Bachelor Thesis

Master's Programme in Computer Science, 300 credits

Digitalising The Smith Machine

Bachelor Thesis, 15 credits

Halmstad 2021-06-06

Sabor Amini, Rana Faheem Iftikhar

HALMSTAD UNIVERSITY

Abstract

Health and fitness sector is becoming increasingly popular, and technology is being integrated into the fitness industry. This project proposed to digitalise the smith machine by creating a system to generate force through an electric motor to replace free weights. Because the exercise machine no longer needs to be loaded and unloaded with free weights, digitalisation of the smith machine will allow several users to utilise the same training machine. It will also be possible to track users' training information, such as weight lifted and the number of sets and reps performed during a workout.
In this thesis, an Electrical speed controller (ESC) called VESC was used to control the BLDC motor. Experiments were conducted to find a method to control the torque. The result was producing a constant torque with a margin of error of 1-1.5 kilograms. Furthermore, two algorithms were developed where the resistance varies during the path of motion, which indicates that it is possible to develop the system to individualise resistance to improve training performance and rehabilitation. The conclusion that could be drawn is that this system can be used to replace free weights with an electric motor.

Sammanfattning

Hälso- och fitness sektorn blir allt populärare och tekniken integreras alltmer i fitnessbranschen. Detta projekt syftar på att digitalisera smithmaskin genom att skapa ett system för att generera kraft genom en elmotor för att ersätta fria vikter. Genom att ersätta lösa vikter med en elmotor så slipper användaren hantera lösa vikter vilket också underlättar för flera användaren att använda samma träningsmaskin. Det kommer också vara möjligt att spåra användarnas träningsinformation, såsom vikt, antalet set och repetitioner som utförts under ett träningspass.

I detta projekt användes en motorstyrenhet som heter VESC för att styra en BLDC-motor. Flertalet experiment utfördes för att hitta en metod för att kontrollera vridmomentet. Detta resulterade till att ett konstant vridmoment med en felmarginal på 1-1.5 kilogram uppnåddes. Dessutom utvecklades två algoritmer där motståndet varierar dynamiskt, vilket indikerar att det är möjligt att i framtiden utveckla systemet för att individualisera motstånd och därmed förbättra resultatet i träning och rehabilitering. Slutsatsen som kan dras är att detta system kan användas för att ersätta fria vikter med en elmotor.

Table of Contents

1. Introduction

Technology is developing rapidly in the fitness industry and is already present and is making its way into all sports domains to assist users in improving their fitness exercises [1].

Much research has been conducted to implement a control system for exercise machines [2-5]. The main components used in these machines are microcontrollers and motors.

There are several exercise machines already on the market. One of these machines is the 1080 Quantum Syncro, a smith machine that generates force through an AC servo motor [6]. Tonal is another exercise machine with an implemented control system to produce force dynamically [7].

One machine that is used for exercising is the smith machine, as seen in Figure 1. The smith machine consists of a barbell connected to a vertical track that can only move vertically. The user has to unlock the hooks and load the machine with free weights to start the exercise. When the user is finished with the exercise, the user has to unload the weights, place the barbell back on the track and lock them on the pegs. [8].

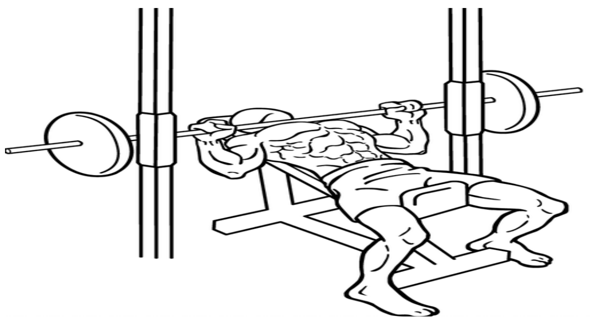


Figure 1:Smith machine [9]

This project proposes digitalising the smith machine by creating a system to generate force through an electric motor to replace free weights. In this project, it was a requirement from the task giver to use an electrical speed controller called VESC. The main difference between this project and existing exercise machines is the electrical speed controller used to control the motor.

This project also proposes to provide a graphical user interface (GUI) where the user can choose weight and store training information such as weight lifted and the number of sets and reps performed during a workout.

By replacing free weights with a motor, the user does not need to load and unload the machine with free weights. Instead, the user can, through the graphical user interface, choose a weight, this will facilitate multiple users to use the same training machine because loading and unloading weight can be done quickly using the graphical user interface. Accidents such as dropping the weight plates may be eliminated by replacing free weights with a motor.

1.1 Purpose & Objectives

This thesis aims to digitalise the smith machine by creating a system to replace free weights with a motor and create a graphical user interface to choose weight and store training information.
It will be attempted to implement algorithms that vary the resistance during the path of motion.

Objectives:

- Create a system to replace free weights with an electric motor.
- Create a constant force during the eccentric (when the muscle lengthens) and concentric (when the muscle shortens) phases.
- Create a graphical user interface for selecting weight.
- Create a storage method to store user training information.

1.2 Engineering questions

- How to control the motor?
- How to create a constant force during the eccentric and concentric phases?
- How to store user training information?
- How is VESC working?
- How to enable communication between different components of the system?

1.3 Scope

In this thesis, there were requirements from the task giver to use an electrical speed controller (ESC) called VESC, a brushless direct current (BLDC) motor, and to implement the software within a Raspberry Pi. The BLDC motor provided by the task giver is capable of producing a maximum weight of 15 kilograms. A smith machine will not be built, considering our educational background as computer engineering students.

1.4 Requirements

In this project, the requirements were given by the task giver. One of the requirements in this project was to create a constant weight during the eccentric and concentric phases. Another requirement was to create a graphical user interface to choose the weight and save training data.

2. Theoretical Background

This chapter explains the theory behind the different parts of the project. It starts with a description of how an electric motor works with an emphasis on BLDC motors. It also includes the mathematical background to control such a motor. The VESC and Vesc Tool parts explain how VESC and Vesc Tool operate and control the motor. The Storage information part describes different database management systems (DBMS) that are available for storing data. The Graphical user interface (GUI) part describes different programming languages that can be used to develop a GUI. Finally, a description of related work is presented.

2.1 Motor and motor theory

2.1.1 Brushless DC Motor

It is necessary to understand how the motor works in order to control it and provide constant torque. A description of the BLDC motor and its equations will be presented in the text below.
Brushless DC motors are widely used in industrial applications. A BLDC motor consists of two parts, a rotor, and a stator. The rotor is a permanent magnet, and the stator consists of several coils. Applying current to the coil will turn it into an electromagnet. As with all other magnets, the electromagnet will generate a magnetic field. The direction of the magnetic field in the electromagnet depends upon which direction the current is going. The generated magnetic field from the coils will attract the permanent magnet. This causes the rotor to rotate. By supplying the right coil at the right moment with current, the motor will continuously rotate. To achieve rotation this way, an electrical speed controller is required [10].

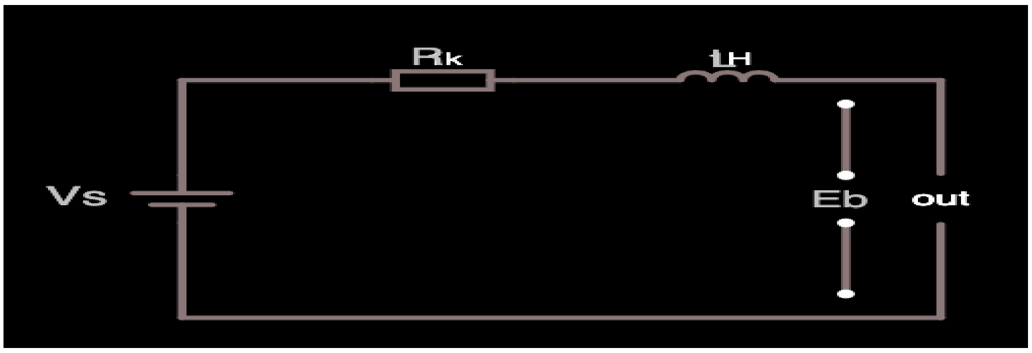


Figure 2: DC motor circuit. Vs. is the voltage in. Rk is the motor resistance, Lh is the inductance, Eb is the back EMF voltage.

Using Kirchhoff's law applied on the circuit shown in Figure 2, Equation 1 was derived.
$E = U - RI(t) + L \quad (1)$

E is back EMF voltage, U[V] is voltage in, R[ohm] is resistance, I(t)[A] is current in the phase windings varying in time, and L[H] is inductance [11].

*A BLD*C motor is supplied with a direct current, which means the current in the windings is constant, and I(t) can be written as I. The inductance L in a BLDC motor is tiny so that it can be neglected [11]. Then Equation 1 can be re-written as Equation 2.

$$E = U - RI \qquad (2)$$

In a BLDC motor, a back EMF voltage is generated when the motor starts to rotate. Back EMF occurs according to Faraday's law when a conductor carrying current is placed in a magnetic field. Then the potential is generated in the conductor. This potential is measured in Volts. The back EMF is resisting the motor's natural movement and is because of that called the "back" EMF [14]. Back EMF is equal to the angular velocity times the back EMF constant according to Equation 3.

$$E = K * w \qquad (3)$$

K [V/ms-1] is the back EMF constant, and w[rad/s] is the angular velocity [11].

Equation 4 can be used to convert from revolutions per minute to angular velocity [11].

$$w = 2 * pi * n/60 \qquad (4)$$

n is revolutions per minute(rpm) [11].

The torque produced from a BLDC motor is equal to the torque constant multiplied by the current motor windings [11].

$$M = Km * I \qquad (5)$$

M[Nm] is torque, Km[Nm/A] is torque constant, and I is current in the motor windings. Equation 5 shows that current is proportional to torque.

