

Final Report

Ground Vibration Simulator for Investigating Vibration Effects on the Development of Turtle Eggs



Team Members

Meia Copeland	100972654
Shawaiz Khan	100917863
Talal Jaber	101167571
Marwan Zeyada	101141759
Ranishka Fernando	101063607

Supervisors

Dr. Lynn Marshall
Dr. Yuu Ono

April 12, 2023

Abstract

The Davy Lab, in the Department of Biology at Carleton University, conducts conservation studies to better understand how environmental changes affect locally vulnerable bats, amphibians, and reptiles. They wish to conduct an experiment that looks at how ground vibrations from nearby industrial activity and highways impact turtle egg growth. In this project, the engineering team developed a tool that can replicate the ground vibrations caused by diverse industrial operations so that the lab may explore these effects in a controlled setting.

The team aimed to create a ground vibration simulating shake table for under \$1000, as industrial shake tables begin at \$5000 each.

With inspiration from the James Webb Space Telescope mirror positioning actuators, the team has created a ground vibration simulation table that costs less than \$1000. Desired vibrations of 0.02 – 0.04 mm linear displacements and frequencies of 5 – 20 Hz were achieved. A User Interface and data collection system were designed to make the simulation table user-friendly.

The final design of the table costs under \$700 for the initial table and data collection system, with the ability to add tables to the system for less than \$200 per table. All costs do not include labour. The simulation table is almost in a research-ready state and will be ready for the Davy lab by the beginning of the Summer 2023 term. The entire project is available as an open-source project on [Github](#), so that the Davy lab and other labs can use the design to achieve their research goals.

Acknowledgements

We would like to express our heartfelt appreciation to the following individuals who have made valuable contributions to the completion of this project:

First, we would like to extend our gratitude to Nagui Mikhail for his help and advice early on regarding motors and achieving the desired results. We are deeply grateful for his unwavering support, which has been invaluable to us.

We would also like to thank Dr. Peter Gordon for his advice regarding measuring very small displacements and working at such a small scale (< 1 mm).

We would like to acknowledge Sam Gauthier for his contributions, including introducing us to Polyfractal's video as inspiration, helping with wood construction, and offering advice on 3D modelling and printing.

Our sincere thanks go to Matt Marshall—"that guy from fencing" as he asked to be called— and to Graham Bell for their help when last-minute 3D printed prototypes were needed.

We are deeply grateful to Davy Lab for giving us the opportunity to do something unique and trailblaze a new approach to the engineering capstone project. Also, for supporting us through the year financially, and teaching us so much about turtles and conservation.

Finally, we would like to extend our heartfelt gratitude to our supervisors for their encouragement throughout the year and for believing in us. Their guidance and support have been instrumental in helping us complete this project.

Thank you all for your support and guidance. We could not have completed this project without your help.

Table of Contents

Abstract	ii
Acknowledgements	iii
Table of Contents	1
List of Figures and Tables	4
1 Introduction	6
1.1 Background	6
1.2 Motivation	7
1.3 Project Objectives	7
1.4 Accomplishments	8
1.5 Report Outline	9
2 The Engineering Project	10
2.1 Health and Safety	10
2.2 Engineering Professionalism	10
2.3 Project Management	11
2.4 Justification and Suitability for Degree Program	13
2.5 Individual Contributions	15
2.5.1 Project Contributions	15
2.5.2 Final Report Contributions	16
2.5.3 Proposal Contributions	18
2.5.4 Progress Report Contributions	19
3 Requirements	20
3.1 Functional Requirements	20
3.2 Non-Functional Requirements	21
3.3 Use Cases	22
3.3.1 Use Case 1 – Begin New Experiment	22
3.3.2 Use Case 2 – Change the vibrational Frequency.	23
3.3.3 Use Case 3 – Exporting the Experiment Data	24
3.3.4 Use Case 4 – Monitoring Shaker Table Performance	24
3.3.5 Use Case 5 – Verify Motor RPM Measurements	25

4	Research on Simulator Design	25
4.1	Measuring Linear Displacement.....	25
4.2	Measuring Frequency.....	26
4.3	Methods of Vibration Simulation	31
4.3.1	Linear Actuator	31
4.3.2	DC Motor.....	33
4.4	Database Selection to Store Acquired Data	34
4.5	User Interface Framework	36
5	Vibration Simulator Design	37
5.1	System Design	37
5.2	Sensors	39
5.2.1	Measuring Linear Displacement	39
5.2.2	Temperature and Humidity	39
5.2.3	RPM Sensor	40
5.3	Methods of Vibration	40
5.3.1	Flexure-guided Linear Actuator Design	40
5.3.2	Table Design.....	43
5.3.3	Motor Feedback System	45
5.4	Database Design.....	46
5.4.1	Designing the Database Schema.....	46
5.4.2	Database Implementation on the Raspberry Pi	47
5.5	User Interface.....	50
	Normal User Flow	53
6	System Evaluation.....	56
6.1	Computer and Microcontroller Devices.....	56
6.2	Sensors	56
6.2.1	Linear Displacement Sensor.....	56
6.2.2	Temperature and Humidity Sensor	57
6.2.3	RPM Sensor	57
	Vibration Simulator	58

6.2.4	Flexure-guided Linear Actuator	58
6.2.5	Motor Feedback System	58
6.3	Database.....	59
6.4	User Interface.....	60
7	Budget Breakdown.....	62
8	Reflections.....	63
8.1	Limitations.....	64
8.2	Future Work and Improvements.....	64
	Future Work	64
	Improvements.....	65
9	Conclusion	66
10	References.....	67
	Appendix 1: Relevant Courses	71
	Appendix 2: Additional Diagrams	72
	Measuring Frequency with an Accelerometer	72
	Motor	73
	Database	75
	User Interface	76
	Appendix 3: Costs.....	78
	Research and Development Costs	78
	Additional Table Costs	79
	Appendix 4: Work Plan	80
	Collaboration.....	80
	Project Milestones	80
	Schedule of Activities/Gantt Chart	82
	Appendix 5: Project Links.....	84
	Appendix 6: Progress Report	84
	Appendix 7: Proposal	84

List of Figures and Tables

Figure 1 – ADXL335 Accelerometer Connected to Arduino Uno R3 for Vibration Data Measurement and Analysis	27
Figure 2 – Code snippet of conducting Fourier transform manually and calculating frequency.	29
Figure 3 – Overview of the code used for Accelerometer Data Processing and Frequency Spectrum Calculation using Manual Fourier Transform	30
Figure 4 – Disk attached to gear	31
Figure 5 – Demonstration of backlash between gears	31
Figure 6 – Strain vs. voltage curve for a typical piezo material when alternating voltages are applied [16]	32
Figure 7 – DC motor frequencies [19].....	33
Figure 8 – L298N Motor Driver	34
Figure 9 – An example of a RDMS in banking [25].....	35
Figure 10 – System block diagram	37
Figure 11 – Vibration table system	38
Figure 12 – Digital indicator set-up.....	39
Figure 13 – DHT22 Sensor Module	40
Figure 14 – IR Sensor Module	40
Figure 15 – Original design of the JWST fine positioning stage by Polyfractal, available on Thingiverse.com [18].	41
Figure 16 – Simulated displacement of flexure	41
Figure 17 – Flexure with gears	42
Figure 18 – Range of motion of flexure	42
Figure 19 – Flexures with couplers for (a) under-mounting and (b) side-mounting, where arrows indicate the direction of vibration.	43
Figure 20 – 3D model of table design.	44
Figure 21 – 3D model of table design, actuator mounted underneath.	44
Figure 22 – Actual build of table, with actuator in side-mounted configuration.....	45
Figure 23 – Motor feedback system diagram	46
Figure 24 – Database file as an excel spreadsheet.	48
Figure 25 – Received email containing attached database file as excel spreadsheet.....	49
Figure 26 – Home screen UI.....	51
Figure 27 – Settings Page UI	52
Figure 28 – Code utilized to calculate frequency using raw data of accelerometer	72
Figure 29 – Motor design schematics	73
Figure 30 – Motor and IR sensor code.....	74
Figure 31 – Database schema from the proposal.....	75
Figure 32 – Database schema from the progress report.....	75

Figure 33 – Home screen UI Wireframe	76
Figure 34 – Setting Screen UI Wireframe	76
Figure 35 – Home screen UI First Version.....	77
Figure 36 – Settings screen UI First Version	77
Figure 37 – Gantt chart	83
Table 1 – Roles, tasks, and relevant experience for each team member.....	13
Table 2 – Contributions of each team member to the project components.	15
Table 3 – Contributions from each team member to the Final Report document.	16
Table 4 – Contributions of each team member to the Proposal report.....	18
Table 5 – Contributions of each team member to the Progress Report.	19
Table 6 – Different gear configurations and their output frequencies	42
Table 7 – Database Schema	47
Table 8 – Hardware required to complete project, with sources and pricing.	62
Table 9 – Cumulative relevant courses for required knowledge and experience.....	71
Table 10 – Components ordered for research and development purposes.	78
Table 11 – Components needed to build additional table for system.	79
Table 12 – Milestone descriptions and completion dates.....	80
Table 13 – Schedule of activities for project completion, documentation, and presentations..	82

1 Introduction

As we enter an era of ever-increasing industrialization, the need for a sustainable and conscious approach towards the environment has become more crucial than ever. At the forefront of this movement are conservation research groups like the Davy Lab at Department of Biology, Carleton University, dedicated to understanding the impact of local environmental changes on threatened species of bats, amphibians, and reptiles.

One of the lab's recent projects is investigating the effects of ground vibrations from adjacent industrial activities and roadways on the growth of turtle eggs. To tackle this issue, a team of engineering students – Meia Copeland, Shawaiz Khan, Talal Jaber, Marwan Zeyada, and Ranishka Fernando – has developed a tool that can simulate these ground vibrations in a controlled environment.

For this engineering project, the team has designed a shake table prototype that can simulate the vibrations produced by industrial activities, enabling the lab to investigate the impacts on turtle eggs more easily. This research will contribute knowledge to the field of reptile conservation, and impact how appropriate measures are taken to preserve these threatened species in the future.

1.1 Background

As cities continue to expand, the need for infrastructure like power-generation facilities, housing, and roads has also grown. Unfortunately, these developments have a significant impact on the local ecosystems and ground-dwelling species in the area.

Power-generation facilities such as wind farms, hydroelectric plants, and steam turbine powered plants also generate ground vibrations, but most research has focused on the impacts of fully aquatic species around hydroelectric plants. The vibrations associated with power plants using steam turbines (nuclear, biomass, natural gas, and coal) have a displacement in the range of 0.8 – 2 μm , and a frequency of approximately 8 Hz [1]. Wind turbine farms create vibrations with a displacement of approximately 300 nm within a 150 m radius around the base of a turbine, with a frequency of 2 – 10 Hz depending on wind speeds [2]. Vibrations attenuate significantly when measured from further away from the turbine. Initially, the project aimed to simulate the ground vibrations from wind turbines (as seen in Appendix 7: Proposal). However, it was determined that larger vibrations should be examined first. If the larger vibrations have no effect on the development of turtle eggs, then smaller vibrations from wind turbines would not have an effect either. The project instead chose to focus on the following larger vibrations and iterate in the future for smaller vibrations if experimental data shows an effect.

Industrial activities such as construction, pile driving, and heavy machinery use cause ground vibrations that disturb the habitats and burrows of local species, potentially affecting their

survival and breeding success. These vibrations have a frequency range of 20 – 45 Hz, and a displacement range of 0.27 – 1 mm [3]. These vibrations do not occur continuously, but rather in intervals over a long period of time [4].

Roadways and railways are significant sources of ground vibrations, with highways producing consistent vibrations for long periods every day. On the other hand, railways produce similar vibrations but not consistently. These ground vibrations have a displacement in the range of 0.16 – 0.8 mm, at frequencies between 10 – 20 Hz [3].

Despite extensive research conducted on the impacts of roadways, construction, and power generation facilities on wildlife, the effects of ground vibrations produced by these sources are still not well understood.

The effects of ground vibrations are particularly important to research due to the behaviour of turtle embryos. During development in the egg, many turtle species communicate via mechanical vibrations. This communication has been shown to initiate hatching among a clutch of eggs. Further, other sources of vibration such as handling, transportation, and thunder have also been shown to initiate hatching among some species of turtle [5]. The Davy Lab wishes to investigate whether vibrations from human industrial and transportation activity could have a similar effect on turtle development and hatch timing.

1.2 Motivation

The rapid growth of urban areas in Ontario and the consequent expansion of industrial and transportation infrastructure has resulted in significant changes to the local ecosystem. Although the effects of roadways, industrial activities, and power generation on wildlife have been studied, the impact of ground vibrations from these sources remains poorly understood.

The project team has developed a tool that can simulate the ground vibrations produced transportation infrastructure so that the Davy Lab can investigate the impact of these vibrations on turtle egg development in a controlled laboratory environment. By measuring environmental parameters like temperature and humidity, the tool will enable the Davy Lab to study the effects of ground vibrations on turtle eggs and other ground-dwelling species. This research will provide valuable insights into the impact of human activities on the local ecosystem and help identify measures to mitigate their negative effects.

1.3 Project Objectives

The objective of our project is to design and develop a tool that can accurately mimic ground vibrations from a range of industrial activities, such as construction and transportation, within a controlled environment. The device will generate vibrations with a frequency range of 5 to 20 Hz and a displacement range of 0.1 to 1 mm, like those produced by various sources of industrial

vibration. The tool will have a surface area of 30 cm x 30 cm, sufficient for one egg incubation tub, with the displacement and frequency measured from this surface.

The device will be able to generate vibrations continuously or at intervals for a period of up to 95 days, which is the incubation period for snapping turtle eggs [6]. By meeting these objectives, we will enable the Davy Lab to investigate the impact of ground vibrations on turtle egg development under controlled and measurable conditions.

Existing shake tables used for industrial and engineering purposes could achieve the vibration needed. However, these machines are often over specification and expensive at over \$6000. Also, it does not meet all the requirements from the Davy lab for monitoring the effect of vibration on turtle egg development. This project aims to create a device sufficient for this area of research, that costs under \$1000 and can be easily put together by anyone interested in ground vibration research. Further, this project intends to provide a more customizable solution that can be used for various species and vibration types. The plans and documentation for the project will be made open-source and posted on GitHub for anyone to use.

Additionally, in response to the lab's request, our objective is to design and develop a Structured Query Language (SQL) database capable of keeping track of data collected from various sensors. The database will also store basic egg batch details to enable the lab staff to analyze the data for their research.

1.4 Accomplishments

Over eight months, the team achieved most goals set out in this report, while keeping the project under a budget of \$1000 for the components before labour costs. The ground vibration simulator device is small enough to place on a table in the lab and have one turtle egg incubator placed on top. The table surface can vibrate up and down smoothly with a customizable displacement range between 0.01 – 0.04 mm. While this is slightly lower than the target of 0.1 – 1 mm, this range is still relevant to some ground vibrations. The linear displacement mechanism to move the table up and down can be used in two configurations, each with a different displacement value. Using 3D modelling software, the mechanism may be simply rebuilt for additional displacements and 3D printed for free through the University Library. An increased range will be investigated in the future.

To vibrate the table at frequencies between 5 and 20 Hz a motor is connected to the linear displacement mechanism. The motor is connected to a sensor to track revolutions -per-minute (RPM) to create a feedback system where the user can select a frequency. The incorporation of an infrared obstacle avoidance sensor module parallel to the motor permitted, non-intrusive detection of the motor's RPM. This method takes advantage of the sensor's inherent versatility, high precision, and low cost, all of which are critical considerations for accurate and dependable RPM measurements. The non-contact measurement method eliminates any potential influence

of vibrations or disturbances on frequency measurement, resulting in an accurate portrayal of the motor's performance. Furthermore, integrating this sensor module with an Arduino microcontroller platform allows for real-time data acquisition and processing.

The ground vibration simulator device is controlled using an Arduino microcontroller which connects to a Raspberry Pi (RPi) computer. The RPi hosts a database and User Interface (UI) to track data and allow the user to control the frequency and duration of an experiment. The RPi can have multiple devices connected, allowing for a central control of all experiments. The Arduino codebase's modification capabilities enable further fine-tuning of data processing and analysis to match individual requirements.

For data collection, a cost-free SQLite database is implemented on the Raspberry Pi using the SQLite3 module in Python programming language. The database works with various sensors, which allows it to populate crucial data for the lab's research. Additionally, an option to send the data over to a user's email address using Google's SMTP (Simple Mail Transfer Protocol) server is implemented, which adds a level of convenience and accessibility to the database.

Finally, all these components come together under the User Interface (UI) providing a simple and user-friendly experience for researchers to control and monitor experiments. The UI allows users to set the frequency and duration of the experiment, as well as to view real-time data from the sensors. The interface also includes a graphical representation of some of the data collected, making it easy for users to analyze and interpret the results. With the integration of the SQLite database and the ability to send data to email, the UI streamlines the research process, allowing users to focus on their experiments and data analysis.

1.5 Report Outline

The following part of the final report for this project will start with an overview of the health and safety concerns, description of engineering professionalism, project management methods, and justification and suitability of the project for each team member. The report will then detail the research that informed the development of the ground vibration simulation device, including successful and unsuccessful approaches. The design of the simulator will be described in detail, including its components and how they work together. Additionally, the report will delve into the research conducted on the development of the User Interface and database, providing insights into successful and unsuccessful approaches. The report will also outline the project plan, including milestones, activities, and budget breakdown. Finally, the report will discuss how the project objective was achieved through the collaboration of the engineering team.

2 The Engineering Project

2.1 Health and Safety

It was crucial to ensure the health and safety of everyone working on a project. Since potential hazards in the project environment needed to be identified and mitigated, it was essential that all members adopt a proactive approach. The Health and Safety Manual supplied by Carleton University served as the foundation for the safety protocols adopted for this project. The manual provides detailed instructions on how to maintain a safe workplace and is intended to assist students in complying with applicable regulations. The steps taken to ensure a safe and healthy system for the project are described, with particular emphasis on Section 13 (Tools and Machinery) and Section 6.12 (Electrical Equipment and Apparatus).

As mentioned before, a key focus was placed on Section 13 of the Health and Safety Guide. This section provided guidelines for ensuring safe operation and good condition of the equipment. As such, a noticeable defect in any equipment was immediately reported to the rest of the team, and appropriate measures were taken to rectify it. Additionally, the user (Davy Lab) is being instructed and trained to properly operate the system, with thorough documentation being provided to enable them to perform regular scheduled maintenance while keeping the equipment in good working condition.

Section 6.12 was another significant section of the Health and Safety Guide that was pertinent to this project. It provided guidelines for ensuring the safe use of electrical equipment and preventing accidents involving electricity. Specifically, extension cords were discouraged for permanent installation of the system to prevent any potential risks that may arise due to the use of outdated or improperly installed equipment. Additionally, they pose a tripping hazard and increase the risk of electrical fires. Moreover, depending on the placement of the system in the biology laboratory, the use of ground fault circuit interrupters was suggested to reduce the possibility of an electric shock in wet areas.

Finally, Section 5 of the manual referred to following the general health and safety principles such as always using appropriate personal protective equipment where applicable. The aim was to prevent accidents and minimize potential hazards through the emphasis on the importance of regular maintenance, proper operation of equipment, and the use of appropriate safety measures. Furthermore, the guidelines and the steps taken were revisited and revised to ensure they remain effective and up-to date.

2.2 Engineering Professionalism

As engineers, we have a responsibility to ensure that our work is conducted in a manner that upholds the highest ethical standards. In Canada, the Code of Ethics for Engineers provides a framework for ethical behavior that engineers must follow in order to maintain their professional

licenses. Throughout our project, we were careful to ensure that we met our professional responsibilities in accordance with this code.

One of the most important principles outlined in the code is the duty to protect the public. This includes a responsibility to ensure that our work is conducted in a safe and responsible manner, and that we do not knowingly put the public at risk. In our project, we took this responsibility very seriously. We conducted extensive testing at every stage of development to ensure that the system was safe and reliable. We also implemented fail-safe mechanisms to prevent any dangerous or unforeseen situations from occurring.

Another key principle outlined in the code is the duty to act with integrity. This includes a responsibility to be honest and transparent in all our dealings, and to not engage in any behavior that would damage our reputation or the reputation of the engineering profession. Throughout our project, we were transparent and honest in all our interactions. We documented our work thoroughly and made our code available to our colleagues for review. We also communicated with the Davy lab and the instructors regularly to keep them informed of our progress and to address any concerns they may have had.

The code also stresses the importance of maintaining a high level of competence in our work. This includes a responsibility to stay up to date with the latest developments in our field, and to continually strive to improve our skills and knowledge. Throughout our project, we worked hard to stay on top of the latest developments in engineering and to ensure that our work met the highest standards of quality. We also took the time to reflect on our work and to identify areas where we could improve in the future.

One key section that we followed was Section 1.1, which requires engineers to “undertake only work that they are competent to perform.” We ensured that all team members were adequately trained and competent in the use of the chosen framework and language for our project. Additionally, we followed Section 1.3, which requires engineers to “avoid real or perceived conflicts of interest whenever possible, and to disclose them to potentially affected parties when they do exist.” We ensured that all stakeholders involved in the project were aware of any potential conflicts of interest and worked to mitigate them. Furthermore, we followed Section 3.3, which requires engineers to “communicate clearly, objectively, honestly, and without bias.” We ensured that all communication with the Davy lab was clear and objective, providing regular updates on the progress of the project and any potential issues that arose. By following these sections of the Code of Ethics for Engineers in Canada, we were able to ensure that our project was carried out in an ethical and professional manner.

2.3 Project Management

One of the goals of this engineering project was to get real-world experience in working on a long-term team project. To achieve this goal, the team adopted industry-level project

management techniques to coordinate, manage, and execute the project. Moreover, instead of relying on a single management technique, a combination of techniques was utilized to maximize efficiency and productivity.

One of the key techniques used was the Work Breakdown Structure (WBS). WBS is a powerful tool for dividing the project into manageable tasks as it provided a structured approach to project planning and management (e.g., Research, Implementation, Testing) [7], enabling the team to comprehend the various project components and associated tasks with ease. This resulted in a better management of the project's scope, timeline, and resources by dividing it into smaller, more manageable chunks.

Another important tool utilized was Gantt charts. This organizational tool served as a visual representation of the project's timeline and task dependencies, highlighting critical milestones and deadlines ranging from report deliverables to chunks broken down from WBS. By utilizing the WBS, tasks were accurately defined in the Gantt chart, allowing for greater clarity and understanding of the project's scope and timeline.

A Waterfall development method was initially considered as the primary project management technique. This approach is often compared to the Agile method, which is a more flexible and iterative approach to project management, as it employed a cyclical process of planning and execution that permitted more iterations and changes than the Waterfall method which is based on a linear and sequential approach [8]. After careful consideration of both methods, the Waterfall method was chosen as it was well suited to the project's distinct phases, which required a linear, sequential approach to development. Activities and tasks in a waterfall method typically flow linearly through five phases (Requirements, Design, Implementation, Verification, and Maintenance), which was ideal for a project with defined objectives and limited iterations.

