Trabajo Fin de Grado
Grado en Ingeniería Electrónica, Robótica y Mecatrónica

Diseño y control de una prótesis de mano de bajo coste

Autor: Federico Vaz Fernández

Tutores: Juana María Mayo Núñez y Joaquin Ojeda Granja

Dpto. Ingeniería Mecánica y Fabricación
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2020

Trabajo Fin de Grado
Grado en Ingeniería Electrónica, Robótica y Mecatrónica

Diseño y control de una prótesis de mano de bajo coste

Autor:
Federico Vaz Fernández

Tutores:
Juana María Mayo Núñez y Joaquin Ojeda Granja
Catedrática de universidad y profesor contratado doctor

Dpto. Ingeniería Mecánica y Fabricación
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2020

Trabajo Fin de Grado: Diseño y control de una prótesis de mano de bajo coste

Autor: Federico Vaz Fernández
Tutores: Juana María Mayo Núñez y Joaquin Ojeda Granja

El tribunal nombrado para juzgar el trabajo arriba indicado, compuesto por los siguientes profesores:

Presidente:

Vocal/es:

Secretario:

acuerdan otorgarle la calificación de:

El Secretario del Tribunal

Fecha:

Agradecimientos

Quiero aprovechar estas líneas para agradecerle a toda mi familia por el cariño, paciencia y apoyo que han tenido conmigo durante los duros años de carrera. Sobre todo, quisiera agradecerle a mi abuelo Agustín Fernández el iniciarme de pequeño en el mundo de la mecánica y electrónica. Sin él, no estaría donde estoy a día de hoy.

He de agradecerles a mis amigos de la carrera, pues sin la ayuda y apoyo de ellos, la carrera no habría sido tan maravillosa experiencia. Además, he de agradecer a mis amigos de Huelva, pues sin la felicidad que estos me aportan no podría aguantar la carga de trabajo de la carrera.

También he de agradecerle a mis tutores Juana María Mayo Núñez y Joaquín Ojeda Granja por su ayuda y atención, sobre todo con la situación de cuarentena que envuelve a todo el trabajo, siendo un trabajo mayormente práctico. Sin su ayuda no podría haber hecho nada de este. Del mismo modo, todo el profesorado que he tenido en la carrera merece mi más sincero agradecimiento por su docencia.

Desde todo mi corazón, gracias a todos.

Federico Vaz Fernández
Sevilla, 2020

Resumen

Este proyecto muestra el desarollo y control de una prótesis de bajo coste para la mano derecha.

Comienza con la impresión 3D de la prótesis, con diferentes materiales y perfiles de impresión. En primer lugar, se describen las impresoras utilizadas para la impresión así como el programa para crear los archivos que utilizarán. Seguidamente, se detallan los perfiles de impresión para cada material y pieza, finalizando con un breve análisis mecánico de la prótesis donde se comparará a una mano biológica en cuanto a grados de libertad.

En segundo lugar, se estudiará el uso de dos sensores EMG completamente diferentes, uno de muy bajo coste y otro de alta precisión. El primero consiste en la medida EMG de un músculo junto a un posterior procesamiento de esta medida, resultando en una señal apta para microcontroladores con entradas analógicas, mientras que el segundo ofrece una señal completamente fiel a la medida EMG del sensor.

En tercer lugar, puesto que los sensores adquieren medidas de músculos cercanos a donde esté colocado, ha de separarse cuándo hay intención de movimiento para un dedo o no. Es por ello que se propone un método de clasificación para las señales EMG del brazo, realizado con el sensor de alta precisión, pero extrapolable para varios sensores de baja precisión.

A continuación, se incluye el diseño e impresión de un antebrazo para que la prótesis impresa se pueda ajustar a las necesidades del sujeto. En este antebrazo se incluyen el microcontrolador, el sensor, las baterías y los actuadores del sistema, de manera que el antebrazo proteja a todos los elementos de colisiones internas y externas.

Además, se describen los modos de funcionamiento programados para la actuación de la prótesis. El primero de estos consiste en controlar todos los dedos simultáneamente en función de si se detecta entrada del sensor EMG por encima de un valor umbral, mientras que el segundo propone manejar cada dedo mediante un contador de pulsos en un intervalo de tiempo.

Finalmente, se incluye un análisis de los costes de producción del proyecto, tanto de la prótesis como de la electrónica que se incluye.

Abstract

This project shows the development and control of a low cost right hand prosthesis.

It begins with the 3D printing of the prosthesis, using various materials and printing profiles, firstly describing the 3D printers and the software used. Afterward, the printing profiles for each material and part will be detailed, finishing with a brief prosthesis mechanic analysis contrasting it with a biologic hand, in therms of degree of freedom.

Secondly, two different EMG sensors will be studied, some of very low cost and others of high precision. The fist one consists of the EMG measure and a subsequent signal processing, resulting in a signal which is capable to be measured in a microcontroller witch analogical inputs, while the second one presents a raw EMG signal.

Thirdly, due to the surrounding EMG measures of a sensor, it has to be divided when there is intention to move a finger or not. Because of that, a classification method for EMG arm signals, made with the high-precision sensor but transferable to some low cost sensors, will be proposed.

In addition, it is included the design and printing of a forearm so that the printed prosthesis can be adjusted to the user needs. Inside this forearm there is included the microcontroller, the EMG sensor, the batteries and the system actuators, so the forearm protects all the elements against internal and external collisions.

It is also described the operating modes programmed for the prosthesis actuation. The first one consists of controlling all fingers at the same time, depending if it is detected a EMG measure which is bigger than the threshold value, while the second one purposes to handle each finger through a pulse counter and a timeout.

Finally, an analysis of the production costs of the project is included, both fot prosthesis and electronics.

Índice Abreviado

Índice

Notación

CNC	control numérico por computador
PLA	ácido poliláctico
PVA	acetato de polivinilo
ABS	acrilonitrilo butadieno estireno
TPE	elastómero termoplástico
GDL	grado de libertad
EMG	electromiografía, electromiográfica
sEMG	electromiografía, electromiográfica en superficie
MDF	modelado por deposición fundida
CMRR	rechazo al modo común
STL	archivo de objeto 3D
RMS	valor cuadrático medio (del inglés root mean square)
MAV	valor medio absoluto (del inglés medium absolute value)
MDF	frecuencia mediana
MEF	frecuencia media
PSD	densidad espectral de potencia (del inglés power spectral density)
CAD	diseño asistido por computador (del inglés computer assisted design)
PCB	circuito impreso
OPAM	amplificador operacional (del inglés operational amplifier)
PWM	modulación por ancho de pulsos (del inglés Pulse-Width Modulator)

1 Introducción

La ingeniería siempre ha buscado satisfacer necesidades y resolver problemas de la sociedad. Es por ello que han surgido a lo largo de los años ramas de la ingeniería aplicadas a las ciencias de la salud, como la ingeniería genética, la ingeniería biomédica o la ingeniería biónica.

Esta última engloba a dos ramas, una con el fin de simular el comportamiento de seres vivos, y otra con el fin de combinar biología y electrónica para mejorar las capacidades de los seres vivos [1].

La ingeniería biónica combina disciplinas como mecatrónica, robótica y biología. Este trabajo es un claro ejemplo de la aplicación conjunta de estas ramas de conocimiento, centrándose en especial en la mecatrónica y adquisición de señales.

1.1 Motivación y Necesidad

El número de personas con miembros amputados crece año tras año, ya sea por enfermedades tales como diabetes, por accidentes de diversos tipos, por herencia o por guerras.

Se estima que el 30 % de las personas que sufren de esta discapacidad parcial padecen depresión y problemas de ansiedad, y que el coste de los cuidados y salud se duplica con respecto a una persona sin esta discapacidad, llegando a costar el total de pacientes en un año tan sólo en Estados Unidos (185000 pacientes) unos 8 billones de dólares al año[2].

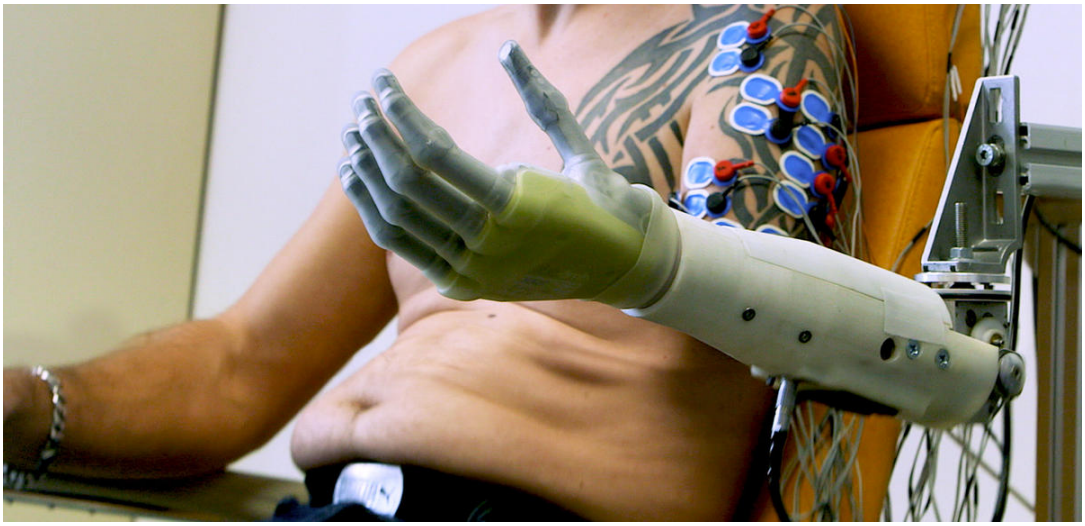


Figura 1.1 prótesis de codo, antebrazo y mano conectada a sensores mioeléctricos. fuente: www.cnrs.fr.

Además, el porcentaje de miembros superiores, según diferentes estudios, suele rondar entre el 27 % y 56 % [3], mientras que las prótesis de miembros inferiores ocupan entre el 49 % y 95 % [3] de los casos.

Según la OMS [4], se estima que el 0.5 % de la población mundial tiene necesidad de prótesis, por lo que estima que en África, Asia y Latinoamérica, habría unos 25 millones de personas que no tendrían esta necesidad solventada.

Es por ello que surge la motivación por parte de numerosos grupos de investigación y empresas para desarrollar prótesis que satisfagan esta necesidad, minimizando el coste para que pueda expandirse por todo el mundo, mediante impresión 3D o diversos usos de materiales. Es aquí donde se encuentra mi motivación personal por el trabajo, pues me gustaría aplicar los conocimientos de la carrera para ayudar al máximo número de personas posibles, y este campo es seguramente uno de los más adecuados para ello.

Además, resulta bastante útil que la prótesis esté actuada. Aunque por ahora no se consiga la funcionalidad completa de una mano biológica, es conveniente que la prótesis permita agarrar objetos, utilizar herramientas, aguantar un determinado peso y tener un tamaño similar al miembro original.

1.2 Objetivo y metodología

El objetivo de este trabajo es la correcta impresión, estableciendo unos perfiles adecuados para conseguir imprimir una prótesis de mano de diversos materiales con un coste muy bajo y la sensorización y actuación sobre esta, tal que se puedan mover los dedos para agarrar objetos.

La prótesis será actuada mediante servomotores que tirarán de unos hilos de nylon atados a las puntas de los dedos, por lo que la fuerza que se podrá ejercer sobre los objetos dependerá directamente del par que nos ofrezcan estos motores.

Dichos motores se activarán mediante señales electromiográficas de los músculos del antebrazo, aunque pueden ajustarse a cualquier músculo y necesidad. Se compararán los resultados obtenidos con sensores de alta precisión *delsys® trigno system* frente a los obtenidos con un dispositivo amplificador y rectificador básico.

La adquisición de dichas señales se realizará con sensores de superficie, por lo que las interferencias entre músculos han de tenerse en cuenta. Para ello se realizará un clasificador lineal que identifique y clasifique los movimientos, aunque no se llegue a implementar en la versión final del trabajo.

Para tener un uso cómodo, es necesario diseñar e imprimir un elemento que albergue los motores, el microprocesador (arduino nano), baterías y demás elementos electrónicos de manera que no estorben al usuario. Para el diseño se usará el programa *SolidWorks®*.

Finalmente, se hará un estudio de los costes de la prótesis con objeto de apreciar la rentabilidad para una empresa al implementar este tipo de proyectos.

1.3 Estado del Arte

1.3.1 Breve historia de las prótesis

Se ha descubierto, gracias a la momia de un sacerdote egipcio [5], que las primeras prótesis datan del milenio 1 a.C. siendo esta un dedo del pie, sin funcionalidad aparente salvo la estética.

Hay hallazgos de una prótesis de la mitad inferior de la pierna que data del año 300 a.C. aproximadamente, fabricada con madera, hierro y bronce, capaz de permitir andar a una persona. También data del 210 a.C. aproximadamente una prótesis de brazo capaz de soportar el peso de un escudo.

No fue hasta el Renacimiento que comenzó a desarrollarse la funcionalidad y diseño de estas, creando dedos articulables que podían fijarse con ayuda de la otra mano, permitiendo al usuario coger objetos y tener una limitada maniobrabilidad.

Una figura importante fue Ambrosio Paré, uno de los cirujanos más importantes de la historia, que diseñaba y construía prótesis para los heridos en guerra del siglo XVI. Llegó a diseñar una prótesis de pierna con la rodilla flexionable y con posibilidad de fijar el pie de esta, siendo el primero en darle utilidad práctica a estas.

En los siglos posteriores, hubo numerosas mejoras tanto en el manejo, calidad y utilidad de las prótesis como en técnicas de amputación que permitiesen mejor control de las prótesis.

Además, se desarrolló bastante el aspecto de estas, añadiendo materiales y funcionalidades que lograsen unas prótesis más naturales y ligeras.

La guerra civil estadounidense supuso un avance en cuanto a los materiales de los que se fabricaban las prótesis, la alta demanda supuso cambiar a prótesis de madera y añadir mecanismos a estas.

Desde entonces, no hubo mejoras notables hasta la Segunda Guerra Mundial, donde el gobierno de Estados Unidos decidió cambiar parte de la inversión en armamento por la prostética, añadiendo plástico a los materiales [6]. Por último, se ha añadido electrónica y microprocesadores a las prótesis de manera que los pacientes puedan recuperar no solo el aspecto sino el estilo de vida.

1.3.2 Actualidad

En el presente, la investigación en el campo de la ingeniería biónica toma varias vertientes:

- **Mecánica**: Centrada tanto en el diseño y desarrollo de modelos de prótesis y los estudios sobre esta, como la resistencia que oponen sus materiales a esfuerzos y otros experimientos mecánicos. España cuenta con varios departamentos de nuestras universidades que trabajan en este campo a fondo, como el departamento de ingeniería mecánica de la Universidad Jaume I de Castellón, que ha permitido usar su modelo *IMMA hand* 1.2 [7] para este proyecto.
- **Elecrónica**: Estudia la adquisición de las señales EMG, su tratamiento y su digitalización en microprocesadores modernos. Además, suelen combinarse otros sensores tales como acelerómetros y giróscopos para mejorar la precision de las medidas de los sensores EMG. Normalmente los departamentos que trabajan en este campo suelen trabajar conjuntamente con el control de esta. Algunos de los avances más novedosos en este campo son la detección de movimiento voluntario, el mantener una postura fija cuando se desee y la clasificación de movimientos en tiempo real mediante patrones.
- **Control**: Trabajan con prótesis completamente actuadas o subactuadas, con casos de hasta un actuador por articulación de la mano, siendo investigaciones muy costosas pero bastante importantes para el desarrollo de estas. Además, se suele estudiar junto a este caso los modelos

dinámicos de la mano. Destaca entre numerosos departamentos el Instituto de Biorobótica de Santa Ana, en Pisa.

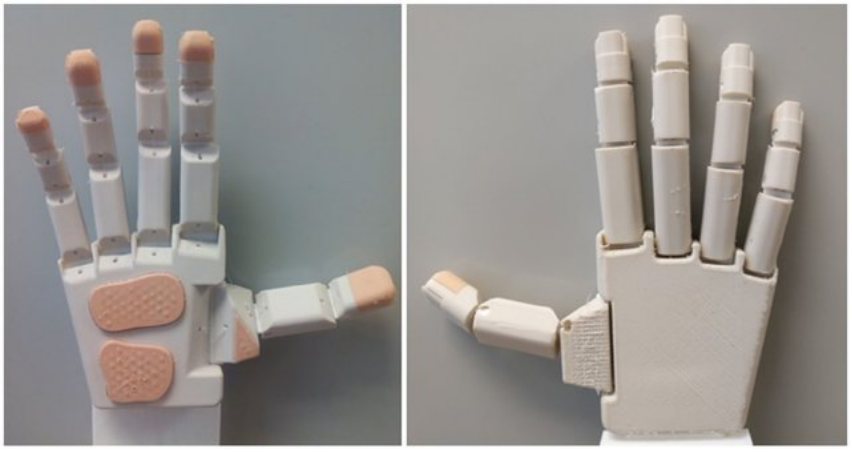


Figura 1.2 prótesis IMMA hand. fuente: System for the experimental evaluation of anthropomorphic hands. Application to a new 3D-printed prosthetic hand prototype. 2017.

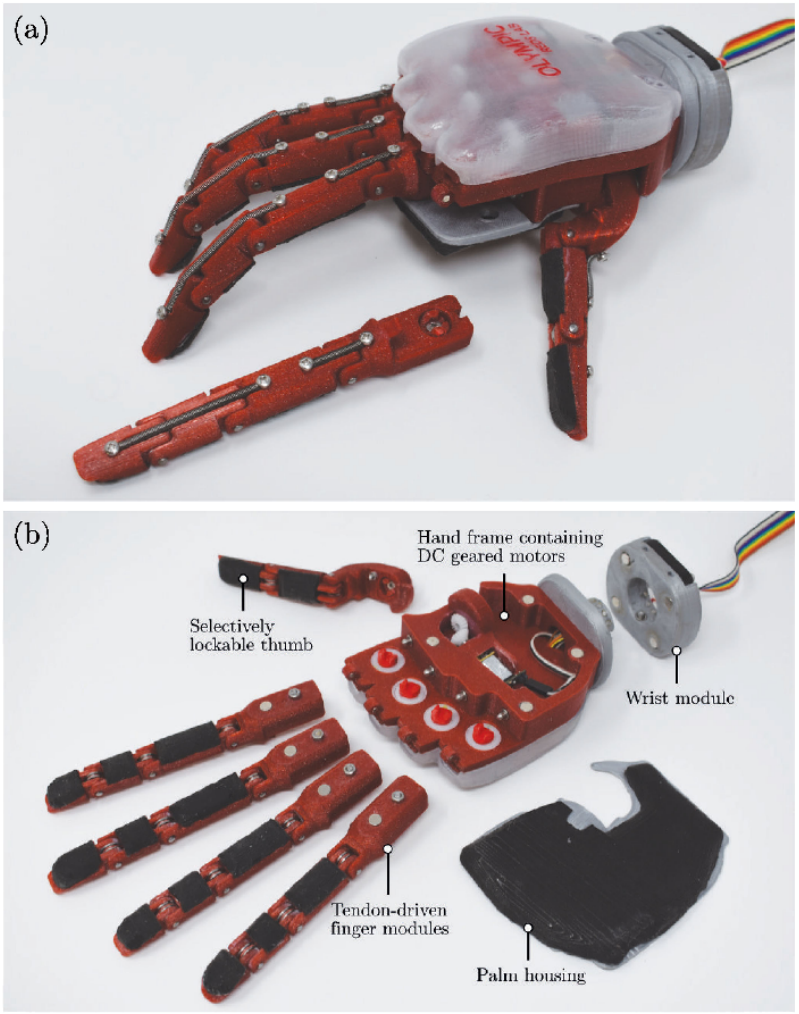


Figura 1.3 prótesis OLYMPIC hand. [8].

2 Impresión 3D de la prótesis

Para la fabricación de la prótesis se han utilizado técnicas de impresión 3D de modelado por deposición fundida, cuyas ventajas frente a otras técnicas, como la impresión 3D por fotopolimerización, son notables en cuanto a costes.

La técnica por deposición fundida consiste en añadir capas de material, una encima de otra, hasta obtener una pieza completa. Las capas se componen de un filamento de material que, al llegar a la boquilla a una temperatura mayor de la fusión del material, crea pequeños hilos de material [9]. Para mover dicha boquilla se dispone de un servomotor por cada eje XYZ, variando la disposición de los servomotores según el fabricante.

2.1 Impresoras Utilizadas

2.1.1 Ultimaker 3 extended

La impresora Ultimaker 3 extended es uno de los modelos que dispone Ultimaker, una de las más famosas empresas del sector. Es una impresora de uso profesional que cuenta con dos motores extrusores situados detrás, es decir, puede imprimir con dos materiales simultáneamente.

Además, cuenta con un volumen de 215x215x300 mm y una velocidad máxima de 300 mm/s. Una característica de esta impresora es que imprime de arriba a abajo, es decir, la plataforma donde depositamos el material baja a medida que este se deposita, de manera que el movimiento en eje z está asociado a esta plataforma y los desplazamientos en el plano XY se controlan desde dos ejes que atraviesan la zona de los extrusores.

Figura 2.1 Ultimaker 3 extended. fuente: 3dnatives.com.

2.1.2 Creality ender 3

Creality fabrica impresoras de bajo coste y de uso personal. Cuenta con solo un extrusor, únicamente imprime con polímeros.

Frente a la Ultimaker 3 extended, esta cuenta con tan solo un volumen de 220x220x250 mm y velocidad máxima de 180 mm/s, lo cual la sitúa muy por debajo de la Ultimaker 3 extended en cuanto a prestaciones, pero teniendo un coste considerablemente menor. Además, la distribución de los servomotores es diferente, en este caso la plataforma es la que recibe el movimiento del eje Y, mientras que X y Z se asocian al movimiento del extrusor.

Figura 2.2 Creality ender 3. fuente:3dnatives.com.

2.2 Ultimaker Cura

La impresora recibe un archivo de tipo *.gcode*, que consiste en un conjunto de órdenes CNC, coordenadas para cada eje y extrusor, los cuales la impresora se encarga de convertir en pasos para los servomotores. Para generar ese archivo, se necesita un programa que permita transformar un objeto 3D *.stl* en ese conjunto de instrucciones. Este tipo de programas son denominados *slicer*, puesto que "rebanan" la pieza en capas.

De entre todos los programas que hay con este propósito, se ha elegido Cura (de Ultimaker) por ser de los más utilizados en el mercado y compatible con la mayoría de impresoras y estar familiarizado previamente con su entorno de trabajo.

El entorno de Cura cuenta con la representación del espacio exacto de impresión, de manera que al situar una pieza se pueda preveer dónde se situará esta dentro de la impresora. Además, para editar la impresión, permite desplazarla, modificar su tamaño, girarla y crear soportes o eliminarlos, voluntariamente.

Por otro lado, permite modificar los ajustes de la impresión con los denominados "perfiles de impresión" y, una vez listos, permite tener una vista preliminar de la pieza a imprimir. En esta vista se puede apreciar capa por capa lo que extruirá la impresora.

