Wen Song
Matriculation number: S2024613
BEng Honours Project Report
<Smartphone Interface For
Peripheral Sensors>
25 April 2024

Mission Statement

Project Definition

This project is motivated by the increasing need for a user-friendly, versatile solution to connect various sensors with smartphones. Recognizing the financial and technical difficulty of integrating a range of sensors within smartphones, a solution using an external microcontroller is investigated. The primary aims include sensor construction with Arduino boards, data transmission, and smartphone app development. This project provides a versatile and easy-to-use tool for individuals, researchers, and businesses by accomplishing these tasks.

Main Tasks

- Arduino programming for data acquisition from various sensors, considering those need external excitation.
- Android app development to collect data and send excitation signals.

Background Knowledge

- Ability to read and use technical report and data sheet for each sensor model.
- Arduino and C programming abilities, including the use of third-party library.
- Familiarization and ability to use Android app development tools.

Acceptance Criteria

- The whole route is tested on two sensors; one requires excitation signals, and one does not
- An app is developed to present data extracted from the sensors in real-time and able to change excitation signals.

Abstract

This paper explores the usage of Arduino and Bluetooth to connect peripheral sensors to the smartphone interface, demonstrates a bidirectional channel between the smartphone and the sensor on the Android system, using a passive temperature and humidity sensor and an active vibrating wire strain gauge. The readings from this channel have been validated to have low errors of 3.7% for temperature, 1.7% for humidity and 9.6% for strain measurements. This project involves sensor principle research, hardware circuit construction, Arduino programming and app creation. The work is divided into two stages, increasing complexity and focusing on different sensor types to allow better progress tracking. The validation process, particularly using Digital Image Correlation, highlights the precision and reliability of the data acquired, emphasising the system's potential for widespread industrial and scientific application. This research not only contributes to sensor principle understanding and application development but also pioneers methods for robust hardware interfacing and efficient data communication. The project's impact extends to various sectors, including environmental monitoring, infrastructure maintenance, and healthcare, by providing a cost-effective and scalable solution for integrating diverse sensor systems with mobile technology.

Declaration of Originality

I declare that this thesis is my
original work except where stated

Wen Song

....

Statement of Achievement

In this BEng project, I built a universal, versatile, user-friendly smartphone interface for various peripheral sensors. This project follows a design approach of researching a specific sensor, implementing it on an Arduino board, programming Arduino with C language, developing corresponding functions in a universal smartphone app, and final data validation. I successfully built a bidirectional communication route between various sensors and smartphone interfaces through Arduino as a microcontroller. Two sensors have been implemented, tested and validated in this project, including a temperature and humidity sensor named DHT-11 and a vibrating wire strain gauge with a readout unit VM501.

The critical difference between these two sensors is that the vibrating wire strain gauge needs excitation signals to start sensing data, while the temperature and humidity sensor does not. My work during the project is split into two stages, each for one sensor. The project began with a temperature and humidity sensor, DHT-11, as a quick start with essential features and only a receiving path from sensors to smartphones. The project was developed for the vibrating wire strain gauge with the readout unit VM501 by adding new features to accommodate it. Therefore, the project finally achieved bidirectional communications between smartphones and sensors.

The validation process of data measured has also been emphasized in this project, as it marks the quality of the sensors and indicates the stability of the communication route. Data from DHT-11 sensor were compared with local weather reports, with errors of 3.7% for temperature and 1.7% for humidity. For the vibrating wire strain gauge data, a cross-discipline validation unitizing Digital Image Correlation (DIC) technique is carried out in collaboration with the civil engineering department in the university. The readings obtained from this strain gauge proved satisfyingly accurate, with an average error of 9.6%. Although validation was not the main task, I still invested as much time as the primary programming process. Therefore, this detailed verification work for the two sensors enriched my project to a higher level.

Contents

Chapter 1

Introduction and Background

1.1 Current Landscape

In the current technological landscape, integrating sensors with smartphones represents a rapidly evolving field driven by the increasing connections between electronic devices and the real world. On the other hand, the financial and technical challenges associated with embedding a broad spectrum of sensors directly into smartphones are addressed as obstacles. Most sensors with uncommon or specific needs are still peripheral and used with independent readout instruments. However, smartphones can now interface with a wide array of sensors as a comprehensive and universal platform, extending their functionality beyond conventional uses.

1.2 Project Motivation

This project is motivated by the pressing demand for a comprehensive and user-friendly platform to bridge the gap between various sensors and smartphones. It proposes an approach utilizing an external microcontroller controlled by a smartphone app. This solution not only overcomes the limitations imposed by modern smartphone design's compact and complex architecture but also amplifies its utility beyond conventional use. By facilitating the data acquisition process from diverse sensors into one accessible readout, the project aims to enhance the user experience significantly. This advancement holds profound implications for a broad range of potential stakeholders. Individual researchers will benefit from convenience in fieldwork; sensor manufacturers will save time and effort designing readout instruments for each sensor, and the system can be regularly updated with new sensors connected as peripheral modules.

1.3 Project Overview

The project aims to build a functional base tested by two sensors with different sensing principles. The selection of the two sensors has been carefully assessed; each represents a general case of the sensing principle, and together, they cover the most cases of sensors in the market. Successful tests and validations on the two sensors will indicate the completion of the project.

1.3.1 Introduction of Excitation

Sensors can be broadly categorized based on their operational requirements, specifically whether they require excitation signals. Passive sensors operate without needing an external excitation signal and only rely on the power supply. On the other hand, active sensors necessitate an external power source, an "excitation signal", to initiate their measurement process with the essential power supply.

This fundamental distinction on the project indicates a requirement for bidirectional data communication. For sensors that do not need excitation, the data path is a single way from the sensor, passing through the microcontroller to the smartphone display. On the other hand, for sensors that require excitation, as the excitation changes to influence the measurement results, the smartphone app must be able to modify the excitation signal. Thus, a dual-way data path will be necessary between the smartphone and sensors, passing through the microcontroller.

1.3.2 Division of Project Tasks

The project starts with a sensor with no excitation, the DHT-11 temperature and humidity sensor. This passive sensor focuses primarily on Arduino programming and app development to construct a proof of concept and implement such a system. Then, with the essential model functions, the project explores a more complex and realistic vibrating wire sensor type. The sensor used in the second stage is an embedded strain gauge. This active sensor will post the challenge of setting excitation signals from a smartphone app and, therefore, the construction of a bidirectional data communication.

Chapter 2

Research and Literature Review

2.1 Introduction to Literature Review

Smartphone's increasing connections with the world represent a rapidly evolving field with significant implications across many applications, from health monitoring to environmental sensing and home automation systems. This literature review aims to delineate the current landscape of research and development, focusing on employing smartphone interfaces for sensor data acquisition and control, utilising Bluetooth technology for peripheral connectivity, and applying Arduino microcontroller as a flexible and accessible platform for sensor integration. By examining these areas, the review highlights the technological advancements, challenges, and solutions presented in existing studies.

2.2 Smartphone-Sensor Interfaces

In recent years, many studies have explored the use of smartphones as interfaces for sensors. These researches commonly highlighted the convenience of accessing sensors on the smartphone interface. They underscored the potential of smartphone-sensor connections across diverse applications.

Cheng et al. [1] made progress in intelligent wearable sensors for health monitoring based on biofluids and introduced the latest applications of these sensors. The article stresses the importance of wearable sensors in providing real-time information feedback to the patient in a more accessible way. The incorporation of wearable sensors with smartphones for data visualization and analysis was mentioned as a possible extension, demonstrating the potential for the integration of smartphone interfaces in healthcare monitoring.

Similarly, research by Beduk et al. [2] discussed the first use of a smartphone-based sensor system for detection of biofluids and presented various sensing techniques, proving the possibility of applying smartphone's portability, high accessibility and fast sample processing to an actual project.

2.3 Bluetooth Technology in Peripheral Connectivity

Bluetooth technology emerges as a pivotal wireless communication standard in exploring peripheral connectivity. It is renowned for its low power consumption, cost-effectiveness, and notable range, crucial for the seamless integration of smartphone sensors. The relevance of Bluetooth in facilitating such connections is underscored by Pham et al. [3], who detailed its architectural superiority in maintaining stable connections across a multitude of devices, highlighting its capacity for the Internet of Things (IoT) domain, such as smart home systems. A similar conclusion was drawn from another research carried out by Rahul N. et al. [4], and developed in the methodology of connecting sensor nodes to Internet-based services and applications. Both analyses provided essential insights into how Bluetooth technology optimizes energy use, a critical consideration for battery-operated sensor modules and connection to mobile devices.

2.4 Arduino as a Microcontroller for Sensor Networks

Massimo Banzi [5], one of the co-founders of the Arduino project, illustrated Arduino's capability to act as a microcontroller for electronics networks and highlighted its versatility, ease of use, and open-source nature. The book also includes tutorials for digital input and output processing in section 8, which can provide direct help and guidance for the project.

2.5 Overview of Sensor Technology

Sensor technology encompasses devices designed to detect changes and respond to environmental conditions. A foundational reference in understanding the breadth of sensor technologies is provided by Göpel et al. [6] in their comprehensive review. This work systematically categorised sensors based on their sensing principles, highlighting the distinctions between sensors that require external excitation signals and those that directly function in response to external physical changes. For the project, this classification is instrumental in selecting sensors that align with the objectives and demonstrate the project's versatility in interfacing with different sensor types. The insights provided by Göpel's book show the operational and integration differences between various sensors, including inherently active ones, such as the vibrating wire sensor, which necessitates external excitation. This distinction sets a fundamental requirement for a versatile sensor application framework, highlighting integration capabilities for different sensors.

Exploring specific models of vibrating wire sensors, the strain gauge frequently plays a crucial role in the measurements for civil and structural engineering projects. Xu et al. [7] provided an in-depth explanation of the operational principle of vibrating wire sensors, which includes converting many physical parameters into a change in frequency of a taut wire. This reference is relevant and necessary to the project as it underscores the sensor's excitation mechanism, response characteristics, and application spectrum. It informs the methodology in integrating this sensor type with the Arduino digital control and emphasises the project's appropriate data acquisition and monitoring approach.

Furthermore, it is also crucial to note that the methodology may contain analogue and digital signals, depending on sensor types. According to the desired route of the project, even if analogue signals exist in the sensor circuit, they end at the microcontroller. Boyu S. [8] pursued a design of cost-efficient analogue-to-digital conversion (ADC) circuits for a temperature sensor. The key concepts come from conversion and re-sampling between time and frequency domains, which is one of the primary learning outcomes of undergraduate discipline in the university. This research linked the materials covered in theoretical, test-based courses to a practical and similar case study of the project and provided valuable advice and guidance.

2.6 Conclusion and Critical Analysis of the Literature Review

This literature review shows a promising landscape of sensor technology, Bluetooth communication, and Arduino's roles in sensor networks, laying a solid foundation for integrating peripheral sensors with smartphones. The key findings highlight the smartphone's evolving capabilities as interfaces for various sensors, the efficiency of Bluetooth wireless connections, and the adaptability of Arduino as a microcontroller to be used in this project.

However, despite comprehensive insights, existing research still presents gaps and unsolved work, particularly in creating a unified, user-friendly system for diverse sensors. Most studies focus on a specific type of sensor or an isolated aspect of this integration, leaving a potential development for this project. The project utilizes passive (DHT-11) and active (vibrating wire strain gauge) sensors, highlighting its intention to create a versatile system capable of supporting various sensor technologies. This intention widens the range of users and lays the groundwork for exploring a broader impact of such work in real-world applications.

In summary, while the existing literature provides valuable insights into the three main sectors, this project aims to fill the research gap by developing a unified, user-friendly, and systematic route that connects smartphone interfaces to various sensors. This effort can be assured by existing technical guidance and demonstrates its potential to influence popular domains like IoT, environmental measurement and healthcare monitoring significantly.

