the project and future iterations upon the quadrupedal robot prototype.

Table of Contents

Acknowledgements.....	1
Abstract.....	1
1. Introduction.....	2
1.1. Project Overview.....	2
1.2. Project Aims and Objectives.....	3
1.3. Glossary/Terminology.....	3
2. Investigation.....	4
2.1. Existing Robotics.....	4
2.1.1. Boston Dynamics' Legacy Robots.....	4
2.1.2. Boston Dynamics' Spot.....	5
2.1.3. DeepRobotics's Jueying Line of Robots.....	6
2.2. Hardware.....	7
2.2.1. Flux.....	7
2.2.1.1. Arduino Mkr Wifi 1010.....	7
2.2.1.2. EMAX ES08MA II Servos.....	7
2.2.1.3. Adafruit PCA9685 16-Channel Servo Driver.....	7
2.2.1.4. MLX90640 Thermal Camera Breakout.....	7
2.2.1.5. PA1010D GPS Breakout.....	7
2.2.1.6. BME688 4-in-1 Air Quality Breakout.....	7
2.2.1.7. HC-SR04 - Ultrasonic Distance Sensor.....	8
2.2.1.8. SEN0395 - Human Presence Detection.....	8
2.2.2. Self Produced Remote Control.....	8

2.2.2.1. Adafruit ST7789 IPS TFT Display.....	8
2.2.2.2. Joystick Modules.....	8
3. Design and Development.....	9
3.1. Design.....	9
3.1.1. Movement Mechanisms.....	9
3.1.1.1. Tracks.....	9
3.1.1.2. Wheels.....	9
3.1.1.3. Legs.....	9
3.1.1.4. Decision.....	10
3.1.2. Initial Design.....	10
3.1.3. Delivered Design.....	12
3.2. Project Development.....	13
3.2.1. Development Methodology.....	13
3.2.2. Hardware Development.....	14
3.2.2.1. Legs.....	14
3.2.2.2. Chassis.....	17
3.2.2.3. Controller.....	18
3.2.3. Software Development.....	19
3.2.3.1. The Environment.....	19
3.2.3.2. Component Programming.....	20
3.2.3.3. Latency and Response Times.....	22
3.3. Testing.....	22
3.3.1. Free Standing Testing.....	22
3.3.2. Knee Joint Movement.....	23
3.3.3. Hip Joint Movement.....	23
3.3.4. Walking Testing.....	24
3.3.5. Controller Testing.....	25
4. Conclusion.....	26
5. Critical Evaluation.....	26
5.1. Project Evaluation.....	26
5.1.1. Successes.....	27
5.1.2. Failures.....	27
5.2. Future Implementation.....	28
5.2.1. Leg and Body Design.....	28
5.2.2. Hardware Overhaul.....	28
6. References.....	29
7. Appendix.....	31
Flux Chassis Pictures.....	31
MLX Thermal Camera output.....	37
03/31/23 Notes.....	38
Trello Board Pictures.....	39
Action Plan.....	42
Original Project Specification.....	43
Elaboration.....	44

Project Aims.....	44
Project deliverable(s).....	44
Action plan.....	45
BCS Code of Conduct.....	47
Publication of Work.....	47
GDPR.....	47
UREC 1 Form.....	47
UREC 1 RESEARCH ETHICS REVIEW FOR STUDENT RESEARCH WITH NO HUMAN PARTICIPANTS OR DIRECT COLLECTION OF HUMAN TISSUES, OR BODILY FLUIDS.....	47
1. General Details.....	48
2. Research in external organizations.....	49
3. Research with Products and Artefacts.....	50
Adherence to SHU policy and procedures.....	51

1. Introduction

1.1. Project Overview

Society has often been fascinated by the thought of having robotic assistants that can perform or aid them in the completion of tasks. One of the next steps of societal progression is utilization of autonomous robotics. The primary purpose of this project is to prototype a quadrupedal robot which is capable of three dimensional movement to be utilized in the assistance of search and rescue related operations by combining programmable circuit boards and the necessary additional components to allow operators to safely investigate and search potentially hazardous locations without having to risk the operators safety. In order to do this, the robot will be equipped with sensors which will enable it to determine its surroundings global positioning, determine air quality, as well as provide the user with a visual output of what's in front of the robot utilizing a 110° wide angle thermal camera and to send the gathered data and information back to the operator via the remote control.

1.2. Project Aims and Objectives

- Prototype a four legged self-standing robotic which can move forward and backward
- Improve my knowledge in robotics and 3D modeling/design
- Consider what additional components might be useful when creating Flux for the aims of scouting dangerous environments
- Create a graphic implementation of a thermal processing component.
- Test movement ability thoroughly and improve upon Flux's coding for leg movement
- If capable, develop code to allow Flux to jump or climb steps
- Research what type of thermal cameras will be suitable for operation with an Arduino type board or various Raspberry Pi models
- Apply good engineering practices to create a quality deliverable prototype with minimal to no defects which can serve as a base for a future iteration.

- Research other types of “search and rescue” robotics to determine what components are best for Flux to have built-in.
- Research inverse kinematics to help with the construction, design, and implementation of the final robotic legs.

1.3. Glossary/Terminology

Breadboard	A device used to create semi-permanent prototypes of electronic circuits
Flux	The name of the robot - referred to as “it”
GFX	Graphics
GPS	Global Positioning System
I2C	Inter-Integrated Circuit
LIDAR	Laser Imaging, Detection, And Ranging
MISO	Master In Slave Out
MOSI	Master Out Slave In
RAM	Random Access Memory
SCLK / SCL / SCK	Serial Clock
SDA	Smart Data Access
SEN	Short for SEN0395 - Human presence detection sensor
Servo	Servomotor - a type of rotary actuator
SPI	Serial Peripheral Interface
SRAM	Static Random Access Memory
SS	Slave-select Signal
TFT	Thin-Film Transistor
UART	Universal Asynchronous Receiver-Transmitter
ZIP	A widely used archive file format

2. Investigation

2.1. Existing Robotics

2.1.1. Boston Dynamics' Legacy Robots

Boston Dynamics has a lineup of 5 quadrupeds robots. Taking a look at the BigDog one may notice the design of the inverted legs as well as the curved band foot nubs for traction; there's also the inclusion of several cameras and monitors/sensors for environment sensing.



Figure 1: BigDog



Figure 2: LS3



Figure 3: WildCat

Similarly with BigDog, LS3 also has legs that have knee joints which are inverted of each other, meaning both will bend outwards or inwards which is a downside to the overall design of the legs being prone to buckling. WildCat has knee joints which both bend the same way; this design choice is carried on into the future generations with the introduction of Spot Classic (*Legacy robots*). The advantages of these legacy robots is the metal chassises which can offer protection from large objects damaging the robots' components; however because these are large metal frames they offer no covering of the overall design meaning smaller objects can enter and damage these internal components.



Figure 4: Spot Classic

2.1.2. Boston Dynamics' Spot

Boston Dynamics' Spot is the most recent iteration of their quadrupedal robots. The design of Spot is smaller when compared to the previous generations. Spot is equipped with numerous sensors and cameras which allow it to navigate its world by creating a 3d map of its

surroundings as demonstrated in a video by Boston Dynamics (*Spot Autonomous Navigation*). The strengths of Spot are its smaller, more compact design over the previous generations, the refined camera and sensor systems which offers more ways for Spot to detect its surroundings by using not only multiple sensor arrays on the sides and rear but also a criss-crossed sensor array in the front to offer potential binocular vision than that of the monocular vision of the previous generations; additionally the improved design of the legs, now using a polymer rather than metal, might offer less stress on the legs' motors which would extend the longevity of the components. The downsides to Spot could be the polymer chassis and construction itself which might get damaged in a hardhat zone (an area of work where one is required to wear a hardhat for one's protection) which is a place where Boston Dynamics often demonstrates Spot working (*What's New in Spot | Boston Dynamics*).



Figure 5: Boston Dynamics' Spot

2.1.3. DeepRobotics's Jueying Line of Robots

DeepRobotics takes a slightly different approach to developing their quadrupedal robots. At the time of writing there are four models: the Jueying, Jueying Mini, Jueying Lite2 and the Jueying X20.

Figure 6: Jueying Lite2



Figure 7: Jueying Mini





Figure 8: Jueying



Figure 9: Jueying X20

All of the models offer a built in depth camera, “high torque joints,” “high-strength limbs,” and either a built in or removable lidar attachment depending on the model (*Deep robotics - the global leader in quadruped Rob.*). As demonstrated in a video by DeepRobotics (*The Quadruped Robot Jueying Mini Lite*) the “Jueying Mini Lite,” presumably another name for the Jueying Lite2 as the name “Jueying Mini Lite” does not appear on DeepRobotics website, is capable of active braking (not moving into walls), self-righting, and the ability to handle sloped surfaces. The Jueying X20 serves a more search and rescue related purpose; a video by DeepRobotics demonstrates the ability to detect gasses, human identification, human following and thereby active pathfinding, radio voice communication as well as stable movement and stair climbing (*Jueying X20 Robot Dog Hazard Detection & Rescue Solution*). The strengths to these robots is the small design and presumably light weight of them which can offer a faster traveling speed. A few of the downsides to these are the requirement of a circular lidar for what I can only imagine is autonomous movement, the lack of side attached cameras or sensors and the front facing camera and sensor array system which doesn’t offer as wide of a forward sensing array than that of Spot’s design with two criss-crossed sensor arrays.

2.2. Hardware

2.2.1. Flux

2.2.1.1. Arduino Mkr Wifi 1010

In order to allow Flux to think and actually perform commands given to it Flux needs to utilize a microcontroller that serves as its brain, so to speak. The Arduino Mkr Wifi 1010 is a microcontroller with a powerful onboard processor equipped with 32KB of SRAM (utilized for storing variables) and 256KB of internal flash memory (utilized for storing bites of code) (*ATECC508A complete data sheet - arduino*). The high amount of SRAM allows the utilization of multiple complex variables that are required to fully operate Flux’s add-ons.

2.2.1.2. EMAX ES08MA II Servos

The EMAX ES08MA II’s are rated as having a high stall torque and quick operating speed (*Emax ES08MA II 12G Mini Metal Gear ANALOG SERVO FOR RC Model&Robot PWM Servo*). Having a high stall torque means that the EMAX servos will be capable of supporting Flux’s weight and allow free standing ability and the quick operating speed can allow Flux’s legs to quickly move from in

to out and vice versa. Additionally the EMAX servos are metal geared which makes it more difficult for the gears to slip, or not connect, under pressure.

2.2.1.3. Adafruit PCA9685 16-Channel Servo Driver

Since I'm going to be using servos as my movement mechanism and choice of motors, I will need something to power and operate each of the servo motors individually. The Adafruit PCA9685 16-Channel Servo Driver board will take a separate power input to power each of the individual motors, up to 16, as any power the board receives from the Arduino MKR Wifi 1010 (a 5 voltage signal) is not sufficient to power more than 2 individual servo motors, much less up to 16. These boards are also chainable, meaning you can attach multiple boards if more than 16 servos are needed; a great starting point for my needs (*Adafruit 16-channel 12-bit PWM/Servo driver - I2C interface*).

2.2.1.4. MLX90640 Thermal Camera Breakout

The MLX90640 is a thermal camera which will enable Flux to send a visual representation of Flux's forward surroundings to the remote control for the operator to use for piloting Flux. This will enable the ability to visually detect differences in the temperature of surrounding objects and make identification of human-like objects easier for an operator. The model I've opted for specifically offers a 110 degree wide angle view to see more of its surroundings.

2.2.1.5. PA1010D GPS Breakout

The inclusion of an onboard GPS for Flux will allow the operator to track Flux's position while powered on. This GPS can also enable the ability for Flux to map its own positioning and track the path it took to a destination to allow the operator to safely navigate to Flux's position. In terms of search and rescue this can serve crucial when navigating wreckage to determine what pathways are safe to take.

2.2.1.6. BME688 4-in-1 Air Quality Breakout

The 4-in-1 air quality sensor will be utilized to allow Flux to precisely detect the temperature, pressure, and humidity of the air in addition to being able to detect "volatile organic compounds, volatile sulfur compounds and presence of carbon monoxide and hydrogen" (*BME688 Digital low power gas, pressure, temperature & humidity sensor with AI*). With this component Flux will be able to determine a general measurement of air quality and determine safe levels for humans.

2.2.1.7. HC-SR04 - Ultrasonic Distance Sensor

Flux will be equipped with 6 ultrasonic sensors which will detect its proximity to nearby objects which Flux will use to prevent itself from running into walls or objects in its way. The HC-SR04 modules are composed of two ultrasonic transducers, one of which acting as the transmitter which produces 40 KHz ultrasonic waves or pulses and the other acting as the receiver of those waves. Ultrasonic sensors are an excellent choice when considering non-contact range detection; the HC-SR04 models are accurate from 2 cm to 400 cm or approximately 13 feet with an accuracy of +-3 mm (Last Minute Engineers, 2022).

2.2.1.8. SEN0395 - Human Presence Detection

Since Flux is designed to be a rescue robot it will need to be able to detect the presence of humans. The SEN0395 is a component that will be responsible for detecting human presence and human movement. This component has all of the detection and processing performed onboard through algorithms and a process of sending various radio waves which are then reflected back by moving objects and converted into electrical signals by the millimeter-wave MMIC circuit. According to its documentation, "The millimeter-wave radar can sense the human presence, stationary and moving people within the detection area. Moreover, it can even detect

static or stationary human presence such as a sleeping person” (*MmWave radar sensor Arduino-Human Presence Detection Wiki*) which will be beneficial in detecting potentially incapacitated individuals.

2.2.2. Self Produced Remote Control

2.2.2.1. Adafruit ST7789 IPS TFT Display

Since Flux has a built-in thermal camera sensor this data will need to be outputted in a visual format for humans to understand. With the addition of an Adafruit IPS Display the operator will be able to see exactly what the cameras see in color. This will be useful for controlling Flux when out of eyesight and to allow a human to make their own judgment calls based on the cameras’ feedback in addition to outputting all the data of the additional sensors for the operator to understand.

2.2.2.2. Joystick Modules

The controller will be equipped with two joystick modules which will control Flux’s movement and looking directions. One joystick will be responsible for Flux’s forwards, backwards, and side to side movement and the other will be responsible for Flux’s pitch and yaw. These will allow the operator to manually move Flux and to look up and down with Flux as well. Additionally, since both of the joysticks have their own buttons, one of these buttons will operate as the mode selector for the display, which I will touch upon later in the software portion.

3. Design and Development

3.1. Design

3.1.1. Movement Mechanisms

In the field of robotics there are many choices when designing movement for a robot. The choice of movement will be tailored to the purpose and environment of a robot.

3.1.1.1. Tracks

Tracks offer more surface area contact than either legs or wheels can offer. This then exerts “a much lower force per unit area on the ground being traversed than a conventional wheeled vehicle of the same weight” (*Tracked Robots*) making it an ideal choice for usage on softer surfaces that offer less friction such as sand, mud, or ice. Tracks are more complex than an implementation of a wheel design and instances such as torn or derailed tracks will cause the track system to fail.

3.1.1.2. Wheels

When it comes to developing a wheel based system for robots, there’s a few I’d like to mention. There are four wheeled and five or more wheeled designs. A four wheeled design can vary in implementation. There is a 2 powered wheels 2 free rotating wheels approach where the former would control motion and the latter controlling direction. There is a 2-by-2 powered wheel approach which is similar to the movement of a tank where both sides work in conjunction or opposition to control motion or direction respectfully. There is also an axle based approach where typically two wheels are powered and one or two axles connect either the front and/or rear pairs of wheels together which is then used for controlling the direction, similarly to that of

a vehicle. Finally implementations of five or more wheels often vary and are utilized for larger robots, though not always a practical decision to do so. The most famous implementation of this being the Mars Rovers used by NASA. Often both the front and rear wheels will be used to control directional movement (*Robotics/types of robots/wheeled*).

3.1.1.3. Legs

Regarding legged robots, there's also several approaches to designing them. I specifically want to mention three: two, four, and six legs. For the purpose of this paper I will keep the explanation simplified as there are a lot more complexities to each of the legged robotic designs than that of what I will discuss.

Two legged robots are often designed similarly to that of a human, relying on the momentum of constantly falling forwards to continuously move and put the other foot in front of it to prevent it from physically falling forward; one such implementation being Boston Dynamics' Atlas. (*Legged robot*)

Four legged robots which exhibit quadrupedal motion, similar to that of a dog or horse which have the benefit of higher stabilization than that of a two legged implementation but less stabilization to that of the six legged approach (*Legged robot*)

The six legged, often referred to as hexapods, offer a greater stability than that of the two legged or four legged implementations. The movement mechanics are often designed after those of insects such as a spider (*Legged robot*).

3.1.1.4. Decision

Ultimately I decided to go with a four legged or quadrupedal design for Flux. The added benefits of more stability over that of a bipedal robot as well as not having to manage weight distribution was a huge upside for four legged over bipedal. When it came to hexapods on the other hand, while they offer six points of structural stability and of weight distribution, I had determined that the gait of hexapods was far too complex for me to construct considering this would be my first time working with any actual moving robotic parts. The simplicity over a hexapod and the ease of design and movement over that of a two legged robot with the additional added complexity over wheeled or track based robots led me to designing a four legged, quadrupedal robot.

3.1.2. Initial Design

The initial design for Flux was rather ambitious. I had first drawn up what I wanted the chassis to look like and where everything would be placed.

