UNIVERSIDAD NACIONAL DE CHIMBORAZO
FACULTAD DE INGENIERIA
CARRERA DE INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

"Diseño de un prototipo de prótesis robótica antropomórfica de mano utilizando bci no invasivo y percepción de temperatura mediante sensores"

Trabajo de Titulación para optar al título de Ingeniero en Electrónica y Telecomunicaciones.

Autor:
Molina Guadalupe, Cristian Marcelo

Tutor:
PhD. Leonardo Fabian Rentería Bustamante

Riobamba, Ecuador. 2024

DERECHOS DE AUTORÍA

Yo, Cristian Marcelo Molina Guadalupe, con cédula de ciudadanía 0603860792, autor del trabajo de investigación titulado: "**DISEÑO DE UN PROTOTIPO DE PRÓTESIS ROBÓTICA ANTROPOMÓRFICA DE MANO UTILIZANDO BCI NO INVASIVO Y PERCEPCIÓN DE TEMPERATURA MEDIANTE SENSORES"**, certifico que la producción, ideas, opiniones, criterios, contenidos y conclusiones expuestas son de mí exclusiva responsabilidad.

Asimismo, cedo a la Universidad Nacional de Chimborazo, en forma no exclusiva, los derechos para su uso, comunicación pública, distribución, divulgación y/o reproducción total o parcial, por medio físico o digital; en esta cesión se entiende que el cesionario no podrá obtener beneficios económicos. La posible reclamación de terceros respecto de los derechos de autor de la obra referida, será de mi entera responsabilidad; librando a la Universidad Nacional de Chimborazo de posibles obligaciones.

En Riobamba, a la fecha de su presentación.

Cristian Marcelo Molina Guadalupe
C.I: 0603860792

ACTA FAVORABLE - INFORME FINAL DEL TRABAJO DE INVESTIGACIÓN
CARRERAS NO VIGENTES

En la Ciudad de Riobamba, a los 14 días del mes de febrero de 2024, luego de haber revisado el Informe Final del Trabajo de Investigación presentado por el estudiante **Molina Guadalupe Cristian Marcelo** con CC: **0603860792**, de la carrera **ELECTRÓNICA Y TELECOMUNICACIONES** y dando cumplimiento a los criterios metodológicos exigidos, se emite el **ACTA FAVORABLE DEL INFORME FINAL DEL TRABAJO DE INVESTIGACIÓN** titulado **"DISEÑO DE UN PROTOTIPO DE PRÓTESIS ROBÓTICA ANTROPOMÓRFICA DE MANO UTILIZANDO BCI NO INVASIVO Y PERCEPCIÓN DE TEMPERATURA MEDIANTE SENSORES"**, por lo tanto se autoriza la presentación del mismo para los trámites pertinentes.

LEONARDO FABIAN
RENTERIA BUSTAMANTE

PhD. Leonardo Rentería
TUTOR

DICTAMEN FAVORABLE DEL TUTOR Y MIEMBROS DE TRIBUNAL

Quienes suscribimos, catedráticos designados Tutor y Miembros del Tribunal de Grado para la evaluación del trabajo de investigación **"DISEÑO DE UN PROTOTIPO DE PRÓTESIS ROBÓTICA ANTROPOMÓRFICA DE MANO UTILIZANDO BCI NO INVASIVO Y PERCEPCIÓN DE TEMPERATURA MEDIANTE SENSORES"**, presentado por Cristian Marcelo Molina Guadalupe, con cédula de identidad número 0603860792, certificamos que recomendamos la APROBACIÓN de este con fines de titulación. Previamente se ha asesorado durante el desarrollo, revisado y evaluado el trabajo de investigación escrito y escuchada la sustentación por parte de su autor; no teniendo más nada que observar.

De conformidad a la normativa aplicable firmamos, en Riobamba a la fecha de su presentación.

Mgs./ PhD. Carlos Ramiro Peñafiel Ojeda
PRESIDENTE DEL TRIBUNAL DE GRADO

Mgs. Klever Hernan Torres Rodríguez
MIEMBRO DEL TRIBUNAL DE GRADO

Mgs. Edgar Giovanny Cuzco Silva
MIEMBRO DEL TRIBUNAL DE GRADO

PhD. Leonardo Fabián Rentería Bustamante
TUTOR

CERTIFICADO DE LOS MIEMBROS DEL TRIBUNAL

Quienes suscribimos, catedráticos designados Miembros del Tribunal de Grado para la evaluación del trabajo de investigación "**DISEÑO DE UN PROTOTIPO DE PRÓTESIS ROBÓTICA ANTROPOMÓRFICA DE MANO UTILIZANDO BCI NO INVASIVO Y PERCEPCIÓN DE TEMPERATURA MEDIANTE SENSORES"**, presentado por Cristian Marcelo Molina Guadalupe, con cédula de identidad número 0603860792, bajo la tutoría de PhD. Leonardo Fabian Rentería Bustamante; certificamos que recomendamos la APROBACIÓN de este con fines de titulación. Previamente se ha evaluado el trabajo de investigación y escuchada la sustentación por parte de su autor; no teniendo más nada que observar.

De conformidad a la normativa aplicable firmamos, en Riobamba a la fecha de su presentación.

Mgs./ PhD. Carlos Ramiro Peñafiel Ojeda
PRESIDENTE DEL TRIBUNAL DE GRADO

Mgs. Klever Hernan Torres Rodríguez
MIEMBRO DEL TRIBUNAL DE GRADO

Mgs. Edgar Giovanny Cuzco Silva
MIEMBRO DEL TRIBUNAL DE GRADO

CERTIFICACIÓN

Que, Molina Guadalupe Cristian Marcelo con CC: 0603860792, estudiante de la Carrera **ELECTRÒNICA Y TELECOMUNICACIONES**, **NO VIGENTE**, Facultad de INGENIERIA; ha trabajado bajo mi tutoría el trabajo de investigación titulado " DISEÑO DE UN PROTOTIPO DE PRÓTESIS ROBÓTICA ANTROPOMÓRFICA DE MANO UTILIZANDO BCI NO INVASIVO Y PERCEPCIÓN DE TEMPERATURA MEDIANTE SENSORES", cumple con el 4%, de acuerdo al reporte del sistema Anti plagio **TURNITIN**, porcentaje aceptado de acuerdo a la reglamentación institucional, por consiguiente autorizo continuar con el proceso.

Riobamba, 16 de 02 de 2024

LEONARDO FABIAN
RENTERIA BUSTAMANTE

PhD. Leonardo Rentería
TUTOR TRABAJO DE INVESTIGACIÓN

DEDICATORIA

«El desarrollo del hombre depende fundamentalmente de la invención. Es el producto más importante de su cerebro creativo. Su objetivo final es el dominio completo de la mente sobre el mundo material y el aprovechamiento de las fuerzas de la naturaleza a favor de las necesidades humanas».

Nikola Tesla.

Este trabajo de tesis se lo dedico con mucho cariño a mi mamá Rosario Guadalupe, quien supo creer ciegamente en mí sin importarle nada; a mi papá Marcelo Molina por apoyarme y a mis hermanos.

Cristian Molina Guadalupe.

AGRADECIMIENTO

«Es posible para la gente normal elegir ser extraordinaria»**.**

***Elon Musk*.**

Empezaré por agradecer al ser celestial que me acompaña todos los días, *Dios*, sus ideas llegan a mí sin aviso previo. Mediante él he logrado materializar mis sueños y mi imaginación.

A mi mami *Rosario Guadalupe* por siempre creer en mí, estoy en deuda con ella, no me va alcanzar la vida para recompensar lo que hace y hará por mí.

A mi papi *Marcelo Molina* por apoyarme en mis proyectos, sin importarle el riesgo que significaba.

A mis hermanos *Kevin y Génesis*, gracias por existir, sólo ustedes logran hacerme reír.

A mi tutor *Leonardo Rentería* quien supo acompañarme en este proyecto en las buenas y en las malas.

A todos los docentes que tuve en la universidad quienes me han enseñado varias lecciones para ser un buen profesional y también como persona.

Cristian Molina Guadalupe.

ÍNDICE GENERAL

DERECHOS DE AUTORÍA

DICTAMEN FAVORABLE DEL TUTOR

DICTAMEN FAVORABLE DEL TUTOR Y MIEMBROS DE TRIBUNAL

CERTIFICADO DE LOS MIEMBROS DEL TRIBUNAL

CERTIFICADO DE ANTIPLAGIO

DEDICATORIA

AGRADECIMIENTO

ÍNDICE GENERAL 9

ÍNDICE DE TABLAS 11

ÍNDICE DE FIGURAS 12

RESUMEN 13

ABSTRACT 14

CAPITULO I 15

1.1. INTRODUCCIÓN 15

1.2. PLANTEAMIENTO DEL PROBLEMA 17

1.3. JUSTIFICACIÓN 18

1.4. OBJETIVOS 18

1.4.1. OBJETIVO GENERAL 18

1.4.2. OBJETIVOS ESPECÍFICOS 18

CAPITULO II. MARCO TEÓRICO 19

2.1 Investigaciones Internacionales 19

2.1.1 Investigaciones Nacionales 20

CAPITULO III. METODOLOGÍA 23

3.1. TIPO DE INVESTIGACIÓN 23

3.2. DISEÑO DE INVESTIGACIÓN 23

3.2.1 Fase de investigación 23

3.3. DESARROLLO DEL PROTOTIPO 24
3.3.1. Diseño de la Prótesis de Mano 24
3.3.2. Diagrama de conexión 28
3.4. TÉCNICAS DE RECOLECCIÓN DE DATOS 29
3.4.1. OPERACIONALIZACIÓN DE VARIABLES 30
3.5. POBLACIÓN DE ESTUDIO Y TAMAÑO DE MUESTRA 30
3.5.1. Población 30
3.5.2. Muestra 30
3.6. PROCESO DE ADQUISICION DE DATOS 31
3.6.1. Procedimiento de Comunicación 31
3.6.2. Software de adquisición de datos 32
3.6.3. Software transmisor de datos del giroscopio MPU6050 34
3.6.4. Software de Control de microservomotores y lectura de temperatura 34
CAPÍTULO IV. RESULTADOS Y DISCUSIÓN 36
4.1. Prueba de flexión y extensión 36
4.2. Prueba de lectura de temperatura 39
CAPÍTULO V. CONCLUSIONES Y RECOMENDACIONES 41
5.1. Conclusiones. 41
5.2. Recomendaciones. 42
BIBLIOGRAFÍA 43
ANEXOS 45

ÍNDICE DE TABLAS

Tabla 1.	Comparativa de prótesis disponibles en el mercado actual.	17
Tabla 2.	Comparativa dispositivos bci	22
Tabla 3.	Norma din 33 402 segunda parte.	26
Tabla 4.	Descripción del dedo.	26
Tabla 5.	Pruebas de normalidad con respecto al tiempo.	37
Tabla 6.	Pruebas de normalidad con respecto al tiempo.	37
Tabla 7.	Test de tukey.	38
Tabla 8.	Pruebas de normalidad con respecto a la posición.	39
Tabla 9.	Prueba de hipótesis respecto a la posición con un nivel de significación de 0.05.	39
Tabla 10.	Pruebas de normalidad con respecto al dispositivo de lectura.	40
Tabla 11.	Prueba de hipótesis respecto al dispositivo con un nivel de significación de 0.05.	40

ÍNDICE DE FIGURAS

Figura 1. Servomotor MG90S, servomotor MG996R. 24
Figura 2. Tarjeta LattePanda 2G/32G. 25
Figura 3. Sensor de Temperatura MLX90614, Driver servomotores PCA9685. 25
Figura 4. Diseño del dedo. 26
Figura 5. Falanges del dedo. 27
Figura 6. Diseño de la palma. 27
Figura 7. Diseño de la muñeca. 27
Figura 8. Soporte de muñón. 28
Figura 9. Ensamble completo de la mano. 28
Figura 10. Diagrama del Sistema Electrónico 1(Tarjeta principal LattePanda, microservos, Led RGB y sensor de temperatura). 29
Figura 11. Diagrama del Sistema Electrónico 2 (MPU6050) 29
Figura 12. Aplicación Emotiv. 31
Figura 13. Algoritmo adquisición de datos entre Lattepanda y Emotiv Insight. 33
Figura 14. Algoritmo Transmisor de datos del giroscopio MPU6050. 34
Figura 15. Algoritmo Control de micro servomotores y lectura de temperatura. 35
Figura 16. Diagrama de cajas: Número de aciertos en función del tiempo. 36
Figura 17. Gráfica de medias del número de aciertos en función del tiempo. 38
Figura 18. Diagrama de cajas: Número de aciertos en función del de la posición. 38
Figura 19. Diagrama de cajas: Temperatura en función del tipo de dispositivo utilizado para medir. 40

RESUMEN

En el presente proyecto, se desarrolló un modelo inicial de prótesis que emplea la tecnología de impresión 3D como alternativa no intrusiva destinada a personas con amputaciones en la extremidad superior, específicamente por debajo del codo. El funcionamiento de este dispositivo se basa en la recepción de señales endógenas generadas por ondas EEG del usuario, lo que posibilita el control de los movimientos de flexión y extensión de la mano. Además, este proporciona la sensación de temperatura mediante indicadores de color. Este trabajo fue desarrollado en tres fases; en la primera fase de llevó a cabo un estudio sobre los diferentes tipos de interfaces cerebro computador y de las tecnologías existentes, en la segunda se diseñó, ensambló e implementó el prototipo de prótesis y en la última se realizaron las pruebas de funcionamiento. El funcionamiento del prototipo fue evaluado respecto al efecto de la posición del usuario y el tiempo de tiempo de espera para ejecutar el comando, en el número de aciertos al momento de ejecutar instrucciones de flexión y extensión. Como resultado principal se pudo determinar que la posición del usuario, sentado o parado no tiene efecto alguno en el número de aciertos, mientras que el tiempo de espera si, siendo 5 segundos el tiempo con menor número de aciertos, 22 de 30 en comparación con 10 y 15 segundos con 26 de 30. Así mismo, los datos arrojados por el sistema de percepción de temperatura fueron comparado con los medidos con un termómetro comercial dando como resultado un promedio estadísticamente igual.

Palabras clave: Prótesis, 3D, percepción de temperatura, ondas cerebrales, flexión y extensión

ABSTRACT

The main objective of this research study was to focus on an initial model of a prosthesis was developed using 3D printing technology as a non-intrusive alternative for people with upper limb amputations, specifically below the elbow. The operation of this device is based on the reception of endogenous signals generated by EEG waves from the user, making it possible to control the flexion and extension movements of the hand. In addition, it provides a temperature sensation using color indicators. This work was developed in three phases; in the first phase a study of the different types of brain-computer interfaces and existing technologies was carried out, in the second phase the prosthesis prototype was designed, assembled, and implemented, and in the last phase, functional tests were carried out. The functioning of the prototype was evaluated concerning the effect of the user's position and the waiting time to execute the command on the number of successes when executing flexion and extension instructions. As a main result, it was determined that the position of the user, sitting or standing, does not affect the number of hits, while the waiting time does, being 5 seconds the time with the lowest number of hits, 22 out of 30 compared to 10 and 15 seconds with 26 out of 30. Likewise, the data provided by the temperature perception system were compared with those measured with a commercial thermometer resulting in a statistically equal average.

Keywords: Prosthesis, 3D, temperature perception, brain waves, flexion and extension.

MARCO ANTONIO AQUINO ROJAS

Reviewed by:
Marco Antonio Aquino
ENGLISH PROFESSOR
C.C. 1753456134

CAPITULO I

1.1. INTRODUCCIÓN

La mano es uno de los órganos más importantes del cuerpo que permite la manipulación física del medio. En la evolución humana, las manos se han convertido en un instrumento de variados y delicados usos que pueden realizar movimientos amplios y precisos (Puchades, 2003).

Las manos y muñecas son la parte del cuerpo que más se exponen a la hora de trabajar y por lo tanto las que más se lesionan, sin embargo, la mayoría de las lesiones en las manos ocurren por actos imprudentes o por realizar acciones con mucha prisa y poca atención (México, 2016). Existen personas alrededor del mundo que sufren pérdida o amputación de sus miembros superiores debido a traumas, sepsis, accidentes, enfermedades como el cáncer o malformaciones congénitas, tumores malignos u otras causantes (Ottobock, 2014).

Por otro lado, la aplicación de la ingeniería junto con otras disciplinas ha permitido el desarrollo de prótesis, ortesis y férulas. Las prótesis permiten la sustitución del miembro perdido devolviendo las habilidades y la apariencia muy cercana a la realidad, algunas, no solo reemplazan en apariencia sino cumplen también con la movilidad parcial en los dedos llevando una vida más normal para la persona con este tipo de discapacidad (Campillo, 2015). Así mismo, se ha llevado a cabo investigaciones de interfaces cerebro-máquina que intentan conectar el "pensamiento" humano con dispositivos electrónicos de forma natural, llamadas neuroprótesis, las mismas que permitirían a una persona paralítica mover miembros artificiales con la mente (Kaku, 2014). Por ejemplo; Alex Pinkerton, cofundador de Brane Interface y CEO, utiliza grafeno y Math CAD para desarrollar una interfaz BCI, intenta llegar al nivel del pensamiento humano con el objetivo de darle aplicaciones para controlar dispositivos físicos como teléfonos celulares y prótesis robóticas (Interface, 2020). Igualmente, NEURALINK fundada por Elon Musk, planea implantar un chip de forma invasiva para crear una interfaz BCI capaz de conectarse a la inteligencia artificial, tratar una gama amplia de trastornos neurológicos, restaurar la función sensorial y de movimiento, además expandir la forma en que interactuamos con nosotros y el mundo (Neuralink, 2021). En este contexto, la presente investigación propone el diseño y creación de una prótesis de mano robótica controlada por ondas del cerebro y con la capacidad de percepción artificial de la temperatura mediante sensores e indicadores. Este trabajo se estructura de los siguientes capítulos:

Capítulo I - Introducción: Se muestra de manera definida la problemática que incentivó a la realización del presente trabajo de investigación, tomando en cuenta investigaciones anteriores como pauta para el diseño en implementación del prototipo. Se describe la justificación y se formulan los objetivos.

Capítulo II - Marco teórico: Este capítulo contempla la información necesaria referente a los diferentes dispositivos electrónicos, tarjetas de desarrollo, sensores y software que se han

utilizado así como un resumen de algunas investigaciones relacionadas con el tema propuesto en relación al diseño y construcción de prótesis.

Capítulo III – Metodología: En la sección de metodología se detalla cada una de las fases del proyecto. Se presentan las técnicas e instrumentos y las variables de la investigación, se plantea la hipótesis y se determinan la población y la muestra.

Capitulo IV: En este capítulo se detallan los resultados obtenidos y experimentales, fruto de la puesta en ejecución del prototipo de prótesis de mano.

Capítulo V: Este es el último apartado, aquí, se exponen las conclusiones del trabajo realizado, se plantean las recomendaciones y se formulan los trabajos futuros.

1.2. PLANTEAMIENTO DEL PROBLEMA

Los traumas son la principal causa de amputaciones y contemplan el 30% de los casos. Así mismo, otras de las cusas comunes son las amputaciones de miembros superiores causadas por enfermedades o malformaciones congénitas (Proteus, s.f.). Según el INEC (Instituto Nacional de Estadística y Censos) en un censo realizado en Ecuador durante el año 2018 de pacientes egresados en hospitales, existen 681 casos de amputación traumática de la muñeca y de la mano, de los cuales 550 personas entre 15 y más de 65 años de edad sufren esta pérdida de miembro superior. Dentro de los casos de amputación de antebrazo existen 12 personas que padecen esta pérdida (INEC, 2021).

Las pérdidas de miembros superiores son cada vez más habituales, sobre todo en personas de bajos recursos que no tienen las posibilidades para acceder a prótesis biónicas. Adicionalmente, existen grupos de amputados que presentan atrofia muscular o alguna enfermedad de parálisis y el uso de una prótesis mioeléctrica no podría ser efectiva debido a la discriminación de señales o la no personalización del encaje de la prótesis.

A los problemas mencionados anteriormente, se suman los costos de las extremidades robóticas de brazo que son bastante elevados y por ende inaccesibles para la mayoría de las personas que las necesitan. Dependiendo de la construcción, metodología y funcionalidad de los prototipos de las prótesis de mano que existen en el mercado se describen en la siguiente tabla [3], [10], [11].