Chapter 3

Methodology

3.1 Project Development Approach

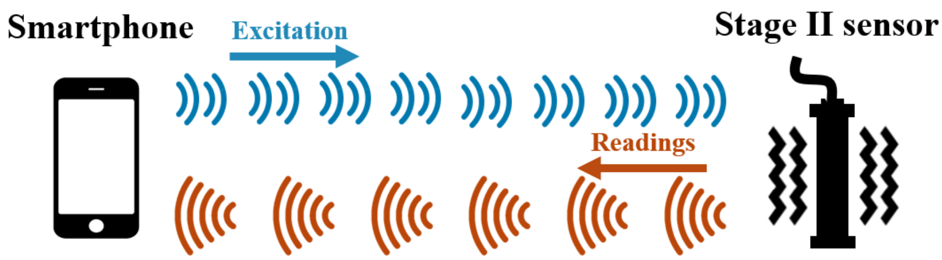
This section outlines the process to develop the smartphone interface for peripheral sensors. The project is structured into two stages, testing with two types of sensors to expand the system's complexity and functionality gradually. The sensors used in this project are a passive DHT-11 temperature and humidity sensor in stage I and an active vibrating wire strain gauge in stage II. Different sensors represent the difference in data paths. Illustrations of the stages are given below.

In stage I, as shown in Figure 3.1a, the smartphone only performs the receiving function. A thermometer icon in the schematic symbolizes the sensor employed in this stage. The data path is unidirectional: "Readings" are captured by the sensor and transmitted to the smartphone interface. The simplicity of this stage aims to build the basic functions.

Moving to stage II, as shown in Figure 3.1b, the system evolves to a more complex one with two data paths, achieving a bidirectional communication. The stage II sensor necessitates excitation to work. For conceptual design, the smartphone issues an "Excitation" signal to activate the sensor. Subsequently, the sensor responds with a "Readings" signal, conveying the measurement data to the smartphone interface.

To ensure an effective progression of work and to guarantee a clear traceability of progress in each stage, the following tasks are undertaken for the two stages described above:

(a) Stage I unidirectional data path



(b) Stage II bidirectional data path

Figure 3.1: Data Communication Paths

1. **Idea and Concept:** The project begins with a conceptual phase where the necessity for a user-friendly, versatile tool to connect various sensors with smartphones is identified. Initial ideas are refined through literature reviews and feasibility studies to establish a clear project scope and objectives. The target of adapting both passive and active sensors is developed in this phase.

2. **Design and Development:** The design and development phase involves meticulous planning and component selection. This process is based on the requirements identified in the conceptual phase and is informed by the theoretical insights gained from the literature review. The software architecture is designed with scalability in mind, allowing for future expansions and sensor integration.

3. **Implementation:** The implementation involves assembling the hardware according to schematics, programming the Arduino for sensor data management, and developing the Android application to facilitate user interaction via Bluetooth communication protocols.

4. **Testing and Validation:** Testing ensures the accurate performance of the sensor-smartphone interface. It involves verifying that the smartphone application correctly displays and reacts to data from the sensors. This stage confirms the system's functionality and reliability.

3.2 Components Used

In addition, a list of all components utilized in the project is also included in this chapter, accompanied by a detailed rationale for each selection. Each component is carefully chosen based on specific criteria such as compatibility, cost-effectiveness and ease of use, ensuring the approach aligns with the overarching project objective.

The components are categorized into three groups according to the three sectors of the route design: the smartphone sector, the microcontroller sector and the sensor sector. They will be introduced in this section in the same order.

3.2.1 Smartphone with Android System

The selection of the Android Operating System for this project is based on its accessibility and widespread adoption. The primary advantage is that it allows most developers to create applications on various platforms, providing a range of development environments and tools. In contrast, Apple's iOS requires a Mac computer for app development, limits the ease of app creation and sets a high barrier for beginners. Furthermore, the choice can be supported by research on market share [9] and preference [10] of the two systems. Both indicate Android was found to be more chosen globally than Apple, though the margin was insignificant and varied even within states of a country. The device used in the project is Xiaomi 13, operating on the Android 14 system.

3.2.2 Microcontroller: Arduino UNO R3 with Bluetooth

The Arduino board is a powerful and widely-chosen microcontroller for small projects. It contains a microprocessor to perform coded programs, a storage to keep the default program and a series of digital and analogue input/output (I/O) pins. All these reasons make it an ideal data transfer and handling platform for this project.

For ease of use, wireless communication methods are preferred. Three leading wireless technologies, including Bluetooth, Wi-Fi and Near Field Communication (NFC), have been investigated in depth and compared. Finally, Bluetooth technology is chosen for several reasons:

- **Ease of Integration:** Bluetooth modules are more accessible to be integrated with Arduino, and a vast amount of documentation and community support can be used as guidance for this project. Libraries within Arduino also help simplify the code required to establish and manage Bluetooth connections.

- **Accessibility:** As a universally adopted standard, Bluetooth is supported on virtually all modern smartphones. This widespread integration eliminates the need for additional peripherals or extensive setup processes, enabling straightforward connectivity.
- **Range and Capacity:** While not as far as Wi-Fi, Bluetooth provides a good connection range for on-field measurements. Its channel capacity is also sufficient for the real-time transmission needs of most sensors, aligning with the project concept of further extension in sensor models.

3.2.3 HC-05 Bluetooth Module

The Bluetooth module chosen to perform wireless communication between Arduino and smartphones is HC-05, presented in Figure 3.2. The HC-XX is an extensive series of Bluetooth modules; current on-market modules have HC-05 and HC-06. Comparing their difference, the project applied HC-05 for its ability to operate both as a master and as a slave device, which means it can both initiate a connection and accept connections without additional commends. At the same time, HC-06 only functions in slave mode.

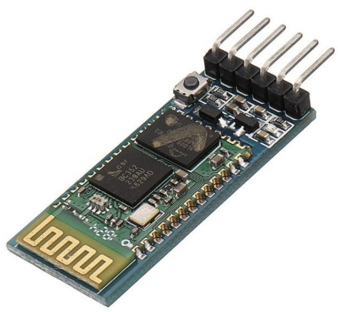


Figure 3.2: HC-05 Bluetooth module graph

3.2.4 Sensor I: DHT-11 Humidity and Temperature Sensor

In selecting an appropriate sensor for stage I of the system architecture, it is vital to consider the operational simplicity and the nature of the environmental parameters to be monitored. The

DHT-11 sensor, shown in Figure 3.3, emerges as an optimal choice because it is a composite sensor capable of measuring temperature and humidity. These two variables can be relatively easily controlled and calibrated under experimental conditions, facilitating subsequent testing and verification.

The DHT-11 sensor is an example of a large class of sensors that do not require excitation signals to initiate measurements. Such sensors are inherently designed for passive operation, continuously monitoring conditions. This characteristic is particularly beneficial in systems where constant, real-time monitoring is crucial. The lack of a necessary excitation phase simplifies such systems' circuit design and software logic.

Additionally, the DHT-11 is selected for its ease of interfacing with microcontrollers. Its digital output allows direct communication with the microcontroller without needing analog-to-digital conversion, further reducing system complexity. Moreover, the DHT-11 is known for its cost-effectiveness and accessibility, making it a popular choice for a proof-of-concept implementation where budget constraints and component availability are considered. Its widespread use and robust community support ensure sufficient resources for troubleshooting and integration into the proposed system architecture.

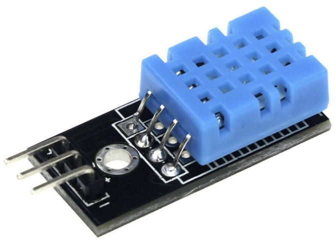
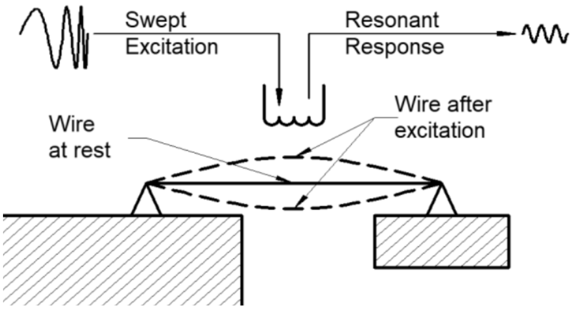


Figure 3.3: DHT-11 sensor graph [11]

3.2.5 Sensor II: Embedded Vibrating Wire Strain Gauge with VM501 Readout Unit

Many active sensors available on the market, such as capacitive pressure sensors, piezoelectric accelerometers, or resonant solid-state temperature sensors, are commonly sold in integrated packages with built-in excitation mechanisms. They are ready for direct use as they only need a power supply to operate, and the excitation component is already embedded within the chip. However, the project aims to achieve intervention of excitation signals from the smartphone interface, so the aforementioned sensors are rejected in research. Finally, for stage II of the



(a) Vibrating wire sensor internal structure [12]

(b) Embedded strain gauge image

Figure 3.4: Vibrating Wire Strain Gauge

project, a vibrating wire strain gauge in Figure 3.4 is selected to present a sensor that necessitates an excitation signal to operate.

The principle of a vibrating wire strain gauge lies in its ability to link external strain to the tension within a wire that is held taut. This tension alters the wire's fundamental natural frequency of vibration, similar to the varying pitches produced by guitar strings under different tensions. Through the readings of the wire's natural frequency, the external strain experienced by the strain gauge can be quantified. Similar to a guitar string, the taut wire in this sensor requires a frequency-swept magnetic field emitted by a built-in coil as an excitation signal to begin vibrating. Following the vibration, data-acquisition equipment measures the wire's natural frequency.

In addition to the strain gauge, a VM501 vibrating wire readout unit is applied. This unit is specifically designed to cater to the unique needs of controlled frequency-swept excitation. It encompasses functions for sensor excitation, signal detection, and data processing. Furthermore, it facilitates data exchange through digital pins, serving as a crucial conversion unit for digital communication between the vibrating wire sensors and the external devices [13]. In this project, it is applied in the connection between the strain gauge and the Arduino board.

3.2.6 Software: App Inventor Integrated Development Environment (IDE)

App Inventor is the tool for developing the app used in this project, chosen for its straight-forward visual programming environment. This platform is suitable for beginners to develop essential functions and robust enough to support complex applications required for interacting with hardware components such as sensors and microcontrollers. The platform also provides real-time feedback for quick design changes and immediate testing on Android devices, which is essential for development. Moreover, App Inventor's compatibility with Android's extensive function suite enables the project to integrate with Bluetooth seamlessly.

This project benefits from the platform's ability to circumvent the steep learning curve of traditional programming, promoting an efficient developmental cycle. As it simplifies the creation of the user interface and logic design, App Inventor allows the project to focus on the application's functionality and programming logic without getting mired in complex code syntax.

3.3 Conclusion

This chapter demonstrates that both unidirectional and bidirectional communication pathways could be effectively established and managed through the integration of a passive DHT-11 sensor and an active vibrating wire strain gauge. The future work for both sensors is listed as hardware connections followed by software Arduino and app programming and final comprehensive test. The selections of components are made, including the use of Arduino as the microcontroller, HC-05 to apply Bluetooth communication, the two sensor models and the use of App Inventor to develop the corresponding app. Rationales behind each decision are given in details to allow for a clear understanding of the methodology and its alignment with the project's goals.

Chapter 4

Stage I: Establishing a Unidirectional Communication

4.1 Aim of Stage I

In the first stage, design concepts will be applied, tested, and evaluated in actual experiments. Details of the entire project will be uncovered, and functional codes will be programmed. This stage is vital for establishing a solid foundation and understanding of the project's outcome, ensuring a smooth transition to the more complex second stage.

The sensor for stage I is a passive sensor without the need for excitation, meaning stage I will focus on receiving data at the smartphone interface. The whole design of the entire information system needs to address the following considerations:

- What power is required for each component, and where does that power come from?
- What is the expansion capability of each component?
- What are the typical scenarios for end-users?

4.2 Sensor

As mentioned in section 1.3.2, the sensor used for stage I is a DHT-11 temperature and humidity sensor. Delving into its structure, the DHT-11 humidity and temperature sensor is comprised of three primary elements as shown in Figure 4.1: a resistive humidity sensor for moisture detection, a negative-temperature-coefficient (NTC) thermistor for temperature measurement, and an 8-bit inherent microcontroller. This microcontroller converts the analogue signals from the humidity and temperature sensors into a single digital output. It has three pins that require connections, referring to Table 4.1.