Equation 6 can be used to calculate torque [11].

$$P = T * w \qquad (6)$$

P is power, and T is torque, and w is the speed of the motor [11].

2.2 Electronic speed controller (ESC)

An electronic speed controller is an electric circuit used for controlling and regulating an electric motor's speed. Transistors called mosfets control which phases in the motor are activated in what order and speed. Phases in the motor are attached to ground and Vcc, and mosfet transistors work as switches by either closing the switch to the Vcc or ground. With some ESCs, it is also possible to run the motor reversibly and perform dynamic braking [10].

The ESC can determine which phase to activate at what time. To do that, the ESC must know the position of the rotor. There are two ways for it to find out the position of the rotor. One is with three Hall-effect sensors in the motor arranged equally 120 degrees or 60 degrees from each other. When the rotor's permanent magnet rotates, the sensors sense the magnetic field and generate a high logic. With this information, ESC knows when to commutate the phases [10].
The second way is through back EMF force. Back-EMF appears when the rotor's moving magnetic field passes through the coil that causes a current to flow in the coil, resulting in a voltage drop in that coil. With the help of dropping voltage, ESC can predict which phase to activate [10].

2.3 VESC

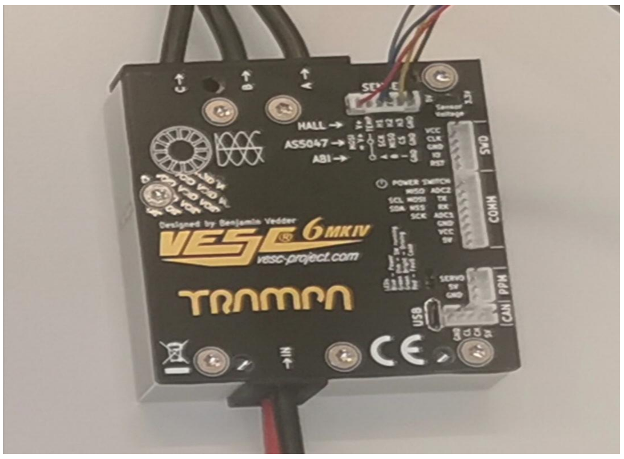


Figure 3: VESC Hardware [12].

As mentioned above, the ESC used in this thesis is called VESC. VESC is an open-source ESC. VESC is using an STM32F4 microcontroller and can operate with a sensor or a sensorless field-oriented control FOC. It can auto-detect all motor parameters. In addition, VESC allows for configurable RPM-, current-, voltage, and power limits [12 - 13].

As shown in Figure 3, there are multiple ways to communicate with VESC. It is possible to communicate with VESC using PPM signals (RC servo), analog, UART, I2C, USB, or CAN-bus. In addition, free, open-source software called VESC Tool has USB implemented and can also communicate with VESC [12 - 13].

2.3.1 Field-Oriented Control (FOC)

The commutation method used in VESC is called Field-oriented control (FOC).
Field-oriented control is an electronic commutation method that strives to get a 90-degree angle between the rotor magnetic field and stator magnetic field, which is when the highest torque is accomplished. The first step in FOC is to find the rotor position, in which direction the rotor's magnetic field is pointing. The next step is to compute the desired stator field vector by controlling the 3 phase currents. In FOC, it is dealt with alternating currents (AC) that have sinusoidal waveforms. A proportional-integral-derivative (PID) controller cannot work with sinusoidal waveforms [14].
Using Park and Clark transform, it is possible to translate the stationary stator reference frame into the rotating reference frame. In other words, converting AC currents to DC currents, which are the direct and quadrature currents [18]. These two measured currents are then compared to the desired reference values and fed into two PI regulators. One PI regulator is used to minimise the direct current, and the other PI controller is used to maximise the quadrature current. The PI regulators have two voltage outputs, Va and Vb, which need to be converted to the 3 phase reference frame with the use of the inverse Clarke and Park transform before being sent back to the motor [15].

2.3.2 Pulse Width Modulation (PWM)

VESC uses Pulse width modulation (PWM) to control the motor. PWM is a method to control a motor's speed by regulating its terminals' voltage. It works by driving the motor with a series of "ON-OFF" pulses and varying the duty cycle. The duty cycle is the amount of time when the signal is active compared to when it is low for a given period. The power applied to the motor is controlled by changing the pulses' width, which results in a changed voltage applied to the motor's terminals. Thus, higher speed can be achieved by either changing the frequency or the pulse width [16].

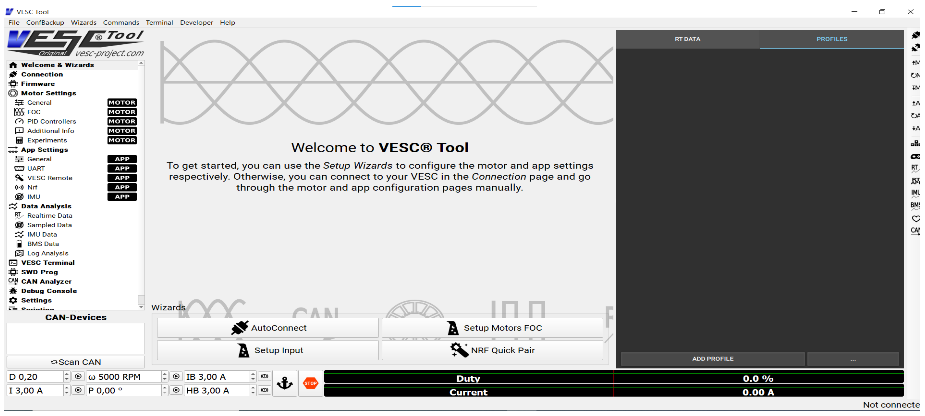
2.3.3 Proportional–Integral–Derivative regulator (PID)

VESC provides a PID controller customised through VESC Tool to control the rotor's speed and position. PID controller is a control-loop system that calculates the error value as a difference between the determining value and the measured value. There are three parts in the PID controller: the first part is the proportional part responsible for error responses of the system, the second part is the integral part that eliminates the steady-state error part, and the third one is the derivative part, which improves the stability of the system[17].

2.3.4 Regenerative Braking System

Regenerative braking is a braking method that converts mechanical energy from the motor to electrical energy [18]. VESC provides this braking method. When the user stops the motor, VESC stops to supply power to the motor and at the same time uses the power produced by the motor to charge the battery [19].

2.4 Vesc Tool



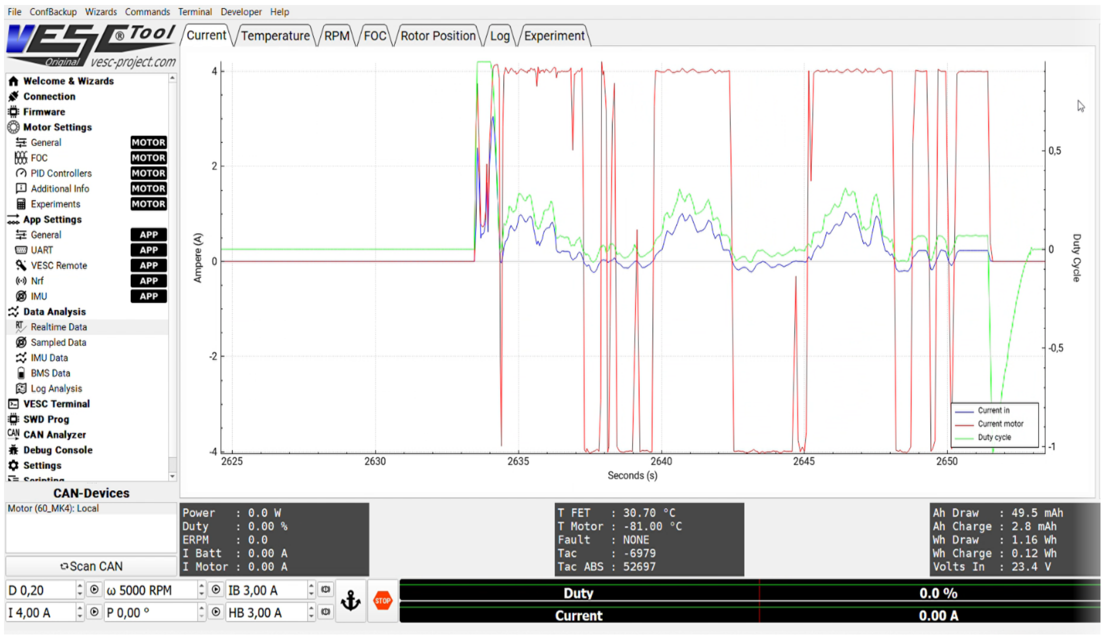
Welcome to **VESC® Tool**

To get started, you can use the *Setup Wizards* to configure the motor and app settings respectively. Otherwise, you can connect to your VESC in the *Connection* page and go through the motor and app configuration pages manually.

```
D 0,20    ω 5000 RPM   IB 3,00 A
I 3,00 A  P 0,00 °     HB 3,00 A
Duty       0.0 %
Current    0.00 A
```

Figure 4: VESC Tool is software used for communication with VESC [12].

