Boston University
Electrical & Computer Engineering
EC464 Senior Design Project

NeuroToys User Manual

NeuroToys

by
Team 9

Team Members

Adam Shaikh adamparz@bu.edu
Andrés Marquez Santacruz afms@bu.edu
Gabriela Porto Machado gmpm@bu.edu
Mete Gumusayak mgumus@bu.edu
Robert Bona rjbona@bu.edu

NeuroToys User Manual

Table of Contents

Executive Summary
1 Introduction
2 System Overview and Installation
    2.1 Overview block diagram
    2.2 User Interface
    2.3 Physical Description
    2.4 Installation, setup, and support
3 Operation of the Project
    3.1 Normal Operation
    3.2 Abnormal Operation
    3.3 Safety Issues
4 Technical Background
5 Relevant Engineering Standards
6 Cost Breakdown
7 Appendices
    7.1 Appendix A - Specifications
    7.2 Appendix B - Team Information

Executive Summary

NeuroToys is a brain-computer interface (BCI) system capable of remote control of a toy car using nothing more than a user's brain waves and eye winks. By translating EEG signals into real-time actionable commands, NeuroToys demonstrates the potential of non-invasive neural control for recreation and productivity. This project blends several aspects of neuroscience, machine learning, and embedded systems, and is designed primarily with accessibility and engagement in mind. NeuroToys offers a compelling alternative to traditional toys, many of which require fine motor skills that not all individuals may possess.

1 Introduction

Individuals with physical disabilities often face significant challenges in controlling devices and interacting with technology, relying on manual inputs or invasive surgeries that may be inaccessible or impractical. For customers seeking accessible solutions, NeuroToys addresses this critical need for user-friendly and non-invasive technologies that enable effortless control of devices through thought alone. This initiative represents a critical step towards the advancements of human-machine interaction, offering an intuitive and inclusive solution for users with mobility impairments and other physical limitations.

As customers seek solutions to enable device control for users with mobility impairments and encounter shortcomings due to traditional inaccessible, invasive, and costly ways, this project seeks to demonstrate the feasibility of neural-based robotic control while ensuring accessibility, affordability, and versatility for diverse applications.

To achieve this, NeuroToys leverages electroencephalography (EEG) to capture brainwave signals, which are gathered through an external headgear. By processing and classifying this neural activity in real-time, the system translates them into actionable commands for a battery-powered car. This project integrates hardware, such as the Muse 2 EEG headset and an ESP32 microcontroller, with software written in Python for signal processing and command execution. The ultimate goal is to provide a seamless, wireless, and portable solution that bridges the gap between user intent and device functionality.

Figure 1.1 Simplified system architecture diagram, showcasing EEG signal acquisition, real-time processing, and motor control

Key highlights of the project include its focus on accessibility, with reduced setup complexity and cost compared to existing alternatives. The lightweight and portable nature of our set up enhances usability, and classification methods allow for precise motor control. In addition, it is designed to be cost-effective, making it an affordable solution for a broader audience. NeuroToys is not only one implementation of assistive technology, but also a foundation for broader applications, such as rehabilitation, smart home automation, and gaming.

Thus, through an interdisciplinary approach and a user-centric design, we offer an modern, accessible, and affordable solution that redefines the way people interact with technology, empowering disabled individuals and advancing the field of human-machine interaction.

2 System Overview and Installation

The Neurotoys system consists of an off-the-shelf non-invasive EEG headset, an RC car, and the user's personal computer to run associated software. The Muse 2 detects raw brain voltages in µV from electrodes placed on the forehead and relays this information to the computer via Bluetooth. This information is taken through several processing stages, depicted in Figure 2.1, to determine the user's intent and relay the command signal to the car's ESP32 to perform the corresponding action after a brief processing delay.

The Python interface processes this signal by performing a Fourier transform on the AF7 electrode to isolate the beta frequency band from the EEG data, which is associated with focus. Trehe beta power is then calculated (expressed in μV^2), representing the user's focus level. A threshold is established to determine whether a command should be sent to the ESP32, which then controls the forward movement of the toy car. A similar signal processing technique is used for right and left control, where the AF7 and AF8 electrodes on both sides of the forehead go through a band pass filter separately to isolate the gamma frequency band from the raw EEG data. Gamma is a lower frequency range (0.5 - 5 Hz), allowing for isolation of corneoretinal motion (eye-blink) detection instead of fast neural activity. Peak detection algorithms are then used to classify between right and left eye blinks (AF8 vs. AF7 peaks), which in turn send the right and left turn commands to the car.