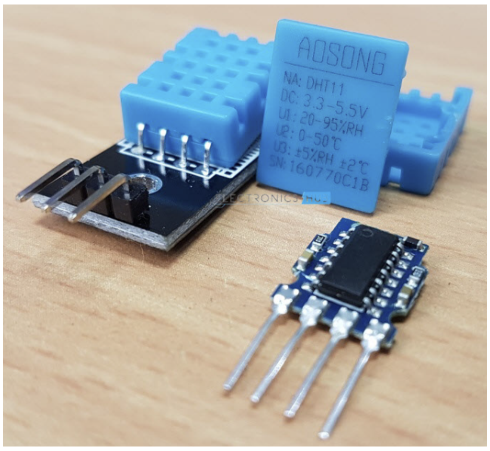


Figure 4.1: DHT-11 contains a humidity sensor, a thermistor and a microcontroller [14]

Table 4.1: Pin configuration of DHT-11 sensor

Pin No.	Pin Name	Pin Description
1	VCC	Power supply 3.3 to 5.5 Volt DC
2	DATA	Digital output pin
3	NC	Not in use
4	GND	Ground

DHT-11 uses the only digital pin for communication, which holds either VCC or GND voltage during communication. To implement this sensor in Arduino, it is necessary to see its communication process as three steps: measurement request (handshake), data transmission and measurement end.

The process in Figure 4.2 begins when the microcontroller sends a start pulse, pulling down the data pin 18ms and then up. In response, the DHT-11 sends a low signal for 54us and then a high signal for 80us to acknowledge receipt of the start pulse, demonstrating the success

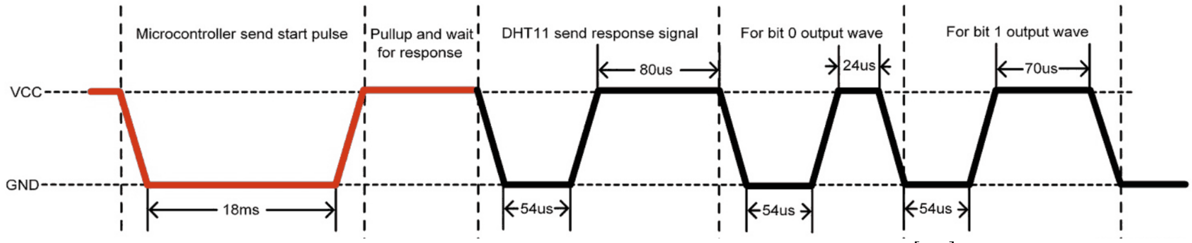


Figure 4.2: DHT-11 digital communication process [14]

of handshaking. Both temperature and humidity readings are transmitted with a 40-bit data frame of five segments: the first two for humidity (integer and decimal parts), the next two for temperature (also in integer and decimal parts), and the final segment for the checksum. The checksum is the binary sum of the humidity and temperature readings, and if it does not match the received checksum, an error is indicated. Outputs of 0 and 1 are different in time that the data pin stays at the high state, which is 24us and 70us, respectively. After the data frame transmission, DHT-11 concludes the communication cycle with a final low signal before it goes into sleep mode.

Corresponding to the aforementioned considerations, DHT-11 requires a power supply of 3.3V or 5.5V, recommended for using power pins from Arduino, as its digital data pin forces wired connections to microcontrollers. Therefore, the usage scenarios depend on the combination of microcontroller and sensor rather than the sensor alone. DHT-11 is compact and consumes a small amount of power. Suppose its paired microcontroller holds the same performance, the combination can be embedded at fixed locations, like inside an air conditioner, rather than just for remote usage of the external module of a measuring instrument.

4.3 Bluetooth

A quick view of the Bluetooth module is required to specify the port connection on Arduino. HC-05 has six pins but only pins 2 to 5 are connected as the project only uses this module in its default mode, as referred to Table 4.2.

Considering the points discussed previously, the HC-05 module needs a 5V power supply, and it is advisable to utilize the power pins from an Arduino for this purpose. Its RX (Receiver) and TX (Transmitter) pins necessitate wired connections to microcontrollers, making the setup and application scenarios similar to those outlined for the DHT-11 sensor.

Table 4.2: Pin configuration of HC-05 Bluetooth module

Pin	**Name**	**Description**
1	Enable	This pin is used to toggle between Data Mode (set low) and AT command mode (set high). By default, it is in Data mode
2	Vcc	Powers the module. Connects to +5V
3	Ground	Ground pin of module. Connects to ground.
4	TX	Transmits Serial Data. Everything received via Bluetooth will be given out by this pin as serial data.
5	RX	Receive Serial Data. Every serial data given to this pin will be broadcasted via Bluetooth
6	State	The state pin is connected to on board LED, it can be used as a feedback to check if Bluetooth is working properly.

4.4 Arduino

Arduino can provide corresponding ports for both DHT-11 and HC-05. While the power and data pins from DHT-11 are straightforward, as they are just constant voltage supply and pure digital signals, the RX and TX pins from HC-05 are more tricky. RX and TX pins necessitate defining a serial port rather than directly reading from the pins since serial communication involves more than changes in signal levels. Serial communication is a time-synchronized method requiring precise timing and protocol management, such as start bits, stop bits and parity bits. Merely reading the high and low levels from the pins is insufficient to fulfil the complex communication protocols of Bluetooth. Hence, a serial port containing RX and TX pins must be defined.

4.4.1 Route Layout and Arduino Connection

There are two types of serial ports that can be implemented on Arduino: hardware and software serial ports. A hardware serial port on a microcontroller like an Arduino is a physical interface that facilitates serial communication according to the Universal Asynchronous Receiver/Transmitter (UART) standard. On the other hand, Software serial is a method of UART communication in software, allowing digital pins on a microcontroller to be used for serial communication. Software serial on Arduino is achieved by simulating the timing and framing of a standard UART port in software, increasing the number of pins that can be used for serial communication at the cost of program storage space and dynamic memory.

There are only two hardware serial ports on Arduino Uno, one connected to the USB interface and another to pin 0 and 1. Therefore, to allow more sensors to be connected to the microcontroller and be consistent with the aim of being versatile and extendable, software serial ports are actively implemented using the "SoftwareSerial.h" library from Arduino. It allows serial communication on digital pins from pin 2 to 13. The complete pin map of Arduino Uno can be found in Figure 4.3.

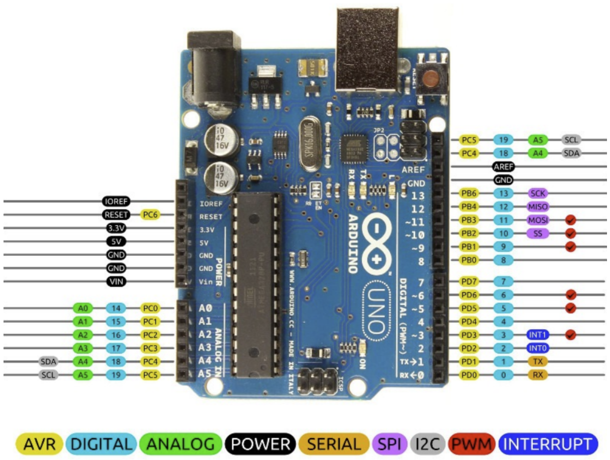


Figure 4.3: Arduino Uno pin map

The final connection of the project is displayed in Figure 4.4. However, DHT-11 contains several types that may have three pins and four pins (with one extra empty pin), depending on different manufacturers.

4.4.2 Arduino Programming

The above components must be configured in Arduino programming, and the logic must be laid out. For general system programming, serial port synchronization in baud rate is the fundamental base of correct communication without bad bits. The baud rate is set to 9600, which is adequate for most sensors to take measurements [6].

The project uses two serial ports: the USB hardware port and the software port of pins 12 and 13. The USB port is not involved in the actual communication with a smartphone, and there is no consequence of disabling or ignoring it; it exists just for the ease of programming. Pins 12 and

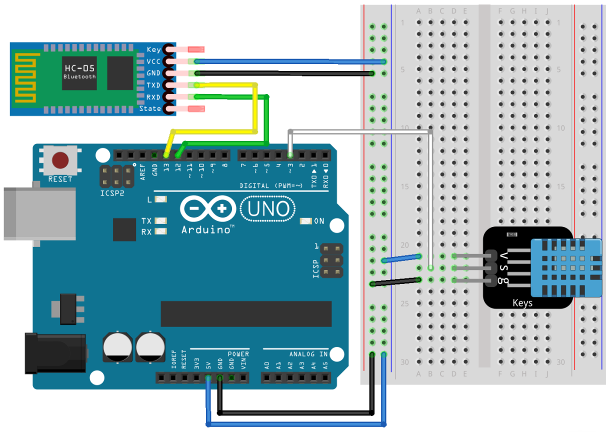


Figure 4.4: Stage I hardware connection

13 are used for HC-05 Bluetooth communication. As mentioned previously, the SoftwareSerial library needs to be included to achieve software serial communications, and a serial port of pins 12 and 13 needs to be defined.

Settings in the "Serial Monitor" tab in the Arduino IDE can activate the USB serial port for both writing and reading functions. DHT-11 outputs temperature and humidity readings as floating points, so two floating points are transmitted at each measurement, one for each. This configuration makes the maximum transmitting bits per second at 64, meaning the baud rate can transmit more information and leaves a possibility of extension or multiple sensor connections.

Coding logic follows the flowchart in Figure 4.5. Since the system does not have a sending function from the smartphone, the sign of start measurement composes power supply check and Bluetooth connecting check using *serial.available()* function. All the checking and initialization steps are done in the *setup()* function to save power and extend the two module's lifespan. Once Bluetooth is connected, the program enters the *loop()* function, where Arduino continuously reads from the sensor and sends it through Bluetooth. The loop stops when Arduino is powered off.

For reading from DHT-11, the "DHT.h" library is included in the code, which directly contains functions of *dht.readTemperature()* and *dht.readHumidity()*. Two float variables are defined, and values are assigned from these two functions for readings. However, for the Bluetooth output, after writing each float representing temperature or humidity, a sign of "a" is written to split numbers. For example, to send information "Temperature 27.5, Humidity 25", the actual output from Bluetooth is "27.50a25.00a".

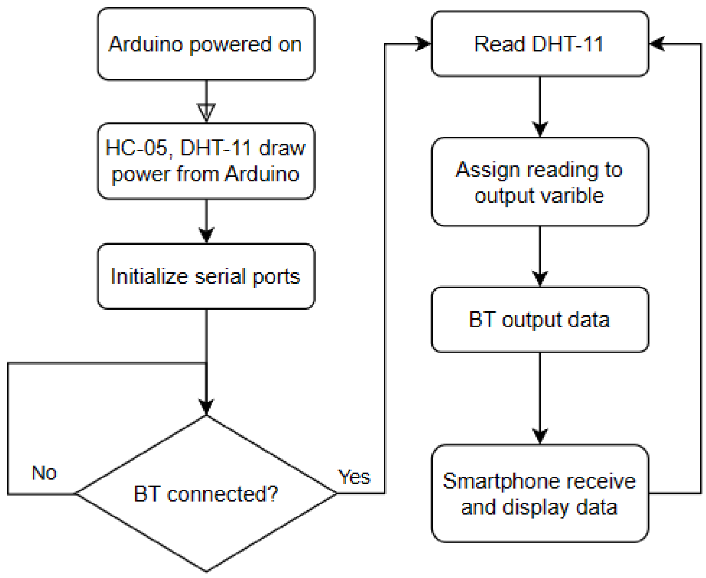


Figure 4.5: Arduino programming logic for stage I

4.4.3 Considerations in Common Scenarios

Arduino powers the whole system without needing an extra power supply. There are four ways to power an Arduino board:

- Constant 5V DC to USB port. This is commonly used for PC and laptop connection for programming.
- Constant 7V to 12V DC to Vin port.
- Constant 5V DC to the 5V pin. This is commonly used for household DC power supplies.
- Constant 9V to 12V DC to the power port, using a 5.5mm/2.1mm DC power connector.

The choice of power supply depends on the scenario. If any modification of the Arduino stored program is desired, a USB port is recommended as it can activate the serial monitor in Arduino IDE and allows the *serial.print()* function. For fixed usage, it can use both household DC power supplies and a 5.5mm/2.1mm DC power connector; however, stopping measurement at this stage would require manual unplugging. For remote use, like a readout instrument, it is recommended to use an On-The-Go (OTG) cable to convert Type-C to USB. Even long-time power consumption of DHT-11 and HC-05 is still within an acceptable range for modern smartphone batteries due to their efficiency. This allows an easier unplugging process and is more convenient as smartphones are intended to be used for the system.

4.5 Smartphone Interface

Smartphone interface is developed using App Inventor IDE, using block codes. Before the app's design, knowing and testing the information sent from the current functioning route is necessary. Without a self-designed app, this can be done by any Bluetooth analyzer app available online. A screenshot taken using an app called Bluetooth Master is shown in Figure 4.6. The test log shows the output data from the current circuit and programs work the desired way. However, it is worth noticing that if the read frequency of the app is too fast, it may break the information string into separate pieces, as the seconds of 22:07:55 and 22:07:56 show. This broken information serves as a warning to the app programming. An internal timer must be set to adjust the reading frequency to avoid collision with the time node of the Arduino transmitting data.

