

ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

**Facultad de Ingeniería en Mecánica y Ciencias de la
Producción**

Diseño e implementación del sistema de control de una mano
robótica antropomórfica mediante señales mioeléctricas

PROYECTO INTEGRADOR

Previo a la obtención del título de:

Ingenieros en Mecatrónica

Presentado por:

Bolívar Steward Núñez Montoya

Mauricio David Valarezo Añazco

GUAYAQUIL - ECUADOR

Año: 2021

DEDICATORIA

Dedico el presente trabajo a Dios, por haberme guiado en mi ciclo universitario brindándome la fuerza y el valor para superar cada obstáculo. A mis padres y hermana, por siempre estar presentes apoyándome en todo momento, brindando sus consejos y experiencias de vida. De igual forma a mi querida sobrina, por sacarme una sonrisa en los momentos más cansados y agotadores; por sorprenderme con sus locuras e inteligencia. A mis profesores, quienes me brindaron todos sus conocimientos y experiencias profesionales para mi crecimiento personal y profesional.

Bolívar Steward Núñez Montoya

DEDICATORIA

Este trabajo lo dedico en primer lugar a Dios, porque gracias a sus bendiciones y sabiduría, hoy estoy terminando otra etapa en mi vida. También lo dedico a mis padres Edwin Valarezo y Rosa Añazco, que siempre se han preocupado y apoyado por más difícil que sea la situación, gracias a su ayuda hoy estoy aquí. Mis hermanos Edwin Valarezo y Eliana Valarezo, que son los mejores profesionales que he conocido y siempre me enseñan algo nuevo e importante. Cada uno de ellos siempre estuvo presente en mi vida dedicándome su tiempo, consejos, sus enseñanzas para que pueda alcanzar mis objetivos. Por eso los quiero mucho, y muchas gracias por su paciencia.

Mauricio David Valarezo Añazco

AGRADECIMIENTO

Agradezco a Dios por toda su ayuda y haberme guiado por toda mi etapa universitaria.

Agradezco a mis padres y mi hermana por siempre apoyarme en todo momento, por haberme brindado sus consejos de vida y profesionales.

También agradezco a todos mis profesores entre ellos el Master Terán y el Doctor Fajardo, quienes me brindaron sus conocimientos en todo momento.

Al Doctor Francis Loayza, al Ingeniero Saravia y al Ingeniero Edwin Valarezo, por haberme guiado y brindado sus conocimientos y experiencias para el desarrollo del presente proyecto con éxito.

De igual forma a mi compañero Mauricio Valarezo, por ser parte de este interesante proyecto de grado.

Bolívar Steward Núñez Montoya

AGRADECIMIENTO

Agradezco a Dios por toda la ayuda, paciencia y todas las bendiciones brindadas en mi etapa académica.

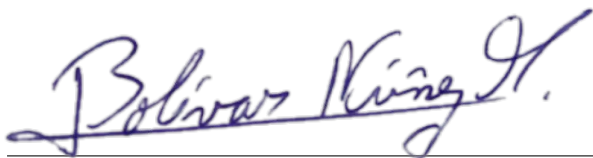
A mis padres Edwin y Rosa por toda la confianza brindada, por ser mi compañía, ser mis guías en los momentos difíciles, y siempre apoyarme en mis decisiones. A mis hermanos Edwin y Eliana por ser unos ejemplos para mí, ya que siempre me dieron sus consejos, enseñándome cosas nuevas, motivándome y ayudándome a seguir adelante.

También agradezco a Aracely Arrata por haberme acompañado en toda esta etapa académica, con sus ánimos y ayuda logre obtener muchos logros. A mi amigo Bolívar Núñez quien me brindó su apoyo en el proyecto y en cualquier error cometido.

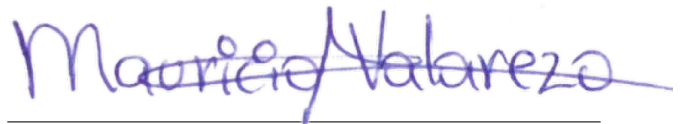
Mauricio David Valarezo Añezco

DECLARACIÓN EXPRESA

“Los derechos de titularidad y explotación, nos corresponde conforme al reglamento de propiedad intelectual de la institución; Bolívar Steward Núñez Montoya, Mauricio David Valarezo Añazco damos nuestro consentimiento para que la ESPOL realice la comunicación pública de la obra por cualquier medio con el fin de promover la consulta, difusión y uso público de la producción intelectual”



Bolívar Steward Núñez Montoya



Mauricio David Valarezo Añazco

EVALUADORES

M.Sc. Efraín Terán
PROFESOR DE LA MATERIA

Ph.D. Francis Loayza
PROFESOR TUTOR

RESUMEN

La continua evolución de conocimientos y tecnologías ha incitado a desarrollar nuevas áreas de conocimiento en la educación ecuatoriana para alcanzar un nivel de aprendizaje similar a las instituciones del exterior. Por ello, el presente proyecto plantea el diseño e implementación del sistema de control de una mano robótica antropomórfica (MRA) con 6 grados de libertad utilizando señales mioeléctricas, para promover el aprendizaje en estudiantes y docentes del laboratorio de Neuroimagen y Bioingeniería de la ESPOL. Primero se recopiló los datos EMG de 10 participantes (4 hombres y 6 mujeres) que imitaban 4 gestos de la mano. El diseño de la implementación estaba basado en una placa electrónica para la adquisición de datos Cyton Board, que se comunica por puerto serial a una PC. La PC aloja el sistema de control y el modelo de clasificación mediante machine learning que emite el movimiento mediante una Rest API (comunicación con POST y GET de HTTP) a un ESP32 que controla el movimiento de la MRA. El algoritmo *Multi-layer Perceptron Classifier* fue seleccionado con tres capas ocultas, cuya exactitud obtenida fue de 84.78 %, permitiendo identificar 7 de 10 movimientos aleatoriamente realizados entre los 4 gestos seleccionados con un tiempo de reacción de 2 segundos. El diseño final de la MRA fue impresa en 3D, conteniendo todo el sistema eléctrico en su interior. Este diseño contribuye a los objetivos 3 y 4 de las ODS por ser un dispositivo didáctico que incentiva la investigación y podría ayudar a personas con manos amputadas. El precio final fue de \$1 570.00 dólares, convirtiéndolo en un equipo biomédico 51 % más económico que otros disponibles localmente. Permitiendo al dispositivo ser más accesible para implementarse en distintos centros educativos y de investigación.

Palabras Clave: Electromiografía, Antropomorfismo, Machine Learning, Clasificadores, Mano robótica.

ABSTRACT

The continuous evolution of knowledge and technologies has prompted the development of new areas of knowledge in Ecuadorian education to achieve a level of learning similar to that of foreign institutions. For this reason, this project proposes the design and implementation of the control system of an anthropomorphic robotic hand (MRA) with 6 degrees of freedom using myoelectric signals to promote learning in students and teachers of the Neuroimaging and Bioengineering laboratory of ESPOL. EMG data was first collected from 10 participants (four men and six women) who imitated four hand gestures. The implementation design was based on an electronic board for data acquisition Cyton Board, which communicates via serial port to a PC. The PC hosts the control system and the classification model through machine learning that issues the movement through a Rest API (communication with HTTP POST and GET) to an ESP32 that controls the movement of the MRA. The Multi-layer Perceptron Classifier algorithm was selected with three hidden layers, whose accuracy was 84.78 %, allowing the identification of 7 out of 10 randomly performed movements among the four selected gestures with a reaction time of 2 seconds. The final design of the MRA was printed in 3D, containing the entire electrical system inside. This design contributes to objectives 3 and 4 of the SDGs as it is a teaching device that encourages research and could help people with amputated hands. The final price was \$1,570.00, making it 51 % cheaper than other locally available biomedical equipment. Allowing the device to be more accessible to be implemented in different educational and research centres.

Key Words: *Electromyography, Anthropomorphism, Machine Learning, Classifiers, Robotic hand.*

ÍNDICE GENERAL

RESUMEN.....	I
ABSTRACT	II
ÍNDICE GENERAL.....	III
SIMBOLOGÍA	V
ABREVIATURAS	VI
ÍNDICE DE FIGURAS	VIII
ÍNDICE DE TABLAS	X
1 INTRODUCCIÓN.....	1
1.1 Descripción del problema	3
1.2 Justificación	4
1.3 Objetivos	4
1.3.1 Objetivo general	4
1.3.2 Objetivos específicos	5
1.4 Marco teórico	5
1.4.1 Mano robótica antropomórfica	5
1.4.2 Señales mioeléctricas	6
1.4.3 Posición y cantidad de electrodos	7
1.4.4 Procesamiento de señales mioeléctricas	8
1.4.5 Filtros	9
1.4.6 Rest API	10
1.4.7 Modelos ML de clasificación	10
1.4.8 Estado del arte	11
2 METODOLOGÍA.....	13
2.1 Requerimientos de Diseño.....	14
2.2 Placa de adquisición de datos	14
2.2.1 RFDuino USB dongle	14
2.2.2 OpenBCI GUI	15
2.3 Alternativas de solución para la implementación.....	16
2.4 Sistema de adquisición de datos	20

2.4.1	Ubicación de electrodos	20
2.4.2	Selección de movimientos a realizar	22
2.4.3	Programación del sistema de adquisición de datos.....	22
2.4.4	Adquisición de datos.	24
2.5	Procesamiento de datos.....	26
2.5.1	Preprocesamiento de datos.....	27
2.5.2	Filtrado de señal.	28
2.5.3	Extracción de características de la señal	31
2.5.4	Selección de características relevantes de la señal	31
2.6	Diseño de control	33
2.6.1	Modelos de clasificación	34
2.7	Implementación	35
2.7.1	Cálculo de consumo energético	36
2.7.2	Rest API	37
3	RESULTADOS Y ANÁLISIS	38
3.1	Adquisición y procesamiento de datos.	38
3.2	Diseño de sistema de control	38
3.2.1	Selección de modelo de clasificación	38
3.2.2	Sistema de control.....	43
3.3	Implementación	45
3.3.1	Sistema eléctrico	45
3.3.2	Diseño final	46
3.3.3	Resultados finales de implementación.	48
3.4	Análisis de Costos	50
4	CONCLUSIONES Y RECOMENDACIONES.....	53
4.1	Conclusiones.....	55
4.2	Recomendaciones.....	56

BIBLIOGRAFÍA

APÉNDICES

SIMBOLOGÍA

mV	milivoltio
μV	microvoltios
Hz	Hercio
A	Amperios
F_s	Factor de seguridad
m_s	milisegundos

ABREVIATURAS

LNB	Laboratorio de Neuroimagen y Bioingeniería
EMG	Electromiografía
EEG	Encefalograma
ECG	Electrocardiograma
ACTI	Actividades de Ciencia, Tecnología e Innovación
PIB	Producto Interno Bruto
LDA	Linear Discriminant Analysis
MRA	Mano Robótica Antropomórfica.
GDL	Grados de Libertad.
CB	Cyton Board
LSL	Lab Streaming Layer
FTT	Fast Furier Transform
ML	Machine Learning
MLPC	Multi-layer Perceptron Classifier
RFC	Random Forest Classifier
CNN	Convolutional Neural Network
KNN	K-Nearest Neighbors
tanh	tangente hiperbólica
ReLU	Rectified Linear Unit
SGD	Stochastic Gradient Descent
GUI	Graphical User Interface
PCA	Principal Component Analysis

IA Inteligencia Artificial
SISO Single Input Single Output
MISO Multiple Input Single Output
ROC Receiver Operating Characteristic

ÍNDICE DE FIGURAS

Figura 1.1	Prótesis pierna 300A.C [1].	1
Figura 1.2	Gasto en actividades de ciencia, tecnología e innovación, 2012-2014 [1].	2
Figura 1.3	Publicaciones científicas por cada 100.000 habitantes, 2010 - 2019 [2].	2
Figura 1.4	Diseño de mano robótica antropomórfica [3].	6
Figura 1.5	Generación de un impulso eléctrico debido al movimiento de las fibras musculares [4].	7
Figura 1.6	Curva de número de canales vs el porcentaje de precisión de clasificación [5].	8
Figura 1.7	Disposición de los electrodos [5].	8
Figura 1.8	Diagrama de bloques del Sistema de adquisición de señales EMG [6].	9
Figura 1.9	Mano robótica mimética [7].	11
Figura 1.10	Diseño de mano antropomórfica [8].	12
Figura 2.1	Metodología de diseño e implementación.	13
Figura 2.2	Placa de biodetección Cyton Board desarrollado por OpenBCI [9].	15
Figura 2.3	Adaptador USB con Bluetooth [9].	15
Figura 2.4	Diseño de implementación de la solución A.	16
Figura 2.5	Diseño de implementación de la solución B.	17
Figura 2.6	Diseño de implementación de la solución C.	17
Figura 2.7	Diseño de implementación de la solución D.	18
Figura 2.8	Ubicación de electrodos en el CB.	20
Figura 2.9	Diagrama de ubicación y distanciamiento de los electrodos a nivel muscular del antebrazo de una persona.	21
Figura 2.10	Gestos seleccionados para implementación.	22
Figura 2.11	Diagrama de flujo de adquisición de datos. Parte I.	23
Figura 2.12	Diagrama de flujo de adquisición de datos. Parte II.	24
Figura 2.13	Muestra obtenida del movimiento 1 repetición 6 de una persona.	27
Figura 2.14	Gráfica de los datos preprocesados de la Figura 2.13.	28
Figura 2.15	Espectro de frecuencias de repetición 6 del gesto 1 de una muestra.	29
Figura 2.16	Señal filtrada del movimiento 1 repetición 6 de una muestra.	31
Figura 2.17	Relevancia de las 20 mejores características.	32

Figura 2.18 Matriz de correlación.	33
Figura 2.19 Diseño base de MRA.	35
Figura 2.20 Diagrama de conexión de ESP32 y PCA9685 [10].	36
Figura 2.21 Fuente de alimentación de protoboards [11].	37
Figura 3.1 Curvas de entrenamiento y validación, escalabilidad y rendimiento del modelo seleccionado.	41
Figura 3.2 Matriz de confusión.	42
Figura 3.3 Curva ROC.	43
Figura 3.4 Diagrama de bloques del sistema de control del clasificador.	44
Figura 3.5 Diagrama de sistema de control de posición de un servomotor [12].	44
Figura 3.6 Diagrama de bloques del sistema de control de los servomotores.	45
Figura 3.7 Diagrama de conexión del sistema eléctrico.	46
Figura 3.8 Diseño final de la MRA.	47
Figura 3.9 Resultados de predicción de implementación del diseño. Parte 1.	48
Figura 3.10 Resultados de predicción de implementación del diseño. Parte 2.	49
Figura 3.11 Diagrama de Flujo de funcionamiento del diseño de implementación. ...	50

ÍNDICE DE TABLAS

Tabla 2.1	Requerimientos para el diseño e implementación del sistema.	14
Tabla 2.2	Requerimientos para el diseño e implementación del sistema.	19
Tabla 2.3	Matriz de decisión de alternativas de solución.	19
Tabla 2.4	Medidas de ubicación de cada zona del antebrazo.	21
Tabla 2.5	Datos de sujetos de prueba parte 1.	25
Tabla 2.6	Datos de sujetos de prueba parte 2.	26
Tabla 3.1	Resultado de entrenamiento y validación de modelos de clasificación	39
Tabla 3.2	Parámetros de evaluación cruzada.	39
Tabla 3.3	Resultados de evaluación cruzada con modelo seleccionado.	40
Tabla 3.4	Reporte del modelo MLPC seleccionado.	42
Tabla 3.5	Cotización de rubros.	51
Tabla 3.6	Comparativo entre prótesis de mano.	52

CAPÍTULO 1

1. INTRODUCCIÓN

El desarrollo de sistemas mecánicos con formas y comportamientos similares a los movimientos de los seres vivos remonta de varias épocas atrás. Un claro ejemplo fue la construcción de la primera pierna artificial realizada con materiales de madera y bronce en el año 300 a.C., según registros históricos [13].



Figura 1.1 Prótesis pierna 300A.C [1].

Estos sistemas también fueron abordados por Leonardo Da Vinci, una persona destacada por su ingenio e invenciones en el siglo XV. Jack Steele en 1959-1960 acuñó el término de biónica como la "Materialización de los funcionamientos reales de los sistemas vivos en aparatos" [14]. Posteriormente surge el término prótesis, como aquel dispositivo cuya finalidad es mejorar o reemplazar una parte, sección o miembro completo del cuerpo humano que haya sido afectado o amputado permitiendo recuperar su movilidad y aspecto [15]. En la última década, se ha desarrollado nuevas y mejores interfaces hombre-máquina mediante la incorporación del mundo digital en la Industria 4.0; entre los cuales se tienen los sistemas ciberfísicos [16].

La continua evolución de conocimientos y tecnologías ha repercutido en la educación ecuatoriana incitándolas a incorporar nuevas áreas de conocimiento para alcanzar un nivel de aprendizaje similar a las instituciones del exterior [17]; y así, poder satisfacer las demandas del sector productivo. En el año 2012, el

gobierno ecuatoriano destinó el 0.28 % del PIB en I+D y un total de 0.98 % del PIB como gasto de ACTI y va en aumento, como se puede apreciar en la Figura 1.2 [18].

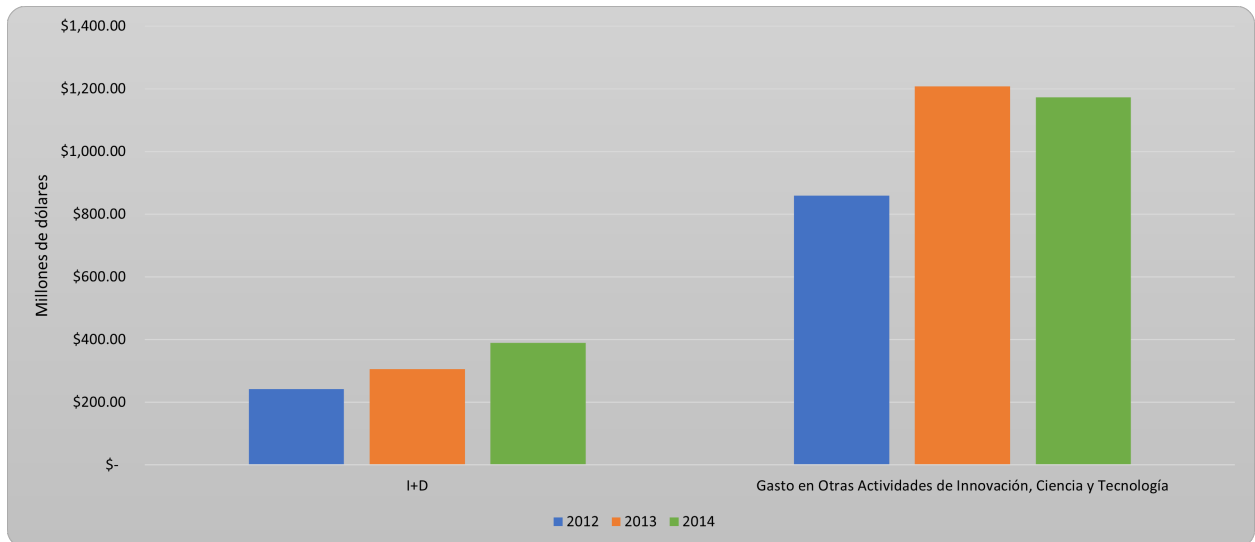


Figura 1.2 Gasto en actividades de ciencia, tecnología e innovación, 2012-2014 [1].

Este aporte ha permitido mantener un rango de 30 publicaciones científicas en Scopus por año. Sin embargo, en la Figura 1.3 se puede apreciar que Chile tiene un promedio de 85 publicaciones con una población similar a Ecuador. Por lo que, el cómo se usan los recursos tecnológicos y del personal marcan la diferencia [18].

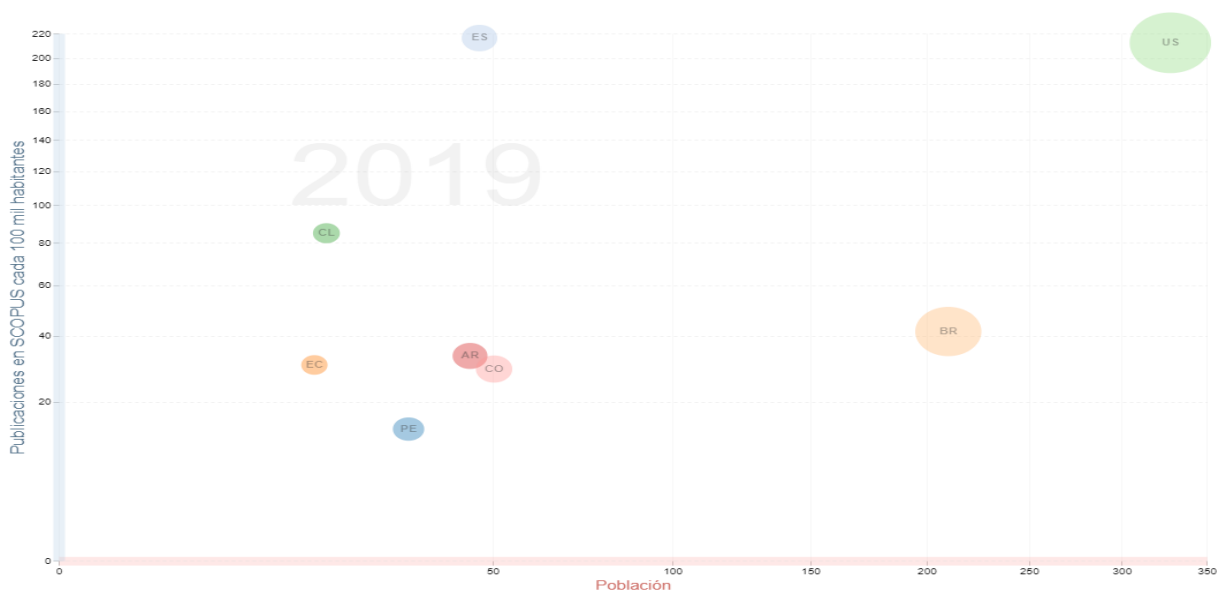


Figura 1.3 Publicaciones científicas por cada 100.000 habitantes, 2010 - 2019 [2].

Por lo tanto, el desarrollo de un recurso didáctico para el Laboratorio de Neuroimagen y Bioingeniería (LNB) de la ESPOL permitirá expandir las bases de conocimiento impartidos a los estudiantes y elevar la calidad de enseñanza de la institución educativa. El presente proyecto está seccionado en 4 capítulos empezando con la descripción del marco teórico y objetivos; el capítulo 2 detalla la metodología implementada para clasificar las señales EMG según el movimiento muscular; el capítulo 3 presenta los resultados obtenidos de la implementación, y el capítulo 4 finaliza el documento con las conclusiones y recomendaciones.

1.1 Descripción del problema

En el transcurso del aprendizaje de una ingeniería se adquieren todo tipo de habilidades y conocimientos científicos que, en cada generación de estudiantes se amplían cada vez más. El desarrollo de fuentes de conocimiento, que permitan ser base para las generaciones predecesoras, ha permitido alcanzar una mayor capacidad de conocimiento en los estudiantes.

Actualmente, en el LNB de la ESPOL no cuenta con dispositivos que permitan la interacción hombre-máquina, los cuales estén basados en el registro de señales de electromiografía (EMG) y cuyo control sea realizado aplicando Machine Learning (ML). Por ello, se requiere del diseño e implementación de un sistema de control de un dispositivo mecatrónico didáctico para el desarrollo de conocimiento en los estudiantes y profesores. El dispositivo deberá tener como base el diseño 3D de una mano robótica antropomórfica (MRA), otorgado por el PhD. Francys Loayza, la cual deberá moverse mediante el análisis e interpretación de las señales mioeléctricas. El dispositivo mecatrónico representa la mano de la extremidad superior derecha, el cual presenta seis grados de libertad que le permitirán reproducir los movimientos naturales de la mano humana. Este dispositivo tiene como finalidad ampliar los recursos utilizados en las clases de laboratorio e investigación de los estudiantes de la ESPOL, la cual podrá ser utilizada para probar e implementar algoritmos de control utilizando inteligencia artificial (IA), y mejorar la interfaz hombre-máquina.

1.2 Justificación

Según los datos estadísticos de la RICYT, la aportación científica de Ecuador es de aproximadamente 30 publicaciones hasta el año 2019. Dicho aporte es similar a países como Argentina y Colombia, pero se encuentra por debajo de Chile y de grandes potencias como Estados Unidos y España, como se aprecia en la Figura 1.3. Mediante el aporte al LNB de la ESPOL con el dispositivo biomecatrónico didáctico (i.e., una mano robótica antropomórfica) se dará un paso adelante en el desarrollo de conocimientos científicos y tecnológicos tanto para estudiantes como docentes. De esta forma, también se incentiva a la comunidad educativa a generar aportaciones científicas basadas en señales bioeléctricas.

El desarrollo de la MRA permitirá a futuro ampliar la malla curricular de los estudiantes de la carrera de Ingeniería en Mecatrónica de la ESPOL, los cuales podrán aplicar nuevos conocimientos de biomecatrónica al implementar distintos sistemas de control basados en IA. Para ello, la MRA constará con el respectivo circuito eléctrico para realizar un control independiente de los movimientos de la mano utilizando sensores mioeléctricos.

De esta forma, no solo se incentiva a la investigación en la comunidad educativa de la ESPOL sino también establecer a futuro el desarrollo de prótesis robóticas a personas necesitadas que presenten amputación de su mano.

1.3 Objetivos

1.3.1 Objetivo general

Diseñar e implementar el sistema de control de una mano robótica antropomórfica con seis grados de libertad (GDL) mediante señales mioeléctricas, para el desarrollo de una herramienta didáctica que permita promover el aprendizaje en estudiantes y docentes del laboratorio de Bioingeniería de la ESPOL.

1.3.2 Objetivos específicos

- Ensamblar un prototipo de la MRA utilizando piezas impresas en 3D.
- Analizar las señales mioeléctricas de los sujetos para el desarrollo de un correcto procesamiento de los datos.
- Diseñar un sistema de control que permita la clasificación de las señales mioeléctricas en 4 gestos de la mano y que emita el control sobre el movimiento de la MRA.
- Evaluar el desempeño de la implementación del sistema de control en la MRA.

1.4 Marco teórico

1.4.1 Mano robótica antropomórfica

La mano es una extremidad con una gran habilidad para manipular objetos y ejecutar varias actividades. Su ausencia reduciría considerablemente la capacidad de agarrar y maniobrar objetos [19]. La complejidad de su movimiento se debe a la cantidad de GDL que posee en un espacio reducido; de ello, también se evidencia el reducido tamaño de sus huesos [20]. En las últimas dos décadas ha aumentado el interés por el desarrollo de manos robóticas con la meta de desarrollar y diseñar sistemas electromecánicos capaces de reproducir la mayoría de las funcionalidades y estética de la mano [21].

Al desarrollar una mano robótica se considera dos puntos importantes: las propiedades cinemáticas y las apariencias estéticas. Esta última define el término "Antropomorfismo" como la capacidad de un robot de imitar parcial o totalmente la forma, consistencia, tamaño, color, entre otros aspectos físicos y visuales de un miembro humano [22] [23], como se puede apreciar en la Figura 1.10.

La teleoperación es uno de los puntos débiles del antropomorfismo, la cual permite realizar distintas tareas con un reducido entrenamiento [24]. Así



Figura 1.4 Diseño de mano robótica antropomórfica [3].

como la versatilidad para manipular objetos mediante movimientos simples y rápidos [25], que se logran mediante el análisis cinemático de fuerzas y torques según los actuadores [26].

1.4.2 Señales mioeléctricas

Las señales EMG son señales bioeléctricas producidas por las fibras musculares o nervios durante el movimiento de una extremidad. Continuamente se ha buscado mejores algoritmos de diferenciación y clasificación de los distintos movimientos de las extremidades para ser replicados en sus semejantes robotizados [27]. Estas señales son medidas por dos métodos: el primero es por inserción de una aguja hipodérmica, pero se considera invasivo e impropio para el cuerpo humano (Figura 1.5); el segundo método son electrodos superficiales. Se diferencian en la cantidad de grupos musculares que logran medir, siendo la intramuscular aquella capaz de seccionar las fibras musculares [4].

