UNIVERSIDAD TÉCNICADEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

ESCUELA DE INGENIERÍA EN MECATRÓNICA

TRABAJO DE GRADO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN MECATRÓNICA

TEMA:
“ALGORITMO DE CONTROL PARA UN BRAZO ROBÓTICO MEDIANTE SEÑALES MIOELÉCTRICAS PROVENIENTES DE LA ACCIÓN MUSCULAR DE LAS EXTREMIDADES SUPERIORES.”

AUTOR: MICHAEL WILLIAM QUINTANA LÓPEZ

DIRECTOR: ING. IVÁN IGLESIAS

IBARRA – ECUADOR
2020

UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

AUTORIZACIÓN DE USO Y PUBLICACIÓN A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

1. IDENTIFICACIÓN DE LA OBRA

En cumplimiento del Art. 144 de la Ley de Educación Superior, hago la entrega del presente trabajo a la Universidad Técnica del Norte para que sea publicado en el Repositorio Digital Institucional, para lo cual pongo a disposición la siguiente información:

DATOS DEL CONTACTO			
CÉDULA DE IDENTIDAD:	131497706-5		
APELLIDOS Y NOMBRES:	MICHAEL WILLIAM QUINTANA LÓPEZ		
DIRECCIÓN:	EDUARDO GARZÓN Y JOSÉ YÉPEZ		
EMAIL:	mwquintanal@utn.edu.ec		
TELÉFONO FIJO:	062616543	**TELÉFONO MOVIL**	0960134111
DATOS DE LA OBRA			
TÍTULO	ALGORITMO DE CONTROL PARA UN BRAZO ROBÓTICO MEDIANTE SEÑALES MIOELÉCTRICAS PROVENIENTES DE LA ACCIÓN MUSCULAR DE LAS EXTREMIDADES SUPERIORES		
AUTOR	MICHAEL WILLIAM QUINTANA LÓPEZ		
FECHA	7/12/2020		
PROGRAMA	PREGRADO		
TÍTULO POR EL QUE OPTA	INGENIERÍA EN MECATRÓNICA		
DIRECTOR	MSC. IVÁN IGLESIAS NAVARRO		

2. CONSTANCIAS

El autor manifiesta que la obra objeto de la presente autorización es original y se la desarrollo, sin violar derechos de autor de terceros, por lo tanto, la obra es original y que es el titular de los derechos patrimoniales, por lo que asume la responsabilidad sobre el contenido de esta y saldrá en defensa de la Universidad en caso de reclamación por parte de terceros.

Ibarra a los 7 días de Diciembre del 2020.

Firma:

Michael William Quintana López

131497706-5

UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

CERTIFICACIÓN

En calidad de director del presente trabajo de titulación denominado: **ALGORITMO DE CONTROL PARA UN BRAZO ROBÓTICO MEDIANTE SEÑALES MIOELÉCTRICAS PROVENIENTES DE LA ACCIÓN MUSCULAR DE LAS EXTREMIDADES SUPERIORES**. Ha sido desarrollado por el Sr. Michael William Quintana López para optar por el título de Ingeniero en Mecatrónica, certifico que el mencionado proyecto fue realizado bajo mi dirección.

Es todo cuanto puedo certificar en honor a la verdad.

Atentamente,

IVAN IGLESIAS NAVARRO

Digitally signed by IVAN IGLESIAS NAVARRO
Date: 2020.12.04 10:14:26 -05'00'

MSc. Iván Iglesias Navarro

DIRECTOR

AGRADECIMIENTO

Agradezco a mis padres, Ramón Quintana y Dolores López por el inmenso apoyo recibido de su parte, nada de lo que haya logrado en mi vida fuese posible sin ellos, ya que soy un reflejo de todos los valores que me han sido inculcados a lo largo de mi corta existencia. Siempre serán lo mejor de mi vida. Los amo.

A mi hermano Andrés Quintana, quién merece una mención especial, por ser mi amigo desde que tengo uso de razón, por las alegrías, las peleas, los regaños y el apoyo que siempre he recibido de su parte.

A mis hermanos, Eduardo, Christian y Steven Quintana, por siempre estar a mi lado y darme la motivación necesaria para seguir avanzando.

A mis mejores amigos David Silva y Samuel De Mera por su apoyo incondicional y por creer en mí, por los buenos momentos, memorias que siempre llevaremos con cariño dentro de nosotros, ¡gracias! siempre serán parte de mi familia.

Al Sr. Eduardo Véliz y familia por brindarme siempre su apoyo en los momentos más difíciles de mi carrera.

A mi grupo de trabajo, Jahir Campoverde, Bryan García y Fernando Lucero; gracias por esa competencia sana de conocimientos que siempre fue emocionante para mí.

A mis amigos y compañeros de carrera, por todos los trabajos cumplidos, el tiempo de convivencia, las salidas, y todos los buenos recuerdos que generaron en mí.

DEDICATORIA

Este trabajo lo dedico a mis padres, por ser siempre mi fuente de inspiración y motivación, por apoyarme, aunque eso significase perder ciertos privilegios, siempre ante poniendo mi seguridad y bienestar por encima de ellos. A todas las personas que marcaron una diferencia en mí, aquellos que me acompañaron en el camino y a aquellos que ahora no se encuentran con nosotros.

CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

LISTA DE ANEXOS

RESUMEN

En el siguiente trabajo de investigación se presenta el desarrollo de un algoritmo de control para el robot Mitsubishi RV-2F basado en interpretación de gestos mediante el procesamiento de señales electromiográficas provenientes de las extremidades superiores, específicamente de los músculos del antebrazo. El objetivo del proyecto es otorgar cierto grado de asistencia en la manipulación de objetos cotidianos por parte de personas con mal formaciones o amputaciones parciales en las extremidades superiores, cuyo número de personas alcanzan el medio millón en Ecuador y continúan en aumento.

Para el diseño del sistema de control mioeléctrico se aplicó la norma VDI 2206, cuyo proceso inicia con la etapa de adquisición de la señal para lo cual se emplea un dispositivo Myo Armband que cuenta con sensores superficiales de grado médico de acero inoxidable. En cuanto a la fase de procesamiento, interviene el software Matlab encargado de generar una interfaz de identificación de gestos y una tarjeta comercial Arduino Uno que activa las entradas del controlador. La comunicación entre la tarjeta comercial y el controlador CR750-D se realiza a través de los pines digitales del Arduino Uno, para poder activar las entradas del slot 1 del controlador, cuya programación interna se encarga de la correcta elección de la trayectoria que será ejecutada por el brazo robótico.

El sistema de trayectorias del robot consta de tres tipos de control: Automático, semiautomático y manual, a las cuales se accede mediante un menú basado en gestos.

ABSTRACT

The following research work presents the development of a control algorithm for the Mitsubishi RV-2F robot based on the interpretation of gestures by processing electromyographic signals from the upper extremities, specifically the muscles of the forearm. The objective of the project is to provide a certain degree of assistance in the handling of everyday objects by people with poorly formed or partial amputations in the upper extremities, whose number of people reaches half a million in Ecuador and continues to increase.

For the design of the myoelectric control system, the VDI 2206 standard was applied, the process of which begins with the signal acquisition stage, for which a Myo Armband device is used that has surface sensors of medical grade stainless steel. Regarding the processing phase, the Matlab software intervenes in charge of generating a gesture identification interface and an Arduino Uno commercial card that activates the controller inputs. The communication between the commercial card and the CR750-D controller is carried out through the digital pins of the Arduino Uno, in order to activate the inputs of slot 1 of the controller, whose internal programming is responsible for the correct choice of the path to be executed by the robotic arm.

The robot trajectory system consists of three types of control: Automatic, semi-automatic and manual, which can be accessed through a gesture-based menu.

INTRODUCCIÓN

ANTECEDENTES

La robótica ha experimentado durante las últimas décadas un avance notable debido a los desarrollos realizados en campos como la computación, sensores, electrónica y software. Llamando así la atención del mundo sobre ella cuando sus conocimientos se han aplicado en tareas que el ser humano no podría ser capaz de realizar por su hostilidad, complejidad, precisión o repetitividad; como lo son las exploraciones a volcanes, localización de naufragios, accidentes nucleares y viajes espaciales [1].

Cada vez se ha vuelto más frecuente que las fábricas estén mediana o completamente automatizadas mediante robots manipuladores. Sin embargo, la robótica no solamente ha incursionado en la industria, sino también en centros de investigación, universidades, centros de recreación, hospitales y hogares. Con relación a este último, se aprecia que la robótica es cada vez más familiar hasta el punto en el que se pueden observar robots realizando tareas domésticas, lo que conlleva a pensar que es cada vez más accesible a la familia promedio [2].

El avance mencionado anteriormente y la familiarización de la robótica han permitido que áreas científicas como la medicina y la ingeniería trabajen en conjunto para la creación de nuevos campos multidisciplinarios, entre ellos la biomecatrónica, que consiste en la aplicación de los principios y técnicas desarrolladas por ambas áreas. Un ejemplo de esta convergencia de ciencias es un sistema integrado que reconozca señales electromiográficas (EMG), que son señales creadas a partir de las contracciones de las fibras musculares, y las transmita a un dispositivo electrónico para su posterior reconocimiento, análisis y aplicación. Dicho ejemplo se ha venido desarrollando en los últimos años para la creación de sistemas de control de prótesis robóticas mioeléctricas como es el caso de [3]. JPL por su parte ha estado trabajando en una interfaz basada en reconocimiento de gestos de la mano para el control de robots

manipuladores llamada BioSleeve que consiste en una banda que cubre la mayor parte del antebrazo e incluye 16 sensores de electromiografía de contacto seco más un par de unidades de medición de inercia [4]. Todo esto resulta en un campo de investigación amplio en el que se proyecta realizar alguna aportación con el presente trabajo de investigación.

DESCRIPCIÓN DEL PROBLEMA

La carrera de ingeniería en Mecatrónica de la Universidad Técnica del Norte ha planteado desde 2015 proyectos de investigación que involucran señales electromiográficas, con objetivo de rehabilitación mediante prótesis en pacientes con extremidades inferiores amputadas. Existen trabajos anteriores que estudian el análisis de estos tipos de señales y diseñan tarjetas de acondicionamiento para el control de prótesis, tal es el caso de los trabajos de grado [5] y [6], entre otros. El uso de señales electromiográficas para la asistencia a personas con discapacidades se encuentra en constante crecimiento gracias a su gran utilidad para controlar dispositivos electrónicos y robots manipuladores.

La carrera desea abordar el tema de las señales electromiográficas correspondiente a las extremidades superiores para el posterior control de un brazo robótico, con el fin de contribuir con el grupo de personas con malformaciones congénitas en las extremidades superiores o personas que han sufrido algún tipo de accidente que los haya dejado incapacitados y se les dificulte la manipulación de objetos. Según el registro nacional de discapacidades [7] en Ecuador existen actualmente 485.325 personas que registran algún tipo de discapacidad de los cuales el 46.64% son personas con discapacidad física, el cual representa el mayor porcentaje entre las discapacidades y este número va en aumento.

El tema del procesamiento de las señales mioeléctricas de las extremidades superiores surge de la necesidad de realizar y probar diferentes estrategias de control de estas señales principalmente para proyectos de inclusión social, según lo anterior mencionado, el problema

que se resolverá será el de añadir un control remoto intuitivo para un robot manipulador que proporcionará ayuda a personas con discapacidades, para el manejo de objetos cotidianos. Además de aportar a la línea de investigación Biomecatrónica y sus consecuentes aplicaciones dentro de proyectos, clases que ayuden a mejorar las destrezas de los estudiantes y demostraciones en diferentes eventos que se realizan en la universidad.

OBJETIVOS

Objetivo General

Diseñar un algoritmo de control para el robot Mitsubishi RV-2F mediante señales mioeléctricas provenientes de la acción muscular de las extremidades superiores.

Objetivos Específicos

- Adquisición y procesamiento de señales mioeléctricas de las extremidades superiores a través del dispositivo Myo Armband.
- Utilizar una interfaz de reconocimiento de gestos predeterminados para el control de la posición del robot.
- Implementar el control del brazo robótico con el brazalete Myo mediante un sistema de trayectorias.
- Realizar pruebas de funcionamiento del algoritmo de control del brazo robótico.

JUSTIFICACIÓN

La interacción hombre-máquina es cada vez más frecuente y versátil permitiendo así a los robots manipuladores recrear movimientos propios de las personas que los operan, todo esto es posible gracias al tratamiento de señales provenientes de las acciones musculares y articulares de los seres vivos, dichas señales son denominadas como mioeléctricas.

Actualmente el uso de tecnologías inalámbricas, como el dispositivo Myo Armband, se encuentra en crecimiento ya que sus posibles implementaciones son amplias en sectores industriales, manufactura, educación, entre otras. Una de las ventajas más grande de este tipo de sistema de control basado en señales mioeléctricas es la adaptabilidad y compatibilidad que tiene con diversos sistemas operativos. Lo que permite que el proyecto sea viable.

Lo mencionado exhorta a la creación de nuevas aplicaciones que faciliten la comunicación humano-máquina para mejorar el rendimiento en tareas cotidianas repetitivas a personas con discapacidades físicas presentes en extremidades superiores. Cumpliendo así con el objetivo 3 del Plan Nacional del Buen Vivir que plantea el mejoramiento de la calidad de vida de la población ecuatoriana y siendo una herramienta útil al área biomecatrónica para futuros estudios orientados al control de robots. Comprendiendo un aporte social e investigativo considerable.

ALCANCE

Se emplearán señales mioeléctricas provenientes de las extremidades superiores, aproximadamente a unos cinco centímetros del codo, para el diseño e implementación de un algoritmo de control del robot Mitsubishi RV-2F que se encuentra en el laboratorio de Manufactura Integrada por Computadora de la Universidad Técnica del Norte. El control consiste en la generación de trayectorias para el robot, el cual realizará diferentes actividades dependiendo de las señales que reciba y según los gestos preestablecidos. Se utilizará un dispositivo con sensores superficiales, específicamente un dispositivo Myo Armband y una tarjeta comercial Arduino para la adquisición y procesamiento de las señales, también se utilizarán librerías para la comunicación entre los distintos softwares, un optoacoplador para elevar la señal enviada por el Arduino de 3,3VDC a 24VDC

CAPÍTULO I

MARCO TEÓRICO

1.1 Señales Bioeléctricas musculares

Las señales bioeléctricas son la manifestación de la activación neuromuscular asociada con una contracción muscular. Las señales representan las corrientes generadas por el flujo iónico a través de la membrana de las fibras musculares que se propaga a través de los tejidos que intervienen. Las fibras musculares están inervadas en grupos llamados unidades motoras, que cuando se activan generan un potencial de acción. La activación del sistema nervioso central se repite continuamente mientras se requiera que el músculo genere fuerza. Esta activación continua genera trenes de potencial de acción de la unidad motora. Estos trenes de las unidades motoras activas simultáneamente se superponen para formar la señal, a medida que aumenta la excitación del sistema nervioso central, para generar una mayor fuerza en el músculo, se activa (o recluta) un mayor número de unidades motoras y aumenta la velocidad de disparo de todas las unidades motoras activas [8]. En la Figura 1 se puede apreciar las partes que constituyen a dicha unidad motora.

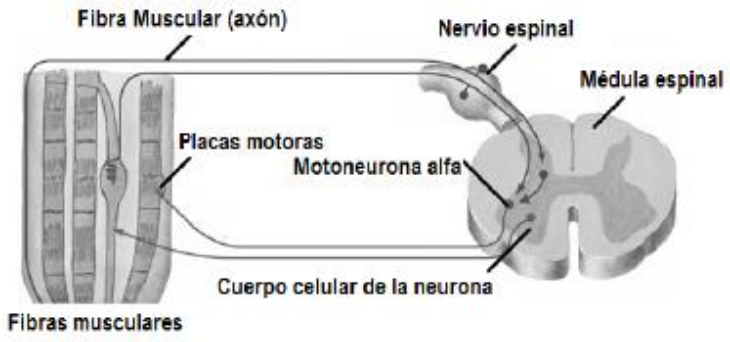

Figura 1: Elementos que constituyen la Unidad Motora (UM)

Fuente: [9]

1.2 Electromiografía

La electromiografía es la disciplina que se ocupa de la detección, análisis y uso de potenciales eléctricos creados por la contracción de los músculos, cuando estos se activan eléctrica o neurológicamente. Dichos potenciales son conocidos como señales electromiográficas (EMG) aunque un número creciente de profesionales optan por usar el término "señal mioeléctrica" (EM). Un ejemplo de una señal EMG se muestra en la Figura 2 que empieza con una amplitud baja pero que a medida que ocurre un aumento en la fuerza de contracción muscular, más fibras se activan y aumenta la velocidad de disparo de estas, lo que produce un aumento en la amplitud [8].

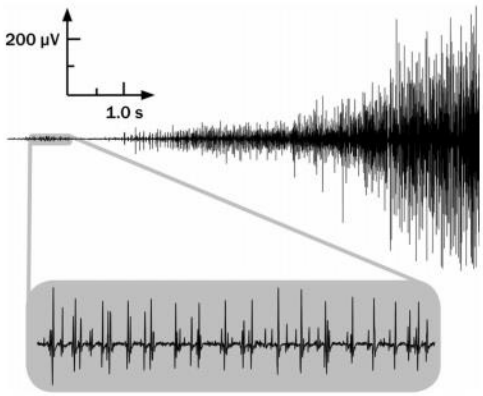

Figura 2: Ejemplo de Señal EMG registrada con electrodos de superficie situados sobre la piel del músculo interóseo dorsal en la mano.

Fuente: [8]

Estas señales, cuya amplitud pico a pico puede variar desde 0 a 10 mV, se obtienen a partir de electrodos y según el tipo que se utilice se consideran dos tipos de electromiografía: de aguja o intramuscular y la de superficie.

La electromiografía intramuscular o de aguja se realiza con el fin de estudiar la patología o fisiología de las unidades motrices, se pueden estudiar las lesiones, como la pérdida de suministro al músculo, la capacidad del nervio para regenerarse, entre otras patologías. Mientras

que la electromiografía de superficie (SEMG) se enfoca a estudios relacionados con el comportamiento muscular, los patrones de actividad temporal y la fatiga muscular. Encuentran aplicaciones importantes en el deporte, la rehabilitación y la medicina laboral donde las técnicas de aguja no son aceptables o cuando las evaluaciones tienen que repetirse con frecuencia [10].

En la Figura 3 se muestran ejemplos de electrodos de agua intramusculares y de superficie utilizados para la detección de señales EMG.

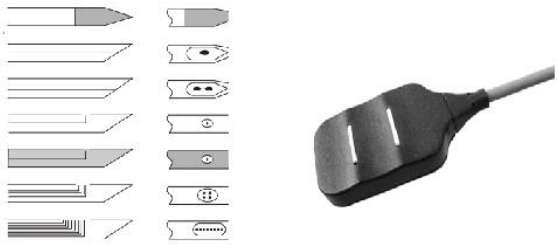

Figura 3: Ejemplos de electrodos de aguja y superficie.