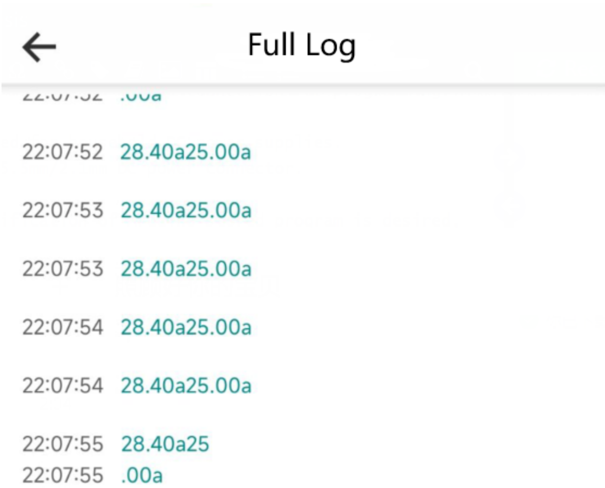


Figure 4.6: Test log from Bluetooth Master app

Components of the app include a user interface design, a Bluetooth client to activate the Bluetooth path, and an internal timer to control the frequency of reading updates. To minimize the aforementioned collision problem, Arduino outputs data every 1500ms, and the timer resets every 1010ms. Programming in App Inventor is visual block codes; the full logic is shown in Figure 4.7. It describes a logic sequence of:

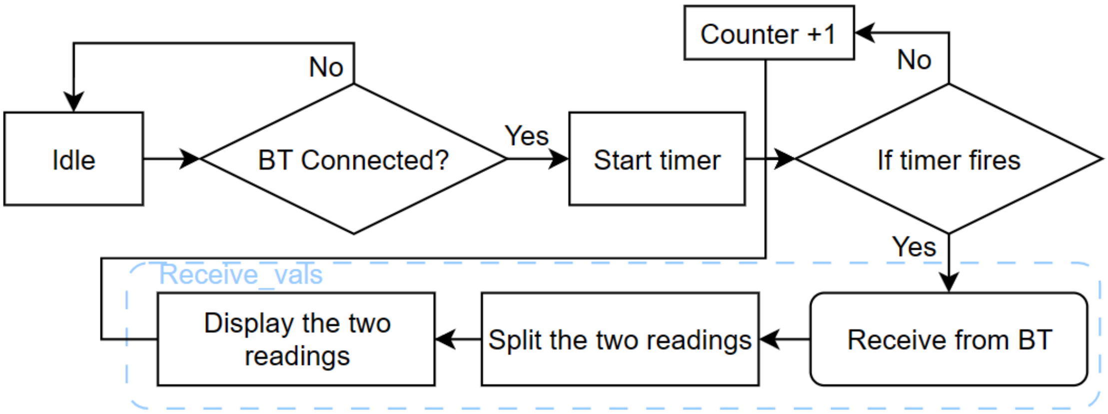


Figure 4.7: App programming logic for stage I

1. Initially, the app is in the idle state.
2. To activate Bluetooth, it lists of Bluetooth device names and addresses in a Bluetooth client. After a Bluetooth device is selected, it initiates a connection to the chosen device and enables a timer.
3. When the timer fires on each tick, it calls the "Receive_vals" function.
4. The "Receive_vals" function checks the data available from the Bluetooth connection.
   - If bytes are available, it reads the incoming text from Bluetooth path.
   - It then splits incoming data at the character "a", as programmed in Arduino.
   - It sets the text of "Temperature" to the first item of the split results.
   - It sets the text of "Humidity" to the second item of the split results.

4.6 Testing and Validation of Stage I

The testing process went smoothly with the previous accumulation of sufficient research, detailed analysis, and arrangement of programming logic. The readings were quickly available on the mobile app. Moreover, the values displayed on the smartphone interface could react accordingly to human intervention. For instance, covering the sensor with fingers could increase the temperature and humidity readings, while removing them could return the original readings. This phenomenon also proved that the number displayed on the smartphone app is the readings from the sensor.

However, the initial results seemed to be not accurate as it gives a humidity reading of 15% in a room, while the standard value should be within 20% to 50%. These unexpected findings emphasized the importance of calibration. To exclude uncontrollable factors in the room, subsequent tests were conducted outside the room to compare with official local weather reports, as shown in Figure 4.8. The validation experiment was conducted at 15:13 p.m., 14th April 2024, on the south side of Edinburgh. Five tests were conducted at the exact location. Each is spaced one hour apart to align with the update frequency of the local report. Full records is shown in Table 4.3.

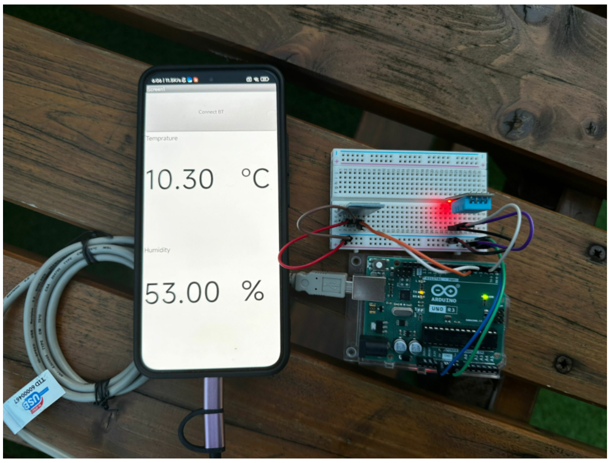


Figure 4.8: Stage I outdoor validation experiment

Table 4.3: Comparison between measured and official data

Test	Measured		Official		Error	
	Temp. °C	**Humi. %**	**Temp. °C**	**Humi. %**	**Temp. %**	**Humi. %**
1	10.3	53	10	54	3.0	1.9
2	10.5	55	10	56	5.0	1.8
3	9.4	56	9	56	4.4	0.0
4	9.3	62	9	60	3.3	3.3
5	8.2	61	8	60	2.5	1.7
Average					3.7	1.7

For temperature, the sensor exhibits a consistent trend of overestimation. The deviation ranges from 2.0% to 5.0%, with an average error of 3.7%. This consistent positive error may indicate a systematic bias with the test environment. The humidity measurements show a more varied pattern, with an average error of 1.7%. Test 3 aligns perfectly with the official data, indicating a 0% error, whereas Test 4 shows a more significant divergence of 3.3%.

A possible reason for the sensor overestimating temperature could be the location of the experiment, which in a depression surrounded by houses may inhibit heat dissipation, leading to higher measured temperatures. The combination of the terrain and buildings could create a microclimate effect, similar to an urban heat island, resulting in relatively higher temperature readings. Additionally, the use of heating appliances by individuals in the neighbourhood during the experiment could have further contributed to this outcome. As for humidity, the variations could be attributed to localized conditions at the measurement site, such as the presence of water sources or variations in airflow patterns, which are not represented in broader official reports.

However, the measurement results obtained from the DHT-11 sensor fall within the acceptable accuracy range for temperature and humidity, as specified by the manufacturer. The deviations observed align with the inherent precision level of the DHT-11, which is ±2°C for temperature and ±5% Relative Humidity (RH) for humidity. Given these parameters, the validation process can be considered complete and successful. The environmental factors contributing to the slight mismatch have been identified and are consistent with the expected sensor behaviours. Overall, the sensor's performance can offer satisfactory performance for real-life applications.

4.7 Summary of Stage I

In stage I of the thesis, the project focuses on the foundation of integrating sensors with smartphones. The stage begins with correct connections of a DHT-11 sensor and an HC-05 Bluetooth module to an Arduino board, followed by the development of both the Android platform and a corresponding Android app. Together, they enable communication from a DHT-11 sensor to a smartphone. The app's essential functions, including Bluetooth connection, receiving and visualizing the collected data, are established at this stage. Challenges were encountered during the testing, particularly concerning the validation of data, which means a more precise validation for stage II is preferred. The successful completion of the first phase laid the foundation for the project's second, more complex stage.

Chapter 5

Stage II: Establishing a Bidirectional Communication

5.1 Aim of Stage II

In the second stage, the goal is to extend the types of sensors and construct a new communication route from smartphones to sensors. Deep code related to sensors and microcontrollers is involved to achieve more detailed and precise control of the entire system. This stage intends to bring the whole project to a close, not only to develop and test new features but also to refine and add more details to the results already implemented in stage I.

The sensor for stage II is an active sensor which requires excitation. With the solid outcome from stage I, the work will focus on communication with a complex active sensor and sending data from a smartphone interface.

5.2 Sensor with Readout Unit

The sensor is an embedded vibrating wire strain gauge commonly used in geotechnical and structural engineering to measure strain within various structures and materials. This sensor is widely used for monitoring the structural health of bridges, dams, tunnels, and buildings, as well as for measuring rock and soil stresses in mines and for other applications.

A typical vibrating strain gauge only has two wires connected to the internal coil. Excitation and measurement signals share the same pair of wires. Since the signals are transmitted at different times, they will not collide.

Some parameters will be involved in this detailed analysis of the measurement process, which will be adjusted in subsequent experiments. To avoid confusing readers in the experiments, this paragraph will be accompanied by corresponding symbols when it comes to relevant parameters. For specific descriptions, please refer to Table 5.2 in section 5.2.3.

5.2.1 Sensor Excitation

Vibration process under excitation signals is shown in Figure 5.1. To ensure the vibration has stabilized, there is always a delay (RD_INTE) after excitation and before sampling a desired quantity (RD_COUNT). Specifically for embedded vibrating wire strain gauges, there are two options for excitation signals:

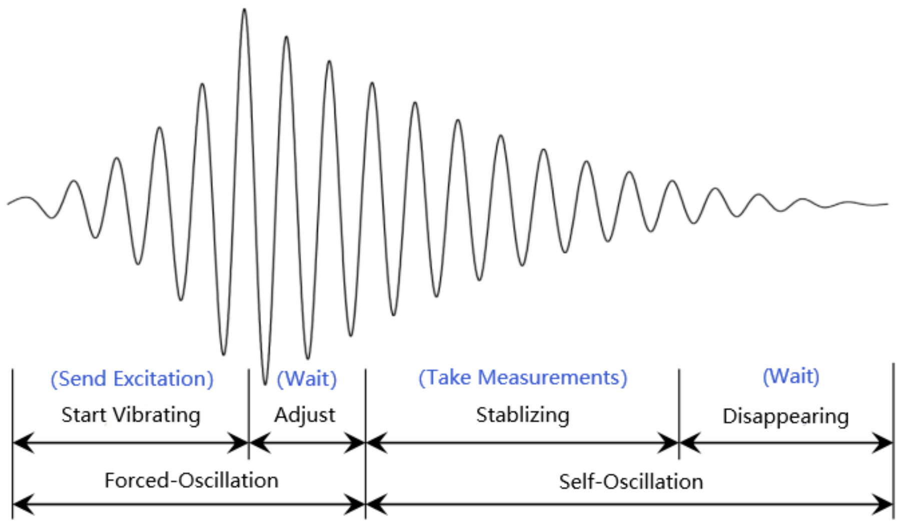


Figure 5.1: The vibration process of the taut wire inside the sensor

- **High-voltage pulse excitation:** a method that uses a high voltage of 100V to 200V, sends a short single pulse to the vibrating wire sensor coil and causes the vibrating wire sensor to produce self-oscillation at all frequencies.
- **Low-voltage sweeping excitation:** a method of sending continuous low-voltage of 3V

to 10V pulse train to the vibrating wire sensor using a frequency close to the sensor's natural frequency to cause self-oscillation of the sensor.

Projects built based on Arduino cannot hold a voltage of 100V. Thus, it is unrealistic to use the high-voltage excitation method. The project needs to apply the low-voltage sweeping method. Therefore, the excitation signal frequency selection and setting have become critical in ensuring the sensor's regular operation.

Delving into the low-voltage excitation pulses, there are two options for excitation:

1. **Stepped Sweeping:** a method of sending out several pulse trains, each containing a different frequency, as shown in Figure 5.2. Samples are taken from each pulse train and added together when all return signals have been received from all excitation pulse trains. In most cases, this method takes more time but gives an accurate reading.

