POLITECNICO DI TORINO

Master's Degree in Mechatronic Engineering

Master's Degree Thesis

Multimodal Haptic Feedback in a Soft Robotic Ball for Therapeutic Human–Robot Interaction (MR2S)

Supervisors

Prof. PANGCHENG David Cen Cheng

Co-Supervisor

Prof. Joffrey BECKER

Candidate

Mohamad Jawad NAIM

July, 2025

Abstract

As part of the MR2S project, this research explores the development of robotic systems intended for therapeutic interaction in healthcare, with a particular focus on autism support. In such contexts, soft and responsive haptic devices offer a compelling non-verbal channel to promote emotional engagement and sensory regulation. The thesis presents the development of a tactile interaction system that responds to varying levels of physical input using capacitive sensing and multimodal feedback. The goal was to create a clear and compact platform capable of detecting graded touch and translating it into both vibration and mechanical motion responses. The sensing was handled by a soft resistive-capacitive material, able to reflect different pressure levels through measurable resistance changes. Five interaction categories were defined (no touch, soft touch, medium touch, hard touch, grab touch) and each was mapped to a corresponding response in vibration motors and servo-based motion. The Arduino Mega 2560 was selected as the most suitable controller because of its wide pin availability and steady analog input handling. Each function of the system including sensor reading, classification, vibration triggering, and servo actuation was first developed and tested independently, before being brought into a unified loop. The logic used was deliberately simple, relying on threshold comparisons and step-wise control to maintain transparency and responsiveness. Python scripts were used to visualize sensor behavior, refine thresholds, and validate the classification process with greater flexibility than the standard Arduino tools. The resulting implementation showed reliable, repeatable feedback patterns across all defined interaction levels. The system's structure allows for easy adaptation, and its components work together to illustrate how simple, soft sensors can drive multimodal feedback in real time. This platform may support future applications in wearable feedback, soft robotics, or therapeutic devices requiring low-latency tactile response.

Acknowledgements

I would like to express my heartfelt gratitude to my thesis supervisor Professor David Cen Cheng PANGCHENG for fostering an academic environment in the Mechatronics department that has significantly contributed to my growth throughout this journey. His guidance have played a central role in shaping my professional development during this thesis.

I am thankful for the opportunity to be a part of the Erasmus+ exchange program at École Nationale Supérieure de l'Électronique et de ses Applications in France, which was a enriching experience in my academic life. The collaborative research at ETIS laboratory between ENSEA and CY Tech allowed me to explore a challenging environment where engineering skills meets social impact. This experience widened both my technical and cultural knowledge.

I would also like to express my sincere gratitude to the MR2S project team, in particular to Dr. Mathilda GAULARD, Professor Joffrey BECKER and Dr. Mehdi ABDELWAHED, for their invaluable support and for trusting me with the opportunity to work on such an engaging topic.

To my friends, I am truly grateful to every one of you, so many that naming you all would be impossible. Your support has made even the most challenging moments more bearable.

Finally, to mum and dad, I am forever thankful to both of you. Your unconditional love and support have given me everything I have today. Even when distance kept us apart, you continued to provide constant encouragement throughout my academic journey. No words can truly express my love and gratitude.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Background and context

Touch-based interaction is becoming more common in systems that aim to give physical feedback. Whether in wearable devices, therapy tools, or soft robotics, physical touch offers an immediate and intuitive way to trigger responses. It often works better than visual or audio cues in situations where attention, comfort, or accessibility is a concern.

In parallel, recent improvements in soft materials and simple sensing techniques have made it easier to build systems that react to touch without the need for complex electronics. Materials that change their resistance or capacitance under pressure can be used to classify input levels in a reliable way. This opens the door to design interaction systems with fewer components and less processing overhead.

Some existing devices already rely on vibration motors or slow breathing-like motion to give clear and calming feedback. These features are common in tools meant to reduce stress, support focus, or deliver basic sensory cues. The advantage is that such feedback is easy to interpret and doesn't rely on screens or sound.

This work was developed with those ideas in mind. However, many current systems rely on complex sensor arrays, signal conditioning, or software filtering to achieve this. These solutions can be costly, hard to scale, or difficult to integrate into soft robotic forms. The focus was on building a simple non-verbal system that could detect graded input and respond using both vibration and motion. One soft sensor was used to handle input, and the entire setup was run on a basic microcontroller without any external filtering or advanced computation. Each part was kept clear and adaptable so that the full interaction could be tested and adjusted in a practical way.

1.2 Motivation

Traditionally, many existing interaction systems focus on detecting a single event, e.g., a press, a tap, or just contact. But physical input is often more than just binary. A soft press is not the same as a strong one, and how a system reacts can affect how intuitive or natural it feels.

The idea behind this work was to explore how a soft sensor could be used to detect not just the presence of touch, but its intensity, and then use that to drive simple, clear feedback. Instead of adding multiple sensors or complex processing, the goal was to do everything with one sensor and basic logic.

Vibration and breathing-like motion were chosen as feedback types because they are easy to feel, quiet, and don't need visual focus. These forms of feedback have also been used in projects aimed at stress relief or sensory stimulation, which makes them useful in soft robotic or therapeutic contexts.

There was also a practical reason for the choices made. Many advanced systems are hard to reproduce or adjust. This setup was intended to show that clear feedback and graded interaction could be achieved without high-cost components or complicated design steps.

1.3 Research Objectives

The thesis set out to build a hands-on prototype that could classify different levels of touch and respond to them in a direct and noticeable way. Rather than just detecting whether contact had occurred, the goal was to interpret how strong the input was and adjust the feedback accordingly.

To make this possible, the project was guided by a few simple but clear targets:

- Use a single stretchable sensor capable of detecting a range of touch intensities through changes in resistance.
- Map those intensity levels to two forms of feedback, for instance, vibration for quick and localized response, and servo motion to mimic a slow, breathing effect.
- Keep the system low-cost, flexible, and easy to adjust or extend.
- Test the system live and tune its behavior by plotting the raw and smoothed data in Python, rather than relying on Arduino's built-in tools.

These steps were meant to explore how far a soft sensor and a few basic components could go in creating a smooth, responsive interaction — all without adding complexity that could block future adaptation.

1.4 Scope of Work

The thesis is focused on building a working prototype that could demonstrate graded tactile interaction through vibration and motion feedback. Rather than pursuing a polished or final product, the effort was directed toward making each component — sensing, classification, and response — functional, reliable, and easy to test.

A single soft sensor was used to detect pressure levels, while four vibration motors and two servos provided feedback. The system ran on an Arduino Mega 2560, which offered steady analog readings and enough pins to handle all outputs without additional circuitry.

Instead of using filters or external computation, all logic was handled in the same loop. Thresholds were tuned directly during testing, based on the real-time behavior of the sensor. To visualize the data more clearly, Python was used in place of the Arduino serial plotter.

At this stage, the main goal was to make the setup work as clearly and consistently as possible. Building a compact or optimized form was not part of this version. That part, along with improvements to timing and parallel behavior, remains for future work, once the core interaction is fully validated.

1.5 Thesis Structure Overview

This thesis presents the development process step by step, starting from the core design decisions and moving toward implementation and testing.

- **Chapter 2** provides a brief overview of related work and existing technologies, focusing on soft sensors, feedback systems, and tactile interaction tools.
- **Chapter 3** describes the materials used, including the stretchable sensor, electronic components, and the structural elements of the prototype.
- **Chapter 4** explains the methodology for testing and classifying different levels of touch. It also outlines how thresholds were selected and shows the Python plots that were used to analyze raw data and adjust values.
- **Chapter 5** focuses on implementation. It includes selected parts of the Arduino code to explain how the sensor reading, classification, vibration control, and servo motion were built and optimized.
- **Chapter 6** reflects on the results and limitations, while suggesting how the system could be improved or extended in future work.

Chapter 2

Related Work

In recent years, numerous academic researches have been proposed to explore the use of robots in the medical field, particularly in therapeutic interventions for children suffering from autism spectrum disorder (ASD). It offers an alternative human-like behavior while maintaining predictable and non-threatening interactions, which makes robots especially useful for children who find social contact stressful or overwhelming. The following sections review several innovative approaches in this domain.

2.1 Therapeutic Robotics for Autism

The use of robotic intervention in autism therapy is not a new idea. Since the early 2000s, several specialized systems have been developed to support therapeutic approaches for children with ASD.

For example, KASPAR (Figure 2.1), a humanoid robot with limited face emotions, has been used to teach social cues. PARO (Figure 2.1), a soft robotic seal, was designed to induce emotional responses through simple touch and sound. NAO (Figure 2.3), a programmable humanoid robot, has been widely used to interact through speech and gesture.

All of these robots mainly handle the visual or auditory communication, such as facial expressions, voice commands, or programmed movement patterns. However, children suffering from autism are highly sensitive to noise and may struggle to reveal their facial expressions. In these cases, such strategies can lead to ineffective or even stressful interaction. Thus, an alternative touch-based interaction method could be the solution.

Touch is a direct form of communication. It creates a clear connection between the child and the robot. Some research has shown that interactive responses can support emotional awareness, reduce anxiety, and help children stay engaged for longer periods. It also helps in controlling of sense and calmness, which is important in therapeutic contexts.

As a result, many recent efforts have focused on designing non-verbal, touch-responsive robots made from soft and friendly materials. These systems aim to respond not just to the presence of touch but also to its quality and intensity. This allows the robot to react physically in a way that mirrors the user's input, through gentle motion, vibration, or deformation. For children who are sensitive to traditional forms of interaction, this type of response can feel more natural and emotionally meaningful.

In the current project, this direction is taken further by developing a soft interactive ball that can detect different levels of touch — from light contact to strong grasp — and respond accordingly through physical feedback like vibration and breathing-like motion. This design avoids the complexity of anthropomorphic features and instead focuses on natural interaction through physical behavior, making it more suitable for therapeutic use with children on the spectrum.

Figure 2.1: KASPAR robot used for social interaction training for children with autism [1].

Figure 2.2: PARO therapeutic seal robot designed for emotional interaction [2].

Figure 2.3: NAO humanoid robot widely used in autism research and interaction studies [3].

2.2 Capacitive Touch Sensing – State of the Art

In this section, touch sensing techniques are analyzed especially the ones using frequency capacitive sensing, which is central to this project. A capacitive sensor is a low-cost solution for detecting simple touch interaction. However, the things are more complex when we want to deal with the integration of different conditions and intensities, especially in soft, deformable structures like the one used in this robot. The following subsections present a classification of capacitive touch approaches

from basic systems to more advanced gesture-recognition techniques, highlighting their potential and limitations in the context of therapeutic robotics.

2.2.1 Basic Capacitive Touch Techniques

The first capacitive sensing methods used basic touch detection systems which only identified whether a user was touching or not touching the surface. The systems detect user contact through capacitance measurements that occur when they touch conductive surfaces. The basic detection system works well for standard applications yet it lacks the capability to measure touch force which remains essential for emotionally responsive and therapeutic systems.

The work presented series-connected sensing electrodes as a solution which uses multiple electrodes connected through capacitors under a single pulse signal power [4]. The method decreases wiring complexity and functions well with flexible materials so it represents an appropriate selection for compact setups. The system supports only two simultaneous touch points yet remains highly sensitive to noise which reduces its reliability in therapeutic settings. The method fails to implement swept frequency analysis which prevents it from distinguishing between soft and firm touch inputs.

The Multi-Touch Kit (MTK) [5] provides a basic method for building DIY capacitive multi-touch sensing systems using standard microcontrollers. The system provides low-cost construction and simple implementation but maintains its focus on touch location instead of gesture understanding. The system fails to detect pressure and cannot understand the strength or softness of user touch.

2.2.2 Capacitive Touch Communication Systems

Some other research has focused on capacitive systems that are not intended for gesture or pressure detection, but rather to identify the ID of the person who is touching the device. These systems are more concerned with user recognition than with the type or intensity of interaction.

One example of this is the technique proposed in the study by Vu et al. [6], where a wearable ring or wristband sends a unique signal through the user's body, which can be detected when the user touches a screen or surface. This approach allows systems to differentiate between multiple users using capacitive conditions, without needing additional sensors or external hardware. While this method is interesting for applications like user authentication, multi-user control, or interactive games, it is not suitable for projects that aim to detect touch intensity or emotional

interactions. It also requires the user to wear additional equipment, which can limit its practicality in therapeutic environments, especially for children.

For this reason, although the idea of using capacitive signals for user recognition is technically innovative, it does not align with the needs of systems focused on non-verbal emotional interaction or touch-based gesture recognition. Nonetheless, the reasoning behind observable touch patterns may be an intriguing sensing technique for this project, particularly when paired with techniques from Swept Frequency Capacitive Sensing (SFCS).

2.2.3 Advanced SFCS-Based Systems

SFCS is an advanced technique used to detect a range of touch interactions by sweeping across multiple frequencies rather than using a single static signal. With this method, the system can record more detailed information about the interaction, such as the type of gesture, the intensity of the touch, and even proximity, depending on the sensor setup. For systems intended for therapeutic or emotionally sensitive applications, this can offer a more responsive and organic feedback format.

Watanabe et al.'s Foamin method is among the best suited for this purpose [7]. This technique employs a block of conductive foam as the sensing element together with a single wire. When a pressure is applied, the foam's impedance changes, enabling SFCS to identify and categorize touches that range from soft to hard. A machine learning classifier was used to improve accuracy, reaching high recognition rates. This simplicity and compatibility with soft, deformable materials make Foamin a highly suitable strategy.

Another design is the system by Lee and Bae [8], which uses addressable capacitive sensors combined with Frequency Division Multiplexing (FDM). Each sensor operates at a unique frequency, allowing a single wire pair to detect multiple simultaneous touches. But even though it deals with touches, it is more suited for rigid surfaces, and its complexity may create challenges in implementation and cost.

Honigman et al. [9] introduced an open-source SFCS platform built around Arduino microcontrollers. This setup, while more experimental, allows DIY developers to implement SFCS using common components. It was mainly developed for musical interaction surfaces but can detect different gestures such as soft touch, press, and grab. The downside is that the processing is slower and more sensitive to

noise, making it less ideal for high-precision or real-time feedback in therapeutic use.

One of the most famous works in this area is Touché, developed by Sato et al. at Disney Research [10]. Touché uses a wide frequency sweep (1 kHz to 3.5 MHz) to capture unique electrical signatures. It can recognize gestures like pinching, grabbing, or sliding with high accuracy. Although it delivers excellent performance, it was originally built with custom electronics and optimized for rigid or liquid surfaces. As such, it requires significant adaptation to work with soft, flexible materials like those in this project.

Another practical application of SFCS is shown in the Smart Steering Wheel developed by Ono et al. [11]. In this work, SFCS was used to monitor grip intensity, hand position, and approach detection. The project demonstrated that SFCS could distinguish different levels of pressure, which is relevant to this project's goal of interpreting different levels of user interaction. However, it was designed more for automotive safety, not therapeutic feedback, and uses rigid surface-mounted electrodes.

Lastly, the system described in Expressive Plant by Gao et al. [12] presents a more emotionally centered use of SFCS. This interactive system detects various gestures on a soft plant structure and responds with visual and auditory cues. It was specifically designed for use in sensory training for children with autism, making it one of the closest in spirit to the goal of this project. The authors showed that combining SFCS with simple actuation can trigger emotional communication in non-verbal settings, validating the therapeutic potential of this technology.

In summary, all reviewed SFCS-based systems have strengths in gesture detection and touch analysis. Nevertheless, many are designed for rigid surfaces, require complex hardware, or prioritize technical precision over therapeutic simplicity. Among them, Foamin (with some adjustments) stands out as the most suitable for this thesis: it is accurate, flexible, works with soft structures, and only needs one wire. Its successful integration with SFCS makes it the best choice for building a touch-sensitive therapeutic robot designed for children with autism.

2.3 Vibration Feedback in Therapeutic Devices

Touch is the primary form of communication in addition to its way of addressing emotional expressions. In therapeutic contexts, particularly when working with children suffering from autism, the addition of vibration physical feedback enhances the child's interactions and offers a direct, immediate, and non-verbal signal that

can reinforce a child's understanding of cause and effect, emotional responses, and interaction consistency.

Several research efforts have explored how vibration-based feedback systems can support therapeutic goals. Muhammad et al. [13] explored personalized vibrotactile and thermal stimulation patterns for affect regulation, showing that simple vibratory cues could evoke calmness or excitement depending on the vibration pattern. Also, Seo et al. [14] developed a plush toy with vibration feedback to help children experience touch in a safe and predictable way, improving their sensory engagement.

Moreover, McDaniel and Panchanathan [15] showed that therapeutic haptics could reduce anxiety levels and improve attention. MacDonald et al. [16] ensured that robotic interaction can create a stable emotional environment for children sensitive to overstimulation. Krishnan et al. [17] further stated that strategic placement of vibrators can influence the child's interactions.