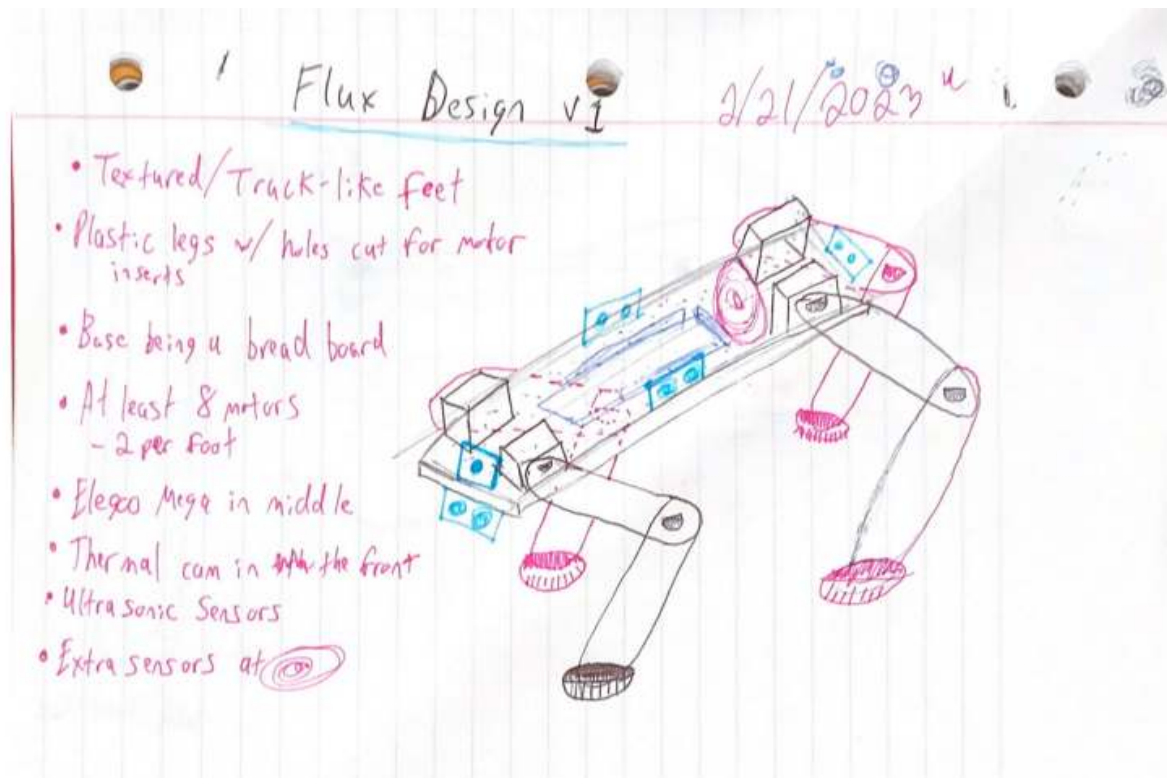


Figure 10: Flux Design V1

In version 1 of my overall Flux design, Flux Design V1, I denote where I desired the placement of all my components. The MLX thermal camera was to be at the front or “head” of Flux, drawn as the blue square with a singular blue dot in it. The HC-SR04 ultrasonic sensors, drawn as a blue rectangle with two blue dots, were to be placed underneath or next to the thermal camera, one on each side of Flux, and an additional sensor at the rear; all four of these would work in conjunction to alert the operator of nearby walls or objects. In the middle of the chassis was to be the processor Flux would require, initially thought to be the Elegoo Mega, though through later testing changed to the Arduino MKR WiFi 1010 due to memory constraints. The placement of the servos were initially thought to be directly on the center of the chassis with the legs connected to the gear of the servo thus serving as a hip joint which would control the equivalent of a plastic femur to swing the whole leg up and down with an additional servo connected as a knee joint to bring up and down the equivalent of the plastic tibia/fibula which had a rubberized surface at the bottom for traction.

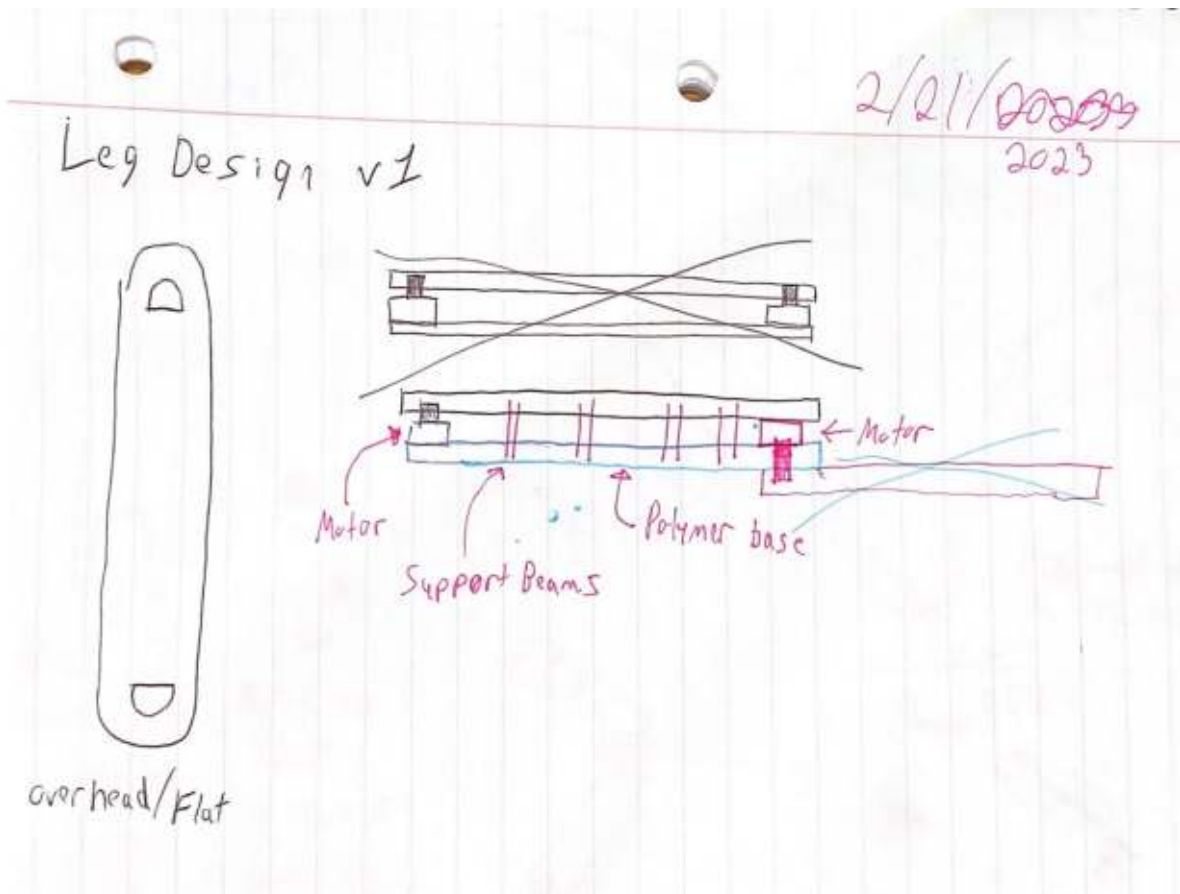


Figure 11: Leg Design V1

In Leg Design V1 I attempted to depict how I initially wanted the legs to be constructed, trying to keep it simple, knowing I would have had to 3D model and then laser cut or print this utilizing my university's equipment. I wanted two upper leg pieces which would connect together through support beams. Both the servos controlling the plastic legs were initially thought to be mounted inside and between the leg pieces with the first controlling the plastic femur and the lower controlling the tibia/fibula plastic which was constructed of only one piece.

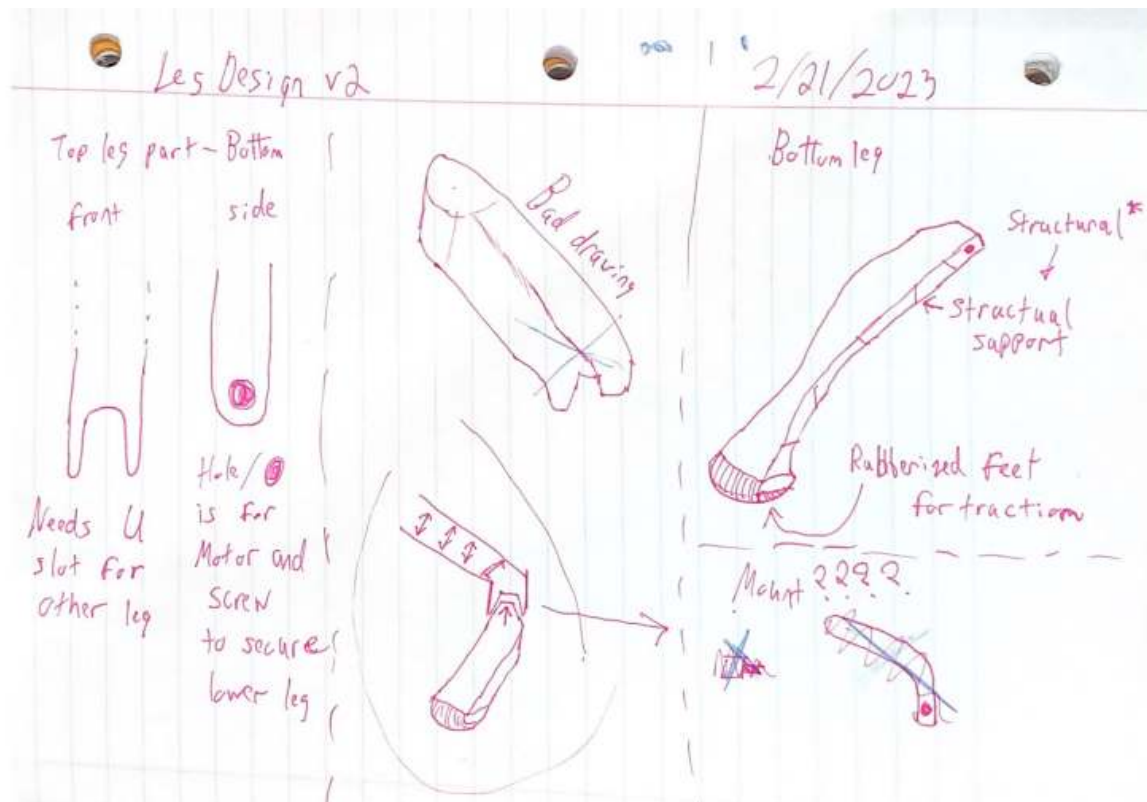


Figure 12: Leg Design V2

I quickly scrapped Leg Design V1 and in Leg Design V2 I had blueprinted what is a more secure and simpler design. The top part of the leg, the plastic femur, was to be a simple straight piece of plastic with a U-shaped bottom for the connection of the lower tibia/fibula leg. The lower leg itself was to be a slightly curved 3D print with a rubberized bottom foot. There would've been a hole cut out to connect to the U-shaped part of the upper leg where a servo or motor would then be placed on the outside to control the lower legs' movements. This also continued to expand upon my initial overall Flux design of having the servo or motors attached directly to the base of the chassis.

3.1.3. Delivered Design

The delivered design for Flux, unlike the drawings, was nowhere near what I initially imagined, rather messy, and a hassle to develop on. Due to lack of knowledge and skill with 3D printing and design, I ended up requiring a development kit for the construction of my legs. Thankfully my university's robotics module had extra robotic arm kits produced by MeArm: an older version of the current "MeArm kit for micro:bit" (*MeArm kit for micro:bit - nuka cola blue*) which actually came with 4 included servos; however I needed to replace a few of the servos due to the weaker stall torque which prevented Flux from supporting its own weight. I utilized this as the basis for my legs as it included pre-cut plastic which would be constructed into the arms. Overall, I do believe that the mounting system that MeArm developed for its base or foundation of the arms was superior to anything I could've developed. The issue with this kit is that there was no way to mount a rubberized foot therefore Flux would struggle to gain traction when moving, if any at all. It did however have a claw at the end of the arm which I locked in place to use as feet

which allowed Flux to easily stand and support its own weight with the new EMAX servos in place. Regarding the chassis itself, there is no space between the mounting of the legs which meant no space for the components in the middle as I had initially designed which led to less than ideal mounting options for all of my components. I ended up utilizing several bread boards spread across the plastic of Flux's chassis with them being taped in place. This also caused messy wiring spewn about which would sometimes cause other components to improperly read and output and was difficult to keep the wires secured in the breadboards due to the lengths of some of the wires.

3.2. Project Development

The aim of this project was to create a proof of concept prototype that would later serve as the base for an improved version of Flux given the approximate 20 week overall work period, noting that I could not receive my MeArm robotic arms until approximately 13 weeks into overall development.

3.2.1. Development Methodology

For the development of Flux I had decided to work in sprint weeks utilizing the agile development methodology rather than slowly but continuously working over the course of the semester. According to Megan Cook, the Group Product Manager for Jira Software at Atlassian “With scrum, a product is built in a series of iterations called sprints that break down big, complex projects into bite-sized pieces” (REHKOPF). The agile approach is one that incorporates working in sprints into the methodology which is perfect for me. First one receives the concept or requirements of an idea or project which is then outlined and designed into something one can use as a potential starting point. Then development is undergone where iterations of the concept are produced into something deliverable; this is later tested and depending on the testing phase is either deployed, delivered, or will return back to the design and development phases (*Agile methodology for software development*).

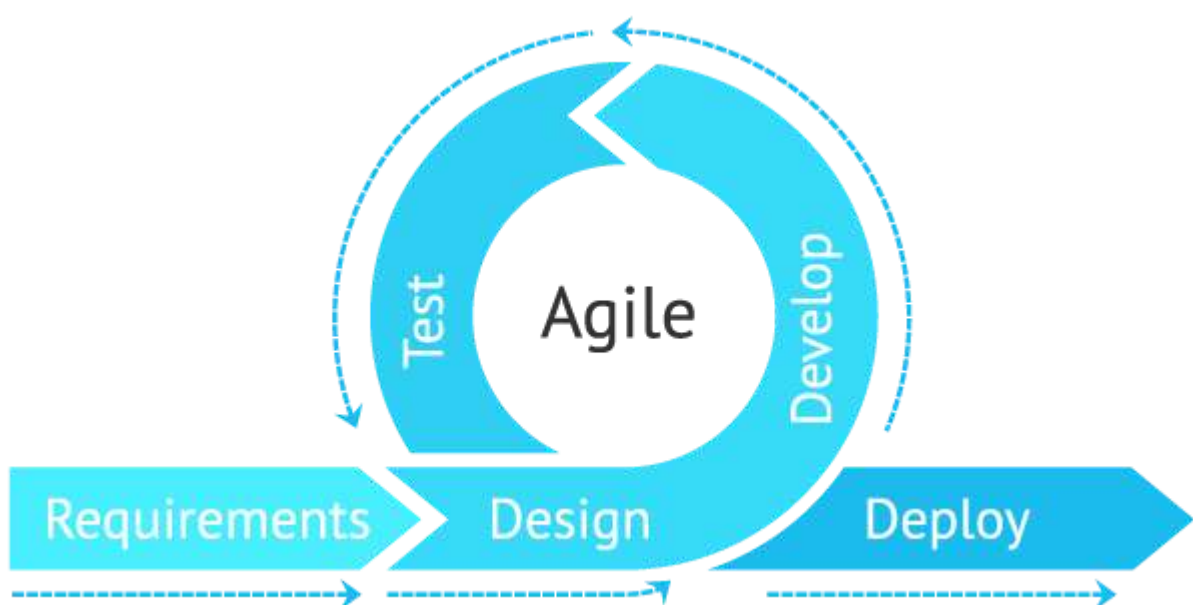


Figure 13: Agile Methodology

During my initial proposal I had laid out an action plan full of milestones that I would undergo during development. I took these milestones and placed them into a trello board, breaking them down so I knew what I needed to do during each of my sprints. This broke the overall development phase into smaller, more manageable steps that were a lot less overwhelming. However, as previously mentioned I had encountered an issue in 3D modeling the legs so I had adjusted my remaining sprint plans to account for the new hardware I was using. Working in sprints offered the best time management and productivity for myself. During the development of Flux I was also working on another robotics project for my course which was going on simultaneously. Both of these projects required lots of detailed and extremely independent work to be done; nothing where I could take the work of one project onto another. The usage of sprints allowed me to separate the work of both projects into manageable lists of “what needs done” which allowed me to create and deliver high quality work. Had I not utilized sprints I undoubtedly would have prioritized the work of one over the other which would have caused the development of Flux to suffer massively.

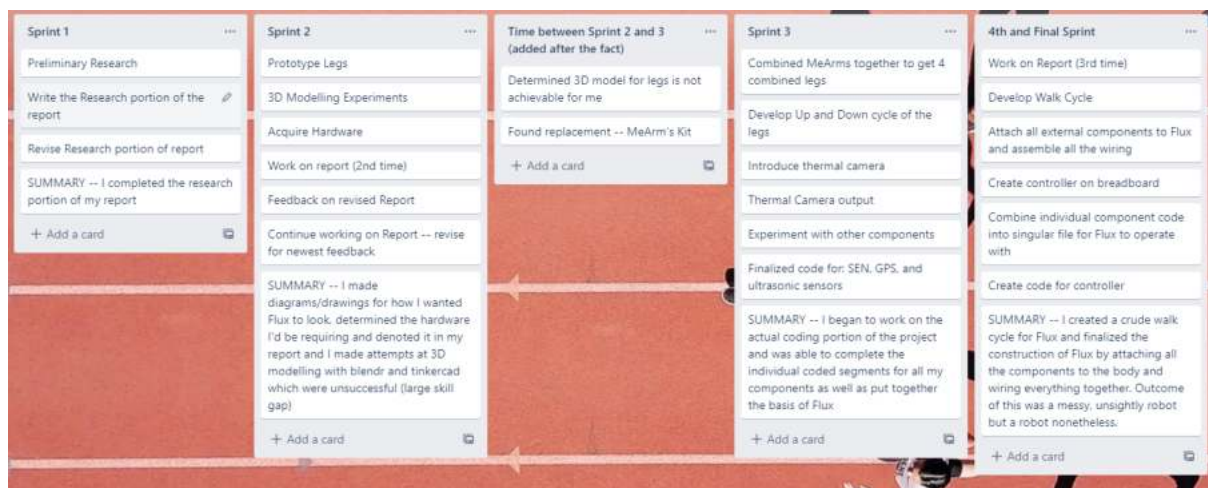


Figure 14: Flux Trello Board
note: to see detailed trello see appendix

3.2.2. Hardware Development

3.2.2.1. Legs

The development of the hardware was more difficult than I had initially anticipated. Since I could not determine how to properly 3D model, I was forced to use what was available to me at the time; that being the 4 MeArm robot arms. This had led to a lot of constraints in terms of power and configurability. The MeArm kit is designed in such a way that it requires the exact model of servo in order to fit into the plastic moldings. The issue with this was that the servos that were included in the kit were extremely weak and required replacing with stronger servos for the operation of the four legs' up and down motion, or the “knee joint” servo. I ran into numerous issues when replacing the servos. The first and often recurring issue was that the plastic was rather fragile and would snap if not handled precisely. There were several points of failure of the plastic of the MeArms, however the most common failure would be the middle connection joint of the legs – the equivalent of one’s femur bone. This joint was critical as it

connected the servos' plastic "tendons" together where one controlled the leg's up and down motion and the other controlled the leg's swing in and out.

