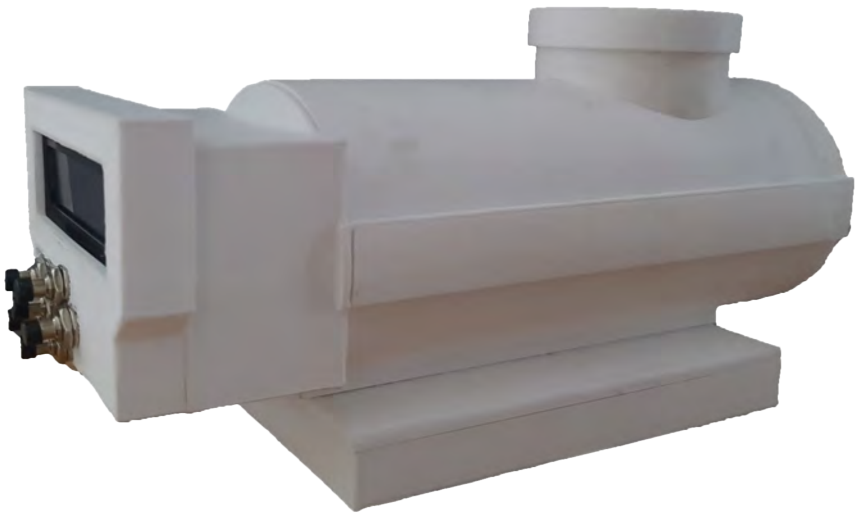
DEGREE PROJECT IN TECHNOLOGY,
FIRST CYCLE, 15 CREDITS
STOCKHOLM, SWEDEN 2021

Secure Fish Feeder

Automatic secure fish feeder to make sure you do not overfeed your fish

ELIAS GRIFFITH

JACOB ILJANS



KTH ROYAL INSTITUTE OF TECHNOLOGY
SCHOOL OF INDUSTRIAL ENGINEERING AND MANAGEMENT

Secure Fish Feeder

Automatic secure fish feeder to
make sure you do not overfeed
your fish

ELIAS GRIFFITH
JACOB ILJANS

Bachelor's Thesis at ITM
Supervisor: Nihad Subasic
Examiner: Nihad Subasic

TRITA-ITA-EX 2021:21

Abstract

This project aims to provide fish keepers with a safe and simple way to feed their pets while abroad or in any other situation where they do not have the ability to feed their fish by hand.
Overfeeding is an unfortunate albeit common reason for mass death in an otherwise safe and healthy aquarium. While overfeeding to land dwelling pets might have negative health implications, the rapid damage seen in an aquarium is unparalleled in other environments. This is due to the release of ammonia that occurs when excess food (and partially fecal matter) is decomposed. Moderate feedings provide adequate nutrition for the pets, while keeping ammonia production at a level where it can be converted to nitrite and lastly the rather harmless compound nitrate.
Most, if not all, feeders on the market today base their feedings solely off of volume, but this can lead to large variations in feeding size, due to variations in air between pellets in the feeding portions. By adding a failsafe with weight, we can fully prevent any chance of overfeeding, and thus ensuring a long and healthy life for our pets. Overall the goals of a safe fish feeder were accomplished, although the weighing was not as accurate as it optimally would have been.

Keywords: mechatronics, fish, aquarium, feeder, food

Referat

Säker fiskmatare

Med det här projektet syftar vi på att tillförse akvarieägare med ett tryggt och enkelt sätt att mata sina fiskar när de är utomlands eller av annan anledning inte har möjlighet att mata för hand. Övermatning är en olycklig men dessvärre vanlig anledning till massdöd i ett annars hälsosamt akvarium. Även om övermatning hos landdjur kan ha ohälsosamma konsekvenser, så är det aldrig lika skadligt på kort sikt som det är i ett akvarium. Det här beror på att överbliven mat (och även till viss del avföring) i en vattenmiljö snabbt omvandlas till ammoniak under förmultning. Lagom stora matningar ger all näring fiskar behöver, medan enbart en halt ammoniak som kan omvandlas till nitrit och därefter det relativt harmlösa nitratet uppstår snabbt nog för att vara ofarligt. De flesta - om än inte alla - fiskmatare på marknaden idag baserar sina portioner på enbart volym. På grund av variationer i mängden luft mellan foder så kan då mängden mat ha stor variation. Genom att lägga till en säkring med vikt, så kan vi helt och hållet motverka all risk för övermatning, och på så vis säkerställa ett långt och hälsosamt liv för våra husdjur. Överlag lyckades vi skapa en säker fiskmatare, men dessvärre var inte vågen lika noggrann som den optimalt hade varit.

Nyckelord: mekatronik, fisk, akvarium, matare, foder

Acknowledgements

We would like to thank everybody who has helped us through this project. Not only the many online sources of information provided to us, but also those who have helped guide us through this course. This especially includes Nihad Subasic for his excellent feedback and guidelines, as well as information and recommendations on how to plan and follow the course through all its steps.

We would also like to thank Staffan Qvarnström for helping us obtain parts for this project, and the course assistants for helping us pick appropriate parts. And lastly, we would like to thank the kind members of the Fishlore community for helping us gather information on proper fish keeping techniques.

Contents

List of Figures

List of Tables

Abbreviations

PMW - Pulse width modulation
KEXPO - Exhibition for the bachelors work in mechatronics at KTH.
IDE - Integrated development environment
CAD - Computer-aided design
KTH - Royal Institute of Technology
RPM - Revolutions per minute
DC - Direct Current
USB - Universal Serial Bus
GUI - Graphical User Interface
LCD - Liquid Crystal Display
3D - Three dimensional

Chapter 1

Introduction

This chapter will give a quick introduction to what this project in mechatronics is about. It is going to be presenting the *Background*, *Purpose*, *Scope* and *Method* of the project.

1.1 Background

As all avid fish owners can attest, dying fish are an unfortunate event that is largely unavoidable. It is however possible to limit this by using proper maintenance and care.
Two leading causes for these events are spikes in ammonia (essentially naturally occurring poison for fish) and bloat. Both are mainly caused due to overfeeding. The already existing automated fish feeders tend to be not as precise in the amount of fish food it delivers each time. This project will be comparing the amount of food the fish will be fed with and without the added security feature of the scale.

1.2 Purpose

The goal with this project is to create an automatic feeder with a safety feature. Current feeders on the market tend to overfeed your aquarium, which is far from optimal, and can cause death of fish. The main idea behind this project is to build a secure automated fish feeder that measures how much food it drops every time it feeds and makes sure that the fishes get the proper amount of food every 24 hours.

To solve this problem, the following questions will be answered:

- *How accurately can the food be measured?*
- *What kind of safety features are going to be required for it to be safe?*
- *Is it safer to measure the food with a scale or the number of rotations the feeder will perform?*
- *How much ammonia is produced per gram of food?*

1.3 Scope

This project is put together by three different requirements, which consists of a mechanical part, an electric part and software to run the construction. These requirements do of course limit the number of available solutions for this project. The time frame of the project was 4 months, which would also affect the final product in the way that maybe not all the desired functions would be ready in time, and just the basic functions of the automated fish feeder. But if time permits, the prototype will be equipped with proper feeding and several safety features to ensure that the fish does not get overfeed. The budget for this project is 1000 SEK provided by KTH, this amount of money is also a factor to be considered in the result. The result was originally supposed to be presented at the fair for bachelors work in mechatronics at KTH KEXPO in May 2021. However due to the Covid-19 global pandemic the presentation and the whole exhibition was moved to an online website.

1.4 Method

To achieve safe feedings, a few milestones must be met. Firstly, find a safe way to store the food. Secondly, a reliable dosing method. Thirdly, a way to weigh the feeding amount, and make sure it is no more and no less than the desired portions.

For this to be accomplished, the following components will be used; an Arduino to steer all the components, a motor or servo to move each meal, a load cell combined with an amplifier for measuring the food that is being released from the feeder, and a display with physical buttons where the user can choose how much and when to feed the fish. All the parts of the fish feeder were designed in CAD and then 3D-printed. All materials that come into direct contact with the fish food needs to be aquarium safe as well. In other words, no plastics that are below food grade, no adhesives with risk of leaching toxic chemicals, and so forth. The project's code was written and developed in the programming language C, which is the language that is being used by the IDE for the Arduino [1]. During the same time as the code was written the electronics were being put together and tested, first in the free online modeling program Tinkercad [2] and then physically to make sure everything worked fine.

Chapter 2

Theory

This chapter will be discussing the theory behind the secure fish feeder. The required information about fish, what consequences overfeeding the fish may have, and the theory behind the components that are used or could be used in this project.

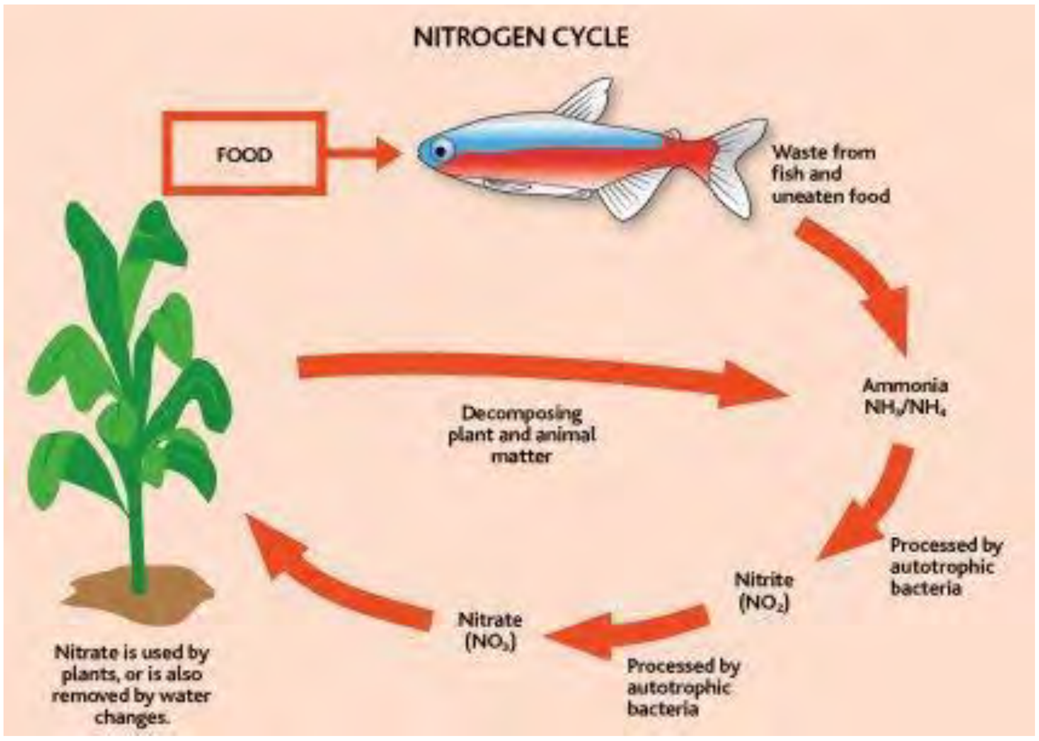
2.1 Fish theory

For this to be a successful project, a fundamental understanding about the fish feeding process must be achieved. This subchapter will be discussing the effects overfeeding fish has and how to properly handle and store the fish food.

2.1.1 The effects of ammonia on fish

One of the biggest factors for fish to be happy is the water quality of the tank. One big factor in the water quality is the amount of ammonia present. Ammonia is composed of nitrogen and hydrogen; it is a colorless gas with a strong pungent smell. Ammonia takes to forms in water, one toxic, one non-toxic. The toxic form of ammonia has the chemical formula NH_3 and is the un-ionised form of the gas. The non-toxic form is the ionised form of the gas with the chemical formula NH_4. The way ammonia enters the water in the aquarium is mainly through the fishes' waste and excess food, since it becomes a by-product of the fishes' metabolism and decomposition of nutrients, see figure 2.1. Nature's way of dealing with this problem is algae, due to algae's ability to absorb some of the ammonia [3]. This however is not a long-term solution for the presence of ammonia in nature and therefore also not the solution in aquariums.