This idea has been improved and used in the current project by integrating tiny vibration motors into the flexible ball. The motors are glued at specific spots inside the ball, ensuring that the vibration propagates uniformly across the soft surface. When the capacitive sensor senses a touch, the vibration motors are activated, detecting the interaction and producing a vibration intensity that corresponds to the force of the touch. A hard grip creates a more severe feedback pattern, whereas a soft touch causes a gentle vibration.

This direct physical response enhances the interaction by providing instant feedback, thus helping the child link their action (touch) to a physical reaction (vibration). It offers non-verbal cues that are easier to process than complex auditory or visual information. Experimental testing of this mechanism showed that vibration feedback is a critical bridge between sensory perception and emotional connection in the therapeutic context.

2.4 Bioinspired Breathing-Like Motion in Soft Robotics

Therapeutic robotics may produce sensations of emotional tranquility and connection by simulating natural processes like breathing. These movements can be useful for children that benefit from gentle and reliable sensory signals.

Recent research has demonstrated the power of soft robots to reproduce breathing behavior for therapeutic purposes. Klausen et al. [18] developed a soft social robot, Breco, which uses servo motors to create subtle breathing motions that were shown to alleviate anxiety symptoms in children. Similarly, Lingampally et al. [19] studied the effect of wearable assistive devices that included breathing-like movements that can assist children with autism in rehabilitation tasks.

The HBS-1 humanoid robot, introduced by Wu et al. [20], had servo-controlled modules that could replicate heartbeat and breathing cycles, facilitating emotional contact through embodied signals. Furthermore, Ishihara and Asada [21] explored the Affetto android, a child-size robot that studied pneumatically actuated breathing-like body movements reflecting connectivity with children.

Moreover, Yoshikawa et al. [22] integrated breathing actuation into an android robot used for psychological support, where air-servo control systems provided lightweight, realistic chest movements.

Benefiting from all these experimental designs and findings, the current project incorporates servo motors installed inside a soft elastic structure, programmed to produce slow inflation and deflation cycles that mimic natural breathing. This is achieved without the need for rigid frames or mechanical complexity, relying instead on the deformation of the ball surface in response to servo movements. The resulting breathing-like feedback strengthens emotional resonance during the interaction, offering a non-verbal way to comfort and engage children in therapeutic sessions.

2.5 From Literature to Implementation: Design Choices for the Therapeutic Ball

Based on the extensive review of existing technologies and approaches discussed earlier, a clear design path for the current project has been defined. The goal is to develop a therapeutic soft robotic ball capable of interpreting different touch intensities and interacting through physical feedback, suitable for children with autism.

For the sensing part, after studying various capacitive touch systems, the Foamin method was selected. Its structure, based on conductive foam and a single-wire configuration, offers an ideal match for soft and deformable surfaces. Unlike other

rigid or complex setups, Foamin allows precise detection of touch pressure variations while maintaining a simple, lightweight design. Its compatibility with SFCS also supports the classification of different touch gestures, which is critical for the intended application.

Regarding haptic feedback, the use of small vibration motors triggered human-like sensing. Drawing from the literature on emotional haptics, it became evident that vibrotactile feedback provides a direct, nonverbal communication channel that is well suited for children who are sensitive to traditional audio-visual cues and thus engaging them more in social life.

Finally, servomotors are embedded inside the ball structure to stimulate breathing-like motion. This solution enables the ball to perform rhythmic expansion and contraction in response to user interaction and making the robot's reaction feel more natural and emotionally meaningful.

These three elements: servo–actuated breathing motion, PWM-controlled vibration feedback, and Foamin-based sensing, create a hybrid interactive platform. This technical choice is closely related to therapeutic goals and beneficial for children in therapeutic environments.

Chapter 3

Methodology

3.1 Project Overview

The purpose of this project is to develop a soft, interactive robotic ball designed to support the interaction of children suffering from autism. The aim of this technology is to provide a haptic and emotional experience that can replace the various sensory inputs with an interactive and predictable feedback. The proposed robot works on sensing the child's touch through pressure levels and then respond with physical feedback vibration and motion that mimics breathing.

To achieve this, the system was designed around three main modules:

- **Touch Sensing Module:** A designed capacitive sensor solution based on SFCS and Foamin technology that allows identification and categorization of various touch intensities on conductive fabric.
- **Haptic Feedback Module:** A vibration-based system using several motors (vibrators) that respond with increasing intensity that depends on the detected contact strength. This non-verbal feedback mechanism supports emotional communication with children without physical touch.
- **Breathing Simulation Module:** A servo-driven actuation mechanism that reproduces a slow natural inflation-deflation cycle. This "breathing-like" motion adds a layer of emotional connection, comfort, and predictability during interaction.

Each of these modules was carefully studied and designed to reflect therapeutic needs, to achieve safe and soft material selection aligned with circuit simplicity and low noise. The individual modules were first independently tested and then integrated into a final assembly inside a flexible ball structure. Arduino uno, mega,

and ESP32 microcontrollers were programmed to test and control the system to process sensor data in real time and choose the optimal control unit. In the following sections, the different subsystems will be presented in details. Each part discusses the theoretical background and component selection aligned with design process, and finally the final functional prototype assembly.

3.2 Capacitive Touch Sensing Subsystem

3.2.1 Capacitive Touch Sensing – Theoretical Background

Capacitive sensing operates on the principle that the presence of a human body, which conducts electricity, can alter the capacitance between two conductive elements. In a basic setup, a capacitor is formed by a conductive sensor pad and the capacitance of the human finger . When a user touches the surface, a change in capacitance occurs, which can be measured to detect the presence and type of the interaction.

In this thesis, a SFCS method was used due to its capability to go beyond single touch detection. Unlike traditional capacitive sensing with fixed frequency, SFCS sweeps across a range of frequencies. This enables the system to analyze the impedance profile of the sensing material (in this case, conductive foam), which varies depending on the intensity and area of contact.

After observing how the impedance changes with different touch conditions, the system can sense the force and duration of the touch. For example:

- A light touch leads to a slight change in the impedance curve.
- A firm or grabbing interaction results in a more significant impedance decrease.

In contrast to simple capacitive methods , this frequency-domain approach widens data collection, particularly when dealing with soft and deformable materials, where touch interactions can be complex and highly varied.

Traditional capacitive systems are suitable for detecting basic touch presence (binary) and often fail in applications that require richer forms of interaction such as touch strength classification, or emotional communication.This limitation is critical in therapy where physical engagement can carry emotional meaning. By adopting SFCS, the system will support adaptive responses based on the intensity and the type of contact, that can be essential for children who rely on physical interactions.

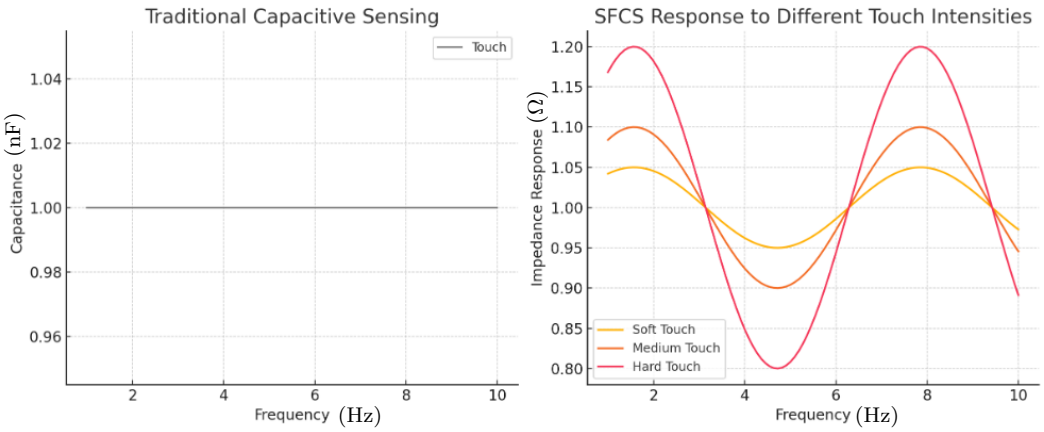


Figure 3.1: Comparison between traditional capacitive touch sensing (left) and SFCS-based touch recognition (right).

As illustrated in Figure 3.1, the SFCS method allows for multilevel classification of pressure-based inputs by analyzing the impedance response over a frequency sweep. To make this possible, a single conductive foam block is connected via a wire to the Arduino board. This foam acts as both a sensing element and a pressure transducer. The measured frequency response is then analyzed using a classification algorithm that separates interactions into multiple types of touch such as soft touch, medium press, hard press, and full grasp.

This theoretical basis sets the foundation for the sensor circuit described in the next section, which transforms these impedance shifts into reliable input for the rest of the system.

3.2.2 Component Selection & Circuit Design

The effective realization of a SFCS system depends on both theoretical concepts and hardware implementation. The development of the electronic circuit used to achieve the desired touch classification, from initial designs inspired by the literature to the final, simplified solution tailored for the specific needs of the soft robotic ball.

Circuit Design Exploration

To begin the experimental construction, the first version of SFCS circuit described by Honigman et al. [9] was implemented. This version, shown in Figure 3.2, includes

key components:

- $R_1 = 10K\Omega$, a pull-up resistor,
- $L_1 = 10mH$ inductor, acting as part of the LC resonant filter,
- $C_1 = 10nF$, forming the capacitive interface to the sensing surface,
- $D_1 = 1N4148$ diode, used for signal rectification,
- $R_2 = 3.3K\Omega$, $R_3 = 1M\Omega$, and $C_2 = 0.1nF$, forming a smoothing and filtering stage to detect and process voltage variations.

The circuit is powered and modulated using Timer 1 / PWM pin on the Arduino Mega, with signal acquisition at an analog pin. This design supports impedance sensing across frequency sweeps, enabling multi-level touch detection.

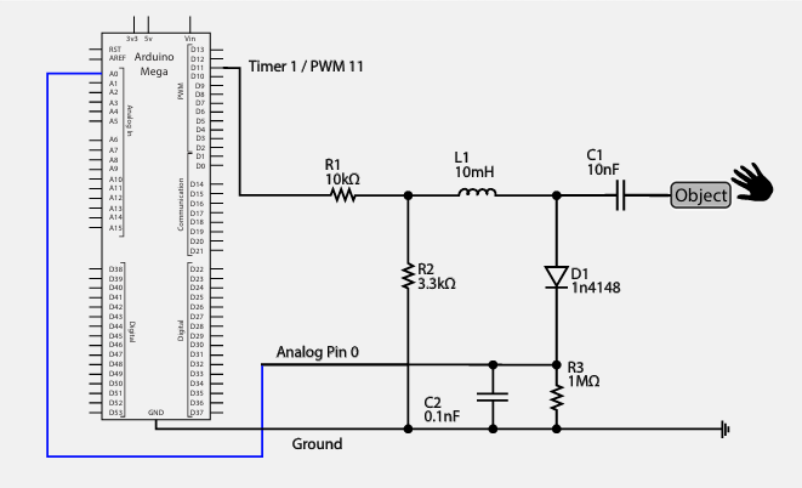


Figure 3.2: Initial SFCS circuit implemented based on [9], employing a resonant LC network with rectification and analog voltage output to detect touch-driven impedance changes,

While this circuit followed the theoretical model accurately, it proved somewhat sensitive to environmental noise and was not always stable with the flexible materials used in the robot and thats due to the complexity and sensitivity of the wiring layout.

Final Simplified Circuit

Finally, after some theoretical considerations and electronic testing, a seond hand-written circuit was designed and tested physically (Figure 3.3), subjected to work directly with the foam and conductive fabric that ensures the flexibility on the soft robotic ball structure.

This last configuration is built around a basic voltage divider approach using only one resistor and the conductive foam (Foamin) acting as a variable impedance classifier. The components include:

- Conductive foam covered with conductive fabric as the touch-sensitive element,
- A 2 MΩ resistor, connected between the analog input and ground,
- Single wire connection from one side of the foam to 5V and the other to analog input (A0),
- Arduino Mega 2560 used for data acquisition at analog pin A0.

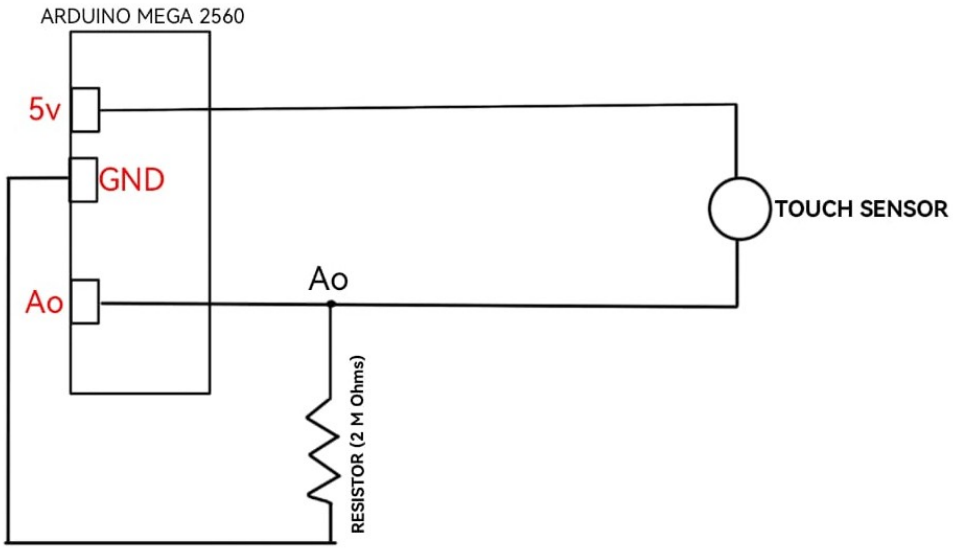


Figure 3.3: Final capacitive touch circuit implemented manually — simplified and optimized for interfacing with soft foam material and Arduino analog input.

In this setup, the **2 MΩ resistor** plays a key role as a pull-down resistor. It ensures that the analog pin reads a stable baseline voltage in the absence of touch, improving signal reliability. When the user touches the foam, the impedance of

the foam changes, modifying the voltage divider ratio and producing a measurable shift in the analog reading. This shift is then analyzed in the software to determine the intensity and type of touch.

Final Component Selection

The finalized sensing system adopted the following component and connection structure:

- Conductive Foam (Foamin) covered with conductive fabric as the sensing medium,
- Single wire connection to analog input for impedance reading,
- 2 MΩ pull-down resistor to stabilize signal and support voltage division,
- Minimal passive components to reduce signal distortion and noise,
- Powered via Arduino Mega or Uno depending on testing phase,
- Classification achieved via software analysis of raw analog readings across multiple frequencies (explained in later software section).

This configuration allowed the sensor to successfully distinguish between various touch interactions with reliable consistency. It also enabled integration with the real-time signal processing framework later developed in the code layer.

3.2.3 Realization & Testing Approach

To examine the performance of the capacitive sensing system, a structured series of functional tests was conducted during the development phase. The goal was to validate that each interaction produced distinct and reliable voltage signals suitable for classification and accurate feedback control.

The initial setup involved testing the sensor alone, using the final circuit design described in Section 3.2.2. Touch inputs were applied across four different levels (soft touch, medium press, hard press, and full grasp). Then, using real-time plotting on the serial monitor and external graphing tools, clear visualization made us able to verify that each interaction produced a measurable change in the analog signal and thus distinguishing analog values for each touch category.

To strengthen this validation, the capacitive sensor was paired with a Force Sensitive Resistor (FSR) during multiple trials. The analysis held for two purposes: Calibration Reference: The FSR was used as a baseline to assess the proportionality

of capacitive measurements and accuracy detection . Signal Consistency Check: The overlap in detection patterns confirmed that the foam sensor output correlated reliably with touch strength. Two different trial setups were performed:

- A linear sequence of touch types (e.g., soft → medium → hard → grab) was applied with rest intervals in between.
- An alternating format, where each touch was followed by a no-touch period, was used to evaluate reset response and sensitivity range.

Visual documentation of these trials was captured and included in supporting figures (see Chapter 4). Bar graphs were also generated to illustrate the classification pattern derived from each interaction stage. This testing confirmed the system's ability to calibrate touch intensities and provided the data needed for software-level classification tuning in later stages.

3.2.4 Final Sensor Setup Summary

After the development and validation of the sensing circuit, the final integration of the capacitive touch system was carried out inside the flexible robotic ball. The sensor was positioned inside the ball and attached to the conductive fabric to ensure both comfort and responsiveness. This placement was used to maintain full surface deformability without compromising signal quality.

One side of the foam was connected to the analog input pin (A0) on the Arduino Mega, while the other was connected to the 5V supply. The 2 MΩ resistor served as a pull-down resistor, improving signal stability.

The entire structure remained lightweight and soft to accommodate the therapeutic context of the design.