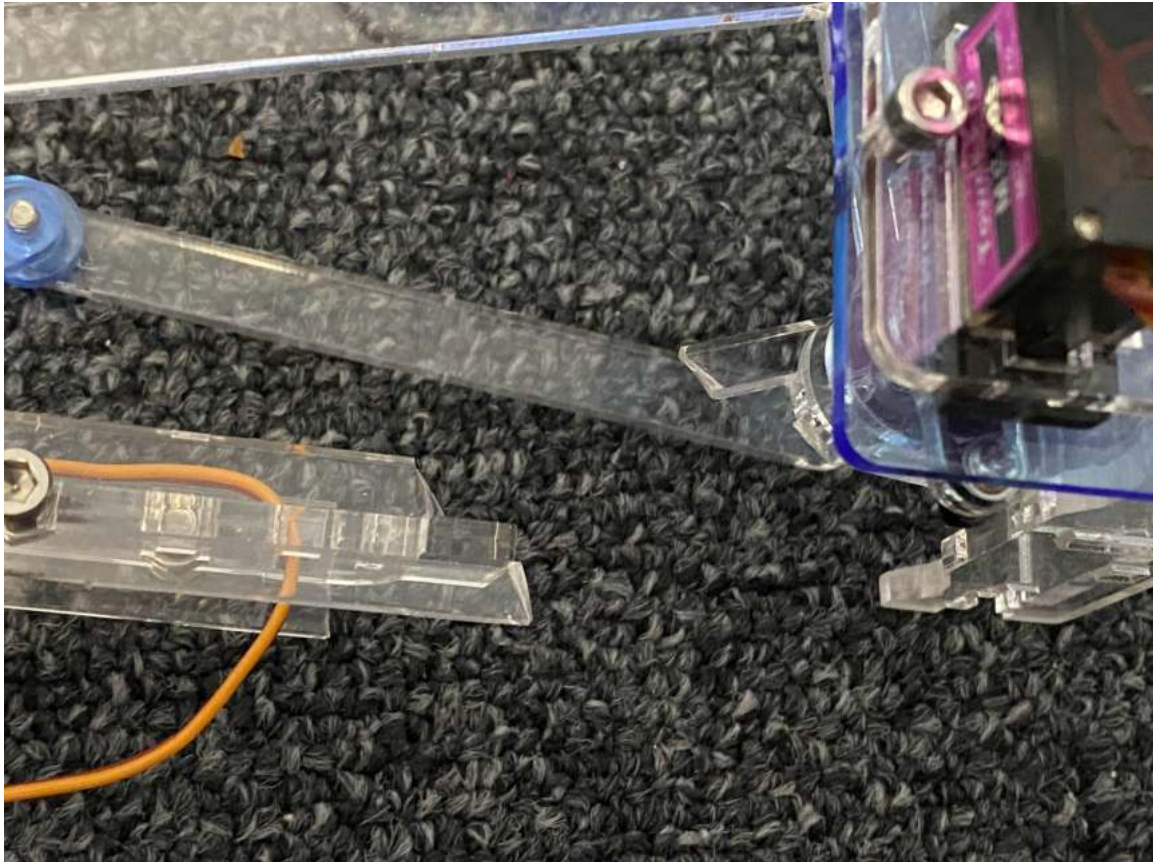


Figure 15: Broken MeArm central arm plastic

The replacement of the knee joint servos on the four MeArms required modification of the servo mounting brackets and the gear's controller mounts. Since the plastic was molded for the specific servos that came with the kit, I had to file approximately 1mm off of the width of the plastic mounting locations and brackets to fit the ever-so-slightly larger servos. After which I later discovered that the gear sizing of the new servos was much smaller than the gears of the previous servos and thus would not fit the plastic gear mount. This required a modification of the gear mounting system so the servo could physically control the leg's movement.

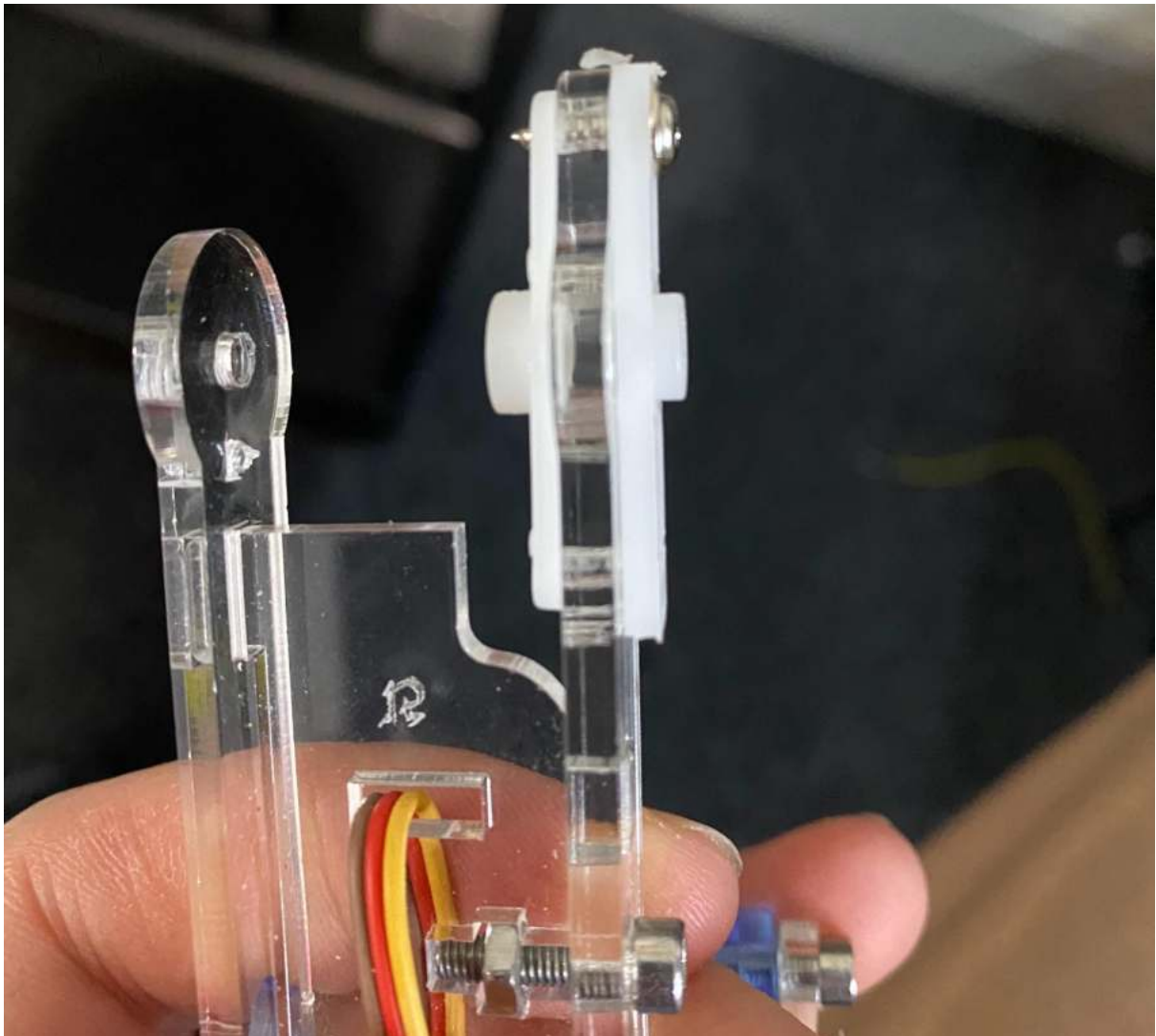


Figure 16: New servo mounting plastic

In the photo the included plastic gear mount is on the left and inside of the plastic femur whereas the plastic mount for the newer EMAX servos is on the right and outside of the plastic femur; note how the plastic circle jutting out of the white pieces is of different sizes. They're connected together by screws which run through the plastic gear mounts where the inside mount is geared to lock the screw in place. With this modification however the left and inner mount would scrape against a connecting part of the plastic used for the legs' foundation.

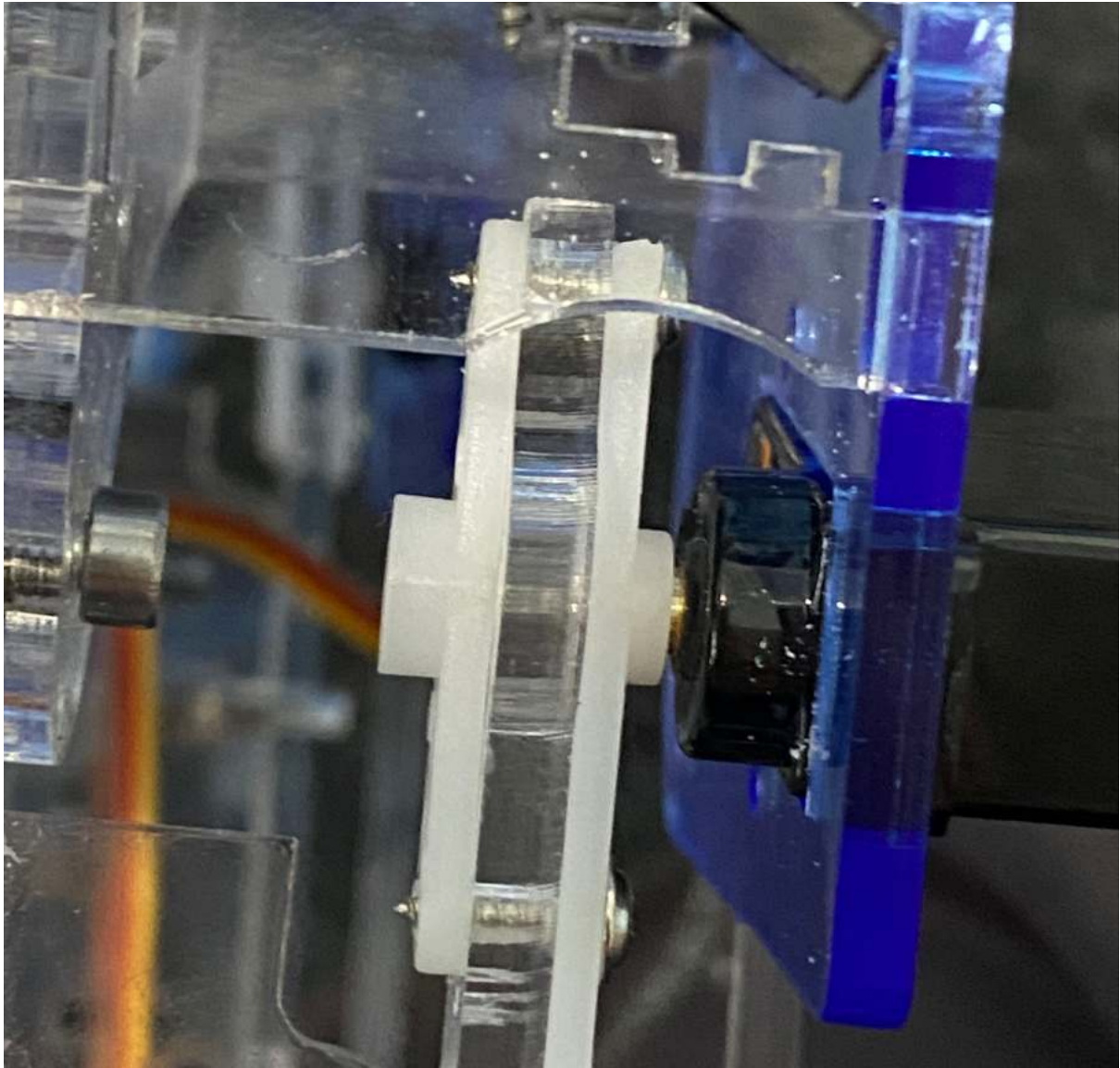


Figure 17: depiction of new mounting plastic interference

3.2.2.2. Chassis

The formation of Flux itself was rather simple after modifying the MeArms. I took two pairs of the MeArms and screwed them together to create two sets of front and hind legs; the two sides of flux were then taped together to secure them – simple yet extremely effective for a prototype. Since I only utilize 2 of the 4 servos in each of the MeArms kits I secured the loose wires with twist ties. Because of the construction of the chassis Flux does not have a central gap where the breadboard and components can be placed. To work around this I attached several smaller breadboards across the chassis via tape and wired my components around, over, and through the gaps in the chassis to the centralized centralized output breadboard which had wires running to the controller from Flux.

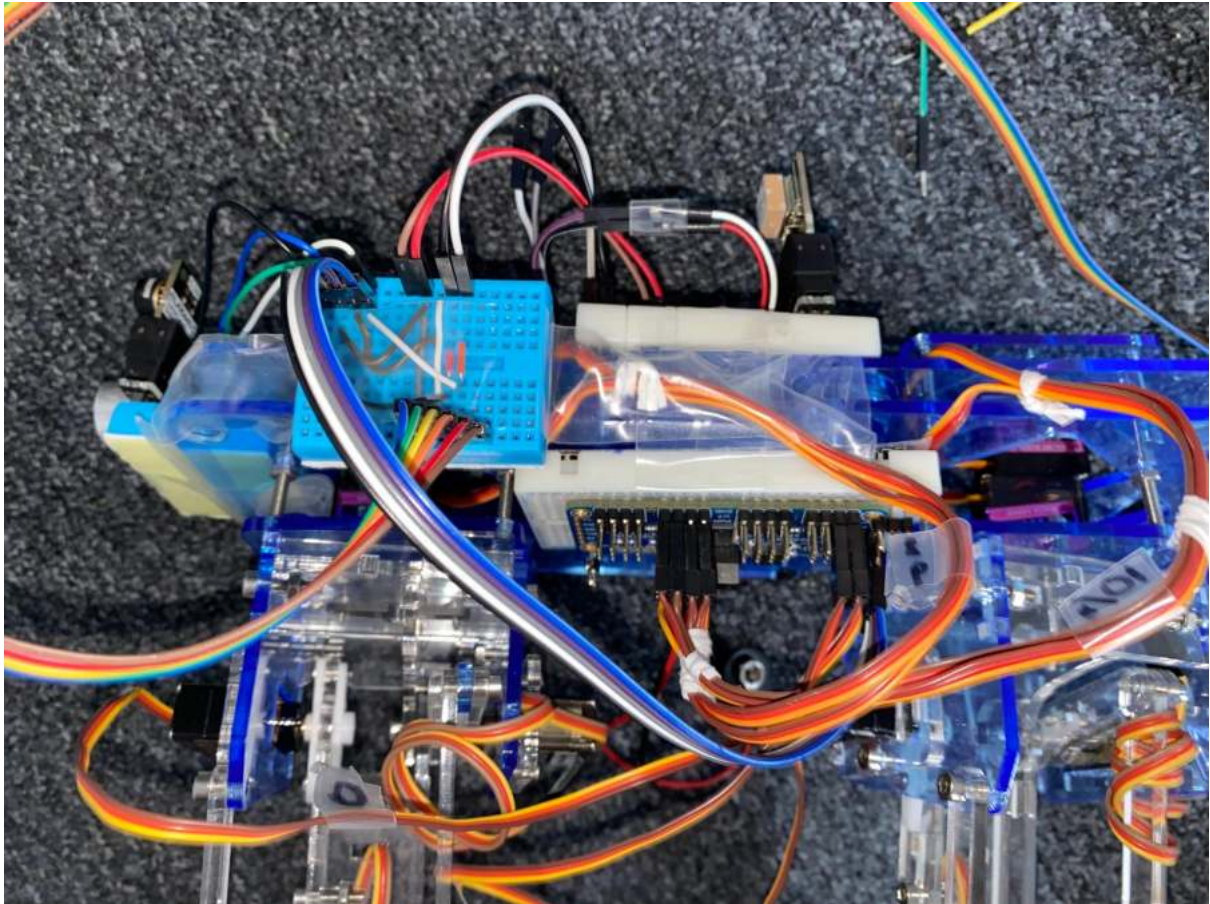


Figure 18: Flux Wiring Overview
note: for more photos see Appendix

3.2.2.3. Controller

The controller is a simple breadboard with the Arduino MKR Wifi 1010, two joysticks, and display attached to it. The wires run from each of the components to various input pins on the Arduino. Both joysticks are wired to analog pins, neither switch connected, and powered; the display is to MOSI, SCK, and 3 additional pins, then powered; a button is wired to control the display's mode; there are four wires running to the back (SDA, SCL, power, ground) which run via a long cable to the centralized board on Flux's back (Figure 18).