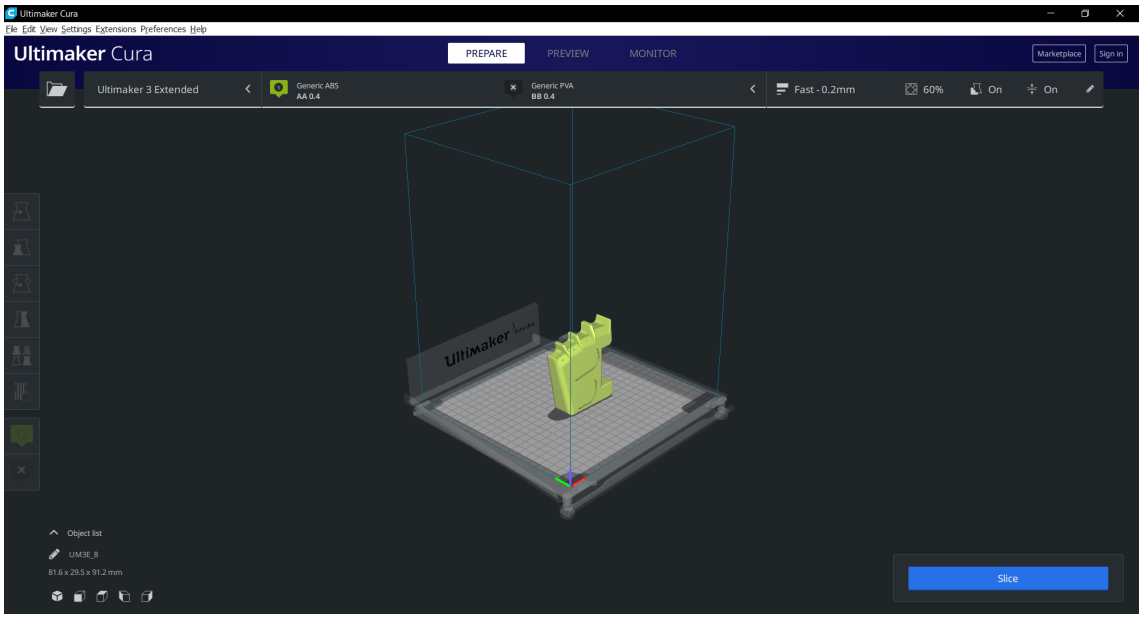
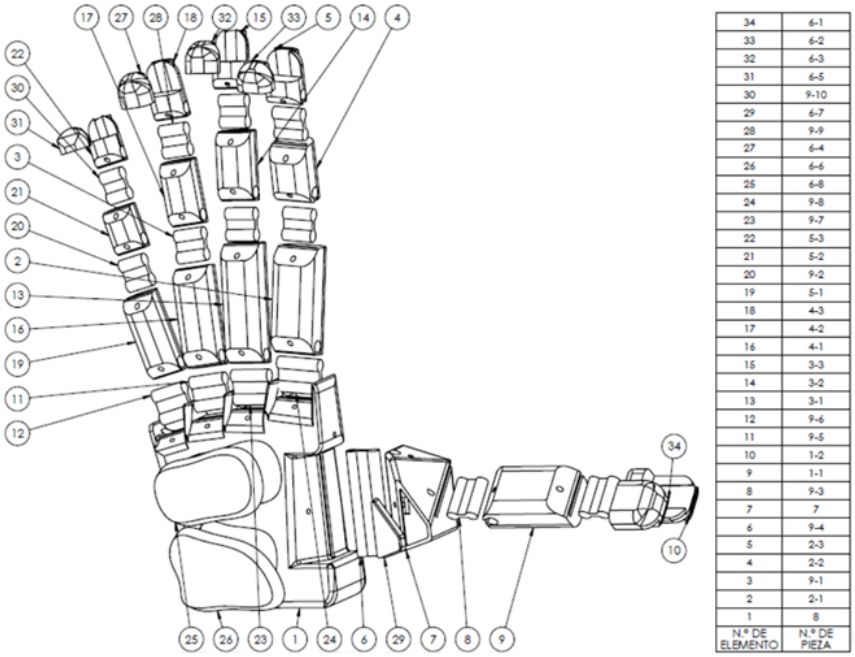


Figura 2.3 Entorno de Ultimaker Cura..

2.3 Modelo de la prótesis

El modelo de la prótesis está creado por Inmaculada Llop-Harillo de la Universitat Jaume I. Se ha decidido utilizar este modelo por las consideraciones que ha tenido la autora para representar un modelo similar a una mano humana estándar y por los buenos resultados que ofrece para diferentes funciones [10]. El modelo se caracteriza además por contraerse tirando de la última falange de los dedos mediante un hilo (ya sea de plástico o metálico) que es conducido por un pequeño agujero que atraviesa los dedos y la palma, y extenderse gracias a las juntas que unen los dedos.



N.º DE ELEMENTO	N.º DE PIEZA
34	6-1
33	6-2
32	6-3
31	6-5
30	9-10
29	6-7
28	9-9
27	6-4
26	6-6
25	6-8
24	9-8
23	9-7
22	5-3
21	5-2
20	9-2
19	5-1
18	4-3
17	4-2
16	4-1
15	3-3
14	3-2
13	3-1
12	9-6
11	9-5
10	1-2
9	1-1
8	9-3
7	7
6	9-4
5	2-3
4	2-2
3	9-1
2	2-1
1	8

Figura 2.4 Piezas que componen la prótesis. fuente: proyecto DEVALHAND.

2.3.1 Falanges

La prótesis consta de 14 falanges de dedos, como una mano biológica (sin contar las falanges de la palma). Para estas se ha buscado tener la sufiente resistencia, precisión y ligereza. Por ello, se ha decidido imprimirlas con PLA, un biopoliéster termoplástico que forma parte de los a-hidroxi-acidos, basado en el ácido láctico, presentando una gran variedad de propiedades físicas, químicas y mecánicas, además de ser biodegradable bajo ciertas condiciones a temperaturas del orden de 60 °C [11]. Este destaca en el caso del proyecto por tener un coste mínimo por kilogramo de material.

El perfil utilizado en este caso, a grandes rasgos, se define por:

Tabla 2.1 Perfil falanges.

Parámetro	Valor
Impresora	Ultimaker®3 extended
Alto de capa	0.1 mm
Ancho de pared	1.2 mm
Relleno	30 %
Material	PLA (210 ºC)
Velocidad	60 mm/s
Soportes	No

2.3.2 Palma

La palma de la mano se divide en dos partes, el cuerpo de la mano como tal y la unión del pulgar a esta. La impresión de esta pieza ha sido también en PLA, como el resto de falanges, pero con un perfil diferente, buscando menos precisión y mayor robustez.

Además, puesto que las piezas son más grandes y tienen formas menos cilíndricas, se han añadido soportes a la estructura de manera que los hilos de filamento que no tuviesen superficie donde depositarse, reposen sobre una estructura firme. Para estos soportes se ha utilizado el material PVA, un polímero que se obtiene mediante la polimerización del acetato de vinilo, cuya principal ventaja es la solubilidad en agua y la biodegradabilidad.

Se realizaron varias palmas tanto de PLA como de ABS, polímero amorfo que ofrece mejores propiedades físicas que el PLA frente a impactos y esfuerzos mecánicos, trabajando además en un rango más amplio de temperaturas. Sin embargo, para la aplicación del trabajo no resultan características realmente necesarias, y debido a su mayor coste y dificultad a la hora de imprimirlo, se descarta su uso.

Las principales características de este perfil son:

Tabla 2.2 Perfil mano.

Parámetro	Valor
Impresora	Ultimaker®3 extended
Alto de capa	0.2 mm
Ancho de pared	1.8 mm
Relleno	60 %
Material	PLA (210 ºC)
Velocidad	60 mm/s
Soportes	Sí (PVA)

2.3.3 Juntas

Las juntas entre las falanges de la prótesis son las encargadas de llevar a los dedos a su estado de reposo y compensar el par que apliquen los motores sobre cada articulación. Por ello, se requiere un material compatible para MDF que al imprimirse sea elástico, capaz de recuperar su estado original. Para ello se ha utilizado filaflex, un TPE (termoplástico elastómero) con base de poliuretano que consigue resultados elásticos y resistentes.

El perfil utilizado para imprimir filaflex, principalmente es:

Tabla 2.3 Perfil juntas.

Parámetro	Valor
Impresora	Creality®ender 3
Alto de capa	0.1 mm
Ancho de pared	1.2 mm
Relleno	20 %
Material	filaflex
Velocidad	20 mm/s
Soportes	No

2.3.4 Yemas

Para las yemas de los dedos se ha decidido usar nuevamente filaflex, pues conviene que tengan cierta elasticidad para poder aplicar la fuerza de agarre sobre una mayor superficie del objeto. Aunque aparentemente el relleno es mucho mayor al de las juntas, ha de puntualizarse que las juntas tienen suficientes capas de material en las paredes como para no necesitar apenas relleno, mientras que en este caso se tendría que subir bastante este valor para conseguir la consistencia deseada.

El perfil para imprimir las yemas es muy similar al de las juntas:

Tabla 2.4 Perfil yemas.

Parámetro	Valor
Impresora	Creality®ender 3
Alto de capa	0.1 mm
Ancho de pared	1.2 mm
Relleno	70 %
Material	filaflex
Velocidad	15 mm/s
Soportes	No

También se han hecho las yemas de ABS, pues además de la alta resistencia que ofrece el material, tiene un acabado bastante liso y suave. Sin embargo, la sujeción que ofrece filaflex no la puede ofrecer el ABS, por lo que la versión final de la mano no contará con ninguna pieza de ABS.

Tabla 2.5 Perfil yemas ABS.

Parámetro	Valor
Alto de capa	0.1 mm
Ancho de pared	1.3 mm
Relleno	20 %
Material	ABS (230 ºC)
Velocidad	55 mm/s
Soportes	No

Figura 2.5 Resultado de la impresión.

2.4 modelo mecánico

Una mano humana consta de 27 huesos [12], representados en la imagen 2.6.

Cada dedo es formado por tres falanges, menos el pulgar, que tiene dos, y cada dedo consta de cuatro grados de libertad (a excepción del pulgar). La articulación de la falange distal con la falange media (DIP en la imagen) y la articulación de la falange proximal con la falange media (PIP en la imagen) tienen ambas un grado de libertad (felxión), mientras que la articulación de la falange proximal con el hueso metacarpiano (MP en la imagen) consta de dos (flexión y abducción) [12].

El caso del pulgar es muy diferente. Pese a tener una falange menos, tiene cinco GDL. Un GDL

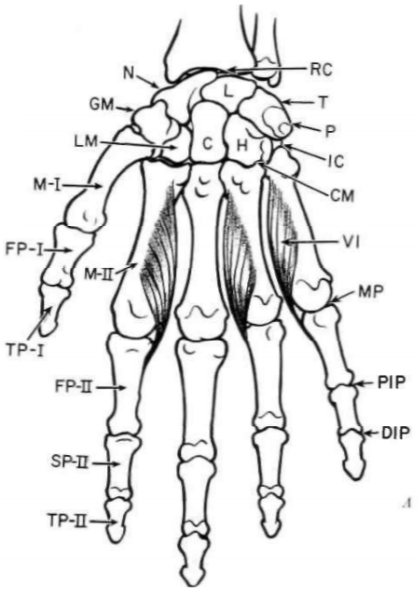


Figura 2.6 Huesos y articulaciones de la mano, incluyendo músculos interóseos [12].

en la flexión entre las dos falanges de este, y dos GDL en cada articulación del hueso metacarpiano, flexión y abducción [13].

En total, contando solamente la estructura de los dedos de la mano, sin contar la muñeca, se obtienen 21 GDL para una mano biológica.

En el caso de la mano IMMA hand, pese a ser una aproximación muy acertada a una mano biológica en tamaño y forma, dista en los grados de libertad que ofrece, pues es una simplificación de una mano real, perdiendo algunos grados de libertad para ser más eficiente en su uso.

Las articulaciones entre cada falange podrían simplificarse por pares de rotación que impliquen un grado de libertad y cinco restricciones cinemáticas.

El gran cambio se aprecia en el pulgar, donde se pasa de contar con cinco grados de libertad a disponer de tres. En el resto de dedos se pierde tan solo un grado de libertad. Tal y como puede suponerse observando la prótesis IMMA, se ha descartado la abducción de los cuatro dedos y las dos abduciones del metacarpiano del pulgar, manteniendo solo las flexiones de cada falange.

Para realizar un análisis cinemático de cada dedo, podría simplificarse un dedo como un mecanismo sencillo de tres barras y tres articulaciones, permitiendo fácilmente el cálculo su modelo cinemático mediante Denavit-Hatemberg [14].

Para ello se utilizarán θ_1 como el valor del álgulo en la articulación del metacarpiano (MP), θ_2 como la falange proximal (PIP), y θ_3 como la falange distal (DIP) , representado en la figura 2.7.

Desde este esquema se puede obtener los parámetros [15]:

	α	a	d	θ
1	L_1	0	0	θ_1
2	L_2	0	0	θ_2
3	L_3	0	0	θ_3

Con dichos parámetros se construyen las matrices de transformación (rotación + traslación) de cada barra, o directamente se forman las matrices homogéneas. Tras obtener dichas matrices, se

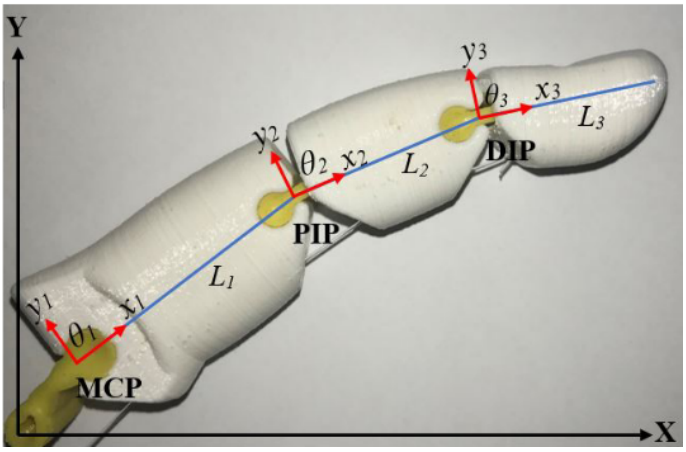


Figura 2.7 Esquema para modelo D-H [15].

multiplican todas y se obtiene una única matriz desde el inicio del mecanismo hasta el final de la última barra, o lo que es equivalente en el caso del proyecto, desde la primera articulación del dedo hasta el extremo de la yema.

La última columna de dicha matriz muestra la traslación desde el inicio del mecanismo hasta el extremo, o lo que es lo mismo, la cinemática directa 2.1. La cinemática directa podría haberse obtenido directamente mediante geometría, pues se trata de un caso bastante simple.

$$\begin{aligned} X &= L_1\cos\theta_1 + L_2\cos(\theta_1+\theta_2) + L_3\cos(\theta_1+\theta_2+\theta_3) \\ Y &= L_1\sin\theta_1 + L_2\sin(\theta_1+\theta_2) + L_3\sin(\theta_1+\theta_2+\theta_3) \end{aligned} \tag{2.1}$$

Sin embargo, como más tarde se repetirá en apartados posteriores, el sistema tan solo cuenta con un grado de actuación por cada dedo, lo que resulta claramente en un sistema subactuado.

3 Sensor EMG de Bajo coste

Uno de los sensores utilizados en el proyecto es un sensor EMG que consiste en un amplificador y rectificador simple de estas señales. Cuando un músculo esquelético se contrae, este genera una pequeña diferencia de potencial entre sus extremos. Esta señal suele tener valores de orden ente microvoltios y el milivoltio, componentes frecuenciales muy altas que introducen ruido en la medida, y oscilación entre valores positivos y negativos (la entrada analógica de Arduino solo admite valores positivos). Es por ello que se necesita una etapa de rectificación, filtrado y amplificación.

Este sensor podría ensamblarse fácilmente, pero por comodidad se ha decidido adquirir uno con las etapas anteriores ya incluidas en un circuito.

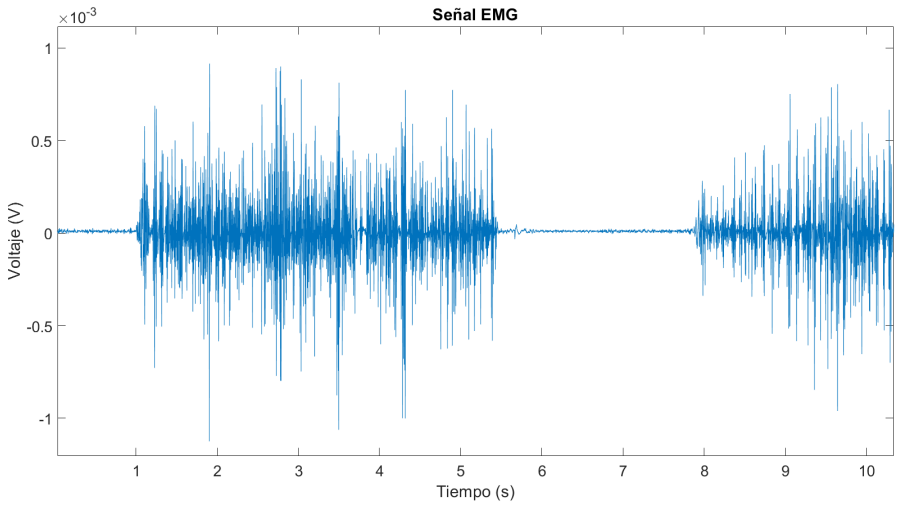


Figura 3.1 Señal EMG sin procesar. [16].

3.1 Componentes del sensor

Para este apartado se ha utilizado la ficha técnica que ha proporcionado el fabricante del sensor.

3.1.1 Etapa de medida

Consiste en un amplificador de instrumentación AD8221 de manera que midiendo en los dos electrodos del músculo y tomando como GND el electrodo de referencia, podemos obtener una señal amplificada. Además, este amplificador puede trabajar hasta 10 kHz sin disminuir apenas su CMRR y, como cualquier amplificador de instrumentación, tiene alta impedancia de entrada, lo que permite mayor seguridad para el organismo que se mida.

Puede ser alimentado desde ±2.3 hasta ±18 voltios. En este caso, tal y como recomienda el fabricante, se alimenta a ±9 V. Además, se puede añadir una resistencia R_G para controlar la ganancia que ofrece el amplificador, siguiendo:

$$G = 1 + (49.4k\Omega / R_G) \approx 207 \tag{3.1}$$

El resultado de la medida es MEASURE.

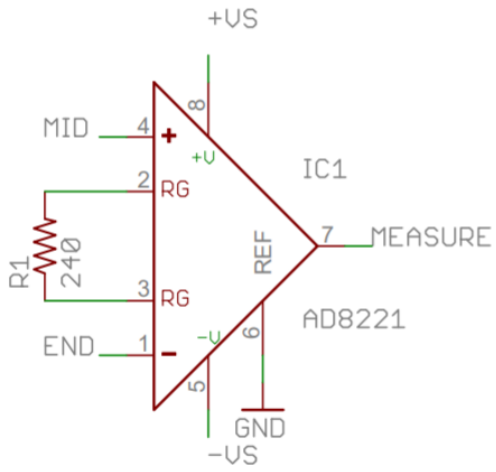


Figura 3.2 Etapa de medida.

3.1.2 Etapa de rectificación

Para que la señal pueda leerse en Arduino, es necesario que sea positiva. Por ello, se necesita una etapa que transforme la señal original medida y amplificada por la etapa de medida en una señal cuyos valores sean mayores o iguales a cero.

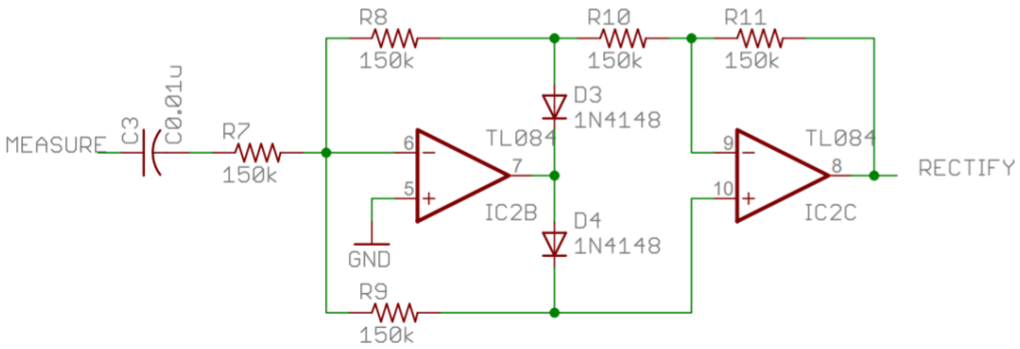


Figura 3.3 Etapa de rectificación.

3.1.3 Etapa de filtrado

Un filtro analógico es un elemento electrónico que modifica las componentes frecuenciales de una señal analógica en función de su frecuencia.

Para reducir el ruido de la señal, se necesita un filtro analógico que deje pasar únicamente bajas frecuencias.

La función de transferencia del filtro es:

$$H(jw) = \frac{-R_{13}/R_2}{R_{13}C_4 j\omega + 1} \tag{3.2}$$

Midiendo la capacidad del condensador, se obtienen 47 nanofaradios. Por lo que a $1/R_{13}C_4$ se calcula la frecuencia de corte, con un valor de 42 Hz aproximadamente.

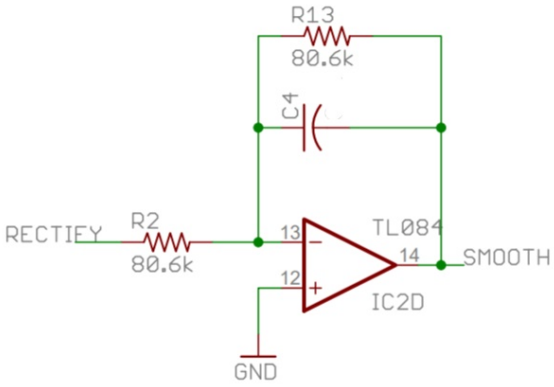


Figura 3.4 Etapa de filtrado.

3.1.4 Etapa de amplificación

En el caso de la versión del circuito, el potenciómetro mostrado en la imagen ha sido reemplazado por una resistencia de 20 kΩ.

Con esta amplitud, junto a la ganancia en el amplificador de instrumentación, la señal alcanza un rango que la entrada analógica de Arduino puede medir, es decir, entre 0 y 5 voltios. Si se multiplican ambas ganancias, se obtiene que la ganancia total del sensor adquiere un valor de 5000 aproximadamente.

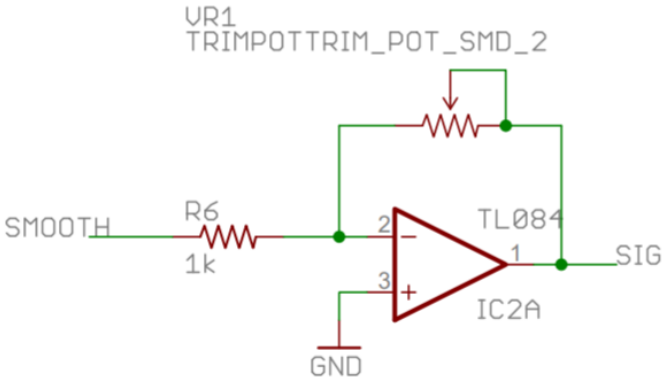


Figura 3.5 Etapa amplificadora.

3.2 Medidas con Arduino

Para medir el sensor, se ha utilizado la entrada analógica de Arduino, la cual tiene una tasa máxima de 10000 medidas por segundo. Es por ello que, para la comunicación con el ordenador, se ha

decidido usar comunicación serie a 9600 baudios, perdiendo pocas medidas, pero sin duplicar la cantidad de datos transmitidos.