In addition to manual interaction, the final configuration was tested using a the ball prototype in addition to 3D-printed robotic arm to simulate repeatable touch inputs. This helped ensure the sensing behavior was consistent under both human and mechanical interaction conditions.

Figures included in Chapter 4 further illustrate the sensor mounting position, wiring layout, and response behavior under simulated testing.

3.3 Haptic Feedback via Vibration

3.3.1 Theoretical Role in Therapy

In therapeutic contexts, children with autism struggle with traditional audio-visual stimuli, and non-verbal feedback channels are often more effective for them.

Vibration, as a form of haptic feedback, offers a direct and predictable response that helps children interpret and react to interaction without requiring verbal or visual interpretation.

Several studies have emphasized that vibrotactile feedback can reduce anxiety, reinforce attention, and support emotional regulation. These tactile cues, when delivered consistently, enable children to form a link between their actions and the system's response, which is especially helpful in developing cause-effect understanding and engagement in therapy.

In the current project, small vibration motors were embedded inside the flexible ball and triggered by the output of the capacitive sensing system. A feedback is generated with increasing intensities depending on the level of pressure.

A general illustration of the vibration motor placement and physical structure is shown in Figure 3.4.

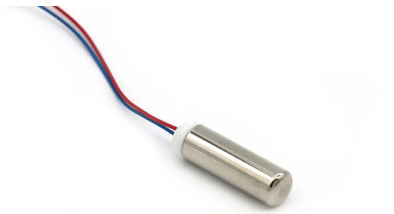


Figure 3.4: Shape and structure of one vibration motor used in the haptic feedback module. The motor is embedded within the ball surface and activates according to touch intensity.[23]

3.3.2 Hardware Setup & Circuit Design

The vibration feedback system uses small cylindrical encapsulated DC motors (Figure 3.4) selected for their low noise, compact size, and suitability for soft integration. Each motor operates at 3–5V and was directly driven by the Arduino Mega using PWM (Pulse Width Modulation). This made them ideal for creating multi-level intensity feedback within the soft ball.

The final circuit design is straightforward: one wire of each motor connects to a PWM-capable digital pin (e.g., D11), while the other goes to ground (Figure 3.5). The vibration intensity is modulated by changing the PWM duty cycle, allowing dynamic feedback that reflects the detected touch strength. No external transistors or drivers were required due to the low current demand of these motors and to maintain simplicity.

This configuration was repeated across four motors, each assigned to a separate PWM pin. The control logic determines the number and intensity of activated motors in real time, based on the sensor's classification of touch intensity.

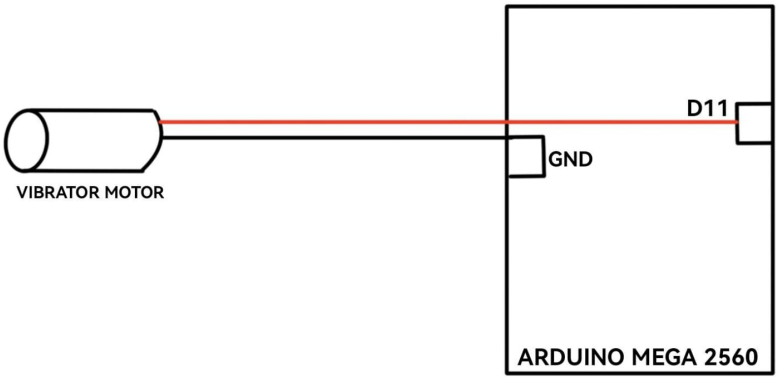


Figure 3.5: Circuit schematic showing PWM-controlled vibration motor connection to Arduino (D11) and ground.

3.3.3 Vibration Feedback Strategy

Our goal was to translate different levels of touch pressure into physically distinct vibration patterns that the child could perceive without requiring visual or auditory cues. However, since changing the intensity of all motors simultaneously was technically not possible, an alternative solution was imposed.

Four touch types were mapped to a corresponding number of vibration motors being activated:

- **Soft Touch** → 1 Motor ON
- **Medium Press** → 2 Motors ON
- **Hard Press** → 3 Motors ON
- **Grasp** → 4 Motors ON

This mapping logic provides a progressive, intuitive physical response to increasing pressure. By gradually modifying feedback, children receive a consistent sensory response in accordance with the degree of their physical input, supporting both emotional regulation and engagement.

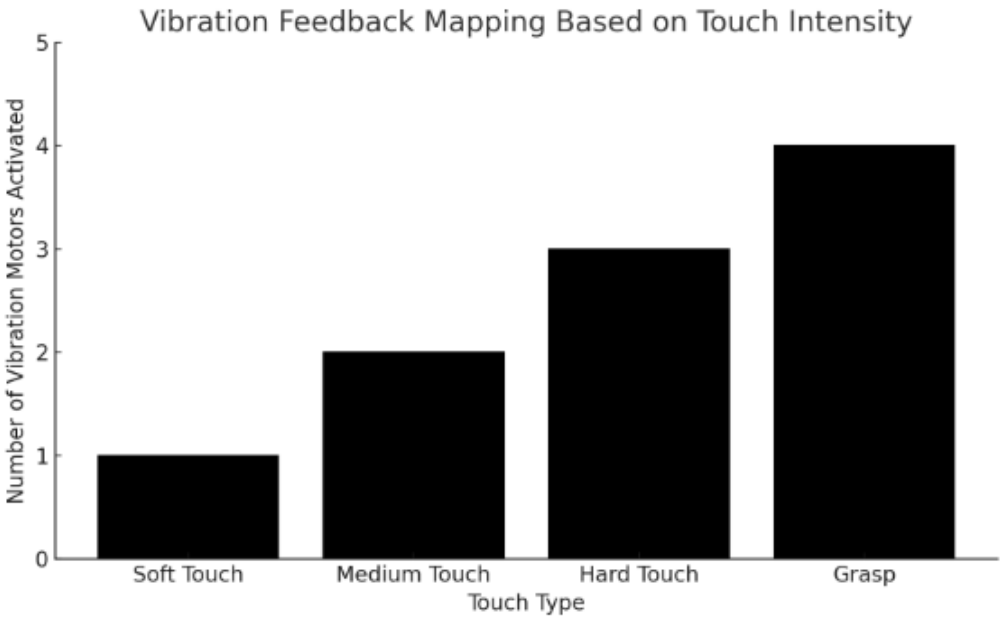


Figure 3.6: Mapping of touch intensity levels to number of activated vibration motors. Each condition corresponds to a discrete step of tactile feedback.

3.3.4 Final Integration

The motors were physically integrated into the interior of the soft robotic ball following the validation of the vibration circuit and control logic. Their positioning was planned to ensure both effective feedback distribution and user comfort during therapeutic interaction.

The vibration motors were positioned carefully on the four opposite sides of the interior beneath the outer fabric. Wires were routed carefully inside the ball's structure to prevent discomfort or unintended resistance during handling or play.

The Arduino Mega microcontroller was responsible for driving the PWM signals that control the motors in real time. Based on the sensor's classification of touch, the number of motors activated changed dynamically, converting touch pressure into spatially distributed haptic cues.

A labeled layout is shown in Figure 3.7, indicating the position of each motor and the corresponding direction of feedback force.

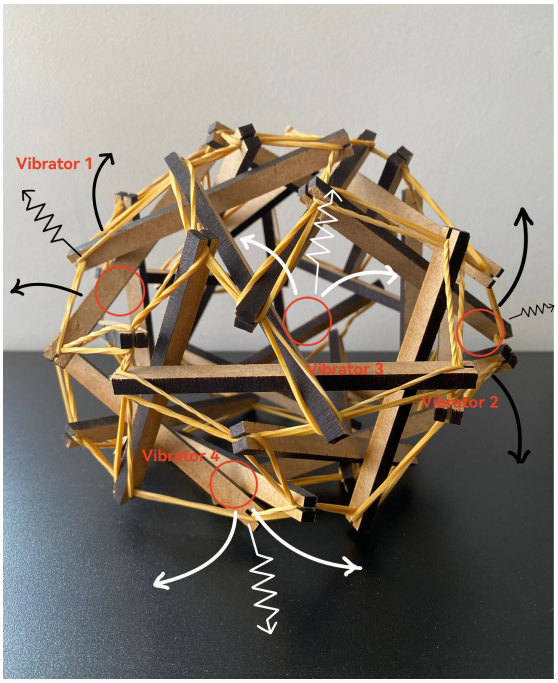


Figure 3.7: Vibration motor placement layout inside the soft robotic ball. The image highlights the four motor zones, each contributing to directional and intensity-based haptic feedback.

3.4 Breathing-like Actuation Subsystem

3.4.1 Emotional Value of Breathing Cues

One of the key emotional design principles in this project is the use of breathing-like motion to offer a sense of calm and presence. This idea is inspired by the natural rhythm of human respiration, which is widely associated with relaxation and emotional grounding.

In therapeutic contexts, especially for children suffering from autism, the predictability and softness of breathing movements can provide a sense of comfort and emotional safety. Unlike visual or auditory stimuli, which can sometimes be overstimulating, rhythmic motion offers a gentle and silent cue that encourages engagement without overwhelming the child.

Several bioinspired robotic systems served as sources of inspiration for this element, such as *Affetto* (Figure 3.9) and *BRECO*(Figure 3.8), both of which demonstrated that slow, breathing-like actuation helps foster emotional bonding.

BRECO (Breathing Companion Robot) was developed specifically for children with neurodevelopmental disorders and uses inflation-deflation cycles to simulate

calm, therapeutic respiration patterns [24]. Similarly, *Affetto* is a humanoid robot designed with soft features and facial actuation to investigate emotional interaction [25].

These examples inspired the current project to integrate a subtle, haptic breathing simulation that does not aim to replicate human life exactly but rather evokes a familiar and reassuring rhythm that supports therapeutic interaction.

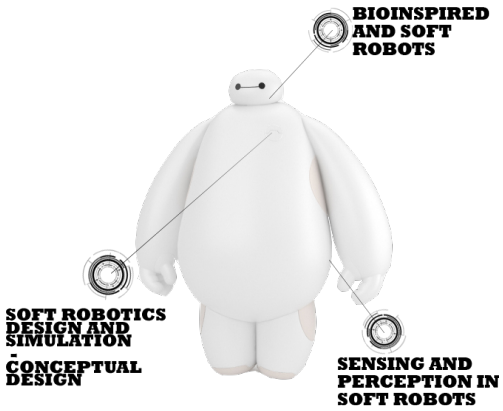


Figure 3.8: Puffy interactive robot [26]

Figure 3.9: Face interactive robot [27]

3.4.2 Mechanical Design & Servo Control

The breathing action in the robotic ball was made feasible by using a micro servo motor. After testing several types, the MF90 servo was chosen for its compact size, metal gear design, and enough torque to bend the flexible outer shell. Its dependability and reactivity under low-power settings make it ideal for repeated motion in a soft structure.

The two servos were mounted inside the ball in a fixed opposite lateral locations. A lightweight elastic bands were attached from the servo arm to the inner sides of the shell. The tension in the band caused an outward expansion and contraction effect, similar to a breathing cycle, as the servo rotating slowly and steadily. This configuration allows for simple inflation-deflation-like deformation without adding bulk or altering the ball's softness.

The mounting was designed to preserve the deformability and safety of the overall system. Soft padding and fabric insulation were added around the servo casing to eliminate hard edges and reduce noise. (Figure 3.10) shows the selected servo model used in the breathing subsystem.

Figure 3.10: MF90 Micro Servo used for breathing simulation, with mounting accessories.[28]

3.4.3 Programming Breathing Motion

To create a breathing-like motion inside the ball, two MG90S servo motors were programmed using a basic Arduino loop. The goal was to smoothly stretch and release the elastic bands that deform the soft surface, making it rise and fall in a calm, repeating rhythm.

A full cycle was broken into two steps:

- **Inhalation**: The servo angle slowly increases from 0° to 180°, pulling the bands outward and creating an expansion in the shell.
- **Exhalation**: The servo returns back from 180° to 0°, letting the ball gently return to its resting shape.

This motion was repeated in a loop. The code below shows how the movement was implemented using two synchronized servo motors.

Listing 3.1: Arduino code controlling breathing motion

```
#include <Servo.h>

Servo myservo;
Servo myservo2;

int pos = 0;

void setup() {
  myservo.attach(7);
  myservo2.attach(12);
}

void loop() {
  for (pos = 0; pos <= 180; pos += 10) {
    myservo.write(pos);
    myservo2.write(pos);
    delay(15);
  }
  for (pos = 180; pos >= 0; pos -= 10) {
    myservo.write(pos);
    myservo2.write(pos);
    delay(15);
  }
}
```

This basic setup allowed the servos to move steadily back and forth. No sensors were used in this phase; the motion runs on its own in the background during use. The aim was to create a predictable and calming breathing effect that repeats with each cycle.

3.4.4 Final Implementation

Once the servo control logic had been tested to validate the breathing mechanism, the two MG90S servos were mounted on opposing sides, aligned to preserve structural balance and assure smooth movement.

Each servo was linked to an elastic band secured inside the ball. The servo's rotation (pull or push) results in the previously stated expansion and contraction. This internal motion occurred in a rhythmic behavior, operated by code executed on the Arduino, and did not require external triggers.

To maintain safety and comfort, the servo housings were cushioned with soft padding and wrapped in fabric to minimize contact with any rigid surfaces. The internal layout ensured that the deformation was perceptible but not disruptive, preserving the ball's soft texture and tactile appeal.

A schematic of the internal configuration is shown in (Figure 3.11), where the placement and directional motion of each servo are highlighted.

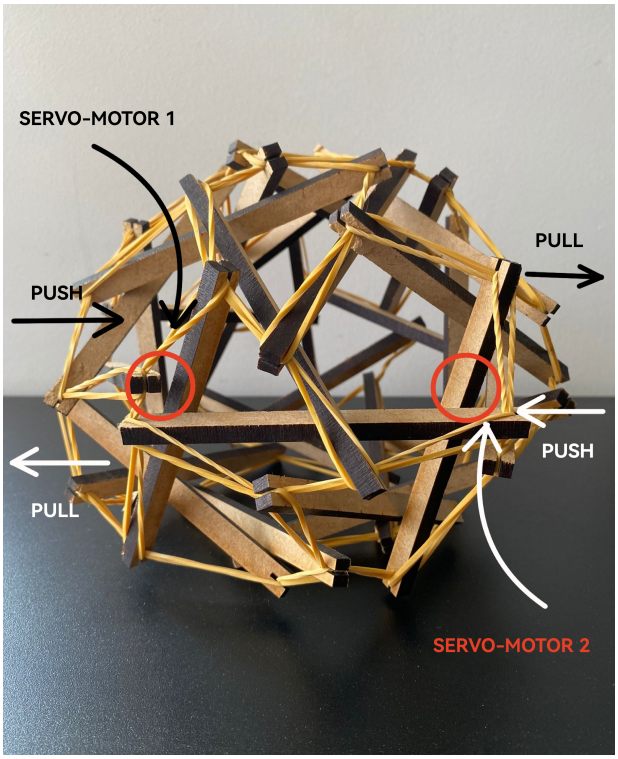


Figure 3.11: Internal configuration of the breathing mechanism. Arrows illustrate the push-pull forces applied by the two servo motors to drive the rhythmic expansion and contraction.

3.5 System Integration

3.5.1 Control Architecture

The Arduino Mega 2560 was selected as the main controller due to its large number of I/O pins, good analog performance, and reliable timing accuracy. These qualities made it well-suited for managing the multiple subsystems involved in sensing and actuation.

Each component was assigned to specific pins:

- The capacitive sensor was connected to analog pin A0.
- Four vibration motors were controlled through PWM-enabled pins D8 to D11.

- Two servo motors responsible for the breathing motion were connected to digital pins D7 and D12.

The control logic was straightforward: based on real-time capacitive sensor readings, the Arduino triggered a certain number of vibration motors and altered the servo angles correspondingly. The readings were continuously processed to ensure that the interface was smooth and responsive.

Figure 3.12 presents an overview of the control architecture, showing the pin layout and how each subsystem was integrated into the board.