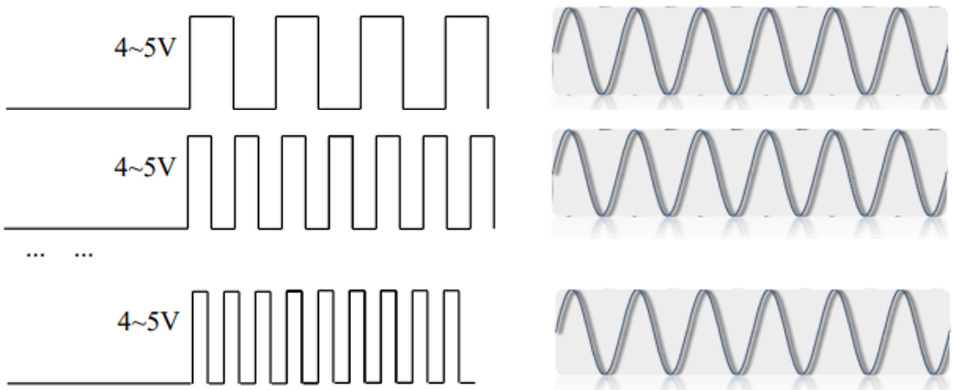


Figure 5.2: The stepped sweeping excitation method

2. **Progressive Sweeping:** a method of sending out one single pulse train containing all frequencies in the setting range, as shown in Figure 5.3. Samples can be taken and added directly after the reading, as there will only be one return signal. This method allows for the possibility of more frequent readings.

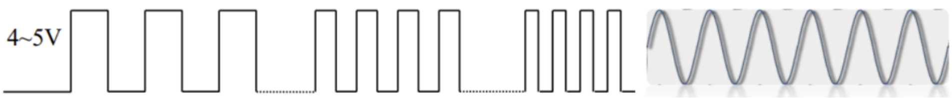


Figure 5.3: The progressive sweeping excitation method

Both methods have advantages, and deciding according to the actual use environment is always recommended. Therefore, this project will implement both methods, leaving the choice to the project's end-users.

5.2.2 Sensor Sampling and Processing Data

The taut steel wire in the sensor starts self-oscillation after receiving the excitation signal, cutting the sensor coil and generating a weak sinusoidal current in the coil. This sinusoidal current signal that changes with the vibration of the wire is the returned signal and will be sampled over time.

In order to process the signal, first, the weak current signal needs to be amplified to make it strong enough for subsequent processing. Secondly, to improve the signal quality, a filter is used to remove noise and non-correlated frequency components, retaining the core signal part that reflects the natural frequency of the steel string. Then, a frequency spectrum analysis is carried out with the processed signal, using spectrum analysis techniques such as fast Fourier transform (FFT) to convert the current signal in the time domain into a frequency domain signal. Signal in the frequency domain allows comparison with the amplitudes of the different frequency components, with the most significant amplitude peak corresponding to the natural frequency of the steel wire. Figure 5.4 shows a flowchart description.

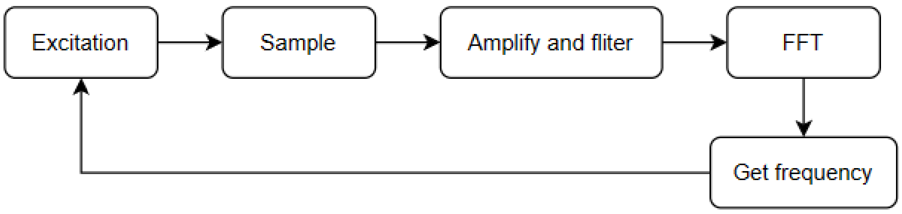


Figure 5.4: Vibrating wire strain gauge measure process

Two modes of measurement were developed for this project: automatic and manual. They are stored in a register located at 0x0A (EX_METH).

- **Automatic mode:** The automatic mode sends excitation skimming through a range of frequencies between 0Hz to 5000Hz, the limit of an embedded vibrating wire strain gauge, and gets a first reading out of this long-term process. Then, it uses the first reading as a base and quickly sweeps the frequency at 100Hz (FSG_TH) above and below the base value. Once there are three continuous measurements of 0, the system will do another all-frequency skimming to re-locate the base value.

Stepped sweeping excitation is used to give precise readings. The use of base values and limits reduces negative effect of slow measurement.

- **Manual mode:** The manual mode requires users to input their sweeping lower limit (FS_FMIN), up limit (FS_FMAX) and step (FS_STEP). Excitation continuously sweeps within that input range at the input step. The system does not react to continuous zero measurements when the natural frequency is out of the input range.

  Progressive sweeping excitation is used for this method to give rapid readings, as in common scenarios of using this mode, a rough range of the natural frequency is known in advance. Otherwise, it is recommended to use the automatic mode instead.

5.2.3 Readout Unit VM501

Due to the complexity of taking readings from the strain gauge, a corresponding unit VM501 is applied, shown in Figure 5.5. Used pins for this project have definitions presented in Table 5.1. For complete definitions, please refer to its datasheet [13]. VM501 is developed for single-coil vibrating wire sensors and can complete the process from generating excitation to taking readouts. It brings a working measurement process architecture. However, parameters involved in the process still requires manual settings.

Figure 5.5: VM501 unit pin map

VM501 stores the parameters within registers, and the working process entirely depends on the register (parameter) value. The register is an integer represented by hexadecimal address and is divided into readable&writable registers and read-only registers. These registers can be accessed and modified through the UART interface to achieve control and interaction with the unit and therefore the whole measurement process. The unit contains over eighty registers; each

Table 5.1: Unit register addresses and their functionalities.

Symbol	Pin	Description
S+	1	Positive pole of sensor coil
S-	2	Negative pole of sensor coil
SIG	5	Vibration signal quality
REF	6	Reference voltage, connected to VDD
VDD	8	Unit power supply 2.5 to 3.6V
GND	9	Connected to ground
TXD	12	UART transmitting pin
RXD	13	UART receiving pin
GND	19	Connected to ground
VSEN	20	Excitation power supply

is assigned a unique address. The project does not need to use all the registers; relevant ones to the project are shown in Table 5.2.

Table 5.2: Unit register addresses and their functionalities

Address	Symbol	Type	Description	Default	Unit
0x01 (1)	BAUD	Read/Write	UART baud rate	0x0060	
0x06 (6)	MM_INTE	Read/Write	Muti-measure interval	0x01F4	ms
0x08 (8)	RD_INTE	Read/Write	Sample delay	0x0064	ms
0x09 (9)	RD_COUNT	Read/Write	Sample quantity	0x14C8	
0x0A (10)	EX_METH	Read/Write	Excitation method	0x0064	
0x0F (15)	FS_FMIN	Read/Write	Sweeping low limit	0x012C	Hz
0x10 (16)	FS_FMAX	Read/Write	Sweeping high limit	0x1388	Hz
0x11 (17)	FS_STEP	Read/Write	Sweeping steps	0x0005	Hz
0x18 (24)	FSG_TH	Read/Write	Feedback method range	0x1414	Hz
0x20 (32)	SYS_STA	Read/Write	System status	0	
0x21 (33)	SFV	Read	Current sweeping frequency	0x0000	Hz
0x23 (35)	S_FRQ	Read	Frequency measured		Hz
0x28 (40)	V_SEN	Read	Current excitation voltage		V

5.3 Bluetooth

The Bluetooth connection part is the same as stage I, thanks to the HC-05 module that can be used as a master and a slave device. The relevant code will be explained in the Arduino section. For more details on Bluetooth, please refer to section 4.3.

5.4 Arduino

Stage II replaces the DHT-11 sensor with the VM501 unit, and a strain gauge is connected to the sensor terminals of the unit. A block diagram showing the connections is provided below in Figure 5.6.

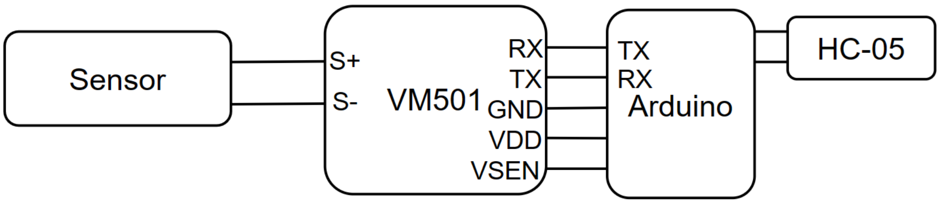


Figure 5.6: Block diagram of components

Arduino starts to receive orders from the smartphone interface through the Bluetooth module, using *Bluetooth.available()* and *Bluetooth.readStringUntil()* functions, which have been included in the "SoftwareSerial.h" library.

5.4.1 Route Layout and Arduino Connection

Routing for HC-05 and Arduino stays the same as that for stage I. VM501 routing is shown in Figure 5.7. This layout refers to the application circuit in the VM501 datasheet; it removes test pin ports for troubleshooting purposes, retaining only essential port connections. The unit requires the connections shown in the figure to work correctly. The addition of the three capacitors is used for different purposes:

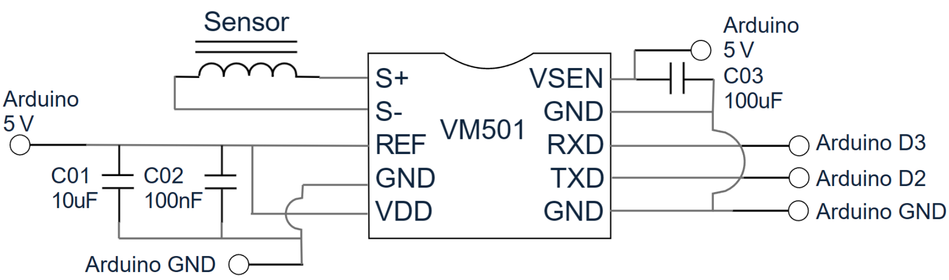


Figure 5.7: VM501 connections to Arduino and the sensor

- **Decoupling / Bypass Capacitors (C01 and C02):** C01 and C02 filter out noise from the Arduino power supply. They should be placed as close as possible to the power supply pins of VM501, providing a short path for high-frequency noise to return to the ground. 10uF and 100nF are used to filter different noise frequencies. The capacitor values are chosen based on the unit manufacturer's recommendation.

- **Bulk Capacitors (C03):** C03 is a larger capacitor acting as a charge reservoir to provide stable power to the unit during sudden changes in demand or supply. It helps to ensure that the voltage level remains constant if the power dips and rises briefly.

5.4.2 Arduino Communication with VM501

The communication protocol used for the interaction between Arduino and VM501 is a modified MODBUS protocol. Compared with the traditional MODBUS communication protocol, the structure is more straightforward, and its checksum bits are more accessible to generate, making it suitable for conducting rapid testing. Although only a single register can be read or written within one command simultaneously, multiple registers can still be accessed sequentially using multiple commands, which is suitable for this project. At the same time, it should be noted that VM501 also requires serial ports for communication, and a new software serial port named "mySerial" has been defined.

Comments are written in hexadecimal forms; when a correct command is received, the unit should return a response. Read and write commands and responses have structures listed below:

Reading command structure:

Head 0xAA 0xBB	**Unit Address**	**Register Address**	**Checksum**
2 bytes	1 byte	1 byte	1 byte

Reading response structure:

Head 0xAA 0xBB	**Unit Address**	**Register Address**	**Value**	**Checksum**
2 bytes	1 byte	1 byte	2 bytes	1 byte

Writing command structure:

Head 0xAA 0xBB	**Unit Address**	**Register Address|0x80**	**Value**	**Checksum**
2 bytes	1 byte	1 byte	2 bytes	1 byte

Writing response structure

Head 0xAA 0xBB	**Unit Address**	**Register Address**	**Value**	**Checksum**
2 bytes	1 byte	1 byte	2 bytes	1 byte

5.4.3 Arduino Programming

The codes written in Arduino follow the logic presented in Figure 5.8. To react to orders from smartphones, a *while (BT.available())* loop is used in the main *loop()* function. This while loop sets an indicator to "Set to manual mode", "Set to auto mode", "Start", or "Stop". A *switch (indicator)* function is followed to conduct the corresponding actions.