La señal medida por los electrodos superficiales es producida por la adición espacial y temporal de las unidades motoras individuales en conjunto con las fibras musculares que se encuentran en la vecindad del electrodo [28]. Su amplitud va de 0 a 10 [mV] con una frecuencia útil en el rango de 0-500 [Hz], siendo el rango entre 50 y los 150 [Hz] la frecuencia con mayor de energía concentrada. [29]

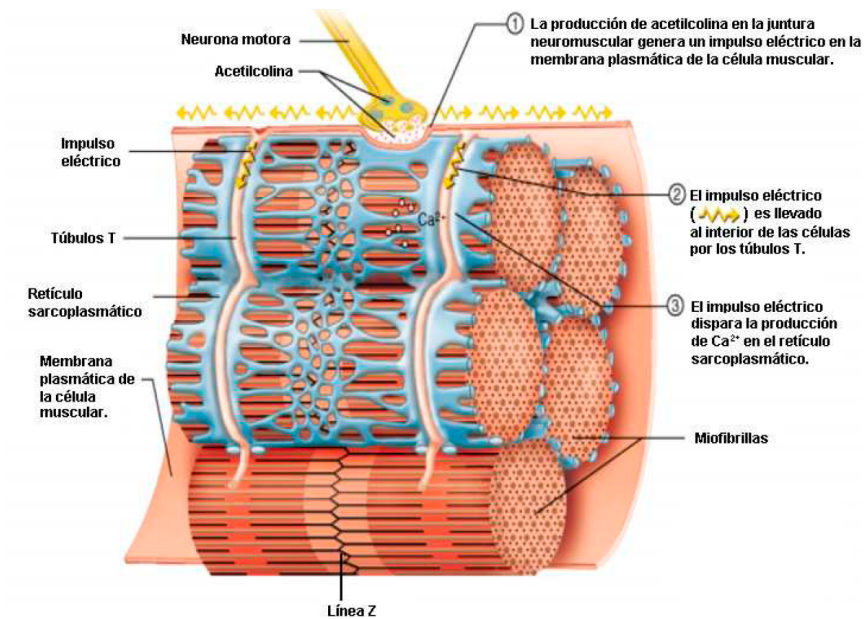


Figura 1.5 Generación de un impulso eléctrico debido al movimiento de las fibras musculares [4].

1.4.3 Posición y cantidad de electrodos

La posición y la cantidad de electrodos son aspectos a considerar durante la recolección de datos EMG. La asistencia de un personal especializado o conocimientos en anatomía muscular pueden ayudar a obtener señales más precisas acorde al movimiento de una extremidad. De acuerdo a los resultados de [5], a partir del cuarto canal no se aprecia mejoras en la exactitud de un modelo de clasificación. Al contrario, este se mantiene o empieza a decrecer como se aprecia en la Figura 1.6.

En el análisis de [30] se demuestra que la mejor disposición de los electrodos es colocarlos de forma longitudinal o mediante una combinación con una disposición transversal y longitudinal respecto a la dirección de las fibras musculares, véase Figura 1.7. Y un electrodo de gran tamaño reduce la sensibilidad del mismo.

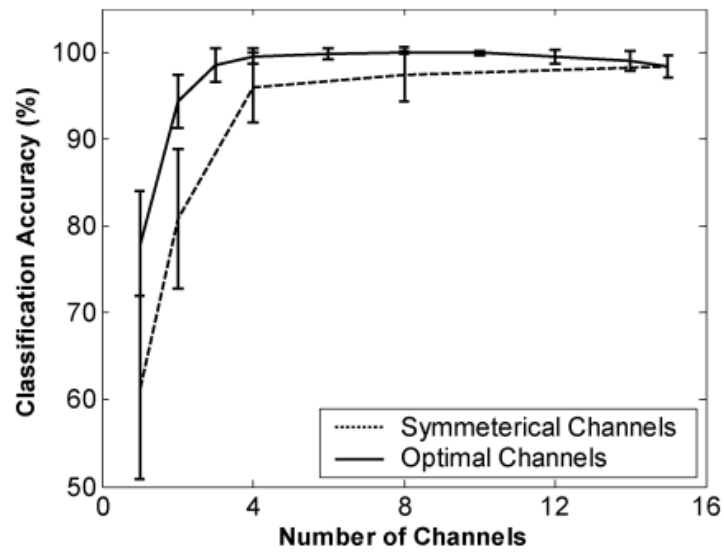


Figura 1.6 Curva de número de canales vs el porcentaje de precisión de clasificación [5].

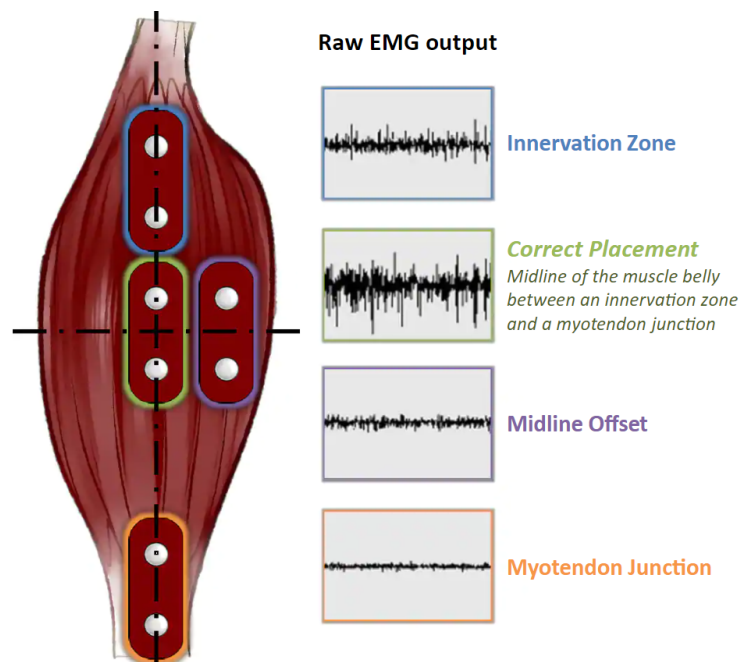


Figura 1.7 Disposición de los electrodos [5].

1.4.4 Procesamiento de señales mioeléctricas

El procesamiento consiste en la amplificación y filtrado de la señal (Figura 1.8), los cuales dependerán del análisis según el propósito final. Los amplificadores diferenciales alcanzan magnitudes de 5000 veces y permiten evitar distorsiones presentes en las señales EMG [31].

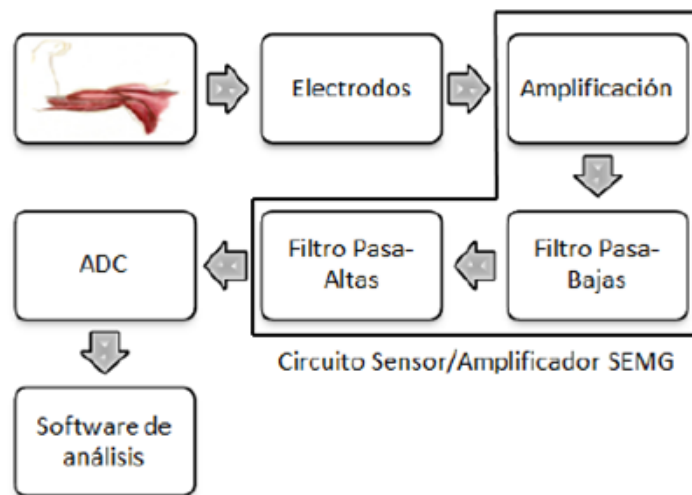


Figura 1.8 Diagrama de bloques del Sistema de adquisición de señales EMG [6].

El cuerpo humano es un buen agente de recepción de ondas electromagnéticas, por lo que fácilmente absorbe ruido ambiental [27]. Estos son atenuados o eliminados mediante la aplicación de filtros, pero su uso podría conllevar a eliminar información útil entre los 50-150 Hz [32]. Entre los ruidos presentes se tiene: ruidos inherentes provenientes de equipos electrónicos (0 a miles de hertz), ruidos comprendidos por la red eléctrica (50 o 60 [Hz]) y finalmente el movimiento en la interfaces electrodo-piel (0-10 [Hz]) [33].

1.4.5 Filtros

Entre los filtros mas utilizados en señales EMG se tiene:

- Filtro notch:** Utilizado para eliminar el ruido generado por la red eléctrica, cuyas frecuencias están en 50 o 60 [Hz]. Su forma mas utilizada pertenece al segundo orden conformado por dos polos y dos ceros, véase Ecuación 1.1 [34].

$$T(s) = \frac{s^2 + \omega_N^2}{s^2 + 2\xi\omega_N s + \omega_N^2} \quad (1.1)$$

- Filtro Butterworth:** Se lo denomina máximamente plano debido a que, dependiendo de su orden, presenta una caída bien nítida donde no se

aprecian los picos en el diagrama de Bode. Su fórmula depende de la paridad del orden, siendo la Ecuación 1.2 para orden "M" impar y la Ecuación 1.3 para orden par en filtros pasa baja [34]. Para filtros pasa alta se deben agregar ceros en en el origen según el orden.

$$T(s) = \left(\frac{\omega_N}{s + \omega_N} \right) \prod_1^{(M-1)/2} \left(\frac{\omega_N^2}{s^2 + 2 \cos(\theta_i) \omega_N s + \omega_N^2} \right), \theta_i = i \times 180/N \quad (1.2)$$

$$T(s) = \prod_1^{M/2} \left(\frac{\omega_N^2}{s^2 + 2 \cos(\theta_i) \omega_N s + \omega_N^2} \right), \theta_i = (i - 0.5) \times 180/N \quad (1.3)$$

1.4.6 Rest API

Rest Api es un conjunto de reglas que utilizan el protocolo HTTP para transmitir datos de una forma mucho mas rápida y sencilla entre cliente y servidor [35]. En [36], se utilizó comunicación websocket con un ESP32 desde un servidor remoto y uno extranjero. Este ultimo es un sistema mas completo que funciona mediante sockets.

1.4.7 Modelos ML de clasificación

La IA es una ciencia cuyo objetivo es buscar una relación con las habilidades que realizan las personas. El ML es un conjunto que depende de la IA, porque con ML se entrena y aprende una máquina [37].

Para el aprendizaje del modelo, es importante dividir los datos en dos conjuntos: un conjunto de entrenamiento con el 70% de los datos, para modelar el comportamiento; y un conjunto de evaluación con el 30% de los datos, para determinar la exactitud del modelo [37]. A través de los datos de entrada, se ejecuta un algoritmo que resulta en la generación de mas información para el problema [38]. Entre los modelos de clasificación se tiene:

- Decisión Tree.
- Random Forest.
- K Nearest Neighbor -KNN.
- Multi-layer Perceptron Clasificador.
- Gaussian Process Clasificador.
- Support Vector Machine Polinomial.
- Ridge Classifier.

1.4.8 Estado del arte

Hasta el presente año, se ha desarrollado una gran cantidad de manos robóticas implementando distintos métodos para alcanzar el antropomorfismo. Entre los cuales se destacan aquellos que utilizan señales EMG como base de análisis. Entre los destacados se tiene [7], en el cual se enfocaron en replicar la funcionalidad de los ligamentos de la mano ampliando el número de GDL's, como se aprecia en la Figura 1.9. Su modelo de clasificación basado en una red neuronal de 3 capas (656, 328, 6 neuronas respectivamente) era capaz de determinar 6 tipos de gestos de la mano con una exactitud del 84.7% mediante el uso de un brazalete MYO. El brazalete presentaba 6 canales para la lectura mioeléctrica.



Figura 1.9 Mano robótica mimética [7].

Otro ejemplo es el desarrollado en [8], cuyo diseño se puede apreciar en la Figura 1.10. Su diseño es similar al anterior, pero únicamente utiliza un solo cordón por dedo para ejercer su respectivo movimiento. Su metodología se basa en la sinergia muscular para la clasificación de varios movimientos de la mano a partir de un pequeño conjunto de movimientos aprendidos. El clasificador implementado era una red recurrente que mezcla un modelo logarítmico-linearizado gaussiano (R-LLGMN). La exactitud de su modelo logró alcanzar el 90 % entre sus pruebas.

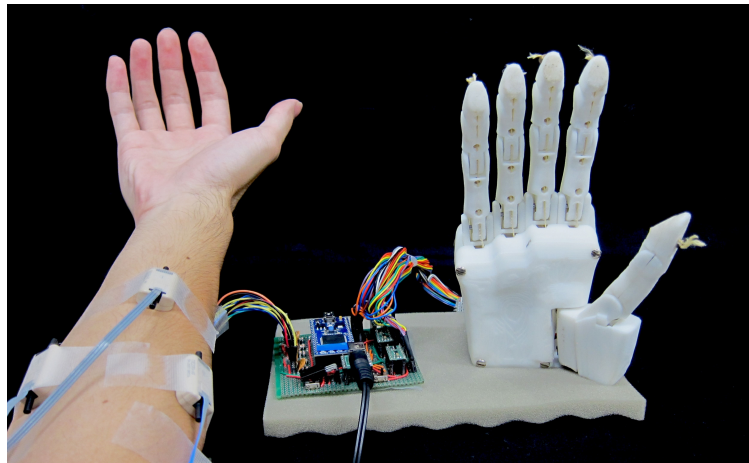


Figura 1.10 Diseño de mano antropomórfica [8].

Por otro lado, en [39] desarrollo el control de una prótesis robótica de una mano mediante el uso de redes neuronales convoluciones (CNN). En el procesamiento de las señales EMG se aplicó el análisis de los componentes principales (PCA) para la extracción de las características de mayor relevancia que ingresarían a la CNN. Sus resultados demostraron un varianza en la exactitud del modelo según el sujeto de prueba, el cual variaba entre un 60 % a un 80 % aproximadamente.

CAPÍTULO 2

2. METODOLOGÍA

En este capítulo se detalla el procedimiento para el cumplimiento del propósito del presente proyecto. Conforme los requerimientos planteados por el representante del LNB, se desarrolló cada etapa del proceso de diseño desde la selección de alternativas de solución hasta su implementación. En la Figura 2.1 se puede apreciar un diagrama de flujo de la metodología aplicada.

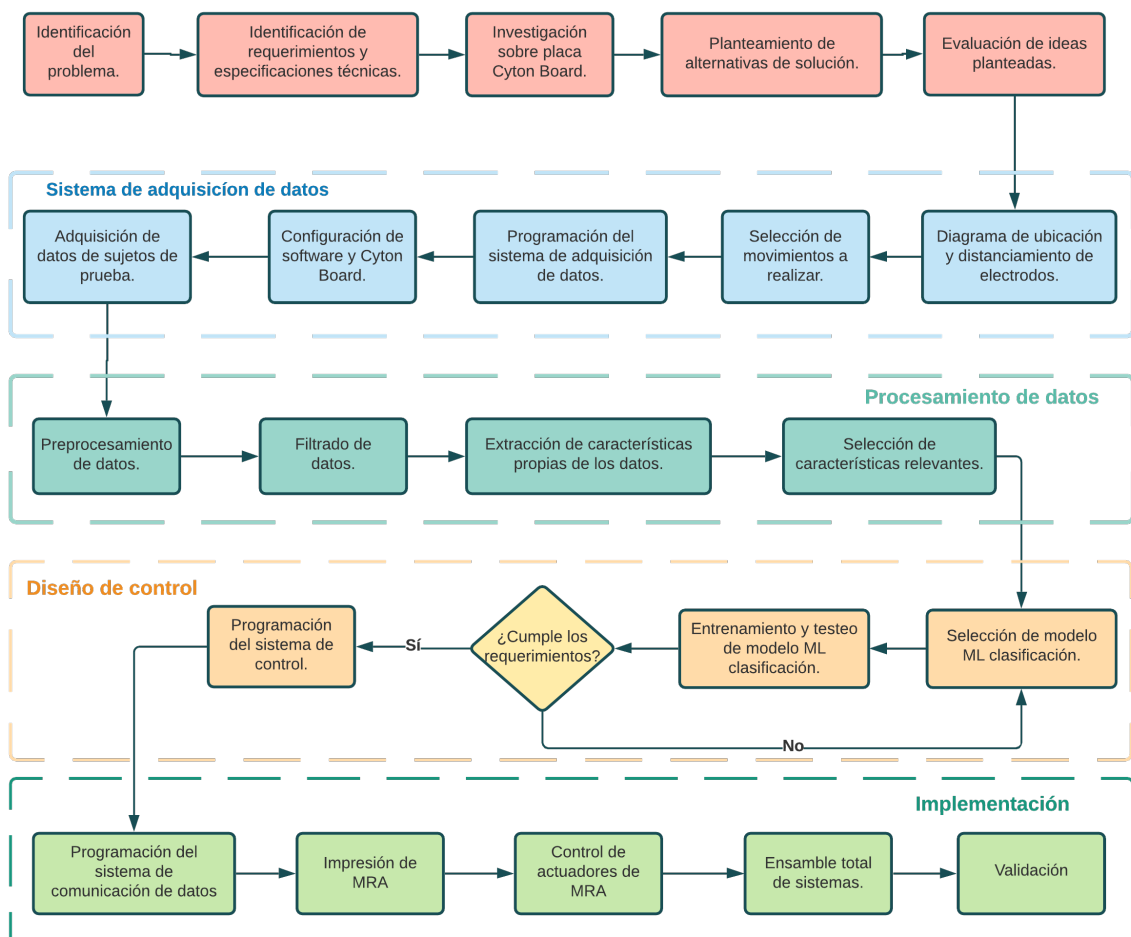


Figura 2.1 Metodología de diseño e implementación.

2.1 Requerimientos de Diseño

El diseño e implementación del sistema de control del MRA fue desarrollado bajo ciertos requerimientos del cliente. Los cuales se describen en la Tabla 2.1.

Tabla 2.1 Requerimientos para el diseño e implementación del sistema.

Criterio	Descripción
Exactitud	La exactitud del modelo del sistema de control debe ser mayor al 80 %.
Costo	La implementación del prototipo no debe superar los \$3 000.
Tiempo de reacción	El tiempo entre que se realiza el movimiento hasta que es imitado por la MRA debe ser menor a 5 segundos.
Ubicación estándar	La ubicación de los electrodos cuando se realice la medición de las señales debe ser la misma para todos los sujetos.
Operatividad	El dispositivo debe ser diseñado para trabajar durante dos horas de manera continua.
Cyton Board	Se debe utilizar la placa de adquisición de datos bioeléctricos Cyton Board.
MRA	Implementar el sistema de control sobre el MRA otorgado.

2.2 Placa de adquisición de datos

Conocidos los requerimientos, se realizó una breve investigación sobre la placa de adquisición de datos bioeléctricos Cyton Board (CB), Figura 2.2. Esta placa permite muestrear la actividad eléctrica del cerebro (EEG), la actividad eléctrica muscular (EMG) e incluso la actividad eléctrica del corazón (ECG) a través de sus 8 canales de alta ganancia con bajo ruido [9]. En el Apéndice A se puede encontrar mas información sobre las especificaciones técnicas de la placa CB.

2.2.1 RFDuino USB dongle

El RFDuino USB dongle o llave/adaptador USB (Figura 2.3) es un dispositivo que permitió establecer una conexión de forma inalámbrica

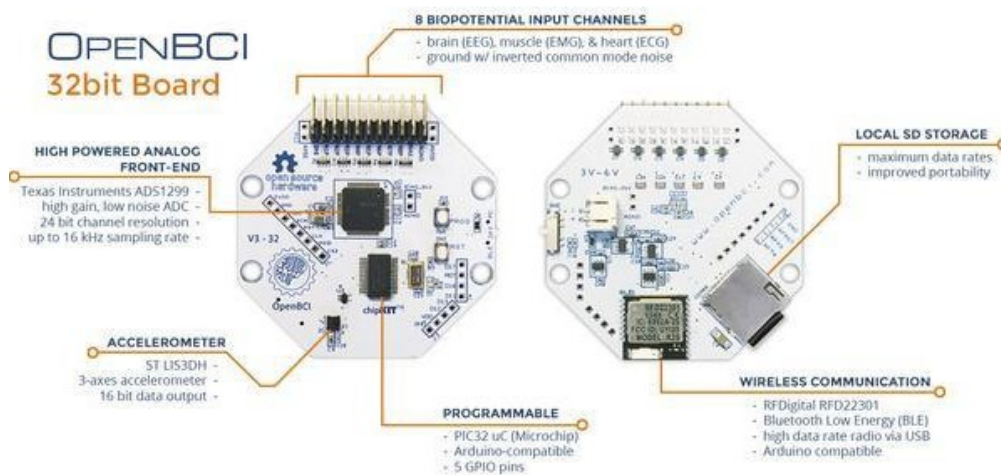


Figura 2.2 Placa de biodetección Cyton Board desarrollado por OpenBCI [9].

con un equipo de cómputo. Contiene un convertidor FTDI USB Serial que permitía recibir datos a una velocidad de 115200 baudios hasta 1 mega baudio [40] [9]. En Apéndice B se describió de forma más detalla su protocolo de comunicación.

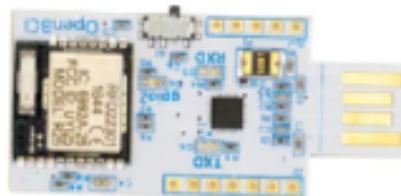


Figura 2.3 Adaptador USB con Bluetooth [9].

2.2.2 OpenBCI GUI

OpenBCI posee un potente GUI que contenía una interfaz muy amigable e iterativa para analizar los datos en tiempo real debido a sus herramientas de visualización, grabación y transmisión de datos. Para ello, se utilizó el lenguaje Python para su programación y el sistema de Lab Streaming Layer (LSL) para la transmisión de datos [41]. Una mayor descripción de la funcionalidad del sistema LSL se encuentra en el Apéndice C.

2.3 Alternativas de solución para la implementación

En relación a la problemática se planteó 4 diseños de implementación, en donde cada alternativa tenía como base la placa CB. El método de adquisición y procesamiento de datos depende del tipo de hardware y medio de comunicación. Las alternativas de solución planteadas, se presentan a continuación:

A) Sistema de control en PC utilizando comunicación serial con un Arduino o ESP32.

Su diseño (Figura 2.4) hace uso de un PC para la adquisición y procesamiento de los datos. También contiene el sistema de control con el clasificador ML, cuya salida emite el movimiento a ejecutar mediante comunicación serial hacia el hardware de control de los motores de la MRA. El hardware puede ser un Arduino o un ESP32 y el GUI es compatible con el PC.

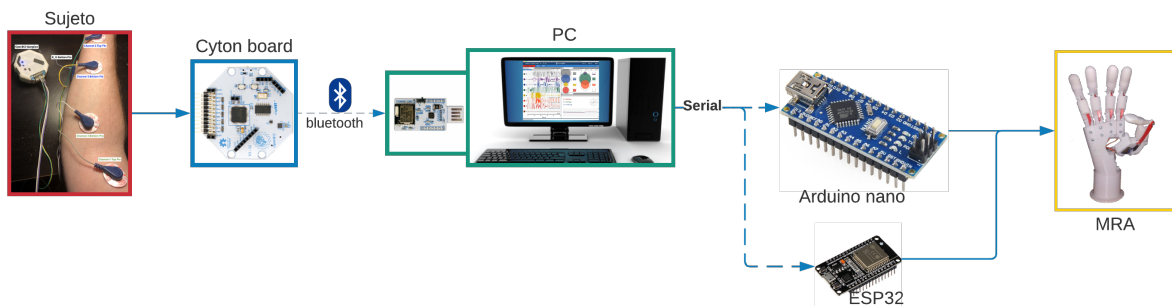


Figura 2.4 Diseño de implementación de la solución A.

B) Sistema de control en PC utilizando comunicación Rest Api con ESP32.

Esta solución (Figura 2.5) se diferencia de la anterior en el uso de la comunicación por Rest Api entre el PC y el ESP32.



Figura 2.5 Diseño de implementación de la solución B.

C) Sistema de control utilizando una Single Board Computer Raspberry Pi.

El diseño de la Figura 2.6 aloja el sistema de control, el modelo de clasificación y el control de la MRA en la Raspberry Pi. El hardware no es compatible con el GUI y la adquisición de datos no presentaba una frecuencia de muestreo estable. Presentaba una mayor complejidad al capturar los datos del movimiento de la mano del usuario.

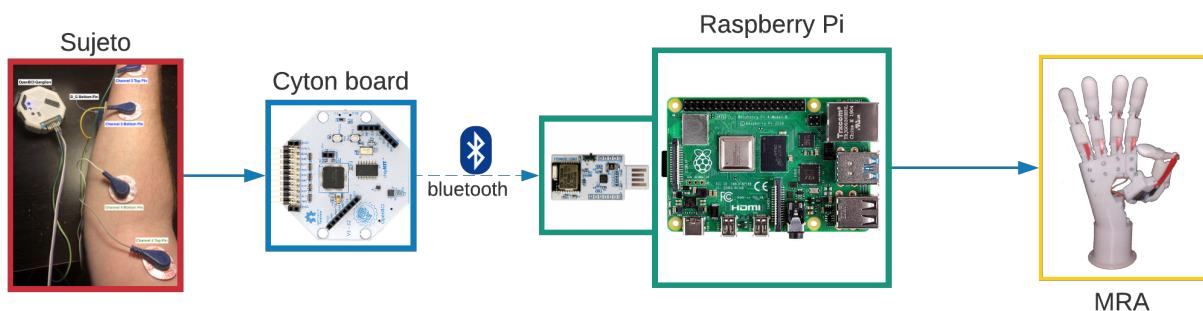


Figura 2.6 Diseño de implementación de la solución C.

D) Sistema de control utilizando una Single Board Computer Jetson Nano.

El diseño de la Figura 2.7 está basado en una Jetson Nano el cual posee una mayor capacidad de cómputo que la raspberry pero menor que un PC. Esta es una opción más costosa y con mejores resultados de cómputo en redes neuronales [42]. Para la salida de control, se requería de un módulo de mosfet para controlar los servomotores.

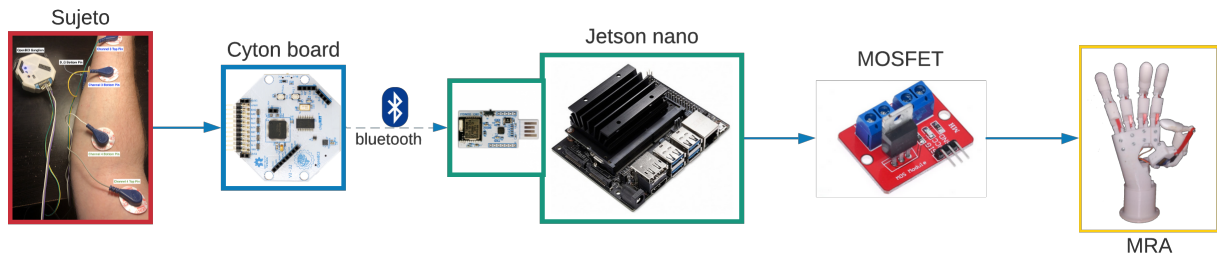


Figura 2.7 Diseño de implementación de la solución D.

La selección de la mejor alternativa de solución se determinó bajo el método ordinal corregido de criterios ponderados explicado en [43]. En la Tabla 2.2 se puede apreciar el orden de los criterios según el grado de importancia y su respectivo peso que influenciaba en la selección. Una breve descripción de los criterios que se evaluaron se muestra a continuación:

- **Transmisión de datos** La velocidad de transmisión de datos debía ser alta para obtener una baja latencia.
- **Capacidad de computo** El hardware debía ser capaz de procesar los datos rápidamente.
- **Interferencia** El diseño debía ser semi robusto reducir las afectaciones por ruido.
- **Portabilidad** El diseño del sistema debía ser pequeño y simple.
- **Tiempo de reacción** Bajo tiempo de reacción de la MRA.
- **Costo** El costo de producción debía ser bajo.

Tabla 2.2 Requerimientos para el diseño e implementación del sistema.

Criterios de evaluación			
Peso	Criterio	Rango de importancia	% de decisión
3.0	Tiempo de reacción	1	26%
2.5	Capacidad de computo	2	21%
2.0	Costo	3	17%
2.2	Transmisión de datos	4	19%
1.5	Portabilidad	5	13%
0.5	Interferencia	6	4%
11.7	Total		100%

Cada alternativa propuesta fue evaluada según los pesos de los criterios especificados. Dado los resultados obtenidos en la Tabla 2.3, se determinó que la mejor alternativa de solución fue la opción B. A pesar de su alto costo por el PC, la capacidad de computo era mayor y este suele estar presente en los laboratorios por lo que el costo final puede ser reducido.

Tabla 2.3 Matriz de decisión de alternativas de solución.

Pesos	Criterios						Resultados		
	3.0	2.5	2.0	2.2	1.5	0.5	Puntaje sin peso	Puntaje con peso	Prioridad
Alternativas	Tiempo de reacción	Capacidad de computo	Costo	Transmisión de datos	Portabilidad	Interferencia			
A	7	9	4	8	5	7	40	80.1	4
B	8	9	4	9	7	9	46	89.3	1
C	9	6	9	6	6	10	46	87.2	3
D	9	8	7	7	5	10	46	88.9	2

2.4 Sistema de adquisición de datos

Su desarrollo partió desde la especificación de la ubicación y la cantidad requerida de electrodos y finalizó con la adquisición de datos a los sujetos de prueba.