As seen in Figure 4, VESC Tool is a feature-rich application that can be used to communicate with VESC. It can also be used to configure, monitor, and retrieve motor configuration data. In Vesc Tool, it is possible to use both TCP and UDP to communicate with other applications. Vesc Tool contains an integrated development environment which supports writing code in JavaScript and QML [13].
It is possible to set boundary limits for parameters like current, voltage, and temperature in Vesc Tool, protecting the system components from the most common user mistakes. For example, one parameter regulates the minimum and maximum input voltage to protect the motor [12].
Vesc Tool provides real-time data where users can track different parameters like power, duty cycle, current in motor, rotor position, rpm, current from the battery, supplied voltage, fault errors like high temperature, and rotor position, as seen in Figure 5 [12].



```
Power   : 0.0 W        T FET   : 30.70 °C      Ah Draw   : 49.5 mAh
Duty    : 0.00 %       T Motor : -81.00 °C     Ah Charge : 2.8 mAh
ERPM    : 0.0          Fault   : NONE          Wh Draw   : 1.16 Wh
I Batt  : 0.00 A       Tac     : -6979         Wh Charge : 0.12 Wh
I Motor : 0.00 A       Tac ABS : 52697         Volts In  : 23.4 V

D 0,20    ω 5000 RPM   IB 3,00 A
I 4,00 A  P 0,00 °     HB 3,00 A
Duty       0.0 %
Current    0.00 A
```

Figure 5: Real-Time data analysis in Vesc Tool [12].

2.5 Storing Information

This project requires a database to store user personal and training information. There are many database management systems available to choose between. Three database management systems that Raspberry PI supports are PostgreSQL, SQLite, and MySql [20]. In the text below, a description of these three databases is presented.

2.5.1 Database

The database is a collection of logically related data, and it has both data and a description of it stored in it. It is designed to meet the data demand of an organisation [20].
Relational databases use structured query language SQL. Relational databases have different categories similar to phone books [20].
A relational database is a highly stable, reliable, and consistent way of storing data. It is easy to integrate data from other systems into a database. Three database management systems are MySQL, PostgreSQL, and SQLite [20].

MySQL

MySQL is a database management system that can run on many platforms, such as Linux, Windows, and macOS X. MySQL provides much documentation and support and is increasingly being used because of its stability, support, and low cost. MySQL is under the GNU General Public License and the GNU Lesser General Public License, which means it is free and no license is required. APIs that can be used with MySQL are C, C++, JDBC/Java, Python, and PHP [21].

PostgreSQL

PostgreSQL is one of the most advanced open-source database management systems available. PostgreSQL can also run on different platforms like Mac, Windows, and Linux. PostgreSQL offers many features which make it easier to create and maintain databases. It provides advanced functionalities like declarative SQL queries, multi-version concurrency control, multi-user support, transactions, inheritance, and arrays. PostgreSQL allows users to interact with other languages such as Java, Python, and C++ [22].

SQLite

SQLite is an open-source embedded database which means it is not a standalone process. It runs and exists within the application it serves. It does not necessitate the use of a separate server process or system. One advantage of this is that there is no need for network configuration, and no administration is required. It is a cross-platform database because the database instance is in a single cross-platform file. SQLite allows users to interact with other languages such as Java, Python, and C++ [23].

2.6. Graphical user interface (GUI)

Several programming languages are available to create a graphical user interface. Three programming languages that Raspberry PI supports are Java, C++/C, and Python [24]. In this chapter, a description of all the different programming languages is presented.

Java

The most famous and known GUI library in java is Swing. Java is suitable for creating a GUI because it is easy to create and manage windows and add components. Java also provides many predefined widgets, such as menus, dialogs, buttons, and other necessary components [25].

Python

There are several modules in Python used to implement a graphical user interface. The most commonly used library to create a GUI in python is Tkinter. Tkinter is a free and open-source cross-platform widget toolkit. Like the Java swing library, Tkinter provides many predefined widgets, making it easier to develop a GUI. Widgets in Tkinter have default functionalities, such as arranging the elements according to layout and handling different types of events [25 - 26].

C++

There are many different C++ compilers available for implementing a GUI. Two commonly used compilers for creating a GUI in C++ are Qtcreator and wxWidgets toolkit [41]. Qt Creator is a cross-platform integrated development environment (IDE) for developing apps for desktop, mobile, and embedded platforms. Qt Creator runs on Windows, Linux, and macOS desktop operating systems [27].

3 Related works

There are various methods to implement a system similar to what this project proposes to design. In the text below, three articles are described and how the control system was implemented in these articles.

In the article "Design of Embedded System for Resistance Type Exercise Machine" [2] an exercise machine that provides an auto-configurable exercise routine and fine motion control is developed. A permanent magnet direct current (PMDC) motor is used to produce force. They have used a microcontroller of model PCM-3370 as the master and an electronic control module as a slave. The master sends instructions to the slave through ISA-Bus architecture. A driver, which consists of a DC power supply and an H-bridge amplifier, amplifies the power of the output signals from the microcontroller before being transmitted to the DC motor to proportionally higher power levels. A PID control algorithm is used to maintain the chosen torque and speed. The algorithm takes an error signal as input and outputs a PWM duty cycle, generating a commanded input current signal. Finally, the signal is sent to the driver, which regulates the motor's armature current according to the difference between the commanded current from the microcontroller and the current feedback [2].

In the article “Torque Control of Brushless DC Motors Applied to Electric Vehicles” it is proposed to create a control system for a brushless DC motor. The goal is to achieve similar output characteristics to that of a continuously variable transmission where the torque is constant at the slow-speed area of the motor, and the power is constant at the high-speed area of the motor. A Texas Instruments TMS320C240 (‘C240) DSP card was used to execute the control program, and an interface circuit to drive the power stage and the motor is used [28]. Field weakening control and phase advance control are used to enable higher motor speeds by reducing the back EMF generated by the motor.
Hall sensors were used to measure the speed of the motor, which was used to decide the working area of the motor according to the torque-speed curve of the motor. From the motor working area, a maximum torque value Tmax, maximum power Pmax and a base speed Wb are determined [28].
To keep the torque constant in the low-speed area of the motor, a current sensor is used to measure the motor current. The relationship between torque and current is obtained from the current values, and an equation is determined. To be able to keep the current constant, an IP current controller was used [28].

In the article "Power Control of Brushless Permanent Magnet Electric Machine for Exercise Bikes,[34]” a power control structure for a brushless permanent magnet electric machine is presented. This power control structure is implemented for an exercise bicycle. For controlling and regulating the resistance created from the motor, a regenerative braking system has been used. While paddling, the power generated by the user moves to the battery in the form of electrical energy [3]. The desired torque (required resistance) is achieved by regulating the phase current as torque is a function of the driving current. Controlling current is done by determining the rotor position with the help of the Hall sensors. The motor used in this system has a trapezoidal Back EMF with current flowing into only two phases at a given time. Ideally, total torque is constant, which means no torque ripples. However, in this case, the total torque is not ideal. Torque ripple has been reduced by adding a factor k (between 0 and 1) in Equation 5. At high speed, k is recommended to be closer to 1, and a smaller k is recommended at the low speed [3].

A customized workout algorithm is implemented to improve users' training performance. In the algorithm, torque is determined by the user's pedaling speed. The torque value is calculated by comparing the base speed to the actual speed. When the actual speed exceeds the base speed, the torque will be decreased by multiplying power in Equation 6 with a factor between 0 and 1 and dividing it by speed. When the actual speed is less than the base speed, T is multiplied by a value between 0 and 1 to keep the torque constant. [3].

4. Method

This chapter begins with a description of the overall system architecture.
This followed by describing how communication was enabled between different parts of the system. The Graphical user interface and MySQL parts describe and motivate what database management system and programming language was chosen to develop a GUI and store data. Finally, a description of how the torque was kept constant and how customised algorithms were implemented to vary the resistance dynamically is presented.

4.1. System architecture

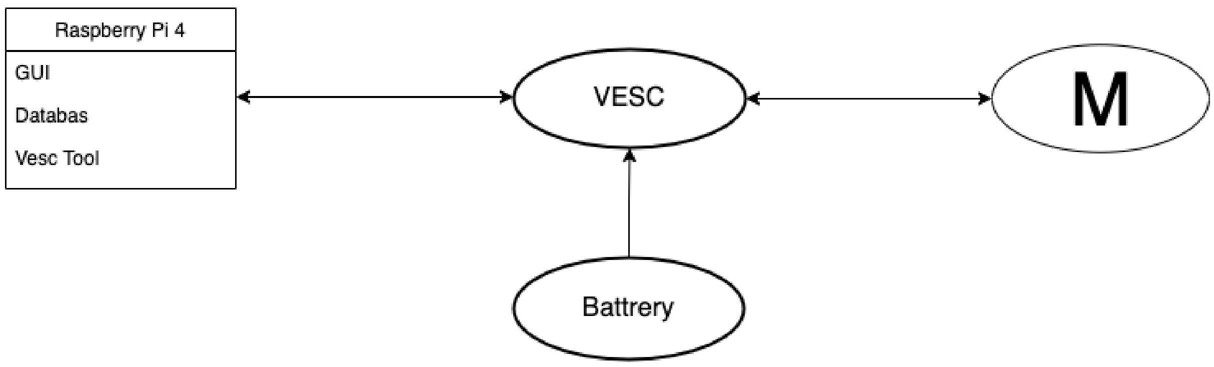


Figure 6: Architecture design for the system.

As shown in Figure 6, a Raspberry PI was utilized where the software Vesc Tool and a database were installed. The GUI has also been implemented in the Raspberry Pi. A user has the possibility to either log in or use the system without logging in. If the user chooses to log in, the user will be connected to the database and can store and retrieve training information. The user can select a weight using the GUI, and the chosen weight will be sent from the GUI to the software Vesc Tool using the communication protocol TCP. In the Vesc Tool, the current corresponding to the selected weight is calculated and is sent to VESC. VESC is then controlling the motor according to the calculated current. VESC also provides Vesc Tool with feedback from the motor such as rpm, motor current, voltage, and power in the motor. To power the system, a battery is used.

