ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

Diseño e implementación de prototipo portátil para captura de señales mioeléctricas en la extremidad superior derecha de una persona

PROYECTO INTEGRADOR

Previo la obtención del Título de:

Ingeniero en Electrónica y Telecomunicaciones

Presentado por:

Germán Alexander Granados Weisson

Katherine Gladys Molina Ochoa

GUAYAQUIL - ECUADOR

Año: 2018

DEDICATORIA

Dedico este proyecto a mis padres, por el amor y apoyo incondicional que siempre me han brindado y por guiarme para ser alguien de bien.

A mis hermanos, Camilo y Diego, sepan que siempre estaré para ayudarlos.

A Edgar, por creer en mí, apoyarme en los buenos y malos momentos y motivarme a siempre aprender más.

Katherine Molina Ochoa

A mis hermanos quienes han sido motor fundamental en mi vida para poder seguir avanzando pese a las circunstancias adversas que nos presentó esta.

A mi novia Melany Adriana Vega Pinto, quien ha llegado a ser una de las personas más importantes en mi vida y ha creído en mi en todo momento sin importar las adversidades; por lo cual me ha dado fuerzas para continuar con paso firme.

Germán Granados Weisson

AGRADECIMIENTOS

Agradezco a mis padres, por todo el esfuerzo y sacrificios que realizaron durante todos mis años de estudio.

Al Ing. Efrén Herrera, por la oportunidad brindada para incursionar en el mundo de la investigación y aportar científicamente a la comunidad politécnica.

Al Ing. Washington Medina, por su guía y paciencia durante la elaboración de este proyecto.

Katherine Molina Ochoa

Mi más sincero agradecimiento a Dios, quien me dio las fuerzas para continuar en la obtención de este logro; así como mi padre Germán Fausto Granados Ponce, dado que gracias a sus sabios consejos me lograba motivar a no rendirme e intentarlo hasta el final. Él me dice "lo peor que puede pasar es no lograrlo, pero al menos sabrás que lo diste todo".

Germán Granados Weisson.

DECLARACIÓN EXPRESA

"Los derechos de titularidad y explotación, nos corresponde conforme al reglamento de propiedad intelectual de la institución; Germán Alexander Granados Weisson y Katherine Gladys Molina Ochoa damos nuestro consentimiento para que la ESPOL realice la comunicación pública de la obra por cualquier medio con el fin de promover la consulta, difusión y uso público de la producción intelectual"

G. Alex. W

.....

Germán Alexander Granados Weisson

.....

Katherine Gladys Molina Ochoa

EVALUADORES

M.Sc Washington Medina Moreira

PROFESOR DE LA MATERIA

Ing. Miguel Molina Villacís

PROFESOR TUTOR

RESUMEN

Las señales mioeléctricas son pequeños impulsos eléctricos que se generan por los movimientos de los músculos en el cuerpo. Aplicaciones que vienen del estudio de estas señales han aumentado con el pasar del tiempo, por ello existen diversas formas para su adquisición. Este proyecto tiene como objetivo diseñar e implementar un prototipo portátil de un equipo para adquirir, visualizar, y transmitir/almacenar las señales mioeléctricas generadas desde el brazo derecho utilizando la Raspberry Pi 3 B+ junto al brazalete de sensores Myo Armband. Este prototipo portátil debe eliminar las limitaciones que provoca el realizar la adquisición de datos en un lugar fijo, tales como el bajo número de voluntarios para realizar experimentos, lo que trae consigo una menor cantidad de datos para analizar, además de un rango limitado en cuanto a las edades de los voluntarios. Los algoritmos para cada tarea requerida fueron desarrollados en Python debido a la versatilidad que ofrece este lenguaje de programación. Se realizaron pruebas a varias personas de diferentes edades y contextura física mientras realizaban diferentes movimientos con la mano derecha, y con las gráficas de EMG[mV] vs t[s] que se generaron cada vez, se observó qué estaba ocurriendo durante cada experimento. Los datos se adquirieron con éxito y fueron enviados vía e-mail o almacenados en la Raspberry dependiendo de si existía o no conexión a internet en el prototipo. Con los resultados obtenidos se pudo comprobar el correcto funcionamiento del prototipo portátil y se logró cumplir con todos los requerimientos exigidos por el cliente.

Palabras Clave: mioeléctrico, raspberry, myo armband, python.

ABSTRACT

Myoelectric signals are small electrical impulses that are generated by the movements of the muscles in the body. The applications that come from the study of these signals have increased over time, so there are several ways to acquire them. This project aims to design and implement a portable prototype of a device to acquire, visualize, and transmit or store the myoelectric signals generated from the right arm using the Raspberry Pi 3 B+ and the Myo Armband. This portable prototype must eliminate the limitations that cause the acquisition of data in a fixed place, such as the low number of volunteers to carry out experiments, which brings with it a smaller amount of data to analyze, in addition to a limited range at the ages of the volunteers. The algorithms for each required task were developed in Python due to the versatility offered by this programming language. Tests were performed on several people of different ages and physical frame while performing different movements with the right hand, and with the EMG[mV] vs. t[s] charts that were generated each time, what was happening during each experiment was observed. The data was acquired successfully and was sent via e-mail or stored in the raspberry depending on whether or not there was an Internet connection on the prototype. With the results obtained, it was possible to verify the correct functioning of the prototype and all the requirements demanded by the client were fulfilled.

Keywords: myoelectric, raspberry, myo armband, python.

ÍNDICE GENERAL

ABREVIATURAS

ESPOL	Escuela Superior Politécnica del Litoral
FIEC	Facultad de Ingeniería en Electricidad y Computación
GIACI	Grupo de Investigación en Automatización y Control Industrial
EMG	Electromiografía, Electromiográfica, mioeléctrica
SBC	Simple Board Computer
SDK	Software Development Kit
CSV	Comma Separated Values
GIMP	GNU Image Manipulation Program
SMTP	Simple Mail Transfer Protocol
SBC	Simple Board Computer
TCP/IP	Transmission Control Protocol/Internet Protocol
PC	Personal Computer
VNC	Virtual Network Computing
SSH	Secure SHell
DMZ	Demilitarized Zone
URL	Uniform Resource Locator

SIMBOLOGÍA

mV	Milivoltio
t	Tiempo
°C	Grados Celsius
lb	Libras
cm	Centímetros

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

CAPÍTULO 1

1. INTRODUCCIÓN

El estudio de las señales mioeléctricas ha ido aumentando con el paso de los años debido a la gran cantidad de aplicaciones que tienen en la vida diaria, tales como mejorar el diseño de prótesis humanas, activar dispositivos electrónicos con movimientos leves de los músculos, e incluso en un futuro, crear un teléfono móvil que se pueda usar o incluso implantar en el cuerpo. La información proveniente de los músculos es de gran importancia para el desarrollo de proyectos en los que el factor principal es el movimiento humano. Por estas razones el desarrollo de diferentes sistemas para la adquisición de señales mioeléctricas es de gran importancia.

Actualmente existen diferentes formas para adquirir la señal mioeléctrica de una parte específica de cuerpo, las más comunes son mediante el uso de electrodos de aguja (forma invasiva) o con electrodos superficiales (forma no invasiva). Estos electrodos son conectados en placas electrónicas para realizar el filtrado y transmisión de las señales y luego ser procesadas en diferentes softwares según el fin que se desee.

Una tercera forma de adquirir las señales mioeléctricas es usando el brazalete de sensores desarrollado por Thalmic Labs, Myo Armband. Debido a la practicidad y mayor movilidad que permite este método fue el seleccionado para el desarrollo de este proyecto.

Conjuntamente se usará la Computadora de Placa Simple (SBC, por sus siglas en inglés) Raspberry Pi 3 B+, la cual se comunicará con el Myo Armband mediante el protocolo Bluetooth. En esta SBC se implementarán los algoritmos necesarios para la adquisición, almacenamiento y transmisión de las señales mioeléctricas, quedando listas para su procesamiento según la necesidad que se presente.

El fin principal de seleccionar estos dispositivos para el diseño del prototipo es su portabilidad. Esto permitirá que los investigadores de GIACI puedan adquirir los datos mioeléctricos de diferentes personas en diferentes lugares.

1.1 Descripción del problema

Dentro de la Escuela Superior Politécnica del Litoral (ESPOL), uno de los proyectos en los que el Grupo de Investigación en Automatización y Control Industrial (GIACI) está actualmente trabajando, trata de la adquisición y análisis de señales mioeléctricas emitidas desde la extremidad superior derecha de una persona con el fin de encontrar una relación entre la temperatura y la señal mioeléctrica. Para ello cuentan con equipos de adquisición de señales mioeléctricas como son el Myo Armband y la placa Olimex EKG-EMG Shield. Estos equipos requieren de una interfaz por computador para adquirir, procesar y analizar las señales mioeléctricas. Para disponer de un mayor rango de edades en los que los resultados de la investigación se cumplan, el Grupo se ha visto en la necesidad de plantear la idea de traer personas a ESPOL para realizar las pruebas, pero esto ocasionaría gastos de transporte y alimentación de los voluntarios, y generaría una pérdida de recursos económicos al proyecto. Otra razón por la que la idea tuvo que ser dejada de lado es que no todos los voluntarios cuentan con el tiempo requerido para trasladarse hasta ESPOL.

A pesar de estos inconvenientes, todo el proceso de adquisición es realizado en un laboratorio dentro de las instalaciones de la Facultad de Ingeniería en Electricidad y Computación (FIEC), de ESPOL. Por ende, todas las pruebas realizadas han sido con estudiantes de la institución, cuyo intervalo de edad se encuentra entre 18 a 26 años.

Para solucionar esta problemática, se presenta la necesidad de la elaboración de un equipo portátil que permita adquirir, almacenar y visualizar las señales mioeléctricas del brazo derecho de una persona.

1.2 Justificación del problema

Las ventajas que traerá llevar a cabo este proyecto son de variada índole. Para empezar, se elimina la necesidad de traer voluntarios de fuera de ESPOL ya que, al ser portátil el equipo, el investigador podrá trasladarse a varios lugares para adquirir datos. Además, se ampliará el rango de edades del estudio, esto permitirá al grupo contar con una mayor variedad en los datos con los que trabajar para el proyecto.

Otra de las ventajas es que el proyecto puede realizarse con dispositivos ya existentes en el mercado, como son el brazalete Myo Armband y la SBC Raspberry Pi, por lo cual se evita el trabajo de diseñar e implementar nuevos equipos para la adquisición y transmisión de las señales. Por ende, el tiempo de implementación del equipo será menor.

1.3 Objetivos

1.3.1 Objetivo General

Diseñar e implementar un prototipo portátil de un sistema de captura de las señales mioeléctricas generadas desde la extremidad superior derecha de una persona.

1.3.2 Objetivos Específicos

1. Estudiar el funcionamiento del Myo Armband.
2. Establecer la interfaz de comunicación entre el sensor Myo Armband y la SBC Raspberry Pi.
3. Diseñar algoritmo para adquisición de datos mioeléctricos obtenidos del Myo Armband.
4. Diseñar algoritmo para almacenamiento y transmisión de los datos mioeléctricos adquiridos.
5. Diseñar algoritmo para mostrar la gráfica de la señal mioeléctrica adquirida en la pantalla del equipo portátil.
6. Instalar software para la creación de un servidor y de nube.

1.4 Marco teórico

1.4.1 Electromiografía y Señales Mioeléctricas

La electromiografía (EMG) es una técnica experimental relacionada con el desarrollo, registro y análisis de señales mioeléctricas. Las señales mioeléctricas están formadas por variaciones fisiológicas en el estado de las membranas de fibras musculares. [1]

1.4.2 Myo Armband

El brazalete de sensores Myo Armband es un dispositivo fabricado por Thalmic Labs, que permite reconocer los movimientos musculares realizados por el brazo y antebrazo. Este dispositivo permite al usuario controlar diversos dispositivos de forma inalámbrica utilizando diferentes movimientos de la mano, siendo su uso mayormente bajo el ambiente Windows. Los sensores electromiográficos (EMG) del brazalete captan la actividad eléctrica que se produce en los músculos al realizar cualquier movimiento, y, combinados con un giroscopio, un acelerómetro y un magnetómetro, permite reconocer diferentes gestos. [2]

La forma física de la pulsera se muestra en la Figura 1.1.

Figura 1.1 Myo Armband [3]

A continuación, se presentan las características técnicas del brazalete [3]:

- Tamaño, Peso y Dimensiones

  En la Tabla 1.1 se describen las características físicas del brazalete.

Tabla 1.1 Características físicas del Myo Armband [3]

TAMAÑO	Ampliable para una circunferencia del antebrazo entre 19 - 34 cm.
PESO	93 gramos
ESPESOR	1.143 cm

- Dispositivos compatibles

  Para establecer una comunicación con el brazalete es necesario que los dispositivos cuenten con un sistema operativo que cumpla con las características necesarias para que haya compatibilidad con el Kit de Desarrollo de Software (SDK, por sus siglas en inglés) del Myo Armband. En la Tabla 1.2 se detallan los dispositivos son compatibles para conectarse con el brazalete.

Tabla 1.2 Sistemas operativos compatibles con el SDK del Myo Armband [3]

Dispositivo		Detalle
WINDOWS	Windows 7	Todos con el adaptador USB Bluetooth® incluido y OpenGL 2.1 o superior
	Windows 8	
	Windows 10	
MAC	OS X 10.8 (Mountain Lion) y superior	Con adaptador USB Bluetooth® incluido
IOS	7.0 o superior	Para los siguientes dispositivos: iPad 3ra y 4ta generación iPad Air, Air 2 iPhone 4s, 5, 5c, 5s, 6, 6 Plus iPod Touch 5ta generación iPad Mini 1ra y 2da generación iPad Mini 3
ANDROID	Android 4.3 (Jelly Bean) y superior	El dispositivo debe tener una radio Bluetooth® compatible con Bluetooth® 4.0 LE.

- Hardware

  En la Tabla 1.3 se detallan las características del hardware del brazalete.

Tabla 1.3 Características del hardware del Myo Armband [3]

Sensores	Sensores EMG de acero inoxidable de grado médico, IMU de nueve ejes altamente sensible que contiene giroscopio, acelerómetro y magnetómetro de tres ejes
LEDs	LEDs indicadores dobles
Procesador	ARM Cortex M4
Retroalimentación Háptica	Vibraciones cortas, medianas y largas

- Gestos y Movimiento

  Posee sensores de movimiento altamente sensibles. Además, tiene un conjunto de gestos manuales detectados por sensores musculares EMG patentados. Los gestos se pueden constatar a continuación (Ver Figura 1.2).

Figura 1.2 Myo Armband: Gestos patentados [3]

- Energía y Batería

  Carga micro-USB. Batería de iones de litio recargable incorporada Un día completo de uso sin carga.

- Comunicación

  Tecnología inalámbrica inteligente Bluetooth®.

- Numeración de los sensores del brazalete

  En la Figura 1.3 se muestra la numeración que identifica a cada sensor.

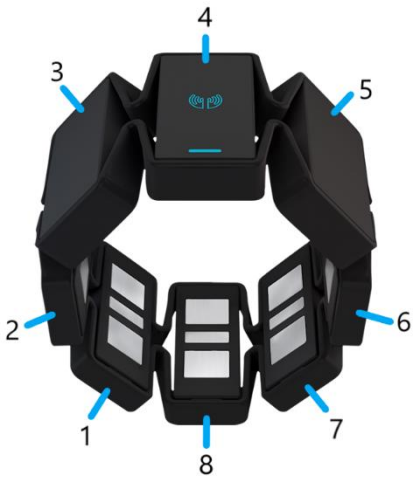


Figura 1.3 Numeración de cada sensor del Myo Armband

1.4.3 Raspberry Pi 3 B+

El Raspberry Pi es un dispositivo con características de software similares a las de un computador, pero con la ventaja de que tiene un tamaño reducido. Fue desarrollado por la Fundación Raspberry Pi con el fin de otorgar la oportunidad de desarrollar una gran variedad de proyectos y fomentar el aprendizaje de ciencias computacionales en las instituciones educativas.

El sistema operativo oficial de la placa es una versión adaptada del Debian/GNU Linux, denominado Raspbian.

En la Tabla 1.4 se detallan las características más relevantes del Raspberry Pi 3 B+.

Tabla 1.4 Especificaciones técnicas de la Raspberry Pi 3 B+ [4]

PROCESADOR	Broadcom BCM2837B0, Cortex-A53 SoC de 64-bit a 1.4GHz
MEMORIA	1GB LPDDR2 SDRAM
CONECTIVIDAD	LAN Inalámbrica de 2.4GHz y 5GHz IEEE 802.11.b/g/n/ac, Bluetooth 4.2, BLE
	Gigabit Ethernet sobre USB 2.0 (rendimiento máximo de 300 Mbps)
	4 puertos USB 2.0
ACCESO	40 pines GPIO
VIDEO Y SONIDO	HDMI de tamaño completo
	Puerto de visualización MIPI DSI
	Puerto de cámara MIPI CSI
	Salida estéreo de 4 polos y puerto de video compuesto
MULTIMEDIA	Decodificación H.264, MPEG-4 (1080p30); Codificación H.264 (1080p30); OpenGL ES 1.1, 2.0 gráficos
SOPORTE DE TARJETA SD	Formato Micro SD para cargar el sistema operativo y el almacenamiento de datos
ALIMENTACIÓN	5V / 2.5A DC a través del conector micro USB
	5V DC a través del encabezado GPIO
	Potencia a través de Ethernet (PoE) habilitada (requiere un PoE HAT independiente)
TEMPERATURA DE FUNCIONAMIENTO	0-50 °C

1.4.4 Estado del arte

El Grupo de Ingeniería en Rehabilitación de la Universidad Nacional del Nordeste, Argentina, trabajó en el proyecto “Sistema de adquisición y visualización de señales mioeléctricas” [5]. Con éxito lograron construir un prototipo portátil para

adquirir las señales mioeléctricas para luego visualizarlas en un PC. Todo esto con el fin de proporcionar información sobre el estado de los músculos. Para la adquisición utilizaron electrodos superficiales sobre un músculo y para la transmisión hacia una PC elaboraron una placa con diferentes elementos electrónicos para la amplificación y filtrado de la señal.

En marzo del 2013, investigadores del Instituto Tecnológico de Orizaba, México, llevaron a cabo un proyecto con título "Diseño de un sistema de adquisición de señales electromiográficas inalámbrico" [6], cuyo objetivo principal era el de adquirir y mostrar en tiempo real la activación muscular. Mediante el uso de 3 electrodos superficiales y amplificadores de instrumentación, lograron adquirir y amplificar las señales producidas por los músculos, para luego aplicarles métodos de filtrado y procesamiento y por último realizar la transmisión inalámbrica de estas señales. Mediante pruebas a diferentes voluntarios pudieron observar con éxito la reacción de las señales mioeléctricas debido a las variaciones de fuerza en los músculos.

En el 2017, investigadores de diferentes universidades en Bucharest, Romania, se reunieron para desarrollar el proyecto “A new Method for Myoelectric Signal Acquisition: Preparing the Patients to Efficiently Use an Artificial Arm” [7]. Este proyecto tenía como objetivo desarrollar una nueva metodología en cirugía de amputación dedicada a los pacientes que desean usar una prótesis mioeléctrica de antebrazo. Las señales de control para una prótesis mioeléctrica son las señales EMG de superficie. Para ello desarrollaron un equipo para adquirir y amplificar la señal mioeléctrica. Usaron electrodos superficiales para la adquisición de las señales mioeléctricas y para la transmisión y procesamiento de las señales los dispositivos EMG que se utilizaron fueron el Contec CMS6600 (PUB) y Nicolet EDX (UEH), junto con su software EMG incluido. También desarrollaron un dispositivo de biofeedback portátil. La metodología propuesta se evaluó experimentalmente en una cirugía de retoque de muñón, permitiendo la comparación de las señales antes de la cirugía con las posteriores a la cirugía y con las obtenidas después de completar el entrenamiento post-cirugía. Al pasar por estas etapas, se encontró una mejora constante de la amplitud e

independencia de las señales EMG recolectadas del muñón. Al final, el paciente pudo controlar un modelo experimental de mano artificial, mediante cinco señales EMG diferentes, recogidas de los músculos del muñón. Los resultados parecen validar, en este caso, las hipótesis de trabajo utilizadas como base para el método quirúrgico y el entrenamiento de biorretroalimentación.

1.5 Alcance del Proyecto

La solución se realizará con una Raspberry Pi debido a que es un dispositivo con características similares a una computadora, con la diferencia que trabaja con un sistema operativo basado en Linux, además cuenta con dimensiones semejantes a las de una tarjeta de crédito. Esta computadora se comunicará con los sensores del Myo Armband usando el protocolo Bluetooth para realizar la adquisición y transmisión de datos. Conjuntamente se grafica la señal miográfica en una pantalla táctil de 7" conectada a la Raspberry.

La alimentación de la Raspberry Pi es de 5 voltios, 2.5 amperios, que podrá ser suministrado por una fuente conmutada conectada a la toma de 110 voltios, y cabe mencionar que esta minicomputadora tiene un bajo consumo de potencia.

El equipo portátil tendrá 2 modos de operación: El primero, en los sitios donde no haya internet, los datos adquiridos serán almacenados en un archivo de formato .csv dentro de la Raspberry. El segundo, si se tiene acceso a internet, los datos se almacenarán de igual forma en un archivo con formato .csv y se enviarán vía internet a un correo electrónico especificado en el código.

1.6 Metodología General

En el diagrama de flujo presentado a continuación se muestra, de una forma general, la metodología que será usada para el desarrollo de este proyecto. El detalle de cada bloque se detalla en el Capítulo 2.