By adopting this combination of project management techniques, the team benefitted from greater organization, increased efficiency, and improved productivity which ultimately led to the project staying on track and meeting its milestones and deadlines.

2.4 Justification and Suitability for Degree Program

Table 1 – Roles, tasks, and relevant experience for each team member.

Team Member	Tasks	Justification
Meia Copeland <i>Computer Systems Engineering</i>	Linear Actuator Design	Designed 3D models for the linear actuator, and accessory parts such as frames, brackets, shafts, etc. ELEC 1010
	Simulation	Created simulations with 3D models to assess how the system would respond to physical interactions. ELEC 3105, SYSC 3600, SYSC 4505
	Signal processing	Wrote code to interact with sensors and process the outputs into useful data. SYSC 3010, SYSC 3310, SYSC 4805
	Hardware-Software interfacing	Created code libraries that interfaced between external sensors and the UI, for data collection and system control purposes.
	System integration	Utilized processes learned throughout degree program to integrate various system components and test them. SYSC 3020, SYSC 3303, SYSC 3310, SYSC 4805
	Project management	Created various project management processes to plan, track, and execute work as learned through degree program and co-op placements. SYSC 3020, SYSC 3310, SYSC 4805
Shawaiz Khan <i>Computer Systems Engineering</i>	Database	Designed and implemented different types of databases in SYSC 3010
	User Interface	Designed and implemented GUIs for phone and web apps in SYSC 3010 and SYSC 2004
	Hardware Integration	Integrated and tested multiple hardware-software projects in SYSC 3010 and SYSC 4805, including remote dispenser and autonomous car
	Software Integration and testing	Experience in software integration and testing on multiple hardware-software projects in SYSC 3010, SYSC 3303 and SYSC 4805
Talal Jaber <i>Electrical Engineering</i>	Electrical design	Designed multiple circuits in ELEC 2501 and ELEC 3509
	Hardware integration	Choosing the right motor, driver, and IR sensor module for the project in Elec 3907

	Hardware-software interfacing	Learned how to code and add libraries to control the sensors in SYSC 2004 and SYSC 2006
Marwan Zeyada <i>Computer Systems Engineering</i>	User Interface	Created GUIs using Python and Java for various systems. SYSC 3010 and SYSC 2004
	Hardware Integration	Designed a mechanical door opener that uses sensors and motors to remotely open doors. SYSC 3010
	Database	Designed multiple databases for plethora of projects SYSC 3010
	Software integration and testing	Great experience in software integration and testing due to working on multiple projects of that nature. SYSC 3010 and SYSC 4805.
Ranishka Fernando <i>Electrical Engineering</i>	Electrical design	Designing required PCB
	Hardware integration	Constructed required circuitry and set up the motor for testing – Experience from ELEC 3907 and past work experience as a Hardware Consultant
	Debugging	Extensive experience in development and operations through CO-OP and knowledge acquired from ELEC 2607 and ELEC 3500
	Simulation	Simulation using ANSYS in ELEC 3105
	Linear Actuator Design	CAD experience from ECOR 1010

Collectively, the team had all the necessary skills and knowledge to be successful on this project. Requirements for this project mostly involved electronics or computer systems techniques previously learned, with some very basic mechanical engineering and CAD that was learned in first year general engineering courses such as ECOR 1010 and ECOR 1101.

Further, this project involved working closely with a collaborator, the Davy lab. Project management and client-interaction skills were learned in courses such as CCDP 2100, ELEC 3907, SYSC 3010, SYSC 4805, and ECOR 4995. Most team members have some co-op experience, where working professionally with other engineers and clients was learned.

2.5 Individual Contributions

2.5.1 Project Contributions

Table 2 – Contributions of each team member to the project components.

Component		Contributor
Linear Actuator	3D modelling	Meia Copeland
	Simulation Testing	Meia Copeland and Ranishka Fernando
	3D printing	Meia Copeland
	Motor Control	Talal Jaber and Ranishka Fernando
	Actuator Testing	Meia Copeland
Shake Table	Table Support Design	Meia Copeland and Talal Jaber
	Assembly	Meia Copeland
Wiring	Assembly	Talal Jaber
Motors	DC Motor testing	Talal Jaber
	RPM Sensor	Talal Jaber
	Motor Feedback System	Talal Jaber
	Accelerometer Testing	Ranishka Fernando
Environmental Sensors	Hardware connection	Meia Copeland
	Software Integration	Shawaiz Khan, Marwan Zeyada, and Meia Copeland
	Testing	Shawaiz Khan and Marwan Zeyada
Database	Schema design	Shawaiz Khan
	Implementation	Shawaiz Khan and Marwan Zeyada
	Testing	Shawaiz Khan
User Interface	UI Design	Marwan Zeyada
	Implementation	Marwan Zeyada and Shawaiz Khan
	Testing	Marwan Zeyada

2.5.2 Final Report Contributions

Table 3 – Contributions from each team member to the Final Report document.

Final Report		Contributor
1 Introduction	Abstract	Ranishka Fernando, Meia Copeland
	Background	Meia Copeland
	Motivation	Meia Copeland
	Project Objectives	Meia Copeland, Shawaiz Khan
	Accomplishments	Shawaiz Khan, Meia Copeland
	Report Outline	Meia Copeland, Shawaiz Khan
2 The Engineering Project	Health and Safety	Shawaiz Khan
	Engineering Professionalism	Marwan Zeyada
	Project Management	Shawaiz Khan
	Justification and Suitability for Degree Program	All members
	Individual Contributions	Project Contributions Meia Copeland
		Report Contributions Meia Copeland
3 Requirements	Functional Requirements	Ranishka Fernando, Shawaiz Khan
	Non-Functional Requirements	Ranishka Fernando, Shawaiz Khan
	User Cases	Ranishka Fernando, Shawaiz Khan, and Marwan Zeyada
4 Research on Simulator Design	Measuring Linear Displacement	Meia Copeland and Ranishka Fernando

	Measuring Frequency		Ranishka Fernando and Talal Jaber	
	Methods of Vibration Simulation	Linear Actuator	Meia Copeland and Ranishka Fernando	
		DC Motor	Talal Jaber	
	Database		Shawaiz Khan	
	User Interface		Marwan Zeyada	
5 Vibration Simulator Design	System Design		Meia Copeland	
	Measuring Linear Displacement		Meia Copeland	
	Simulation of Linear Displacement	Flexure-guided Linear Actuator Design	Meia Copeland, Ranishka Fernando	
		Table Design	Meia Copeland	
		Motor Feedback System	Talal Jaber	
	Database		Shawaiz Khan	
	User Interface		Marwan Zeyada	
6 Tradeoff Analysis	Computer and Microcontroller Devices		Meia Copeland	
	Sensors	Linear Displacement Sensor	Meia Copeland	
		Temperature and Humidity Sensor	Meia Copeland	
		RPM Sensor		
	Vibration Simulator	Linear Actuator	Meia Copeland	
		Motor Feedback System	Talal Jaber	
	Database		Shawaiz Khan	
	User Interface		Marwan Zeyada	
7 Budget Breakdown			Meia Copeland	
8 Reflections			General Reflections	Ranishka Fernando, Meia Copeland

	Limitations of Design	Marwan Zeyada
	Future Work and Improvements	Talal, Shawaiz Khan, Marwan Zeyada
9 Conclusion		Marwan Zeyada, Talal Jaber, and Meia Copeland
Appendix 1: Relevant Courses		Meia Copeland
Appendix 2: Additional Diagrams	Measuring Frequency with an Accelerometer	Ranishka Fernando
	Motor	Talal Jaber
	Database	Shawaiz Khan
	User Interface	Marwan Zeyada
Appendix 3: Costs		Meia Copeland
Appendix 4: Work Plan		Meia Copeland and Shawaiz Khan
Appendix 5: Progress Report		All members
Appendix 6: Proposal		All members

2.5.3 Proposal Contributions

Table 4 – Contributions of each team member to the Proposal report.

Proposal			Contributor
Introduction	Background		Meia Copeland
	Motivation		Meia Copeland
	Project Objectives		Meia Copeland
Research	Vibrations to be Simulated		Meia Copeland
	Methods of Simulation	Earthquake Shake Table	Meia Copeland
		Sub-woofers and Haptic Transducer	Meia Copeland
		DC Brushless Motor	Meia Copeland, Talal Jaber, and Ranishka Fernando
		Stepper Motor and Fine Control Linear Actuator	Meia Copeland
	Software	Database Options	Shawaiz Khan
		User Interface Framework	Marwan Zeyada
	Actuator Test Plan		Meia Copeland

System Design	Motor Test Plan		Ranishka Fernando and Talal Jaber
	Shake Table Test Plan		Meia Copeland
	Measurements		Meia Copeland
	Database		Shawaiz Khan
	User Interface		Marwan Zeyada
	Use Cases		Shawaiz Khan
	Evaluation		Shawaiz Khan
Work Plan	Project Team	Roles and Tasks	All Members
		Collaboration	Meia Copeland
	Contributions		Meia Copeland
	Project Milestones		Meia Copeland and Shawaiz Khan
	Schedule of Activities/Gantt Chart		Meia Copeland and Shawaiz Khan
	Risks and Mitigation Strategies		Meia Copeland and Shawaiz Khan
	Project Requirements	Project Requirements	
Stretch Goals		Meia Copeland and Shawaiz Khan	
Budget Breakdown	Hardware		Meia Copeland
	Services		Meia Copeland
Conclusion			Meia Copeland

2.5.4 Progress Report Contributions

Table 5 – Contributions of each team member to the Progress Report.

Progress Report		Contributor
Introduction	Abstract	Ranishka Fernando
	Background	Meia Copeland
	Motivation	Meia Copeland
	Project Objectives	Meia Copeland
	Report Summary	Meia Copeland
	Health and Safety	Shawaiz Khan
	Project Management	Shawaiz Khan

The Engineering Project	Justification and Suitability for Degree Program		All members
	Individual Contributions	Project Contributions	Meia Copeland
		Report Contributions	Meia Copeland
Research	Measuring Linear Displacement		Ranishka Fernando and Meia Copeland
	Methods of Vibration Simulation	Linear Actuator	Ranishka Fernando
		DC Brushless Motor	Talal Jaber
	Database Options		Shawaiz Khan
	User Interface Framework		Marwan Zeyada
Vibration Simulator Design	Simulation of Linear Displacement		Meia Copeland, Talal Jaber, and Ranishka Fernando
	Database		Shawaiz Khan
	User Interface		Marwan Zeyada
Work Plan	Project Milestones		Meia Copeland and Shawaiz Khan
	Schedule of Activities/Gantt Chart		Meia Copeland and Shawaiz Khan
Budget Breakdown	Hardware		Meia Copeland
Conclusion			Meia Copeland

3 Requirements

This project involved an extensive requirement gathering process. The team worked with the Davy lab to determine what research the lab wanted to commit to, and how the team could help. When the research goal of analyzing the effects of ground vibrations on the development of turtle eggs was reached, the engineering team conducted interviews with members of the Davy lab to determine and document the lab member's needs and requirements to meet the research objective. The following section outlines these requirements.

3.1 Functional Requirements

Functional requirements pertain to the essential features and capabilities of the device. In the context of the Davy Lab project, the device must be able to generate controlled vibrations replicating various sources, such as highways and industrial activities, that could affect turtle egg hatching. These vibrations can disrupt the orientation and position of the eggs, potentially leading to developmental abnormalities, reduced hatching success, or even death of the developing hatchlings [5].

To mimic these vibrations given the low frequency (5 – 20 Hz) and RPM, a 3-phase brushless DC motor and a motor driver are ideal to control the motor using pulse width modulation. This motor was chosen because it hits the frequency needed without causing too much noise.

The motor's rotational motion must be converted into linear motion (up and down) of the vibration table surface. Further, the linear motion must be controlled to a low displacement of 0.1 – 10 mm. A flexure-guided linear actuator was used to accomplish this, as it could easily convert rotational motion into linear motion and was customizable through 3D modelling and printing.

Given that the device would only vibrate the surface vertically, a linear displacement sensor is necessary to correctly measure the surface movement and ensure the correct displacement is obtained. Furthermore, the inclusion of an infrared obstacle avoidance sensor module parallel to the motor allows for non-intrusive RPM detection, which is critical for precise vibration frequency control.

In addition to the requirements mentioned, the device should also meet the functional requirements related to data collection, data management, user-friendly interface, and backup and recovery. The device should collect and store the collected data once every 30 minutes, without missing any data points. Moreover, the device needs a feature that allows the user to back up the collected data regularly and recover it in case of any system failures or data losses.

3.2 Non-Functional Requirements

Non-functional requirements, such as scalability and reliability, are vital to ensuring the device's long-term effectiveness. Addressing scalability involves designing the system to accommodate additional tables if necessary, enabling larger sample sizes or concurrent testing of multiple vibration scenarios. The system's scalability is cost-effective, with each additional table costing only 1/3 of the initial system's price. Thus, while the entire system's cost is approximately \$800, adding another table requires an investment of just \$280, as the need for extra hardware is mitigated with the primary system.

Reliability plays a crucial role in the system design, with the device needing to operate continuously for up to 95 days to provide a consistent and accurate assessment of the vibrations' impact on turtle egg development. Ensuring long-term operation necessitates the meticulous selection of components, a robust design, and rigorous testing to validate the device's performance and durability. The current setup meets the required incubation period for continuous operation. Furthermore, maintenance considerations have been tailored to suit a non-technical audience. Most of the system components are 3D printed, with design files openly accessible, and electronic parts are both readily available and affordable as off-the-shelf replacements. This approach ensures ease of maintenance and cost-effective component

replacement. In the case of an off-the-shelf part no longer being available, the design files can be altered to accommodate new parts.

3.3 Use Cases

In the use cases section, we will explore various real-world scenarios where the developed device can be effectively employed.

3.3.1 Use Case 1 – Begin New Experiment

Table 1

<u>Intent</u>	Setting up a new experiment to investigate the impact of ground vibrations on turtle egg growth
<u>Primary Actor</u>	Researcher/Student
<u>Precondition</u>	The system is connected to power and turned on. Simulation table is operational and connected to the UI
<u>Postcondition</u>	The user successfully sets up a new experiment. The parts operate at the intended settings.
<u>Failed Postcondition</u>	Either one of the actuators, accelerometer, temperature sensor or humidity sensor fails to operate. The User Interface does not provide the desired frequency or fail during the experiment
<u>Basic Flow</u>	<ol style="list-style-type: none"> 1. Researcher sets up turtle eggs on the simulation table. 2. Navigate through the touch screen interface. 3. Touch the “Settings” UI button on the top right of the screen. 4. Enter an experiment name, desired frequency, desired amplitude, email addresses, etc. in the given spaces. 5. Choose the unit for Temperature, review the details entered and touch the ‘Save Changes’ UI button. 6. Press start experiment and simulation table starts vibrating.

3.3.2 Use Case 2 – Change the vibrational Frequency.

Table 2

<u>Intent</u>	Changing the vibrational frequency applied by the actuator.
<u>Primary Actor</u>	Researcher/Student
<u>Precondition</u>	The system is connected to power, ON, and an experiment has been set up and is running; The UI displays the home screen.
<u>Postcondition</u>	The user successfully changed the vibrational frequency to the desired setting. The system uses the infrared sensor to measure RPM and calculate the frequency and display a confirmation notification.
<u>Failed Postcondition</u>	Vibration profile is not saved or does not provide the desired frequency
<u>Basic Flow</u>	<ol style="list-style-type: none"> 1. Navigate through the touch screen interface. 2. Researcher selects the “Settings” UI button on the top right of the screen. 3. Researcher inputs desired frequency and amplitude parameters. 4. Vibration parameters are saved and applied on the experiment.
<u>Alternate Flow</u>	<p>Change only vibrational frequency.</p> <ol style="list-style-type: none"> 1. Researcher touches the Vibration frequency indicator. 2. Frequency value is now editable, and Researcher inputs the new value. 3. The appropriate vibration parameter gets saved and applied on the system

3.3.3 Use Case 3 – Exporting the Experiment Data

Table 3

<u>Intent</u>	Extract the data collected for analysis during experiment is running
<u>Primary Actor</u>	Researcher/Student
<u>Precondition</u>	The system is connected to power, ON, and an experiment has been running; The UI displays the home screen.
<u>Postcondition</u>	The user returns to main screen of the UI and real time data is saved locally and sent to email.
<u>Basic Flow</u>	<ol style="list-style-type: none"> 1. Navigate through the touch screen interface to the Settings Page 2. Add any desired email addresses in the respective fields (Left Side of the Page) 3. Touch the UI box/button displaying the export data. 4. Choose to send the data to email. 5. Data sent to the location of the researcher's choice.

3.3.4 Use Case 4 – Monitoring Shaker Table Performance

Table 4

<u>Intent</u>	Monitor the performance of the shake table, including frequency, temperature, and time
<u>Primary Actor</u>	Researcher/Student
<u>Precondition</u>	The system is connected to power, ON, and an experiment is running; The UI displays the home screen.
<u>Failed Postcondition</u>	Shake table does not provide the desired frequency or fails during monitoring
<u>Postcondition</u>	Researcher has access to real-time performance data of the shake table
<u>Basic Flow</u>	<ol style="list-style-type: none"> 1. Researcher accesses the UI home screen. 2. Researchers can monitor different important information in real-time displayed clearly in the UI. (Frequency, temperature, time, etc.) 3. More in depth data with timestamps is available using the flow described in Use Case 3
<u>Alternate Flow</u>	None

3.3.5 Use Case 5 – Verify Motor RPM Measurements

Table 5

<u>Intent</u>	Validate the accuracy of the infrared sensor RPM measurements to ensure reliable operation
<u>Primary Actor</u>	Service Technician
<u>Precondition</u>	Simulation table is connected to the UI, motor and infrared sensor are operational but not running an experiment in order to conduct routine maintenance.
<u>Postcondition</u>	Infrared sensors RPM measurements are verified as accurate and reliable
<u>Failed Postcondition</u>	Infrared sensors RPM measurements are inaccurate or unreliable
<u>Basic Flow</u>	<ol style="list-style-type: none"> 1. Service Technician access the UI. 2. Service Technician to run a series of tests at different motor speeds using the UI. 3. Compare infrared sensor measurements with an external reference or manual calculation. 4. If the measurements are accurate, the test is successful; otherwise, the troubleshoots the sensor and repeats the test.
<u>Alternate Flow</u>	None

4 Research on Simulator Design

4.1 Measuring Linear Displacement

To ensure that the correct displacement is achieved, the displacement of the surface must be measured. Since the device will only be vibrating the surface up and down, a linear displacement sensor can be used. These sensors are available with many different technologies, such as linear potentiometers, linear variable differential transformers, capacitive displacement sensors, laser displacement sensors, and digital indicators. Due to budget constraints of the project that will be discussed later in this document, the sensor used would ideally be under \$500 to implement.

Linear potentiometers are sensors that use variable resistance to measure position. Linear motion is converted to a changing resistance which can be directly converted to a voltage or current output. This output can be read by a computer to determine the displacement of the measured surface. Due to being an analog sensor, potentiometers have infinite resolution, however the smaller the measurement, the more it is affected by factors such as noise and the number of bits needed to convert to a digital value. The downside to using a potentiometer is the physical movement of the measuring device. Most potentiometers use a mechanical piece that moves to create resistance, which is subject to wear and tear. Since for one experiment, a slider would be expected to move up to 164 million times (20 Hz for 95 days), this greatly impacts

the choice of sensor. Most sensors within a reasonable price range only have a lifespan of up to a 20 million cycle [9].

Linear variable differential transformers (LVDTs) convert the mechanical motion of an object, usually a tubular element that can move along a rod, into an electrical signal using induction. Preliminary searches show that LVDTs capable of measuring within the desired range of 0.1 – 1 mm and with sampling frequency of over 100 Hz are within budget [10]. However, these sensors also have a short life span relative to what is required, with one such LVDT we inquired about only capable of 10 million cycles, which would only last one quarter of the duration of an experiment of 95 days that uses a frequency of 5 Hz.

Capacitive sensors are no-contact sensors, which use an electric field to detect the target's position thanks to interference with the electric field. They are analog sensors and technically have an infinite resolution with the same stipulations as the linear potentiometer. Measuring displacement using an electric field eliminates the risk of wear and tear, as the device would not be in contact with a moving surface. However, upon inquiring about prices, sensors capable of detecting displacement within the desired range were over the budget of \$500 for the sensor, and an additional \$5000 for the required signal processing unit and cables.

Laser displacement sensors, often known as point lasers, use triangle reflection to measure a single point. Laser profilers, on the other hand, measure the complete length of a line. The measurement precision of laser displacement sensors is great, but the efficiency is low due to point-by-point data collection [9]. Given the precision, using a laser to measure linear displacement would be ideal. However, it does not fit within the price point. Conducting laser measurements for the desired range of 0.1 – 1 mm exceeds over a thousand dollars per sensor or instrument that is available to be purchased off the shelf.

Finally, digital indicators were investigated as a simple measurement tool [11]. These are the digital counterpart to dial indicators and have a rod that can be moved up and down to measure displacement. This rod has an attached optical reader, which reads microscopic marks along a thin glass scale. This allows the digital indicator to be extremely precise for a relatively low price, with a resolution of 0.001 mm available for under \$100. However, a digital indicator cannot measure at as high a frequency as some of the other options. This has been discussed with the lab, and a solution will be provided in the project design section.

4.2 Measuring Frequency

To determine the frequency at which the shaker table vibrates, the process of connecting an accelerometer to an Arduino microcontroller is employed to measure and analyze the z-axis vibration data. The raw data from the accelerometer undergoes processing and conversion into the frequency domain through the Fourier Transform. By calculating the frequency spectrum of the accelerometer data, valuable insights into the system's vibrational characteristics can be

obtained. This information is beneficial for a variety of applications, including condition monitoring, modal analysis, and vibration control. Furthermore, several metrics, such as changes in velocity, displacement, and direction, may be calculated by evaluating this data [12].

The connection between the Arduino Uno R3 and the ADXL 335 accelerometer is established through the analog pins on the Arduino board. The accelerometer's z-axis output pin is specifically attached to the Arduino board's A2 analog pin. Moreover, the VCC (power supply) and GND (ground) pins of the accelerometer are linked to the Arduino's VCC and GND pins, respectively, to guarantee correct power supply and reference ground [12]. The diagram in Figure 1 illustrates the wiring connections between the ADXL335 accelerometer and the Arduino Uno R3 for effective data measurement and analysis.