Tabla 1. *Comparativa de prótesis disponibles en el mercado actual.*

Comparativa			
Prótesis		**Principal característica**	**Costo aproximado $**
Mioeléctrica	I-Limb quantum	Diseño para gesto de pinza y responde a movimientos preprogramados.	70000-140000
	Michelangelo prosthetic hand (Ottobock, 2019)	Manipulación de objetos y responde a movimientos preprogramados.	140000
	Bebionic 3	Diseño para gesto de pinza y responde a movimientos preprogramados.	25000-35000
	Probionics	Control mioeléctrico con movimientos programados.	2800-8600
	Hero Arm	Capaz de realizar gesto de pinza.	3000

1.3. JUSTIFICACIÓN

Con base a los antecedentes antes mencionados, el presente trabajo de titulación "DISEÑO DE UN PROTOTIPO DE PRÓTESIS ROBÓTICA ANTROPOMÓRFICA DE MANO UTILIZANDO BCI NO INVASIVO Y PERCEPCIÓN DE TEMPERATURA MEDIANTE SENSORES" propone el diseño de un prototipo funcional controlado por ondas cerebrales que permita manipular objetos y al mismo tiempo brinde la percepción de temperatura. El prototipo será para pacientes que presenten atrofia muscular y no puedan utilizar prótesis Mioeléctricas. La prótesis será controlada por ondas cerebrales sin la necesidad de cirugía con un método no invasivo utilizando comunicación inalámbrica y contará con: un dispositivo electrónico lector de ondas cerebrales, un sensor de medición inercial, una tarjeta de control, diseño antropomórfico de la mano, prototipo impreso con tecnología 3D, actuadores para cada dedo y muñeca, además de la implementación de un sensor e indicadores de temperatura para seguridad de la prótesis y el paciente.

1.4. OBJETIVOS

1.4.1. OBJETIVO GENERAL

Diseñar y controlar un prototipo de prótesis antropomórfica de una mano utilizando un método no invasivo BCI mediante lectura de ondas cerebrales, y detectar la temperatura de objetos o superficies como método de seguridad tanto del prototipo como al usuario.

1.4.2. OBJETIVOS ESPECÍFICOS

- Estudiar y elegir el dispositivo BCI no invasivo lector de ondas cerebrales, la tarjeta electrónica de desarrollo, motores, sensor de temperatura y alimentación adecuados para el funcionamiento del prototipo.

- Diseñar e implementar un sistema BCI para el control de una prótesis robótica de mano con percepción de temperatura.

- Evaluar el funcionamiento del sistema mediante experimentación y pruebas del prototipo.

CAPITULO II. MARCO TEÓRICO

En este capítulo se abordan los trabajos de investigación y desarrollo de sistemas y/o prototipos relacionados con el tema del proyecto, en el ámbito mundial, nacional y local. Se presentan los autores, métodos utilizados y resultados más importantes.

2.1 Investigaciones Internacionales

En la actualidad existen estudios de neuro-prótesis capaces de ser controladas por estímulos del cerebro como por ejemplo Luke Skywalker, DeTop y BrainCo.

La prótesis de brazo biónico Luke Skywalker está dirigida especialmente para personas con amputación de brazo por debajo del hombro, para veteranos de guerra o voluntarios. La "DARPA" junto con el Laboratorio Johns Hopkins de Física Aplicada desarrollaron un brazo robótico mediante osteointegración. El prototipo es controlado por la mente por medio de los nervios y músculos que se mantienen en el brazo de la persona, estos envían señales al brazo robótico para que responda como uno real (Moncrieffe, 2017).

En el proyecto DeTop los científicos utilizaron una tecnología OHMG (pasarela humano-máquina osteointegrada) que es capaz de crear una conexión física entre una persona y una prótesis robótica. El sistema es instalado directamente en el hueso del brazo receptor, unos electrodos son conectados a los nervios y músculos para extraer las señales y controlar una mano robótica proporcionando sensaciones táctiles (DeTOP, 2016).

La prótesis de mano de BrainCo funciona con una diadema e inteligencia artificial, capta ondas cerebrales y señales musculares del amputado para poder intuir el movimiento. En este trabajo, descubrieron una nueva forma de entender las señales eléctricas procedentes del cerebro, desde la extremidad residual de la persona (Rayome, 2020).

Generalmente, las prótesis mioeléctricas del mercado presentan diseños cómodos para el paciente, Hero Arm es liviana y asequible, equipada con sensores que detectan movimientos musculares, además de vibraciones hápticas que proporcionan notificaciones de su uso, pesa menos de 1kg (Bionics O. , 2018).

Otro ejemplo es BeBionic, una mano protésica diseñada en Reino Unido que posee 14 agarres diferentes de mano, diseño estético, aspecto moderno, ajustable, distribución óptima de peso y además del movimiento sincronizado de los dedos (Ackerman, 2012).

Prótesis robustas de mano como el prototipo Luke Ar, protegen al usuario del polvo y de la lluvia ligera, este brazo protésico está programado con múltiples patrones de agarre, la muñeca puede rotar. Es alimentado con 14.8v y una corriente de 5 a 7 amperios (Bionics M. , 2021).

Otro prototipo de diseño robusto es el HY5 MYHAND o el prototipo I-Limb quantum, que fue fabricada con dígitos de titanio, 36 agarres diferentes, 15 grados de libertad, rotación de muñeca de 360°, la fuerza que ejerce los actuadores es de 5kgf cada uno, pesa aproximadamente 1,5kg, hasta un 30% más de fuerza de agarre y un 30% de aumento de velocidad para mejorar el movimiento natural, la fuerza y la funcionalidad (Össur, Össur Life whitout limitations, 2021), (Industries, 2021).

TASKA también tiene un diseño seguro, robusto y versátil, controlado por EMG. Opera con 7,4v-2000mAh, puede cargar un peso de hasta 20Kg, es a prueba de agua; pesa 611g. Está programado con múltiples patrones de agarre, proporciona un agarre seguro de objetos, muñeca de desconexión rápida y además un puerto de carga magnética (Fillauer, 2020).
Vincent Evolution es otro ejemplo de mano mioeléctrica robusta construida con los mejores materiales en acero inoxidable junto con una aleación de magnesio, es ligera y robusta. Incorporado 6 motores que mueven independientemente cada dedo y rotación del pulgar para cada patrón de agarre, programado con 9 movimientos diferentes intuitivos y fáciles de aprender (Ortosur, 2018).

La revolución tecnológica ha concebido diseños asequibles es el caso de Hackberry, es un proyecto de brazo biónico imprimible con tecnología 3D de código abierto, fundada por exii Inc., tiene como característica principal su muñeca y dedos dúctiles, recoge objetos pequeños, ojea libros. Utiliza un smartphone como cerebro y baterías de cámara para funcionar (ICHI.PRO, 2015).

Un prototipo innovador y muy interesante es el caso de la mano biónica Impact Hand, es impresa con tecnología 3D, construida con tecnología de tinta conductiva; incorporada con sensores musculares que controlan el pulgar articulado permitiendo funcionar como una mano. Su tiempo de fabricación es de 10 horas, es el modelo de prótesis capaz de ser impreso completamente en 3D el dispositivo con los circuitos electrónicos integrados para conectar sensores y actuadores (WARWICK, 2019).

2.1.1 Investigaciones Nacionales

Hand of Hope.- La mano robótica “Hand of Hope” fue diseñada y construida en Loja – Ecuador en el año del 2014 entre estudiantes y docentes pertenecientes a la Escuela de Electrónica y Telecomunicaciones de la UTPL bajo la tutela del Ing. Carlos Calderón (Cueva, 2015).

El prototipo puede emular seis diferentes posiciones y agarres: posturas cilíndricas, verticales, horizontales y una para presionar botones. Funciona mediante bioseñales leídas por los sensores musculares y procesada por una tarjeta que ordena los movimientos de la mano protésica (Cueva, 2015).

Brazo Artificial UTN .- Docentes y estudiantes de la UTN trabajan en un prototipo de prótesis que, mediante el desarrollo de un entrenador mío-eléctrico, la persona entrena los músculos para generar señales que activen la prótesis. La tarjeta de adquisición de señales tiene 3 canales, los cuales efectúa 8 movimientos; dichos movimientos son en base a las señales que envía el músculo del usuario a través de electrodos que son tratadas más adelante por una tarjeta de adquisición y permite la ejecución de los actuadores del brazo robótico. Su funcionamiento depende del entrenamiento del usuario, se trabaja actualmente en movimientos de agarre, pinza y supinación (UTN, 2016).

Prótesis robótica ESPOCH.- La prótesis fue diseñada por docentes de la facultad de Mecánica de la ESPOCH, la cual trabaja mediante estímulos eléctricos del cerebro y de los músculos para poder moverse. Su diseño es ergonómico, ligero y fácil de manipular. En la cabeza del usuario se colocan sensores los cuales captan señales electromiográficas del cerebro para que pueda trasladar objetos con el brazo artificial. Se estima que el coste de esta prótesis sea de 3000 dólares (COMERCIO, 2020).

ECUAPRÓTESIS.- Esta empresa ecuatoriana construye prótesis personalizadas con tecnología de impresión 3D, entre las cuales existe las prótesis bajo el codo, ideales para cubrir una amputación del antebrazo bajo el codo. Este prototipo es mecánico totalmente funcional con el movimiento del muñón, de fácil mantenimiento y liviana con respecto a otras prótesis tradicionales (Ecuaprótesis 3D, 2021).

Por otro lado, con relación a dispositivos BCI, la tabla 2, muestra una comparativa entre diferentes tecnologías en lo que respecta a costo, canales, usos y plataforma en la que se desarrolló el proyecto:

Tabla 2. *Comparativa Dispositivos BCI*

Comparativa de Dispositivos EEG no Invasivos							
Dispositivos EEG		**Usos**	**Electrodos**	**Conectividad**	**Duración de la batería**	**Plataforma**	**Precio**
Neurosky Mindwave Mobile 2		Salud, entretenimiento, educación, investigación de mercado.	Sensor FP1 y referencia en el oído.	Bluetooth	8 horas	Windows, Mac, iOS, Android.	99,99
Next-Mind		Entretenimiento, controlar dispositivos electrónicos, aplicaciones con Realidad Virtual	9 electrodos secos en forma de peine.	Bluetooth	8 horas	Windows 10, MacOS.	448
Muse 2		Neurociencia, investigación científica, salud.	7 canales: 2 en la frente, 2 detrás de las orejas y 3 sensores de referencia.	Bluetooth	5 horas	iOS 11, Android 5 o superior.	219
EMOTIV	EPOCX	Investigación Cientifica, uso personal, neurociencia, desarrollo de aplicaciones.	14 canales: AF3, F7, F3, FC5, T7, P7, O1, O2, P8, T8, FC6, F4, F8, AF4	Bluetooth	9 horas	Windows, Mac, iOS, Android.	799
	EPOC+	Investigación Cientifica, uso personal, neurociencia, desarrollo de aplicaciones.	14 canales: AF3, F7, F3, FC5, T7, P7, O1, O2, P8, T8, FC6, F4, F8, AF4	Bluetooth	6 horas	Windows, Mac, iOS, Android.	699
	EPOC FLEX	Investigación Cientifica, uso personal, neurociencia, desarrollo de aplicaciones.	Hasta 32 canales	Bluetooth	9 horas	Windows, Mac.	2099
	INSIGHT	Investigación Cientifica, uso personal, neurociencia, desarrollo de aplicaciones.	5 canales: AF3, AF4, T7, T8, Pz	Bluetooth	4 horas	Windows, Mac, iOS, Android.	300

CAPITULO III. METODOLOGÍA

3.1. TIPO DE INVESTIGACIÓN

El método inductivo es aplicado en la fase investigativa ya que se analiza situaciones particulares partiendo de antecedentes en las ciencias médicas, robótica, diseño mecánico, programación y desarrollo de algoritmos, dando lugar al diseño e implementación de un prototipo robótico de mano haciendo uso de interfaz cerebro-máquina no invasivo y utilizando percepción de temperatura mediante un sensor.

Se utiliza el método heurístico en las fases de: selección, diseño, ensamble e implementación puesto que resuelven un problema que se está tratando y le dan solución al mismo; se seleccionan los elementos necesarios para el control adecuado de la prótesis y un diseño, ensamble e implementación ideales para su correcto funcionamiento.

Mediante el método experimental se realiza pruebas de funcionamiento y las respectivas correcciones para que pueda ser adecuado a una persona amputada de miembro superior de mano.

3.2. DISEÑO DE INVESTIGACIÓN

Para el presente proyecto de investigación se aplican los métodos de investigación: inductivo, heurístico y experimental que a continuación se detallan por fases:

3.2.1 Fase de investigación

- Recopilar la información necesaria para la realización del prototipo.
- Realizar un estudio sobre los diferentes softwares de diseño, simulación y programación del prototipo.
- Investigación sobre los diferentes dispositivos BCI no invasivos existentes en el mercado nacional y extranjero.

Fase de selección

- Se selecciona los elementos electrónicos para el prototipo tales como: servo-motores, driver de los motores, placa electrónica de desarrollo y alimentación del prototipo.

Fase de diseño

- Diseño del prototipo robótico de mano en el software SolidWorks 2018.
- Ensamble y cinemática virtual del prototipo robótico de mano en el software SolidWorks 2018.

Fase de ensamble

- Transformar los archivos del software SolidWorks a extensión.stl para ser impresos con tecnología 3D.
- Realizar el ensamble de las piezas impresas de la prótesis.
- Colocar los elementos electrónicos, tarjeta de desarrollo, sensores de temperatura y alimentación de la prótesis de mano.

Fase de implementación

- Desarrollo de algoritmos programables en el lenguaje de programación escogido.
- Realización de la comunicación entre la tarjeta de desarrollo y el lector de ondas cerebrales BCI no invasivo.

Fase de pruebas, correcciones y resultados

- Realizar las pruebas de funcionamiento y posteriormente correcciones del prototipo de mano robótica con respecto a: programación, agarres del prototipo, indicadores, sensores, comunicación y calibración de la prótesis.
- Elaboración y presentación de la investigación del prototipo de prótesis robótica de mano.

3.3. DESARROLLO DEL PROTOTIPO

3.3.1. Diseño de la Prótesis de Mano

El diseño del prototipo desarrollado en este proyecto se basó en el modelo Youbionic Hand 2021 creado por Federico Ciccarese, sobre todo, debido a que se pueden utilizar componentes electrónicos de bajo costo.

En primera instancia, se diseñaron los servomotores con las respectivas medidas para que se realice el movimiento de los dedos y muñeca, figura 1.

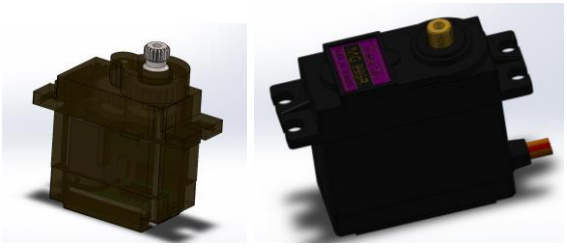

Figura 1. Servomotor MG90S, servomotor MG996R.

Los modelos de componentes electrónicos como la tarjeta de desarrollo, el controlador de servomotores y el sensor de temperatura fueron tomados de librerías existentes de la web, figura 2 y 3.

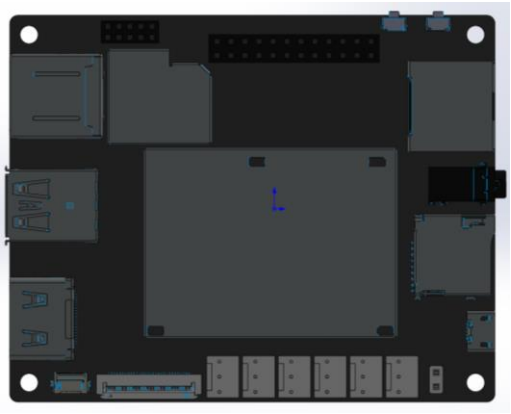

Figura 2. Tarjeta LattePanda 2G/32G.

Figura 3. Sensor de Temperatura MLX90614, Driver servomotores PCA9685.

Para el diseño mecánico del prototipo se consideraron las medidas antropomórficas de la mano diestra según la Norma DIN 33 402 Segunda Parte. Esta normativa que asegura la calidad de productos científicos e industriales, a continuación, se observa la Tabla 3 expresada en percentiles:

Tabla 3. Norma DIN 33 402 Segunda Parte.

Dimensiones en cm		Percentiles		
		5%	50%	95%
22	Ancho del meñique en la palma de la mano	1,8	1,7	1,8
23	Ancho del meñique próximo de la yema	1,4	1,5	1,7
24	Ancho del dedo anular en la palma de la mano	1,8	2,0	2,1
25	Ancho del dedeo anular próximo a la yema	1,5	1,7	1,9
26	Ancho del dedo mayor en la palma de la mano	1,9	2,1	2,3
27	Ancho del dedo mayor próximo a la yema	1,7	1,8	2,0
28	Ancho del dedo índice en la palma de la mano	1,9	2,1	2,3
29	Ancho del dedo índice próximo a la yema	1,7	1,8	2,0
30	Largo del dedo meñique	5,6	6,2	7,0
31	Largo del dedo anular	7,0	7,7	8,6
32	Largo del dedo mayor	7,5	8,3	9,2
33	Largo del dedo índice	6,8	7,5	8,3
34	Largo del dedo pulgar	6,0	6,7	7,6
35	Largo de la palma de la mano	10,1	10,9	11,7
36	Largo total de la mano	17,0	18,6	20,1
37	Ancho del dedo pulgar	2,0	2,3	2,5

El diseño de las falanges de los dedos se basó en el modelo Youbionic Hand, se realizó un escalamiento de las 4 falanges que constituyen un dedo, cada dedo con dos grados de libertad como se muestra en la figura 4.

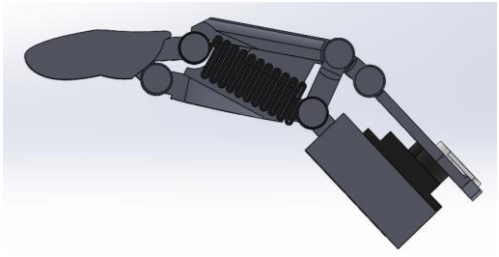

Figura 4. Diseño del dedo.

En la tabla 4 y figura 5 se detalla cómo está compuesto el diseño del dedo:

Tabla 4. *Descripción del dedo.*

N°	Falanges	Descripción
1	Distal-Media	Activado por el sistema adaptativo.
2	Proximal	Sistema adaptativo
3	Metacarpo	Porta servos
4	Eslabón	Activa el movimiento del dedo

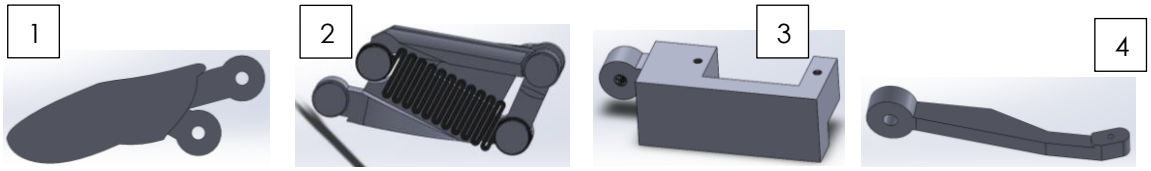

Figura 5. Falanges del dedo.

El dieño de la palma de la mano se muestra en la figura 6; esta fue diseñada para contener las baterías, el controlador de servomotores, el sensor de temperatura y un servomotor.

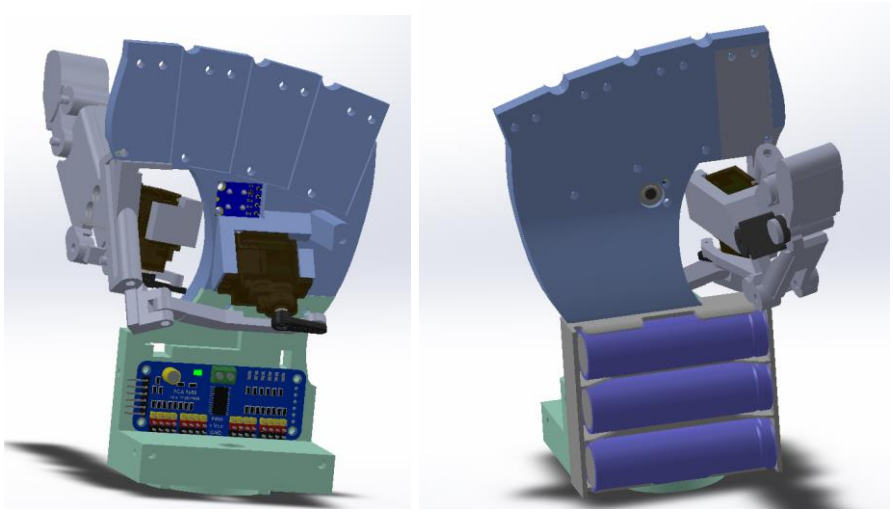

Figura 6. Diseño de la palma.