The level of toxicity in the ammonia is dependent on many different factors, the type of ammonia, the water temperature, and the length of exposure towards the ammonia. When fish get exposed to ammonia it affects the central nervous system and causes loss of equilibrium, increased breathing, cardiac output, and oxygen uptake. In extreme cases it can cause the fish to die. So, overfeeding the fish will result in uneaten food that will produce dangerous amounts of ammonia in the tank. Levels of ammonia should always remain at 0 ppm, and at the very most below 0.25 ppm [4].


NITROGEN CYCLE
FOOD
Waste from fish and uneaten food
Ammonia NH_3/NH_4
Decomposing plant and animal matter
Processed by autotrophic bacteria
Nitrite (NO_2)
Nitrate (NO_3)
Processed by autotrophic bacteria
Nitrate is used by plants, or is also removed by water changes.
Figure 2. 1. The Nitrogen Cycle, which shows that ammonia comes from uneaten and eaten food [5].

2.1.2 Food handling

How much food a fish eats varies vastly in what kind of fish you have. A good rule of thumb is to give them as much as they can eat in 5 minutes, it is always better to underfeed the fish if there is any uncertainty [6]. There is no exact amount that is right, just like with humans, every fish is different. The same goes for how often the fish should eat. There is no precise right amount of time to wait between feedings. A good rule of thumb here is to feed the fish once every day, but this varies between species. Most fishes take from 16-24 hours to digest their food. The number of times you feed the fish is not as important as the amount of food you feed. There is no problem with feeding the fish two times a day if the portions together add up to the daily amount of food that is right [6].

Different plastics have different categories, these categories are used to help consumers in choosing what plastic they want to use for different things. In the case of food handling in an aquatic setting there is a requirement for the plastic to be able to store food safely, without any chemicals or liquid leeching into the food and destroying it. In this case the best type of plastic to use is the HDPE (High-Density Polyethylene) due to it not letting in or out anything that will compromise the food, but any food grade plastic is considered safe [7].

2.2 Electrical theory

The other crucial part of this project is the understanding of the electrical components that will be used. This subchapter will discuss the potential components that will be considered for this project.

2.2.1 Servo Motor or DC Motor

A servo motor is very useful when wanting to achieve great rotating precision. Since it normally comes equipped with a control circuit that provides feedback with the current position of the motor shaft, see figure 2.2.

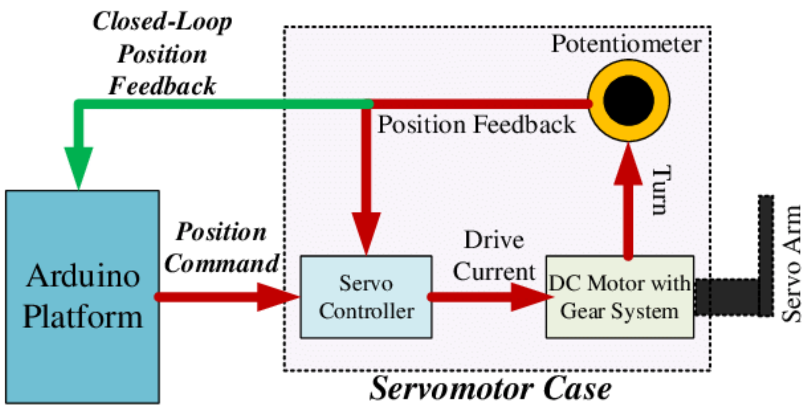


Figure 2. 2. Diagram that shows the internal parts of a servo motor [8].

The working mechanism in the motor consists of three different parts as displayed in figure 2.2.
One servo controller, one output sensor, in the case of figure 2.2, the potentiometer, and finally a feedback system. The closed-loop position feedback is not part of the servo motor. The position feedback inside the servo motor case is based on positive feedback and is used to control the motion and final position of the servo arm. This is done by comparing the output signal and the input signal, also known as the reference signal [9]. This control system can be observed in figure 2.3.

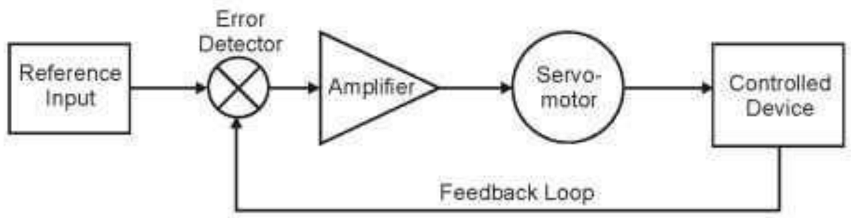


Figure 2. 3. Closed loop system of a servo motor [9].

An alternative to a standard servo motor would be a DC motor. A DC motor features a fixed magnet which is called a stator and a turning coil that is called an armature. It is powered by a DC source, in this case a battery; this is visualised in figure 2.4.

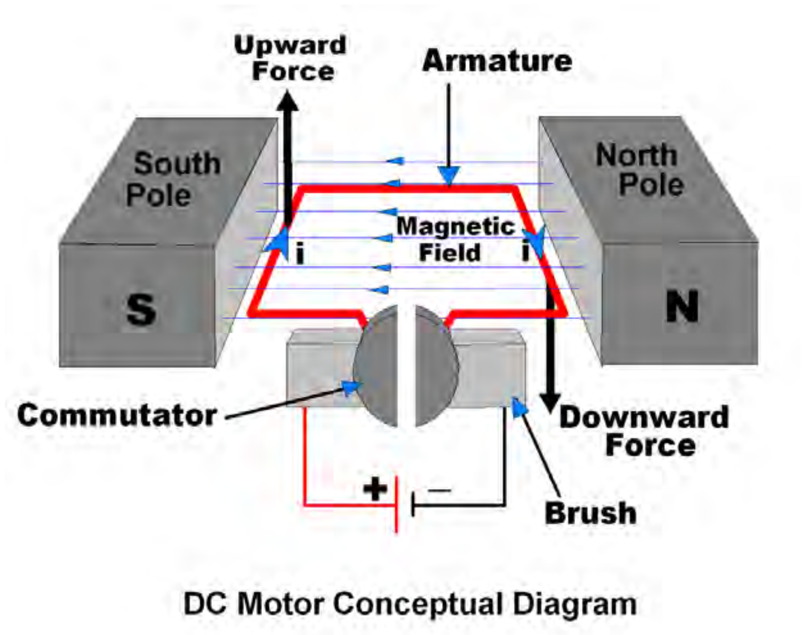


Figure 2. 4. Simple DC motor circuit [10].

The armature is connected to the battery making it carry a current. The current loaded armature is then generating a magnetic field making it an electromagnet. Placing the armature in the middle of the stator generates the force that makes the motor rotate [11]. There are different ways to control a DC motor, one example is to control it with PWM control.

2.2.2 PWM

To control the servomotor the use of PWM, Pulse Width Modulation is implemented. This technology is based around maximum and minimum pulses with a repetition rate between them. The PWM sends a pulse to the servo motor every 20 MS and the length of the maximum pulse determines how much the motor rotates [12]. For example, a maximum pulse that lasts less than 1 MS does not turn the motor at all, a maximum pulse that lasts 1.5 MS rotates the motor 90 degrees and a maximum pulse that lasts 2 MS rotates the motor 180 degrees, this is shown in figure 2.5.

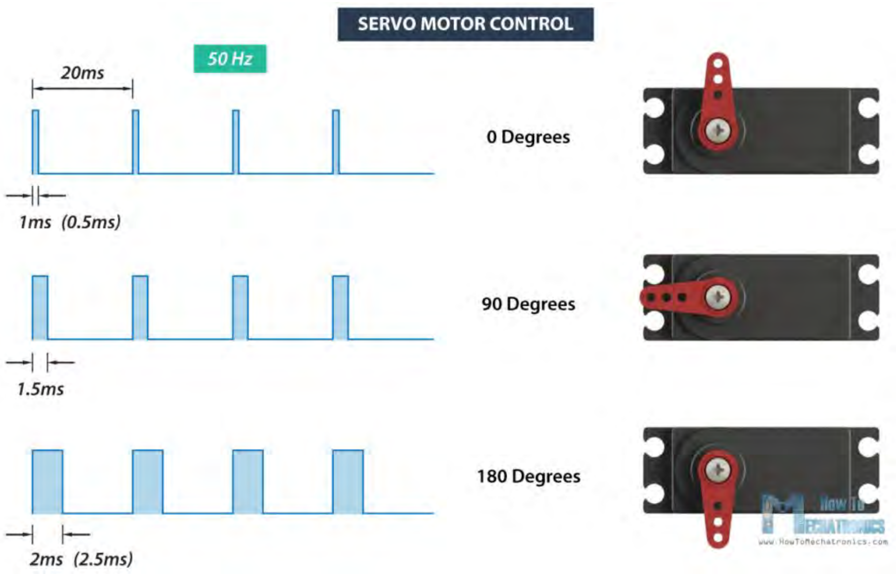


Figure 2. 5. Shows how big of a rotation the motor does for each length of the maximum pulse [13].

These numbers can differ depending on the make of servo motor, but generally they are as stated in figure 2.5.

The way it controls the DC motor is that it sends the pulses of electricity to the DC motor. The way that these pulses are sent will determine how much average voltage gets to the DC motor and thus control the rpm of the motor [14]. This can be visualised in figure 2.6.

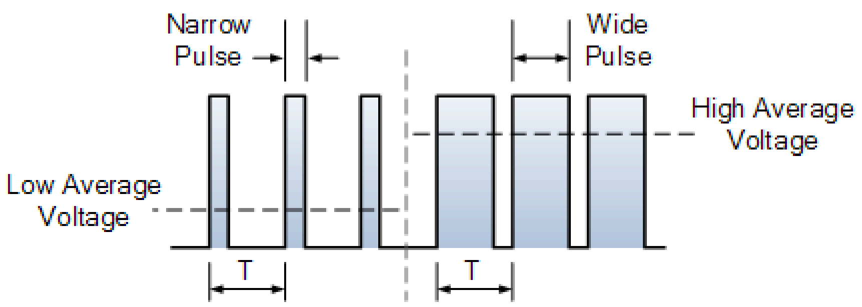


Figure 2. 6. Illustrating how the pulses give different average voltage [15].

A lower voltage gives a lower rpm, and a higher voltage gives a higher rpm. The motor however is always at full power since the amplitude of the voltage does not change. This means that even if lower average voltage is being used and the motor is rotating slower, the motor will not be stalling, it will just be rotating slower [15]. This is the same as having a geared down DC motor. It is also possible to combine these two to make the rpm even lower.

2.2.3 Microcontroller

A microcontroller is a small integrated circuit that controls a specific operation in an embedded system. It consists of processor, memory and inputs and outputs for different peripherals, all of this is fit together on a single chip. The microcontroller takes temporary data from the inputs and then stores it in the microcontrollers data memory [16]. Here it can be accessed whenever the processor needs it to perform the desired action of the program. The outputs are then used to communicate the actions of the program to the peripherals that are connected to the microcontroller [17]. A diagram of a microcontroller is shown in figure 2.7.

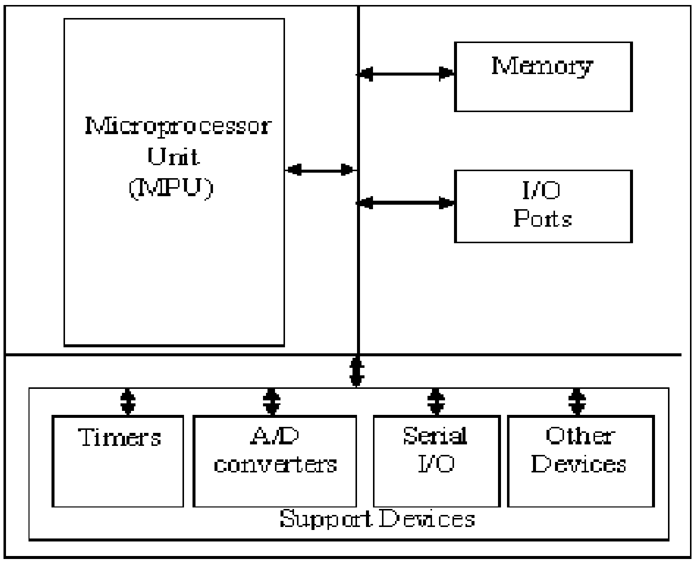


Figure 2. 7. A diagram of a microcontroller [18].