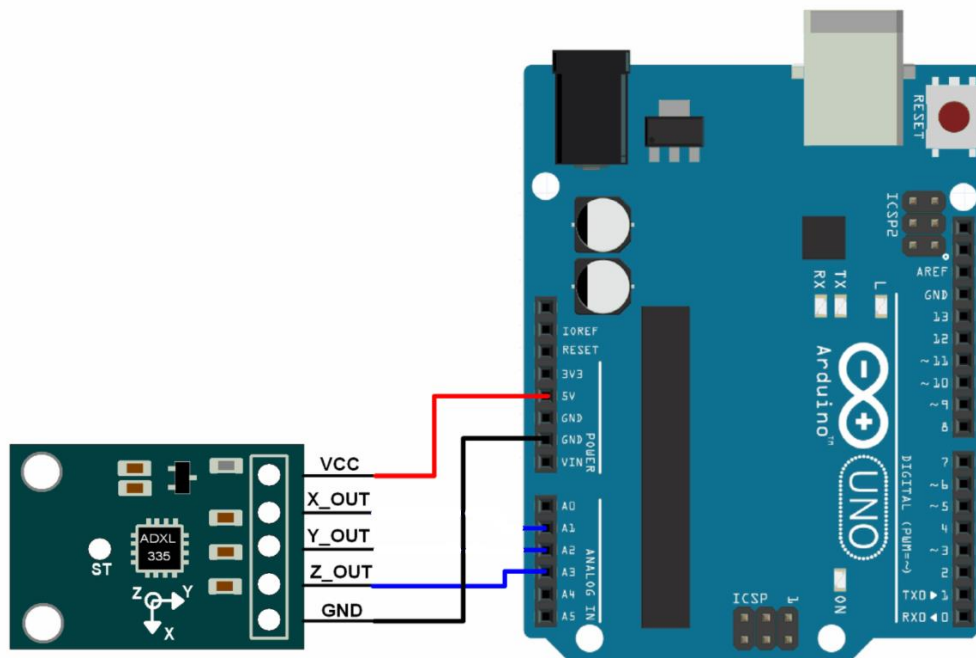


Figure 1 – ADXL335 Accelerometer Connected to Arduino Uno R3 for Vibration Data Measurement and Analysis

The accelerometer measures the acceleration in three axes (x, y, and z), and the Arduino processes the raw data from the z-axis, converting it into meaningful acceleration values and calculating the frequency using the code designed that is attached in the appendix. The Arduino IDE, running on the PC, is utilized to program the Arduino board and monitor the output data, making it an essential tool in the process of measuring frequency using the accelerometer.

One common approach to measuring frequency is to use a Fourier transform on the accelerometer's raw data. The included Arduino code (see Appendix 2: Additional Diagrams - Measuring Frequency with an Accelerometer) shows how to detect frequency using an ADXL 335 accelerometer attached to an Arduino Uno R3. The code begins by defining the number of samples (SAMPLES), sampling frequency (SAMPLING FREQ), ADC range (ADC RANGE), and

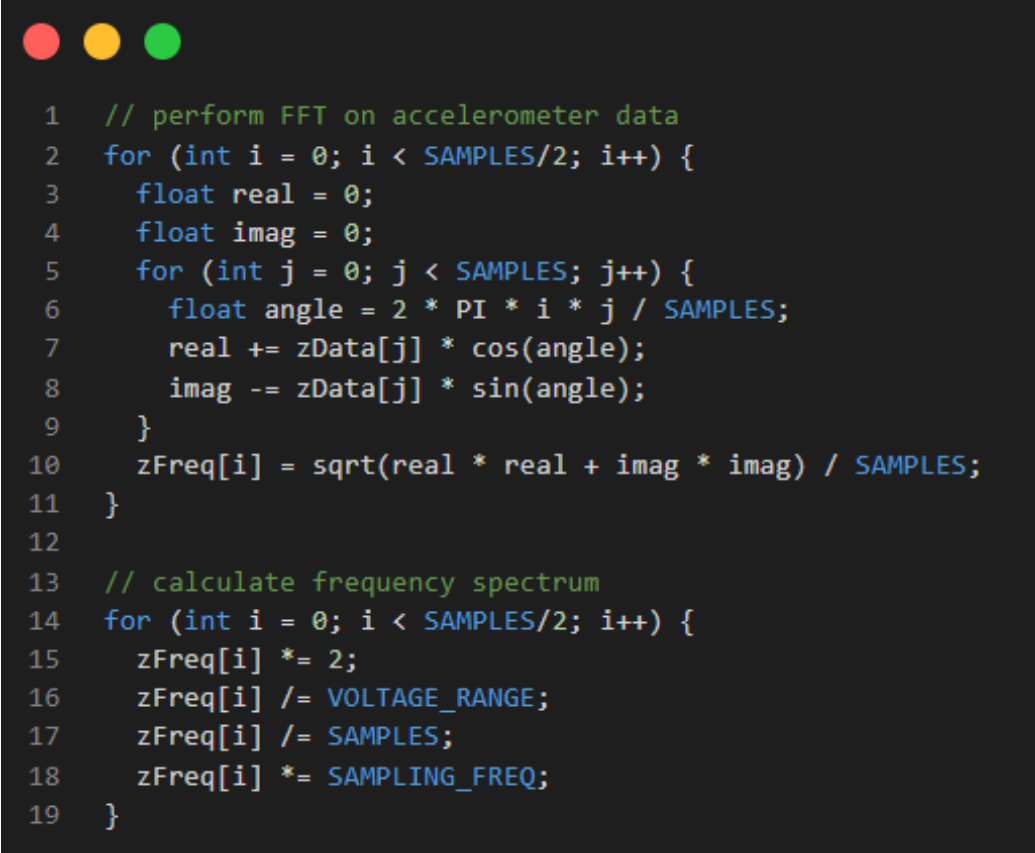
accelerometer output voltage range (VOLTAGE RANGE). It then defines the analog pin for the z-axis input and initializes variables for storing raw sensor readings, accelerometer readings in g (gravitational force), and arrays for storing accelerometer data and frequency spectrum.

The serial communication is configured to 9600 bps in the `setup()` function. The `loop()` function starts by reading the raw accelerometer data and converting it to an accelerometer reading in g. The conversion considers the ADC range, voltage range, and accelerometer sensitivity (330 mV/g for the z-axis). After that, the accelerometer data is added to the FFT array.

The team stumbled across a problem with the Arduino IDE, as it was unable to recognize the Fourier Transform library, making it impossible to use the `FFTArduino` library for this project. To overcome this obstacle, the team opted to perform the Fourier Transform manually within the code. By defining the Fourier Transform function directly in the code, the team ensured that the frequency measurement could still be accurately calculated without relying on external libraries, ultimately maintaining the functionality intended.

The code performs the Fourier transform manually on the accelerometer data by iterating over it and computing the real and imaginary components of the frequency spectrum. The frequency spectrum is calculated by multiplying the result by 2, dividing it by the voltage range and sample number, and then multiplying it by the sampling frequency. The reason for dividing by 2 is related to the fact that the Discrete Fourier Transform (DFT) represents both positive and negative frequencies, with each frequency component mirrored around the Nyquist frequency. In practice, the DFT is symmetric, meaning that the values for positive and negative frequencies are equal. Consequently, by dividing by 2, the frequency components are effectively averaged, considering only the unique information from the first half of the DFT. The frequency spectrum is then normalized by dividing by the voltage range and the number of samples. This normalization process ensures that the frequency values are independent of the input signal's amplitude and the number of samples used in the DFT. Finally, the frequency components are scaled by the sampling frequency to convert them into real-world frequency values (in Hz) [13].

The code snippet provided in figure 2 demonstrates how to conduct the DFT manually and calculate the frequency spectrum, highlighting the steps of calculating the real and imaginary components, normalizing the results, and scaling by the sampling frequency.



```

1  // perform FFT on accelerometer data
2  for (int i = 0; i < SAMPLES/2; i++) {
3      float real = 0;
4      float imag = 0;
5      for (int j = 0; j < SAMPLES; j++) {
6          float angle = 2 * PI * i * j / SAMPLES;
7          real += zData[j] * cos(angle);
8          imag -= zData[j] * sin(angle);
9      }
10     zFreq[i] = sqrt(real * real + imag * imag) / SAMPLES;
11 }
12
13 // calculate frequency spectrum
14 for (int i = 0; i < SAMPLES/2; i++) {
15     zFreq[i] *= 2;
16     zFreq[i] /= VOLTAGE_RANGE;
17     zFreq[i] /= SAMPLES;
18     zFreq[i] *= SAMPLING_FREQ;
19 }

```

Figure 2 – Code snippet of conducting Fourier transform manually and calculating frequency.

The code outputs the frequency to the serial monitor after computing the frequency spectrum. To achieve an acceptable sampling rate, a delay of 100 ms is introduced before capturing the next sample.

Although this approach used an accelerometer to determine frequency, it was discovered to have a 4.7-5.3 Hz offset when the accelerometer remained stationary. Because the frequency range of interest is 5-20 Hz, this offset is relevant for the experiment. To acquire the true frequency value, a 5 Hz offset would be required, but the team could not confirm whether the offset is constant or caused by a malfunctioning accelerometer.

Consequently, the team opted not to pursue this option, although it did give useful insight into alternate frequency measuring methods. Section 5.2.3 of the report explains the chosen solution. The flow diagram in Figure 3 illustrates a high-level overview of the implementation of the code.

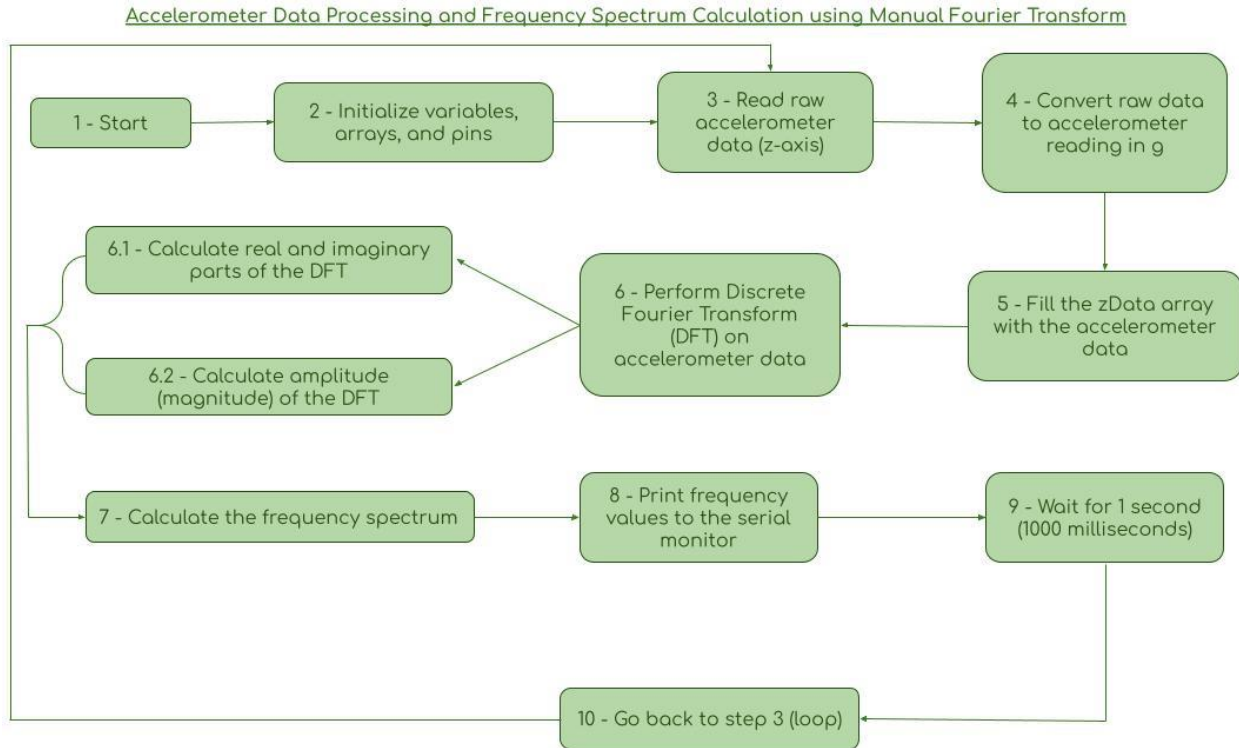


Figure 3 – Overview of the code used for Accelerometer Data Processing and Frequency Spectrum Calculation using Manual Fourier Transform

The team opted to instead explore the use of a tachometer as a potential solution for calculating frequency by detecting the RPM of the DC motor shaft. However, upon further investigation, it was discovered that a suitable tachometer small enough to detect the rotation of a 2 mm DC motor shaft was not readily available. Subsequently, the team shifted its focus to exploring alternative options.

After extensive research and testing, the team arrived at a viable solution which entailed the use of an infrared (IR) sensor module. This module can detect the rotation of the gear by reflecting the light of a spinning disk, as illustrated in Figure 4. As the black strip passes in front of the sensor, the module counts one spin and calculates the RPM accordingly.

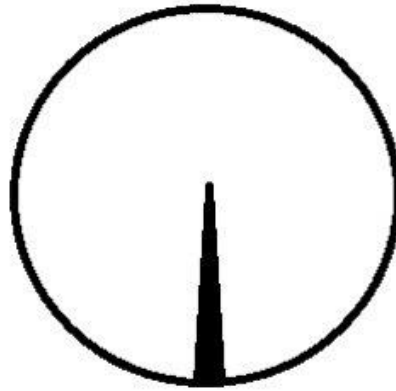


Figure 4 – Disk attached to gear

The IR sensor module is powered by the 5V input of the Arduino, and further details on how this technology works can be found in section 5.2.3. This solution has proven to be highly effective, and the team is confident in its ability to accurately measure frequency for our project's requirements.

4.3 Methods of Vibration Simulation

Over the course of the turtle eggs' 65 to 95-day incubation cycle, vibration treatment would need to be applied continuously or in intervals. Methods to create the up and down linear displacement needed for the treatment will be presented here.

4.3.1 Linear Actuator

A linear actuator is a device that converts some type of input into linear motion for the purpose of pushing, pulling, sliding, or otherwise moving things [14]. Traditionally this is done with a series of gears and a lead screw that converts the rotational motion of a motor into linear motion. Other linear actuators use flexures powered by motors or piezoelectric devices.

Linear actuators were investigated due to the desired motion being contained to one axis. Simple, traditional linear actuators are available off the shelf, however the precision needed (0.1 mm – 1mm) is a limiting factor with these actuators. Further, the actuator needs to change direction at 5 Hz – 20 Hz. Mechanical systems that utilize gears are subject to backlash (Figure 5), which occurs when there is space between gear teeth—or “play”—which the gear must overcome before contact between teeth is achieved. This occurs when gears begin moving or change direction. Backlash can cause additional vibration and affect gear performance until teeth meet. Due to this effect and the fact that the actuator would have to change direction at high speeds, achieving the

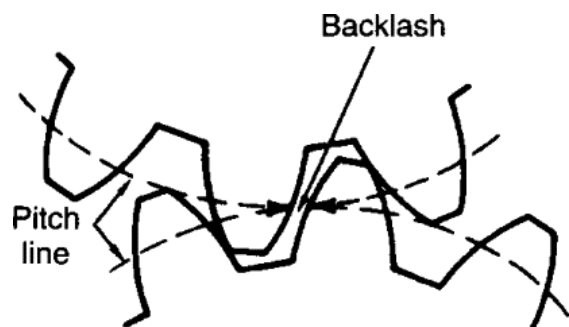


Figure 5 – Demonstration of backlash between gears

desired frequency would not be possible without severely affecting the performance and positioning of the actuator.

Flexure-guided linear actuators use a flexure system that can either amplify small movements or reduce larger movements while translating the movement into a linear motion. Piezoelectric actuators use a piezo device which deforms when voltage is applied [15]. This deformation is then converted into linear motion with a flexure. These devices are common in nano positioning devices for very precise movement. The desired linear displacement is easily achieved with these

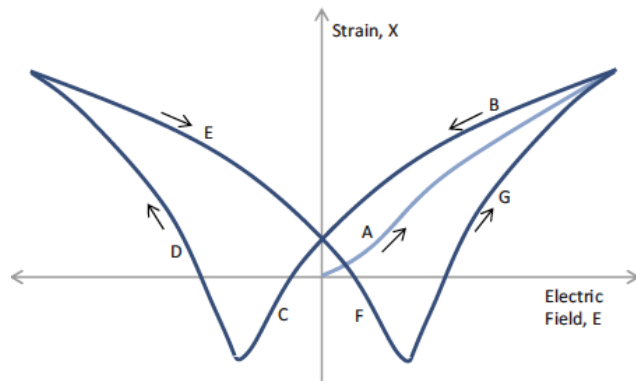


Figure 6 – Strain vs. voltage curve for a typical piezo material when alternating voltages are applied [16]

devices, however the frequency of the movements presents a challenge. Piezo devices are prone to hysteresis (Figure 6) which could affect the performance as it must reverse the electric field at high speeds to change the direction of displacement at the desired frequency [16]. This has a similar effect to the backlash described for traditional linear actuators, where hysteresis causes an response when the direction of motion changes that impacts the accuracy of displacement.

Flexures can also be used to translate rotational motion from a motor into linear motion. One example of this type of flexure is the fine stage actuator of the James Webb Telescope (JWST) mirror positioning system, designed by Ball Aerospace [17]. These types of flexures can be driven by DC or stepper motors depending on the movement desired, which is coupled to the flexure with an offset camshaft. This type of flexure is capable of precise movements at high speeds, with the speed depending on the driving motor. Neither backlash or hysteresis are an issue with this type of flexure, as the offset camshaft only needs to rotate in one direction to drive the mechanical up and down motion of the flexure.

A replica of the JWST flexure design is available under a Creative Commons – Attribution – Non-Commercial license from user Polyfractal (Zachary Tong) on Thingiverse.com [18]. This design can be almost fully 3D printed, apart from some shafts, screws, bearings, and the motor. It is capable of a 26 μm displacement on its own and can be altered to achieve the desired displacement of 0.1 – 1 mm by changing the dimensions of the flexure. This solution is extremely cost-effective, as the Carleton Library offers free commercial-grade 3D printing services to students and faculty. Other components are inexpensive and easy to find on online marketplaces such as Digikey and Amazon.

4.3.2 DC Motor

Initially, a vibrating DC motor was investigated to produce vibrations by acting directly upon the table surface. This involved a simple vibrating motor attached under a tabletop with some insulating pads near the legs for eliminating noise. First looked at was a brushless 3-phase motor with a pulse-width modulation (PWM) controller to control the revolutions per minute (RPM) of the motor. PWM is a way to control the input power by pulsing the signal which reduces the average voltage. RPM is directly proportional with the frequency of vibrations. However, most small motors have a limit where the vibration frequency stops decreasing at around 30 Hz which is the minimum vibrations a normal vibrating motor can go due to the size and weight constraint of the motor and shaft. We are trying to achieve 5 to 20 Hz which is not possible with these types of motors. (Figure 7) shows that most vibration motors that are not custom built have a cut-off frequency 30Hz when decreasing voltage supply to control the speed.

TYPICAL DC MOTOR PERFORMANCE CHARACTERISTICS

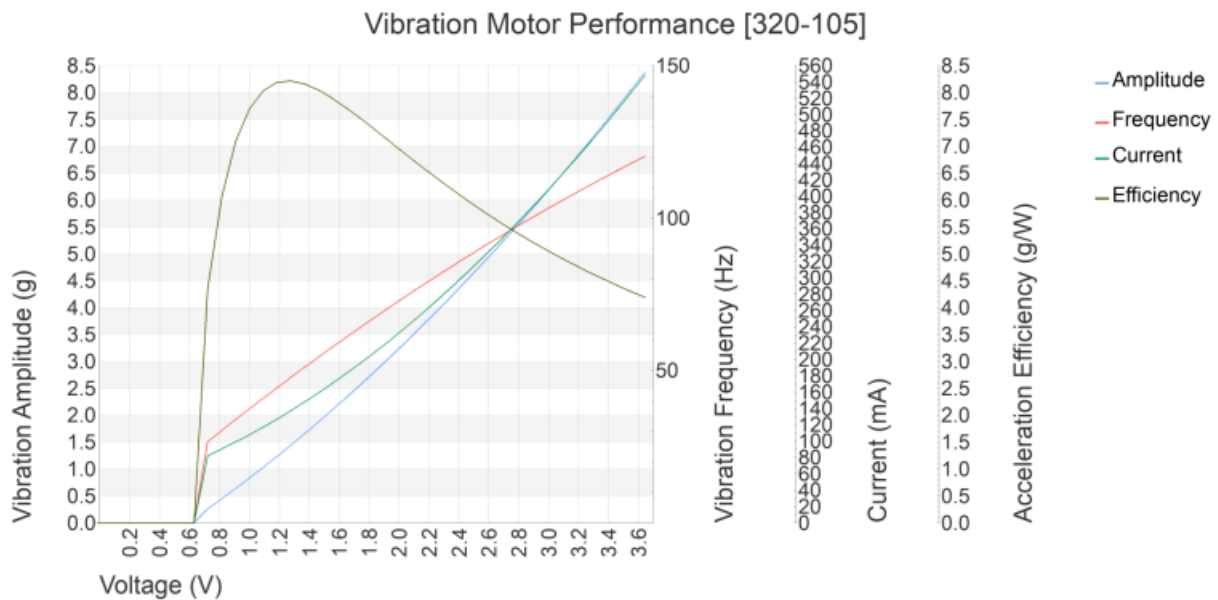


Figure 7 – DC motor frequencies [19]

Once the flexure-guided linear actuator was considered, a new solution was needed. The JWST design uses a stepper motor to achieve precision adjustments. However, this design requires continuous and smooth operation, which a DC motor can provide. The motor used in this project is a 5V DC motor, which is connected to gears to transmit mechanical power to the linear actuator. To control the motor, an L298N driver (Figure 8) is used, which is connected to an Arduino Uno microcontroller. This driver provides the necessary voltage and current to the motor to control its speed and direction. The driver can run 2 motors at the same time and control them using different PWM signal. The figure below shows the connections of the L298N driver. The driver needs a 12V dc power supply to run [20].

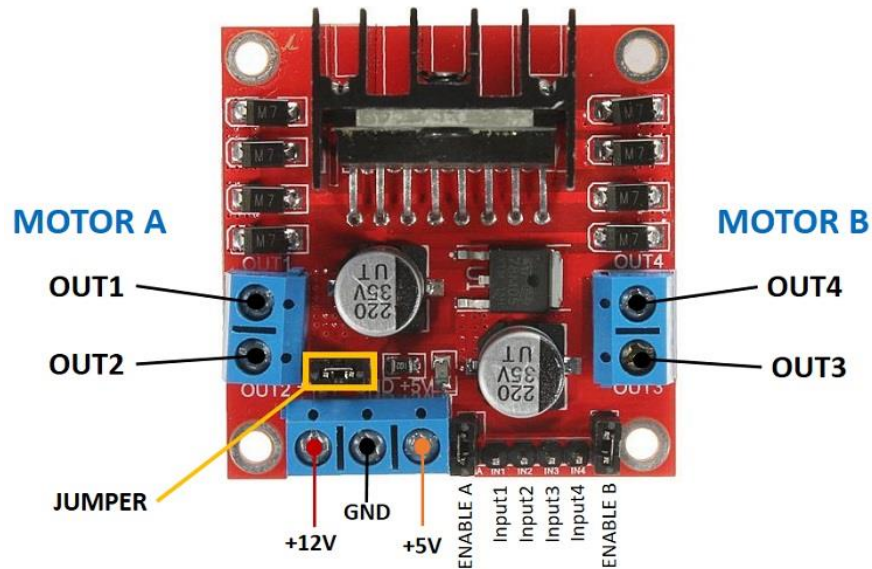


Figure 8 – L298N Motor Driver

By controlling the motor's RPM, the speed and performance of the gears can be optimized to achieve the desired output. This setup enables the motor to run smoothly and efficiently while providing accurate feedback on its speed. Overall, this system provides a reliable and effective way to control the motor and ensure the proper functioning of the gears.