Para el movimiento de la muñeca se diseñó una transmisión simple con el servomotor hacia el disco de la muñeca de la mano, como se observa en la figura 7:

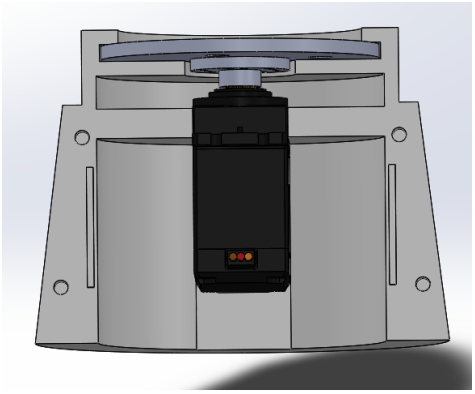

Figura 7. Diseño de la muñeca.

Para apoyar el muñón del usuario se diseñó una pieza particular que conecta con la muñeca, como se muestra en la figura 8:

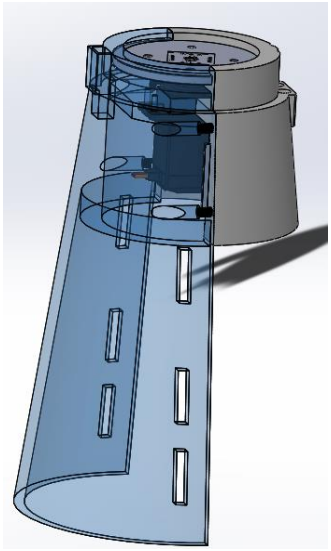

Figura 8. Soporte de muñón.

En la figura 9 se puede apreciar el ensamble completo de la mano:

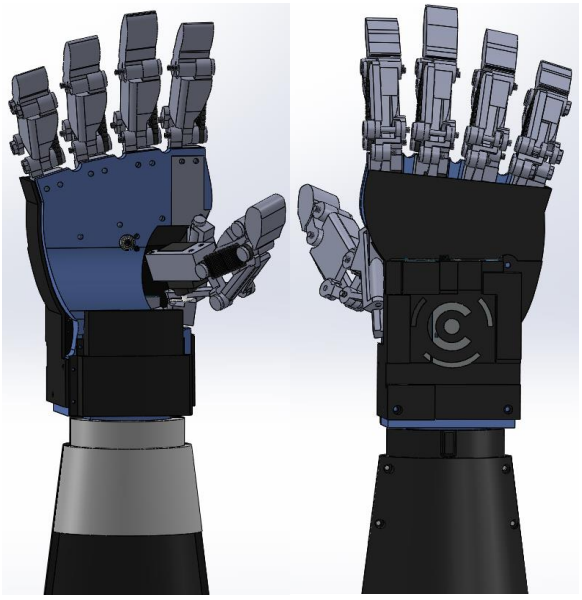

Figura 9. Ensamble completo de la mano.

3.3.2. Diagrama de conexión

Para que el sistema este en óptimas condiciones de funcionamiento, se calculó la alimentación necesaria para un funcionamiento autónomo de 4 horas. El diagrama de las figuras 10 y 11 muestran el diagrama completo de conexión de la prótesis:

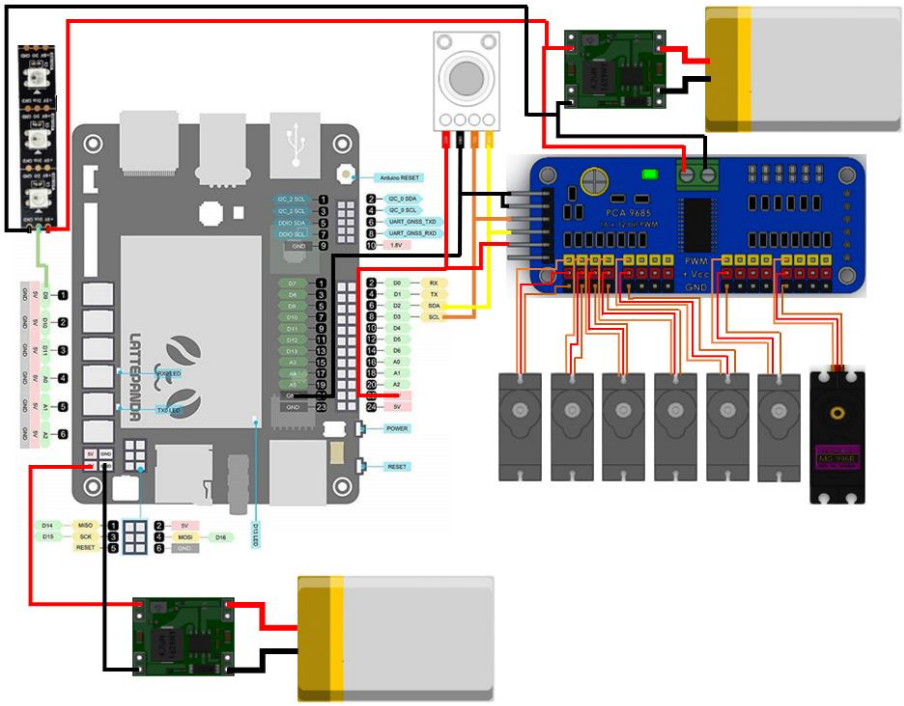

Figura 10. Diagrama del Sistema Electrónico 1(Tarjeta principal LattePanda, microservos, Led RGB y sensor de temperatura).

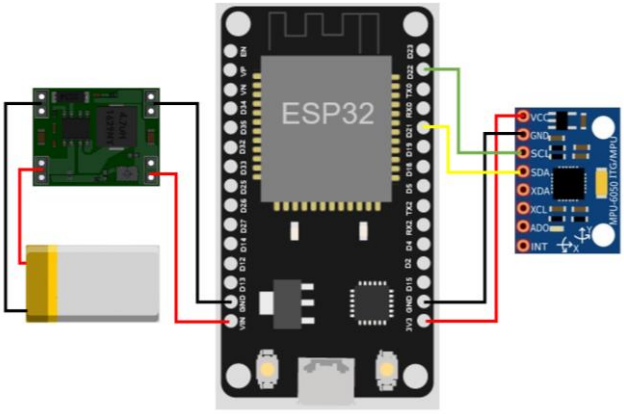

Figura 11. Diagrama del Sistema Electrónico 2 (MPU6050)

3.4. TÉCNICAS DE RECOLECCIÓN DE DATOS

En esta investigación se ha realizado un análisis de varias fuentes de información fiables y científicas tales como: artículos de IEEE, revistas digitales, páginas de prótesis de brazo y publicaciones científicas que realizan análisis de ondas EEG con sus respectivos dispositivos; conectando una interfaz BCI entre estos dispositivos con una aplicación, se ha obtenido la información necesaria de cada dispositivo electrónico haciendo uso de la hoja de datos técnicos de los mismos con el fin de conocer las distintas características para el óptimo funcionamiento del prototipo.

3.4.1. OPERACIONALIZACIÓN DE VARIABLES

VARIABLE DEPENDIENTE

Nombre	Descripción	Indicador	Métodos e instrumentos
Número de aciertos del prototipo	Corresponde al número de instrucciones ejecutadas correctamente	Numero entero	Observación

VARIABLES INDEPENDIENTE

Nombre	Descripción	Indicador	Métodos e instrumentos
Tiempo de respuesta	Corresponde al tiempo máximo que tiene el usuario para ejecutar un comando	Segundos	Cronometro
Postura del usuario	Corresponde a la posición del cuerpo que adopta el usuario mientras ejecuta los comandos	Nominal: Senado-Parado	Observación

3.5. POBLACIÓN DE ESTUDIO Y TAMAÑO DE MUESTRA

3.5.1. Población

En este proyecto, la población fue constituida por los 64 datos del número de aciertos del usuario al momento de ejecutar una instrucción. El experimento se llevó a cabo durante 8 días en dos sesiones, una en la mañana y una en la tarde, tato en la posición corporal sentado y parado.

3.5.2. Muestra

Conociendo el total de la población, se obtuvo el tamaño de la muestra para una población finita, a partir de la siguiente fórmula:

$$n = \frac{k^2 * N * q * p}{e^2 * (N-1) + k^2 * q * p}$$

Siendo:

N=64 datos

k=95%

p=50%

q=50%

e=5%

$$n = \frac{1.96^2 * 64 * 0.5 * 0.5}{0.05^2 * (64 - 1) + 1.96^2 * 0.5 * 0.5}$$

$$n = 54.98$$

$$n \approx 55$$

Los datos para la muestra se extraen de forma aleatoria.

3.6. PROCESO DE ADQUISICION DE DATOS

3.6.1. Procedimiento de Comunicación

En primera instancia se creó una cuenta en EMOTIV App con su respectivo Emotiv-ID, el mismo que genera un client-ID y un client secret, necesarios para el desarrollo de la presente aplicación.

El procedimiento para ejecutar en Cortex 2.0 de Emotiv se realiza una sola vez y se lo describe a continuación:

- Escribir las credenciales en la aplicación de Emotiv, como se muestra en la figura 12.

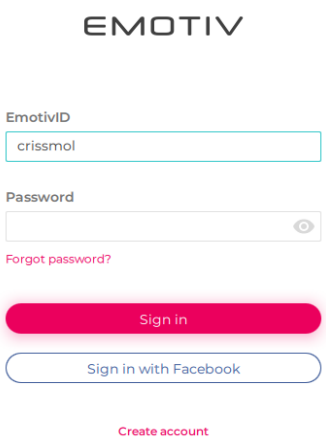

Figura 12. Aplicación Emotiv.

- Agregar en el código del programa en Python las credenciales: client-ID, client secret e identificación de la diadema Emotiv-Insight.
- Pedir un requerimiento de acceso desde el código del programa (request access), luego aceptarlo desde la aplicación EMOTIV.
- Encender la diadema Emotiv-Insight y emparejar el bluetooth del dongle USB en Lattepanda.
- La miniPC reconoce si existe un dispositivo EEG, luego lo conecta.
- Requerir desde el programa principal la autorización para obtener un "token".
- Crea una nueva sesión.

3.6.2. Software de adquisición de datos

Para realizar la interfaz BCI se utilizó el lenguaje de programación Python, de acuerdo a los algoritmos del Cortex versión 2.0 para comunicar la diadema Emotiv Insight y el minicomputador LattePanda. Así mismo, se programó en el coprocesador del mini pc en el IDE de Arduino la comunicación serial, el control de los servomotores y la lectura del sensor infrarrojo de temperatura en base al algoritmo que se explica a continuación, figura 13:

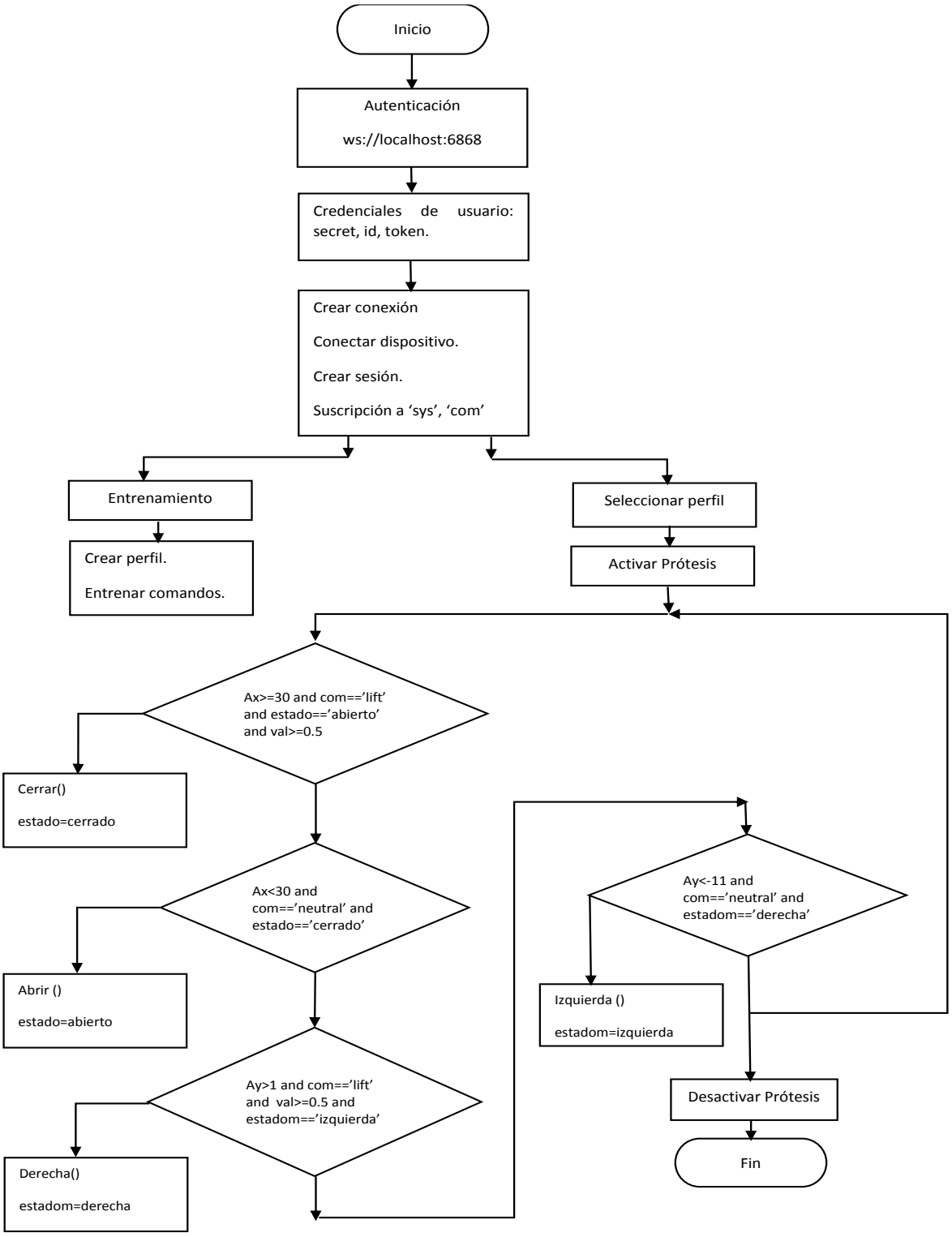

Figura 13. Algoritmo adquisición de datos entre Lattepanda y Emotiv Insight.

3.6.3. Software transmisor de datos del giroscopio MPU6050

Un programa en micropython fue creado para mantener el movimiento activado de los comandos mentales, este permite leer los valores medidos del giroscopio y acelerómetro MPU6050; para posteriormente enviarlos mediante comunicación wifi al minicomputador Lattepanda.

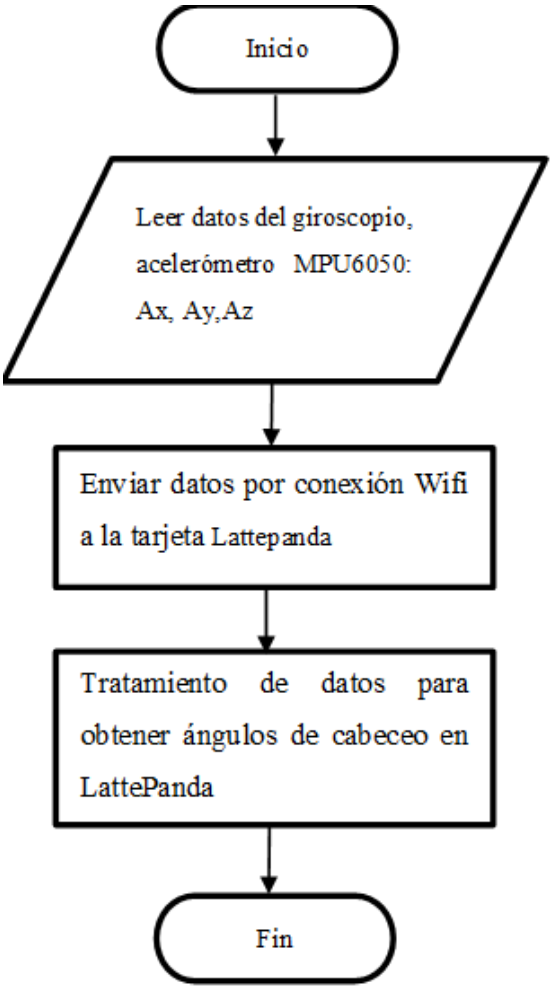

Figura 14. Algoritmo Transmisor de datos del giroscopio MPU6050.

3.6.4. Software de Control de microservomotores y lectura de temperatura

Desde el coprocesador de Arduino se programaron los movimientos de la mano; abducción, extensión y giro de muñeca; el control se lo realiza haciendo uso de la comunicación serial por puerto COM hacia la mini PC LattePanda.

Para la lectura de temperatura se utilizó el sensor infrarrojo MLX90614 que permite medir la temperatura a una distancia determinada sin necesidad de contacto; adicionalmente, una cinta LED RGB cambia de color dependiendo de la variación de temperatura.

El siguiente diagrama de flujo de la figura describe el código programado en Arduino correspondiente a la salida y entrada de datos para el control de servomotores y detección de temperatura, figura 15:

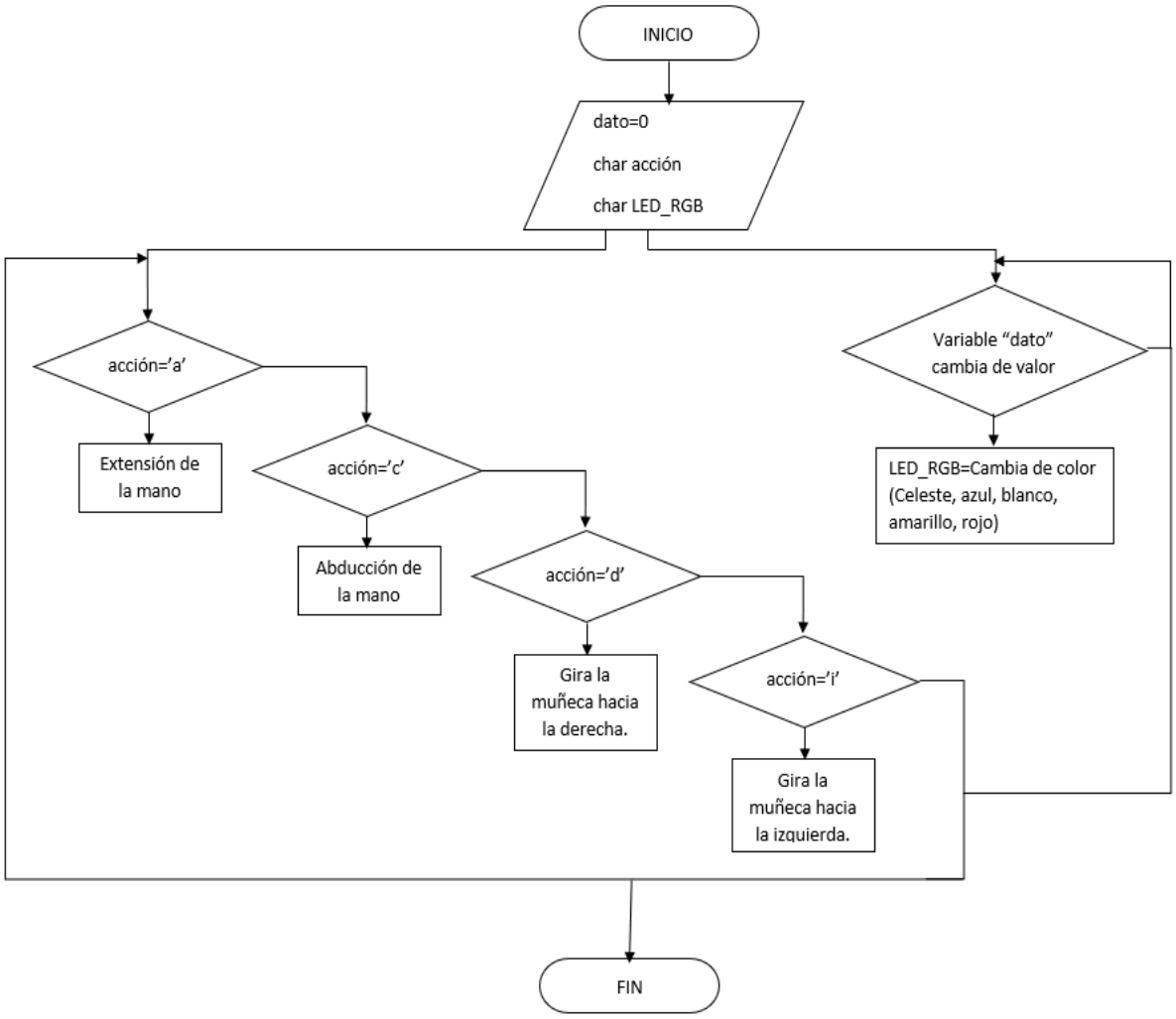

Figura 15. Algoritmo Control de micro servomotores y lectura de temperatura.