2.2.4 Load Cell

To accurately keep a steady diet for the fish, the feeding dosage needs to be weighed. To do this, a load cell will be implemented into the design of the feeder.

A load cell is essentially a transducer or a sensor that takes the kinetic energy that is placed on it and converts that force so that a computer can read the signal. Without the load cell the computer would not be able to measure the force that is being applied. There are many ways to achieve this, but what they all have in common is that they all exploit some type of physical property of the load cell. There are different types of forces that can be measured by different types of load cells. The different types of load cells are the following: Strain Gauge Load Cell, Hydraulic Load Cell, Pneumatic Load Cell, Capacitive Load Cell and Piezoelectric Transducers. All these load cells have a different way of measuring the force that is being applied to them.
The most widely known and used load cell of is the Strain Gauge Load Cell, which uses electric resistance to measure the amount of force that is applied to it. The load cells components consist of at least one string gauge, a loading platform for the force to be applied on, an excitation voltage source and output wires to measure the change in voltage caused by the change in resistance. The string gauge is essentially a wire that is etched onto a non-conductive material that connects to each end of the elastic wire. When the wire then

stretches or compresses the resistance will increase respectively decrease, this change in resistance is then converted by the load cell into a measurement. The fundamental circuitry that makes this kind of conversion possible is called a Wheatstone Bridge whose primary use is to measure electrical parameters such as resistance. The primary benefits of the String Gauge Load Cell are that it has high accuracy and low cost which makes it perfect for this project [19].
However, the strength of the signal given from a load cell is proportional to the force that is applied, in the case for this project which will be measuring fish food servings the force will not be too high and therefore an amplifier will have to be used. The amplifier that will be used is the common HX711, the measurement can be calibrated to quite accurate measurements. Certainly, a precision good enough for fish feedings, even in smaller tanks. Figure 2.8 shows what a common load cell looks like. The HX711 uses the changes in resistor values from the load cell and then converts those signals to electrical signals. Electrical signals which then can be used by any type of microcontroller to read the weight that has been placed on the load cell [19].

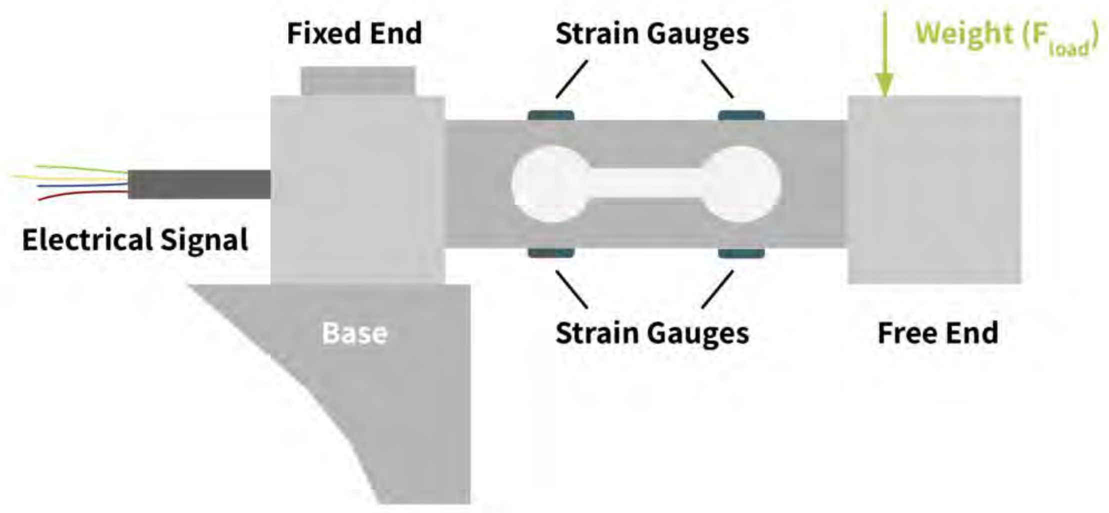


Figure 2. 8. A Strain Gauge Type Load Cell [19].

Chapter 3

Assembling the components

This chapter will be going into all the different types of components this project consists of and how these different components interact with each other.

3.1 Electrical components

The fish feeder is constructed by one load cell, an amplifier, a microcontroller in the form of an Arduino Uno and a DC-motor. The load cell is connected to the amplifier which is in turn connected to the Arduino Uno to receive the data from the load cell, which is used to control the amount of food that has been released into the fish tank. How this is connected can be seen in figure 3.3. The DC-motor is also connected to the Arduino Uno, so that the code on the microcontroller can control when the motor starts and feeds the fish and when it stops. The source of energy for this comes from either a battery pack that is also connected to the controller, or a USB power supply. At the back of the feeder there is an LCD screen and a series of buttons to control and view the data that is being fed to the Arduino. The screen is going to act as the GUI of the feeder, where the changes being made, and the current feeding will be displayed. All the components can be seen in figure 3.1.

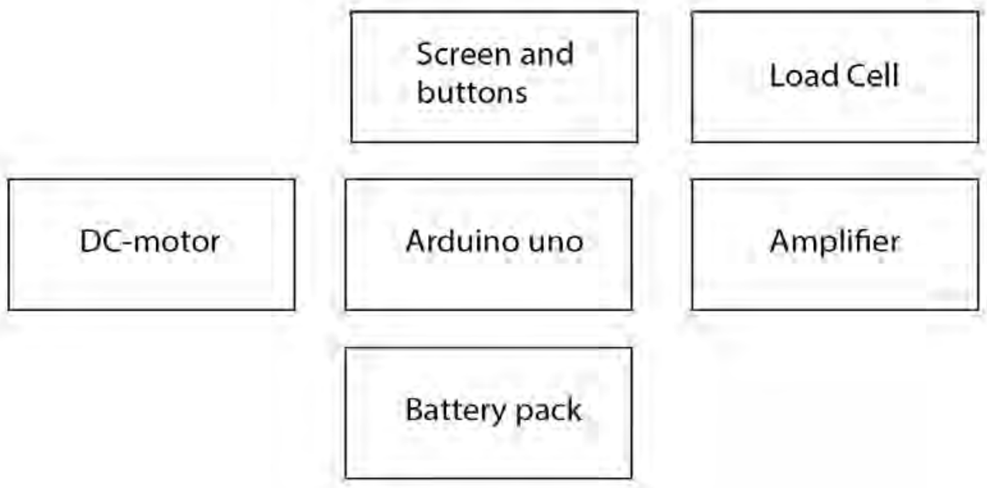


Figure 3. 1. All the major electric components of the fish feeder. Made in Adobe Illustrator [20].

When the feeding process starts, the microcontroller (Arduino Uno) sends a signal to the motor to start rotating, so the food can be sent down into the aquarium. As the feeding proceeds, the load cell sends a signal to the amplifier which then amplifies that signal for greater precision and sends it further to the microcontroller. In the microcontroller the signal is received and checked if the right amount of food has been fed, and if so, it sends a signal to the motor to stop the feeding process.

3.1.1 Load Cell and Amplifier

For the fish feeder to realise it has released food into the aquarium, it needs to be able to measure its own weight on a precise level. This can easily be achieved with a load cell, in particular a strain gauge load cell with an amplifier that amplifies the signal and can give a more precise reading of the food. The load cell and amplifier that was used in this project are 4 regular strain gauge load cells which can be observed in figure 3.2. They are connected to an HX711 amplifier. This connection and the connection from the amplifier to the Arduino Uno can be seen in figure 3.2. There are multiple combinations to hook up the load cells. There is a 1x configuration that uses only one load cell, there is a 2x configuration that uses two load cells and the configuration with 4x load cells that can be viewed in figure 3.2. This project is using the 4x configuration since it appears to be the most stable for measuring smaller doses of fish food and would have the best fit considering the square design of the base.

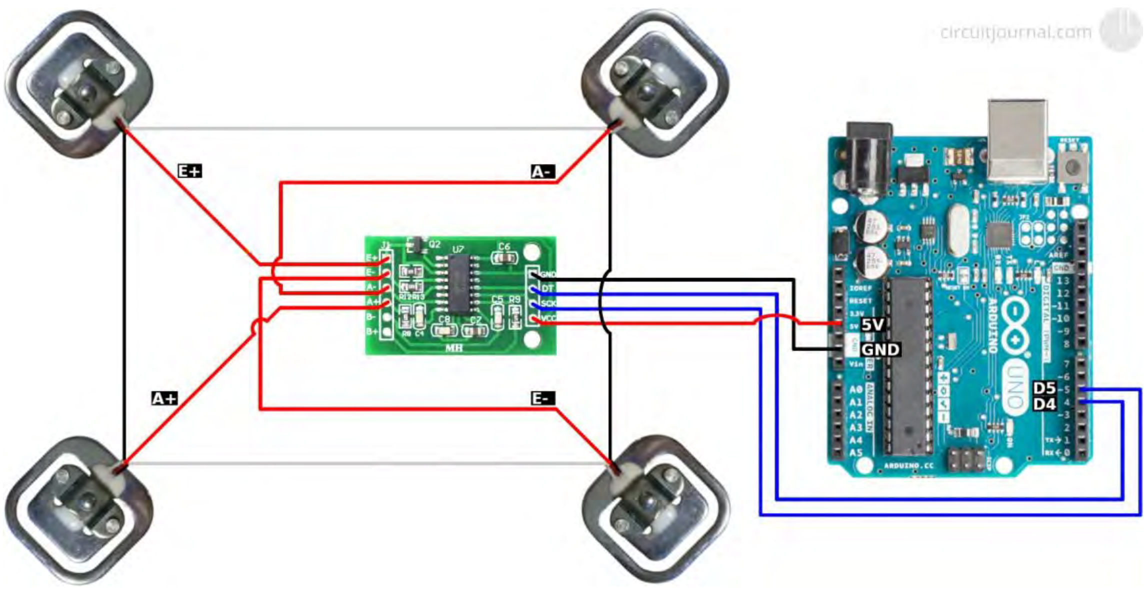


Figure 3. 2. An example of how the connection between the microcontroller, the amplifier and the load cells can look like [21].

3.1.2 Motor

The motor that will be used for this project is a geared down DC motor with an operating voltage of 12V that runs fine from a standard USB output of 5V. Its low current rating of 280mA makes it easy to power alongside an Arduino board, from just a standard USB outlet or a battery pack. The motor alone offers torque up to 2.9 mNm, but with the matching gearbox this number can be greatly increased (at the cost of slower rotation, but this is desirable for more accurate feedings). This motor does the job perfectly for this project since it has both low rotation speed and has high torque at the same time. This is important in case some of the fish food gets stuck in between the rotating spiral and the feeding hole or the housing for the spiral. Should this happen, the motor would have enough strength to crush the food so the spiral can continue to spin. The speed of the motor is ideal for this kind of work, since the low rpm it outputs goes hand in hand with the precise feeding the machine wants to achieve.

3.1.3 Arduino

The microcontroller of choice for this project is an Arduino Uno. The Arduino is the brain of the machine that controls everything from how much food will be distributed during each feeding to how many times a day feeding will occur. Everything in the machine is connected to the Arduino as can be observed in figure 3.3.