2.4.1 Ubicación de electrodos

Tomando en cuenta los resultados de la Figura 1.6, se utilizó 6 canales para la adquisición de datos por EMG. En la disposición de los electrodos, se seleccionó aquellos grupos musculares de flexión y extensión del antebrazo que incidían en el movimiento de los dedos de la mano mediante la plataforma de Complete Anatomy [44]. Los cuales fueron:

- I. Flexor superficial de los dedos.
- II. Flexor largo del pulgar.
- III. Flexor profundo de los dedos.
- IV. Extensor largo del pulgar + extensor del índice.
- V. Extensor corto del pulgar.
- VI. Extensor del meñique + extensor de los dedos.

Cada grupo muscular se encuentra ubicado en su respectiva área enumerada en la Figura 2.9. Cada par de área enumerada corresponde a un canal de la placa CB, Figura 2.8.

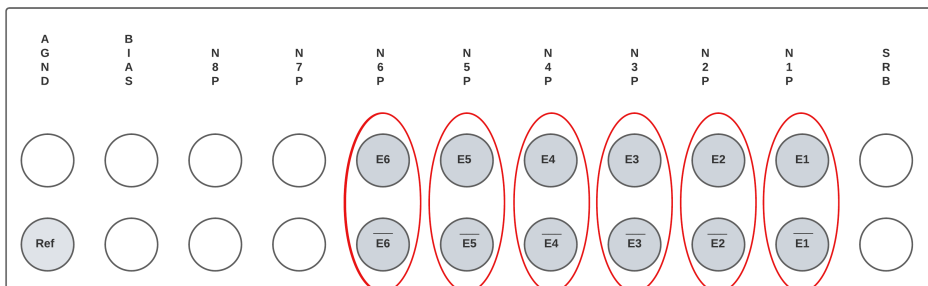


Figura 2.8 Ubicación de electrodos en el CB.

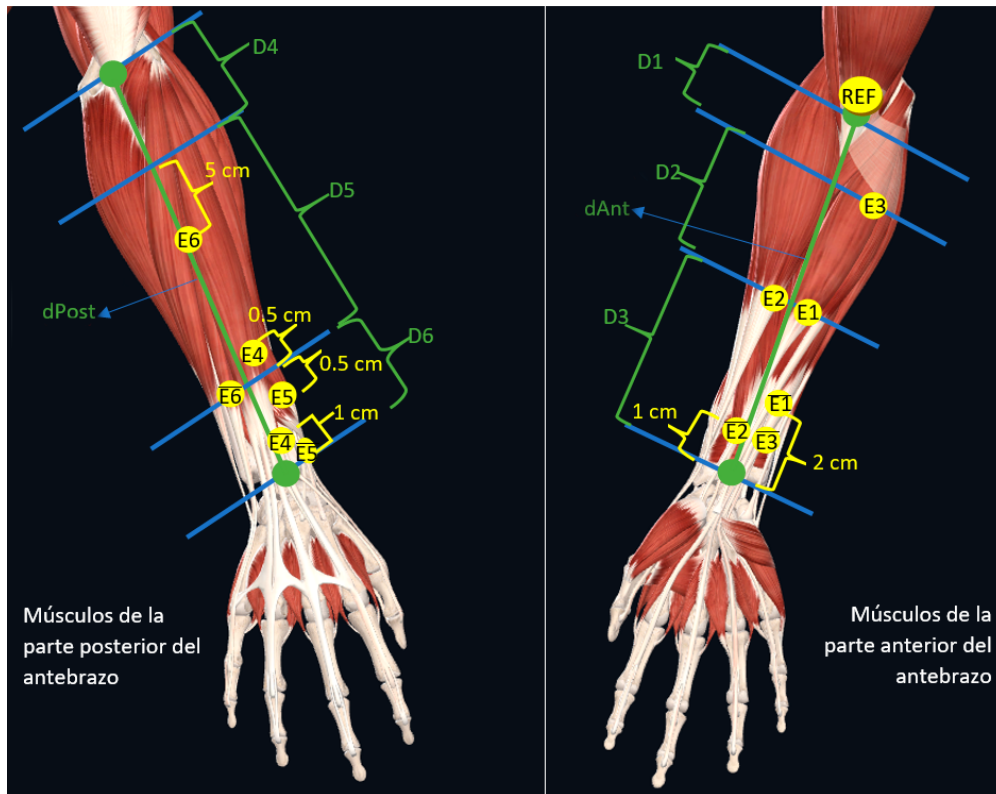


Figura 2.9 Diagrama de ubicación y distanciamiento de los electrodos a nivel muscular del antebrazo de una persona.

Para facilitar la localización de las áreas, se segmentó la parte posterior y anterior en tres zonas acorde a la Tabla 2.4. Esto permitió mantener un estándar en la colocación de cada electrodo y un orden en la lectura de las señales mioeléctricas.

Tabla 2.4 Medidas de ubicación de cada zona del antebrazo.

Variable	Medida	Descripción
DAnt		Distancia del Antebrazo parte anterior.
D1	$0.205 \cdot DAnt$	zona 1.
D2	$0.295 \cdot DAnt$	zona 2.
D3	$0.5 \cdot DAnt$	zona 3.
DPost		Distancia del Antebrazo parte posterior.
D4	$0.273 \cdot dPost$	zona 4.
D5	$0.545 \cdot dPost$	zona 5.
D6	$0.182 \cdot dPost$	zona 6.

2.4.2 Selección de movimientos a realizar

Se escogió 4 movimientos distintos a los realizados en los experimentos de [8], [45]. En la Figura 2.10(a), se puede apreciar el primer gesto de la mano el cual es utilizado de forma habitual para el pulso de botones, o manejo de pantallas táctiles. El segundo (Figura 2.10(b)) expresa el estado "OK", o la expresión "todo esta bien". El tercer (Figura 2.10(c)) es utilizado frecuentemente para manipular objetos pequeños, como una tarjeta, un lápiz o pluma. El cuarto (Figura 2.10(d)) representa la letra 'V' dentro del lenguaje de señas, y expresa conformidad.

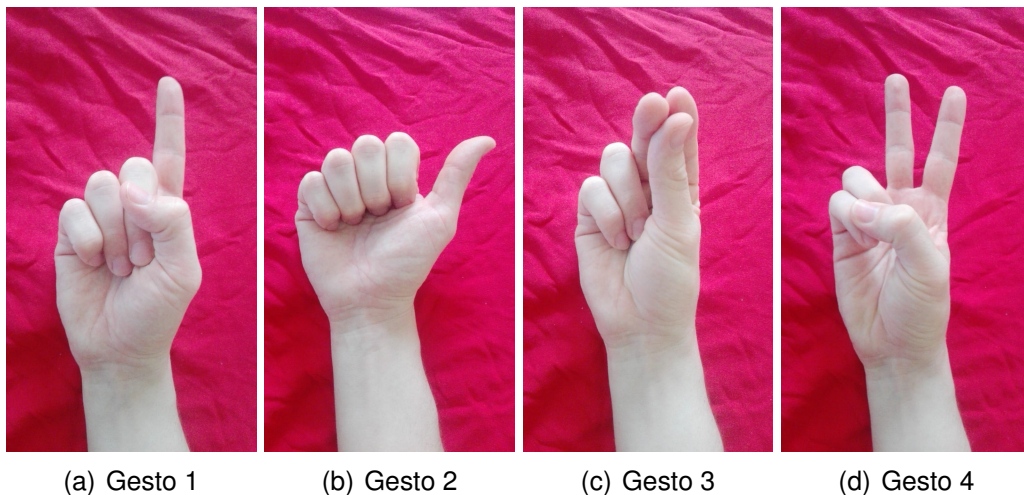


Figura 2.10 Gestos seleccionados para implementación.

2.4.3 Programación del sistema de adquisición de datos

Se desarrolló el programa en python para establecer la comunicación GUI mediante LSL, el cual se encuentra en el Apéndice E Código E.1.1. En el Apéndice C se puede encontrar más información sobre LSL, así como la respectiva configuración del GUI en el Apéndice D. En la Figura 2.11 y la Figura 2.12 se puede apreciar la secuencia de la adquisición de datos y la aleatoriedad del orden de los gestos en cada repetición. Los datos de cada gesto fueron capturados en 2 segundos y al finalizar cada sesión fueron guardados en un archivo *.csv con el siguiente encabezado: índice, time, ch1, ch2, ch3, ch4, ch5, ch6, tag ("tag" indica el tipo de gesto).

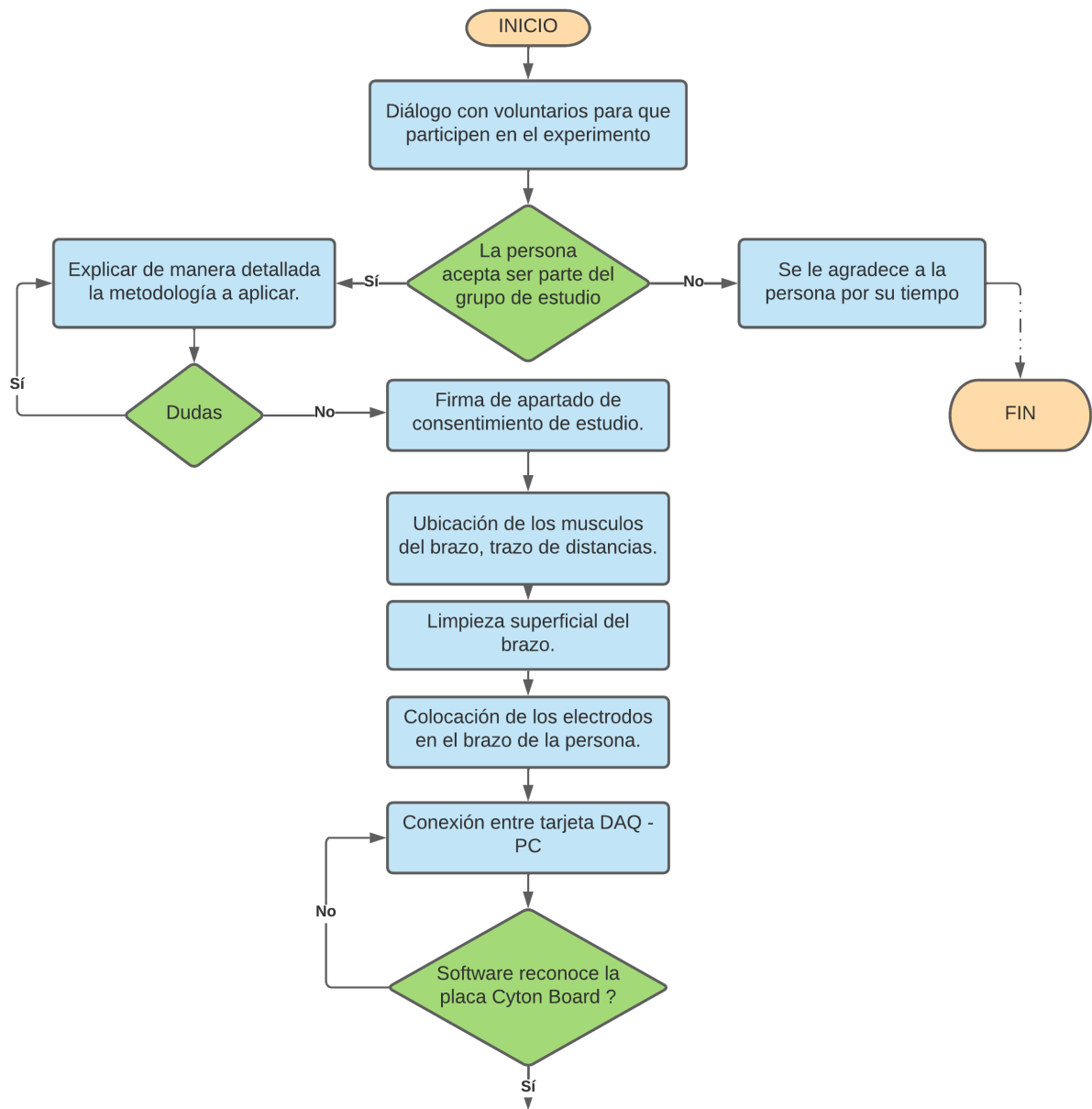


Figura 2.11 Diagrama de flujo de adquisición de datos. Parte I.

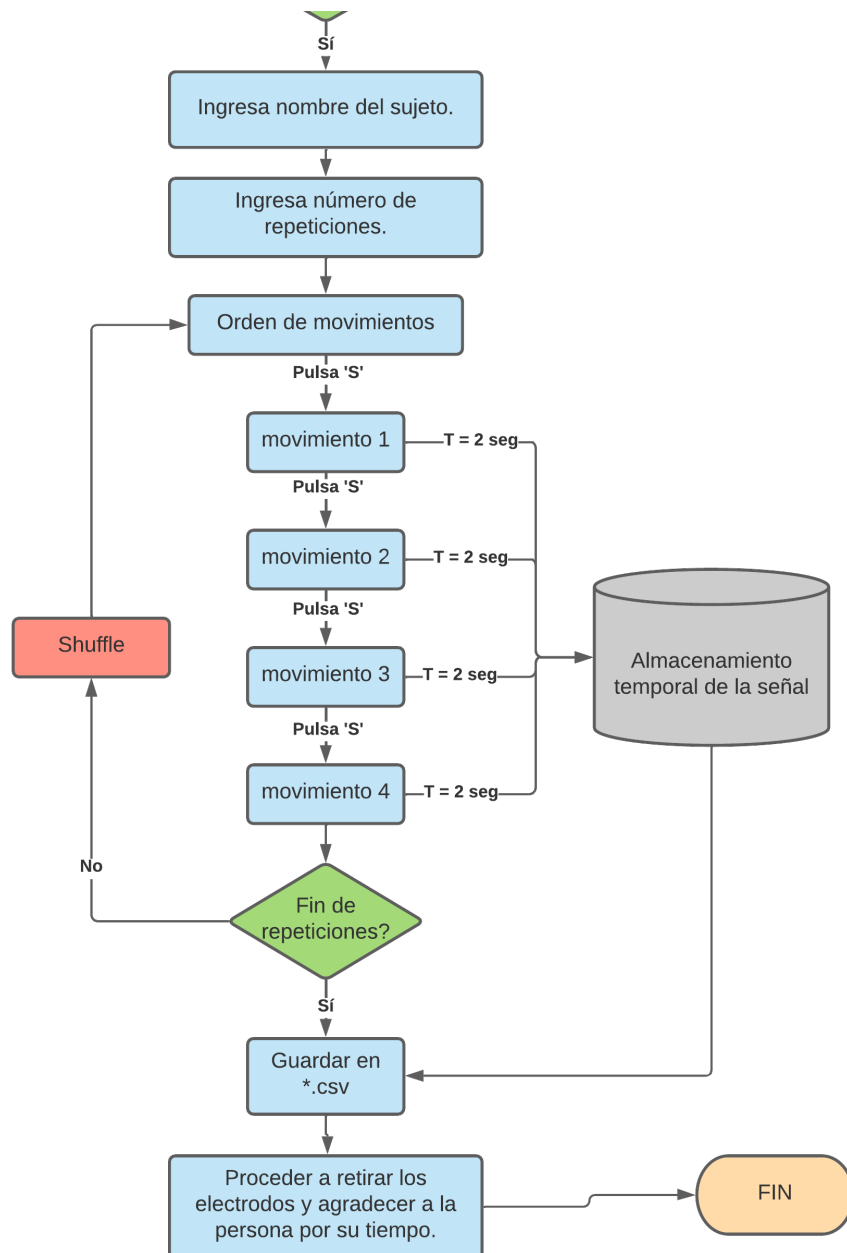


Figura 2.12 Diagrama de flujo de adquisición de datos. Parte II.

2.4.4 Adquisición de datos.

Se seleccionó personas que estuvieran en un rango entre 18 a 50 años de edad debido a que este selecto grupo presenta una mayor actividad eléctrica en los músculos. Los participantes firmaron un acuerdo de autorización de uso de datos (véase Apéndice F), y luego realizaron 20 repeticiones de cada gesto. En la Tabla 2.5 se puede apreciar aquellos sujetos que participaron en una o varias sesiones.

Tabla 2.5 Datos de sujetos de prueba parte 1.



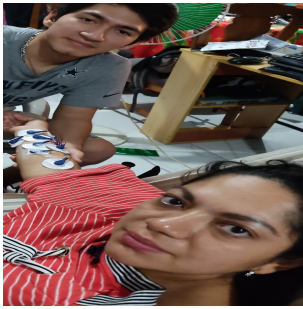
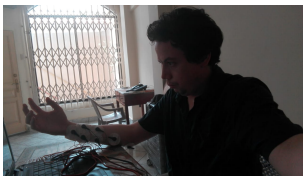






Foto	Edad	Contextura muscular	Dexteridad
	20	Media	Diestra
	18	Media	Diestra
	32	baja	Diestra
	25	baja	Diestro
	34	baja	Diestra

Tabla 2.6 Datos de sujetos de prueba parte 2.

Foto	Edad	Contextura muscular	Dexteridad
	25	Media	Diestra
	48	Media	Diestra
	43	Media	Diestro
	44	Media	Diestro
	43	Media	Diestra

2.5 Procesamiento de datos.

Los datos obtenidos de las muestras fueron preprocesados antes de filtrarlos. Cada gesto tenía un total de 500 datos debido a la frecuencia de muestreo de 250 Hz de la placa CB. En la Figura 2.13 se puede apreciar los datos crudos.

Graficas canales vs tiempo del movimiento 1 repetición 6

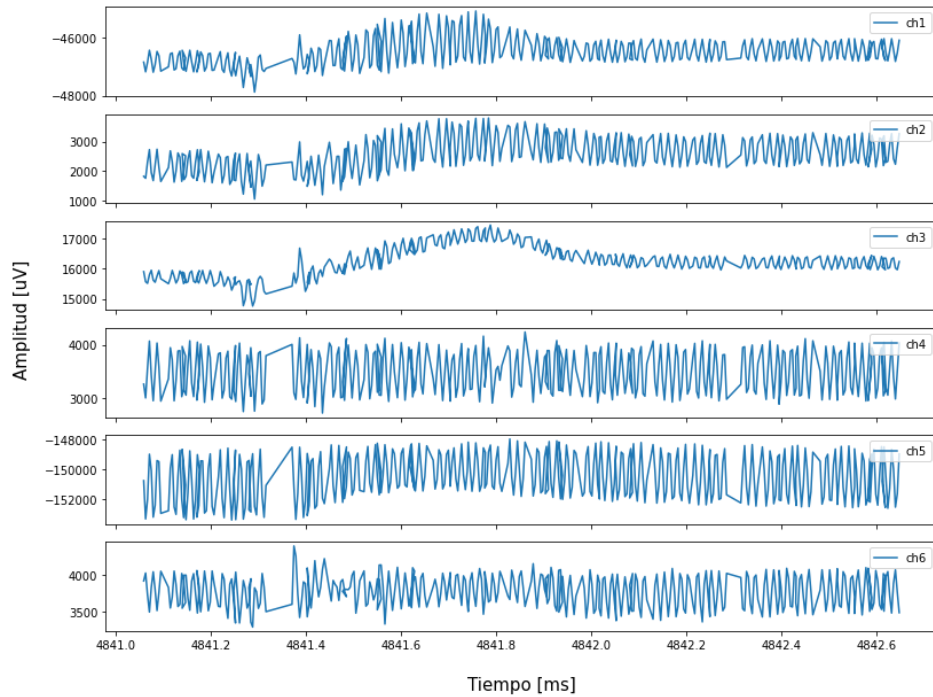


Figura 2.13 Muestra obtenida del movimiento 1 repetición 6 de una persona.

2.5.1 Preprocesamiento de datos.

El preprocesado de los datos fue aplicado a cada repetición de la muestra, cuya secuencia fue:

- A cada canal se le restó su media aritmética para centrar la señal respecto al origen. De igual forma, se precedió con el tiempo para establecer su inicio en cero milisegundos.
- Se normalizó mediante la implementación del método escalador Min-Max (ecuación 2.1), para reducir la escala de los datos originales a valores de 0 a 1 [46]. En la ecuación 2.2 se puede apreciar el cálculo para el canal 1 de la muestra de la Figura 2.13.

$$X_{sc} = \frac{X - X_{min}}{X_{max} - X_{min}} \quad (2.1)$$

$$X_{sc} = \frac{X - (-153416.984375)}{17447.056640625 - (-153416.984375)} \quad (2.2)$$

Por facilidad, se desarrolló el Código E.2.2 del Apéndice E que permitió aplicar la ecuación 2.1 para cada dato de la serie temporal y su repetición por cada canal, cuyo resultado se puede apreciar en la Figura 2.14.

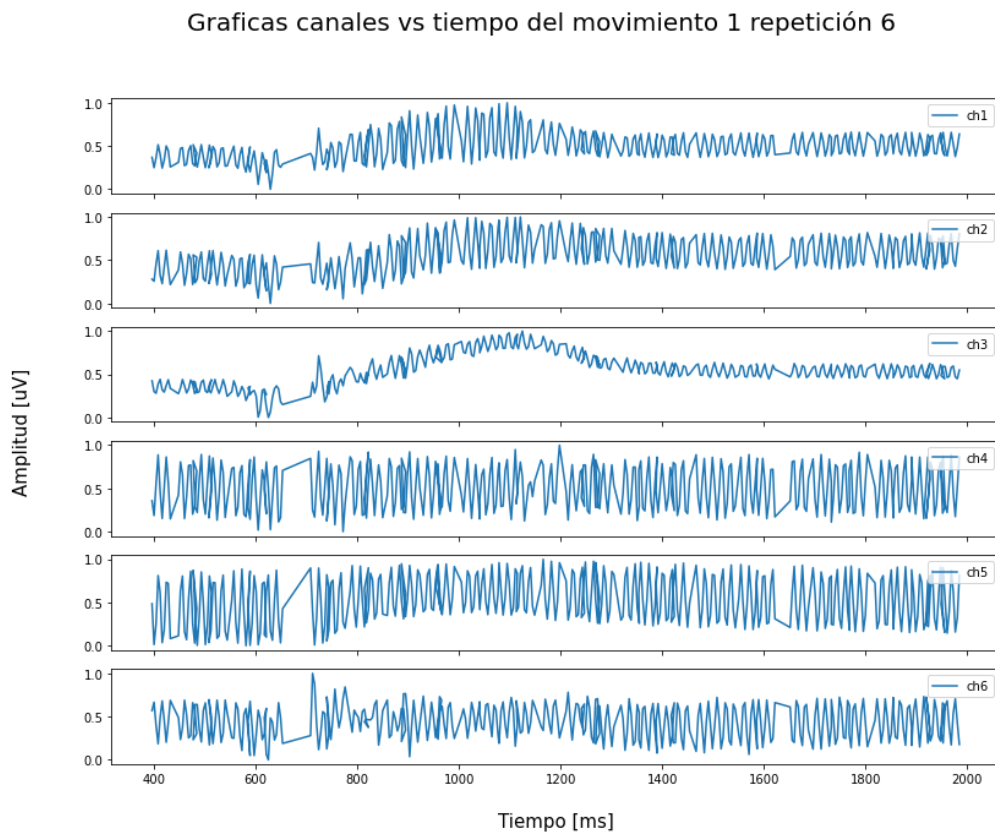
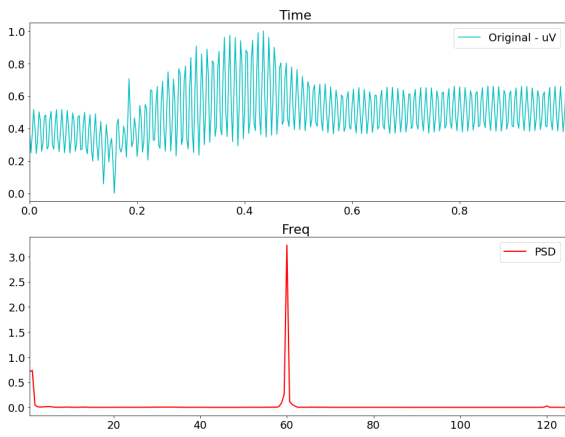


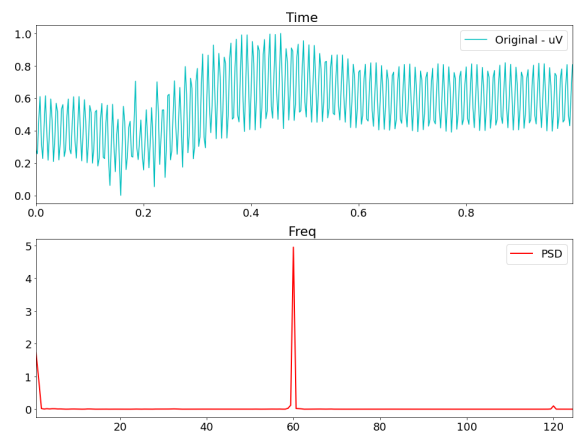
Figura 2.14 Gráfica de los datos preprocesados de la Figura 2.13.

2.5.2 Filtrado de señal.

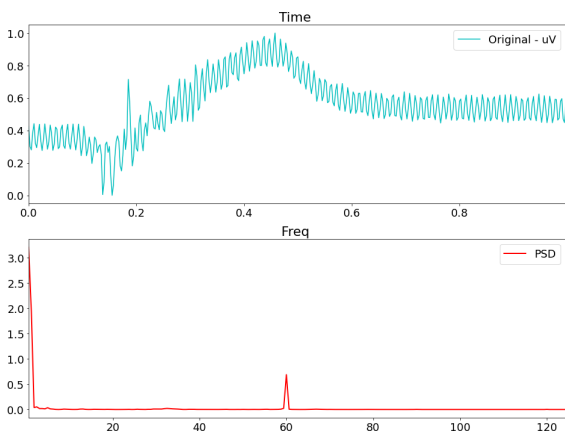
Mediante un análisis de la transformada rápida de Fourier (FTT), se logró evidenciar aquellas frecuencias atípicas de la señal que opacaban las demás frecuencias, como se aprecia en la Figura 2.15. Dichos picos se debían al ruido por la red eléctrica domestica siendo en Ecuador de 60 [Hz]. También, debido a la interacción electrodo-piel y a las señales procedentes de dispositivos electrónicos cercanos. Estas señales oscilan a una frecuencia entre 0 y 10 [Hz] [33].



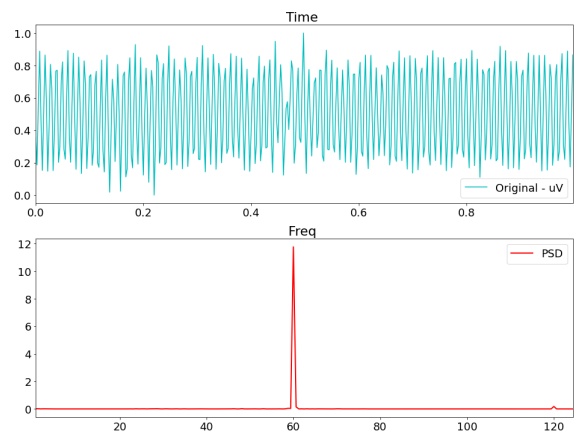
(a) Canal 1



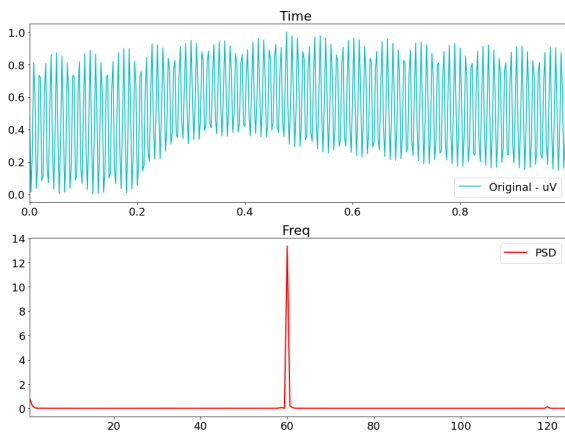
(b) Canal 2



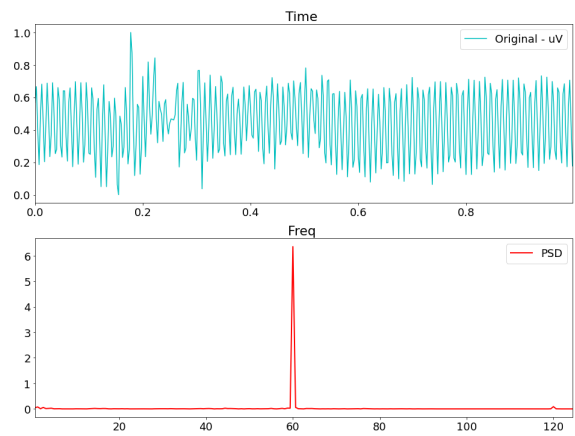
(c) Canal 3



(d) Canal 4



(e) Canal 5



(f) Canal 6

Figura 2.15 Espectro de frecuencias de repetición 6 del gesto 1 de una muestra.