CAPÍTULO IV. RESULTADOS Y DISCUSIÓN

Para evaluar el funcionamiento, el dispositivo implementado fue sometido a pruebas de agarre durante 30 días. Estas pruebas se desarrollaron considerando diferentes circunstancias tales como: horario (mañana y tarde), tiempo de espera respuesta (5, 10 y 15s), posición del cuerpo del usuario (sentado y parado), tipo de objetos (cautín, manzana y helado).

4.1. Prueba de flexión y extensión

La figura 16 muestra el diagrama de cajas correspondiente al número de aciertos en función del tiempo de espera de ejecución del comando. Los tiempos de espera se establecieron en 5, 10 y 15 segundos.

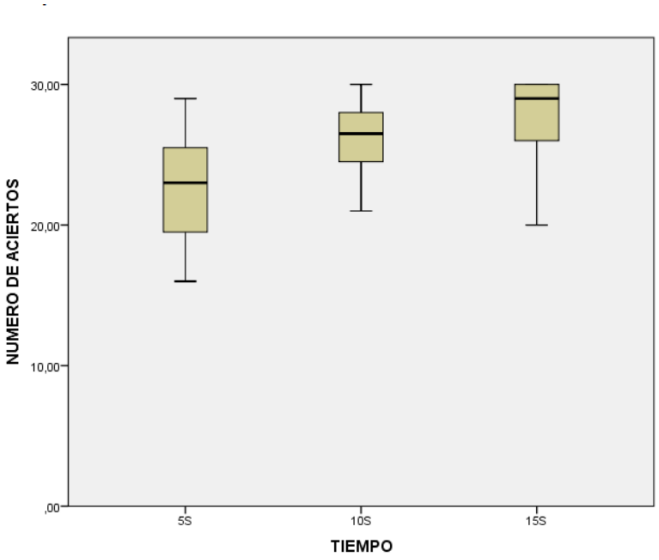

Figura 16. Diagrama de cajas: Número de aciertos en función del tiempo.

Para conocer si los datos de la muestra corresponden o no a una distribución normal, se procedió a realizar la prueba de normalidad de Shapiro-Wilk, partiendo de la fórmula:

$$W = \frac{\left(\Sigma_{j=1}^{n} a_i x_{(i)}\right)^2}{\Sigma_{i=1}^{n}(x_i - \bar{x})^2}$$

Donde:

x_i: Son valores de muestra aleatorios ordenados
a_i: Son constantes generadas a partir de las covarianzas, varianzas y medias de la muestra de una muestra normalmente distribuida.
$\bar{x}$: Media muestral

La hipótesis planteada es la siguiente:

H_0: Los datos provienen de una distribución normal.
H_1: Los datos no provienen de una distribución normal.

La tabla 6, muestra los resultados de la prueba de hipótesis. Para 5s y 10s, el p-valor (Sig.) es mayor que 0.05 por lo tanto se acepta la hipótesis nula es decir los datos se distribuyen normalmente, sin embargo, para 15s el p-valor es menor que 0.05 por tanto se rechaza la hipótesis nula, es decir los datos no se distribuyen normalmente. En este sentido se aplica una prueba de hipótesis no paramétrica.

Tabla 5. *Pruebas de Normalidad con respecto al tiempo.*

PRUEBAS DE NORMALIDAD							
TIEMPO		**Kolmogorov-Smirnov**			**Shapiro-Wilk**		
		Estadístico	**gl**	**Sig.**	**Estadístico**	**gl**	**Sig.**
NÚMERO DE ACIERTOS	**5S**	0.114	20	0.2	0.966	20	0.671
	10S	0.159	20	0.2	0.949	20	0.346
	15S	0.244	20	0.003	0.826	20	0.002

La hipótesis planteada es la siguiente:

H_0: las medianas del número de aciertos en la ejecución de los comandos son iguales con los diferentes tiempos de espera.
H_1: las medianas del número de aciertos en la ejecución de los comandos no son iguales (al menos una) con los diferentes tiempos de espera.

Para determinar si existe una diferencia significativa en las medias del número de aciertos en la ejecución de los comandos con diferentes tiempos de espera, se aplicó una prueba Anova cuyos resultados se muestran en la tabla 7. En este caso, el p-valor es 0.0 por lo tanto se rechaza la hipótesis nula y se acepta la alternativa, es decir al menos una de las medias es diferente.

Tabla 6. Pruebas de Normalidad con respecto al tiempo.

	Suma de cuadrados	**gl**	**Media cuadrática**	**F**	**Sig.**
Entre grupos	243,233	2	121,617	12,661	,000
Dentro de grupos	547,500	57	9,605		
Total	790,733	59			

Para determinar cuál de las medias es diferente, se aplicó la prueba de comparaciones múltiples de Tukey. Los resultados que se muestran en la tabla 7 revelan que para 10 y 15

segundos el número de aciertos es el mismo, 26 en promedio sobre un máximo de 30, mientras que para 5 segundos es diferente. Así mismo, la figura 17 ratifica no solo que para 5 segundos la media es diferente si no que es menor, en promedio 22 sobre 30.

Tabla 7. *Test de Tukey.*

TIEMPO	N	Subconjunto para alfa = 0.05	
		1	2
5S	55	22,7500	
10S	55		25,9500
15S	55		27,6000
Sig.		1,000	,220

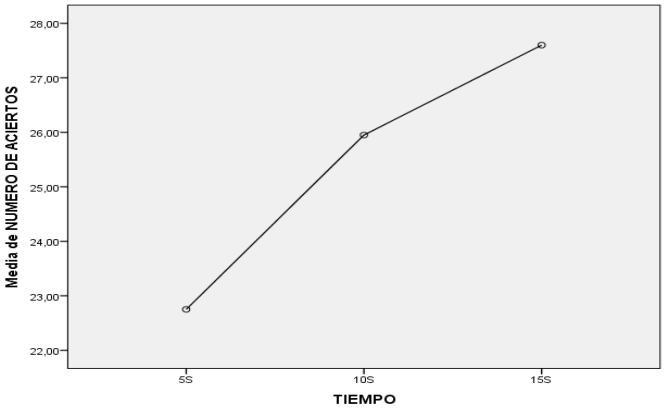

Figura 17. Gráfica de medias del número de aciertos en función del tiempo.

Por otro lado, la figura 18 muestra el diagrama de cajas del número de aciertos en función de la posición del cuerpo durante el experimento, en este caso sentado y parado.

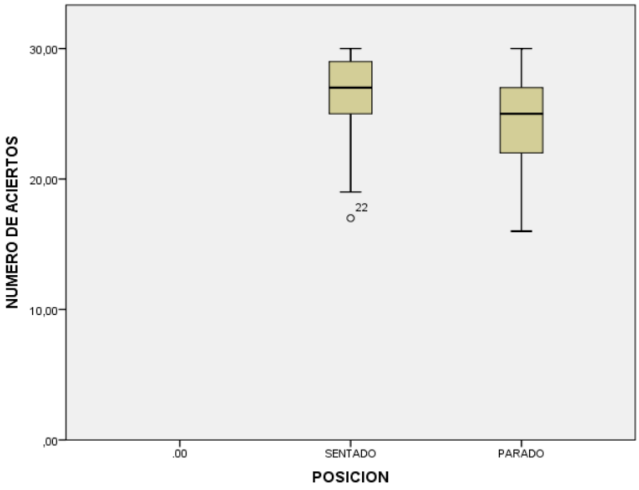

Figura 18. Diagrama de cajas: Número de aciertos en función del de la posición.

Así mismo, se aplicó la prueba de normalidad de Shapiro-Wilk para determinar si los datos corresponden a una distribución normal, los resultados se muestran en la tabla 8. Como se puede apreciar, el p-valor de la posición sentado en menor que 0.05 por lo tanto, los datos no se distribuyen normalmente, mientras que, para la posición parado, el p-valor es mayor por lo que los datos si se distribuyen normalmente.

Tabla 8. *Pruebas de Normalidad con respecto a la posición.*

PRUEBAS DE NORMALIDAD							
POSICIÓN		**Kolmogorov-Smirnov**			**Shapiro-Wilk**		
		Estadístico	**gl**	**Sig.**	**Estadístico**	**gl**	**Sig.**
NÚMERO DE ACIERTOS	**SENTADO**	0.188	30	0.008	0.878	30	0.002
	PARADO	0.111	30	0.2	0.959	30	0.296

La tabla 9 muestra los resultados de la prueba de hipótesis no paramétrica. Tomando en cuenta que el p-valor calculado es mayor a 0.05 se acepta la hipótesis nula, es decir, no hay una diferencia significativa en el número de aciertos al cambiar la posición mientras se ejecutan los comandos. El promedio de aciertos es de 25 sobre un total de 30 intentos.

Tabla 9. *Prueba de hipótesis respecto a la posición con un nivel de significación de 0.05.*

Hipótesis	Prueba	Sig.	Decisión
La distribución de **NUMERO DE ACIERTOS** es la misma entre las categorías **POSICION**	Prueba U de Mann-Whitney para muestras independientes.	0.073	Conserva la hipótesis nula.

4.2. Prueba de lectura de temperatura

Para verificar si el prototipo mide la temperatura correctamente, se realizó una comparativa con un termómetro comercial. Se consideraron un total de 120 mediciones de temperatura tomadas de tres objetos diferentes (helado, manzana y un cautín), tanto con el prototipo implementado y con el termómetro. La figura 19, muestra el diagrama de cajas de la temperatura en función del dispositivo utilizado para medirla.

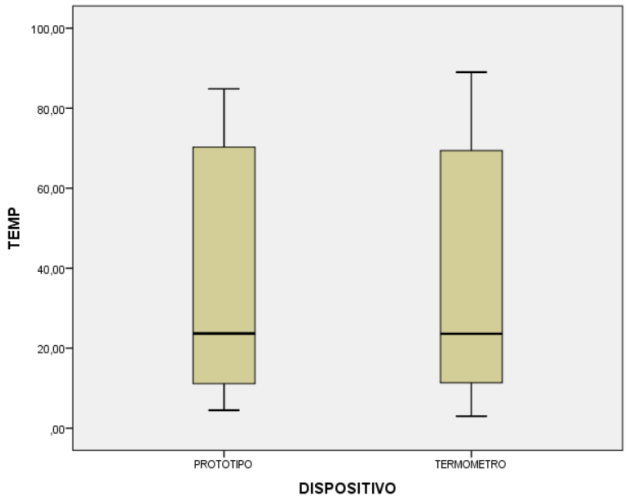

Figura 19. Diagrama de cajas: Temperatura en función del tipo de dispositivo utilizado para medir.

Igual que antes, también se aplicó la prueba de normalidad de los datos como se puede apreciar en la tabla 10. En ambos casos, el p-valor es menor que 0.05 por lo tanto, las distribuciones no son normales y es necesario aplicar una prueba de hipótesis no paramétrica.

Tabla 10. *Pruebas de Normalidad con respecto al dispositivo de lectura.*

		Kolmogorov-Smirnov[a]			Shapiro-Wilk		
	DISPOSITIVO	Estadístico	gl	Sig.	Estadístico	gl	Sig.
TEMP	PROTOTIPO	,325	120	,000	,776	120	,000
	TERMOMETRO	,331	120	,000	,785	120	,000

Finalmente, al aplicar la prueba de hipótesis se obtuvo un p-valor de 0.819 que es mayor a 0.05, lo que indica que no hay diferencia significativa en la medición de temperatura al utilizar el prototipo y el termómetro comercial, es decir que el dispositivo implementado funciona correctamente.

Tabla 11. *Prueba de hipótesis respecto al dispositivo con un nivel de significación de 0.05.*

Hipótesis	Prueba	Sig.	Decisión
La distribución de **TEMPERATURA** es la misma entre las categorías **DISPOSITIVO**	Prueba U de Mann-Whitney para muestras independientes.	0.819	Conserva la hipótesis nula.

CAPÍTULO V. CONCLUSIONES Y RECOMENDACIONES

5.1. Conclusiones.

- En este trabajo se construyó un prototipo de prótesis utilizando la tecnología de impresión 3D como una solución no invasiva para personas con pérdida de miembro superior por debajo del codo. Este sistema funciona por medio de estímulos endógenos proporcionados por ondas EEG del usuario y permite controlar movimientos de flexión y extensión de la mano, adicionalmente, brinda la percepción de temperatura a través de indicadores de color.

- El funcionamiento del prototipo implementado fue validado a través de pruebas de campo en un ambiente controlado en dos escenarios, el primero con tres tiempos de espera para la ejecución de un comando, 5, 10 y 15 segundos y el segundo con diferentes posiciones del cuerpo, sentado y parado; la variable de experimentación fue el número de aciertos en la ejecución de la instrucción. A través de un análisis estadístico y pruebas de hipótesis se determinó que con un tiempo espera de 5 segundos se obtiene un promedio menor de aciertos 22 sobre un máximo de 30, mientras que para los tiempos de 10 y 15 segundos el promedio fue de 26 sobre 30. Así mismo, se encontró que la posición del usuario no afecta al número de aciertos al momento de ejecutar el comando alcanzando un promedio de 25 acierto sobre 30 tanto parado como sentado.

- Finalmente, para determinar si el prototipo implementado mide correctamente la temperatura, los datos adquiridos fueron comparados con los datos tomados con un termómetro comercial. La prueba de hipótesis arrojó que no existe diferencia entre las mediciones realizadas por ambos dispositivos validando de esta manera el funcionamiento el sistema propuesto.

5.2. Recomendaciones.

- Calibrar y humedecer con gel los sensores del dispositivo Emotiv Insight antes de su operación.

- Colocar los sensores de la diadema en la zona que pertenece para una buena señal y lectura correcta.

- Calibrar adecuadamente el sensor de temperatura MLX90614.

- Utilizar redes neuronales en la etapa de entrenamiento para reducir el tiempo de adaptación al prototipo.

- Utilizar dos fuentes de energía DC por separado para la etapa de potencia y la minicomputadora respectivamente.

- Imprimir con filamento TPU los distales y filamento de fibra de carbono el soporte del muñón del usuario, puesto que soportaría mejor los impactos o caídas.

BIBLIOGRAFÍA

Ackerman, E. (2012). Bebionic3 Cyborg Hand. *IEEE SPECTRUM*.

Bionics, M. (2021). *Mobius Bionics*. Obtenido de https://www.mobiusbionics.com/luke-arm/

Bionics, O. (2018). *Open Bionics*. Obtenido de https://openbionics.com/hero-arm/

Campillo, S. (17 de junio de 2015). *Así es la vida de alguien con una mano biónica*. Obtenido de https://hipertextual.com/2015/06/bebionic

COMERCIO, E. (6 de julio de 2020). *LÍDERES*. Obtenido de LÍDERES: https://www.revistalideres.ec/lideres/protesis-robotica-costo-universidad-chimborazo.html

Cueva, E. (2015). Crean Mano Robótica cubierta de piel en Ecuador. *EL UNIVERSO*.

DeTOP. (2016). *DeTOP*. Obtenido de Summary and Figures: http://www.detop-project.eu/summary-and-figures/

Ecuaprótesis 3D. (2021). Ecuaprótesis 3D Prótesis y Órtesis. *Catálogo de Productos*, 12.

Fillauer. (2020). *Fillauer* . Obtenido de https://fillauer.com/products/taska-hand-with-quick-disconnect-wrist/

ICHI.PRO. (2015). *ICHI.PRO*. Obtenido de Cyborg Hand de código abierto hace que las prótesis sean más accesibles que nunca: https://ichi.pro/es/31373037393034303335

Industries, C. P. (2021). *College Park Industries.* Obtenido de https://www.college-park.com/hy5-myhand

INEC. (23 de Marzo de 2021). *Instituto Nacional de Estadística y Censos.* Obtenido de Instituto Nacional de Estadística y Censos: www.ecuadorencifras.gob.ec/documentos/web-inec/Estadisticas_Sociales/Camas_Egresos_Hospitalarios/Cam_Egre_Hos_2018/Tabulados_series_ECEH_2018.xlsx

Interface, B. (2020). *Brane Interface*. Obtenido de http://braneinterface.com/

Kaku, M. (2014). *The Future of the Mind.* New York: Doubleday.

México, A. N. (2016). *Los Amputados y su Rehabilitación, un reto para el estado.* México D.F: Intersistemas.

Moncrieffe, M. V. (2017). EL INVENTO PROTÉSICO MÁS NUEVO DE APL PERMITE A LOS USUARIOS RECUPERAR SU SENTIDO DEL TACTO. *Johns Hopkins Magazine*.

Neuralink. (2021). *Neuralink*. Obtenido de https://neuralink.com/

Ortosur. (2018). *Ortosur*. Obtenido de Ortosur: https://www.ortosur.es/catalogo-de-productos/protesis/miembro-superior/mano-mioelectrica/vincent-evolution/

Össur. (25 de Marzo de 2021). *Össur*. Obtenido de https://www.ossur.com/en-us/prosthetics/arms/i-limb-quantum

Össur. (1 de Abril de 2021). *Össur Life whitout limitations*. Obtenido de https://www.ossur.com/en-us/prosthetics/hands

Ottobock. (2014). *Ottobock*. Obtenido de https://www.ottobock.es/protesica/miembro-superior/sistemas-de-brazo-y-mano/bebionic/

Proteus, C. d.-E. (s.f.). *PROTÉUS*. Obtenido de http://www.proteus-ec.com/protesis/

Puchades, A. (2003). *La mano, admirable don del hombre.* Editorial del Cardo.