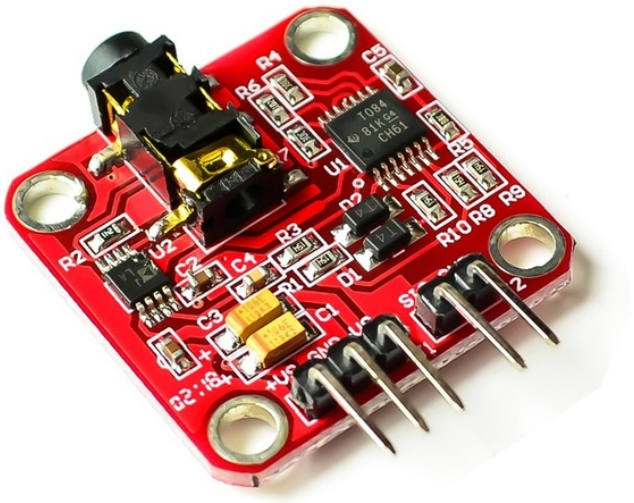


Figura 3.6 Sensor EMG.

La distribución general de los electrodos será uno en una zona no muscular, como podría ser el codo, para tomar referencia de potencial, y dos sensores en el músculo a medir, uno en la mitad de este y otro al final del músculo siguiendo la dirección longitudinal de este.

3.2.1 Colocación de sensores

Las medidas se obtendrán principalmente de los músculos flexor común superior e inferior del antebrazo, aunque al ser sensores sEMG, el segundo será de difícil acceso. Además, para la medida del pulgar, se usará el músculo abductor largo del pulgar.

Aunque solo se disponga de un sensor, se realizarán medidas para cada dedo, seleccionando las zonas apropiadas de estos músculos para tener la mayor amplitud posible con este sensor en la señal.

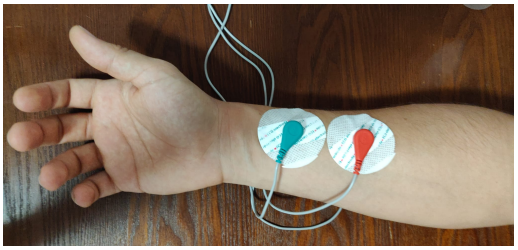
El electrodo de color rojo ha de situarse sobre el músculo a medir, mientras que el electrodo verde tiene como propósito mejorar la medida, y el amarillo tomar una referencia. Realmente, solo dos serían necesarios, pero la precisión al añadir un segundo electrodo de medida aumenta con creces.

Los electrodos también pueden ser situados en diferentes músculos, llegando a tener señales de mayor amplitud en músculos de mayor tamaño.

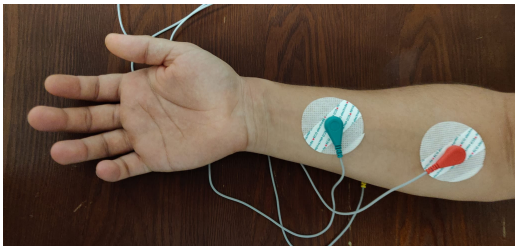
Realmente, el músculo encargado de la flexión de los dedos es común para estos, siendo complicado saber cuándo se flexiona uno u otro. Aun así, pese a ser un único músculo, hay zonas de este que se contraen más o menos en función del dedo deseado a contraer o extender, y diferentes terminaciones de este músculo para cada tendón [17].

Por ello, se colocan aproximadamente los electrodos sobre estas zonas en función de cada dedo. El dedo índice y el dedo meñique corresponden a la zona interior de este músculo, por lo que son considerablemente más difíciles de leer.

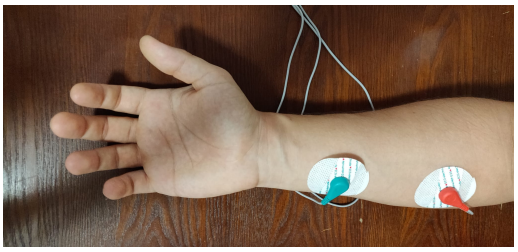
Para el pulgar, como el abductor está en la palma de la mano, ha de medirse el abductor largo, un músculo pequeño y de difícil acceso que se extiende por el radio.



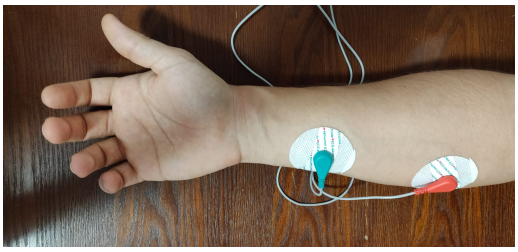
(a) pulgar.



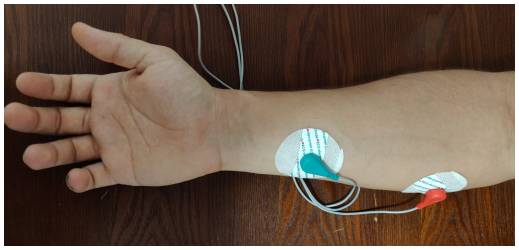
(b) índice.



(c) corazón.



(d) anular.



(e) meñique.

Figura 3.7 Posicionamiento de sensores.

Hay numerosos estudios sobre la colocación de los electrodos para obtener diferentes resultados, pero no es el tema central de este trabajo. En este caso se ha optado por seguir la colocación de los electrodos, reduciendo el número a 5 posiciones, basándose en el artículo [18] para la situación de los electrodos de medida, y utilizando el codo como referencia.

3.2.2 Gráficas de medidas

Las señales EMG que se mostrarán a continuación son el resultado de la medida ya procesada por el sensor, es decir, amplificadas varios órdenes de magnitud, filtradas y rectificadas. Sin embargo, para tener mayor claridad de estas o trabajar fácilmente, podemos aplicar filtros más estrictos en frecuencia.

El movimiento realizado consiste en contraer y relajar suavemente el bíceps, con bastante intensidad en la contracción.

Observando la gráfica 3.8, podría establecerse como caso general que, a partir de un valor de 100 en la entrada analógica filtrada, el músculo estaría contraído, o que, si se alcanzasen picos de 300 en la señal sin filtrar, habría contracción igualmente.

Para las pruebas se ha contado con la ayuda de dos sujetos para realizar medidas EMG en sus antebrazos. De este modo se puede apreciar la diferencia de resistencia eléctrica de cada individuo, que depende de muchos factores, como las diferencias entre las características de los músculos (grosor, longitud) y antebrazos [19].

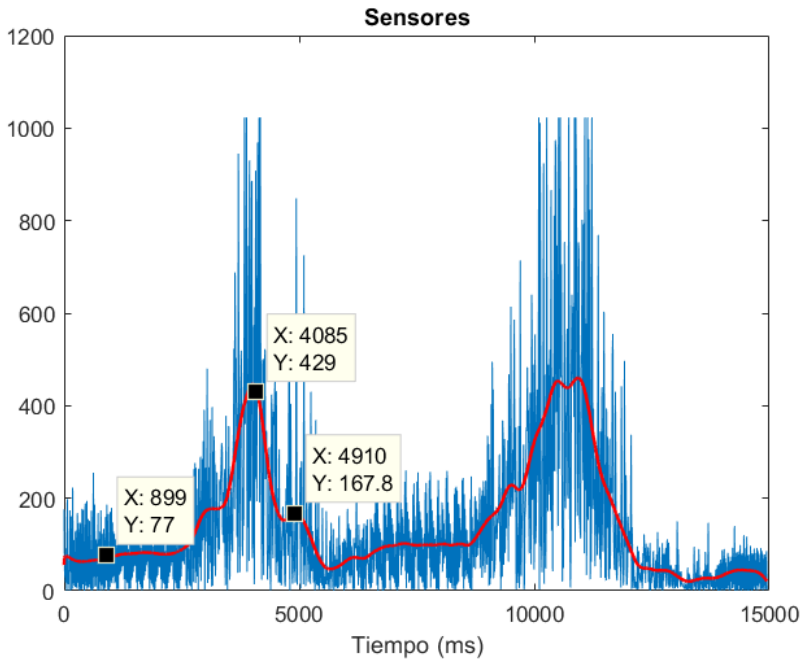


Figura 3.8 Gráfica medida EMG bíceps. Filtro a 1 Hz en rojo.

En todas las señales de cada dedo se ha seguido el patrón de contracción durante un pequeño periodo de tiempo, contracción algo más duradera y repeticiones de contracciones pequeñas.

Sujeto 1

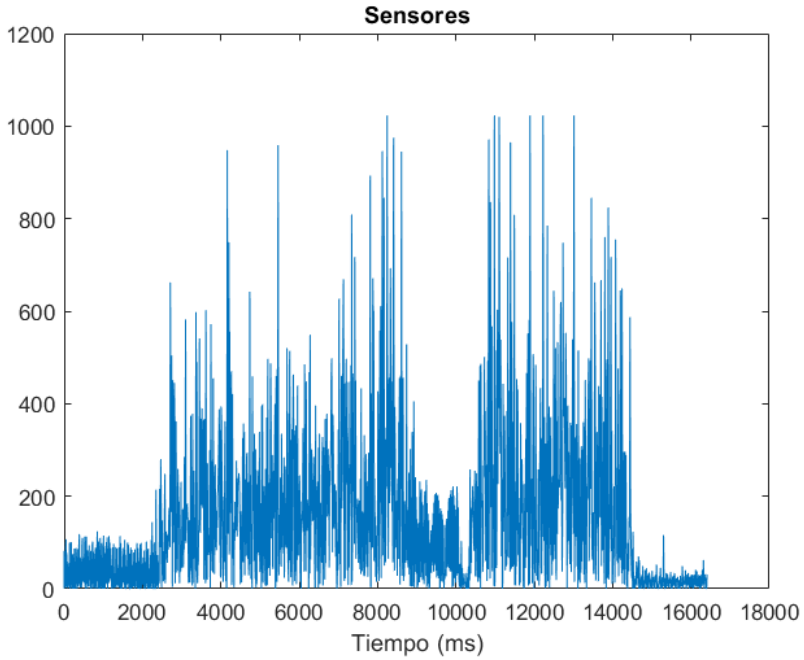


Figura 3.9 puño contraído y relajado en intervalos largos.

Tal y como puede observarse en la figura 3.10, ante picos de fuerza se producen picos de intensidad, mucho más destacados en intervalos cortos. Esto implica que ante medidas de señales

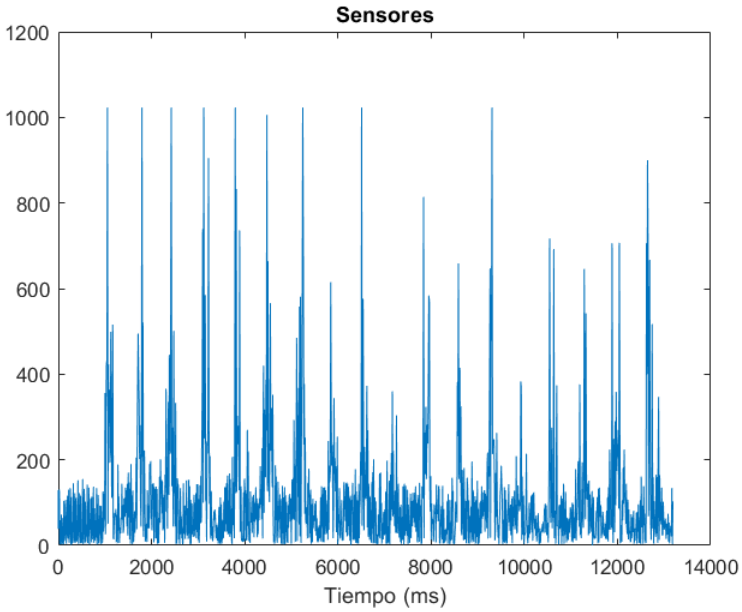


Figura 3.10 puño contraído y relajado en intervalos muy cortos.

de contracciones mantenidas no se actuará fielmente según la intensidad del pulso, mientras que al reconocer pequeños impulsos, podría crearse un código de contracciones para mover cada dedo con un solo sensor.

En el caso de contraer dedos, tal y como se ha mencionado, la señal del pulgar y del índice son considerablemente peores, pues tienen peor acceso que el resto. Esto se aprecia debidamente al comparar las figuras 3.13a y 3.13b con el resto.

Sujeto 2

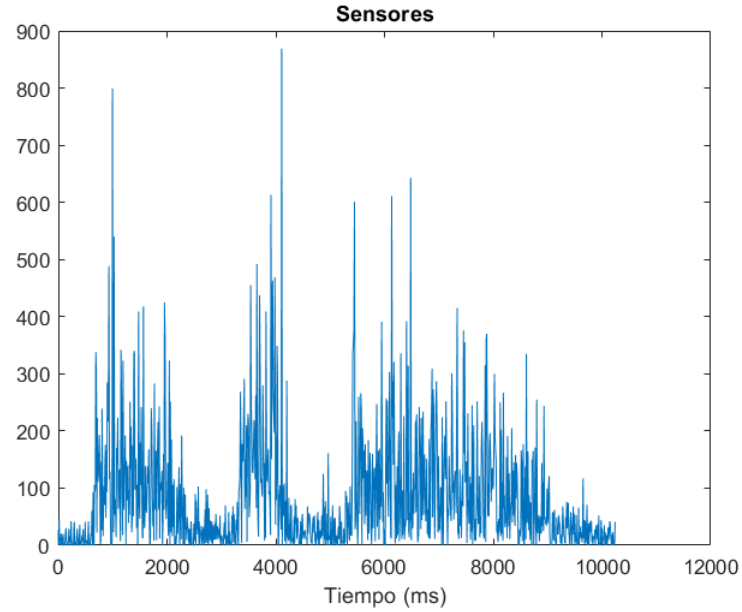


Figura 3.11 puño contraído y relajado en intervalos largos.

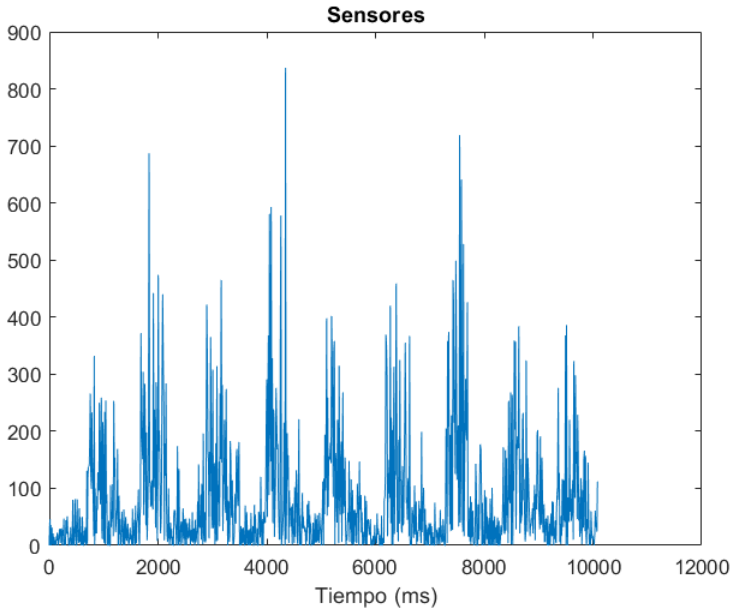


Figura 3.12 puño contraído y relajado en intervalos muy cortos.

La calidad de la señal se verá siempre alterada por diversos factores, entre ellos, el ruido introducido por los componentes electrónicos del sensor y electrodos, el ruido electromagnético ambiental de las fuentes de corriente a 60 Hz o la adquisición de datos de otros músculos no deseados. Además, los OPAM del sensor introducen una tensión de offset dependiente de la temperatura que se amplificará en las etapas del sensor [19].

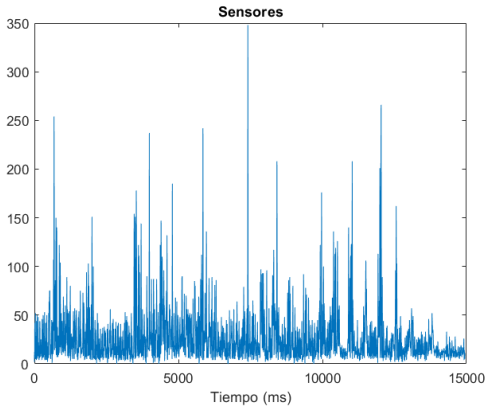
Comparando las gráficas de ambos sujetos, se observa que no comparten el mismo "offset" al relajar los músculos, y que la amplitud máxima de la señal EMG de cada uno varía con creces. Es por esto que es estrictamente necesario hacer un estudio electromiográfico a un paciente para poder establecer límites para determinar si hay intención de movimiento o no, o diseñar un método que automatice este procedimiento.

Por ejemplo, en el caso del sujeto 1, dado que, como se aprecia en 3.9, al relajar la mano se mide en la entrada analógica de Arduino unos 0.5 V (100) mientras que, al contraerla se aprecian picos que saturan en 5 V (1024). Es por ello que, se necesitaría establecer un valor para discriminar entre intención de cerrar el puño o no por encima de los 0.5 V.

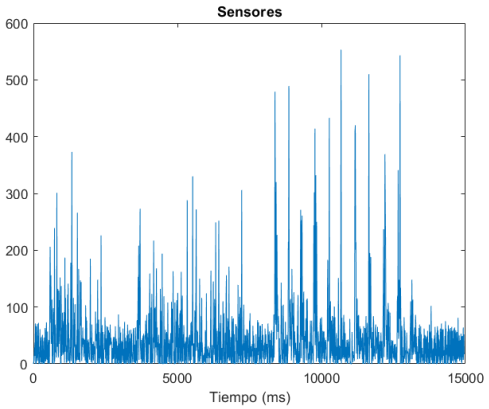
Apreciando la figura 3.9, un valor lógico resultaría 1 V (200). Sin embargo, observando 3.11, este valor podría mantenerse en 0.5 V (100), pues apenas llega a 0.2 V el valor que mide el sensor cuando se relajan de los músculos del antebrazo.

En el apéndice A, concretamente en el código A.1, se muestra el código utilizado para la adquisición de medidas, que, junto con *putty* para leer el puerto serie y guardar los logs de este en un archivo .txt, se representa en matlab con el código A.3. Para el filtro, por la necesidad de que la caída de la ganancia tras la frecuencia de corte sea rápida, se requiere un orden relativamente alto.

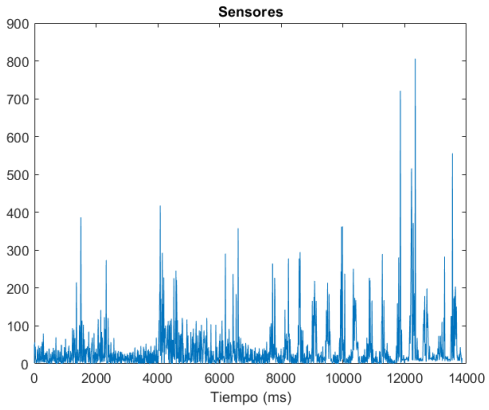
En este apartado se han utilizado sensores de bajo coste y por ende bajas prestaciones, sin embargo, para comprobar su validez se compararán con unos sensores de gran precisión expuestos en el siguiente capítulo.



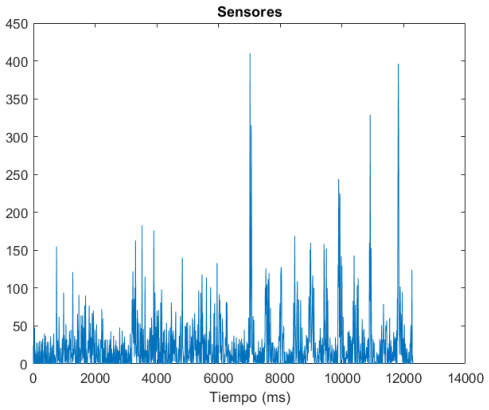
(a) pulgar.



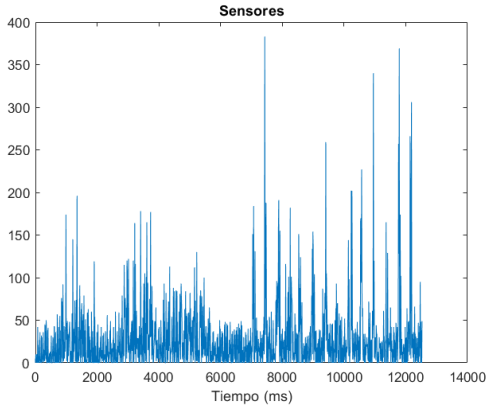
(b) índice.



(c) corazón.

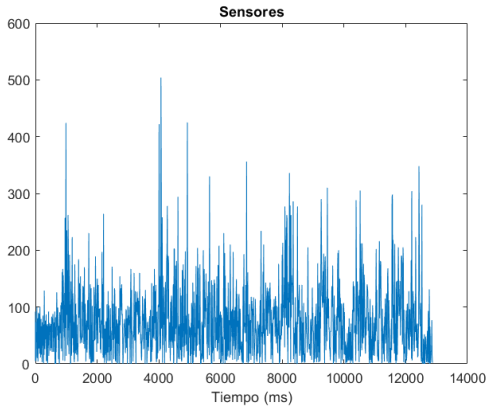


(d) anular.

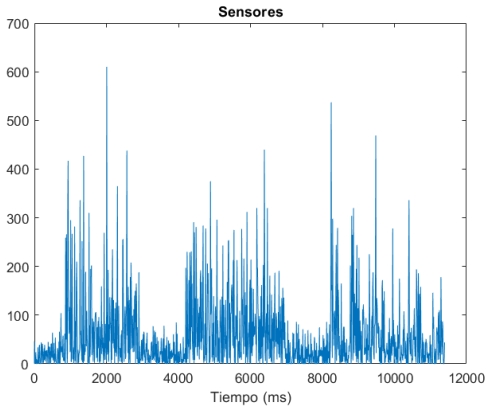


(e) meñique.

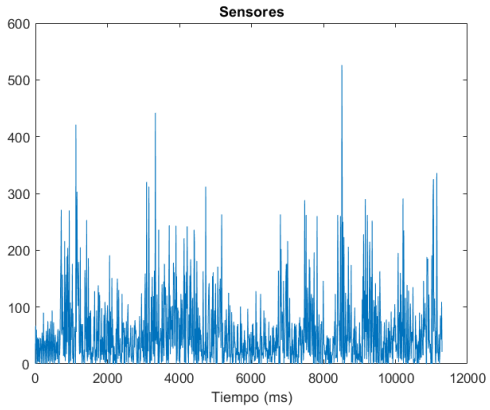
Figura 3.13 Gráficas de cada dedo del sujeto 1.



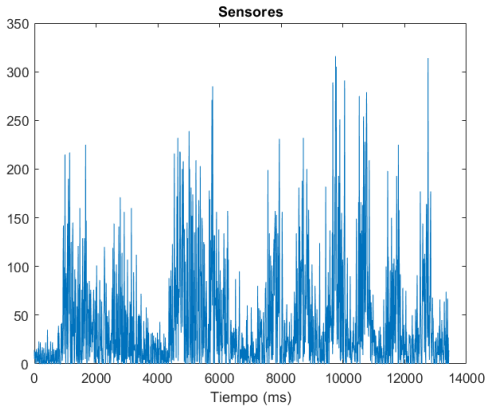
(a) pulgar.



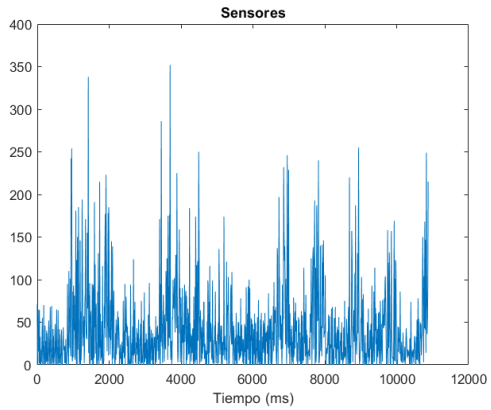
(b) índice.