2.1 Overview block diagram

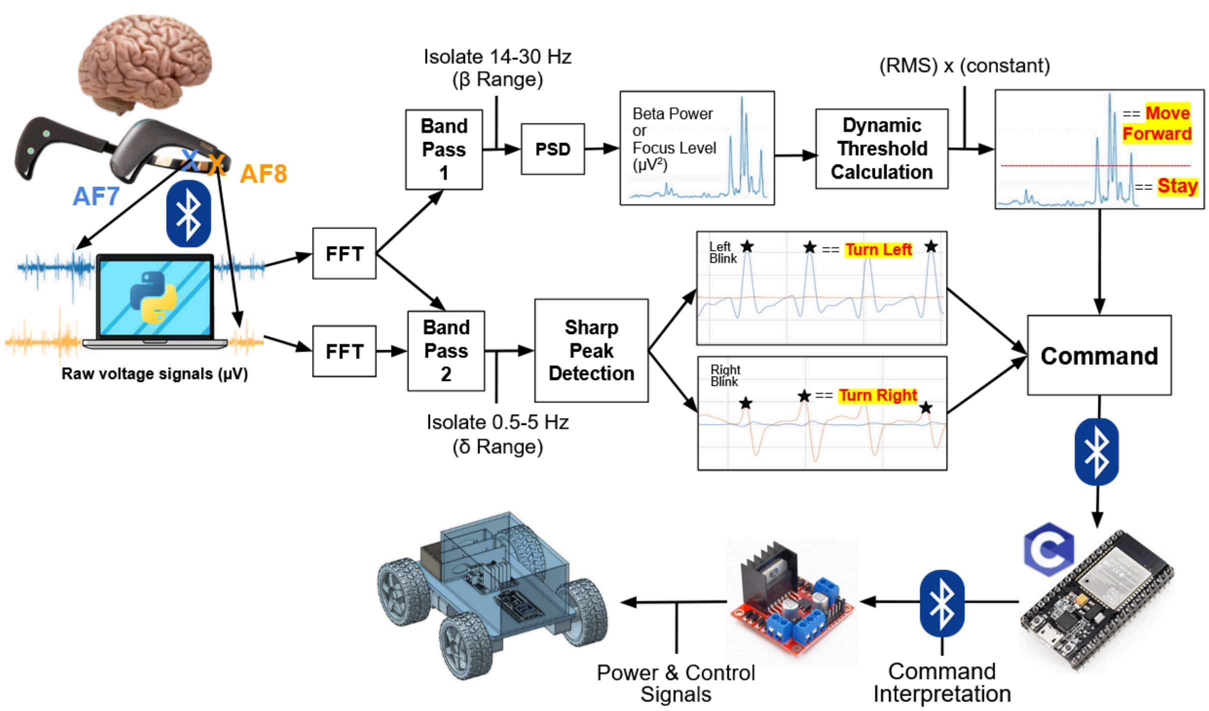

Figure 2.1 Full system architecture diagram, showcasing EEG signal acquisition, real-time processing, and motor control