Rayome, A. D. (6 de Enero de 2020). *CNET.* Obtenido de You can move this prosthetic hand with your mind: https://www.cnet.com/news/cut-your-own-hair-with-this-28-haircut-kit-the-lowest-price-ever/

UTN. (2016). *Carrera de Ingeniería Mecatrónica UTN*. Obtenido de https://www.utn.edu.ec/fica/carreras/mecatronica/?p=1860

WARWICK. (14 de Noviembre de 2019). *WARWICK News y Events.* Obtenido de https://warwick.ac.uk/newsandevents/pressreleases/bionic_hand_made/

ANEXOS

ANEXO 1. CÓDIGO PARA CREAR CONEXIÓN

```
import json
import ssl
from websocket import create_connection

receivedData = create_connection("wss://localhost:6868", sslopt={"cert_reqs": ssl.CERT_NONE})
def getToken(id,secret):
    try:

        receivedData.send(json.dumps(
        {
            "id": 1,
            "jsonrpc": "2.0",
            "method": "requestAccess",
            "params": {
                "clientId": id,
                "clientSecret": secret
            }
        }
        ))
        print("\n                Bienvenido a Prótesis Mental")
        print("Acceso: "+json.loads(receivedData.recv())["result"]["message"])
        receivedData.send(json.dumps({
            "id": 3,
            "jsonrpc": "2.0",
            "method": "authorize",
            "params": {
                "clientId": id,
                "clientSecret": secret
            }

        }))
        token = json.loads(receivedData.recv())["result"]["cortexToken"]
        print("\nToken para EMOTIV-Insight:\n\n"+str(token))
        return token
    except KeyError as result:
        pass
```

ANEXO 2. CÓDIGO ADQUISICIÓN DE DATOS ENTRE LATTEPANDA Y EMOTIV INSIGHT

```
from tkinter import *
from tkinter.ttk import *
from tkinter import ttk
import tkinter
import json
from websocket import create_connection
from fysom import *
import ssl
#from pyfirmata import Arduino
import time
import serial
import requests
import autorizacion
import numpy as np
from array import *
import os
import threading
import continuous_threading
from matplotlib.figure import Figure
from matplotlib.backends.backend_tkagg import (FigureCanvasTkAgg,
NavigationToolbar2Tk)
from PIL import Image, ImageTk
from time import sleep
import socket
import ast
import math
arduinoPort = serial.Serial('COM20', 9600)
val1 = 0
index = []
s=socket.socket()
s.connect(("192.168.137.47",2020))

def readserial():
    global val1
    ser_bytes = arduinoPort.readline()
    ser_bytes = ser_bytes.decode("utf-8")
    #print(type(ser_bytes.rstrip()))
    val1 = ser_bytes
    index.append(val1)
    if len(index) == 2:
```

```
        #display1 = tk.Label(root,text=index[0]).place(x=10,y=10)
        display2 = Label(ventana,text=index[1], font=("Helvetica", 30)).place(x=600,y=75)
    if len(index) == 2:
        #print("Done")
        index.clear()

    time.sleep(0.2)

t1 = continuous_threading.PeriodicThread(0.05, readserial)

#Maquina de estado MANO
mano = Fysom({'initial': 'abierto',
              'final': 'red',
              'events': [
                  {'name': 'cerab', 'src': 'cerrado', 'dst': 'abierto'},
                  {'name': 'abcer',  'src': 'abierto',   'dst': 'cerrado'}]})
##Maquina de estado muñeca
mun = Fysom({'initial': 'izquierda',
              'final': 'red',
              'events': [
                  {'name': 'deriz', 'src': 'derecha', 'dst': 'izquierda'},
                  {'name': 'izder',  'src': 'izquierda',   'dst': 'derecha'}]})
def cerrar():
    arduinoPort.write(str('c').encode())
def abrir():
    arduinoPort.write(str('a').encode())

def derecha():
    arduinoPort.write(str('d').encode())
def izquierda():
    arduinoPort.write(str('i').encode())
#credenciales
secret='VFMk8PsviQM1iND4xGQgTcdPf4GEGYbMlUzkogmpYteXbZ9xCr2bWxeA39f
uYZBU1c1JMcZGgB8LqeeOhr5bZzsDlwmRQtWvfborAzayNzXx4h7hDiaEfyj1Irx6buis'
id='cQQVfLWK6yAdvYZDi7pGCoaYeTt3wdIw6HPzLxjI'
TOKEN=autorizacion.getToken(id,secret)
# Crear objeto-clase create_connection
ws = create_connection("wss://localhost:6868", sslopt={"cert_reqs": ssl.CERT_NONE})
#Conectar Emotiv Insight
ws.send(json.dumps({
    "id": 2,
    "jsonrpc": "2.0",
    "method": "controlDevice",
```

```
    "params": {
        "command": "connect",
        "headset": "INSIGHT-5A68CBD2"
    }
}))
info=json.loads(ws.recv())["result"]["message"]
print("\nInformación del Dispositivo: "+info)

#Crear sesion con Emotiv Insight
ws.send(json.dumps({
    "id": 3,
    "jsonrpc": "2.0",
    "method": "createSession",
    "params": {
        "cortexToken": TOKEN,
        "headset": "INSIGHT-5A68CBD2",
        "status": "open"
    }
}))
print("\nCreación de Sesión:")
sid=json.loads(ws.recv())["result"]["id"]
print(sid)

#Suscripción
ws.send(json.dumps({
    "id": 4,
    "jsonrpc": "2.0",
    "method": "subscribe",
    "params": {
        "cortexToken": TOKEN,
        "session": sid,
        "streams": ["sys"]
        }
    }
                ))
print("Suscrito a Sistema de Eventos: " + ws.recv())

#query Perfil
def infoper():
    ws.send(json.dumps({
        "id": 5,
        "jsonrpc": "2.0",
        "method": "queryProfile",
```

```
        "params": {
            "cortexToken": TOKEN
            }
        }))
def cargarpn():
    ws.send(json.dumps({
        "id": 3,
        "jsonrpc": "2.0",
        "method": "setupProfile",
        "params": {
            "cortexToken": TOKEN,
            "profile": combo.get(),
            "headset": "INSIGHT-5A68CBD2",
            "status": "load"
            }
        }))
    print(ws.recv())


def unload():
    ws.send(json.dumps({
        "id": 3,
        "jsonrpc": "2.0",
        "method": "setupProfile",
        "params": {
            "cortexToken": TOKEN,
            "profile": combo.get(),
            "headset": "INSIGHT-5A68CBD2",
            "status": "unload"
            }
        }))
    print(ws.recv())

def umbralent():
    ws.send(json.dumps({
        "id": 1,
        "jsonrpc": "2.0",
        "method": "mentalCommandTrainingThreshold",
        "params": {
            "cortexToken": TOKEN,
            "session": sid
            }
        }))
```

```
    umbral=json.loads(ws.recv())["result"]
    valor=umbral.get("lastTrainingScore")
    print("Umbral de Entrenamiento: "+str(valor))
    im = Image.open(r'C:\Users\LattePanda\Documents\Programacion Tesis\programacion
mentepro1\mano.gif'); img = ImageTk.PhotoImage(im)
    delay = float(im.info['duration'])/600; # Delay used in the GIF file 800
    lbl = Label(image=img); lbl.place(x=600,y=200)

    if(valor>=0.75):
        for frame in range(0,4,1):
            sleep(delay);
            im.seek(frame); img = ImageTk.PhotoImage(im)
            lbl.config(image=img); ventana.update()

    elif(valor<0.75 and valor>=0.5):
        for frame in range(0,3,1):
            sleep(delay);
            im.seek(frame); img = ImageTk.PhotoImage(im)
            lbl.config(image=img); ventana.update()
    elif(valor<0.5 and valor>=0.25):
        for frame in range(0,2,1):
            sleep(delay);
            im.seek(frame); img = ImageTk.PhotoImage(im)
            lbl.config(image=img); ventana.update()
    elif(valor<0.25):
        for frame in range(0,1,1):
            sleep(delay);
            im.seek(frame); img = ImageTk.PhotoImage(im)
            lbl.config(image=img); ventana.update()

def ratinglift():
    ws.send(json.dumps({
        "id": 1,
        "jsonrpc": "2.0",
        "method": "mentalCommandGetSkillRating",
        "params": {
            "cortexToken": TOKEN,
            "session": sid,
            "action": "lift"
            }
        }))

    valor=json.loads(ws.recv())["result"]
```

```
    print("Rating: "+str(valor))

    im = Image.open(r'C:\Users\LattePanda\Documents\Programacion Tesis\programacion mentepro1\mano.gif'); img = ImageTk.PhotoImage(im)
    delay = float(im.info['duration'])/600; # Delay used in the GIF file 800
    lbl = Label(image=img); lbl.place(x=600,y=200)

    if(valor>=0.75):
        for frame in range(0,4,1):
            sleep(delay);
            im.seek(frame); img = ImageTk.PhotoImage(im)
            lbl.config(image=img); ventana.update()

    elif(valor<0.75 and valor>=0.5):
        for frame in range(0,3,1):
            sleep(delay);
            im.seek(frame); img = ImageTk.PhotoImage(im)
            lbl.config(image=img); ventana.update()
    elif(valor<0.5 and valor>=0.25):
        for frame in range(0,2,1):
            sleep(delay);
            im.seek(frame); img = ImageTk.PhotoImage(im)
            lbl.config(image=img); ventana.update()
    elif(valor<0.25):
        for frame in range(0,1,1):
            sleep(delay);
            im.seek(frame); img = ImageTk.PhotoImage(im)
            lbl.config(image=img); ventana.update()


def ratingneu():
    ws.send(json.dumps({
        "id": 1,
        "jsonrpc": "2.0",
        "method": "mentalCommandGetSkillRating",
        "params": {
            "cortexToken": TOKEN,
            "session": sid,
            "action": "neutral"
            }
        }))
    valor=json.loads(ws.recv())["result"]
    print("Rating: "+str(valor))
```

```
    im = Image.open(r'C:\Users\LattePanda\Documents\Programacion Tesis\programacion
mentepro1\mano.gif'); img = ImageTk.PhotoImage(im)
    delay = float(im.info['duration'])/600; # Delay used in the GIF file 800
    lbl = Label(image=img); lbl.place(x=600,y=200)

    if(valor>=0.75):
        for frame in range(0,4,1):
            sleep(delay);
            im.seek(frame); img = ImageTk.PhotoImage(im)
            lbl.config(image=img); ventana.update()

    elif(valor<0.75 and valor>=0.5):
        for frame in range(0,3,1):
            sleep(delay);
            im.seek(frame); img = ImageTk.PhotoImage(im)
            lbl.config(image=img); ventana.update()
    elif(valor<0.5 and valor>=0.25):
        for frame in range(0,2,1):
            sleep(delay);
            im.seek(frame); img = ImageTk.PhotoImage(im)
            lbl.config(image=img); ventana.update()
    elif(valor<0.25):
        for frame in range(0,1,1):
            sleep(delay);
            im.seek(frame); img = ImageTk.PhotoImage(im)
            lbl.config(image=img); ventana.update()

def ratingrig():
    ws.send(json.dumps({
        "id": 1,
        "jsonrpc": "2.0",
        "method": "mentalCommandGetSkillRating",
        "params": {
            "cortexToken": TOKEN,
            "session": sid,
            "action": "rotateClockwise"
            }
        }))
    valor=json.loads(ws.recv())["result"]
    print("Rating: "+str(valor))
```

```
    im = Image.open(r'C:\Users\LattePanda\Documents\Programacion Tesis\programacion mentepro1\mano.gif'); img = ImageTk.PhotoImage(im)
    delay = float(im.info['duration'])/600; # Delay used in the GIF file 800
    lbl = Label(image=img); lbl.place(x=600,y=200)

    if(valor>=0.75):
        for frame in range(0,4,1):
            sleep(delay);
            im.seek(frame); img = ImageTk.PhotoImage(im)
            lbl.config(image=img); ventana.update()

    elif(valor<0.75 and valor>=0.5):
        for frame in range(0,3,1):
            sleep(delay);
            im.seek(frame); img = ImageTk.PhotoImage(im)
            lbl.config(image=img); ventana.update()
    elif(valor<0.5 and valor>=0.25):
        for frame in range(0,2,1):
            sleep(delay);
            im.seek(frame); img = ImageTk.PhotoImage(im)
            lbl.config(image=img); ventana.update()
    elif(valor<0.25):
        for frame in range(0,1,1):
            sleep(delay);
            im.seek(frame); img = ImageTk.PhotoImage(im)
            lbl.config(image=img); ventana.update()

def ratingizq():
    ws.send(json.dumps({
        "id": 1,
        "jsonrpc": "2.0",
        "method": "mentalCommandGetSkillRating",
        "params": {
            "cortexToken": TOKEN,
            "session": sid,
            "action": "rotateCounterClockwise"
            }
        }))
    valor=json.loads(ws.recv())["result"]
    print("Rating: "+str(valor))

    im = Image.open(r'C:\Users\LattePanda\Documents\Programacion Tesis\programacion mentepro1\mano.gif'); img = ImageTk.PhotoImage(im)
```

```
    delay = float(im.info['duration'])/600; # Delay used in the GIF file 800
    lbl = Label(image=img); lbl.place(x=600,y=200)

    if(valor>=0.75):
        for frame in range(0,4,1):
            sleep(delay);
            im.seek(frame); img = ImageTk.PhotoImage(im)
            lbl.config(image=img); ventana.update()

    elif(valor<0.75 and valor>=0.5):
        for frame in range(0,3,1):
            sleep(delay);
            im.seek(frame); img = ImageTk.PhotoImage(im)
            lbl.config(image=img); ventana.update()
    elif(valor<0.5 and valor>=0.25):
        for frame in range(0,2,1):
            sleep(delay);
            im.seek(frame); img = ImageTk.PhotoImage(im)
            lbl.config(image=img); ventana.update()
    elif(valor<0.25):
        for frame in range(0,1,1):
            sleep(delay);
            im.seek(frame); img = ImageTk.PhotoImage(im)
            lbl.config(image=img); ventana.update()

def cerebromap():
    #cargarpn()
    ws.send(json.dumps({
        "id": 5891714,
        "jsonrpc": "2.0",
        "method": "mentalCommandBrainMap",
        "params": {
            "cortexToken": TOKEN,
            "profile": combo.get(),
            "session": sid
            }
        }))
    coordenadas=json.loads(ws.recv())["result"]
    n1=coordenadas[0].get("coordinates")[0]
    n2=coordenadas[0].get("coordinates")[1]
    r1=coordenadas[1].get("coordinates")[0]
    r2=coordenadas[1].get("coordinates")[1]
    l1=coordenadas[2].get("coordinates")[0]
```

```
    l2=coordenadas[2].get("coordinates")[1]
    print(n1, n2)
    print(r1, r2)
    print(l1, l2)

    fig = Figure(figsize = (2, 2),
                 dpi = 100)

    plot1 = fig.add_subplot(111)
    plot1.plot(n1,n2,'bo', r1,r2,'co',l1,l2,'go')
    canvas = FigureCanvasTkAgg(fig,
                               master = ventana)
    canvas.draw()
    canvas.get_tk_widget().place(x=300,y=325)

def crearperfiles():

    infoper()
    entrenamientos = json.loads(ws.recv())["result"]
    n=len(entrenamientos)
    perfn=[]
    for i in range(0,n,1):
        perfn.append(entrenamientos[i].get("name"))
    perfc=np.append(perfn,E1.get())
    listaperfc=perfc.tolist()
    ws.send(json.dumps({
        "id": 2,
        "jsonrpc": "2.0",
        "method": "setupProfile",
        "params": {
            "cortexToken": TOKEN,
            "profile": E1.get(),
            "headset": "INSIGHT-5A68CBD2",
            "status": "create"
            }
        }))
    combo['values']= listaperfc

def acneutral():
    print("Entrenamiento accion neutral")
    print("Relájese, espere 3 segundos")
    for i in range(2,-1,-1):
        print(i)
```

```
        time.sleep(1)
    ws.send(json.dumps({
        "id": 4,
        "jsonrpc": "2.0",
        "method": "training",
        "params": {
            "action": "neutral",
            "cortexToken": TOKEN,
            "detection": "mentalCommand",
            "session": sid,
            "status": "start"
            }
        }))
    print(ws.recv())
    time.sleep(5)
    print(ws.recv())

    for var in range(3,-1,-1):
        Lt=Label(ventana, textvariable=str(var)).place(x=600,y=300)
        time.sleep(1)
    #time.sleep(4)
    L8=Label(ventana, text="RELÁJESE COMPLETAMENTE", font=("Helvetica",
14)).place(x=625,y=350)
    style = ttk.Style()
    style.theme_use('default')
    style.configure("skyblue.Horizontal.TProgressbar", background='skyblue')

    #progress=ttk.Progressbar(ventana, orient=HORIZONTAL,
length=400,mode='determinate', maximum=100, value=1)
    progress=ttk.Progressbar(ventana,length=350, style='skyblue.Horizontal.TProgressbar')
    progress.place(x=600,y=400)


    for i in range(0,100,5):

        progress['value']+=5
        progress.update()
        time.sleep(1)



    print(ws.recv())
    #umbralent()
```

```
    ratingneu()

def aceptarneutral():
    ws.send(json.dumps({
        "id": 5,
        "jsonrpc": "2.0",
        "method": "training",
        "params": {
            "action": "neutral",
            "cortexToken": TOKEN,
            "detection": "mentalCommand",
            "session": sid,
            "status": "accept"
            }
        }))
    print(ws.recv())
    time.sleep(2)
    print(ws.recv())
    ws.send(json.dumps({
        "id": 6,
        "jsonrpc": "2.0",
        "method": "setupProfile",
        "params": {
            "cortexToken": TOKEN,
            "profile": combo.get(),
            "headset": "INSIGHT-5A68CBD2",
            "status": "save"
            }
        }))
    print(ws.recv())
    #ratinglift()
def rechazarneutral():
    ws.send(json.dumps({
        "id": 5,
        "jsonrpc": "2.0",
        "method": "training",
        "params": {
            "action": "neutral",
            "cortexToken": TOKEN,
            "detection": "mentalCommand",
            "session": sid,
            "status": "reject"
            }
```

```
        }))
    print(ws.recv())
    time.sleep(2)
    print(ws.recv())
    ws.send(json.dumps({
        "id": 6,
        "jsonrpc": "2.0",
        "method": "setupProfile",
        "params": {
            "cortexToken": TOKEN,
            "profile": combo.get(),
            "headset": "INSIGHT-5A68CBD2",
            "status": "save"
            }
        }))
    print(ws.recv())

def aclift():
    print("Entrenamiento para sujetar objetos")
    print("Concéntrese")
    for i in range(2,-1,-1):
        print(i)
        time.sleep(1)
    ws.send(json.dumps({
        "id": 4,
        "jsonrpc": "2.0",
        "method": "training",
        "params": {
            "action": "lift",
            "cortexToken": TOKEN,
            "detection": "mentalCommand",
            "session": sid,
            "status": "start"
            }
        }))
    print(ws.recv())
    time.sleep(5)
    print(ws.recv())

    for var in range(3,-1,-1):
        Lt=Label(ventana, textvariable=str(var)).place(x=500,y=300)
        time.sleep(1)
    #time.sleep(4)
```

```
    L8=Label(ventana, text="Concéntrese en la palabra 'CERRAR'", font=("Helvetica", 14)).place(x=625,y=350)
    progress=ttk.Progressbar(ventana, orient=HORIZONTAL, length=350,mode='determinate', maximum=100, value=1)
    progress.place(x=600,y=400)


    for i in range(0,100,5):
        #print(i)
        progress['value']+=5
        progress.update()
        time.sleep(1)
        #umbralent()
        #ratinglift()

    print(ws.recv())
    #umbralent()
    ratinglift()

def aceptarlift():
    ws.send(json.dumps({
        "id": 5,
        "jsonrpc": "2.0",
        "method": "training",
        "params": {
            "action": "lift",
            "cortexToken": TOKEN,
            "detection": "mentalCommand",
            "session": sid,
            "status": "accept"
            }
        }))
    print(ws.recv())
    time.sleep(2)
    print(ws.recv())
    ws.send(json.dumps({
        "id": 6,
        "jsonrpc": "2.0",
        "method": "setupProfile",
        "params": {
            "cortexToken": TOKEN,
            "profile": combo.get(),
            "headset": "INSIGHT-5A68CBD2",
```

```
            "status": "save"
            }
        }))
    print(ws.recv())
    #ratinglift()
def rechazarlift():
    ws.send(json.dumps({
        "id": 5,
        "jsonrpc": "2.0",
        "method": "training",
        "params": {
            "action": "lift",
            "cortexToken": TOKEN,
            "detection": "mentalCommand",
            "session": sid,
            "status": "reject"
            }
        }))
    print(ws.recv())
    time.sleep(2)
    print(ws.recv())
    ws.send(json.dumps({
        "id": 6,
        "jsonrpc": "2.0",
        "method": "setupProfile",
        "params": {
            "cortexToken": TOKEN,
            "profile": combo.get(),
            "headset": "INSIGHT-5A68CBD2",
            "status": "save"
            }
        }))
    print(ws.recv())

def acright():
    print("Entrenamiento para sujetar objetos")
    print("Relájese")
    for i in range(2,-1,-1):
        print(i)
        time.sleep(1)
    ws.send(json.dumps({
        "id": 4,
        "jsonrpc": "2.0",
```

```
        "method": "training",
        "params": {
            "action": "rotateClockwise",
            "cortexToken": TOKEN,
            "detection": "mentalCommand",
            "session": sid,
            "status": "start"
            }
        }))
    print(ws.recv())
    time.sleep(5)
    print(ws.recv())
    for var in range(3,-1,-1):
        Lt=Label(ventana, textvariable=str(var)).place(x=600,y=300)
        time.sleep(1)
    #time.sleep(4)
    L8=Label(ventana, text="RELÁJESE COMPLETAMENTE", font=("Helvetica",
14)).place(x=625,y=350)
    style = ttk.Style()
    style.theme_use('default')
    style.configure("skyblue.Horizontal.TProgressbar", background='turquoise')

    #progress=ttk.Progressbar(ventana, orient=HORIZONTAL,
length=400,mode='determinate', maximum=100, value=1)
    progress=ttk.Progressbar(ventana,length=350, style='skyblue.Horizontal.TProgressbar')
    progress.place(x=600,y=400)

    for i in range(0,100,5):
        progress['value']+=5
        progress.update()
        time.sleep(1)
    print(ws.recv())

    ratingrig()

def aceptarright():
    ws.send(json.dumps({
        "id": 5,
        "jsonrpc": "2.0",
        "method": "training",
        "params": {
            "action": "rotateClockwise",
            "cortexToken": TOKEN,
```

```
            "detection": "mentalCommand",
            "session": sid,
            "status": "accept"
            }
        }))
    print(ws.recv())
    time.sleep(2)
    print(ws.recv())
    ws.send(json.dumps({
        "id": 6,
        "jsonrpc": "2.0",
        "method": "setupProfile",
        "params": {
            "cortexToken": TOKEN,
            "profile": combo.get(),
            "headset": "INSIGHT-5A68CBD2",
            "status": "save"
            }
        }))
    print(ws.recv())
    #ratinglift()
def rechazarright():
    ws.send(json.dumps({
        "id": 5,
        "jsonrpc": "2.0",
        "method": "training",
        "params": {
            "action": "rotateClockwise",
            "cortexToken": TOKEN,
            "detection": "mentalCommand",
            "session": sid,
            "status": "reject"
            }
        }))
    print(ws.recv())
    time.sleep(2)
    print(ws.recv())
    ws.send(json.dumps({
        "id": 6,
        "jsonrpc": "2.0",
        "method": "setupProfile",
        "params": {
            "cortexToken": TOKEN,
```

```
            "profile": combo.get(),
            "headset": "INSIGHT-5A68CBD2",
            "status": "save"
            }
        }))
    print(ws.recv())

def acleft():
    print("Entrenamiento para sujetar objetos")
    print("Relájese")
    for i in range(2,-1,-1):
        print(i)
        time.sleep(1)
    ws.send(json.dumps({
        "id": 4,
        "jsonrpc": "2.0",
        "method": "training",
        "params": {
            "action": "left",
            "cortexToken": TOKEN,
            "detection": "mentalCommand",
            "session": sid,
            "status": "start"
            }
        }))
    print(ws.recv())
    time.sleep(5)
    print(ws.recv())
    for var in range(3,-1,-1):
        Lt=Label(ventana, textvariable=str(var)).place(x=600,y=300)
        time.sleep(1)
    #time.sleep(4)
    L8=Label(ventana, text="RELÁJESE COMPLETAMENTE", font=("Helvetica",
14)).place(x=625,y=350)
    style = ttk.Style()
    style.theme_use('default')
    style.configure("skyblue.Horizontal.TProgressbar", background='turquoise')

    #progress=ttk.Progressbar(ventana, orient=HORIZONTAL,
length=400,mode='determinate', maximum=100, value=1)
    progress=ttk.Progressbar(ventana,length=350, style='skyblue.Horizontal.TProgressbar')
    progress.place(x=600,y=400)
```

```
    for i in range(0,100,5):

        progress['value']+=5
        progress.update()
        time.sleep(1)
    print(ws.recv())

    ratingrig()

def aceptarleft():
    ws.send(json.dumps({
        "id": 5,
        "jsonrpc": "2.0",
        "method": "training",
        "params": {
            "action": "left",
            "cortexToken": TOKEN,
            "detection": "mentalCommand",
            "session": sid,
            "status": "accept"
            }
        }))
    print(ws.recv())
    time.sleep(2)
    print(ws.recv())
    ws.send(json.dumps({
        "id": 6,
        "jsonrpc": "2.0",
        "method": "setupProfile",
        "params": {
            "cortexToken": TOKEN,
            "profile": combo.get(),
            "headset": "INSIGHT-5A68CBD2",
            "status": "save"
            }
        }))
    print(ws.recv())
    #ratinglift()
def rechazarleft():
    ws.send(json.dumps({
        "id": 5,
        "jsonrpc": "2.0",
```

```
        "method": "training",
        "params": {
            "action": "left",
            "cortexToken": TOKEN,
            "detection": "mentalCommand",
            "session": sid,
            "status": "reject"
            }
        }))
    print(ws.recv())
    time.sleep(2)
    print(ws.recv())
    ws.send(json.dumps({
        "id": 6,
        "jsonrpc": "2.0",
        "method": "setupProfile",
        "params": {
            "cortexToken": TOKEN,
            "profile": combo.get(),
            "headset": "INSIGHT-5A68CBD2",
            "status": "save"
            }
        }))
    print(ws.recv())

def entrenarcom():

    cargarpn()
    cerebromap()
    L5=Label(ventana, text="BrainMap", font=("Helvetica", 14)).place(x=350,y=250)
    L6=Label(ventana, text="Coordenadas", font=("Helvetica", 12)).place(x=350,y=275)
    neutral=tkinter.Button(ventana, text="Neutral",bd=8, width=15, height=3, bg="skyblue",
command=acneutral).place(x=50,y=250)
    aceptarne=tkinter.Button(ventana, text="Aceptar", bd=8, width=8, height=1,
bg="skyblue",command=aceptarneutral).place(x=200,y=250)
    rechazarne=tkinter.Button(ventana, text="Rechazar", bd=8, width=8,
height=1,command=rechazarneutral).place(x=200,y=290)
    lift=tkinter.Button(ventana, text="Sujetar Objeto",bd=8, width=15, height=3, bg="teal",
command=aclift).place(x=50,y=350)
    aceptarli=tkinter.Button(ventana, text="Aceptar", bd=8, width=8, height=1,
bg="teal",command=aceptarlift).place(x=200,y=350)
    rechazarli=tkinter.Button(ventana, text="Rechazar", bd=8, width=8,
height=1,command=rechazarlift).place(x=200,y=390)
```

```
   right=tkinter.Button(ventana, text="Giro Derecha",bd=8, width=15, height=3,bg="turquoise", command=acright).place(x=50,y=450)
   aceptarri=tkinter.Button(ventana, text="Aceptar", bd=8, width=8, height=1, bg="turquoise",command=aceptarright).place(x=200,y=450)
   rechazarri=tkinter.Button(ventana, text="Rechazar", bd=8, width=8, height=1,command=rechazarright).place(x=200,y=490)

def activar():
   global running
   running = True
   activarcom()

def activarcom():
   if running:
      ws.send(json.dumps({
         "id": 3,
         "jsonrpc": "2.0",
         "method": "setupProfile",
         "params": {
            "cortexToken": TOKEN,
            "profile": combo.get(),
            "headset": "INSIGHT-5A68CBD2",
            "status": "load"
            }
         }))
      print("Perfil: "+ws.recv())
      #time.sleep(3)

      #Suscripción
      def commental():
         ws.send(json.dumps({
            "id": 8,
            "jsonrpc": "2.0",
            "method": "subscribe",
            "params": {
               "cortexToken": TOKEN,
               "session": sid,
               "streams": ["com"]
               }
            }
                     ))
```

```
def giroscopio():
    data=s.recv(1024)
    if(len(data)<=84):
        datastr=str(data, 'utf8')
        datadict=data.decode("UTF-8")
        trama=ast.literal_eval(datadict)
        ax=trama['AcX']
        ay=trama['AcY']
        az=trama['AcZ']
        gx=trama['GyX']
        gy=trama['GyY']
        gz=trama['GyZ']
        ###calcular angulos de rotación pitch roll yaw
        ##-180:180
        sgz1=(az>=0)-(az<0)
        sgz=float(sgz1)
        a=np.sqrt(az*az+ax*ax)
        b=np.sqrt(az*az+ay*ay)
        c=np.sqrt(ax*ax+az*az)
        Angulox=round(np.rad2deg(math.atan2(ay,sgz*a)),1)
        Anguloy=round(np.rad2deg(-math.atan2(ax,b)),1)
        Anguloz=round(np.rad2deg(math.atan2(c,az)),1)
        Tnuevo=millis()
        dt=(Tnuevo-preintervalo)*0.001
        preintervalo=Tnuevo
        angx=round((0.98*(angx+(gx/131)*dt)+0.02*Angulox),1)
        angy=round((0.98*(angy+(gy/131)*dt)+0.02*Anguloy),1)
        angz=round((0.98*(angz+(gz/131)*dt)+0.02*Anguloz),1)

        angxprev=angx
        angyprev=angy
        angzprev=angz


commental()
millis=lambda:int(round(time.time()*1000))
preintervalo=millis()
angx=0
angy=0
angz=0

while True:
    try:
```

```
            inicio = time.perf_counter()
            dato1=json.loads(ws.recv())["com"]
            comando=dato1[0]
            valor=dato1[1]
            fin = time.perf_counter()
            tiempo_lectura = fin - inicio
            hora_actual = time.strftime("%H:%M:%S")

            data=s.recv(1024)
            if(len(data)<=84):
               datastr=str(data, 'utf8')
               datadict=data.decode("UTF-8")
               trama=ast.literal_eval(datadict)
               ax=trama['AcX']
               ay=trama['AcY']
               az=trama['AcZ']
               gx=trama['GyX']
               gy=trama['GyY']
               gz=trama['GyZ']
               sgz1=(az>=0)-(az<0)
               sgz=float(sgz1)
               a=np.sqrt(az*az+ax*ax)
               b=np.sqrt(az*az+ay*ay)
               c=np.sqrt(ax*ax+az*az)
               Angulox=round(np.rad2deg(math.atan2(ay,sgz*a)),1)
               Anguloy=round(np.rad2deg(-math.atan2(ax,b)),1)
               Anguloz=round(np.rad2deg(math.atan2(c,az)),1)
               Tnuevo=millis()
               dt=(Tnuevo-preintervalo)*0.001
               preintervalo=Tnuevo
               angx=round((0.98*(angx+(gx/131)*dt)+0.02*Angulox),1)
               angy=round((0.98*(angy+(gy/131)*dt)+0.02*Anguloy),1)
               angz=round((0.98*(angz+(gz/131)*dt)+0.02*Anguloz),1)
               #angz=round((angz+(gz/131)*dt),1)
               angxprev=angx
               angyprev=angy
               angzprev=angz

               print("Comando: "+comando,"Valor: "+str(valor), "AngX: "+str(Angulox),"AngZ: "+str(angz), "AngY: "+str(Anguloy),"Hora: ", hora_actual, ":", valor, "- Retardo: ", round(tiempo_lectura,7))
```

```
                #print("Comando: "+comando,"Valor: "+str(valor), "AngX: "+str(roll),"AngZ: "+str(yaw), "AngY: "+str(pitch),"Hora: ", hora_actual, ":", valor, "- Retardo: ", round(tiempo_lectura,7))


                if(Angulox>=30)and (comando=='lift')and (valor>=0.5)and (mano.current=='abierto'):

                    ##print("Aciertos: "+str(contadorcom))
                    print(mano.current)
                    cerrar()
                    mano.abcer()
                if(Angulox<30)and(comando=='neutral')and(mano.current=='cerrado'):
                    print(mano.current)
                    abrir()
                    mano.cerab()
                if(Anguloy>1 and mun.current=='izquierda')and(comando=='lift')and(valor>=0.5):
                    print(mun.current)
                    derecha()
                    mun.izder()
                if(Anguloy<-11 and mun.current=='derecha')and(comando=='lift')and(valor>=0.5):
                    print(mun.current)
                    izquierda()
                    mun.deriz()

            except KeyError as com:
                pass
    else:
        print("Saliendo del sistema")



        #os.system("cls")

def desactivar():
    global running
    running = False
    ventana.destroy()

running = True
ventana = tkinter.Tk()
```

```
ventana.title('BCI Insight-Prótesis')
ventana.iconbitmap(r'C:\Users\LattePanda\Documents\Programacion Tesis\cerebro2.ico')

L1=Label(ventana, text="Ingrese el nombre del Perfil Nuevo: ")
L1.place(x=50,y=50)
L2=Label(ventana, text="Seleccione un perfil de entrenamiento: ")
L2.place(x=50,y=125)
L3=Label(ventana, text="Temperatura")
L3.place(x=650,y=25)
L4=Label(ventana, text=" °C", font=("Helvetica", 30)).place(x=700,y=75)
E1=Entry(ventana, width=20)
E1.place(x=275, y=50)
crearperfil=tkinter.Button(ventana, text="Crear Perfil",bd=8, width=15,
height=3,bg="skyblue", command=crearperfiles)
crearperfil.place(x=425,y=25)

def callbackFunc(event):
   print("Perfil escogido: ", combo.get())
combo = Combobox(ventana)
infoper()

entrenamientos = json.loads(ws.recv())["result"]
n=len(entrenamientos)
perfn=[]
for i in range(0,n,1):
   perfn.append(entrenamientos[i].get("name"))
perfc=np.append(perfn,E1.get())

listaperfc=perfc.tolist()
combo['values']= listaperfc
combo.grid(column=0, row=0)
combo.place(x=275,y=125)
combo.bind("<<ComboboxSelected>>", callbackFunc)
entrenarnp=tkinter.Button(ventana, text="Entrenar Comandos",bd=8, width=15, height=3,
command=entrenarcom)
entrenarnp.place(x=425,y=110)
activar1=tkinter.Button(ventana, text="Activar Prótesis", bd=8, width=20, height=5,
bg="green", command=activar)
activar1.pack()
activar1.place(x=800,y=50)

desactivar1=tkinter.Button(ventana, text="Desactivar Prótesis", bd=8, width=20, height=5,
bg="red", command=desactivar)
```

```
desactivar1.pack()
desactivar1.place(x=800,y=150)

t1.start()
ventana.geometry("1000x600")
ventana.mainloop()
```