(c) corazón.



(d) anular.



(e) meñique.

Figura 3.14 Gráficas de cada dedo del sujeto 2.

4 Sensor EMG Trigno

Con objetivo de apreciar el verdadero alcance del sensor EMG utilizado en la prótesis, se realiza la comparativa al sensor TRIGNO™ Wireless Sensor. Este sensor pertenece a la compañía DELSYS®. La gama de sensores EMG que ofrece esta compañía es amplia, variando desde sensores individuales cableados hasta sistemas de sensores inalámbricos, como el que se dispone en el laboratorio de ingeniería mecánica y que se ha usado en este proyecto.

La comparativa de ambos sensores resulta interesante para observar la calidad de la señal medida mediante un sensor de 20€ - 30€ frente a un sistema de 20000€. Claramente, ambos tienen diferente alcance, los sensores de Delsys cuentan con una precisión mucho mayor, y están acompañados de un software diseñado por ellos cuyas funciones sobrepasan este proyecto, dándoseles uso en aplicaciones médicas.

La versión que se encuentra en el laboratorio incluye 16 sensores EMG que a su vez contienen, cada uno, un acelerómetro de 3 ejes, conectados a una antena de la base. Sin embargo, las nuevas versiones que han comercializado contienen además conexiones wifi y *bluetooth* para cada sensor, de manera que pueden monitorizarse desde cualquier dispositivo. Para este capítulo se utilizará el software de Delsys para tratar las señales EMG obtenidas por los sensores, llegando a utilizarse cinco sensores simultáneamente.

4.1 Software de Delsys

Para este apartado se han utilizado los siguientes programas.

4.1.1 Trigno Analog Output

Trigno Analog Output ayuda al usuario a emparejar los sensores al PC, dando información sobre el estado de estos tales como la batería o la calidad de la señal que llega. Para conectar un sensor hay que encenderlo (mantener pulsado su único botón) y una vez encendido, acceder al modo emparejamiento (mantener pulsado hasta que parpadee su luz verde).

4.1.2 EMGworks Acquisition

EMGworks es el programa facilitado por la compañía para adquirir y visualizar la señal leída por los sensores. Permite crear perfiles con diferentes configuraciones de sensores, usuarios, y establecer un orden de tareas para trabajar sobre los sensores, siendo el total del proceso monitoreado tanto para la realización de las tareas como para la colocación de los sensores. Las tareas que se han trabajado para este proyecto son:

- Plot and Store: Permite visualizar la señal sin procesar y almacenarla si se desea.
- MVC (EMG RMS): Realiza y muestra el valor eficaz de la onda que mide mientras la adquiere.

- Analyze: Permite exportar a EMGworks Analysis las tareas realizadas anteriormente a esta.

4.1.3 EMGworks Analysis

Permite crear gráficas para representar los datos proporcionados por Acquisition. Es una herramienta muy similar a las figuras de matlab, permite además realizar cálculos y filtros sobre la propia gráfica y exportarla como vector.

4.2 Medidas EMG del antebrazo

Por simetría con el apartado anterior, se ha decidido realizar experimentos siguiendo el mismo patrón de movimiento, por lo que en primera instancia, el sujeto contrae el antebrazo firmemente 4 segundos aproximadamente, descansa un segundo y realiza otra contracción más intensa que la anterior durante 5 segundos. Los resultados se muestran en 4.1.

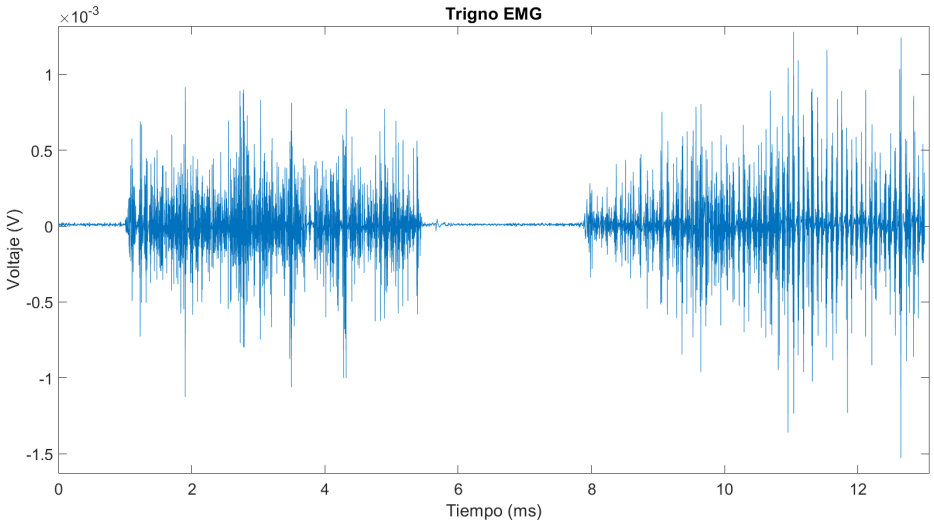


Figura 4.1 Señal EMG recibida por el sensor trigno sin procesar.

A simple vista se observa mejora en cuanto a la reducción de ruido en intervalos de reposo, donde prácticamente la medida es nula. Tal y como se aprecia, es una señal sin procesar, con picos del orden del milivoltio.

Del apartado anterior se obtuvo una ganancia para el sensor EMG utilizado de 5000, por lo que, observando esta medida, los picos de 1 milivoltio saturan en 1023 en la entrada analógica de arduino, tal y como se puede ver claramente , por ejemplo, en 3.8.

Como segundo experimento, el sujeto realiza contracciones rápidas del antebrazo, originando picos de gran amplitud en la medida.

Del mismo modo que 4.1, los picos que sobrepasasen el milivoltio en el otro sensor, saturarían en la entrada de arduino, a falta de tener potenciómetro. Sin embargo con estos sensores, al no tener procesamiento analógico de la señal, podría modificarse el valor máximo perceptible digitalmente.

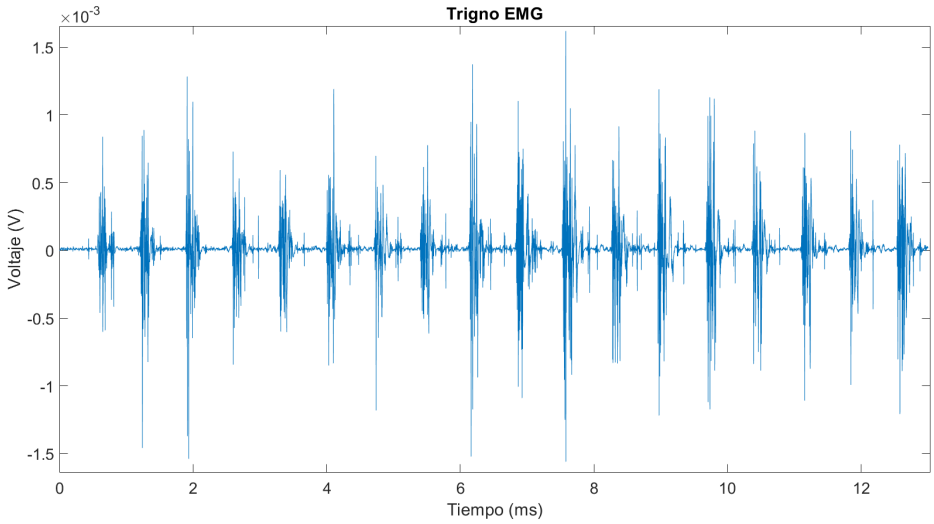


Figura 4.2 Señal EMG recibida por el sensor trigno sin procesar.

4.2.1 Localización de sensores en el antebrazo

Para la colocación de los sensores trigno se ha optado por seguir el artículo [20] sustituyendo los sensores de la parte exterior por dos sensores que acompañen al resto.

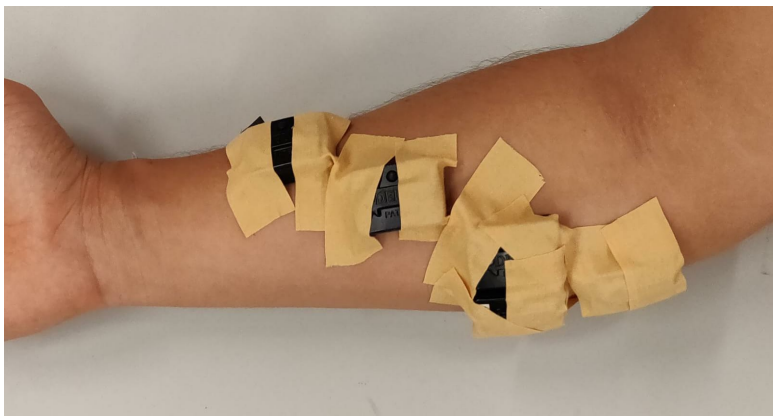


Figura 4.3 Colocación de los sensores en el antebrazo.

4.2.2 Gráficas de medidas de cada sensor

Al igual que en el capítulo anterior, se realiza una contracción larga de cada dedo y tras ello una serie de contracciones cortas. Sin embargo, como se dispone de suficientes sensores simultáneamente, se medirá cada dedo junto al ruido que producen los músculos circundantes.

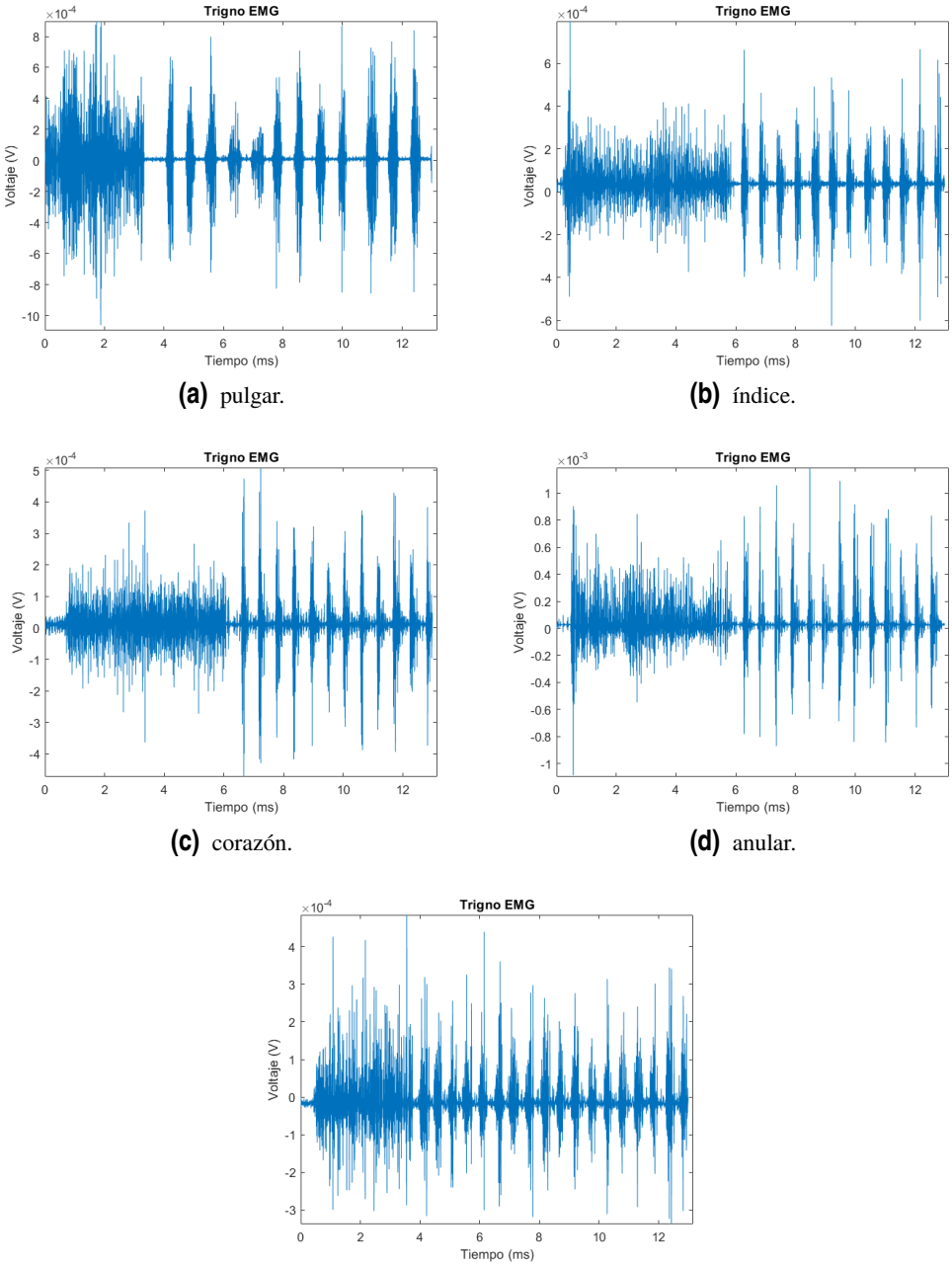
Aunque en 4.4 se observen claramente los resultados de medir cada dedo, en el resto de sensores se medirá un valor no correspondiente al movimiento del dedo del sensor, lo que podría desencadenar en movimientos de dedos de la prótesis no voluntarios.

Un ejemplo de esto es 4.5, donde se puede apreciar que el movimiento del anular provoca una medida en el sensor del meñique, aun siendo de un orden menor que la del anular, del mismo orden que mediría el sensor del meñique cuando el movimiento de este es intencional.

Sin embargo, si se calcula su valor RMS (figura 4.6), es decir, el valor típico de la onda, en este caso con un tamaño de ventana de 0.25 segundos, se puede comprobar que al mantener contracción,

el valor medio de la señal obtenida en el sensor del meñique al mover el anular tiene la mitad de amplitud que la medida en el sensor del meñique al mover este.

Para el índice y corazón se obtienen resultados similares a este. Además, en el sensor del pulgar también se encuentra ruido del movimiento del dedo índice, sin embargo es decartable sin cálculos de RMS, pues su amplitud (de la señal sin procesar) es muy inferior a la medida del sensor pulgar cuando este se contrae.



(a) pulgar.

(b) índice.

(c) corazón.

(d) anular.

(e) meñique.

Figura 4.4 Medidas EMG de sensores Trigno.

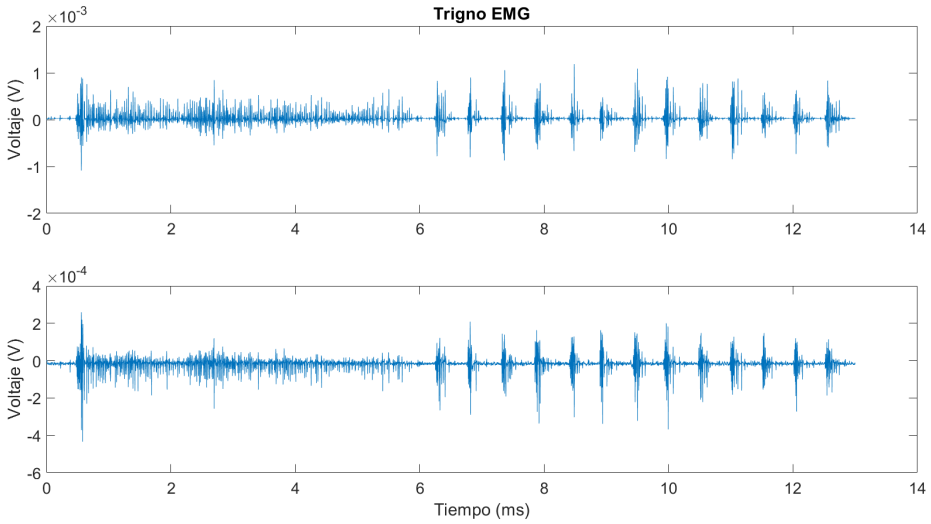


Figura 4.5 Señal obtenida al mover el dedo anular medida en el sensor de anular (arriba) y meñique(abajo).

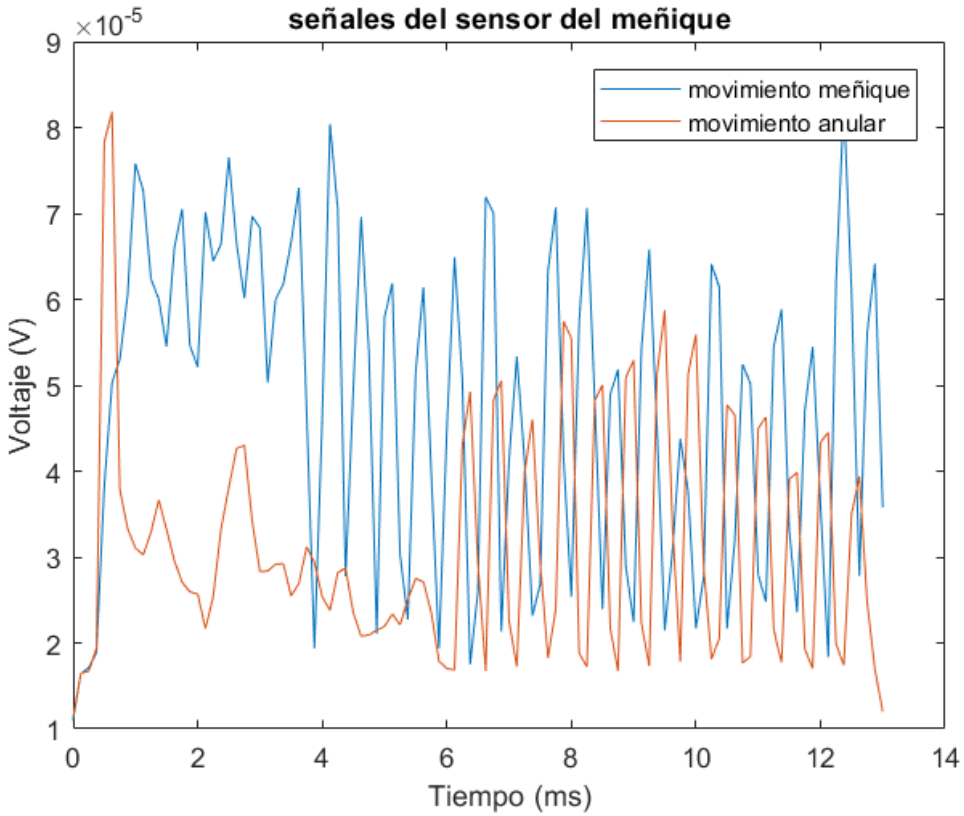


Figura 4.6 RMS del sensor meñique.

4.3 Conclusión

Los resultados que ofrecen los sensores de trigno, además de poder ser acompañados del acelerómetro, son claramente mejores, se dispone de una redución del *offset* en toda la señal en el caso del sensor trigno. Además, la tasa de muestreo que ofrece la señal de trigno es 27 veces mayor.

Por otra parte, cabe recordar que la señal que miden los sensores de trigno no tiene ningún tipo de filtro, ni rectificador, ni amplificador, por lo que para analizarla y trabajar con ella es mejor que la señal que ofrece la salida del sensor EMG de bajo coste. Sin embargo, para tratar la señal con arduino, es claramente preferente y necesario que la señal resultante en la entrada analógica de arduino sea positiva y esté escalada en un determinado rango.

Debido a la gran diferencia de precio, como el objetivo del proyecto es realizar una prótesis minimizando el coste, el sensor que se utilizará en la version final es el sensor de bajo coste, siendo filtrado digitalmente también. Sin embargo, en el siguiente apartado se trabajará con muestras obtenidas de los sensores trigno.

Con la comparación hecha, partiendo del problema observado en las imágenes 4.5 y 4.6, ha de realizarse un clasificador para poder diferenciar entre dedos cuando el ruido de un músculo sobre un sensor donde no debería haber medida interfiera en el dedo deseado. Para ello se utilizarán los sensores Trigno de este mismo capítulo.

5 Clasificación movimientos

En microcontroladores de bajo coste suele discriminarse según la amplitud de las señales mientras que en casos mas precisos se incluyen métodos de inferencia bayesiana o características que impliquen mayor coste computacional.

5.1 Características típicas

Para obtener las características principales se ha utilizado EMGworks Analysis, programa que permite el tratamiento rápido de señales y cálculos sobre estas. La señal sobre la cual se calcularán las características de ejemplo es 5.1. Se mostrarán tanto características del dominio temporal como frecuencial, trabajadas en estudios como [21] o [22].

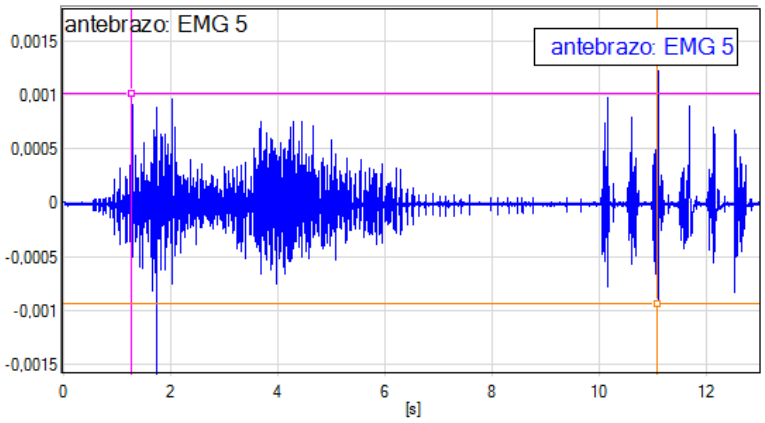


Figura 5.1 Señal EMG sin procesar obtenida del antebrazo. Gráfica de EMGworks Analysis.

Son muchos los cálculos que se pueden realizar sobre las señales EMG, pero en la mayoría de estudios [23] [24] [25] [26] suelen utilizarse los siguientes:

5.1.1 Valor eficaz (RMS)

El valor eficaz o valor cuadrático medio de una onda consiste calcular la media de sus valores, calculándose sobre un determinado número N de muestras.

$$RMS = \sqrt{\frac{1}{N}\sum_{n=1}^{N-1}|x_n^2|} \tag{5.1}$$

Es un valor de mucho interés en casos de una sola característica, pues permite establecer un valor discriminante con facilidad 5.2. Los resultados del valor RMS como característica suelen rondar el 88 % [24] o incluso más cuando se combina junto a otras características.

Además, como se puede apreciar en 5.3, pese a tener menos amplitud que la onda original, no dista de la forma rectificada que tendría la onda.

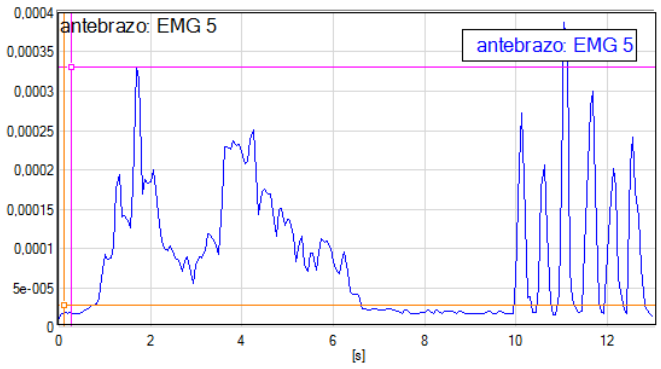


Figura 5.2 RMS de señal EMG . Gráfica de EMGworks Analysis.