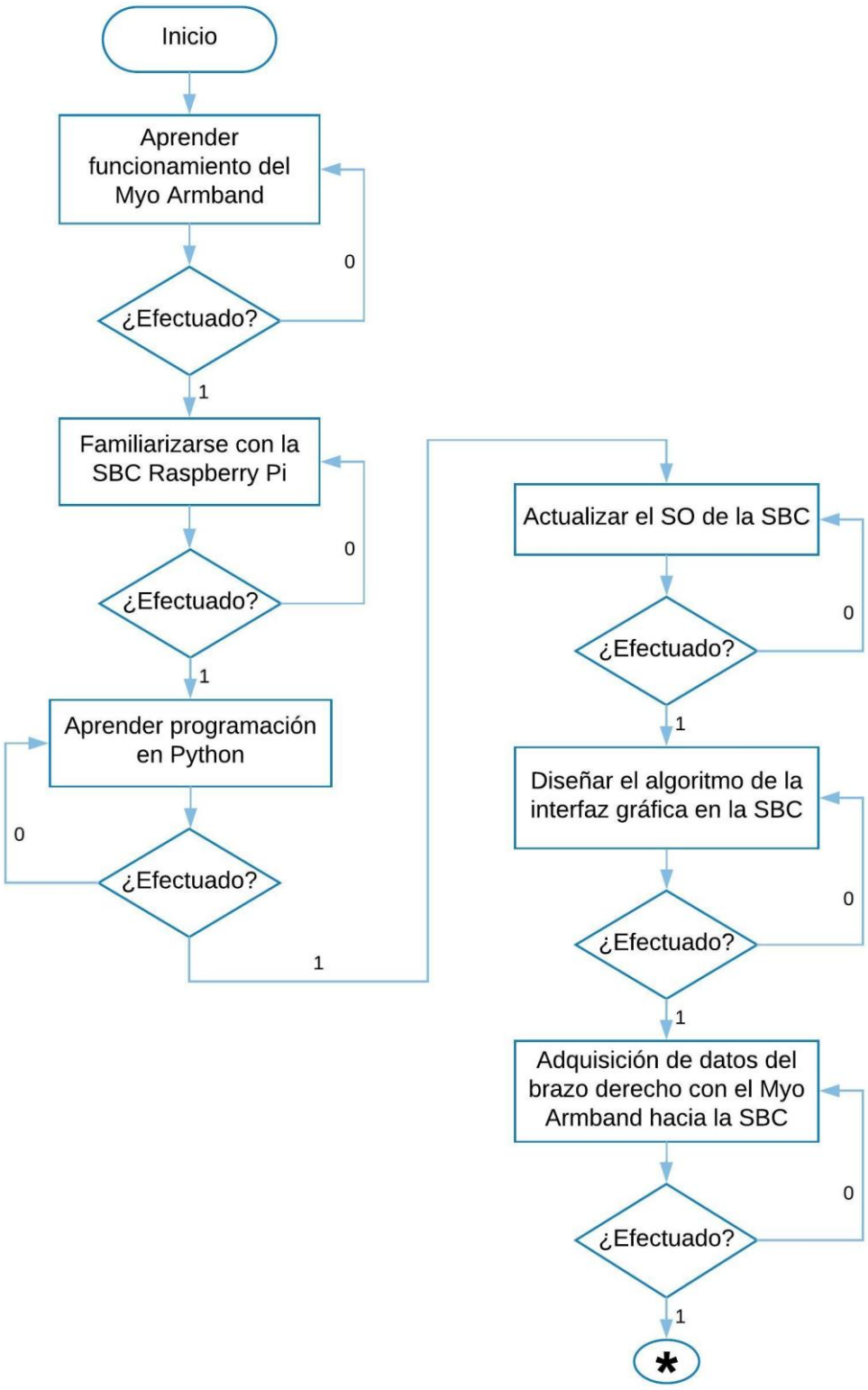


Figura 1.4 Diagrama de Flujo de todo el sistema (parte 1)

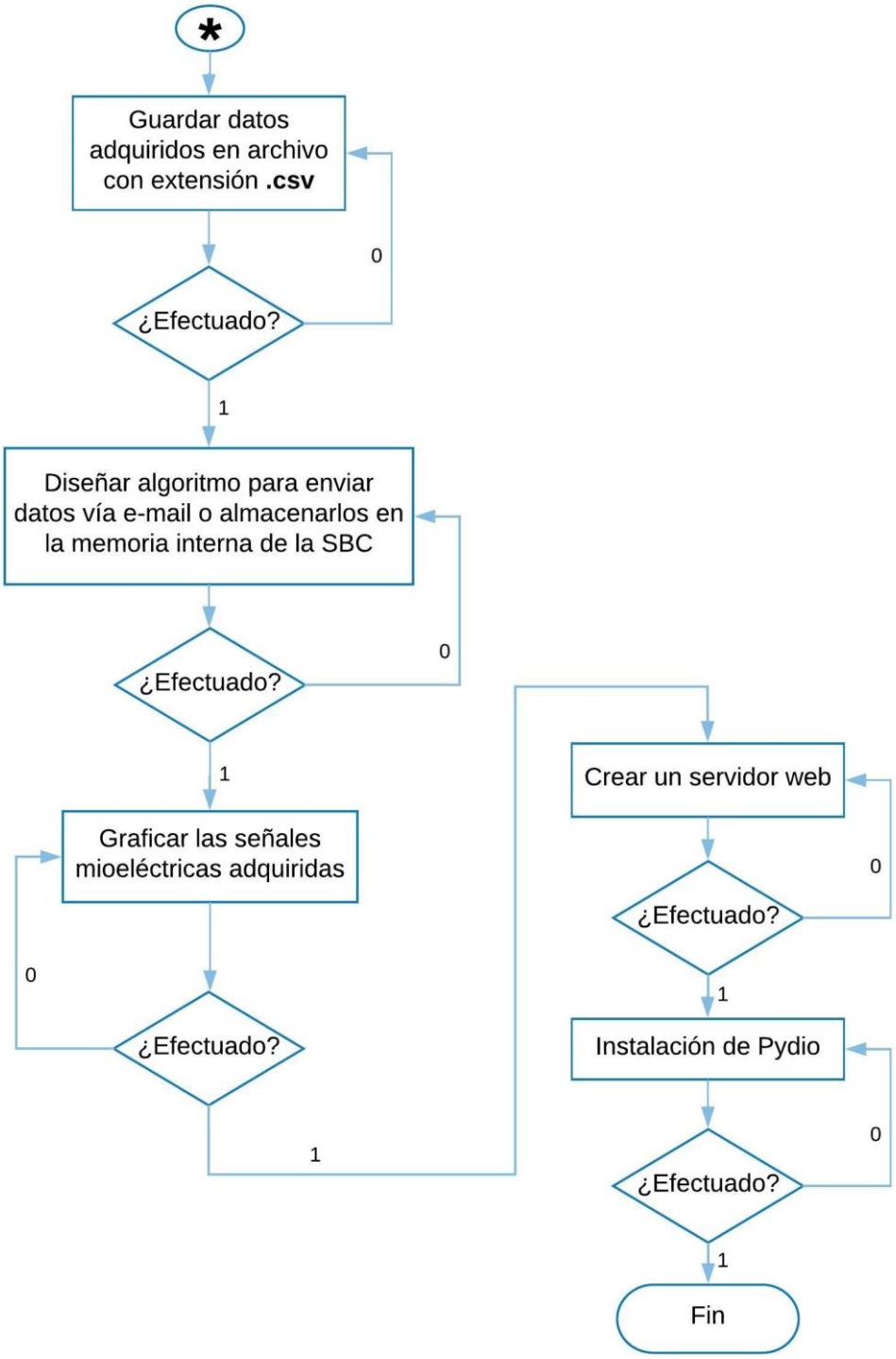


Figura 1.5 Diagrama de Flujo de todo el sistema (parte 2)

CAPÍTULO 2

2. METODOLOGÍA Y DISEÑO DEL PROTOTIPO

2.1 Metodología

Actualmente existen varias alternativas para la adquisición de señales mioeléctricas, dos de ellas se describen de forma general a continuación:

- Ejecutar la adquisición con electrodos sobre partes determinadas del brazo derecho; estos electrodos están conectados a una tarjeta electrónica que procesará y guardará las señales y permitirá la visualización de estas en un osciloscopio.
- La adquisición realizarla con 3 electrodos (Rojo, Blanco y Negro) conectados a una tarjeta Olimex EKG-EMG Shield, la cual mediante un amplificador operacional de instrumentación mide la diferencia de potencial entre los electrodos rojo y blanco con el electrodo negro como referencia. Luego de amplificar y filtrar la señal, se la adquiere con una DAQ, y esta tarjeta estará conectada a una computadora para transmitir las señales y procesarlas mediante el uso del software Simulink.

El inconveniente de estos métodos es que es necesario un computador para poder guardar y visualizar las señales adquiridas.

Otra opción es utilizar el Myo Armband para adquirir y transmitir vía Bluetooth las señales mioeléctricas del brazo derecho, estas señales serán receptadas en una computadora para realizar el procesamiento requerido en diferentes softwares.

Usando como base este método se propone el uso del Myo Armband junto a la Raspberry Pi para elaborar el prototipo de este proyecto y así simplificar la cantidad de hardware que se usa actualmente y permitir que el equipo sea portátil.

Se seleccionó la Raspberry Pi debido a que esta tarjeta es un computador de dimensiones reducidas, similares a las de una tarjeta de crédito.

A continuación, se describe la metodología para la elaboración del equipo.

Se empezará con un estudio del funcionamiento del brazalete de sensores Myo Armband, para ello se conectará el brazalete a un PC con sistema operativo Windows 10. Esta conexión se realizará primero instalando el software Myo Connect. Al terminar la instalación se conectará el módulo Bluetooth del brazalete al PC, y se comprueba en el Armband Manager del Myo Connect que el brazalete está correctamente conectado.

Con la conexión establecida se procederá a probar las funciones que tiene la pulsera y analizar de forma general su funcionamiento en Windows 10.

Luego de haberse familiarizado con el funcionamiento del brazalete, se procederá a conectarlo con la Raspberry Pi, la cual tiene un sistema operativo basado en Linux.

De igual manera se utilizará el protocolo bluetooth para establecer la comunicación entre el brazalete y la SBC, y se deberá instalar el software Pyo Connect para realizar la conexión.

La SBC cuenta con una pantalla táctil de 7" que permite interactuar con los usuarios, por lo cual se dispuso a crear una aplicación que será visualizada en dicha pantalla y que sirva de enlace entre la SBC y el Myo Armband con el fin de poder manipular los dispositivos. Bajo esta interfaz creada se pretende obtener información relevante de los voluntarios, al igual que almacenar y/o enviar los datos adquiridos. La información obtenida será almacenada en un archivo con formato .csv (comma separated values) y enviada mediante correo electrónico, por lo cual, el modo de operación dependerá de si exista o no conexión a internet mediante WiFi; dado el caso en el que no se cuente con dicha red, los datos se almacenarán en la memoria propia del dispositivo hasta poder conseguir una conexión a internet. Todos los archivos que se obtengan podrán ser subidos a una nube que se creará desde cero. Dado que la información es muy importante no se pueden usar servicios como Dropbox, Google Drive o similares debido a la falta de privacidad en los mismos.

2.2 Diagrama de Bloques

En la Figura 2.1 se visualiza la estructura que se seguirá en la elaboración del prototipo.

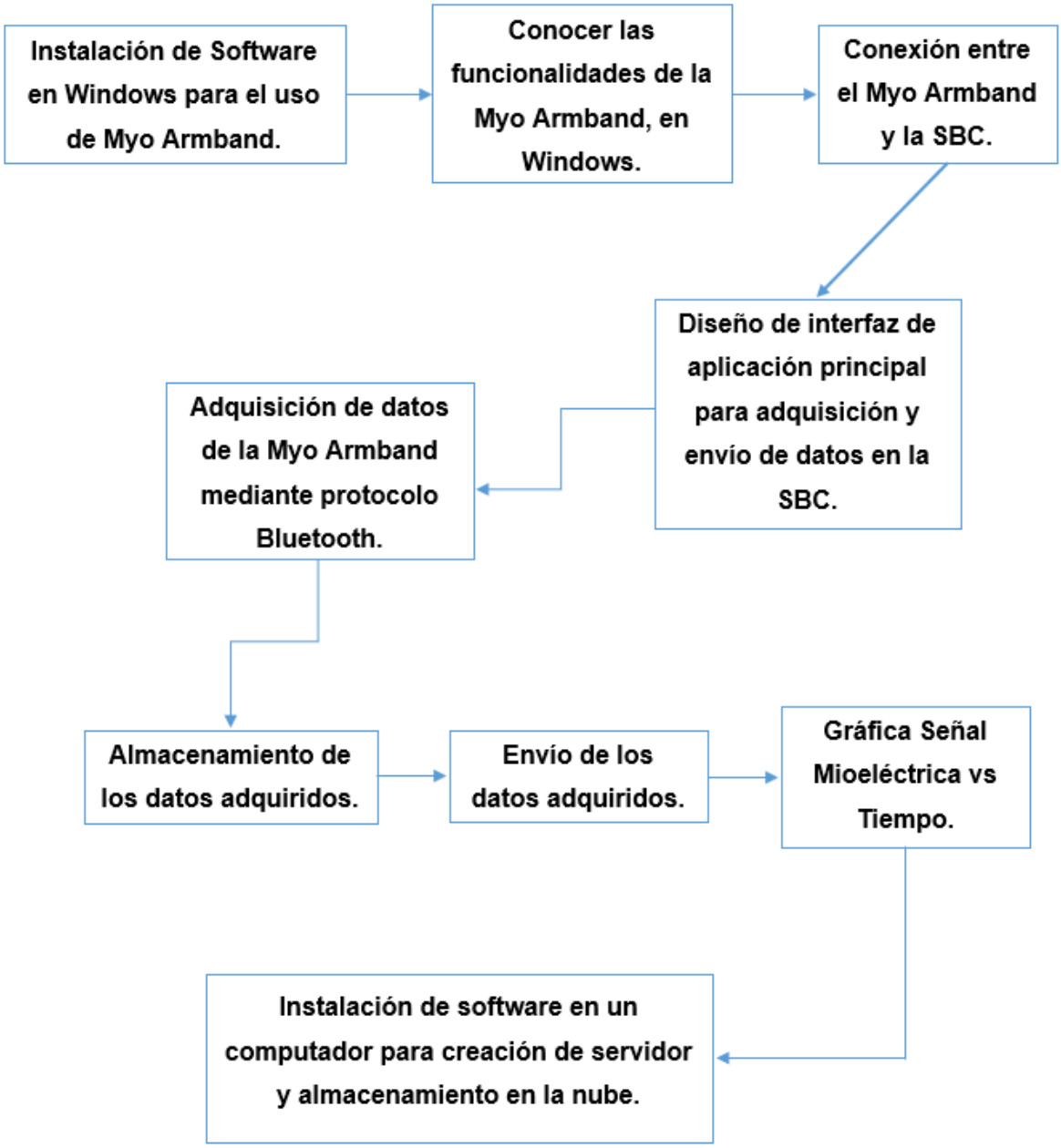


Figura 2.1 Diagrama de bloques principal del sistema

2.3 Funcionamiento del brazalete de sensores Myo Armband

Antes de establecer una conexión con la Raspberry hay que familiarizarse con el funcionamiento del brazalete en un sistema operativo de uso más común, como lo es Windows 10.

Existe un software para establecer una conexión entre el Myo Armband y Windows, llamado Myo Connect. Este software se lo puede descargar desde la página oficial de Myo, https://www.myo.com/start. Luego de seguir los pasos respectivos para la instalación y al dar doble clic en el programa, en la barra de tarea se mostrará el ícono principal del Myo Connect (ver Figura 2.2)

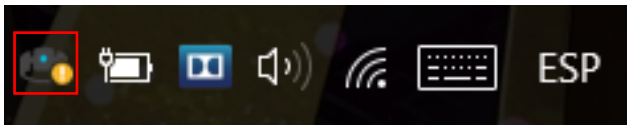


Figura 2.2 Ícono de Myo Connect

Al dar clic derecho se muestran las opciones de la Figura 2.3, para conectar el brazalete a la computadora se dará un clic sobre Armband Manager.

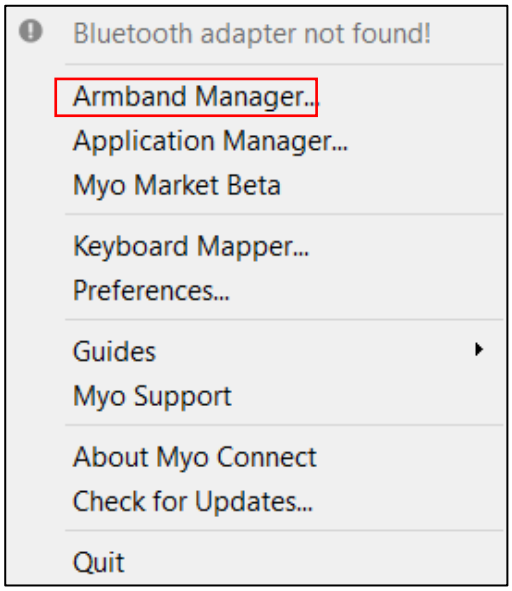


Figura 2.3 Lista de herramientas que proporciona Myo Connect

Se abrirá la ventana principal del programa (ver Figura 2.4). Aquí se establece la conexión con el brazalete además de otras funciones como hacer que vibre o apagarlo.

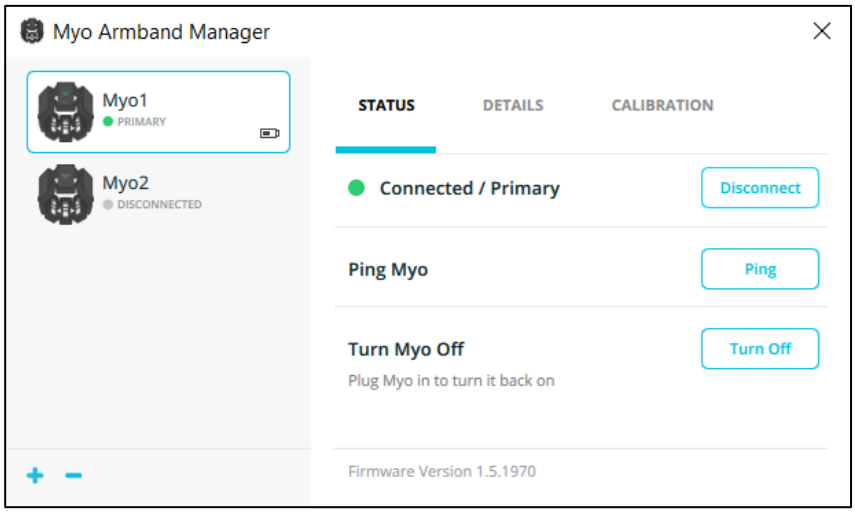


Figura 2.4 Ventana principal del Myo Armband Manager

Para corroborar el buen funcionamiento y el estado del brazalete, existe una página web desarrollada por los creadores del Myo Armband, llamada MyoDiagnostics. En las Figuras 2.5 y 2.6 se observa la ventana general de la página web, aquí se pueden apreciar distintas funcionalidades del brazalete, como son: estado de bloqueo, brazo en el que está colocado, nivel de batería, gráficas de las señales captadas por los 8 sensores, etc.

Figura 2.5 Captura de MyoDiagnostics, parte 1 [8]

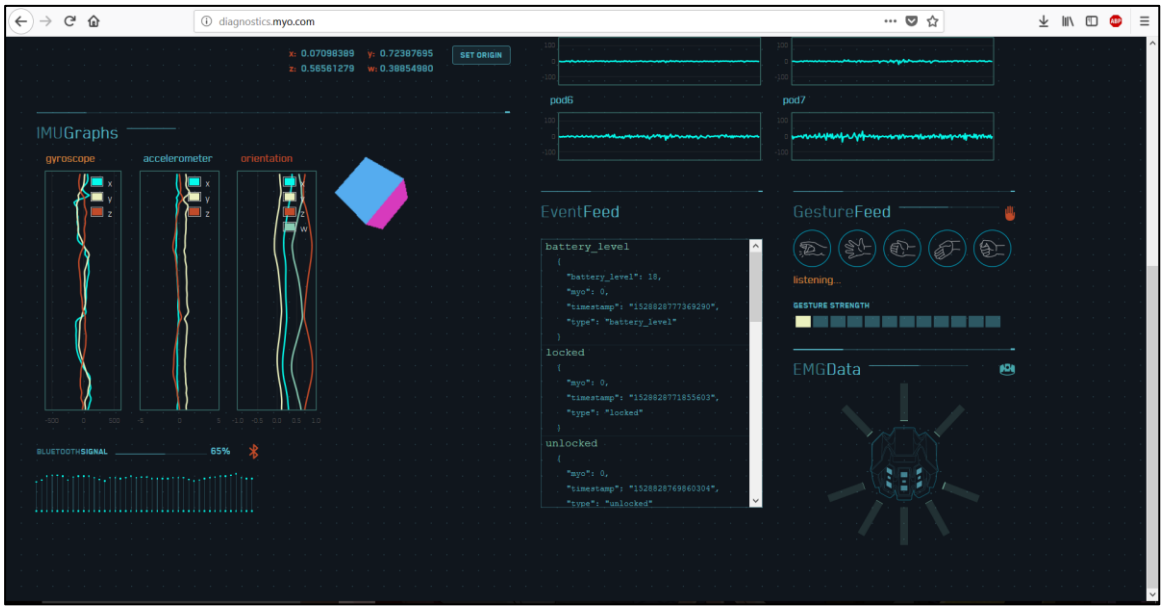


Figura 2.6 Captura de MyoDiagnostics, parte 2 [8]

Al tener el brazalete conectado a la computadora, queda a disposición para realizar cualquier actividad que el usuario desee. Hay funciones ya establecidas al instalar el Myo Connect, por ejemplo, el usuario puede utilizar el brazalete para controlar el mouse o el teclado de su computador. En caso de que se desee conocer a fondo los valores mioeléctricos captados por los sensores o los valores que tienen el acelerómetro, giroscopio o magnetómetro, uno de los softwares de mayor uso para adquirir y procesar estos datos es MATLAB.

Luego de haberse familiarizado con el funcionamiento del Myo Armband bajo el ambiente de Windows, en la siguiente sección se procederá a hacer el cambio al sistema operativo deseado para este proyecto, Raspbian, y se detallará el trabajo realizado para la conexión con el brazalete.

2.4 Comunicación entre el brazalete Myo Armband y la SBC Raspberry Pi 3 B+

El software que permitirá comunicar el brazalete con la SBC se conoce como PyoConnect. Este es el equivalente en Linux a lo que hace el scripting de Myo Connect en Windows. A continuación, se detalla el paso a paso de la instalación de PyoConnect en la Raspberry Pi 3 B+.

Para empezar, se conecta el módulo Bluetooth del brazalete en la SBC, luego se abre el emulador del Terminal para realizar los pasos necesarios para la instalación.

Se ejecutarán los códigos mostrados en la Figura 2.7:

```
~ $ sudo usermod -a -G dialout $USER
~ $ sudo apt-get install python-pip
~ $ sudo pip install pySerial --upgrade
~ $ sudo pip install enum34
~ $ sudo pip install PyUserInput
~ $ sudo apt-get install python-xlib
~ $ sudo apt-get install python-tk
```

Figura 2.7 Códigos de configuración para instalar PyoConnect

El resultado final luego de haberse ejecutado las líneas de código anteriores se observa en la Figura 2.8.