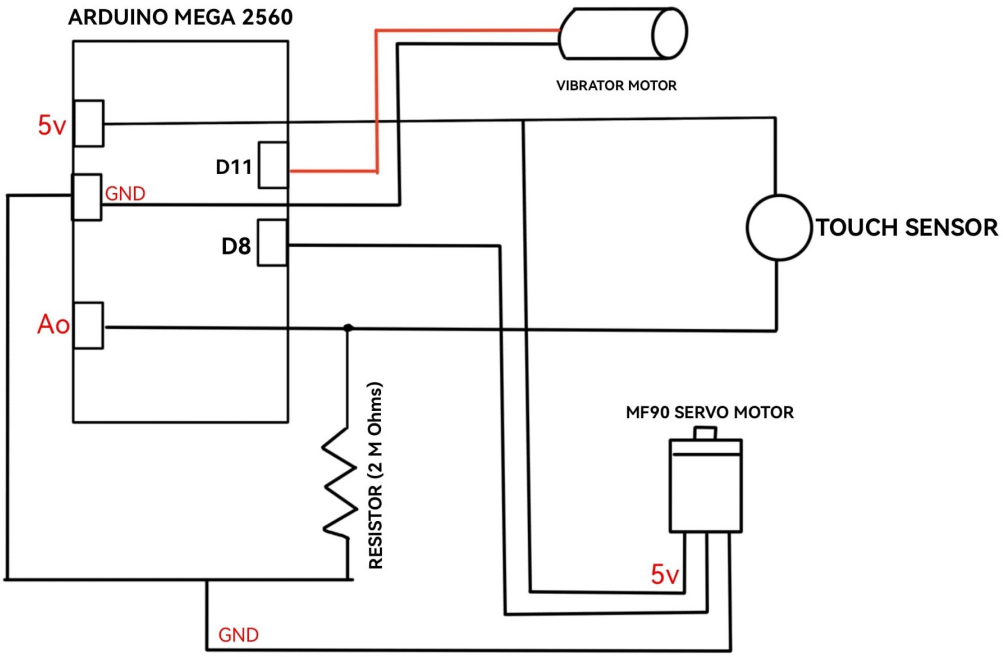


Figure 3.12: Complete system circuit diagram showing the integration of the capacitive sensor, vibration motors, and servo-controlled breathing system with the Arduino Mega.

3.5.2 Software Overview

The main control algorithm was implemented on the Arduino Mega in C++, using a modular structure that coordinated the sensor input, classification, and actuation outputs. Each loop cycle collected the analog signal from the foam sensor (pin A0), calculated the equivalent resistance, and processed it using a smoothing function to eliminate high-frequency noise.

The smoothed resistance was then classified into five interaction levels: no touch, soft touch, medium press, hard press, and full grasp. These categories triggered the appropriate number of vibration motors through digital output pins (D8, D9, D11, D2), while simultaneously adjusting the servo movement for breathing simulation.

The servo control was integrated into the same logic using incremental rotation loops. Each pressure level changed the servo step size to simulate varied breathing rates: light touches caused delayed expansion, while firm grasps resulted in faster contraction cycles.

Additionally, a stretch classification layer was included to detect deformation intensity beyond typical pressure. While not linked directly to feedback, it provided diagnostic logs for future expansion.

Figure 3.13 summarizes the overall logic of the system, showing the real-time sensor reading, resistance filtering, classification thresholds, and synchronized motor outputs.

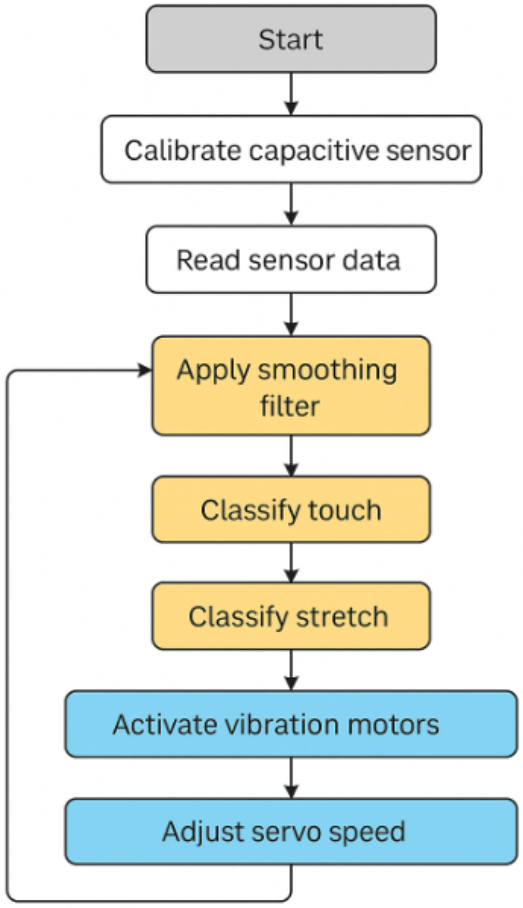


Figure 3.13: Flowchart representing the software logic implemented on the Arduino. It illustrates the sequence of data acquisition from the capacitive sensor, classification of the touch intensity, and triggering of the appropriate vibration and breathing response.

3.5.3 Assembly of the Ball

At the final stage of the system's development, each component was assembled inside the soft robotic ball, taking into account the system's mechanical harmony and flexibility.

Integration of Fabric, Circuits, and Sensors

The capacitive sensing element, made of conductive foam and fabric, was positioned where interaction typically occurs—along the upper interior arc of the ball. The connecting wires from the foam were installed along the internal sticks and fixed

to prevent any physical obstruction or discomfort during use.

Vibration motors were spaced evenly around the interior shell. The servo motors were installed on opposing sides of the inner wall, anchored with elastic bands that allowed for subtle inward and outward motion to mimic breathing.

To keep the electronic components safe, the Arduino Mega board was enclosed inside a cushioned pocket at the center of the ball. Its connections were distributed to the motors and sensors with careful insulation and flexible wire management, preserving the deformability of the entire structure.

Visual Documentation

To help illustrate this configuration, two real-world photographs were captured:

- **Figure 3.14** Internal setup without outer layers — highlighting the sensor, motor, and servo placement as well as routing of internal wires.
- **Figure 3.15** Final assembly with fabric covering — showing how the outer appearance remains soft and coherent despite embedded electronics.

These visuals support a better understanding of how the system was built and show that internal feedback mechanisms do not compromise the ball's therapeutic qualities.

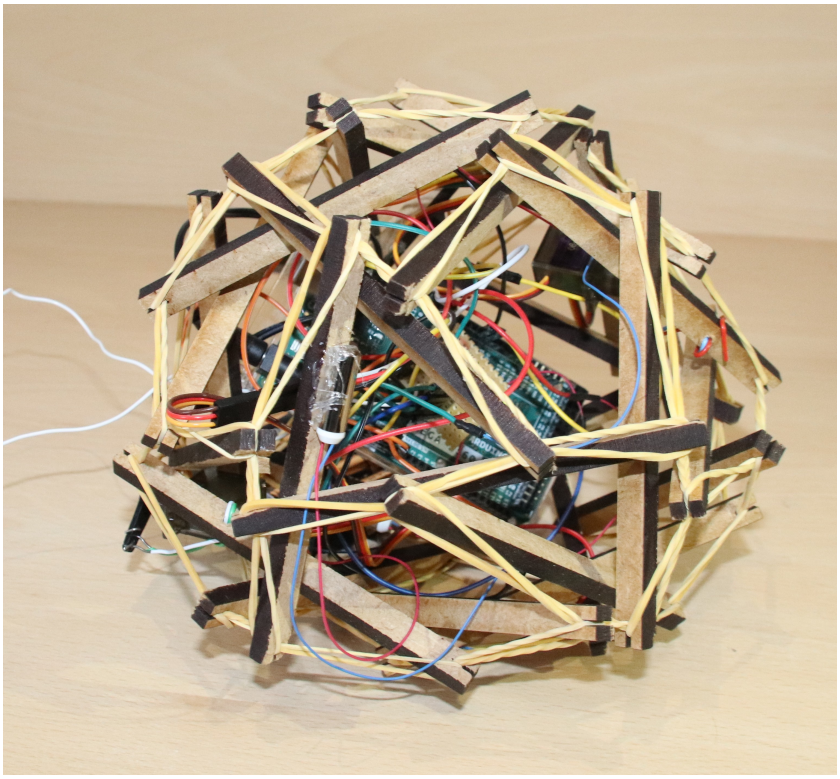


Figure 3.14: Internal setup without outer layers.

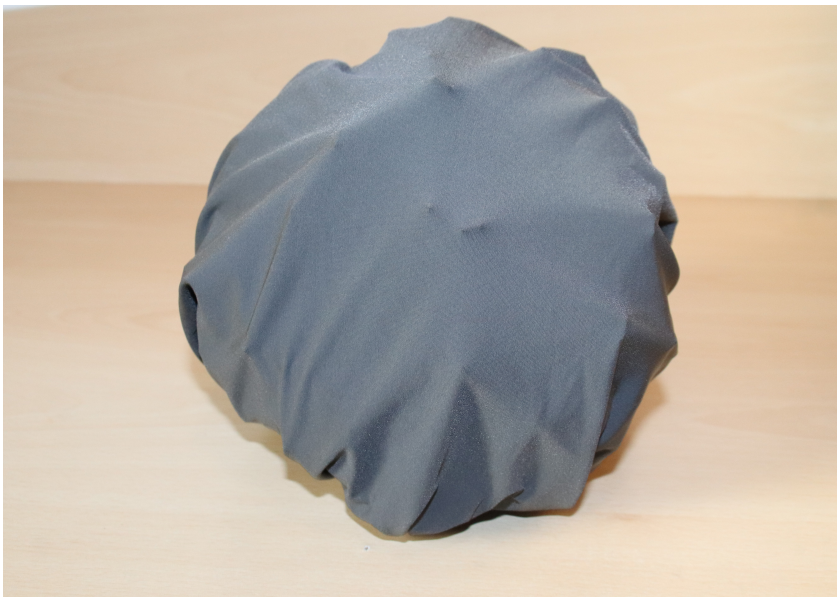


Figure 3.15: Final assembly with fabric covering.

3.6 Summary of Methodological Innovations

Key Technical and Therapeutic Decisions From the early stages of development, design choices were shaped by therapeutic goals—particularly the need for a system that could offer calming, non-verbal interaction. The decision to use a soft capacitive foam sensor enabled pressure-based input without requiring firm contact, making it more accessible for children with sensory sensitivities. Likewise, the use of vibration motors and breathing-like motion was grounded in their ability to deliver gentle, consistent cues without visual or auditory overload. These features were not chosen in isolation, but in response to specific interaction needs observed in therapeutic settings.

Modularity, Simplicity, and Ease of Use Each component of the system—the sensor, the vibration actuators, and the servo-based motion unit—was designed to operate independently, allowing for easy testing and troubleshooting. At the same time, they were built to work together through a unified control logic, executed on a single Arduino Mega. This approach minimized wiring complexity and made the system more reliable. Materials like foam, elastic bands, and flexible fabric were selected not only for their mechanical function but also to support tactile comfort and maintain the softness of the overall structure.

Advantages Over Earlier Designs Compared to many previous therapeutic robots that rely on screens or sound-based feedback, this system offers a quieter, more tactile form of interaction. By focusing on physical sensations and rhythmic motion, it provides a more grounded and emotionally supportive experience. The single-board control, reduced circuitry, and lack of bulky hardware make it both practical and easier to replicate. This simplicity does not come at the expense of richness—it enables nuanced, real-time feedback in a form that fits comfortably into therapeutic environments.

Chapter 4

Experimental Testing & Results

This chapter presents the experimental testing and validation of the developed robotic ball system. Each part (capacitive touch sensing, vibration-based haptic feedback, and breathing simulation) was tested independently to verify performance under real time conditions. Several trials were performed using different microcontrollers and circuits to achieve the optimal signal stability and interactive classification . Final integration testing was performed on the full setup, both with and without the outer fabric, to confirm the system's responsiveness, comfort, and reliability during interaction. The results are supported by photos, graphs, and real-time output recordings.

4.1 Testing Strategy and Objectives

A stepwise testing strategy was developed to evaluate the performance and integration of the soft robotic ball's main components under realistic conditions. Rather than proceeding directly to full-system testing, each subsystem was verified in isolation before moving on to combined evaluations. This approach supported early identification of issues and allowed adjustments to be made incrementally.

The testing workflow was structured into four main phases:

Isolated Component Verification

Each hardware element was tested individually. This phase was used to validate correct wiring, confirm basic responsiveness, and adjust initial control parameters such as actuation thresholds or delay timing, including for the capacitive sensor, vibration motors, and servo motors.

Microcontroller Evaluation

The capacitive sensor was tested using three different boards: Arduino Uno, Arduino Mega 2560, and ESP32-WROOM-S3 to determine the most stable and accurate platform for classification.

Progressive System Integration

Following successful individual testing, components were connected in logical pairs to evaluate combined functionality — for example, sensor-to-vibration or sensor-to-servo linkages. This refined the control logic used for actuation. Gradual integration ensured that subsystem behavior remained predictable when placed under multi-component coordination.

Final Assembly Testing

Complete system tests were then conducted with and without the foam and fabric layers. This phase assessed whether the external covering affected the sensor's responsiveness, and whether the overall setup preserved the intended softness, comfort, and user-friendly interaction profile in its final embedded state.

Key evaluation metrics across all tests included:

- Accuracy of touch and stretch classification
- Response speed and smoothness of actuation
- Signal stability and noise suppression
- Consistency of physical appearance and tactile performance during operation

Phased testing improved overall system consistency and made debugging easier by ensuring that each functional layer was separately validated prior to integration.

4.2 Capacitive Sensor Testing

The capacitive sensor was the first part to be implemented for detecting and classifying interactions. Early trials focused on verifying signal behavior to various touch intensities, and selecting the most stable microcontroller. The aim was to ensure that its output could be used reliably in later integration stages.

Testing was done progressively, starting with a basic setup on Arduino Uno, followed by improved trials using the Arduino Mega 2560 and later the ESP32. At each stage, changes were made to the circuit or platform to reduce signal noise

and improve classification accuracy. These tests helped define threshold values for different levels of contact and showed which platform gave the cleanest and most consistent readings.

4.2.1 First Trial: Arduino Uno

The initial testing phase used an Arduino Uno connected to the foam-based capacitive sensor. This trial aimed to confirm whether the sensor could detect and reflect basic variations in touch — from light taps to strong presses figure 4.1.

The results showed that the sensor was responsive, but the signal data lacked consistency. Lighter touches, in particular, were difficult to separate from ambient fluctuations. The Uno's limited ADC resolution and relatively slow analog sampling made it hard to maintain a stable signal baseline. As a result, small changes in pressure did not always translate clearly in the readings, and classification was not reliable figure 4.2.

The trial confirmed that the sensing mechanism worked in principle, but the platform's hardware constraints limited its practical use for distinguishing finer levels of interaction. This phase was useful to establish early response behavior, though it also made clear that a more stable board would be needed in later stages figure 4.3.

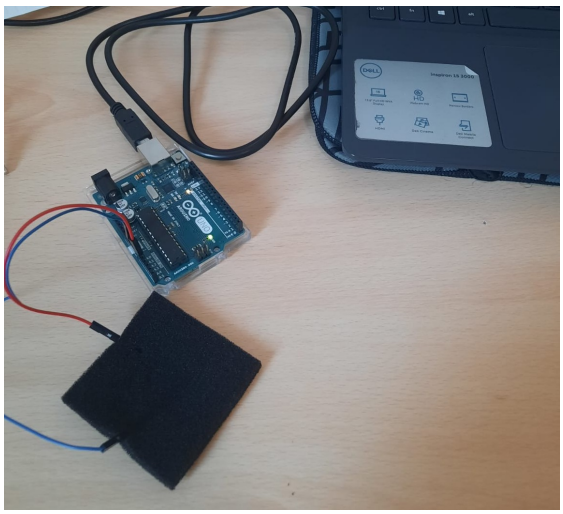


Figure 4.1: Sensor connected to Arduino Uno.

```
Hard Touch Detected
Hard Touch Detected
Hard Touch Detected
Medium Touch Detected
Medium Touch Detected
Soft Touch Detected
Soft Touch Detected
Soft Touch Detected
No Touch
No Touch
```

Figure 4.2: Serial classification output.

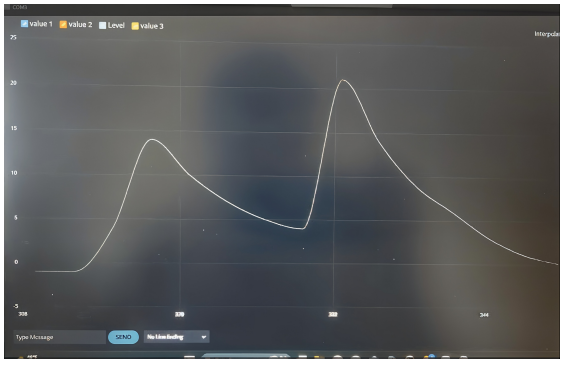


Figure 4.3: Touch response graph.

4.2.2 Second Trial: Arduino Mega 2560

After the first trial, the setup was upgraded to the Arduino Mega 2560, which provided improved analog resolution and better stability for signal handling. A 1 MΩ resistor was added in series with the sensor to reduce fluctuations and improve noise immunity.

This setup showed significant improvements. The sensor output became noticeably smoother and more reliable. Touch interactions of varying intensity (ranging from light contact to full press) resulted in distinguishable resistance variations. Unlike the Uno, which often produced inconsistent readings, the Mega was able to separate different levels of pressure more clearly.