Fuente: [8]

1.2.1 Norma de adquisición de señales electromiográficas.

Debido a los avances generados en la detección y procesamiento de señales electromiográficas, en la última década se empezaron a realizar grandes esfuerzos para la normalización de las diferentes metodologías elaboradas por los distintos laboratorios de investigación clínica. Esto con el fin de que los datos presentados en dichas investigaciones puedan ser comparables entre sí y para crear un gran repositorio común de información sobre la correcta adquisición y uso de las señales EMG en sus distintos campos de aplicación. Por esta razón aparece en 1996 una asociación denominada “Electromiografía de superficie no invasiva para la evaluación de los músculos” o SENIAM por sus siglas en inglés. El principal objetivo de SENIAM es crear un consenso sobre los elementos involucrados en la detección de las señales EMG (sensores y su correcto posicionamiento), el intercambio de datos, procesamiento de la señal y modelado [11]. La conclusión del estudio realizado por SENIAM

es un estándar de características que deben poseer los electrodos, amplificadores, el muestreo y la conversión A/D, presentado en 6 volúmenes y 7 revistas. Un resumen de las características es presentado a continuación:

- Electrodo (Montaje Bipolar): Diámetro < 10 mm; Distancia entre electrodos < 20 mm ó < 1/4 de la longitud total del músculo; Localización del electrodo en cualquier distancia más pequeña entre la zona distal de la inervación y el tendón distal; Localización del electrodo de referencia entre la zona más próxima a la inervación y el tendón proximal.
- Amplificador: Filtro pasa alto para análisis espectral EMG < 10 Hz, para análisis de movimiento se utiliza una frecuencia de 20 Hz; Filtro pasa bajo aproximadamente 500 Hz; Nivel de ruido a la entrada de voltaje < 1 uVrms en el ancho de banda de 10-500 Hz; Nivel de ruido a la entrada de corriente < 10 pArms en el ancho de banda de 10-500 Hz.
- Muestreo y conversión A/D: Frecuencia de muestreo > 1000 muestras; n Bit para A/D 12 para amplificador con ganancia variable y 16 para ganancia fija.

1.2.2 Factores que afectan la información en las señales electromiográficas.

Las señales EMG son señales complicadas que se ven afectadas por las propiedades fisiológicas de los músculos, así también como por las propiedades de los instrumentos que se utilizan para su detección y medición. En la tabla 1 se muestran los principales factores que afectan estas señales y la influencia que tienen sobre ellas.

Tabla 1:Factores que influyen en el contenido de la información de la señal de la electromiografía.

Factor	Influencia
Neuro activación	- Velocidad y sincronización de disparo de los potenciales de acción de la unidad motora. - Número de unidades motoras reclutadas.
Fisiología de fibra muscular	- Velocidad de conducción de las fibras musculares.
Anatomía muscular	- Diámetro, distribución y orientación de las fibras musculares.
Tamaño y orientación del electrodo	- Número de fibras musculares dentro del área de detección del electrodo. - Número de unidades motoras dentro del área de recolección de la superficie de detección del electrodo relativo a las fibras musculares.
Interfaz electrodo – electrolito	- Materiales y preparación del electrodo. - Ubicación del electrodo. - Impedancia con frecuencia en aumento.
Configuración de electrodos bipolares	- Efecto de la distancia entre los electrodos de detección y de ancho de banda (filtro de paso de banda). - La orientación de los electrodos de detección con respecto al eje de las fibras musculares.

Fuente: [12]

1.3 Extremidad superior

El proyecto se basa en el estudio de las señales mioeléctricas de las extremidades superiores, específicamente las señales recibidas del antebrazo a aproximadamente cinco centímetros desde

el codo, debido a que los dispositivos de detección de señales mioeléctricas comerciales frecuentemente se utilizan en dicha posición. Estos músculos cambian su longitud al desarrollar tensiones en las fibras musculares lo que genera una fuerza de contracción que se ve graduada por el número y la frecuencia de axones que se estimulan. En la Figura 4 se observan los músculos y nervios del miembro superior.

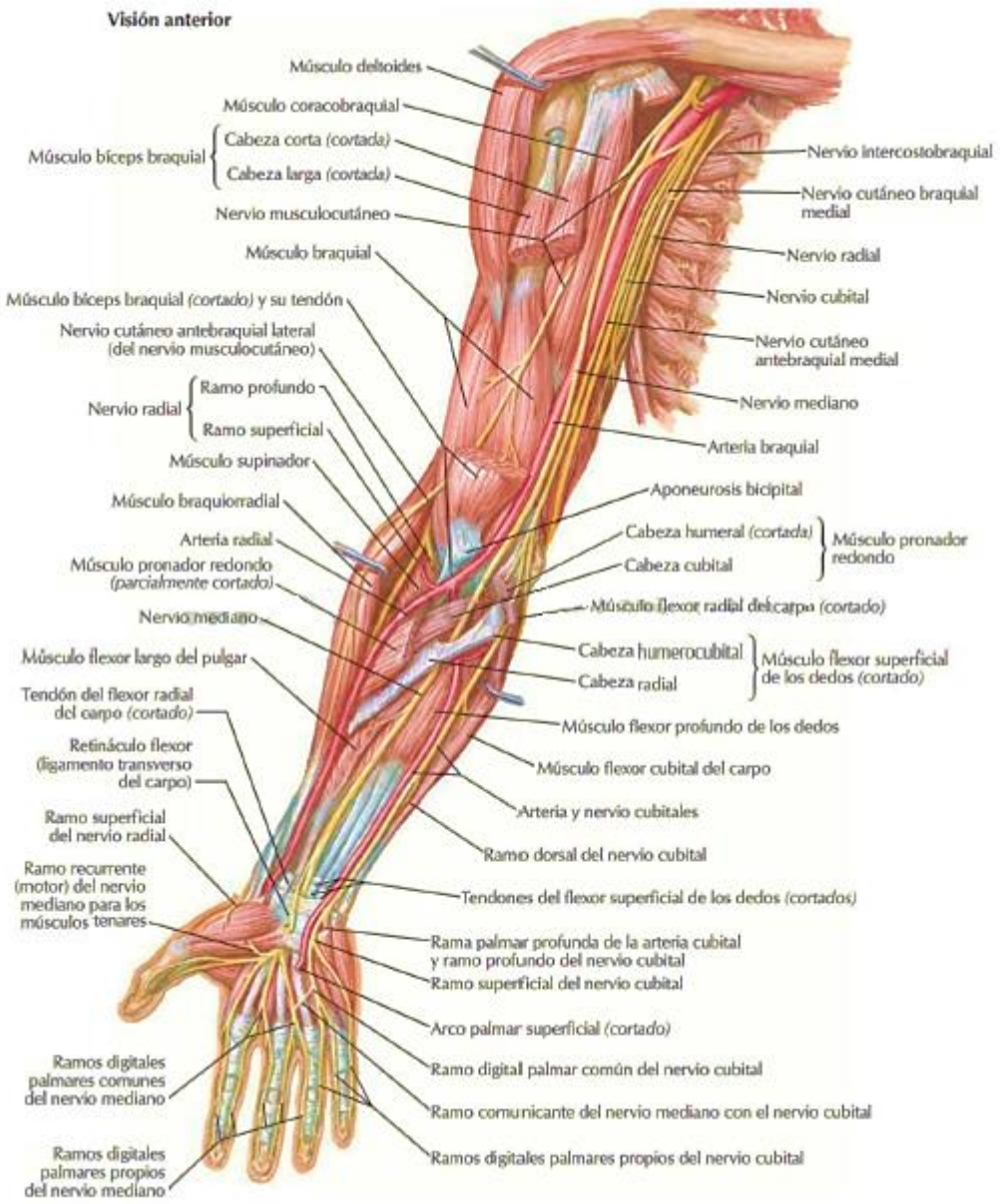

Figura 4: Nervios y músculos de la extremidad superior.

Fuente: [13]

1.4 Campos de aplicación de señales EMG

1.4.1 Medicina

Utilizado sobre todo en aplicaciones de rehabilitación, consisten en sistemas de adquisición y procesamiento de señales mioeléctricas conformados por un bloque de sensores que realizan el acondicionamiento y adquisición de la señal, una tarjeta para el procesamiento de la señal y actuadores para realizar el movimiento de rehabilitación según el caso.

1.4.2 Biónica

La convergencia de técnicas entre la ingeniería, robótica y biociencia constituyen las bases para una nueva disciplina denominada biónica, cuyo objetivo principal es la creación y desarrollo de prótesis y procedimientos tecnológicos que ayudan a recuperar las funciones perdidas de las partes del cuerpo que sustituyen. Estas pueden ser controladas por señales electromiográficas por medio de electrodos de la misma forma que en las aplicaciones mencionadas anteriormente. Un ejemplo de aplicación en este campo es la prótesis inteligente C-Leg® (Computerized Leg) que se muestra en la Figura 5. Cuenta con un control hidráulico de la fase de apoyo, es capaz de asimilar los datos obtenidos en cada una de las fases de la marcha y puede ser configurada a través de un software de soporte en una computadora personal [14].

Figura 5: Prótesis de rodilla C-Leg.

Fuente: [14]

1.4.3 Aeroespacial

En el presente es común encontrar que la velocidad y posición de los drones sea controlada mediante los gestos del brazo del usuario. Este tipo de control otorga una mejor maniobrabilidad a estos dispositivos aeroespaciales. Como principal ejemplo tenemos el presentado en la Figura 6, el dron Parrot AR drone 2.0. Posee una cámara HD 720p, estabilizador de vuelo y vuelta a casa, para usarlo solo se necesita un Myo Armband, una computadora personal con Windows y el software del fabricante (AutoFlight). Se utilizan los gestos preestablecidos en el dispositivo Myo para poder controlar el dron.

Figura 6: Parrot AR Drone 2.0

Fuente: [15]

1.4.4 Industria

En este campo de aplicación se destacan el control de actuadores y robots manipuladores mediante señales mioeléctricas que permiten que se reduzca la posibilidad de accidentes ya que el control se lo realiza a distancia evitando la intervención directa del operador con el área de producción.

1.5 Brazalete de gestos GForce-100

Es un dispositivo comercial desarrollado por la empresa OYMotion. Utiliza electrodos secos diferenciales EMG de grado médico de 8 canales, sensor de movimiento de 9 ejes, adquisición

en tiempo real de Quaternión o datos de movimiento sin procesar. Cuenta con 6 gestos preestablecidos detectados desde el antebrazo del usuario, compatibilidad con Arduino, STM32 MCU a través de gForce-Joint Bluetooth al módulo UART y respuesta de vibración. El dispositivo se muestra en la Figura 7.

Figura 7: Brazalete GForce-100 de OYMotion.

Fuente: [16]

1.6 Myo Armband

Es un dispositivo comercial desarrollado por la empresa Thalmic Labs. Es un Wearable que permite utilizar los gestos y movimientos del antebrazo del usuario para interactuar con computadoras, celulares y otros dispositivos sin necesidad de hacer contacto físico. El dispositivo Myo Armband es un brazalete compuesto por 8 electrodos que se distribuyen equidistantemente alrededor del antebrazo del usuario, capaces de extraer datos EMG a una velocidad de muestreo de 200Hz. Utiliza bluetooth como medio de transmisión de datos hacia la computadora. Posee un software propietario para la detección de gestos y varias aplicaciones prácticas disponibles en su tienda online [17]. En la Figura 8 se muestra el dispositivo.

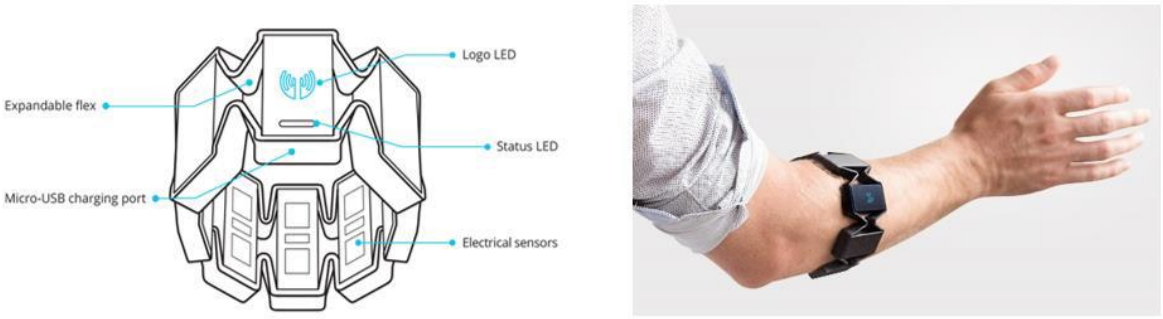

Figura 8: Izquierda: Estructura del dispositivo Myo Armband. Derecha: Dispositivo siendo correctamente vestido por un usuario.

Fuente: Propia

1.6.1 Caracterización de las señales EMG

El dispositivo Myo Armband captura las señales electromiográficas a una frecuencia de 200 Hz la cual es una medida estándar para los sensores que utiliza el brazalete. La amplitud pico - pico de estas señales se encuentra dentro de unos pocos milivolts, aproximadamente 0,01 mV con una frecuencia promedio que está entre 1 – 300 Hz. La Figura 9 muestra un ejemplo de la amplitud de la señal versus el espectro de esta, reflejando la mayor cantidad de energía entre 50 – 150 Hz.

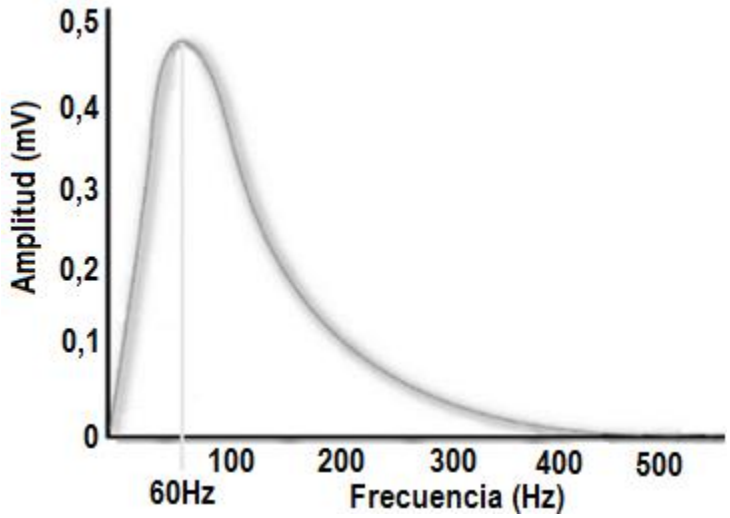

Figura 9: Representación esquemática del espectro de una señal típica de SEMG.

Fuente: [6]

Los sensores del brazalete cuentan con clasificadores en sus salidas las cuales son procesadas por algoritmos internos del dispositivo, lo que se traduce en que el sensor no muestra la señal en bruto (RAW) si no que la convierte en una señal normalizada, esto ocurre debido a que entre dos personas realizando el mismo gesto existe una diferencia notable, como se muestra en la Figura 10.

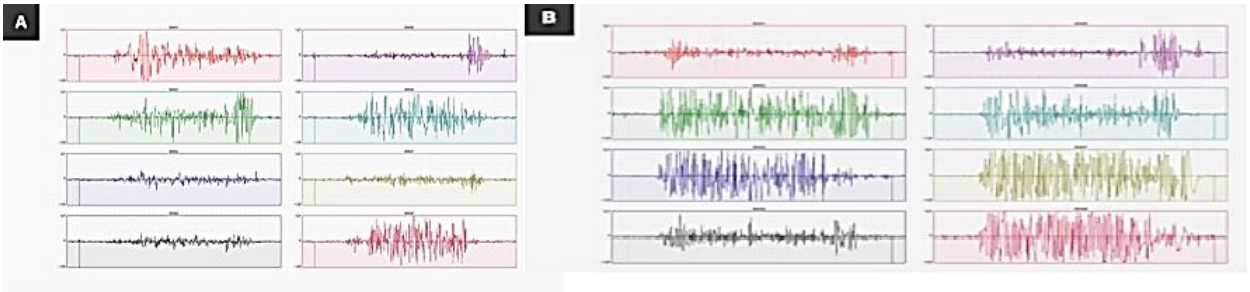

Figura 10: Señales electromiográficas de distintos sujetos realizando el mismo gesto.

Fuente: [18]

1.6.2 Especificaciones técnicas

Las especificaciones se obtienen de la página de soporte de Myo [19].

- Dimensiones: Expandible de 19 a 34 centímetros de diámetro para el antebrazo.
- Peso: 93 gramos.
- Grosor: 1.14 centímetros.
- Dispositivos compatibles: Windows, Mac OS X y macOS, IOS y Android.
- Hardware: 8 Sensores EMG de acero inoxidable de grado médico, estos electrodos son de tipo multi-electrodo superficial, IMU de 9 ejes altamente sensible con giroscopio de tres ejes, acelerómetro de tres ejes, magnetómetro de tres ejes.
- LEDs: Indicador Led duales.
- Procesador: ARM Cortex M4.

- Comunicación: Bluetooth 4.0 de baja energía.
- Batería: Batería de iones de litio recargable incorporada, de duración de un día.

En la Figura 11 se muestran los gestos preestablecidos que puede detectar Myo Armband.

Figura 11: Gestos reconocidos por Myo Armband.

Fuente: [19]

1.6.2 Myo Connect

Es el software privativo de Thalmics Labs encargado de controlar el dispositivo Myo Armband. Establece la conexión vía bluetooth a través de un USB Dongle que debe conectarse al PC, este software incluye una guía de introducción para la sincronización del Myo con la computadora y el entrenamiento de los diferentes gestos. Este software es el único que puede encender, apagar, conectar y desconectar el dispositivo por lo que siempre será necesario la sincronización del dispositivo con el software. Además, esta aplicación permite configurar el dispositivo y gestionar las aplicaciones descargadas de Myo Market (tienda online) [17].

1.6.3 Myo SDK.

Es el Software Development Kit oficial creado por Talmic Labs, disponible en la página de desarrolladores de Myo Armband. Este SDK permite al desarrollador acceder a dos tipos de datos:

• Datos Espaciales (Spatial Data): proporciona información acerca de la orientación y movimiento del brazo del usuario.

• Datos de Gestos (Gestural Data): informa que está realizando el usuario con su mano, detecta cuando un usuario realiza un gesto. El núcleo de Myo SDK es la librería “Libmyo”. Esta librería permite a las aplicaciones interactuar con el Myo Armband. Toda la funcionalidad de la librería es expuesta en una API desarrollada en C. Generalmente, las aplicaciones no interactúan directamente con la librería “LibMyo” sino que utilizan un “Language Binding” que corresponde al lenguaje de programación en el cual se va a desarrollar la aplicación. Oficialmente Myo Developers ofrece el conector para el lenguaje C++. Sin embargo, también existen en internet conectores para otros lenguajes como C# y Matlab. [17].

1.7 Tarjetas de adquisición y procesamiento de señales

1.7.1 Arduino

Arduino es una plataforma de desarrollo basado en hardware libre que contiene un microcontrolador reprogramable y una serie de pines hembra que permiten la conexión entre la placa y los diferentes sensores y actuadores [20]. El desarrollo de hardware libre permite la reutilización de recursos y ahorro de costes, además de la realización de proyectos de robótica, internet de las cosas, impresión 3D, entre otras. Cuenta con una comunidad llena de tutoriales, cursos, herramientas, librerías y proyectos realizados por terceros los cuales ayudan a que el aprendizaje sea de una forma más sencilla e intuitiva.