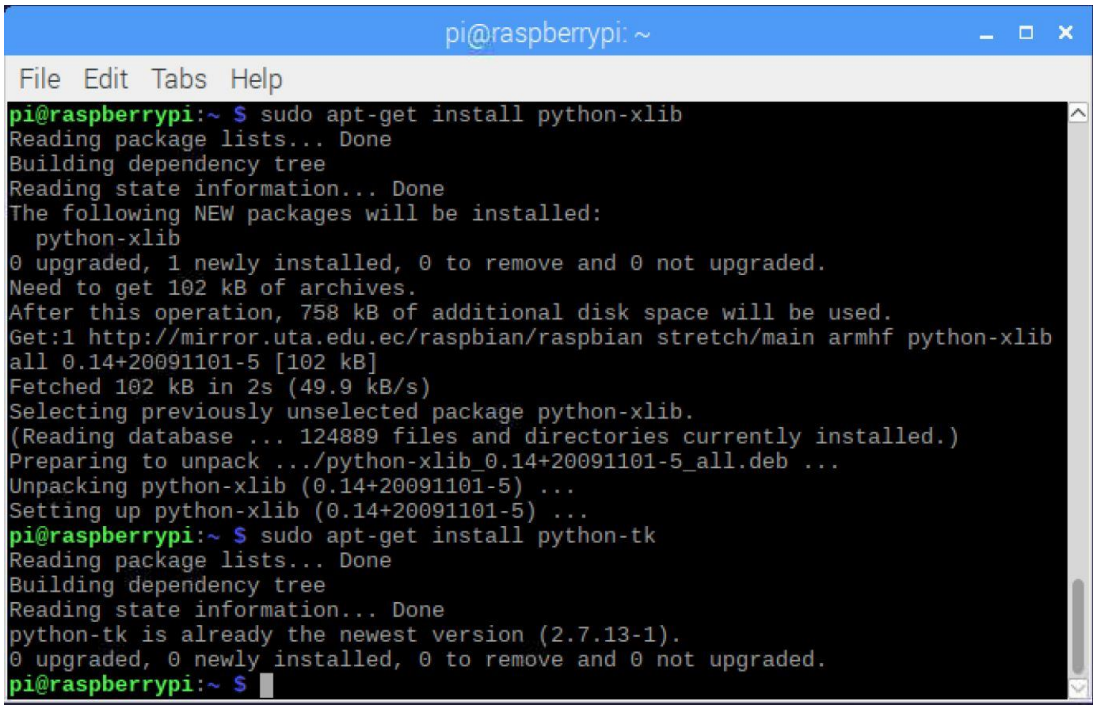


Figura 2.8 Resultado final de la ejecución de código

Para abrir el PyoConnect desde el Terminal primero se debe cambiar el directorio predeterminado, para ello se utiliza el comando de cambio de directorio, cd, como se muestra en la Figura 2.9.

Al haber definido la dirección en donde se encuentra el PyoConnect se procede a abrir el programa para administrar el uso del brazalete, llamado PyoManager. Para

ello se usa el comando $ python PyoManager.pyc. En la Figura 2.10 se muestra la ventana al abrir el programa.

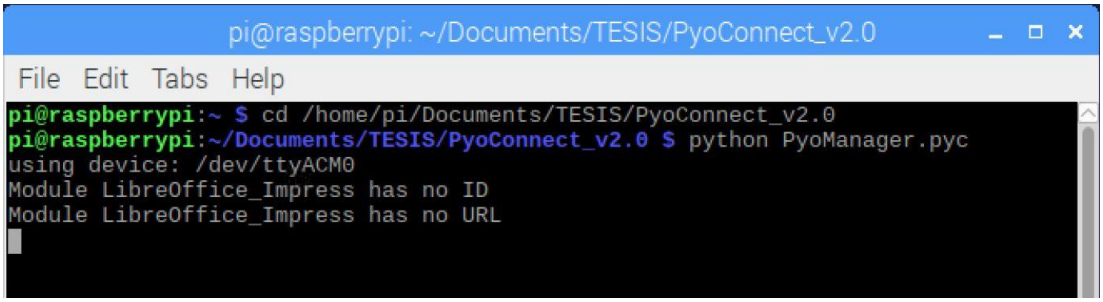


Figura 2.9 Cambio de directorio y ejecución del comando para abrir el mánager de PyoConnect

Figura 2.10 Ventana principal del PyoConnect

Con PyoConnect instalado en la SBC se logra exitosamente la conexión con el brazalete y mediante algoritmos elaborados en Python se podrá tener acceso a la información que este provee.

Por ejemplo, en la Figura 2.11 se muestra la ejecución de un algoritmo programado para mostrar qué pose se encuentra realizando la mano del usuario del brazalete.

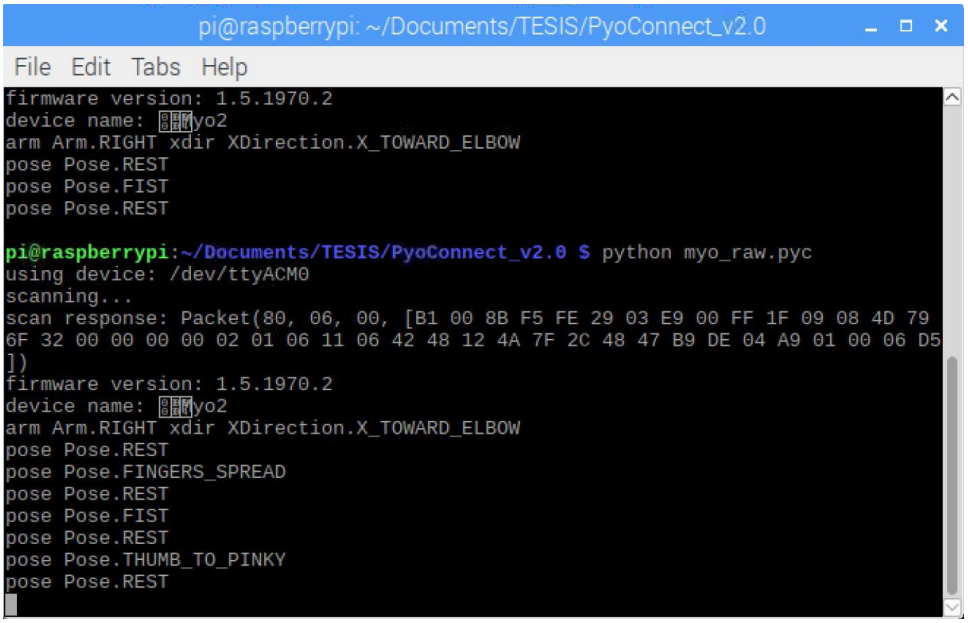


Figura 2.11 Resultado de ejecución de myo_raw, se muestran las poses que realiza la mano en tiempo real

2.5 Aplicación principal del equipo

La aplicación que se creará establecerá comunicación entre el usuario y el Myo Armband. Por tal motivo se hizo uso de la librería pygame dentro del lenguaje de programación python de fuente abierta y gratuita, se usa para hacer aplicaciones multimedia, como juegos; se construye sobre la excelente biblioteca SDL. Al igual que SDL, pygame es muy portátil y se ejecuta en casi todas las plataformas y sistemas operativos. Millones de personas han descargado el pygame en sí, que es una gran cantidad de bits volando a través de las interwebs [9]. Dicha librería viene por defecto en la Distribución Raspbian de Linux.

Se diseñó la ventana de presentación en donde se estableció un fondo de pantalla, así como el título de la misma, como se observa en la Figura 2.12.

En el diseño se colocó el logotipo de la facultad, así como el de la universidad, por lo cual se editaron dichas imágenes en la herramienta GNU “Image Manipulation Program” (GIMP), que es un software libre y gratuito.

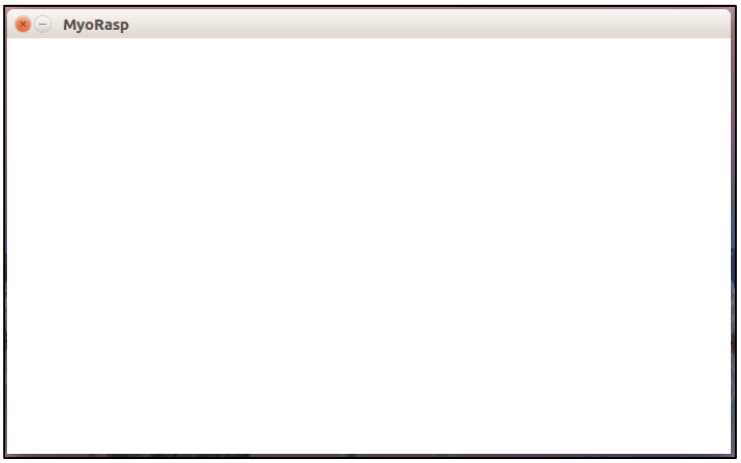


Figura 2.12 Creación de ventana de la aplicación

Este software pertenece al Proyecto GNU y se lo puede adquirir bajo la Licencia pública general de GNU y GNU Lesser General Public License [10]. En la Figura 2.13 se observan las modificaciones realizadas.

Figura 2.13 Modificación de Logos

Estos logos modificados se procedió a colocarlos de la siguiente manera: en la esquina superior izquierda el logo de FIEC y en la esquina inferior derecha el logo de ESPOL, (ver Figura 2.14).

Para que el usuario pueda estar al tanto acerca de la conexión a internet del dispositivo se crearon dos imágenes para este propósito en GIMP; ver Figura 2.14 y Figura 2.15.

Figura 2.14 Colocación de logos en la ventana de la aplicación

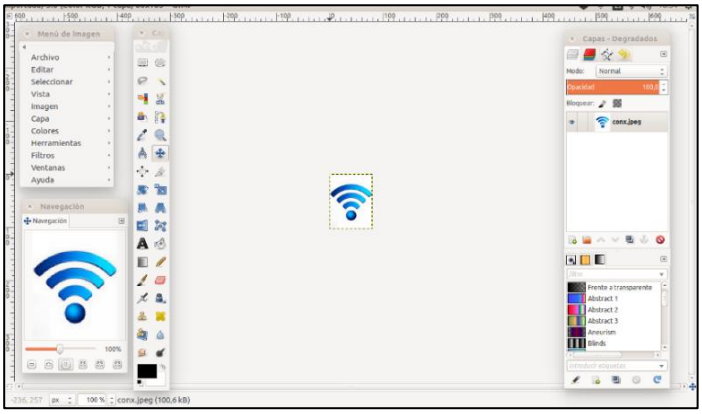


Figura 2.15 Diseño de logo: con conexión a internet

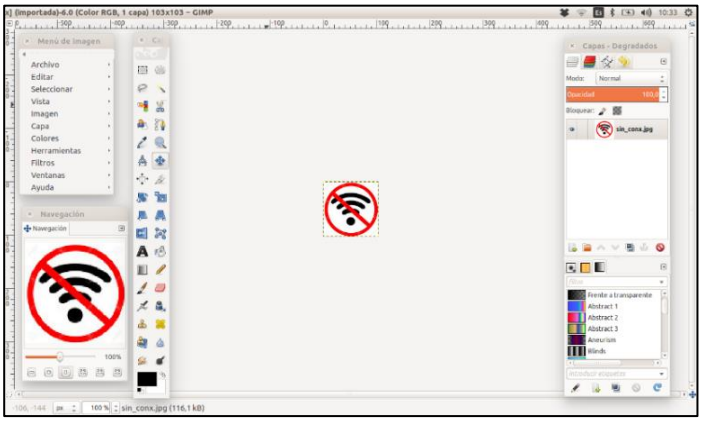


Figura 2.16 Diseño de logo: sin conexión a internet

Para poder clasificar cada uno de los datos que se tomarán es necesario conocer cierta información de los voluntarios, tales como: género, edad, peso y altura; por lo que se procedió a crear en GIMP dicha imagen con cada uno de esos parámetros; ver Figura 2.17.

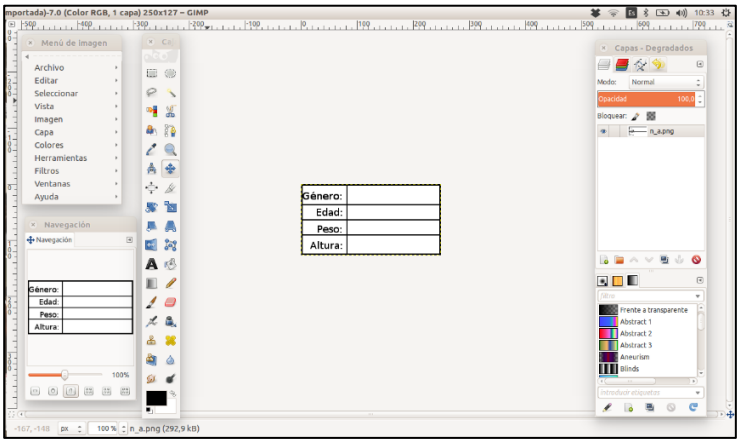


Figura 2.17 Diseño de tabla de datos de los voluntarios

Al colocar este nuevo elemento en el código el aspecto del programa toma la forma observada en la Figura 2.18.

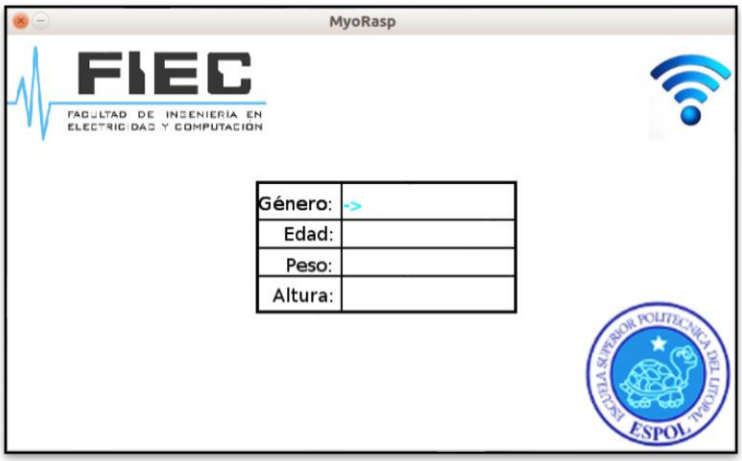


Figura 2.18 Ventana principal de la aplicación para ingreso de datos de los voluntarios

Dado que el proceso de adquisición y visualización de datos los realizan la SBC Raspberry Pi y el Myo Armband; en la aplicación bastó con colocar un interruptor que permita ejecutar dichas acciones; y dichos botones fueron diseñados en GIMP, ver Figura 2.19.

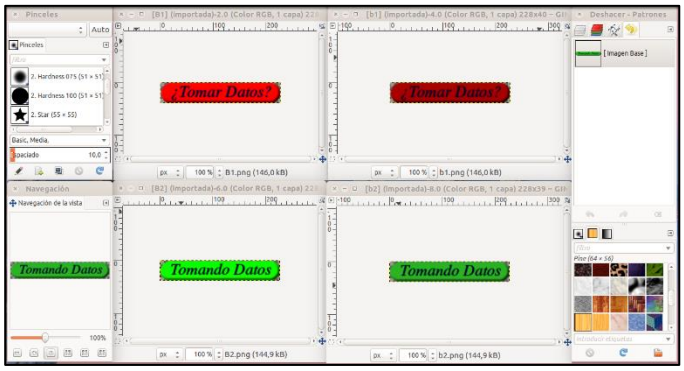


Figura 2.19 Diseño de botones

La creación de estos botones crea un efecto visual a los usuarios al colocar el puntero del mouse sobre ellos. La implementación de dicho proceso se observa en las Figuras 2.20 y 2.21.

Figura 2.20 Puntero fuera del botón

Figura 2.21 Puntero sobre el botón

La adquisición de datos desde el brazalete hacia la SBC se ejecuta al accionar el botón creado anteriormente.

Posibles errores por parte del usuario al querer usar la aplicación pueden ser: dar clic en otra parte de la ventana de la aplicación donde no esté el botón o accidentalmente dar clic derecho para empezar a tomar datos. Para solucionar estos inconvenientes se crearon cuadros de textos que indiquen la manera correcta de ejecutar el programa; la creación y ejecución se muestra en las Figuras 2.22 - 2.25.

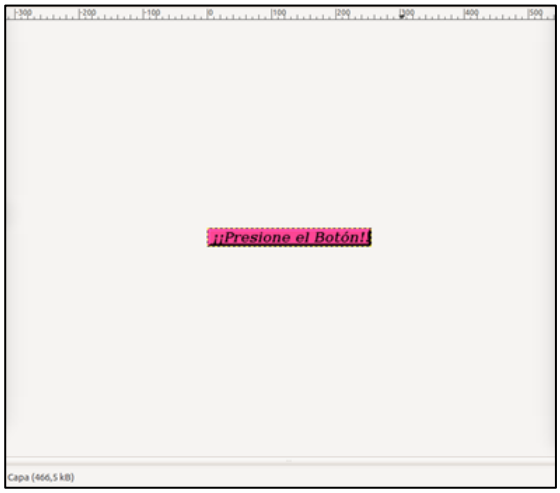


Figura 2.22 Creación del texto a mostrarse cuando se presiona fuera del botón

Figura 2.23 Ejecución de texto cuando se presiona fuera del botón

Los datos que se obtengan de los voluntarios serán adquiridos así haya o no internet; la diferencia se da en que estos se enviarán o almacenarán según sea el caso; ver Figura 2.26 y Figura 2.27.

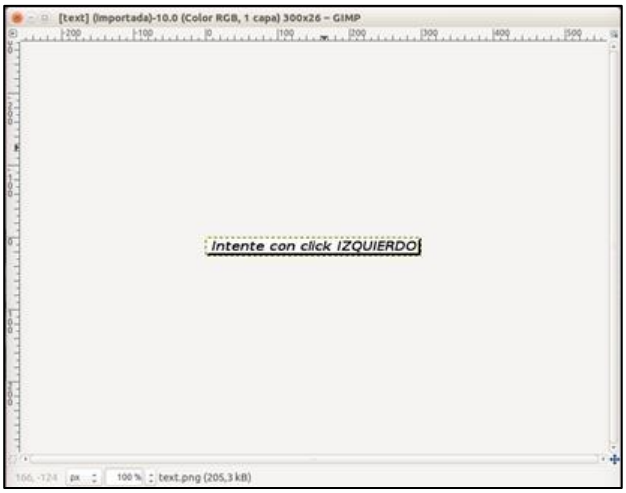


Figura 2.24 Creación del texto a mostrarse cuando se presiona dentro del botón, pero con la tecla incorrecta

Figura 2.25 Ejecución del texto mostrado cuando se presiona dentro del botón, pero con la tecla incorrecta

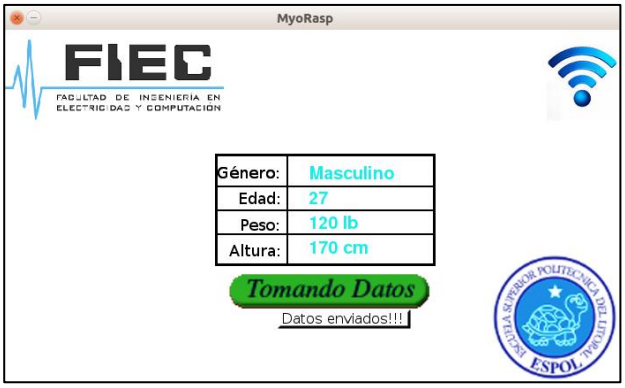


Figura 2.26 Adquisición y envío de datos

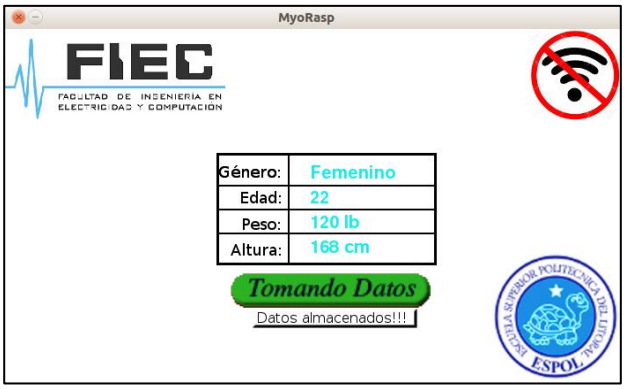


Figura 2.27 Adquisición y almacenamiento de datos

Se puede dar el caso de tener dos o más archivos almacenados debido a que no exista acceso a internet en el área geográfica donde se estén realizando la adquisición de las señales mioeléctricas; en dicho caso no se podrá salir de la aplicación a menos de que se pueda establecer una conexión a internet con el objetivo de enviar dichos datos de manera automática; otra manera de cerrar la aplicación es forzar su cierre mediante el Terminal o apagando la SBC; en cualquiera de estos dos últimos casos se tendrá que enviar los datos manualmente ya que al ejecutar nuevamente la aplicación las variables iniciarán en cero; ver Figura 2.28 y Figura 2.29.

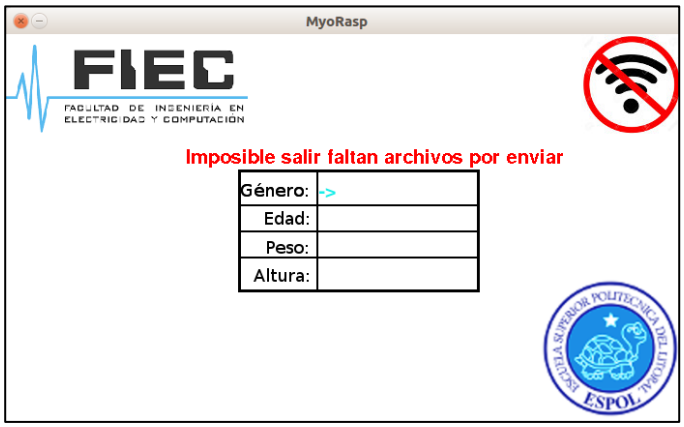


Figura 2.28 Con datos almacenador no se puede salir del programa

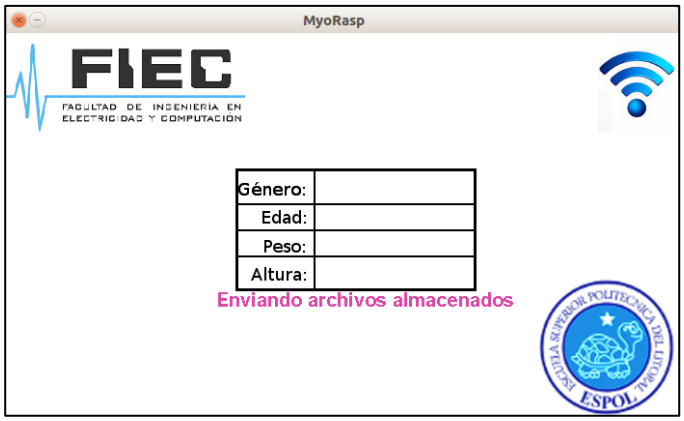


Figura 2.29 Cuando se establezca una conexión a internet se enviarán los datos almacenados

2.6 Adquisición de los datos mioeléctricos

Habiendo establecido anteriormente la conexión entre la SBC y el brazalete se procede a crear el algoritmo que servirá para la adquisición de los datos mioeléctricos generados en el brazo derecho. Para ello, usando Python, se importará el módulo myo_raw [11], código en el cual se tiene la base para obtener información del brazalete. Este código es un equivalente al SDK para Windows del Myo Armband. Además de myo_raw, se importará la librería time para establecer durante cuánto tiempo se desean adquirir los datos mioeléctricos.