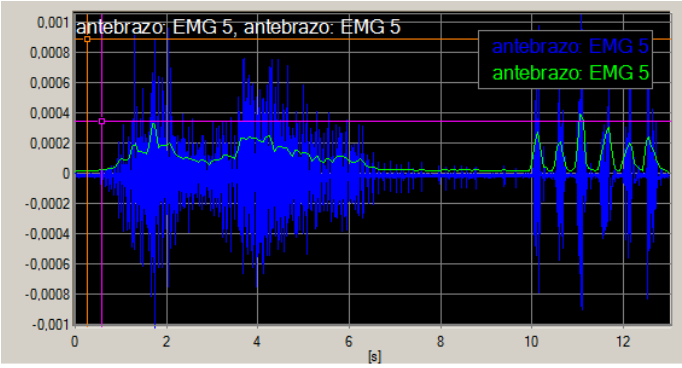


Figura 5.3 Señal EMG sin procesar con RMS sobrepuesto. Gráfica de EMGworks Analysis.

5.1.2 Valor medio absoluto (MAV)

El valor medio absoluto de una onda se define por sí mismo, consiste en calcular la media integral de tamaño N de los valores absolutos de la onda.

$$MAV = \frac{1}{N}\sum_{n=1}^{N-1} \omega_n |x| \tag{5.2}$$

Además, se suelen añadir a este cálculo ventanas ω_n como función a trozos para ponderar más los valores centrales al punto de la ventana.

Tal y como se aprecia en 5.4, el resultado tiene forma muy similar al cálculo de RMS de la onda, sin embargo se aprecia menor amplitud en este caso.

De hecho, según [24], los resultados que se obtienen al aplicar el MAV son muy similares, pero levemente menores, a los resultados de tratar al RMS como característica única. Esto es debido a la pérdida de amplitud de una señal con respecto a la otra.

Un cálculo muy común es el iEMG, la integral de los valores absolutos de la onda, pero no se entra en detalle por ser muy similar al MAV.

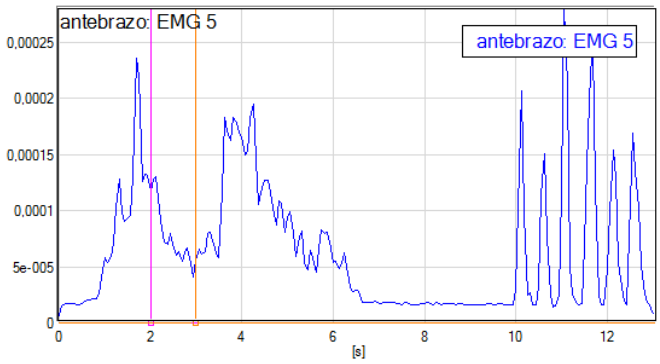
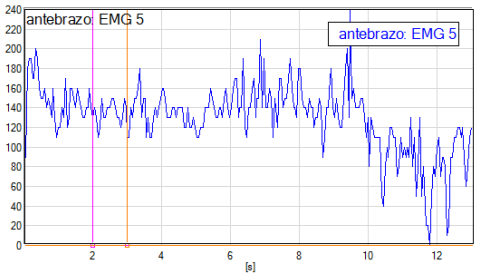


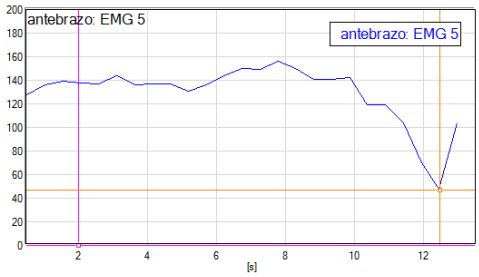
Figura 5.4 MAV de Señal EMG. Gráfica de EMGworks Analysis.

5.1.3 Frecuencia media y mediana (MEF, MDF) y densidad espectral (PSD)

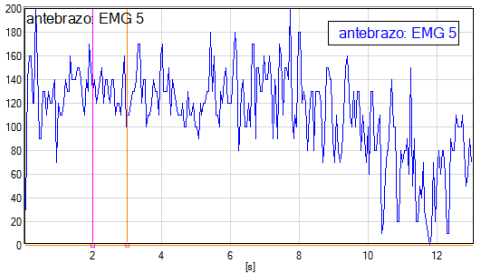
En el dominio frecuencial pueden obtenerse características tales como la frecuencia media 5.5a 5.5b o mediana 5.5c 5.5d o la densidad espectral. Las dos primeras permiten conocer los valores medios y medianos de la onda para intervalos de tiempo, de tal manera que reúne la media o mediana de un número N de datos y lo mantiene durante ese periodo.



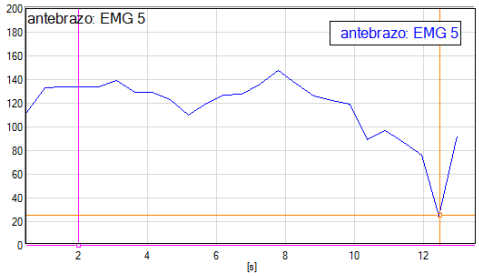
(a) MEF de Señal EMG ventana 0.1s.



(b) MEF de Señal EMG ventana 1 s.



(c) MDF de Señal EMG ventana de 0.1s.



(d) MDF de Señal EMG ventana 1 s.

Figura 5.5 Media y mediana de frecuencia. Gráficas de EMGworks Analysis.

La densidad espectral de una onda muestra la distribución de los valores de esta en frecuencia. Es similar a un histograma de frecuecias, pero sin tener en cuenta la frecuencia como tal sino la potencia de la onda.

En 5.6 se observa cómo una señal EMG sin filtrar tiene su máximo de componentes frecuenciales en torno a los 100 Hz, con gran parte de su espectro en aún mayor frecuencia. Cuando el sensor del capítulo 3 filtra a 50 Hz, gran parte del espectro se ve filtrado, por lo que obtenemos una muy pequeña proporción de este.

La PSD sin embargo solo ofrece un 75 % de fiabilidad, muy inferior a los resultados de MAV y RMS.

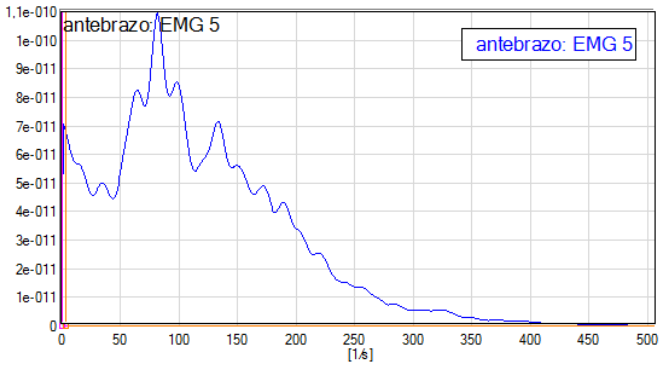


Figura 5.6 PSD de Señal EMG. Gráfica de EMGworks Analysis.

5.2 Driscriminante lineal

Puesto que para discriminar entre tipos de movimientos mediante amplitud sólo habría confusión entre meñique y anular y entre índice y corazón, realizar dos discriminantes lineales de dos dimensiones (dos características) debe ser suficiente.

Los discriminantes lineales ofrecen gran sencillez en estos casos, pese a ser poco generales. Consisten en una recta que separe clases según características, expresada como función discriminante lineal.

Como característica para discriminar, se elige el valor MAV de la señal. Dicho valor se puede calcular en matlab según el código 5.1.

Código 5.1 Código de matlab para calcular MAV.

```
L=200; %L numero de muestras para cada calculo de MAV
for j = 1:size(file(:,2))-L
    sum = 0;
    for i = 1:L
        sum = sum + abs(file(i+j-1,2));
    end
    MAV(j+L/2) = sum/L;
end
```

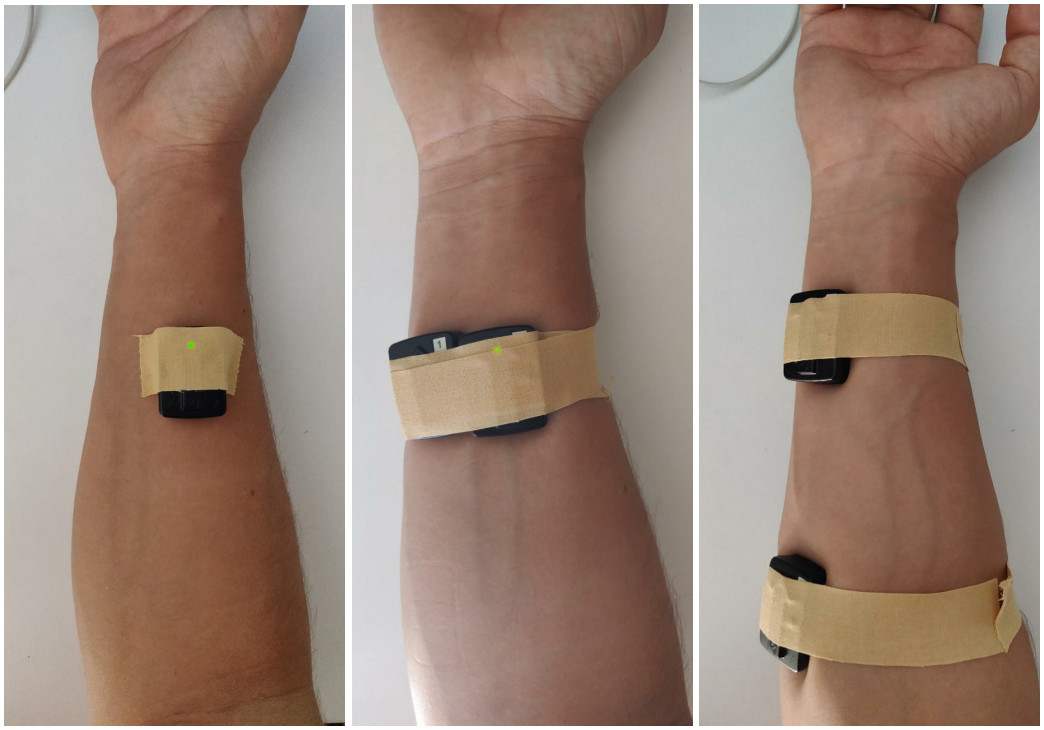
Para este apartado se han recolocado los sensores de manera que se aislen pulgar, índice y corazón, y anular y meñique.

5.2.1 Pulgar

El caso del pulgar es bastante simple, pues al no tener apenas ruido por otros músculos, basta con clasificar si hay intención de movimiento en este o no, resultando en dos clases (contracción pulgar o extensión pulgar) según una sola característica (MAV pulgar).

Para ello se han extraído muestras de varias pruebas, creando un conjunto de datos con contracciones y relajaciones.

Tras analizar 5.8, claramente se puede establecer $0.2 \cdot 10^{-4}$ como valor límite entre las dos clases. Para este conjunto de 170 muestras, la discriminación es perfecta y simple. Sin embargo, si se aumenta el número de muestras analizadas hasta diez veces más, se obtiene que las clases no están tan separadas, y que ciertas muestras de cada clase se mezclan en la otra 5.9.



(a) Pulgar. **(b)** Índice y corazón. **(c)** Anular y meñique.

Figura 5.7 Colocación de los sensores.

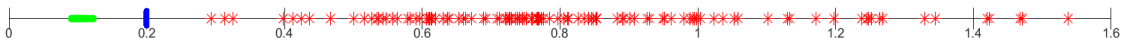


Figura 5.8 MAV de muestras de señales EMG del pulgar (170 muestras).

Figura 5.9 MAV de muestras de señales EMG del pulgar (1700 muestras).

En concreto, se observan 5 muestras mal clasificadas 5.10, pero realmente son 6 muestras erróneas según el algoritmo desarrollado 5.2, lo que equivale (a falta de probar más conjuntos de muestras) a un 0.35 % de muestras mal clasificadas, o lo que es lo mismo, un 99.65 % de fiabilidad.

Figura 5.10 MAV de muestras de señales EMG del pulgar (1700 muestras) con zoom.

Código 5.2 Código de matlab para clasificaciones erróneas.

```
mal_clasificadas = 0;
valor_threshold = 0.00002;
for m = 2:length(conjunto_datos)
    if(conjunto_datos(m) < valor_threshold)
        mal_clasificadas = mal_clasificadas+1;
    end
end
```

5.2.2 Índice y corazón

Para el caso de dos dedos o, equivalentemente, dos características se requieren 4 clases, correspondiendo cada una a cada estado de cada dedo. Este caso es más complicado que la discriminación del pulgar por tratarse de señales que se contaminan entre ellas.

Las funciones de decisión en casos bidimensionales consisten en rectas que separan el plano formado por las dos características a tratar. Para este caso, esas características son el MAV del sensor asociado al dedo índice y el MAV del sensor asociado al dedo corazón.

Es por ello que la dificultad reside en encontrar rectas que separen adecuadamente los elementos en las diferentes clases. Se diferencian cuatro clases:

- C_0: ambos dedos relajados.
- C_1: contracción del dedo índice.
- C_2: contracción del dedo corazón.
- C_3: contracción de ambos dedos.

Para un primer estudio, se seleccionan 341 muestras de clases conocidas, y se representan en el plano según estas características 5.11.

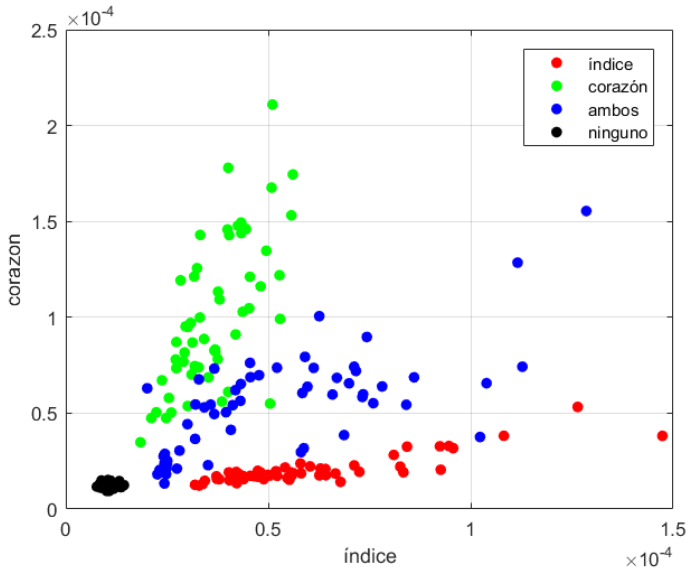


Figura 5.11 Representación de medidas según el MAV de ambas.

Aunque las clases no estén perfectamente separadas, se puede intuir la forma que tendrían las rectas que separasen a estas. El caso de separar las tres diferentes contracciones de la relajación es sencillo, si se traza una recta que vaya desde [0.3, 0] a [0, 0.3] 10^{-4} pueden diferenciarse perfectamente. Para ello se necesita transformar dicha recta, agrupando las dos características en un vector, para evaluar dichas funciones de decisión rápidamente. Es un proceso que se hará para cada punto.

La recta se transforma tal que:

$$x_2 = mx_1 + n \Rightarrow x_2 = -x_1 + 0.00003 \Rightarrow \begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} - 0{,}00003 = 0 \tag{5.3}$$

Con ello, se definen:

$$\overline{n}^T = -\begin{bmatrix} 1 & 1 \end{bmatrix} \tag{5.4}$$
$$\rho = -0.00003 \tag{5.5}$$
$$\overline{w} = \begin{bmatrix} \overline{n} \\ -\rho \end{bmatrix} \tag{5.6}$$
$$fd_1(\overline{x}) = \overline{w}^T \overline{x}_{Ampliado} \tag{5.7}$$

Siendo $\overline{x}_{Ampliado} = \begin{bmatrix} x_1 & x_2 & 1 \end{bmatrix}'$ y w el vector de pesos [27] [28].

Si se sustituye cada muestra que se obtenga en $fd_1(\overline{x})$ se obtendrán valores que, de ser positivos, corresponderán a esa clase según el clasificador diseñado. Por ejemplo, si se sustituye el punto (0,0) obtendríamos que la función de decisión devuelve 0.00003, por lo que dicho punto correspondería a la clase 0, es decir, a la extensión o relajación de ambos dedos.

Con el mismo procedimiento, se obtienen las siguientes vectores de pesos para las clases 1, 2 y 3.

$$\begin{aligned} \overline{w}_2 &= \begin{bmatrix} 1 \\ -2 \\ 0 \end{bmatrix} \\ \overline{w}_3 &= \begin{bmatrix} \frac{9}{5} \\ -1 \\ 0 \end{bmatrix} \end{aligned} \tag{5.8}$$

Representando dichas rectas, el plano queda distribuído según 5.12.

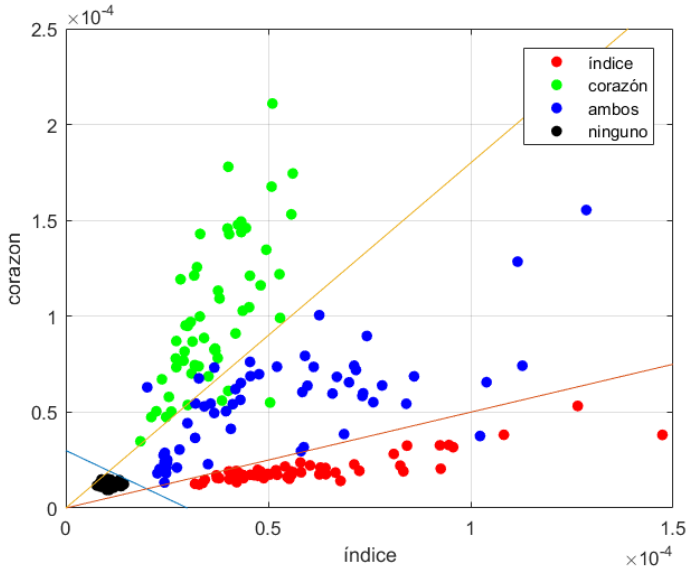


Figura 5.12 Funciones de decisión.

Según el algoritmo programado A.4, de las 172 muestras de contracciones, tan solo 13 están mal clasificadas (figura 5.13), lo que equivale a un 92.44 % de fiabilidad del clasificador (sin contar éxitos al clasificar la clase 0, que es un 100 % de éxitos), un resultado eficiente que cumple con lo observado en estudios como [24] a falta de probar en más muestras.

Para hacer un análisis más exhaustivo, el número de muestras se amplia a 1700 aproximadamente. En este caso, incluso algunos de los elementos de la clase cero se pueden encontrar fuera de esta.

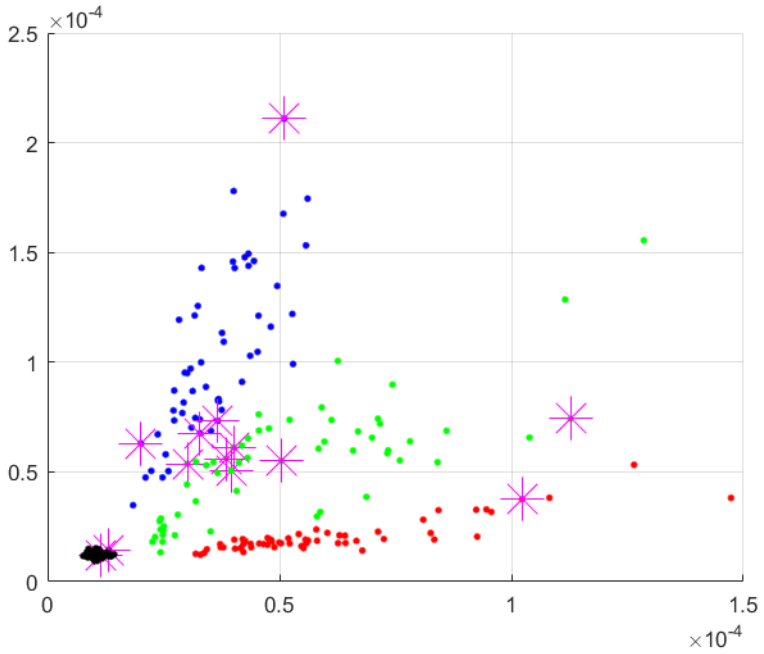


Figura 5.13 Muestras clasificadas por el algoritmo, con marcas para muestras mal clasificadas.

Pese a ello, la fiabilidad solo baja hasta el 92.25 %, lo que sigue siendo un resultado muy eficiente. Tal y como se puede observar en 5.14, la mayoría de fallos se concentran en la frontera entre clases, por lo que mejorando la función de decisión o realizando la discriminación con métodos probabilísticos podría mejorarse la clasificación de estos.

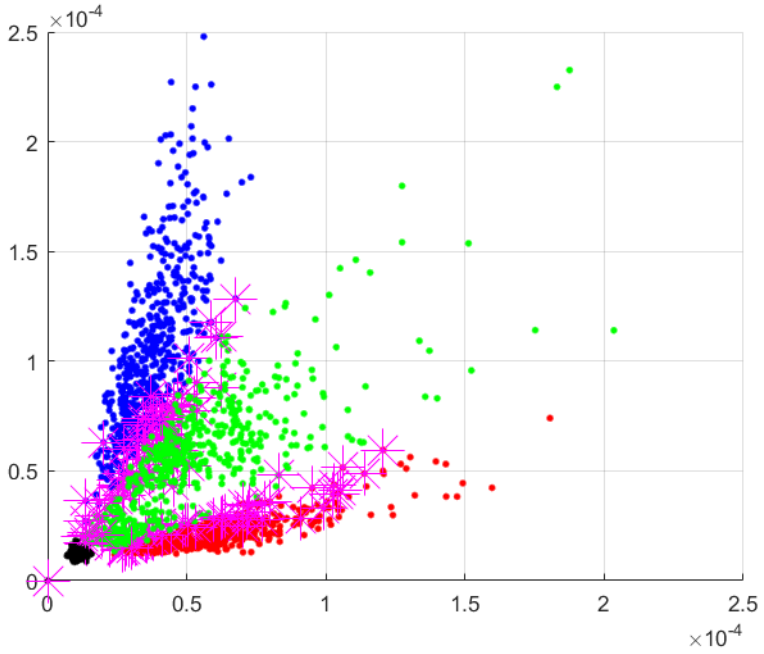


Figura 5.14 Muestras (1700) clasificadas por el algoritmo, con marcas para muestras mal clasificadas.

5.2.3 Anular y meñique

Siguiendo con el mismo procedimiento que el subapartado anterior, se procede a diferenciar entre ambos dedos, pues el ruido que generan uno a otro impide tratarlos por separados.

Tras analizar otras 170 muestras, se determinan las funciones de decisión según los vectores de pesos:

$$\overline{w}_1 = \begin{bmatrix} 0.2 \\ -1 \\ 0.00002 \end{bmatrix} \tag{5.9}$$

$$\overline{w}_2 = \begin{bmatrix} 0.3 \\ 1 \\ 0.000008 \end{bmatrix} \tag{5.10}$$

$$\overline{w}_3 = \begin{bmatrix} 2.9 \\ -1 \\ 0.0000544 \end{bmatrix} \tag{5.11}$$

Con ello se logra un 90.11 % de fiabilidad, algo más bajo que en el caso anterior, pero igualmente del orden de los estudios de los que se parte. Si se aumentan las muestras a 1700 (realmente serían 3400 aproximadamente, pero obviamos la clase 0 por ser mucho más fiable que el resto), se observa que el porcentaje crece al 92.84 %, lo que mejora aún el resultado anterior.