Arduino cuenta con una serie de placas entre ellas Arduino Uno, Nano y Mega. La placa más utilizada es el Arduino Uno, mostrada en la figura 12, que cuenta con un microprocesador Atmega328, 14 pines de entradas/salidas digitales, 6 pines para entradas analógicas, Plug USB para la comunicación con el entorno de desarrollo integrado de Arduino, conector de fuente externa, pines de voltaje de 3.3V y 5V y tierra [21].

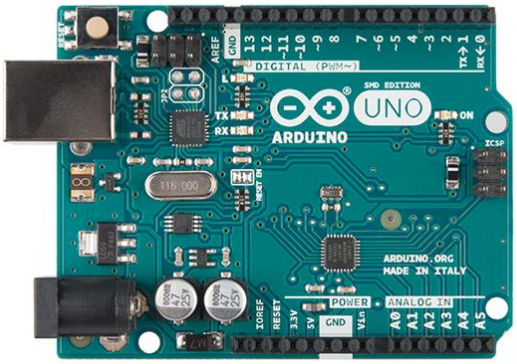

Figura 12: Arduino Uno.

Fuente: [21]

1.7.2 Raspberry PI

Raspberry PI es un ordenador de placa simple desarrollado en Reino Unido por Raspberry Pi Foundation que puede conectarse con diferentes periféricos como teclado, mouse, pantalla, entre otros. Es comúnmente utilizado para aprender a programar como en un ordenador personal. Al igual que Arduino, Raspberry cuenta con una serie de placas con características diferentes según el modelo, pero a nivel general tiene de 26 a 40 pines de entradas/salidas de propósito general, incluye un UART, un bus I2C, un bus SPI, audio I2S, salidas de 3V y 5V y su respectiva tierra, puerto de cámara CSI-2 y puerto de pantalla DSI [22].

La placa más común se muestra en la Figura 13, cuenta con un procesador con chip de 32 bits, sistema de 700MHz. El modelo B tiene una RAM de 512 MB y el modelo A tiene 256 de memoria RAM.

Figura 13: Raspberry PI 2 modelo B.

Fuente: [22]

1.7.3 MyRIO

MyRIO es una placa de procesamiento que cuenta con un procesador ARM COrtex-A9, para aplicaciones de sistemas embebidos, control, sistemas embebidos y mecatrónicos. Proporciona salidas y entradas analógicas y digitales, acelerómetro, Wireless, audio, salida de potencia y puerto USB para comunicación en un dispositivo compacto integrado. Además de presentar 3 canales A, B y C, de los cuales los dos primeros son iguales y poseen salidas y entradas digitales y analógicas, pines PWM, comunicación UART, I2C y SPI, fuentes de salida de 3V y 5V. El canal C tiene menos entradas y sus fuentes son de 15V y 5V [23]. En la Figura 14 se muestra el diagrama de bloques de la tarjeta.

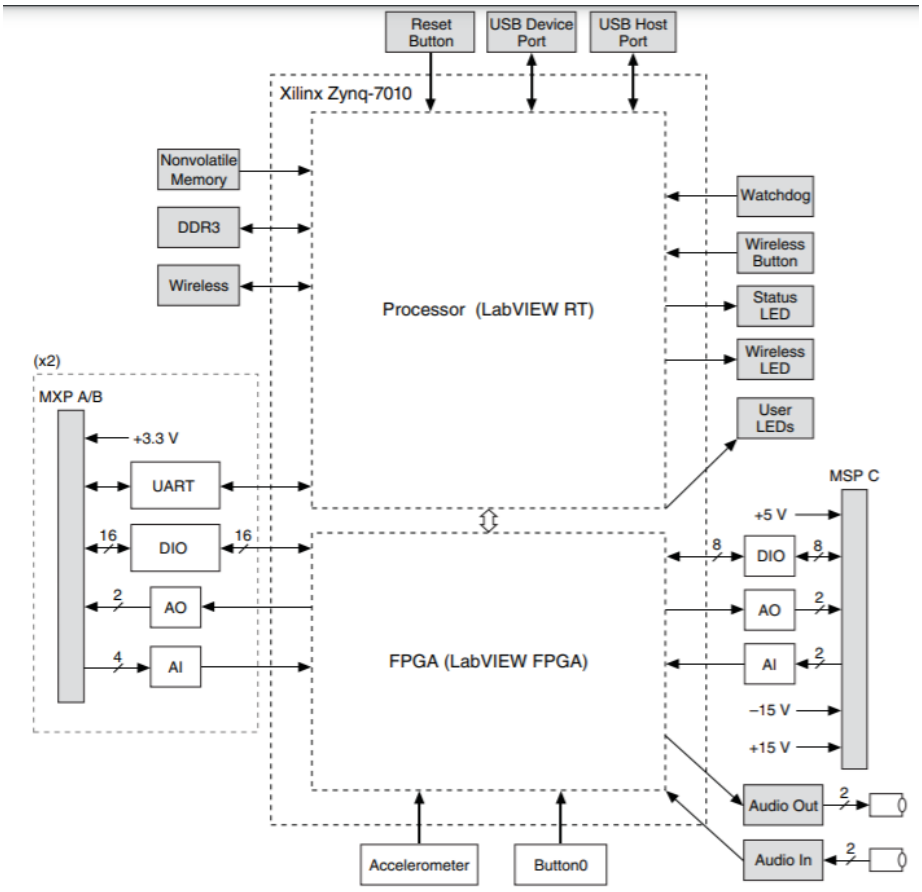

Figura 14: Bloque de diagrama del hardware de MyRIO.

Fuente: [23]

1.8 Software para el procesamiento de la señal

El software utilizado en el presente trabajo es Matlab. Es una plataforma de programación enfocada en la matemática computacional que permite realizar: análisis de datos, desarrollo de algoritmos y creación de modelos. Esta herramienta es ampliamente usada por ingenieros y científicos para aplicaciones de Deep Learning, Machine Learning, procesamiento y comunicaciones de señales, procesamiento de imágenes y video, sistemas de control, finanzas, biología computacional, etc. Se utilizaron las versiones Matlab R2015a y R2017b [24].

1.8.1 Application compiler

Matlab Compiler permite compartir programas desarrollados en Matlab como aplicaciones independientes también llamadas Standalone Applications. Todas las aplicaciones creadas con Matlab Compiler utilizan el Matlab Runtime, el cual es un conjunto de librerías que permite la ejecución de aplicaciones compiladas sin la necesidad de tener instalado Matlab. El Matlab Runtime puede ser empaquetado e instalado cuando se crea la aplicación [25].

1.8.2 Library compiler

Extiende la funcionalidad del Matlab Compiler y permite convertir código M (Lenguaje Matlab) en librerías para C/C++, Java y .NET. Estos componentes pueden ser integrados y desplegados en aplicaciones web o de escritorio libre de impuestos. Esta funcionalidad, al ser una extensión del Matlab Compiler también hace uso del Matlab Runtime [26].

1.8.3 Myo MEX

Es un SDK desarrollado para Matlab que permite a los usuarios obtener datos de uno o dos dispositivos Myo Armband. Es posible obtener señales EMGs de los 8 sensores electromiográficos en forma de Stream y los datos IMU (Inertial Measurement Unit) que incluyen: Cuaternios (Orientación), tres ejes del giroscopio (Velocidad Angular) y tres ejes del acelerómetro (Aceleración Lineal). Este SDK se encuentra disponible en GitHub [27].

1.9 ROS (Robot Operating System)

ROS es ampliamente utilizado en robótica ya que su objetivo es crear un programa para trabajar con varios manipuladores a la vez con solo unos pequeños cambios en el código, con esto es posible crear funciones compartidas que pueden ser utilizadas por otros robots sin emplear demasiado trabajo. Cada vez se hace más común el adaptar sensores y actuadores para utilizarlos con esta plataforma. ROS está basado en una arquitectura gráfica donde el procesamiento se realiza en nodos que pueden recibir y enviar datos de sensores, control, estado

y otros mensajes a través de un entorno de programación C++/Python para controlar el hardware, creando una red donde los procesos están interconectados [28]. Los componentes básicos que conforman ROS se muestran en la Figura 15.

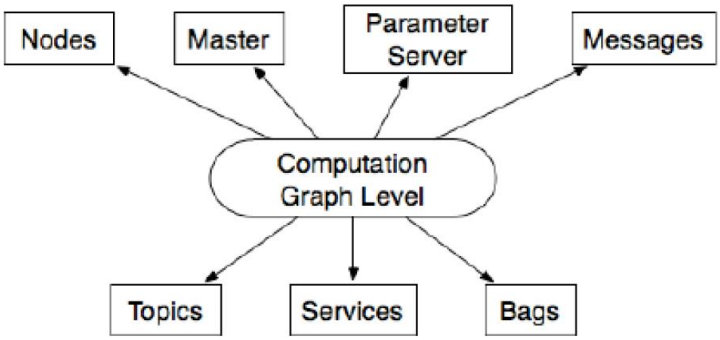

Figura 15: Componentes básicos de ROS.

Fuente: [28]

1.10 Robot Mitsubishi RV-2F.

Es un brazo robótico industrial de la serie RV-2F de 6 grados de libertad adaptado a la fabricación de celdas con alta velocidad, rendimiento de alta precisión y combina tecnología inteligente. Tiene una fácil conectividad con los PLC y equipos FA de Mitsubishi, lo que permite ser utilizado en sistemas industriales de automatización. Su capacidad de carga es de 2 kg con un radio de trabajo de 504 mm. En la tabla 2 se muestran las principales características del robot Mitsubishi RV-2F.

Tabla 2: Características principales del Robot Mitsubishi RV-2F.

Características	Especificaciones
Grados de libertad	6
Montaje	Pared, suele o techo
Accionamiento	Servo motor AC (J2, J3, J5)
Detección de posición	Encoder de valor absoluto

Radio de alcance	504 mm
Rango de movimiento	J1 480° (-240+240) J2 240° (-120+120) J3 160° (0 - + 160) J4 400° (-200+200) J5 240° (-120+120) J6 720° (-360+360)
Velocidad de movimiento	J1 300°/s J2 150°/s J3 300°/s J4 450°/s J5 450°/s J6 720°/s
Velocidad máxima resultante	4955 mm/s
Peso	19 Kg
Momento Nominal	J4 4.17 Nm J5 4.17 Nm J6 2.45 Nm
Momento de inercia nominal	J4 0.18 Kgm J5 0.18 Kgm J6 0.04 Kgm
Cableado de Herramienta	4 entradas/ 4 salidas
Presión neumática de alimentación	0.5 ± 10%
Controlador del Robot	CR750-D

Fuente: [29]

1.11 Controlador CR750-D

El controlador del robot permite el correcto funcionamiento de este, también limita la capacidad de memoria, entradas y salidas e interfaz. En la tabla 3 se muestran las principales características del controlador CR750-D.

Tabla 3: Características principales del controlador CR750-D.

Modelo		**CR750-D**
Método de control de ruta		Control PTP y control CP
Número de ejes del controlador		6
Lenguaje del robot		Melfa-Basic
Capacidad de memoria		39000 de puntos aprendidos 78000 número de pasos 512 Número de programas
Entradas/salidas externas	**I/O generales**	0 entradas/ 0 salidas (256/256)
	I/O dedicadas	Asignadas para I/O generales
	Apertura y cierre de la mano	8 entradas/ 8 salidas
	Entrada paro de emergencia	1 (redundante)
	Entrada interruptor de puerta	1 (redundante)
	Habilitar entrada de dispositivo	1 (redundante)
	Salida paro de emergencia	1 (redundante)
	Salida de modo	1 (redundante)
	Salida de error del robot	1 (redundante)
	Ejes adicionales	1 (redundante)
Interfaz	**Rs-422**	1 (Teaching Pendant)
	Ethernet	1

USB	Solo funciones dispositivos 2.0
Slot dedicado a la mano	1 (Dedicado para interfaz mano)
Ejes adicionales	1 (SSCNET III)
Slot de extensión	2
Entrada encoder	2

Temperatura ambiente	0 – 40 °C
Humedad relativa	45 – 85 %RH
Alimentación	200 – 230 V (voltaje de entrada) 0.5 KVA (capacidad de potencia)
Dimensiones externas	430*425*174 mm
Peso	18 Kg

Fuente: [29]

Entre los dispositivos que se pueden conectar al controlador CR750-D tenemos los siguientes:

- Unidad Paralela I/O: Usada para expandir a 32 entradas/salidas externas del controlador.
- Interfaz CC-Link: Agrega datos de bits al controlador y permite la transmisión cíclica de datos.
- Interfaz paralelas I/O: Se utiliza para expandir a 32 entradas y salidas externas, que también es posible el uso combinado con la unidad de entrada y salida paralela.

1.12 Diseño Mecatrónico

Para el diseño de un sistema de control de un brazo robótico se optará por un diseño basado en la norma VDI 2206 que enfoca todos los procesos necesarios para obtener un producto final, este diseño de sistema básico se muestra en la Figura 16.

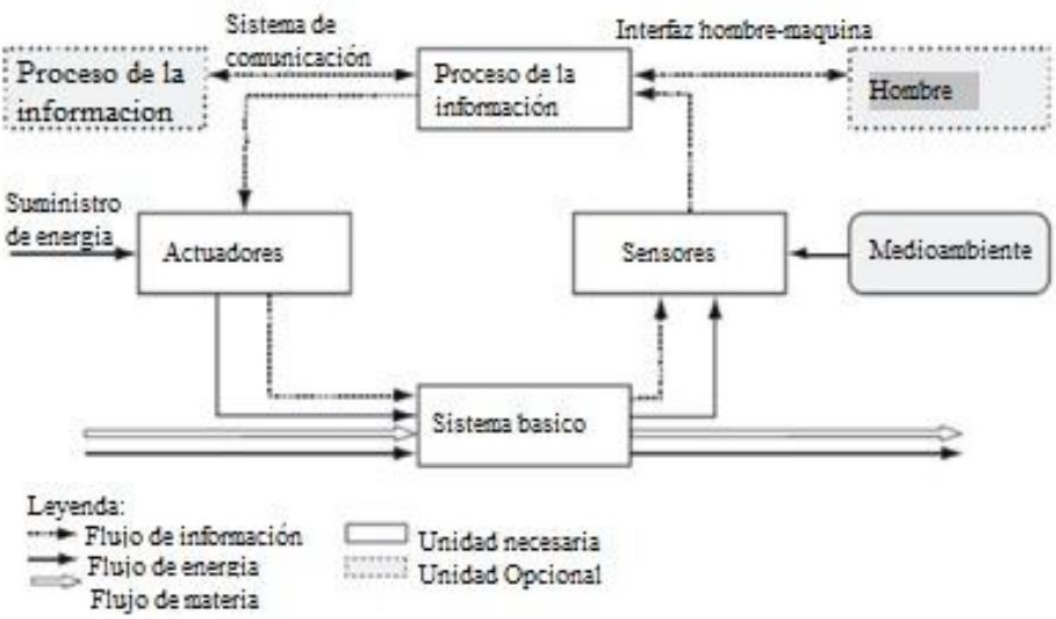

Figura 16: Diseño Mecatrónico basado en la norma VDI 2206.

Fuente: [30]

Los elementos que constarán en el sistema básico para el control de un robot manipulador mediante señales mioeléctricas son los siguientes:

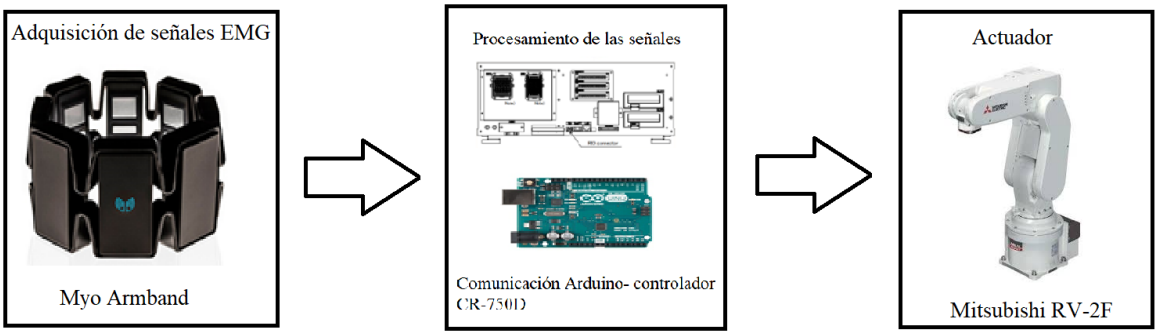

Figura 17: Elementos del sistema básico mecatrónico.

CAPÍTULO II

METODOLOGÍA

2.1 Diseño conceptual

2.1.1 Estructura funcional

La Figura 18 muestra un mapa conceptual con los diferentes procesos a realizar para un sistema de control del robot Mitsubishi RV-2F mediante señales mioeléctricas.

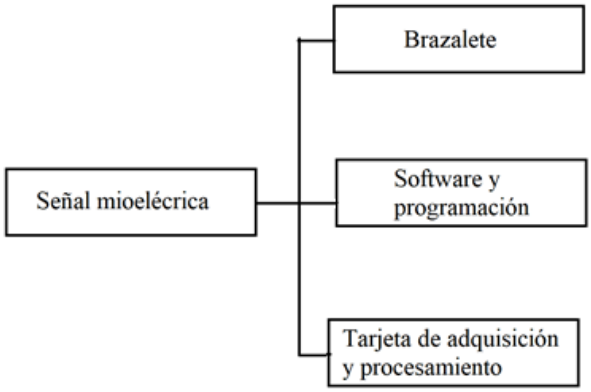

Figura 18:Diagrama funcional para el control del robot Mitsubishi.

En la Figura 19 se puede apreciar la función global del control del brazo robótico mediante el uso de la herramienta caja negra [30]. Donde se puede apreciar la relación entre los flujos de entrada y salida presentes en el proceso.

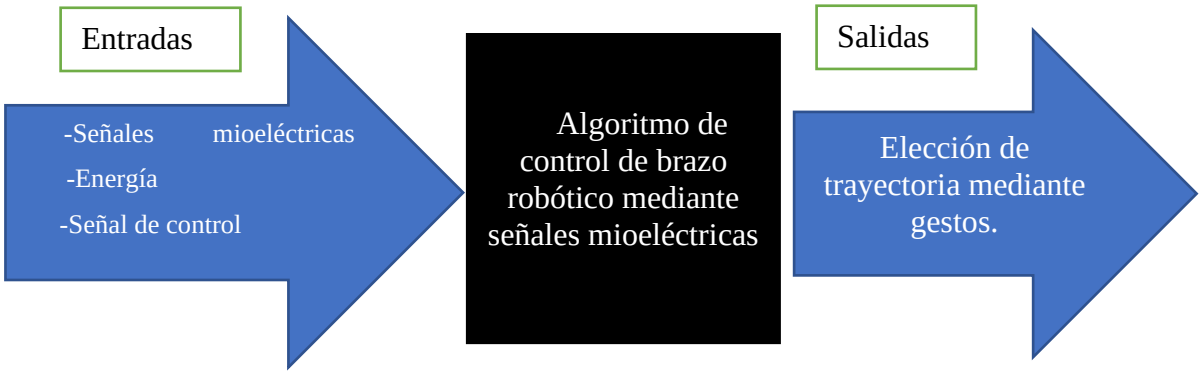

Figura 19: Caja negra del control de un brazo robótico mediante señales EMG.