Table 6 in part 5.3 shows the speeds needed to achieve the desired frequencies. The motor is connected to the linear actuator with a gear reduction system. Since DC motor have less noise when running at higher speeds.

4.4 Database Selection to Store Acquired Data

Keeping in mind the requirements of this project and at the request of the Davy lab, a database was needed to store and retrieve data collected from various sensors of the design. A database comprises of three primary components: a database management system (DBMS), a query language, and a database schema [21]. The actual storing and retrieval of data is handled by the DBMS, and users may interact with the data by sending queries to the DBMS using the query language. Furthermore, a database schema describes how data is arranged and stored inside a database's logical structure, acting as a guide for the database management system to ensure the consistency and correctness of the data. A database schema is included in Section 5.4.1 of the proposal to further explain and illustrate the design.

Given the duration of the incubation period (up to 95 days), our goal was to find a reliable database management system that could efficiently store and retrieve the large volumes of data gathered from the sensors. Different types of database management systems were identified during the research, offering various options for the project. "Relational Database Management Systems" (RDMS) were compared with "Not only Structured Query Language" (NoSQL) during the

research. Initially, consideration was given to using a NoSQL database, specifically Firebase, due to its real-time updates to a remote server [22]. However, after careful evaluation of the cost implications associated with using Firebase, the decision was made to opt for a RDMS instead. RDMS was considered a reliable choice for the project due to its widespread use and prior experience working with it. Moreover, according to industry statistics, RDMS is the most used type of database management system [23] which made it a reliable choice for this project. A day-to-day example of RDMS is banking [24]. A bank uses an RDMS to store customer account details, transaction details, loan details, and other financial data. An example can be seen in Figure 9, where all the information is stored in a relational table like format.

Date	Description	Ref.	Withdrawals	Deposits	Balance
2003-10-08	Previous balance				0.55
2003-10-14	Payroll Deposit - HOTEL			694.81	695.36
2003-10-14	Web Bill Payment - MASTERCARD	9685	200.00		495.36
2003-10-16	ATM Withdrawal - INTERAC	3990	21.25		474.11
2003-10-16	Fees - Interac		1.50		472.61
2003-10-20	Interac Purchase - ELECTRONICS	1975	2.99		469.62
2003-10-21	Web Bill Payment - AMEX	3314	300.00		169.62
2003-10-22	ATM Withdrawal - FIRST BANK	0064	100.00		69.62
2003-10-23	Interac Purchase - SUPERMARKET	1559	29.08		40.54
2003-10-24	Interac Refund - ELECTRONICS	1975		2.99	43.53
2003-10-27	Telephone Bill Payment - VISA	2475	6.77		36.76
2003-10-28	Payroll Deposit - HOTEL			694.81	731.57
2003-10-30	Web Funds Transfer - From SAVINGS	2620		50.00	781.57
2003-11-03	Pre-Auth. Payment - INSURANCE		33.55		748.02
2003-11-03	Cheque No. - 409		100.00		648.02
2003-11-06	Mortgage Payment		710.49		-62.47
2003-11-07	Fees - Overdraft		5.00		-67.47
2003-11-08	Fees - Monthly		5.00		-72.47
*** Totals ***			1,515.63	1,442.61	

Figure 9 – An example of a RDMS in banking [25]

After determining that a RDMS was the best choice for the project, it was compared to different options available under RDMS, including MySQL, PostgreSQL, and SQLite. Following evaluation, the choices were narrowed down to SQLite and MySQL [26]. Initially a MySQL database was considered and partially implemented as our RDMS for the project due to its ability to handle larger amounts of data and remote access capabilities, however, its implementation on the Raspberry Pi proved to be challenging due to the use of MariaDB, a MySQL fork [27]. The lack of adequate documentation and resources online further reinforced our decision to select SQLite. Additionally, SQLite's lightweight nature, small memory footprint, and serverless architecture [28] made it an efficient and cost-effective solution for storing and retrieving data collected. A trade-off analysis is presented later in Section 6.3 of this report.

Choosing SQLite meant using SQL as the query language. This meant taking advantage of a well-established and widely used language, allowing for efficient data management and manipulation [29]. The language's syntax is straightforward, making it easier to write complex queries and

retrieve specific data points from large datasets. Additionally, SQL's ability to join tables and perform complex operations enables the lab to perform a more in-depth analysis of the data collected.

4.5 User Interface Framework

The User Interface (UI) is one of the main components that determines the user experience and how effectively the device can be used, it's composed of several components, including buttons, menus, text fields, and graphics. Each of these components plays a critical role in providing a clear and intuitive interface for users. To design a UI properly, it is essential to understand the purpose of the application, the target audience, and the user's goals. It is also crucial to maintain consistency throughout the design, using the same visual language and layout to create a cohesive user experience. After researching the various options of frameworks available on the market. 2 options seemed the most suitable for this project: Kivy and PyQt5.

Kivy and PyQt5 [30] are two of the most used Python Graphical User Interface (GUI) frameworks in the industry. Kivy, was made from the ground up for mobile GUI design, with the purpose of making clean modern looking GUIs that can be used on most system software like Linux, Windows, macOS, and Raspberry pi. It has great documentation but lacks community online resources.

PyQT5 on the other hand, has almost all the same features as Kivy but has an extra important feature that will be of great use in this project. This feature is called QTDesigner [31], it is a program that allows you to seamlessly create GUIs by designing them on a 2D plane by adding elements in a drag and drop fashion, and then adding functionality and style using python code. This allows for beautiful looking graphical interfaces that can be made with ease.

Based on the discussed features of PyQt5, mainly the QT Designer, it was chosen to be the best framework to be used to design and build the User Interface. The Davy lab and the project team on multiple occasions discussed and agreed on features and elements expected to be in the UI's design, multiple iterations were provided, and the design and functionalities were tweaked until a satisfactory version was reached. Because of the designer software that is part of PyQt5, the development times of each iteration were significantly reduced because it proved the ability to change the design and elements of the pages and dynamically add them in the code without removing any previously written user code. Overall, the use of PyQt5 proved to be a wise choice, providing an efficient, fast, and highly customizable solution that is cross-platform for building the project's UI.

5 Vibration Simulator Design

This section discusses the design of the entire ground vibration simulator system solution. An overview of the system in its entirety is presented, followed by detailed designs of each component. These details cover design choices, trade-off analyses, and how the components contribute to the rest of the design.

A video demonstrating the system has been produced and is available:

<https://www.youtube.com/watch?v=OBzAEXO5CB8>

5.1 System Design

The vibration simulator design system can be broken down into three systems that interact with one another, as seen in Figure 10.

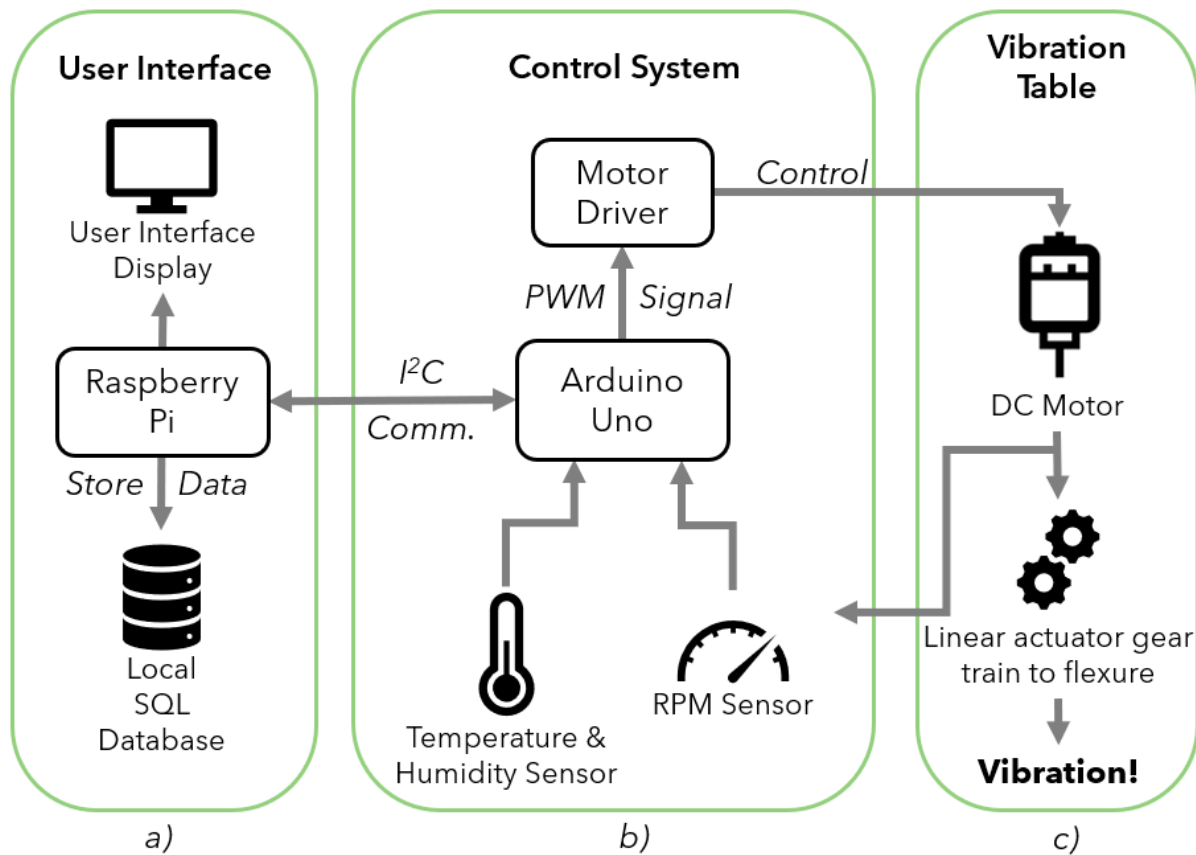


Figure 10 – System block diagram

The front- and back-end (Figure 10a) is a system with a Raspberry Pi 4 Model B computer at its heart. This computer hosts the User Interface application and local databases, which will be described in later sections. To interact with the application, only a monitor, keyboard and mouse are needed. The Raspberry Pi is capable of wired or wireless internet connection, which is recommended so that experimental results can be sent to users via email. The Raspberry Pi

collects data and sends commands through an Inter-Integrated Circuit (I²C) connection with the microcontroller system. I²C is a serial connection protocol that allows data to be transmitted bit by bit, where data is addressed and can be accessed on either side through that address. The User Interface allows the user to setup new experiments, filling pertinent data such as turtle species (or other animals species), length of experiment, and desired frequency. When the experiment is started, the Raspberry Pi will send a signal to the Arduino to begin vibration. It will collect data every 30 minutes from the sensors connected to the Arduino and add this data to the local database. Once the experiment ends, the user will have options on the User Interface to export the collected data to a spreadsheet that is sent to emails submitted by the user.

The microcontroller system (Figure 10b) is controlled by an Arduino Uno microcontroller. The Arduino collects data through various sensors such as a temperature and humidity sensor and an RPM sensor. These sensors will be described section 5.2.2. The data collected from these sensors is sent to the Raspberry Pi through an I²C connection when requested. The Arduino receives commands through the same connection, specifically regarding the frequency of vibration. This frequency is translated into RPM, and a feedback system consisting of the motor driver and RPM sensor is used to change the speed of the motor. Details of this system will be given in a later section.

The vibration table (Figure 10c) is a mechanical system, seen in Figure 11, consisting of a DC motor coupled to a gear train (1) that controls the vibration of a surface. The vibration is created by a flexure-guided linear actuator (2) that translates rotational motion into linear motion and will be described in section 5.3.1. The surface being vibrated is suspended between springs (3) and is attached to a linear bearing (4) to isolate any motion to one axis, up and down. This system will be further described in a later section 5.3.2.

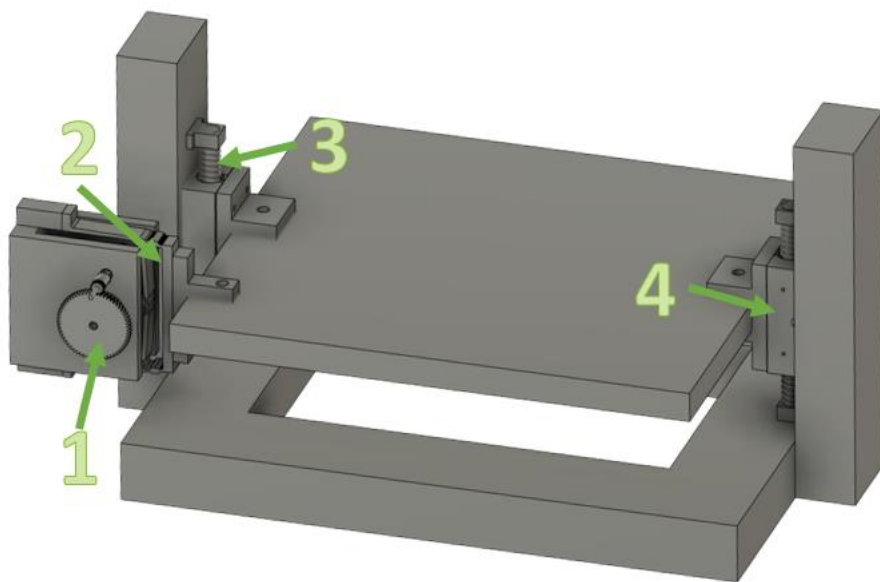


Figure 11 – Vibration table system

5.2 Sensors

5.2.1 Measuring Linear Displacement

To measure the small displacement created by the simulation table, a Mitutoyo 543-783 Absolute Digital Indicator was selected. This solution is low cost (approximately \$300), and one sensor can be used for all experiments. The indicator has a resolution of 0.01 mm.



Figure 12 – Digital indicator set-up

As can be seen in Figure 12, the indicator (1) is attached to an adjustable arm with a magnetic base (2). A metal plate (3) is attached to the frame of the simulation table so that the base can stay in place. The indicator is placed over the table, with the needle touching and slightly depressed. As the table moves, the displacement can be read on the indicator.

The purpose of measuring the displacement is to ensure that the table has not run into any problems, as the displacement values should stay consistent throughout the experiment. If abnormal values are found, the linear displacement mechanism may have broken or is not operating properly, and the experiment is compromised.

In the current design, the values must be manually read and tracked by the user. To do so, a setting was designed for the User Interface that executes a measuring protocol.

The motor slows down to a very low frequency where the user can read the values on the indicator as the table moves. The user then enters the values in the User Interface, and it is tracked in the database with other variables. The measuring only needs to occur for one to two minutes at intervals of once every two to three weeks. Since the experiment takes place over 65-95 days, these small periods for measuring should not affect experimental results.

This solution using a digital indicator was selected for its simplicity and low cost, with minimal effects on the experiment. The inconvenience of manually reading the measurements was approved by the Davy Lab and is offset by the cost and convenience of only needing one sensor.

5.2.2 Temperature and Humidity

For the duration of the experiment, the Davy Lab will be maintaining the environmental conditions of the laboratory to ensure optimal development of the turtle eggs. A sensor to measure temperature and humidity is required for the lab to measure these environmental variables, and to track in data collection.

A DHT22 sensor module (Figure 13) was selected for this project due to its accuracy and longevity. Further, the module has a compact size, low energy consumption, and is easy to interface with using the Arduino Uno. It has an accuracy of $\pm 2\%$ RH for humidity and $< \pm 0.5^\circ\text{C}$ for temperature, which is sufficient for the sake of maintaining an appropriate environment for reptile eggs. The DHT22 has a sensing time of 2 seconds per reading, which is sufficient for this use case where readings will be taken every 30 minutes.

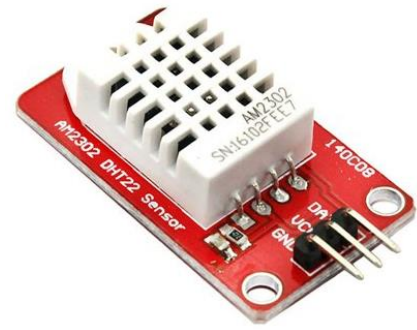


Figure 13 – DHT22 Sensor Module

The most attractive aspect of using the DHT22 is the Arduino library provided by Adafruit for ease of use [32]. This library is also available for the DHT11, which was considered but not selected due to its inaccuracy in comparison to the DHT22.

5.2.3 RPM Sensor

The team has incorporated a simple, yet effective IR sensor module (Figure 14) to accurately measure the RPM of the motor. This sensor operates by detecting the interruptions caused by the revolutions of the motor's shaft, transmitting signals to the microcontroller to calculate the RPM utilizing a mathematical formula.

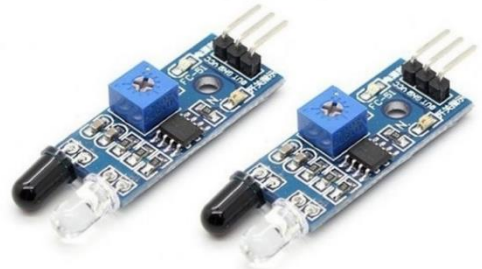


Figure 14 – IR Sensor Module

Despite its simplicity, this sensor is highly effective in its ability to accurately measure the motor's RPM. In fact, it has been specifically designed to measure speeds up to 13,000 RPM, which far exceeds the requirements of the project. While it may not possess the most sophisticated technology, the team has determined that it is an ideal fit for the project due to its reliability, affordability, and ease of use.

By utilizing this sensor, the team has ensured that the system is both accurate and reliable, which is essential for collecting precise data and ensuring optimal experimental outcomes. Thus, the incorporation of this simple yet efficient sensor has been a valuable asset in enhancing the overall performance of the system.

5.3 Methods of Vibration

5.3.1 Flexure-guided Linear Actuator Design

The 3D model replica (Figure 15) of the JWST Mirror fine stage actuator used as a starting point for the design is available under a Creative Commons Non-Commercial license, provided by user Polyfractal on Thingiverse.com [18]. The specific components that will be used from the print are the flexure (1), frame (2), and camshaft (3).

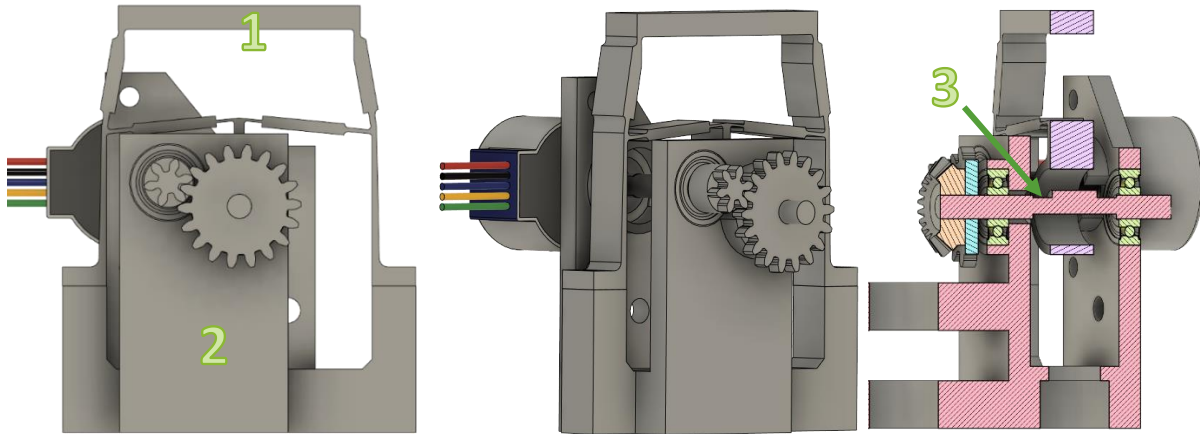


Figure 15 – Original design of the JWST fine positioning stage by Polyfractal, available on Thingiverse.com [18].

This model replicates the James Webb Space Telescope (JWST) mirror actuator fine positioning stage, as described in a research paper by Ball Aerospace engineer Robert Warden [17]. The flexure and camshaft of this design can be altered to achieve the correct amount of displacement. Polyfractal's flexure is capable of 26 μm of displacement given a 1 mm offset on the camshaft. When attached to the table, this displacement is less than 26 μm and so small it cannot be reliably measured. By changing the camshaft offset or the width/height of the flexure, the displacement can be increased to achieve desired results.

The flexure model was simulated in Fusion 360 [33] to test displacement capability (Figure 16). This method allowed for faster iteration on the flexure and camshaft design, which could then be printed and integrated into the system for further testing. To increase the displacement, decreasing the height and/or widening the top of the flexure proved to be effective. Increasing

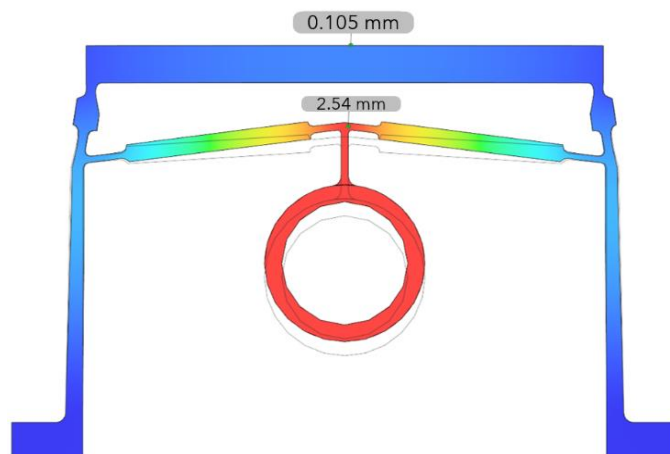


Figure 16 – Simulated displacement of flexure

the camshaft offset also increased the displacement.

A desirable displacement of 0.105 mm was achieved by widening the top of the flexure by about 2 mm and shortening the height by about 8 mm. The entire design was then scaled up by a factor of 1.375 mm. The simulation for this flexure can be seen in Figure 16, using a camshaft offset of 5 mm.

The camshaft was scaled up to have a diameter of 8 mm (previously 5 mm), with different offsets available. By changing the offset, the displacement also changes. As mentioned above, an offset of 5 mm provided 0.105 mm of

displacement. An offset of 3 mm had a displacement of 0.629 mm. These camshafts are very easy to design, 3D print, and swap out in the linear actuator, allowing for customizability by the user.

The gears from the original design were replaced by metal spur gears. These gears are available as a 64-tooth output gear, and options of 17-tooth, 21-tooth, 26-tooth, and 29-tooth input gears. The various configurations and possible output frequencies can be seen in Table 6. Gears are connected to the flexure using 3 mm and a 5 mm protrusion on the camshaft (Figure 17). Note that different gear configurations require different designs for the flexure frame due to the different sizes of the gears.