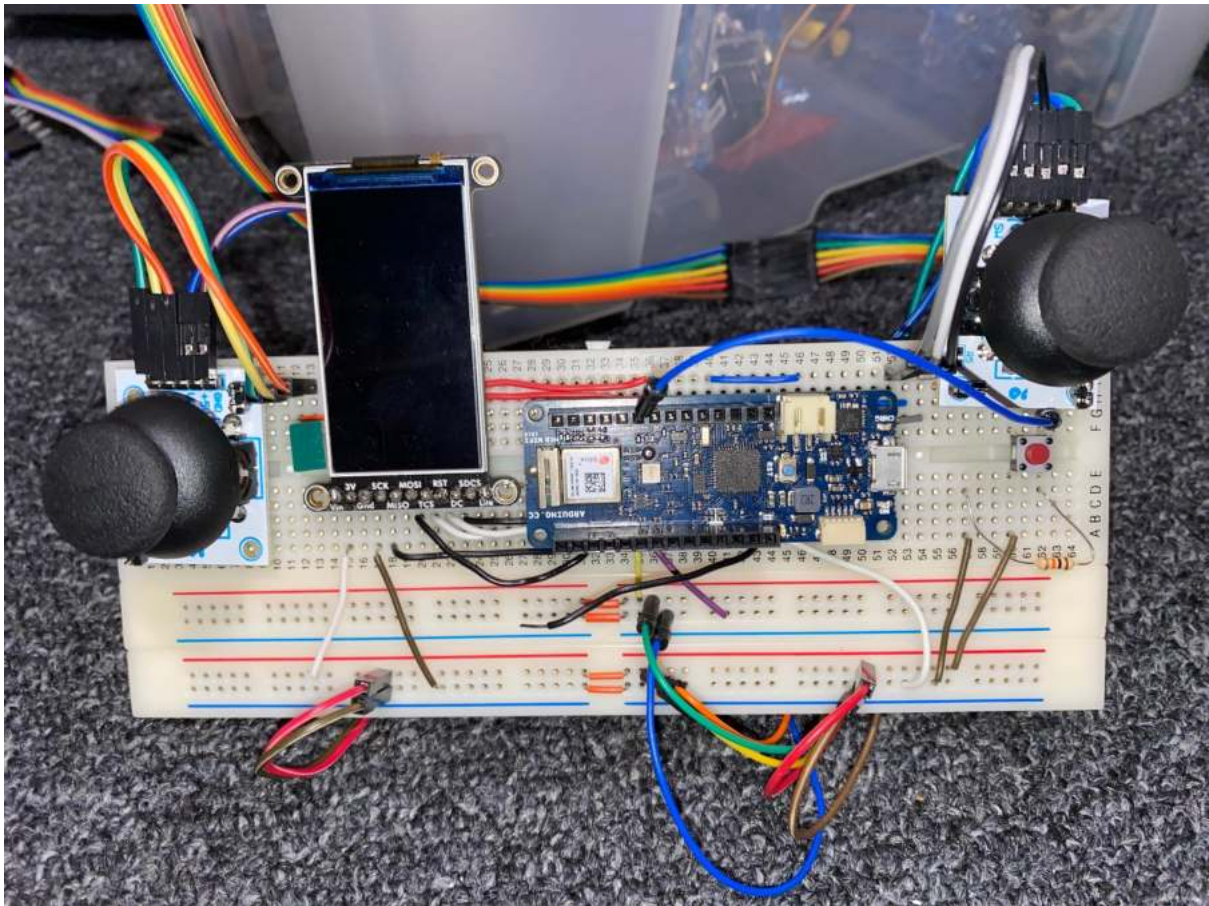


Figure 19: Controller Front

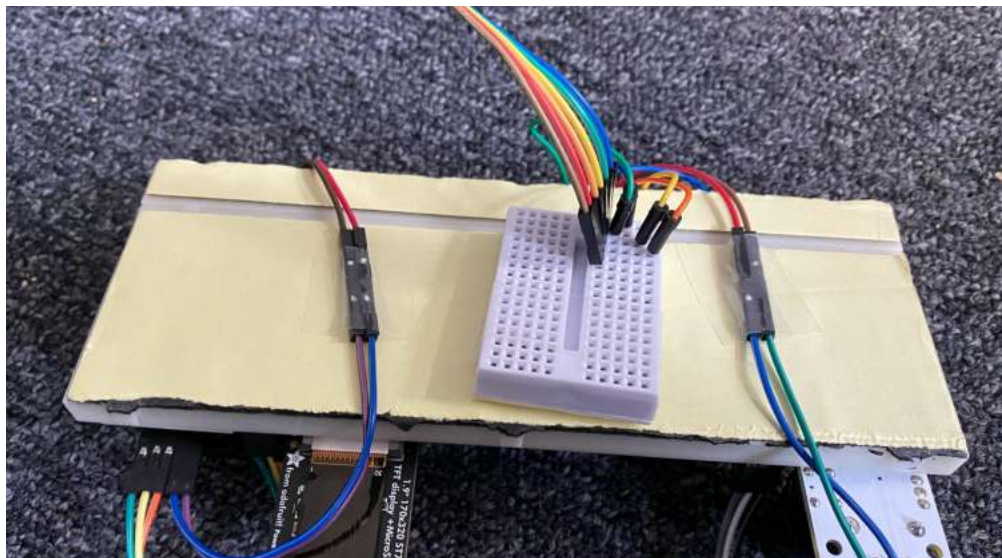


Figure 20: Controller backside

3.2.3. Software Development

3.2.3.1. The Environment

Developing the software side of the project wasn't nearly as difficult as the hardware aspect of it. Since I was developing on an Arduino board, I used ArduinoIDE as the development platform and C++ as the development language. ArduinoIDE produces code in a setup to loop

environment where when the code is first written in C++, compiled, then translated to something the programmable board can understand, and written to it. Once the board is powered on it will first perform all the code written within the setup function as an initialization and then repeat all the code written within the loop function until power is either removed or lost (such as depleting batteries) or the board is reset where it will then act as if turned off and on again and re-perform the setup function and return to the looping function afterwards (CIRCUITO TEAM, 2019). ArduinoIDE offers support for a multitude of libraries and even support for local based ZIP libraries. Most of Flux's components were extremely complex to work with but also included several page lengths of libraries which I had utilized. All of the libraries I used, except the library for the SEN0395, were produced by Adafruit, a company which also produces and distributes a wide variety of arduino and general circuit board components. Even if another company had produced the components themselves, the functionalities of those components would typically come from Adafruit libraries.

3.2.3.2. Component Programming

Components can communicate with programmable circuit boards via I2C, SPI, and UART protocols. SPI is often used for high speed data communication between the component and the board which requires a SCLK, a way for the board to communicate with the component (MOSI), or a way for the component to communicate with the board (MISO) in addition to its own SS port. UART is a protocol in which two UARTs will communicate directly with each other through a transmitter (Tx) and a receiver (Rx) in the form of packets. The I2C protocol utilizes two connection points for communication, a serial clock (SCL) and a serial data (SDA) where multiple devices called slaves can be connected to a singular (or multiple) board called a master; each slave will have its own unique address which the master uses to separate the all the data it receives from its slaves (Hopkins, 2021).

Most of my components utilized I2C for communication except for the ST7789 display which uses SPI, the SEN which uses UART, and the joysticks which are analog devices, meaning they are connected to the Arduino's analog connection pins and act solely as an input device. The libraries of the I2C components would initialize the device into a single variable for me without myself having to work with the components' individual addresses.

Rather than producing code for every component simultaneously I broke it down to one piece at a time. After looking over all the components' libraries the GPS tracker and the 4-in-1 air quality sensor seemed simple and a great place to start as both of these implementations were as simple as utilizing the libraries built in functions to make calls to gather the data.

The SEN0395 human presence detection sensor would send either a 0 or 1 to the Arduino; 0 indicating there was no human presence detected and 1 indicating the presence of a human. With this I would then send a message to the display that would read "Human presence detected" if the component had detected something.

The two HC-SR04 ultrasonic sensors were initialized into an array of values. I would only print these values if they were within a reasonable range; some sensors can realistically only detect up to 4 meters in the best circumstances; my thought being that an operator doesn't need to know if there's an object two or more meters in front of Flux as these sensors were implemented

for close proximity. I would then output the proximity of Flux to the display when something approached within range.

Development for the MLX thermal camera and the accompanying display output took the longest of my components to work with. While the library for the MLX camera did initialize the device and send the values into an array it did not provide any functionality to create a graphical output for a TFT display. It would however print ASCII art if told; useless for this project. I spent a lot of time researching the best way to create a graphical representation of the data from the thermal camera and ultimately settled with the idea of mapping the 32 by 24 pixel array directly onto the 170 by 320 pixel display. Obviously too small to comprehend, I then wanted to multiply the pixel sizes by 10, thereby making the 32 by 24 a 320 by 240 size up but this would cut off some of the display, but potentially offer a more pleasant viewing experience. After several iterations I ended up accepting a square of imaging data rather than a rectangle, accepting that there'd be an empty portion of the screen. This took more advantage of the full 110 degree wide angle camera.

The ST7789 relied on the Adafruit GFX library. I've previously worked with a similar library type in the java language called Java Swing. Both libraries use function calls to draw polygons, lines, color in shapes, create text, and so on. With the GFX library I created a boot up screen to display while the components are initializing during the setup phase and the later formatting of the screen and all the accompanying data. Besides the complexity of the thermal camera visualization, I had wanted to keep everything else simple so the display was able to update as fast as it would receive data. This led me down to develop two main modes for the screen: a data only mode and a visualization for the thermal camera mode. Since I could not determine how to speedup and optimize the rendering of the thermal camera, which took approximately half a second to draw, I decided this was the best approach.

The servo controller works by using a function called "setPWM" which sends pulses to the servos to control how the servos turn. I did a lot of testing in order to calibrate and find the values I needed for my legs' movements as the values don't map evenly to that of a normal 0-180 degrees non-continuous servo. Once I had these values I then implemented my idea for a walking cycle (Figure 21). Initially I only tested using one side as I had snapped yet another leg piece rendering it unusable and later was able to test using three legs.

Constructing the legs themselves was easy enough, and in theory, I know what values I need to input to create movement, or at least what I hope will be movement. To expand upon this, I need to first turn **front left** and **rear right** hip joints forwards, extend the knee joints to contact the ground, turn all four hip joints backwards to propel Flux forwards until the **front left and rear right** legs are straight up and down and the **front right and rear left** legs are behind Flux, then contract the **front right** and **rear left** knee joints and turn the hip joints to be forwards and repeat first steps with **front right and rear right**.

*Figure 21: portion of notes written
note: see appendix "03/31/23 Notes"*

The setup to loop structure of ArduinoIDE made combining all of the code extremely simple. I just copied what I needed from one file to the main program and everything worked coherently

together after importing everything. I began taking the outputs for each of my components and displaying them using only text indicators with labels.

Inside the main file I then added code for the legs to respond to the movement of the joysticks where if the right joystick was pushed forwards Flux would perform a forwards walk, if it was pushed back Flux would perform a backwards walk cycle. If the left joystick was pushed forwards Flux would go from its neutral position to a sitting position and if it was pushed back Flux would go from its neutral position to a standing position.

3.2.3.3. Latency and Response Times

One of the slowest components of Flux as a whole is the response time of the display. The issues occur when the display is put into the camera's view where it takes approximately half a second to refresh and draw the camera's captured data. The camera itself has no issue constantly capturing and update the values in its array but the display fails to update as fast as the program is trying to draw to the screen which causes a slow sweep of the screen draw to occur thus sometimes intermixing data values and outputting a mixture of two or more captures of the camera's data resulting in a somewhat blurred and confusing output for that second. This issue can easily be fixed by replacing the screen and using something that has a faster response time. The display was rated for 60Hz but the highest I've seen it initialize at was 27Hz. Aside from the visualization of the thermal camera everything else seemed to respond in millisecond response times which was normal for the components, especially with some of the sizes of the libraries of the more complex components like the GPS.

3.3. Testing

3.3.1. Free Standing Testing

The testing performed for the free standing ability occurred over the development of Flux, first starting with nothing attached to the chassis and the final tests performed while all of the breadboards and additional components were attached to Flux.

Input	Expected Outcome	Actual Outcome	Next Steps
Original MeArm Servos (GoTeck brand) as elbow joints – unpowered	Weight distribution of all 4 servos will allow standing	Collapsed after I removed my hand from the middle	Power servos
Original MeArm Servos (GoTeck brand) as elbow joints – powered	Powering the servos will lock them in place allowing Flux to stand	Slowly collapses this time but the servos sound like they're being damaged	Replace servos
EMAX servos as elbow joints – unpowered	Higher stall torque will allow standing capability	Flux is able to stand on its own. Even while missing 1 leg	

3.3.2. Knee Joint Movement

As previously mentioned the function “setPWM” does not directly correlate to ranges 0-180 degrees. The testing was used to calibrate my minimum and maximum values for setPWM where it would set to minimum, delay 1 second, then set to maximum testing 1 leg only as all legs will have the same values. After the tests were concluded I determined a neutral/midpoint for the legs to be exactly half of the min and max values: 250.

Input	Expected Outcome	Actual Outcome	Next Steps
Min: 0 Max: 500	Leg will go from a sitting position to a standing position	Leg was getting stuck on the plastic piece in figure 17 and breaking	increase Min value
Min: 300 Max: 500	Leg will go from a sitting position to a standing position	Leg was extended for Min and overextended and breaking for Max	decrease Min Max value
Min: 150 Max: 300	Leg will fully extend and might hit plastic when sitting	Leg was fully extended and hit plastic when sitting, causing servo stress	increase Min value
Min: 200 Max: 300	Leg will go from a sitting position to a standing position	Leg fully stands and sits with no issues	

3.3.3. Hip Joint Movement

The same testing for the knee joint was done with the hip joint.

Input	Expected Outcome	Actual Outcome	Next Steps
Min: 0 Max: 500	Hip will swing forwards and swing backwards	Hip is overextended forwards and breaking and swings back and knocks into rear leg	increase Min decrease Max
Min: 100 Max: 300	Hip will swing forwards and swing backwards	Hip swings too far forwards and not far enough backwards Will use 100 for a “reach” value	increase Min increase Max
Min: 200 Max: 350	Hip will swing forwards and swing backwards	Hip swings back perfectly, but not far enough forwards	Small decrease Min
Min: 175 Max: 350	Hip will swing forwards and swing	Hip swings forwards and backwards	Determine a Midpoint

	backwards		
Mid: 300	Hip will rest at a neutral/midpoint for the legs	Hip is slightly crooked backwards	minimal decrease of Mid
Mid: 295	Hip will rest at a neutral/midpoint for the legs	Hip rest at a neutral/midpoint.	

3.3.4. Walking Testing

I wrote all the code for the walking cycle at once without testing it based on my idea for leg movement. The testing was then adjusting the code.

```

495
496 > void walkCycleSideForward(int FrontLeg, int FrontHip, int RearLeg, int RearHip, bool isFrontRightSide){-
571 }
572 void walkForwards(){
573     //move one side then the other side
574     walkCycleSideForward(FrontLeftLeg, FrontLeftHip, RearRightLeg, RearRightHip, false);
575     walkCycleSideForward(FrontRightLeg, FrontRightHip, RearLeftLeg, RearLeftHip, true);
576     //thus completing a "walk cycle"
577 }
578 > void walkCycleSideBackwards(int FrontLeg, int FrontHip, int RearLeg, int RearHip, bool isFrontRightSide){
653 }
654 |
655 void walkBackwards(){
656     //move one side then the other side
657     walkCycleSideBackwards(FrontLeftLeg, FrontLeftHip, RearRightLeg, RearRightHip, false);
658     walkCycleSideBackwards(FrontRightLeg, FrontRightHip, RearLeftLeg, RearLeftHip, true);
659     //thus completing a "walk cycle"
660 }

```

Figure 22: function calls for walk cycle
note: see appendix for full code

Input	Expected Outcome	Actual Outcome	Next Steps
"walkCycleSide" function – only have one side connected to Flux with 2 legs	Front Left Leg (<i>and in theory Rear Right Leg</i>) will swing forward, extend the leg as it swings backwards then recenter and extend back to "neutral" position	Same as expected	Adjust code to call the cycle for one pair of legs then the other pair of legs
"walkForwards" function – only have one side connected to Flux with 2 legs	Front Left Leg will do the cycle, then Rear Left Leg will do the cycle	As expected	Create Backwards Cycle
"walkBackwards" function – only have one side connected to Flux with 2 legs	First Front Left Leg will swing backwards, extend the leg as it swings forwards, then retract the	As expected	Attach other leg and test the cycles

	leg, recenter, and extend the leg to neutral position then Rear Left Leg will do same		
"walkForwards" function – Front Left, Rear Left, and Rear Right legs attached (front Right leg indefinitely broken)	Front Left Leg and Rear Right Leg will swing forward, extend the leg as it swings backwards then recenter and extend back to "neutral" position and repeat with Rear Left and Front Right (if attached)	Front Left Leg worked – Rear Right leg's hip swung the wrong direction	Add boolean to determine the side the front leg is on – this then determines what hip (rear or front) needs to have its swing inverted.
"walkForwards" function with NEW "walkCycleSideForward" function call containing a bool – Front Left, Rear Left, and Rear Right legs attached (front Right leg indefinitely broken)	Front Left Leg and Rear Right Leg will swing forward, extend the leg as it swings backwards then recenter and extend back to "neutral" position and repeat with Rear Left and Front Right (if attached)	As expected	Copy Paste and Adjust backwards cycle to account for new boolean variable and hip inversion.
"walkBackwards" function with NEW "walkCycleSideBackwards" function call containing a bool – Front Left, Rear Left, and Rear Right legs attached (front Right leg indefinitely broken)	Front Left Leg and Rear Right Leg will swing forward, extend the leg as it swings backwards then recenter and extend back to "neutral" position and repeat with Rear Left and Front Right (if attached)	As expected	

3.3.5. Controller Testing

Testing of the controller was mostly testing the Joysticks' values and the button to switch modes.