ANEXO 3. CÓDIGO DE CONTROL DE MICROMOTORES Y SENSOR DE TEMPERATURA

```
#include <Wire.h>
#include <Adafruit_PWMServoDriver.h>
#include <Boards.h>
#include <Adafruit_MLX90614.h>
Adafruit_MLX90614 mlx = Adafruit_MLX90614();
#include <Adafruit_NeoPixel.h>
#include <avr/power.h>
#define PIN 9
#define NUMPIXELS      10
int dato=0;
int accion=0;
Adafruit_PWMServoDriver servos = Adafruit_PWMServoDriver(0x40);
unsigned int pos0=172; // ancho de pulso en cuentas para pocicion 0°
unsigned int pos180=565;
unsigned int pos90=368;
Adafruit_NeoPixel pixels = Adafruit_NeoPixel(NUMPIXELS, PIN, NEO_GRB + NEO_KHZ800);
void iniciomano(){ for (int duty = pos0; duty < pos180; duty=duty+1) {
  servos.setPWM(4,0,duty);
  servos.setPWM(3,0,duty);
  servos.setPWM(2,0,duty);
  servos.setPWM(1,0,duty);
  servos.setPWM(5,0,duty);
  servos.setPWM(0,0,duty);
  //servos.setPWM(6,0,duty);
  delay(1);

}
for (int duty = pos180; duty > pos90; duty=duty-2) {
  servos.setPWM(6,0,duty);
  delay(1);
}
}
void setup() {
Serial.begin(9600);
mlx.begin();
pixels.begin();
servos.begin();
servos.setPWMFreq(50); //Frecuecia PWM de 60Hz o T=16,66ms
iniciomano();
```

```
pixels.begin();
pixels.setBrightness(10);  // Ajuste de Brillo de todos los leds
pixels.show();

}

void temperatura(){Serial.println(mlx.readObjectTempC());
 dato=mlx.readObjectTempC();
 uint32_t rojo = pixels.Color(150,0,0);
uint32_t amarillo = pixels.Color(150,150,0);
uint32_t blanco = pixels.Color(150,150,150);
uint32_t celeste = pixels.Color(0,150,150);
uint32_t azul = pixels.Color(0,0,150);
uint32_t verde = pixels.Color(0,150,0);
uint32_t naranja=pixels.Color(255,128,0);
if(dato>=0&&dato<30){  //Serial.println(mlx.readObjectTempC());
     for(int i=0;i<NUMPIXELS;i++){

    pixels.setPixelColor(i, verde); // Brillo moderado en rojo
    pixels.show();   // Mostramos y actualizamos el color del pixel de nuestra cinta led RGB
    delay(50); // Pausa por un periodo de tiempo (en milisegundos).
}}
else if(dato>=30&&dato<40){//Serial.println(mlx.readObjectTempC());
  for(int i=0;i<NUMPIXELS;i++){
    pixels.setPixelColor(i, amarillo); // Brillo moderado en azul
    pixels.show();   // Mostramos y actualizamos el color del pixel de nuestra cinta led RGB
    delay(50); // Pausa por un periodo de tiempo (en milisegundos).
}}
 else if(dato>=40&&dato<50){  for(int i=0;i<NUMPIXELS;i++){
    pixels.setPixelColor(i, naranja); // Brillo moderado en azul
    pixels.show();   // Mostramos y actualizamos el color del pixel de nuestra cinta led RGB
    delay(50); // Pausa por un periodo de tiempo (en milisegundos).
}}
  else if(dato>=50){  for(int i=0;i<NUMPIXELS;i++){
    pixels.setPixelColor(i, rojo); // Brillo moderado en azul
    pixels.show();   // Mostramos y actualizamos el color del pixel de nuestra cinta led RGB
    delay(50); // Pausa por un periodo de tiempo (en milisegundos).
}}
   else if(dato<0&&dato>=-20){  for(int i=0;i<NUMPIXELS;i++){
    pixels.setPixelColor(i, blanco); // Brillo moderado en azul
    pixels.show();   // Mostramos y actualizamos el color del pixel de nuestra cinta led RGB
    delay(50); // Pausa por un periodo de tiempo (en milisegundos).
}}
```

```
else if(dato<=-21&&dato>-30){  for(int i=0;i<NUMPIXELS;i++){
    pixels.setPixelColor(i, celeste); // Brillo moderado en azul
    pixels.show();   // Mostramos y actualizamos el color del pixel de nuestra cinta led RGB
    delay(50); // Pausa por un periodo de tiempo (en milisegundos).
}}
else if(dato<=-31){  for(int i=0;i<NUMPIXELS;i++){
    pixels.setPixelColor(i, azul); // Brillo moderado en azul
    pixels.show();   // Mostramos y actualizamos el color del pixel de nuestra cinta led RGB
    delay(50); // Pausa por un periodo de tiempo (en milisegundos).
}}
}

void loop(){
  temperatura();
   accion=Serial.read();
if(accion=='a'){
for (int duty = pos0; duty < pos180; duty=duty+1) {
   servos.setPWM(4,0,duty);
   servos.setPWM(3,0,duty);
   servos.setPWM(2,0,duty);
   servos.setPWM(1,0,duty);
   servos.setPWM(5,0,duty);
   servos.setPWM(0,0,duty);
   //servos.setPWM(6,0,duty);
   delay(1);}

}
else if(accion=='c'){
  for (int duty = pos180; duty > pos0; duty=duty-2) {
   servos.setPWM(4,0,duty);
   servos.setPWM(3,0,duty);
   servos.setPWM(2,0,duty);
   servos.setPWM(1,0,duty);
   servos.setPWM(5,0,duty);
   servos.setPWM(0,0,duty);
   //servos.setPWM(6,0,duty);
   delay(1);}
  }
else if(accion=='d'){for (int duty = pos90; duty < pos180; duty=duty+1) {
   servos.setPWM(6,0,duty);
   delay(1);
}}
else if(accion=='i'){for (int duty = pos180; duty > pos90; duty=duty-2) {
```

```
  servos.setPWM(6,0,duty);
  delay(1);
}}
}
```

ANEXO 4. Código DEL SENSOR DE INERCIA

```
import machine
escalacel=9.81*16384
escalgir=250/32768
class accel():
    def __init__(self, i2c, addr=0x68):
        self.iic = i2c
        self.addr = addr
        self.iic.start()
        self.iic.writeto(self.addr, bytearray([107, 0]))
        self.iic.stop()

    def get_raw_values(self):
        self.iic.start()
        a = self.iic.readfrom_mem(self.addr, 0x3B, 14)
        self.iic.stop()
        return a

    def get_ints(self):
        b = self.get_raw_values()
        c = []
        for i in b:
            c.append(i)
        return c

    def bytes_toint(self, firstbyte, secondbyte):
        if not firstbyte & 0x80:
            return firstbyte << 8 | secondbyte
        return - (((firstbyte ^ 255) << 8) | (secondbyte ^ 255) + 1)

    def get_values(self):
        raw_ints = self.get_raw_values()
        vals = {}
        vals["AcX"] = self.bytes_toint(raw_ints[0], raw_ints[1])
        vals["AcY"] = self.bytes_toint(raw_ints[2], raw_ints[3])
        vals["AcZ"] = self.bytes_toint(raw_ints[4], raw_ints[5])
        #vals["Tmp"] = self.bytes_toint(raw_ints[6], raw_ints[7]) / 340.00 + 36.53
        vals["GyX"] = self.bytes_toint(raw_ints[8], raw_ints[9])
        vals["GyY"] = self.bytes_toint(raw_ints[10], raw_ints[11])
        vals["GyZ"] = self.bytes_toint(raw_ints[12], raw_ints[13])
        return vals
```

```
def val_test(self):
    from time import sleep
    while 1:
        print(self.get_values())
        sleep(0.05)
```