La Figura 20 muestra el proceso completo partiendo de la adquisición de señales hasta la ejecución del actuador y el desglosado de las subfunciones del control del brazo robótico, de esta manera se puede tener una mejor visión de las posibles alternativas de solución. El módulo naranja representa la adquisición de la señal, el módulo verde el procesamiento y la activación de entradas y el módulo azul el control de las trayectorias.

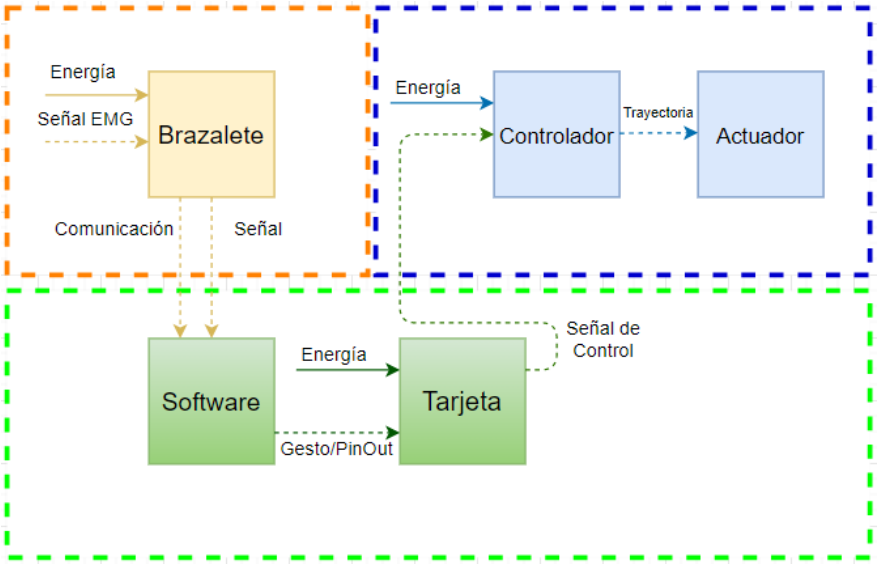

Figura 20:Estructura funcional del control del brazo robótico.

2.1.2 Matriz Morfológica

La matriz morfológica es una técnica que permite generar una tormenta de ideas, descompone el proyecto en sus elementos primarios para así producir una matriz que permita relacionar todas sus partes con el propósito de obtener diferentes soluciones, de las cuales se escogerá la mejor basado en el método de selección por criterios ponderados.

Mediante esta técnica se obtiene la tabla 4, donde se muestra la descomposición del proyecto en diferentes dispositivos básicos para un sistema de control de un brazo robótico mediante señales mioeléctricas. Estos dispositivos básicos son:

- Brazalete: Dispositivos comerciales para la adquisición de señales mioeléctricas.

- Tarjeta de adquisición: Para el procesamiento de la señal y el envío de señales al PLC del robot.
- Software: Diferentes métodos de programación y comunicación.

Tabla 4: Matriz Morfológica

Brazalete			
	GForce-100	Myo Armband	
Tarjeta de Adquisición			
	Raspberry PI	Arduino Uno	NI MyRIO
Software			
	Matlab	ROS	Labview

2.1.3 Método de evaluación

Los criterios para tomar en cuenta en la elección de las alternativas de solución en el apartado de método de criterios ponderados son costo, conectividad, dificultad de programación y compatibilidad entre dispositivos.

2.1.3.1 Alternativas de solución

Solución amarilla: La etapa de adquisición de las señales las realiza el dispositivo de gestos GForce-100, este se conecta la tarjeta Raspberry PI en la que se procede a realizar la programación en Linux mediante ROS.

Solución azul: El dispositivo GForce-100 se conecta a la tarjeta Arduino la que se procede a conectar con Matlab mediante el uso de librerías.

Solución verde: El dispositivo Myo Armband se encarga de la adquisición de las señales y se conecta a un Arduino que a su vez se conecta con Matlab para realizar el procesamiento y envío de señales al robot.

Solución roja: El dispositivo Myo Armband se conecta a la tarjeta de adquisición NI MyRIO y se procede a realizar el procesamiento de las señales y la programación gráfica en LabVIEW.

2.1.3.2 Selección de solución

Se empleará el método de criterio ponderados que permite comparar distintas alternativas de solución y obtener resultados globales significativos realizando un análisis cuantitativo mediante los criterios establecidos. En la tabla 5 se muestra el peso específico de cada criterio.

Conectividad > Costo > Compatibilidad > Programación

Tabla 5: Evaluación de criterios.

Criterio	Costo	Conectividad	Programación	Compatibilidad	Σ+1	Ponderación
Costo		0,5	1	1	3,5	0,292
Conectividad	1		1	1	4	0,333
Programación	0	0,5		0,5	2	0,167
Compatibilidad	0,5	0,5	0,5		2,5	0,208
				Suma	12	1

En las tablas 6, 7, 8 y 9 se realizan las evaluaciones de los pesos de los criterios: costo, conectividad, facilidad de operación y compatibilidad. En la tabla 10 se muestra la conclusión del método ordinal de criterios ponderados.

Solución Verde = Solución Azul > Solución Amarilla > Solución Roja

Tabla 6: Evaluación de alternativas con respecto al criterio conectividad.

Conectividad	Solución Amarilla	Solución Azul	Solución Verde	Solución Roja	Σ+1	Ponderación
Solución Amarilla		0,5	0	1	2,5	0,238
Solución Azul	0,5		0,5	1	3	0,286
Solución Verde	0,5	0,5		1	3	0,286
Solución Roja	0,5	0,5	0		2	0,190
				Suma	10,5	1

Solución Verde > Solución Azul > Solución Amarilla > Solución Roja

Tabla 7: Evaluación de alternativas con respecto al criterio costo.

Costo	Solución Amarilla	Solución Azul	Solución Verde	Solución Roja	Σ+1	Ponderación
Solución Amarilla		0	0	1	2	0,200
Solución Azul	1		0	0,5	2,5	0,250
Solución Verde	1	1		1	4	0,400
Solución Roja	0	0,5	0		1,5	0,150
				Suma	10	1

Solución Verde = Solución Azul > Solución Amarilla = Solución Roja

Tabla 8: Evaluación de alternativas con respecto al criterio compatibilidad.

Compatibilidad	Solución Amarilla	Solución Azul	Solución Verde	Solución Roja	Σ+1	Ponderación
Solución Amarilla		0	0	0,5	1,5	0,150
Solución Azul	1		0,5	1	3,5	0,350
Solución Verde	1	0,5		1	3,5	0,350
Solución Roja	0,5	0	0		1,5	0,150
				Suma	10	1

Solución Verde > Solución Azul > Solución Amarilla = Solución Roja

Tabla 9: Evaluación de alternativas con respecto al criterio dificultad de programación.

Programación	Solución Amarilla	Solución Azul	Solución Verde	Solución Roja	Σ+1	Ponderación
Solución Amarilla		0	0	0,5	1,5	0,150
Solución Azul	1		0	1	3	0,300
Solución Verde	1	1		1	4	0,400
Solución Roja	0,5	0	0		1,5	0,150
				Suma	10	1

Tabla 10: Conclusión de criterios ponderados.

Criterio	Conectividad	Costo	Compatibilidad	Programación	Σ	Prioridad
Amarilla	0,238*0,292	0,200*0,333	0,150*0.167	0,150*0,208	0,19	**3**
Azul	0,286*0,292	0,250*0,333	0,350*0,167	0,300*0,208	0,29	**2**
Verde	0,286*0,292	0,400*0,333	0,350*0,167	0,400*0,208	0,36	**1**
Roja	0,190*0,292	0,150*0,333	0,150*0,167	0,150*0,208	0,16	**4**

La solución óptima es la solución verde, incluye un dispositivo Myo, programación M en Matlab y una tarjeta Arduino Uno. En el siguiente flujograma presentado en la Figura 21 se muestra cómo actúa el sistema de control mediante señales mioeléctricas para la solución óptima.

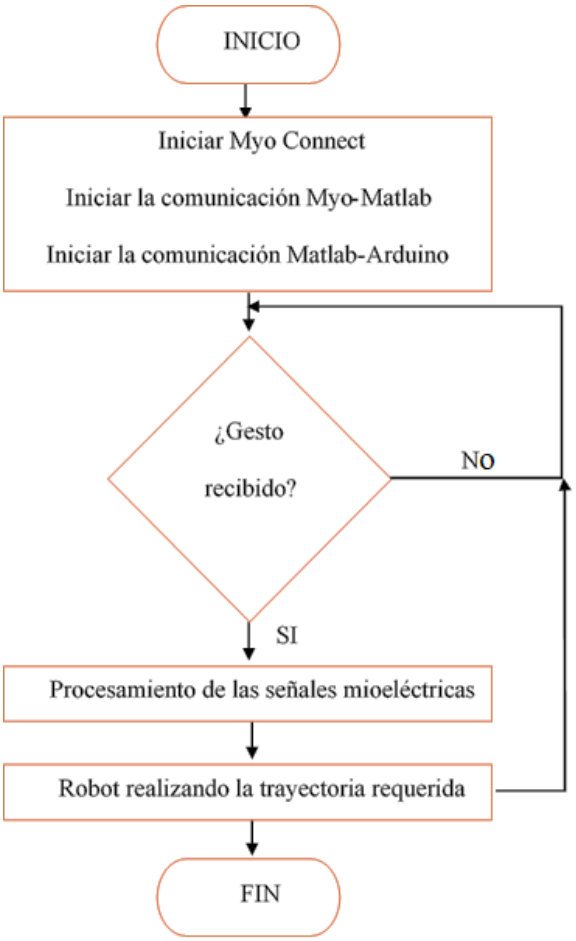

Figura 21: Flujograma para el control del brazo robótico.

El control del robot manipulador mediante señales mioeléctricas se implementa en base a un sistema mecatrónico básico cuyos elementos se muestran en la Figura 22. En este sistema se identifica a los electrodos del brazalete como los sensores, encargados de la adquisición de la señal proveniente del usuario. Como parte de la etapa de procesamiento de la señal actúan el software Matlab y la tarjeta Arduino Uno, identificando el gesto realizado y enviando una señal al controlador del robot que decidirá mediante programación MELFA BASIC V la trayectoria que deberá realizar el actuador, en este caso el robot Mitsubishi RV-2F.

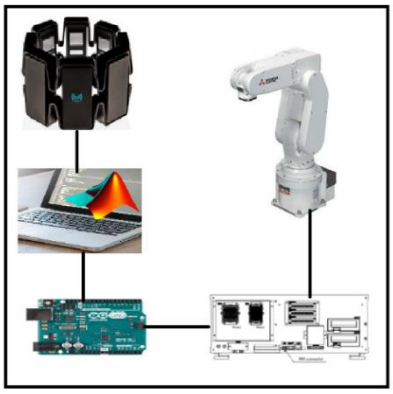

Figura 22: Sistema de control mediante señales mioeléctricas.

2.2 Desarrollo

2.2.1 Comunicación Myo Armband-Matlab

Como se menciona en el punto anterior, la mejor opción es utilizar el entorno de Matlab para el procesamiento de las señales EMG. Para realizar exitosamente la conexión entre Myo y Matlab utilizaremos el software Myo Connect y dos kits de desarrollo de software (SDK).

2.2.1.1 Myo Connect

Lo primero es descargar el software Myo Connect desde la página de soporte de Myo. Una vez instalado el programa aparecerá una ventana con una guía de introducción para registrar el dispositivo y sincronizarlo con la aplicación como se muestra en la Figura 23.

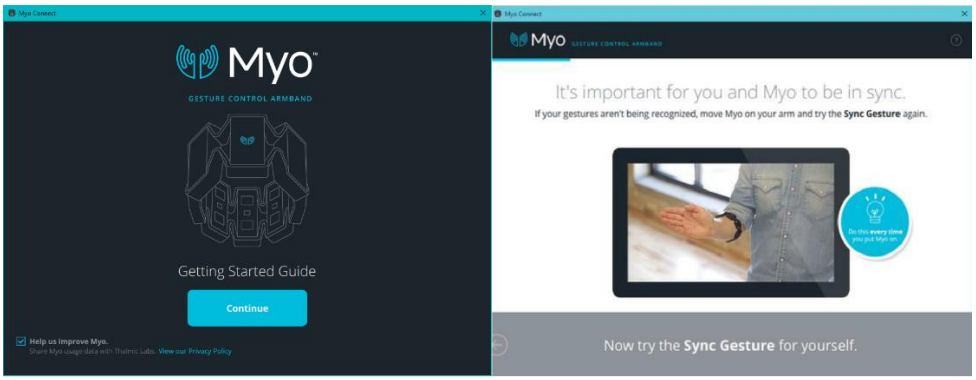

Figura 23: Primeros pasos y sincronización de Myo

Es necesario que el Myo esté conectado mediante un cable USB al computador y que su dongle bluetooth también esté conectado. Esta acción solo es necesaria la primera vez que se conecta el Myo a la computadora personal.

2.2.1.2 Myo SDK

Una vez sincronizado Myo, se debe descargar el Myo SDK que es un conjunto de librerías, cabeceras, código ejecutable, clases y demás información con códigos de ejemplo. Para interactuar con Myo es necesaria la librería libmyo, incluida en el SDK, que proporciona una serie de bindings para C++. Esto permite el acceso de aplicaciones en lenguajes de programación distintos al usado en la librería [31]. En la Figura 24 se puede observar el desarrollo de una aplicación que utiliza el SDK de Myo.

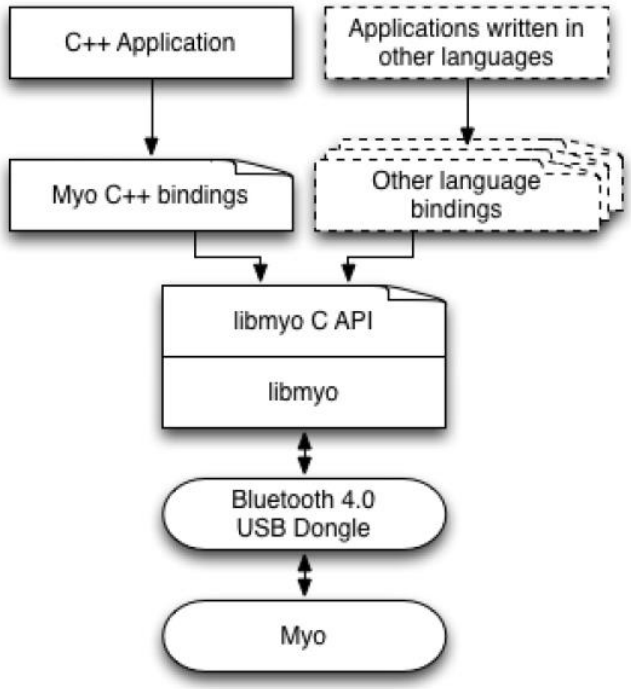

Figura 24: Diagrama de flujo del SDK Myo para el desarrollo de aplicaciones.

Fuente: [31]

Se recomienda extraer el kit de desarrollo en el disco C, se debe agregar la ruta de la carpeta del SDK en Matlab para que el compilador pueda acceder al DLL requerido según el sistema operativo (myo32.dll o myo64.dll).

2.2.1.3 Myo SDK MATLAB Mex Wrapper

Como se menciona anteriormente los bindings C++ permiten manipular las librerías y distintas clases del SDK de Myo para el posterior análisis y procesamiento de la señal a través de Matlab. Se debe ejecutar paralelamente el software Myo Connect para poder acceder a los datos del dispositivo.

El siguiente SDK que se utiliza en la comunicación Myo-Matlab es el Myo SDK MATLAB Mex Wrapper desarrollado por Mark Tomaszewski, que contiene una clase simplificada MyoMex que permite transmitir datos en las frecuencias reales del dispositivo (200Hz EMG y 50Hz IMU). En la Figura 25 se muestra el código para instalar el SDK.

Figura 25: Instalando MyoMex.

2.2.1.4 Clase MyoMex.

Esta clase es un envoltorio cuya función es generar un vínculo entre los datos generados por Myo SDK y los objetos MyoData. Usar esta clase es relativamente sencillo. Solo debemos iniciar MyoMex, quién llama a la función myo_mex y esta inicializa la transferencia de datos

durante el tiempo que se ejecute el código, de esta manera tiene acceso a los objetos tipo MyoData. Al final se debe limpiar la instancia de MyoMex. El código para realizar esto se muestra en la Figura 26.

```
%% Uso del MyoMex para leer Objetos tipo MyoData

mm = MyoMex(countMyos);
m1 = mm.myoData(1);
if countMyos == 2, m2 = mm.myoData(2); end
mm.delete();
```

Figura 26: Código para crear una clase MyoMex que lea objetos tipo MyoData.

2.2.1.5 Clase MyoData.

El objetivo de esta clase es recolectar los datos del dispositivo Myo físico, ofrece esos datos a los usuarios a través de las propiedades <data> y <data>_log. Los usuarios adquieren un puntero inteligente que es utilizado cuando un programa hace referencia a bloques de memoria u objetos controlados por otros sistemas, en otras palabras, el objeto MyoMex llama repetidamente al objeto MyoData para darle los datos que se encuentran disponibles [31]. Las propiedades <data> se muestran en la Figura 27. Todas las propiedades EMG de la clase MyoData son muestreadas a una frecuencia de 200 Hz mientras que las propiedades IMU de esta clase se muestrean a una frecuencia de 50 Hz. Cada vez que el dispositivo es usado se crean archivos de registros escritos a estas frecuencias.

```
47 -    ml.timeIMU
48 -    ml.quat
49 -    ml.rot
50 -    ml.gyro
51 -    ml.gyro_fixed
52 -    ml.accel
53 -    ml.accel_fixed
54 -    ml.pose
55 -    ml.pose_rest
56 -    ml.pose_fist
57 -    ml.pose_wave_in
58 -    ml.pose_wave_out
59 -    ml.pose_fingers_spread
60 -    ml.pose_double_tap
61 -    ml.arm
62 -    ml.arm_right
63 -    ml.arm_left
64 -    ml.arm_unknown
65 -    ml.xDir
66 -    ml.xDir_wrist
67 -    ml.xDir_elbow
```

Figura 27: Propiedades <data> del objeto MyoData.

Estos datos son usados para el procesamiento de las señales y la interpretación de los gestos de Myo en tiempo real.