4.2 Method Description

4.2.1 Communication with VESC

The first step was to decide how to communicate with the electrical speed controller VESC. As mentioned in the background, there are several ways of doing it. Implementing a communication protocol from scratch was not possible because of the time limit in this project. After some research, it could be found that there are implementations of UART, CANBUS, and USB available online. Implementation of USB is within the desktop application Vesc Tool.
The communication protocol that was chosen was USB through the software VESC Tool. Vesc Tool was chosen because, except for the implementation of USB, there are other helpful functions available to control the motor.

4.2.2 Communication between the graphical user interface and Vesc Tool

As mentioned in the background, there are different ways to communicate with Vesc Tool. Both TCP and UDP could be utilised to communicate between the GUI and Vesc Tool.
The communication protocol that was chosen to communicate between the GUI and Vesc Tool is Transmission Control Protocol (TCP). TCP was chosen over UDP because it is more reliable than UDP. Reliability is essential in this project because the motor should always produce the chosen weight and if a user has to stop the motor, the command should always reach the motor or it can lead to injuries. The client-side was created in the graphical user interface using Java Socket class, and the server-side was created inside Vesc Tools using JavaScript and QML, QTcpServer class [29].

4.2.3 Graphical user interface (GUI)

Raspberry PI supports various programming languages such as Java, C++/C, and Python for creating graphical user interfaces [24].
The programming language that was chosen to create a GUI was Java. Java was chosen because of better documentation and support than Python and C++, which facilitates using it and looking for support when encountering a problem.
Java is also suitable for creating GUI because it provides many predefined widgets, making it easier to create a GUI. It was also chosen because earlier knowledge and experience with Java are better compared to the other programming languages.

4.2.4 MYSQL

As mentioned in the background, there are different kinds of DBMS.
In this project, all three databases described in the background could be utilised because the differences between them are minor.
Using this system in several machines in the future facilitates the maintenance of the database if all of the machines are connected to the same database server.
Considering this, SQLite is not a good option because it has no server, as mentioned earlier in the background.

In this project, MySQL was chosen because of better documentation and support compared to PostgreSQL [30].

4.3 Controlling the motor

As mentioned earlier in the background, Vesc Tool contains an integrated development environment which supports writing code in JavaScript and QML. The algorithms to control the motor were implemented within the application Vesc Tool. Vesc Tool was also used to retrieve motor information such as current and rpm. This chapter describes how experiments were conducted and what algorithm was implemented to keep the torque constant. The Custom algorithm part describes how algorithms were implemented to vary the torque dynamically.

4.3.1 Relation between current and torque

The current was used to regulate the torque because, as mentioned earlier in the background, torque is proportional to the current (Equation 5). A dynamometer was used to conduct experiments to determine the relationship between current and torque. Figure 7 shows the experiment setup, which included a dynamometer connected to a rope and a motor. The dynamometer was pulled in concentric and eccentric phases to record the force produced by the motor. Because the dynamometer used in this project was not electric, the values of the dynamometer had to be recorded using a phone during the experiments. The test was carried out with various current levels ranging from 0 to 8 A and increased in 0.5 A increments. Due to the fact that the same current value did not correlate to the same weight in both the concentric and eccentric phases throughout the tests, two equations, 7 and 8, were derived. The concentric phase is represented by Equation 7, while the eccentric phase is represented by Equation 8.

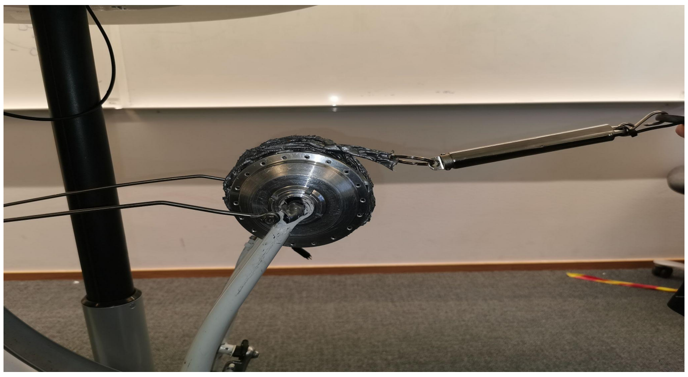


Figure 7: Experimental setup to determine the relationship between current and force.

The current for a given weight can be calculated using Equations 7 and 8. In order to identify which equation to use, it was necessary to compute the motor's direction, as shown in Figure 8. The direction of the motor was determined by using the command *COMM_GET_VALUES*, which provides a positive rpm value during the eccentric phase and a negative rpm value during the concentric phase. Equations 7 or 8 were used to compute the correct amount of current to send to the motor when the direction was confirmed. The method *mCommands.setCurrent()* was used to send the current value to the motor.

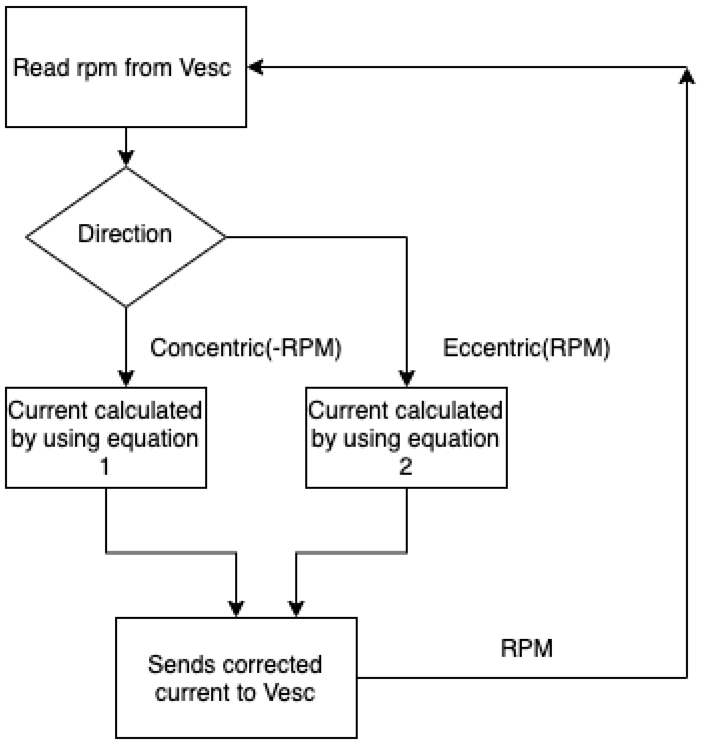

Read rpm from Vesc
Direction
Concentric(-RPM)
Eccentric(RPM)
Current calculated by using equation 1
Current calculated by using equation 2
Sends corrected current to Vesc
RPM

Figure 8: Flowchart for achieving a constant torque.

4.3.2 Custom algorithms

Two algorithms were developed where the resistance is changing dynamically during a workout. These were developed to test if it is possible to change the resistance dynamically with this system. The idea and inspiration for the algorithms were found from related work [3, 28].

Algorithm 1

The first algorithm decreases the weight by 1 kilogram when the weight has not moved for five seconds and increases the weight by 1 kilogram if the weight has not been stopped for ten seconds. The weight cannot increase more than the weight that was chosen from the beginning.

This is done by reading the rpm value from VESC using the command *COMM_GET_VALUES* in Vesc Tool, which returns the rpm in the motor. A timer is used to continuously read the rpm value and measure for how long the weight has not moved. If the rpm is zero for five seconds, a flag will be activated and current corresponding to 1 kilogram will be reduced from the original current value, and the command *mCommands.setCurrent()* is used to send the reduced current to the motor. A flowchart of the algorithm is shown in Figure 9.

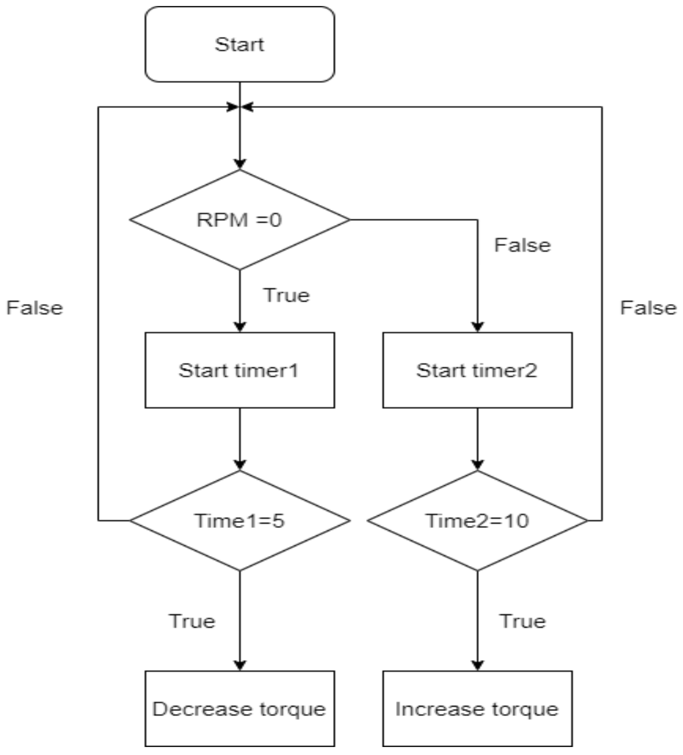


Figure 9: A custom algorithm that increases and decreases weight during a stall.

Algorithm 2

The second algorithm is working by holding the torque constant when the user performs the exercise between 0-3000 rpm during the concentric and eccentric phases. When a user exceeds the limit of 3000 rpm during the concentric phase, the weight will increase linearly with 1 kilogram for every 1000 rpm. When the user exceeds 3000 rpm during the eccentric phase, the weight decreases linearly with 1 kilogram for every 1000 rpm.