Una vez importadas las librerías necesarias, se procede a la creación del algoritmo para adquirir, mostrar y guardar los datos mioeléctricos del brazo derecho del voluntario. El diagrama de flujo que se observa en la Figura 2.30 explica el algoritmo diseñado.

2.7 Creación de algoritmo para el almacenamiento de una copia y/o transmisión de los datos adquiridos.

Los datos serán guardados en un archivo con formato .csv para luego ser almacenados o transmitidos, dependiendo de si se dispone o no de conexión a internet. El envío de la información adquirida se realizará través de correo electrónico, por lo cual se debe contar con una librería que realice dicho proceso; smtplib cumple esta tarea. Esta librería cuenta con el protocolo Simple Mail Transfer

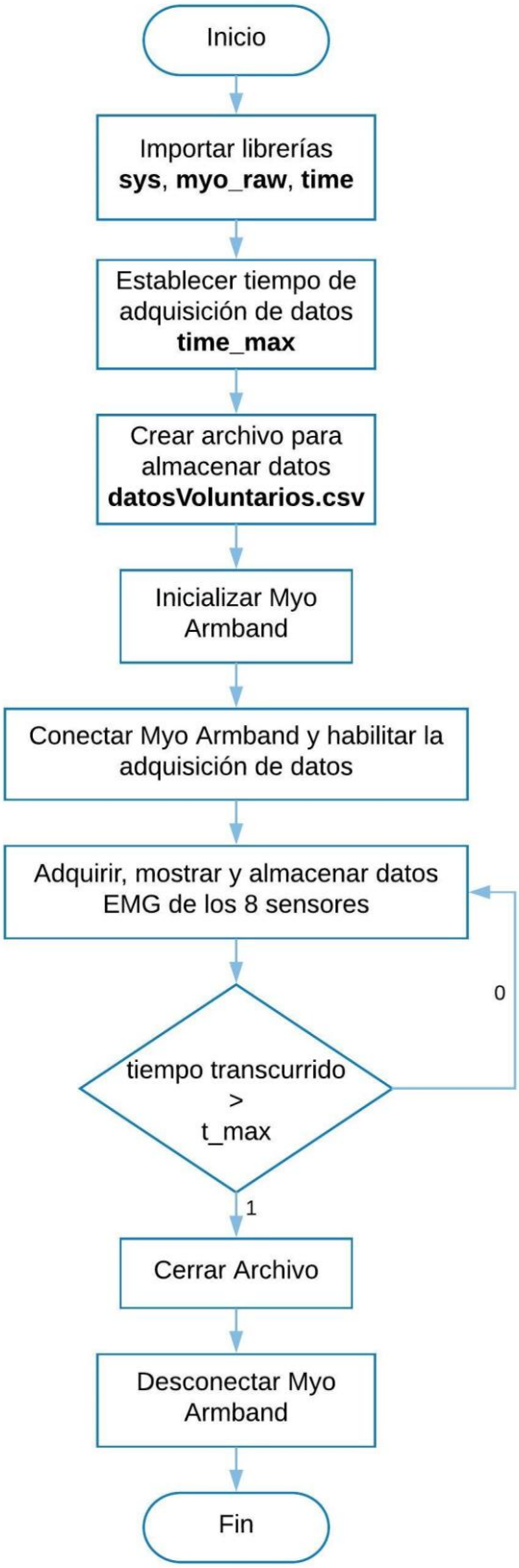


Figura 2.30 Diagrama de flujo de la adquisición de datos mioeléctricos

Protocol (SMTP), el cual es un conjunto de normas para la transmisión de mensajes desde el origen al destino, y proporciona una forma muy conveniente para enviar correo electrónico. [12]

El almacenamiento de datos o el envío de estos dependerá exclusivamente de la conexión a internet del equipo. Dado el caso de que no exista la misma, se realizará una copia del archivo original, y en el momento que se pueda establecer conexión a internet se enviarán y borrarán todos los archivos almacenados. Para poder conocer dicho evento se realizará un PING; este comando proporciona información del estado de conexión de un Host Local con uno o varios equipos contemplados en una red de tipo TCP/IP. [13]

Todo el proceso mencionado se detalla en el diagrama de flujo mostrado en la Figura 2.31:

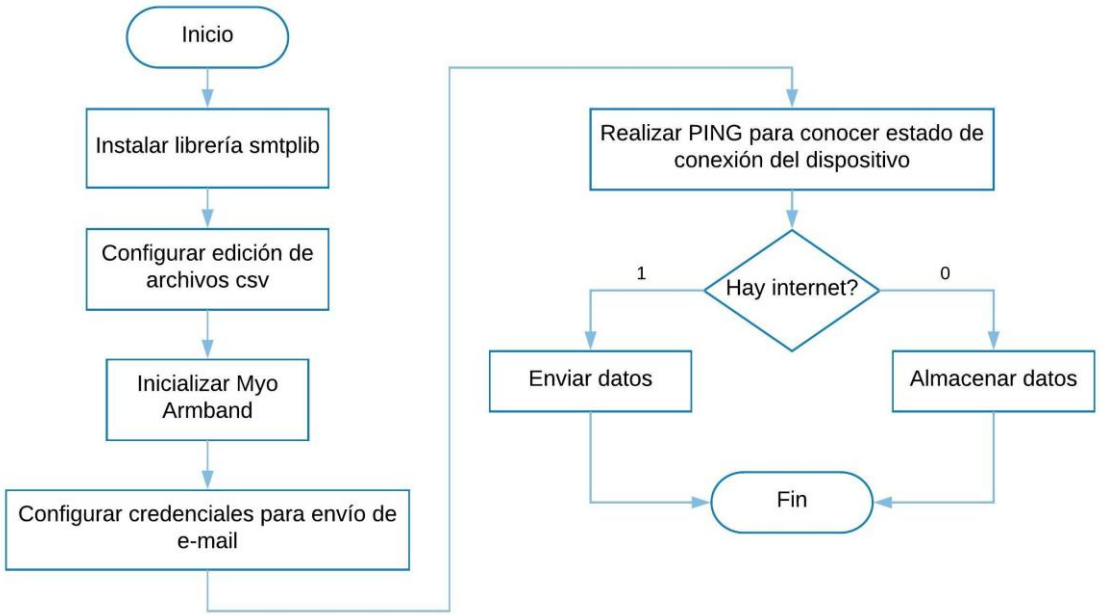


Figura 2.31 Pseudocódigo de almacenamiento y transmisión de datos

2.8 Gráfica de las señales mioeléctricas

Luego de adquirir y guardar los datos mioeléctricos, se desea mostrar una gráfica de EMG vs t para observar el comportamiento que tuvieron las señales durante la adquisición, y comprobar que los datos se adquirieron con éxito y de forma correcta. Para ello es necesario el manejo de las librerías numpy y matplotlib, las cuales

permiten trabajar con matrices y gráficas. Además de estas, se necesitará la librería csv que sirve para manipular archivos que contienen datos separados por comas. En la Figura 2.32 se observa en forma de pseudocódigo el procedimiento para graficar los datos mioeléctricos.

```
- Importar numpy, matplotlib, csv
- Crear listas; t, emg
- Crear figura; fig.
- Crear una subgráfica para mostrar los ejes; ax
- Definir valores mínimos y máximos de los ejes x y y
- Activar cuadrícula en la gráfica
- Abrir archivo datos.csv en modo lectura
    Leer el archivo csv
            Asignar valores del archivo csv a las listas t y emg
- Graficar (t,emg) en fig
- Activar títulos de ejes y de la gráfica
- Guardar figura
- Mostrar figura
```

Figura 2.32 Pseudocódigo para graficar las señales mioeléctricas

En la Figura 2.33 se observa cómo se muestra la gráfica de las señales mioeléctricas en la pantalla de la SBC.

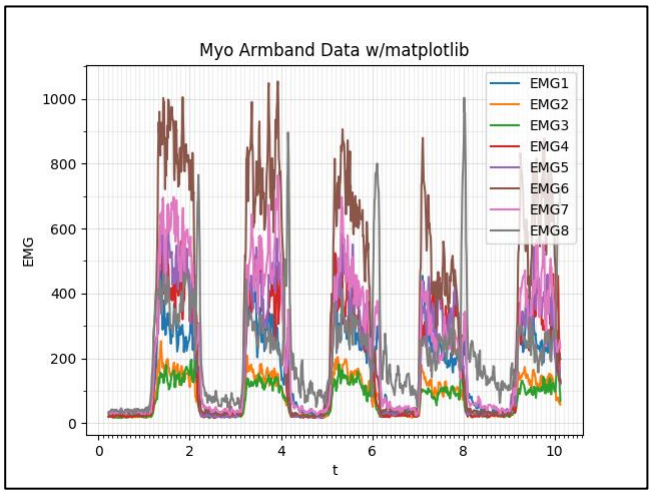


Figura 2.33 Señales mioeléctricas de los 8 sensores del brazalete

2.9 Creación de Servidor-Nube

Para montar un servidor lo primero que se debe hacer es entrar a la configuración del router, y establecer una IP estática a la PC donde se almacena dicho servidor, esto se hace debido a que los routers se reinician automáticamente cada cierto tiempo, pre-establecido por el fabricante, lo cual ocasiona que las IP en todos los dispositivos cambien, esto provoca un problema al momento de crear un servidor o ingresar a los dispositivos remotamente mediante VNC o SSH. En el router también se abren los puertos necesarios para la comunicación del servidor con el mundo exterior, si no se realiza este paso, solo los dispositivos conectados a la misma red del servidor podrán acceder a la misma; dichos puertos son: del 20 – 50; del 20-100 y el 8080. Se debe activar el DMZ en el router e introducir la IP del PC.

Para poder conectarse al servidor desde el exterior se debe tener una IP pública, pero las IP's que ofrecen los routers son privadas. Existen dos maneras de obtener dicha IP, la primera es pagando por el servicio y la segunda es obtenerla gratis, la segunda opción es más conveniente. Para ello se accede a la página web www.noip.com**;** donde se crea un Hostname y automáticamente se asignará la IP pública. Lo que se debe tener en consideración al usar esta página es que cada 30 días se tendrá que actualizar los datos ya que se caduca la IP. Luego se instala el software **DDCLIENT,** el cual sirve para poder vincular IP privada con la IP pública creada, dicho proceso se muestra a continuación en las Figuras 2.34 - 2.40:

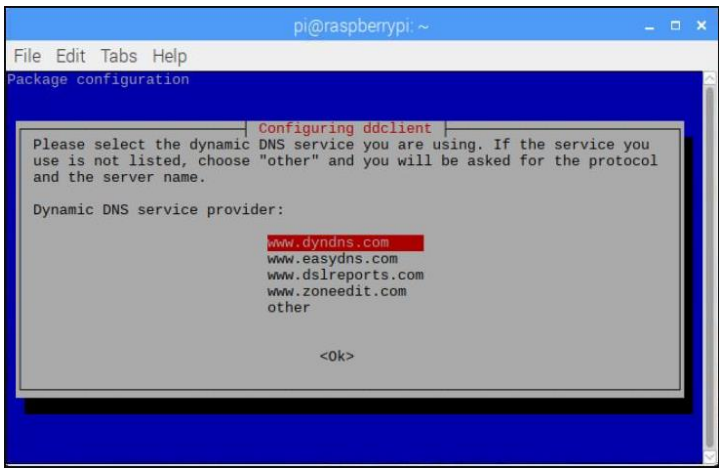


Figura 2.34 Configuración del DNS

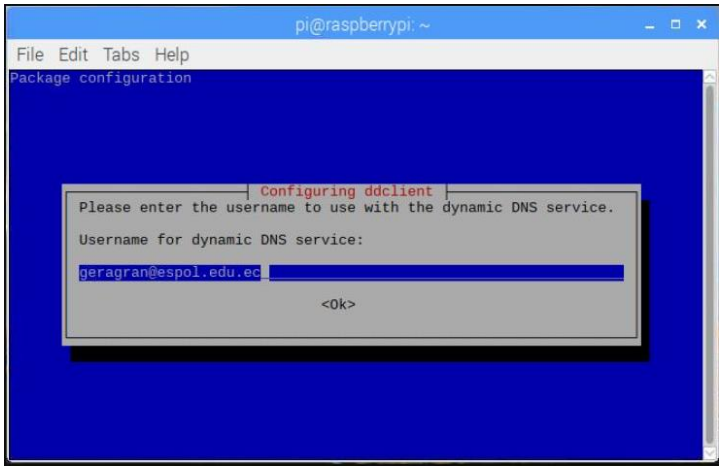


Figura 2.35 Configuración del usuario creado en noip.com

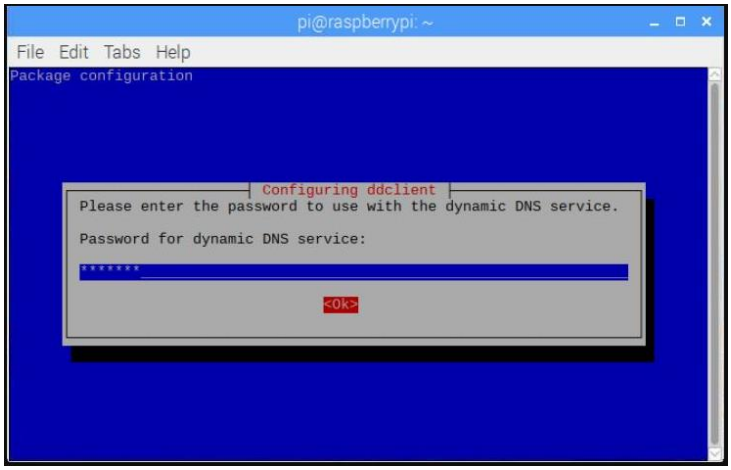


Figura 2.36 Configuración de clave usada en noip.com

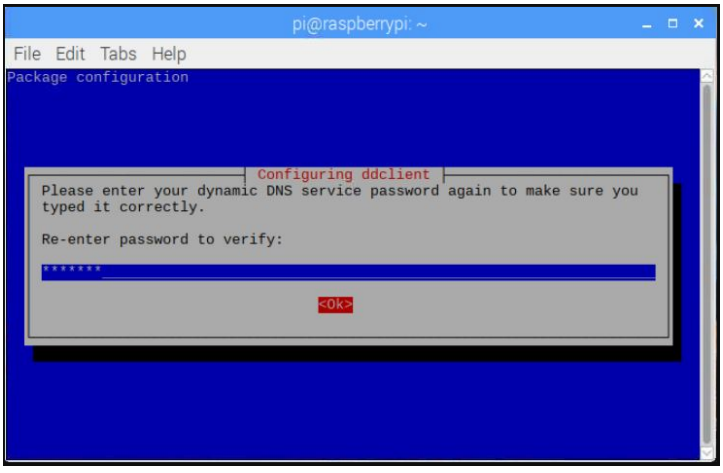


Figura 2.37 Confirmación de clave usada en noip.com

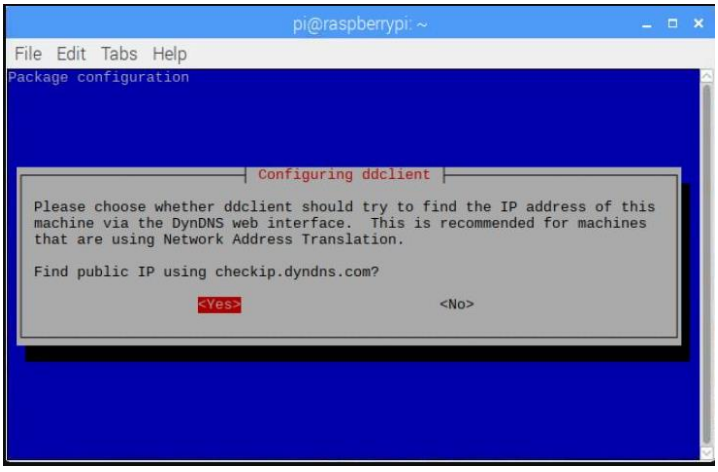


Figura 2.38 Selección del DyDNS

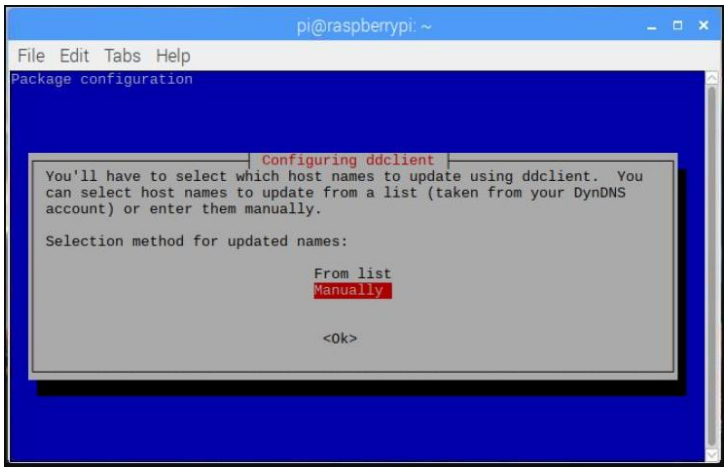


Figura 2.39 Establecimiento manual del HostName

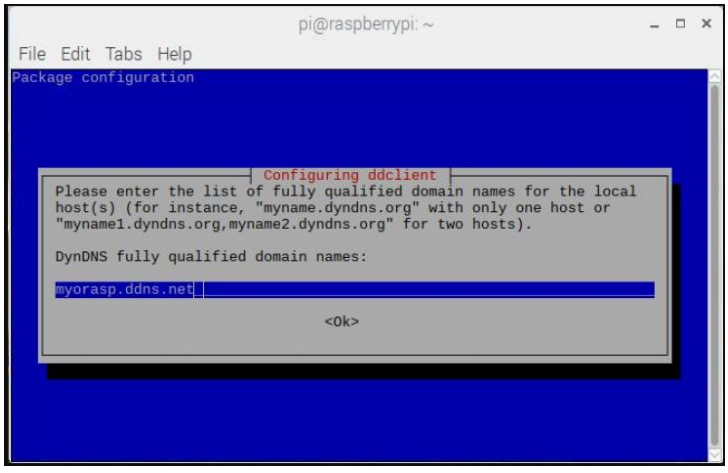


Figura 2.40 HostName creado

Ahora se puede instalar Apache2 y algunas librerías que se usarán para el buen funcionamiento del servidor; ver Figuras 2.41 y 2.42.

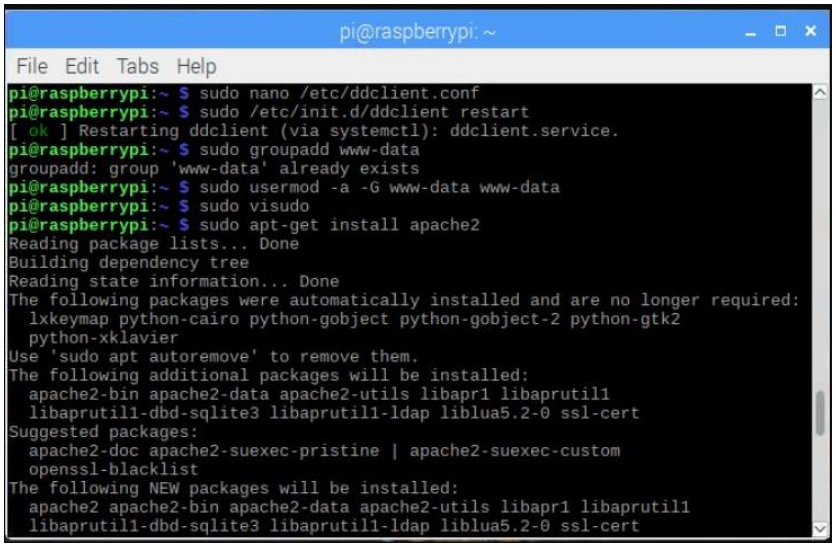


Figura 2.41 Instalación de apache2

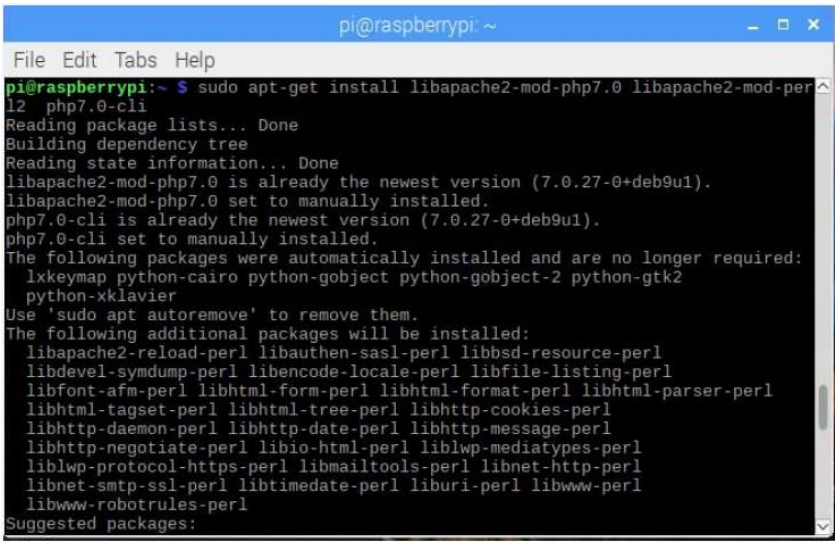


Figura 2.42 Instalación de librerías para MYSQL

Luego de que se instalaron todos estos paquetes se dirige al navegador y se introduce la IP de la PC. Se obtuvo el resultado que se observa en la Figura 2.42.

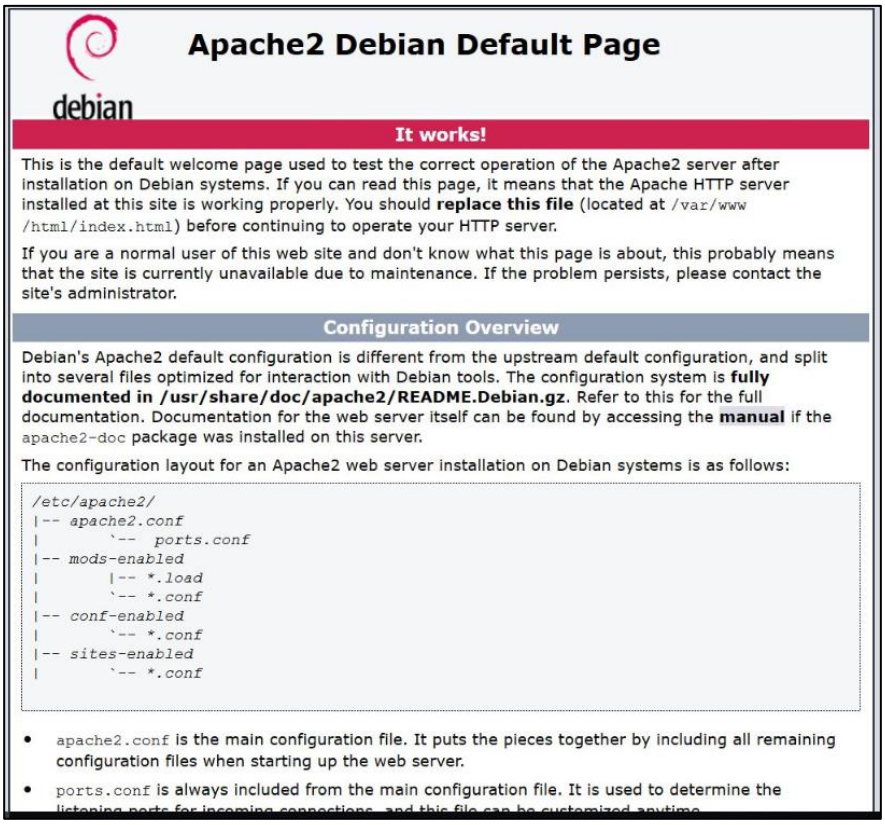


Figura 2.43 Visualización de Apache2 en nuestro navegador

Luego se instala MySQL y todas sus dependencias; y se procede a modificar la contraseña del usuario **root,** y de igual forma se crea una base de datos para la nube, tal como se muestra en las Figuras 2.44 - 2.47.