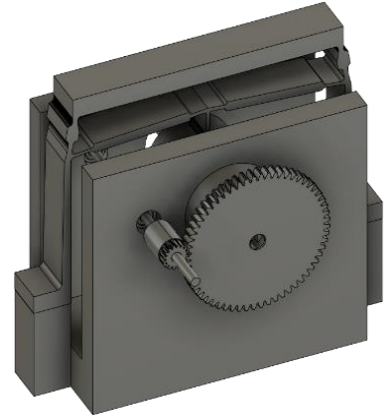


Figure 17 – Flexure with gears

Table 6 – Different gear configurations and their output frequencies

Gear Ratio	Input Rotational Speed (RPM)	Output Frequency (Hz)
17/64	1500	6.6
	3500	15.5
21/64	1500	8.2
	3500	19.1
26/64	1500	10.2
	3500	23.7
29/64	1500	11.3
	3500	26.4

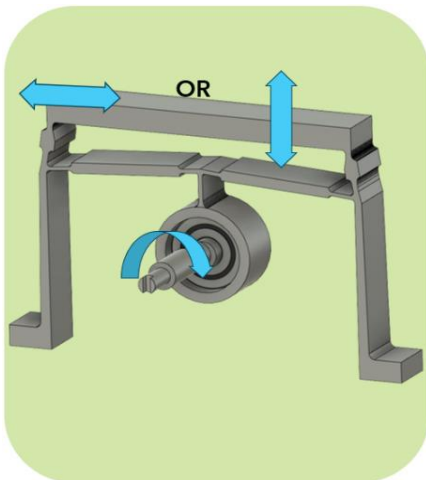


Figure 18 – Range of motion of flexure

During testing of the actuator, it was noticed that the flexure had up-and-down and side-to-side motion (Figure 18). Initially, the up-and-down motion was isolated by designing a coupling plate (Figure 19a) that could be attached to the bottom of the table surface so the flexure could not move side-to-side. However, when testing the actuator flipped on its side, to use the side-to-side motion instead, it was found that the displacement was different. A coupling bracket was designed to allow the flexure to be mounted in this fashion (Figure 19b). These discoveries show that the actuator can be used in both configurations to provide two different

displacements. This makes the linear actuator more customizable, as any change results in two more options for experimentation.



Figure 19 – Flexures with couplers for (a) under-mounting and (b) side-mounting, where arrows indicate the direction of vibration.

When a 5 mm offset is used, the displacement of the actuator is reduced to 0.04 mm for the side-mount configuration, and 0.02 mm for the under-mount configuration. This is lower than the target vibration, however it is still relevant to the research of lower displacement vibrations. Further, these values are from current tests prior to integration with the motor and implementation of a more secure frame, as the motor feedback system was not completed in time for this report. Higher displacement values are expected once the flexure frame is better secured to the table, due to some force being placed on the frame in its current form.

5.3.2 Table Design

As seen in Figure 20, the surface (1) on which the eggs are placed is fastened to a linear bearing pillow block (PIL) (2) on each side with custom 3D printed brackets (3). The PIL slides along a 12 mm metal shaft with minimal friction, isolating any motion to one axis (up and down). The metal shafts are clamped to wood (4), which is fastened to a sturdy base that can be placed on a table. To reduce strain on the flexure, shock absorbing springs (5) are used to sandwich the PIL. This significantly reduces the force needed to push the table upward, overall reducing additional strain on the flexure.

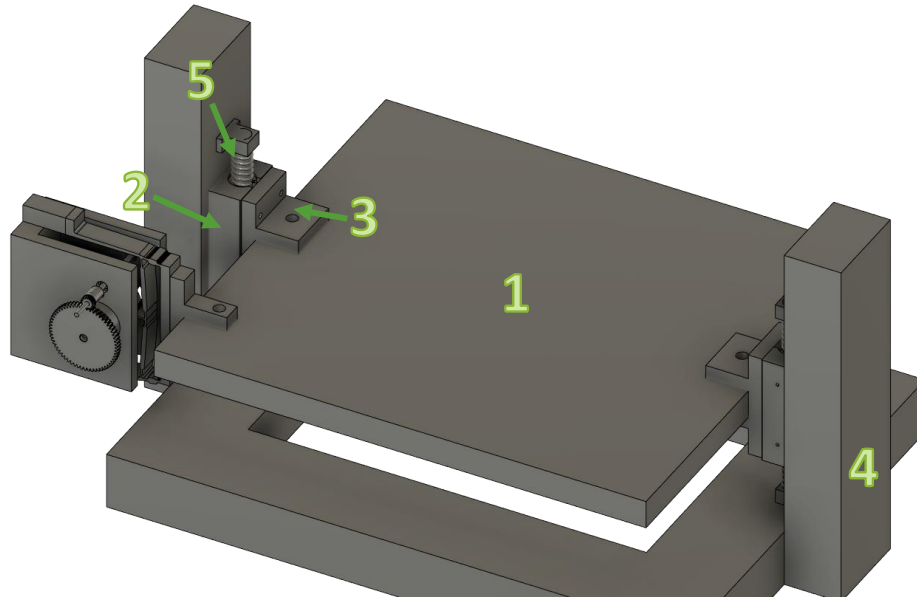


Figure 20 – 3D model of table design.

Figure 20 shows the linear actuator in a side-mounted configuration, where it is coupled to a custom 3D printed bracket. Further 3D modelling is needed to shape the frame of the actuator to attach securely to the table frame.

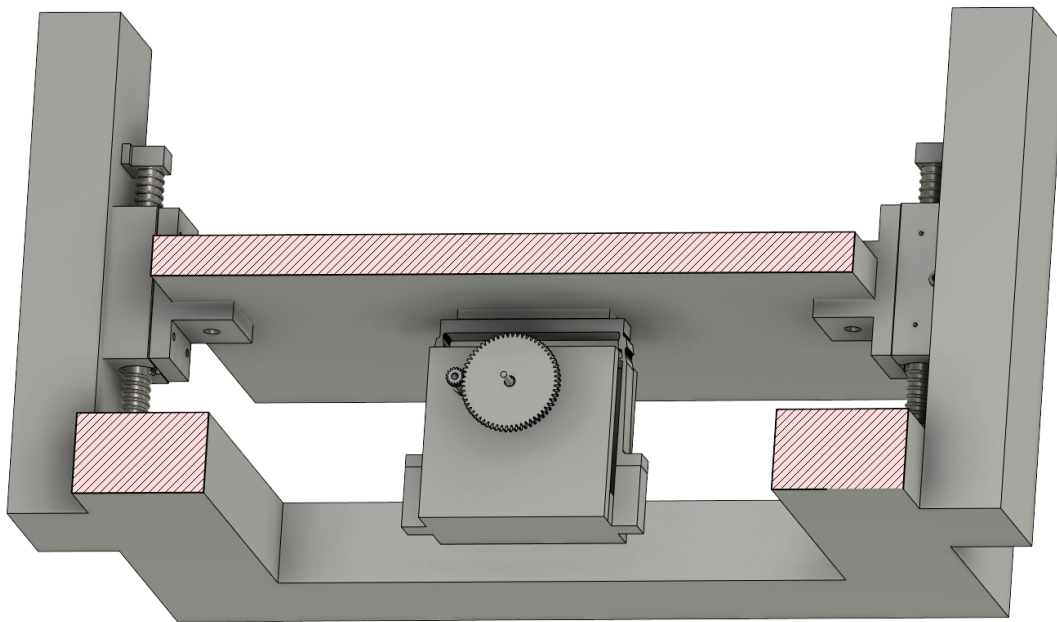


Figure 21 – 3D model of table design, actuator mounted underneath.

Figure 21 shows the option of mounting the linear actuator underneath the table, using a custom 3D printed coupling plate.

The wood base is constructed by hand using a 2x3x8 pine board, as can be seen in Figure 22. Leveling feet can be added as needed. The wooden design of the table is incredibly sturdy and inexpensive, however it requires appropriate tools and a person knowledgeable in wood construction to accurately build. A consideration for the future is to design the frame so that it can be 3D printed and snapped together, making it accessible and affordable to anyone with access to a 3D printer.



Figure 22 – Actual build of table, with actuator in side-mounted configuration.

5.3.3 Motor Feedback System

Using the I²C communication protocol, the speed of a motor can be precisely controlled. The user can set the desired motor RPM or vibration frequency through the User Interface (UI), and this information is then sent to the Raspberry Pi microcontroller. Upon receiving the command, the microcontroller generates a Pulse Width Modulation (PWM) signal that adjusts the motor's speed accordingly.

To maintain a consistent speed, the system utilizes readings from an RPM sensor to regulate the PWM signal. This feedback loop allows the system to adjust the voltage supplied to the motor based on the actual measured speed. As a result, the system can effectively work with a wide range of motors since it adapts the voltage to maintain the desired speed. This approach also guarantees that any damage or faults in the motor will not impact its speed since the system will automatically adjust the voltage to maintain the commanded speed.

By utilizing the I²C protocol and integrating an RPM sensor into the system, precise and reliable motor speed control is achieved. The system can be easily customized to suit various applications and can ensure optimal motor performance with minimal risk of damage.

Figure 23 presents a graphical representation of the feedback system, illustrating the relationship between the various components of the system. It provides a visual depiction of the flow of information within the system, highlighting the key components and their respective roles in ensuring the system's optimal performance.

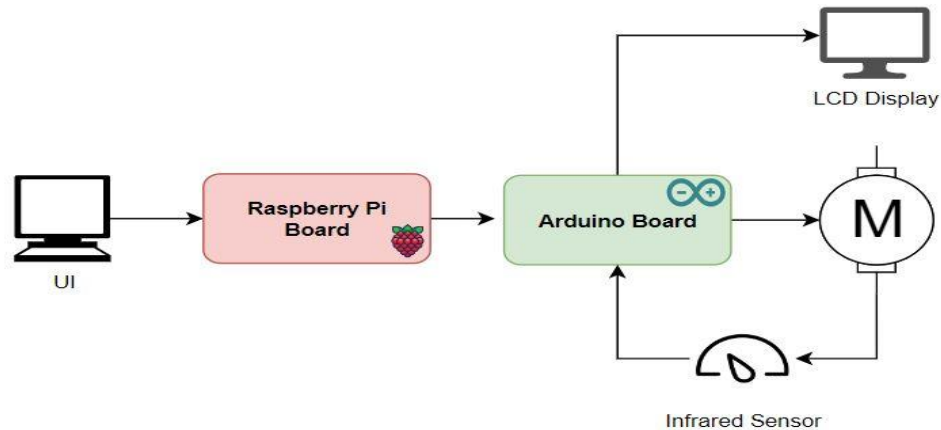


Figure 23 – Motor feedback system diagram

5.4 Database Design

5.4.1 Designing the Database Schema

As mentioned in Section 4.4, one of the three primary components of a database is a database schema. After choosing the database management system, and the query language, the next important task was to create a database schema. To recap, a database schema is a blueprint representation of a database that defines the organization and logical structure of the data [34]. It specifies the tables, attributes, and relationships that are necessary to store and manage data in a database.

Initially, the purpose and scope of the database were identified, along with the tables and fields that needed to be included. After initial meetings with the Davy lab team, a list of values they were interested in tracking was formed, with which the first iteration of the schema was created. Over the development period, the schema went through various iterations (included in the appendix for reference) and was finally updated to look like the one presented in Table 7. As seen from the figure, each field has a ‘type’, which represents what kind of an entry it was. Having multiple options such as Integer, Text, Date, Date-Time, and Float, every field was assigned a logical and appropriate field type.

Field	Type
id	INTEGER
species_name	TEXT
species_id	INTEGER
egg_count	INTEGER
site_location	TEXT
collection_date	DATE
email	TEXT
start_time	DATETIME
frequency	FLOAT
displacement	FLOAT
temperature	FLOAT
humidity	FLOAT
timestamp	DATETIME

Table 7 – Database Schema

Talking about the column themselves, each experiment is assigned a unique ID number, which corresponds to a specific entry in the database table. In addition to the ID number, each experiment is associated with a ‘species name’ and ‘ID’, providing information on the type of species under investigation. The ‘egg count’ field tracks the number of eggs utilized in the experiment. Information on the ‘location’ and ‘date of egg collection’ is stored in their dedicated columns. An ‘email’ address column is also included for contact purposes, which is set during the creation of a new experiment using the User Interface. Moreover, the time will be noted at the point of creating a new experiment using the User Interface and stored in the ‘start time’ column. These columns are set once and will not be updated during the experiment.

The subsequent columns in the database are dedicated to monitoring sensor readings. These readings are taken at 30-minute intervals, which means that the ‘timestamp’ column is updated accordingly. The vibration frequency, ‘displacement’, ‘temperature’, and ‘humidity’ sensors each have their own dedicated column for data collection. These values are critical for the laboratory's research, as they provide crucial information on the conditions under which the eggs will be kept.

5.4.2 Database Implementation on the Raspberry Pi

After finalizing the design of the database schema, the next step was to implement it on the Raspberry Pi using SQLite. The decision was made to interact with the database programmatically using Python due to its seamless integration with the User Interface. Both the database and User Interface were developed using Python as their core programming language. By interacting with the database programmatically through Python, data manipulation was streamlined, and data retrieval was efficient, leading to enhanced functionality and performance [35]. The SQLite3

module was installed on the Raspberry Pi, as it was the appropriate library/driver to interact with the SQLite database [36]. With the installation, data could be read, updated, and deleted from the database using Python code, making it as simple as using SQL language commands.

The work began by creating a database file on the device using the 'CREATE DATABASE' SQL command through Python [37]. This enabled a functional database file that could be utilized for further SQL commands. The database file was stored locally on the Raspberry Pi as a .db (database) file.

Next, a Python file was constructed containing many functions, each of which played an integral role in populating the database. The first function was programmed to utilize the 'CREATE TABLE' SQL command, which established within the database file, a new table with columns according to the attributes of the designed database schema. In addition, numerous more functions were developed within the same Python file to retrieve real-time sensor information from the sensors themselves. At set intervals, relevant data were acquired and stored in the database, by making use of the 'INSERT' SQL command, which enabled the real-time insertion of modified values into the database (once every 30 minutes) [37]. Lastly, a function was developed that employed the SQL 'DELETE' command to delete an existing database table [37]. This allowed redundant data to be eliminated from the database when the trial concluded, ensuring effective data management procedures.

In the event that the experiment was terminated (using the User Interface), the .db file was automatically converted to a .xlsx (excel) file. This file was then attached to an email and automatically sent to the email address associated with the experiment. A temporary Gmail account associated with the lab was set up to facilitate the file transfer. The excel file format of the received file made it easy for the user to view their experiment data once the experiment had concluded, resembling the screenshot shown in Figure 24 below. Overall, this approach allowed for a convenient and streamlined process for transferring experiment data to the user (as seen in the screenshot in Figure 25). All the described process, including the automatic conversion of .db file to .xlsx, attachment to an email, and transfer to the user's email address, was made possible through the creation of a third Python file that incorporated the necessary functions utilized in this process.

ID	Species	Species ID	Eggs	Site Location	Collection Date	Start Time	Frequency	Displacement	Temperature	Humidity	Email	Time Stamp
1	snapback	3454	9	Medeola Woods	2023-06-23	2023-06-24 13:49	123.456	123.459	22.3	35	dlab@gmail.com	24-06-2023 13:49
2	snapback	3454	9	Medeola Woods	2023-06-23	2023-06-24 13:49	123.456	123.459	22.3	35	dlab@gmail.com	24-06-2023 14:19
3	snapback	3454	9	Medeola Woods	2023-06-23	2023-06-24 13:49	123.456	123.459	22.4	36	dlab@gmail.com	24-06-2023 14:49
4	snapback	3454	9	Medeola Woods	2023-06-23	2023-06-24 13:49	123.457	123.459	22.4	36	dlab@gmail.com	24-06-2023 15:19
5	snapback	3454	9	Medeola Woods	2023-06-23	2023-06-24 13:49	123.457	123.459	22.3	35	dlab@gmail.com	24-06-2023 15:49

Figure 24 – Database file as an excel spreadsheet.

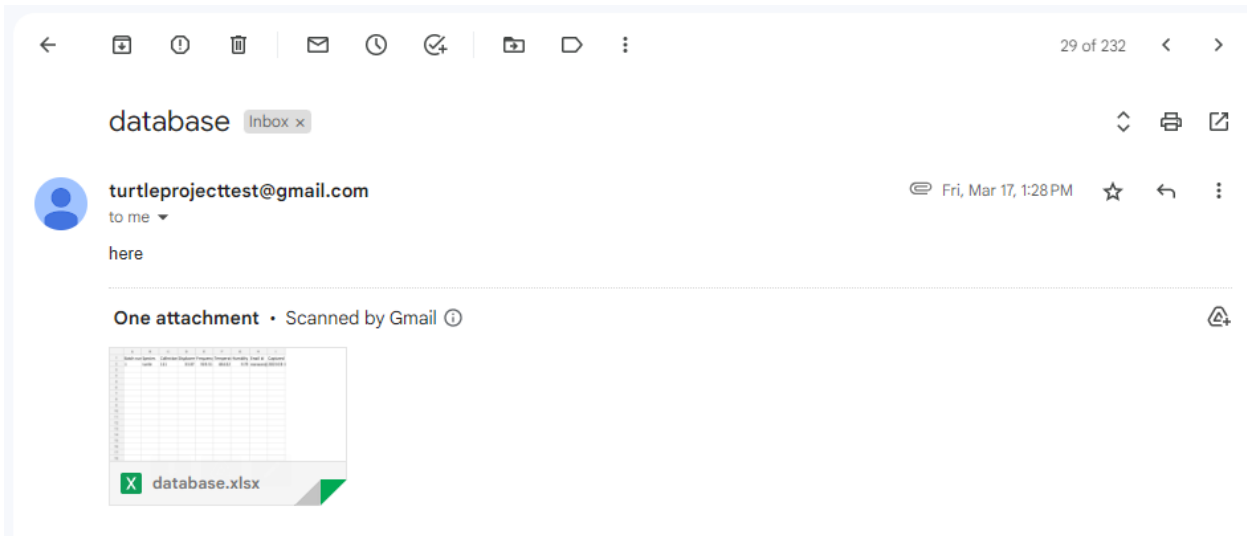


Figure 25 – Received email containing attached database file as excel spreadsheet.

During the development process for the database, a couple of obstacles were encountered. The initial approach was to automatically send emails with attachments using Google's SMTP server, which is a widely used method for sending emails from a Gmail account to other email clients using a Python script. Typically, this server only requires a username and password for authentication. However, since May 2022, two-factor authentication has been required on all Google accounts, which involves linking a functional cell phone number to the account [38]. A workaround that didn't require a phone number for privacy reasons was attempted but proved unsuccessful within the time constraints of the semester. Consequently, a cell phone number was used to enable the email-sending process.

Additionally, as mentioned earlier in Section 4.4, initial attempt to implement the database involved using MySQL. However, we discovered that its implementation on the Raspberry Pi used a fork of MariaDB, which added complexity to the task and would have likely delayed development [27]. Furthermore, a lack of documentation available to guide us through the implementation process would have made it difficult to troubleshoot issues as they arose. Therefore, an ultimate decision to use SQLite was made. While this solution may not have had all the functionality that MySQL could have offered, its straightforward implementation process was the best course of action given the time constraints of the deliverables over the semester. A more elaborate explanation of the benefits we traded is listed later in section 6.3 of this report.

Testing

Throughout the development process, several aspects were tested thoroughly and consistently throughout the various iterations to ensure the functionality and performance of the database. Firstly, the creation of the database file using the 'CREATE DATABASE' SQL command was used to verify that a functional database file was successfully created and stored locally on the RPi.

Similarly, the function that utilized the 'CREATE TABLE' SQL command was tested to ensure that the tables created were of the correct column lengths and fields. Additionally, functions that used the 'INSERT' SQL command were tested to determine if real-time sensor information was accurately acquired from the sensors and stored in the database at set intervals. The function that employed the 'DELETE' SQL command was tested to see if an existing database table was eliminated from the database file. Every aspect of sending the database through email was tested, including the file conversion and email attachments. Finally, the overall functionality and performance of the database in combination with the user interface was tested by conducting multiple experiments to check for any potential errors or issues that may arise during the day-to-day operation of the device.

During the testing phase, a major bug was discovered that resulted in corrupted excel files being attached to email, rendering the experiment data unreadable for the user. Further testing revealed that the bug was caused by an incorrect encoding format used during the file conversion process. This was fixed by updating the encoding format to ensure proper conversion and attachment of the .xlsx file to the email. This fix, in addition to some minor bugs highlighted the significance of keeping testing as a high priority throughout the development process.

5.5 User Interface

The User Interface (UI) provides a way that a user can interact with our device; therefore, it must be designed for efficiency and simplicity. Our design philosophy is to create an interface that is simple and highly functional which always allows for full control over the experiment and its parameters.

The framework and language to be used in the building of this UI is PyQt5 and Python respectively. A great feature of PyQt5 is its Qt designer which allowed for real-time designing of the UI on a 2D plane to then add functionality using code. This streamlined the process of building the UI greatly and allowed for designs that were not as easily achievable using a pure code approach.

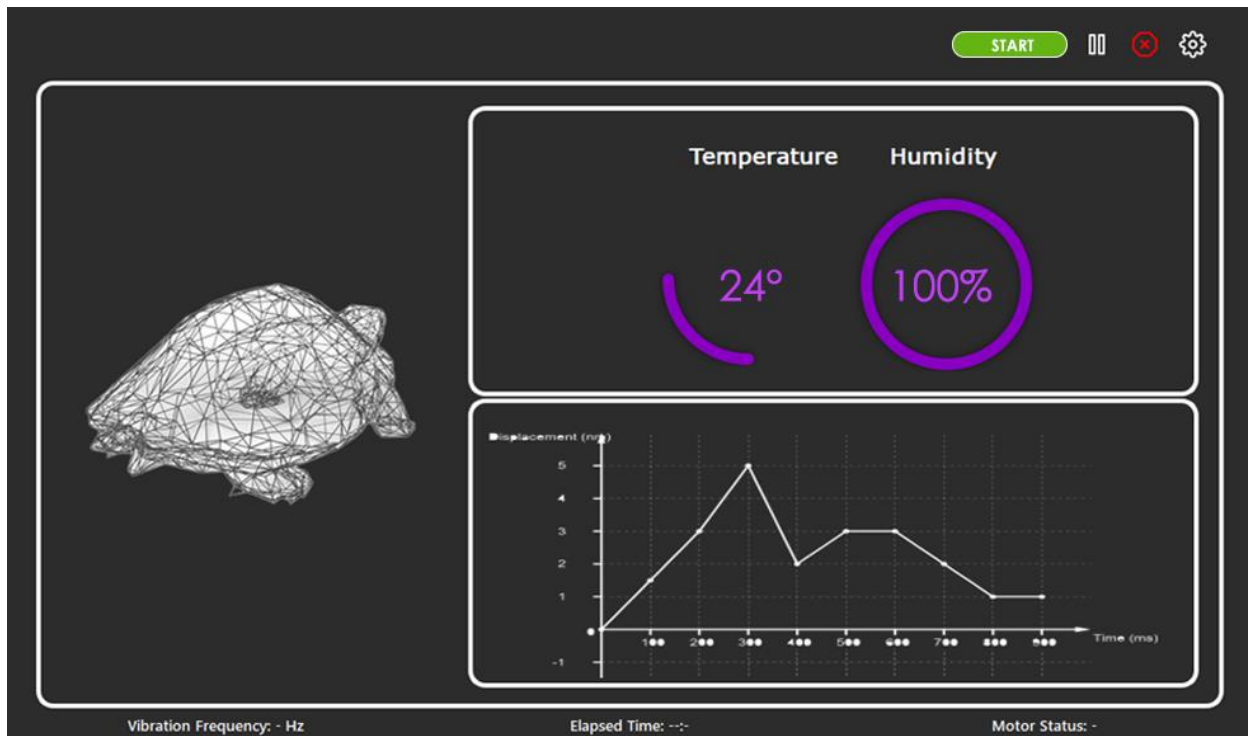


Figure 26 – Home screen UI

The home screen is the main page of the system, it houses important experiment information such as the vibrational frequency, elapsed time, sensor information, and motor status. These data are always displayed for the researcher in a clear manner to allow for quick and easy monitoring of the most important parts of the system. A graphical view of the displacement graph is also provided in the interface as a quality-of-life improvement.