2.2 User Interface

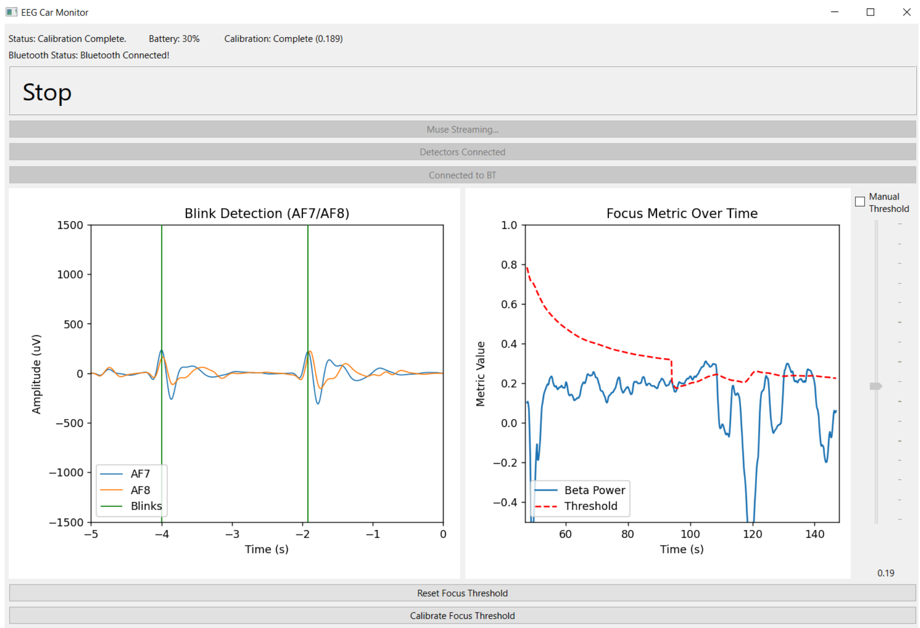

Figure 2.2 Screenshot of Graphical User Interface. Illustrating blink detection and focus levels plots.