This was done by first deriving a linear equation that increases or decreases weight with 1 kilogram for every 1000 rpm depending on which direction the motor is moving. To determine when the rpm exceeds 3000 rpm, the command *COMM_GET_VALUES* was sent to VESC. If rpm was over 3000 during the concentric phase, Equation 7 was used to calculate what current to send to VESC using the command *mCommands.setCurrent().* During the eccentric phase, when the rpm is over 3000, Equation 8 and the same commands were used to decrease the weight and send it to the motor. In Figure 10, a flowchart of the algorithm can be seen.

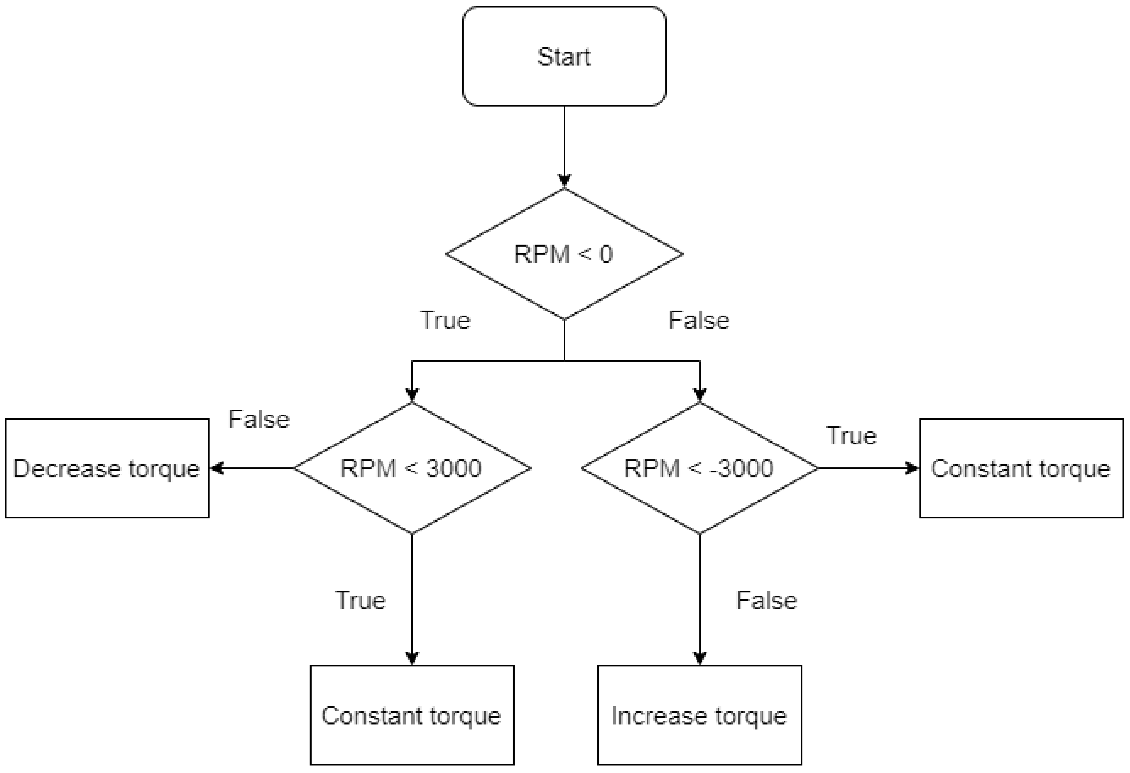
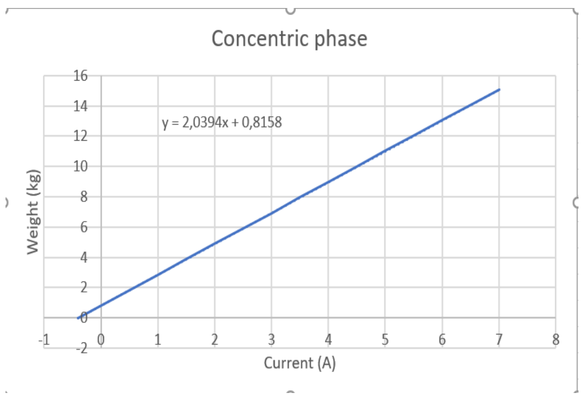


Figure 10: Custom algorithm 2 where the weight increases/decreases at high speed and maintains the same weight at low speed.

5. Result

5.1 Relation between current and torque



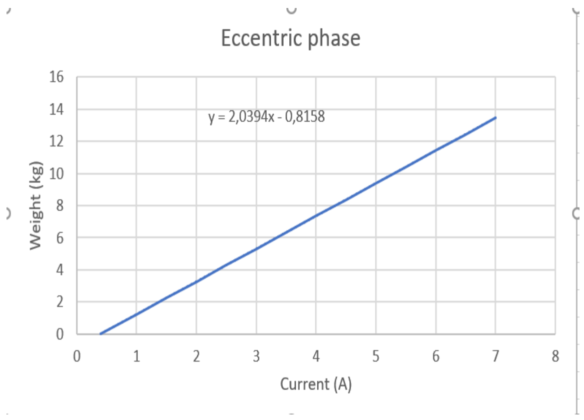


Figure 11: Weight-current relationship graph during the concentric and eccentric phase

The relationship between weight and current is displayed in Figure 11. The graphs are linear, indicating that current and torque are proportionate. Equations describing the relationship between current and weight could be obtained by plotting the graphs. Two equations, 7 and 8, were discovered.

$$y = 2.0394x + 0.8158 \qquad (7)$$
$$y = 2.0394x - 0.8158 \qquad (8)$$

Y is the weight in kg, and x is the current in ampere.

Equation 7 is for the concentric phase, and Equation 8 is for the eccentric phase and can be used to calculate the current value for a given weight. For example, a weight of 5 kilograms corresponds to a current of 2.05168 A during the concentric phase and 2.85172A during the eccentric phase.

5.2 Custom Algorithms

Two algorithms were developed which vary the resistance dynamically.
The first one is decreasing the weight with 1 kilogram when rpm is 0 during five seconds and increasing the weight with 1 kilogram when rpm is not equal to 0 during teen seconds. The weight cannot increase more than the weight that was chosen from the beginning.
The second algorithm is working by holding the torque constant when the user performs the exercise at a speed of 0-3000 rpm during the concentric and eccentric phases. When a user exceeds the limit of 3000 rpm during the concentric phase, the weight will increase linearly with 1 kilogram for every 1000 rpm. When the user exceeds 3000 rpm during the eccentric phase, the weight decreases linearly with 1 kilogram for every 1000 rpm.

5.3 Graphical user interface (GUI)

A Graphical user interface has been developed in the Raspberry Pi, where it is possible to sign up, log in, choose a weight, insert and retrieve training information. The chosen weight will be stored in a database if the user is logged in. Users can store and retrieve information such as repetitions, sets, workout time, and weight lifted. The start page of the GUI is shown in Figure 12.

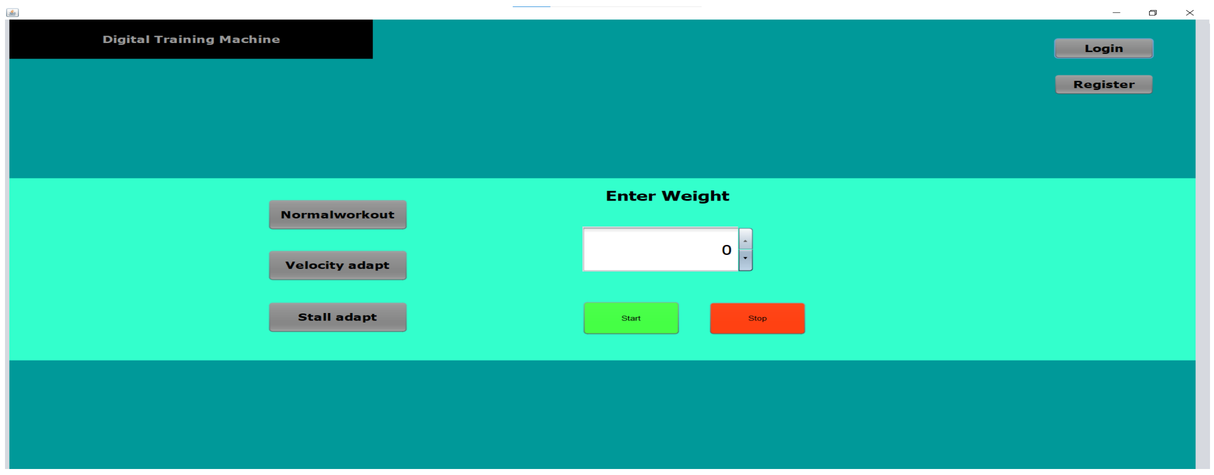


Figure 12: Start page of the graphical user interface.

5.4 Database

A MySQL database has been developed in the Raspberry Pi, which consists of two tables, as shown in Figure 13. One table for user personal information and another one for user's training information. The entity idTraining_information in the table training_information is foreign to the primary key idUser in the table user.