2.2.1.6 MyoMex_Quickstart.

Una vez que se han instalado los recursos mencionados anteriormente se procede a ejecutar un programa llamado MyoMex_Quickstart.m incluido en la carpeta del kit de desarrollo de software. Este código permite reconocer el número de dispositivos Myo que están sincronizados en Myo Connect, con un máximo de dos dispositivos y lo almacenará en el argumento countMyos. Luego, durante un periodo de muestreo de cinco segundos, el programa almacenará las propiedades del objeto MyoData que detecte en la trasmisión de datos en sus respectivos argumentos, se podrá detener la trasmisión de datos, así como borrar el registro de datos temporales [31]. Además de todo lo mencionado, el programa también mostrará en tiempo real el gesto que se realiza durante la transmisión en la parte inferior derecha de la pantalla como se muestra en la Figura 28.

Figura 28: Gesto reconocido por MyoMex_Quickstart.

Luego de la recopilación de datos, se mostrará en una ventana las gráficas relacionadas a los valores adquiridos del giroscopios, acelerómetros y señales EMG de los ocho cuales superpuestos como lo muestra la Figura 29.

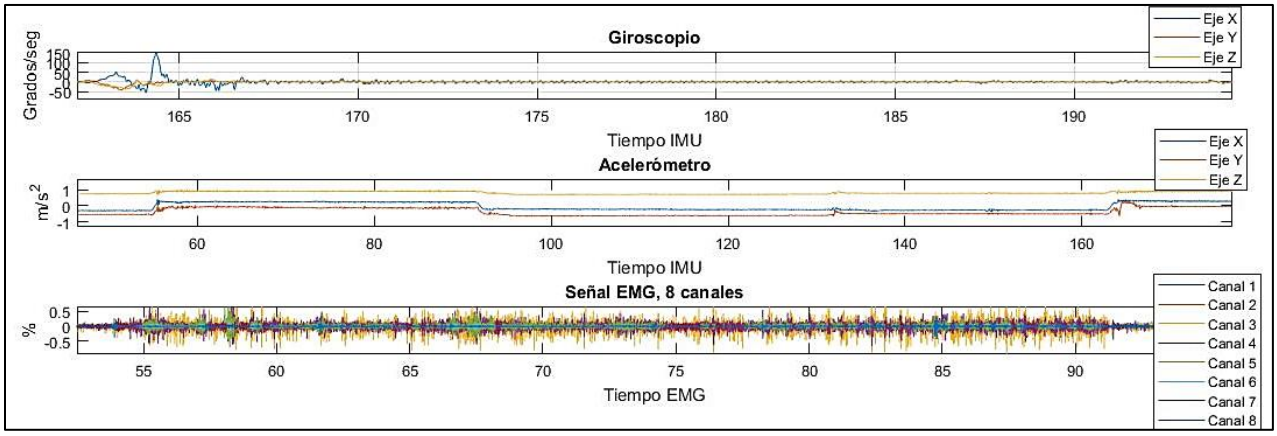

Figura 29: Gráficas de los datos de gyro, accel y EMG's.

Fuente: [31]

Estos datos superpuestos no representan la señal cruda de la actividad muscular, si no una representación normalizada mediante el acondicionamiento de esta, para ello se incluye un filtro pasa alto entre una frecuencia de 10 – 20 Hz y un filtro pasa bajo con una frecuencia de corte entre 100 – 450 Hz. Se utilizan estos rangos debido a que la mayor parte de información de la señal se encuentra entre los valores de 50 – 150 Hz aproximadamente y los armónicos no deseados de las señales mioeléctricas se encuentran a menudo fuera de esos valores.

Para reducir el ruido producido por los equipos eléctricos se puede incluir un filtro Notch reduciendo así la interferencia dentro de las frecuencias de 50 o 60 Hz. Aunque no es una buena

opción debido a que quita energía, produce rotaciones de fase y cambia la forma de onda del electromiograma. Por lo que se opta por utilizar filtros mencionados anteriormente.

2.2.1.7 Ruido de línea base y pérdida de información de la señal

Los causantes principales del ruido y la pérdida de la información de la señal provienen de las señales mioeléctricas mínimas que existen cuando el usuario se encuentra en la posición de descanso y de las señales provenientes de los equipos eléctricos cuyas frecuencias (50 o 60 Hz) se encuentran dentro de la señal útil, por lo que es necesario implementar un filtro con el menor ancho de banda de rechazo para evitar estas pérdidas. Para visualizar correctamente estas señales de interferencia se realiza un análisis de la señal cuando está en inactividad como se muestra en la Figura 30.

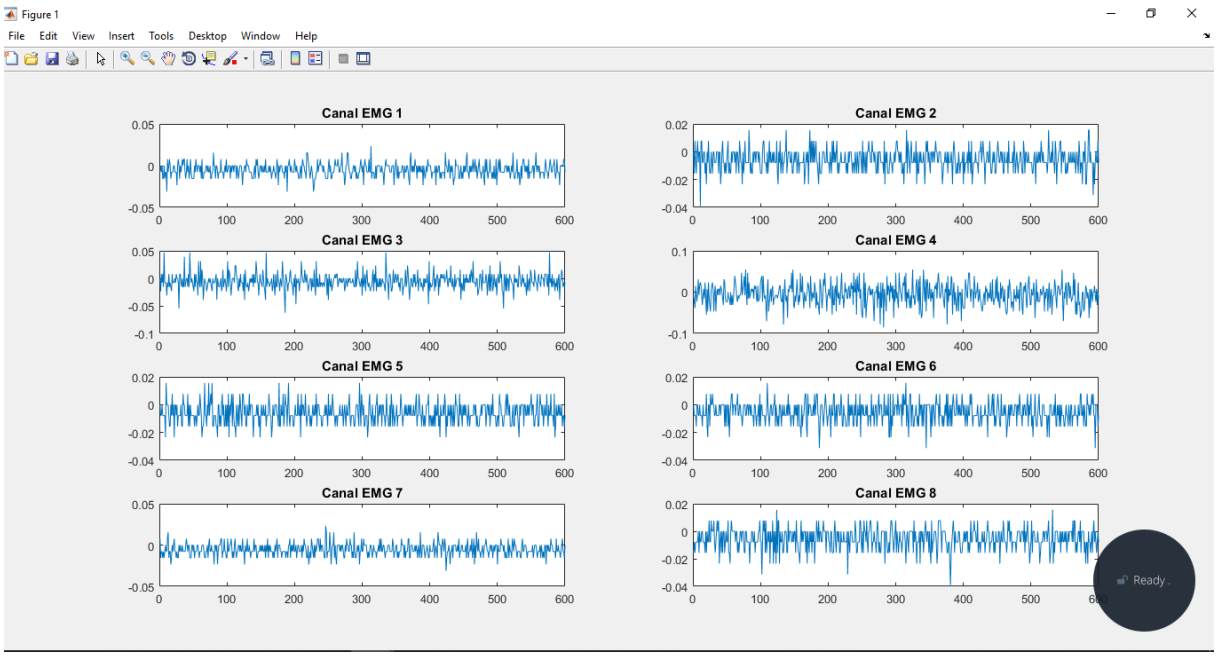

Figura 30: Señales de ruido en la posición de reposo.

Fuente: Propia

Como se aprecia en la figura 30 existen señales mínimas que se detectan constantemente, el canal 4 se muestra con mayor perturbación que el resto. Si estás señales alcanzan amplitudes considerables entonces afectaría el resultado del gesto a clasificar. Estas pequeñas

perturbaciones varían entre personas y dependen también del medio en el que se encuentre el individuo. Los picos altos en cada uno de los canales se deben a otras fuentes de ruido, por ejemplo, el ruido del sistema.

2.2.1.8 Interfaz gráfica Myo-Matlab.

Dentro del kit de desarrollo de software Matlab Mex Wrapper se incluyen dos monitores con interfaz gráfica para el estudio de las señales del Myo Armband, estos monitores también son útiles al momento de verificar si la señal recibida es real y correcta. Dichos monitores se presentan a continuación:

- MyoMexGUI_Monitor: Presentada en la Figura 31. Se utiliza para la instanciación y eliminación de objetos MyoMex. Contiene instrucciones para que los usuarios ingresen el número de dispositivos Myo a analizar. Después de especificar el número de dispositivos, se instancian pulsando el botón “Init MyoMex”. Una vez creado el objeto MyoMex los usuarios pueden iniciar una o dos instancias de monitores MyoData, este número depende de la cantidad de brazaletes.

Figura 31: MyoMexGUI_Monitor.

Fuente: [31]

- MyoDataGUI_Monitor: Presentada en la Figura 32, su argumento de entrada es un objeto MyoData que recibe mediante el monitor anterior. La interfaz permite el inicio

y el paro de trasmisión en cualquier momento. Contiene botones para alternar la representación de los datos mostrados en el gráfico. Estos son presentados en coordenadas del sensor o del marco fijo de referencia. En la parte izquierda del monitor se muestran los datos quat, representados por un cilindro que muestra la orientación del antebrazo del usuario con el codo en el origen de coordenadas. Los controles deslizantes permitirán al usuario ajustar el marco fijo de modo que se pueda hacer coincidir el marco de referencia fijo con el modelo físico del antebrazo. En la parte inferior izquierda se mostrará el gesto detectado siempre y cuando el gesto no sea una pose de descanso o un gesto desconocido.

La parte derecha del monitor muestran los datos más recientes del giroscopio, acelerómetro y los ocho canales junto con la representación de la magnitud de la señal que se ve graficada con una línea negra gruesa, en la parte superior de los datos EMG se muestran una serie de círculos cuyo color depende del gesto para mejorar así la visualización del mismo, en el caso de la Figura 32 los círculos se muestran en un color naranja vinculado al gesto de puño.

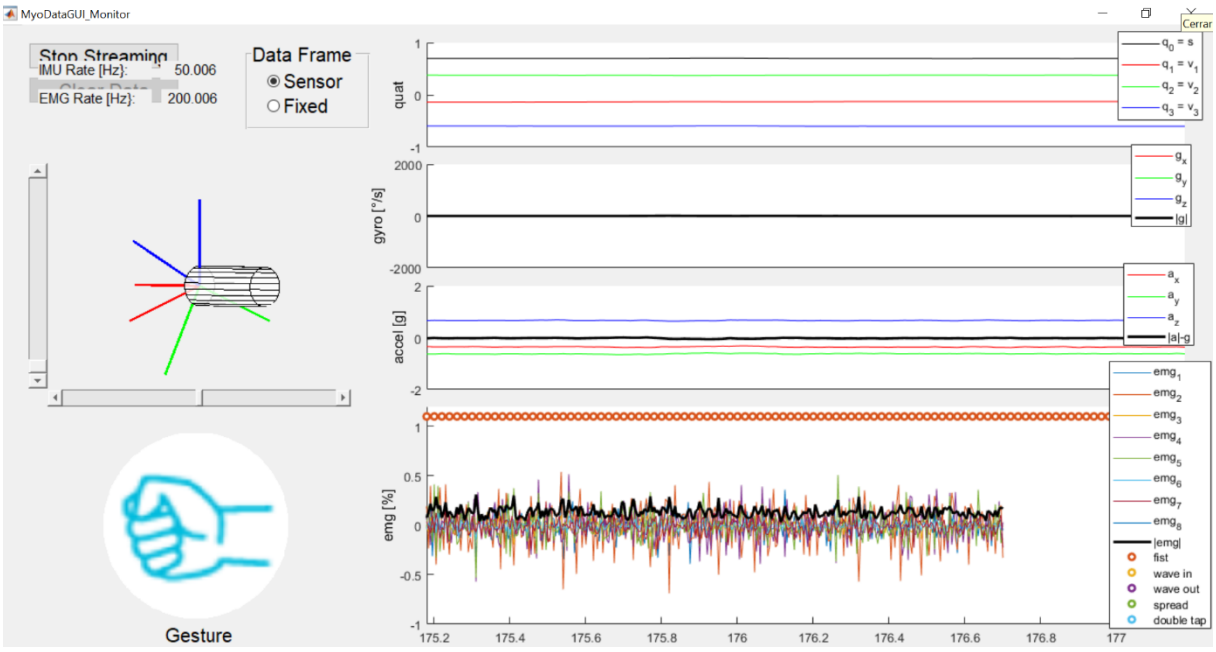

Figura 32: MyoDataGUI_Monitor.

2.2.2 Comunicación Matlab-Arduino

Para esta parte de la comunicación necesitamos tener una cuenta en Mathworks para hacer uso de los soportes de Hardware que se pueden descargar en el apartado Ads-on de la pestaña Home al inicio del software de Matlab como se muestra en la Figura 33.

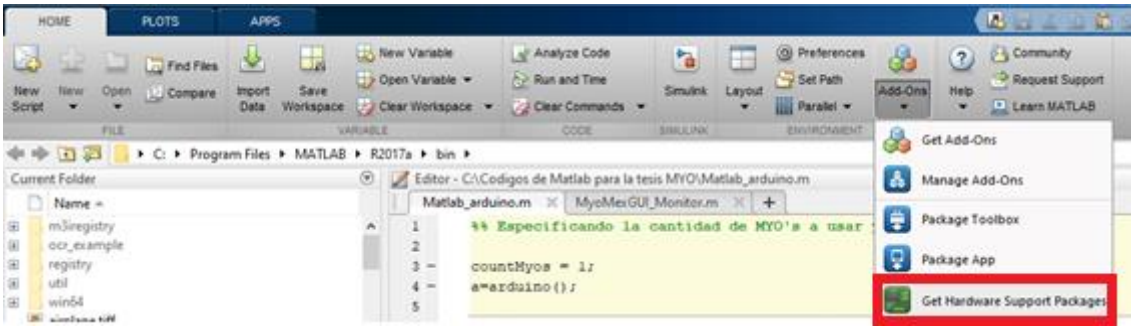

Figura 33: Soporte de Hardware para Matlab.

Al hacer clic en esta opción saltará una nueva ventana donde buscaremos el soporte de hardware Arduino para Matlab y seguimos los pasos que se indica para la instalación.

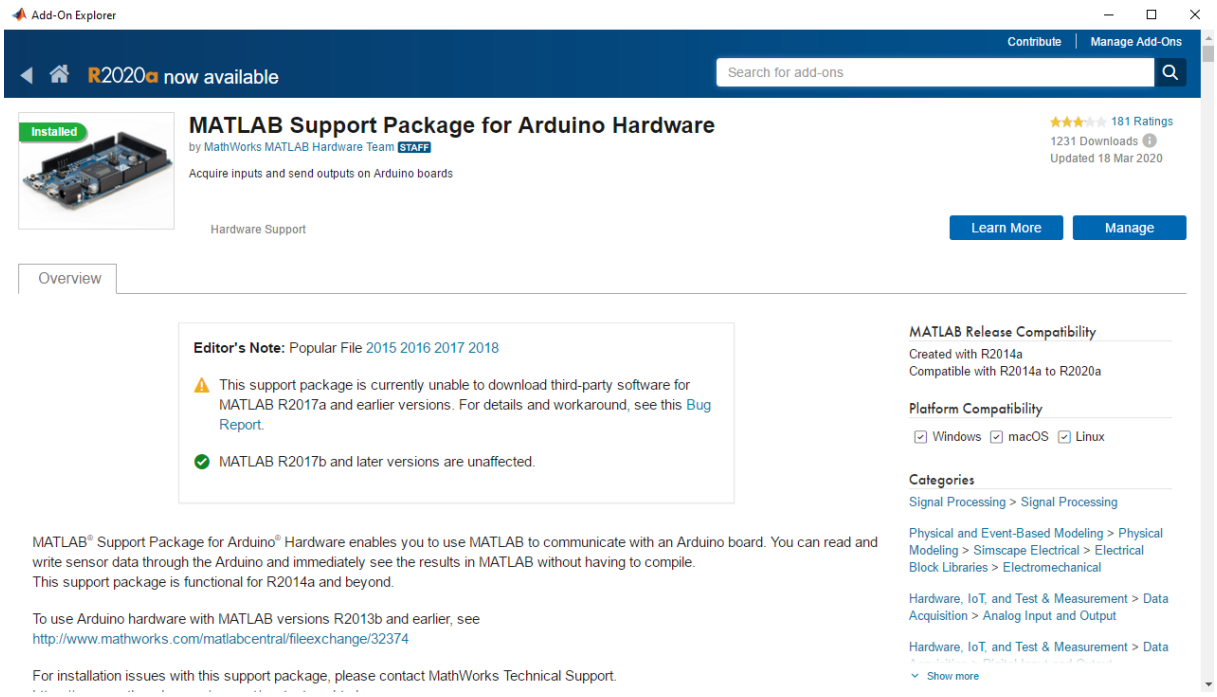

Figura 34: Soporte de hardware Arduino para Matlab.

Luego de descargar el soporte de hardware tenemos la instrucción arduino() para iniciar la comunicación serial entre Matlab y Arduino a través de algún COM de la computadora

personal. Entre las funciones que este soporte ofrece está el controlar los pines de entradas/salidas de Arduino, lo cual es posible a través de la instrucción writeDigitalPin(a,'D5',1); Donde: a es el objeto Arduino instanciado, D5 es el pin digital y 1 es el estado de la salida para encendido. Utilizando estás funciones podremos enviar una señal al controlador desde la tarjeta según el gesto reconocido por Matlab.

2.2.3 Comunicación Arduino-Controlador CR750-D

2.2.3.1 Entradas/Salidas CR750-D.

Este controlador contiene una interfaz paralela de entradas y salidas en el slot 1 que administra las salidas del PLC como entradas al microcontrolador.

Estas entradas y salidas se interpretan como “1” cuando la tensión alcanza un valor de 24 VDC y como “0” cuando el valor es 0 VDC. Las entradas y salidas reservadas del controlador se muestran en la tabla 11.

Tabla 11: Entradas/Salidas reservadas CR750-D.

Entrada	Salida PLC	Entrada Slot 1	Salida	Salida Slot 1
Stop		0	Run	0
Servo OFF	O02	1	Servo On	1
Error Reset	O03	2	Error	2
Start	16	3	Oper. Enable	3
Servo On	O01	4	Assembling	4
Oper. Enable	07	5	Cycle End	5
Robot Auto/Manual	O04	6	Close Gripper	6
Robot Start	O01	7	Screwier Start	7
Cycle Start	O05	8	Screwing	8

Tall Bearing	O06	9

Fuente: [32]

Existen 23 entradas disponibles para conectar las salidas del Arduino, si se desea visualizar estas entradas se puede hacer uso del Teach Pendant siguiendo las siguientes instrucciones: Entrar en la opción Menú de la pantalla inicio, luego iremos a Monitor/General Purpose Signal como se muestra en la Figura 35.