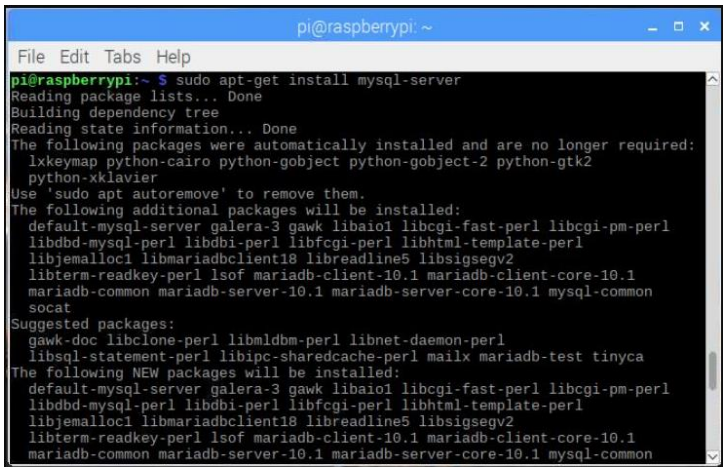


Figura 2.44 Instalación de MYSQL

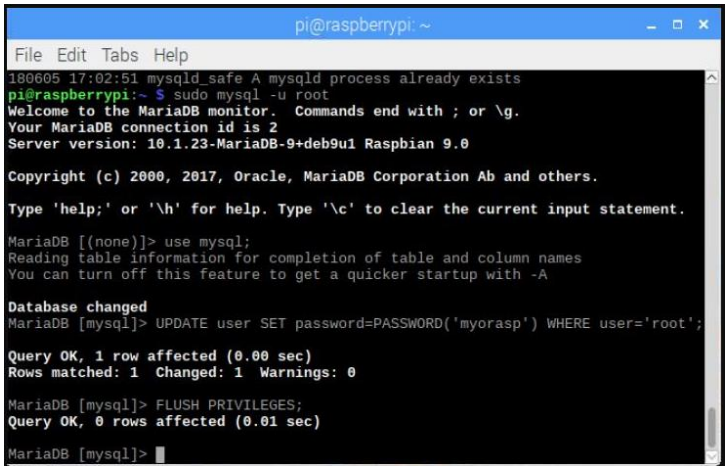


Figura 2.45 Creación de usuario MYORASP

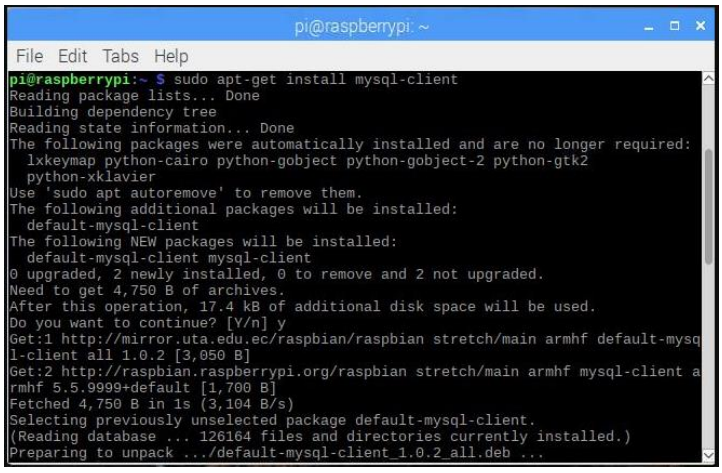


Figura 2.46 Instalación de mysql-client

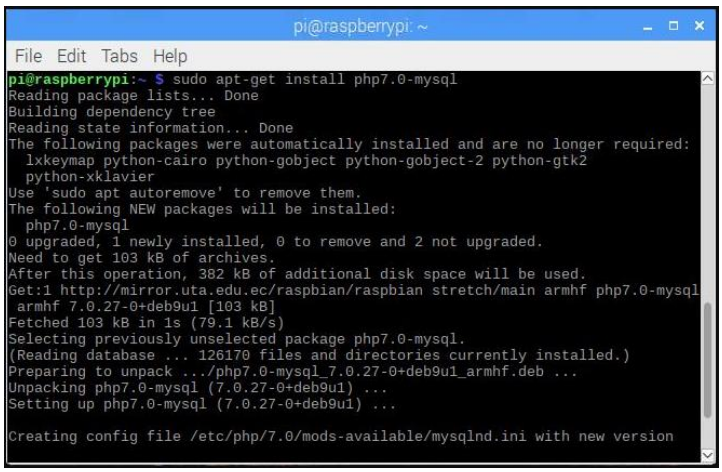


Figura 2.47 Instalación de php7.0-mysql

La instalación de **Phpmyadmin** es mucho más sencilla; y en este proceso también se crea la base de datos y la contraseña respectiva. Para comprobar si toda la configuración ha sido exitosa se abre el navegador y se introduce la URL **IPRaspberry/phpmyadmin.** En la página cargada se verifican los usuarios creados en **MySQL;** es decir el usuario **root,** y el usuario **phpmyadmin;** como se observa en las Figuras 2.48 - 2.52**.**

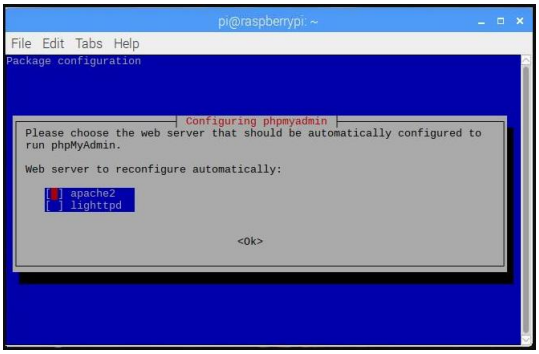


Figura 2.48 Establecimiento del servidor usado

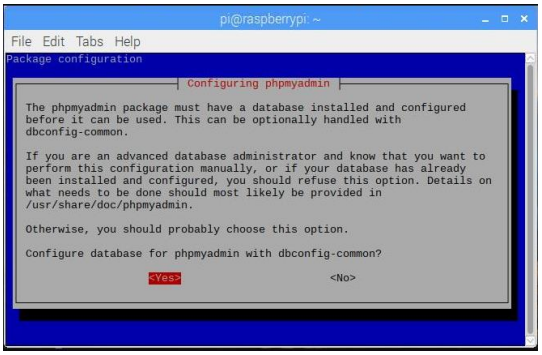


Figura 2.49 Configuración de Base de Dato

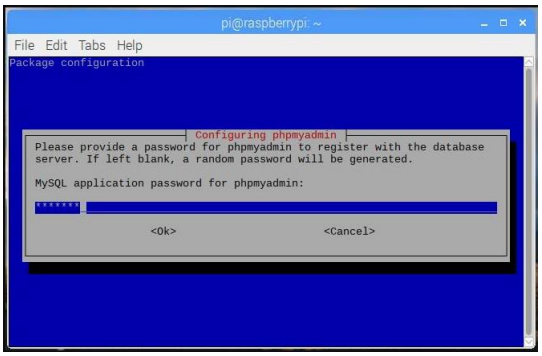


Figura 2.50 Establecimiento de la contraseña de phpmyadmin

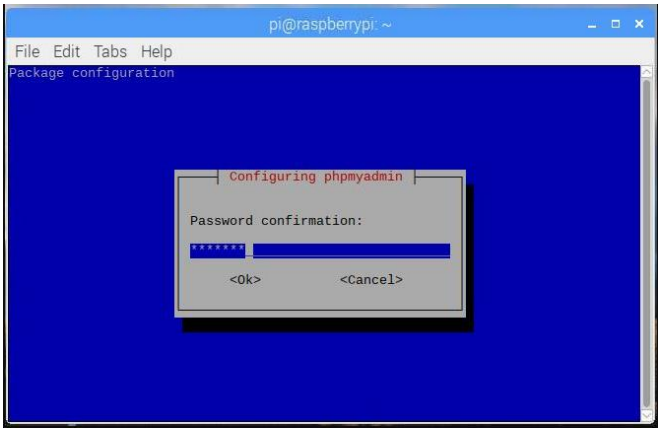


Figura 2.51 Confirmación de la contraseña de Phpmyadmin

Figura 2.52 Instalación finalizada de Phpmyadmin

Con todos estos programas instalados se tiene listo el servidor para proceder con la instalación de la nube; la que se usará es **Pydio,** este es un software libre y tiene características similares a plataformas como Dropbox; con la ventaja de que el acceso a los datos almacenados solo será para las personas a las que se les proporcionen las claves respectivas. [14]

El proceso de instalación y configuración de Pydio se muestra en las Figuras 2.53 - 2.56.

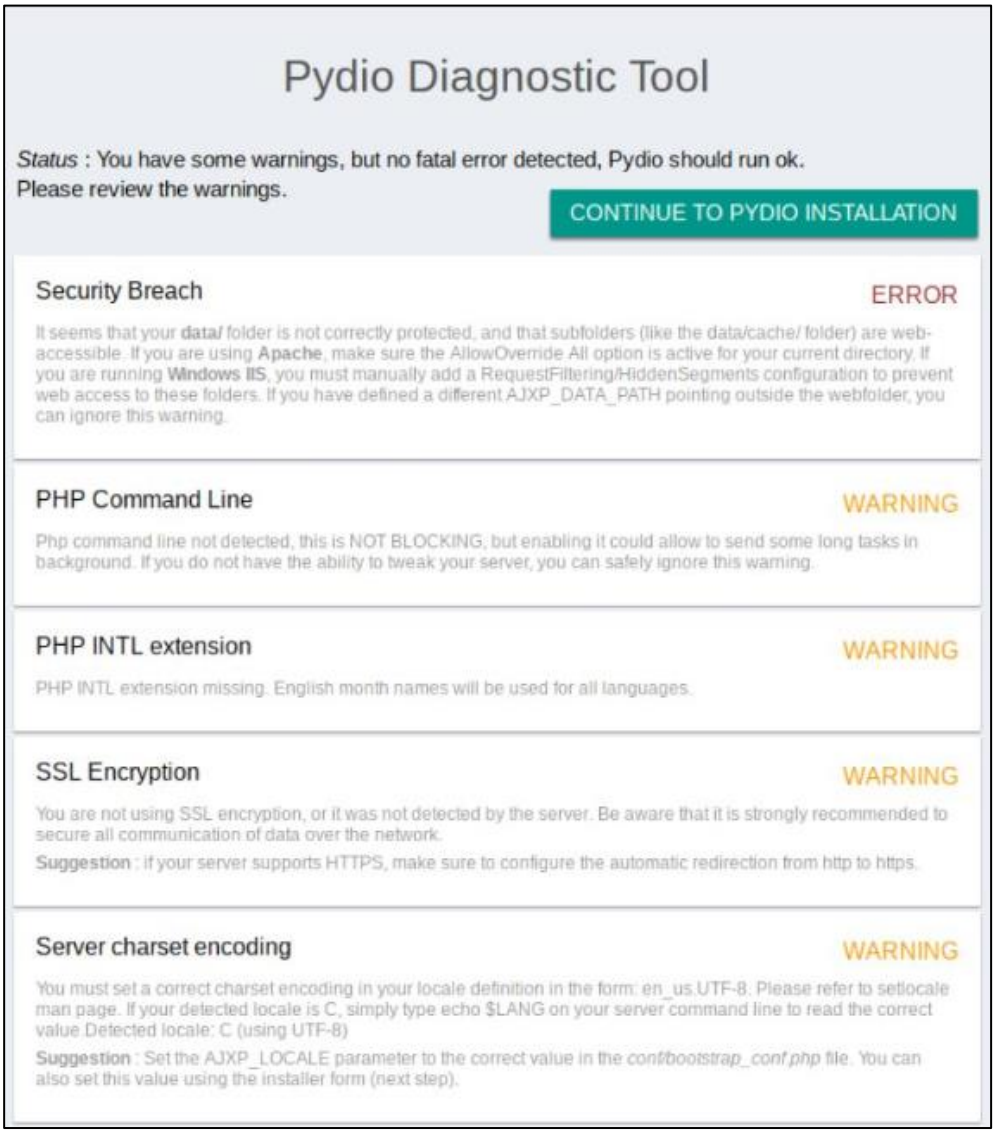


Figura 2.53 Inicio de configuración Pydio

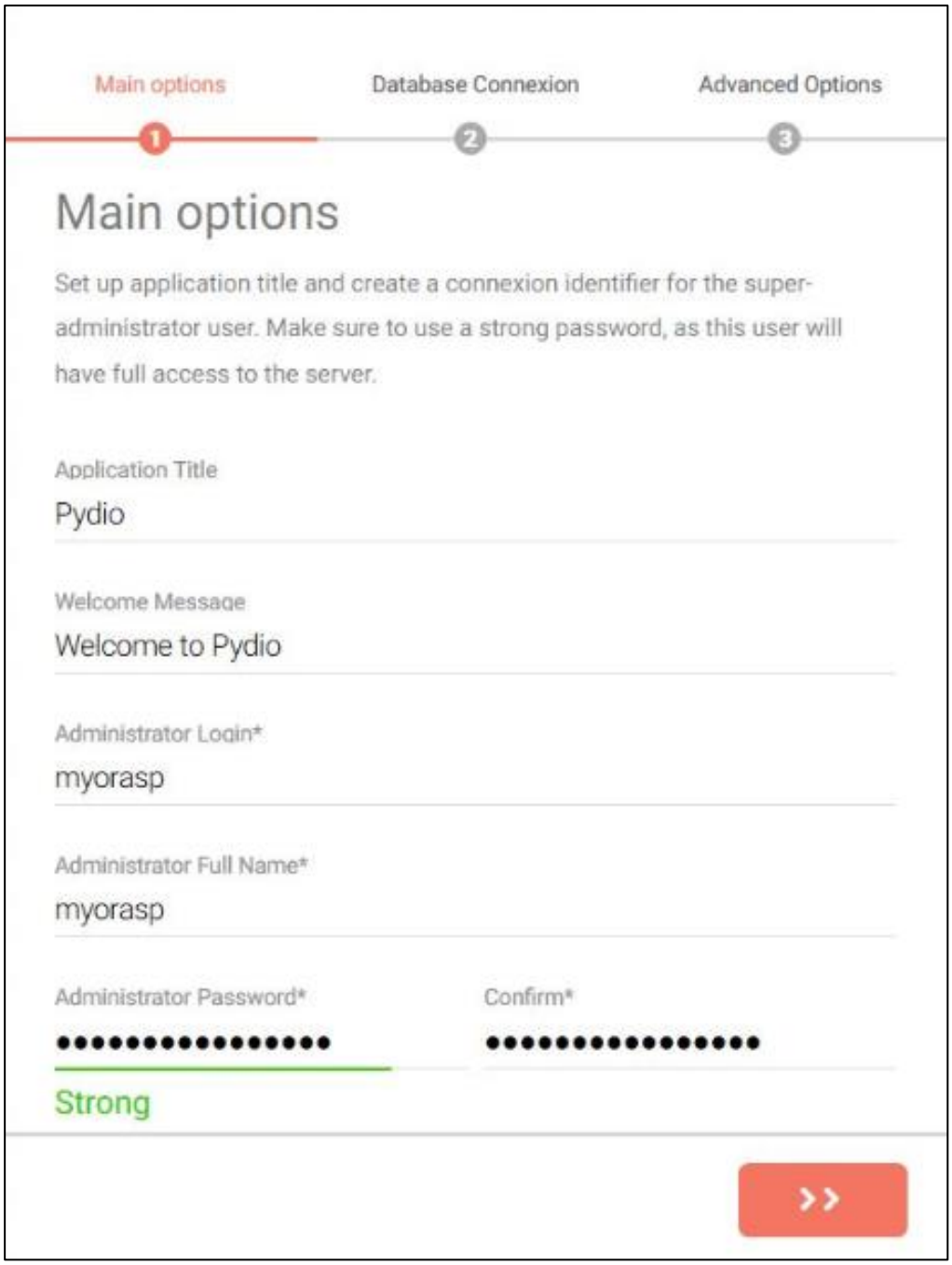


Figura 2.54 Establecimiento de usuario y contraseña de Pydio

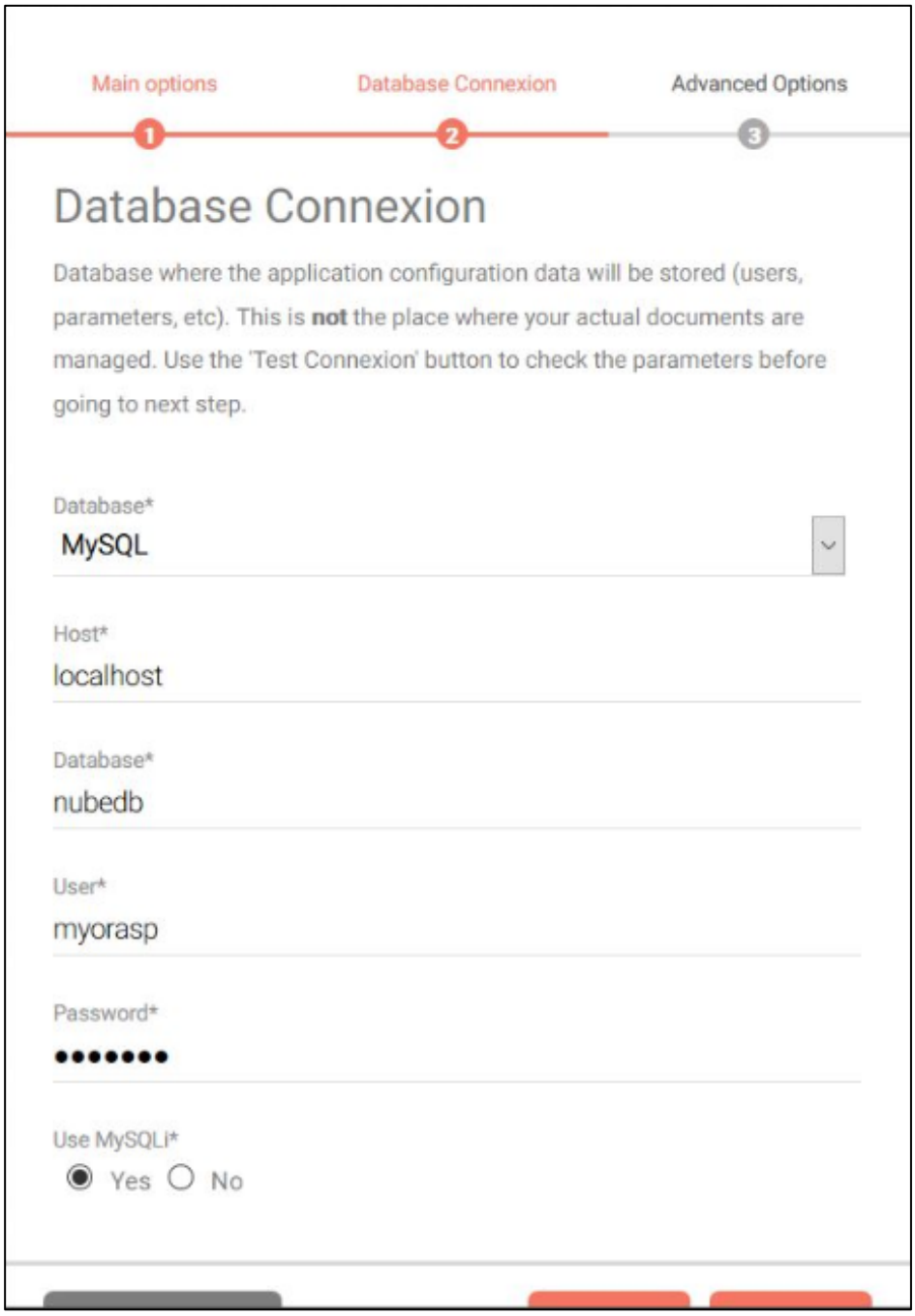


Figura 2.55 Configuración de la base de datos en MySQL

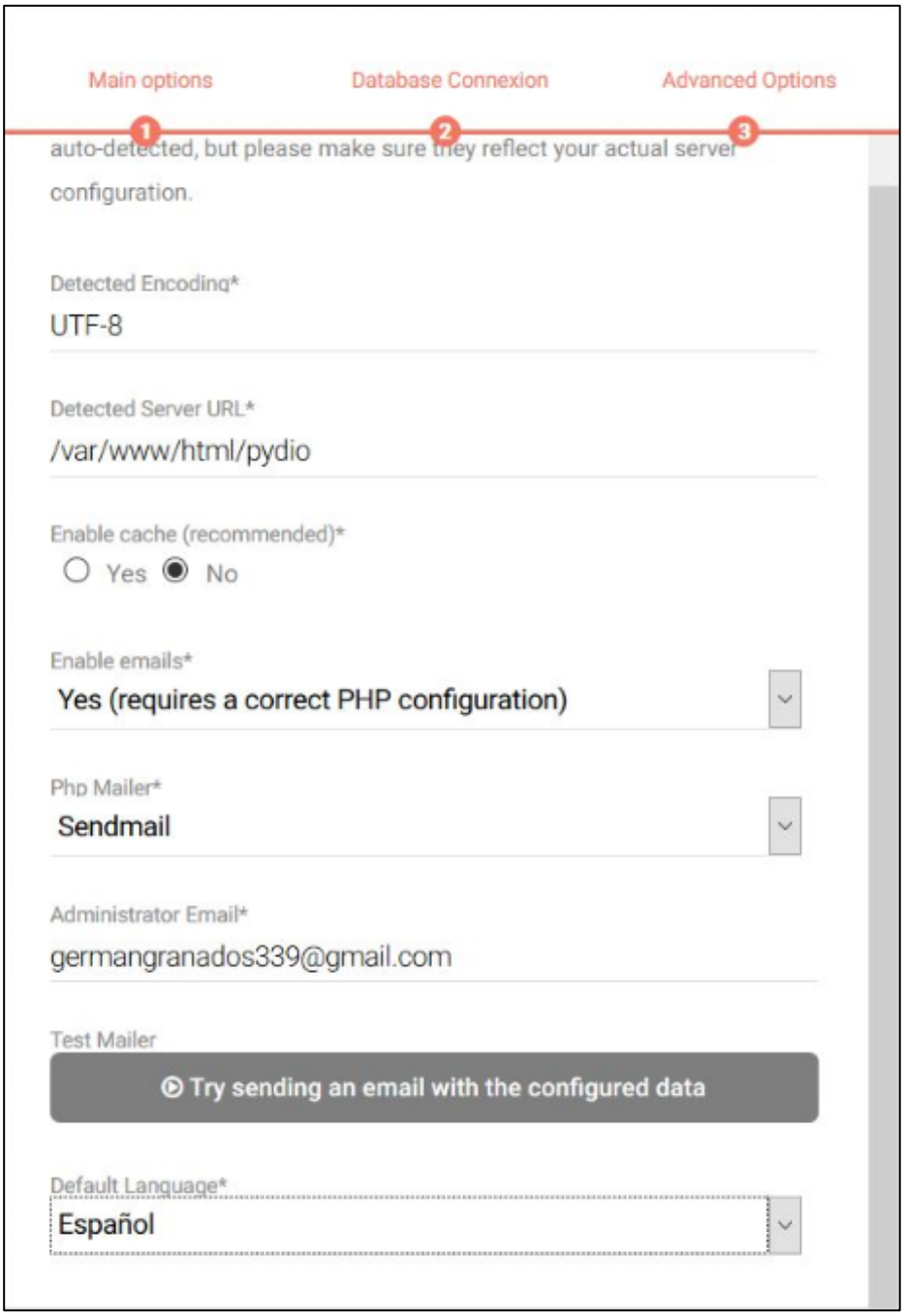


Figura 2.56 Configuración de Idioma y correo para el envío de a-mail

CAPÍTULO 3

3. RESULTADOS Y ANÁLISIS

Para comprobar el correcto funcionamiento del equipo portátil se realizó la adquisición de los datos mioeléctricos de un pequeño grupo de voluntarios realizando diferentes movimientos de la mano, los cuales se detallan en la Tabla 3.1.