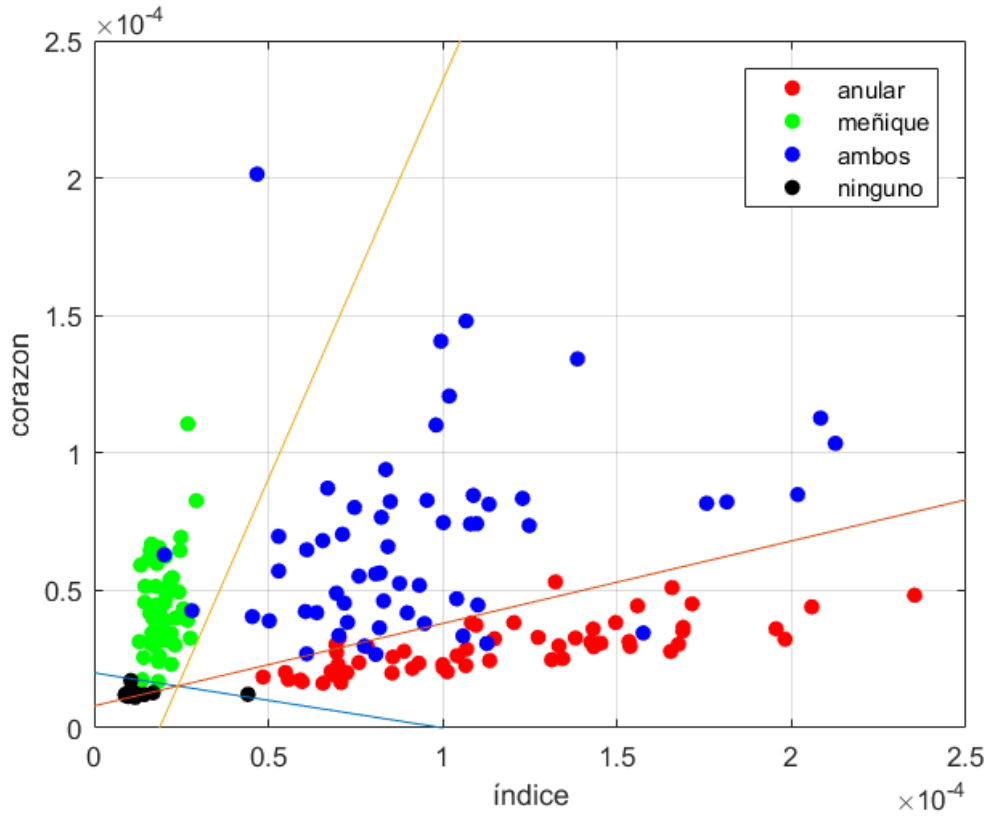


Figura 5.15 Funciones de decisión.

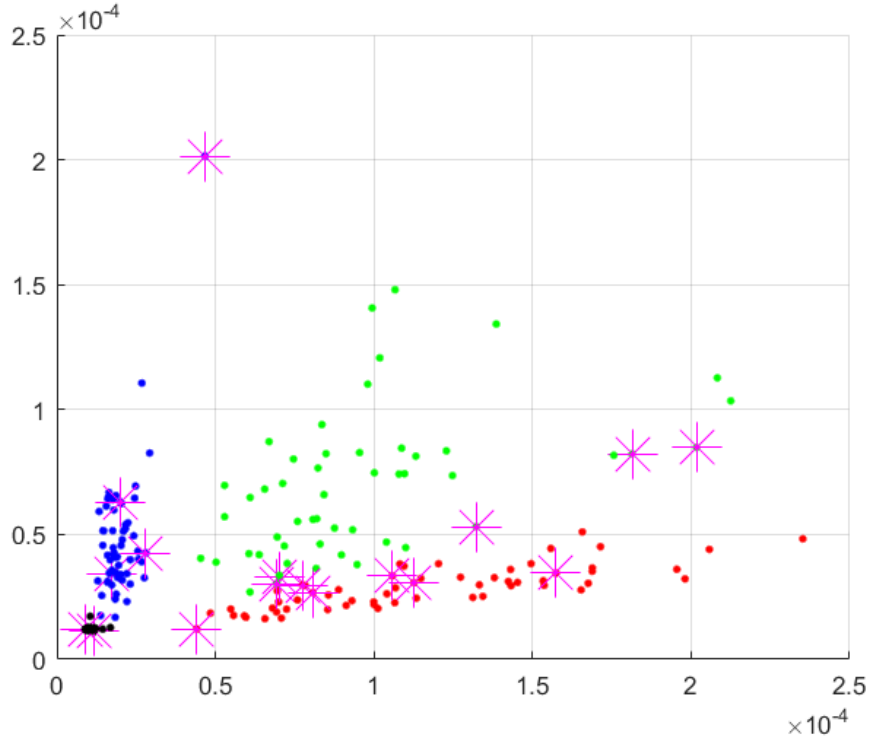


Figura 5.16 Muestras clasificadas por el algoritmo, con marcas para muestras mal clasificadas.

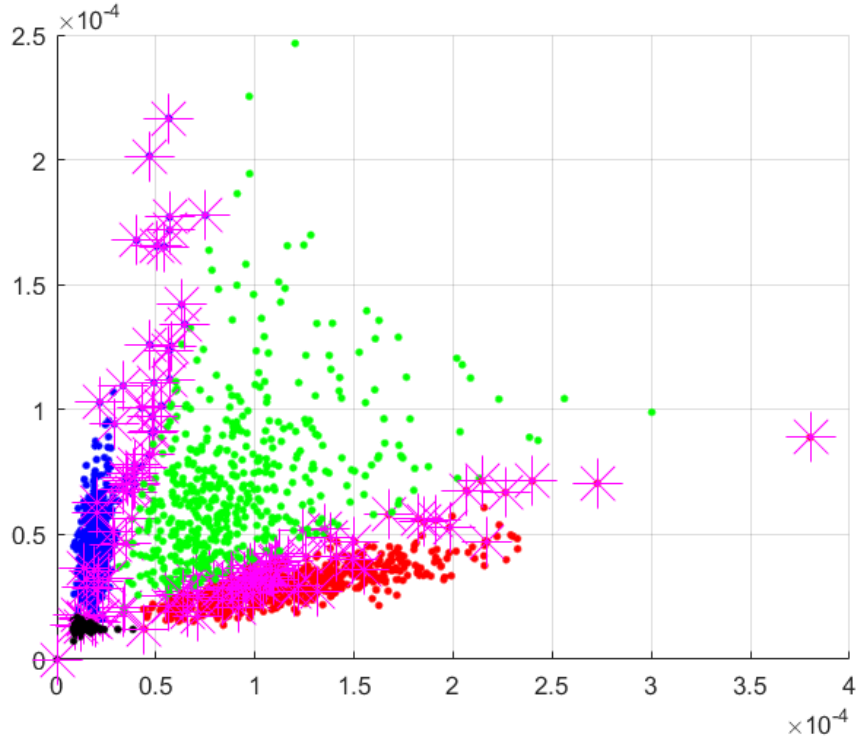


Figura 5.17 Muestras (1700) clasificadas por el algoritmo, con marcas para muestras mal clasificadas.

6 Diseño del antebrazo

El modelo de prótesis del capítulo 2 incluye una ranura en la palma de la mano para ensamblar otra pieza que se una a esta. Es por ello, y por la necesidad de crear una pieza que mantenga los servomotores que traccionarán los hilos, junto a las baterías y el microprocesador y el sensor EMG, que se realiza el diseño de un "antebrazo" que albergue estos elementos.

El diseño de este antebrazo contará con tres piezas, un armazón que mantenga cada motor alineado con la salida del hilo del dedo corresponciente, a la vez que se introduce en la ranura de la palma de la mano, y una carcasa para el antebrazo dividida en dos partes, donde se incluirá la pieza anterior, de manera que una de estas guarde las baterías, el microprocesador y el sensor EMG y la otra mantenga el chasis de los servomotores libre de colisiones con el resto de elementos.

Para el diseño de las piezas se ha utilizado SolidWorks®, un software CAD (diseño asistido por computadora) para diseños mecánicos que croquiza, acota y modela [29]. Se ha decidido usar SolidWorks por ser un CAD de gran relevancia en la industria actual, aunque podría haberse sustituido perfectamente por CATIA, que es de uso más extendido en la escuela.

6.1 Armazón para servomotores

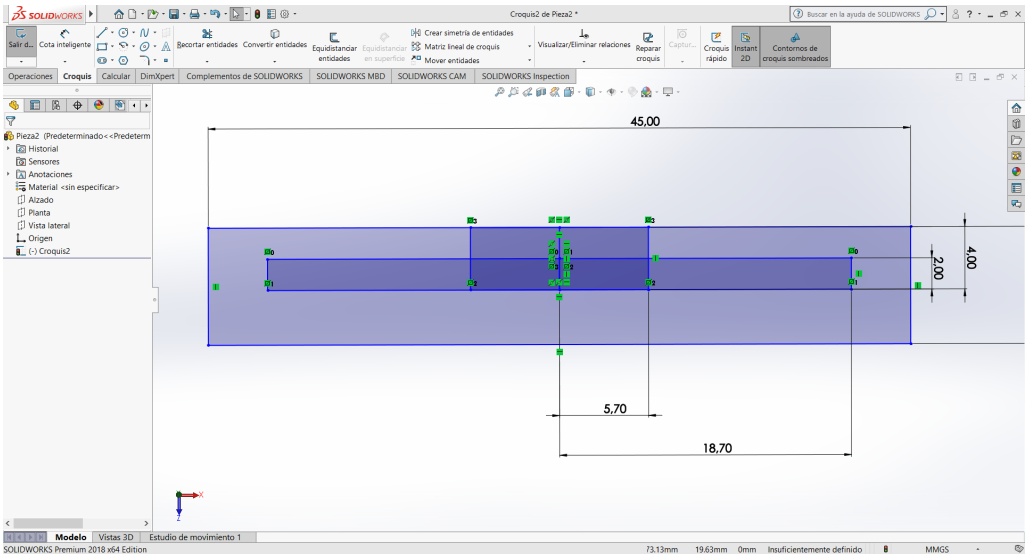


Figura 6.1 Captura de SolidWorks. Croquización de la pieza a introducir en la mano.

En 6.1 se observa el espacio de trabajo de SolidWorks, a la izquierda se puede apreciar una pestaña que contiene los diferentes elementos y partes que componen la pieza, arriba se observan varias pestañas de trabajo, cada una con sus correspondientes herramientas y, en el centro, la pieza que se modela.

En este caso, se croquzia la pieza a introducir por la parte inferior de la mano, y tras ello se acota las diferentes dimensiones de la pieza.

En concreto, se han utilizado las herramientas de operaciones y croquis en su totalidad, y la herramienta de medidas de cálculo.

Si se aplica la herramienta "recorte" se pueden eliminar entidades del croquis, de manera que la pieza resultante es 6.2. Además, se le aplica una pequeña tolerancia a la pieza para compensar errores en la impresión.

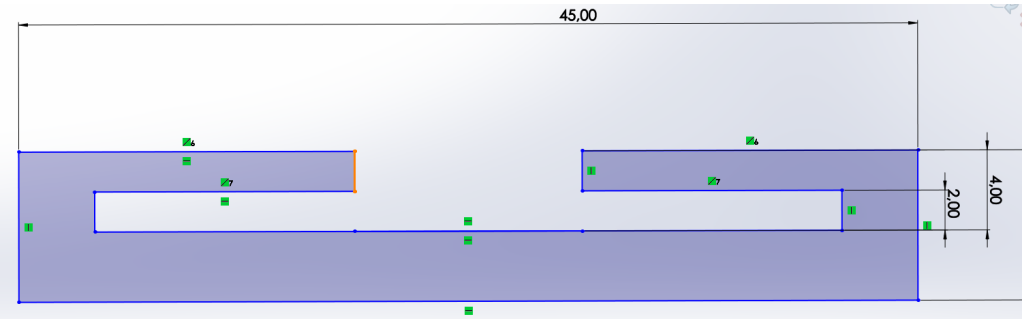


Figura 6.2 Plano de sección de la pieza a introducir en la mano.

El siguiente paso es modelar el sólido desde el plano, dandole una profundidad de treinta milímetros. Para ello se utiliza la herramienta de "extruir saliente/ base".

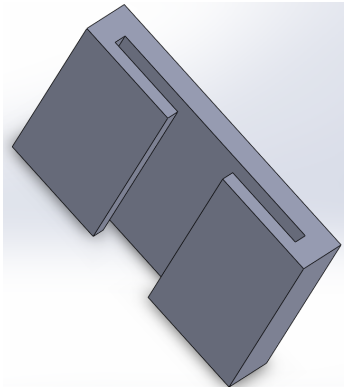


Figura 6.3 Sólido a introducir en la mano.

Para poder comprobar que las tolerancias eran correctas, se imprimió esta pieza por separado. Para ello se exporta a STL el sólido generado y desde Cura se crea el gcode. Tras comprobar esto, se procedió a continuar con el resto de la pieza.

Los servomotores pueden orientarse vertical u horizontalmente, mientras que el giro de este suponga tracción de los hilos. Además, puede asignárseles una rueda que traccione del hilo.

Tras comprobar que las ruedas supondrían un problema para el espacio en el antebrazo, han sido descartadas. Por ello, los servomotores estarán anclados en las paredes del armazón de manera que su movimiento se genere en un plano vertical al de la mano.

Las medidas para que el servo pueda acoplarse son 23.5 milímetros de ancho y 12.7 milímetros de alto, y el taladro a generar para la fijación tiene un diámetro de siete milímetros. Además, se

necesitarán 6 servomotores, uno por cada hilo, por lo que se requiere del suficiente espacio para albergarlos, y que estén separados 5 milímetros entre sí en el plano transversal a los hilos.

Con todo ello, y utilizando las herramientas nombradas anteriormente para croquizar, extruir y la herramienta "extruir corte", se puede crear la pieza final que sostendrá los motores y encajará en la hendidura de la mano, obteniendo 6.4.

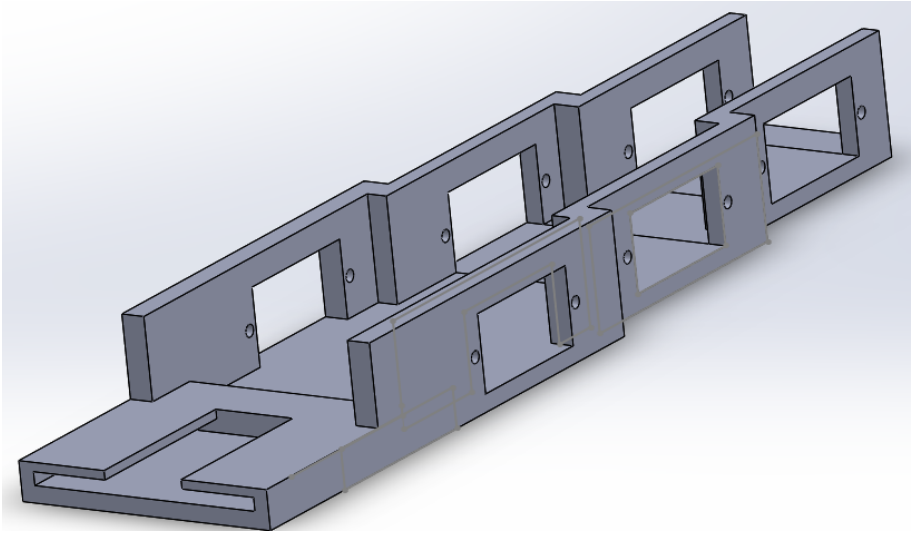


Figura 6.4 Armazón para sujetar motores e introducirse en la mano.

El resultado final, junto a los servomotores acoplados, se aprecia en la figura 6.5.

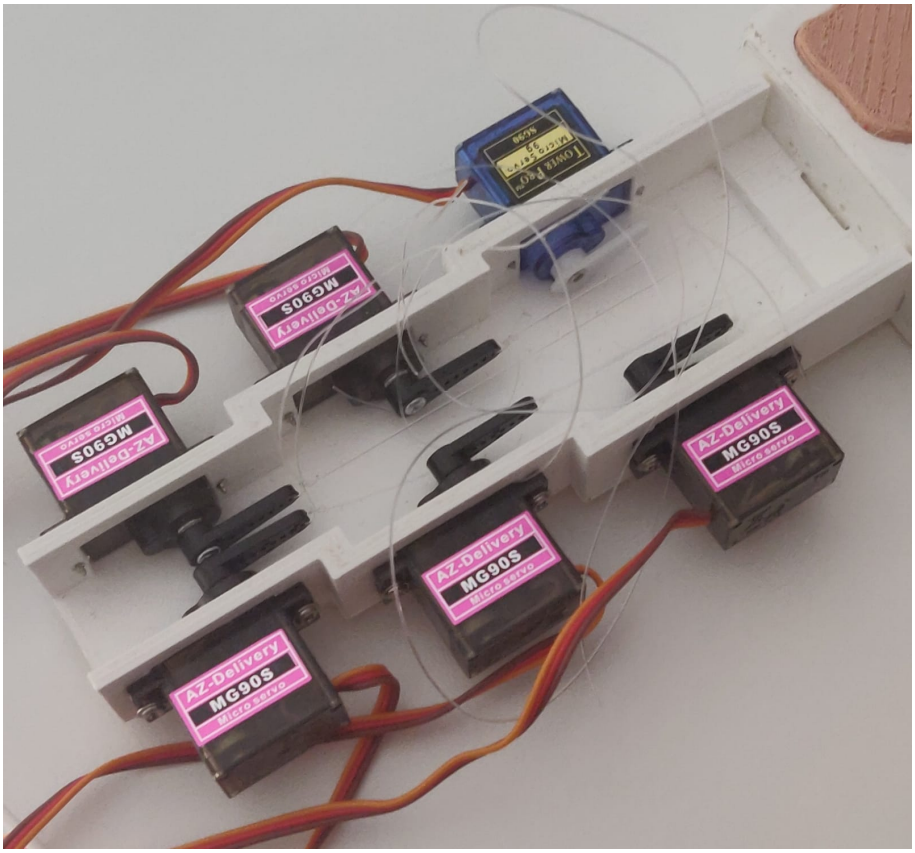


Figura 6.5 Armazón impreso.

Para la impresión del armazón y de la carcasa se han utilizado los siguientes parámetros:

Tabla 6.1 Perfil armazón y antebrazo.

Parámetro	Valor
Impresora	Ultimaker®3 extended
Alto de capa	0.2 mm
Ancho de pared	1.8 mm
Relleno	60 %
Material	PLA (210 ºC)
Velocidad	60 mm/s
Soportes	No

6.2 Antebrazo

Siguiendo los procedimientos del apartado anterior, han de diseñarse dos piezas que mantengan el microcontrolador (Arduino en esta ocasión), las baterías, el sensor EMG, los motores y el cableado necesario para esto.

La primera mitad de este elemento se diseñará para albergar el armazón y los servomotores de manera que dispongan del espacio necesario para no colisionar con ningún elemento en su trayectoria. La segunda mitad debe tener un volumen suficiente como para contener al resto de elementos y poder acoplarse a la otra mitad.

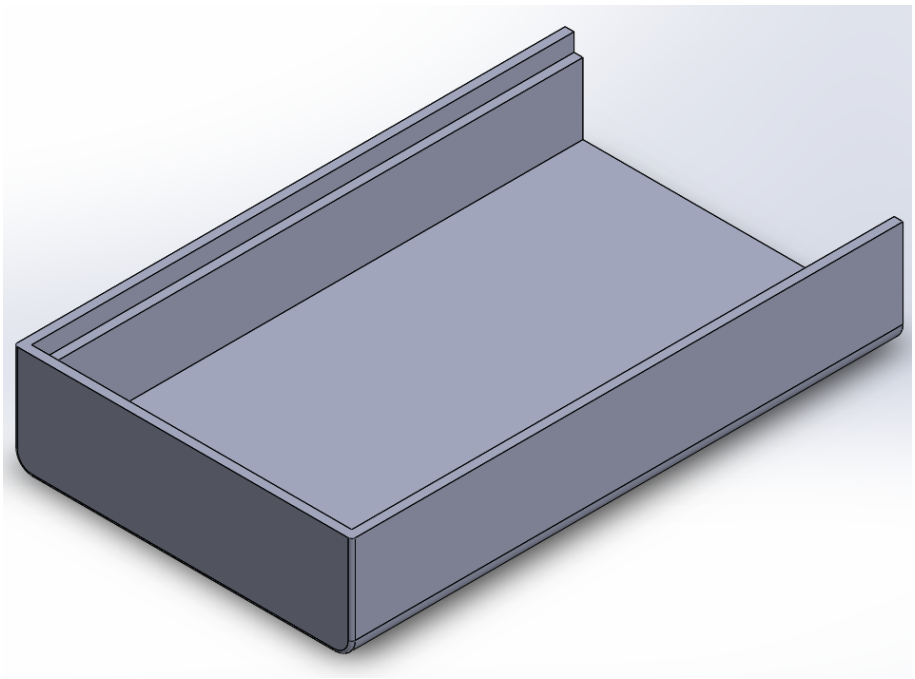


Figura 6.6 Carcasa para motores.

Como puede apreciarse en las imágenes 6.6 y 6.7, los bordes están suavizados con la herramienta "Redondeo". Además, las piezas se acoplan entre sí mediante una inserción de 5mm de una en otra.

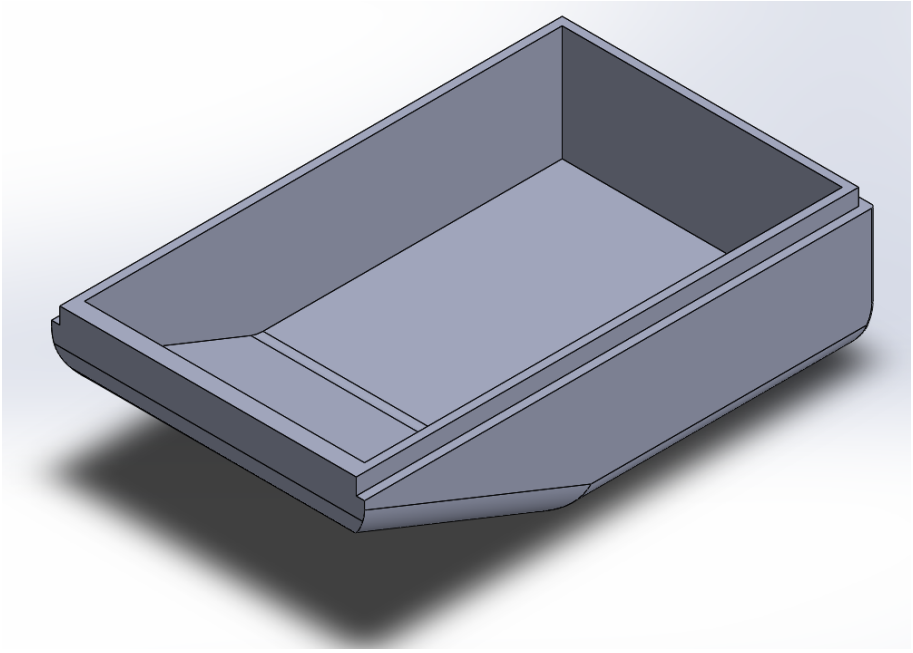


Figura 6.7 Carcasa para el resto de elementos.

La impresión de ambas piezas se realiza siguiendo los parámetros del armazón para motores, obteniendo los mismos resultados.