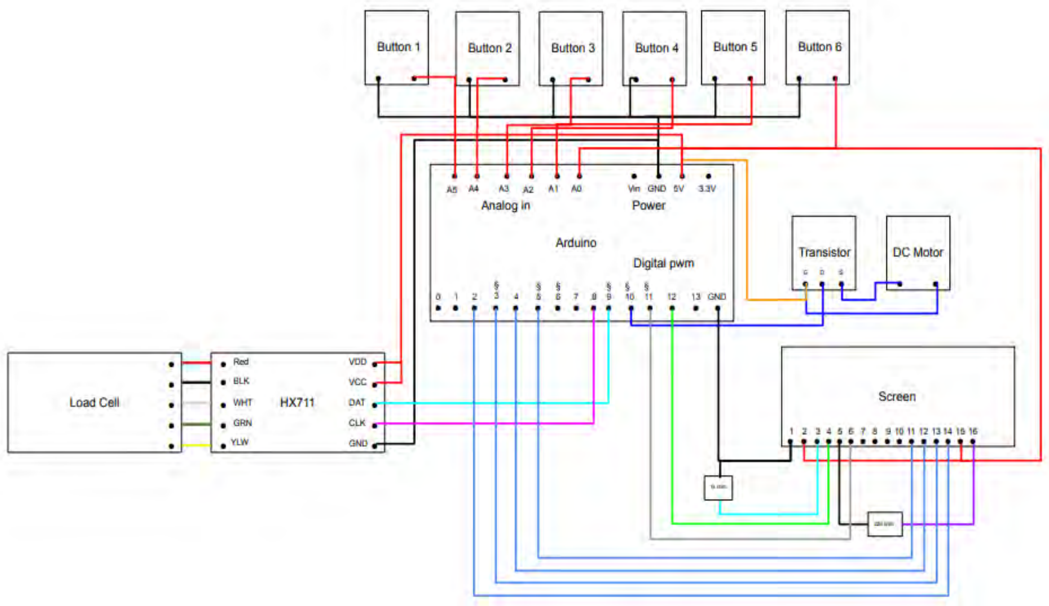


Figure 3. 3. Schematic coupling of electronics, created in Google Drawings [22].

3.1.4 Buttons and Screen

The fish feeder will be equipped with 6 buttons and an on/off switch, which can be observed in figure 3.3. The 6 buttons will enable the user to choose when and how much food to give to the fish. These buttons will be attached directly to the Arduino board and can be used at any time. This means that the amount of food and number of feedings can be changed from day to day if the information is fed to the Arduino through the buttons. For the user to be able to control what changes they make, a screen is also placed just over the buttons at the back of the feeder. The screen that is used is a regular 16 by 2 LCD screen. The information that can be displayed on the screen is how long it is until the next feeding and the current time. It will also display what button to use when wanting to set the amount of food that is going to be fed and the time it is going to be feeding. The connection of the screen to the Arduino can be observed in figure 3.3 and in figure 3.4.

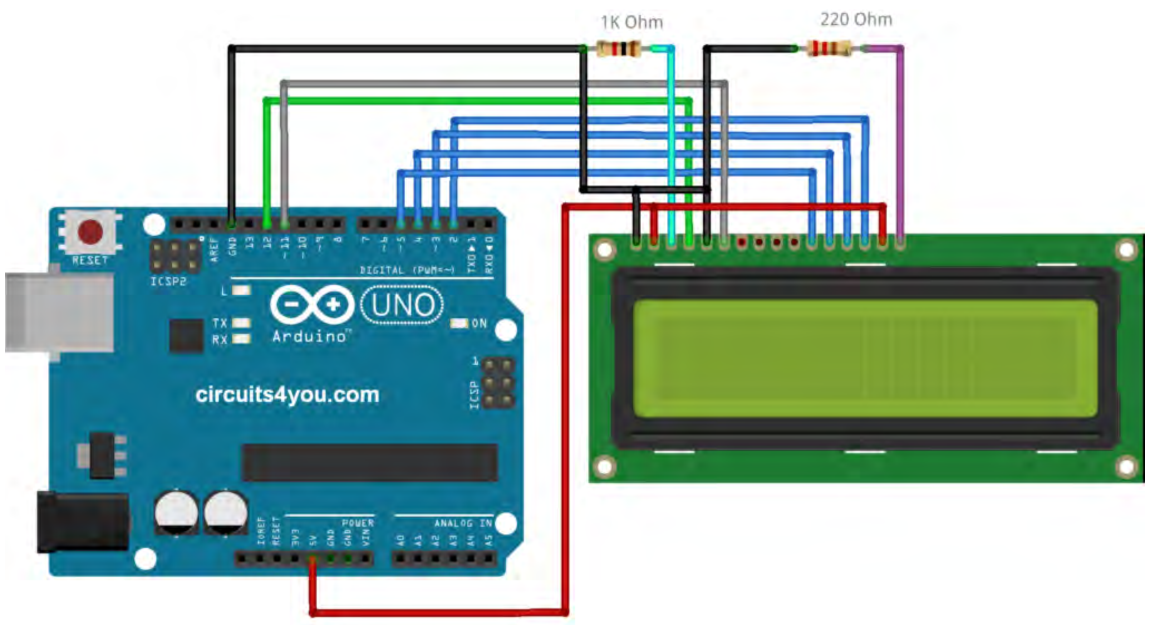


Figure 3. 4. An example of how to connect the screen to the Arduino. Made in Tinkercad [2].

3.2 Construction of the fish feeder

The main body will be made from food grade plastic, for safety reasons. Fish and invertebrates are quite sensitive to certain metals, and some plastics can leach toxins into the food over time. Food grade plastic is however considered safe, as stated by its name.
To ensure only plastic gets in contact with the food, the container for the food is designed to assemble without screws visible on the inside. All components are also designed with the possibility of printing with minimal support material and cleanup required. The entire feeder can be printed on a regular consumer grade 3D printer in a single go, and it could easily be scaled to larger production using a variety of other production methods.

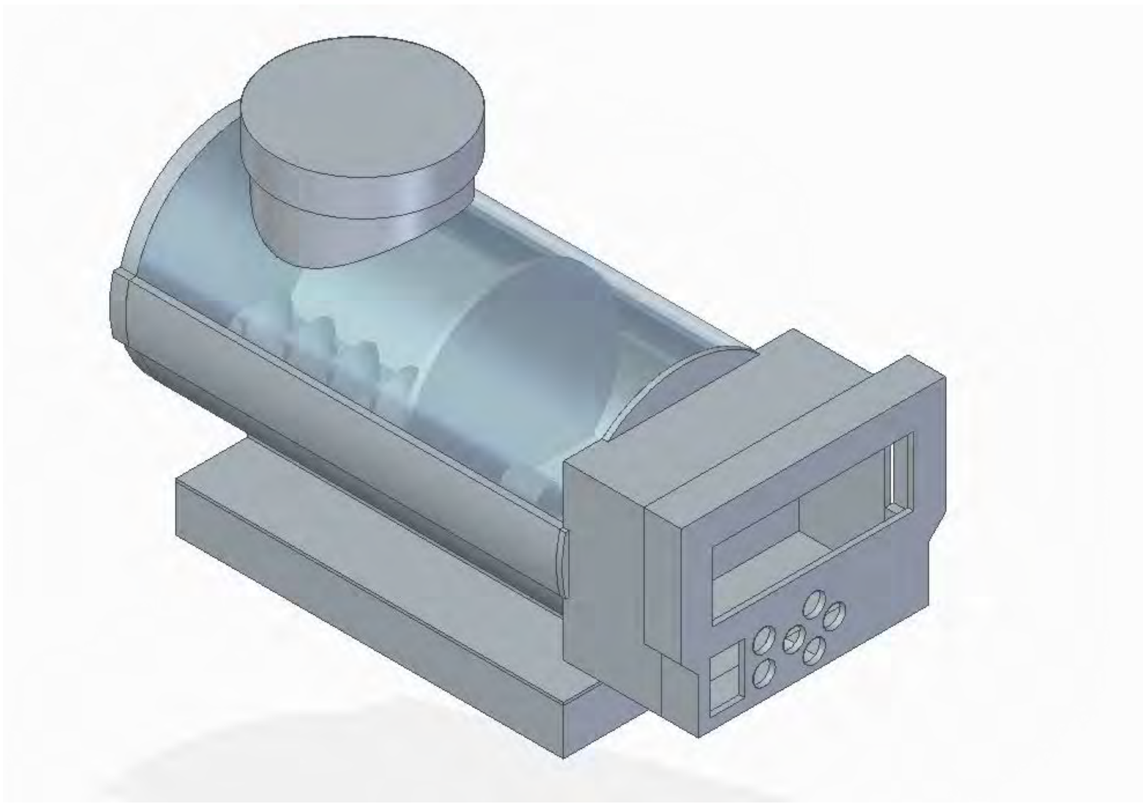


Figure 3. 5. The main body as designed in Solid Edge ST10, assembled view [23].

3.2.1 The spiral

To get the food from the storage in the feeder a rotating spiral will be used. When it is time for feeding the Arduino will start the DC motor which is connected to the spiral. The spiral then starts spinning and due to the sloped design of the spiral it will transport the food from where it is towards the front of the feeder and then into the fish tank. When the motor does not spin the spiral prevents any of the food from getting towards the front of the feeder and therefore preventing any food from getting into the fish tank when it is not supposed to. The spiral is also attached to a ball bearing at the front so that it can rotate freely without friction.

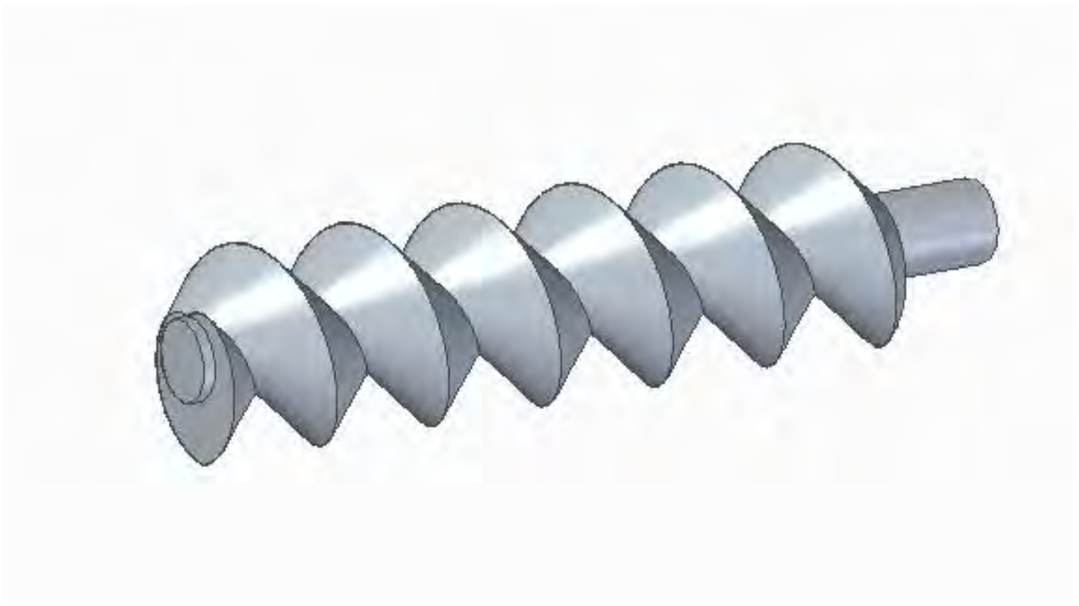


Figure 3. 6. The spiral that pushes the food forward in the feeder. Designed in Solid Edge ST10 [23]

3.2.2 The inside of the feeder

The inside of the feeder consists of two spaces, the one to the left in the figures 3.5 and figure 3.7 is where the food is kept. The space to the right is where the optional battery pack and the DC motor is being stored. To divide the spaces there is a 3D printed food grade plastic disc that prevents the battery pack and the motor from coming in direct contact with the food, thus keeping the food as safe as possible. The disc also has a tiny hole so that the DC motors axis can be connected to the hole in the spiral. The left space is also designed with sloped sides to help the food get to the bottom of the feeder to make sure that it can be transported forward by the rotating spiral. Underneath the sloped sides there is a circular cut-out that the spiral is placed into that further enables the spiral to bring food towards the front of the feeder.