Tabla 3.1 Movimientos realizados durante la adquisición de datos

Movimiento1	Extensión de los dedos - Descanso	
Movimiento 2	Puño - Descanso	
Movimiento 3	Hombre araña - Descanso	
Movimiento 4	Pulgar arriba - Descanso	
Movimiento 5	Pulgar a anular - Descanso	

En la figura 3.1 se muestra una fotografía de los elementos del equipo portátil, de la aplicación principal abierta en el equipo y del brazo de uno de los voluntarios con el brazalete colocado. Se recalca el hecho de que el sensor 4 debe estar ubicado siempre en la posición observada antes de realizar la adquisición de datos.

Figura 3.1 Equipo portátil para adquisición de datos mioeléctricos

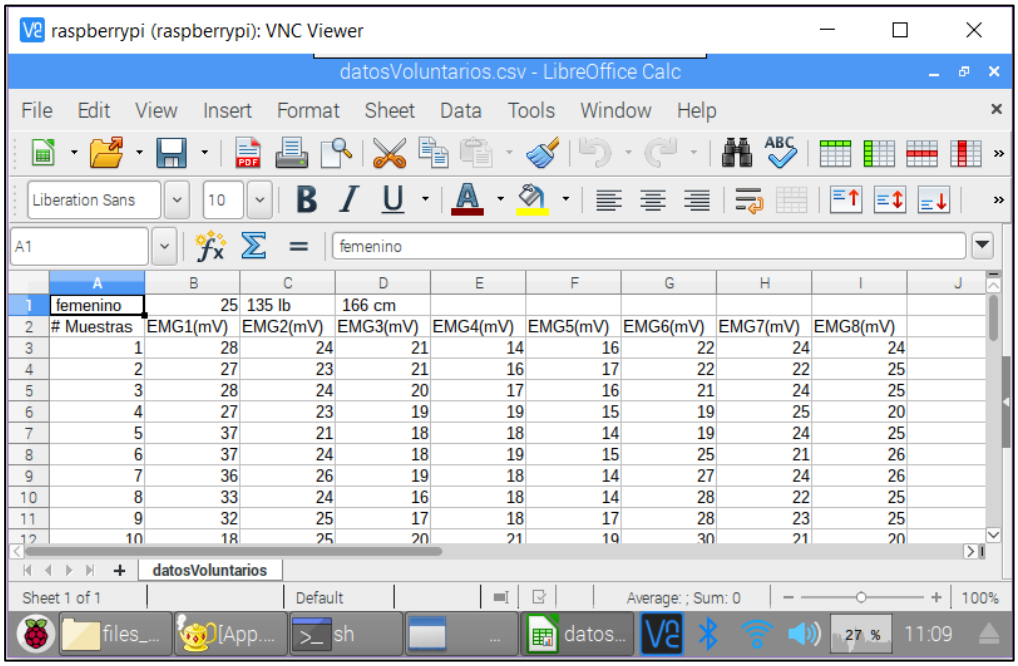
3.1 Resultados obtenidos

La Tabla 3.2 proporciona información personal sobre los tres voluntarios que realizaron los movimientos. Esta información se ingresó también en la aplicación y se almacenó junto a los datos mioeléctricos adquiridos durante las diferentes pruebas.

Tabla 3.2 Información de los voluntarios

Voluntario	Género	Edad (años)	Peso (lb)	Altura (cm)
1	Femenino	25	154.1	165
2	Masculino	22	158.4	173
3	Femenino	52	193.6	162

Al ejecutar el programa en el equipo portátil, los datos mioeléctricos se guardaron correctamente en un archivo con formato .csv, como se muestra en la Figura 3.2. En la primera fila se almacenaron los datos personales del voluntario, siendo estos género, edad, peso y altura. A partir de la segunda fila en adelante se encuentran los datos mioeléctricos detectados por los 8 sensores. En la primera columna se almacenó el número de muestras obtenidas y en las 8 columnas siguientes el voltaje de las señales mioeléctricas, en mV, detectado por los 8 sensores del brazalete.



	A	B	C	D	E	F	G	H	I
1	femenino	25	135 lb	166 cm					
2	# Muestras	EMG1(mV)	EMG2(mV)	EMG3(mV)	EMG4(mV)	EMG5(mV)	EMG6(mV)	EMG7(mV)	EMG8(mV)
3	1	28	24	21	14	16	22	24	24
4	2	27	23	21	16	17	22	22	25
5	3	28	24	20	17	16	21	24	25
6	4	27	23	19	19	15	19	25	20
7	5	37	21	18	18	14	19	24	25
8	6	37	24	18	19	15	25	21	26
9	7	36	26	19	18	14	27	24	26
10	8	33	24	16	18	14	28	22	25
11	9	32	25	17	18	17	28	23	25
12	10	18	25	20	21	19	30	21	20

Figura 3.2 Captura de pantalla de los datos mioeléctricos almacenados

Después de realizar la adquisición y almacenamiento de los datos, se generaron las gráficas respectivas para observar con más claridad el comportamiento de las señales durante los experimentos.

Las gráficas, EMG[mV] vs t[s], con los datos adquiridos de tres diferentes voluntarios se muestran en las Figuras 3.3 – 3.17. Estas gráficas proporcionan información de los movimientos detectados por los 8 sensores, diferenciándose cada uno por los distintos colores asignados. En cada una se observó claramente el momento en que se realizaron los diferentes movimientos, variando solamente las magnitudes de las señales EMG, las cuales dependían de la intensidad de fuerza con la que el voluntario realizaba el movimiento respectivo.

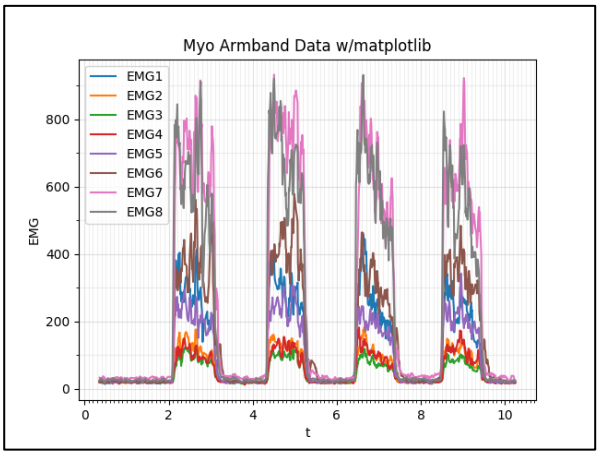


Figura 3.3 Resultado del movimiento 1 del voluntario 1

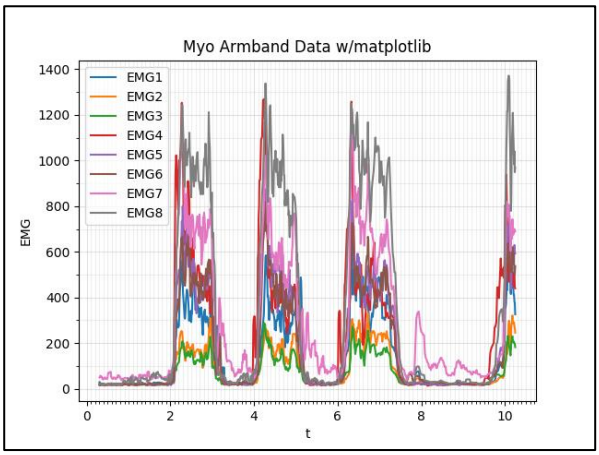


Figura 3.4 Resultado del movimiento 1 del voluntario 2

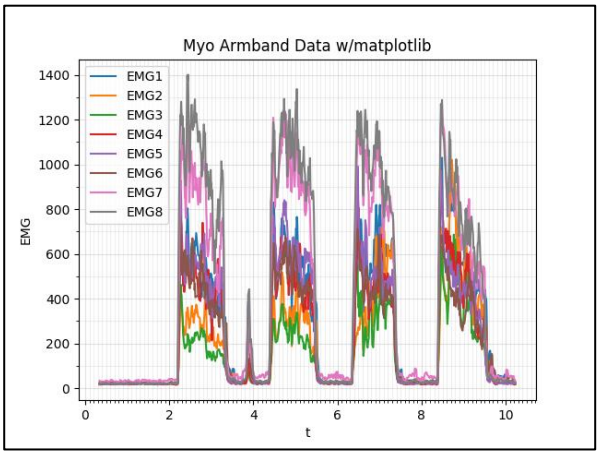


Figura 3.5 Resultado del movimiento 1 del voluntario 3

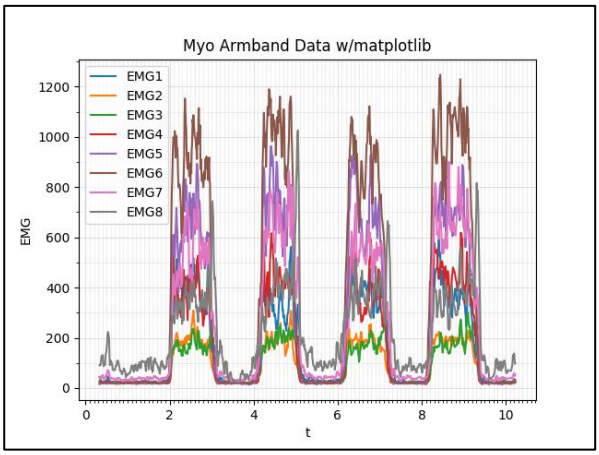


Figura 3.6 Resultado del movimiento 2 del voluntario 1

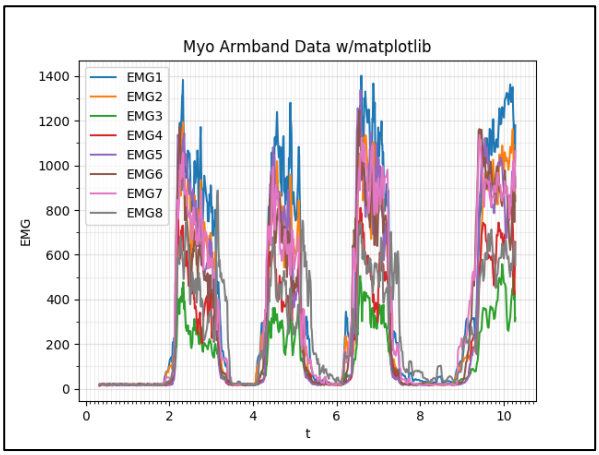


Figura 3.7 Resultado del movimiento 2 del voluntario 2

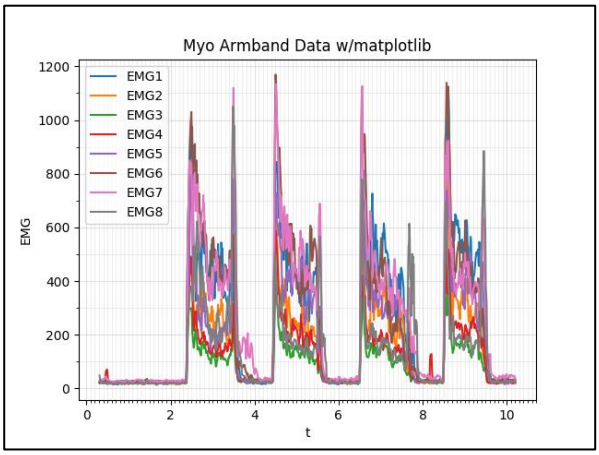


Figura 3.8 Resultado del movimiento 2 del voluntario 3

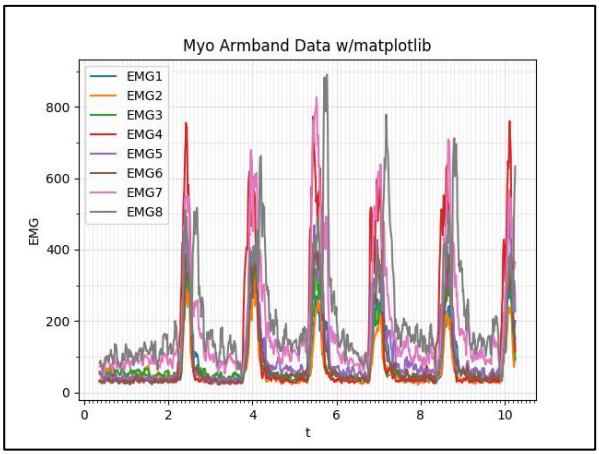


Figura 3.9 Resultado del movimiento 3 del voluntario 1

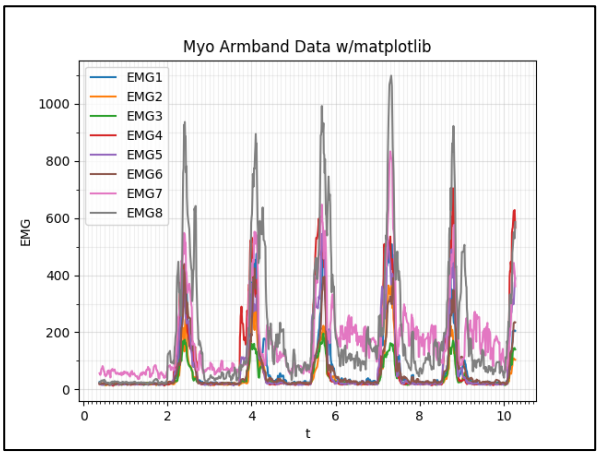


Figura 3.10 Resultado del movimiento 3 del voluntario 2

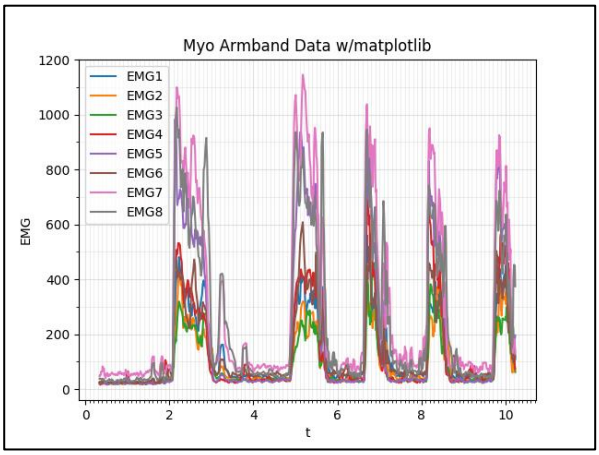


Figura 3.11 Resultado del movimiento 3 del voluntario 3

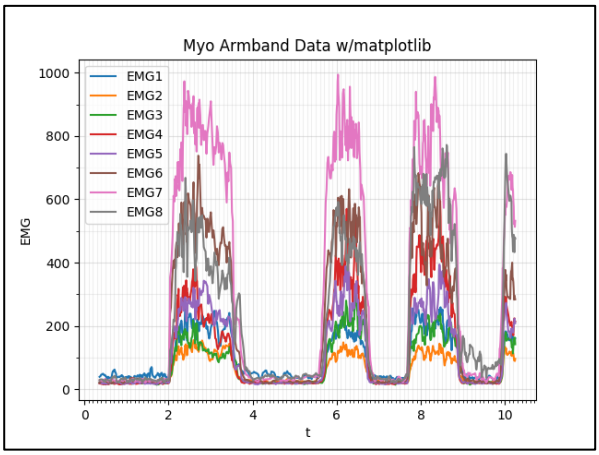


Figura 3.12 Resultado del movimiento 4 del voluntario 1

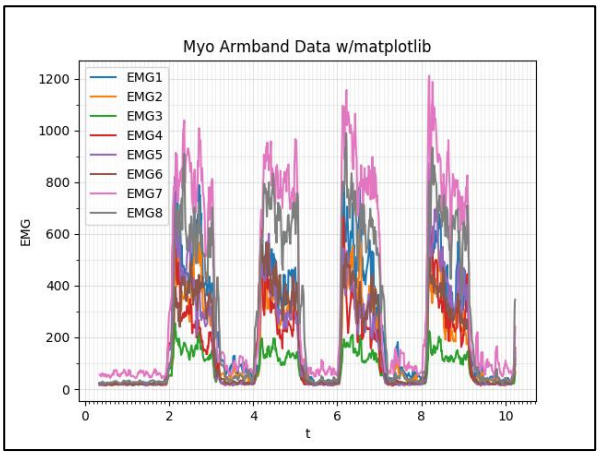


Figura 3.13 Resultado del movimiento 4 del voluntario 2

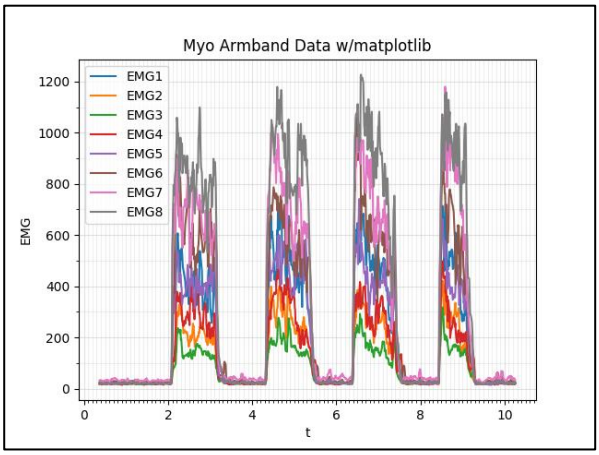


Figura 3.14 Resultado del movimiento 4 del voluntario 3

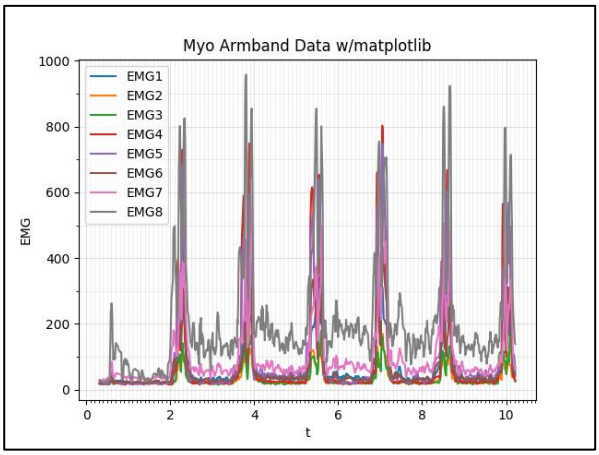


Figura 3.15 Resultado del movimiento 5 del voluntario 1

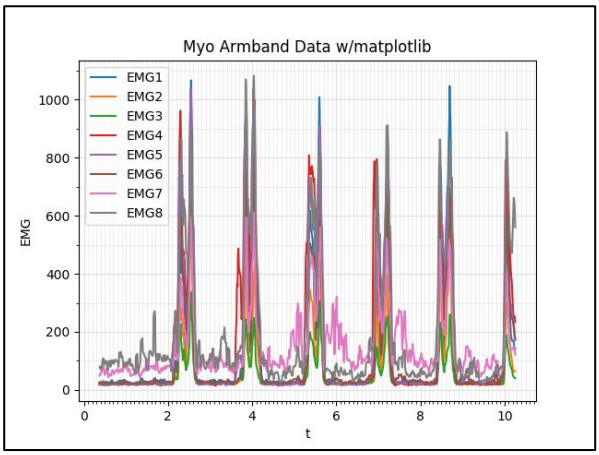


Figura 3.16 Resultado del movimiento 5 del voluntario 2

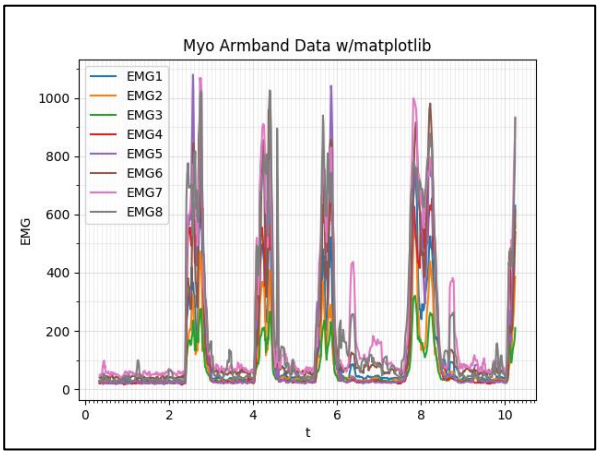


Figura 3.17 Resultado del movimiento 5 del voluntario 3

3.2 Análisis de Resultados

Las gráficas permitieron observar las señales detectadas por los sensores durante la realización de cada movimiento. De ellas podemos concluir que los voluntarios sí realizaron el movimiento que debían en cada ocasión, esto es porque los sensores que actuaban eran los mismos, diferenciándose solo en magnitud, lo cual dependía de la fuerza aplicada por cada persona.

Tomando como ejemplo las gráficas del movimiento 1, para los tres voluntarios, los sensores que presentaron mayor actividad fueron el 7 y 8 debido a que estos estaban ubicados encima de los músculos que actuaban durante el movimiento. De igual forma sucede para los demás movimientos. Con estas observaciones se pudo confirmar que los datos que se adquirieron en cada ocasión fueron coherentes.