Dado que a priori se desconoce el caso de amputación del usuario, no se podría establecer una zona de sujeción genérica para todos los casos, además de la incomodidad que ofrecería el PLA, ABS o filaflex como acabado para la piel del usuario. Por ello se decide establecer el tamaño mínimo de la prótesis para albergar todos los componentes tal que fácilmente pueda diseñarse la unión entre usuario y prótesis.

6.3 Elementos del antebrazo

Además de las tres piezas mostradas anteriormente, en el antebrazo se encuentran varios componentes que permiten el correcto funcionamiento de esta.

Como se observa en la imagen 6.5, hay seis servomotores, uno por cada dedo a excepción del pulgar, que tiene dos. Para accionar esos servomotores es necesario alimentarlos entre 4.8V y 7.2V [30], preferiblemente, de una fuente portátil. En el caso de esta prótesis, se ha probado a alimentarlos con cuatro pilas R6 en serie de manera que se dispongan de 6V a 2550 mAh de capacidad. A 6V los servomotores suelen consumir unos 700mA como pico de corriente, que se dará generalmente en esta aplicación, pues estos picos se dan cuando se requieren mayores momentos de fuerza, como por ejemplo, al girar completamente el servomotor.

Para aplicaciones donde no se dan dichos picos en el mismo instante, esta solución es adecuada, aunque no es la más cómoda para el usuario si las baterías están dentro de la carcasa de la prótesis. Además, en caso de querer coordinar más de tres servomotores, teniendo en cuenta que el Arduino nano solo consume 15 mA, la batería no sería capaz de alimentarlos, generando una pérdida de corriente en el Arduino que conllevaría la pérdida de los PWM con los que se controlan a los servomotores.

Para facilitar el acceso al usuario a reponer la batería de la prótesis a la par que solucionar la carencia de corriente y capacidad de las pilas, se ha optado por utilizar las baterías internas de una *Powerbank* de 10000 mAh de capacidad, siendo además capaz de suministrar suficiente corriente a los seis servomotores y pudiendo recargarse mediante micro-usb. Pese a que el voltaje suministrado

por las baterías sea de 3.7V, incluye en una pcb un regulador para ofrecer 5V en la salida, además de permitir la recarga de las baterías.

Dicha batería alimenta por separado a los servomotores y al Arduino. Debido a que Arduino tiene un fusible de 500mA, no podría hacerse pasar la corriente de los seis servomotores por este. Se incluye también un interruptor para cortar la corriente ofrecida por las baterías de manera que pueda apagarse o reiniciarse la prótesis en su totalidad.

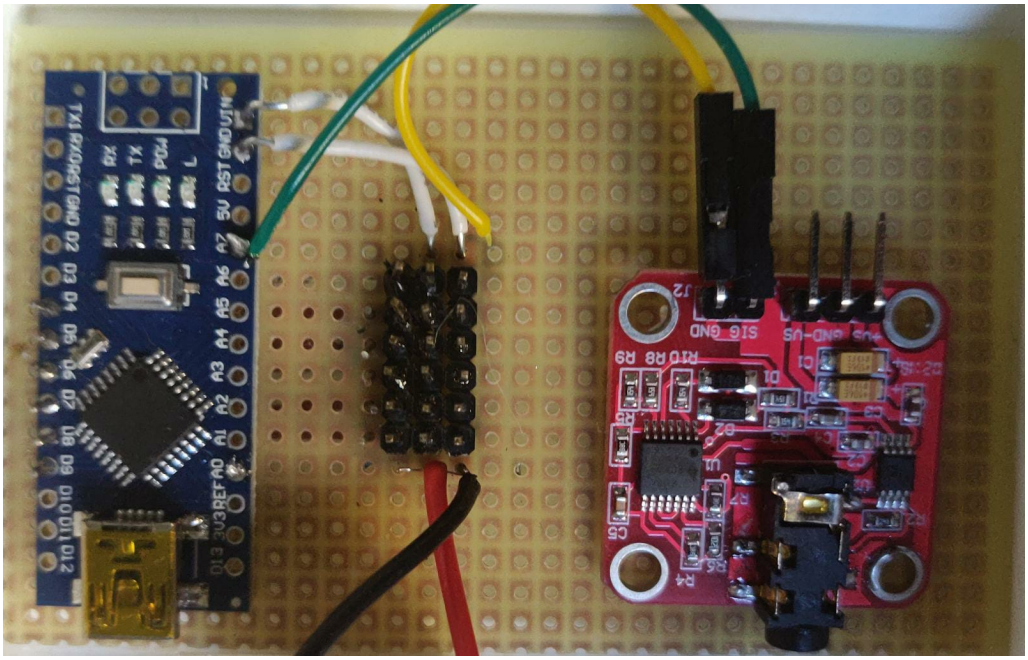


Figura 6.8 Arduino y sensor EMG soldados en PCB de prototipado, junto a pines de servomotores.

Otros elementos que deben ir en el interior del antebrazo son el sensor EMG, las dos baterías de 9V y el propio microcontrolador. Dicho microcontrolador será programado una finalidad específica, que se detallará en el siguiente capítulo.

7 Modos de funcionamiento de la prótesis

Principalmente se han diseñado dos modos de funcionamiento para la prótesis en función de la aplicación y uso que se requiera. Varios estudios como [31] o [32] proponen métodos para validar el agarre y las diferentes posiciones que cada prótesis puede lograr. Sin embargo, como la prótesis utilizada es la misma que en [32], no se realizará un análisis de esta prótesis.

Por otro lado, dado el enfoque comercial de este proyecto, se expondrá cada modo de funcionamiento desarrollado y el código de este.

7.1 Agarre completo

La abdución completa de los dedos permite realizar un agarre sencillo de numerosos objetos. Es por ello que el principal modo de funcionamiento, partiendo de que solo se dispone de un sensor de bajo coste, será este, pues es algo complicado realizar el movimiento de cada dedo mediante un solo sensor.

Tal y como puede observarse en el capítulo 3, la medida que adquiere el sensor al mantener contraído el antebrazo no es nada lineal debido al ruido, sin embago su RMS, valor medio o los picos de esta señal son algo más lineales . Por ello, se ha realizado un código que, en función de la detección de picos o, por otro lado, en función del valor medio, sea capaz de discernir si hay contracción voluntaria o no.

El cálculo del MAV en arduino se realiza mediante el código 7.1.

Código 7.1 Código de arduino para calcular el MAV.

```
// Para realizar MAV continuo, se calculan con las ultimas x muestras,
// por lo que en cada punto se resta la muestra que salga de la
    ventana
// y se añade la nueva
total = total - medidas[indice];
//medida unitaria
medidas[indice] = analogRead(sensor);
// añadimos la medida al total
total = total + medidas[indice];
// aumentamos indice
indice = indice + 1;
// para reiniciar indice del vector
```

```
  if (indice >= numero_medidas) {
    indice = 0;
  }
  // por errores de calculo a veces llega a negativo
  if (total < 0) {
    total = 0;
  }
  // Calculo del MAV. Como la medida analogica de arduino es absoluta,
  // no ha de hacerse otra vez
  MAV = total / numero_medidas;
```

En el código, *medidas[]* es un vector de longitud determinada donde se va almacenando cada muestra que se lee en arduino. Estas muestras se van acumulando en una variable *total*, restándole el valor almacenado previamente en el índice de *medidas[]* correspondiente y incluyendo en este el nuevo valor medido, recorriendo constante el vector. Para cada iteración, se calcula el MAV simplemente dividiendo la variable total entre el numero de medidas, que es el tamaño del vector *medidas[]*.

Para programar en arduino con los 6 sensores, se ha utilizado la librería <Servo.h> propia de este. Dicha librería permite utilizar hasta 12 servos modificando las salidas digitales para poder utilizarlas como PWM.

Tras la declaración e inicialización de todos los servomotores y variables, se controlan a todo o nada en función de si se ha superado el umbral en un determinado intervalo de tiempo. Si se observan las imágenes del capítulo 3 donde se contraía el puño en ambos sujetos con detalle, se puede observar que las muestras que tienen un valor alto son picos relativos cuyas muestras inmediatamente circundantes tienen un valor muchísimo mas bajo, y no se dan dichos picos en intervalos fijos.

Una solución propuesta para ello resulta realizar un bucle *do while* en el cual se entre si se supera un determinado valor, y del cual no se pueda salir si en un intervalo *timeout* se ha vuelto a alcanzar este valor. Dicho código es 7.2.

Código 7.2 Código de arduino para modo de funcionamiento inicial.

```
unsigned long timeout = 300; //300 ms va bien
unsigned long time_act = millis();
//posicion relajada por defecto
servo_1.write(1);
servo_2.write(1);
servo_3.write(1);
servo_4.write(179);
servo_5.write(179);
servo_6.write(179);

if (valor_sensor > 200){ //si supera valor en la entrada analogica
  do {
  servo_1.write(179); //todo o nada, si se supera min, se contrae
      completamente
  servo_2.write(179);
  servo_3.write(179);
  servo_4.write(1);
  servo_5.write(89);
```

```
  servo_6.write(89);
  valor_sensor = analogRead(sensor_1);
  if (valor_sensor > 200){ //si se detecta un solo pico por encima del
      valor min, se resetea timeout
    time_act = millis();
  }
  } while (millis() - time_act < timeout); //mientras no se supere el
      timeout
}
```

La prueba de qué intervalo utilizar y qué valor tomar como umbral ha de realizarse para cada sujeto y, en este caso, se realizó gracias al monitor serie de arduino. En este caso se puede apreciar que el valor umbral se ha establecido en 200 y que el timeout en 300 milisegundos. El valor de timeout podría ser menor, permitiendo mayor velocidad para extender los dedos, pero no se considera necesaria la velocidad en ese aspecto y podría ayudar a no extender la mano involuntariamente.

También se puede observar que, dada la distribución de los servomotores en el antebrazo, las posiciones iniciales y finales difieren entre estos, además de que los servomotores utilizados no son exactamente el mismo modelo en todos los dedos y que el pulgar consta de dos servomotores que no pueden alcanzar los 180 grados por el pequeño recorrido que realiza el pulgar.

7.2 Movimiento por pulsos

Como opción para el movimiento de dedos individualmente, se incluye un método que mueva un determinado dedo según el número de pulsos que se dé con el músculo flexor del antebrazo en un determinado intervalo de tiempo. Para facilitar los movimientos, se incluye la opción de contraer y extender todos los dedos.

Una clara ventaja de esto, además de poder mover cada dedo, es la facilidad para fijar una postura sin esfuerzo para mantenerla, dado que en el caso anterior, para cerrar la mano había que mantener contraído el músculo flexor del antebrazo.

Para ello se va a tratar cada dedo con dos estados (extendido y contraído) de manera que los pulsos correspondientes a cada dedo cambien este, y con un solo pulso se pueda cambiar el estado de todos los dedos a la vez. El número de pulsos correspondiente a cada acción se muestra en la tabla 7.1.

Tabla 7.1 Número de pulsos por acción.

dedo	nº de pulsos
pulgar	2
índice	3
corazón	4
anular	5
menique	6
Todos	1

El código en este modo consiste en el cálculo del MAV en cada iteración del *loop*, tal y como se expuso en el código 7.1, seguido del código 7.3 cuando detecte que se ha superado el límite establecido. Una vez detectado, se seguirán tomando medidas y calculándose el MAV durante un determinado *timeout*. Dicho timeout se reseteea si vuelve a superarse el límite una vez se haya

dejado de superar. Siempre que se detecte un nuevo pulso, la variable *posicion* se incrementa en uno, llevando así la cuenta de pulsos que se dan para poder actuar en funcion de estos.

Código 7.3 Código de arduino para detectar pulsos.

```
if (MAV > limite) { //si supera valor en la entrada analogica (200 es
    aceptable)
  do {
    if (posicion == 0) {
      posicion = 1;
    }
    total = total - medidas[indice];
    medidas[indice] = analogRead(sensor);
    total = total + medidas[indice];
    indice = indice + 1;
    if (indice >= numero_medidas) {
      indice = 0;
    }
    if (total < 0) {
      total = 0;
    }
    MAV = total / numero_medidas;
    if (MAV > limite) {
      if ( MAV1<= limite) { //si se detecta un solo pico por encima
          del valor min, se resetea timeout
        time_act = millis();
        posicion = posicion + 1;
        delay(500);
      }
    }
    MAV1 = MAV; //actualiza MAV_ant
  } while (millis() - time_act < timeout); //mientras no se supere el
      timeout
}
```

Tras calcular *posicion* en el bucle anterior, se pasa a un *switch case* que dependa de posición, donde en función del valor de esta, se cambia el estado del dedo correspondiente o, en caso de valer 1, todos los dedos. Un código del dedo pulgar como ejemplo se muestra en 7.4.

Código 7.4 Código de arduino para cambiar estados.

```
  case 2:  // 2 pulsos: cambia estado pulgar
    if (dedo_pulgar == 0) {
      dedo_pulgar = 1;
    } else {
      dedo_pulgar = 0;
    }
    posicion = 0;
    break;
```

Tras cambiar el estado del dedo (1 contraído, 0 relajado) se da valor nulo a *posicion* para no cambiar más estados hasta que el bucle para detectar pulsos cambie dicho valor.

Una vez terminado el *switch case* se procede con una serie de bucles *if* a actuar sobre los servomotores en función de los estados de cada dedo. Se muestra en el ejemplo 7.5 el caso del pulgar, pues las posiciones iniciales y finales ya han sido mostradas en 7.2.

Código 7.5 Código de arduino para actuar sobre servomotores.

```
if (dedo_pulgar == 1) {
  // contrae
  servo_6.write(89);
  servo_5.write(89);
} else {
  // relaja
  servo_6.write(179);
  servo_5.write(179);
}
```

7.3 Autocalibración

Tal y como se ha expresado en los dos anteriores modos, la determinación del umbral para contracción de los dedos se realiza para cada cliente, pero con un pequeño cálculo basado en el MAV (o en su defecto podría ser la amplitud o picos) puede determinarse un valor válido para determinar si hay intención de movimiento o no.

El código consistiría en comprobar en bucle durante un determinado tiempo (*tiempo calibrar* en el código) desde que comience el bucle *loop*, para que solo se dé durante la primera ejecución del bucle. En caso de utilizar el MAV como característica, ha de calcularse este en cada iteración del bucle *do while*. Tras ello, se comprueba si se ha superado el valor máximo hasta esta iteración y, en caso de superarse, se sustituye por el nuevo valor.

Código 7.6 Código de arduino para calibrar con MAV.

```
if (millis() - time_init < tiempo_calibrar) {
  do {
    //Calculo de MAV
    total = total - medidas[indice];
    medidas[indice] = analogRead(sensor);
    total = total + medidas[indice];
    indice = indice + 1;
    if (indice >= numero_medidas) {
      indice = 0;
    }
    MAV = total / numero_medidas;

    //actualiza maximo si se da
    if (MAV > MAV_max_calib) {
      MAV_max_calib = MAV;
      delay(1);
    }
```

```
      //establece el limite en el maximo detectado
      limite = MAV_max_calib;
      delay(1);
    } while (millis() - time_init < tiempo_calibrar);
  }
```

El tiempo de calibración se ha establecido en 5 segundos, y durante este tiempo se almacenan valores máximos, por lo que sería necesario ejercer fuerza moderada en el brazo para establecer un límite consistente, pero no imposible de alcanzar.

Para el caso de la autocalibración en el primer modo mostrado, solo basta con realizar este mismo bucle sin el cálculo del MAV.

Código 7.7 Código de arduino para calibrar con MAV.

```
if (millis() - time_init < tiempo_calibrar) {
do {
  valor_sensor = analogRead(sensor_1);
      //actualiza maximo si se da
      if (valor_sensor > valor_sensor_calib) {
        valor_sensor_calib = valor_sensor;
        delay(1);
      }
      //establece el limite en el maximo detectado
      limite = valor_sensor_calib;
      Serial.println(limite);
      delay(1);
    } while (millis() - time_init < tiempo_calibrar);
}
```

8 Coste de la prótesis

Pese al análisis y clasificación de los sensores de alta presición de Trigno, el objetivo de este proyecto es realizar una prótesis minimizando el coste en medida de lo posible. Por ello, en el producto final no se incluyen estos, sino un sensor EMG que podría producirse en serie por minimizar aún más los costes.

Tabla 8.1 costes de cada pieza de la prótesis.

	gramos	minutos	coste plastico(€)	coste energia(€)	coste total (€)
pulgar 1	9	128	0,27	0,06600533	0,33600533
pulgar 2	4	76	0,12	0,03919067	0,15919067
indice 1	8	115	0,24	0,05930167	0,29930167
indice 2	4	58	0,12	0,02990867	0,14990867
indice 3	2	45	0,06	0,023205	0,083205
corazon 1	7	108	0,21	0,055692	0,265692
corazon 2	4	57	0,12	0,029393	0,149393
corazon 3	2	39	0,06	0,020111	0,080111
anular 1	6	86	0,18	0,04434733	0,22434733
anular 2	3	51	0,09	0,026299	0,116299
anular 3	2	38	0,06	0,01959533	0,07959533
meñique 1	4	66	0,12	0,034034	0,154034
meñique 2	2	34	0,06	0,01753267	0,07753267
meñique 3	1	29	0,03	0,01495433	0,04495433
mano 1	11	164	0,385	0,08456933	0,46956933
mano 2	93	725	2,79	0,37385833	3,16385833
juntas	16	532	0,8	0,27433467	1,07433467
yema 1	3	159	0,09	0,081991	0,171991
yema 2	1	56	0,03	0,02887733	0,05887733
yema 3	1	59	0,03	0,03042433	0,06042433
yema 4	1	60	0,03	0,03094	0,06094
yema 5	1	34	0,03	0,01753267	0,04753267
almohadillas	9	256	0,27	0,13201067	0,40201067
armazón	36	254	1,08	0,13097933	1,21097933
antebrazo 1	85	377	2,55	0,19440633	2,74440633
antebrazo 2	101	468	3,03	0,241332	3,271332

Para la impresión de la prótesis se han utilizado, en su versión final, únicamente PLA y filaflex, teniendo un precio por kilogramo, respectivamente, de 31.93€ y 52 €. El precio de estos materiales

puede variar según marcas, este es el precio de los materiales usados en la prótesis del proyecto. Sin embargo, este precio podría reducirse casi a la mitad si se utilizasen diferentes marcas.

Para cada elemento de la prótesis se ha calculado su coste en material, simplemente calculando su peso por el coste de material, y su coste eléctrico, mediante el consumo medio de las impresoras (221 W).

Para los elementos del sensor se ha buscado el precio de cada componente por páginas como mouser, texas instruments o RS components, y el precio adoptado ha sido el de bobinas de 1000 elementos, por lo que el precio va orientado a la producción de un gran número de dispositivos.

Para resistencias y condensadores, como sus valores son diferentes, su precio no es exactamente el mismo, sin embargo, apenas difieren, por lo que se ha optado por tomar un precio medio para este caso.

Tabla 8.2 Costes de los componentes del sensor.

	unidades	precio unitario (€)	total (€)
AD	1	2,75	2,75
OPAM (encapsulado de 4)	1	0,09	0,09
resistencias	10	0,02	0,2
conector mini-jack	1	1,44	1,44
condensadores	5	0,05	0,25
diodo	2	0,05	0,1
		total	4,83 €

Sin embargo a este precio de componentes habría que añadir el coste de los electrodos, los cables de estos, la PCB y el ensamblaje de la PCB. Aun siendo un coste diez euros menor al sensor completo, como en este trabajo se ha utilizado el sensor ya ensamblado, se tendrá en cuenta para el coste final sólo el coste del sensor ya ensamblado.

Tabla 8.3 Coste total.

ATMEGA328	2 €
batería 5V recargable	10,8 €
sensor EMG	14,49 €
2 baterías 9V recargables	3,13 €
prótesis completa	14,955826 €
6 servomotores	5,16 €
total	49,605826 €

El cálculo de los costes de la prótesis se ha realizado únicamente para los costes materiales de esta, sin embargo, el coste de la prótesis para una empresa tendría muchísimos más factores y un plan avanzado de costes que, al no ser objetivo del trabajo, no se realizará.

9 Conclusiones y futuras líneas de trabajo

9.1 Conclusiones

La prótesis IMMA es una gran aproximación a una mano humana, con más restricciones cinemáticas que esta, pero sin perder apenas funcionalidad. Sin embargo, solo ofrece la posibilidad de controlarla mediante seis actuadores mediante hilos, perdiendo el control sobre cada articulación.

Esto es en parte beneficioso para nuestro caso, pues no es tan interesante la posibilidad de mover cada falange por separado sino poder actuar sobre la mano como tal mediante pocas señales de control. Dicho de otro modo, al disponer de un solo sensor EMG, ampliable a 5, no tiene especial interés la posibilidad de actuar sobre cada uno de los 15 grados de libertad que dispone la prótesis IMMA.

El sensor EMG implementado en la versión actual del trabajo es un sensor de bajo coste y de poca precisión, como ya se detalló en el capítulo 3. Esto implica que no se podría realizar un control proporcional a la entrada, proporcional a la medida de este sensor, pues pese a filtrarse, se aprecia que al mantener la contracción, la señal oscila, lo que impediría mantener la prótesis fija en una posición determinada. Si se disponiese de mayor presupuesto para la prótesis, se podría estudiar el uso de sensores trigno conectados mediante bluetooth a un microcontrolador, pudiendo no depender de cables y facilitando mucho el control gracias a la gran precisión que estos ofrecen.

El clasificador lineal implementado parte de unas medidas excelentes, por lo que, de usarse para clasificar mediante cinco sensores EMG de baja precisión, podría degradar mucho su fiabilidad. O bien habría que hacer un segundo estudio para un clasificador con cinco sensores EMG de baja precisión, siguiendo las mismas pautas que se han seguido para realizar el clasificador del apartado 5, o bien afinar el clasificador actual para las medidas que estos sensores ofrecerían.

En cuanto al control que se puede realizar actualmente, ha de puntualizarse que al ser todo-nada y en bucle abierto, al agarrar objetos muy grandes podrían quemarse los motores, pues al darle un pulso PWM para rotarlo completamente, si está obstruído el dedo, el motor consumirá mayor corriente hasta que consiga rotar, bien restándole corriente al microcontrolador implicando el reinicio del sistema, o bien obteniendo mayor corriente que la máxima soportada, pudiendo derivar en la pérdida de dicho servomotor. Es por ello que las pruebas realizadas son únicamente en objetos de pequeño tamaño.

Además, precisamente por la misma razón, la prótesis no aplica mucha fuerza para agarrar objetos, pues a mayor presión, mas par realizan los motores, y si se supera el par máximo para estos, comenzará a consumir más potencia, solicitando más corriente a las baterías.

Bajo pocas pruebas, la prótesis como tal resiste adecuadamente a diversos esfuerzos, siendo en primer lugar los tendones el primer elemento en ceder (por una leve elasticidad del material), seguido de los servomotores de bajo coste y bajas prestaciones. Los motores solo cederán en el caso explicado en el párrafo anterior, ya sea por reinicio del controlador o por fallo de estos.

Con respecto a los modos de funcionamiento que ofrece la versión actual, el modo que funciona por impulsos es mucho más fiable que el otro debido a que los sensores reaccionan mejor a impulsos frente a mantener contracciones. Si agarrando un objeto, el sensor da un pico de señal menor al límite establecido, que puede darse por la oscilación de valores en el sensor, la mano se abriría una cantidad de tiempo establecida por *timeout*, cerrándose justo después pero habiendo soltado el objeto agarrado. De ser este el caso, la mejor solución es aumentar *timeout* hasta que se minimice la pérdida de picos, derivando en mayor retardo para abrir y cerrar la mano.