2.3 Physical Description

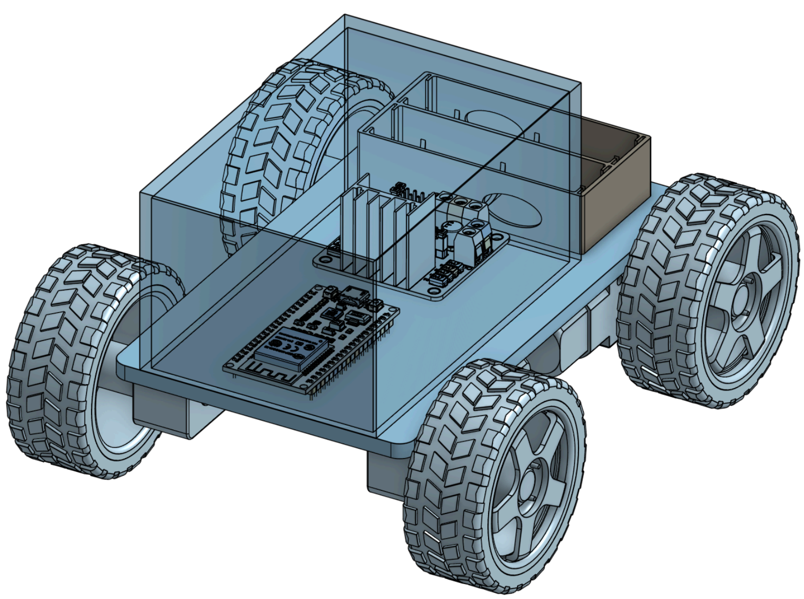

Figure 2.3.1 CAD model of RC car assembly, simplified for visual clarity

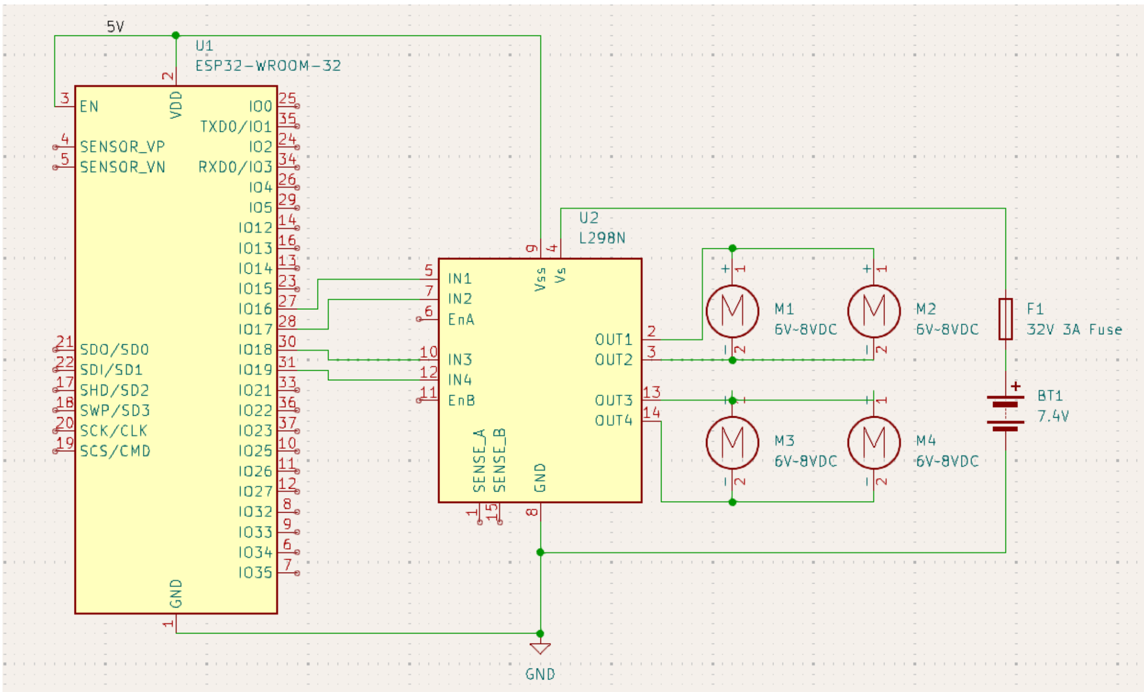

Figure 2.3.2 RC car circuit schematic including ESP32, L298N motor driver, and DC motors

2.4 Installation, setup, and support

The user should only operate the car in a wide open area to prevent collisions. To begin the setup process, download and run the software according to the instructions posted on the NeuroToys GitHub.

Power on the Muse 2 EEG and connect it to your device via Bluetooth. The console log should display a successful connection message — if any issues arise, power cycle the headset and retry. Place the headset firmly against your forehead immediately above the eyebrow, ensuring full contact with the electrodes.

Insert two fully charged 18650 Li-ion batteries into the battery spring clips located at the back of the car. If done correctly, the UI should display two graphs corresponding to real-time focus level and blink detection respectively.

For best results, operate the car in a distraction-free environment, as switching between focused and relaxed states are fundamental for accurate input signal processing. If there is difficulty in triggering the focused state to drive the car forward, users may find it helpful to think of a single, clear tone. If misclassifications are occurring frequently, ensure firm headset contact with the forehead and reset the concentration threshold via the GUI. The car may rapidly turn left and right when there is not full contact with the electrodes, including when the headset is removed. This is considered to be normal behavior caused by low frequency noise present in the environment — consider powering off the car via the power switch located on the side of the battery spring clip prior to removing the headset.

3 Operation of the Project

3.1 Operating Mode 1: Normal Operation

This section outlines the standard procedure for operating the EEG-controlled car using the provided graphical user interface (GUI). Before starting, ensure your system meets the prerequisites: Python 3.12.7 (or compatible) must be installed on a computer with Bluetooth enabled. You also need to install the necessary libraries. Open a terminal or command prompt, navigate to the main NeuroToys project directory (where requirements.txt is located), and execute pip install -r requirements.txt. Finally, carefully put on the EEG headset according to the detailed instructions provided in Section 3.1.1 (Detailed EEG Headset Set-up Procedure).

To launch the application, open a terminal or command prompt, navigate into the muse subdirectory within the main NeuroToys project folder, and run the command python new_gui.py. This will open the main application window (GUI), which resembles Figure 2.2. The GUI features several status indicators to keep you informed: the EEG Status shows the connection state of the Muse headset and data detectors; the Battery Status displays the headset's charge level; the Calibration Status indicates the current stage of the focus threshold calibration; the Bluetooth Status shows the connection state to the car; and the central Movement Display shows the last command sent or the current intended action based on EEG signals (like "Stop", "Forward", "Left", "Right").

The first step in the GUI is to establish a connection with the EEG headset. Click the "Scan and Start Muse Stream" button. The application will search for nearby Muse devices, check the battery of the first one found, and automatically begin the LSL data stream. Ensure that the headset you connected to is the one that is intended. The headset name is displayed at the top of the GUI and on the inner left side of the actual headset. Watch the EEG Status label for updates. Alternatively, you could manually start the stream by running muselsl stream in a separate terminal (this command is included when you install the requirements.txt), but the button method is recommended as it includes the battery check. Once the EEG Status confirms the stream is active, click the "Connect to EEG Stream" button. This action activates the blink and focus detection algorithms, and the corresponding plots will begin displaying live data from the headset.