Figure 4.4 shows the updated setup with the Arduino Mega. A sample output of touch classification is shown in Figure 4.5, where each touch level is correctly

distinguished. The improved signal stability is also evident in the plotted graphs , showing clean transitions and repeatable waveform behavior.

This trial confirmed that the Arduino Mega was a suitable platform for the final integration, especially when reliable classification and real-time response were required.

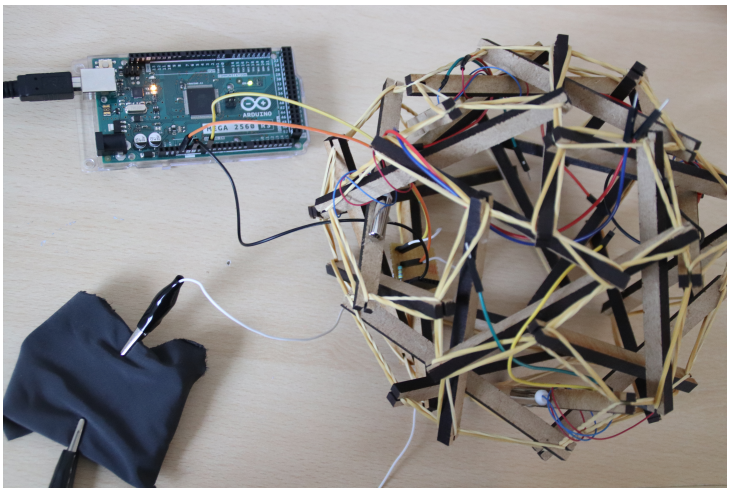
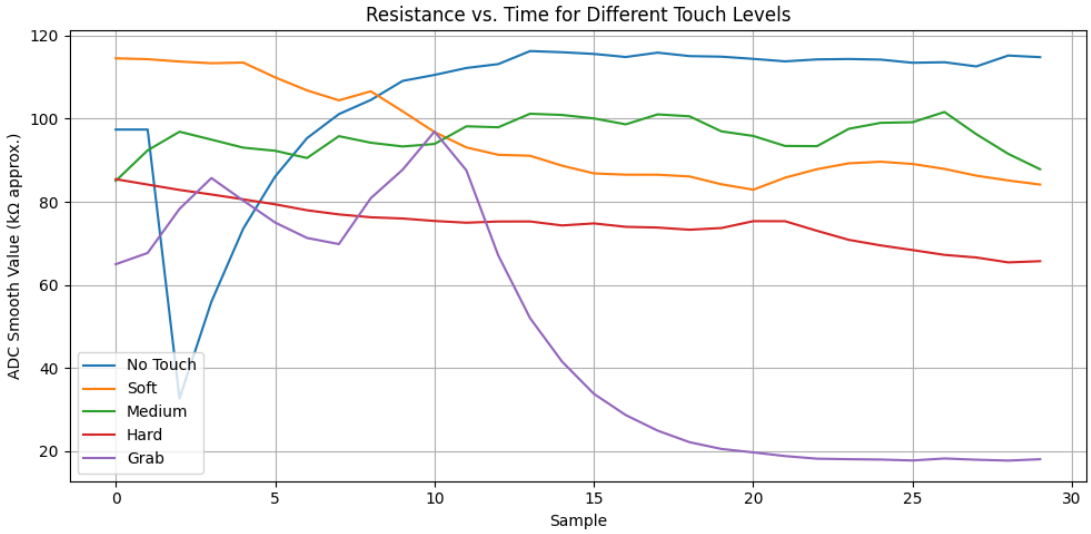
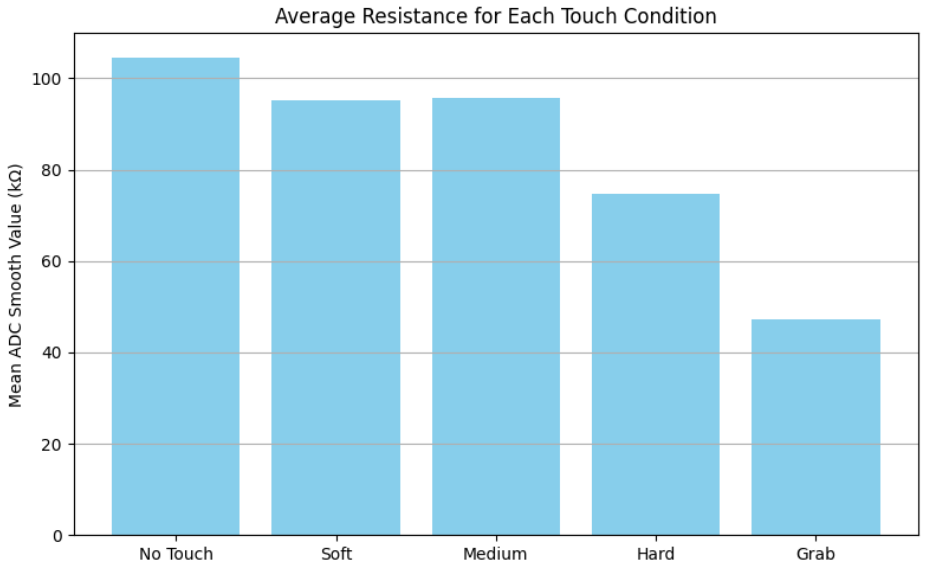


Figure 4.4: Updated setup using Arduino Mega 2560 with conductive fabric and foam sensor.



(a) Touch response graph.



(b) Bar chart: resistance per touch level.

Figure 4.5: Resistance patterns and classification output.

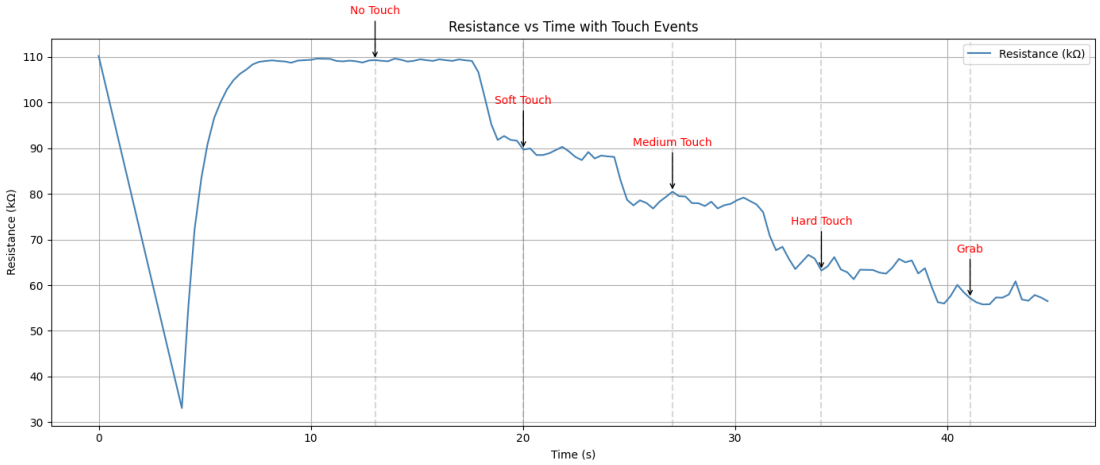


Figure 4.6: Grouped resistance curves showing response over time for various touch levels.

```
ADC smooth value = 185.04
Soft Touch Detected
No Stretch
Resistance: 176354.03 Ω
--------------------------------
ADC smooth value = 182.44
Medium Touch Detected
No Stretch
Resistance: 149746.18 Ω
--------------------------------
ADC smooth value = 172.63
Hard Touch Detected
No Stretch
Resistance: 135235.73 Ω
--------------------------------
ADC smooth value = 161.41
Grab Detected
No Stretch
Resistance: 125916.88 Ω
--------------------------------
ADC smooth value = 150.76
Medium Touch Detected
No Stretch
Resistance: 142409.04 Ω
--------------------------------
ADC smooth value = 148.26
Medium Touch Detected
No Stretch
Resistance: 154731.45 Ω
```

Figure 4.7: Real-time serial output illustrating touch classification levels and corresponding resistance values.

4.2.3 Third Trial: ESP32-WROOM-S3

The ESP32-WROOM-S3 was the final board used in the series of sensor trials. Unlike the Arduino Uno and Mega, this microcontroller included built-in wireless features and a faster processor, which made it a candidate for potential mobile or networked applications. However, the aim at this stage was to check whether its analog performance would be consistent enough for reliable touch detection.

The test configuration remained the same: the same foam-based capacitive sensor and resistor combination were used (Figure 4.8), and the output was monitored via a serial interface. In comparison to the Mega, the ESP32 produced higher fluctuation in resistance values. These variations were more obvious when using a light or moderate touch, making it difficult to establish constant thresholds (Figure 4.9).

Although the board was able to detect simple contact changes, some ADC noise induced instability. This behavior created uncertainty in the middle ranges when categorization boundaries were more rigid, but it was less significant in situations with high pressure. Positively, during the experiments, the ESP32 enabled seamless wireless transmission of sensor data, demonstrating its applicability for remote feedback configurations.

In this context, the ESP32 was found useful for future applications where mobility and network integration are required, but the Arduino Mega remained the better option for high-resolution sensing in controlled conditions.

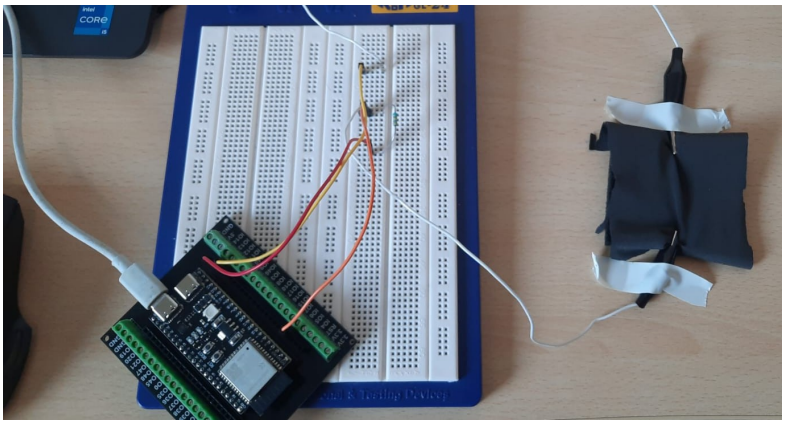


Figure 4.8: Sensor circuit setup using ESP32-WROOM-S3, foam sensor, and signal stabilizing resistor.

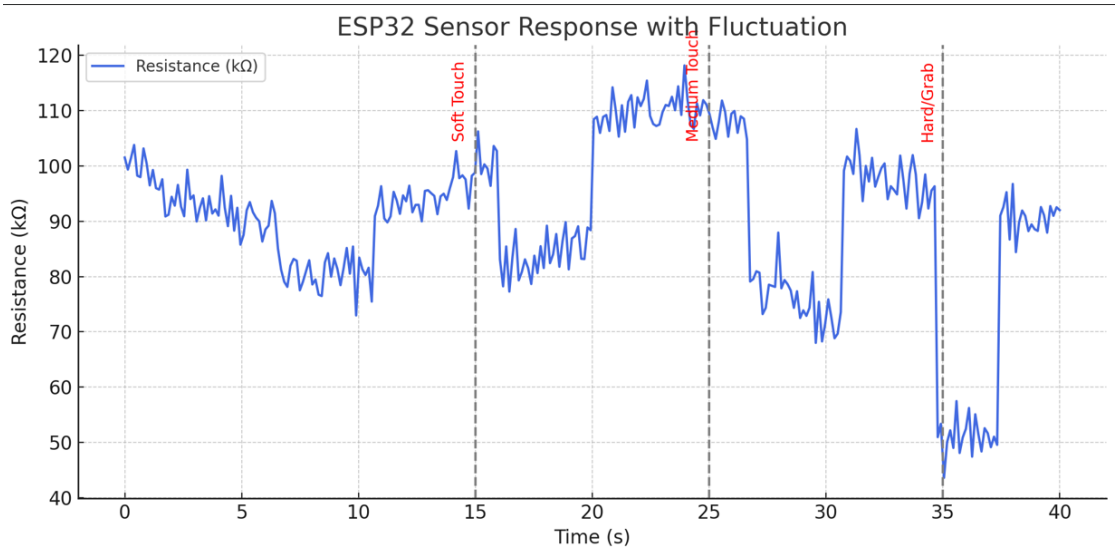


Figure 4.9: Sample sensor output response using ESP32 showing signal fluctuations.

4.2.4 Final Integration and Validation

To finalize the sensor evaluation, two applied tests were carried out. The first used an FSR to verify basic detection capabilities, and the second focused on confirming the performance of the capacitive sensor once embedded in a structural setup.

The FSR setup allowed a quick check of binary contact states Figure 4.10. A standard push was applied using a fingertip, and the voltage response was monitored. As shown in Figure 4.11, the sensor output clearly dropped during contact and returned when released. This behavior confirmed a binary distinction between “touch” and “no touch,” though it lacked the resolution needed for graded interaction levels.

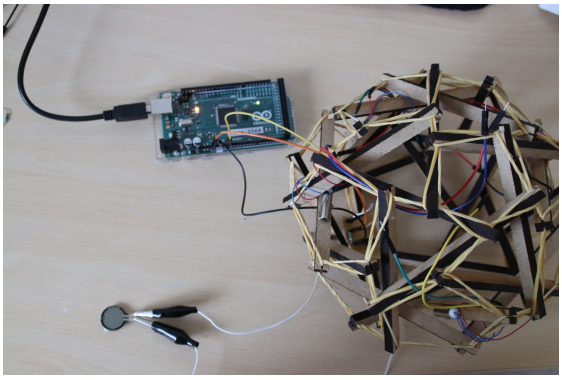
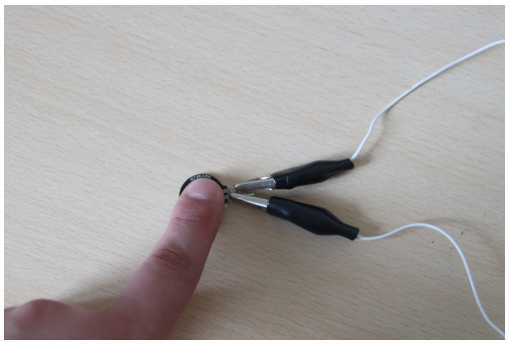


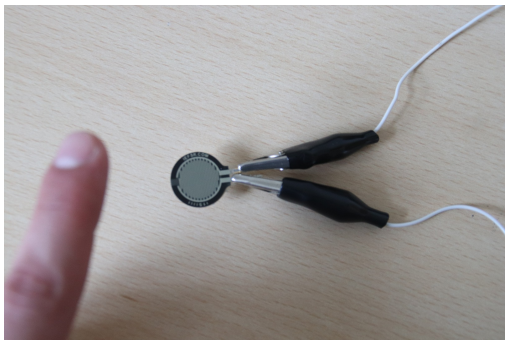
Figure 4.10: FSR setup

A simulated output of the FSR over time is presented in Figure 4.12, showing clear and consistent transitions between contact and release states. While useful for basic detection, the FSR's on–off behavior limited its use for nuanced classification tasks.

In the second test, the capacitive sensor was embedded in a 3D-printed arm structure to validate performance under structural constraint. The result, shown in Figure 4.13, confirmed that the sensor maintained responsiveness even when fixed along a flexible support. Pressure changes continued to generate smooth and accurate resistance shifts, validating its reliability in embedded conditions.



(a) FSR under finger pressure.



(b) FSR in resting state.

Figure 4.11: Binary state response using an FSR sensor.

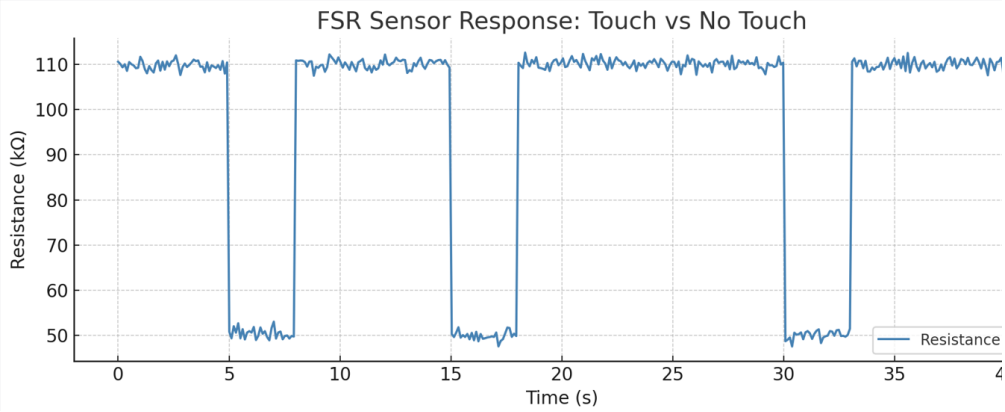


Figure 4.12: Simulated FSR signal showing "touch" and "no touch" transitions.