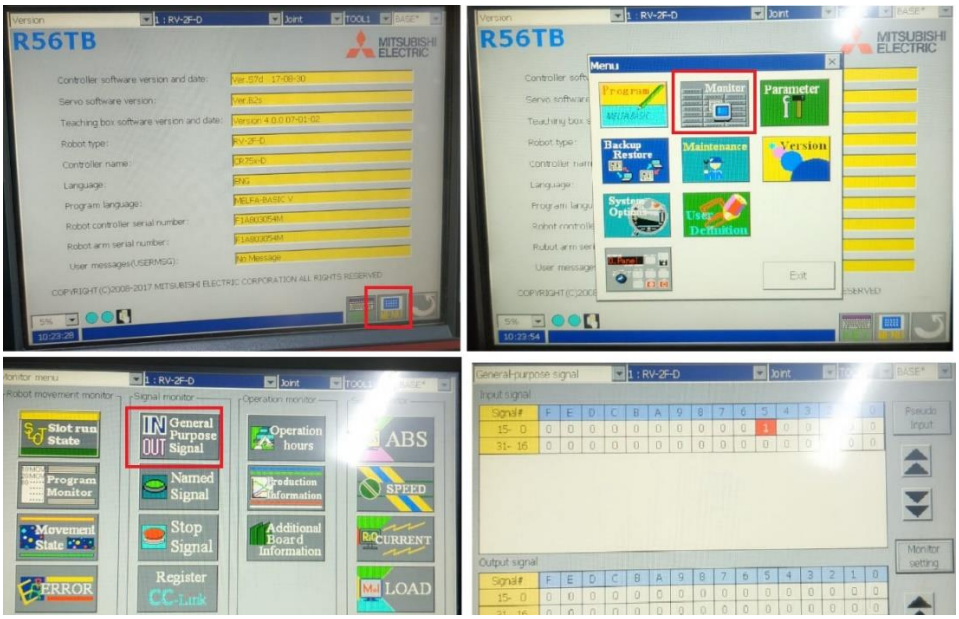

Figura 35: Visualización de las entradas y salidas del controlador CR750-D.

2.2.3.2 Optoacoplador PC817

Las señales que recibe el PLC en sus entradas deben tener una amplitud de 24 VDC para poder ser leída por lo tanto se debe aplicar un circuito que incremente el voltaje de salida del Arduino Uno de 3,3V a 24V para lo que empleamos un optoacoplador PC817. El diseño de este optoacoplador se obtiene de [32], el cual está implementado con las respectivas conexiones en el robot Mitsubishi RV-2F. Este optoacoplador es un sistema de aislamiento eléctrico entre dos etapas de un circuito, basado en el acoplamiento de un fotodiodo infrarrojo que cumple la función de emisor y un fototransistor que es el colector emisor, este sistema funciona como un

interruptor activando el fototransistor mediante la luz emitida por el fotodiodo, garantizando el aislamiento y protección eléctrica de la parte digital de control [32].

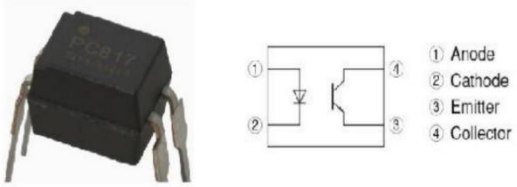

Figura 36: Optoacoplador PC187.

Fuente: [32]

El circuito implementado para el optoacoplador se muestra en la Figura 37, donde Vcc1 es la salida de la tarjeta Arduino de 3,3V, R1 es igual a 180 Ω, R2 es igual a 5,45 kΩ y Vcc2 es igual a 24V.

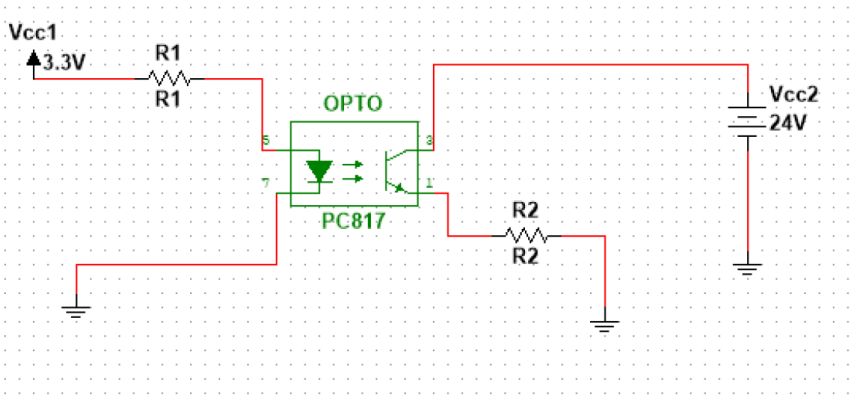

Figura 37: Diseño del circuito con optoacoplador.

Fuente: [32]

Este diseño es implementado para cada una de las salidas de la tarjeta Arduino que a su vez estarán enviando señales a las entradas del slot 1 del controlador. De la siguiente manera:

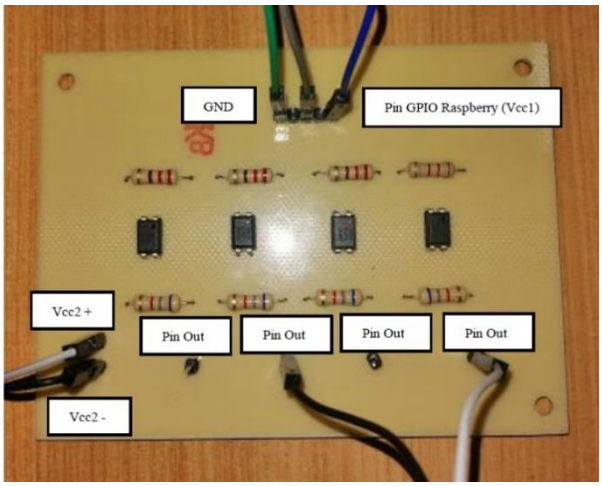

Figura 38: Circuito Amplificador de voltaje para 4 entradas.

En la tabla 12 se muestran los pines de la tarjeta Arduino conectados a las entradas del slot 1.

Tabla 12: Conexión entre pines de Arduino y entradas del slot 1.

Pin de Arduino Uno		**Entrada Slot 1**
Pin Físico	**Puerto del Pin**	
6	PD4	10(A)
7	PD5	11(B)
8	PD6	12(C)
9	PD7	13(D)

2.2.4 Programación de trayectorias

El lenguaje de programación usado para este algoritmo de control es el lenguaje MELFA BASIC V. En el anexo B se muestra el código de programación empleado.

Las trayectorias para el control de la posición del robot se pueden clasificar en tres tipos: Automática (Trayectoria simultánea de ejes), semiautomática (que consiste en una trayectoria pick and place modificada) y manual (Trayectoria Cartesiana). Esto se logra a través de un

menú interactivo de gestos en el que el usuario ingresa a un segundo nivel del código haciendo el gesto *Wave In* seguido del gesto *Fist* o un tercer nivel haciendo el gesto *Wave In* seguido del gesto *Wave Out* como se muestra en la Figura 39.

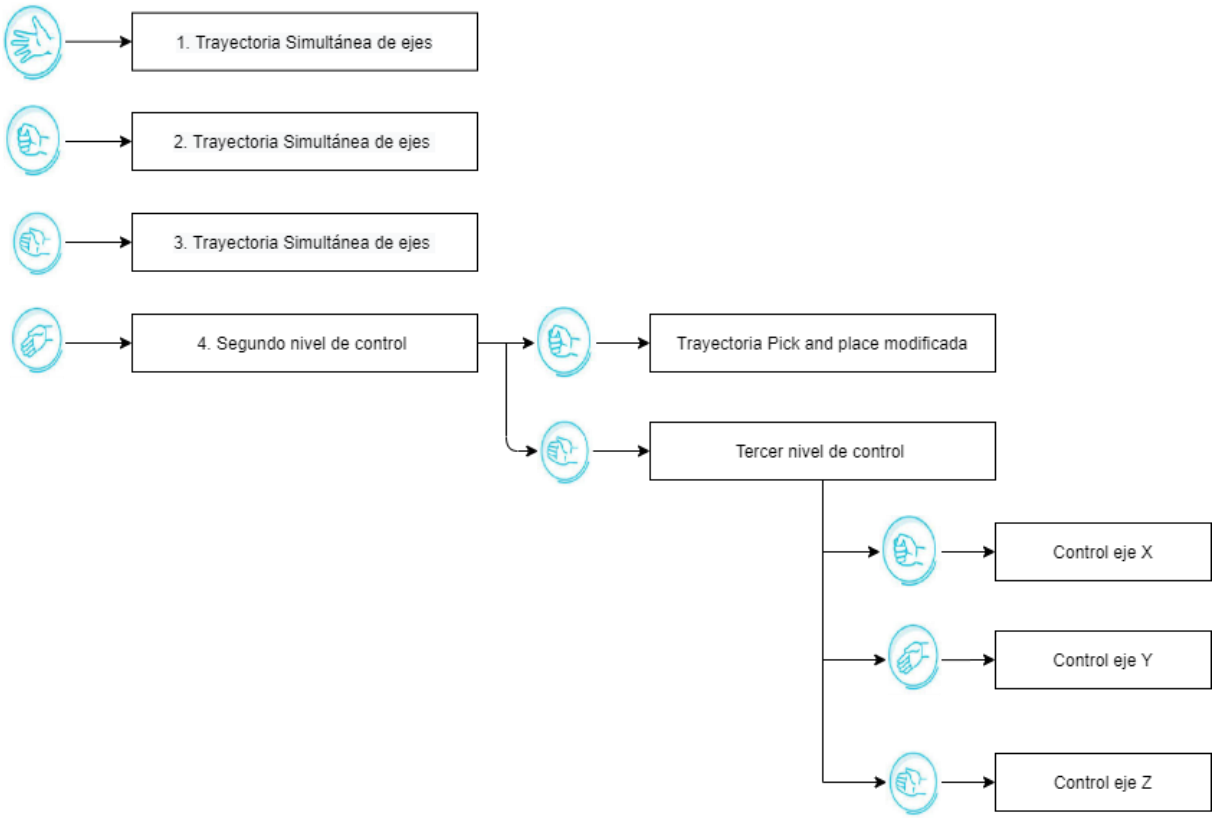

Figura 39: Esquema del menú gestual desarrollado.

Fuente: Propia.

En el primer tipo de control de trayectoria (simultánea de ejes) el robot esperará a que el usuario realice uno de los tres gestos predeterminados para este nivel de control (*Spreadfingers, Fist* y *Wave out*). Al realizar el gesto *Spreadfingers* el robot realizará la actividad de recoger un objeto empleando la garra del robot, en este caso será una tapa circular. La recogerá en una posición determinada, luego la ubicará en otra posición y por último la regresará a su lugar inicial. Al finalizar cada trayectoria el robot volverá a esperar una siguiente instrucción. Si el usuario realiza el gesto de *Fist* el robot agarrará un eje de una determinada posición prevista con anterioridad y la llevará hacia el centro, dibujará un círculo con la pieza y lo regresará a su

posición inicial. Al hacer el gesto *Wave Out*, el robot irá a la posición de un elemento a ensamblar y lo atornillará con su segunda herramienta.

Para ingresar al segundo nivel de control el usuario deberá hacer el gesto de *Wave In* seguido de *Fist*. Después de ingresar a la trayectoria de pick and place modificada, el robot esperará a que el usuario realice uno de tres gestos (*Fist, Wave In, Spreadfingers*) para determinar la pieza a ser recogida, de esta manera los gestos están relacionados a tres posiciones diferentes que corresponden a tres piezas que podrían ser distintas en forma o tamaño, tomando el ejemplo de las trayectorias anteriores estos pueden ser ejes, tapas, cilindros o cualquier pieza que sea capaz de recoger el robot, una vez el usuario hace el gesto el robot se dirige a la pieza y espera el gesto *Fist* para el cierre de la garra del robot, levanta el objeto y lo lleva a otra posición y volverá a esperar el gesto *Fist* para soltar la pieza y regresar a la posición inicial. Esta es una de las posibles configuraciones de trayectoria que se pueden aplicar en este segundo nivel, se pueden utilizar, de ser necesario, trayectorias más complejas y que requieran más gestos.

Para ingresar al tercer nivel de control de trayectorias cartesianas el usuario debe realizar el gesto *Wave In* seguido de *Wave Out*, el robot esperará un tercer gesto que puede ser *Fist* para controlar el eje “x”, *Wave In* para controlar el eje “y” y *Wave Out* para controlar el eje z. Una vez se elige el eje de coordenadas a controlar se utilizan los gestos *Wave In* para ir en dirección positiva y *Wave Out* para la dirección negativa del eje. El gesto *SpreadFingers* en este nivel permite salir del eje de coordenadas para cambiarlo por otro y si se realiza dos veces entonces saldrá al menú del inicio. El gesto *double tap* se reserva para parar la comunicación con el robot.

CAPÍTULO III

RESULTADOS

3.1 Pruebas de funcionamiento

Una vez establecidas las conexiones físicas entre los dispositivos descritos en el apartado anterior procedemos a realizar las pruebas de funcionamiento de las distintas subfunciones del algoritmo de control para un brazo robótico Mitsubishi basado en señales mioeléctricas dentro del laboratorio de Manufactura Integrada por Computadora de la Universidad Técnica del Norte. Estas pruebas fueron realizadas tomando en cuenta las siguientes características: El dispositivo deberá ser colocado en el brazo derecho a unos 5 centímetros desde la articulación del codo. Siempre manteniendo el primer sensor (marcado con el logo de Thalmics) en la misma posición como se muestra en la siguiente figura.

Figura 40: Correcta posición del brazalete.

3.1.1 Adquisición de señales en Matlab

Se empezó con las pruebas de comunicación, para ello vamos a emplear el código detallado en el Anexo A para graficar los datos del giroscopio, acelerómetro y los datos de los canales EMG como se muestra en la Figura 41.

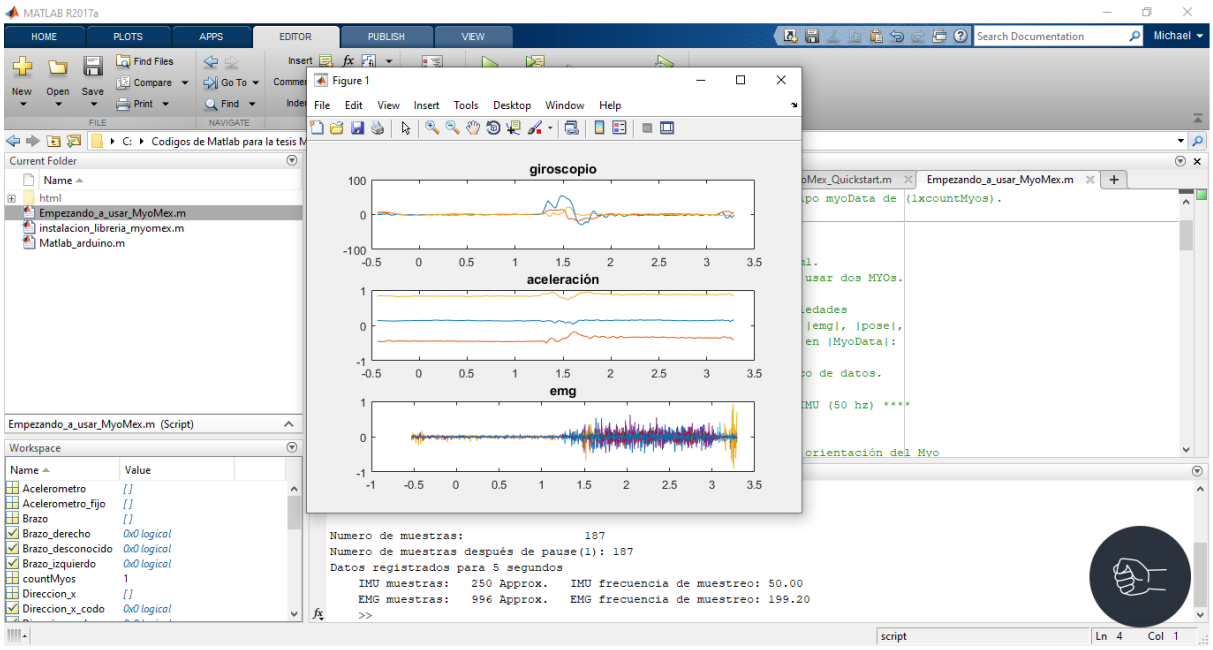

Figura 41: Pruebas de conexión entre Myo y Matlab.

En la parte inferior de la imagen anterior se puede apreciar el número de muestras tomadas y los datos registrados en MyoData referente al muestreo con las frecuencias IMU (50 Hz) y EMG (200 Hz). Una vez comprobado que el software recibe las señales del dispositivo procedemos a realizar el código de reconocimiento de gestos (Anexo C) en conjunto con los monitores para comprobar que se esté enviando la señal correcta correspondiente al gesto realizado.

3.1.2 Interpretación de gestos y activación de salidas

Como se mencionó en el capítulo anterior, el SDK de Myo para Matlab incluye funciones que son de gran utilidad al momento de reconocer los gestos predeterminados del Myo, para el posterior envío de señal hacia el controlador. En el anexo C se encuentra el código utilizado que emplea las funciones integradas en el SDK mencionado.

Se hace uso del monitor incluido en el kit de desarrollo para comprobar que Matlab envíe la señal correcta correspondiente al gesto realizado hacia el Arduino Uno. En la Figura 42 se muestra el monitor detectando el gesto Fist y activando una salida digital de la tarjeta Arduino.

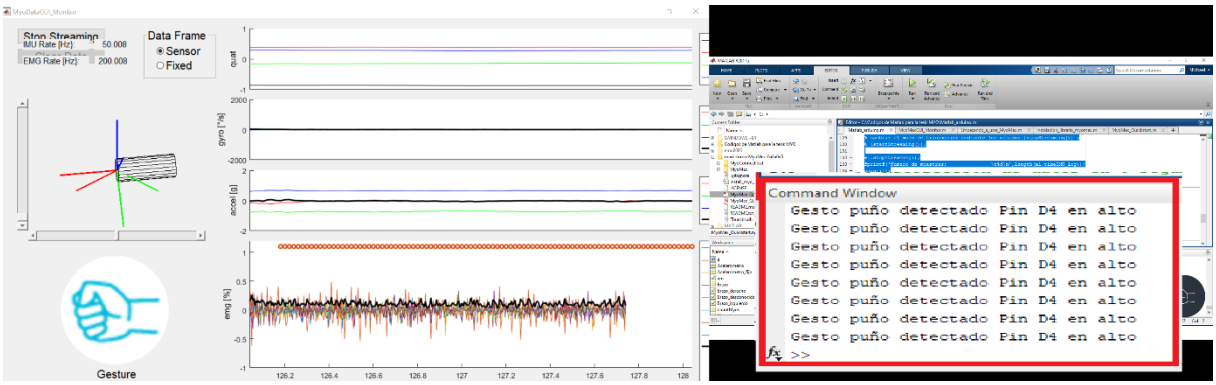

Figura 42: Pruebas de interpretación de gestos y envío de señal a Arduino.

Como se puede apreciar en la imagen anterior, la salida del Arduino se activa cuando detecta el gesto y permanece inactiva cuando no reconoce el gesto o el antebrazo se encuentra en reposo.

3.1.3 Envío de señal y ejecución de trayectorias del robot

En los siguientes puntos se describe el envío de señales desde el Arduino al controlador, el diseño de un case para la tarjeta y la correcta interpretación de los gestos recibidos mediante la comparación de la activación de los 8 diferentes canales para señales EMG que posee Myo.