After connecting to the EEG stream, an automatic calibration process for the focus threshold will commence, as detailed in Section 3.1.2 (Calibration). Follow the on-screen prompts in the EEG Status and Calibration Status labels, which will guide you to relax for about 5 seconds and then concentrate for about 10 seconds. Note that all movement commands to the car are temporarily disabled during this calibration phase. You can re-run this calibration anytime by clicking the "Calibrate Focus Threshold" button.

With the EEG stream connected and initial calibration complete, you can connect to the car. Ensure the Bluetooth car is switched on and positioned near your computer.

The "Connect to Bluetooth Car" button should now be enabled. Monitor the Bluetooth Status label, which should change from "Connecting..." to "Connected!" upon success.

To control the car using your brain activity and eye blinks, follow the methods described in Section 3.1.3 (Control). If desired, you can manually override the automatically calibrated focus threshold. Check the "Manual Threshold" checkbox located beside the Focus plot. Use the vertical slider to set your preferred threshold level. The value is displayed numerically below the slider and visually as a red dashed line on the plot. To revert to the automatic threshold, simply uncheck the box.

To properly shut down the application and disconnect from all devices, close the main GUI window. The program is designed to automatically handle the termination of connections and background processes.

3.1.1 Detailed EEG Headset Set-up Procedure

To correctly put on the EEG headset, gently extend the adjustable earpieces while being mindful not to extend further once you feel tension. Overextension can lead to internal damage in the device. It should be extended so that it is slightly larger than your head size before adjusting it on your head. Place it along the middle of your forehead with the rubber ear sensors resting behind your ears. Earpieces should sit behind your ears, just as a pair of glasses. Adjust both sides simultaneously to tighten it back up for a snug, comfortable fit. Both earpieces should be equally extended to ensure no gaps and that all sensors have good skin-to-sensor contact. The headband should run across the middle of the forehead, not too high (near your hairline) or too low (near your eyebrows).

Ensure there is no hair between the sensors and your skin, as this can prevent the electrodes from getting a good signal. Move any hair from above or behind your ears as you adjust the fit for the best results. If you have long hair, we recommend tying it up. You may need to adjust the headband a few times to obtain the best fit and most consistent signal quality. Please pause for 30-60 seconds after making any adjustments to allow the signal to settle. In time, you will find the fit that is most optimal for you, and this step will become easier over time. The band should sit snug and comfortably around your head, ensuring full contact with the electrodes.

Refer to this YouTube Video for an additional visual guide

3.1.2 Calibration

As soon as the focus and blink detectors are connected, the focus threshold is first automatically calibrated. Calibration is undertaken by first relaxing for 5 seconds, then concentrating hard for 10 seconds. The initial threshold is then set as just below the maximum focus level reached during the calibration period. The threshold can be recalibrated at any time by pressing the “Calibrate Focus Threshold” button. During the calibration period, all movement commands sent to the car are suspended until calibration is complete. Calibration status can be seen to the right of Battery Level.

3.1.3 Control

To drive the car forward, a user should concentrate on *something*. Some effective examples we have tested include:

a) focusing vision only on the robotic car
b) repeatedly thinking about a word
c) imagining a single harmonic tone

Beta Power levels should appear to rise over time until they exceed the threshold (displayed as a red line), at which point the car will move forward. To stop the car, simply break concentration.

Turns are performed in-place at 30° intervals, independently of driving forward. To turn, the user should blink the eye corresponding to the direction of rotation intended. Note that in order to minimize false positive turn detections, the system is optimized for coarse eye blinks, e.g. the user is expected to *forcefully* blink one eye. Intentional blinks generate stronger electrical signals from the ocular muscles; therefore, our system is designed to filter out weak or involuntary blinks to reduce false positives.

3.1.4 Noise & Troubleshooting

A successful setup process provides the means to operate the RC car remotely within a reasonable timeframe using user concentration level to move forward and right/left blinks to turn in place. Noise may significantly impact performance over time as a result of the system's utilization of previously recorded voltage values in updating the threshold — the "Reset" button should be utilized for extended sessions, as well as whenever the headset is removed from a user.

If you're having difficulty getting a good signal quality from the EEG headset there are a couple of different ways to remedy this:

a) Apply a damp tissue or cloth (moistened with water) to the Muse sensors. Give them a gentle wipe to assist conduit of the sensors with your skin.
b) Or use the damp cloth or tissue and run across your forehead and behind the ears to improve sensor connection. You can also take a damp cloth to your forehead and behind your ears to improve signal quality.