Para corroborar, se realizó la correlación entre los tres grupos de datos del movimiento 1 y del movimiento 4.

Para obtener los valores de los coeficientes de correlación se utilizó el software Matlab, que cuenta con la función **corrcoef** [15]. Esta función calcula la correlación entre 2 variables aleatorias utilizando la ecuación 3.1.

$$\rho(X,Y) = \frac{cov(X,Y)}{\sigma_X \sigma_Y} \tag{3.1}$$

Donde:

$\rho(X,Y)$: correlación entre las variables X y Y.

$cov(X,Y)$: covarianza de X y Y.

σ_X, σ_Y: desviación estándar de las variables X y Y.

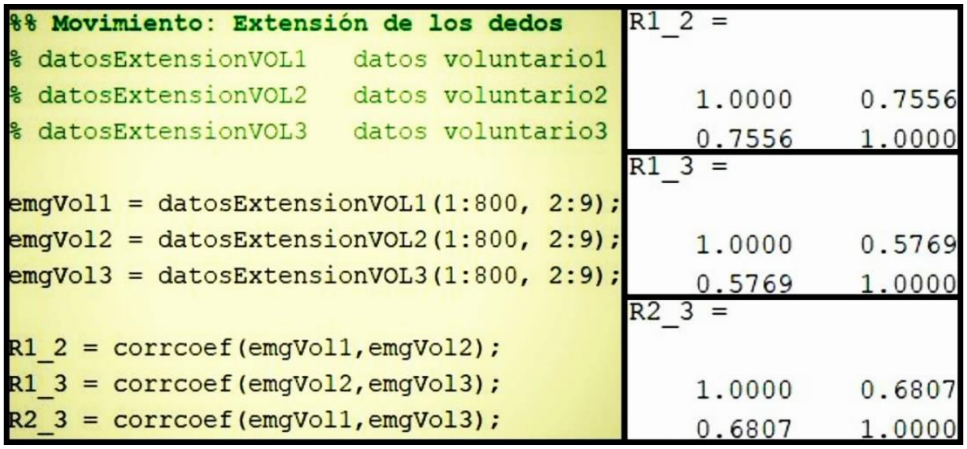
El resultado al ejecutar la función corrcoef se obtendrá en forma de matriz, así:

$$R = \begin{pmatrix} 1 & \rho(X,Y) \\ \rho(Y,X) & 1 \end{pmatrix}$$

Los valores de 1 de la diagonal principal corresponden a la correlación entre (X,X) y (Y,Y).

En la Figura 3.18. se observan los coeficientes de correlación de todos los datos mioeléctricos entre los tres voluntarios; R1_2 corresponde a la correlación entre los

datos del voluntario 1 y 2, R1_3 a la de los voluntarios 1 y 3 y R2_3 a la de los voluntarios 2 y 3.

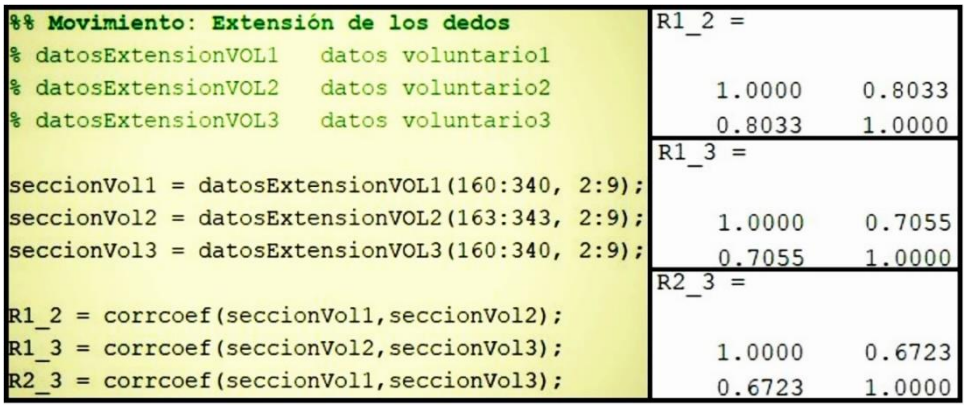


```
%% Movimiento: Extensión de los dedos
% datosExtensionVOL1   datos voluntario1
% datosExtensionVOL2   datos voluntario2
% datosExtensionVOL3   datos voluntario3

emgVol1 = datosExtensionVOL1(1:800, 2:9);
emgVol2 = datosExtensionVOL2(1:800, 2:9);
emgVol3 = datosExtensionVOL3(1:800, 2:9);

R1_2 = corrcoef(emgVol1,emgVol2);
R1_3 = corrcoef(emgVol2,emgVol3);
R2_3 = corrcoef(emgVol1,emgVol3);
```

```
R1_2 =

    1.0000    0.7556
    0.7556    1.0000

R1_3 =

    1.0000    0.5769
    0.5769    1.0000

R2_3 =

    1.0000    0.6807
    0.6807    1.0000
```

Figura 3.18 Coeficientes de correlación de todos los datos del movimiento 1

Los valores obtenidos demuestran que existe una buena correlación entre los datos de los tres voluntarios, siendo la más alta entre los voluntarios 1 y 2 con un valor de 0.7556. Estos resultados nos confirman que los voluntarios ejecutaron el mismo movimiento. Además, se realizó la correlación entre los voluntarios justo en el momento de realizar el movimiento 1. En la Figura 3.19 se observan los resultados obtenidos.



```
%% Movimiento: Extensión de los dedos
% datosExtensionVOL1   datos voluntario1
% datosExtensionVOL2   datos voluntario2
% datosExtensionVOL3   datos voluntario3

seccionVol1 = datosExtensionVOL1(160:340, 2:9);
seccionVol2 = datosExtensionVOL2(163:343, 2:9);
seccionVol3 = datosExtensionVOL3(160:340, 2:9);

R1_2 = corrcoef(seccionVol1,seccionVol2);
R1_3 = corrcoef(seccionVol2,seccionVol3);
R2_3 = corrcoef(seccionVol1,seccionVol3);
```

```
R1_2 =

    1.0000    0.8033
    0.8033    1.0000

R1_3 =

    1.0000    0.7055
    0.7055    1.0000

R2_3 =

    1.0000    0.6723
    0.6723    1.0000
```

Figura 3.19 Coeficientes de correlación de los datos adquiridos en el primer momento de realizar el movimiento 1

Al analizar los resultados obtenidos en este caso, se pudo comprobar con mayor seguridad que todos los voluntarios realizaron el movimiento “Extensión de los

dedos”, y esto permitió comprobar de igual forma el correcto funcionamiento del equipo portátil.

De igual forma, se realizó el mismo análisis para el movimiento 4, tanto de todos los datos obtenidos en el experimento y del primer momento en el cual realizaron el movimiento. Los resultados obtenidos se detallan en las Figuras 3.20 y 3.21.

```
%% Movimiento: Pulgar Arriba
% datosPulgarVOL1   datos voluntario1
% datosPulgarVOL2   datos voluntario2
% datosPulgarVOL3   datos voluntario3

emgVol1 = datosPulgarVOL1(1:790, 2:9);
emgVol2 = datosPulgarVOL2(1:790, 2:9);
emgVol3 = datosPulgarVOL3(1:790, 2:9);

R1_2 = corrcoef(emgVol1,emgVol2);
R1_3 = corrcoef(emgVol2,emgVol3);
R2_3 = corrcoef(emgVol1,emgVol3);
```

```
R1_2 =

    1.0000    0.8282
    0.8282    1.0000

R1_3 =

    1.0000    0.6785
    0.6785    1.0000

R2_3 =

    1.0000    0.7269
    0.7269    1.0000
```

Figura 3.20 Coeficientes de correlación de todos los datos del movimiento 4

```
%% Movimiento: Pulgar Arriba
% datosPulgarVOL1   datos voluntario1
% datosPulgarVOL2   datos voluntario2
% datosPulgarVOL3   datos voluntario3

seccionVol1 = datosPulgarVOL1(158:340, 2:9);
seccionVol2 = datosPulgarVOL2(160:342, 2:9);
seccionVol3 = datosPulgarVOL3(161:343, 2:9);

R1_2 = corrcoef(seccionVol1,seccionVol2);
R1_3 = corrcoef(seccionVol2,seccionVol3);
R2_3 = corrcoef(seccionVol1,seccionVol3);
```

```
R1_2 =

    1.0000    0.7967
    0.7967    1.0000

R1_3 =

    1.0000    0.7689
    0.7689    1.0000

R2_3 =

    1.0000    0.8265
    0.8265    1.0000
```

Figura 3.21 Coeficientes de correlación de los datos adquiridos en el primer momento de realizar el movimiento 4

Los resultados nuevamente confirmaron que los tres voluntarios realizaron el mismo patrón de movimiento, y también permitió corroborar que el equipo portátil funciona de forma correcta.

3.3 Comparación entre datos adquiridos con Electrodos superficiales vs. datos adquiridos con el Myo Armband

Se realizaron dos pruebas para comparar los datos mioeléctricos obtenidos con el brazalete (prueba 1) y los datos que los investigadores de GIACI obtenían utilizando 3 electrodos superficiales junto a la Olimex EKG-EMG shield (prueba 2).

El voluntario 1 realizó el movimiento 3 en ambas ocasiones, lo resultados gráficos se observan en las Figuras 3.22 y 3.23.

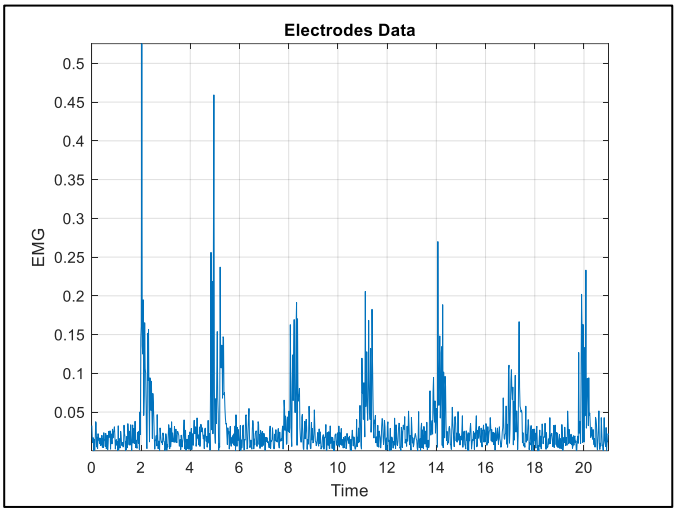


Figura 3.22 Datos mioeléctricos adquiridos con 3 electrodos superficiales

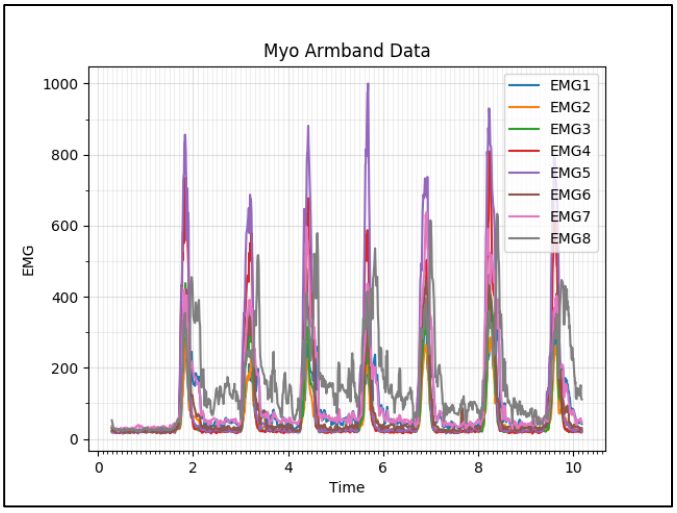


Figura 3.23 Datos mioeléctricos adquiridos con el Myo Armband

Antes de realizar la correlación entre ambas señales se las normalizó, debido a que la magnitud de las señales EMG en cada prueba eran muy diferentes; esto ocurrió porque el brazalete tiene un coeficiente de amplificación distinto al que tiene la Olimex. Las figuras 3.24 y 3.25 muestran las señales mioeléctricas normalizadas.

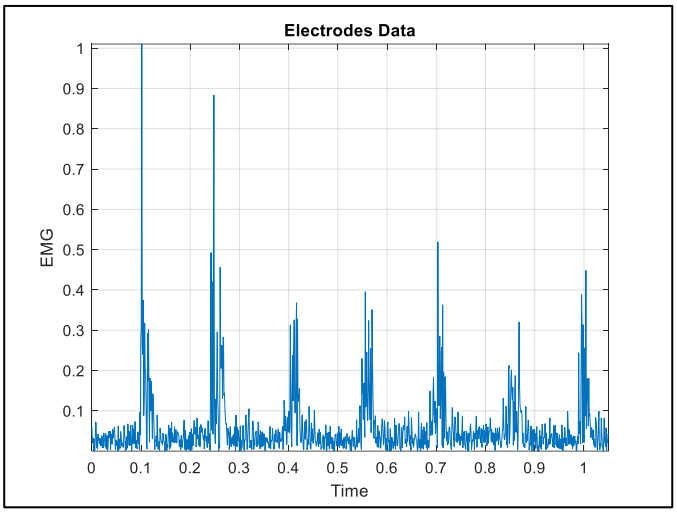


Figura 3.24 Datos normalizados (adquiridos con electrodos superficiales)

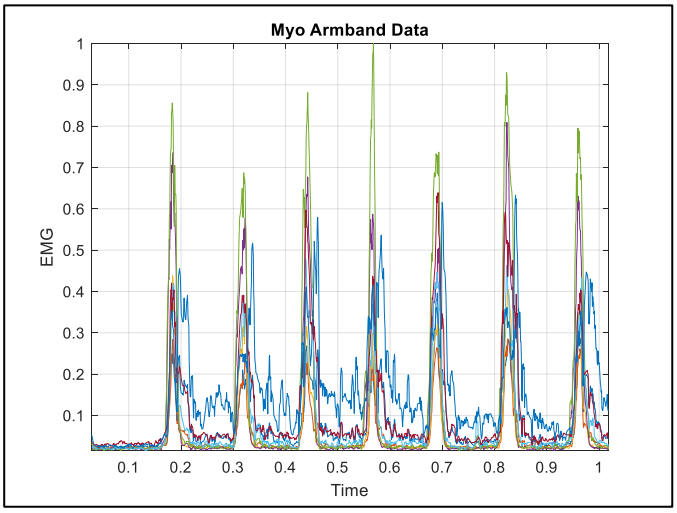


Figura 3.25 Datos normalizados (adquiridos con Myo Armband)

Sólo observando las gráficas se pudo concluir que el comportamiento de las señales mioeléctricas durante la adquisición con ambos equipos es el mismo. Con el fin de confirmar esto, se realizó la correlación entre los datos adquiridos con los electrodos y los datos adquiridos con el brazalete, para esta operación se usaron

los datos de los sensores 4 y 5 del brazalete ya que para la segunda prueba los electrodos se colocaron en la posición donde estaban ubicados estos sensores. Se observan los resultados en la Figura 3.26

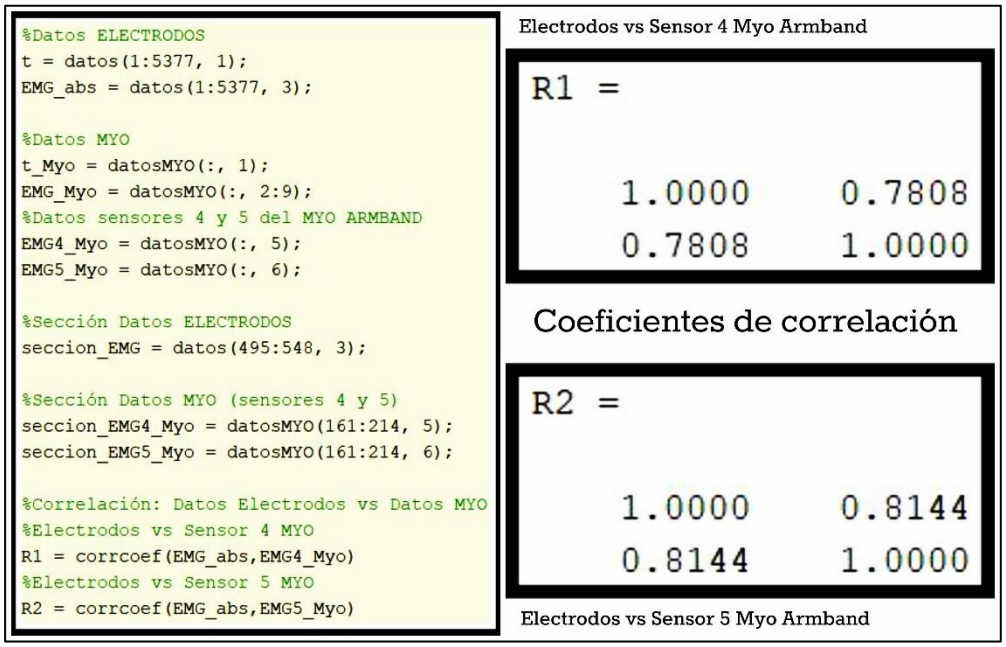


Figura 3.26 Coeficientes de correlación entre los datos adquiridos con ambos equipos

Como se observa en la Figura 3.26, R1 y R2 representan los coeficientes de correlación entre los datos adquiridos con los electrodos y los datos adquiridos por los sensores 4 y 5 del Myo Armband, respectivamente.

Estos resultados indicaron que no habría inconvenientes al querer realizar la adquisición con el equipo portátil ya que los datos que se obtendrán se comportarán de la misma forma que cuando se realiza la adquisición con los equipos del laboratorio en donde los investigadores de GIACI trabajan actualmente.

3.4 Efectos aplicando temperatura

Se realizaron también dos pruebas para observar la reacción de los músculos al verse sometidos al calor. Para dichas pruebas se mantuvo el brazo del voluntario en estado de reposo y luego se aproximó una pequeña fuente de calor a una parte

del brazo que estuviera en la dirección de un sensor en específico. Estas pruebas se realizaron con una temperatura ambiente de 28°C. Ver Figuras 3.27 y 3.28.

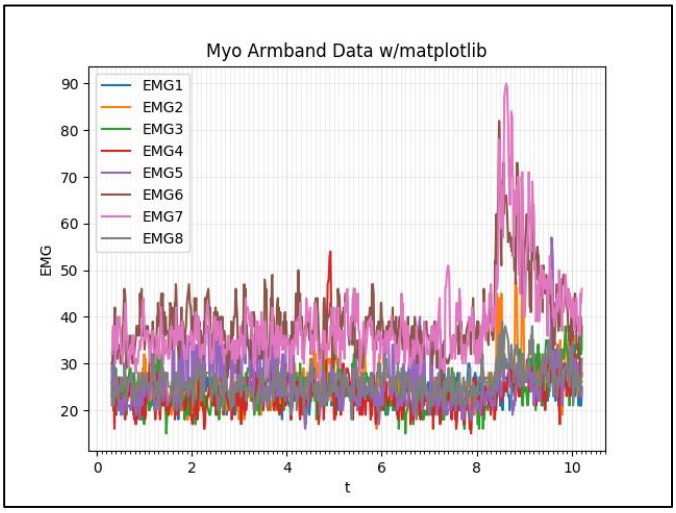


Figura 3.27 Datos obtenidos al aplicar temperatura a la parte posterior del brazo derecho del voluntario 1

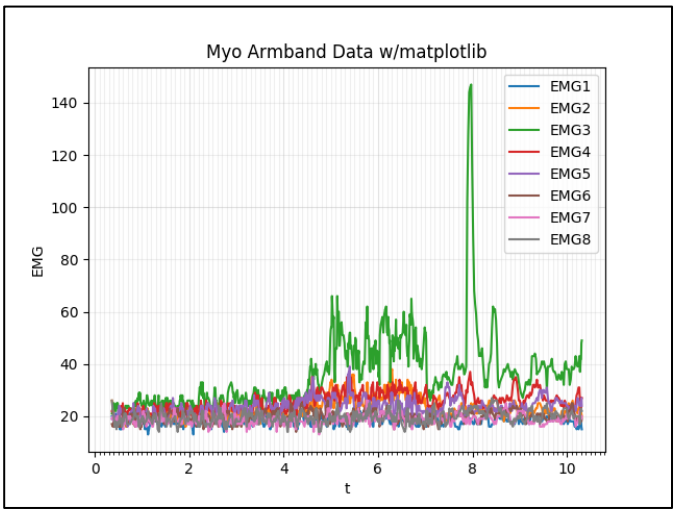


Figura 3.28 Datos obtenidos al aplicar temperatura a la parte anterior del brazo derecho del voluntario 1

En la primera prueba se aplicó el calor en la parte posterior del brazo derecho, y la gráfica de la Figura 3.27 lo confirmó ya que los sensores 6, 7 y 8 del brazalete se ubican en esa posición y fueron justamente estos los que se activaron.

En la segunda prueba, se aplicó el calor en la parte anterior del brazo derecho, es decir, la parte en donde en donde están ubicados los sensores 2, 3 y 4. Claramente

se observó el momento en que los sensores detectaron el calor, como se observa en la Figura 3.28. En este caso fue el sensor 3 que tuvo una mayor reacción ya que el calor se aplicó a un músculo que se encontraba en la misma dirección que este. Es importante tener un control sobre la temperatura del ambiente en el que se realizarán los experimentos que requieran aplicar calor al brazo de un voluntario, esto permitirá obtener datos más precisos y mayor facilidad al momento de realizar el procesamiento necesario.

Todos los experimentos realizados confirmaron el correcto funcionamiento del equipo portátil durante la adquisición, transmisión y almacenamiento de los datos mioeléctricos, cumpliendo así con los requerimientos exigidos por el cliente.

CAPÍTULO 4

4. CONCLUSIONES Y RECOMENDACIONES

4.1 Conclusiones

Se realizó la comunicación entre el Myo Armband y la Raspberry Pi mediante el protocolo bluetooth junto con varias librerías de Python; dichas librerías se tuvieron que estudiar profundamente debido a la programación espesa que contienen las mismas; esta fue una de las etapas que exigió más trabajo durante el diseño e implementación del prototipo.

Se logró crear una aplicación en la SBC que permita adquirir y visualizar los datos mioeléctricos obtenidos del Myo Armband.

Se guardaron los datos de los voluntarios en un documento de extensión .csv debido a la facilidad de manipular este tipo de archivos en otros softwares, como Matlab, para realizar el procesamiento respectivo. Los datos se enviaron vía e-mail a los investigadores después de realizar la adquisición, en dicho documento se guardaron tanto la información personal como los datos mioeléctricos de los voluntarios. Además para saber si hubieron errores durante la adquisición de datos, como movimientos adicionales por parte de los voluntarios, se mostró la gráfica EMG[mV] vs t[s] para observar con mayor claridad qué estaba ocurriendo durante las pruebas.