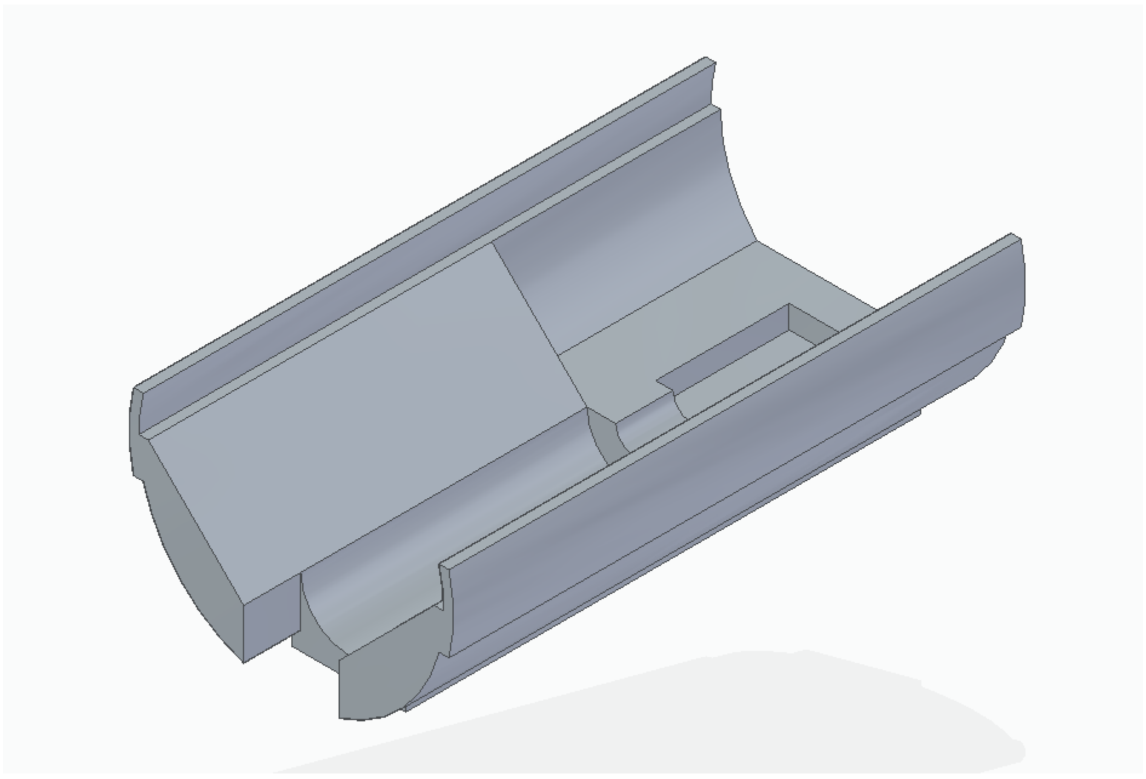


Figure 3. 7. The inside of the feeder, with the sloped sides and space at the back for the motor and battery pack. Designed in Solid Edge ST10 [23].

3.2.3 The top of the feeder

The top of the feeder mainly consists of a roof with a hole in it, where the food is placed into the feeder. The hole also has a removable lid to protect the food in the feeder from outside factors. The hole is designed so that when the food is dropped into the feeder, it is placed on the opposite side from the output of the feeder. This is for safety reasons as the food is supposed to be transported from the rear of the feeder to the front by the rotating spiral, to avoid feeding the fish simultaneously as the feeder is refilled. To achieve this a sloping plastic wall has been mounted to guide the food to the end of the feeder, as can be seen in figure 3.8.

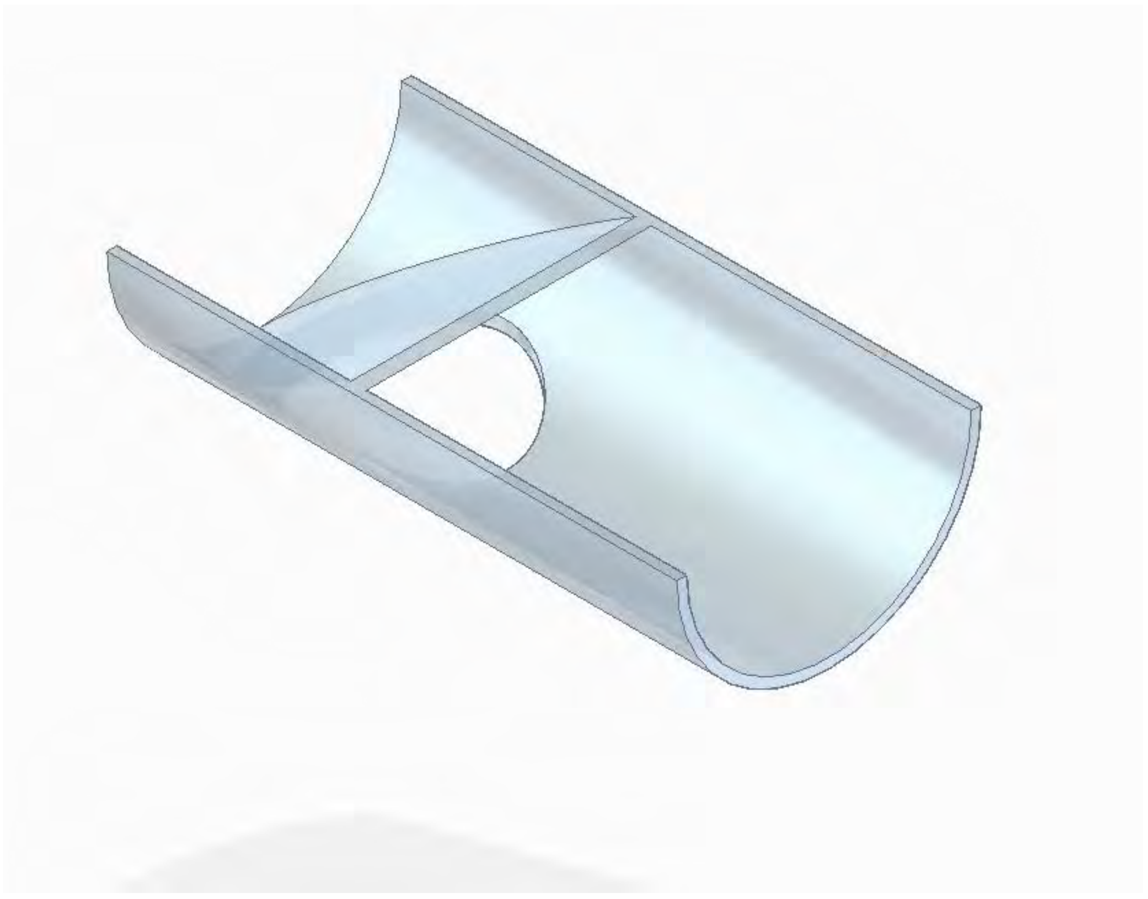


Figure 3. 8. The top of the feeder turned upside down so that the sloped plastic wall easier can be observed. Designed in Solid Edge ST10 [23].

3.2.4 Base of the feeder

The base of the feeder is placed on a slab that houses the load cell which controls the weight of the feeder. It is mounted this way because the weight of the food that is being released from the feeder easily can be measured if the whole construction is mounted on the scale. Another good thing about mounting it atop of the scale is that the total amount of food in the feeder can be easily measured and communicated to the user of the feeder at any given time. This makes managing the food and refilling the food a much easier task, then it would have been without it.

3.2.5 Housing for the electronics

The housing for the electronics is placed on the backside of the feeder to keep all the electronics close to each other to minimize the chance of any kind of disturbance of the signal between the components. The housing contains all the buttons for the fish feeder, the Arduino and the screen for viewing the amount of food and when the food is to be fed to the fish. The housing and the screen are shown in figure 3.9.

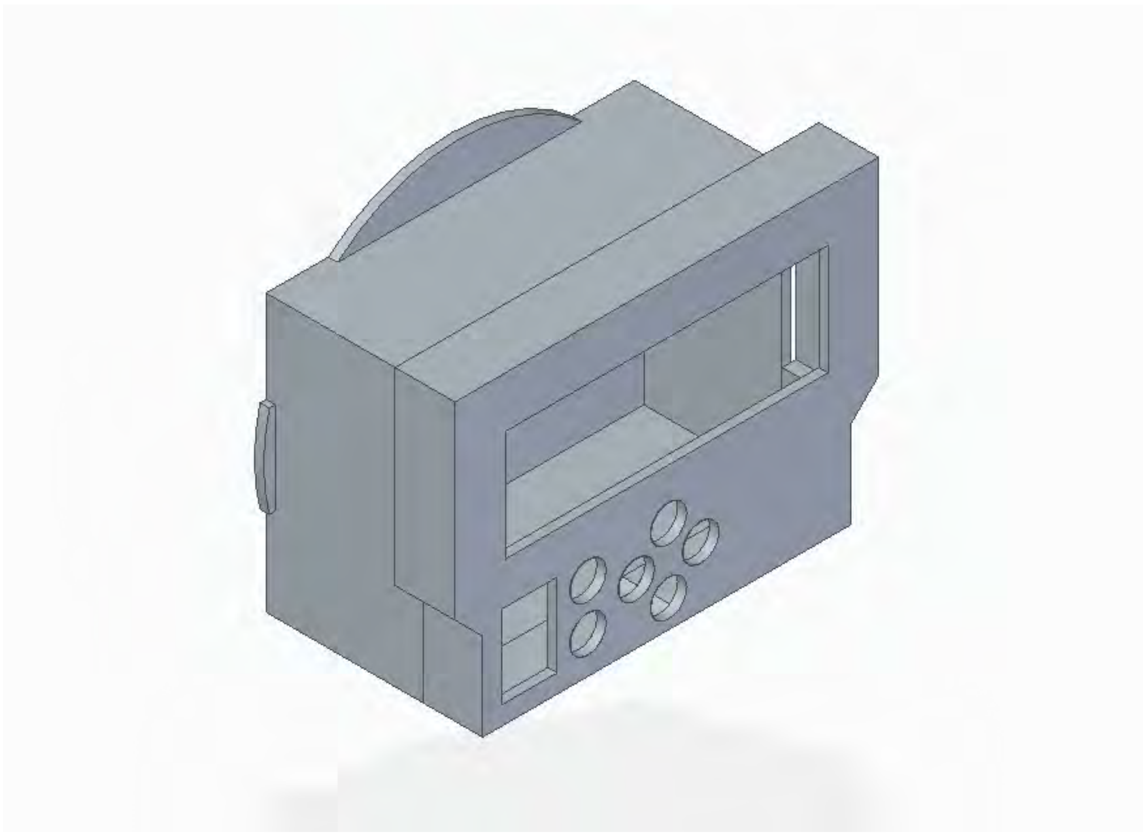


Figure 3. 9. The housing for the screen and electronics**.** Designed in Solid Edge ST10 [23].

3.3 Software

The software is written in C and divided into several functions. Depending on the state read from the analog input pins on the Arduino device, a corresponding function is called. In other words, when a button is pressed the function relating to the button is executed. Pressing the add button runs the add function, the remove button runs the remove function.

Pressing the menu buttons does not simply detect and run a corresponding function, rather it determines the appropriate piece of code to be run based on what has previously been run.