Input	Expected Outcome	Actual Outcome	Next Steps
Read left and right joystick X and Y values	Joysticks range from 0 to 1025 so I expect a neutral range around 550-650 give or take	Neutral values ranging from 700 to 800	Add external power source – ALL of my components were running on the Arduino's single 5V
Read left and right joystick X and Y values	expect a neutral range around 550-650 give or take	Neutral range achieved	Add button

Wired button pin 7, no resistor	Will read 0 when not pressed and 1 when pressed	No solid value, constantly changing between 0 and 1	Change pin – I had experienced issues with pin 7 and other devices too
Wired button pin 0, no resistor	Will read 0 when not pressed and 1 when pressed	No solid value, constantly changing between 0 and 1	Add a resistor
Wired button pin 0 resistor	Will read 0 when not pressed and 1 when pressed	As expected	

4. Conclusion

The project as a whole can widely be considered successful despite the handicaps Flux requires to operate and the lack of goals achieved. As stated in my aims and objectives for the project, I was attempting to create a prototype of a quadrupedal robot which was self standing and can move forwards and backwards. In addition I also set out to improve my knowledge of robotics and 3D design as well as to apply good engineering practices when creating Flux.

This project has widely given me the opportunities I wanted to do just that. I was able to acquire a basic understanding of what's required to do 3D modeling and design in the Blender environment. I developed a prototype which will serve as a basis for my future iterations of Flux, FluxV2 for example, and I was able to experiment with camera feeds and output them to a device like a display, which I was very interested in when beginning this project.

5. Critical Evaluation

5.1. Project Evaluation

Throughout the entire research and development process of the project I was constantly improving upon my knowledge and interest in the field of robotics and discovering something which I truly enjoy doing. The project as a whole has provided me with great insights into what it requires to create operable robotics, especially anything that can move no matter how simple the robotic; movement in robots is an extremely tedious undertaking. I certainly endured more than my fair share of frustration, anger, and stress while developing this project too. The constant snapping of plastic parts of the MeArms kits would mean having to wait up to 4 or 5 days to get back into the university class where the spare and broken MeArm kits were stored to make replacements; this definitely took a major toll on the quality of the deliverable itself.

Researching the project spanned the longest length of time, between researching what design would work best for search and rescue robots, other existing quadrupedal robots, what components I should use, and how the component libraries would work. I definitely could have spent more time researching and experimenting with the 3D modeling aspect of my project so I

could have achieved a better operating leg system, most likely one with less overall failures; however I'm still content with my final deliverable and did still learn what it takes to do 3D modeling, just the very basics of it and nothing to the level I would have needed to design and print a leg model. Designing Flux really didn't take me nearly as long as I spent researching everything else. From the moment I had the idea of a quadrupedal robot I knew approximately what I wanted the design to look like. I had always strived to create something akin to Boston Dynamic's Spot, at least in the long run, but for a prototype I wanted something more functional. Even with the design I drew up originally, I will continue to work off of that base for future iterations. Regarding future design however, the most work will need to be done with the legs which I'll expand upon later.

The actual development and work of the project was what I dedicated the most actual time on. I had undoubtedly poured more hours into physically coding and constructing Flux than I had spent on researching but I was researching for a few months whereas I completed Flux in about a month or so. Development was at times overwhelming knowing an approaching deadline was hanging over my head with the amount of work I felt I had left to do. Ultimately what really helped me progress was focusing on each of the individual components' code, ensuring that one piece worked properly and would do what I needed it to, save that code, and continue on to the next component while starting from scratch. As mentioned previously this was a great way to go about turning what felt like a very daunting and complicated project into smaller, manageable pieces to later bring together into one.

As for the ethics of this whole project, I initially had thought to take an approach towards developing a search and rescue type of robot, one that's designed to rescue individuals. Had a future implementation of this project been used to rescue individuals, what happens if there's a failure? People could die. While on the one hand the usage of such a robot can save the lives of the individuals who risk theirs to save others, in the event of a failure the risk is then potentially worse than if they had not used one in the first place. This leads to an ethical dilemma where one has to weight the value of people's lives: do you risk using a robot that can save the lives of those who are knowledgeable in rescuing individuals, those who can then go on to save more lives overall or do you risk those professionals' lives to save a few individuals now, knowing that they can be injured or killed and unable to save others in the future?

5.1.1. Successes

My biggest achievement of the project was the code behind the movement of the legs. Determining what servos needed moving when and to what degree or pulse as the library documents it, was incredibly daunting at first. For most of the development I was actually operating only two legs with Flux on its side because I had snapped the plastic femur of one of the legs rendering it unusable and was unable to get a replacement.

Another notable success is the thermal camera. I know a few of my processors were rather impressed at the demonstration of the operating thermal camera. I'm quite proud of the work I did, in addition to coming up with the two modes of output for it, one where the view was the whole 32x24 array and another that took up the entire screen, but offered a lesser overall view.

5.1.2. Failures

There were a few failures throughout the development of Flux. Firstly, the leg design is terrible. It barely supports its own weight and can't even move without being held up, much less actually walking. The plastic of the legs also kept breaking which meant that I ended up not having a front right leg for the end product because there were no replacement parts. As mentioned earlier, this was a huge recurring problem that just tanked the possibility of Flux.

I wasn't able to implement the HC-SR04s how I wanted, which was to have one on both sides of Flux, one in the front, and one at the back. Additionally I had wanted to display the sensors' values around a rectangle that would represent Flux so the operator knew which sensor was detecting something in close proximity and where that was located.

Another hardware failure was the 4-in-1 air quality sensor which I found out was broken throughout my development. I had the library's example code to get the sensor working and it would only ever tell me that the sensor could not be found. This meant that either the wiring was incorrect or the device was faulty. My wiring was not correct as I could insert the GPS or MLX cam into the same slot and load up those components' code and that would work. Additionally with the GPS sensor, it needed to be calibrated and I simply could not find any documentation that would discuss how to calibrate it, so my values for the GPS device were a longitude and latitude of 0.

5.2. Future Implementation

5.2.1. Leg and Body Design

In future implementations of Flux, a Flux V2 so to speak, the leg design greatly needs improving to accommodate its own weight. I believe creating something akin to what I had sketched in the drawing "Leg Design V2" would create a more structurally sound support for the body of Flux V2. Additionally, creating my own legs from scratch allows me to utilize whatever motors I'd like for the legs without modifying the overall design, but only altering the mounting location for said motors. In Flux V2 I'd like to have a design that requires only two motors, one that acts as knee joint and one that serves as a hip joint; I mind you this is indeed similar to Flux V1 (the delivered implementation) where one servo moves the whole leg and one moves the lower leg but the placement of the servos in Flux V1 is, in my opinion, less than ideal. I would design something where the motors sit inside the parts they are to turn, truly acting as the joints themselves. Regarding the motors, during my research I came across a more powerful type of motor called a stepper motor instead of a servo motor which I've previously touched on as to why I did not utilize them. As a refresher, stepper motors are capable of more torque than the small servo motors and would more than likely be capable of lifting a heavier Flux design in the future which is why I want a leg design capable of holding and supporting more powerful stepper motors. As for the chassis or body of Flux, something that's more standalone and something that would allow for swapping in and out the legs of Flux is the next steps for the body; something that is independent of the legs whereas the delivered implementation of Flux is entirely dependent on the legs as legs themselves but also as Flux's centralized chassis where the components rest. This allows me to easily alter the leg design time and time again without needing to entirely rework my design for Flux V2.

5.2.2. Hardware Overhaul

In Flux's V2 I'll be removing most of the external components and prioritizing the ability to walk and move over fancy sensors that might be useful in search and rescue. I honestly believe that I focused too much on external components in the development of Flux V1 and that that had cost me some quality of work I put towards the legs and movement of Flux. I was focused on creating something that was complex and eye-catching rather than something that was fully functional. While I need to do more research on the topic, I'm open to the idea of swapping the Arduino MKR WiFi 1010 to something more powerful like a Raspberry Pi which would allow a unified place to work as Raspberry Pi's can have their own operating system, simply remove the wires from the Raspberry Pi and hook it up to a screen, adjust the code for the legs, plug it back into Flux, and rewire the components; something that's incredibly simple in theory. A rather large negative to this however is needing to learn the language that Raspberry Pi's use to interact with their external components; but this is contrasted by the sheer processing power Raspberry Pi's have over Arduino and similar programmable boards. Additionally in future interactions, something after Flux V2, I would really like to implement a direct feedback camera and improve upon the implementation of the visualization of the MLX thermal camera. Wireless communications is also an important feature to add in the future. I'd like to be able to have a standalone, no wires attached, version of Flux that sends the camera and any additional component data to a controller that the operator is using. While I did do a very simplified version of the controller, in future iterations the controller would house a larger display, and additional buttons. Design wise, something that would look similar to a Steam Deck is actually what I had initially imagined when I set out to create a controller but ended up delivering something lackluster.

6. References

Agile methodology for software development. Amplifyre. (n.d.).

<https://www.amplifyre.com/articles/agile-methodology-for-software-development>

Bosch Sensortec. (n.d.). *BME688 Digital low power gas, pressure, temperature & humidity sensor with AI.*

<https://cdn.shopify.com/s/files/1/0174/1800/files/bst-bme688-ds000.pdf>

Boston Dynamics. (2018, May 10). *Spot Autonomous Navigation.* YouTube

https://www.youtube.com/watch?v=Ve9kWX_KXus

Boston Dynamics. (2022, May 3). *What's New in Spot | Boston Dynamics.* YouTube

<https://www.youtube.com/watch?v=zIdyhGyXcUg>

CIRCUITO TEAM. (2019, January 1). *Everything you need to know about arduino code.*

circuito.io blog. <https://www.circuito.io/blog/arduino-code/>

Deep Robotics [DeepRobotics]. (2021, March 9). *The Quadruped Robot Jueying Mini Lite.*

YouTube <https://www.youtube.com/watch?v=IBUb910ktpg>

Deep Robotics [DeepRobotics].(2022, July 22). *Jueying X20 Robot Dog Hazard Detection & Rescue Solution*. YouTube <https://www.youtube.com/watch?v=Q8KWZB4kgTY>

Deep Robotics. (n.d.). *Deep robotics - the global leader in quadruped Rob.*
<https://www.deeprobotics.cn/en/index.html>

DFRobot. (n.d.). *MmWave radar sensor Arduino-Human Presence Detection Wiki*. DFRobot.
https://wiki.dfrobot.com/mmWave_Radar_Human_Presence_Detection_SKU_SEN0395

Emax ES08MA II 12G Mini Metal Gear ANALOG SERVO FOR RC Model&Robot PWM Servo.
Emax. (n.d.).
<https://emaxmodel.com/products/emax-es08ma-ii-12g-mini-metal-gear-analog-servo-for-rc-model-robot-pwm-servo>

Hopkins, J. (2021, December 1). *I2C vs SPI VS UART – introduction and comparison of their similarities and differences*. Total Phase Blog.
<https://www.totalphase.com/blog/2021/12/i2c-vs-spi-vs-uart-introduction-and-comparison-similarities-differences/#:~:text=Unlike%20communication%20protocols%20like%20I2C,send%20and%20receive%20the%20data.>

Industries, A. (n.d.). *Adafruit 16-channel 12-bit PWM/Servo driver - I2C interface*. adafruit industries blog RSS. <https://www.adafruit.com/product/815>

Last Minute Engineers. (2022, June 11). *How HC-SR04 ultrasonic sensor works & how to interface it with Arduino*. Last Minute Engineers.
<https://lastminuteengineers.com/arduino-sr04-ultrasonic-sensor-tutorial/>

Legacy robots. Boston Dynamics. (n.d.). <https://www.bostondynamics.com/legacy>

MeArm kit for micro:bit - nuka cola blue. MeArm Store. (n.d.).
<https://shop.mearm.com/products/mearm-kit-for-micro-bit-nuka-cola-blue>

Microchip Technology Inc. (n.d.). *ATECC508A complete data sheet - arduino*.
https://content.arduino.cc/assets/mkr-microchip_atecc508a_cryptoauthentication_device_summary_datasheet-20005927a.pdf

REHKOPF, M. A. X. (n.d.). Scrum sprints: Everything you need to know. Scrum sprints: Everything you need to know | Atlassian.
<https://www.atlassian.com/agile/scrum/sprints#:~:text=%22Sprints%20make%20projects%20more%20manageable,They're%20not.>

Robotics/types of robots/wheeled. Wikibooks, open books for an open world. (n.d.).
https://en.wikibooks.org/wiki/Robotics/Types_of_Robots/Wheeled

Robotpark. (n.d.). *Tracked Robots*. TRACKED Robots. from
<http://www.robotpark.com/academy/all-types-of-robots/wheeled-robots/tracked-robots/>

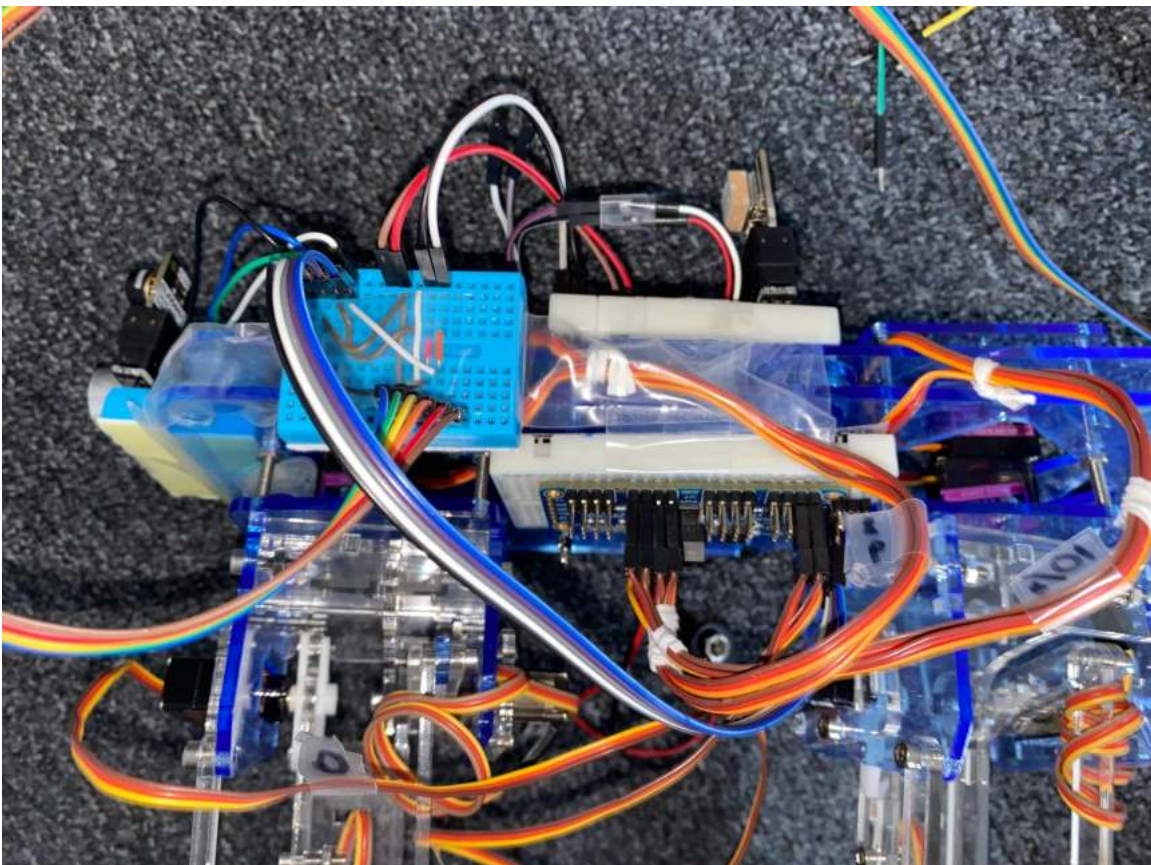
The Arduino Team. (n.d.). *Using the arduino software (IDE): Arduino documentation*.
Arduino Documentation | Arduino Documentation.
<https://docs.arduino.cc/learn/starting-guide/the-arduino-software-ide>

Wikimedia Foundation. (2023, March 8). *Legged robot*. Wikipedia.
https://en.wikipedia.org/wiki/Legged_robot