Para atenuar las frecuencias atípicas se utilizó el filtro digital notch a una frecuencia de 60 [Hz] [47], ecuación 2.3. Y un filtro pasa banda de 10 - 100 [Hz] para atenuar las frecuencias fuera del espectro seleccionado. El

filtro digital aplicado fue del tipo Butterworth de orden 5, cuya expresión matemática para el filtro pasa baja se aprecia en la ecuación 2.4 y para el filtro pasa alta en la ecuación 2.5.

$$H(s) = \left(A \frac{s^2 + (2\pi 60)^2}{s^2 + s \frac{2\pi 60}{2\pi 4} + (2\pi 60)^2} \right) \quad (2.3)$$

$$T(s) = \left(\frac{2\pi 100}{s + (2\pi 100)} \right) \left(\frac{(2\pi 100)^2}{s^2 + 2 \cos(36) 2\pi 100 s + (2\pi 100)^2} \right) \quad (2.4)$$

$$\left(\frac{(2\pi 100)^2}{s^2 + 2 \cos(72) 2\pi 100 s + (2\pi 100)^2} \right)$$

$$T(s) = \left(\frac{2\pi 10 s}{s + 2\pi 10} \right) \left(\frac{(2\pi 10)^2 s^2}{s^2 + 2 \cos(36) 2\pi 10 s + (2\pi 10)^2} \right) \quad (2.5)$$

$$\left(\frac{(2\pi 10)^2 s^2}{s^2 + 2 \cos(72) 2\pi 10 s + (2\pi 10)^2} \right)$$

En el código E.2.4 del Apéndice E se puede apreciar la función de los filtros digitales. En la Figura 2.16 se puede observar el resultado de la muestra analizada ya filtrada, en la cual se aprecia con claridad la captura del gesto realizado por el sujeto.

Graficas canales vs tiempo del movimiento 1 repetición 6

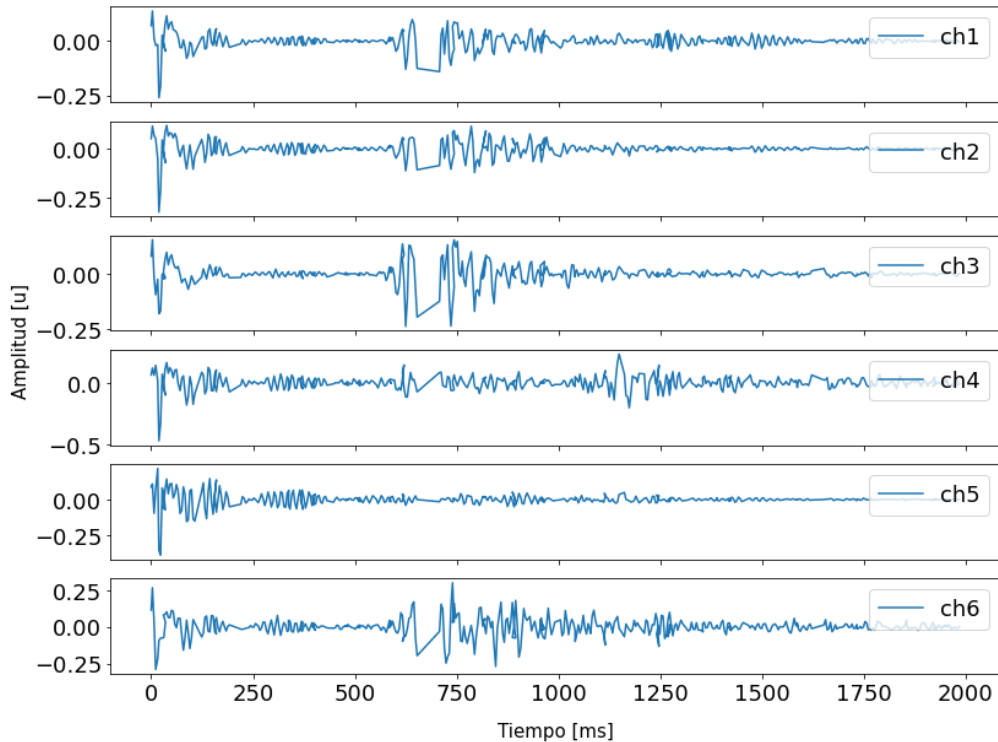


Figura 2.16 Señal filtrada del movimiento 1 repetición 6 de una muestra.

2.5.3 Extracción de características de la señal

En la extracción de características propias de los datos se utilizó la librería TSFresh que permite calcular 77 ecuaciones estadísticas bajo distintos parámetros [48]. Ciertos resultados presentaban valores muy altos y otros por los decimales, por lo que los datos de las características fueron normalizados utilizando la ecuación 2.1.

2.5.4 Selección de características relevantes de la señal

De todo el conjunto de características, se filtró aquellas más relevantes mediante la función *Calculate_relevance_table* de la librería TSFresh [49]. Su aplicación se encuentra en el código E.2.6 del Apéndice E, en donde se detalla los parámetros para obtener aquellas características predictoras de 4 clases (gestos de la mano). De este conjunto se escogió los 20 mejores basados en un algoritmo de *Decision Tree* cuya función se encuentra en el código E.2.7 y el resultado obtenido se puede apreciar en la Figura 2.17.

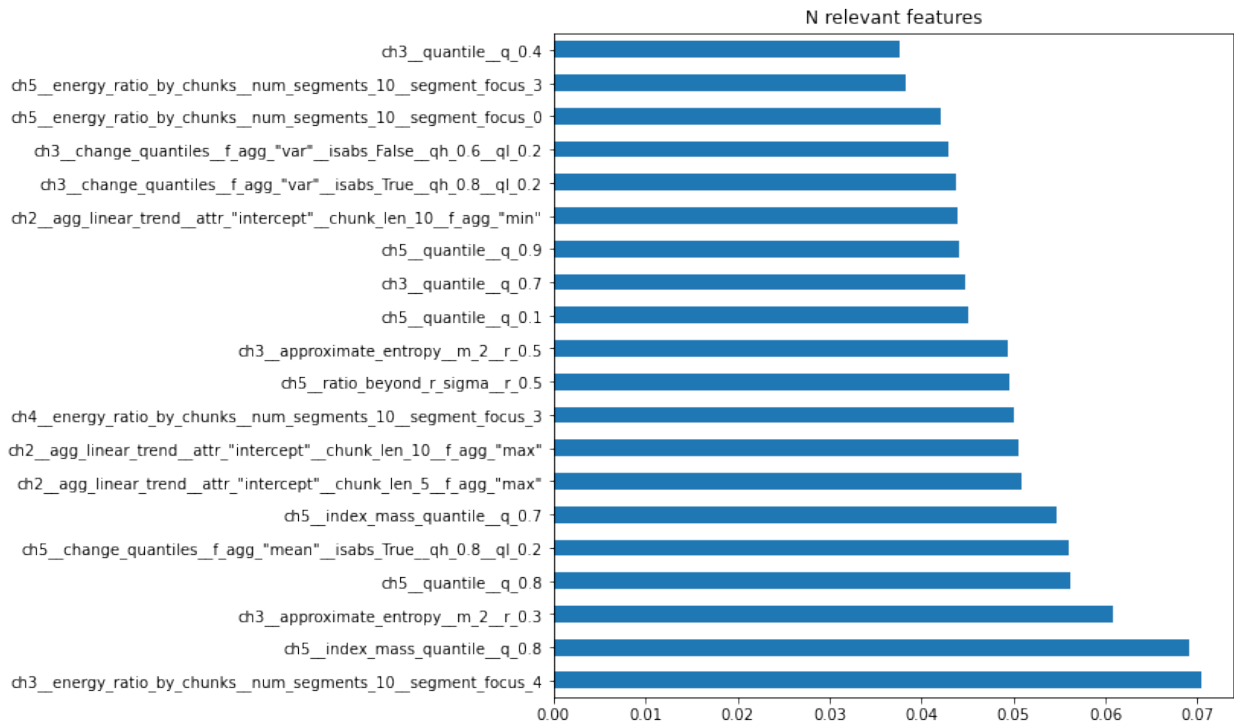


Figura 2.17 Relevancia de las 20 mejores características.

En la Figura 2.18 se aprecia la matriz de correlación obtenida de las 20 características seleccionadas, las cuales pertenecieron a los canales 2, 3, 4 y 5 de la placa CB. El índice de cada eje de la matriz corresponde a la característica ubicada en la Figura 2.17 en orden descendente.

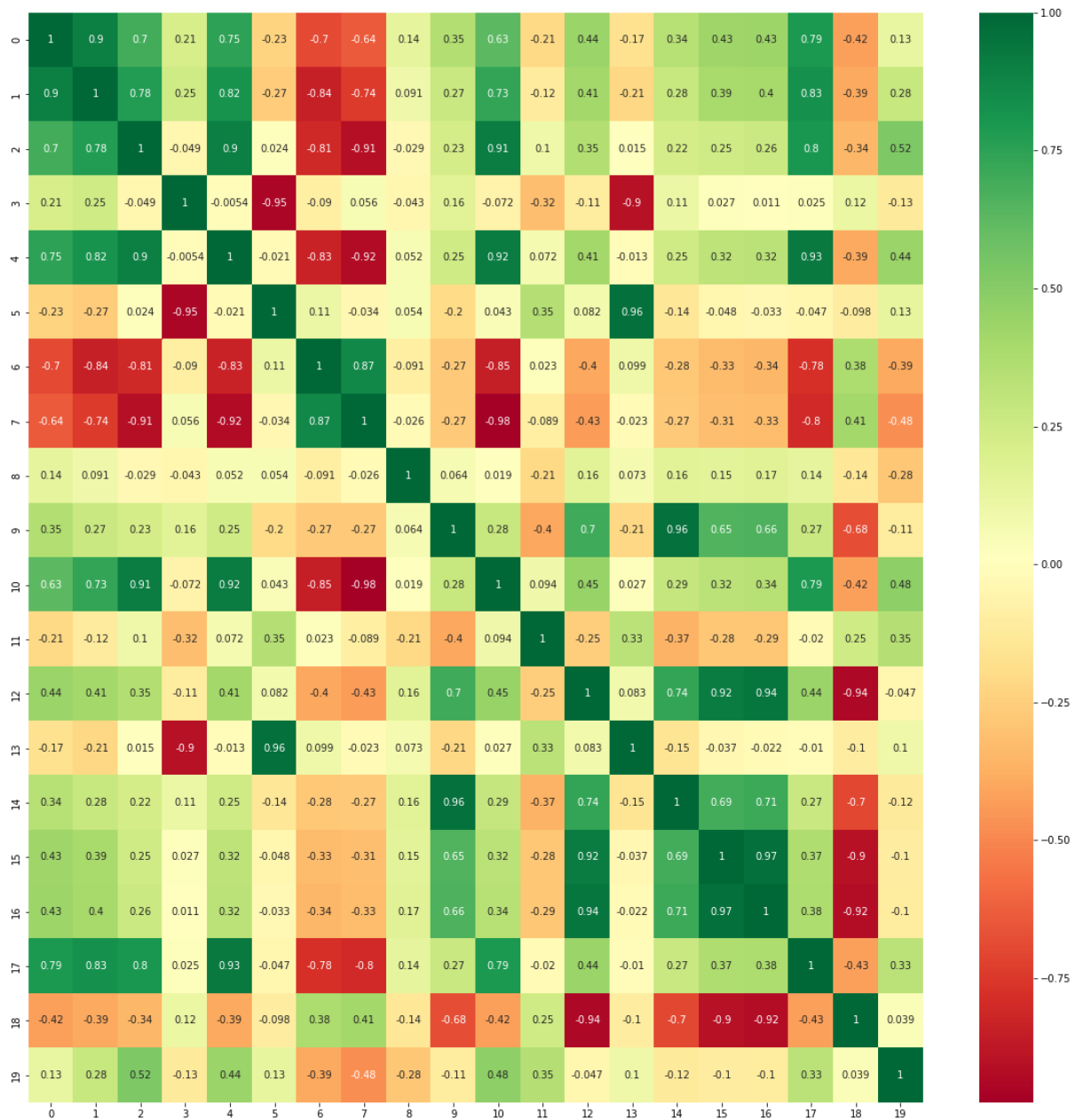


Figura 2.18 Matriz de correlación.

2.6 Diseño de control

Para el diseño del sistema de control se utilizó varios modelos de clasificación ML que permitieran alcanzar una alta exactitud. Para ello, el 80 % del dataset de los sujetos fue destinado para entrenamiento, y un 20 % destinado a evaluación.

2.6.1 Modelos de clasificación

Los parámetros de los modelos fueron modificados hasta haber obtenido el mejor resultado posible, y cuyos parámetros son descritos a continuación, con su respectiva función:

- **Decision tree:** No se requirió de la entrada de algún parámetro, debido a que el modelo creaba sus propias reglas de decisión basándose en las características de los datos de entrenamiento. Su función fue: *"DecisionTreeClassifier()"*.
- **Polynomial:** Este modelo fue basado en un vector de soporte de clasificación cuyo kernel estaba basado en una función de 8^{vo} orden con un parámetro de regularización (C) de 0.001. La función utilizada fue: *"svm.SVC(kernel='poly', degree=8, C=0.001)"*.
- **KNN:** Para el KNN se estableció la cantidad de valores vecinos en 20. El resto de parámetros se mantuvieron en sus valores predeterminados, la cual quedó expresada en: *"KNeighborsClassifier(n_neighbors= 20)"*.
- **Random forest:** Se modificó únicamente el 'random_state' en 21 para afectar la aleatoriedad de las muestras al iniciar la construcción de los árboles. Su función fue: *"RandomForestClassifier(random_state=21)"*.
- **Multi-layer perceptron:** El modelo consta de una red neuronal de 3 capas ocultas con 250, 150 y 50 neuronas respectivamente. Se definió un máximo de 400 iteraciones para la convergencia del optimizador 'adam' con la función de activación 'tanh'. La función implementada fue: *"MLPClassifier((250,150,50), max_iter=400, activation = 'tanh', solver='adam', random_state=1)"*

- **Gaussian process:** Se mantuvo los valores predeterminados de sus parámetros y se modificó el estado inicial aleatorio en 10. La función implementada fue: *"GaussianProcessClassifier(random_state=10)"*.
- **Ridge:** No se requirió de la entrada de algún parámetro, debido a que el modelo creaba sus reglas de decisión basándose en las características de los datos de entrenamiento. Su función fue: *"RidgeClassifier()"*.
- **SGD:** Se mantuvo los valores por defecto de la función quedando como: *"SGDClassifier()"*.

2.7 Implementación

El sistema de control se implementó sobre el diseño de la MRA otorgada por el representante del LNB el cual se aprecia en la Figura 2.19. El diseño consistió en un dispositivo de 6 GDL, uno por cada dedo y solo el pulgar contenía 2 GDL. La MRA fue impresa en 3D con material PLA.

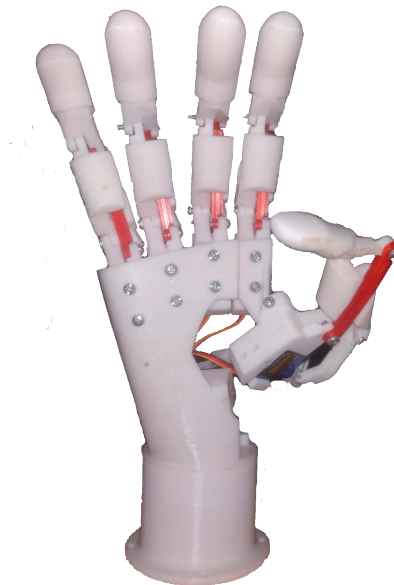


Figura 2.19 Diseño base de MRA.

Para el movimiento de cada GDL se utilizó un pequeño servomotor cuyo par de torque era de 2.5 [kg-cm], todos los demás datos técnicos se encuentran en el Apéndice G. Mediante el módulo PCA9685 se logró establecer el control de los 6 servomotores mediante comunicación I2C con el ESP32. Este módulo actuaba como un pequeño driver de control, sus datos técnicos se encuentran en el Apéndice H.

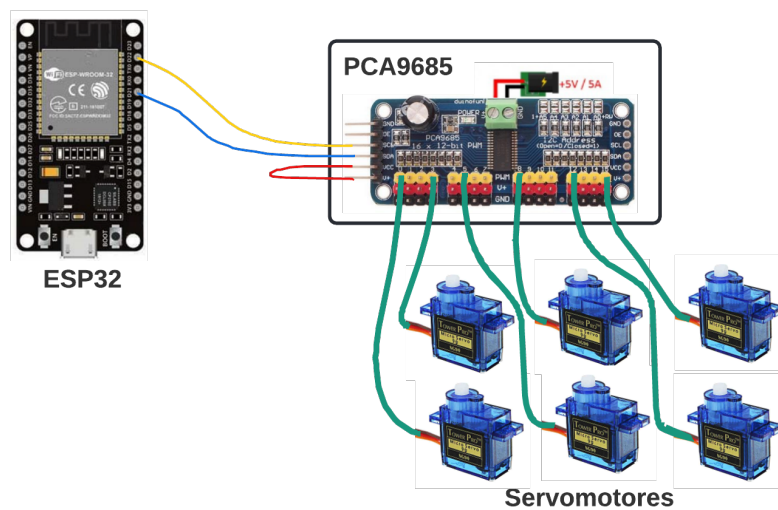


Figura 2.20 Diagrama de conexión de ESP32 y PCA9685 [10].

2.7.1 Cálculo de consumo energético

Para alimentar los distintos componentes de la alternativa, esta fue separada en dos fuentes: una fuente de fuerza para los servomotores y otra fuente para los componentes de lógica de control. La fuente de lógica de control escogida era un módulo de alimentación de 5 [V] y 700 [mA] (véase Apéndice I), los cuales eran suficientes para alimentar el módulo PCA9685 y al ESP32, cuyo consumo utilizando WIFI fue aproximadamente de 180 [mA] (véase Apéndice J) al igual que el controlador de servos.



Figura 2.21 Fuente de alimentación de protoboards [11].

Por otro lado, se dimensionó la fuente de fuerza mediante el consumo aproximado de los servomotores, los cuales variaban entre 150 [mA] hasta 1100 [mA] según la velocidad de movimiento de la carga y su torque [50]. En la ecuación 2.6 se puede apreciar que el consumo promedio de los 6 servomotores fue de 4.88 [A] aplicando un factor de seguridad (F_s) de 1.3, por lo que se utilizó una fuente de poder con su inmediato superior siendo esta de 5 [V] con 5 [A].

$$\begin{aligned}\bar{I} &= (150 + 1100)/2 = 625[mA] \\ I_{total} &= n \times \bar{I} \times F_s \\ I_{total} &= 6 \times 625 \times 1.3 \\ I_{total} &= 4875[mA] \approx 4.88[A]\end{aligned}\tag{2.6}$$

2.7.2 Rest API

Se desarrolló una Rest API que permitía comunicar el sistema de control alojado en el PC con el ESP32, el cual controlaba los servomotores de la MRA. Este medio de comunicación trabaja de forma local con la IP de la red WIFI por lo que no hubo muchos problemas de latencia. El programa desarrollado se encuentra en el Apéndice E código E.4.2.

CAPÍTULO 3

3. RESULTADOS Y ANÁLISIS

En el presente capítulo se presenta: los resultados obtenidos de cada modelo de clasificación ML y su elección, el diseño final del sistema de control y los resultados de la implementación. También se muestra el diagrama eléctrico y el diseño 3D con las modificaciones realizadas en la MRA.

3.1 Adquisición y procesamiento de datos.

De los sujetos de la Tabla 2.5 se obtuvieron un total de 920 repeticiones entre los 4 gestos escogidos y un total de 4736 características propias de los datos, de los cuales se seleccionaron las 20 mejores que se muestran en la Figura 2.17 para el entrenamiento y validación del modelo de clasificación.

3.2 Diseño de sistema de control

3.2.1 Selección de modelo de clasificación

En la Tabla 3.1 se aprecian los resultados del entrenamiento y validación con distintos modelos de clasificación ML. En ella se puede apreciar que los modelos como el *decision tree*, *polynomial*, *random forest* y *MLPC* tuvieron una exactitud mayor al 85% en el entrenamiento e incluso algunos de los anteriores mencionados lograron alcanzar el 100%. Sin embargo, la exactitud en la validación bajó considerablemente en el *decision tree* y en el *SGD*. Solo el modelo *MLPC* y *polynomial* tuvieron una exactitud de evaluación mayor al 80%, siendo el ganador el modelo *MLPC* con un 84.78%. Las funciones desarrolladas pueden encontrarse en el Apéndice E código E.3.1.

Tabla 3.1 Resultado de entrenamiento y validación de modelos de clasificación

Modelo de clasificación	Exactitud de entrenamiento	Exactitud de validación
Decision tree	100.00 %	60.33 %
Polynomial	86.53 %	80.98 %
KNN	75.51 %	72.28 %
Random forest	100.00 %	73.37 %
MLPC	91.16 %	84.78 %
Gaussian process	72.38 %	71.74 %
Ridge	72.65 %	70.11 %
SGD	72.65 %	57.07 %

El mejor resultado entre los modelos de clasificación fue el *MLPC* pero no necesariamente el modelo bajo esos parámetros era el óptimo. Por lo que se realizó una evaluación cruzada aplicando el método *5-fold* con distintos parámetros del modelo *MLPC*, para lo cual se tuvo la Tabla 3.2. El resultado obtenido fue el modelo *MLPC* bajo los siguientes parámetros: *MLPClassifier(activation = 'relu', alpha= 0.0001, hidden_layer_sizes = (250, 150, 50), max_iter=400, solver = 'adam', random_state=1)*, el cual fue el mismo con el que se inició el análisis. Se determinó que desde el inicio el modelo establecido presentaba los mejores parámetros para obtener una alta exactitud en el modelo *MLPC*.

Tabla 3.2 Parámetros de evaluación cruzada.

Parámetros	Datos
alpha	[0.0001]
hidden_layer_sizes	[(250, 150, 50), (250, 500, 60), (1200, 400, 50), (600,250, 60), (400,250, 50)]
solver	[lbfgs, Adam]
activation	[relu,tanh]
max_iter	[400]

Para los resultados de la Figura 3.1 se aplicó aleatoriedad a los datos y se escogía cada vez el 20 % de los datos para validación y el resto para entrenamiento. De esta forma se podía determinar la evolución del modelo conforme se entrenaba con datos virtualmente nuevos. De ello, se puede apreciar que el rendimiento del modelo llegó alcanzar casi el 80 % con una varianza semi constante de aproximadamente de 2.5 %. El modelo obtenido es escalable, pero debido a la variabilidad de los sujetos y el reuso de los electrodos al adquirir los datos, se requería de un entrenamiento cada vez que se colocan los electrodos.

Posteriormente, se realizó una evaluación cruzada con *5-fold*, en el que se determinó el comportamiento de la exactitud frente a datos de entrenamiento y validación virtualmente distintos. Los resultados obtenidos se precian en la Tabla 3.3, en la cual se observó que el promedio obtenido para el valor *F1* fue de 0.7961. Este resultado tenía un buen promedio ponderado de precisión y recuperación del modelo siendo cercano a 0.8. Por otro lado, debido al tipo de entrenamiento y validación ejecutado la exactitud promedio obtenida fue un poco inferior comparada con la obtenida en los resultados de la Tabla 3.1.

Tabla 3.3 Resultados de evaluación cruzada con modelo seleccionado.

K-fold	valor F1	Exactitud
1-fold	78.48 %	75.67 %
2-fold	74.42 %	76.74 %
3-fold	80.55 %	75.67 %
4-fold	80.99 %	81.08 %
5-fold	83.59 %	83.33 %
Promedio	0.7961 %	0.7850 %

Del reporte de la Tabla 3.4 y de su respectiva verificación con la Figura 3.2, se determinó que el modelo poseía un buen recall y precisión al momento de clasificar las señales de entrada. La diagonal de la Figura 3.2 indica las estimaciones correctamente predichas (verdaderos

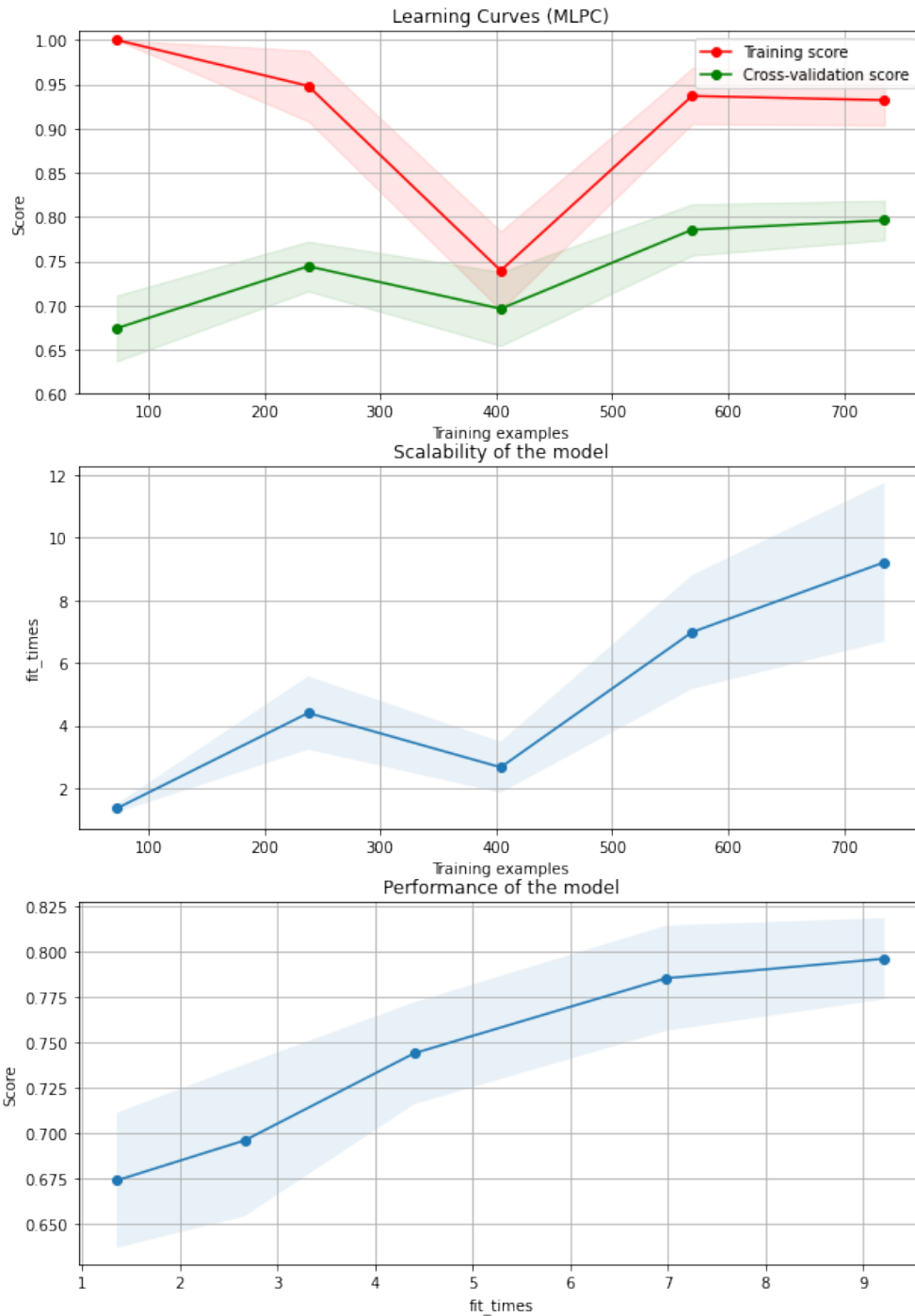


Figura 3.1 Curvas de entrenamiento y validación, escalabilidad y rendimiento del modelo seleccionado.

positivos) y los valores de alrededor indican aquellas estimaciones que no se lograron predecir correctamente (falsos positivos). Tomando en cuenta ello, se logró observar con más claridad que el gesto 1 fue el que tuvo más problemas al confundirse 7 veces con el gesto 3, por lo que un entrenamiento con más repeticiones del gesto 1 y del gesto 4 (el cual también presenta problemas en comparación con los demás) podrían mejorar la exactitud del modelo.

Tabla 3.4 Reporte del modelo MLPC seleccionado.