If there are errors with the connection or the application we suggest closing and restarting the application.

3.2 Operating Mode 2: Abnormal Operations

The Neurotoys system is susceptible to interference and performance-reducing factors of several kinds including but not limited to noise, system latency, distractions to users, and operation distance. While we have optimized performance for typical usage, misclassifications should be expected due to our use of a cost-effective EEG. As mentioned previously, the “Reset” button should be utilized in the case of faulty readings or any related issues.

Concerning safety, always assume the car may move forward, turn, or stop at any point regardless of user input. Power off the car before removing the headset to prevent noise in the environment from impacting the car’s motion.

3.3 Safety Issues

NeuroToys employs the use of an EEG headset, it should not be used by any individuals:

a) with implanted medical devices
b) with uncontrolled epilepsy or seizure disorders
c) with skin conditions or wounds on or near the forehead

Two 3.4V lithium-ion batteries are utilized along with an inline 3A fuse to power the car safely. Keep away from heat, sparks, and flame. Do not operate with wet hands.

4 Technical Background

The decision to use EEG was driven by its ability to measure brain activity associated with intention, providing an intuitive method for control. By filtering the raw voltage signals to show only the focus/alertness frequency range, we are able to ensure accurate classification of user commands.

The remote controlled device is a four-wheel-drive toy car designed to respond to commands derived from EEG signals, using an ESP32 microcontroller as its primary processing unit. The chassis is fabricated from laser-cut acrylic and supports the car's structural components and electronics. Its lightweight yet durable design includes dedicated locations for the ESP32, motor driver, and power supply components, with the microcontroller being near the front of the cart, the motor driver in the middle (equidistant from the wheels' motors), and the battery holder on the back end of the cart to allocate for spacing and weight distribution. The cart operates using four DC motors, connected via a motor driver circuit as illustrated in Figure 2.3.2.

The ESP32 receives control signals wirelessly via Bluetooth Low Energy (BLE) from the EEG headset. These commands are converted into precise motor control signals, enabling differential power delivery for synchronized motion of the required forward motion. A 7.4V lithium-ion battery powers the system, with voltage regulation provided by a buck converter to ensure consistent operation. The physical assembly includes four tires mounted, attached and aligned to ensure proper torque transfer and stability during operation. The cart's chassis is engineered to minimize friction and optimize power transfer, ensuring efficient and reliable performance. The design focuses on mechanical robustness and functionality. Its integration of precision-engineered components and thoughtful cable management makes it both reliable and maintainable.

This system enables Bluetooth connection between a computer and an ESP32 microcontroller that acts as a Bluetooth Low Energy (BLE) Generic Attribute Profile (GATT) server, enabling control of a toy car. It uses the Bleak Python library for BLE interactions and asyncio for asynchronous communication. The system first scans for our ESP32 device, identified by the name, and establishes a BLE connection. After connecting it examines the ESP32's services and characteristics to understand the device's capabilities. This exploration involves listing all available services and their characteristics to give information on how the computer can interact with the ESP32. The system can then send forward and turn commands to a specific characteristic on the ESP32 to control the toy car's movement. It also handles notifications sent by the ESP32 which can include updates on the car's status, and displays the received data in both hexadecimal and ASCII formats for debugging purposes. The system also provides a command-line interface that allows users to manually input commands to control the car's movements in real-time specifically for testing. Another method uses a TCP server that manages the connection between the computer and the ESP32. Aside from allowing for remote control of the toy car from other devices, it gives the advantage of having a persistent connection between the computer and the ESP32. The persistent connection allows us to bypass the need to reconnect to the device after each test.

The ESP32 microcontroller serves as the central control unit for the toy car, receiving commands with BLE and translating those commands into actions such as forward movement, stopping, and turning. The ESP32 supports a BLE GATT server, advertising its services and characteristics, which are defined using specific UUIDs within a GATT profile. This profile provides a structured framework for data exchange between the computer, sending the commands, and the ESP32. The server manages the BLE connections, handles incoming data, and interprets the commands to control the car's motors through GPIO pins. The specific motor control functions are implemented in code, defining the logic for activating the motors in different combinations to achieve the desired car movements.