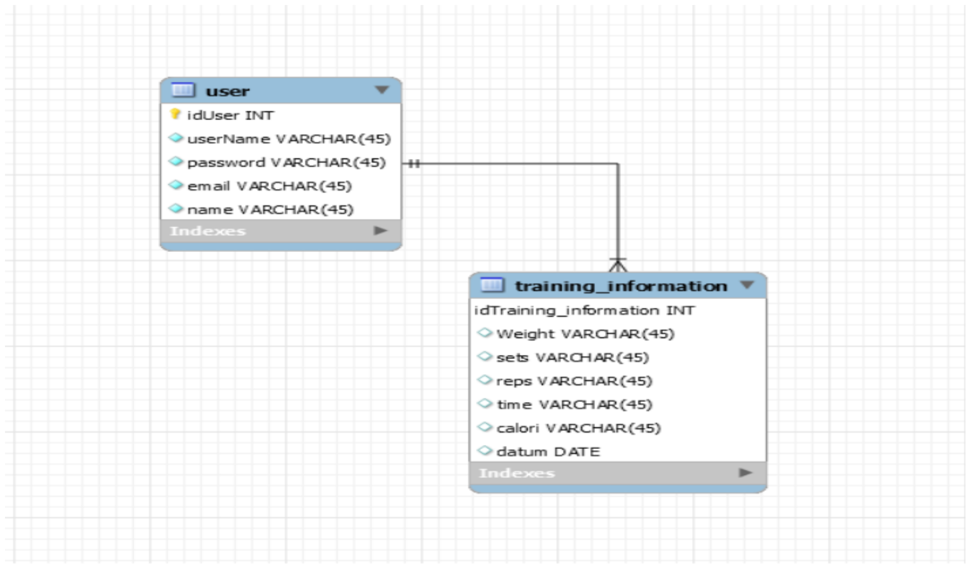


Figure 13: Database structure

6.Testing

6.1 Constant Torque

Several experiments were carried out with various weights to see if the torque was constant, as illustrated in Figures 14 - 17. There are two plots in each of Figures 14 - 17. The blue graph represents the weights, obtained by recording the dynamometer during the tests and converting the values into kilograms. The orange plot represents the weights obtained by reading the motor current values from Vesc Tool and converting them to weight by using Equations 7 and 8.

The weights obtained from Vesc Tools and the dynamometer were synchronised by recording the dynamometer using a camera that included timestamps, the timestamps from the recordings were then compared to the timestamps from Vesc Tool and both weights were plotted, as seen in Figures 14-17.

The tests below show that the weight produced is not entirely constant. Torque ripple occurs mainly during changing of direction. The margin of error is approximately between 1 and 1.5 kilograms.

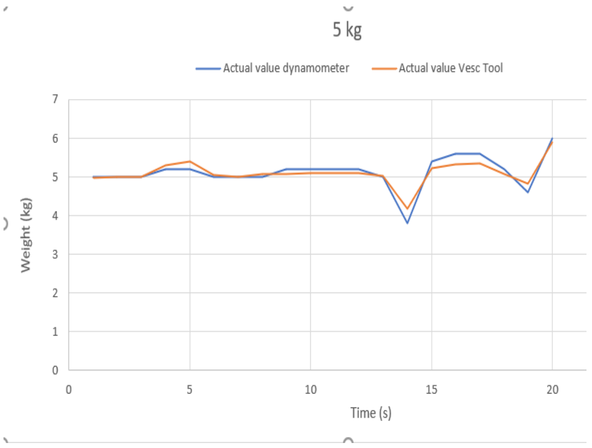


Figure 14: Test with 5 kg

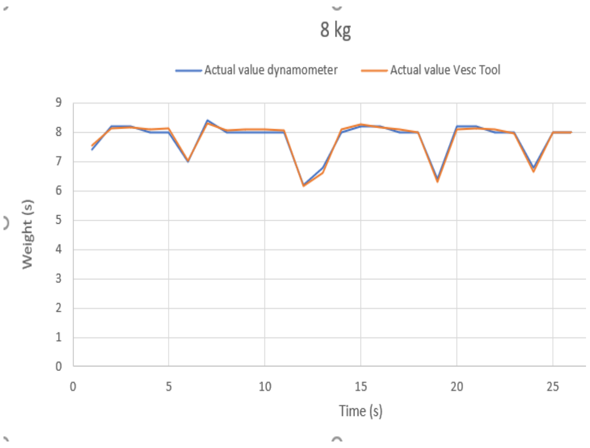


Figure 15: Test with 8kg

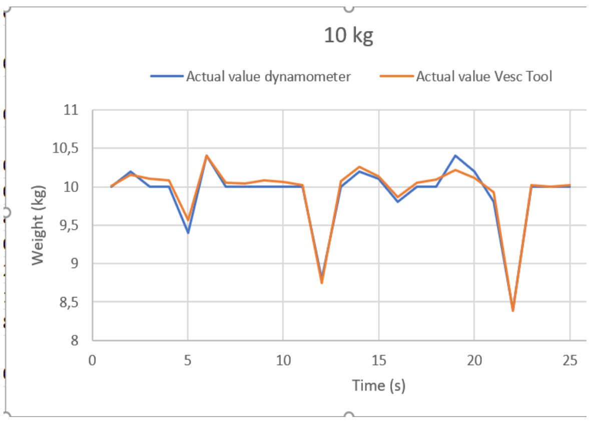


Figure 16: Test with 10 kg

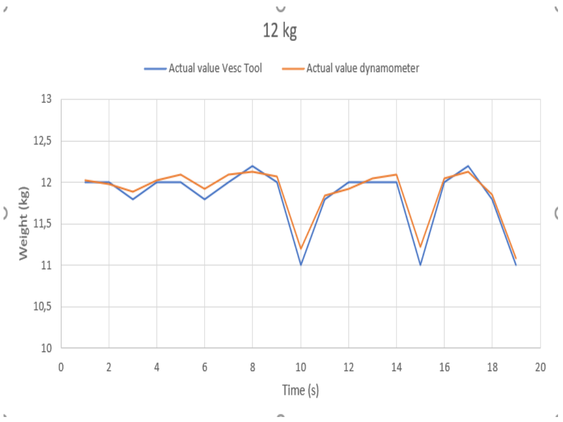


Figure 17: Test with 12 kg

Table 1: Motor current values during changing of direction

Time (s)	Current (A)
54	-2,12
55	-2,13
55	-2,07
55	-2,07
55	-2,06
55	-2,08
56	1,28
56	2,05
56	2,84
56	2,85
56	2,85
57	2,85

During changing direction, the current is going from a negative value to a positive value, and this could be seen when experiments were conducted, as shown in Table 1. Going from a negative value to a positive value does not occur immediately. It requires time, and because of this, most torque ripples occurred during changing of direction.

Cogging torque is another reason why torque does not remain constant while shifting directions. The torque produced by the magnetic interaction between the rotor magnetic field and the stator slots, which can create torque ripple at low speeds, is known as cogging torque. Cogging torque occurs at high speeds, but the rotor's inertia smooths out the torque ripple and makes it less noticeable [31].

There are some methods to reduce the cogging torque, by mechanically changing the construction of the motor, for example, changing the size of the magnet in the motor, which was not possible in this project [32].
According to [34], torque ripple during changing direction can occur because extra torque is required to overcome the inertia of the load and the rotor itself. This happens for a short period during acceleration and when the motor starts or changes direction.

The control method that VESC uses is FOC, which is not entirely free from torque ripple. The torque ripple in FOC occurs when the current is commutated from one phase to another [33].
As mentioned earlier in the background, a voltage called back EMF is generated, resisting the motor's natural movement when the motor starts to rotate. If the back EMF waveform does not match the waveform of the phase currents, it can lead to torque ripple. In this project, the motor had a trapezoidal back EMF waveform, and the waveform of FOC is sinusoidal, leading to torque ripple [31].
Another reason for torque ripple could be because of human errors when conducting the experiments. One human error could be that wrong values were registered from the dynamometer, and the dynamometer could also have some margin of error.

6.2 Custom Algorithms

Algorithm 1

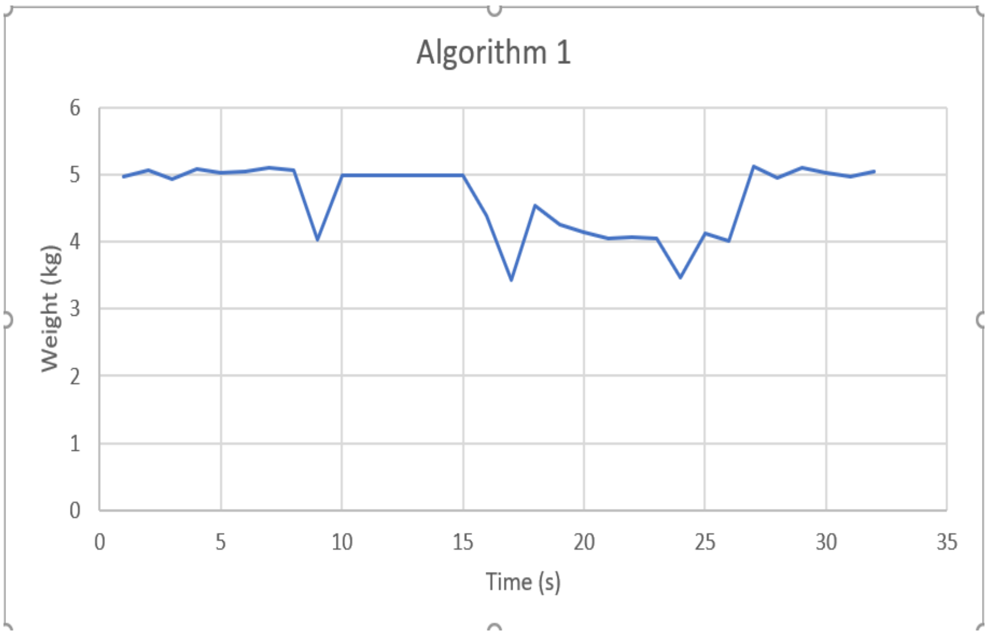
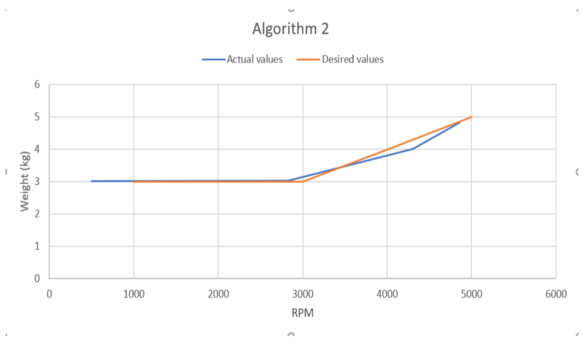


Figure 18: Test on algorithm 1

The test on algorithm 1 was performed by retrieving the motor current values with Vesc Tool. Every second Vesc Tool provided multiple motor current values. An average current value was calculated for every second, and was converted to the corresponding weight using Equation 7 and 8.
The result is plotted as shown in Figure 18. The weight is decreased with 1 kilogram when rpm is 0 during five seconds and increased with 1 kilogram when rpm is not equal to 0 during ten seconds. The torque ripples in algorithm 1 occur primarily during changing of direction.