Class	Precision	Recall	F1-score	Support
1	0.86	0.74	0.79	42
2	0.93	0.93	0.93	44
3	0.72	0.84	0.77	37
4	0.87	0.87	0.87	61
Accuracy			0.85	184
macro avg	0.85	0.84	0.84	184
weighted avg	0.85	0.85	0.85	184

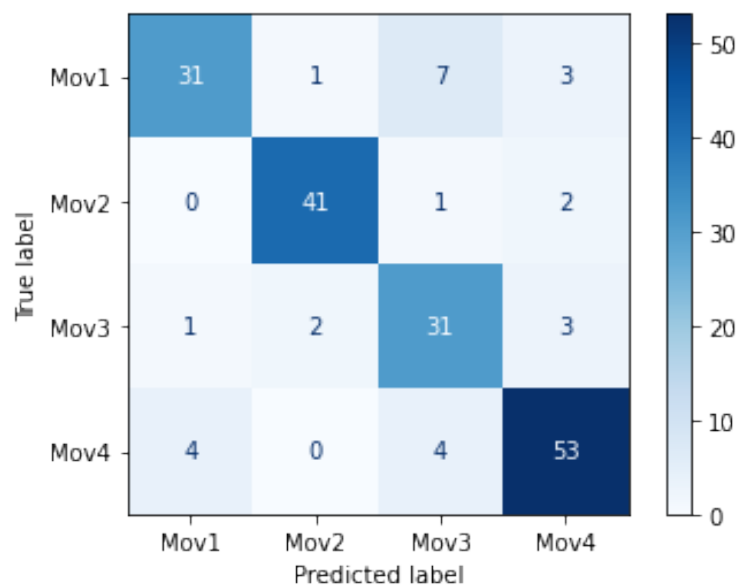


Figura 3.2 Matriz de confusión.

Finalmente, se realizó una curva ROC (Receiver Operating Characteristic) por cada clase. En ella se pudo apreciar que las 4 clases poseen un alta

área bajo la curva por lo que se procedió analizar una buena relación entre sensibilidad y especificidad. En la Figura 3.3 se determinó que en la zona encerrada en rojo, las curvas son un poco más estables, su sensibilidad no es menor a 0.8 y la especificidad es cercana a 0.1 siendo esta la zona mas óptima del modelo. Un mayor grado de sensibilidad podría llevar a una mayor falencia debido a que se incrementa considerablemente los falsos positivos.

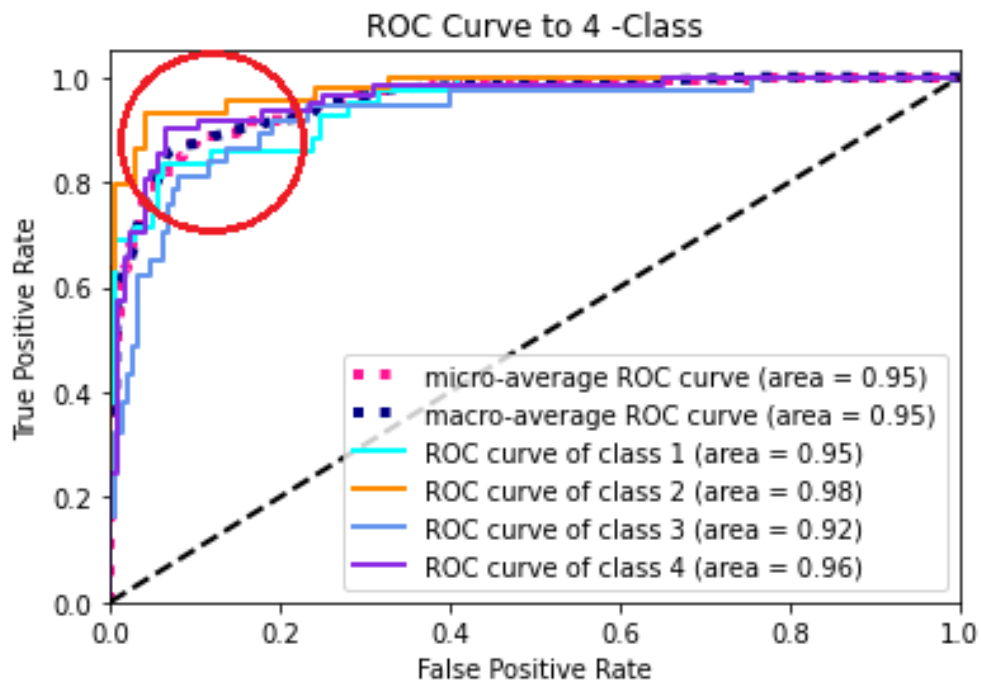


Figura 3.3 Curva ROC.

3.2.2 Sistema de control

Una vez entrenado el modelo de clasificación escogido, dicho modelo fue incorporado en el sistema de control. El sistema fue de lazo abierto debido a que la señal de salida no afectaba la funcionalidad del sistema, véase Figura 3.4. Este sistema de control fue alojado en la PC, la cual adquiría la señal mioeléctrica de 6 canales del CB mediante comunicación serial con el dispositivo USB. Debido a ello el sistema es de tipo MISO porque emite una sola salida de control y recibe 6 conjuntos de varios datos de entrada.

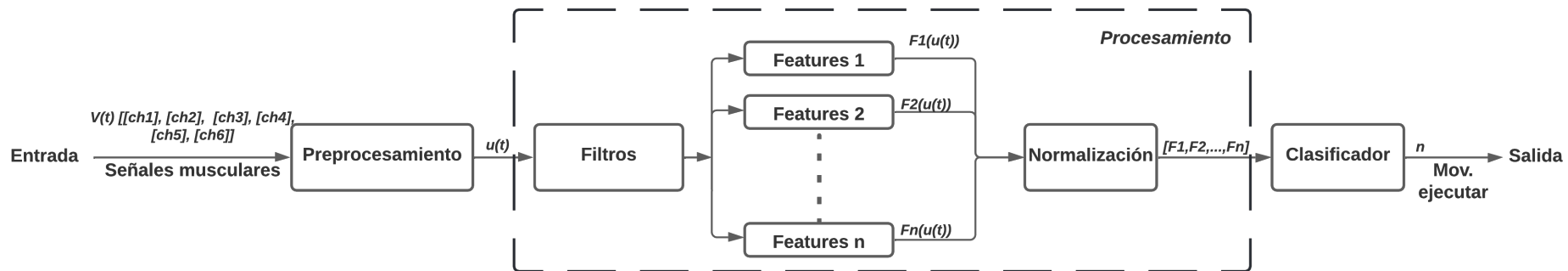


Figura 3.4 Diagrama de bloques del sistema de control del clasificador.

44

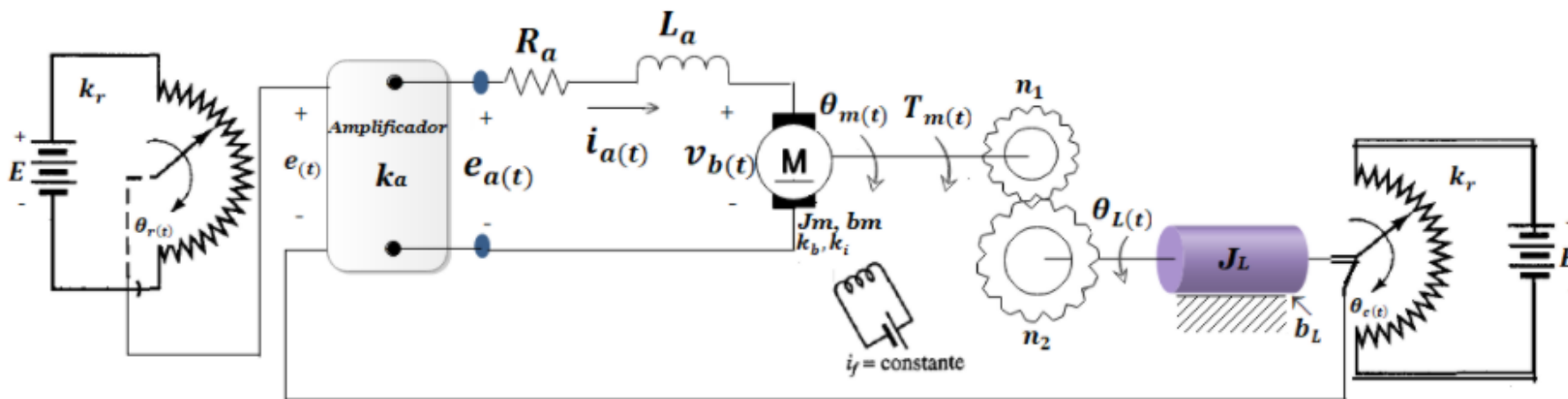


Figura 3.5 Diagrama de sistema de control de posición de un servomotor [12].

Por otro lado, para el control de los servomotores por parte del ESP32 se estableció un control proporcional de tipo SISO (*Single Input Single Output*) mediante el manejo de las señales PWM (*Pulse Width Modulation*), véase Figura 3.6. En [51] se puede encontrar un análisis mas detallado del sistema de control por PWM basado en la Figura 3.5. Dicho control establecía la posición para cada servomotor hacia el controlador PCA9685, el cual tenía incorporado en su lógica de control un controlador de posición (semejante al de la Figura 2.21). Por lo que se utilizó dichas funciones para pre-establecer los movimientos que cada dedo debe realizar para imitar el gesto. En el Apéndice E código E.4.1 se puede encontrar el programa para el ESP32, el cual también efectuaba la comunicación con el servidor alojado en el PC.

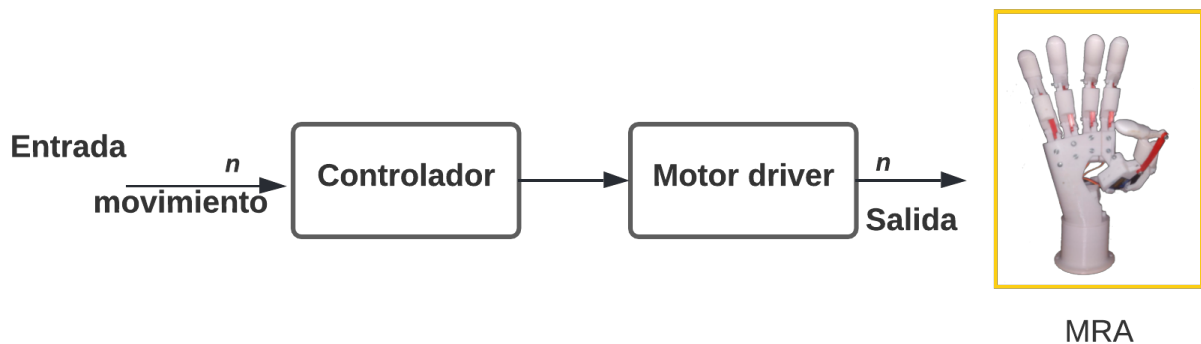


Figura 3.6 Diagrama de bloques del sistema de control de los servomotores.

3.3 Implementación

En esta sección se presenta los resultados obtenidos durante las pruebas realizadas con el modelo de clasificación escogido, así como también las distintas modificaciones en el diseño de la MRA y sus diagramas de conexión y desarrollo de la Rest API.

3.3.1 Sistema eléctrico

Tomando en cuenta las dimensiones que deben tener las fuentes de la parte lógica y de control, se estableció la respectiva conexión entre los elementos necesarios para el funcionamiento de la MRA, los cuales se

aprecian en la Figura 3.7.

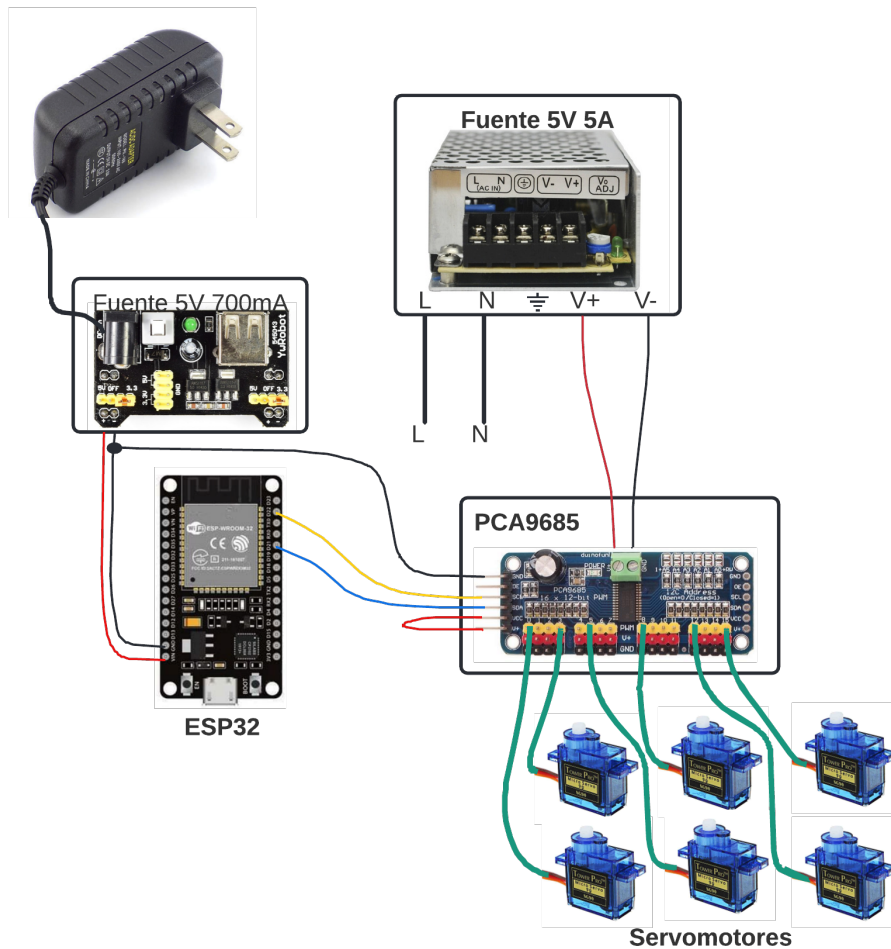
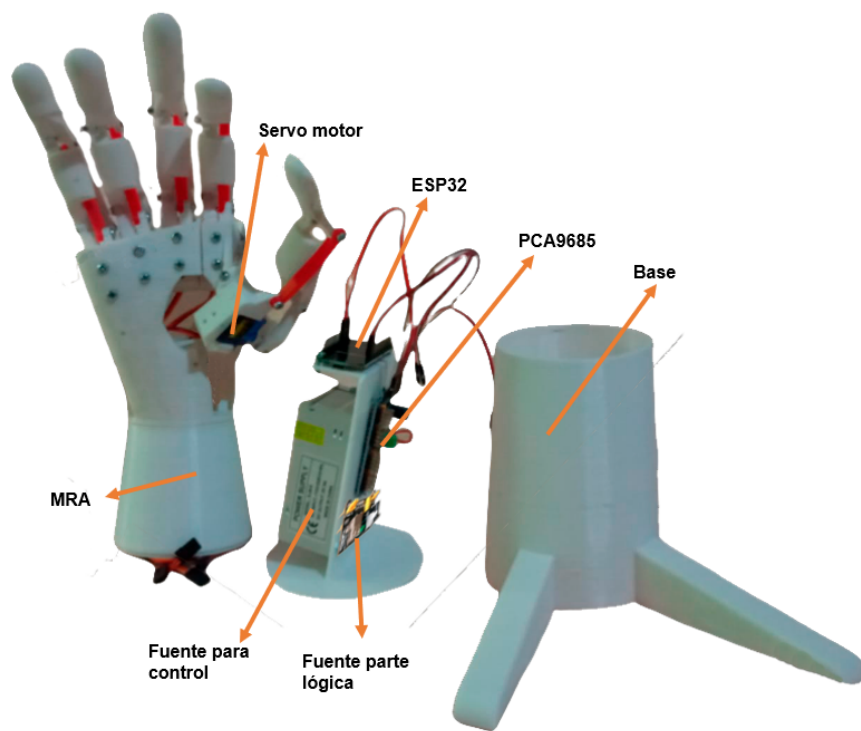


Figura 3.7 Diagrama de conexión del sistema eléctrico.

3.3.2 Diseño final

Entre los elementos con mayor tamaño se tenía la fuente de control de los servomotores, por lo que se rediseñó la base de la MRA para incorporar todas los elementos eléctricos en su interior. En la Figura 3.8 se puede apreciar las 3 secciones que complementan la base de la MRA. En el Apéndice K se encuentran las respectivos planos de la base.



(a) Segmentos de la MRA

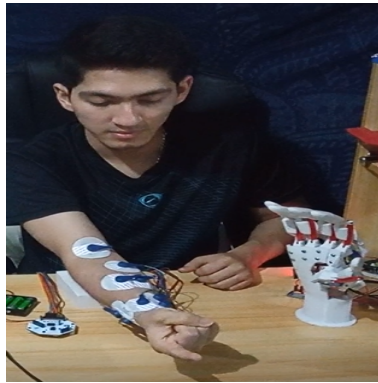


(b) Diseño armado de la MRA.

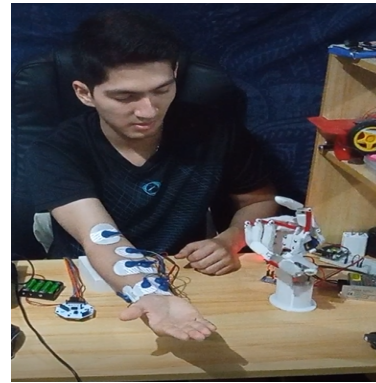
Figura 3.8 Diseño final de la MRA.

3.3.3 Resultados finales de implementación.

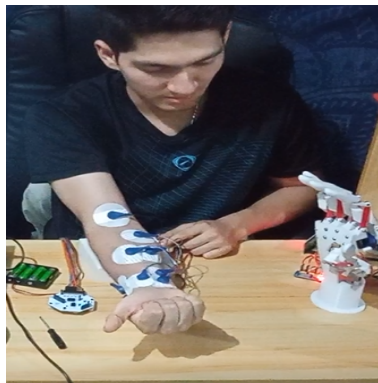
Para la sección de prueba se utilizó la base pequeña para comodidad de las fotos y evaluación de los resultados. En la Figura 3.9 se puede apreciar los distintos movimientos que fueron imitados por la MRA.



(a) Gesto 1



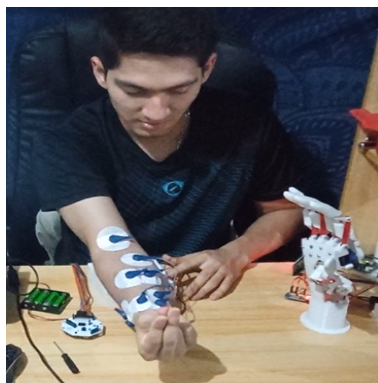
(b) Gesto 1 en MRA



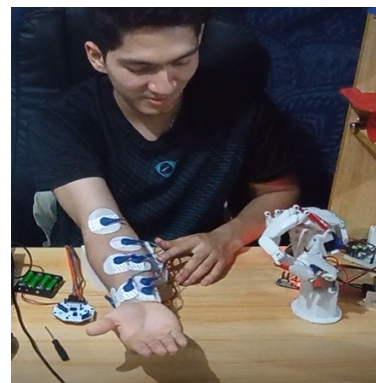
(c) Gesto 2



(d) Gesto 2 en MRA

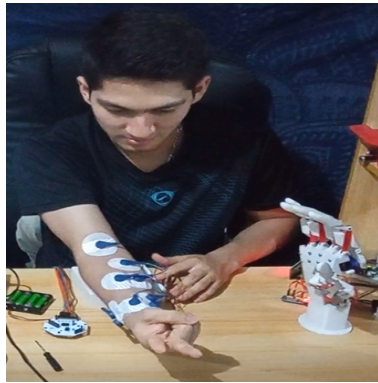


(e) Gesto 3



(f) Gesto 3 en MRA

Figura 3.9 Resultados de predicción de implementación del diseño. Parte 1.



(a) Gesto 4



(b) Gesto 4 en MRA

Figura 3.10 Resultados de predicción de implementación del diseño. Parte 2.

En la Figura 3.11 se puede apreciar la secuencia de operación de la solución. El tiempo de reacción, desde que se realiza el gesto en la mano hasta que inicia el movimiento de los dedos en la MRA, es de máximo 2 segundos. En este tiempo interviene las variables del periodo de lectura del ESP32 en la Rest API y el tiempo en que el sistema de control determina el tipo de movimiento realizado.

En la prueba de implementación se realizó 10 gestos entre los 4 establecidos de forma aleatoria, de los cuales 7 fueron correctamente determinados por el sistema de clasificación del modelo seleccionado. Sin embargo, previamente se realizó un entrenamiento del modelo con una sesión de 20 repeticiones por gesto debido a que se determinó que los datos adquiridos también variaban dependiendo del estado del electrodo. Por lo general, los electrodos solo soportaban ser extraídos hasta un máximo de 3 a 4 veces; y fue un factor clave a tomar en cuenta para no afectar en gravedad la exactitud del modelo.

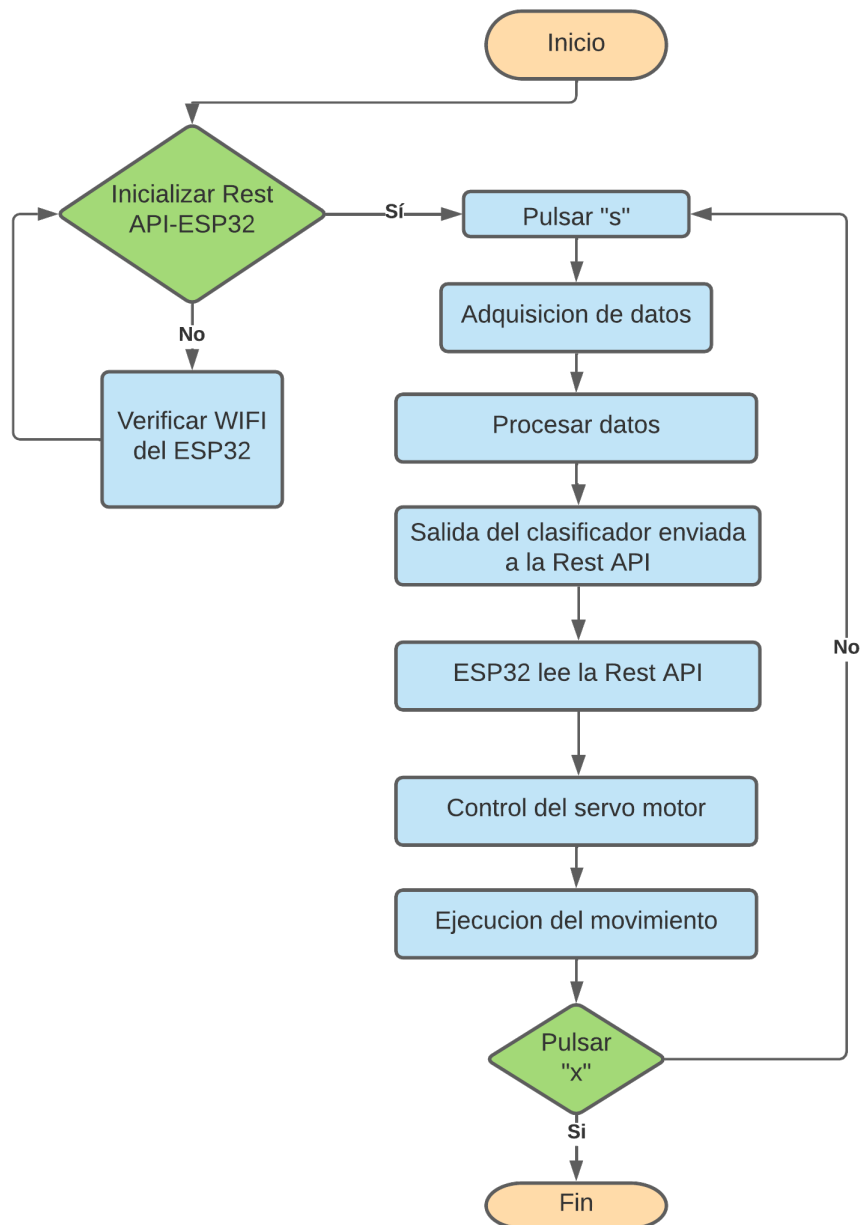


Figura 3.11 Diagrama de Flujo de funcionamiento del diseño de implementación.

3.4 Análisis de Costos

En la Tabla 3.5, se detalla los rubros para el diseño mecánico, eléctrico y de control; así como también el costo de la mano de obra para este proyecto.

Como se determinó que el sistema utiliza únicamente 4 de los 6 canales que inicialmente se estableció, se cotizó por uno de 4 canales como el Ganglion Board [52]. Este dispositivo trabaja de manera similar y su costo es menor.

Tabla 3.5 Cotización de rubros.

Tipo de sistema	Descripción	Cantidad	unidad	Precio Unit (USD)	Total (USD)
Mecánico	Impresión 3D	1	u	\$150.00	\$150.00
	Servomotor	6	u	\$4.25	\$25.50
Eléctrico y de control	Ganglio Board	1	u	\$375.00	\$375.00
	Paquete de electrodos	1	u	\$15.00	\$15.00
	USB Ganglio Board	1	u	\$15.00	\$15.00
	Cables de electrodos	1	u	\$45.00	\$45.00
	Baterías recargables	4	u	\$1.50	\$6.00
	Fuentes de poder 5V 5A	1	u	\$10.00	\$10.00
	Módulo PCA9685	1	u	\$8.00	\$8.00
	ESP32	1	u	\$12.00	\$12.00
	Cargador 12V	1	u	\$5.00	\$5.00
	Fuente 5V	1	u	\$3.50	\$3.50
Ingeniería	Computadora	1	u	\$400.00	\$400.00
	Diseño mano de obra	2	u	\$200.00	\$400.00
	Viáticos	1	u	\$100.00	\$100.00
				Total	\$1570.00

A nivel nacional los precios por equipos médicos son muy elevados debido a que en su mayoría son exportados y cuyos impuestos son altos. Por lo tanto, la opción que se presentó es más económica en comparación con otros equipos similares tanto nacionales como internacionales, como se puede apreciar en la Tabla 3.6. La MRA desarrollada es mucho mas económica, dándonos la ventaja de producirla localmente y pueda ser utilizada como un recurso didáctico de aprendizaje en las distintas entidades de educación superior o centros de investigación.

Tabla 3.6 Comparativo entre prótesis de mano.

Marca / Academia	Foto	Sistema Didáctico	Implementación AI	Precio
Proteus		No	Si	\$3.500,00
ESPOCH		No	No	\$3.000,00
Open Bionics		No	Si	\$5.000,00
Universidad Salesiana		No	No	\$15.000,00
Diseño Propuesto		Si	Si	\$1.570,00

CAPÍTULO 4

4. CONCLUSIONES Y RECOMENDACIONES

En este trabajo se presenta el diseño e implementación de una mano robótica antropomórfica con seis grados de libertad, la cual identifica cuatro movimientos distintos mediante la captura de señales mioeléctricas de los músculos del antebrazo. Para ello, se utilizó 6 canales de la placa Cyton Board para la adquisición de datos. En el análisis realizado se trabajó con un total de 10 personas, de las cuales 6 fueron de mujeres y 4 de hombres. Estas señales fueron filtradas y se les determinó las características más notables para entrenar un modelo de clasificación ML. El modelo con los mejores resultados fue escogido para incorporarlo en el sistema de control que está alojado en un PC. La señal de control es emitida a un ESP32 mediante el uso de una REST API. El ESP32 es el encargado de realizar el control de los servomotores para recrear el movimiento del gesto de la mano.

Como resultado de la comparación de 8 tipos de modelos de clasificación ML, se observó que 4 de ellos tuvieron un resultado mayor al 85% en la etapa de entrenamiento. Dos de ellos presentaban un resultado del 100%, los cuales fueron inmediatamente descartados debido a que presentaban una baja exactitud en la etapa de validación siendo un indicativo de que los modelos presentaban problemas de *overfitting* (modelo sobreajustado, no se adapta a nuevas muestras). Por lo que, utilizando el modelo *MLPC* se muestra una mejora en los resultados de clasificación, la cual tiene una precisión del 84,78%. Su curva de aprendizaje (Figura 3.1) permitía evidenciar una buena adaptabilidad en la recuperación del modelo ante la variabilidad a los datos de los sujetos. Con el modelo seleccionado se obtuvo un tiempo de reacción de 2 segundos para la MRA. Además analizando su reporte en la Tabla 3.4, se evidenció que el segundo movimiento tenía una precisión del 93%, siendo la más alta en comparación con los movimientos restantes. Por otro lado, el tercer movimiento presentó el valor de precisión mas bajo equivalente al 72%. Dichos resultados, permitían evidenciar una mayor

complejidad de determinar el gesto 3 para el modelo seleccionado. Estos valores también los podemos comprobar cuando se realiza la curva ROC en la Figura 3.3, en la que se aprecian las cuatro clases (o gestos). La curva mas cercana a la unidad es la curva del segundo gesto, con una alta tasa de verdaderos positivos. En la gráfica se seleccionó la zona donde la sensibilidad de las curvas no es menor a 0.8 y la especificidad es cercana a 0.1, la cual indicaba el equilibrio mas óptimo para el modelo. Cabe mencionar que un mayor grado de sensibilidad podría llevar a una mayor falencia debido al incremento considerable de los falsos negativos.