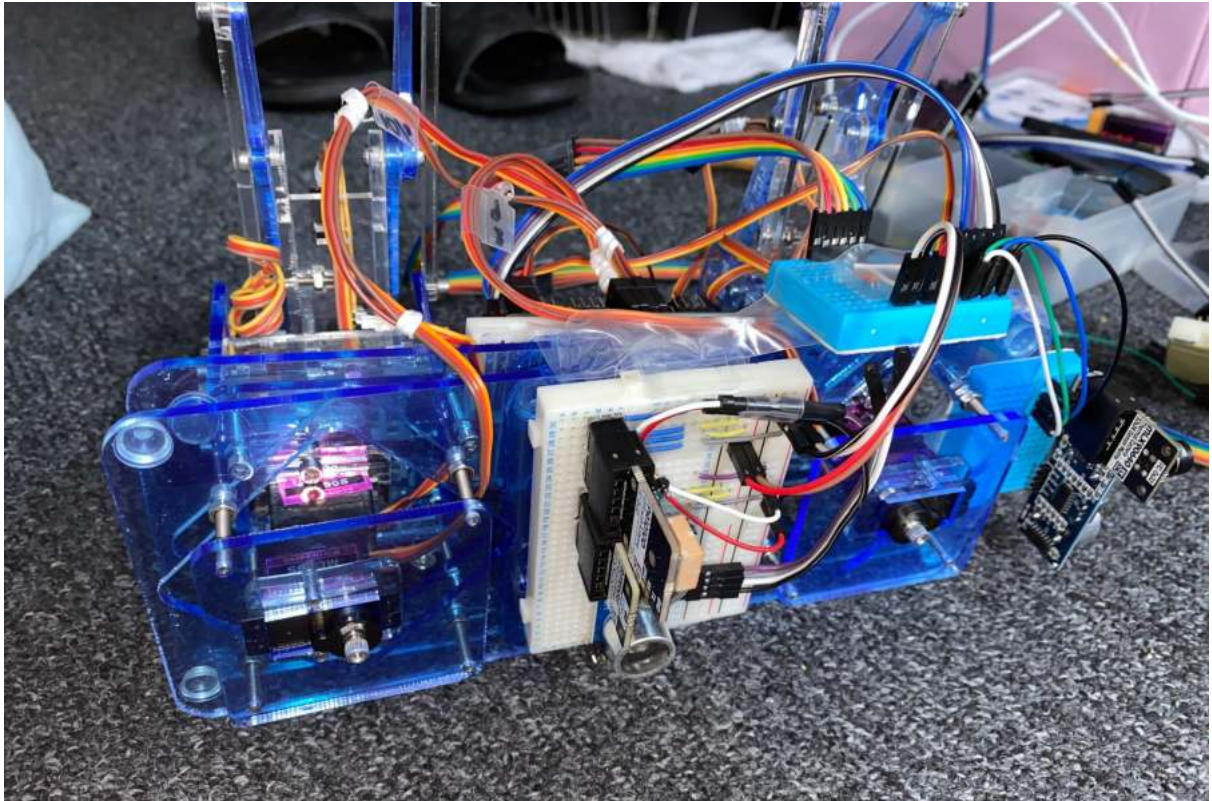
7. Appendix

Flux Chassis Pictures

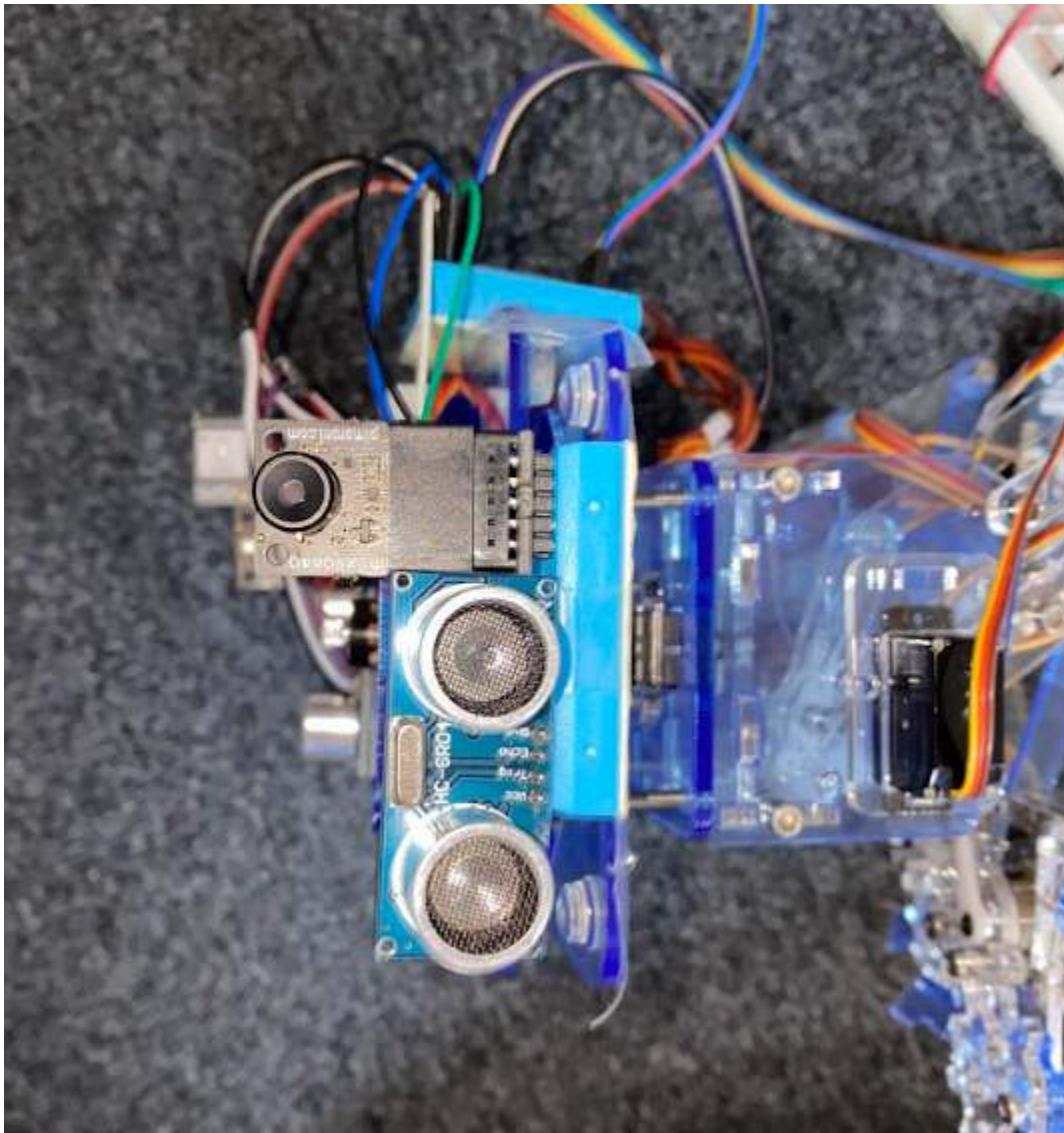
Photos in various stages of development in no particular order.



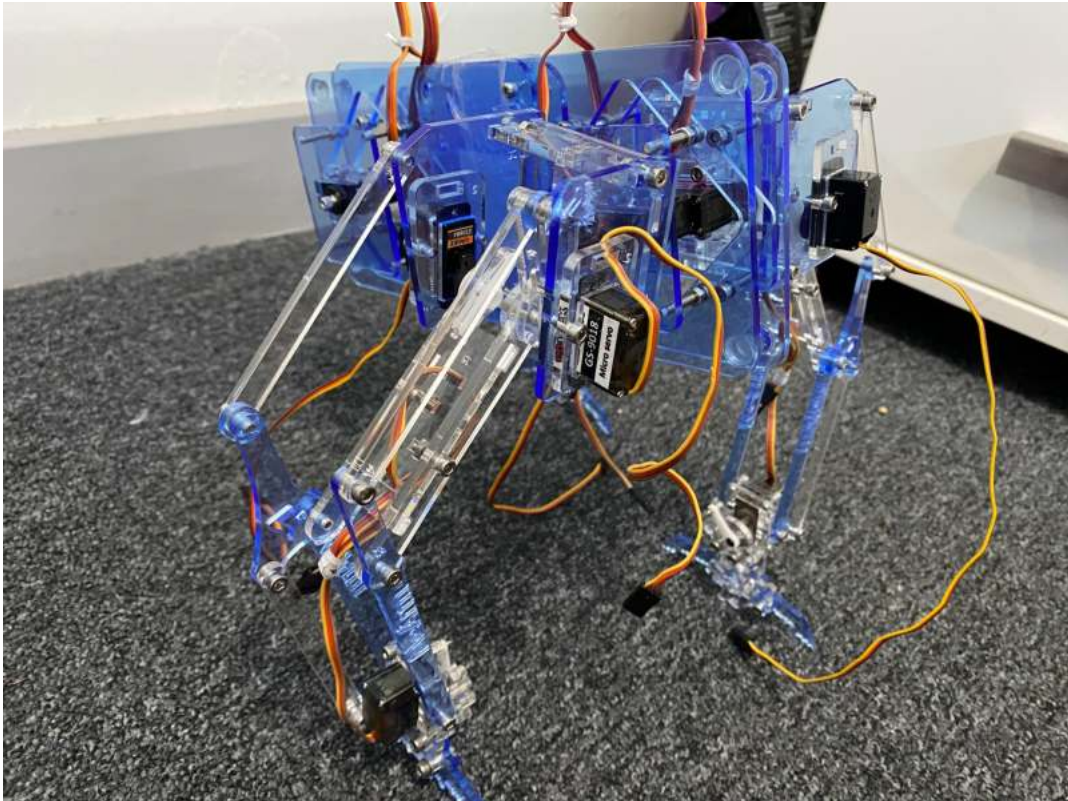
Flux Near Final wiring - Top overview



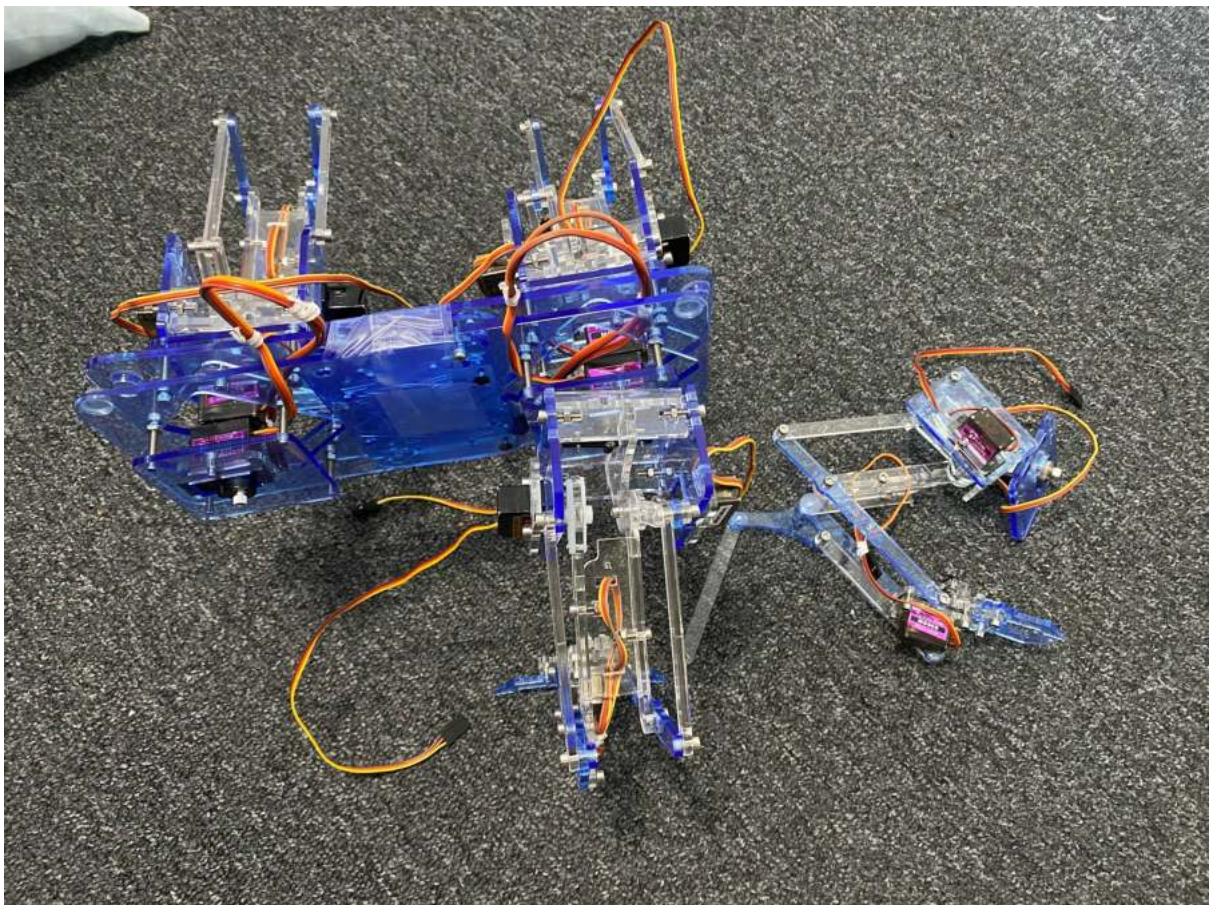
Flux Near Final Wiring - Right side



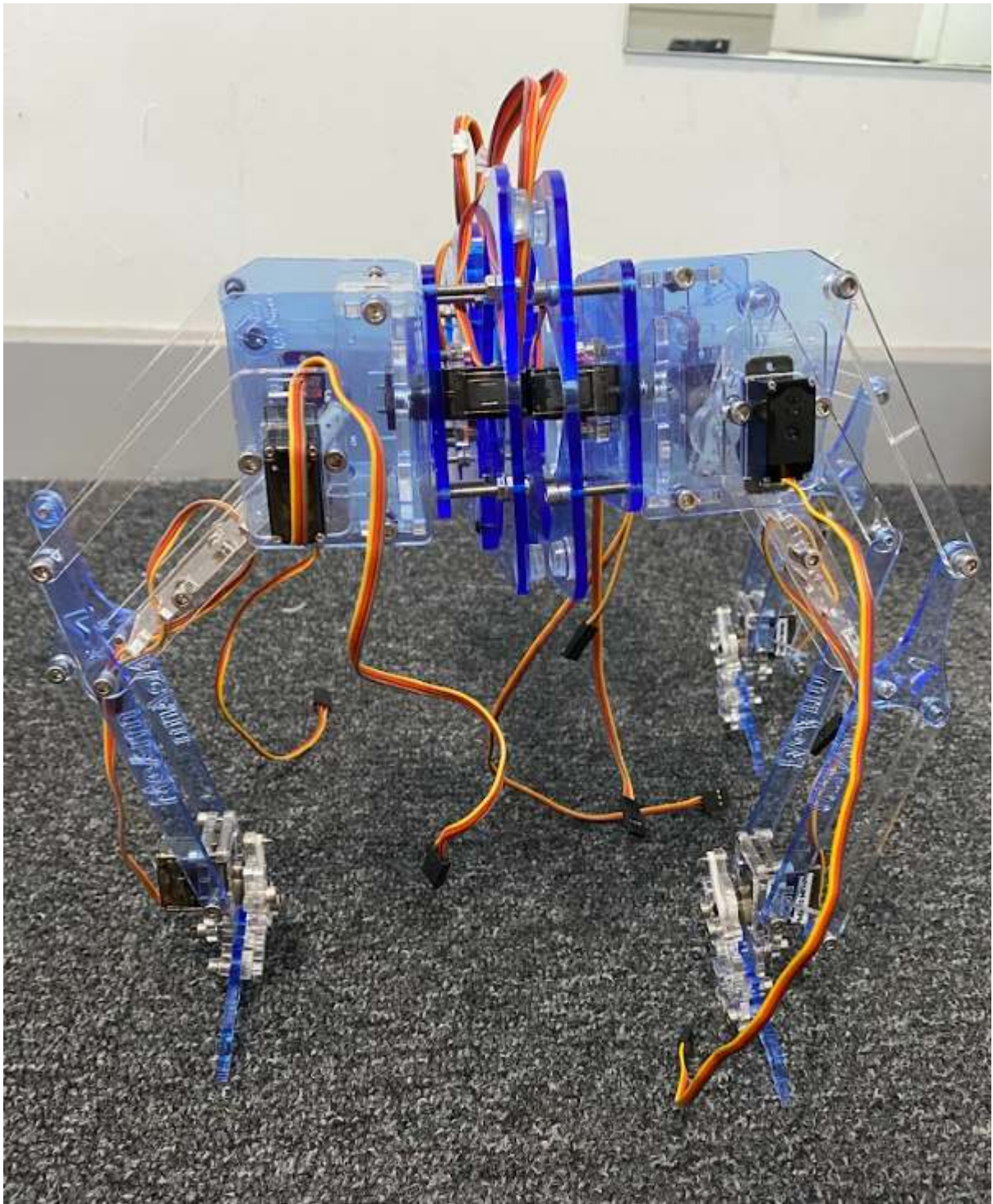
Flux Near Final Wiring - Front camera & sensor