Se crearon diferentes copias del archivo en donde se guardan los datos de la adquisición durante cada prueba cuando no hubo conexión a internet, esto permitía al usuario no tener que preocuparse de perder información ni tampoco tener que manualmente ir creando estas copias.

Se demostró, con los experimentos realizados, que el equipo portátil adquiere sin problemas las señales mioeléctricas del brazo derecho de los voluntarios durante el tiempo que el usuario desee, esto indicó también que la conexión entre el brazalete y la Raspberry se realizó de forma correcta.

Se realizó la comparación entre los datos adquiridos con tres electrodos superficiales junto a la Olimex EKG-EMG Shield y los datos adquiridos con el Myo Armband; mediante coeficientes de correlación se demostró que que el

comportamiento de las señales durante la adquisición de datos es el mismo. Por ende, con el equipo portátil implementado en este proyecto se pueden realizar los experimentos que se realizaban anteriormente con otros materiales o equipos.

Se eliminó la limitación de adquirir los datos mioeléctricos de diferentes personas en un lugar fijo. Con este equipo portátil, los investigadores podrán adquirir una mayor cantidad de información en diferentes lugares y de un mayor número de personas para los análisis y desarrollo de proyectos que requieran.

Se creó una Nube propia llamada Pydio, para en un futuro almacenar todos los datos en su interior.

4.2 Recomendaciones

Utilizar Windows 10 para estudios previos del Myo Armband antes de utilizarlo en otros sistemas operativos. Bajo este ambiente el aprendizaje para el correcto manejo y eso del brazalete es menos complicado.

Mientras no exista conexión a internet y la aplicación esté activa, no forzar al cierre de esta cuando existen datos almacenados; dado que al encender de nuevo el equipo se tendrían que enviar estos datos manualmente; o en su defecto, en un fututo mejorar el programa para que verifique si existen datos almacenados cada que se ejecute la aplicación, ya sea que el equipo haya estado apagado o encendido, y enviarlos si existe conexión a internet.

En caso de que la adquisición de datos se lo realice en un lugar en donde haya que movilizarse entre salas o habitaciones, una batería portátil que cumpla con los requerimientos de alimentación de la Raspberry Pi evitaría tener que apagar y encender el equipo cada vez que se lo movilice.

Se recomienda habilitar la nube Pydio creada para poder almacenar en ella directamente los datos adquiridos con la SBC.

En un futuro se podría mejorar el programa colocando una opción que permita escoger si se desea enviar o eliminar los datos adquiridos, dado que en caso de que hubiera errores durante la adquisición, se podrían desechar esos datos y volver a realizar las pruebas.

BIBLIOGRAFÍA

[1] P. Konrad, The ABC of EMG, Scottsdale: Noraxon U.S.A. Inc., 2006.

[2] Thalmic Labs Inc., «Getting To Know The Myo Armband,» 17 Diciembre 2014. [En línea]. Available: https://developer.thalmic.com/docs/api_reference/platform/getting-started.html. [Último acceso: 21 Mayo 2018].

[3] Thalmic Labs Inc., «Myo Armband Tech Specs,» 17 Diciembre 2014. [En línea]. Available: https://www.myo.com/techspecs. [Último acceso: 26 Mayo 2018].

[4] Raspberry Pi Foundation, «Raspberry Pi 3 Model B+,» 12 Marzo 2018. [En línea]. Available: https://www.raspberrypi.org/products/raspberry-pi-3-model-b-plus/. [Último acceso: 1 Junio 2018].

[5] M. N. López, V. Toranzos y O. G. Lombardero, «Sistema de adquisición y visualización de señales mioeléctricas,» *SABII 2011,* Septiembre 2011.

[6] R. N. Aguilar Serena, D. F. Hinojosa de la Rosa, O. O. Sandoval González, I. Herrera Aguilar y B. E. Gonzáles Sánchez, «Diseño de un sistema de adquisición de señales electromiográficas inalámbrico,» *CIINDET 2013,* Marzo 2013.

[7] P. L. Milea, A. Barbilian, M. Moga, M. E. Pogarasteanu, A. M. Oproiu, V. Lazo y C. I. Stoica, «A new Method for Myoelectric Signal Acquisition: Preparing the Patients to Efficiently Use an Artificial Arm,» *ROMANIAN JOURNAL OF INFORMATION SCIENCE AND TECHNOLOGY,* vol. XX, nº 2, pp. 115-123, 2017.

[8] Thalmic Labs Inc., «MyoDiagnostics,» 17 Diciembre 2014. [En línea]. Available: http://diagnostics.myo.com/. [Último acceso: 15 Junio 2018].

[9] Pygame Community, «Pygame Tutorials,» 12 Marzo 2018. [En línea]. Available: https://www.pygame.org/wiki/tutorials. [Último acceso: 2 Julio 2018].

[10] S. Kimball y P. Mattis, «Tutoriales de GIMP,» 25 Septiembre 1997. [En línea]. Available: http://www.gimp.org.es/modules/downloadse/. [Último acceso: 5 Junio 2018].

[11] D. Zhu, «Myo Raw - GitHub,» 1 Diciembre 2014. [En línea]. Available: https://github.com/dzhu/myo-raw. [Último acceso: 22 Junio 2018].

[12] Python Software Foundation, «smtplib — SMTP protocol client,» [En línea]. Available: https://docs.python.org/3/library/smtplib.html. [Último acceso: 18 Junio 2018].

[13] CISCO, «Using the Extended ping and Extended traceroute Commands,» 28 Julio 2006. [En línea]. Available: https://www.cisco.com/c/en/us/support/docs/ip/routing-information-protocol-rip/13730-ext-ping-trace.html. [Último acceso: 5 Julio 2018].

[14] ABSTRIUM SAS, «Pydio Documentation Library,» 22 Mayo 2013. [En línea]. Available: https://pydio.com/en/docs/administration-guides. [Último acceso: 20 Julio 2018].

ANEXOS

ANEXO A

Algoritmo – Aplicación Principal

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-

import csv
import pygame      #módulo usado para la creación de la consola
import sys,os      #módulo encargado de realizar ping
import smtplib     #módulo para enviar e-mail

#librerías importantes para envio de e-mail
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
from email.MIMEBase import MIMEBase
from email import encoders
from pygame import *
from pygame.locals import *

#representa el color de fondo de la consola
b=(255,255,255)

s=0
cnt=0
d=0

#variable usada para determinar si existe conexión a internet
hostname = "192.168.0.1"

#función para exportar arxhivo que se enviará via e-mail
def load_file(file, file_name):
    read_file = open(file,'rb')
    attach = MIMEBase('multipart', 'encrypted')
    attach.set_payload(read_file.read())
    read_file.close()
    encoders.encode_base64(attach)
    attach.add_header('Content-Disposition', 'attachment', filename=file_name)
    return attach
def sendemail(addr_to_mail, addr_from_mail,n,s):
    smtp_server = 'smtp.gmail.com:587'
    smtp_user   = 'germangranados339@gmail.com'
    smtp_pass   = 'german31189105'
    email = MIMEMultipart()
    email['To'] = addr_to_mail
    email['From'] = addr_from_mail
    email['Subject'] = 'Archivo adjunto desde MYORASP'
    email.attach(MIMEText('<p style="color:red;" >Datos mioelectricos de
voluntario/a.</p>','html'))
    if n==0:
       email.attach(load_file('/home/pi/Documents/PyoConnect/PyoConnect_v2.0/files_TE
SIS/datosVoluntarios.csv','Datos.csv'))
    else:
```

```
      email.attach(load_file('/home/pi/Documents/PyoConnect/PyoConnect_v2.0/files_TE
SIS/datosVoluntarios'+str(s)+'.csv','Datos'+str(s)+'.csv'))
            os.remove('datosVoluntarios'+str(s)+'.csv')

    smtp = smtplib.SMTP(smtp_server)
    smtp.starttls()
    smtp.login(smtp_user,smtp_pass)
    smtp.sendmail(addr_from_mail, addr_to_mail, email.as_string())
    smtp.quit()

#función diseñada para detectar la posición del cursor
class Cursor (pygame.Rect):
      def __init__(self):
            pygame.Rect.__init__(self,0,0,1,1)
      def update (self):
            self.left, self.top=pygame.mouse.get_pos()

#función diseñada para el cambio de imagen al pasar el cursor sobre estas
class Boton (pygame.sprite.Sprite):
      def __init__ (self,b1,b2,x=200,y=200):
            self.imagen_normal = b1
            self.imagen_seleccion = b2
            self.imagen_actual = self.imagen_normal
            self.rect = self.imagen_actual.get_rect()
            self.rect.left,self.rect.top=(x,y)
      def update (self,pantalla,cursor):
            if cursor.colliderect (self.rect):
                  self.imagen_actual=self.imagen_seleccion
            else: self.imagen_actual = self.imagen_normal

            pantalla.blit (self.imagen_actual, self.rect)

#función encargada de detectar las teclas presionadas y escribirlas en la pantalla
class Entrada():
      def __init__ (self,):
            self.lineas = 0
            self.caracteres = [' ',]
            self.fuente = pygame.font.Font(None,30)

            self.distancia = 29

      def teclas(self,evento):
            for accion in evento:
                  if accion.type == KEYDOWN:
                        if accion.key == K_RETURN:
                              self.caracteres.append (' ')
                              self.lineas += 1
                        elif accion.key == K_BACKSPACE:
                              if self.caracteres[self.lineas] == '' and
self.lineas > 0:
                                    self.caracteres = self.caracteres[0:-1]
                                    self.lineas -= 1
                              else :
                                    self.caracteres[self.lineas] =
self.caracteres[self.lineas] [0:-1]
```

```
                        else:
                              self.caracteres[self.lineas] =
str(self.caracteres[self.lineas]) + accion.unicode

      def mensaje(self,superficie,x,y):
            for self.lineas in xrange(len(self.caracteres)):
                  Img_letra =
self.fuente.render(self.caracteres[self.lineas],True,(23,238,233))
                  superficie.blit(Img_letra,( x+25,y+ self.distancia *
self.lineas))
            if self.lineas == 0:
                  superficie.blit (self.fuente.render('-
>',True,(23,238,233)),(x+8,y+self.distancia * self.lineas))
            elif self.lineas == 1:
                  superficie.blit (self.fuente.render('-
>',True,(23,238,233)),(x+8,y+self.distancia * self.lineas))
            elif self.lineas == 2 :
                  superficie.blit (self.fuente.render('-
>',True,(23,238,233)),(x+8,y+self.distancia * self.lineas))
            elif self.lineas == 3 :
                  superficie.blit (self.fuente.render('-
>',True,(23,238,233)),(x+8,y+self.distancia * self.lineas))

def main():
      global s
      global cnt
      global d
      c=0

      #inicialización de la librería pygame
      pygame.init()

      #creación del área de la consola
      ventana = pygame.display.set_mode((700,400))

      #definición del título de la de consola
      pygame.display.set_caption("\t\t\t\t\t\t\tMyoRasp")
      #definición del tamaño, mensaje y color de letra
      fuente = pygame.font.Font(None, 30)
      enviando = fuente.render("Enviando archivos almacenados",0,(220,71,169))
      imp = fuente.render("Imposible salir faltan archivos por enviar",0,(255,0,0))
      #exportación de las imágenes usadas para la consola
      img_fiec =
pygame.image.load("/home/pi/Documents/Appy/Logos/Fiec.png").convert_alpha()
      img_espol =
pygame.image.load("/home/pi/Documents/Appy/Logos/ESPOL.png").convert_alpha()
      img_B1 =
pygame.image.load("/home/pi/Documents/Appy/Logos/B1.png").convert_alpha()
      img_B2 =
pygame.image.load("/home/pi/Documents/Appy/Logos/B2.png").convert_alpha()
      img_b1 =
pygame.image.load("/home/pi/Documents/Appy/Logos/b1.png").convert_alpha()
      img_b2 =
pygame.image.load("/home/pi/Documents/Appy/Logos/b2.png").convert_alpha()
```

```
	img_txt =
pygame.image.load("/home/pi/Documents/Appy/Logos/text.png").convert_alpha()
	img_err =
pygame.image.load("/home/pi/Documents/Appy/Logos/error.png").convert_alpha()
	img_mail = pygame.image.load("/home/pi/Documents/Appy/Logos/e-
mail.png").convert_alpha()
	img_almc =
pygame.image.load("/home/pi/Documents/Appy/Logos/almc.png").convert_alpha()
	img_conx =
pygame.image.load("/home/pi/Documents/Appy/Logos/conx.jpeg").convert_alpha()
	img_sinconx =
pygame.image.load("/home/pi/Documents/Appy/Logos/sin_conx.jpg").convert_alpha()
	img_na =
pygame.image.load("/home/pi/Documents/Appy/Logos/n_a.png").convert_alpha()

	#definición de las posiciones de las imágenes
	posX=0
	posY=0
	posX1=550
	posY1=250
	x=255
	y=275
	t = 0
	net = os.system("ping -c 1 " + hostname)
	#variables usadas para el cambio de botones
	boton1=Boton(img_B1,img_b1,x,y)
	boton2=Boton(img_B2,img_b2,x,y)
	cursor1=Cursor()

	escritura = Entrada()

	#ciclo usado para detectar los eventos realizados mediante teclado o mouse
	while True:
		if t == 500:
			net = os.system("ping -c 1 " + hostname)
			t=0
		t=t+1
		#actualiza la consola para permitir conocer los eventos en marcha
		pygame.display.update()
		#establecer el color de fondo de la consola
		ventana.fill(b)
		#establecer las imágenes y la posición de las mismas
		ventana.blit(img_fiec,(posX,posY))
		ventana.blit(img_espol,(posX1,posY1))
		ventana.blit(img_na,(240,140))
		if net == 0:
			ventana.blit(img_conx,(616,0))
		else:
			ventana.blit(img_sinconx,(595,0))

		eventos = pygame.event.get()
		#Este if sirve para enviar archivos almacenados por motivos de ausencia
		de internet
		if cnt >0 and net == 0:
			for i in range(0,cnt):
```

```
                s=s+1
                while d<=3:
                        pygame.display.update()
                        ventana.blit(enviando,(220,265))
                        d=d+1
                        pygame.time.delay(600)


                sendemail('kgmolina@espol.edu.ec','kathy.molina2509@gmail.c
                om',cnt,s)
        escritura.lineas=0
        escritura.caracteres = [' ',]
        cnt = s = d= 0
elif escritura.lineas <= 3 and cnt >= 0 :
        escritura.teclas(eventos)
        escritura.mensaje(ventana,315,151)
        info = escritura.caracteres
        for evento in eventos:
                #con este if se cierra la venta de la consola
                if evento.type == QUIT:
                        if cnt > 0:
                                while d<=3:
                                        pygame.display.update()
                                        ventana.blit(imp,(185,116))
                                        d=d+1
                                        pygame.time.delay(600)
                                d=0
                        else:
                                pygame.quit()
                                sys.exit()
#se realiza un ciclo para determinar si existe algún tipo de evento
dentro de la consola
else:
        boton1.update(ventana,cursor1)
        cursor1.update()
        salir=False
        escritura.mensaje(ventana,315,151)
        for evento in eventos:
                #con este if se cierra la venta de la consola
                if evento.type == QUIT:
                        if cnt > 0:
                                while d<=3:
                                        pygame.display.update()
                                        ventana.blit(imp,(185,116))
                                        d=d+1
                                        pygame.time.delay(600)
                                d=0
                        else:
                                pygame.quit()
                                sys.exit()
                #se verifica qué de botón es presionado en el mouse
                if evento.type==pygame.MOUSEBUTTONDOWN:
                #valida si se presiona sobre el botón
                        if cursor1.colliderect(boton1.rect):
                #valida si se hizo click izquierdo
```

```
                                if evento.button==1:
                            #se realiza el proceso de aquisición de datos
                                        while salir != True:
                                               pygame.display.update()
                                               boton2.update(ventana,cursor1)

                                               cursor1.update()
                                               pygame.time.delay(500)

                                               if(net == 0):
                                      #llamada de la función encargada de enviar e-mail
                                                      while d<=3:

       pygame.display.update()
       ventana.blit(img_mail,(310,315))
       d=d+1

       pygame.time.delay(600)
       escritura.lineas=0
       escritura.caracteres = [' ',]
                                                      dataFile =
open("datosVoluntarios.csv","w")

       dataFile.write(str(info[0])+','+str(info[1])+','+str(info[2])+','+str(info[3])
+'\n')
                                                      dataFile.write("#
Muestras"+','+"EMG1(mV)"+','+"EMG2(mV)"+','+"EMG3(mV)"+','+"EMG4(mV)"+','+"EMG5(mV)"+
','+"EMG6(mV)"+','+"EMG7(mV)"+','+"EMG8(mV)"+'\n')
                                                      dataFile.close()
                                                      os.system("/usr/bin/env
python /home/pi/Documents/PyoConnect/PyoConnect_v2.0/files_TESIS/test2.py")
                                                      os.system("/usr/bin/env
python /home/pi/Documents/PyoConnect/PyoConnect_v2.0/files_TESIS/plotEMG.py")
                                                      d=0
                                                      salir=True

       sendemail('kgmolina@espol.edu.ec','kathy.molina2509@gmail.com',cnt,s)
                                                elif(net != 0):
                                                      while d<=3:

       pygame.display.update()

       ventana.blit(img_almc,(280,315))
                                                            d=d+1
       pygame.time.delay(600)
                                                            escritura.lineas=0
                                                            escritura.caracteres
= [' ',]
                                                      cnt=cnt+1
                                                      dataFile =
open("datosVoluntarios.csv","w")#open("Datos"+str(cnt)+".csv","w")

       dataFile.write(str(info[0])+','+str(info[1])+','+str(info[2])+','+str(info[3])
+'\n')
                                                      dataFile.close()
```

```
                                                        os.system("/usr/bin/env
python /home/pi/Documents/PyoConnect/PyoConnect_v2.0/files_TESIS/test2.py")
                                                        f =
open("datosVoluntarios"+str(cnt)+".csv","w")
                                                        n = open
("datosVoluntarios.csv" , "r")
                                                        while True:
                                                              linea = n.readline()
                                                              if not linea:
                                                                    f.close()
                                                                    n.close()
                                                                    break
                                                              f.write(linea)
                                                        os.system("/usr/bin/env
python /home/pi/Documents/PyoConnect/PyoConnect_v2.0/files_TESIS/plotEMG.py")
                                                        d=0
                                                        salir=True
                                                  for evento in pygame.event.get():
                                                  #se sale del ciclo while que
vigila la actidad del mouse
                                                        if evento.type == QUIT:
                                                              if cnt > 0:
                                                                    while d<=3:

      pygame.display.update()

      ventana.blit(imp,(185,116))
                                                                          d=d+1


      pygame.time.delay(600)
                                                                    d=0
                                                              else:
                                                                    pygame.quit()
                                                                    sys.exit()
                        #muestra un mensaje de advertencia al usuasio cuando no se
presiona el botón con click izquierdo
                                      else:
                                            ventana.blit(img_txt,(225,300))
                                            pygame.time.delay(1500)
                        #muestra un mensaje de advertencia al usuasio cuando se
presiona fuera del botón
                                 else:
                                      ventana.blit(img_err,(253,300))
                                      pygame.time.delay(1500)
#ejecución del programa principal
main()
```

ANEXO B

Algoritmo – Adquisición de Datos

```
from __future__ import print_function
import os
import sys
import myo_raw2
import time

# tiempo de adquisición de datos
time_max = 10.000000
start_time = time.clock()

# crear archivos en donde se almacenarán las señales mioeléctricas
dataFile = open("datos.csv","w")
f = open("datosVoluntarios.csv","a")
# variable de control del Myo Armband
m = myo_raw2.MyoRaw(sys.argv[1] if len(sys.argv) >= 2 else None)
contador = 0

# función para acceder a los datos EMG de los 8 sensores
# y almacenarlos en los archivos
def proc_emg(emg, moving, times=[]):
	global contador
	contador+=1
	print(emg)
	dataFile.write(str(time.clock())+","+str(emg[0])+","+str(emg[1])+","+str(emg[2
])+","+str(emg[3])+","+str(emg[4])+","+str(emg[5])+","+str(emg[6])+","+str(emg[7])+"\
n")
	f.write(str(contador)+","+str(emg[0])+","+str(emg[1])+","+str(emg[2])+","+str(
emg[3])+","+str(emg[4])+","+str(emg[5])+","+str(emg[6])+","+str(emg[7])+"\n")

#función principal
def getData():
	m.add_emg_handler(proc_emg)
	m.connect()
	while (time.clock() - start_time) < time_max:
		m.run(1)
	dataFile.close
	m.disconnect()
	print("\nTiempo de inicio (s): ",start_time)
	print("\nTiempo total de adquisicion de datos (s): ", time.clock())

#ejecución de la función principal
getData()
```

ANEXO C

Algoritmo – Gráfica de Señales Mioeléctricas

```
# -*- coding: utf-8 -*-
import numpy as np
import matplotlib.pyplot as plt
import csv

# listas para almacenar el tiempo y los datos mioeléctricos
t = []
emg=[[],[],[],[],[],[],[],[]]

# inicializar una ventana para mostrar la gráfica EMG vs t
fig = plt.figure()
ax = fig.add_subplot(1, 1, 1)

# definir valor de inicio y fin del intervalo en el eje Y, y el espacio entre cada
valor en el eje.
major_ticks = np.arange(0, 2000, 100)
# definir valor de inicio y fin del intervalo en el eje X, y el espacio entre cada
valor en el eje.
minor_ticks = np.arange(0, 15, 0.1)

#ax.set_xticks(major_ticks)
ax.set_xticks(minor_ticks,minor=True)
ax.set_yticks(major_ticks,minor=True)

# activar cuadrícula
ax.grid(which='both',alpha=0.2)

# acceder al archivo para almacenar los datos en las listas t y emg
with open('datos.csv','r') as csvfile:
	plots = csv.reader(csvfile, delimiter=',')
	for row in plots:
		t.append(float(row[0]))
		emg[0].append(float(row[1]))
		emg[1].append(float(row[2]))
		emg[2].append(float(row[3]))
```

```
                emg[3].append(float(row[4]))
                emg[4].append(float(row[5]))
                emg[5].append(float(row[6]))
                emg[6].append(float(row[7]))
                emg[7].append(float(row[8]))

# graficar los valores almacenados en las listas
plt.plot(t,emg[0], label='EMG1')
plt.plot(t,emg[1], label='EMG2')
plt.plot(t,emg[2], label='EMG3')
plt.plot(t,emg[3], label='EMG4')
plt.plot(t,emg[4], label='EMG5')
plt.plot(t,emg[5], label='EMG6')
plt.plot(t,emg[6], label='EMG7')
plt.plot(t,emg[7], label='EMG8')
plt.grid(which='minor', alpha=0.2)

plt.xlabel('t')
plt.ylabel('EMG')
plt.title('Myo Armband Data w/matplotlib')
plt.legend()
ax.grid(which='both',alpha=0.2)

plt.savefig('myoData.png')
plt.show()
```