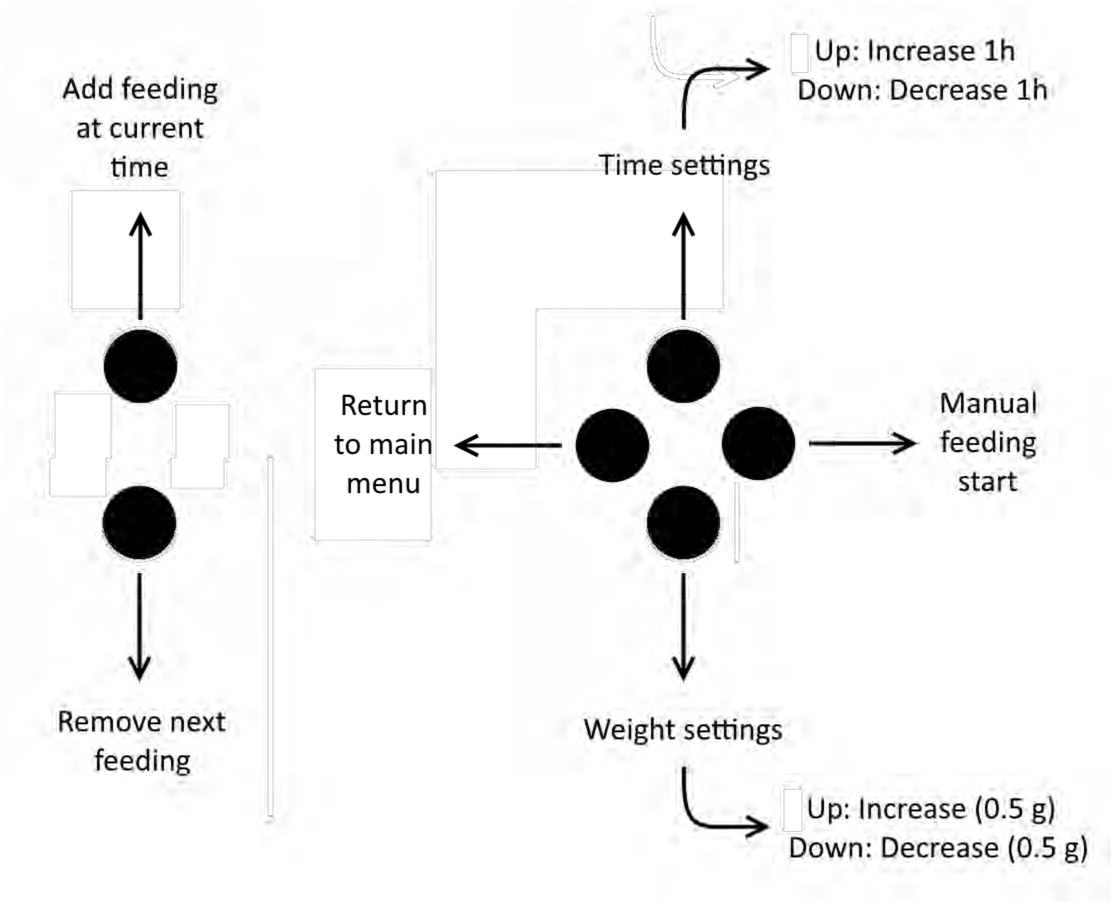


Figure 3. 10. Diagram of each button and corresponding function. Done in paint.net [24].

Chapter 4

This chapter will be discussing the results of the work.

Results

The results will be presented in three different categories.

- How accurate can the food be measured?
- Does the addition of the scale make the feeding more precise or is the difference negligible?
- How much ammonia is produced per gram of food?

4.1 Accuracy of the feeder

Unfortunately, the accuracy of the load cell used is not good enough to handle the dosage of the feeding by itself. It does however provide enough accuracy to provide a failsafe in the form of stopping when overfed.

The loadcell we are using is only providing accuracy to around 0.3 grams. This can be observed in table 4.1.

While this is plenty for a feeding of 1-3g (which is the standard for larger aquariums), it is not accurate enough for smaller tanks where the feedings can be as small as 0.1 grams.
In order to circumvent this, we have opted to scale the feedings based on 'on time' for the motor and use the weight as a security measurement to end the feeding if it exceeds a selected size. This can be seen as an added safety feature for the feeder.

Table 4. 1. Accuracy of the load cells. Average of 10 measurements.

Known weights. [grams]	10	30	50	100
Weights given by our load cell. [grams]	10.1	30.3	50.2	100.3

4.2 Difference in precision with scale

This test was being done when the fish feeder was fully filled since it is the only way for his test to be fair for both kinds of feeding. This is due to the method of using a set amount of time for feeding, which only works when the food is already at the front of the feeder. This is because if the food is at the back of the feeder the rotations that will occur within the given time period will only take the food forward a small way. Thus, not being able to feed it to the fish. So, this comparison is only to see if the scale adds a level of security to the feeder. The results of this comparison are presented in the table 4.2 below.

Table 4. 2. Overview of the amount of food fed with and without the scale.

Preferred amount of fish food [grams]	1.0	2.0	3.0	5.0
Amount of fish food without the scale [grams]	1.4 0.8 1.3 1.2	2.9 2.8 1.4 2.1	3.4 4.0 3.9 2.2	6.0 5.9 5.1 4.5
Amount of fish food with the scale [grams]	1.1 1.0 1.1 0.9	2.1 1.6 2.0 2.1	3.2 2.3 2.8 2.9	4.9 4.9 5.2 4.4

4.3 Amount of ammonia produced per gram of food

To determine a rough guideline of how much ammonia is released when food is decomposed, we filled four separate bowls with 1 liter of distilled water. After this, we added 1, 5, 10 and 20 grams of food to the bowls. This was left to sit at room temperature for 3 days, and then using a liquid test kit we measured the amount of ammonia in each bowl. This does not provide an accurate reading on how much ammonia is created in a live aquatic environment, but it does give a rough guideline of how much excess food is acceptable.

Table 4. 3. Amount of ammonia produced per gram of fish food in a clear water bowl with 1 liter of distilled water.

Amount of fish food [grams]	1.0	5.0	10.0	20.0
Amount of ammonia [ppm]	0.0	0.15	0.3	0.6

Chapter 5

This chapter will be discussing the closing thoughts and discussions for this project.

Conclusion and discussion

Unfortunately, the load cell we used for this project was not as precise as we would have wanted and that is clearly affecting the results in the comparison with only a set amount of time for the rotations of the spiral. But as stated earlier the scale is still good enough to act as a failsafe to keep the feeder from overfeeding the fish. This unfortunate fact can not give a definitive answer to the question of how much the accuracy of the feeding size improves with the addition of the scale to the feeder. Given the precision of the scale it can however be used in larger aquariums without a problem, but as stated above not as reliably in the smaller aquarium. Also as mentioned in the result section of the paper, the scales addition to the automatic feeder acts as a good failsafe. A function that in cooperation with other methods of controlling the amount of food that the fish gets will result in a more precise and safer way of feeding your aquarium fish no matter how big of an aquarium you have.

As for what method is safer for feeding the fish is hard to say with absolute certainty, since the accuracy of the load cells that were used in this project are not as precise as we would have hoped for. What can be said however is that the scale system is more practical then the ‘rotations of the spiral in a set amount of time’ system, since the set amount of time system requires the feeder to be full for it to even have a shot at consistent feedings. That can nevertheless be fixed with a different design to the feeder with separate housings for the spiral and the food. This way the housing for the food does not always have to have food pushed all the way forward after being emptied, it just feeds to the second housing where the spiral is located and then it can always feed without a problem.

We chose to use the DC motor over the servo motor since it ran fine with only the use of power on and power off. We then felt that it was good enough to use for this project. The number of extra components that would have come with the servo was also a factor in the decision. Since it would have been unnecessary with that number of components, when the work could get done by using a DC motor.

Chapter 6

This chapter will be discussing the future of this project, what can be improved upon and what can be added to it for a better user experience.

Future Work

We as developers of this project see a lot of potential for future additions and revisions to the automatic feeder. A load cell with greater accuracy is a simple and desirable improvement, as well as a more insulated motor. While hard to prove, the vibrations from the motor are affecting the accuracy on the weight readings. By isolating the motor from the load cells more than we already have, we could achieve more accurate readings during feeding. This could be done by enlarging the motor mount and adding more rubber between the motor and the mount. More rubber between the housing and bottom pad would also achieve the same result, albeit changing the overall appearance of the feeder unless aesthetically redesigned.

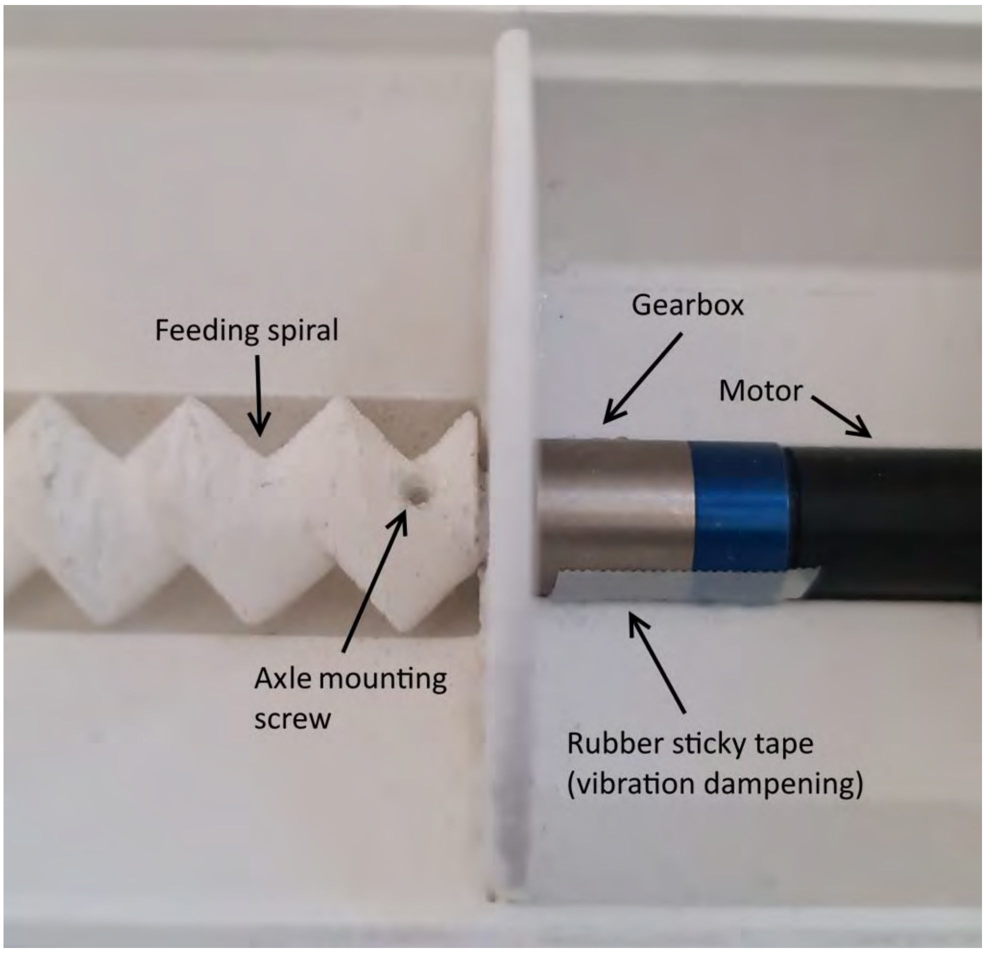


Figure 6. 1. Motor mount and its attachment to the spiral.

Another possible upgrade would be the usage of a touchscreen for menu options, instead of the current mechanical button controller. This gives the ability to add text for each button and corresponding function, although it could possibly prove problematic in the humid environment an aquarium creates. This problem could be nullified using a waterproof, or at the very least water resistant, screen.