Flux Base Chassis - Missing rear leg



Flux Base Chassis - Depicting broken leg

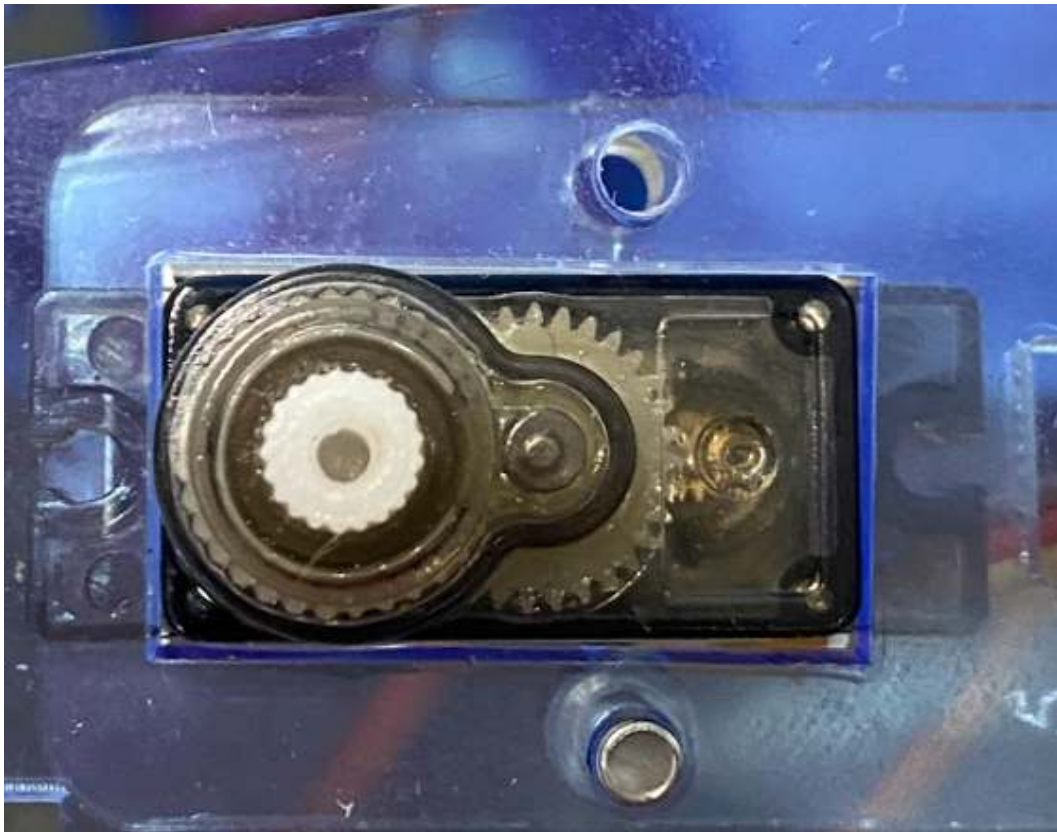


Flux Base Chassis - Straight on

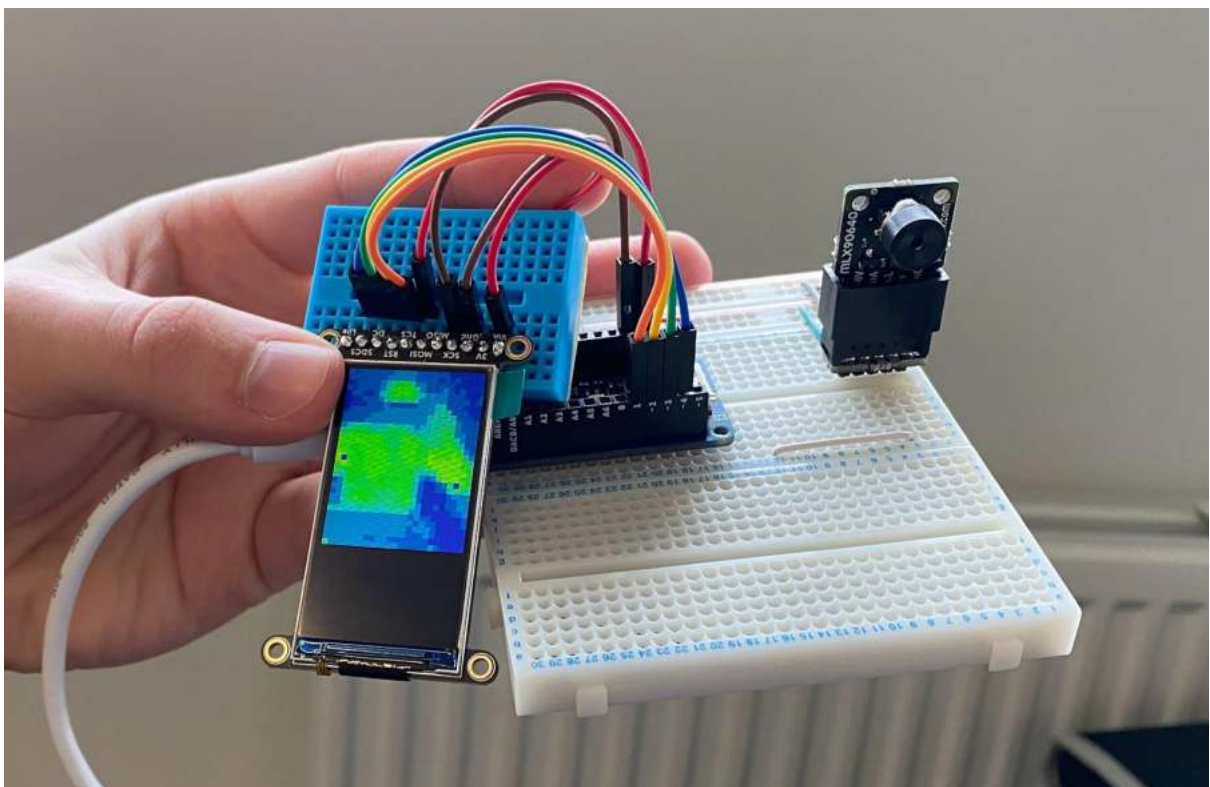
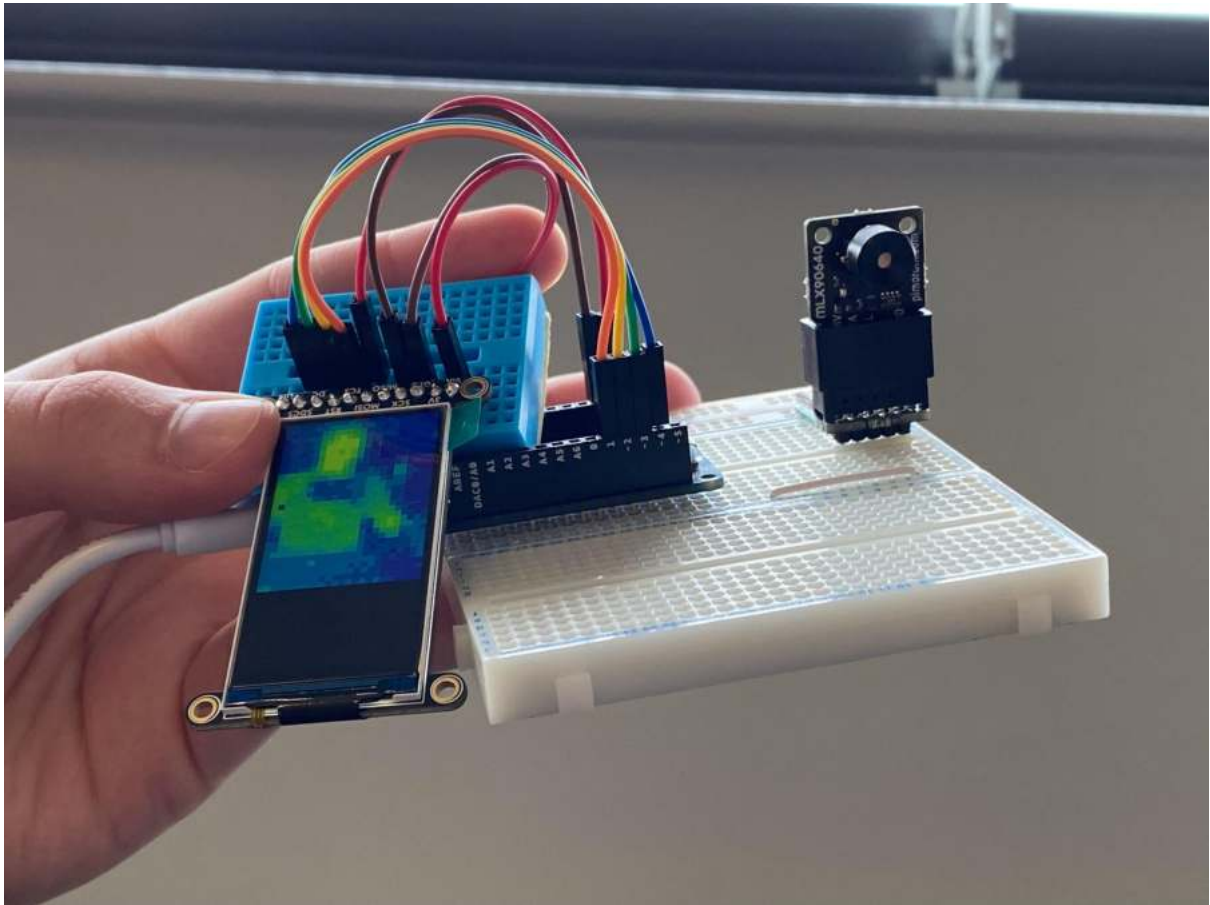


Photo of grinded plastic to fit EMAX servo – Note the slight curve in the plastic and scratch marks

Compared to the non-grinded side with no imperfections



MLX Thermal Camera output



03/31/23 Notes

These are the notes I wrote one day after various experimentations and testing of components. My ideas for the other controller (denoted as right, but in practice was left) were never implemented and instead when pushed forwards Flux would sit and if pushed back Flux would stand.

03/31/23

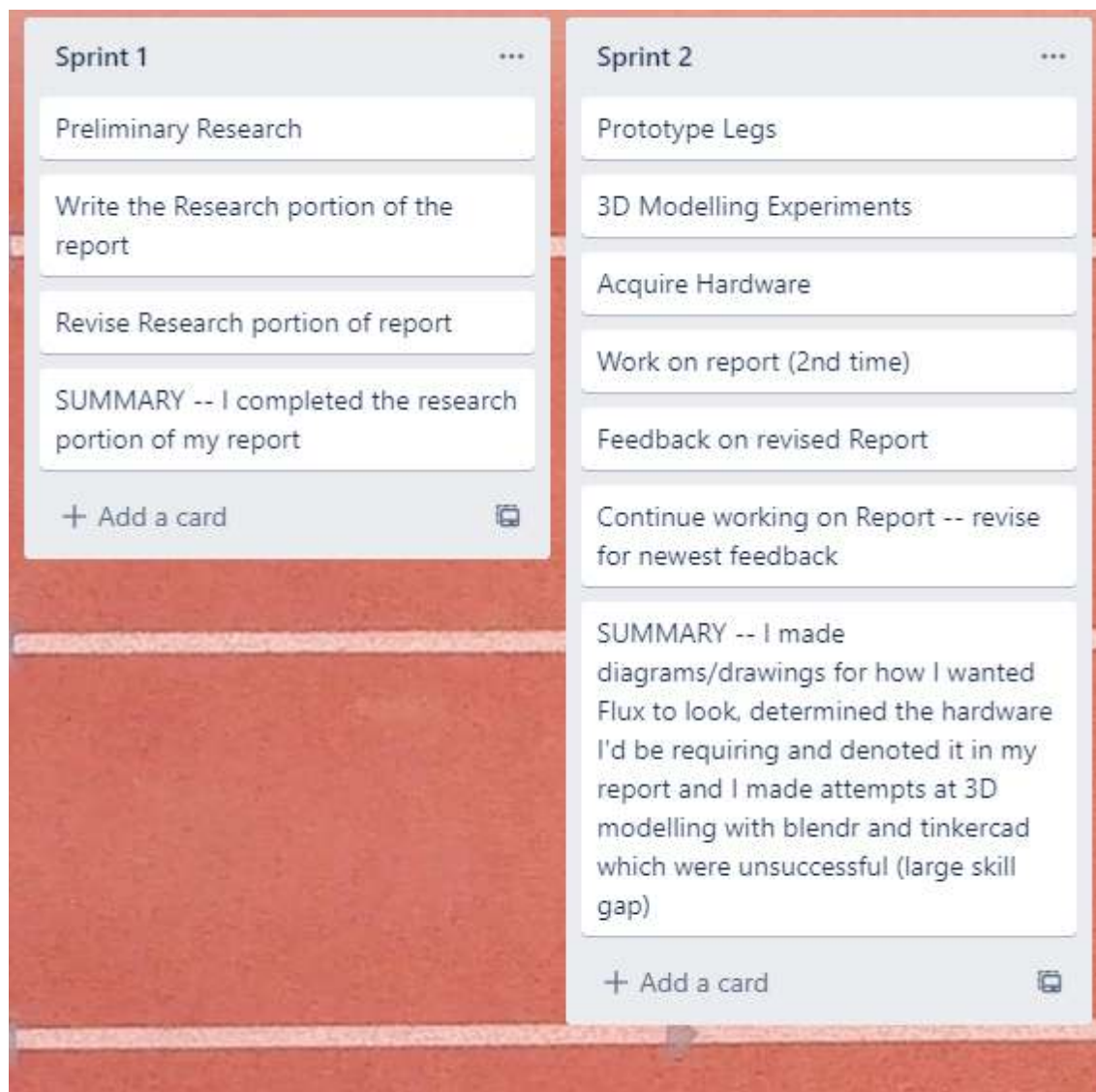
The plastic of the MeArms keeps snapping. The replacement of the Servos for the allegedly stronger servos has proven to be extremely tedious and painful to deal with. The plastic is extremely fragile and will snap at the slightest pressure. There's several points of failure for the MeArms, the most common being where I've had to create a new mount for the new Servos.

Constructing the legs themselves was easy enough, and in theory, I know what values I need to input to create movement, or at least what I hope will be movement. To expand upon this, I need to first turn **front left** and **rear right** hip joints forwards, extend the knee joints to contact the ground, turn all four hip joints backwards to propel Flux forwards until the **front left and rear right** legs are straight up and down and the **front right and rear left** legs are behind Flux, then contract the **front right** and **rear left** knee joints and turn the hip joints to be forwards and repeat first steps with **front right and rear right**.

Regarding external components, the thermal camera is outputting and working with the display. I still need to construct the controller itself, but I'm more concerned with the movement of Flux than having a wireless controller. The GPS needs calibrating then should simply work using its library. The sen sensor for human presence works well enough and the 4-1 air sensor is broken. Once again, less concerned with that than I am with the movement.

Regarding the controller, assuming I have time, it should be simple enough to construct a wired controller. Just utilize the two joysticks and when the left joystick is "forward" more forwards when back go back. Simple enough. **IF** I implement the right joystick, then when the joystick is pushed up, Flux looks up (extension of front legs, contraction of hind legs), when pushed back, Flux looks down (extension of hind legs, contraction of front legs). If I have a stroke of genius inspiration then I'll implement turning flux right and left which in and of itself could take another 3 weeks (time I don't have).

Trello Board Pictures



Time between Sprint 2 and 3 (added after the fact)

Determined 3D model for legs is not achievable for me

Found replacement -- MeArm's Kit

+ Add a card



Sprint 3



Combined MeArms together to get 4 combined legs

Develop Up and Down cycle of the legs

Introduce thermal camera

Thermal Camera output

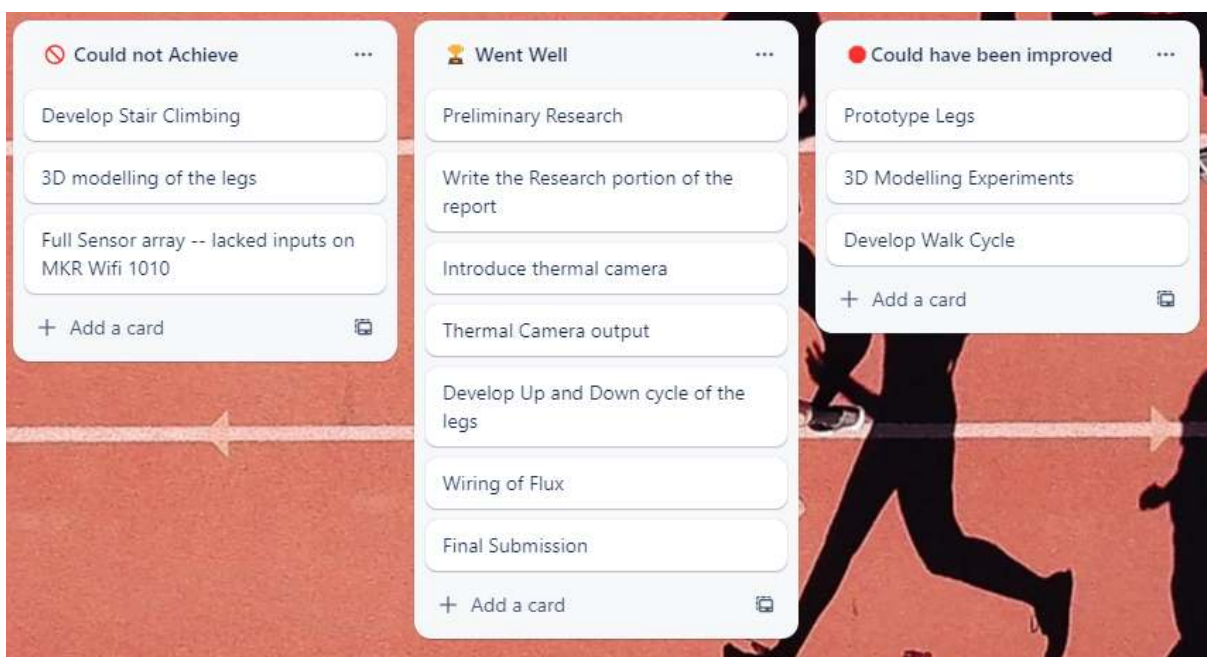
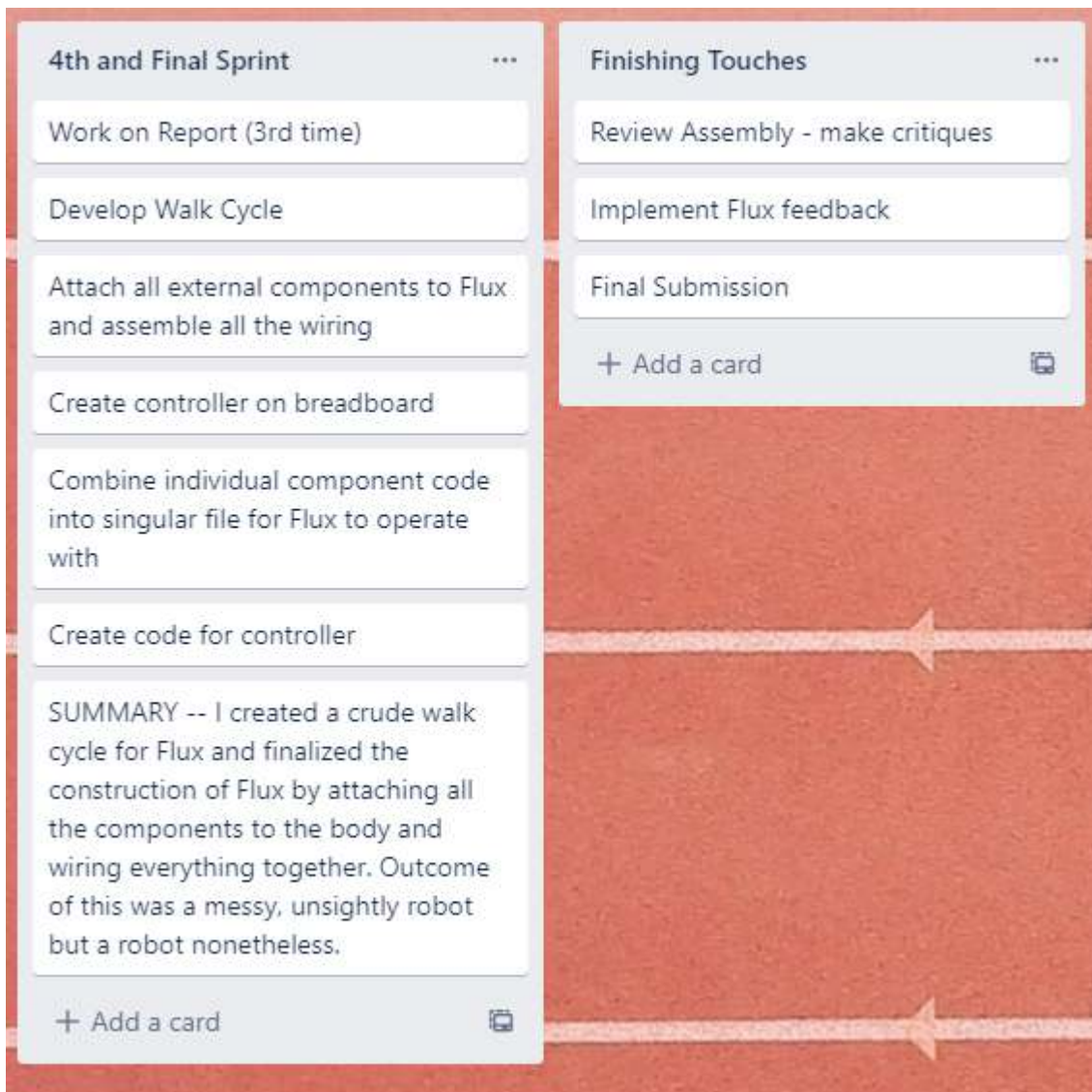
Experiment with other components