3.1.3.1 Envío de señal Arduino-Optoacoplador-Controlador

En este apartado se utiliza el menú del Teach Pendant para comprobar que las salidas del Arduino se amplifican en el optoacoplador y pueden activar las entradas del controlador. Para ello se muestra en la Figura 43 la conexión del Arduino con las entradas del optoacoplador

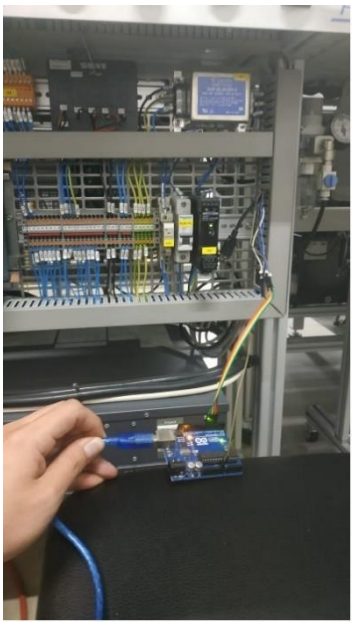

Figura 43: Conexión Arduino-Optoacoplador-Controlador.

Como se muestra en la figura 44 la entrada B del controlador CR750-D se activa cuando se detecta el gesto de puño y se desactiva cuando no hay presencia de este.

Figura 44: Prueba de activación de entrada del controlador.

En cuanto al case, se optó por un diseño sencillo que deje la posibilidad de acceder a todos los pines de la tarjeta. La parte trasera del case tiene una forma que encaja con el riel din que sostiene los elementos del robot Mitsubishi. El case se muestra en la Figura 45.

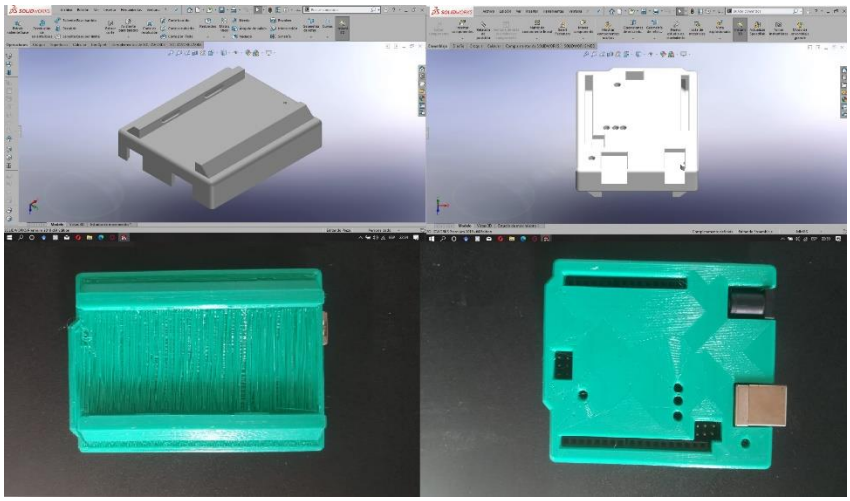

Figura 45: Case para colocar el Arduino en un riel DIN.

3.1.3.2 Ejecución de las trayectorias del robot

Se hizo una prueba por separado de las trayectorias programadas para comprobar que no existiese algún error o colisión al momento de probar el algoritmo final. Luego se hizo una prueba para verificar que el menú basado en gestos funcionara correctamente, ingresando y saliendo de los diferentes niveles de programación, así como también probando las distintas trayectorias una a una y en un orden aleatorio. El laboratorio de Manufactura Integrada por Computadora de la Universidad Técnica del Norte facilitó el uso del robot Mitsubishi RV-2F para las pruebas de funcionamiento.

Como se ha mencionado anteriormente la trayectoria que realiza el actuador y el ingreso a los subniveles de control dependen del gesto que se reciba, por lo tanto las lecturas de estos gestos deben ser precisos. En las Figuras 46, 47, 48 y 49 se presentan las lecturas de los ocho canales EMG del dispositivo Myo Armband para los gestos *Fist, Wave In, Wave Out y Spreadfingers* respectivamente, se analizarán los canales que posean mayor actividad para cada gesto y así visualizar la clasificación que realiza el programa de una forma general.

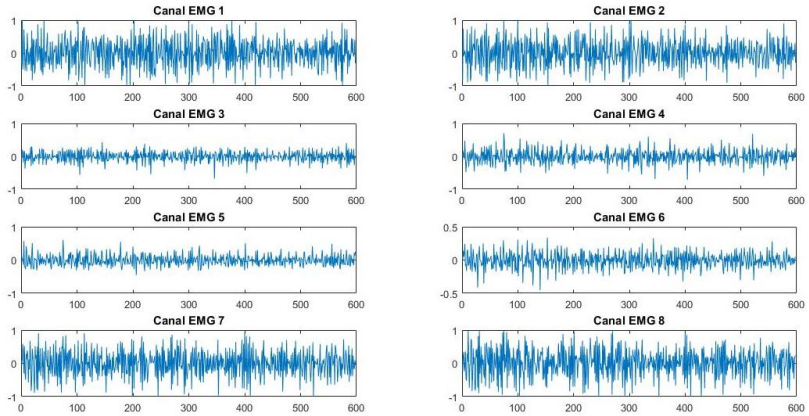

Figura 46: Actividad en los ocho canales EMG para el gesto Fist.

Para el gesto *Fist* se puede apreciar una actividad constante en los canales 1, 2, 7 y 8 seguido de una actividad moderada en 6 y casi nula en 3,4 y 5.

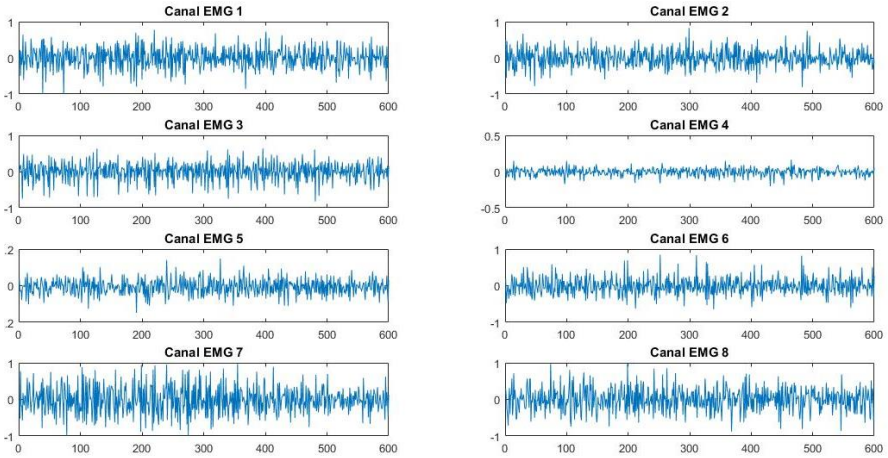

Figura 47: Actividad en los ocho canales EMG para el gesto Wave In.

La mayor actividad para el gesto *Wave In* se presenta en el canal 5 donde el valor de la señal claramente supera el valor de 1. Los canales 7 y 8 también demuestran actividad, puesto que su valor es constantemente cercano a uno, se presenta actividad moderada en los canales 1, 2, 3 y 6 y casi nula en el canal 4.

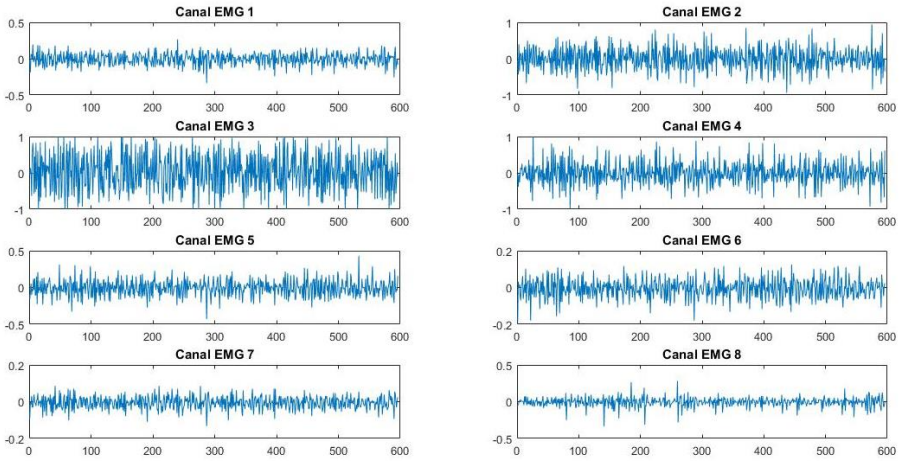

Figura 48: Actividad en los ocho canales EMG para el gesto Wave Out.

El canal 3 para el gesto *Wave out* es el de mayor actividad, en los canales 2 y 4 se observa una actividad constante mientras que en el resto de los canales la actividad es considerablemente menor.

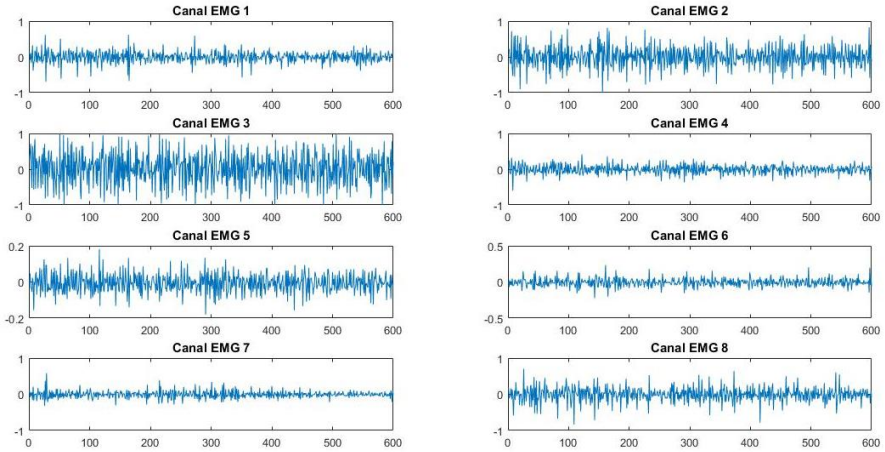

Figura 49: Actividad en los ocho canales EMG para el gesto Spreadfingers.

Para el gesto *Spreadfingers* la mayor actividad se presenta en el canal 3, una actividad moderada en 2 y casi nula en el resto de los canales.

En la tabla 13 se muestra la recopilación general de la actividad de los canales para los gestos que se utilizan en el control del brazo robótico. El color verde indica actividad constante, el color amarillo detona actividad moderada y el rojo nula actividad.

Tabla 13: Comparación de actividad en los canales para los 4 gestos.

Canal / Gesto	Canal 1	Canal 2	Canal 3	Canal 4	Canal 5	Canal 6	Canal 7	Canal 8
Fist								
Wave In								
Wave Out								
Spread Fingers								

Como se puede observar en la tabla anterior todos los gestos tienen al menos un canal con el que se puede identificar qué tipo de gesto se está realizando. Por ejemplo, para identificar el gesto *Fist* se puede verificar la actividad en 1 y 2, para el gesto de Spreadfingers se puede realizar la comparación con el canal 3 o el canal 5 para identificar el gesto de Wave In. El error podría radicar en los picos de los canales con actividad moderada, si estos superan un porcentaje alto de la actividad se podrían interpretar como actividad constante y el algoritmo clasificaría al gesto de una manera distinta. También se debe tener en cuenta que los electrodos del dispositivo Myo se deben colocar en la misma posición, el desplazamiento del dispositivo de solamente unas decenas de milímetros produce un cambio notable en las lecturas de los canales.

En la tabla 14 obtenida de [31] se muestran los valores medios absolutos (MAV) totales y recortados para cada uno de los canales EMG del dispositivo según el gesto realizado, de esta forma podremos visualizar de manera cuantitativa la cantidad de actividad que realizan los ocho canales.

Tabla 14: Valores medios Absolutos totales y recortados de los ocho canales según el gesto.

		EMG1	EMG2	EMG3	EMG4	EMG5	EMG6	EMG7	EMG8
Fist	MAVT	0.109	0.046	0.056	0.111	0.149	0.175	0.128	0.190
	MAVR	0.127	0.060	0.066	0.138	0.144	0.216	0.181	0.312
Spread Fingers	MAVT	0.058	0.037	0.037	0.099	0.090	0.045	0.075	0.0526
	MAVR	0.156	0.056	0.084	0.278	0.262	0.087	0.169	0.126
Wave In	MAVT	0.152	0.112	0.064	0.078	0.108	0.065	0.073	0.129
	MAVR	0.225	0.112	0.063	0.060	0.149	0.105	0.103	0.225
Wave Out	MAVT	0.036	0.125	0.080	0.215	0.167	0.056	0.035	0.025
	MAVR	0.046	0.108	0.120	0.372	0.295	0.121	0.060	0.038

Fuente: Propia

3.1.4 Análisis de resultados

La implementación del algoritmo de control del brazo robótico mediante señales mioeléctricas muestra una correcta ejecución de las trayectorias programadas, debido a que la caracterización de las señales muestra que existen diferencias notables entre las lecturas de los distintos canales. Los datos recibidos son interpretados en Matlab con un error mínimo en sus lecturas. Dicho margen de error se debe a que el dispositivo no siempre será colocado de la misma manera ya que la posición de los sensores varía unos cuantos milímetros en cada prueba. En ocasiones estos desplazamientos cambian completamente las lecturas del musculo implicado en el movimiento, de tal manera que los canales con actividad moderada se podrían interpretar como canales con señales constantes. La navegación del menú basado en gestos cumple su función correctamente permitiendo acceder a la trayectoria requerida realizando los movimientos establecidos con anterioridad.

3.1.5 Análisis Financiero

En este apartado se realizará una estimación de costos realizados en la implementación del algoritmo de control, se dice estimación ya que los elementos para desarrollar el proyecto fueron otorgados por la Universidad Técnica del Norte y los ingenieros de esta, estos elementos son: Myo Armband perteneciente a la carrera de Mecatrónica, un Arduino Uno del laboratorio de Manufactura Integrada por computador y Optoacoplador del mismo laboratorio. Los elementos como el brazo robótico con su respectivo PLC y controlador no serán analizados. La tabla 15 muestra los distintos elementos utilizados y su precio comercial.

Tabla 15: Costos del proyecto.

Descripción	**Unidad**	**Costo Unitario**	**Costo Total**
Brazalete Myo Armband	1	$250,00	$250,00
Arduino Uno	1	$10,00	$10,00
Optoacoplador PC-817	4	$1,00	$4,00
Total			$319,00

El costo del dispositivo Myo Armband puede variar según la región. La implementación de este proyecto resulta relativamente barata si se considera el horizonte de posibilidades para la creación de aplicaciones y configuración de trayectorias programables. En la tabla 16 se muestran los elementos utilizados que fueron costeados por la Universidad Técnica del Norte.

Tabla 16: Elementos otorgados por la Institución.

Descripción	**Unidad**	**Costo Unitario**	**Costo Total**
Robot Mitsubishi RV-2F	1	$0	$0
Licencias de Software	2	$0	$0
Total			$0

CAPÍTULO IV

CONCLUSIONES Y RECOMENDACIONES

4.1 Conclusiones

- El dispositivo de adquisición de señales mioeléctricas Myo Armband permite la obtención de las señales EMG normalizadas de los ocho canales correspondiente a los multi-electrodos superficiales, dichas señales son procesadas dentro del dispositivo mediante algoritmos de clasificación y filtros para obtener una serie de gestos predeterminados, los cuales se utilizan para el posterior control de trayectorias del robot.
- Se obtuvo una interfaz que es capaz de reconocer gestos mediante el procesamiento de las señales que se reciben desde el dispositivo Myo, la misma que simplifica el acondicionamiento de la señal gracias a las funciones, objetos y clases incluidos en el kit de desarrollo de software.
- La implementación del sistema de control mediante señales mioeléctricas para un brazo robótico consta de un dispositivo Myo Armband para la adquisición de señales, una tarjeta comercial Arduino Uno para la comunicación entre las salidas del microcontrolador y las entradas del controlador CR750-D. Mediante una etapa de amplificación de señal y programación en lenguaje Melfa Basic V se obtiene el sistema de trayectorias. Las señales enviadas hacia el controlador se utilizan para la elección de la trayectoria que el usuario desee que realice el actuador.
- Por medio de las pruebas realizadas en las distintas etapas de comunicación y control de trayectorias se observa el correcto funcionamiento del algoritmo de control de un brazo robótico basado en reconocimiento de gestos mediante señales EMG provenientes de las extremidades superiores. Estas pruebas mostraron también que existen pérdidas de

información dentro de la señal útil debido al ruido provocado por los equipos eléctricos y las señales mínimas que se obtienen constantemente en los músculos.

4.2 Recomendaciones

- Implementar una clasificación mejorada de gestos mediante redes neuronales, esto otorgaría mayor naturalidad al momento de interpretar el gesto del usuario.
- Creación de nuevos gestos para mejorar la navegación entre trayectorias, ya que entre más de estos se tengan, mayor orden tendrá la programación del controlador del robot y mejor será el aprendizaje intuitivo para el usuario.
- Implementar una interfaz gráfica que muestre la posición del usuario dentro del menú y las trayectorias que se pueden seleccionar en esa parte del código, disminuyendo de esta manera la elección equivocada por parte del usuario.
- Realizar nuevas rutinas con el brazo robótico, añadiendo variables de otros tipos, como las de tipo array para el control de paletización de elementos.

BIBLIOGRAFÍA

[1] A. Ollero Baturone, Robótica: Manipuladores y Robots Móviles, Barcelona: Marcombo S.A, 2001.

[2] J. F. Reyes Cortés , Robótica - control de robots manipuladores, México: Alfaomega , 2011.

[3] Z. Pinwei, «Design of surface electromyography detection circuit,» de *International Conference on Future Information Technology and Management Engineering (FITME)*, Changzhou, 2010.

[4] JPL, «IEEE Spectrum,» 16 Mayo 2013. [En línea]. Available: https://spectrum.ieee.org/automaton/robotics/robotics-hardware/jpl-biosleeve-enables-precise-robot-control-through-hand-and-arm-gestures. [Último acceso: 23 Mayo 2020].

[5] F. S. Cadena Meneses, «Tarjeta de acondicionamiento para prótesis de rodilla accionada por señales electromiográficas,» Ibarra, 2015.

[6] J. P. Sanipatín Díaz, «Sistema para la adquisición y acondicionamiento de señales electromiográficas para el accionamiento de un tobillo robótico,» Ibarra, 2015.

[7] C. N. p. l. I. d. Discapacidades, «Consejo Nacional para la Igualdad de Discapacidades,» 25 Mayo 2020. [En línea]. Available: https://www.consejodiscapacidades.gob.ec/estadisticas-de-discapacidad/. [Último acceso: 25 Mayo 2020].

[8] C. De Luca, «Electromyography,» de *Encyclopedia of Medical Devices and Instrumentation*, Boston, John Wiley & Sons Inc, 2006, pp. 81 - 105.

[9] E. Criswell, Cram's Introduction to Surface Electromyography, Jones & Bartlett Publishers, 2010.

[10] J. Trontel, J. Jabre y M. Mihelin, «Needle and wire detection techniques,» de *Electromyography: Physiology, Engineering, and Non-Invasive Applications.*, USA, Wiley-IEEE Press, 2004, pp. 27- 46.