Proyecto similares se han desarrollado, pero aplicando distintas metodologías y modelos de clasificación. Siendo como base de la metodología la interpretación de datos mioelectricos para el accionamiento de manos robóticas antropomórficas, se tiene la aplicación de una red neuronal CNN obteniendo una variación del 60% a un 80% aproximadamente entre distintos sujetos [39]. En su investigación utilizaron las señales mioeléctricas de la parte superior tanto del brazo como del cuerpo sin considerar los músculos de la muñeca, que a diferencia del presente trabajo se utilizó la mayor parte de los músculos del antebrazo que contienen los músculos que inciden directamente en el movimiento de los dedos de la mano. Por otro lado, en [7] se utilizó un brazalete EMG que se adaptaba al tamaño del antebrazo del usuario. Dicho brazalete mantenía la ubicación de los electrodos, pero ciertos canales presentaban dificultad de lectura debido a la disposición de los músculos que a diferencia de la metodología aplicada en el presente proyecto, se trató de localizar los músculos mas potenciales al movimiento de la mano. Sin embargo el resultado obtenido del modelo fue muy similar, siendo la exactitud en [7] del 84.7% y en el modelo *MLPC* del 84.78%. Por otro lado, en [8] se implementó una red neuronal recurrente mezclando un modelo logarítmico-linealizado gaussiano (R-LLGMN) obteniendo una precisión del 90% entre sus pruebas realizadas. Su resultado fue mayor debido a que es un modelo mucho más robusto permitiendo un mejor aprendizaje de los datos ante cualquier variación de los sujetos.

4.1 Conclusiones

- En la adquisición de datos mioeléctricos de 6 canales de los participantes se descartaron aquellas sesiones en las que el sujeto no realizó el movimiento con la suficiente fuerza, se equivocaba, o por la medición de los electrodos. Los electrodos, según su tiempo de uso, presentan una alta resistencia generando un detenimiento o baja amplitud de la señal adquirida. Las sesiones escogidas fueron filtradas para luego extraer 20 características estadísticas propias seleccionadas para el modelo de clasificación. El escoger otras características podrían afectar la exactitud del modelo.
- El modelo de clasificación implementado en el sistema de control tuvo una exactitud del 84.78 % y los datos pertenecían únicamente a 4 canales de la placa debido a las características seleccionadas. Por lo tanto, para el reconocimiento de los 4 gesto seleccionados, solo es necesario el registro de los canales ch2, ch3, ch4 y ch5, según el orden establecido en la Figura 2.8.
- Durante la prueba de implementación se tuvo un error del 30% en los gestos aleatorios realizados, reduciendo la exactitud inicial. Esto fue debido a la variabilidad de los sujetos, el estado de los electrodos, y la dificultad de determinar un gesto. El gesto 3 fue el que presentó una mayor dificultad, véase la Figura 3.4, debido a fue un movimiento que se realiza con menor intensidad y se flexionaba parcialmente los dedos. Por lo tanto, el clasificador va a presentar un mejor resultado para cualquier gesto cuyo movimiento de los dedos presente una mayor actividad muscular, caso contrario se va a requerir una mayor cantidad de datos para el gesto.
- EL modelo MLPC seleccionado tuvo un buen desempeño en la implementación, pero el modelo escogido no es definitivo. Por lo tanto, la aplicación del modelo con otros gestos puede presentar ciertas falencias que puede llegar a requerir de un mayor *dataset* al obtenido para el entrenamiento de nuevos gestos de la mano. De igual forma, la arquitectura desarrollada para la clasificación de los datos EMG recogidos

a 250Hz es compatible con el equipo Cyton Board y sus semejantes como el Ganglion Board que pertenecen al mismo proveedor.

- El tiempo de reacción de la MRA fue de 2 segundos el cual era afectado por el tiempo de procesamiento y de clasificación de los datos, así como el tiempo de muestreo del consumo de la ESP32 en el servidor. Una de las ventajas de tener un tiempo de respuesta alto en nuestro proyecto, es que permitirá al estudiante comprender la complejidad del procesamiento interno del proyecto, también la necesidad de recursos computacionales para lograr un mecanismo robusto y con un bajo tiempo de respuesta.
- El diseño desarrollado cumple con los objetivos 3 y 4 de desarrollo sostenible debido a que es un dispositivo mecatrónico didáctico que permitirá desarrollar nuevos conocimientos y habilidades de los estudiantes de educación superior.
- Aunque los equipos biomédicos tiene un alto valor, el precio final del proyecto fue de \$1 570.00 dolares, siendo un 51 % mas económico que otros proyectos de similares características. El precio fue tomando en consideración el cambio de la placa de adquisición de datos a uno de 4 canales, lo cual representa un precio asequible como equipo de laboratorio comparado con equipos comerciales de similares características.

4.2 Recomendaciones

- Para la selección de las personas que participaron en el presente trabajo, se limitó a aquellas personas que estén en un rango de edad entre los 18 a 50 años y que no posean un alto nivel de grasa corporal. Esto es debido a que la grasa en el tejido puede opacar la señal adquirida; y el rango de edad escogido presentaba una mayor actividad eléctrica en los músculos. De igual forma, se recomienda que durante las sesiones, los participantes realicen los movimientos con normalidad sin ejercer una alta ni baja fuerza, de forma que se pueda obtener los movimientos más naturales posibles.
- Para una mejor medición de las señales musculares se recomienda primero: limpiar toda el área del brazo con alcohol para la eliminación de grasa y gérmenes; y usar un gel para electrodos, el cual mejora la

conductividad en la medición mioeléctrica. También se recomienda utilizar esparadrapos sobre los electrodos para establecer un mejor contacto con la piel. Sin embargo, estos no deben hacer una alta presión sobre los electrodos debido a que afectaría en la amplitud de la señal adquirida.

- Para brindarle un mayor nivel de antropomorfismo a la MRA, se recomienda elaborar un guante de latex cuyo diseño sea lo mas parecido a una mano humana. Este guante debe ser delgado para no incrementar la resistencia en el movimiento de los dedos de la MRA, lo que conllevaría a un consumo mayor de corriente o al detenimiento de la MRA.
- Para tener una alta exactitud del modelo durante la implementación, se recomienda realizar al menos una sesión de 20 repeticiones por gesto para volver a entrenar el modelo de clasificación. Sobretudo, si es un sujeto nuevo quien probaría el dispositivo mecatrónico didáctico, debe ser una cantidad mayor a 20 repeticiones por gesto.

BIBLIOGRAFÍA

- [1] “Encuesta nacional de actividades de ciencia, tecnología e innovación-acti —.”
- [2] “Indicadores – ricyt.”
- [3] D. Hwang, E. J. Barron, A. B. M. T. Haque, and M. D. Bartlett, “Shape morphing mechanical metamaterials through reversible plasticity,” *Science Robotics*, vol. 7, 2 2022.
- [4] J. V. Pinzón, R. P. Mayorga, and G. C. Hurtado, “Brazo robótico controlado por electromiografía,” *Scientia et technica*, vol. 1, no. 52, pp. 165–173, 2012.
- [5] L. J. Hargrove, K. Englehart, and B. Hudgins, “A comparison of surface and intramuscular myoelectric signal classification,” *IEEE Transactions on Biomedical Engineering*, vol. 54, no. 5, pp. 847–853, 2007.
- [6] J. Correa-Figueroa, E. Morales-Sánchez, J. Huerta-Ruelas, J. González-Barbosa, and C. Cárdenas-Pérez, “Sistema de adquisición de señales semg para la detección de fatiga muscular,” *Revista mexicana de ingeniería biomédica*, vol. 37, no. 1, pp. 17–27, 2016.
- [7] H. Wang, J. Qian, and Z. Zhang, “Research on mechanism design and gesture recognition application of high biomimetic anthropomorphic robotic hand,” in *2020 IEEE International Conference on Information Technology, Big Data and Artificial Intelligence (ICIBA)*, vol. 1, pp. 343–347, 2020.
- [8] A. Furui, S. Eto, K. Nakagaki, K. Shimada, G. Nakamura, A. Masuda, T. Chin, and T. Tsuji, “A myoelectric prosthetic hand with muscle synergy-based motion determination and impedance model-based biomimetic control,” *Science Robotics*, vol. 4, 6 2019.
- [9] “Cyton biosensing board (8-channels) – openbci online store.”
- [10] J. Damián, “Tutorial módulo controlador de servos pca9685 con arduino - electrogeek,” 5 2020.

- [11] SoloArduino, "Mb102 - alimentación para breadboards," 2016.
- [12] carakenio73, "Sistema de control de posición – servomotores.," 2018.
- [13] C. S. Johnson, *A Brief History of Bionic Sonars*, pp. 769–771. Boston, MA: Springer US, 1988.
- [14] J. M. D. González, P. R. Murillo, I. F. Luna, and A. J. Mendoza, "Robótica y prótesis inteligentes," *Universidad Nacional Autónoma de México*, pp. 1–15, 2005.
- [15] B. G. Andrews, E. E. Harris, H. Hederick, K. P. Robinson, and M. V. et al, *Amputaciones y protesis*, vol. 1. 1 ed., 1985.
- [16] I. Vujaklija, D. Farina, and O. C. Aszmann, "New developments in prosthetic arm systems," *Orthopedic Research and Reviews*, vol. 8, pp. 31–39, 7 2016.
- [17] P. J. S. Guamán, M. P. G. Sánchez, and K. P. S. Guamán, "La educación superior en el ecuador vs las nuevas tecnologías," *RECIMUNDO*, vol. 1, pp. 484–496, 12 2017.
- [18] L. Schwartz and C. Guaipatín, "Ecuador: Análisis del sistema nacional de innovación: Hacia la consolidación de una cultura innovadora," *Monografía del BID (Instituciones para el Desarrollo. División de Competitividad e Innovación). IDB-MG-223*, 2014.
- [19] R. R. Serrezuela, J. L. A. Trujillo, D. R. Delgado, V. K. O. Benavides, R. S. Zamora, and E. M. Reyes, "Diseño e implementación de una prótesis de mano robótica antropomórfica subactuada," in *Memorias de Congresos UTP*, pp. 165–172, 2018.
- [20] Y. O. Rodríguez, A. O. Prado, L. M. Acosta, and R. G. Carbonell, "Modelo cad de mano humana de 28 grados de libertad para el estudio cinemático," pp. 1–9, 9 2019.
- [21] T. Iberall, G. Sukhatme, D. Beattie, and G. A. Bekey, "Control philosophy and simulation of a robotic hand as a model for prosthetic hands," pp. 824–831, 1993.
- [22] M. Vukobratović, V. Ćirić, D. Hristić, and J. Stepanenko, "Contribution to the study of anthropomorphic robots," *IFAC Proceedings Volumes*, vol. 5, pp. 88–96, 6 1972.
- [23] L. Biagiotti, F. Lotti, C. Melchiorri, and G. Vassura, "How far is the human hand? a review on anthropomorphic robotic end-effectors," 2002.

- [24] E. Martinez-Garcia, "Diseño y modelado de mano robótica antropomórfica de músculos artificiales," *Instituto de Ingeniería y Tecnología*, 2020.
- [25] Z. Li and S. Sastry, "Issues in dextrous robot hands," *Dextrous Robot Hands*, pp. 154–186, 1990.
- [26] A. M. Okamura, N. Smaby, and M. R. Cutkosky, "Overview of dexterous manipulation," *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 1, pp. 255–262, 4 2000.
- [27] G. A. BETANCOURT, E. G. Suárez, and J. F. Franco, "Reconocimiento de patrones de movimiento a partir de señales electromiográficas," *Scientia et Technica*, vol. 10, no. 26, pp. 53–58, 2004.
- [28] S. Ferguson and G. R. Dunlop, "Grasp recognition from myoelectric signals," in *Proceedings of the Australasian Conference on Robotics and Automation, Auckland, New Zealand*, vol. 1, 2002.
- [29] C. J. De Luca, "Surface electromyography: Detection and recording," *De/Sys Incorporated*, vol. 10, no. 2, pp. 1–10, 2002.
- [30] A. J. Young, L. J. Hargrove, and T. A. Kuiken, "The effects of electrode size and orientation on the sensitivity of myoelectric pattern recognition systems to electrode shift," *IEEE Transactions on Biomedical Engineering*, vol. 58, no. 9, pp. 2537–2544, 2011.
- [31] B. Hudgins, P. Parker, and R. N. Scott, "A new strategy for multifunction myoelectric control," *IEEE Transactions on Biomedical Engineering*, vol. 40, pp. 82–94, 1993.
- [32] H. A. Romo, J. C. Realpe, and P. E. Jojoa, "Análisis de señales emg superficiales y su aplicación en control de prótesis de mano," *Revista avances en sistemas e informática*, vol. 4, no. 1, pp. 127–136, 2007.
- [33] C. A. Quinayas, M. Galindo, J. Avendaño, M. Lucy, *et al.*, "Identificación de la flexión y extensión de una mano a partir de señales emg," *I+ T+ C-Investigación, Tecnología y Ciencia*, vol. 1, no. 4, pp. 43–46, 2010.

- [34] G. Ellis, "Filters in control systems," *Control System Design Guide*, pp. 165–183, 1 2012.
- [35] J. T. Park, H. Hwang, J.-s. Yun, and I.-y. Moon, "Study of html5 websocket for a multimedia communication," *International Journal of Multimedia & Ubiquitous Engineering*, vol. 9, no. 7, pp. 61–72, 2014.
- [36] S. Khruangsakun, S. Nuratch, and P. Boonpramuk, "Design and development of cyber physical system for real-time web-based visualization and control of robot arm," in *2020 5th International Conference on Control and Robotics Engineering (ICCRE)*, pp. 11–14, 2020.
- [37] B. Romero Rojas, "Una introducción a los modelos de machine learning," B.S. thesis, 2020.
- [38] E. M. Rojas, "Machine learning: análisis de lenguajes de programación y herramientas para desarrollo," *Revista Ibérica de Sistemas e Tecnologías de Informação*, no. E28, pp. 586–599, 2020.
- [39] K. T. Kim, S. Park, T. H. Lim, and S. J. Lee, "Upper-limb electromyogram classification of reaching-to-grasping tasks based on convolutional neural networks for control of a prosthetic hand," *Frontiers in Neuroscience*, vol. 15, 10 2021.
- [40] "Cyton specs — openbci documentation."
- [41] "Compatible software — openbci documentation."
- [42] A. A. Suzen, B. Duman, and B. Sen, "Benchmark analysis of jetson tx2, jetson nano and raspberry pi using deep-cnn," *HORA 2020 - 2nd International Congress on Human-Computer Interaction, Optimization and Robotic Applications, Proceedings*, 6 2020.
- [43] C. R. Romeva, "Métodos de evaluación de soluciones," 4 2002.
- [44] 3D4MEDICAL, "Complete anatomy," 2016.
- [45] J. Segen and S. Kumar, "Human-computer interaction using gesture recognition and 3d hand tracking," *IEEE International Conference on Image Processing*, vol. 3, pp. 188–192, 1998.

- [46] Sklearn, “Minmaxscaler.”
- [47] D. T. Mewett, H. Nazeran, and K. J. Reynolds, “Removing power line noise from recorded emg,” *Annual Reports of the Research Reactor Institute, Kyoto University*, vol. 3, pp. 2190–2193, 2001.
- [48] “tsfresh.feature_extraction package — tsfresh documentation,”
- [49] J. D. Nobel, H. Wang, and T. Baeck, “Explorative data analysis of time series based algorithm features of cma-es variants,” *GECCO 2021 - Proceedings of the 2021 Genetic and Evolutionary Computation Conference*, pp. 510–518, 6 2021.
- [50] DronProfesional, “Tutorial teórico práctico con servos y arduino.,” 2019.
- [51] M. Á. Castillo-Martínez, B. E. Carvajal-Gámez, and F. J. G. Funes, “Efectos en la resolución de servomotores con interfaz pwm por la generación de señales en microcontroladores.,” *Res. Comput. Sci.*, vol. 147, no. 7, pp. 89–98, 2018.
- [52] O. BCI, “Ganglion board..”
- [53] “Cyton data format — openbci documentation.”
- [54] W. J. Alvarado Díaz, “Implementación de una interfaz cerebro-máquina para el control de una silla de ruedas,” 2019.
- [55] P. P. Pesántez Pacheco and M. J. Romero Palacios, “Sistema de análisis de actividad cerebral eeg durante la ejecución de tareas cognitivas,” B.S. thesis, Universidad del Azuay, 2021.

APÉNDICES

APÉNDICE A

Datos técnicos de la placa Cyton Board

Entre las especificaciones técnicas más relevantes de la placa CB, se presentan las siguientes:

- Es compatible con electrodos activos y pasivos.
- Convertidor ADC de 24 Bits especializado para mediciones bioeléctricas con una tasa de muestreo de 250 Hz en cada canal. Procedente de Texas Instruments.
- Ganancia programable: 1, 2, 4, 6, 8, 12, 24
- Voltaje de operación: 3.3V
- Voltaje de operación analógica; $\pm 2.5V$
- Voltaje de entrada: 3.3-12V
- 5 pines GPIO.
- RFDuino de radio Bluetooth de bajo consumo.
- Acelerómetro: LIS3DH
- Ranura SD card

Su sistema incorporado de Bluetooth le permite comunicarse de forma inalámbrica con cualquier dispositivo móvil, tableta o PC. Sin embargo, OpenBCI recomienda utilizar el adaptador USB que es parte del conjunto de herramientas que proporciona la marca para poder alcanzar la alta tasa de muestreo que brinda la placa de adquisición de datos [9].

APÉNDICE B

RFDuino USB dongle

El protocolo de comunicación serial que mantiene el adaptador USB esta conformado por una cabecera, del conjunto de datos de los 8 canales ADC, el resultado medido por los 3 ejes del acelerómetro, y finaliza con el pié del protocolo o byte de parada [53]. Cada sección anteriormente nombrada está conformada por la siguiente estructura:

■ Cabecera

- Byte 1: 0xA0
- Byte 1: Número de muestra.

■ Datos EEG/ECG/EMG

- Bytes 3-5: Lectura del canal 1
- Bytes 6-8: Lectura del canal 2
- Bytes 9-11: Lectura del canal 3
- Bytes 12-14: Lectura del canal 4
- Bytes 15-17: Lectura del canal 5
- Bytes 18-20: Lectura del canal 6
- Bytes 21-23: Lectura del canal 7
- Bytes 24-26: Lectura del canal 8

■ Datos Auxiliares

- Bytes 27-32: 6 bytes de datos que se definen y analizan según el byte de parada.

■ Byte de parada

- Byte 33: 0xCX donde X va de 0-F en hexadecimal.

Las distintas configuraciones que pueden optar los 6 bytes auxiliares dependiendo del bit de parada, se pueden apreciar en la Figura B.1. Por lo general, se utiliza los bits de parada 0xC3 y 0xC5 para obtener las marcas de tiempo de la muestra realizada las cuales permiten realizar otros tipos de análisis. Con el byte de parada 0xC0 se otorga los valores de los 3 ejes del acelerómetro a los bytes 27-32 [53].

Stop Byte	Byte 27	Byte 28	Byte 29	Byte 30	Byte 31	Byte 32	Name
0xC0	AX1	AX0	AY1	AY0	AZ1	AZ0	Standard with accel
0xC1	UDF	UDF	UDF	UDF	UDF	UDF	Standard with raw aux
0xC2	UDF	UDF	UDF	UDF	UDF	UDF	User defined
0xC3	AC	AV	T3	T2	T1	T0	Time stamped set with accel
0xC4	AC	AV	T3	T2	T1	T0	Time stamped with accel
0xC5	UDF	UDF	T3	T2	T1	T0	Time stamped set with raw aux
0xC6	UDF	UDF	T3	T2	T1	T0	Time stamped with raw aux

Figura B.1 Configuraciones de los 6 bytes auxiliares según el byte de parada [53].

APÉNDICE C

Lab Streaming Layer (LSL).

La capa de transmisión de laboratorio o también conocido como Lab Streaming Layer (LSL), es un sistema para la recopilación unificada de series de tiempo de medición, como se muestra en la Figura C.1. Esta capa de transición facilita la comunicación vía streaming, también consta de una biblioteca central [54].

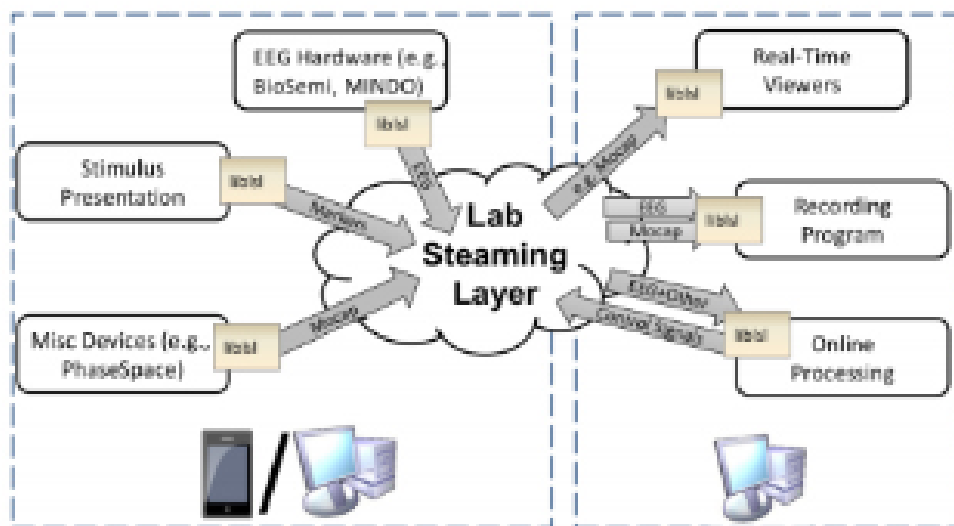


Figura C.1 Esquema del funcionamiento de LSL [55].

El sistema de recopilación se lo realiza a través de un conjunto de bibliotecas, la recopilación se realiza de distintos sistemas y nos facilita la vinculación de diferentes experimentos de investigación. La interfaz de Lab Recorder permite guardar todos los streams que están en la capa de transmisión como se muestra en la Figura C.1, el lab y su formato de guardado es de tipo Extend Disk Format (xdf) [55].

Esta biblioteca tiene ya integrada herramientas como son las aplicadas a experimentos de investigación, en los cuales se maneja la conexión de red, la sincronización y el acceso en tiempo real, opcionalmente la recopilación centralizada. La biblioteca de transporte principal es libsl y sus interfaces de lenguaje son (C, C++, Python, Java, MATLAB). La biblioteca es de uso personal y también de multiplataforma.

APÉNDICE D

Configuración de placa CB.

En el GUI se establecieron los siguientes widgets: time series para la visualización de la señal en tiempo real, networking para activar la comunicación por LSL, y cualquier otro opcional para la visualización de la señal como el FFT Plot que se aprecia en la Figura D.1. De igual forma, se estableció el filtro notch 60 [Hz] y butterworth a 15-50 [Hz]. Cabe recalcar que los datos transmitidos por LSL no contenían estos filtros, se les agregó solo para visualización.

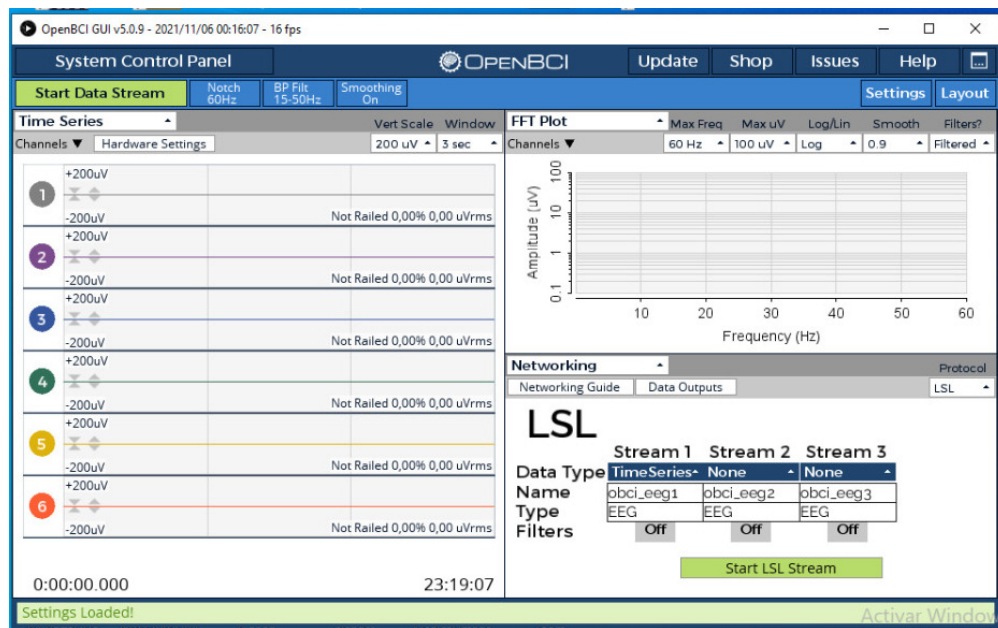


Figura D.1 Configuración de los widgets a utilizar en el Software OpenBCI GUI [9].

En la Figura D.2 se puede apreciar la configuración de los canales en el widget time series. En ella se estableció una ganancia de x24 y se desactivo los demás puertos no utilizados.



Figura D.2 Configuración de los canales en el widget Time Series del Software OpenBCI GUI [9].

APÉNDICE E

En esta sección se presentará todos los códigos de programación utilizados para procesamiento y control del presente proyecto.

E.1 Sistema de adquisición de datos

```
1 import cv2
2 from cv2 import data
3 from pylsl import StreamInlet, resolve_stream
4 import pandas as pd
5 import numpy as np
6 import time
7
8 class muestra:
9
10     def __init__(self, sujeto, inlet, Repeat):
11
12         self.Duration = 2.0
13         self.Repeat = Repeat
14         self.inlet = inlet
15
16         self.data = {'time': [], 'ch1': [], 'ch2': [], 'ch3': [], 'ch4': [], 'ch5': [], 'ch6': [], 'tag': [] }
17         self.interfaz ()
18
19         df = pd.DataFrame(self.data)
20         df.to_csv(sujeto+ '.csv', columns = ['time', 'ch1', 'ch2', 'ch3', 'ch4', 'ch5', 'ch6', 'tag'], mode='a')
21
22
23     def testLSLSamplingRates(self, movimiento):
24         print( "Testing Sampling Rates for {} seconds".format(self.Duration) )
25         start = time.time()
26
27         num_samples = 0
28         self.inlet.flush()
29
30         last_time = 0
31         start_time = 0
32
33         while time.time() <= (start + self.Duration):
34
35             # get a new sample (you can also omit the timestamp part
36             # if you're not interested in it)
37
38             sample, timestamp = self.inlet.pull_sample()
39
40             if timestamp:
41
42                 num_samples += 1
43
44                 self.data[ 'time' ].append(timestamp)
45                 self.data[ 'ch1' ].append(sample[0])
46                 self.data[ 'ch2' ].append(sample[1])
47                 self.data[ 'ch3' ].append(sample[2])
48                 self.data[ 'ch4' ].append(sample[3])
49                 self.data[ 'ch5' ].append(sample[4])
50                 self.data[ 'ch6' ].append(sample[5])
51                 self.data[ 'tag' ].append(movimiento)
52
```

```

53     if num.samples == 1 :
54         start_time = timestamp
55         last_time = timestamp
56
57         print(f"Numero de muestras {num.samples}")
58         print(last_time , " ***** " , start_time)
59         print(f"frecuencia de muestreo = {num.samples/(last_time-start_time)}")
60
61
62     def interfaz (self):
63
64         portada = cv2.imread("Imágenes/Portada.png")
65         cv2.imshow("Portada", portada)
66         if cv2.waitKey(0) & 0xFF == ord('S'):
67             cv2.destroyAllWindows()
68
69         cant_mov = [1, 2, 3, 4]
70
71         for i in range(self.Repeat): #forma aleatoria
72
73             for j in cant_mov:
74
75                 if j == 1:
76                     mov1 = cv2.imread("Imágenes/1.jpg")
77                     cv2.imshow("Movimiento_1", mov1)
78                     if cv2.waitKey(0) & 0xFF == ord('S'):
79                         self.testLSLSamplingRates(1)
80                         cv2.destroyAllWindows()
81
82                 if j == 2:
83                     mov2 = cv2.imread("Imágenes/2.jpg")
84                     cv2.imshow("Movimiento_2", mov2)
85                     if cv2.waitKey(0) & 0xFF == ord('S'):
86                         self.testLSLSamplingRates(2)
87                         cv2.destroyAllWindows()
88
89                 if j == 3:
90                     mov3 = cv2.imread("Imágenes/3.jpg")
91                     cv2.imshow("Movimiento_3", mov3)
92                     if cv2.waitKey(0) & 0xFF == ord('S'):
93                         self.testLSLSamplingRates(3)
94                         cv2.destroyAllWindows()
95
96                 if j == 4:
97                     mov4 = cv2.imread("Imágenes/4.jpg")
98                     cv2.imshow("Movimiento_4", mov4)
99                     if cv2.waitKey(0) & 0xFF == ord('S'):
100                         self.testLSLSamplingRates(4)
101                         cv2.destroyAllWindows()
102
103             np.random.shuffle(cant_mov)
104             print("ESTAMOS EN" , i)
105
106
107     def main ():
108
109         Sujeto = input('Ingresar identificación del sujeto: ')
110
111         Repeat = int(input('Ingrese el numero de repeticiones: '))
112
113         print("looking for an EEG stream...")
114         streams = resolve_stream('type', 'EEG')
115         inlet = StreamInlet(streams[0])
116         print("EMG stream found!")
117
118         muestra(Sujeto, inlet, Repeat)

```



```

119
120 if __name__ == "__main__":
121     main()

```

Código E.1.1 Programa para adquisición de datos EMG.