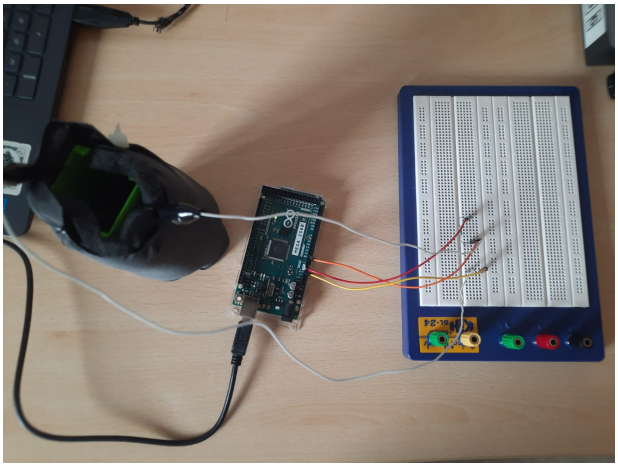


Figure 4.13: Capacitive sensor embedded in a 3D-printed arm setup.

4.3 Vibration Feedback Trials

4.3.1 Single Motor Test

The initial testing of the vibration feedback system began with a single motor to validate basic responsiveness and wiring stability. This level sought to ensure that vibration could be consistently induced by the Arduino microcontroller code.

The Arduino Mega was used to connect a single vibration motor, which was then programmed to turn on when an identified resistance threshold was achieved. This motor was installed with the FSR without any additional covering to allow for tactile inspection and direct viewing.

Once activated, the motor produced consistent pulses with sufficient strength to be perceived through the soft interface. While no graded feedback was implemented at this point, the trial confirmed that the hardware performed as expected and could be used as a foundation for developing intensity-based mapping in later stages.

This test provided a baseline for actuation behavior, confirming that the hardware setup was sound before introducing more motors or complex control logic.

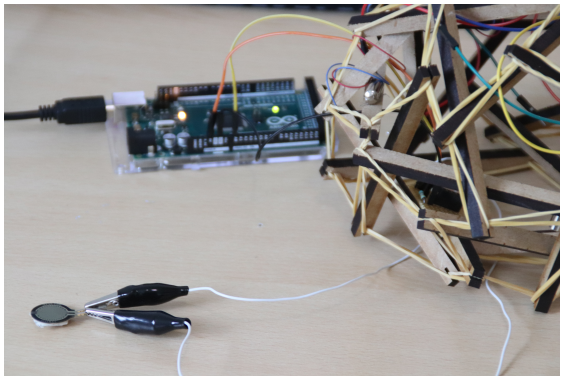


Figure 4.14: Initial vibration motor test setup with FSR placement.

4.3.2 Full Motor Array Test

To evaluate the system's capability to manage multiple actuators, three vibration motors were connected in a shared circuit. This setup allowed all motors to draw power from the same source while being triggered independently through their respective control lines.

Each motor was programmed to turn on sequentially, based on sensor input conditions defined earlier. The goal of this trial was to confirm that signal stability and actuation strength were preserved as more motors were introduced. The Arduino Mega was used again for control due to its sufficient GPIO availability and consistent output performance.

The test confirmed that each motor could be activated in order without interference or delay. Each trigger point led to a corresponding change in the response signal, indicating proper actuation. This verified that simultaneous or staggered feedback could be supported without affecting system reliability.

4.3.3 Final Mapping Strategy

Once the full motor array was confirmed to be operational, a response mapping system was developed to link touch levels with vibration intensity. The aim was

to create a clear and scalable feedback method that could reflect the degree of physical interaction without requiring complex control logic.

The mapping was straightforward: each increase in touch intensity activated one additional motor. This provided four distinct stages of vibration, as listed below:

- Soft Touch → 1 motor
- Medium Touch → 2 motors
- Hard Touch → 3 motors
- Grab → 4 motors

All motors were distributed evenly inside the structure, allowing both the strength and spatial coverage of vibration to change based on the interaction. This setup made the response more intuitive and allowed users to sense feedback variation through both amplitude and location.

The mapping was tested under actual interaction situations. Each touch level precisely activated a suitable number of motors, without delay or misreading. The system handled transitions correctly and demonstrated no instability in repeated trials.

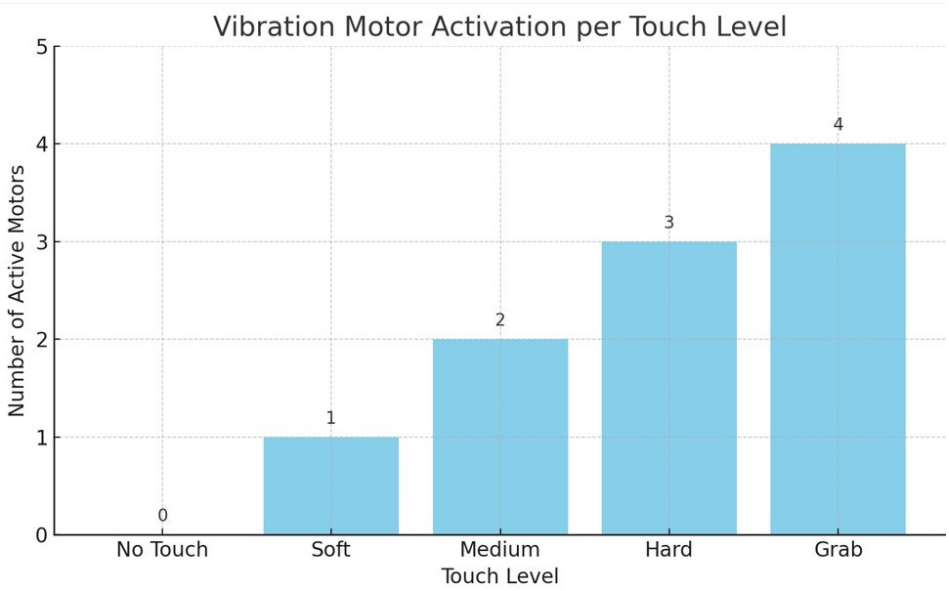


Figure 4.15: Bar graph showing the mapping between touch intensity and number of activated motors.

4.4 Breathing Simulation Trials

4.4.1 Servo Synchronization

The first breathing simulation test focused on verifying the synchronized motion of two servo motors using sweep code (as described in Chapter 5). These motors were installed in opposing positions within the structure and connected to the Arduino Mega, which handled their control in parallel.

The motors were programmed to move in a smooth sweep between two defined angles, replicating the slow rise and fall of a relaxed breathing pattern. Timing parameters were adjusted manually to avoid sudden motion or jitter, and to ensure both servos moved consistently with no noticeable delay between them.

As shown in Figure 4.16, the test setup confirmed that the dual-motor configuration produced a unified and repeatable motion. The mechanical parts moved without noise or backlash, and the transition between positions was smooth enough to be felt as a continuous inflation–deflation cycle when touching the foam. This step served to validate the mechanical alignment and basic feasibility of the breathing effect before adding variation based on input intensity.

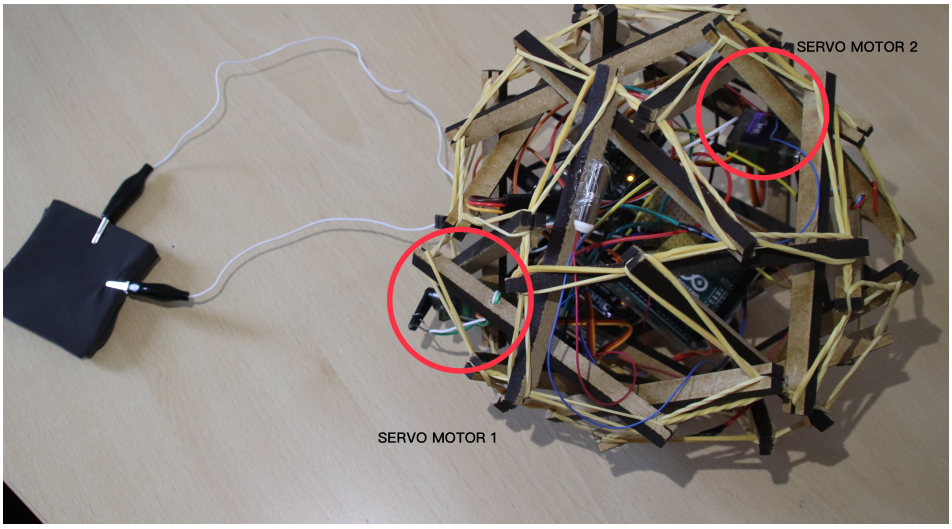


Figure 4.16: Dual-servo motor setup for breathing simulation. Motors are positioned on opposite sides of the internal structure and synchronized for smooth sweeping motion.

4.4.2 Intensity-Linked Breathing

Following successful servo synchronization, the breathing simulation was expanded to vary in real time according to interaction strength. This allowed the system to produce richer feedback by adjusting the movement intensity of the servo motors based on touch level.

The input from the capacitive sensor was classified into five stages. Each one triggered a specific servo behavior defined by motion speed and angular range:

- **No Touch** → 0° (stationary)
- **Soft Press** → sweep between 0° and 180°
- **Medium Press** → sweep between 4° and 180°
- **Hard Press** → sweep between 8° and 180°
- **Grasp** → sweep between 10° and 180°

Each increase in intensity expanded both the amplitude and speed of motion. The breathing rhythm thus became more expressive as interaction strength grew.

Timing and angle values were adjusted manually to maintain fluid transitions, avoiding mechanical jitter. As shown in Figure 4.17, the servo angle pattern closely follows interaction changes, allowing the structure to emulate slow or fast breathing in a way that is perceptible through both sight and touch.

This phase validated the motor logic for pressure-adaptive motion and showed that a simple linear mapping could deliver a believable breathing effect when embedded inside the structure.

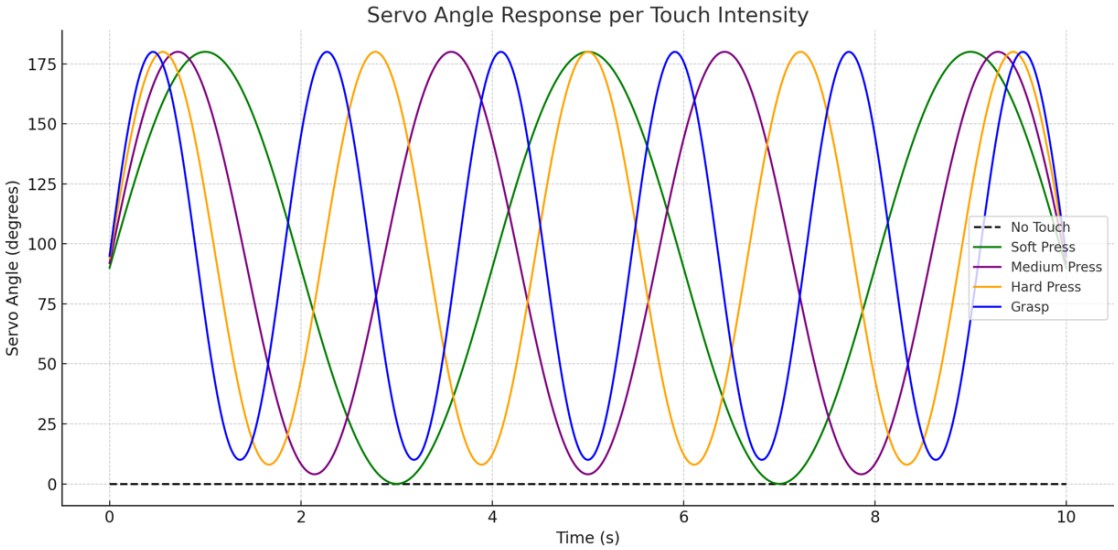


Figure 4.17: Servo response angle over time for varying interaction intensities. Each touch level maps to a different sweep range, demonstrating adaptive breathing simulation.

4.5 Full System Testing

4.5.1 Optimized Fabric-Embedded Circuit Testing

The final configuration of the capacitive sensor circuit was verified at this point once it was fully integrated into the ball. This configuration was used since it demonstrated the best classification consistency in earlier experiments using an Arduino Mega 2560 and a 2 MΩ resistor.

To preserve the wireless usability while still benefiting from the Mega's superior analog performance, a 7.5V battery pack was integrated directly inside the structure. This allowed the setup to run independently from USB power, unlike the ESP32-based tests, and still preserved the untethered, user-friendly interaction feel (Figure 4.18).

Figure 4.18: Embedded battery pack powering the Arduino Mega.

The actual sensor was placed under the layer of fabric and connected via soft internal connections. Response was verified by testing a series of interaction types:

- **No Touch** (idle state)
- **Soft Touch** (light fingertip contact)
- **Medium Press** (moderate finger pressure)
- **Hard Press** (firm pressing with the finger)
- **Grab** (entire hand squeezing the surface)

Each interaction consistently produced a distinct and measurable resistance signal. These values were analyzed using multiple methods. As shown in Figure 4.19, a clear downward trend in resistance was observed as pressure increased. The bar chart further confirmed that each interaction category held a reliably separate resistance band.

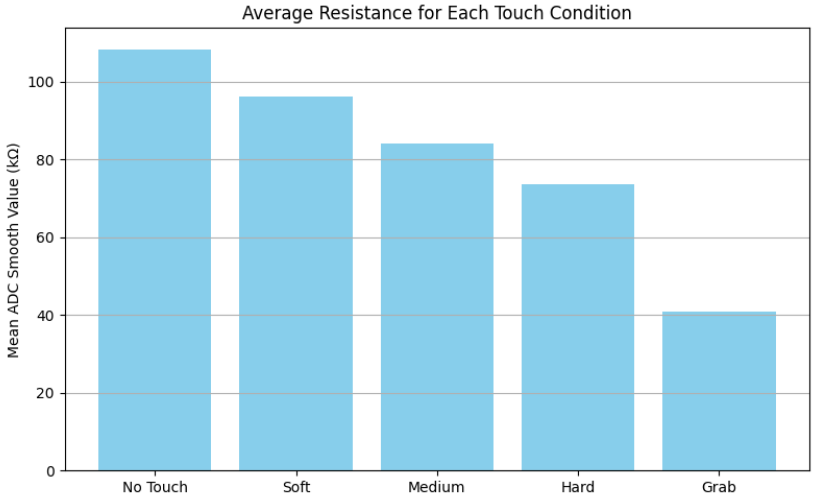


Figure 4.19: Bar graph of resistance values per interaction level.

Figure 4.20 presents the resistance output over time, capturing the real-time variation and pauses between events. Additionally, grouped sensor response curves in Figure 4.21 reinforce the system's stability across repeated applications of the same interaction levels

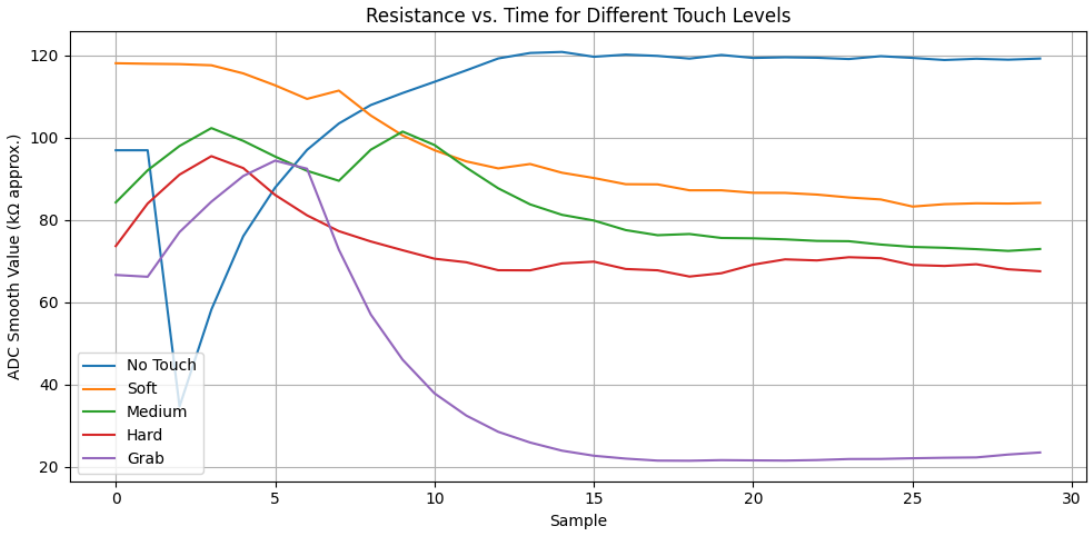


Figure 4.20: Time-based sensor output for sequential interactions.