ANEXO 5. PRUEBA COMANDO NEUTRAL-EXTENSIÓN

```
Comando: neutral Valor: 0.0 AngX: 12.7 AngZ: 21.2 AngY: -1.4 Hora:  14:37:48 : 0.0
Comando: neutral Valor: 0.0 AngX: 12.6 AngZ: 21.4 AngY: -1.8 Hora:  14:37:48 : 0.0
Comando: neutral Valor: 0.0 AngX: 13.3 AngZ: 21.7 AngY: -1.6 Hora:  14:37:48 : 0.0
Comando: neutral Valor: 0.0 AngX: 13.3 AngZ: 22.1 AngY: -1.3 Hora:  14:37:48 : 0.0
Comando: neutral Valor: 0.0 AngX: 13.6 AngZ: 22.5 AngY: -1.7 Hora:  14:37:48 : 0.0
Comando: neutral Valor: 0.0 AngX: 12.4 AngZ: 22.9 AngY: -1.2 Hora:  14:37:49 : 0.0
Comando: neutral Valor: 0.0 AngX: 12.7 AngZ: 23.2 AngY: -1.0 Hora:  14:37:49 : 0.0
Comando: neutral Valor: 0.0 AngX: 11.8 AngZ: 23.5 AngY: -1.1 Hora:  14:37:49 : 0.0
Comando: neutral Valor: 0.0 AngX: 12.6 AngZ: 23.7 AngY: -1.1 Hora:  14:37:49 : 0.0
Comando: neutral Valor: 0.0 AngX: 12.5 AngZ: 24.0 AngY: -1.0 Hora:  14:37:49 : 0.0
Comando: neutral Valor: 0.0 AngX: 12.6 AngZ: 24.1 AngY: -1.1 Hora:  14:37:50 : 0.0
Comando: neutral Valor: 0.0 AngX: 11.1 AngZ: 24.3 AngY: -0.5 Hora:  14:37:50 : 0.0
Comando: neutral Valor: 0.0 AngX: 10.2 AngZ: 24.5 AngY: -0.0 Hora:  14:37:50 : 0.0
Comando: neutral Valor: 0.0 AngX: 9.0 AngZ: 24.7 AngY: 0.5 Hora:  14:37:50 : 0.0 -
Comando: neutral Valor: 0.0 AngX: 9.2 AngZ: 25.0 AngY: 0.1 Hora:  14:37:51 : 0.0 -
Comando: neutral Valor: 0.0 AngX: 10.0 AngZ: 25.3 AngY: 0.0 Hora:  14:37:51 : 0.0 -
Comando: neutral Valor: 0.0 AngX: 9.9 AngZ: 25.6 AngY: -0.9 Hora:  14:37:51 : 0.0 -
Comando: neutral Valor: 0.0 AngX: 11.0 AngZ: 25.8 AngY: -0.8 Hora:  14:37:51 : 0.0
Comando: neutral Valor: 0.0 AngX: 11.0 AngZ: 26.1 AngY: -0.4 Hora:  14:37:51 : 0.0
Comando: neutral Valor: 0.0 AngX: 10.4 AngZ: 26.5 AngY: -0.2 Hora:  14:37:52 : 0.0
Comando: neutral Valor: 0.0 AngX: 10.4 AngZ: 26.7 AngY: -0.2 Hora:  14:37:52 : 0.0
Comando: neutral Valor: 0.0 AngX: 11.3 AngZ: 27.0 AngY: -0.4 Hora:  14:37:52 : 0.0
Comando: neutral Valor: 0.0 AngX: 10.3 AngZ: 27.2 AngY: 0.0 Hora:  14:37:52 : 0.0 -
Comando: neutral Valor: 0.0 AngX: 11.0 AngZ: 27.4 AngY: -0.3 Hora:  14:37:52 : 0.0
Comando: neutral Valor: 0.0 AngX: 10.7 AngZ: 27.7 AngY: -1.0 Hora:  14:37:53 : 0.0
Comando: neutral Valor: 0.0 AngX: 10.8 AngZ: 27.9 AngY: -0.3 Hora:  14:37:53 : 0.0
Comando: neutral Valor: 0.0 AngX: 10.4 AngZ: 28.0 AngY: -0.4 Hora:  14:37:53 : 0.0
Comando: neutral Valor: 0.0 AngX: 10.7 AngZ: 28.1 AngY: -0.2 Hora:  14:37:53 : 0.0
Comando: neutral Valor: 0.0 AngX: 10.5 AngZ: 28.2 AngY: -0.3 Hora:  14:37:54 : 0.0
Comando: neutral Valor: 0.0 AngX: 10.1 AngZ: 28.2 AngY: -0.6 Hora:  14:37:54 : 0.0
Comando: neutral Valor: 0.0 AngX: 10.2 AngZ: 28.4 AngY: -0.1 Hora:  14:37:54 : 0.0
Comando: neutral Valor: 0.0 AngX: 10.2 AngZ: 28.4 AngY: -0.1 Hora:  14:37:54
Comando: neutral Valor: 0.0 AngX: 9.7 AngZ: 28.6 AngY: -0.6 Hora:  14:37:54 :
Comando: neutral Valor: 0.0 AngX: 10.0 AngZ: 28.8 AngY: -0.9 Hora:  14:37:54
Comando: neutral Valor: 0.0 AngX: 10.4 AngZ: 29.1 AngY: -1.4 Hora:  14:37:55
Comando: neutral Valor: 0.0 AngX: 11.2 AngZ: 29.1 AngY: 0.1 Hora:  14:37:55 :
Comando: neutral Valor: 0.0 AngX: 10.6 AngZ: 29.2 AngY: -0.3 Hora:  14:37:55
Comando: neutral Valor: 0.0 AngX: 11.2 AngZ: 29.2 AngY: -0.5 Hora:  14:37:55
Comando: neutral Valor: 0.0 AngX: 11.2 AngZ: 29.4 AngY: -1.1 Hora:  14:37:56
Comando: neutral Valor: 0.0 AngX: 11.1 AngZ: 29.6 AngY: -1.1 Hora:  14:37:56
Comando: neutral Valor: 0.0 AngX: 11.5 AngZ: 29.7 AngY: -0.9 Hora:  14:37:56
Comando: neutral Valor: 0.0 AngX: 11.0 AngZ: 29.8 AngY: -0.5 Hora:  14:37:56
Comando: neutral Valor: 0.0 AngX: 11.0 AngZ: 29.9 AngY: -1.3 Hora:  14:37:57
Comando: neutral Valor: 0.0 AngX: 10.9 AngZ: 30.1 AngY: -0.7 Hora:  14:37:57
Comando: neutral Valor: 0.0 AngX: 11.0 AngZ: 30.3 AngY: -1.1 Hora:  14:37:57
Comando: neutral Valor: 0.0 AngX: 11.2 AngZ: 30.4 AngY: -0.9 Hora:  14:37:57
Comando: neutral Valor: 0.0 AngX: 11.8 AngZ: 30.6 AngY: -0.6 Hora:  14:37:57
Comando: neutral Valor: 0.0 AngX: 11.3 AngZ: 30.7 AngY: -1.6 Hora:  14:37:58
Comando: neutral Valor: 0.0 AngX: 11.2 AngZ: 30.8 AngY: -0.5 Hora:  14:37:58
Comando: neutral Valor: 0.0 AngX: 11.3 AngZ: 30.9 AngY: -0.4 Hora:  14:37:58
Comando: neutral Valor: 0.0 AngX: 11.1 AngZ: 31.1 AngY: -0.3 Hora:  14:37:58
Comando: neutral Valor: 0.0 AngX: 11.4 AngZ: 31.2 AngY: -0.8 Hora:  14:37:59
Comando: neutral Valor: 0.0 AngX: 11.0 AngZ: 31.3 AngY: -0.6 Hora:  14:37:59
Comando: neutral Valor: 0.0 AngX: 11.4 AngZ: 31.4 AngY: -0.6 Hora:  14:37:59
Comando: neutral Valor: 0.0 AngX: 11.1 AngZ: 31.5 AngY: -1.0 Hora:  14:37:59
Comando: neutral Valor: 0.0 AngX: 10.9 AngZ: 31.7 AngY: 0.3 Hora:  14:37:59 :
Comando: neutral Valor: 0.0 AngX: 11.2 AngZ: 31.8 AngY: -0.6 Hora:  14:38:00
Comando: neutral Valor: 0.0 AngX: 11.5 AngZ: 31.9 AngY: -0.4 Hora:  14:38:00
Comando: neutral Valor: 0.0 AngX: 11.5 AngZ: 32.0 AngY: -1.0 Hora:  14:38:00
Comando: neutral Valor: 0.0 AngX: 11.0 AngZ: 32.1 AngY: -0.6 Hora:  14:38:00
```

ANEXO 6. PRUEBA COMANDO LIFT-FLEXIÓN

```
Comando: lift Valor: 0.61086 AngX: 54.5 AngZ: 39.2 AngY: -8.2 Hora:  15:12:38 :
Comando: lift Valor: 0.521299 AngX: 53.9 AngZ: 39.2 AngY: -8.7 Hora:  15:12:38 :
Comando: lift Valor: 0.411465 AngX: 54.5 AngZ: 39.2 AngY: -8.3 Hora:  15:12:38 :
Comando: lift Valor: 0.35897 AngX: 54.8 AngZ: 39.4 AngY: -8.4 Hora:  15:12:38 :
Comando: lift Valor: 0.40428 AngX: 54.7 AngZ: 39.3 AngY: -8.7 Hora:  15:12:38 :
Comando: lift Valor: 0.491784 AngX: 54.5 AngZ: 39.3 AngY: -8.4 Hora:  15:12:39 :
Comando: lift Valor: 0.582636 AngX: 54.5 AngZ: 39.3 AngY: -8.1 Hora:  15:12:39 :
Comando: lift Valor: 0.661504 AngX: 54.1 AngZ: 39.2 AngY: -8.2 Hora:  15:12:39 :
Comando: neutral Valor: 0.0 AngX: 54.1 AngZ: 39.2 AngY: -8.4 Hora:  15:12:39 : (
Comando: neutral Valor: 0.0 AngX: 54.7 AngZ: 39.3 AngY: -8.5 Hora:  15:12:39 : (
Comando: neutral Valor: 0.0 AngX: 54.3 AngZ: 39.4 AngY: -8.0 Hora:  15:12:40 : (
Comando: lift Valor: 0.722519 AngX: 54.4 AngZ: 38.3 AngY: -8.9 Hora:  15:12:41 :
Comando: lift Valor: 0.765098 AngX: 53.9 AngZ: 38.4 AngY: -8.9 Hora:  15:12:41 :
Comando: lift Valor: 0.806762 AngX: 54.1 AngZ: 38.5 AngY: -8.3 Hora:  15:12:41 :
Comando: lift Valor: 0.840446 AngX: 54.2 AngZ: 38.4 AngY: -8.0 Hora:  15:12:42 :
Comando: lift Valor: 0.859675 AngX: 54.5 AngZ: 38.4 AngY: -8.6 Hora:  15:12:42 :
Comando: lift Valor: 0.861601 AngX: 54.2 AngZ: 38.3 AngY: -8.3 Hora:  15:12:42 :
Comando: lift Valor: 0.848915 AngX: 54.8 AngZ: 38.4 AngY: -8.2 Hora:  15:12:42 :
Comando: lift Valor: 0.823797 AngX: 54.0 AngZ: 38.4 AngY: -8.8 Hora:  15:12:42 :
Comando: lift Valor: 0.819438 AngX: 54.3 AngZ: 38.5 AngY: -8.4 Hora:  15:12:43 :
Comando: lift Valor: 0.831194 AngX: 54.5 AngZ: 38.7 AngY: -8.3 Hora:  15:12:43 :
Comando: lift Valor: 0.834273 AngX: 54.0 AngZ: 38.7 AngY: -8.8 Hora:  15:12:43 :
Comando: lift Valor: 0.809252 AngX: 54.0 AngZ: 38.7 AngY: -9.3 Hora:  15:12:43 :
Comando: lift Valor: 0.811075 AngX: 54.0 AngZ: 38.7 AngY: -8.3 Hora:  15:12:43 :
Comando: lift Valor: 0.839834 AngX: 53.9 AngZ: 38.7 AngY: -8.6 Hora:  15:12:44 :
Comando: lift Valor: 0.866514 AngX: 54.3 AngZ: 38.9 AngY: -8.1 Hora:  15:12:44 :
Comando: lift Valor: 0.886321 AngX: 54.3 AngZ: 38.9 AngY: -8.6 Hora:  15:12:44 :
Comando: lift Valor: 0.899702 AngX: 54.4 AngZ: 38.8 AngY: -8.8 Hora:  15:12:44 :
Comando: lift Valor: 0.910888 AngX: 54.5 AngZ: 38.8 AngY: -8.3 Hora:  15:12:45 :
Comando: lift Valor: 0.916562 AngX: 54.3 AngZ: 38.8 AngY: -8.2 Hora:  15:12:45 :
Comando: lift Valor: 0.898888 AngX: 54.3 AngZ: 38.6 AngY: -8.3 Hora:  15:12:45 :
Comando: neutral Valor: 0.0 AngX: 54.3 AngZ: 38.7 AngY: -8.4 Hora:  15:12:45 : (
Comando: neutral Valor: 0.0 AngX: 54.0 AngZ: 38.8 AngY: -8.1 Hora:  15:12:45 : (
Comando: neutral Valor: 0.0 AngX: 54.3 AngZ: 38.9 AngY: -8.6 Hora:  15:12:46 : (
Comando: neutral Valor: 0.0 AngX: 54.8 AngZ: 39.0 AngY: -8.5 Hora:  15:12:46 : (
Comando: lift Valor: 0.913732 AngX: 54.0 AngZ: 39.0 AngY: -8.3 Hora:  15:12:46
Comando: lift Valor: 0.912356 AngX: 55.1 AngZ: 38.9 AngY: -8.2 Hora:  15:12:46
Comando: lift Valor: 0.895838 AngX: 54.4 AngZ: 38.9 AngY: -8.0 Hora:  15:12:46
Comando: lift Valor: 0.893581 AngX: 54.1 AngZ: 38.9 AngY: -8.1 Hora:  15:12:47
Comando: lift Valor: 0.905961 AngX: 54.3 AngZ: 38.9 AngY: -8.4 Hora:  15:12:47
Comando: lift Valor: 0.918076 AngX: 54.4 AngZ: 38.9 AngY: -8.2 Hora:  15:12:47
Comando: lift Valor: 0.924419 AngX: 54.5 AngZ: 38.9 AngY: -7.9 Hora:  15:12:47
Comando: lift Valor: 0.917317 AngX: 54.2 AngZ: 38.9 AngY: -8.5 Hora:  15:12:48
Comando: lift Valor: 0.880189 AngX: 53.6 AngZ: 39.1 AngY: -8.5 Hora:  15:12:48
Comando: lift Valor: 0.802674 AngX: 54.0 AngZ: 39.1 AngY: -8.6 Hora:  15:12:48
Comando: lift Valor: 0.720178 AngX: 55.3 AngZ: 39.1 AngY: -8.3 Hora:  15:12:48
Comando: lift Valor: 0.701019 AngX: 54.5 AngZ: 39.1 AngY: -9.0 Hora:  15:12:48
Comando: lift Valor: 0.697897 AngX: 54.6 AngZ: 39.2 AngY: -8.4 Hora:  15:12:49
Comando: lift Valor: 0.698585 AngX: 55.0 AngZ: 39.3 AngY: -8.4 Hora:  15:12:49
Comando: lift Valor: 0.716066 AngX: 54.0 AngZ: 39.3 AngY: -8.4 Hora:  15:12:49
Comando: lift Valor: 0.749064 AngX: 54.6 AngZ: 39.3 AngY: -8.4 Hora:  15:12:49
Comando: lift Valor: 0.785372 AngX: 54.5 AngZ: 39.3 AngY: -8.6 Hora:  15:12:49
Comando: lift Valor: 0.804437 AngX: 54.2 AngZ: 39.3 AngY: -8.1 Hora:  15:12:50
Comando: lift Valor: 0.765503 AngX: 54.6 AngZ: 39.3 AngY: -8.3 Hora:  15:12:50
Comando: lift Valor: 0.617224 AngX: 54.4 AngZ: 39.4 AngY: -8.5 Hora:  15:12:50
Comando: lift Valor: 0.453024 AngX: 54.5 AngZ: 39.4 AngY: -8.1 Hora:  15:12:50
Comando: lift Valor: 0.315526 AngX: 54.4 AngZ: 39.4 AngY: -8.9 Hora:  15:12:50
Comando: lift Valor: 0.218717 AngX: 54.3 AngZ: 39.5 AngY: -8.2 Hora:  15:12:51
Comando: lift Valor: 0.17052 AngX: 54.8 AngZ: 39.5 AngY: -8.4 Hora:  15:12:51 :
Comando: lift Valor: 0.20489 AngX: 55.2 AngZ: 39.4 AngY: -8.3 Hora:  15:12:51 :
Comando: lift Valor: 0.314931 AngX: 55.0 AngZ: 39.4 AngY: -8.4 Hora:  15:12:51
Comando: lift Valor: 0.42719 AngX: 54.4 AngZ: 39.3 AngY: -8.4 Hora:  15:12:52 :
```