As for design elements, the use of proportions and spacing really helps to design the page in a way that is understandable by the user. In addition, the use of different and strong colors to imply a specific function was used to further enforce certain elements' functions, for example the use of green for the start button and red for the stop button. Both these colors are usually associated with these functions which makes the user instantly make the correlation and thus make it easier for the UI to be understood by the user.

Using the home screen, you can effectively control the experiment status. The controls are clear and grouped together at the top right of the page, present and accessible at all times by the user. A start button to initiate the experiment, a pause/play button to temporarily pause the experiment, and a stop button to completely halt the experiment. Lastly there is a settings button signified by the cog icon that navigates the user to the settings page.



The settings page features a dark background with a white border. At the top left is a back arrow icon. Below it is a gear icon for settings. The page is organized into two main columns of input fields. The left column contains: 'Experiment Name' with a person icon, 'Desired Frequency' with a frequency symbol 'f', 'Desired Amplitude' with a waveform icon, and 'Email Addresses' with an envelope icon and a plus button at the bottom. The right column contains: 'Species Name' with a DNA helix icon, 'Site Location' with a location pin icon, and 'Number of eggs in clutch' with an egg icon. At the top right, there is a 'Temperature Unit' section with 'C' and 'F' buttons and a circular slider. At the bottom right, there are two buttons: 'Export Data' with a document and arrow icon, and 'Save Changes' with a floppy disk icon.

Figure 27 – Settings Page UI

Once the user has navigated to the settings page, they are presented with editable fields to change the parameters of the experiment or the units of your information. These parameters range from important experiment related settings like desired amplitude and frequency to supplementary info like site location or species name to be added in the logging of the data in the database. Researchers can input their email addresses so exported data can be sent as a spreadsheet file directly to them, or the data can instead be saved locally on the user machine.

A big part of building the User Interface was testing. At each step of implementation testing was done to assure that all components of the system still function as they should and to find any hidden issues that might cause fatal errors in the system. One issue that was faced earlier in the project and was causing major problems had to do with loops, due to the use of any loops in the system without the proper preparations would crash the application in the middle of the experiment. This issue was found early on thanks to vigorous testing and was solved by switching to timer-based loops, fixing the crashing problem, and enhancing performance and timing precision. Not all issues found by testing were technical, some were related to design. It was found that the original start button (a white play button) was too ambiguous and did not clearly specify its function also users seemed to sometimes confuse it with the pause/play button understandably. Based on the findings the start button was changed into a clearer green button with letters that show it is the start button.

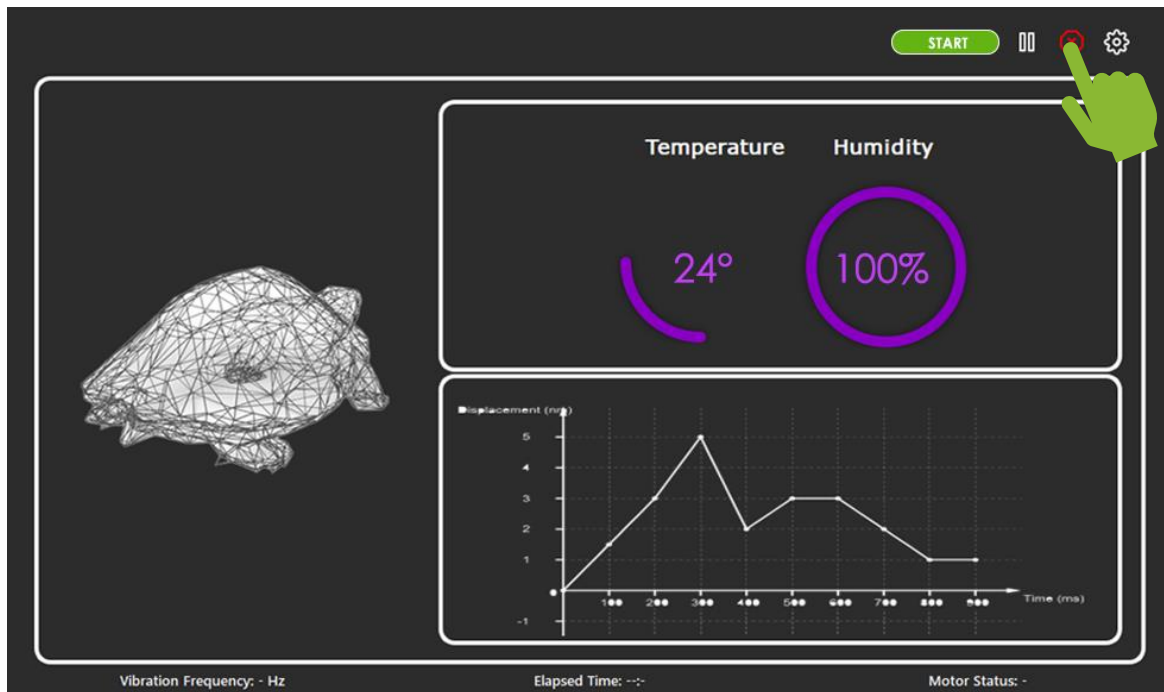
In conclusion, a well-designed User Interface is essential for any device or system, especially in the case of an experiment control system. The use of PyQt5 and Qt designer allowed for the creation of a simple yet highly functional interface that enables full control over the experiment and its parameters. The home screen provides all the necessary information to the researcher in a clear and concise manner, while the settings page allows for the customization of experiment parameters and data logging. Vigorous testing was done throughout the development process to ensure that all components of the system functioned as intended. Technical and design issues were identified and resolved. Overall, a well-designed User Interface can enhance the user experience and make the operation of the system more intuitive and efficient.

Normal User Flow

The typical user flow in this User Interface is simple and easy to understand, here is how it goes:

Firstly, the experiment settings need to be set up before the experiment can be started.

1 – The user presses the settings icon.



2 – Fill the fields with the respective information

The screenshot shows a dark-themed form for the Ground Vibration Simulator. The form contains several input fields and buttons. A green hand cursor is pointing to the 'Species Name' field. The fields are arranged in two columns. The right column has a 'Temperature Unit' section with a slider between 'C' and 'F'. At the bottom right are 'Export Data' and 'Save Changes' buttons.

Field Label	Field Type	Value
Experiment Name	Text	
Species Name	Text	
Desired Frequency	Text	
Site Location	Text	
Desired Amplitude	Text	
Number of eggs in clutch	Text	
Email Addresses	Text	

Buttons: Export Data, Save Changes

3 – User presses on “Save Changes” button

The screenshot shows the same form as before, but now with a green hand cursor pressing the 'Save Changes' button. An alert dialog box is displayed in the center of the screen, indicating that the changes have been saved successfully. The fields are now populated with sample data.

Field Label	Field Type	Value
Experiment Name	Text	Experiment 1
Species Name	Text	Snapback
Desired Frequency	Text	5
Site Location	Text	Medford
Desired Amplitude	Text	0.2
Number of eggs in clutch	Text	6
Email Addresses	Text	marwanzeyada@gmail.com

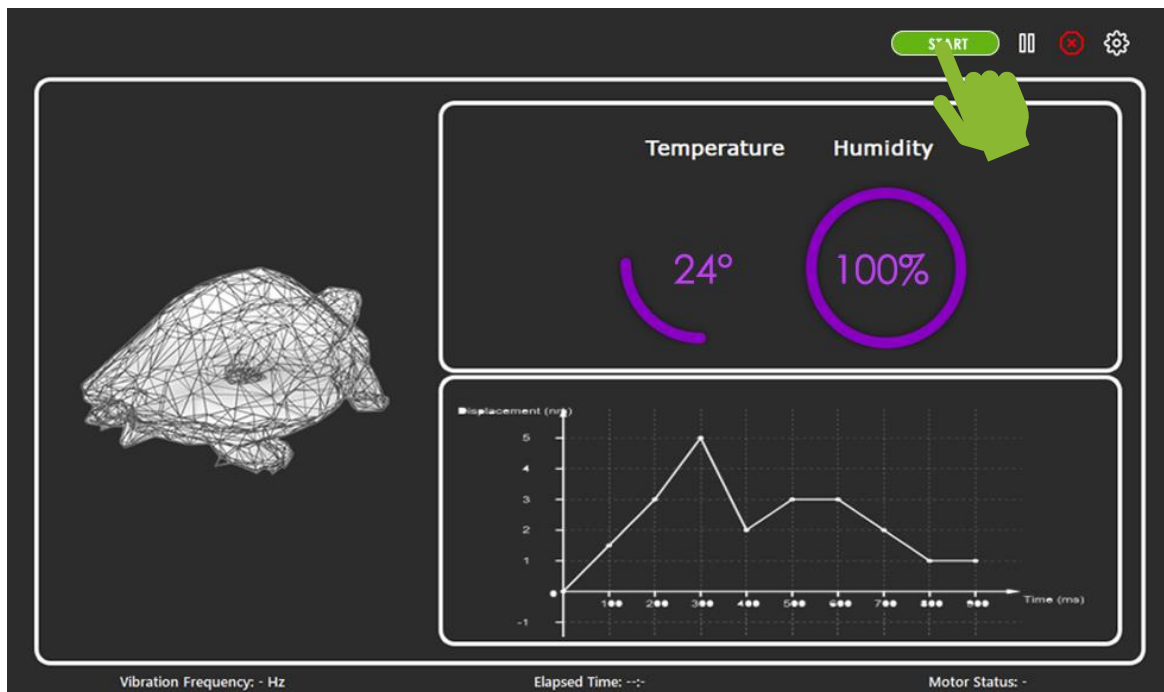
Buttons: Export Data, Save Changes

Alert: Changes Saved! OK

4 – Navigate back to home screen

The settings screen features a dark background with white text and input fields. A green hand icon points to a back arrow in the top left corner. The form includes fields for Experiment Name (Experiment 1), Species Name (Snapping Turtle), Desired Frequency (5), Site Location (Medecia Woods), Desired Amplitude (0.2), Number of eggs in clutch (5), and Email Addresses (marwanzeyada@gmail.com). There are also buttons for Export Data and Save Changes, and a Temperature Unit selector (C/F).

Field	Value
Experiment Name	Experiment 1
Species Name	Snapping Turtle
Desired Frequency	5
Site Location	Medecia Woods
Desired Amplitude	0.2
Number of eggs in clutch	5
Email Addresses	marwanzeyada@gmail.com

5 – Press the green “START” button

Other use cases are documented in section 3.3, as well as alternate flows.

6 System Evaluation

6.1 Computer and Microcontroller Devices

For the main computer, a Raspberry Pi 4 Model B was selected due to its low cost, ease of use, and compact size. The computing power of a Raspberry Pi is sufficient for this project, as it runs a simple User Interface and local database. The Raspberry Pi is capable of wired or wireless internet connection, which is needed to send the collected experimental data to users' emails. For ease of use, the Raspberry Pi has easily interfaceable pins that were used for the I²C connection to the microcontroller. It also has significant documentation so that users with little to no coding experience can easily pick one up and install necessary project files.

Two options for the microcontroller were considered and compared for cost, ease of use, and I²C compatibility. The microcontroller would also have to be beginner-friendly, as the expected user is a biology researcher with little to no coding experience. The user will have to program microcontrollers for additional tables, so beginner-friendliness was a deciding factor.

The Texas Instruments MSP430 LaunchPad microcontroller was initially used as it was a donation from Jelena Nikolic-Popovic, a Senior Member of Technical Staff at Texas Instruments Canada. Additional LaunchPads are inexpensive at about \$25 per unit. The LaunchPad is capable of I²C communication and was successfully programmed to send a PWM signal to a motor driver. Due to the low-level C programming language used, and that the User Interfaces directly with registers, the LaunchPad is extremely fast and efficient. However, this low-level programming language and concept of interfacing with registers is extremely difficult for beginner coders to work with. This was deemed not user friendly enough for the expected user.

An Arduino Uno was ultimately selected, as it is capable of I²C communication and has extensive documentation and libraries that are easy to pick up for beginner coders. It also has libraries for specific sensors used in this project. It is also inexpensive at about \$25 per unit. The Arduino uses a higher-level version of C++ designed specifically for Arduino hardware, which is not as efficient as the LaunchPad's low-level C and register interfacing. However, experienced coders can interface with the Arduino's registers if they desire. While it is less efficient than the LaunchPad, it is equal in most other aspects and is significantly more beginner friendly.

6.2 Sensors

6.2.1 Linear Displacement Sensor

Many sensor options were investigated for precisely measuring the linear displacement. The biggest challenge was to find a sensor that could accurately measure 0.1 – 1 mm of displacement and be tracked in data collection.

Sensors investigated were linear potentiometers (LinPot), linear variable differential transformers (LVDT), capacitive displacement sensors, laser displacement sensors, and digital indicators.

LinPots and LVDTs were rejected due to how many full cycles they could measure before losing accuracy. Both had a limit up to 10 million cycles, which would only cover one quarter of the 95 days duration of an experiment at 5 Hz, less for higher frequencies.

Capacitive displacement sensors and laser displacement sensors were too expensive for sensors sensitive enough to measure 0.1 mm of displacement, with many sensors and data processing systems for the sensors costing more than \$1000.

While the mentioned sensors could measure linear displacement throughout the duration of the experiment, they are either not durable enough or too expensive to be used. Digital indicators were examined as a low-cost option, as only one sensor would be needed to measure displacement of all tables. However, it comes with the trade-off of not measuring 100% of the time. This was deemed acceptable by the Davy Lab and the project team, as measuring linear displacement was important primarily to ensure that all components continue to work as expected and does not change throughout the duration of the experiment.

6.2.2 Temperature and Humidity Sensor

Temperature and humidity sensors are low-cost sensors that are widely available and easy to use. Two sensors were considered as they have well-documented libraries available for the Arduino. DHT11 and DHT22 sensors are capable of both temperature and humidity measurement. The DHT11 is extremely inexpensive at about \$2 per unit but is known for not being reliably accurate. The DHT22 has a slightly higher cost at about \$12 per unit and is very accurate for the purpose of maintaining the correct environment for reptile egg incubation. Further, only one temperature and humidity sensor is required for the entire system. The DHT22 was selected, as the higher cost was determined to be acceptable as a trade-off for the accuracy provided.

6.2.3 RPM Sensor

The selection of a suitable sensor for measuring the motor's RPM involved careful evaluation of multiple factors to ensure optimal performance and cost-effectiveness. The team prioritized achieving the highest level of accuracy while also considering the overall cost of the system. Several options were considered, and a rigorous selection process was undertaken to determine the best choice for the project.

Initially, the team considered using a tachometer to measure the motor's RPM. However, after further evaluation, it became apparent that the shaft of the motor was too small to fit in the

tachometer chamber. As a result, alternative options were explored to identify a more suitable sensor.

After careful consideration, the team decided to use an infrared (IR) module to calculate the motor's RPM. The IR module was found to be highly accurate and reliable, while also being available at a relatively low cost. The price tag for a set of five units was \$11, which made it an affordable and practical choice.

The integration of the IR module was found to be simple and straightforward, which meant that it could be easily replicated for future builds, if needed. Furthermore, the IR module provided a non-intrusive way of measuring the motor's RPM, which reduced the risk of any damage to the motor or the system.

In conclusion, by considering multiple factors and evaluating various options, the team was able to select an appropriate sensor for measuring the motor's RPM. The IR module proved to be a cost-effective, accurate, and reliable choice, which ensured the system's optimal performance while also being easy to replicate for future builds.

Vibration Simulator

6.2.4 Flexure-guided Linear Actuator

As was discussed in depth in sections 4.3.1 and 5.2.1, most methods of creating the linear displacement needed for vibration were deemed too costly or not accurate enough. For example, using a traditional lead screw linear actuator has too much backlash when the direction is changed. Similarly, using a piezo flexure-guided linear actuator would not be accurate due to the hysteresis when changing directions. In both cases, since direction would need to be changed 5-20 times per second, the displacement would not be predictable.

Existing industrial shake tables were considered but begin at \$5000 for a single shake table. Therefore, it was determined that the team would develop a new, cost-effective shake table design.

The JWST fine positioning stage linear actuator was ultimately used for its low cost. Since the design was available for free as a 3D model, it could be iterated upon for no cost, and printed for no cost at Carleton's MacOdrum Library. The design eliminated backlash that makes other linear actuator designs unfavourable.

6.2.5 Motor Feedback System

The backbone of the feedback system is comprised of a micro-controller, which plays a pivotal role in ensuring the system's efficient operation. After careful consideration, the team decided to work with the Arduino Uno micro-controller, primarily because of its user-friendliness and versatility in integrating various sensors. The Arduino Uno has multiple programmable pins,

which can control the motor's speed using pulse width modulation, making it an ideal choice for this project.

To facilitate communication between the User Interface (UI) and the Arduino Uno, the team incorporated a Raspberry Pi micro-controller. The Raspberry Pi serves as a bridge between the UI and the Arduino Uno, enabling the transmission of commands from the UI to the Arduino Uno to control the motor's speed. This approach allows for seamless integration between the various components of the system, ensuring smooth and efficient operation.

The Arduino Uno's compatibility with multiple sensors and ease of integration, coupled with the Raspberry Pi's ability to facilitate communication between the UI and the Arduino Uno, results in a highly efficient and effective feedback system. The system's components work in harmony to achieve optimal motor performance, making it an ideal solution for a range of applications.

The team's selection of a normal DC motor for the project was based on a careful evaluation of various motor types. As a crucial consideration in maintaining an affordable pricing model, the team determined that a normal DC motor was both easily attainable and relatively inexpensive, costing only \$3 each. However, it was recognized that these motors may produce noise and small vibrations when running at lower speeds, which could potentially impact the precision of the experiment. In response to this challenge, the team designed and implemented a gear reduction system to connect the motor to the linear actuators, which effectively mitigated the issue. The team also explored the possibility of utilizing stepper motors, but ultimately determined that they were too slow to meet the desired frequencies required for the project.

To enable the motor to run at different speeds using PWM signals, the team selected a L298N driver. This driver was found to be both cost-effective and user-friendly, enabling seamless integration with the Arduino platform. Furthermore, the L298N driver has the capability to power two motors simultaneously, which not only reduces costs associated with building an additional table, but also enhances the overall efficiency of the system.

6.3 Database

Choosing the right database management system for a project required careful consideration of various factors, including performance, scalability, cost, ease of use, and availability of support and documentation. In the case of this project, the team had to weigh the pros and cons of using either a relational database management system (RDMS) or a non-relational database management system (NoSQL). While NoSQL databases like Firebase offered real-time updates and scalability, they also came with a higher cost and a steeper learning curve [39]. On the other hand, RDMS like SQLite and MySQL were more established and widely used, making them easier to work with and offered better support and documentation. However, they weren't as scalable as NoSQL databases and may require more maintenance and setup [40].

After evaluating several RDMS options, the team had to choose between SQLite and MySQL. While MySQL had more advanced features and better remote access capabilities, it proved to be more challenging to implement on the Raspberry Pi due to its forked version, MariaDB. On the other hand, SQLite's lightweight nature and serverless architecture made it an efficient and cost-effective solution for storing and retrieving data collected from various sensors [41]. However, SQLite may not be as suitable for larger datasets and complex operations as MySQL.

Another important trade-off we considered was the choice of query language. While SQL is a well-established and widely used language, allowing for efficient data management and manipulation, it may not be as flexible and scalable as other query languages used in NoSQL databases. Therefore, the team had to weigh the benefits of using a familiar and efficient language against the potential limitations of SQL.

Finally, the team had to consider the design of the database schema, which determines the organization and logical structure of the data. While a well-designed schema can ensure the consistency and correctness of the data, it may also require more upfront planning and may be less flexible to changes in the future. Therefore, the team had to balance the need for a well-structured and organized database schema with the potential drawbacks of a rigid and inflexible design.

6.4 User Interface

Kivy and PyQt5 are both popular frameworks used for developing User Interfaces in Python. Kivy is an open-source framework that provides cross-platform support for creating interactive applications with multi-touch support. PyQt5, on the other hand, is a Python binding for the Qt application framework, which is a powerful and mature framework that has been around for over 20 years.

One advantage of Kivy is its ease of use for creating applications with complex, multi-touch User Interfaces. It has a simple syntax and provides a wide range of widgets and tools for creating dynamic interfaces. Kivy is also open-source, which means it has a large community of developers who contribute to its development and provide support for users.

PyQt5, on the other hand, provides a mature and stable development environment with a powerful and flexible toolset for creating User Interfaces. One of the major advantages of PyQt5 is its Qt Designer, which allows for real-time designing of the UI on a 2D plane to then add functionality using code. This streamlines the process of building the UI and allows for designs that may not be easily achievable using a pure code approach. PyQt5 also offers seamless integration with other Python libraries and tools.

However, one disadvantage of Kivy is its limited documentation, which can make it difficult for users to get started with the framework. PyQt5, on the other hand, has extensive documentation

and tutorials, which makes it easier for users to learn and use the framework. Another potential disadvantage of PyQt5 is that it is not open-source and requires a license for commercial use. In addition, Kivy's default styling is very basic, and it can be difficult to create more complex designs without a deep understanding of the framework, but PyQt5 give you a lot of flexibility with styling even before you start writing any code.

Ultimately the team chose to go with PyQt5 for its advantage in having a designer which almost cut developing and implementation time in half. In addition, it allowed the team to design a beautiful UI that's completely functional and easily updatable.

7 Budget Breakdown

Note, hardware does not include 3D printed components, as Carleton University's MacOdrum Library provides commercial-grade 3D printing services to students and faculty for free. Further, costs presented are as of March 2023, and are subject to change.

The project budget was \$1000, provided by the Davy Lab. Additional funding was available as needed from the Faculty of Engineering and Design, but was not used. The breakdown in Table 8 reflects the items needed to build the final design of the shake table before taxes. Additional breakdowns of research and development costs and the cost per additional table are available in Appendix 3: Costs.

Table 8 – Hardware required to complete project, with sources and pricing.