Algorithm 2



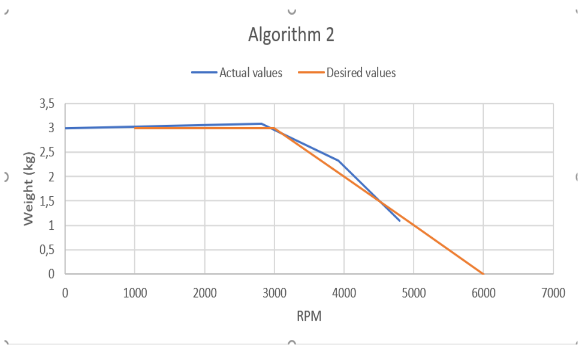


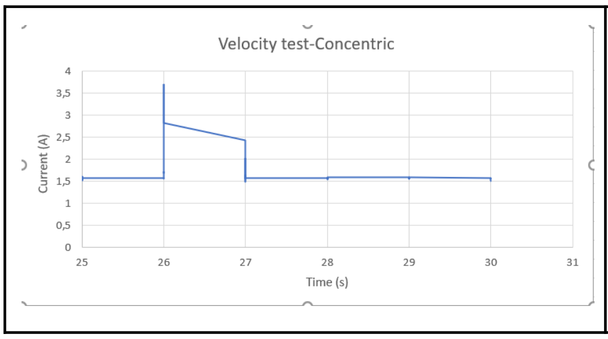
Figure 19: Algorithm 2 Concentric phase

Figure 20: Algorithm 2 Eccentric phase

Algorithm 2 was tested in the same way as algorithm 1 was tested. Vesc Tool provided the motor current values, which were then transformed to weights using Equations 7 and 8.

Figures 19 and 20 demonstrate the result of algorithm 2, where the weight remains nearly constant across the speed range of 0 rpm to 3000 rpm.
Figure 19 shows that the actual graph in the eccentric phase follows the desired graph, which decreases by almost 1 kilogram for every 1000 rpm. Figure 20 shows that the actual graph in the concentric phase also follows the desired graph, which increases with almost 1 kilogram for every 1000 rpm. Figures 19 and 20 do not include changing direction because, during the changing direction, the rpm is always below 3000 rpm and is not relevant to have in this experiment.

6.3 Velocity Test



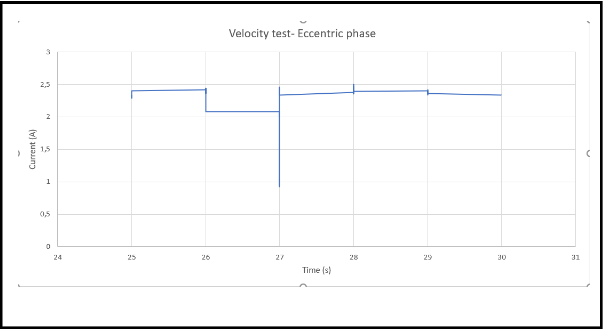


Figure 21: Velocity test in concentric phase Figure 22: Velocity test in the eccentric phase

As seen in Figure 22, the current decreases, which means the torque decreases when the weight is not dragged with a constant speed during the eccentric phase. This can be explained by the torque-speed curve of a DC motor, as seen in Figure 23, where the torque decreases when the speed of the motor increases.
This curve is derived by combining Equations 3,5, and 2, which leads to a new equation, Equation 9. By using Equation 9, a relationship between torque and speed can be determined. Equation 9 shows that torque is inversely proportional to speed in a DC motor.

Equation 9: $(Km \div R) * (V - K \times w)$

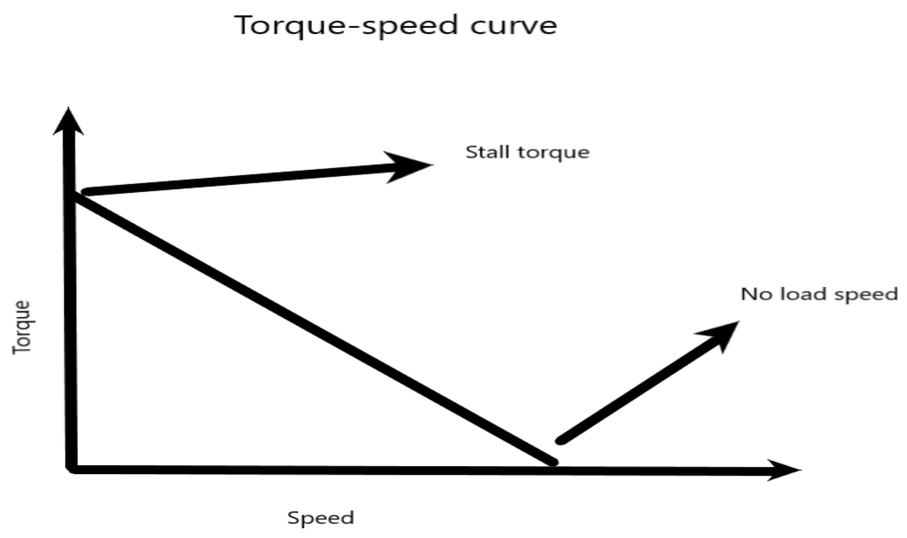


Figure 23: Torque-speed curve of a DC motor

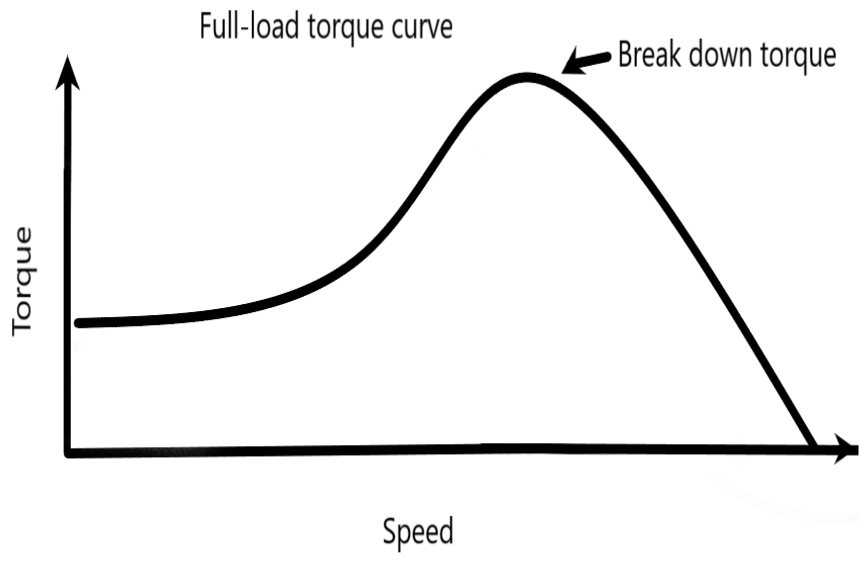


Figure 23: Full-load torque curve

As seen in Figure 21, the torque is increasing during acceleration in the concentric phase. This can be explained with the full-load torque curve of a DC motor, as seen in Figure 23 where the torque is increasing when the speed is increasing up to the breakdown torque point. Break down torque point is the highest torque available before the torque decreases when the machine accelerates to working conditions. In the experiment above, the breakdown torque point was not reached [38].

7. Discussion

In the result, a margin error of 1-1,5 kilograms occurred mostly during the change of direction where the force increased for a brief period. In this project, it was attempted to solve this problem by predicting when a change of direction will occur and in advance to correct the current, but this did not work because it was not possible to go fast enough from positive current to negative current.

As mentioned earlier, torque ripple might have occurred because of human errors when conducting the experiments. This could have been avoided if the experimental setup was better. If the whole system was placed within a smith machine, the result of the tests could have been more reliable. If an electrical dynamometer was used instead of a traditional dynamometer, the values would be registered more accurately.

As mentioned in the test section, torque ripple can occur when the waveform of back EMF does not match the waveform of the phase currents. This could have been avoided by using a motor with a sinusoidal back EMF, however it was not possible to change the motor in this project.

In the article “Performance Analysis and Comparison of BLDC Motor Drive using PI and FOC" a comparison between FOC control and PI control for a BLDC motor was made. Simulations were used instead of an actual system. The result when comparing torque ripple between PI control and FOC control was that more torque ripple occurred with PI control than FOC control when the motor started to rotate. More torque ripple occurred with PI control compared to FOC control even after the starting point.
The margin of error during the start phase was approximately 0-0.9 Nm for FOC control and 10 Nm for PI control. The margin of error during constant speed was approximately 0-0.5 Nm for FOC control and 1 Nm for PI control [37]. Comparing these results to our result indicates that our result was close to them because we have a margin of error of approximately 0-1 Nm during the start phase and around 0-0.5 Nm after the start phase.