Finalized code for: SEN, GPS, and ultrasonic sensors

SUMMARY -- I began to work on the actual coding portion of the project and was able to complete the individual coded segments for all my components as well as put together the basis of Flux

+ Add a card





Action Plan

Task	Details	Done by Date (2023)
Find Project Supervisor	Discuss with supervisor the details of the project and its likelihood to produce desirable deliverable	9th October <u>(2022)</u>
Project Specifications and Ethics Form	Complete everything outlined in the Specification and Ethics Submission Details 2022-23	26th January
Preliminary Research	<p>Perform research into other “search and rescue” type of robotics</p> <p>Look into what components are needed for Flux’s construction</p> <p>Research how to 3D model and laser cut</p>	10th February
Writing of Research into Report	Write the Research portion of the Report	11th February
Research Review	Get advisor to review all the research and get feedback on the research aspect of the paper I’ve written so far	11-15th February Time allotted for revising and improving
Prototype Legs	Mount 4 rudimentary leg models to a simple base that will be controlled manually via several rotary encoders to seek out improvements in overall leg mechanics.	20th February
Work on Report	Work on the report and writing aspect of the project	21-22nd February (Full days of writing)
Feedback on Report	Get feedback on the report as a whole so far	23rd February
3D modelling experiments	Experiment with 3D modelling for the legs, base, and outer chassis	22nd February
Develop Up and Down movement	<p>Create code to allow Flux to move up and down with its legs</p> <p>Allow Flux to become free-standing</p>	23rd February

Develop Walk Cycle	Produce code to allow Flux to move on its own without manually adjusting the legs.	2nd March (1 week after successful free standing)
Work on Report	Work on the report and writing aspect of the project	4-6th March (Full days of writing)
Develop Climbing	Produce code to allow Flux the ability to move up steps without falling over or backwards	13th March (5 weeks until deadline)
Introduce Thermal Camera	Attach a thermal camera or processor to allow Flux to “see” and survey its surroundings.	14th March <i>(day after completion of climbing)</i>
Work on Report	Work on the report and writing aspect of the project	17-20th March (Full days of writing)
Feedback on Report	Get feedback for the section just written and the report as a whole.	21st March
Completion of Jumping Mechanism	Finalize the jumping of Flux so it does not fall over when landing or break apart	8th April (1 week until deadline)
Thermal Cam Output	Produce the output of the thermal camera for the user to see what Flux is seeing	8th April (1 week until deadline)
Review Assembly	Assemble Flux for review and get <u>visual</u> feedback	9th April
Implement feedback (if possible)	Ideally implement visual feedback via reprinting or remodelling.	14th April
Final Submission	Submit everything onto Blackboard and however it needs to be submitted	15th April 19th at the LATEST
Flux’s Demonstration	Flux gets to show off!	TBD? 1st May for Robotics course

Original Project Specification

PROJECT SPECIFICATION - Project (Technical Computing) 2022/23

Student:	Aiden Moncavage
Date:	20/01/2023
Supervisor:	Michael Bass

Degree Course:	Computer Science
Title of Project:	Flux: The Miniature Robotic Scout

Elaboration

Society has often been fascinated by the thought of having robotic assistants that can perform or aid them in completion of tasks. One of the next steps of societal progression is utilization of autonomous robotics. For this project I strive to achieve the creation of a miniature free-standing dog-like robotic that will be utilized to survey or scout unsafe or unknown locations.

The usage of this robotic could primarily benefit the defense and armed forces sector or sectors that work in close conjunction to life threatening environments, like the public sector where police or firefighters can send Flux in for search and rescue operations. This isn't to say it can't be used elsewhere, however during the development of this project the public sector and the defense and armed forces sector is the primary inspiration for Flux's development.

The usage of this robot is to allow access into hazardous or unsafe environments where it will then provide the user with feedback about its surroundings through the form of various sensors, components and thermal imaging while still being able to navigate basic obstacles without becoming unable to further operate.

Project Aims

This should be a bullet-point list of **aims** for the project which describes what you are hoping to achieve. Your project report should refer to this list to provide information on the extent to which these aims have been achieved.

Aims are **what** you want to achieve. Objectives are **how** you get to your aims.

Aims

- Prototype a four legged self-standing robotic which can jump in the air and land without collapsing
- Improve my knowledge in robotics and 3D modelling/design
- Consider what additional components might be useful when creating Flux for the aims of scouting dangerous environments
- Create a graphic implementation of a thermal processing component.

Objectives

- Test movement and jumping ability thoroughly and improve upon Flux's coding for leg movement
- Research what type of thermal cameras will be suitable for operation with an Arduino type board or various Raspberry Pi models
- Apply good engineering practices to create a quality deliverable prototype with minimal to no defects which can serve as a base for a future iteration.
- Research other types of "search and rescue" robotics to determine what components are best for Flux to have built-in.
- Research inverse kinematics to help with the construction, design, and implementation of the final robotic legs.

Project deliverable(s)

What are you going to produce and on what platform are you going to deliver it? What engineering approaches are you going to use?

The deliverable of this project will be a self-standing dog-like robotic that can move, jump, and climb

up stairs while being controlled remotely. Users will be able to use Flux to move to a hazardous location and provide feedback to the user.

Action plan

This should be a **table** listing the jobs that need doing to succeed with your project. This is your list of **objectives**. Against each job you should put a date by when it needs to be done. You and your supervisor will use this to ensure that the project remains on schedule. You could also use a graphical technique to present the information. Include optional deadlines such as the information review, contents page, and draft critical evaluation submissions.

Task	Details	Done by Date
Find Project Supervisor	Discuss with supervisor the details of the project and its likelihood to produce desirable deliverable	9th October
Project Specifications and Ethics Form	Complete everything outlined in the Specification and Ethics Submission Details 2022-23	26th January
Preliminary Research	Perform research into other "search and rescue" type of robotics Look into what components are needed for Flux's construction Research how to 3D model and laser cut	10th February
Writing of Research into Report	Write the Research portion of the Report	11th February
Research Review	Get advisor to review all the research and get feedback on the research aspect of the paper I've written so far	11-15th February Time allotted for revising and improving
Prototype Legs	Mount 4 rudimentary leg models to a simple base that will be controlled manually via several rotary encoders to seek out improvements in overall leg mechanics.	20th February
Work on Report	Work on the report and writing aspect of the project	21-22nd February (Full days of writing)
Feedback on Report	Get feedback on the report as a whole so far	23rd February
3D modelling experiments	Experiment with 3D modelling for the legs, base, and outer chassis	22nd February

Develop Up and Down movement	Create code to allow Flux to move up and down with its legs Allow Flux to become free-standing	23rd February
Develop Walk Cycle	Produce code to allow Flux to move on its own without manually adjusting the legs.	2nd March (1 week after successful free standing)
Work on Report	Work on the report and writing aspect of the project	4-6th March (Full days of writing)
Develop Climbing	Produce code to allow Flux the ability to move up steps without falling over or backwards	13th March (5 weeks until deadline)
Introduce Thermal Camera	Attach a thermal camera or processor to allow Flux to “see” and survey its surroundings.	14th March <i>(day after completion of climbing)</i>
Work on Report	Work on the report and writing aspect of the project	17-20th March (Full days of writing)
Feedback on Report	Get feedback for the section just written and the report as a whole.	21st March
Completion of Jumping Mechanism	Finalize the jumping of Flux so it does not fall over when landing or break apart	8th April (1 week until deadline)
Thermal Cam Output	Produce the output of the thermal camera for the user to see what Flux is seeing	8th April (1 week until deadline)
Review Assembly	Assemble Flux for review and get <u>visual</u> feedback	9th April
Implement feedback (if possible)	Ideally implement visual feedback via reprinting or remodelling.	14th April
Final Submission	Submit everything onto Blackboard and however it needs to be submitted	15th April 19th at the LATEST
Flux’s Demonstration	Flux gets to show off!	TBD? 1st May for Robotics course

BCS Code of Conduct

I confirm that I have successfully completed the BCS code of conduct on-line test with a mark of 70% or above. This is a condition of completing the Project (Technical Computing) module.

Signature: 

Publication of Work

I confirm that I understand the "Guidance on Publication Procedures" as described on the Bb site for the module.

Signature: 

GDPR

I confirm that I will use the "Participant Information Sheet" as a basis for any survey, questionnaire, or participant testing materials. The participant information sheet form is available on the Bb site for the module and as an appendix in the handbook.

Signature: 

UREC 1 Form



UREC 1 RESEARCH ETHICS REVIEW FOR STUDENT RESEARCH WITH NO HUMAN PARTICIPANTS OR DIRECT COLLECTION OF HUMAN TISSUES, OR BODILY FLUIDS.

All University research is required to undergo ethical scrutiny to comply with UK law. The University Research Ethics Policy (<https://www.shu.ac.uk/research/excellence/ethics-and-integrity/policies>) should be consulted before completing the form. The initial questions are there to check that completion of the UREC1 is appropriate for this study. The supervisor will approve the study, but it may also be reviewed by the College Teaching Program Research Ethics Committee (CTPREC) as part of the quality assurance process (additional guidance can be obtained from your College Research Ethics Chair¹)

The final responsibility for ensuring that ethical research practices are followed rests with the supervisor for student research.

¹ College of Social Sciences and Arts – Dr. Antonia Ypsilanti (a.ypsilanti@shu.ac.uk)
College of Business, Technology and Engineering – Dr. Tony Lynn (t.lynn@shu.ac.uk)
College of Health, Wellbeing and Life Sciences – Dr. Nikki Jordan-Mahy (n.jordan-mahy@shu.ac.uk)

Note that students and staff are responsible for making suitable arrangements to ensure compliance with the General Data Protection Regulations (GDPR), for keeping data secure and if relevant, for keeping the identity of participants anonymous. They are also responsible for following SHU guidelines about data encryption and research data management. Guidance can be found on the SHU Ethics Website <https://www.shu.ac.uk/research/excellence/ethics-and-integrity>

Please note that it is mandatory for all students to only store data on their allotted networked drive space and not on individual hard drives or memory sticks etc.

The present form also enables the University and College to keep a record confirming that research conducted has been subjected to ethical scrutiny. Students should retain a copy for inclusion in their research projects, and a copy should be uploaded to the relevant module Blackboard site.

The form must be completed by the student and approved by supervisor and/or module leader (as applicable). In all cases, it should be counter-signed by the supervisor and/or module leader and kept as a record showing that ethical scrutiny has occurred. Students should retain a copy for inclusion in the appendices of their research projects, and a copy should be uploaded to the module Blackboard site for checking.


Please note that it may be necessary to conduct a health and safety risk assessment for the proposed research. Further information can be obtained from the University's Health and Safety Website

1. General Details

Details	
Name of student	Aiden Moncavage
SHU email address	c0000279@hallam.shu.ac.uk
Department/College	Computer Science
Name of supervisor	Michael Bass
Supervisor's email address	m.bass@shu.ac.uk
Title of proposed research	Blotch: The Miniature Robotic Scout
Proposed start date	2/1/2023

Details	
Proposed end date	4/20/2023
Brief outline of research to include, rationale (reasons) for undertaking the research & aims, and methods (250-500 words).	<p>The goal of this project is to create a free-standing robotic that can monitor the temperature of its surroundings, output this to a monitor as a thermal camera would, as well as the ability to move up steps and the ability to jump all while being controlled remotely. Additionally Blotch will be contained in an ideally “eye pleasing” outer chassis.</p> <p>I’m going to be researching other “search and rescue” type robotics that I can use as a basis for determining what are important features to include and what isn’t necessary when developing Blotch. Additionally, there will be a fair amount of research done towards the construction of the robot itself. I’ll need to look into inverse kinematics for the leg movements as this is typically how robotic appendage movement is performed; this will determine if I need extra components on each of the legs to allow smoother up and down movement.</p> <p>3D modeling and laser cutting will be important for the creation of the legs, especially when I have no prior experience so this will be the crucial research element of the project. Thorough research will be done towards the usage of which individual components Blotch will need; this will also tie into the prior research into “search and rescue” robotics. Once I’ve selected the components I’ll look into what, if any, difference there is between the quality of components. While researching into what’s needed I’ll also cover why I’ve chosen the parts I need and the difference between what I could have chosen, specifically the difference between servomechanisms and motors.</p>

I confirm that this study does not involve collecting/using data or samples from human participants

Please tick 

2. Research in external organizations

Question	Yes/No
1. Will the research involve working with/within an organization (e.g., school, business, charity, museum, government department, international agency, etc.)?	no
2. If you answered YES to question 1, do you have granted access to conduct the research? <i>If YES, students please show evidence to your supervisor. PI should retain safely.</i>	

Question	Yes/No
<p>3. If you answered NO to question 2, is it because:</p> <p>A. you have not yet asked</p> <p>B. you have asked and not yet received an answer</p> <p>C. you have asked and been refused access.</p> <p><i>Note: You will only be able to start the research when you have been granted access.</i></p>	

3. Research with Products and Artefacts

Question	Yes/No
1. Will the research involve working with copyrighted documents, films, broadcasts, photographs, artworks, designs, products, programs, databases, networks, processes, existing datasets, or secure data?	no
<p>2. If you answered YES to question 1, are the materials you intend to use in the public domain?</p> <p><i>Notes: 'In the public domain' does not mean the same thing as 'publicly accessible'</i></p> <ul style="list-style-type: none"> Information which is 'in the public domain' is no longer protected by copyright (i.e., copyright has either expired or been waived) and can be used without permission. Information which is 'publicly accessible' (e.g., TV broadcasts, websites, artworks, newspapers) is available for anyone to consult/view. It is still protected by copyright even if there is no copyright notice. In UK law, copyright protection is automatic and does not require a copyright statement, although it is always good practice to provide one. It is necessary to check the terms and conditions of use to find out exactly how the material may be reused etc. <p><i>If you answered YES to question 1, be aware that you may need to consider other ethics codes. For example, when conducting Internet research, consult the code of the Association of Internet Researchers; for educational research, consult the Code of Ethics of the British Educational Research Association.</i></p>	
<p>3. If you answered NO to question 2, do you have explicit permission to use these materials as data?</p> <p><i>If YES, please show evidence to your supervisor.</i></p>	
<p>4. If you answered NO to question 3, is it because:</p> <p>A. you have not yet asked permission</p> <p>B. you have asked and not yet received and answer</p> <p>C. you have asked and been refused access.</p> <p><i>Note You will only be able to start the research when you have been granted permission to use the specified material.</i></p>	


4. **Does this research project require a health and safety risk assessment for the procedures to be used?** Discuss this with your supervisor and consult the [Risk Assessment Toolkit](#) for teaching research.

☐ Yes

☒ No

(If **YES** the completed Health and Safety Risk Assessment form should be attached). You can find a [Blank/Sample Risk Assessment Form](#) at the Checklist, Generic and TORS Risk Assessments on the [Risk Assessment Toolkit](#)

Adherence to SHU policy and procedures

Ethics sign-off	
Personal statement	
I can confirm that: <ul style="list-style-type: none"> I have read the Sheffield Hallam University Research Ethics Policy and Procedures I agree to abide by its principles. 	
Student	
Name: Aiden Moncavage	Date: 1/25/2023
Signature: 	
Supervisor or another person giving ethical sign-off	
I can confirm that completion of this form has confirmed that this research does not involve human participants. The research will not commence until any approvals required under Sections 2 & 3 have been received and any health and safety measures are in place.	
Name:	Date:
Signature:	
Additional Signature if required:	
Name:	Date:
Signature:	

Please ensure that you have attached all relevant documents. Your supervisor must approve them before you start data collection:

Relevant Documents

Research proposal if prepared previously

Yes☐**No**☐**N/A**☐

Any associated materials (e.g., posters, letters, etc.)

☐☐☐

Health and Safety Risk Assessment Form

☐☐☐