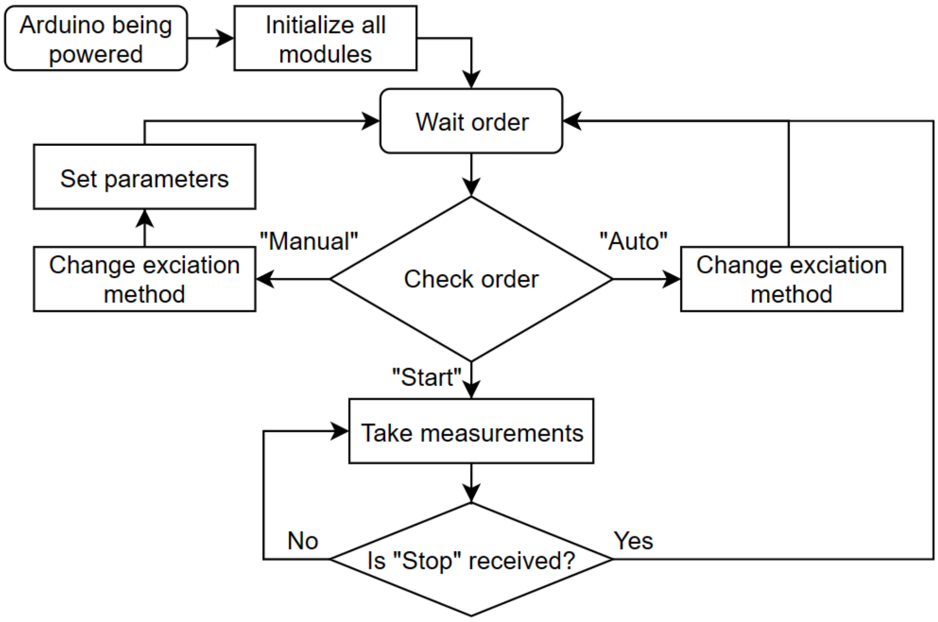


Figure 5.8: Arduino programming logic for stage II

- **"Set to Manual Mode":** Manual mode settings receive orders in the form of "C#Low limit#Step#Up limit" from the smartphone interface, which "C" is an indicator of "beginning manual mode setting". Then Arduino will send the write command to the VM501 unit, changing registers located at 0x0A (EX_METH for excitation method), 0x0F (FS_FMIN for low limit), 0x10 (FS_FMAX for up limit) and 0x11 (FS_STEP for step). The last bit "8" of register 0x0A (EX_METH) corresponds to this method. When dealing with one, the unit cannot receive and store a new command. Thus, each order is only sent when the last one gets its response from the unit. A *while (!mySerial.available())* loop is used after each sending order to stuck the program in a loop until it gets a response from the unit. It should also be noted that the communication between Arduino and VM501 is in hexadecimal codes, but Arduino receives decimal numbers for inputs from the smartphone.

Thus, a function is included in the program to identify the decimal number, convert it to hexadecimal, and write the high and low-order bytes to the corresponding positions in the command array.

- **"Set to Auto Mode":** Auto mode settings receive an order of "B" from the smartphone interface, which switches the system to auto mode. Arduino will send the write command to VM501 to change the value of the register located at 0x0A (EX_METH for excitation method). The last bit "4" of register 0x0A (EX_METH) indicates auto mode. The more detailed coding process was written inside the VM501 unit using their development IDE during the project.
- **"Start":** Start order is received as "D" from smartphone. The measurement is triggered by directly reading the frequency register (S_FRQ). A command of "0xAA 0xBB 0x01 0x23" is sent to the unit to read the register located at 0x23, the one storing calculated frequency readings. The command is written as a case under *switch(indicator)*, inside the *loop()* function, so it will be carried infinitely until the indicator changes.
- **"Stop":** The stop order is received as "E" from a smartphone; it sets the indicator to stop, waiting for new orders from a smartphone.

To give a complete demonstration, if the incoming order from the smartphone interface is "C#1500#15#2500", the Arduino program will first recognize "C" as the indicator of "manual mode setting" and send a command of "0xAA, 0xBB, 0x01, 0x0A | 0x80, 0x00, 0x68". The unit will return "0xAA, 0xBB, 0x01, 0x0A, 0x00, 0x68", showing this command has successfully set the register "EX_METH" located at 0x0A and controls excitation method to value "0x00 0x68", where the final bit "8" is the number representing "stepped sweeping method". Then the first number of the incoming string "1500" is extracted and converted to hexadecimal value "05DC", then a commanding setting the lower limit is sent as "0xAA, 0xBB, 0x01, 0x0F | 0x80, 0x05, 0xDC". The unit will return "0xAA, 0xBB, 0x01, 0x0F, 0x05, 0xDC", showing this command has set the register "FS_FMIN" located at 0x0F and controls lower frequency limit to value "0x05 0xDC", which is 1500Hz in decimal. The program will continue to do the same work for the frequency step and the up limit of the frequency.

It should be noted that all commands mentioned above are missing one byte of checksum, which should be included to complete the whole structure of a command. The checksum byte is added to the end of a command array using an *appendChecksum()* function written in the program, which adds up all bytes in the command array but only keeps the lowest byte as the checksum byte.

5.5 Smartphone Interface

The complex work done in Arduino programming saves a significant workload for developing smartphone software, but it is still a complex and effort-demanding task. With the DHT-11 sensor developed from stage I, two sensors will be available in the current app. It follows a logic referred to Figure 5.9. Moreover, line charts that update with incoming readings are drawn to improve the simple appearance of the previous app. The user interface of four pages' content is shown in Figure 5.10. The four pages contain the functions below:

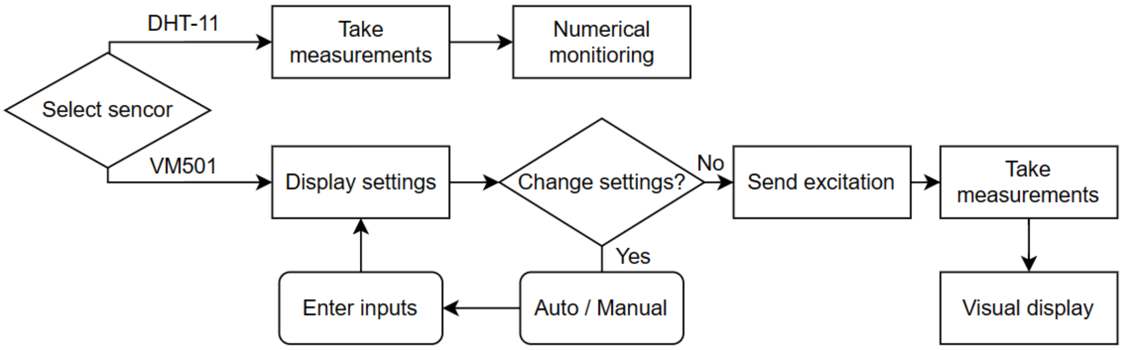


Figure 5.9: APP programming logic for stage II

- **Page 1 5.10a: Sensor Selection** The two sensors that were developed are available on this page. More sensors can be developed and added on this page as a separate tab for future extension possibilities.
- **Page 2 5.10b: DHT Readings** This is suppressed from stage I and adapted to the multi-page app architecture. It performs the same function shown in section 4.5.
- **Page 3 5.10c: Vibrating Wire Readings** This page shows the current excitation method for vibrating wire strain gauge and the relevant parameters when using the manual mode in the order of lower limit, step and up limit. It sends the "Start" and "Stop" orders to Arduino to change the indicator there.

  Readings are presented with values and line charts to show variation over time. The strain value is calculated from the received data.

  The strain ε is calculated as:

$$\varepsilon = K(F_i^2 - F_o^2) \tag{5.1}$$

(a) Page 1: Sensor selection **(b)** Page 2: DHT-11 readings **(c)** Page 3: Vibrating wire readings **(d)** Page 4: Vibrating wire settings

Figure 5.10: New smartphone user interfaces

Where:

$$\text{constant } K = 4.219 \times 10^{-4}\, \mu\varepsilon/\text{Hz}^2 \tag{5.2}$$

And:

$$F_i = 2108\text{Hz, the initial reading of the sensor under no pressure} \tag{5.3}$$

- **Page 4 5.10d: Vibrating Wire Settings** This page allows users to choose from the auto mode or manual mode of measurement and input the parameters for the manual mode.

The line charts are developed in local HTML files. They are uploaded to App Inventor as media and accessed using the WebViewer component. They are developed using Adobe Dreamweaver and can be modified according to needs using any HTML editor. Quick and simple changes can be carried in basic text editors like Notepad for Windows or TextEdit for macOS.

5.6 Testing and Debugging of Stage II

The testing process in stage II was not as smooth as before. The main reason was that the new addition of VM501 brought potential problems in programming. Since the initiation of the testing phase for this stage, three critical bugs have been identified and solved.

5.6.1 Failure Due to Unacknowledged Commands in Arduino to VM501 Communication

Based on the logic described in Arduino Programming section 5.4.3. There are several empty *while()* loops applied to stuck the code while waiting for a response from the unit. However, this method fails when Arduino successfully sends out the order, but VM501 does not receive it. This situation may happen because the unit cannot store order when dealing with a previous one; the new command will just be ignored. A timer is added to each empty *while(!mySerial.available)* loop to solve this issue. The timer will resend the order three times after a short delay if no response is received from VM501. After the command has been resent three times without receiving a response, the software will be forced to exit the loop and report an error.

5.6.2 Serial Port Information Confusion with Multiple Serial Connections

In practical tests, the system may fail to receive orders from HC-05 and VM501. The reason is that Arduino can create multiple software serial connections, but it can only listen to one software serial port at a time. In stage II, two software serial ports are defined for HC-05 and VM501, each with the duties of receiving and sending data. Thus, the *mySerial.listen()* function is used inside the software serial reading loops (the *while(mySerial.available)* loops). This function is part of the "SoftwareSerial.h" library, so it does not need extra definitions. It activates the specified software serial port and ensures that the correct port is actively listening for response and capable of processing incoming data.

5.6.3 Improvements and Alternatives to Perform Delay Function in Arduino

One final possible reason for not receiving the response from the unit and the Bluetooth is the choice of delay implementation. The *delay()* function is a simple way to implement a delay in

Arduino programs. However, it halts the processor on that line of code, blocking any other operations or responses, including reading sensors and handling communications. For this project, Arduino cannot be blocked because new commands may be transmitted at any time through the software serial ports. Thus, timers are used where delays are needed, as they do not block Arduino while tracking the time.

5.7 Validation of Readings with Digital Image Correlation (DIC)

After solving the abovementioned bugs, the system can present data on the smartphone interface. Under conditions without external pressure, the readings will fluctuate within a range of plus or minus 10Hz around the initial values of 2108Hz, indicating 0 strain. When pressure is applied, the frequency and strain will decrease or increase depending on whether the force on the sensor is compressive or tensile. This means that the received data indeed originates from the sensor. However, these readings still need to be verified to check if the data has been corrupted during transmission and processing and to ensure that they accurately reflect the physical deformation of the sensor.

5.7.1 Validation Background Research

Given the nature of the strain gauge's application, which often intersects with structural dynamics and integrity, the validation process requires a collaborative approach with the structural engineering discipline. People in the university's structural engineering departments offered valuable and crucial insights for designing the validation experiment. This collaborative experience incorporated the Digital Image Correlation (DIC) technique into the project.

Digital Image Correlation (DIC) is a non-contact optical method used to measure deformation in the strain of objects. The technique, particularly in its 2-dimensional form, is highly valued for its ability to measure strain across a surface during testing precisely.

5.7.2 DIC Validation Process

The analysis of images is done using software called VIC-2D from Correlated Solutions Inc. The experimental procedures are also referenced in their guide [15]. The whole process, along with measurements taken from the stage II system, is:

1. **Sensor Preparation:** The surface of the specimen is prepared with a random speckle pattern, which is essential for the technique. The pattern is formed by directly spraying white paint onto the black surface of the sensor.

   In order to ensure that its position does not change during the measurement process, the sensor is squeezed and then slowly released to make it rebound to the original value. Such a process gives a frequency curve that first decreases and then increases.