The results of the test performed on algorithm 2 show that it works as desired. As shown in Figure 20, we got a margin error of 0-0.5Nm when rpm was below 3000. In [3], they implemented a similar algorithm to vary the torque during the eccentric phase. They produced a constant torque between 0 and 3000 rpm with a margin error of 0-1Nm, which is nearly identical to our result. They were able to get a torque that was almost identical to the desired torque. However, it was obvious that they had a margin of error of 1-2 Nm at speeds between 3000 and 6000 rpm, which is similar to our result [3].

One significant strength with this control system compared to a regular smith machine is the possibility of varying resistance. In the future, this can be developed in the same way as in the article “Microprocessor controlled robotic exercise machine for athletics and rehabilitation,” where they used two identical hydraulic motors and a microcomputer to create a constrained path of motion that is programmed to the individual needs. The resistance varies over the path of a motion, which is adapted to the user's strength potential and position at each point, and also feedback is provided for the user. This is done by measuring position with potentiometers, velocity by tachometer-generators, and force by transducers attached at the bar [4]. Varying the resistance dynamically can improve muscular strength and be beneficial in rehabilitation [4].

Another strength of this system compared to a regular smith machine is the GUI, where a user can choose a weight without loading and unloading weight on the machine. The possibility to store training data is also beneficial compared to a regular smith machine.

The whole system was implemented in a Raspberry PI, which is a strength considering that a Raspberry PI is small and does not require much space.

One limitation of this system is that the motor can overheat due to overloading. In the case of overheating, the motor's bearing can seize up, the stator coils and motor windings and hall-sensors can be damaged [35 - 36]. This leads to the following limitation, which is the need for reparation. Compared to a regular smith machine, we think this system will require more reparation. Another limitation is that the motor in this control system can only produce torque equal to 15 kilograms. To increase the weight limit, a more powerful motor or more than one motor is required.

7.1 Social requirements

Comparing this system to a regular smith machine, this system is less environmentally friendly considering electricity usage. As mentioned in the background, the system uses regenerative braking, reducing the usage of electricity. While users perform an exercise, the energy that earlier was converted to heat is now converted to electricity and charges the battery.

Regarding privacy aspects, it has been attempted to minimise the amount of private information stored in the database.

This system is using a battery which can have security issues if the battery is overheated, then it can explode,

8. Conclusion

In this project, the first requirement was to create a constant torque with the electrical speed controller VESC and BLDC motor during the concentric and eccentric phases, which are partially accomplished. As mentioned in the test section, during the concentric and eccentric phases, the torque had a margin of 1-1.5 kilograms, mostly during changing of direction. The second requirement was to create a graphical user interface where a user can choose a weight, and this requirement has been fulfilled as shown in the result. The third requirement was to store user training information that has successfully been fulfilled, as shown in the result section where the database structure is presented.

In this project, we have learned how to construct a system consisting of several components and enable communication between them, so the different parts interact and work together. We have also learned how an electric speed controller (VESC) and a motor works.

In the future, this system can be improved by eliminating torque ripple, especially during changing of direction. Custom algorithms can be improved by implementing new algorithms to individualize resistance to increase efficiency in training and rehabilitation.

9. Reference list

1. Kos, A., Wei, Y., Tomažič, S., & Umek, A. (2018). The role of science and technology in sport. Procedia Computer Science, 129, 489-495.

2. Motamarri, S. S., & Cetinkunt, S. (2008, October). Design of Embedded System for Resistance Type Exercise Machine. In *2008 IEEE/ASME International Conference on Mechtronic and Embedded Systems and Applications* (pp. 602-607). IEEE.

3. Tsai, T. C., & Tsai, M. C. (2002). Power control of a brushless permanent magnet electric machine for exercise bikes. *IFAC Proceedings Volumes*, *35*(1), 419-424.

4. Book, W. J., Ruis, D. A., & Polhemus, R. (1979). A microprocessor controlled robotic exercise machine for athletics and rehabilitation. Georgia Institute of Technology.

5. Stewart, G. D., Johnson, T. M., Orman, P. B., Lambright, J. W., Orman, G. M., & Schwartz, R. E. (1989). *U.S. Patent No. 4,869,497*. Washington, DC: U.S. Patent and Trademark Office.

6. Tonal.(2021).THE ULTIMATE STRENGTH TRAINING MACHINE.https://www.tonal.com/equipment/

7. 1080motion.(2021).1080 QUANTUM SYNCRO. https://1080motion.com/products/1080-quantum-syncro/

8. Gallo, R. A., Reitman, R. D., Altman, D. T., Altman, G. T., Jones, C. B., & Chapman, J. R. (2004). Flexion-distraction injury of the thoracolumbar spine during squat exercise with the Smith machine. The American journal of sports medicine, 32(8), 1962-1967.

9 . Smith machine, wikimedia https://commons.wikimedia.org/wiki/File:Incline-bench-press-2-2.png

10. HowToMechatronics. (2021) How Brushless Motor and ESC work. .https://howtomechatronics.com/how-it-works/how-brushless-motor-and-esc-work/

11. Khalaf, M., & Pihl, E. (2016). Styrning av en elcykel med borstlös likströmsmotor och analog teknik.

12. VESC.(2020). VESC Project https://VESC-project.com/

13. VESC.(2020). Benjamin’s robotics http://vedder.se/

14. Rosenbecker, L. (2020). Vibration Attenuation for Satellite Reaction Wheels through the use of Field-Oriented Control.

15. Anders, A., & Jonathan, P. (2020). Att skydda BLDC motorer mot oaktsam användning: Övervakning av temperatur i statorlindningar för handhållna produkter.

16. Barr, M. (2001). Pulse width modulation. Embedded Systems Programming, 14(10), 103-104.

17 . Shamseldin, M. A., & EL-Samahy, A. A. (2014, September). Speed control of BLDC motor by using PID control and self-tuning fuzzy PID controller. In 15th International Workshop on Research and Education in Mechatronics (REM) (pp. 1-9). IEEE.

18 . Yoong, M. K., Gan, Y. H., Gan, G. D., Leong, C. K., Phuan, Z. Y., Cheah, B. K., & Chew, K. W. (2010, November). Studies of regenerative braking in electric vehicle. In *2010 IEEE Conference on Sustainable Utilization and Development in Engineering and Technology* (pp. 40-45). IEEE.

19. Sayyapureddi, S. V., Nandyala, V. R., Komarneni, A., & Seth, D. (2018, July). Design and Development of an Electric Skateboard Controlled Using Weight Sensors. In *International Conference on Distributed, Ambient, and Pervasive Interactions* (pp. 275-285). Springer, Cham.

20 . Connolly, T. M., & Begg, C. E. (2015 SIXTH EDITION). Database systems: a practical approach to design, implementation, and management. Pearson Education.

21 . MySQL, A. B. (2001). MySQL.

22 . Drake, J. D., & Worsley, J. C. (2002). Practical PostgreSQL. " O'Reilly Media, Inc..”

23 . Owens, M., & Allen, G. (2010). *SQLite*. Apress LP.

24 . Maksimović, M., Vujović, V., Davidović, N., Milošević, V., & Perišić, B. (2014). Raspberry Pi as Internet of things hardware: performances and constraints. *design issues*, *3*(8).

25 . Reed, P. K. (2002). A Comparison of Programming Languages for Graphical User Interface Programming.

26 . Alomari, Z., Halimi, O. E., Sivaprasad, K., & Pandit, C. (2015). Comparative studies of six programming languages. *arXiv preprint arXiv:1504.00693*.

27 [. Qt Creator - A Cross-platform, https://www.qt.io/product/development-tools

28 . Chu, C. L., Tsai, M. C., & Chen, H. Y. (2001, June). Torque control of brushless DC motors applied to electric vehicles. In *IEMDC 2001. IEEE International Electric Machines and Drives Conference (Cat. No. 01EX485)* (pp. 82-87). IEEE.

29 . QT.(2020). Use Case - Integrating JavaScript in QML. https://doc.qt.io/qt-5/qtquick-usecase-integratingjs.html

30 . Yang, X. (2011). Analysis of DBMS: MySQL Vs PostgreSQL.

31 Sumega, M., Zoššák, Š., Varecha, P., & Rafajdus, P. (2019). Sources of torque ripple and their influence in BLDC motor drives. *Transportation Research Procedia*, *40*, 519-526.

32 Levin, N., Orlova, S. V. E. T. L. A. N. A., Pugachov, V., Ose-Zala, B., & Jakobsons, E. (2013). Methods to reduce the cogging torque in permanent magnet synchronous machines. *Elektronika IR elektrotechnika*, *19*(1), 23-26.

33. IJSTE - International Journal of Science Technology & Engineering (2015), Torque Ripple Reduction using Field Oriented Control(FOC)-A Comparison Between BLDCM and PMSM, http://www.ijste.org/articles/IJSTEV2I4073.pdf

34 . Yedamale, P. (2003). Brushless DC (BLDC) motor fundamentals. *Microchip Technology Inc*, *20*, 3-15.

35 . Kumar, D., Verma, P. K., Singh, B. K., Singh, R., & Singh, V. Single phasing phase reversal overvoltage under voltage and overheating protection of three phase induction motor. International Journal of Scientific Research and Management Studies (IJSRMS), 1(1).

36 . Moseler, O., & Isermann, R. (2000). Application of model-based fault detection to a brushless DC motor. IEEE Transactions on industrial electronics, 47(5), 1015-1020.

37. Sharma, P. K., & Sindekar, A. S. (2016, December). Performance analysis and comparison of BLDC motor drive using PI and FOC. In 2016 International Conference on Global Trends in Signal Processing, Information Computing and Communication (ICGTSPICC) (pp. 485-492). IEEE.

38. Electrical Classroom, Motor torque calculator | Full load torque of a motor, https://www.electricalclassroom.com/full-load-motor-torque-calculator/

Rana Faheem Iftikhar

Sabor Amini