E.2 Procesamiento de datos

```

1
2 #librerias
3 import os
4 from tsfresh import extract_features
5 from tsfresh.feature_selection.relevance import calculate_relevance_table
6 from tsfresh.utilities.dataframe_functions import impute
7 import pandas as pd
8 import matplotlib.pyplot as plt
9 import numpy as np
10 from brainflow.board_shim import BoardShim, BrainFlowInputParams, LogLevels, BoardIds
11 from brainflow.data_filter import DataFilter, FilterTypes, AggOperations, NoiseTypes
12 from tqdm.auto import tqdm, trange
13
14 from sklearn.preprocessing import MinMaxScaler
15 from sklearn.utils import shuffle
16 from sklearn.tree import DecisionTreeClassifier
17 from sklearn.metrics import mean_squared_error

```

Código E.2.1 Librerías utilizadas.

```

1
2 def oneForAll_tags(DFrame, ID_number ):
3     df = DFrame
4     df = df.filter(['time', 'ch1', 'ch2', 'ch3', 'ch4', 'ch5', 'ch6', 'tag'])
5
6     tag1 = pd.DataFrame()
7     tag2 = pd.DataFrame()
8     tag3 = pd.DataFrame()
9     tag4 = pd.DataFrame()
10
11     groups = [0]
12
13     for i in trange(df.shape[0]-1, desc="Extracting groups"):
14         if (df.time.iloc[i+1] - df.time.iloc[i]) > 1.0:
15             groups.append(i+1)
16
17     groups.append(df.shape[0])
18
19     print(f'Existen {len(groups)-1} repeticiones')
20
21     Secuencia = np.ones([4,1])
22
23     for j in trange(len(groups)-1, desc="Cluster by tags"):
24
25         df_aux=df[groups[j]:groups[j+1]]
26
27         df_aux = impute(df_aux)
28
29         df_aux = df_aux.assign(ID = ID_number)
30         df_aux['time'] = (df_aux.time - df_aux.time.iloc[0])*1000
31         df_aux.time = df_aux.time.astype(int)
32
33         channels = df_aux.iloc[:, 1:7]

```

```

34 channels = channels - channels.mean()
35 scaler = MinMaxScaler()
36 df_aux.iloc[:, 1:7] = scaler.fit_transform(channels)
37
38 if df_aux.tag.iloc[j] == 1:
39     df_aux = df_aux.assign(Repeticion = int(Secuencia[0]))
40     tag1 = tag1.append(df_aux, ignore_index= True)
41     Secuencia [0] += 1
42
43 if df_aux.tag.iloc[j] == 2:
44     df_aux = df_aux.assign(Repeticion = int(Secuencia[1]))
45     tag2 = tag2.append(df_aux, ignore_index= True)
46     Secuencia [1] += 1
47
48 if df_aux.tag.iloc[j] == 3:
49     df_aux = df_aux.assign(Repeticion = int(Secuencia[2]))
50     tag3 = tag3.append(df_aux, ignore_index= True)
51     Secuencia [2] += 1
52
53 if df_aux.tag.iloc[j] == 4:
54     df_aux = df_aux.assign(Repeticion = int(Secuencia[3]))
55     tag4 = tag4.append(df_aux, ignore_index= True)
56     Secuencia [3] += 1
57
58
59
60 #print(Secuencia)
61 return tag1, tag2, tag3, tag4

```

Código E.2.2 Función para normalización y división de datos por tipo de movimiento.

```

1
2 def graficas_tag(DFrame, filter, name):
3     df = DFrame
4
5     id = df.ID.iloc[0]
6     mov = df.tag.iloc[0]
7     rept = df.Repeticion.iloc[0]
8
9     fig, axs = plt.subplots(6, figsize=(13,10), sharex=True, sharey=True)
10    fig.suptitle('Graficas canales vs tiempo del movimiento '+ str(mov))
11
12    labels = ['ch1', 'ch2', 'ch3', 'ch4', 'ch5', 'ch6']
13
14    for i, ax in enumerate(axs):
15        axs[i].plot(df["time"], df[labels[i]], label = labels[i])
16        axs[i].legend(loc="upper right")
17
18    path = 'Figuras_mov/' + name
19    os.makedirs(path, exist_ok=True)
20
21    if filter:
22        plt.savefig(path + '/Filter_mov_' + str(mov) + ' rep_' + str(rept) + ' ID_' + str(id) + '.png')
23    else:
24        plt.savefig(path + '/mov_' + str(mov) + ' rep_' + str(rept) + ' ID_' + str(id) + '.png')
25
26    plt.close()
27    #plt.show()

```

Código E.2.3 Función para graficar y guardar datos puros y normalizados.

```

1
2 def filter_process_df (DFrame, name):
3
4     df = DFrame
5     # de ser necesario despejarlo
6     #df = df.filter(['time', 'ch1','ch2','ch3', 'ch4','ch5', 'ch6', 'tag', 'ID', 'Repeticion'])
7
8     # escalamos respecto a la media por cada grupo independientemente
9     scaler = MinMaxScaler()# StandardScaler()
10    df.iloc[:, 1:7] = scaler.fit_transform(df.iloc[:, 1:7] )
11
12    df_filter = pd.DataFrame()
13
14    groups = []
15
16    #extraccion de repeticiones sin importar el id
17    for i in range(0,df.shape[0]):
18        if df.Repeticion.iloc[i]!=df.Repeticion.iloc[i-1]:
19            groups.append(i)
20
21    groups.append(df.shape[0])
22
23    for j in range(0,len(groups)-1):
24        lim_bajo = groups[j]
25        lim_alto = groups[j+1]
26
27        df_aux=df[lim_bajo:lim_alto]
28
29        graficas_tag(df_aux, filter= False, name= name)
30
31        labels = ['ch1', 'ch2', 'ch3', 'ch4', 'ch5', 'ch6']
32
33        for k in range(len(labels)):
34            DataFilter.remove_environmental_noise(df_aux[labels[k]].values, 250, NoiseTypes.SIXTY.value)
35            DataFilter.perform_bandpass(df_aux[labels[k]].values, 250, 55.0, 90.0, 5, FilterTypes.BUTTERWORTH.value, 0)
36
37        graficas_tag(df_aux, filter= True, name = name)
38
39        df_filter = df_filter.append(df_aux, ignore_index=True)
40
41    return df_filter

```

Código E.2.4 Función para filtrar datos.

```

1
2 def AllforOne_filter (list_dfNames, id):
3
4     All_data_mov = pd.DataFrame()
5
6     for i in trange(len(list_dfNames), desc= "Aplicando filtros"):
7
8         df = pd.read_csv(list_dfNames[i]+ ".csv", index_col=0)
9
10        tag1, tag2, tag3, tag4 = oneForAll_tags(df, id)
11
12        df_filter_mov1 = filter_process_df(tag1, list_dfNames[i] + '_' + str(id) )
13        df_filter_mov2 = filter_process_df(tag2, list_dfNames[i] + '_' + str(id) )
14        df_filter_mov3 = filter_process_df(tag3, list_dfNames[i] + '_' + str(id) )
15        df_filter_mov4 = filter_process_df(tag4, list_dfNames[i] + '_' + str(id) )
16
17        df_filter_mov1.to_csv(list_dfNames[i] + "_dfFilter_mov1.csv")
18        df_filter_mov2.to_csv(list_dfNames[i] + "_dfFilter_mov2.csv")
19        df_filter_mov3.to_csv(list_dfNames[i] + "_dfFilter_mov3.csv")

```

```

20     df_filter_mov4.to_csv(list_dfNames[i] + "_dfFilter_mov4.csv")
21
22     id +=1
23
24
25     All_data_mov = pd.DataFrame()
26
27     for i in range(1,5):
28         movimiento = '_dfFilter_mov'+str(i)+'.csv'
29         data_mov = pd.DataFrame()
30
31         for j in range(len(list_dfNames)):
32             rf = pd.read_csv(list_dfNames[j] +movimiento, index_col=0)
33             rf["Repeticion"] = (np.array(rf.ID.astype(int))-1) *20 + np.array(rf.Repeticion.astype(int))
34
35             data_mov = data_mov.append(rf, ignore_index= True)
36
37         data_mov.to_csv("Data" + movimiento)
38
39         All_data_mov = All_data_mov.append(data_mov, ignore_index= True)
40
41     All_data_mov.to_csv("Data_filter_allMove.csv")
42
43     return All_data_mov

```

Código E.2.5 Función principal para normalizar, graficar y separar los datos por movimiento.

```

1
2 relevance_table = calculate_relevance_table(data_features, tag, 'classification', True, 4)
3 print(f'Tabla de relevancia con {relevance_table.shape[0]} features')
4 relevance_table = relevance_table[relevance_table.relevant]
5 relevance_table.to_csv("relevance_table_"+name+"_allMov.csv")
6 print(f'Tabla de relevancia reducida con {relevance_table.shape[0]} features')
7
8 data_relevant = data_features[relevance_table.feature]

```

Código E.2.6 Filtración de características relevantes.

```

1
2 # feature importance based on Tree Based Classifiers
3 data_rv = data_relevant
4 model = ExtraTreesClassifier()
5 model.fit(data_rv, tag)
6
7 feat_importances = pd.Series(model.feature_importances_, index=data_rv.columns)
8 feat_importances.nlargest(20).plot(kind='barh', figsize=(8,8))
9 plt.title("N relevant features")
10 plt.show()
11
12 feat_imp = feat_importances.nlargest(20)
13 n_data_rv = data_relevant[feat_imp.index]
14 n_data_rv

```

Código E.2.7 Selección de características relevantes.

E.3 Análisis de modelos de clasificación ML.

```
1
2 def train_classifiers (data_train , tag_train):
3     print('*** Resultados de entrenamiento *** \n')
4
5     #Decision tree
6     dt = DecisionTreeClassifier().fit(data_train , tag_train)
7     dt_score = dt.score(data_train , tag_train)
8     print(f'Accuracy using Decision tree: {dt_score*100:.2f}')
9
10    #svm Polynomial
11    poly = svm.SVC(kernel='poly' , degree=8, C=0.001).fit(data_train , tag_train)
12    poly_score = poly.score(data_train , tag_train)
13    print(f'Accuracy using Polynominal: {poly_score*100:.2f}')
14
15    # KNN
16    knn=KNeighborsClassifier(n_neighbors= 20).fit(data_train , tag_train)
17    knn.score = knn.score(data_train , tag_train)
18    print(f'Accuracy using KNN: {knn.score*100:.2f}')
19
20    # Random Forest Classifier
21    rf=RandomForestClassifier(random_state=21).fit(data_train , tag_train)
22    rf_score = rf.score(data_train , tag_train)
23    print(f'Accuracy using Random Forest: {rf_score*100:.2f}')
24
25    # Multi-layer Perceptron classifier
26    mlpc = MLPClassifier((250,150,50) , max_iter=400,activation = 'relu' ,solver='adam' ,
27        random_state=1)
28    mlpc_score = mlpc.score(data_train , tag_train)
29    print(f'Accuracy using MLPC1: {mlpc_score*100:.2f} ')
30
31    # Gaussian Process Classifier
32    #kernel = 1.0 * RBF(1.0) is used
33    gpc = GaussianProcessClassifier(random_state=10).fit(data_train , tag_train)
34    gpc_score = gpc.score(data_train , tag_train)
35    print(f'Accuracy using Gaussian Process: {gpc_score*100:.2f}')
36
37    # Ridge Classifier
38    rc = RidgeClassifier().fit(data_train , tag_train)
39    rc_score = rc.score(data_train , tag_train)
40    print(f'Accuracy using Ridge: {rc_score*100:.2f}')
41
42    # Ridge Classifier
43    sgd = SGDClassifier().fit(data_train , tag_train)
44    sgd_score = sgd.score(data_train , tag_train)
45    print(f'Accuracy using SGD: {sgd_score*100:.2f}')
46
47    return dt, poly, knn, rf, mlpc, gpc, rc, sgd
```

```

48 def test_classifiers (data_test, tag_test, DecisionT, POLY, KNN, RF, MLPC, GPC, RC, SGD):
49     Targets = ['tag_1', 'tag_2', 'tag_3', 'tag_4']
50     print('\n*** Resultados de testeo *** \n')
51
52     #Decision tree
53     dt_pred = DecisionT.predict(data_test)
54     dt_accuracy = accuracy_score(tag_test, dt_pred)
55     print(f'Accuracy using Decision Tree: {dt_accuracy*100:.2f}')
56     #svm Polynomial
57     poly_pred = POLY.predict(data_test)
58     poly_accuracy = accuracy_score(tag_test, poly_pred)
59     print(f'Accuracy using Polynomial Kernel: {poly_accuracy*100:.2f}')
60     # KNN
61     knn_pred = KNN.predict(data_test)
62     knn_accuracy = accuracy_score(tag_test, knn_pred)
63     print(f'Accuracy using KNN: {knn_accuracy*100:.2f}')
64     # Random Forest Classifier
65     rf_pred = RF.predict(data_test)
66     rf_accuracy = accuracy_score(tag_test, rf_pred)
67     print(f'Accuracy using RF: {rf_accuracy*100:.2f}')
68     # Multi-layer Perceptron classifier
69     mlpc_pred = MLPC.predict(data_test)
70     mlpc_accuracy = accuracy_score(tag_test, mlpc_pred)
71     print(f'Accuracy using MLPC1: {mlpc_accuracy*100:.2f}')
72     # Gaussian Process Classifier
73     gpc_pred = GPC.predict(data_test)
74     gpc_accuracy = accuracy_score(tag_test, gpc_pred)
75     print(f'Accuracy using GPC: {gpc_accuracy*100:.2f}')
76     # Ridge Classifier
77     rc_pred = RC.predict(data_test)
78     rc_accuracy = accuracy_score(tag_test, rc_pred)
79     print(f'Accuracy using RC: {rc_accuracy*100:.2f}')
80     # stochastic gradient descent Classifier
81     sgd_pred = SGD.predict(data_test)
82     sgd_accuracy = accuracy_score(tag_test, sgd_pred)
83     print(f'Accuracy using SGD: {sgd_accuracy*100:.2f}')
84
85 X_train, X_test, Y_train, Y_test = train_test_split(n_data_rv, tag, test_size=0.2, random_state
    =300)
86
87 dt, poly, knn, rf, mlpc, gpc, rc, sgd = train_classifiers (X_train, Y_train)
88
89 print('*'*80)
90
91 test_classifiers(X_test, Y_test, dt, poly, knn, rf, mlpc, gpc, rc, sgd)

```

Código E.3.1 Funciones de entrenamiento y testeo de modelos de clasificación ML.

E.4 Programación de ESP32

```
1
2 #include <Arduino.h>
3 #include <WiFi.h>
4 #include <HTTPClient.h>
5 #include <Wire.h>
6 #include <Adafruit_PWMServoDriver.h>
7
8 //Configuracion de conexion Wifi
9 const char* ssid = "SSID";
10 const char* password = "password";
11 ///Configuracion de ip server
12 ip = "Colocar la direccion ip del server"
13 const char* serverName = ip+"/read";
14
15 // the following variables are unsigned longs because the time, measured in
16 // milliseconds, will quickly become a bigger number than can be stored in an int.
17 unsigned long lastTime = 0;
18 // Set timer to 2.5 seconds (2500)
19 unsigned long timerDelay = 2500;
20
21 // Configuracion movimientos
22 String movimiento;
23 int t = 200; // tiempo entre movimiento de dedos
24 //int t2 = 4000; tiempo de retorno a movimiento base de la mano
25
26
27 // Configuracion de la placa pca9685
28 Adafruit_PWMServoDriver servos = Adafruit_PWMServoDriver(0x40);
29 //pulsos = ms(frecuencia/1000)*4096
30 unsigned int pos_0 = 184; // Calculo de pulsos para 0 grados
31 unsigned int pos_180 = 574; //Calculo de pulsos para 180 grados
32
33 uint8_t mov = 0;
34
35
36 void setup() {
37
38     Serial.begin(115200);
39     WiFi.begin(ssid, password);
40
41     Serial.print("Connecting to WiFi");
42
43     while (WiFi.status() != WL_CONNECTED)
44     {
45         Serial.print(".");
46         delay(500);
47     }
48
```

```

49 Serial.print("\nConnected to the WiFi network");
50 Serial.print("IP address: ");
51 Serial.print(WiFi.localIP());
52
53 servos.begin();
54 servos.setPWMPFreq(50); //Frecuencia PWM de 50Hz
55 }
56
57 String httpGETRequest(const char* serverName) {
58     WiFiClient client;
59     HTTPClient http;
60
61     // Your IP address with path or Domain name with URL path
62     http.begin(client, serverName);
63
64     // Send HTTP GET request
65     int httpResponseCode = http.GET();
66
67     String payload = "{}";
68
69     if (httpResponseCode > 0) {
70         Serial.print("HTTP Response code: ");
71         Serial.println(httpResponseCode);
72         payload = http.getString();
73     }
74     else {
75         Serial.print("Error code: ");
76         Serial.println(httpResponseCode);
77     }
78     // Free resources
79     http.end();
80
81     return payload;
82 }
83
84 void movServo(uint8_t n_servo, int angulo) {
85     int duty;
86     duty = map(angulo, 0, 180, pos_0, pos_180);
87     servos.setPWM(n_servo, 0, duty);
88 }
89
90 // (dedo, canal)
91 // (indice, 8) (Medio, 5) (Anular, 3) (Me ique, 0) (pulgar_1, 12) (pulgar_1, 15)
92
93 void movimiento_0 ()
94 {
95     movServo(12, 100);
96     delay(t);
97     movServo(15, 25);

```



```
98 delay(t);
99 movServo(8, 170);
100 delay(t);
101 movServo(5, 170);
102 delay(t);
103 movServo(3, 170);
104 delay(t);
105 movServo(0, 170);
106 delay(t);
107 }
108
109 void movimiento_1 ()
110 {
111     movServo(0, 60);
112     delay(t);
113     movServo(3, 75);
114     delay(t);
115     movServo(5, 100);
116     delay(t);
117     movServo(15, 110);
118     delay(t);
119     movServo(12, 20);
120     delay(t);
121 }
122
123 void movimiento_2 ()
124 {
125     movServo(0, 60);
126     delay(t);
127     movServo(3, 75);
128     delay(t);
129     movServo(5, 100);
130     delay(t);
131     movServo(8, 80);
132     delay(t);
133     movServo(15, 110);
134     delay(t);
135     movServo(12, 100);
136     delay(t);
137 }
138
139 void movimiento_3 ()
140 {
141     movServo(15, 110);
142     delay(t);
143     movServo(12, 30);
144     delay(t);
145     movServo(0, 60);
146     delay(t);
```

```

147  movServo(3, 75);
148  delay(t);
149  movServo(5, 120);
150  delay(t);
151  movServo(8, 105);
152  delay(t);
153 }
154
155 void movimiento_4 ()
156 {
157  movServo(8, 170);
158  delay(t);
159  movServo(5, 170);
160  delay(t);
161  movServo(3, 75);
162  delay(t);
163  movServo(0, 60);
164  delay(t);
165  movServo(15, 110);
166  delay(t);
167  movServo(12, 30);
168  delay(t);
169 }
170
171
172 void loop() {
173  //read an HTTP GET request every 5 seconds
174  if ((millis() - lastTime) > timerDelay) {
175    //Check WiFi connection status
176    if (WiFi.status() == WL_CONNECTED) {
177
178      movimiento = httpGETRequest(serverName);
179      Serial.println(movimiento);
180
181      if (movimiento.toInt() != mov) {
182
183        switch (movimiento.toInt()) {
184          case 1:
185            movimiento_1();
186            //delay(t2);
187            //movimiento_0();
188            //delay(t2);
189            Serial.println("movimiento 1");
190            break;
191
192          case 2:
193            movimiento_2();
194            //delay(t2);
195            //movimiento_0();

```

```

196     //delay(t2);
197     Serial.println("movimiento 2");
198     break;
199
200     case 3:
201         movimiento_3();
202         //delay(t2);
203         //movimiento_0();
204         //delay(t2);
205         Serial.println("movimiento 3");
206         break;
207
208     case 4:
209         movimiento_4();
210         //delay(t2);
211         //movimiento_0();
212         //delay(t2);
213         Serial.println("movimiento 4");
214         break;
215
216     default:
217         movimiento_0();
218         //delay(t2);
219         Serial.println("movimiento 0");
220         break;
221 };
222
223     mov = movimiento.toInt();
224 }
225
226 }
227 else {
228     Serial.println("WiFi Disconnected");
229 }
230     lastTime = millis();
231 }
232
233 }

```

Código E.4.1 Sistema de control de servomotores.

```

1
2 from flask import Flask, request
3 import os
4 import json
5
6
7 app = Flask(__name__)
8

```

```

9 @app.route('/')
10 def home():
11     return 'HELLO WORLD'
12
13 @app.route('/_mov')
14 def movimiento():
15     state = request.args.get("movimiento")
16     s = {"movimiento": state}
17
18     fname = os.path.join(app.static_folder, "movimiento.json")
19
20     with open(fname, 'w') as outfile:
21         json.dump(s, outfile)
22
23     return 'success'
24
25 @app.route('/read')
26 def readJson():
27     fname = os.path.join(app.static_folder, "movimiento.json")
28
29     with open(fname, 'r') as openfile:
30         json_obj = json.load(openfile)
31
32     return json_obj['movimiento']
33
34 if __name__ == '__main__':
35     app.run(host='IP', debug = True, port= 8080)

```

Código E.4.2 Rest API.

APÉNDICE F




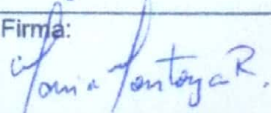
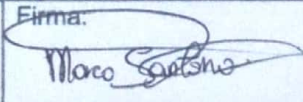
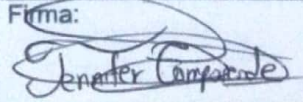
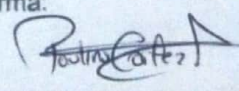
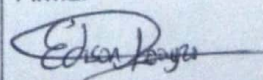
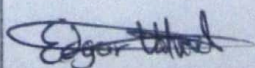

Se presenta la carta de autorización que los participantes firmaron para dar su consentimiento de uso de datos mioeléctricos con fines investigativos y académicos.

Carta de aceptación para el uso de datos personales.

Guayaquil, noviembre 8 del 2021

A quien interese,

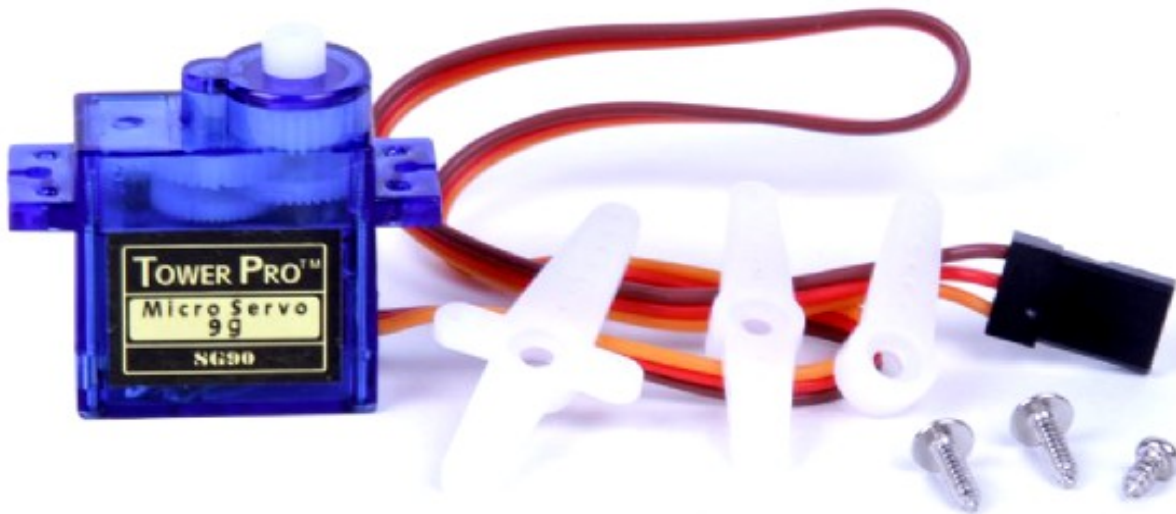
Mediante mi firma del presente documento otorgo de manera expresa, mi consentimiento y autorización para que los estudiantes Mauricio Valarezo y Bolívar Núñez, realicen la toma de mis datos mioeléctricos personales. Con la finalidad de que estos sean procesados, analizados y utilizamos con fines investigativos y académicos.

Voluntario 1	Firma: 
Voluntario 2	Firma: 
Voluntario 3	Firma: 
Voluntario 4	Firma: 
Voluntario 5	Firma: 
Voluntario 6	Firma: 
Voluntario 7	Firma: 
Voluntario 8	Firma: 
Voluntario 9	Firma: 
Voluntario 10	Firma: 

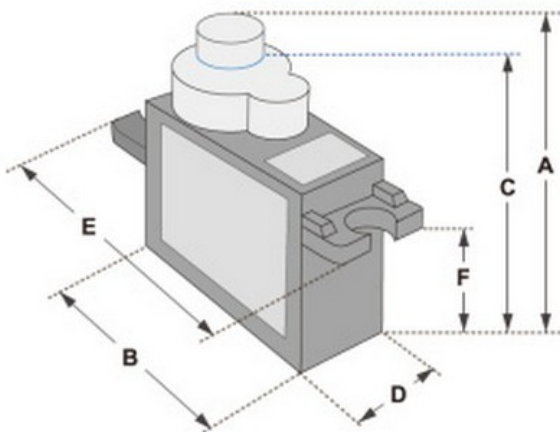
APÉNDICE G

SERVO MOTOR SG90

DATA SHEET

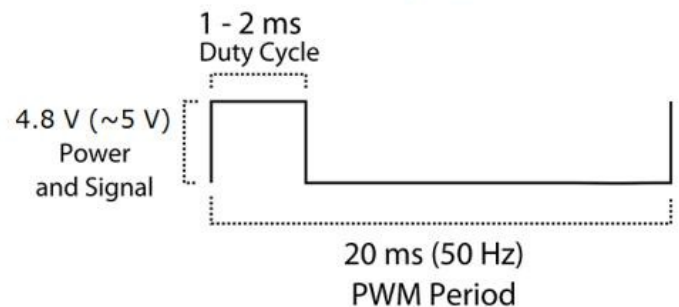
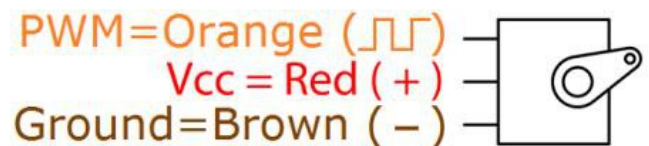


Tiny and lightweight with high output power. Servo can rotate approximately 180 degrees (90 in each direction), and works just like the standard kinds but smaller. You can use any servo code, hardware or library to control these servos. Good for beginners who want to make stuff move without building a motor controller with feedback & gear box, especially since it will fit in small places. It comes with a 3 horns (arms) and hardware.



Dimensions & Specifications
A (mm) : 32
B (mm) : 23
C (mm) : 28.5
D (mm) : 12
E (mm) : 32
F (mm) : 19.5
Speed (sec) : 0.1
Torque (kg-cm) : 2.5
Weight (g) : 14.7
Voltage : 4.8 - 6

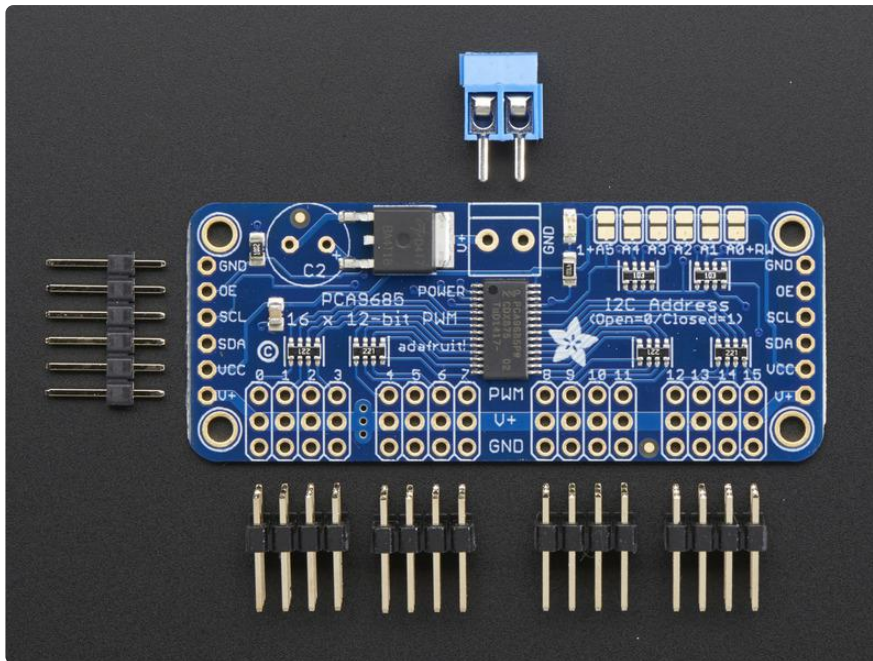
Position "0" (1.5 ms pulse) is middle, "90" (~2ms pulse) is middle, is all the way to the right, "-90" (~1ms pulse) is all the way to the left.