Item	Source	Price
DC motor	BC-Robotics.com	\$1.95
Motor driver	BC-Robotics.com	\$9.99
Arduino	Amazon.ca	\$23.99
Differential Gears 64T, 17T, 21T, 26T, 29T	Amazon.ca	\$16.29
3mm Motor shaft	Amazon.ca	\$7.59
2mm to 3mm Shaft coupler	BCRobotics	\$3.95
Shaft collar 8mm	Amazon.ca	\$9.99
Linear shaft 12mm x 150 mm (x2)	Amazon.ca	\$14.99
Linear shaft 12 mm bearing PIL (x2)	Digikey.ca	\$18.74
Linear shaft 12mm clamp	Amazon.ca	\$17.49
Stiff springs (x4)	Amazon.ca	\$43.66
Melamine surface	Homedepot.ca	\$26.75
Knotty Pine Board 2x3x8	Homedepot.ca	\$16.98
5/16-18 Nylon Insert Nut (x3) (\$ 0.16 each)	Ottawa Fastener Supply	\$0.48
5/16 X 2 Tap Bolts (x3) (\$0.70 each)	Ottawa Fastener Supply	\$2.10
A2 M5x16 Phillips Head Screw (x8) (\$ 0.22 each)	Ottawa Fastener Supply	\$1.76
#12 1 ¼" Sharp Point screw (Pack of 8)	Ottawa Fastener Supply	\$3.79
Incubator	Provided by lab	\$0
Raspberry Pi 4, 4GB	Digikey.ca	\$80.95
Temperature & Humidity sensor	Amazon.ca	\$11.60
Mitutoyo 543-789 Digimatic Indicator	Amazon.ca	\$264.43
Mitutoyo SPC Connecting Cable	Amazon.ca	\$72.58
IR Sensor Module	Amazon.ca	\$10.89
Indicator Holder	Amazon.ca	\$30.00
	TOTAL	\$690.94

8 Reflections

Throughout the research and design phase of the project, the team has gained invaluable insights into the engineering process. Recognizing when an idea might not be feasible, salvaging work done, and producing a final product have been essential learning experiences. Three major challenges encountered were measuring displacement, frequency, and converting rotational motion to linear motion. The team tested and reverse-engineered multiple methods, experiencing failures along the way. These experiences have broadened the team's understanding of engineering a product and expanded their knowledge within the academic year.

Initially, the project aimed to simulate vibrations detected in the presence of wind turbines. However, during the research and design phase, the team determined that achieving the target displacement of about 300 nm would be challenging and difficult to measure. Following discussions with the Davy Lab, the team decided to focus on vibrations with the same frequency but larger displacement, including a range of industrial vibration sources.

To address frequency measurement challenges, two team members individually explored two different solutions: using an accelerometer or an IR sensor to measure RPM and process data to obtain frequency. Due to a comparably bigger offset for the system's use, the accelerometer was ruled ineffective. As a result, the team proceeded with the IR sensor. The member working on the accelerometer then collaborated with the other member to refine the code required for the IR sensor, improving data precision and collection efficiency. This experience demonstrated the importance of individual and collaborative work, accepting failure and success, and embracing the most effective method to enhance the design.

Reflecting upon the project, the collaborative experience has significantly contributed to the team members' understanding of their individual strengths and weaknesses. The flexible task allocation allowed each member to explore various aspects related to the project. Effective collaboration, coupled with thorough documentation, enabled every team member to participate in multiple tasks and contribute input based on their strengths, while the expert member efficiently finalized the tasks. This process facilitated the efficient finalization of tasks and allowed the team to learn valuable lessons about teamwork, personal identity, and engineering ethics.

Through this project, the team has gained a deeper appreciation for the engineering process, the various stages of product development, and the essential role of strong ethical skills for success in the engineering field. These invaluable experiences and insights will undoubtedly benefit the team members as they continue their engineering careers and face future challenges.

8.1 Limitations

The design of the linear actuator presents a few challenges that need to be addressed. One of these is the fixed linear displacement, which requires more swappable designs for the flexure. However, the actuator does offer some variety in its different camshafts and two different configurations, which can help with this issue. Another challenge is that the linear displacement measurement is not continuous, which means that any issues may only be detected during the lab's measurement protocol every other week. Finally, the frame of the actuator is only designed for a 64/21 tooth ratio, which may require more designs to be made so that gears can be swapped out. However, this is a relatively easy fix. Overall, the linear actuator design has some limitations, but with some modifications, it can still be a useful component in a larger project.

It was observed that there were a couple of limitations in the database management process albeit being minor. Firstly, the naming convention used for the database tables was static. This could potentially lead to confusion when trying to identify specific data points, especially if there were multiple tables in use. Although this is not considered a critical issue, implementing a dynamic naming convention could make the database more user-friendly and easier to navigate.

The database can only be exported via email or locally at the end of the experiment. While this is a common practice, it can result in delays in data processing and analysis, especially if there were any errors or inconsistencies in the data. A better approach would be to give the system the ability to export the database whenever it's needed without stopping the experiment.

The current user interface (UI) for controlling experiments has two limitations that impact its usability. Firstly, it lacks the ability to control multiple different experiments simultaneously. This means that researchers or experimenters who need to manage several experiments will have to switch between different UIs or control panels. Secondly, while the UI currently performs well, there is still room for improvement in terms of its overall performance.

8.2 Future Work and Improvements

Future Work

The integration of the UI to control the motor speed is under works. This will make it easier for labs to obtain the desired frequencies for their experiments. Additionally, the printing of the base for the motor and the installation of the motor to the linear actuator are prioritized tasks, that still need to be completed. A pressure sensor needs to be integrated to collect additional data for the labs research. The current driver utilized has the capability to operate two motors concurrently. However, due to the project requirements, only one motor was utilized at the moment. In the event that a new table is constructed in the future, it is possible to seamlessly integrate an additional DC motor into the system by connecting it to output 3-4 of the driver. Another feature that needs to be worked on and will be integrated before handing over the work to the Davy lab is to enable the user to access the updated database at any time during the

experiment. This will be accomplished by including a new UI button that, when pressed, sends the user an email with the current database file attached that they could access whenever convenient. Even though the user can access the running database using 'SQL-browser' for Raspberry Pi, this feature would make it easier for them to examine the most recent data and provide them additional insight into how the experiment is going. If the experiment ends up not showing any effect on the turtle eggs because the displacement is very small, we will print different linear actuators that have a higher displacement for different with higher vibration magnitude such as next a wind turbine farm or dams.

It is important to note that the team is currently in the process of integrating a motor feedback system to enable the motor to adjust speed based on the input received from the user interface. The team recognizes that this critical enhancement is necessary to ensure the system operates with optimal precision and stability, and is thus prioritizing the integration of this system in the near future.

Currently, the system is operational but has demonstrated some inconsistencies in its speed stability, which the team attributes to a lack of a robust feedback system. However, the team is confident that with the implementation of the motor feedback system, the speed control of the motor will be significantly improved, and the overall precision of the experiment will be enhanced.

The team's commitment to addressing this issue underscores its dedication to developing a high-performance system that delivers reliable and precise results. Therefore, the integration of the motor feedback system is a critical step towards achieving this objective, and the team is actively working to ensure that this is accomplished in a timely and effective manner.

Improvements

To enhance the device's functionality, multiple linear flexures can be printed to allow the lab to test different displacement scenarios, such as vibrations near dams, wind turbines, or factories. This will provide flexibility in adjusting the displacement to suit different experimental conditions. In addition, a laser-based measurement system can be considered for more accurate displacement measurements, although this may be a more expensive solution. Moreover, the functionality and privacy of the email-sending process in this project can be improved in a few specific areas. Initially, a different way of sending emails without using the phone number might be investigated to assure greater privacy. For instance, the Google SMTP server might be set up to verify the user without using the phone number using a different authentication method, such as OAuth 2.0 [42]. This would offer a way to send emails that is more private and secure. Other benefits of utilizing SQLite include enhanced scalability, dependability, and security features when integrating a real-time database like Firebase, therefore a real-time database would be the

best option for large-scale trials since it would enable more effective data synchronization and better management of concurrent connections [39].

In future updates of the current user interface for controlling experiments, the two main limitations that impact its usability are likely to be addressed. Multi-experiment control will be implemented, allowing researchers and experimenters to manage several experiments simultaneously from a single UI or control panel. This would greatly improve the efficiency of managing multiple experiments and make it easier to compare data and results across different experiments. This will come with performance optimizations to allow for smooth control over all systems.

9 Conclusion

The main objective of this project was to provide a vibration simulator with which the Davy lab can investigate the effects of ground vibration from various industrial sources on the growth of turtle eggs. Shaking tables exist that can produce frequencies between 5 – 20 Hz, however none were found that could achieve the fine precision of 0.1 – 1 mm displacement that is required. Further, these tables were prohibitively expensive for the lab, the least expensive being \$5000.

The team created vibrations of 0.02 – 0.04 mm of displacement and frequency of 5 – 20 Hz. This extreme precision in displacement can be attained by utilizing the James Webb Space Telescope mirror positioning system's elegant yet straightforward flexure-guided linear actuator design. From this starting point, the team customized the flexure to achieve higher displacements, and created a motor feedback system so that frequency of vibration could be selected and changed by the user. Moreover, the team designed a user-friendly interface and data collection system that enables researchers to monitor environmental variables such as temperature and humidity in addition to vibration frequency and displacement.

Undoubtedly, the team's ground vibration simulator device has provided an innovative solution to meet the unique needs of the Davy lab. This device offers a custom and expandable design, which provides flexibility to adapt to the changing needs of the lab. It is worth noting that such a solution is not commercially available, making the team's contribution even more significant. Moreover, the device is designed to be cost-effective, with a budget of only \$1000 for the components. However, it is essential to keep in mind that the selling price of the device, including labour costs of 200 hours each for five engineers, could easily exceed \$5000 if commercialized. Due to the nature of this being a fourth-year engineering project, labour costs were not accounted for regarding this prototype. The device's customizability, flexibility, and cost-effectiveness make it a valuable addition to the Davy lab and for conservation research.

Without labour costs, the vibration simulator and data collection system can be built for \$695.34, and additional tables can be added to the system for \$176.16. This price point and open-source

design makes researching ground vibration effects on species accessible to the Davy lab and other labs.

The Davy lab is excited to begin experiments with the system this summer. Not only will they be researching the effects on turtle eggs but have found the system can accommodate other reptile species as well, such as snakes. The successful implementation of this project will pave the way for exciting research opportunities for the Davy lab and other research institutions that are interested in studying the effects of ground vibration on different species.

10 References

- [1] R. Sreekala, N. Lakshmanan, K. Muthumani, N. Gopalakrishnan and K. Sathishkumar, "Potential of Vibrations Studies in the Soil Characterization Around Power Plants - A Case Study," in *International Conference on Case Histories in Geotechnical Engineering*, Arlington, 2008.
- [2] P. Botha, "Ground Vibration, Infrasound and Low Frequency Noise Measurements from a Modern Wind Turbine," *Acta Acustica united with Acustica*, vol. 99, no. 4, pp. 537-544, 2013.
- [3] M. Srbulov, "2.3 Industry," in *Practical Soil Dynamics: Case Studies in Earthquake and Geotechnical Engineering*, United Kingdom, Springer Science+Business Media, 2011, pp. 35-36.
- [4] M. Srbulov, "12 Ground Vibration Cause by Industry," in *Practical Soil Dynamics: Case Studies in Earthquake and Geotechnical Engineering*, United Kingdom, Springer Science+Business Media, 2011, pp. 233-257.
- [5] S. J. Doody, B. Stewart, C. Camacho and K. Christian, "Good vibrations? Sibling embryos expedite hatching in a turtle," *Animal Behaviour*, vol. 83, no. 3, pp. 645-651, 2012.
- [6] B. Serralheiro-O'Neill, "Snapping Turtle," The Canadian Encyclopedia, 6 05 2021. [Online]. Available: <https://www.thecanadianencyclopedia.ca/en/article/snapping-turtle>. [Accessed 15 10 2022].
- [7] WBS.com, "What is a Work Breakdown Structure?," WorkBreakdownStructure.com, [Online]. Available: <https://www.workbreakdownstructure.com/>. [Accessed 27 03 2023].

- [8] J. Towner, "What is the Difference Between Agile and Waterfall?," Forecast, 27 06 2022. [Online]. Available: <https://www.forecast.app/blog/difference-between-agile-waterfall>. [Accessed 03 04 2023].
- [9] "p3 American LMC8 Series," p3 America, [Online]. Available: <https://p3america.com/lmc8-series>. [Accessed 02 11 2022].
- [10] "Miniature Lightweight LVDT," TE Connectivity, [Online]. Available: <https://www.te.com/usa-en/product-CAT-LVDT0021.html>. [Accessed 9 12 2022].
- [11] T. Cucchi, "Quality 101: The World of Indicators," Quality Magazine, 15 04 2022. [Online]. Available: <https://www.qualitymag.com/articles/96978-quality-101-the-world-of-indicators>. [Accessed 10 03 2023].
- [12] "Accelerometer Basics," SparkFun Electronics, [Online]. Available: <https://learn.sparkfun.com/tutorials/accelerometer-basics/all>. [Accessed 2 April 2023].
- [13] S. M. R. R. Sanjeev R. Kulkarni, "Frequency Domain and Fourier Transforms," 2002. [Online]. Available: https://www.princeton.edu/~cuff/ele201/kulkarni_text/frequency.pdf. [Accessed 7 April 2023].
- [14] "Heason - Linear Acuator Applications," 24 August 2020. [Online]. Available: <https://www.heason.com/news-media/technical-blog-archive/linear-actuator-applications>. [Accessed 9 December 2022].
- [15] D. Collins, "What are piezo flexure stages and how do they work?," Linear Motion Tips, 08 03 2022. [Online]. Available: <https://www.linearmotiontips.com/what-are-piezo-flexure-stages-and-how-do-they-work/>. [Accessed 12 03 2023].
- [16] D. Collins, "How does hysteresis affect piezo actuator performance?," Linear Motion Tips, 23 09 2021. [Online]. Available: <https://www.linearmotiontips.com/how-does-hysteresis-affect-piezo-actuator-performance/>. [Accessed 12 03 2023].
- [17] R. Warden, "Cryogenic Nano-Actuator for JWST," in *Proceedings of the 38th Aerospace Mechanisms Symposium*, Hampton, Virginia, 2006.
- [18] Polyfractal, "JWST Mirror Actuator," Thingiverse, 7 02 2022. [Online]. Available: <https://www.thingiverse.com/thing:5232214>. [Accessed 1 10 2022].
- [19] "DC motors and mechanisms," Precision Microdrives, 2022. [Online]. Available: <https://www.precisionmicrodrives.com/>. [Accessed 1 October 2022].

- [20] R. Santos and S. Santos, "ESP32 with DC Motor and L298N Motor Driver – Control Speed and Direction," Random Nerd Tutorials, [Online]. Available: <https://randomnerdtutorials.com/esp32-dc-motor-l298n-motor-driver-control-speed-direction/>. [Accessed 23 03 2023].
- [21] S. Wickramasinghe and M. Raza, "DBMS: Database Management Systems Explained," bmc, 09 12 2021. [Online]. Available: <https://www.bmc.com/blogs/dbms-database-management-systems/>. [Accessed 17 03 2023].
- [22] "Firebase Realtime Database," Google, 21 03 2023. [Online]. Available: <https://firebase.google.com/docs/database>. [Accessed 29 03 2023].
- [23] N. Fatima, "A quick overview of different types of databases," Astera, 11 06 2019. [Online]. Available: <https://www.astera.com/type/blog/a-quick-overview-of-different-types-of-databases>. [Accessed 01 04 2023].
- [24] "10 Database examples in real life," Liquid Web, 21 06 2021. [Online]. Available: <https://www.liquidweb.com/blog/ten-ways-databases-run-your-life/>. [Accessed 03 04 2023].
- [25] Wikipedia, "Bank Statement," Wikipedia, 28 March 2023. [Online]. Available: https://en.wikipedia.org/wiki/Bank_statement#/media/File:BankStatementChequing.png. [Accessed 30 March 2023].
- [26] Statista Research Department, "Ranking of the most popular relational database management systems worldwide, as of January 2022," Statista, 23 05 2022. [Online]. Available: <https://www.statista.com/statistics/1131568/worldwide-popularity-ranking-relational-database-management-systems/>. [Accessed 21 10 2022].
- [27] "How to install MariaDB on RPi," Raspberry Tips, 2023. [Online]. Available: <https://raspberrytips.com/install-mariadb-raspberry-pi/>. [Accessed 08 04 2023].
- [28] "SQLite as an application file format," SQLite org, 08 01 2022. [Online]. Available: https://www.sqlite.org/aff_short.html. [Accessed 08 04 2023].
- [29] A. Panwar, "C-Sharpcorner," 17 01 2022. [Online]. Available: <https://www.c-sharpcorner.com/UploadFile/65fc13/types-of-database-management-systems/>. [Accessed 21 10 2022].
- [30] ActiveState, "ActiveState," 9 August 2022. [Online]. Available: <https://www.activestate.com/blog/top-10-python-gui-frameworks-compared>.

- [31] L. P. Ramos, "RealPython," 2021. [Online]. Available: <https://realpython.com/qt-designer-python/>.
- [32] Adafruit, "DHT sensor library," Adafruit, [Online]. Available: <https://www.arduino.cc/reference/en/libraries/dht-sensor-library/>. [Accessed 20 01 2023].
- [33] "Fusion 360," Autodesk, [Online]. Available: <https://www.autodesk.ca/en/products/fusion-360>. [Accessed 20 11 2022].
- [34] IBM.com, "What is a database schema?," IBM, [Online]. Available: <https://www.ibm.com/topics/database-schema>. [Accessed 08 04 2023].
- [35] C. Dickson, "How to create and manipulate sql databases with python," freecodecamp, 31 09 2020. [Online]. Available: <https://www.freecodecamp.org/news/connect-python-with-sql/>. [Accessed 26 03 2023].
- [36] "docs.python," Python, 04 2023. [Online]. Available: <https://docs.python.org/3/library/sqlite3.html>. [Accessed 08 04 2023].
- [37] W3Schools, "SQL tutorial," W3schools, [Online]. Available: https://www.w3schools.com/sql/sql_ref_table.asp. [Accessed 17 03 2023].
- [38] A. Bottijen, "Google no longer allows username and passwords on third-party email applications," Neowin, 19 06 2022. [Online]. Available: <https://www.neowin.net/news/google-no-longer-allows-username-and-passwords-on-third-party-email-applications/>. [Accessed 08 12 2022].
- [39] B. Guriev, "Firebase Pros and Cons," OSDB, 15 03 2021. [Online]. Available: <https://osdb.io/firebase-pros-and-cons-when-you-should-and-shouldnt-use-firebase-osdb/>. [Accessed 08 04 2023].
- [40] P. Pedamkar, "RDMS vs NoSQL," EDUCBA, [Online]. Available: <https://www.educba.com/rdbms-vs-nosql/>. [Accessed 07 04 2023].
- [41] E. S, "SQLite vs MySQL – What's the Difference," 23 08 2022. [Online]. Available: <https://www.hostinger.com/tutorials/sqlite-vs-mysql-whats-the-difference/>. [Accessed 21 10 2022].

[42] J. Hrisko, "Arduino Internet of Things," MakerPortal, 18 04 2018. [Online]. Available: <https://makersportal.com/blog/2018/4/7/arduino-internet-of-things-part-5-raspberry-pi-as-a-smart-hub-for-sending-data-to-google-sheets>. [Accessed 17 03 2023].

Appendix 1: Relevant Courses

The team has collectively taken many different courses relevant to the project. For reference, the course codes and course titles mentioned throughout the document will be outlined here:

Table 9 – Cumulative relevant courses for required knowledge and experience.

Course Codes	Title
ECOR 1010	Introduction To Engineering
ECOR 1101	Mechanics I
SYSC 2004	Object Oriented Software Development
SYSC 3600	Systems and Simulation
SYSC 3010	Computer Systems Development Project
SYSC 3020	Introduction to Software Engineering
SYSC 4805	Computer Systems Design Lab
SYSC 4505	Automatic Control Systems I
ELEC 2501	Circuits and Signals
ELEC 2507	Electronics I
ELEC 3105	Basic EM and Power Engineering (previously), Electromagnetic Fields (current)
ELEC 3907	Engineering Project
ELEC 3509	Electronics II
CCDP 2100	Communication Skills for Engineering
ECOR 4995	Professional Practice

Appendix 2: Additional Diagrams

Measuring Frequency with an Accelerometer

Figure 28 illustrates the complete code utilized to calculate frequency using the raw data obtained from the accelerometer connected to the Arduino R3 Uno.

```

1  #define SAMPLES 256 // number of samples to take for FFT
2  #define SAMPLING_FREQ 1000 // sampling frequency in Hz
3  #define ADC_RANGE 1024 // analog-to-digital conversion range
4  #define VOLTAGE_RANGE 3.3 // voltage range of accelerometer output
5
6  const int zPin = A2; // define analog pin for z-axis input
7
8  float zRaw; // variable to store raw sensor reading
9  float zAcc; // variable to store accelerometer reading in g
10 float zData[SAMPLES]; // array to store accelerometer data for FFT
11 float zFreq[SAMPLES/2]; // array to store frequency spectrum
12
13 void setup() {
14   Serial.begin(9600); // initialize serial communication at 9600 bps
15 }
16
17 void loop() {
18   // read accelerometer raw data
19   zRaw = analogRead(zPin);
20
21   // convert raw data to accelerometer reading in g
22   zAcc = ((zRaw / ADC_RANGE) * VOLTAGE_RANGE - VOLTAGE_RANGE/2) / 0.330; // assuming a sensitivity of 330 mV/g for z-axis
23
24   // add accelerometer data to FFT array
25   for (int i = 0; i < SAMPLES; i++) {
26     zData[i] = zAcc;
27   }
28
29   // perform FFT on accelerometer data
30   for (int i = 0; i < SAMPLES/2; i++) {
31     float real = 0;
32     float imag = 0;
33     for (int j = 0; j < SAMPLES; j++) {
34       float angle = 2 * PI * i * j / SAMPLES;
35       real += zData[j] * cos(angle);
36       imag -= zData[j] * sin(angle);
37     }
38     zFreq[i] = sqrt(real * real + imag * imag) / SAMPLES;
39   }
40
41   // calculate frequency spectrum
42   for (int i = 0; i < SAMPLES/2; i++) {
43     zFreq[i] *= 2;
44     zFreq[i] /= VOLTAGE_RANGE;
45     zFreq[i] /= SAMPLES;
46     zFreq[i] *= SAMPLING_FREQ;
47   }
48
49   // Print frequency to serial monitor
50   Serial.print("Frequency: ");
51   for (int i = 0; i < SAMPLES/2; i++) {
52     Serial.print("Frequency: ");
53     Serial.print(zFreq[0]);
54     Serial.println(" Hz");
55   }
56   Serial.println();
57   delay(100); // wait for 1 second before taking the next sample
58 }
59

```

Figure 28 – Code utilized to calculate frequency using raw data of accelerometer

Motor

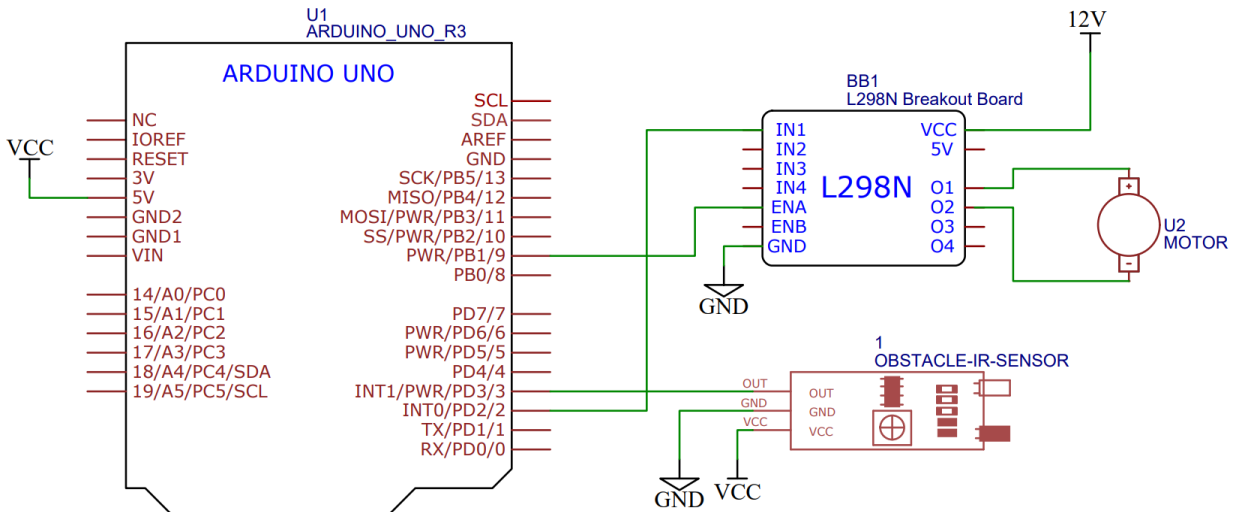


Figure 29 – Motor design schematics

Figure 29 shows the wiring of the motor design. If an extra motor needs to be added it can be plugged in O3 and O4 and IN3 plugged to any of the programmable pins and adjust the code accordingly. It is important to note that when connecting the 12V VCC of the L298N driver, the ground should be common between the Arduino and the L298N driver. That can be achieved by connecting a wire between the ground of the driver and any of the GND pins that are built in the Arduino board.