ANEXO 7. PRUEBA GIRO DE MUÑECA A FAVOR DE LAS MANECILLAS DEL RELOJ

Python 3.6.8 Shell

File Edit Shell Debug Options Window Help

```
Comando: lift Valor: 0.736411 AngX: 32.8 AngZ: 10.8 AngY: 1.1 Hora:  14:23:08 : 0.736411 - Retardo:  0.0007507
izquierda
Comando: lift Valor: 0.779011 AngX: 33.1 AngZ: 9.8 AngY: 4.3 Hora:  14:23:09 : 0.779011 - Retardo:  0.000847
Comando: lift Valor: 0.807871 AngX: 35.1 AngZ: 10.9 AngY: 3.8 Hora:  14:23:09 : 0.807871 - Retardo:  0.000532
Comando: lift Valor: 0.805718 AngX: 35.4 AngZ: 12.6 AngY: 2.1 Hora:  14:23:09 : 0.805718 - Retardo:  0.0006882
Comando: lift Valor: 0.80409 AngX: 36.4 AngZ: 12.8 AngY: 2.2 Hora:  14:23:10 : 0.80409 - Retardo:  0.0007945
Comando: lift Valor: 0.787056 AngX: 36.3 AngZ: 13.8 AngY: 1.3 Hora:  14:23:10 : 0.787056 - Retardo:  0.0009061
Comando: lift Valor: 0.751165 AngX: 36.4 AngZ: 14.0 AngY: 2.5 Hora:  14:23:10 : 0.751165 - Retardo:  0.0007907
Comando: lift Valor: 0.747871 AngX: 37.1 AngZ: 14.5 AngY: 1.8 Hora:  14:23:10 : 0.747871 - Retardo:  0.0008285
Comando: lift Valor: 0.819599 AngX: 36.3 AngZ: 16.9 AngY: 0.7 Hora:  14:23:11 : 0.819599 - Retardo:  0.0008023
Comando: lift Valor: 0.850161 AngX: 36.1 AngZ: 17.8 AngY: 0.6 Hora:  14:23:11 : 0.850161 - Retardo:  0.0007495
Comando: lift Valor: 0.821846 AngX: 35.6 AngZ: 18.3 AngY: -0.2 Hora:  14:23:11 : 0.821846 - Retardo:  0.0007247
Comando: lift Valor: 0.750538 AngX: 35.4 AngZ: 18.9 AngY: -0.0 Hora:  14:23:12 : 0.750538 - Retardo:  0.0008673
Comando: lift Valor: 0.716501 AngX: 36.0 AngZ: 19.7 AngY: -0.8 Hora:  14:23:12 : 0.716501 - Retardo:  0.0008602
Comando: lift Valor: 0.698474 AngX: 35.8 AngZ: 20.1 AngY: -0.7 Hora:  14:23:12 : 0.698474 - Retardo:  0.0008313
Comando: lift Valor: 0.460582 AngX: 35.6 AngZ: 21.1 AngY: -1.8 Hora:  14:23:17 : 0.460582 - Retardo:  0.0006299
Comando: lift Valor: 0.628981 AngX: 35.7 AngZ: 17.6 AngY: -0.4 Hora:  14:23:19 : 0.628981 - Retardo:  0.0006527
Comando: lift Valor: 0.672604 AngX: 35.2 AngZ: 18.5 AngY: -1.9 Hora:  14:23:19 : 0.672604 - Retardo:  0.0006966
Comando: lift Valor: 0.707332 AngX: 35.2 AngZ: 18.4 AngY: -0.6 Hora:  14:23:20 : 0.707332 - Retardo:  0.0007498
Comando: lift Valor: 0.732084 AngX: 35.0 AngZ: 18.9 AngY: -1.5 Hora:  14:23:20 : 0.732084 - Retardo:  0.0005599
Comando: lift Valor: 0.753947 AngX: 34.5 AngZ: 19.4 AngY: -1.4 Hora:  14:23:20 : 0.753947 - Retardo:  0.0005984
Comando: lift Valor: 0.772482 AngX: 35.8 AngZ: 19.7 AngY: -1.7 Hora:  14:23:20 : 0.772482 - Retardo:  0.0010742
Comando: lift Valor: 0.790174 AngX: 35.1 AngZ: 20.1 AngY: -0.8 Hora:  14:23:20 : 0.790174 - Retardo:  0.0014327
Comando: lift Valor: 0.816986 AngX: 35.6 AngZ: 20.4 AngY: -1.7 Hora:  14:23:21 : 0.816986 - Retardo:  0.0011473
Comando: lift Valor: 0.845551 AngX: 34.9 AngZ: 20.9 AngY: -1.7 Hora:  14:23:21 : 0.845551 - Retardo:  0.0010079
Comando: lift Valor: 0.881852 AngX: 34.7 AngZ: 21.1 AngY: -1.0 Hora:  14:23:21 : 0.881852 - Retardo:  0.0007296
Comando: lift Valor: 0.88557 AngX: 35.3 AngZ: 21.4 AngY: -1.8 Hora:  14:23:22 : 0.88557 - Retardo:  0.0008306
Comando: lift Valor: 0.889768 AngX: 34.8 AngZ: 21.9 AngY: -0.9 Hora:  14:23:22 : 0.889768 - Retardo:  0.0084107
Comando: lift Valor: 0.869302 AngX: 34.9 AngZ: 22.1 AngY: -1.3 Hora:  14:23:22 : 0.869302 - Retardo:  0.0009101
Comando: lift Valor: 0.584844 AngX: 34.1 AngZ: 22.3 AngY: -1.5 Hora:  14:23:23 : 0.584844 - Retardo:  0.0006979
Comando: lift Valor: 0.3872 AngX: 34.5 AngZ: 22.6 AngY: -1.0 Hora:  14:23:23 : 0.3872 - Retardo:  0.0008682
```

Python 3.6.8 Shell

File Edit Shell Debug Options Window Help

```
Comando: lift Valor: 0.3872 AngX: 34.5 AngZ: 22.6 AngY: -1.0 Hora:  14:23:23 : 0.3872 - Retardo:  0.0008682
Comando: lift Valor: 0.215158 AngX: 35.6 AngZ: 22.8 AngY: -1.3 Hora:  14:23:23 : 0.215158 - Retardo:  0.0008033
Comando: lift Valor: 0.134997 AngX: 35.4 AngZ: 23.1 AngY: -1.3 Hora:  14:23:24 : 0.134997 - Retardo:  0.0006725
Comando: lift Valor: 0.228053 AngX: 34.9 AngZ: 23.1 AngY: -1.9 Hora:  14:23:24 : 0.228053 - Retardo:  0.0006311
Comando: lift Valor: 0.297987 AngX: 35.0 AngZ: 23.3 AngY: -1.3 Hora:  14:23:24 : 0.297987 - Retardo:  0.0010063
Comando: lift Valor: 0.335284 AngX: 35.4 AngZ: 23.5 AngY: -2.1 Hora:  14:23:25 : 0.335284 - Retardo:  0.0007704
Comando: lift Valor: 0.264189 AngX: 34.9 AngZ: 23.7 AngY: -1.6 Hora:  14:23:25 : 0.264189 - Retardo:  0.0006708
Comando: lift Valor: 0.261681 AngX: 35.9 AngZ: 24.1 AngY: -1.8 Hora:  14:23:25 : 0.261681 - Retardo:  0.0010793
Comando: lift Valor: 0.276244 AngX: 35.0 AngZ: 24.4 AngY: -1.4 Hora:  14:23:26 : 0.276244 - Retardo:  0.0007057
Comando: lift Valor: 0.290467 AngX: 35.4 AngZ: 24.5 AngY: -1.5 Hora:  14:23:26 : 0.290467 - Retardo:  0.0008796
Comando: lift Valor: 0.331419 AngX: 34.9 AngZ: 24.9 AngY: -2.2 Hora:  14:23:26 : 0.331419 - Retardo:  0.0011824
Comando: lift Valor: 0.422533 AngX: 34.9 AngZ: 25.0 AngY: -2.0 Hora:  14:23:26 : 0.422533 - Retardo:  0.0008512
Comando: lift Valor: 0.521008 AngX: 35.2 AngZ: 25.1 AngY: -1.6 Hora:  14:23:27 : 0.521008 - Retardo:  0.0006808
Comando: lift Valor: 0.652972 AngX: 34.4 AngZ: 25.1 AngY: -1.5 Hora:  14:23:27 : 0.652972 - Retardo:  0.0005417
Comando: lift Valor: 0.688732 AngX: 34.4 AngZ: 25.3 AngY: -1.9 Hora:  14:23:27 : 0.688732 - Retardo:  0.0006927
Comando: lift Valor: 0.706771 AngX: 35.4 AngZ: 25.5 AngY: -1.9 Hora:  14:23:28 : 0.706771 - Retardo:  0.0015663
Comando: lift Valor: 0.704722 AngX: 34.7 AngZ: 25.7 AngY: -1.5 Hora:  14:23:28 : 0.704722 - Retardo:  0.0006261
Comando: lift Valor: 0.722976 AngX: 34.7 AngZ: 26.0 AngY: -1.1 Hora:  14:23:28 : 0.722976 - Retardo:  0.0007046
Comando: lift Valor: 0.766356 AngX: 34.6 AngZ: 26.2 AngY: -1.2 Hora:  14:23:28 : 0.766356 - Retardo:  0.0007233
Comando: lift Valor: 0.808048 AngX: 34.2 AngZ: 26.3 AngY: -1.9 Hora:  14:23:29 : 0.808048 - Retardo:  0.0006654
Comando: lift Valor: 0.865809 AngX: 34.3 AngZ: 26.5 AngY: -1.4 Hora:  14:23:29 : 0.865809 - Retardo:  0.0007947
Comando: lift Valor: 0.873027 AngX: 35.2 AngZ: 26.6 AngY: -1.8 Hora:  14:23:29 : 0.873027 - Retardo:  0.0008548
Comando: lift Valor: 0.817575 AngX: 34.4 AngZ: 26.9 AngY: -1.2 Hora:  14:23:30 : 0.817575 - Retardo:  0.0014221
Comando: lift Valor: 0.57014 AngX: 34.7 AngZ: 27.1 AngY: -1.7 Hora:  14:23:30 : 0.57014 - Retardo:  0.000719
Comando: lift Valor: 0.254129 AngX: 35.6 AngZ: 27.4 AngY: -1.7 Hora:  14:23:30 : 0.254129 - Retardo:  0.0007166
Comando: lift Valor: 0.243689 AngX: 35.1 AngZ: 17.0 AngY: -0.9 Hora:  14:23:37 : 0.243689 - Retardo:  0.0011785
Comando: neutral Valor: 0.0 AngX: 35.6 AngZ: 16.3 AngY: 0.4 Hora:  14:23:37 : 0.0 - Retardo:  0.0006173
Comando: neutral Valor: 0.0 AngX: 34.5 AngZ: 16.8 AngY: -0.5 Hora:  14:23:38 : 0.0 - Retardo:  0.0005865
Comando: neutral Valor: 0.0 AngX: 35.2 AngZ: 17.3 AngY: -0.7 Hora:  14:23:38 : 0.0 - Retardo:  0.0005966
Comando: neutral Valor: 0.0 AngX: 35.2 AngZ: 17.6 AngY: -0.5 Hora:  14:23:38 : 0.0 - Retardo:  0.0006175
Comando: neutral Valor: 0.0 AngX: 35.1 AngZ: 17.2 AngY: -1.7 Hora:  14:23:39 : 0.0 - Retardo:  0.0006975
```

ANEXO 8. PRUEBA GIRO DE MUÑECA EN CONTRA DE LAS MANECILLAS DEL RELOJ

Python 3.6.8 Shell

File Edit Shell Debug Options Window Help

```
Comando: lift Valor: 0.907555 AngX: 38.4 AngZ: 42.1 AngY: -11.3 Hora:  15:17:25 : 0.907555 - Retardo:  0.0005474
derecha
Comando: lift Valor: 0.921042 AngX: 38.2 AngZ: 42.5 AngY: -12.8 Hora:  15:17:25 : 0.921042 - Retardo:  0.0007983
Comando: lift Valor: 0.929199 AngX: 38.5 AngZ: 42.4 AngY: -13.4 Hora:  15:17:25 : 0.929199 - Retardo:  0.0005254
Comando: lift Valor: 0.771188 AngX: 39.2 AngZ: 41.3 AngY: -12.2 Hora:  15:17:26 : 0.771188 - Retardo:  0.0005659
Comando: lift Valor: 0.416027 AngX: 40.2 AngZ: 39.7 AngY: -10.9 Hora:  15:17:26 : 0.416027 - Retardo:  0.0006698
Comando: lift Valor: 0.126658 AngX: 41.4 AngZ: 38.6 AngY: -7.7 Hora:  15:17:26 : 0.126658 - Retardo:  0.0005526
Comando: lift Valor: 0.023464 AngX: 41.3 AngZ: 38.5 AngY: -9.3 Hora:  15:17:26 : 0.023464 - Retardo:  0.0006069
Comando: lift Valor: 0.045698 AngX: 41.4 AngZ: 38.5 AngY: -8.5 Hora:  15:17:26 : 0.045698 - Retardo:  0.0005528
Comando: lift Valor: 0.137182 AngX: 41.9 AngZ: 38.4 AngY: -8.0 Hora:  15:17:27 : 0.137182 - Retardo:  0.0006129
Comando: lift Valor: 0.281949 AngX: 41.4 AngZ: 38.4 AngY: -8.5 Hora:  15:17:27 : 0.281949 - Retardo:  0.0006903
Comando: lift Valor: 0.410753 AngX: 41.9 AngZ: 38.3 AngY: -8.2 Hora:  15:17:27 : 0.410753 - Retardo:  0.0007284
Comando: lift Valor: 0.51909 AngX: 41.9 AngZ: 38.2 AngY: -8.4 Hora:  15:17:27 : 0.51909 - Retardo:  0.0005774
Comando: lift Valor: 0.609998 AngX: 42.5 AngZ: 38.2 AngY: -8.7 Hora:  15:17:28 : 0.609998 - Retardo:  0.0005415
Comando: lift Valor: 0.683878 AngX: 41.6 AngZ: 38.3 AngY: -7.3 Hora:  15:17:28 : 0.683878 - Retardo:  0.0005328
Comando: lift Valor: 0.735544 AngX: 41.7 AngZ: 38.2 AngY: -8.5 Hora:  15:17:28 : 0.735544 - Retardo:  0.0007189
Comando: lift Valor: 0.740262 AngX: 41.9 AngZ: 38.1 AngY: -8.2 Hora:  15:17:28 : 0.740262 - Retardo:  0.0005657
Comando: lift Valor: 0.772416 AngX: 42.2 AngZ: 38.1 AngY: -8.1 Hora:  15:17:28 : 0.772416 - Retardo:  0.0005515
Comando: lift Valor: 0.81383 AngX: 42.4 AngZ: 38.1 AngY: -8.6 Hora:  15:17:29 : 0.81383 - Retardo:  0.0006338
Comando: lift Valor: 0.833088 AngX: 41.7 AngZ: 38.1 AngY: -8.6 Hora:  15:17:29 : 0.833088 - Retardo:  0.0005973
Comando: lift Valor: 0.828798 AngX: 41.2 AngZ: 38.0 AngY: -7.8 Hora:  15:17:29 : 0.828798 - Retardo:  0.0005314
Comando: lift Valor: 0.838485 AngX: 42.5 AngZ: 37.9 AngY: -8.7 Hora:  15:17:29 : 0.838485 - Retardo:  0.0005351
Comando: lift Valor: 0.859743 AngX: 42.8 AngZ: 37.9 AngY: -8.3 Hora:  15:17:29 : 0.859743 - Retardo:  0.0006918
Comando: lift Valor: 0.877924 AngX: 41.8 AngZ: 37.9 AngY: -8.5 Hora:  15:17:30 : 0.877924 - Retardo:  0.0006107
Comando: lift Valor: 0.885865 AngX: 42.0 AngZ: 37.8 AngY: -8.2 Hora:  15:17:30 : 0.885865 - Retardo:  0.000559
Comando: lift Valor: 0.866981 AngX: 42.6 AngZ: 37.8 AngY: -8.4 Hora:  15:17:30 : 0.866981 - Retardo:  0.0006283
Comando: lift Valor: 0.808735 AngX: 41.6 AngZ: 37.8 AngY: -8.4 Hora:  15:17:30 : 0.808735 - Retardo:  0.0005674
Comando: lift Valor: 0.730463 AngX: 41.9 AngZ: 37.8 AngY: -8.4 Hora:  15:17:31 : 0.730463 - Retardo:  0.0005213
Comando: lift Valor: 0.682169 AngX: 41.6 AngZ: 37.8 AngY: -8.9 Hora:  15:17:31 : 0.682169 - Retardo:  0.0005406
Comando: lift Valor: 0.685824 AngX: 42.7 AngZ: 37.8 AngY: -8.7 Hora:  15:17:31 : 0.685824 - Retardo:  0.000738
Comando: lift Valor: 0.710896 AngX: 41.9 AngZ: 37.9 AngY: -8.6 Hora:  15:17:31 : 0.710896 - Retardo:  0.0005957
```

Python 3.6.8 Shell

File Edit Shell Debug Options Window Help

```
Comando: lift Valor: 0.686143 AngX: 41.7 AngZ: 37.7 AngY: -7.3 Hora:  15:17:31 : 0.686143 - Retardo:  0.0006113
Comando: lift Valor: 0.574612 AngX: 42.2 AngZ: 37.7 AngY: -7.8 Hora:  15:17:32 : 0.574612 - Retardo:  0.0006177
Comando: lift Valor: 0.507259 AngX: 42.9 AngZ: 37.7 AngY: -7.3 Hora:  15:17:32 : 0.507259 - Retardo:  0.0005996
Comando: lift Valor: 0.568961 AngX: 42.3 AngZ: 37.8 AngY: -8.9 Hora:  15:17:32 : 0.568961 - Retardo:  0.0006752
Comando: lift Valor: 0.64995 AngX: 42.2 AngZ: 37.8 AngY: -8.5 Hora:  15:17:32 : 0.64995 - Retardo:  0.0005856
Comando: lift Valor: 0.714415 AngX: 42.3 AngZ: 37.8 AngY: -7.6 Hora:  15:17:32 : 0.714415 - Retardo:  0.0006448
Comando: lift Valor: 0.765054 AngX: 42.6 AngZ: 37.7 AngY: -9.1 Hora:  15:17:33 : 0.765054 - Retardo:  0.0006958
Comando: lift Valor: 0.807667 AngX: 42.6 AngZ: 37.8 AngY: -9.3 Hora:  15:17:33 : 0.807667 - Retardo:  0.0006228
Comando: lift Valor: 0.842083 AngX: 42.4 AngZ: 37.8 AngY: -8.0 Hora:  15:17:33 : 0.842083 - Retardo:  0.0006758
Comando: lift Valor: 0.867294 AngX: 42.4 AngZ: 37.7 AngY: -7.7 Hora:  15:17:33 : 0.867294 - Retardo:  0.000604
Comando: lift Valor: 0.86602 AngX: 42.7 AngZ: 37.6 AngY: -8.6 Hora:  15:17:33 : 0.86602 - Retardo:  0.0005329
Comando: lift Valor: 0.806233 AngX: 41.9 AngZ: 37.7 AngY: -8.4 Hora:  15:17:34 : 0.806233 - Retardo:  0.0005598
Comando: lift Valor: 0.753003 AngX: 42.1 AngZ: 37.7 AngY: -8.2 Hora:  15:17:34 : 0.753003 - Retardo:  0.0006426
Comando: lift Valor: 0.749153 AngX: 42.4 AngZ: 37.7 AngY: -7.8 Hora:  15:17:34 : 0.749153 - Retardo:  0.0006098
Comando: lift Valor: 0.751426 AngX: 43.1 AngZ: 37.7 AngY: -7.7 Hora:  15:17:34 : 0.751426 - Retardo:  0.0010687
Comando: lift Valor: 0.730968 AngX: 42.5 AngZ: 37.8 AngY: -7.7 Hora:  15:17:35 : 0.730968 - Retardo:  0.0005317
Comando: lift Valor: 0.68557 AngX: 42.5 AngZ: 37.7 AngY: -8.8 Hora:  15:17:35 : 0.68557 - Retardo:  0.0005726
Comando: lift Valor: 0.633569 AngX: 42.4 AngZ: 37.8 AngY: -8.0 Hora:  15:17:35 : 0.633569 - Retardo:  0.0008102
Comando: lift Valor: 0.519195 AngX: 43.2 AngZ: 38.0 AngY: -8.3 Hora:  15:17:35 : 0.519195 - Retardo:  0.0007487
Comando: lift Valor: 0.29798 AngX: 42.3 AngZ: 37.9 AngY: -7.9 Hora:  15:17:35 : 0.29798 - Retardo:  0.0006915
Comando: neutral Valor: 0.0 AngX: 42.2 AngZ: 37.8 AngY: -7.6 Hora:  15:17:36 : 0.0 - Retardo:  0.0006141
Comando: neutral Valor: 0.0 AngX: 42.5 AngZ: 37.9 AngY: -8.6 Hora:  15:17:36 : 0.0 - Retardo:  0.0006333
Comando: neutral Valor: 0.0 AngX: 42.5 AngZ: 38.1 AngY: -8.2 Hora:  15:17:36 : 0.0 - Retardo:  0.0005085
Comando: neutral Valor: 0.0 AngX: 42.5 AngZ: 38.1 AngY: -7.2 Hora:  15:17:36 : 0.0 - Retardo:  0.0005474
Comando: lift Valor: 0.367532 AngX: 42.8 AngZ: 38.1 AngY: -8.6 Hora:  15:17:36 : 0.367532 - Retardo:  0.0005667
Comando: lift Valor: 0.461856 AngX: 42.5 AngZ: 38.1 AngY: -7.9 Hora:  15:17:37 : 0.461856 - Retardo:  0.0007116
Comando: lift Valor: 0.5593 AngX: 42.4 AngZ: 38.3 AngY: -7.7 Hora:  15:17:37 : 0.5593 - Retardo:  0.0007387
Comando: lift Valor: 0.640837 AngX: 42.7 AngZ: 38.3 AngY: -9.1 Hora:  15:17:37 : 0.640837 - Retardo:  0.00058
Comando: neutral Valor: 0.0 AngX: 42.0 AngZ: 38.3 AngY: -7.8 Hora:  15:17:37 : 0.0 - Retardo:  0.0007909
Comando: neutral Valor: 0.0 AngX: 43.0 AngZ: 38.3 AngY: -7.5 Hora:  15:17:37 : 0.0 - Retardo:  0.0005479
Comando: neutral Valor: 0.0 AngX: 43.4 AngZ: 38.4 AngY: -8.8 Hora:  15:17:38 : 0.0 - Retardo:  0.0005711
```