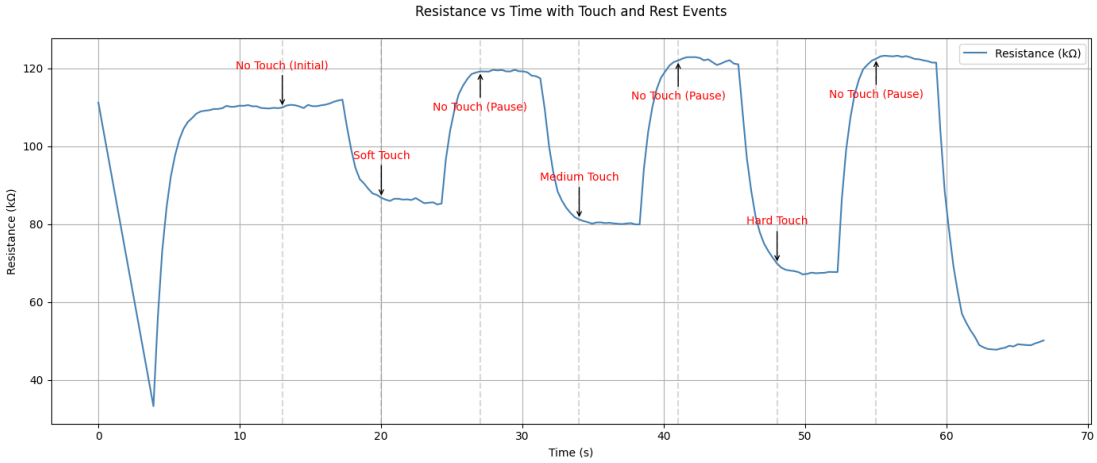


Figure 4.21: Grouped sensor curves across interaction stages.

To visually document the classification process, Figures 4.22 to 4.26 display actual photographs of finger placement and contact for each of the five pressure types, from idle to full grab. These images link the physical gesture directly to the data being recorded.

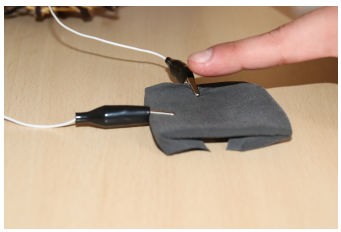


Figure 4.22: No Touch – idle sensor state.

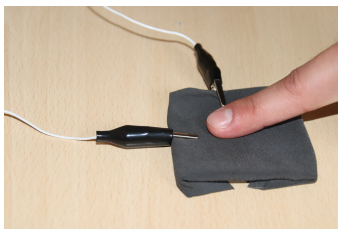


Figure 4.23: Soft Touch – light fingertip contact.

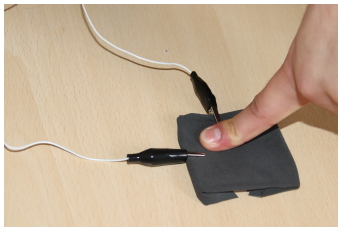


Figure 4.24: Medium Press – moderate finger pressure.

Figure 4.25: Hard Press – firm finger pressure.

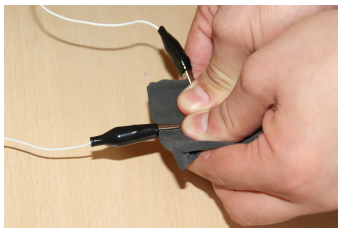


Figure 4.26: Grab – full-hand pressure.

Finally, Figures 4.27 and 4.28 show the final hardware assembly—first uncovered, and then with the full fabric wrap. In both cases, the system preserved its sensitivity and output clarity. This confirmed that the embedded configuration worked as intended and delivered consistent classification accuracy under normal usage.

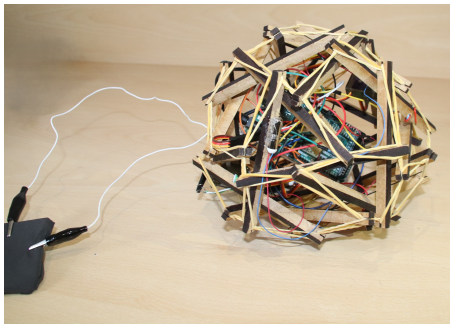


Figure 4.27: Optimized circuit inside the ball without fabric.

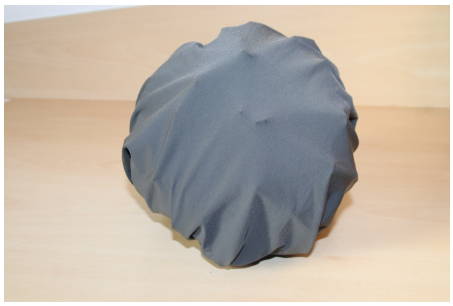


Figure 4.28: Final assembly with fabric covering.

4.5.2 Classification Accuracy Analysis

After embedding the final circuit inside the ball, classification performance was evaluated based on five touch levels. Each interaction type was tested repeatedly to confirm that the sensor could produce consistent and separable outputs without overlap or drift.

For every category, the resistance levels did not change during each stage of the testing. Between experiments, no adjustments and recalibration or just a minor once were required. As can be seen in Figure 4.19, each level had a distinct resistance range, and the value decreased significantly as pressure approached.

Figure 4.20 illustrates the sensor output over time during a full interaction cycle. The signal responded clearly to both quick and prolonged contact, and it reliably returned to baseline after each touch. Transition timing was smooth, and no significant lag or signal noise was observed.

It should be noted that this configuration is an operational prototype. Although the system performed reliably during testing, adjustments may yet be necessary for future applications. For instance, if the fabric structure changes in size or if sensor wires are redirected within, the resistance ranges may need to be modified. These enhancements are expected to be included in subsequent editions.

Despite this, the results confirm that the sensor logic is effective in its current form, and the signal quality is high enough to drive responsive feedback without the need for additional filtering or correction.

4.6 Comparative Overview Table

This section presents a summary of the main improvements made during testing. Each system element was adjusted or rebuilt more than once to meet both performance and usability targets. The following table outlines how each component changed from its initial version to the current state used in the final prototype.

Table 4.1: Comparison of system modules between early trials and final implementation.

Subsystem	**Initial Trial**	**Final Outcome**	**Remarks**
Sensor	Readings were unstable using Uno or ESP32 with lower resistance values	Arduino Mega and 2 MΩ resistor gave consistent and repeatable readings	The Mega's better analog handling and stable reference helped improve touch separation
Vibration	Single motor, on/off feedback without stages	Multi-motor setup with feedback mapped to four pressure levels	Gradual response gave more clarity to users and matched interaction stages better
Breathing	Simple sweep loop, constant speed	Touch-controlled variation in sweep angle and speed	Behavior became more expressive and adjusted naturally to pressure level
Full System	No fabric, USB-powered, exposed components	Final build with internal power and full outer fabric	Wireless setup maintained stability, and comfort improved with fabric layering

While some adjustments were minor, such as tuning the resistor or servo angles, others — like embedding the power supply or mapping feedback more clearly — had a significant effect on how the prototype behaved in real usage conditions. The summary here provides a practical overview of how the system evolved.

4.7 Summary of Experimental Findings

This section outlines the test results. Each component of the system (sensor, vibration, and breathing) was adjusted through practical experiments until it could respond consistently and meaningfully during human interaction.

For the capacitive sensor, the biggest gain came from switching to the Arduino Mega with a 2 MΩ resistor. This configuration provided the most steady readings, especially when embedded into the fabric. Thresholds remained consistent across touch levels, and once tuned, recalibration was rarely required.

In terms of vibration feedback, the system moved from a single motor test to a four-stage setup where motor count and intensity reflected pressure strength. The result was a more understandable and intuitive response pattern, which users could feel clearly through both amplitude and coverage.

The breathing simulation also progressed. The early tests used a constant sweep angle, but this was later adapted based on touch input. That made the movement feel more natural, especially when the servo angles changed smoothly depending on how much pressure was applied.

For the final assembly, all subsystems were installed together inside the soft ball with no external wiring. Even after the foam and fabric layers were added, the system kept working smoothly. The sensor stayed responsive, and both feedback modules reacted without delay or signal distortion.

While some areas still need tuning—such as adjusting thresholds when the fabric geometry changes or fine-tuning the feedback curves—the current setup shows that the design is sound. The system reliably translates user input into output feedback, without needing extra filters or correction layers.

Chapter 5

System Programming & Implementation

5.1 System Control Overview

This chapter outlines how the control logic was built and tested using the Arduino platform. Each function (sensing, vibration, and servo motion) was developed and tested individually before being integrated into a complete system.

The Arduino Mega 2560 was used in all final tests due to its stable analog readings and sufficient I/O support. While the ESP32 was initially considered, the Mega proved more reliable when working with resistive sensors, especially after switching to a 2 MΩ resistor.

Development started with basic sketches. For example, early code was used to read analog values from the FSR to confirm that touch pressure produced readable and repeatable resistance changes. Another sketch tested servo motion through a basic sweep function to verify synchronization. These individual tests were critical before attempting to combine subsystems.

The final version of the code combined the capacitive sensor classification, vibration mapping, and servo control into one continuous loop. No filtering or external libraries were used, aside from `Servo.h`. Each part of the interaction was managed through basic conditional logic, keeping the implementation simple and adjustable. Based on the input intensity, the code triggered a matching number of motors and set servo motion to the appropriate angle range.

The following sections walk through each step of this progression—from early sensor reading to full-system control—highlighting which sketches were used, what was tested, and how the system behavior evolved from isolated functions to a synchronized feedback loop.

5.2 Interaction System Logic

5.2.1 Capacitive Touch Classification Logic

This project used a capacitive sensor made of conductive foam to detect different levels of physical interaction. The sensor was connected to analog pin A0 and acted as a variable resistor within a voltage divider. To ensure signal consistency across tests, a calibration phase was added at the beginning of the program.

During calibration, the sensor was left untouched, and 100 analog readings were collected. The average of these readings was used as a baseline for future comparisons:

Listing 5.1: Sensor Calibration Code

```
Serial.println("Calibrating... DO NOT TOUCH OR STRETCH the sensor.");
delay(1000);

long sum = 0;
for (int i = 0; i < 100; i++) {
    sum += analogRead(SENSOR_PIN);
    delay(10);
}
baseline = sum / 100;
Serial.print("Calibration Complete. Baseline: ");
Serial.println(baseline);
```

Based on the analog measurement after calibration, the system calculated resistance values using a condensed form of Ohm's Law. Real-time processing of the values was then done within the loop:

Listing 5.2: Sensor Resistance Calculation

```
rawValue = analogRead(SENSOR_PIN);
float voltage = rawValue * (5.0 / 1024.0);
float I = voltage / 500000.0;
float VRx = 5.0 - voltage;
resistance = VRx / I;
```

To smooth out fluctuations and ensure better consistency, a simple filter was applied:

Listing 5.3: Exponential Smoothing

```
float smoothed = (prev * 0.7) + (input * 0.3);
```

The classification of touch levels was based on raw resistance (not the smoothed value).The increase in the pressure applied causes the resistance decrease, allowing a clear separation between different levels of interaction. Using the standard `if-else` structure, the logic was developed, giving each resistance range a unique interaction label:

Listing 5.4: Touch Classification Logic

```
if (resistance > noTouch) {
    Serial.println("No Touch");
}
else if (resistance > softTouch) {
    Serial.println("Soft Touch Detected");
}
else if (resistance > mediumTouch) {
    Serial.println("Medium Touch Detected");
}
else if (resistance > hardTouch) {
    Serial.println("Hard Touch Detected");
}
else {
    Serial.println("Grab Detected");
}
```

The threshold values were tuned manually based on observation:

Listing 5.5: Threshold Values

```
float noTouch = 200000;
float softTouch = 170000;
float mediumTouch = 140000;
float hardTouch = 130000;
float grabTouch = 100000;
```

This classification was first tested on its own, without any motor or servo feedback. Values were printed on the serial monitor to confirm that the system could consistently detect each interaction level. The modular structure made it easier to reuse this logic when vibration and motion feedback were added later.

5.2.2 Vibration Feedback Control

After classifying the different touch levels, the system needed a way to provide physical feedback. For this, four vibration motors were added. Each one was connected to a digital output pin, and the idea was simple: the stronger the touch, the more motors would turn on.

The control part was handled in the same `classifyTouch()` function. Based on the resistance value from the sensor, one to four motors would be activated using a set of `digitalWrite()` commands.

The code used a basic step-wise logic like this:

Listing 5.6: Vibration Motor Activation

```
if (resistance > noTouch) {
    digitalWrite(8, LOW);
    digitalWrite(11, LOW);
    digitalWrite(9, LOW);
    digitalWrite(10, LOW);
}
else if (resistance > softTouch) {
    digitalWrite(8, HIGH);
    digitalWrite(11, LOW);
    digitalWrite(9, LOW);
    digitalWrite(10, LOW);
}
else if (resistance > mediumTouch) {
    digitalWrite(8, HIGH);
    digitalWrite(11, HIGH);
    digitalWrite(9, LOW);
    digitalWrite(10, LOW);
}
else if (resistance > hardTouch) {
    digitalWrite(8, HIGH);
    digitalWrite(11, HIGH);
    digitalWrite(9, HIGH);
    digitalWrite(10, LOW);
}
else {
    digitalWrite(8, HIGH);
    digitalWrite(11, HIGH);
    digitalWrite(9, HIGH);
    digitalWrite(10, HIGH);
}
```

Each `digitalWrite()` statement turns one motor on or off. Pin 8 was used for the first motor, then 11, 9, and 10. A soft press triggered just one of them. With a harder press, two or three would turn on. A full grasp activated all four at once.

Binary behavior was used by the motors so there was no need for PWM and variable speed integration. This prevented flickers or delays that might appear with more intricate control techniques and kept things straightforward and responsive.

This setup worked well in testing. The feedback was easy to feel and easy to adjust. If needed, the thresholds could be changed quickly, since everything was defined with constants at the top of the script. Having each output clearly mapped

also helped when debugging or isolating one part of the system.

5.2.3 Servo Motion Control

Before connecting the servo motor to any interaction logic, a single sweep sketch was used to test it. By doing this, the motor was less likely to noise or become stuck and to move as designed. The motion was programmed to move from 0° to 180° and back again using two servos attached to pins 7 and 12.

The movement happened in 10-degree steps with a short delay, just to keep things smooth and easy to observe. Here's the basic structure:

Listing 5.7: Sweep test for two servos

```
myservo.attach(7);
myservo2.attach(12);

for (pos = 0; pos <= 180; pos += 10) {
    myservo.write(pos);
    myservo2.write(pos);
    delay(15);
}
for (pos = 180; pos >= 0; pos -= 10) {
    myservo.write(pos);
    myservo2.write(pos);
    delay(15);
}
```

Running this gave a quick check that everything was wired correctly and that the motion was clean across the full range. It also made it easier to see what kind of speed or range might feel right later on, once linked to interaction.

At this stage, the servo was not yet connected to the sensor or any feedback system. The goal was just to confirm it worked properly on its own. That made debugging easier and avoided confusion while tuning motor response. The way this motion was later tied to touch intensity and integrated with the rest of the system is explained in the next section.

5.3 Full System Integration

After testing the sensor, vibration motors, and servo motion separately, the final step was to combine them into one working loop. This was done using the `trialwithservo3.ino` sketch, which brought everything together in a way that kept the system responsive and easy to adjust.

The first thing added was the breathing feedback. Instead of using fixed angle values for each touch level like before, a custom function called `breathing(step)`

was used. The idea was simple: smaller step values made the servo move slowly, while higher steps made the motion quicker and wider — kind of like simulating deeper or faster breathing.

Listing 5.8: Breathing function for servo motion

```
void breathing(int step){
  for (pos = 0; pos <= 180; pos += step) {
    myservo.write(pos);
    myservo2.write(pos);
    delay(15);
  }
  for (pos = 180; pos >= 0; pos -= step) {
    myservo.write(pos);
    myservo2.write(pos);
    delay(15);
  }
}
```

This function was called from inside the `classifyTouch()` function, depending on the level of resistance detected. For example, a soft touch would only trigger the first motor and a slow servo motion, while stronger inputs would increase both the number of motors and the breathing intensity.

Listing 5.9: Conditional breathing based on touch level

```
else if (resistance > softTouch) {
    digitalWrite(8, 1);
    breathing(1);
}
else if (resistance > mediumTouch) {
    digitalWrite(8, 1);
    digitalWrite(11, 1);
    breathing(2);
}
```

Each motor was wired to a separate pin (8, 11, 9, and 2), and they were turned on one by one as the resistance got lower. This made it easy to adjust feedback just by changing the thresholds defined at the top of the code.