5 Relevant Engineering Standards

The NeuroToys project integrates multiple engineering disciplines — electrical, software, and mechanical — into a unified brain-computer interface (BCI) system. Several engineering standards across these domains are relevant to ensuring the project is safe, reliable, and interoperable.

NeuroToys uses Bluetooth for wireless communication between the EEG headset and the ESP32 microcontroller. Compliance with Bluetooth Core Specification ensures that the device transmits data reliably and efficiently. FCC Part 15 standards for electromagnetic compatibility also play a role, although the product does not emit any abnormal RF signals.

Given that NeuroToys uses EEG hardware in close contact with the human body, compliance with IEC 60601-1 (General Requirements for Basic Safety and Essential Performance) is highly relevant. While NeuroToys is not a commercial nor certified medical device, adherence to key principles such as electrical isolation, current leakage limits, and protection against electrical shock is crucial for user safety. Additionally, the National Electrical Safety Code (NESC) provides guidelines for safe electrical design practices which we have followed regarding our low-voltage DC circuit.

The system is also intended to be used by children or general consumers — referencing ISO 9241 (Ergonomics of Human-System Interaction) helps ensure our project is intuitive and accessible to all.

6 Cost Breakdown

The NeuroToys project prioritizes the cost-effectiveness of our solution, as accessibility to all is a core consideration. The most costly component is our choice of EEG, which we needed to have high accuracy for functionality — however, all other components are easily manufactured and may be sourced from a multitude of vendors. This enables our project to be a viable option for widespread adoption, especially in educational environments, hobbyist communities, and assistive technology development. By intentionally designing around low-cost, easily sourced components, we ensure that our project remains accessible not just to commercial institutions, but also to individuals themselves.

Table 6.1: Project cost breakdown for production

Item	Description	Quantity	Cost
1	18650 Batteries	2	$16.98
2	18650 Battery Spring Clip	1	$6.99
3	Motor Driver	1	$6.99
4	Toy Car Tires, DC motors	4	$9.99
5	ESP32	1	$5.00
6	Muse 2 EEG Headset	1	$249.99
7	3A glass fuse	1	$0.60
8	Fuse Holder	1	$0.70
9	Acrylic Chassis	1	$8.00
10	Dupont Connectors	10	$0.50
	TOTAL COST:		$305.74

7 Appendices

7.1 Appendix A - Specifications

Table 7.1: Engineering standards

Requirements	Value, range, tolerance, units
Car Dimensions	25cm x 14cm x 14cm
Operation Time	6 hours (normal operation), 2 hours (continuous operation)
Operating distance (user to car)	Up to 30m
Accuracy	>80% classification accuracy over five 30s trials
Battery Capacity	18 Wh
Cost	$305.75
Total Response Time	~500ms

7.2 *Appendix B – Team Information*

Adam is a graduating senior in Computer Engineering with an interest in VR and embedded systems. He plans to apply his skills and knowledge gained from Boston University in the hardware industry.

Andrés is a graduating senior in Mechanical Engineering with a minor in Entrepreneurship and Innovation, and a Concentration in Technology Innovation. Along with his insatiable drive to mass knowledge, he plans to employ his skills gained throughout his years at Boston University into becoming a world-renowned inventor as well as a product designer. He, indeed, will pursue graduate education in the near future in the business and anthropology fields.

Gabriela is a graduating senior completing a dual degree in Biomedical Engineering (B.S.) and Neuroscience (B.A.), with a minor in Electrical Engineering. Founder of this project, she is fascinated by the interactions between technology and the brain; specifically by the development and advancement of BCIs through robotics and machine learning. Following graduation, Gabriela has committed to a Ph.D. at Brown University, where she plans to investigate bidirectional feedback mechanisms of the motor and sensory cortices for closed-loop BCI systems.

Mete is a graduating senior in Computer Engineering with a concentration in machine learning. After graduation, he has committed to pursuing an M.S. in Computer Science at Stanford University, intending to specialize in the field. He plans to work in robotics or other machine learning applications, and use his work in ML to develop intelligent software systems, solving real-world problems.

Robert is a graduating senior in Computer Engineering passionate about biotech, product development, and machine learning. He would like to work in a startup environment to bring lab scale research devices to viable medical devices.