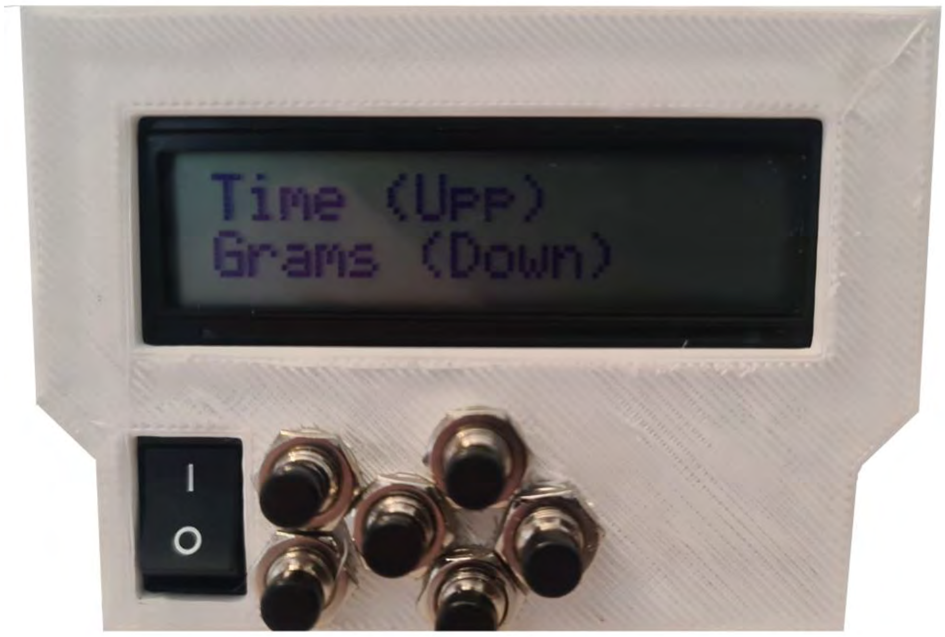


Figure 6. 2. User control and display.

There is also a potential for the feeder, to in co-occurrence to the physical GUI, have a mobile application that connects with the feeder via Bluetooth or Wi-Fi, thus giving the user the option to control the feeder from another place than at home. This is a reasonable addition to the product since a big advantage of having an automatic feeder is that one is not bound to the house to feed one's fishes. Also being able to adjust the feedings from another place in the world is a real quality of life improvement to this product.

One way to take the product one step further is to develop a complementary product that essentially is a sensor for detecting the amount of ammonia in the aquarium. This combined with the feeder will give the ultimate fish feeding experience. The sensor will continuously give the feeder information about the levels of ammonia in the aquarium and the feeder can from there adjust the amount of food it feeds to the fishes. Thus, giving the owner of the aquarium fish complete confidence that the fishes are getting the proper amount of food and not being exposed to any dangerous amounts of ammonia. While the accuracy our loadcells achieved were not as good as we had hoped, it was good enough to ensure feeding sizes within the safe limits of ammonia as tested in table 4.3.

Bibliography

[1]. Arduino, *Arduino IDE*. URL: https://www.arduino.cc (20-05-2021)

[2]. Autodesk, *Tinkercad*. URL: https://www.tinkercad.com (20-05-2021)

[3]. Sergeant, C (2014). *The management of ammonia levels in an aquaculture environment*. Available: URL: https://www.pollutionsolutions-online.com/article/water-wastewater/17/cancer-research-uk/the-management-of-ammonia-levels-in-an-aquaculture-environment/1557#:~:text=By%20aerating%20a%20pond%2C%20dissolved,increase%20the%20ammonia%20levels%20further (20-05-2021)

[4]. Stuart M. Levit, MS, JD (2010). *A Literature Review of Effects of Ammonia on Fish*. Available: URL: https://www.conservationgateway.org/ConservationByGeography/NorthAmerica/UnitedStates/alaska/sw/cpa/Documents/L2010ALR122010.pdf (20-05-2021)

[5]. Hill, N (2021). *The Age of Maturity*. Available: URL: https://www.practicalfishkeeping.co.uk/features/the-age-of-maturity/ (20-05-2021)

[6]. Y. Atoum, S. Srivastava and X. Liu, *Automatic Feeding Control for Dense Aquaculture Fish Tanks*, in IEEE Signal Processing Letters, vol. 22, no. 8, pp. 1089-1093, Aug. 2015, doi: 10.1109/LSP.2014.2385794.

[7]. R. Cole, D. McDowell, M. Kirwan (2003). *Food Packing Technology*. Blackwell Publishing Ltd.

[8]. Jalani, J (2015). *A Comparative Study on the Position Control Method of DC Servo Motor with Position Feedback by using Arduino*. Available: URL: https://www.researchgate.net/figure/The-Block-Diagram-of-DC-Servo-motor-with-Closed-Loop-voltage-Feedback_fig6_304435252 (20-05-2021)

[9]. D. Ibrahim. *ARM-Based microcontroller projects using MBED*. In: 2019, pp. 261-279, ISBN: 9780081029695, DOI: URL: https://doi.org/10.1016/B978-0-08-102969-5.00009-4 (20-05-2021)

[10]. *DC motor*. Available: URL: https://www.electronicwings.com/sensors-modules/dc-motor (20-05-2021)

[11]. J. Linares-Flores and H. Sira-Ramirez, *DC motor velocity control through a DC-to-DC power converter*, 2004 43rd IEEE Conference on Decision and Control (CDC) (IEEE Cat. No.04CH37601), 2004, pp. 5297-5302 Vol.5, doi: 10.1109/CDC.2004.1429649.

[12]. H. Zhou, *DC Servo Motor PID Control in Mobile Robots with Embedded DSP*, 2008 International Conference on Intelligent Computation Technology and Automation (ICICTA), 2008, pp. 332-336, doi: 10.1109/ICICTA.2008.426.

[13]. *How Servo Motors Work and How to Control Servos Using Arduino* URL: https://howtomechatronics.com/how-it-works/how-servo-motors-work-how-to-control-servos-using-arduino/ (20-05-2021)

[14]. K. M. Cho, W. S. Oh, Y. T. Kim and H. J. Kim, *A New Switching Strategy for Pulse Width Modulation (PWM) Power Converters*, in IEEE Transactions on Industrial Electronics, vol. 54, no. 1, pp. 330-337, Feb. 2007, doi: 10.1109/TIE.2006.888793.

[15]. *Pulse Width Modulation*. Available: URL: https://www.electronics-tutorials.ws/blog/pulse-width-modulation.html (20-05-2021)

[16]. *Basics of Servo Motor and Its Types*. Available: URL: https://project-engi.blogspot.com/2019/02/basics-of-servo-motor-and-its-types.html (20-05-2021)

[17]. Chou, Ortega and Borriello, *Synthesis of the hardware/software interface in microcontroller-based systems*, 1992 IEEE/ACM International Conference on Computer-Aided Design, 1992, pp. 488-495, doi: 10.1109/ICCAD.1992.279322.

[18]. Kumar Reddy, K., Jagadeesh, P., Venkatramana Reddy, S. *Traffic Signals Generation with Bicolor LEDS using PIC 18F Series Microcontroller*. Available: URL: https://www.researchgate.net/figure/Block-diagram-of-Microcontroller-with-support-devices_fig5_216663454 (20-05-2021)

[19]. *An Overview of Load Cells*. Available: URL: https://tacunasystems.com/knowledge-base/an-overview-of-load-cells/ (20-05-2021)

[20]. Adobe, *Adobe illustrator*. Available: URL: https://www.adobe.com/products/illustrator.html (20-05-2021)

[21]. Luuk, I. *50kg Load Cells with HX711 and Arduino. 4x, 2x, 1x Diagrams*. Available: URL: https://circuitjournal.com/50kg-load-cells-with-HX711 (20-05-2021)

[22]. Google. *Google drawings*. Available: URL: https://docs.google.com/drawings (20-05-2021)

[23]. Siemens AG, *Solid Edge ST10*. Available: URL: https://solidedge.siemens.com/en/ (20-05-2021)

[24]. Paint.net. *Paint.NET*. Available: URL: https://www.getpaint.net (20-05-2021)

Appendix

A. Arduino Code

```
//Created by Elias Griffith and Jacob Iljans, as part of Mechatronics KEX work.
//Used to control Automatic Fish Feeder. Reads loadcells, provides GUI, and handles feedings.
//2021-05-01, KTH, MF133X

//Variable and library definitions
int feeds[24];
long feedstime[24];
int done = 0;
float grams = 0;
float gramsize = 1;

int addbtn = 0;
int rembtn = 1;
int LEFTbtn = 11;
int DOWNbtn = 10;
int UPPbtn = 4;
int RIGHTbtn = 5;
int motor = 10;

int LR = 0;
int UD = 0;

int menu = 1;

#include <TimeLib.h>

#include "HX711.h"

#include <LiquidCrystal.h>

#define cf 125000 //calibration of scale here

HX711 scale;
const int rs = 12,
  en = 11,
  d4 = 5,
  d5 = 4,
  d6 = 3,
  d7 = 2;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

//---------------------------------------------------------------

void setup() //Power-on defaults
{
  pinMode(motor, OUTPUT);
  pinMode(addbtn, INPUT); //add button
  pinMode(rembtn, INPUT); //remove button
  pinMode(LEFTbtn, INPUT); //Button Left
  pinMode(DOWNbtn, INPUT); //Button Down
  pinMode(UPPbtn, INPUT); //Button Upp
  pinMode(RIGHTbtn, INPUT); //Button Right
  Serial.begin(9600);
  lcd.begin(16, 2);
  scale.begin(9, 8); //9 = data, 8 = clk
  scale.set_scale(cf);
  scale.tare();

  lcd.print("Time (Upp)"); //testing
  lcd.setCursor(0, 1);
  lcd.print("Grams (Down)");
}

void loop() //Main menu reading loop
{
  analogWrite(motor, 255); //make sure motor is ALWAYS off, unless feeding
  if (analogRead(addbtn) == 0) { //add
    add();
  }
  if (analogRead(rembtn) == 0) { //Removal
    feeds[hour()] = 0;
    Serial.println(feeds[hour()]);
  }

  if (analogRead(DOWNbtn) == 0) { //Grams
    text("Hold left ", "to return");
    delay(2000);
    menuDOWN();
  }
  if (analogRead(UPPbtn) == 0) { //Time
    text("Hold left ", "to return");
    delay(2000);
    menuUPP();
  }
  if (analogRead(RIGHTbtn) == 0) { //Manual Feed
    text("Manual 2nd feed", "(Current hour)");
    feed();
  }

  if (menu == 0) { //Main menu LCD
    int temp = 999;
    int feed = 0;
    for (int i = 0; i < 24; i++) {
      if (hour() - i < temp && feeds[i] > 0) { //Find nearest feed
        temp = hour() - feeds[i];
        feed = i;
      }
    }
    if (feeds[feed] != 0) {
      text("Current hour: " + String(hour()), "Next feed: " + String(feed));
    } else {
      text("Current hour: " + String(hour()), "No feedings");
    }
  }

  if (done == 0 && minute() < 30) { //turned new hour. Feed?
    if (feeds[hour()] == 1) {
      Serial.println("Feeding time!");
      done = 1;
      feed();
    }

  } else {
    if (done == 1 && minute() >= 30) { //reset done
      done = 0;
    }
  }

  delay(100);
}

void weight() //Read loadcell
{
  Serial.println("weight");
  grams = scale.get_units() * 453.592; //Set as grams
  Serial.println(grams);
  delay(100);
  return;
}

void feed() //Power motor, check failsafe weight
{
  Serial.println("feed");
  weight();
  float prefeed = grams;
  analogWrite(motor, 50); //Motor startup
  int curhour = hour();
  long feedstimeHolder = feedstime[curhour];
  while (feedstimeHolder > 0) { //Feed wait
    weight();
    if (prefeed - grams > gramsize) {
      break;
    }
    feedstimeHolder = feedstimeHolder - 100;
    delay(100);
  }
  analogWrite(motor, 255); //Motor shutdown
}

void add() //Adding new feeding at current selected hour
{
  Serial.println("add");
  int curhour = hour();
  feeds[curhour] = 1;
  weight();
  long feedtime = -300; //Accounts for 300 ms delay on button release
  analogWrite(motor, 50);
  while (analogRead(motor) != 255) { //Read holddown time
    if (analogRead(addbtn) != 0) { //First release check (accidental stop protection)
      Serial.println("first");
      delay(300);
      feedtime = feedtime + 300;
      if (analogRead(addbtn) != 0) { //300ms delay confirmed, end feeding
        Serial.println("second");
        break;
      }
    }
    weight();
    delay(100);
    feedtime = feedtime + 100; //Add 100ms for delay compensation
  }
  analogWrite(motor, 255); //Motor off
  done = 1; //Don't double feed
  feedstime[curhour] = feedtime; //add amount fed
  Serial.println(feeds[curhour]);
  Serial.println(feedstime[curhour]);
  return;

}

void menuUPP() { //time settings
  menu = 1;
  text("Time: ", String(hour()));
  int selecthour = hour();
  while (menu != 0) {
    if (analogRead(UPPbtn) == 0) {

      delay(500);
      if (analogRead(UPPbtn) == 0) {
        if (selecthour < 23) {
          selecthour++;
        } else {
          selecthour = 0;
        }
        text("Time: ", String(selecthour));
      }
    }
    if (analogRead(DOWNbtn) == 0) {
      delay(500);
      if (analogRead(DOWNbtn) == 0) {

        if (selecthour > 0) {
          selecthour--;
        } else {
          selecthour = 23;
        }
        text("Time: ", String(selecthour));
      }
    }
    if (analogRead(LEFTbtn) == 0) { //First check settings exit
      delay(1000);
      if (analogRead(LEFTbtn) == 0) { //Second check settings exit
        menu = 0;
        break;
      }
    }
  }
  setTime(selecthour, 00, 00, 00, 00, 0000);
  return;
}

void menuDOWN() { //grams settings
```

```
  text("Feeding: ", String(gramsize) + " g");
  menu = 1;
  while (menu != 0) {
    if (analogRead(LEFTbtn) == 0) { //First check settings exit
      delay(1000);
      if (analogRead(LEFTbtn) == 0) { //Second check settings exit
        menu = 0;
        break;
      }
    }

    if (analogRead(UPPbtn) == 0) {
      delay(500);
      if (analogRead(UPPbtn) == 0) {
        if (gramsize < 10) {
          gramsize = gramsize + 0.5;
        } else {
          gramsize = 0;
        }
        text("Feeding: ", String(gramsize) + " g");
      }
    }
    if (analogRead(DOWNbtn) == 0) {
      delay(500);
      if (analogRead(DOWNbtn) == 0) {
        if (gramsize > 0) {
          gramsize = gramsize - 0.5;
        } else {
          gramsize = 10;
        }
        text("Feeding: ", String(gramsize) + " g");
      }
    }
  }
  return;
}

void text(String s, String ss) { //LCD text
  lcd.clear(); //Remove old
  lcd.print(s); //First row
  lcd.setCursor(0, 1); //Focus second row
  lcd.print(ss); //Second row
  return;
}
```