To make sure everything worked reliably together, the loop was kept simple. No heavy libraries were used apart from `Servo.h`, and the smoothing function for the resistance value was lightweight. Also, a short calibration was done at the beginning to set the baseline for the sensor:

Listing 5.10: Sensor calibration loop

```
for (int i = 0; i < 100; i++) {
    sum += analogRead(SENSOR_PIN);
    delay(10);
}
baseline = sum / 100;
```

This helped the sensor stay accurate even if environmental conditions shifted slightly. The whole system ran in a single loop with basic `if-else` logic, which made it easy to debug and adjust later.

In the end, the combined script worked as intended. The touch input controlled both tactile (vibration) and motion (servo) feedback, and the interaction felt clear and responsive. All the separate tests came together in one system that was stable and easy to tune.

5.4 Python-Based Plotting and Visualization

The plots shown in the previous chapter were created using Python scripts instead of the Serial Plotter built into the Arduino IDE. While the Serial Plotter was useful in early tests, it lacked flexibility for saving graphs, adding labels, or adjusting the view. Using Python made it easier to read the data in real time, mark different phases of interaction, and generate clearer results.

Three scripts were written and used during different parts of the development. The first one, `python_for_plotting.py`, was written to simply read resistance values from the serial port and plot them live. This was mainly used to check whether the sensor was working properly and how the signal responded to different touch levels. A small part of the code used looked like this:

```
resistance = float(line.strip().split('=')[1])
plt.plot(timestamps, resistance_values)
```

This gave a simple line graph that updated in real time.

The second script, `pthon_plotting_condition_with_notouch.py`, was used to test each interaction level separately. The idea was to apply one type of touch at a time, record the data, and then calculate the average resistance value. The results were shown as a bar chart. The section below shows how the labels were used:

```
labels = ['No Touch', 'Soft', 'Medium', 'Hard', 'Grab']
plt.bar(labels, average_resistances)
```

This made it easier to compare the values and check that the thresholds used in the Arduino code were reasonable.

The last script, `python_optimized_code_plotting.py`, was used to create the final labeled plots. It added vertical lines and text annotations to mark when each new touch phase started. This made the plots easier to read when showing how the resistance changed across time. A typical part of the code used for this was:

```
plt.annotate(labels[i], ...)
plt.axvline(...)
```

This setup made it easier to see whether the classification was working as intended and was also helpful during debugging.

All the plots shown earlier in chapter 4 were created using these scripts. Depending on the test, one of them was used with small adjustments. Using Python instead of the Serial Plotter made a big difference in how clearly the results could be presented.

5.5 Discussion

Some practical choices were made during the integration to keep the system simple and reliable. The Arduino Mega was used instead of the ESP32 mainly because it gave more stable analog readings and had more pins, which helped during testing.

For vibration, digital control was enough. There was no need for PWM since just turning the motors on and off was clear and effective. The same idea was followed with the servo — step-based motion was easier to tune and didn't add complexity.

A basic calibration was added at startup to adjust the baseline each time. This helped reduce noise and made the readings more consistent.

The smoothing of resistance was done with a small filter directly in the code, without any libraries. This kept the response fast.

Python-based plotting was also a useful part of the testing process. Instead of relying on the Arduino Serial Plotter, all sensor responses were logged and visualized using custom Python scripts. This made it easier to verify whether the classification logic and thresholds were working as expected.

Still, some limits remained. The thresholds were hardcoded, and the servo motion used blocking delays. Everything ran in one loop, so there was no multitasking. But for the scope of this project, it was enough to show the concept clearly.

Chapter 6

Conclusion and Future Work

In recent years, there has been growing interest in using simple, adaptive systems to explore human–machine interaction beyond conventional interfaces. This thesis focused on developing a prototype that interprets the intensity of human touch and responds through vibration and mechanical motion. By combining a capacitive sensor with straightforward control logic, the aim was to create a compact and transparent interaction model. The work prioritized clarity, simplicity, and adaptability over complex architecture, offering a foundation for future exploration in soft robotics, wearable feedback, and responsive therapeutic devices.

6.1 Conclusion

This work demonstrated the feasibility of using a minimal setup to deliver graded tactile interaction in real time. A stretchable capacitive-resistive sensor was used to detect varying levels of pressure, and the response was mapped to a multimodal output system based on vibration and servo motor motion. The intent was not to build a final product, but to validate a design path that prioritizes usability, signal clarity, and low system complexity.

The development was guided by a hands-on, modular approach. Each subsystem—sensor reading, signal smoothing, threshold calibration, vibration feedback, and servo actuation—was tested in isolation before integration. The logic relied on simple comparisons and blocking delays to maintain clarity and ease of adjustment. Despite these basic tools, the final integration proved robust, with reliable detection across five levels of input.

The main contributions of this work can be summarized as follows:

- **Single Sensor Integration:** A soft, stretchable capacitive-resistive sensor was employed to detect graded physical input, avoiding the need for multiple sensing layers or external conditioning hardware.

- **Touch Classification Logic:** Custom thresholds were defined and calibrated to categorize five interaction levels (no touch, soft touch, medium touch, hard touch, grab), based solely on raw analog resistance values. The logic remained simple, yet showed stable detection across multiple tests.
- **Multimodal Feedback System:** Distinct vibration patterns were assigned to each touch category using digital signals. Additionally, servo motion was programmed to mimic a breathing-like response, offering both immediate and slow feedback to complement the sensor input.
- **Independent Microcontroller Implementation:** The entire interaction loop ran on an Arduino Mega 2560, chosen for its stable analog readings and generous I/O capacity. No libraries or multitasking approaches were used, ensuring clarity and full control of timing and feedback.
- **Data Visualization via Python:** To better understand and refine the sensor behavior, Python plotting scripts were created. These allowed real-time visualization of both raw and smoothed resistance values, offering more control and insight than Arduino's built-in tools.
- **Incremental Testing and Debugging:** Each component — from reading and calibration to feedback triggering — was implemented and verified individually before integration. This stepwise development helped avoid interference between parts and made debugging clearer.

In summary, this prototype confirms that a compact, low-cost design can still provide meaningful and responsive tactile feedback. Although this version focused on simplicity and validation of the interaction principle, it sets the stage for future improvements, such as feedback tuning, modular packaging, or wireless operation in wearable or therapeutic scenarios.

6.2 Future Work

While the current prototype met the main goals of clarity and functionality, several areas remain open for improvement or extension. Future work could build on the foundation laid here in the following directions:

1. **Hardware Optimization:** Miniaturizing the setup by using a smaller board (such as an ESP32) and reducing wiring could make the system more suitable for wearable or embedded use.
2. **Non-blocking Control Logic:** Replacing blocking delays with interrupt-based or non-blocking loops would improve responsiveness and open the door to multitasking or wireless communication.

3. **Modular Feedback Control:** Splitting vibration and motion control into separate modules would make it easier to tune them independently, or even switch between modes dynamically.

4. **Dynamic Threshold Calibration:** Adding a learning or adaptive mechanism to calibrate resistance thresholds per user or environment could increase robustness and usability in varied contexts. This could also include automatic adjustment in case of changes to the sensor's physical layout or wiring distances.

5. **3D-Printed Shape Redesign:** The outer shell could be redesigned to be more flexible and ergonomic, especially to ensure safe use by children or in therapeutic settings. Material softness and deformation tolerance could be further improved.

6. **Testing on Larger Conductive Materials:** Extending the sensor tests to wider and less constrained conductive fabrics would help evaluate system performance in more realistic, uncontrolled and harsh environments.

7. **Wireless Operation and Power Management:** Making the system wireless and battery-powered would enable mobile or field use, especially in applications like rehabilitation or stress-relief tools.

8. **Data Logging and User Testing:** Adding onboard data storage or remote logging could help study long-term trends or run structured user studies to evaluate the system's effectiveness.

9. **Expansion to Multi-sensor Inputs:** Extending the platform to support multiple sensors would allow for spatial mapping of touch or more complex interaction patterns.

These enhancements would not only make the system more versatile but also support its adaptation into real-world soft robotics or assistive devices. The current design offers a stable testbed to begin such explorations.

Bibliography

[1] Adaptive Systems Research Group, University of Hertfordshire. *KASPAR the social robot*. https://www.herts.ac.uk/kaspar/the-social-robot. Accessed: 2025-06-16. 2025.

[2] Aaron Biggs / Wikimedia Commons. *Paro therapeutic robot seal (Flickr, 2005)*. https://commons.wikimedia.org/wiki/File:Paro_robot.jpg. CC BY-SA 2.0; accessed 2025-06-16. 2005.

[3] SoftBank Robotics (Aldebaran). *NAO Robot*. https://us.softbankrobotics.com/nao. Accessed: 2025-06-16. 2025.

[4] Hiroyuki Manabe and Wataru Yamada. «A Capacitive Touch Sensing Technique with Series-Connected Sensing Electrodes». In: *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology (UIST)*. ACM, 2017, pp. 565–574. DOI: 10.1145/3126594.3126625.

[5] Bjoern Hartmann and Scott R. Klemmer. «Multi-Touch Kit: A Do-It-Yourself Technique for Capacitive Multi-Touch Sensing Using a Commodity Microcontroller». In: *Proceedings of the ACM International Conference on Tangible and Embedded Interaction (TEI)*. ACM, 2006, pp. 139–146. DOI: 10.1145/3332165.3347895.

[6] Tam Vu, Akash Baid, Simon Gao, Marco Gruteser, Richard Howard, Janne Lindqvist, Predrag Spasojevic, and Jeffrey Walling. «Distinguishing Users with Capacitive Touch Communication». In: *Proceedings of the ACM International Conference on Mobile Systems, Applications, and Services (MobiSys)*. ACM, 2012, pp. 197–210. DOI: 10.1145/2348543.2348569.

[7] Keisuke Watanabe, Ryosuke Yamamura, and Yasuaki Kakehi. «Foamin: A Deformable Sensor for Multimodal Inputs Based on Conductive Foam with a Single Wire». In: *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems (CHI)*. ACM, 2021, pp. 1–13. DOI: 10.1145/3411763.3451547.

[8] Dongyoung Lee and Joonbum Bae. «Individually Addressable Multitouch Sensors Using a Sweep Signal for Minimal Wiring Complexity». In: *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology (UIST)*. ACM, 2017, pp. 793–801.

[9] Colin Honigman, Jordan Hochenbaum, and Ajay Kapur. «Techniques in Swept Frequency Capacitive Sensing: An Open Source Approach». In: *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)*. Goldsmiths, University of London, 2014, pp. 441–444. URL: https://www.nime.org/proceedings/2014/nime2014_515.pdf.

[10] Munehiko Sato, Ivan Poupyrev, and Chris Harrison. «Touché: Enhancing Touch Interaction on Humans, Screens, Liquids, and Everyday Objects». In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI)*. ACM, 2012, pp. 483–492. DOI: 10.1145/2207676.2207743.

[11] Yutaro Ono, Yuhei Morimoto, Reiji Hattori, Masayuki Watanabe, Nanae Michida, and Kazuo Nishikawa. «Smart Steering Wheel with Swept Frequency Capacitive Sensing». In: *IEICE Transactions on Electronics* E100.C.12 (2017), pp. 1015–1021. URL: https://globals.ieice.org/en_transactions/electronics/10.1587/transele.E100.C.972/_p5.

[12] Jie Gao, Leijing Zhou, Miaomiao Dong, and Fan Zhang. «Expressive Plant: A Multi-Sensory Interactive System for Sensory Training of Children with Autism». In: *Proceedings of the 2018 ACM International Joint Conference on Pervasive and Ubiquitous Computing and the 2018 ACM International Symposium on Wearable Computers (UbiComp/ISWC)*. ACM, 2018, pp. 195–198. URL: https://dl.acm.org/doi/abs/10.1145/3267305.3267588.

[13] Umer Muhammad, Corina Sas, and Niousha Chalabianloo. «Exploring Personalized Vibrotactile and Thermal Patterns for Affect Regulation». In: *ResearchGate* (2021). URL: https://www.researchgate.net/publication/352802012_Exploring_Personalized_Vibrotactile_and_Thermal_Patterns_for_Affect_Regulation.

[14] Jee-Hye Seo, Preeti Aravindan, and Apirat Sungkajun. «Toward Creative Engagement of Soft Haptic Toys with Children with Autism Spectrum Disorder». In: *Proceedings of the Creativity and Cognition Conference*. ACM, 2017. DOI: 10.1145/3059454.3059474.

[15] Thomas McDaniel and Sethuraman Panchanathan. «Therapeutic Haptics for Mental Health and Wellbeing». In: *Haptic Interfaces for Accessibility, Health, and Enhanced Quality of Life*. Springer, 2019, pp. 129–143. DOI: 10.1007/978-3-030-34230-2_6.

[16] Samantha MacDonald, Edward Freeman, and Frank Pollick. «Prototyping and Evaluation of Emotionally Resonant Vibrotactile Comfort Objects as a Calming Social Anxiety Intervention». In: *ACM Transactions on Computer-Human Interaction (TOCHI)* 31.1 (2024). DOI: 10.1145/3648615.

[17] K. Krishnan, Nor Azan Mat Zin, and R. K. Ayyasamy. «Haptic Feedback: An Experimental Evaluation of Vibrations as Tactile Sense in Autistic People». In: *IEEE Access* 12 (2024), pp. 38812–38822. URL: https://ieeexplore.ieee.org/abstract/document/10551815.

[18] Rasmus E. Klausen, Julie C. Andersen, Vanessa F. S. Svarer, Dan W. P. Brændholt, Erik B. Olsen, Jonathan F. Woodward, Heni B. B. Andersen, and Kasper S. W. Pedersen. «Breco: A Soft Social Robot to Alleviate Anxiety Symptoms in Children». In: *Social Robotics (ICSR 2024)*. Springer, 2024. DOI: 10.1007/978-981-96-3522-1_12.

[19] V. Lingampally, M. C. Kumar, and B. K. Sahu. «Wearable Assistive Rehabilitation Robotic Devices—A Comprehensive Review». In: *Machines* 12.6 (2024), p. 415. DOI: 10.3390/machines12060415.

[20] Hao Wu, Bin Fang, Juan Wang, Weidong Chen, and Xinyu Wu. «HBS-1: A Modular Child-Size 3D Printed Humanoid Robot». In: *Robotics* 5.1 (2016), p. 1. DOI: 10.3390/robotics5010001.

[21] Hiroshi Ishihara and Minoru Asada. «Design of 22-DOF Pneumatically Actuated Upper Body for Child Android 'Affetto'». In: *Advanced Robotics* 29.14 (2015), pp. 937–948. DOI: 10.1080/01691864.2015.1046923.

[22] Yuichiro Yoshikawa, Hiroshi Ishiguro, and Minoru Asada. «Development of an Android Robot for Psychological Support of Hospitalized Children». In: *2011 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2011, pp. 2456–2461. DOI: 10.1109/ROBIO.2011.6181654.

[23] *Miniature Waterproof Vibration Motor, DC 3–5V, 18000RPM*. https://www.ebay.com/itm/176776950795. Brand new; eBay item #135607221356; listing ended July 7, 2025. July 2025. URL: https://www.ebay.com/itm/176776950795.

[24] Nicole Penati, Giovanni Beltrami, Francesco Rea, Alessandra Sciutti, and Giulio Sandini. «BRECO: A Breathing Robotic Companion for Children with Neurodevelopmental Disorders». In: *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*. IEEE, 2016, pp. 1086–1091. URL: https://inria.hal.science/hal-01678515v1/document.

[25] Giuseppe Crifaci, Hideki Kozima, and Minoru Asada. «Affetto: A Humanoid with Facial Expressions for Studying Emotional Communication». In: *2015 IEEE International Conference on Development and Learning and Epigenetic Robotics (ICDL-EpiRob)*. IEEE, 2015, pp. 157–158. URL: https://www.researchgate.net/publication/224676913_Development_of_Face_Robot_for_Emotional_Communication_between_Human_and_Robot.

[26] Franca Garzotto, Mirko Gelsomini, Yosuke Kinoe, et al. «Puffy: a mobile inflatable interactive companion for children with neurodevelopmental disorders». In: *Proceedings of the 2017 ACM Interaction Design and Children Conference*. Image taken from research paper. 2017. URL: https://www.researchgate.net/figure/The-inflatable-body-of-Puffy_fig1_316612287.

[27] Robotics Today / Osaka University. *Affetto humanoid baby robot (Osaka University)*. https://www.roboticstoday.com/robots/affetto. Accessed: 2025-06-16. 2011.

[28] Barnabas Robotics. *9g Metal Standard Servo Motor (0–180°)*. https://shop.barnabasrobotics.com/products/9g-metal-standard-servo-motor-180-degrees. Includes metal gears and servo horns; rotates 0–180°. Regular price USD$4.95. 2025. URL: https://shop.barnabasrobotics.com/products/9g-metal-standard-servo-motor-180-degrees.