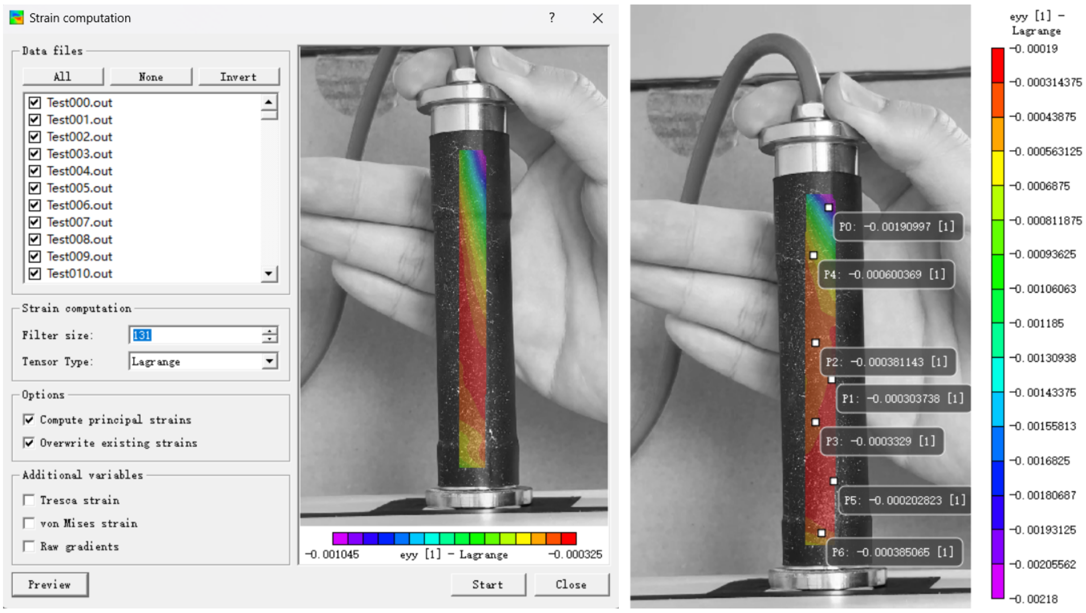
2. **Image Capture and Data measurement:** A video is recorded and converted to continuous images with fixed intervals for the specimen sensor while applying deformation. This is done using a high-resolution camera on another smartphone at a fixed position. While the images are being captured, the readings from the sensor are collected over time. Figure 5.11 shows the whole experiment setup.

Figure 5.11: DIC validation experiment setup

3. **Image Analysis:** The core of DIC lies in image analysis. The software VIC-2D compares the reference image (before deformation) with subsequent images taken as the specimen is deforming. The software tracks the displacement of the speckle patterns across the images.

4. **Strain Calculation:** The strain of the length on the sensor's surface is calculated from the amount of displacement. The software provides a detailed strain map, showing how it varies over samples. As an illustration, a single map analysis is shown in Figure 5.12, where *eyy* is the strain in the vertical direction. The image of the example is taken when the frequency drops to one of its lowest values.

5. **Comparison with Measurements:** The calculated strain is compared to the readings drawn from the sensor system. These data are imported to Matlab to generate the following



(a) Strain calculations over sample images on a decreasing trend

(b) Strain map for the Lowest frequency moment

Figure 5.12: Strain analysis using VIC-2D

comparison chart in Figure 5.13. If the values from DIC validation and direct measurement match, the data transmitted from the sensor to the smartphone system is proven not corrupted and can accurately reflect the sensor's physical deformation.

5.7.3 Validation Results

The deformation experiment, conducted over a span of 109 seconds, provided an extensive dataset comprising 48 frequency recordings from the sensor and 198 images processed via the DIC tool. The result shows a strong correlation between the DIC-analysed strain and the strain as measured by the sensor-smartphone system.

The measured strain data shows a clear vibrating motion along the actual deformation change captured by DIC technology. This is due to the sensor's principle that the taut wire inside always tries to follow the deformation trend. This phenomenon in a continuously changing experiment leads to fluctuations around the deformation line. In real application, this strain gauge is embedded in concrete, which has a steady changing trend and ends at a stable pressure level. Measurements in reality will be much more accurate than this validation experiment.

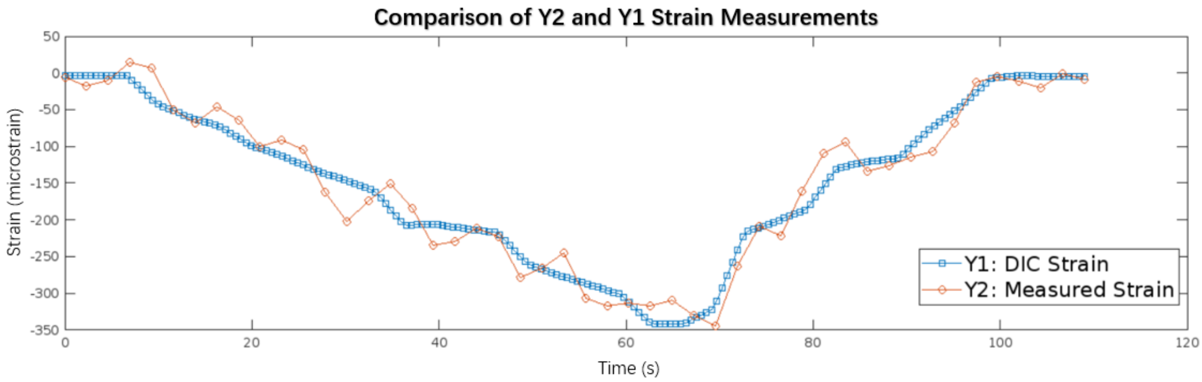


Figure 5.13: Sensor-smartphone readings and VIC-2D strain comparison

However, to quantify the accuracy of the sensor-smartphone system based on the measurements, understanding the nature of the error is key. In this case, the Mean Absolute Error (MAE) will tell the average magnitude of differences between the two signals, whereas the standard deviation of the error will tell how those errors are distributed around the average. The strain gauge in this case has a fixed, systematic error due to fluctuations; that error will be included in the MAE because MAE is only concerned with the size of the error, not its direction or distribution. On the other hand, the standard deviation, reflecting the consistency of the error, tells how the errors are fluctuating around their average value (MAE). A low standard deviation of the error means that, despite there possibly being a constant bias caused by the nature of the sensor, the sensor-smartphone path is delivering consistent readings. Thus, the standard deviation can better reflect the accuracy of the communication path. The equations for the calculation are provided below.

$$MAE = \frac{1}{n}\sum_{i=1}^{n} |y_{1i} - y_{2i}| \tag{5.4}$$

$$\sigma_{\text{error}} = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n} (|y_{1i} - y_{2i}| - MAE)^2} \tag{5.5}$$

Where:

$$y_{1i}, y_{2i} \text{ represents the values from the DIC Strain and the Measured Strain, respectively} \tag{5.6}$$

To apply these equations, the 48 measured strain data points need to be oversampled to match the quantity of the DIC strain. This task is done using the Matlab *resample()* function. The oversampled data is shown in Figure 5.14. Thus, MAE is calculated as 18.18 microstrain using equation 5.4, and the standard deviation is calculated as 14.01 microstrain using equation 5.5. These results give a coefficient of variation (standard deviation divided by mean) of 9.6%, indicating that the readings from the sensor-smartphone measurement are consistent.

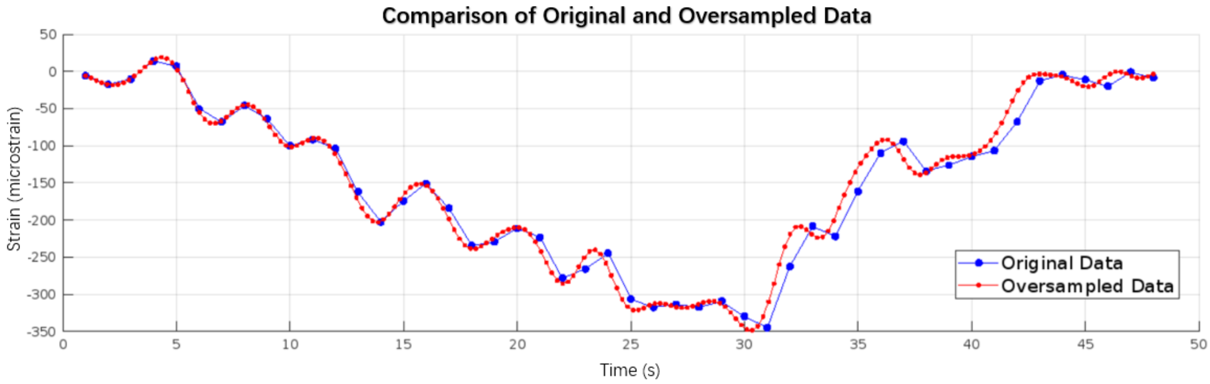


Figure 5.14: Oversampled data of measured strain set

In conclusion, the validation result suggests high accuracy in the sensor-smartphone system measurements. It indicates that data is not corrupted or damaged through the path, and a successful validation shows its precision in capturing deformation.

5.8 Summary of Stage II

In stage II of the thesis, the project delves into a more complex sensor type. It effectively accomplished a new path of sending information from smartphones to sensors throughout the process. Implementation of a vibrating wire strain gauge with the VM501 unit not only marks the adaptability of this project to other vibrating wire sensors but also indicates the potential for more active requirements that require excitation.

During this stage, as the sensor has become more complex, many issues are encountered and solved along Arduino programming and smartphone app development. They have been recorded, which can be used as a guide for anyone who wishes to continue working on this project.

A precise validation is carried out for this stage, the results of it has become solid evidence to indicate the success of stage II, which concludes the whole project.

Chapter 6

Impact and Exploitation

6.1 Product Development Cycle Analysis

This project sits at the beginning of a typical product development cycle. It starts with a concept and evolves into a functional model. During this project, concepts were embodied in the actual implementation process of two sensors that represent passive and active types of sensors; functions have been iterated from unidirectional communication to bidirectional communication with a visual presentation on a smartphone interface. However, upon completing this project, there are still steps to take to become a commercial product from a product's development cycle.

A thorough market research must be undertaken to transition the project from its current progress to a commercial state. This research should identify the potential market size and the target audience that could benefit and be attracted by integrating peripheral sensors with smartphones. The IoT devices market has been projected to expand significantly over the past decade, driving a surging demand for sensor integration technologies. The method proposed in this thesis is well-positioned to tap into these trends.

From a product perspective, the microcontroller part and Bluetooth module will be packaged into a device, with external interfaces reserved for connecting various sensors. The app will become an accessory of the device and be sold together. The device will be battery-powered for portability reason, eliminating the need for a power supply cable. All of these require extra designing and testing work as prerequisites for commercialization.

However, to become a viable commercial product, it is necessary to consider and compare the cost and benefit of implementing such a device. This device is regarded as a comprehensive alternative for many sensor readout instruments currently on the market. Thus, it needs to answer the question of how many sensors it can be adapted to and how many instruments it can replace. The project uses two sensors to represent its ability to adapt passive and active sensor categories. However, commercialization would require more detailed research on this matter, possibly to be detailed to sensor models. On the other hand, cost should also be controlled, which may require scalable production strategies for such a device. For instance, the device might be designed for smaller or larger models according to the number of their external interfaces, which decides the number of sensors that can be connected at a time and impacts the possible user of each model. To this end, forming strategic partnerships with established technology firms and manufacturers will also be crucial to handle mass production and distribution. These collaborations could also aid in the technology supply chain and quality control.

6.2 Societal Impact

The societal benefits of integrating peripheral sensors with smartphones are considerable and diverse. This project enhances public safety by enabling more effective monitoring of civil infrastructures such as bridges and buildings. Early detection of structural faults through sophisticated sensor technologies prevents accidents and increases the lifespan of public assets, thereby contributing significantly to the safety of human lives.

The adaptability and ease of use of the sensor integration technology developed here open up its application to various sectors including environmental monitoring and healthcare. For instance, in environmental monitoring, these sensors can be used for tracking air quality or water pollution levels, providing data that can lead to better-informed decisions and policies. In healthcare, portable devices equipped with these sensors can monitor patient health metrics in real time, offering improvements in preventive care and reducing the need for hospital visits.

The project's implications extend to enhancing public trust in technology and engineering. By demonstrating how technological advancements can be directly applied to solve real-world problems, the project fosters a deeper understanding and appreciation of engineering solutions among the general population. This integration not only showcases the practical applications of engineering research but also promotes a culture of innovation and responsiveness to societal needs, setting a precedent for future technological developments.

6.3 Economic Impact

The integration of peripheral sensors with smartphones, as explored in this project, is poised to drive substantial economic growth within the sensor industry. This technology enables a wide array of industrial applications, thus increasing demand for advanced sensors. According to market forecasts, the industrial sensors market is projected to grow from USD 27.9 billion in 2024 to USD 42.1 billion by 2029 [16], exhibiting a robust compound annual growth rate (CAGR) of 8.5%. This growth is attributed to the increasing adoption of these sensors across various sectors, facilitated by the advancements made in this project.