[11] J. I. G. Angarita, La Electromiografía: Un Acercamiento al Concepto Fisiológico, la Construcción de un Equipo Electromiográfico con Registro no Invasivo; y la Resistencia Galvánica de Piel como Método de Relajación Muscular., Pereira, 2009.

[12] G. Soderberg, Selected topics in surface electromyography for use in the occupational setting: expert perspectives., Cincinnati: Department of Health and Human Services., 1992.

[13] F. H. Netter, Atlas de anatomía humana, Ámsterdam: Elsevier, 2014.

[14] Ottobock, «Ottobock,» 2014. [En línea]. Available: https://www.ottobock.es/protesica/miembro-inferior/articulaciones-de-rodilla/c-leg/. [Último acceso: 8 Septiembre 2020].

[15] P. D. SAS, «Parrot Drone SAS,» 2020. [En línea]. Available: https://www.parrot.com/us/drones/parrot-ardrone-20-elite-%C3%A9dition. [Último acceso: 8 Septiembre 2020].

[16] OYMotion, «OYMotion Magic Now,» 2020. [En línea]. Available: http://oymotion.com/en/product32/150. [Último acceso: 8 Septiembre 2020].

[17] F. Ramírez, Diseño de una arquitectura de software y aplicaciones prácticas para explotar las capacidades de un sistema de reconocimiento de gestos del brazo humano, 2018.

[18] S. M. A, «Myo | Announcing Raw EMG Data for Developers from the Myo Armband.,» NORTH, 2014. [En línea]. Available: https://developerblog.myo.com/big-data/. [Último acceso: 21 Noviembre 2020].

[19] T. Labs, «NORTH,» 2020. [En línea]. Available: https://support.getmyo.com/hc/en-us/articles/202648103-Myo-Gesture-Control-Armband-tech-specs. [Último acceso: 8 Septiembre 2020].

[20] Arduino, «Arduino Cl,» Arduino, 2020. [En línea]. Available: http://arduino.cl/que-es-arduino/. [Último acceso: 8 Septiembre 2020].

[21] Arduino, «Arduino Cl,» Arduino, 2020. [En línea]. Available: https://arduino.cl/producto/arduino-uno/. [Último acceso: 8 Septiembre 2020].

[22] R. P. Foundation, «Raspberry PI,» 2020. [En línea]. Available: https://www.raspberrypi.org/documentation/faqs/#introduction. [Último acceso: 8 Septiembre 2020].

[23] N. Instrument, «National Instrument,» 2020. [En línea]. Available: http://www.ni.com/pdf/manuals/376047c.pdf. [Último acceso: 8 Septiembre 2020].

[24] MathWorks, «What is MATLAB?,» 2019. [En línea]. Available: https://www.mathworks.com/discovery/what-is-matlab.html.

[25] MathWorks, «MATLAB Compiler Documentation - MathWorks America Latina,» 2019. [En línea]. Available: https://www.mathworks.com/help/compiler/index.html.

[26] MathWorks, «MATLAB Compiler SDK,» 2019. [En línea]. Available: https://www.mathworks.com/help/compiler_sdk/index.html.

[27] MathWorks, «Myo SDK MATLAB MEX Wrapper,» 2019. [En línea]. Available: https://www.mathworks.com/matlabcentral/fileexchange/55817-myo-sdk-matlab-mex-wrapper.

[28] A. Martínez, Learning ROS for Robotics Programming, Reino Unido: Packt Publishing , 2013.

[29] M. Electric, «RV-2F-D Series Standard Specifications Manual,» 2018. [En línea]. Available: https://dl.mitsubishielectric.com/dl/fa/document/manual/robot/bfp-a8900/bfp-a8900aa.pdf. [Último acceso: 8 Septiembre 2020].

[30] C. R. Romeva, Diseño concurrente, Barcelona: UPC, 2020, pp. 126-139.

[31] A. S. Anillo, Matriz de electrodos EMG para detección de intención de movimiento de la mano, Madrid, 2017.

[32] J. Ruano, Sisitema de visión artificial para autonomía de acción del robot industrial Mitsubishi RV-2F., Ibarra, 2020.

ANEXO A

Instanciación de MyoMex e interpretación de gestos

```
%% Especificando la cantidad de MYO's a usar

countMyos = 1;

%% Instanciando MyoMex e inspeccionando las propiedades de MyoMex

mm = MyoMex(countMyos); %Su única propiedad será un objeto tipo myoData de
(1xcountMyos).

%% Inspeccionando MyoData

m1 = mm.myoData(1); %representamos el dispositivo físico en m1.
% if countMyos == 2, m2 = mm.Myodata(2); end % En el caso de usar dos MYOs.

%  Los datos más recientes de Myo se almacenarán en las propiedades
%  relevantes del objeto |MyoData| (|quat|, |gyro|, |accel|, |emg|, |pose|,
%  etc.). la siguiente es una lista de todas las propiedades en |MyoData|:

pause (0.1) %Tiempo de espera para la llegada del primer marco de datos.

%**** Propiedades de datos muestreados en la base de tiempo IMU (50 hz)
****

TiempoIMU           = m1.timeIMU;
Quat                = m1.quat; %Cuaternión que representa la orientación
del Myo
Matriz_de_rotacion  = m1.rot;
Giroscopio          = m1.gyro;
Giroscopio_fijo     = m1.gyro_fixed;
Acelerometro        = m1.accel;
Acelerometro_fijo   = m1.accel_fixed;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%*** Propiedades de datos muestreados en la base de tiempo EMG (200 Hz) ***

TiempoEMG               = m1.timeEMG;
EMG                     = m1.emg; %Datos sin procesar de los 8 sensores de
superficie
Pose                    = m1.pose;
Pose_descanso           = m1.pose_rest;
Pose_puno               = m1.pose_fist;
Pose_palma_dentro       = m1.pose_wave_in;
Pose_palma_fuera        = m1.pose_wave_out;
Pose_dedos_abiertos     = m1.pose_fingers_spread;
Pose_doble_toque        = m1.pose_double_tap;
Brazo                   = m1.arm;
Brazo_izquierdo         = m1.arm_left;
Brazo_derecho           = m1.arm_right;
Brazo_desconocido       = m1.arm_unknown;
Direccion_x             = m1.xDir; %Indica la dirección del eje x del
sensor Myo en el brazo del usuario
Direccion_x_muneca      = m1.xDir_wrist;
Direccion_x_codo        = m1.xDir_elbow;
Direccion_x_desconocido = m1.xDir_unknown;


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%% Usando datos registrados

% Como | MyoData | recibe datos de | MyoMex |, se acumulan automáticamente
% en las llamadas propiedades | <data> _log | , es decir, | quat_log |,
% | accel_log |, etc. Nos referimos a esto como el modo de transmisión de
un
% Objeto| MyoData |. Este estado es indicado por la propiedad | isStreaming
|

Transmitiendo = m1.isStreaming;

%--------------- Inspeccionando la acumulación de registros ---------------

fprintf('%8s%13s\n','tiempo','muestras');
for ii = 1:5
  fprintf('% 8.2f%10d\n',m1.timeIMU,size(m1.quat_log,1));
  pause(0.2);
end
fprintf('\n\n');

%--------------------------------------------------------------------------


%------ Cambiando el modo de transmisión para que no pase a MyoData -------


% Aunque no podemos evitar que los datos pasen a |MyoData|, podemos
% cambiar el modo de transmisión mediante los métodos |stopStreaming()| y
% |startStreaming()|.

m1.stopStreaming();
fprintf('Numero de muestras:                \t%d\n',length(m1.timeIMU_log));
pause(1);
fprintf('Numero de muestras después de
pause(1):\t%d\n',length(m1.timeIMU_log));

%% Graficando los datos

%se deben tener en cuenta los vectores correctos de tiempo (IMU o EMG)
figure;
subplot(3,1,1); plot(m1.timeIMU_log,m1.gyro_log);  title('giroscopio');
subplot(3,1,2); plot(m1.timeIMU_log,m1.accel_log); title('aceleración');
subplot(3,1,3); plot(m1.timeEMG_log,m1.emg_log);   title('emg');

%% Recolección de datos en T segundos

T = 5; % Segundos
m1.clearLogs() % Borra las propiedades del data_log para iniciar una nueva
prueba de registro
m1.startStreaming();
pause(T);
m1.stopStreaming();
fprintf('Datos registrados para %d segundos\n\t',T);

fprintf('IMU muestras: %5d\tApprox.   IMU frecuencia de muestreo:
%5.2f\n\t',...
  length(m1.timeIMU_log),length(m1.timeIMU_log)/T);
```

```
fprintf('EMG muestras: %5d\tApprox.   EMG frecuencia de muestreo:
%5.2f\n\t',...
  length(m1.timeEMG_log),length(m1.timeEMG_log)/T);

%% Limpiando los datos para terminar

mm.delete; %Borra los datos
% clear mm;  %Borra la variable
% clear all;
% clc
```

ANEXO B

Sistema de trayectorias, lenguaje

MELFA BASIC V

```
1 Ovrd 30
2 M_Out(6)=0
3 P13=P1
4 *Inicio
5 On M_In(13)=1 GoSub *Sub1
6 On M_In(10)=1 GoSub *Sub2
7 On M_In(12)=1 GoSub *Sub3
8 On M_In(11)=1 GoSub *Sub4
9 GoTo *Inicio
10 *Sub1
11 Ovrd 30
12 Mov P1
13 Mov P2, -50
14 Ovrd 10
15 Mvs P2
16 Dly 1.0
17 M_Out(6)=1
18 Dly 1.0
19 Mvs P2, -50
20 Ovrd 30
21 Mov P1
22 Mov P3, -50
23 Ovrd 10
24 Mvs P3
25 Dly 1.0
26 M_Out(6)=0
27 Dly 1.0
28 M_Out(6)=1
29 Dly 1.0
30 Mvs P3, -50
31 Ovrd 30
32 Mov P1
33 Mov P2, -50
34 Mov P2
35 Dly 1.0
36 M_Out(6)=0
37 Dly 1.0
38 Mov P2, -50
39 Mov P1
40 Return
41 *Sub2
42 Ovrd 30
43 Mov P1
44 Mov P4, -50
```

```
45 Ovrd 10
46 Mvs P4
47 Dly 1.0
48 M_Out(6)=1
49 Dly 1.0
50 Mvs P4, -50
51 Ovrd 30
52 Mov P1
53 Mvc P1, P5, P6
54 Mov P4, -50
55 Ovrd 10
56 Mvs P4
57 Dly 1.0
58 M_Out(6)=0
59 Dly 1.0
60 Mvs P4, -50
61 Ovrd 30
62 Mov P1
63 Return
64 *Sub3
65 Ovrd 30
66 Mov P1
67 Mov P14, -50
68 Ovrd 10
69 Mvs P14
70 Dly 1.0
71 M_Out(7)=1
72 Dly 1.0
73 M_Out(7)=0
74 Dly 1.0
75 Mvs P14, -50
76 Ovrd 30
77 Mov P1
78 Return
79 *Sub4
80 On M_In(10)=1 GoSub *Sub5
81 On M_In(12)=1 GoSub *Sub6
82 GoTo *Sub4
83 *Sub5
84 Ovrd 30
85 Mov P1
86 On M_In(10)=1 GoSub *Sub7
87 On M_In(11)=1 GoSub *Sub8
88 On M_In(12)=1 GoSub *Sub9
```

```
89 On M_In(13)=1 GoTo *Sub6
90 GoTo *Sub5
91 *Sub7
92 Mov P7, -50
93 Mov P7
94 Wait M_In(10)=1
95 Dly 1.0
96 M_Out(6)=1
97 Mov P7, -50
98 Mov P1
99 Mov P10,-50
100 Mov P10
101 Wait M_In(10)=1
102 M_Out(6)=0
103 Dly 1.0
104 Mov P10, -50
105 Mov P1
106 GoTo *Sub5
107 *Sub8
108 Mov P8, -50
109 Mov P8
110 Wait M_In(10)=1
111 Dly 1.0
112 M_Out(6)=1
113 Mov P8, -50
114 Mov P1
115 Mov P11,-50
116 Mov P11
117 Wait M_In(10)=1
118 M_Out(6)=0
119 Dly 1.0
120 Mov P11, -50
121 Mov P1
122 GoTo *Sub5
123 *Sub9
124 Mov P9, -50
125 Mov P9
126 Wait M_In(10)=1
127 Dly 1.0
128 M_Out(6)=1
129 Mov P9, -50
130 Mov P1
131 Mov P12,-50
132 Mov P12
```

```
133 Wait M_In(10)=1
134 M_Out(6)=0
135 Dly 1.0
136 Mov P12, -50
137 Mov P1
138 GoTo *Sub5
139 *Sub6
140 Dly 1
141 On M_In(10)=1 GoSub *Sub10
142 On M_In(11)=1 GoSub *Sub11
143 On M_In(12)=1 GoSub *Sub12
144 On M_In(13)=1 GoSub *Sub13
145 GoTo *Sub6
146 *Sub10
147 On M_In(11)=1 GoSub *Sub14
148 On M_In(12)=1 GoSub *Sub15
149 On M_In(13)=1 GoTo *Sub6
150 GoTo *Sub10
151 *Sub14
152 Ovrd 30
153 P13.X=P13.X+1
154 Mov P13
155 Return
156 *Sub15
157 Ovrd 30
158 P13.X=P13.X-1
159 Mov P13
160 Return
161 *Sub11
162 On M_In(11)=1 GoSub *Sub16
163 On M_In(12)=1 GoSub *Sub17
164 On M_In(13)=1 GoTo *Sub6
165 GoTo *Sub11
166 *Sub16
167 Ovrd 30
168 P13.Y=P13.Y+1
169 Mov P13
170 Return
171 *Sub17
172 Ovrd 30
173 P13.Y=P13.Y-1
174 Mov P13
175 Return
176 *Sub12
```

```
177 On M_In(11)=1 GoSub *Sub18
178 On M_In(12)=1 GoSub *Sub19
179 On M_In(13)=1 GoTo *Sub6
180 GoTo *Sub12
181 *Sub18
182 Ovrd 30
183 P13.Z=P13.Z+1
184 Mov P13
185 Return
186 *Sub19
187 Ovrd 30
188 P13.Z=P13.Z-1
189 Mov P13
190 Return
191 *Sub13
192 Ovrd 30
193 Mov P1, -50
194 Mov P1
195 End
P13=(+248.88,-2.63,+453.45,+179.39,-1.50,+178.77)(7,0)
P1=(+248.88,-2.63,+453.45,+179.39,-1.50,+178.77)(7,0)
P2=(+170.25,-241.29,+340.87,+179.39,-1.50,+178.77)(7,0)
P3=(+169.80,+205.69,+340.08,+179.39,-1.50,+178.77)(7,0)
P4=(+167.25,-288.28,+340.17,+179.39,-1.50,+178.77)(7,0)
P5=(+298.88,-52.63,+453.45,+179.39,-1.50,+178.77)(7,0)
P6=(+348.88,-2.63,+453.45,+179.39,-1.50,+178.77)(7,0)
P14=(+376.87,-28.63,+387.41,-179.92,-1.56,+179.52)(7,0)
P7=(+169.00,-287.25,+342.40,-179.92,-1.56,+179.52)(7,0)
P10=(+170.89,+257.14,+341.25,-179.92,-1.56,+179.52)(7,0)
P8=(+118.71,-287.25,+338.17,-179.92,-1.56,+179.52)(7,0)
P11=(+119.74,+257.14,+338.56,-179.92,-1.56,+179.52)(7,0)
P9=(+67.28,-235.50,+339.25,-179.92,-1.56,+179.52)(7,0)
P12=(+70.44,+207.85,+339.26,-179.92,-1.56,+179.52)(7,0)
```

ANEXO C

Envío de interpretación de gesto

Matlab-Arduino-Controlador

```
%% Especificando la cantidad de MYO's a usar y conexión con arduino

countMyos = 1;
a=arduino();

%% Instanciando MyoMex e inspeccionando las propiedades de MyoMex

mm = MyoMex(countMyos); %Su única propiedad será un objeto tipo myoData de
(1xcountMyos).

%% Inspeccionando MyoData

m1 = mm.myoData(1); %representamos el dispositivo físico en m1.
% if countMyos == 2, m2 = mm.Myodata(2); end % En el caso de usar dos MYOs
%  Los datos más recientes de Myo se almacenarán en las propiedades
%  relevantes del objeto |MyoData| (|quat|, |gyro|, |accel|, |emg|, |pose|,
%  etc.). la siguiente es una lista de todas las propiedades en |MyoData|:

pause (0.1) %Tiempo de espera para la llegada del primer marco de datos.

%**** Propiedades de datos muestreados en la base de tiempo IMU (50 hz)
****

TiempoIMU           = m1.timeIMU;
Quat                = m1.quat; %Cuaternión que representa la orientación
del Myo
Matriz_de_rotacion  = m1.rot;
Giroscopio          = m1.gyro;
Giroscopio_fijo     = m1.gyro_fixed;
Acelerometro        = m1.accel;
Acelerometro_fijo   = m1.accel_fixed;

%************************************************************************

proba=1;
while proba==1
TiempoEMG               = m1.timeEMG;
EMG                     = m1.emg; %Datos sin procesar de los 8 sensores de
superficie
Pose                    = m1.pose;
Pose_descanso           = m1.pose_rest;
Pose_puno               = m1.pose_fist;
Pose_palma_dentro       = m1.pose_wave_in;
Pose_palma_fuera        = m1.pose_wave_out;
Pose_dedos_abiertos     = m1.pose_fingers_spread;
Pose_doble_toque        = m1.pose_double_tap;
Brazo                   = m1.arm;
Brazo_izquierdo         = m1.arm_left;
Brazo_derecho           = m1.arm_right;
Brazo_desconocido       = m1.arm_unknown;
Direccion_x             = m1.xDir; %Indica la dirección del eje x del
sensor Myo en el brazo del usuario
Direccion_x_muneca      = m1.xDir_wrist;
Direccion_x_codo        = m1.xDir_elbow;
Direccion_x_desconocido = m1.xDir_unknown;


%PinLed='D8';
```

```
    if Pose_puno == 1
        writeDigitalPin(a,'D4',1);
    else
        writeDigitalPin(a,'D4',0);
    end

    if Pose_palma_dentro == 1
        writeDigitalPin(a,'D5',1);
    else
        writeDigitalPin(a,'D5',0);
    end

    if Pose_palma_fuera == 1
        writeDigitalPin(a,'D6',1);
    else
        writeDigitalPin(a,'D6',0);
    end

    if Pose_dedos_abiertos == 1
        writeDigitalPin(a,'D7',1);
    else
        writeDigitalPin(a,'D7',0);
    end

    if Pose_doble_toque == 1
        writeDigitalPin(a,'D4',0);
        writeDigitalPin(a,'D5',0);
        writeDigitalPin(a,'D6',0);
        writeDigitalPin(a,'D7',0);
        proba=0;
    end

    if Pose_descanso == 1
        writeDigitalPin(a,'D4',0);
        writeDigitalPin(a,'D5',0);
        writeDigitalPin(a,'D6',0);
        writeDigitalPin(a,'D7',0);
    end

end
```