The following code (figure 29) is used to control the motor and IR sensor with the connections shown above.

```

1  // Define the pin for the IR sensor
2  int IR_PIN = 3;
3  int motor1pin1 = 2;
4  // Define variables to hold the values read from the IR sensor
5  int IR_STATE = 0;
6  int last_IR_STATE = 0;
7
8  // Define variables to hold the time intervals between sensor readings
9  unsigned long current_time = 0;
10 unsigned long last_time = 0;
11
12 // Define a variable to hold the RPM value
13 int RPM = 0;
14
15 void setup() {
16     // Initialize the serial port
17     Serial.begin(9600);
18     pinMode(motor1pin1, OUTPUT);
19     pinMode(9, OUTPUT);
20     // Set the IR sensor pin as an input
21     pinMode(IR_PIN, INPUT);
22 }
23
24 void loop() {
25     analogWrite(9, 0);
26
27     digitalWrite(motor1pin1, HIGH);
28
29     // Read the state of the IR sensor
30     IR_STATE = digitalRead(IR_PIN);
31
32     // If the state of the sensor has changed
33     if (IR_STATE != last_IR_STATE) {
34         // Record the time of the change
35         current_time = micros();
36
37         // Calculate the time interval since the last change
38         unsigned long time_interval = current_time - last_time;
39
40         // If the time interval is greater than 200 microseconds (corresponding to a frequency of 5000 Hz)
41         if (time_interval > 200) {
42             // Calculate the RPM value
43             RPM = 60000000 / (time_interval * 20); // 20 is the number of slots on the encoder disc
44
45             // Print the RPM value to the serial port
46             Serial.println(RPM);
47             last_time = current_time;
48         }
49
50         // Update the last state variable
51         last_IR_STATE = IR_STATE;
52     }
53 }

```

Figure 30 – Motor and IR sensor code

Database

Figure 31 below illustrates the database schema presented in the initial proposal.

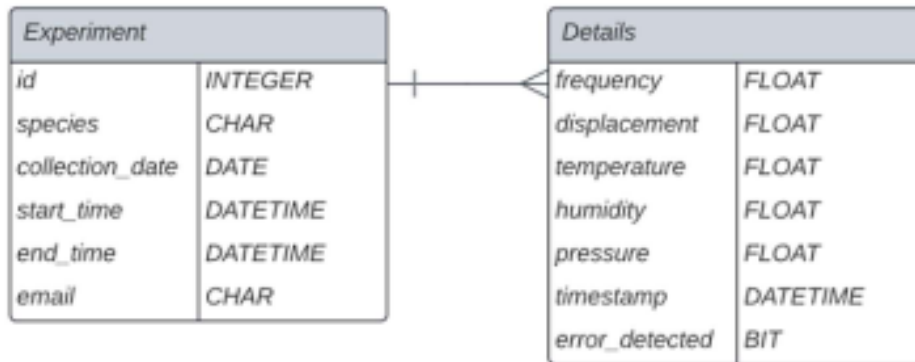


Figure 31 – Database schema from the proposal.

Figure 32 below illustrates the database schema presented in the progress report.

id	INTEGER
species	TEXT
collection_date	DATE
start_time	DATETIME
email	TEXT
frequency	FLOAT
displacement	FLOAT
temperature	FLOAT
humidity	FLOAT
pressure	FLOAT
timestamp	DATETIME
error_detected	TEXT

Figure 32 – Database schema from the progress report.

User Interface

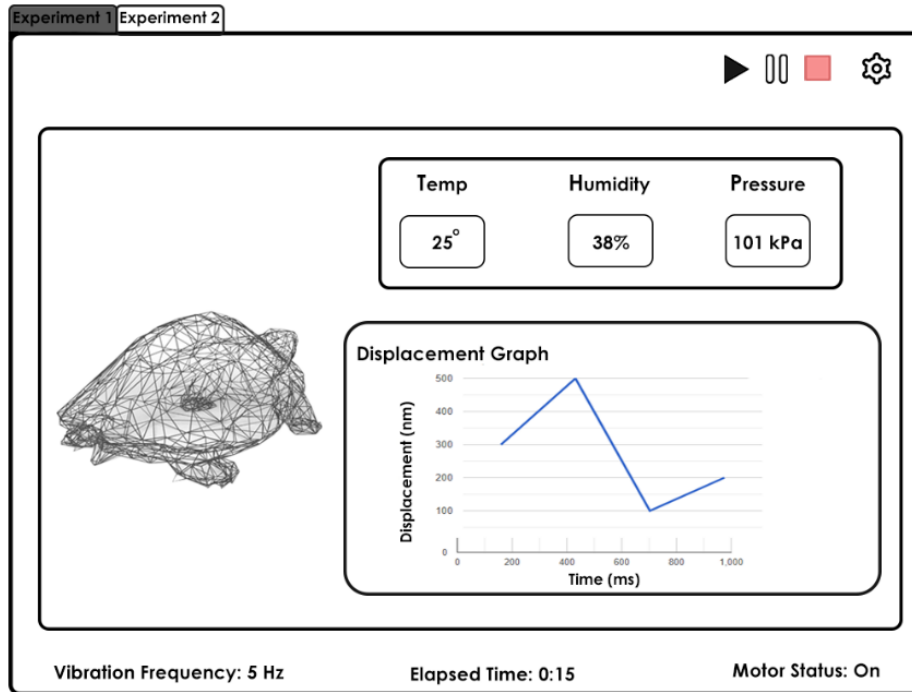


Figure 33 – Home screen UI Wireframe

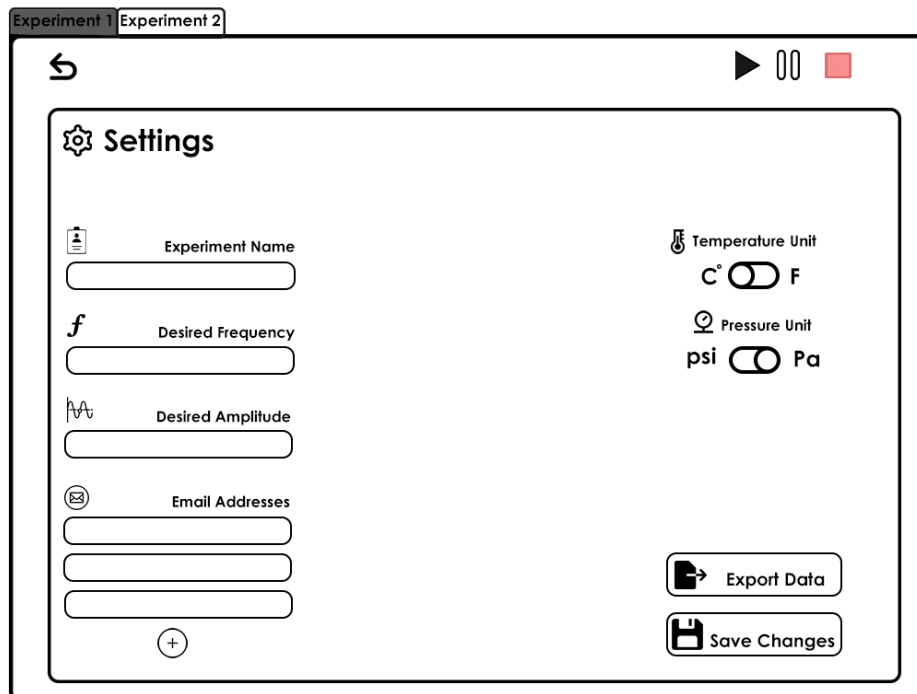


Figure 34 – Setting Screen UI Wireframe

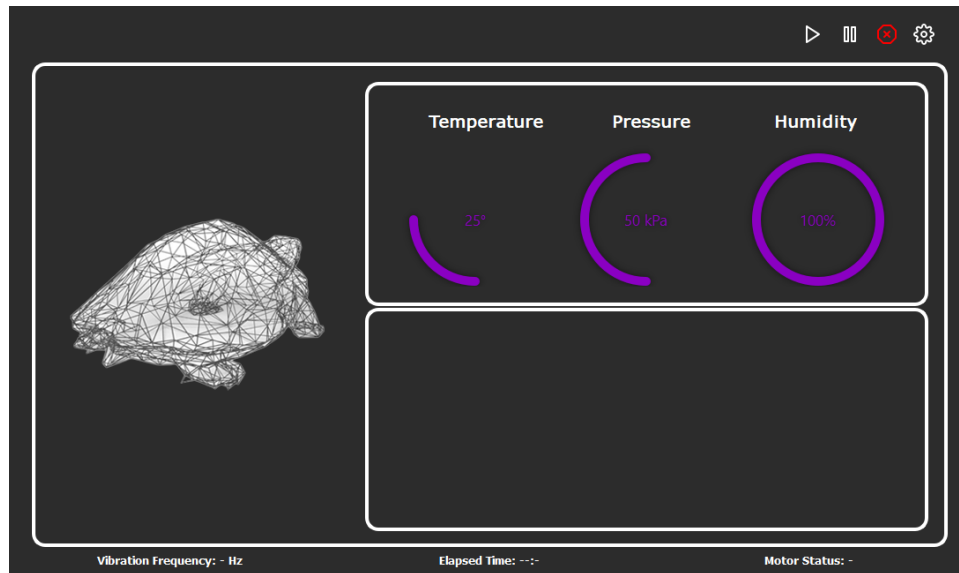


Figure 35 – Home screen UI First Version



Figure 36 – Settings screen UI First Version

Appendix 3: Costs

These cost breakdowns contain before-tax prices.

Research and Development Costs

Table 10 – Components ordered for research and development purposes.

Item	Source	Price
DC Brushless Motor	BC-Robotics.com	\$1.95
DC Brushless Motor	Amazon.ca	\$23.53
Motor driver	TexasInstrument.com	\$54.53
Motor Driver	BC-Robotics.com	\$9.99
MC3479 Accelerometer	Digikey.ca	\$21.47
MSP430 LaunchPad MCU	Digikey.ca	\$22.94
Arduino Uno	Amazon.ca	\$23.99
Differential Gears 64T, 17T, 21T, 26T, 29T	Amazon.ca	\$16.29
5mm Motor shaft	Amazon.ca	\$8.54
3mm Motor shaft	Amazon.ca	\$7.59
3mm to 5mm Shaft coupler	Amazon.ca	\$11.99
2mm to 3mm Shaft coupler	BCRobotics	\$3.95
Shaft collar 8mm	Amazon.ca	\$9.99
625-2rs Ball Bearings (x4)	Amazon.ca	\$16.03
Linear shaft 12mm x 150 mm (x2)	Amazon.ca	\$14.99
Linear shaft 12 mm bearing PIL (x2)	Digikey.ca	\$18.74
Linear shaft 12mm clamp	Amazon.ca	\$17.49
Stiff springs (x4)	Amazon.ca	\$43.66
Melamine surface	Homedepot.ca	\$26.75
Knotty Pine Board 2x3x8	Homedepot.ca	\$16.98
5/16-18 Nylon Insert Nut (x3)	Ottawa Fastener Supply	\$0.48
5/16 X 2 Tap Bolts (x3)	Ottawa Fastener Supply	\$2.10
A2 M5x16 Phillips Head Screw (x8)	Ottawa Fastener Supply	\$1.76
#12 1 ¼" Sharp Point screw (Pack of 8)	Ottawa Fastener Supply	\$3.79
Incubator	Provided by lab	\$0
Raspberry Pi 4, 4GB	Digikey.ca	\$80.95
Temperature & Humidity sensor	Amazon.ca	\$16
Mitutoyo 543-789 Digimatic Indicator	Amazon.ca	\$264.43
Mitutoyo SPC Connecting Cable	Amazon.ca	\$72.58
IR Sensor Module	Amazon.ca	\$10.89
Indicator Holder	Amazon.ca	\$30.00
	TOTAL	\$854.37

Additional Table Costs

Table 11 – Components needed to build additional table for system.

Item	Source	Price
DC Brushless motor	BC-Robotics.com	\$1.95
Motor driver	BC-Robotics.com	\$9.99
Arduino	Amazon.ca	\$23.99
Differential Gears 64T, 17T, 21T, 26T, 29T	Amazon.ca	\$16.29
2mm to 3mm Shaft coupler	BCRobotics	\$3.95
Linear shaft 12mm x 150 mm (x2)	Amazon.ca	\$14.99
Linear shaft 12 mm bearing PIL (x2)	Digikey.ca	\$18.74
Linear shaft 12mm clamp	Amazon.ca	\$17.49
Stiff springs (x4)	Amazon.ca	\$43.66
Knotty Pine Board 2x3x8	Homedepot.ca	\$16.98
5/16-18 Nylon Insert Nut (x3) (\$ 0.16 each)	Ottawa Fastener Supply	\$0.48
5/16 X 2 Tap Bolts (x3) (\$0.70 each)	Ottawa Fastener Supply	\$2.10
A2 M5x16 Phillips Head Screw (x8) (\$ 0.22 each)	Ottawa Fastener Supply	\$1.76
#12 1 ¼" Sharp Point screw (Pack of 8)	Ottawa Fastener Supply	\$3.79
Incubator	Provided by lab	\$0
	TOTAL	\$176.16

Appendix 4: Work Plan

Collaboration

To communicate updates to one another, supervisors, and the Davy lab, the team used a variety of communication platforms. We used Microsoft Teams for sharing documents and for discussion. A repository for code, documents, and designs was used on [GitHub](#). All work completed was published in the repository as open source so that other labs may use it for future research. Every Monday at 2:30 pm, the team members, managers, and Dr. Davy gathered for progress meetings. Additionally, Jelena Nikolic-Popovic, a Senior Member of Technical Staff at Texas Instruments Canada, worked with the team (TI). She donated some hardware and offered her knowledge of the TI equipment that was used for research in the project, but ultimately not used.

Project Milestones

Not all milestones were reached within the scope of the course, as development was bottlenecked by the motor feedback system being incomplete. The team plans to finish the project so that the Davy Lab may use it for Summer 2023. Completion dates reflect these altered goals.

As is reflected in this report, incomplete milestones are approximately 80% - 90% complete as of submission of this report.

Table 12 – Milestone descriptions and completion dates.

Milestone	Completed	Description	Status and Remarks
Hardware Design	Oct. 21	Research will be used to come up with final designs for the vibration mechanism and measurement to be implemented.	Complete
Software Design	Oct. 21	Different components of the software system will be chosen and designed, such as GUI wireframes, framework for GUI development, operating system to be used, database schemas, and languages/libraries to use for software-hardware interfacing.	Complete
Finalize Part Orders	Oct. 21	Parts required for the final designs can be ordered to enable development as soon as possible.	Complete
Hardware Testing Phase	May 1	Tests will be conducted as the linear actuator design and hardware setup is	In-progress

		iterated upon. Different motors, gear configurations, and flexure and camshaft designs will be examined to determine the final design is able to hit all required frequencies and displacements.	
Software Development	April 20	Various components will be implemented such as GUI, database, communication protocols, custom libraries to retrieve data, and signals to control vibration mechanism.	In-progress
Hardware Development	April 20	Once a final design is decided upon, the project's hardware components will be assembled. This phase includes having parts manufactured as needed.	In-progress
Software-Hardware Integration	April 20	Final integrations between the hardware and software will be completed, such as retrieving data from all sensors and controlling the vibration mechanism.	In-progress
Integration Testing	May 10	Final integrations between the hardware and software will be thoroughly tested.	In-progress
Acceptance Testing	May 10	Tests will be conducted to ensure the project works as is required by the Davy lab.	In-progress

Schedule of Activities/Gantt Chart

Table 13 – Schedule of activities for project completion, documentation, and presentations.

Task	Begin	Draft	Completed	Status
Kickoff meeting between engineering team and lab	-	-	Aug. 26	Complete
Research	Summer	Sept. 7	Sept. 30	Complete
Proposal	Sept. 7	Sept. 30	Oct. 21	Complete
Hardware & Software Designs	Sept. 7	Oct. 8	Oct. 21	Complete
Finalize Part Orders	-	-	Sept. 30	Complete
Hardware Test Phase	Oct. 1	-	April 20	In-progress
Test Plan	Oct. 31	-	Jan. 15	Complete
Development – Software & Hardware	Sept. 30	Jan. 1	April 20	In-progress
Software-Hardware Integration	Feb. 2	-	April 20	In-progress
Progress Report	Nov. 1	Nov. 18	Dec. 9	Complete
Oral Presentations	Jan. 9	Form – Dec. 9	Jan. 23-27	Complete
Integration Testing	Feb. 15	-	May 10	In-progress
Acceptance Testing	Feb. 28	-	May 10	In-progress
Poster Fair	March 1	-	March 17	Complete
Final Report and Video	Jan. 15	1 st – Feb. 17 2 nd – March 24	April 12	Complete

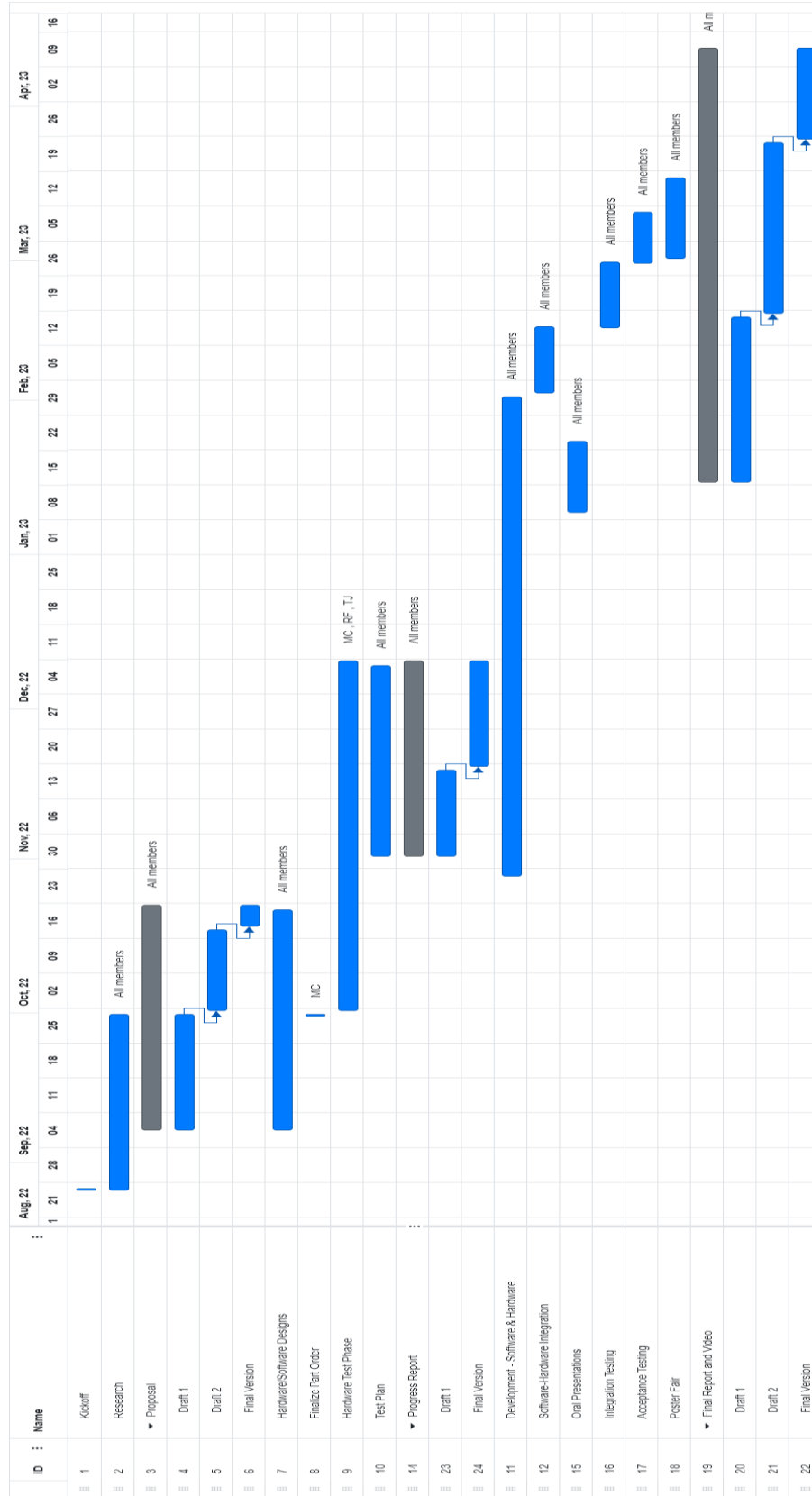


Figure 37 – Gantt chart

Appendix 5: Project Links

Demonstration video available: <https://youtu.be/OBzAEXO5CB8>

Github containing project available:

https://github.com/bienemeia/SYSC4907_Group44_GroundVibrationSimulator.git

Appendix 6: Progress Report

Progress Report is available: [Progress Report](#)

Appendix 7: Proposal

Proposal Document begins on next page.