Particularly, the project's focus on the integration of vibrating wire strain gauges for structural health monitoring in civil engineering is noteworthy. The market for such applications is expected to expand from USD 1.9 billion in 2022 to around USD 4.0 billion by 2028. This surge is driven by the growing need for automated and precise monitoring systems that ensure the longevity and safety of infrastructure.

Manufacturers of sensor technologies stand to benefit significantly from this trend. As they capitalize on the expanding market, they can expect not only increased sales but also opportunities for innovation in product development. Additionally, the broader economic impact includes job creation in manufacturing and research and development sectors, spurred by the heightened demand for sophisticated sensor solutions. Ultimately, the economic potential of this project extends beyond direct financial gains. They catalyze broader industrial advancements and contribute to a cycle of innovation and economic vitality in the high-tech sector.

6.4 Knowledge Impact

The development and integration of peripheral sensors with smartphones in this project contribute significantly to the knowledge base in both the fields of sensor technology and smart device application. This research facilitates the generation of new insights into how different sensors can be effectively combined with consumer electronics to enhance their functionality and application range.

The project advances understanding in the area of sensor technology by demonstrating practical implementations of peripheral sensor integration. This includes novel approaches to data transmission, sensor calibration, and real-time data processing, which are critical for real-world applications. By documenting these methods and their effectiveness, the project serves as a valuable resource for other researchers and developers in the field.

On the other hand, the project explores the impact of these technologies on user interaction and usability. Insights into how users can seamlessly interact with these integrated systems contribute to the broader field of human-computer interaction (HCI). This research helps in designing more intuitive and user-friendly interfaces that can accommodate a diverse range of sensor inputs, thereby enhancing user experience and accessibility.

In addition to practical applications, this project also stimulates academic inquiry and curriculum development. By bridging gaps between theoretical knowledge and practical implementation, it provides a framework for future educational programs that aim to equip students with the skills necessary to innovate and apply sensor technologies in various industries.

6.5 People Impact

This project not only requires the application of advanced technical skills but also fosters the development of new competencies that are essential in the modern workforce. This technology pushes individuals to acquire and refine skills in areas such as data analytics, system integration, and software development. The necessity to handle complex datasets and manage interconnected systems encourages a deeper understanding of data science and information technology among users. As these skills are highly transferable and increasingly in demand across various sectors, participants in this project gain a competitive edge in the job market.

The project's implications extend into educational settings, influencing curriculum development at educational institutions. By demonstrating the practical applications of sensor technologies, it provides a compelling case for the inclusion of more comprehensive, hands-on training in STEM (Science, Technology, Engineering, and Mathematics) education. This prepares students not just for theoretical understanding but for practical, problem-solving skills that are directly applicable in real-world settings.

Importantly, by lowering barriers to entry and providing tools that are more accessible to a diverse population, the project promotes inclusivity in the field of technology. This inclusivity ensures that people from various backgrounds have the opportunity to contribute to and benefit from advances in technology, enriching the talent pool with a wide range of perspectives and fostering a culture of diversity and innovation.

Chapter 7

Improvements and Future Work

7.1 Expansion of Sensor Types

At this stage, the project provides a systematic and functional route that is able to link a smartphone device with external sensors through Bluetooth communication. The current system demonstrates a robust integration with the DHT-11 sensor and a strain gauge, as passive and active sensor examples, respectively, to cover temperature, humidity, and strain measurements. Future iterations of this project could greatly benefit from incorporating a broader spectrum of sensor types, such as light intensity, air quality, or sound levels. This expansion would enhance the system's utility across different environmental conditions and cater to a broader range of applications, from smart home systems to industrial monitoring. Furthermore, including sensors with varying operational complexities, such as those requiring different excitation mechanisms or having higher precision, will test and potentially improve the versatility of the communication interface designed.

7.2 Enhanced Data Transmission Techniques

While Bluetooth provides a convenient method for short-range communication, there may be better choices for some applications, especially those requiring higher data throughput or extended range. Exploring alternative data transmission technologies like Wi-Fi or Zigbee could address these limitations. Wi-Fi would enable the integration of the system into IoT networks with existing infrastructure, allowing for greater scalability and remote monitoring capabilities. Additionally, employing a mesh network protocol like Zigbee could enhance the system's robustness and reliability in industrial environments where obstacles or long distances can impede Bluetooth signals.

7.3 Cross-Platform Compatibility

The system currently utilizes an Android-based application to interface with the sensors. To increase the accessibility and user base of our technology, developing a cross-platform application that also supports iOS and desktop (Windows, Linux, MacOS) systems would be beneficial. This approach ensures that users are not limited by device type and can seamlessly integrate this technology into diverse technological systems. Moreover, compatibility with proprietary industrial platforms is also considered to facilitate the adoption of this system in specialized fields that rely on specific software environments.

7.4 Conclusion

By addressing these critical areas for improvement, the project can move towards a more inclusive, versatile, and robust system. Each step forward helps this project reach its full potential of integrating smartphones with peripheral sensors, making advanced monitoring and data analysis tools more accessible to all.

Chapter 8

Conclusion

The smartphone has become a highly functional and expandable comprehensive platform equipped with various components. It is widely used and is changing people's habits. People are getting used to operating various things and devices on their smartphones. Therefore, this project leverages its connectivity capabilities to interface with peripheral sensors that can not be integrated into smartphones. Based on the work done for this project, an example route has been created, tested and validated.

In this project, the connection between the smartphone and sensors is achieved with a microcontroller, which handles data transmission, excitation creation, and power supply. The creation of a communication path is divided into two stages: stage I for a unidirectional path from the sensor to the smartphone interface and stage II for a bidirectional path between them.

In the first stage, the DHT-11 temperature and humidity sensor was used in the design, serving as an example of a simple type of passive sensor. It helped establish the basic functionality and features of the communication path, including the code for the app to receive data and Bluetooth communication. In the validation experiment of stage I, the data received by the system was used to compare with official local weather reports, ultimately verifying its accuracy and stability.

For the second stage, a vibrating wire strain gauge is an example of an active sensor requiring complex excitation. With this new requirement proposed by the sensor, the communication path is developed into a bidirectional one that allows the smartphone to send information out and influence the sensor. Accordingly, the app has been redesigned to accommodate this feature and improved data visualisation by adding line charts to show the variation trends of readings. Validation of this strain gauge applied the Digital Image Correlation (DIC) technique using VIC-2D, which directly analysed the photos taken of the sensor as a standard to measure the overall error of readings.

The project built a prototype for demonstration with low errors of 3.7% for temperature, 1.7% for humidity and 9.6% for strain measurements. Necessary steps to complete a product development cycle and possible improvements for technical aspects are documented in chapters 6 and 7, respectively.

Acknowledgements

This project spanned an entire year and was the most complex, effort-demanding, and challenging project I undertook during my undergraduate studies at the University of Edinburgh, but it remains my favorite. I was fortunate and honored to have proactively contacted my supervisor, Dr. Jiabin Jia, at the start of my senior year to secure this opportunity. Dr. Jia held weekly meetings with me to discuss my progress and provided all the necessary support when I encountered challenges. He also proposed the crucial step of validation and reviewed my draft for thesis, which helped me perfect the structure of the complete report.

I would also like to express my gratitude to my examiner, Prof. John Thompson. Although we met only once during the second project presentation, his kindness and interest in my project, the questions he raised during the QA session, and the feedback he provided made me reconsider how to present each detail and decision made on each step more clearly and accurately in my thesis, and better prepared me for the final viva-voce presentation.

During the validation process of the second stage, Dr. Thomas Reynolds and Dr. Hwa Kian Chai from the Civil Engineering Department provided invaluable insights and assisted in designing various experiment methods. They also guided me on using the VIC-2D software for Digital Image Correlation analysis of images. Without their help, validating the readings from the vibrating wire strain gauge would have been nearly impossible.

I am also thankful to three other individuals who greatly assisted me during my undergraduate years: Prof. Wasiu O Popoola, my lecturer for Analogue Mixed Signal Laboratory 3 course, two lab technicians Mr. Ali Christie and Mr. Iain Gold in the electronic lab. Although not directly involved in this project, they taught me how to use essential equipment, such as oscilloscopes and soldering stations, in other classes and provided theoretical knowledge relevant to this project.

Lastly, I want to thank my parents and my girlfriend. My parents give me the opportunity to study at the University of Edinburgh. I am also very fortunate to have been born into a harmonious family, as my parents have supported me unconditionally in all significant life decisions. The third and fourth years of undergraduate study were tough and stressful, my girlfriend has always given me necessary emotional support and encouragement. Without her, these two years would have been painful for me. Despite being far from home, I never lacked a sense of being loved because of them.

References

[1] Cheng, S., Gu, Z., Zhou, L. *et al.*, 'Recent progress in intelligent wearable sensors for health monitoring and wound healing based on biofluids,' *Frontiers in Bioengineering and Biotechnology*, vol. 9, no. 12, pp. 5–6, 2021. DOI: 10.3389/fbioe.2021.765987.

[2] Beduk, T., Beduk, D., Hasan, M. *et al.*, 'Recent progress in intelligent wearable sensors for health monitoring and wound healing based on biofluids,' *Biosensors*, vol. 12, no. 8, pp. 1–2, 2022. DOI: 10.3390/bios12080583.

[3] Tri, M. T. P. and Jinqiu, Y., 'Exploring bluetooth communication protocols in internet-of-things software development,' *2020 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, vol. 12, no. 8, pp. 792–793, 2020. DOI: 10.1109/ICSME46990.2020.00093.

[4] Gore, R. N., Kour, H., Gandhi, M., Tandur, D. and Varghese, A.: 'Bluetooth based sensor monitoring in industrial iot plants,' in *2019 International Conference on Data Science and Communication (IconDSC)*, pp. 1–6. DOI: 10.1109/IconDSC.2019.8816906.

[5] Massimo, B. and Michael, S.: 'Getting Started with Arduino: The Open Source Electronics Prototyping Platform'. (Maker Media, Inc., 2014).

[6] Göpel, W., Hesse, J. and Zemel, J.: 'Sensors: A comprehensive survey', vol. 8, pp. 1–551. DOI: 10.1002/9783527620180.

[7] Xu, C., Luo, D. and Chang, Y., 'Research on feasibility of vibrating wire sensor in dynamic testing,' vol. 36, pp. 107–110, Feb. 2012. DOI: 10.3963/j.issn.1006-2823.2012.01.025.

[8] Shui, B.: 'Design and implementation of cost-efficient analog-to-digital conversion circuits for sensor applications,' Ph.D. dissertation, Jan. 2022.

[9] Statcounter. 'Mobile operating system market share worldwide', Statcounter. (2024), https://gs.statcounter.com/os-market-share/mobile/worldwide, accessed Apr. 2024. Open Source.

[10] Navarro, R. 'Android vs apple phones: Which does the world love most?' ElectronicsHub. (2024), https://www.electronicshub.org/android-vs-apple-which-does-the-world-love-most/, accessed Apr. 2024. Open Source.

[11] AZDelivery. 'Azdelivery dht11 temperature and relative humidity sensor module including e-book', Product image retrieved from Amazon UK website. (2023), https://www.amazon.co.uk/AZDelivery-Temperature-Relative-Humidity-Including/dp/B089W8DB5P?th=1.

[12] SCIENTIFIC, C. 'Vibrating wire sensors and their measurement', CAMPBELL SCIENTIFIC. (2024), https://www.campbellsci.com/vspect/principle.

[13] winkooo, H. 'Vibrating wire readout unit.vm5x1/6x4/6x8/704/s datasheet', Hebei WINCOM Technology Co., Ltd. (Oct. 2022), http://www.winkooo.com/uploads/ueditor/php/upload/file/20221014/1665712709671822.pdf, accessed Apr. 2024. Open Source.

[14] Anusha. 'Dht11 humidity sensor on arduino', ElectronicsHub. (2024), https://www.electronicshub.org/dht11-humidity-sensor-arduino/.

[15] 'Vic-2d testing guide', Correlated Solutions. (18th Mar. 2024), https://www.wiki.ed.ac.uk/download/attachments/158708302/VIC-2D_Testing_Guide.pdf?version=1&modificationDate=1686655452000&api=v2, accessed 16th Apr. 2024. Open Source.

[16] Dhapte, A.: 'Sensor Market Research Report Information By Type - Industry Forecast Till 2030'. (Market Reseach Future, 2020).