B. Acumen simulation code

```
//Created by Elias Griffith and Jacob Iljans, as part of Mechatronics KEX work.
//Used to generate a model and animation of an Automatic Fish Feeder.
//2021-24-03, KTH, MF133X

model Main(simulator) =  //START OF CODE
initially
 feed = create feeder((0,0,0),white), //create feeder with base color white
 x1=0, x1'=0, x1''=0
always //rotate
 if x1<10 //if x1 < 10
  then x1'' = -(x1'-1) //set new x1''
  else if x1'>0 //if x1 !< 10 and x1' > 0
        then x1'' = -1 //set new x1''
        else x1'' = 0, feed.pos = (0,x1,0) //set new x1'' and change feed.pos (if x1 !< 10 and x1' !> 0)

model feeder(pos,col) = //generate feeder model
initially
 _3D = (),_Plot=()
always
_3D = (
            //Below is creation of the model and animation of the model
            Cylinder center = (0,0,0) size = (8,2) color = white rotation=(0,0,0) //main body

            Cylinder center=(0,1,2) size = (2,1.5) color = blue rotation=(pi/2,0,0) //lid

            Box center = (0,-0.5,-2) size = (6.5,1,2) color = black rotation = (pi/2,0,pi/2) //base, bottom plate

            Box center = (0,3.3,-2.7) size = (2,0.3,1.5) color = white rotation = (pi/2,0.7,pi/2) //ramp for food

            Box center = (0,3.7,-2) size = (1,0.5,0.5) color = white rotation = (0,0,0) //mouth of feeder

            Cylinder center = (0.7,2.5,-3) size = (1.1,0.2) color = blue rotation = (pi/2,0,0) //leg 1
            Cylinder center = (-0.7,2.5,-3) size = (1.1,0.2) color = blue rotation = (pi/2,0,0)//leg 2
            Cylinder center = (0.7,-3.5,-3) size = (1.1,0.2) color = blue rotation = (pi/2,0,0)//leg 3
            Cylinder center = (-0.7,-3.5,-3) size = (1.1,0.2) color = blue rotation = (pi/2,0,0) //leg 4

            Cylinder center = (5,0,0) size = (6,0.2) color = white rotation=pos+(0,0,0) //feeder body
                  //Below is feeder rotation settings
            Cylinder center = (5,-3,0) size = (0.2,0.6) color = white rotation=pos+(0.4,0,0) //feed spin
            Cylinder center = (5,-2,0) size = (0.2,0.6) color = white rotation=pos+(0.4,0,0) //feed spin
            Cylinder center = (5,-1,0) size = (0.2,0.6) color = white rotation=pos+(0.4,0,0) //feed spin
            Cylinder center = (5,0,0) size = (0.2,0.6) color = white rotation=pos+(0.4,0,0) //feed spin
            Cylinder center = (5,1,0) size = (0.2,0.6) color = white rotation=pos+(0.4,0,0) //feed spin
            Cylinder center = (5,2,0) size = (0.2,0.6) color = white rotation=pos+(0.4,0,0) //feed spin
            Cylinder center = (5,3,0) size = (0.2,0.6) color = white rotation=pos+(0.4,0,0) //feed spin
            Cylinder center = (5,-2.5,0) size = (0.2,0.6) color = white rotation=pos+(0.4,0,0) //feed spin
            Cylinder center = (5,-1.5,0) size = (0.2,0.6) color = white rotation=pos+(0.4,0,0) //feed spin
            Cylinder center = (5,-0.5,0) size = (0.2,0.6) color = white rotation=pos+(0.4,0,0) //feed spin
            Cylinder center = (5,0.5,0) size = (0.2,0.6) color = white rotation=pos+(0.4,0,0) //feed spin
            Cylinder center = (5,1.5,0) size = (0.2,0.6) color = white rotation=pos+(0.4,0,0) //feed spin
            Cylinder center = (5,2.5,0) size = (0.2,0.6) color = white rotation = pos+(0.4,0,0)) //feed spin
```

C. Photos of the final product

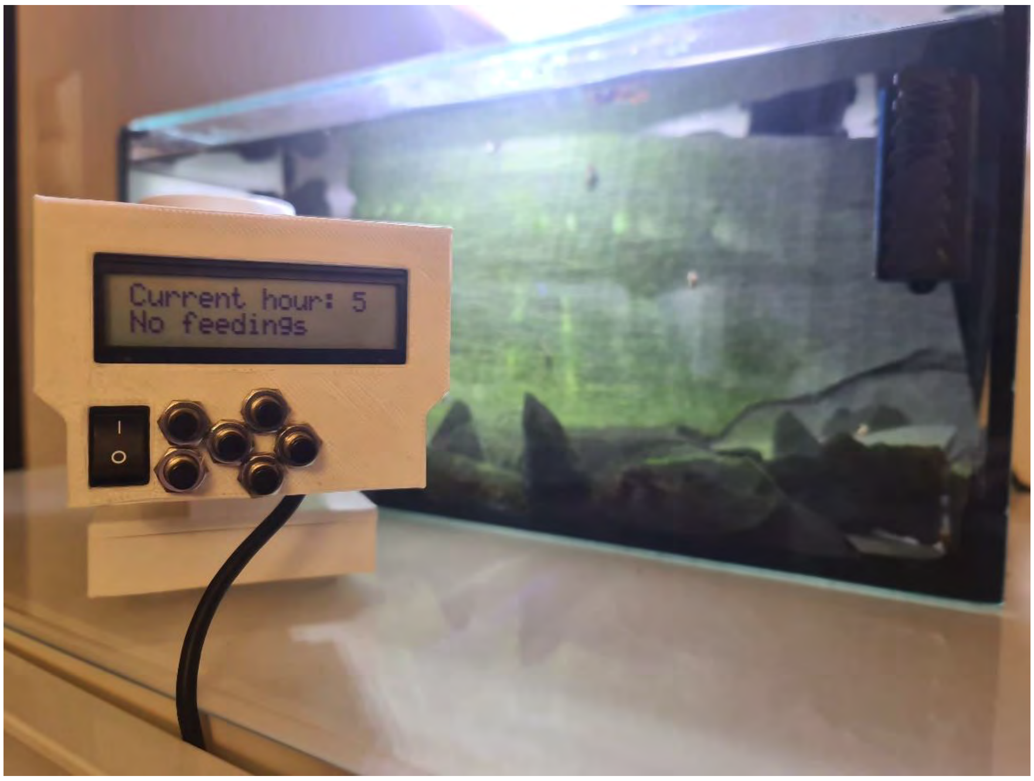


Figure B. 1. Finished product from the back.

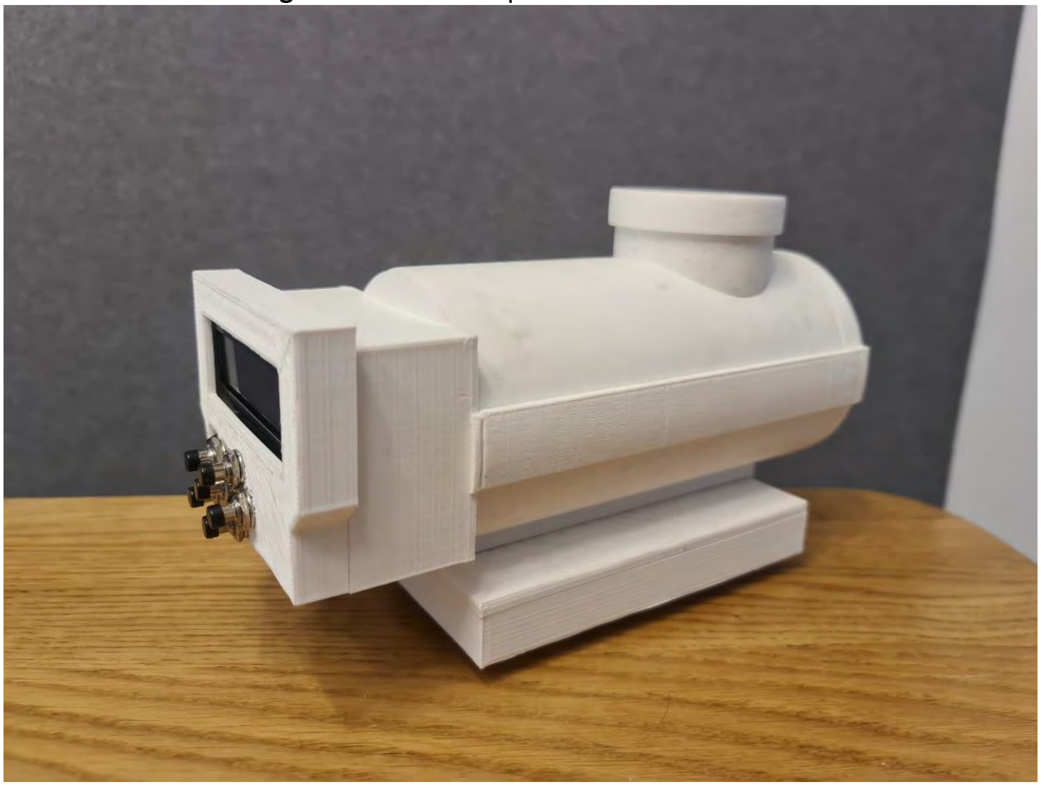


Figure B. 2. Finished product from side.

TRITA -ITM-EX 2021:21