APÉNDICE H

Datasheet del driver de control PCA9685

Pinouts



There are two sets of control input pins on either side. Both sides of the pins are identical! Use whichever side you like, you can also easily chain by connecting up two side-by-side

Power Pins

- GND - This is the power and signal ground pin, must be connected
- VCC - This is the logic power pin, connect this to the logic level you want to use for the PCA9685 output, should be 3 - 5V max! It's also used for the 10K pullups on SCL/SDA so unless you have your own pullups, have it match the microcontroller's logic level too!
- V+ - This is an optional power pin that will supply distributed power to the servos. If you are not using for servos you can leave disconnected. It is not used at all by the chip. You can also inject power from the 2-pin terminal block at the top of the board. You should provide 5-6VDC if you are using servos. If you have to, you can go higher to 12VDC, but if you mess up and connect VCC to V+ you could damage your board!

Control Pins

- SCL - I2C clock pin, connect to your microcontrollers I2C clock line. Can use 3V or 5V logic, and has a weak pullup to VCC

- SDA - I2C data pin, connect to your microcontrollers I2C data line. Can use 3V or 5V logic, and has a weak pullup to VCC
- OE - Output enable. Can be used to quickly disable all outputs. When this pin is low all pins are enabled. When the pin is high the outputs are disabled. Pulled low by default so it's an optional pin!

Output Ports

There are 16 output ports. Each port has 3 pins: V+, GND and the PWM output. Each PWM runs completely independently but they must all have the same PWM frequency. That is, for LEDs you probably want 1.0 KHz but servos need 60 Hz - so you cannot use half for LEDs @ 1.0 KHz and half @ 60 Hz.

They're set up for servos but you can use them for LEDs! Max current per pin is 25mA.

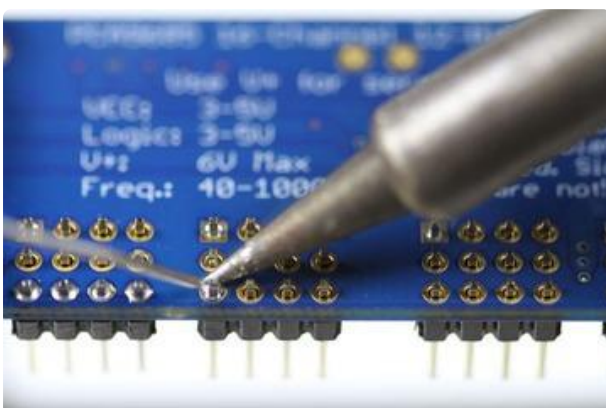
There are 220 ohm resistors in series with all PWM Pins and the output logic is the same as VCC so keep that in mind if using LEDs.

Assembly



Install the Servo Headers

Install 4 3x4 pin male headers into the marked positions along the edge of the board.



Solder all pins

There are a lot of them!

APÉNDICE I

Datasheet de fuente para protoboard

Fuente para protoboard 3.3v o 5V, 700ma

(Versión 30-9-18)

Este modulo posee salidas de 3.3V y 5V, posee un regulador para cada voltaje, Estos voltajes se seleccionan con jumpers.

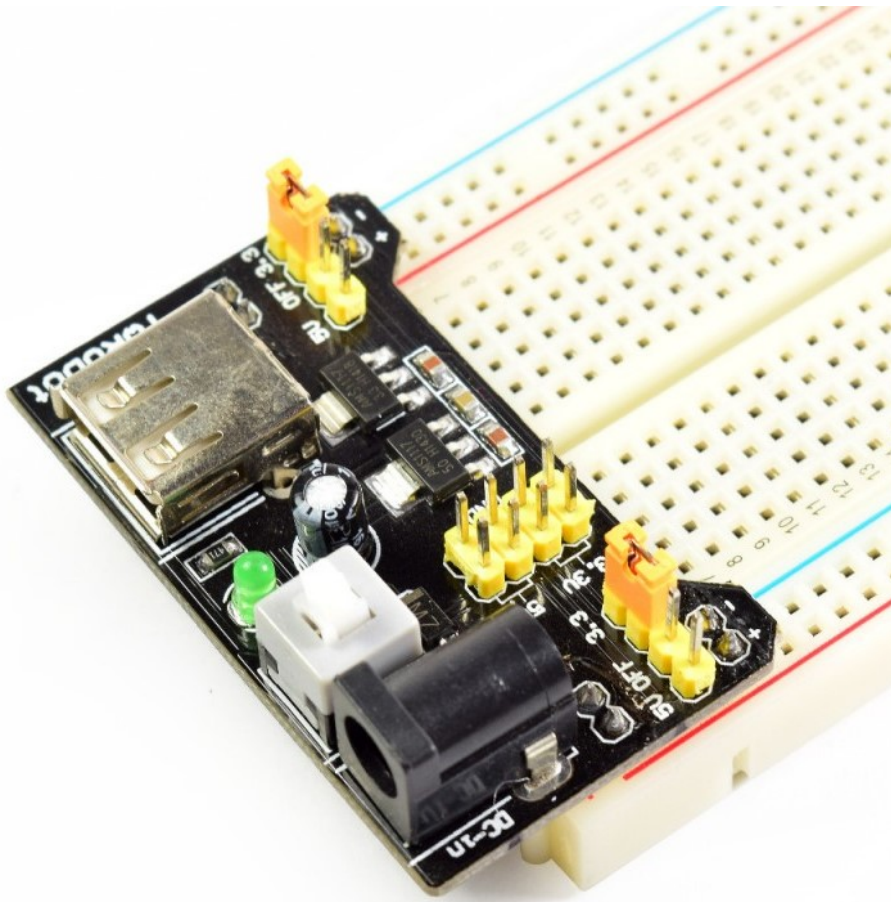
La entrada de voltaje al módulo puede ser por el conector Jack o a través del conector USB (*no recomendable, la corriente se limita a la que puede entregar el puerto USB de la PC*). Si se usa el conector Jack, el conector USB se puede utilizar como salida de 5V.

Permite alimentar un protoboard mediante plug a un transformador de pared hasta 12v. Proporciona dos salidas independientes que son seleccionables mediante jumpers y permiten suministrar 5V o 3.3V. Cuenta además con salida de 5V por conector USB.

Práctica fuente que simplifica la alimentación de la protoboard y circuitos electrónicos especialmente los de carácter digital, ideal con cualquier versión de Arduino y microcontroladores PIC. Cada lado de la fuente tiene un jumper de encendido / apagado y de selección de voltaje.

ESPECIFICACIONES TÉCNICAS

- Voltaje de entrada: 6.5V a 12V
- Voltaje de salida 1: 3.3V o 5V (seleccionable por jumper)
- Voltaje de salida 2: 3.3V o 5V (seleccionable por jumper)
- Salida de tensión USB: 5V
- Corriente máxima de salida: 700 mA
- LED indicador de encendido
- Botón de encendido / apagado



APÉNDICE J

Datasheet de consumo de poder del ESP32

- Ping
- Bluetooth Low Energy
 - Advertising
 - Scanning
 - Simultaneous advertising and scanning
 - Multiple connections
 - Asynchronous data reception and transmission
 - Adaptive Frequency Hopping and Channel assessment
 - Connection parameter update
 - Data Length Extension
 - Link Layer Encryption
 - LE Ping

3.7 RTC and Low-Power Management

With the use of advanced power-management technologies, ESP32 can switch between different power modes.

- Power modes
 - **Active mode:** The chip radio is powered on. The chip can receive, transmit, or listen.
 - **Modem-sleep mode:** The CPU is operational and the clock is configurable. The Wi-Fi/Bluetooth baseband and radio are disabled.
 - **Light-sleep mode:** The CPU is paused. The RTC memory and RTC peripherals, as well as the ULP coprocessor are running. Any wake-up events (MAC, host, RTC timer, or external interrupts) will wake up the chip.
 - **Deep-sleep mode:** Only the RTC memory and RTC peripherals are powered on. Wi-Fi and Bluetooth connection data are stored in the RTC memory. The ULP coprocessor is functional.
 - **Hibernation mode:** The internal 8 MHz oscillator and ULP coprocessor are disabled. The RTC recovery memory is powered down. Only one RTC timer on the slow clock and certain RTC GPIOs are active. The RTC timer or the RTC GPIOs can wake up the chip from the Hibernation mode.

Table 6: Power Consumption by Power Modes

Power mode	Description		Power consumption
Active (RF working)	Wi-Fi Tx packet		Please refer to Table 15 for details.
	Wi-Fi/BT Tx packet		
	Wi-Fi/BT Rx and listening		
Modem-sleep	The CPU is powered on.	240 MHz *	Dual-core chip(s) 30 mA ~ 68 mA
			Single-core chip(s) N/A
		160 MHz *	Dual-core chip(s) 27 mA ~ 44 mA
			Single-core chip(s) 27 mA ~ 34 mA

Power mode	Description		Power consumption
	Normal speed: 80 MHz	Dual-core chip(s)	20 mA ~ 31 mA
		Single-core chip(s)	20 mA ~ 25 mA
Light-sleep	-		0.8 mA
Deep-sleep	The ULP coprocessor is powered on.		150 μ A
	ULP sensor-monitored pattern		100 μ A @1% duty
	RTC timer + RTC memory		10 μ A
Hibernation	RTC timer only		5 μ A
Power off	CHIP_PU is set to low level, the chip is powered off.		1 μ A

Note:

- * Among the ESP32 series of SoCs, ESP32-D0WD-V3, ESP32-U4WDH, ESP32-D0WD ([NRND](#)), ESP32-D0WDQ6 ([NRND](#)), and ESP32-D0WDQ6-V3 ([NRND](#)) have a maximum CPU frequency of 240 MHz, ESP32-S0WD has a maximum CPU frequency of 160 MHz.
- When Wi-Fi is enabled, the chip switches between Active and Modem-sleep modes. Therefore, power consumption changes accordingly.
- In Modem-sleep mode, the CPU frequency changes automatically. The frequency depends on the CPU load and the peripherals used.
- During Deep-sleep, when the ULP coprocessor is powered on, peripherals such as GPIO and RTC I2C are able to operate.
- When the system works in the ULP sensor-monitored pattern, the ULP coprocessor works with the ULP sensor periodically and the ADC works with a duty cycle of 1%, so the power consumption is 100 μ A.

Test Item	Test Condition	Test Standard
TCT (Temperature Cycling Test)	-65 °C / 150 °C, 500 cycles	JESD22-A104
Autoclave Test	121 °C, 100% RH, 96 hours	JESD22-A102
uHAST (Highly Accelerated Stress Test, unbiased)	130 °C, 85% RH, 96 hours	JESD22-A118
HTSL (High Temperature Storage Life)	150 °C, 1000 hours	JESD22-A103

1. JEDEC document JEP155 states that 500 V HBM allows safe manufacturing with a standard ESD control process.
2. JEDEC document JEP157 states that 250 V CDM allows safe manufacturing with a standard ESD control process.

5.5 RF Power-Consumption Specifications

The power consumption measurements are taken with a 3.3 V supply at 25 °C of ambient temperature at the RF port. All transmitters' measurements are based on a 50% duty cycle.

Table 15: RF Power-Consumption Specifications

Mode	Min	Typ	Max	Unit
Transmit 802.11b, DSSS 1 Mbps, POUT = +19.5 dBm	-	240	-	mA
Transmit 802.11g, OFDM 54 Mbps, POUT = +16 dBm	-	190	-	mA
Transmit 802.11n, OFDM MCS7, POUT = +14 dBm	-	180	-	mA
Receive 802.11b/g/n	-	95 ~ 100	-	mA
Transmit BT/BLE, POUT = 0 dBm	-	130	-	mA
Receive BT/BLE	-	95 ~ 100	-	mA

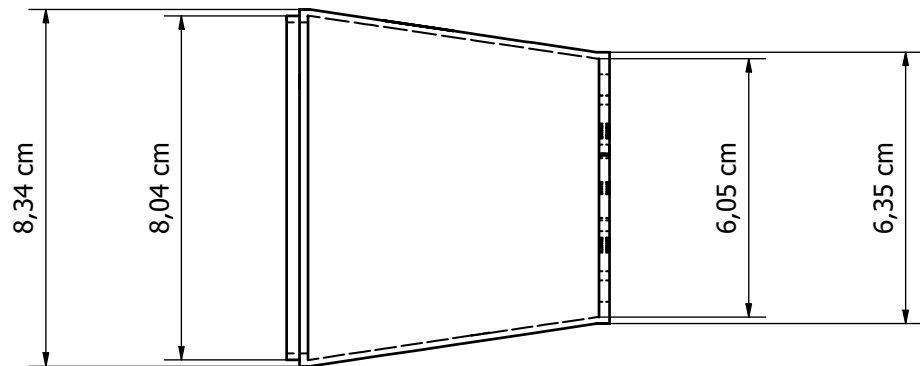
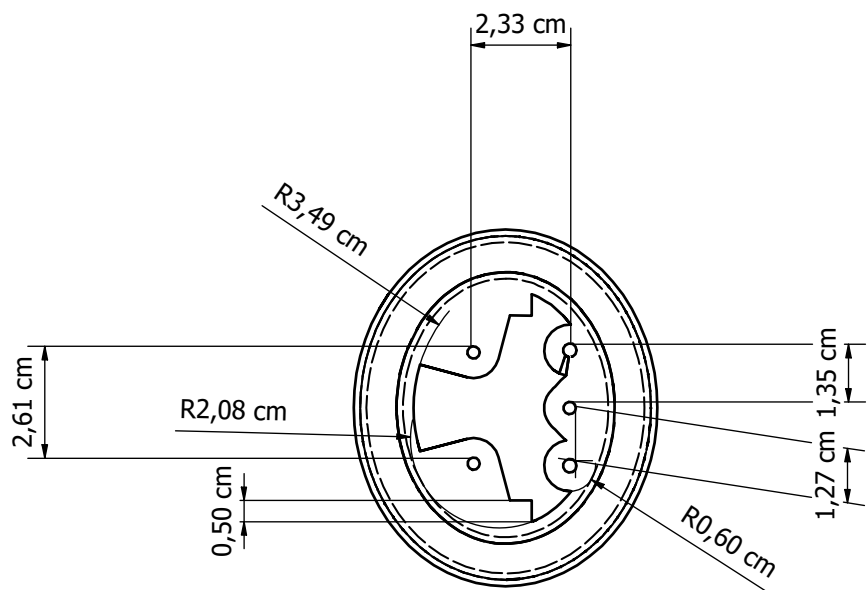
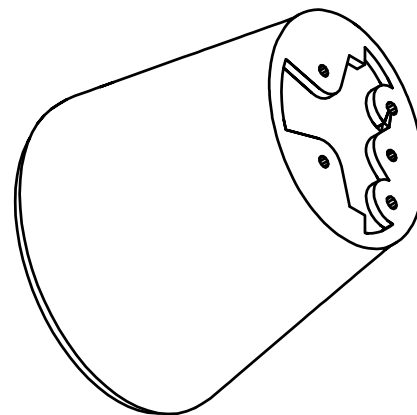
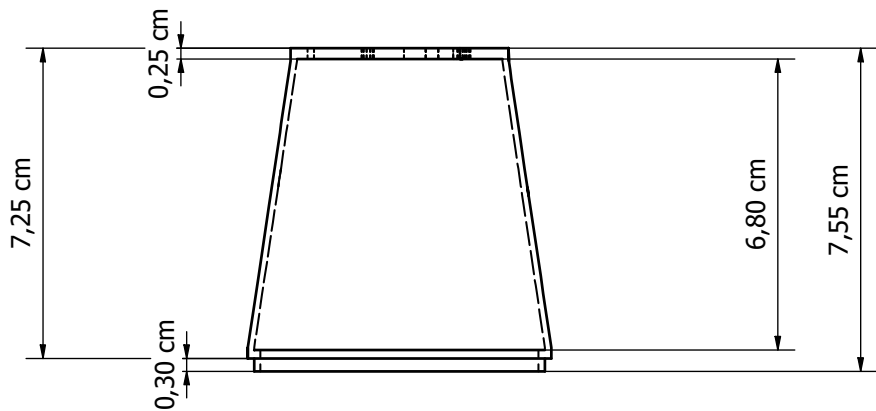
5.6 Wi-Fi Radio

Table 16: Wi-Fi Radio Characteristics

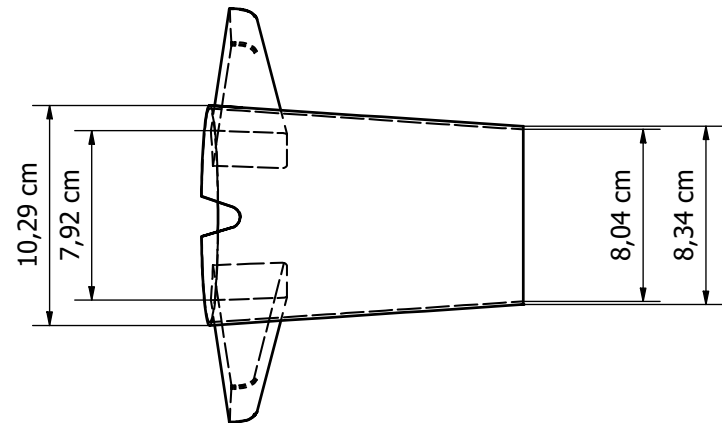
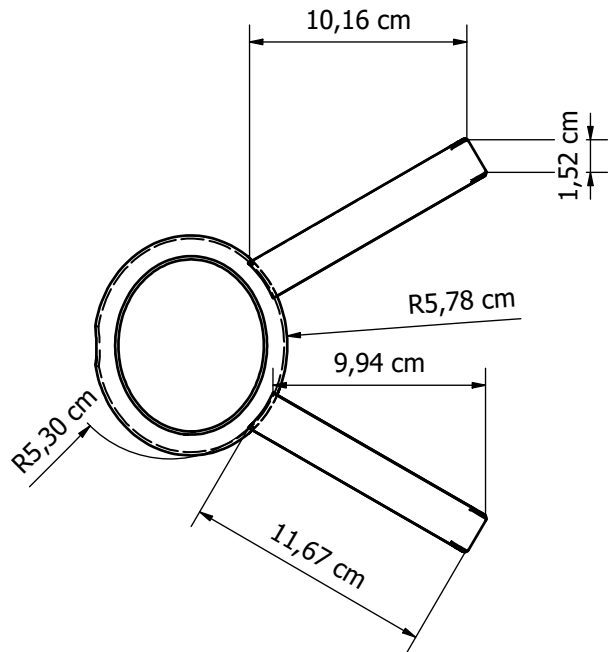
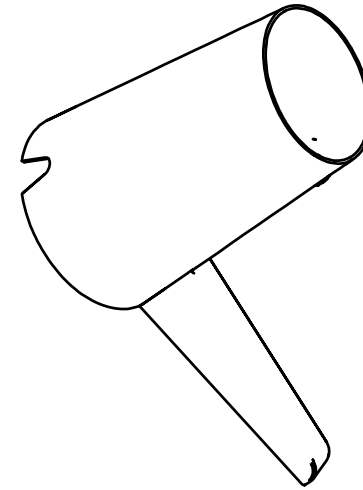
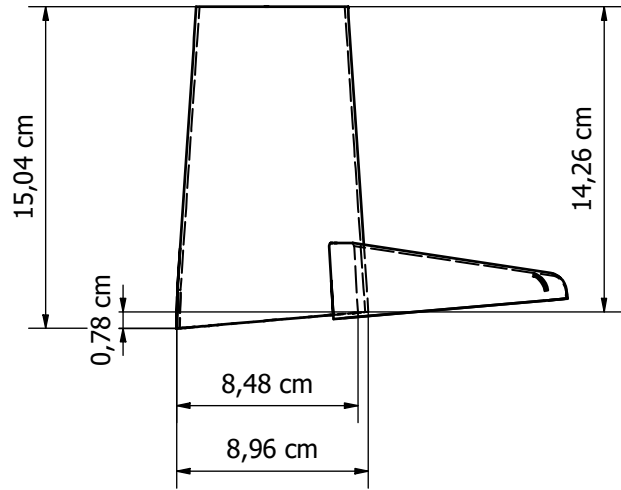
Parameter	Condition	Min	Typ	Max	Unit
Operating frequency range ^{note1}	-	2412	-	2484	MHz
Output impedance ^{note2}	-	-	<i>note 2</i>	-	Ω
TX power ^{note3}	11n, MCS7	12	13	14	dBm
	11b mode	18.5	19.5	20.5	dBm
Sensitivity	11b, 1 Mbps	-	-98	-	dBm
	11b, 11 Mbps	-	-88	-	dBm
	11g, 6 Mbps	-	-93	-	dBm
	11g, 54 Mbps	-	-75	-	dBm
	11n, HT20, MCS0	-	-93	-	dBm
	11n, HT20, MCS7	-	-73	-	dBm
	11n, HT40, MCS0	-	-90	-	dBm
11n, HT40, MCS7	-	-70	-	dBm	

APÉNDICE K

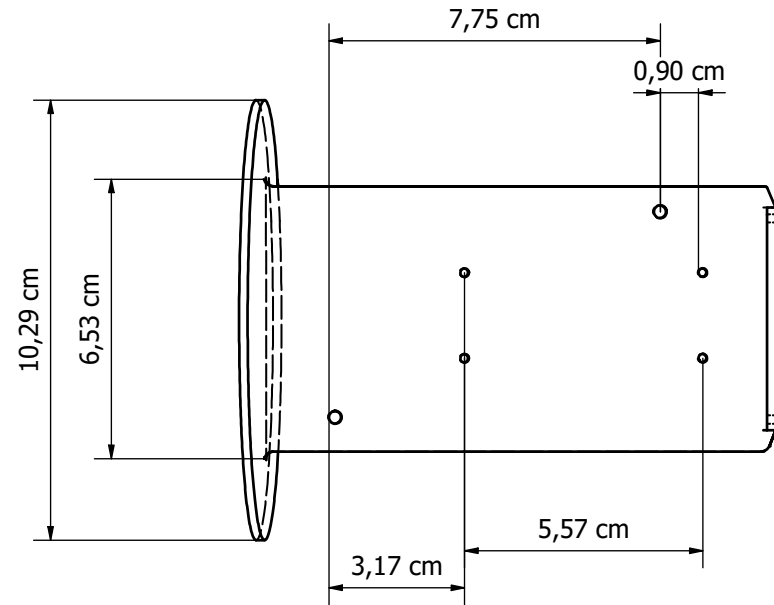
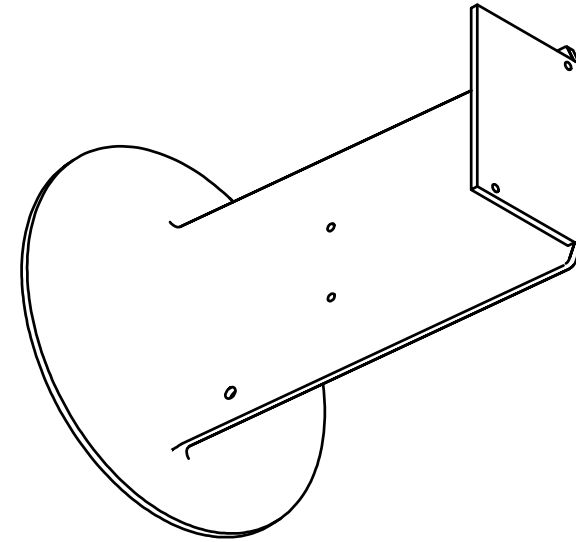
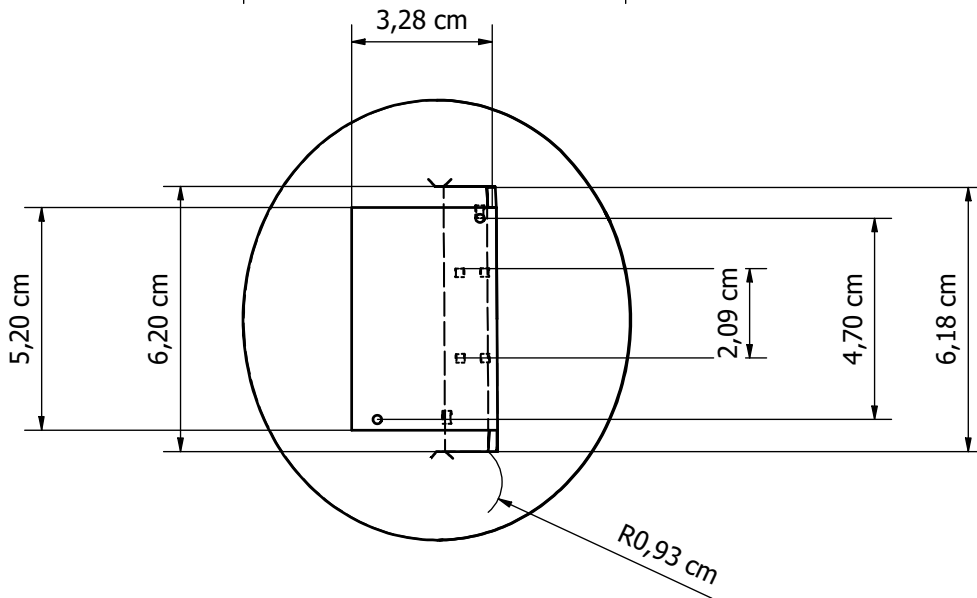
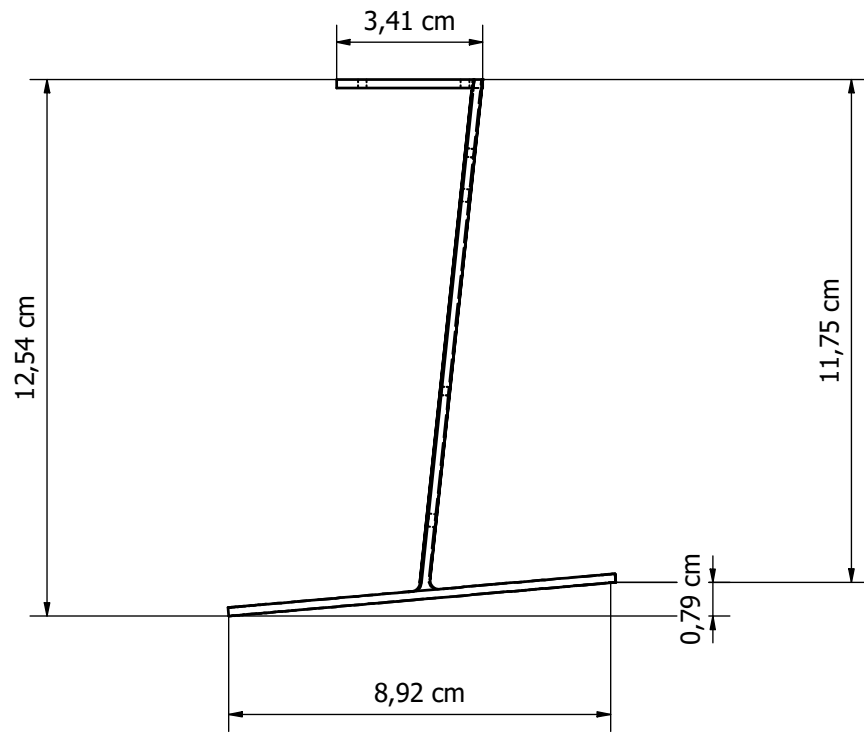
Planos de la base de la MRA



Nombre M.V.B.N	Archivo Diseño en 3D	Dibujado Comprobado	Escala 8:1	Fecha 10/02/2022	
Proyecto de Grado			Apoyo		
Num: 1 de 3				Firmas	



Nombre	Archivo	Dibujado	Escala	Fecha	
M.V.B.N	Diseño en 3D	Comprobado	4:1	10/02/2022	
Proyecto de Grado			Base de MRA		
			Num: 2 de 3	Firmas	



Nombre	Archivo	Dibujado	Escala	Fecha
M.V.B.N	Diseño en 3D	Comprobado	8:1	10/02/2022
Proyecto de Grado		Inserto de MRA		
		Num: 3 de 3	Firmas	