Para el caso de funcionamiento mediante pulsos, el único problema posible sería la detección de pulsos no voluntarios, es decir, que levemente se supere el límite establecido inconscientemente, aumentando en uno los pulsos medidos, implicando el cambio de estado de un dedo no deseado, o, de solo detectar uno, la mano entera. Este problema no es tan usual como el anterior, pero sigue siendo igualmente un gran inconveniente no solucionado.

9.2 Futuras líneas de trabajo

Todos los objetivos iniciales del trabajo se han cumplido, sin embargo, hay muchos aspectos del trabajo que pueden ser mejorados aprovechando que la prótesis ya está impresa.

En primer lugar, la implementación de los sensores Trigno no se ha llevado a cabo, una gran modificación de este trabajo sería conseguir conectar los sensores Trigno al microcontrolador de la prótesis, pudiendo conectar varios e implementar el sistema de clasificación desarrollado en el trabajo, pese a no poder considerarse entonces una prótesis de bajo coste.

Otra posible mejora sería diseñar un sensor EMG según los elementos del capítulo 3 de este trabajo, de manera que puedan diseñarse 5 sensores y lograr controlar cada dedo con cada sensor, adaptando el sistema de clasificación diseñado a estos nuevos sensores.

Además, podrían realizarse ensayos mecánicos a la prótesis, imprimiendo varias de estas, para comprobar los límites que ofrece la prótesis como tal, incluyendo o no la electrónica que esta conlleva para comprobar los pares reales que ofrecen los motores o la fuerza máxima que puede aplicar cada dedo.

Para poder controlar la prótesis en bucle cerrado, podrían implantarse sensores piezoeléctricos en las yemas de los dedos, entre las piezas de PLA y filaflex, de manera que se pudiese realimentar los esfuerzos en esta zona para ajustar la presión deseada en el agarre. Es estrictamente necesario para poder realizar operaciones con mayor precisión desarrollar este aspecto, o bien dotar a los sensores de muchisima más precisión y rapidez en el microcontrolador como para que el usuario sea capaz de poder aplicar la presión deseada en el agarre sin depender de dichos sensores piezoeléctricos.

Una característica común de otros trabajos actuales similares a este supone el desarrollo de redes neuronales para construir un modelo que sirva como clasificador. No se ha realizado en este trabajo por la facilidad con la que se separaban las muestras de los sensores Trigno, pero de utilizarse varios sensores EMG básicos, muy probablemente sea necesaria esta característica por el ruido que estos implican.

Apéndice A
Códigos

Código A.1 Código de adquisición por puerto serie simple.

```
int val;
static long t_ini;

void setup()
{
  val = 0;
  pinMode(0, INPUT);
  Serial.begin(9600);
  t_ini = millis();
}

void loop()
{
  val = analogRead(0);
  Serial.print(millis() - t_ini); //Para medir tiempo desde inicio
  Serial.print("\t");
  Serial.println(val);
}
```

Código A.2 Código control todo o nada de un servo (BA).

```
#include <Servo.h> // Incluimos la biblioteca Servo

Servo servo_1; // Definimos los servos que vamos a utilizar

int sensor_1 = 0; // Pin usado para conectar el sensor
int valor_sensor;  // Esta variable definirá la posición del servo

void setup() {
servo_1.attach(9); // Definimos el pines de señal para el servo
Serial.begin(9600);
}
```

```
void loop() {

valor_sensor = analogRead(sensor_1);
// leemose valor de entrada analogica (valor entre 0 y 1023)
Serial.println(valor_sensor);
unsigned long timeout = 300; //300 ms va bien
unsigned long time_act = millis();
servo_1.write(0); //posicion relajada por defecto

if (valor_sensor > 200){ //si supera valor en la entrada analogica (200
    es aceptable)
  do {
  servo_1.write(179); //todo o nada, si se supera min, se contrae
      completamente
  valor_sensor = analogRead(sensor_1);
  if (valor_sensor > 200){ //si se detecta un solo pico por encima del
      valor min, se resetea timeout
    time_act = millis();
  }
  } while (millis() - time_act < timeout); //mientras no se supere el
      timeout
}
    Serial.println(analogRead(0));
// Esperamos para reiniciar el bucle
}
```

Código A.3 Código de matlab para representar medidas.

```
clear all;
file=load('puño_entero.log');
y=file(:,2);
x=file(:,1);

 Fs=50;                   % frecuencia de las muestras
 Fnorm = 1/(Fs/2);         % frecuencia normalizada
 df = designfilt('lowpassfir','FilterOrder',40,'CutoffFrequency',Fnorm);
 D = mean(grpdelay(df)) % retraso del filtro
 y1 = filter(df,[y; zeros(D,1)]); %añadimos tantos ceros como retraso
     tenga el filtro y filtramos
 y1 = y1(D+1:end); %compensamos retraso
%[y1,y2] = envelope(y,40,'peak'); %para ver envolvente sup/inf
figure(1);
plot(file(:,1),file(:,2)); hold on;
plot(file(:,1),y1,'r','linewidth',1.5);
xlabel('Tiempo (ms)');
title('Sensores');
```

Código A.4 Código de matlab para clasificar medidas.

```
%clear all; %><
indice=load('ind_con_cor.txt');
corazon=load('cor_con_ind.txt');
ambos = load('ind_y_cor.txt');
ninguno = load('ind_cor_relajado.txt');
L=200; %L es el tamaño de la ventana
for j = 1:length(indice(:,1))-L %%indice contraccion
    sum = 0;
    for i = 1:L
        sum = sum + abs(indice(i+j-1,2));
    end
    MAVind(j+L/2) = sum/L;
end

for j = 1:length(indice(:,1))-L %%indice contraccion
    sum = 0;
    for i = 1:L
        sum = sum + abs(indice(i+j-1,3));
    end
    MAVind_cor(j+L/2) = sum/L;
end

for j = 1:length(corazon(:,1))-L %%corazon contraccion
    sum = 0;
    for i = 1:L
        sum = sum + abs(corazon(i+j-1,3));
    end
    MAVcor(j+L/2) = sum/L;
end

for j = 1:length(corazon(:,1))-L %%corazon contraccion
    sum = 0;
    for i = 1:L
        sum = sum + abs(corazon(i+j-1,2));
    end
    MAVcor_ind(j+L/2) = sum/L;
end

for j = 1:length(ambos(:,1))-L %%contracción de ambos
    sum_ind = 0;
    sum_cor = 0;
    for i = 1:L
        sum_ind = sum_ind + abs(ambos(i+j-1,2));
        sum_cor = sum_cor + abs(ambos(i+j-1,3));
    end
    MAVambos_ind(j+L/2) = sum_ind/L;
    MAVambos_cor(j+L/2) = sum_cor/L;
end
```

```
for j = 1:length(ninguno(:,1))-L %%contracción de ambos
    sum_ind = 0;
    sum_cor = 0;
    for i = 1:L
        sum_ind = sum_ind + abs(ninguno(i+j-1,2));
        sum_cor = sum_cor + abs(ninguno(i+j-1,3));
    end
    MAVninguno_ind(j+L/2) = sum_ind/L;
    MAVninguno_cor(j+L/2) = sum_cor/L;
end

MAVind_sampled = interp1(MAVind, linspace(1,length(MAVind), length(
    MAVind)/700));
ind_cor_sampled = interp1(MAVind_cor, linspace(1,length(MAVind_cor),
    length(MAVind_cor)/700));
%ind_cor_sampled = interp1(indice(:,3), linspace(1,length(indice(:,3)),
    length(indice(:,3))/700)); %%sensor cor al medir ind
MAVcor_sampled = interp1(MAVcor, linspace(1,length(MAVcor), length(
    MAVcor)/700));
cor_ind_sampled = interp1(MAVcor_ind, linspace(1,length(MAVcor_ind),
    length(MAVcor_ind)/700));
%cor_ind_sampled = interp1(corazon(:,2), linspace(1,length(corazon(:,2))
    , length(corazon(:,2))/700));%%sensor ind al medir cor

MAVambos_ind_sampled = interp1(MAVambos_ind, linspace(1,length(
    MAVambos_ind), length(MAVambos_ind)/700));
MAVambos_cor_sampled = interp1(MAVambos_cor, linspace(1,length(
    MAVambos_cor), length(MAVambos_cor)/700));

MAVninguno_ind_sampled = interp1(MAVninguno_ind, linspace(1,length(
    MAVninguno_ind), length(MAVninguno_ind)/700));
MAVninguno_cor_sampled = interp1(MAVninguno_cor, linspace(1,length(
    MAVninguno_cor), length(MAVninguno_cor)/700));

all_samples_ind = [MAVind_sampled cor_ind_sampled MAVambos_ind_sampled
    MAVninguno_ind_sampled];
all_samples_cor = [ind_cor_sampled MAVcor_sampled MAVambos_cor_sampled
    MAVninguno_cor_sampled];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%funciones de decision
n1 = -[1 1]'; rho1 = -0.00003; % entre inmovil y el resto
n2 = -[-1 2]'; rho2 = 0; % entre indice y ambos
n3 = -[-2 1]'; rho3 = 0; % entre ambos y corazon


%vectores de pesos
w1 = [n1; -rho1];
w2 = [n2; -rho2];
```

```
w3 = [n3; -rho3];

figure(1);
hold on;
mal_clasificadas = 0;

for k=2:length(all_samples_ind)
    x1=abs(all_samples_ind(k));
    x2=abs(all_samples_cor(k));
    xA=[x1; x2; 1];
    %cálculo distancia
    fd1 = w1' * xA;
    fd2 = w2' * xA;
    fd3 = w3' * xA;

    if(k<length(MAVind_sampled))
        clase_original = 1;
    else
        if(k>length(MAVind_sampled) & k<(length(MAVind_sampled) + length
            (MAVcor_sampled)))
            clase_original = 2;
        else
            if(k>(length(MAVind_sampled) + length(MAVcor_sampled)) & k<(
                length(MAVind_sampled) + length(MAVcor_sampled) + length(
                MAVambos_ind_sampled)))
                clase_original = 3;
            else
                clase_original = 0;
            end
        end
    end

    if fd1>0
        clase = 0;
    else
        if fd2>0
            clase = 1;
        end
        if fd3<0
            clase = 2;
        end
        if fd2<0 & fd3>0
            clase = 3;
        end

    end

    if clase == 0
        plot (x1, x2, 'k.');
    end
```

```
    if clase == 1
        plot (x1, x2, 'r.');
    end
    if clase == 2
        plot (x1, x2, 'b.');
    end
    if clase == 3
        plot (x1, x2, 'g.');
    end

    if(clase ~= clase_original)
        mal_clasificadas = mal_clasificadas+1;
        plot (x1, x2, 'y*');
    end
end
mal_clasificadas


figure(2);
plot(MAVind_sampled(2:end),abs(ind_cor_sampled(2:end)),'r.', 'MarkerSize
    ',20);hold on;
plot(abs(cor_ind_sampled(2:end)),MAVcor_sampled(2:end),'g.', 'MarkerSize
    ',20);hold on;
plot(MAVambos_ind_sampled(2:end),MAVambos_cor_sampled(2:end),'b.', '
    MarkerSize',20);hold on;
plot(MAVninguno_ind_sampled(2:end),MAVninguno_cor_sampled(2:end),'k.', '
    MarkerSize',20);hold on;
legend('índice','corazón','ambos','ninguno');
lim=axis;
fplot(@(x)-x + 0.00003,'HandleVisibility','off');
fplot(@(x)0.5*x,'HandleVisibility','off');
fplot(@(x)2*x,'HandleVisibility','off');
axis(lim);
grid;
xlabel('índice');
ylabel('corazon');
```

Código A.5 Código de arduino para modo de agarre completo.

```
#include <Servo.h> // Incluimos la biblioteca Servo

// Definimos los servos que vamos a utilizar
Servo servo_1;
Servo servo_2;
Servo servo_3;
Servo servo_4;
Servo servo_5;
Servo servo_6;
```

```
int sensor_1 = A0; // Pin usado para conectar el sensor
int valor_sensor;  // Esta variable definirá la posición del servo

// valor a partir del cual se actuará
int limite;

//valores para temporizadores
unsigned long time_init;
unsigned long tiempo_calibrar = 5000;

int valor_sensor_calib = 0;

void setup() {
  servo_1.attach(4); // Definimos el pines de señal para el servo
  servo_2.attach(5);
  servo_3.attach(6);
  servo_4.attach(7);
  servo_5.attach(8);
  servo_6.attach(9);
  Serial.begin(9600);
  time_init = millis(); // indica fin setup
}

void loop() {
  //calibracion: define valor limite
  if (millis() - time_init < tiempo_calibrar) {
    do {
      valor_sensor = analogRead(sensor_1);
      //actualiza maximo si se da
      if (valor_sensor > valor_sensor_calib) {
        valor_sensor_calib = valor_sensor;
        delay(1);
      }
      //establece el limite en el maximo detectado
      limite = valor_sensor_calib;
      Serial.println(limite);
      delay(1);
    } while (millis() - time_init < tiempo_calibrar);
  }
  valor_sensor = analogRead(sensor_1);
  // leemose valor de entrada analogica (valor entre 0 y 1023)
  unsigned long timeout = 300; //300 ms va bien
  unsigned long time_act = millis();
  servo_1.write(1); //posicion relajada por defecto
  servo_2.write(1);
  servo_3.write(1);
  servo_4.write(179);
  servo_5.write(179);
  servo_6.write(179);
```

```
  if (valor_sensor > limite) { //si supera valor en la entrada analogica
      (200 es aceptable)
    do {
      servo_1.write(179); //todo o nada, si se supera min, se contrae
          completamente
      servo_2.write(179);
      servo_3.write(179);
      servo_4.write(1);
      servo_5.write(89);
      servo_6.write(89);
      valor_sensor = analogRead(sensor_1);
      if (valor_sensor > limite) { //si se detecta un solo pico por
          encima del valor min, se resetea timeout
        time_act = millis();
      }
    } while (millis() - time_act < timeout); //mientras no se supere el
        timeout
  }
  Serial.println(valor_sensor);
}
```

Código A.6 Código de arduino para modo por impulsos.

```
#include <Servo.h> // Incluimos la biblioteca

// numero de muestras en sumatorio de MAV
const int numero_medidas = 50;

Servo servo_1;
Servo servo_2;
Servo servo_3;
Servo servo_4;
Servo servo_5;
Servo servo_6;

int medidas[numero_medidas];   // medidas para MAV
int indice = 0;           // indice para sumatorio
int total = 0;               // max de sumatorio
int MAV = 0;             // MAV
int MAV1;              // MAV anterior

int sensor = A0;

// para elegir dedos
int posicion = 0;

// para estados de dedos
int dedo_pulgar = 0;
int dedo_indice = 0;
```

```
int dedo_corazon = 0;
int dedo_anular = 0;
int dedo_menique = 0;

int MAV_max_calib = 0;
int limite;

unsigned long time_init;
unsigned long tiempo_calibrar = 5000;

unsigned long timeout = 1000; // 300 ms va bien
unsigned long time_act;

void setup() {
  Serial.begin(9600);
    servo_1.attach(4);
    servo_2.attach(5);
    servo_3.attach(6);
    servo_4.attach(7);
    servo_5.attach(8);
    servo_6.attach(9);

  // inicializar vector de medidas (sumatorio) a 0:
  for (int ind_cero = 0; ind_cero < numero_medidas; ind_cero++) {
    medidas[ind_cero] = 0;
  }

  analogRead(sensor); // estabiliza la adquisicion a los 5 segundos
  delay(5000);
  time_init = millis();
}

void loop() {
  // calibracion
  if (millis() - time_init < tiempo_calibrar) {
    do {
      //Calculo de MAV
      total = total - medidas[indice];
      medidas[indice] = analogRead(sensor);
      total = total + medidas[indice];
      indice = indice + 1;
      if (indice >= numero_medidas) {
        indice = 0;
      }
      MAV = total / numero_medidas;

      //actualiza maximo si se da
      if (MAV > MAV_max_calib) {
        MAV_max_calib = MAV;
        delay(1);
```

```
    }
    //establece el limite en el maximo detectado
    limite = MAV_max_calib;
    Serial.println(limite);
    delay(1);
  } while (millis() - time_init < tiempo_calibrar);
}

// Para realizar MAV continuo, se calculan con las ultimas x muestras,
// por lo que en cada punto se resta la muestra que salga de la
    ventana
// y se añade la nueva
total = total - medidas[indice];

//medida unitaria
medidas[indice] = analogRead(sensor);

// añadimos la medida al total
total = total + medidas[indice];

// aumentamos indice
indice = indice + 1;

// para reiniciar indice del vector
if (indice >= numero_medidas) {
  indice = 0;
}

// no va a suceder, pero por si acaso
if (total < 0) {
  total = 0;
}

// Calculo del MAV. Como la medida analogica de arduino es absoluta,
// no ha de hacerse otra vez
MAV = total / numero_medidas;
MAV1 = MAV;
Serial.println(MAV);
delay(1);

// detectar con timeout el numero de pulsos, de manera que cuando
    detecte
// pulso reinicie timeout, y cuando no reciba pulso en timeout para de
// modo adquisicion a actuar. Tras ello, vuelve a esperar que haya
    pulso
// para adquirir mas pulsos.
time_act = millis();

if (MAV > limite) { //si supera valor en la entrada analogica (200 es
    aceptable)
```

```
  do {
    if (posicion == 0) {
      posicion = 1;
    }
    total = total - medidas[indice];
    medidas[indice] = analogRead(sensor);
    total = total + medidas[indice];
    indice = indice + 1;
    if (indice >= numero_medidas) {
      indice = 0;
    }
    if (total < 0) {
      total = 0;
    }
    MAV = total / numero_medidas;
    Serial.println(MAV);
    if (MAV > limite) {
      if ( MAV1<= limite) {
        //si se detecta un solo pico por encima del valor min, se
            resetea timeout
        time_act = millis();
        posicion = posicion + 1;
        delay(500);
      }
    }
    MAV1 = MAV; //para no volver a entrar en siguiente iteracion
  } while (millis() - time_act < timeout); //mientras no se supere el
      timeout
}

// switch case para cambiar de estados los dedos según el resultado
// del anterior do while en posicion
switch (posicion) {
  case 0:
    break;
  case 1:  // 1 pulso: mano completa, cambia cada dedo
    if (dedo_pulgar == 0) {
      dedo_pulgar = 1;
    } else {
      dedo_pulgar = 0;
    }

    if (dedo_indice == 0) {
      dedo_indice = 1;
    } else {
      dedo_indice = 0;
    }

    if (dedo_corazon == 0) {
      dedo_corazon = 1;
```

```
    } else {
      dedo_corazon = 0;
    }

    if (dedo_anular == 0) {
      dedo_anular = 1;
    } else {
      dedo_anular = 0;
    }

    if (dedo_menique == 0) {
      dedo_menique = 1;
    } else {
      dedo_menique = 0;
    }
    posicion = 0;
    break;
  case 2:  // 2 pulsos: cambia estado pulgar
    if (dedo_pulgar == 0) {
      dedo_pulgar = 1;
    } else {
      dedo_pulgar = 0;
    }
    posicion = 0;
    break;

  case 3:  // 3 pulsos: cambia estado indice
    if (dedo_indice == 0) {
      dedo_indice = 1;
    } else {
      dedo_indice = 0;
    }
    posicion = 0;
    break;
  case 4:  // 4 pulsos: cambia estado corazon
    if (dedo_corazon == 0) {
      dedo_corazon = 1;
    } else {
      dedo_corazon = 0;
    }
    posicion = 0;
    break;

  case 5:  // 5 pulsos: cambia estado anular
    if (dedo_anular == 0) {
      dedo_anular = 1;
    } else {
      dedo_anular = 0;
    }
    posicion = 0;
```

```
    break;

  case 6: // 6 pulsos: cambia estado meñique
    if (dedo_menique == 0) {
      dedo_menique = 1;
    } else {
      dedo_menique = 0;
    }
    posicion = 0;
    break;
}


// actuar sobre los servomotores correspondientes
if (dedo_pulgar == 1) {
  // contrae
  servo_6.write(89);
  servo_5.write(89);
} else {
  // relaja
  servo_6.write(179);
  servo_5.write(179);
}

if (dedo_indice == 1) {
  // contrae
  servo_4.write(1);
} else {
  // relaja
  servo_4.write(179);
}

if (dedo_corazon == 1) {
  // contrae
  servo_3.write(179);
} else {
  // relaja
  servo_3.write(1);
}

if (dedo_anular == 1) {
  // contrae
  servo_2.write(179);
} else {
  // relaja
  servo_2.write(1);
}

if (dedo_menique == 1) {
  // contrae
```

```
    servo_1.write(179);
  } else {
    // relaja
    servo_1.write(1);
  }
}
```

Índice de Figuras

Índice de Tablas

Índice de Códigos

Bibliografía

[1] "Ingeniería biónica - Wikipedia, la enciclopedia libre." [Online]. Available: https://es.wikipedia.org/wiki/Ingeniería_biónica

[2] G. Jeong, Y. Kim, W. Choi, G. Gu, H. Lee, M. B. Hong, and K. Kim, "On the design of a novel underactuated robotic finger prosthesis for partial hand amputation," in *2019 IEEE 16th International Conference on Rehabilitation Robotics (ICORR)*, 2019, pp. 861–867.

[3] K. A. Raichle, M. A. Hanley, I. Molton, N. J. Kadel, K. Campbell, E. Phelps, D. Ehde, and D. G. Smith, "Prosthesis use in persons with lower- and upper-limb amputation." in *Journal of rehabilitation research and development*, vol. 45, no. 7, 2008, p. 961–972.

[4] World Health Organisation (WHO) and International Society for Prosthetics and Orthotics (ISPO), "Guidelines for training personnel in developing countries for prosthetics and orthotics services," *WHO Library Cataloguing-in-Publication Data*, pp. 1–57, 2005. [Online]. Available: http://apps.who.int/iris/bitstream/10665/43127/1/9241592672.pdf

[5] "sofisticación en el antiguo egipto: una prótesis de hace 3.000 años." [Online]. Available: https://historia.nationalgeographic.com.es/a/sofisticacion-antiguo-egipto-protesis-hace-3000-anos_11639

[6] K. Norton, "A brief history of prosthetics," *InMotion*, vol. 17, no. 7, pp. 11–3, 2007.

[7] "Design and evaluation of anthropomorphic hands by using grasping simulation. application to the design and control of prosthethic hands." [Online]. Available: https://sites.google.com/a/uji.es/devalhand/

[8] L. Liow, A. B. Clark, and N. Rojas, "OLYMPIC: A modular, tendon-driven prosthetic hand with novel finger and wrist coupling mechanisms," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 299–306, 2020.

[9] "Impresión 3d - wikipedia, la enciclopedia libre." [Online]. Available: https://es.wikipedia.org/wiki/Impresión_3D

[10] I. L. Harillo, A. P. González, J. C. Ramis, F. J. Andrés, and U. J. I. Uji, "Evaluación y comparación de manos protésicas de impresión 3D mediante el Anthropomorphic Hand Assessment Protocol (AHAP) Material y métodos Resultados y discusión," pp. 24–25, 2019.

[11] L. Serna C., A. Rodríguez de S., and F. Albán A., "Ácido poliláctico (pla): Propiedades y aplicaciones," *INGENIERÍA Y COMPETITIVIDAD*, vol. 5, no. 1, pp. 16–26, jun. 2011. [Online]. Available: https://revistaingenieria.univalle.edu.co/index.php/ingenieria_y_competitividad/article/view/2301

[12] R. J. Schwarz and C. Taylor, "The anatomy and mechanics of the human hand," *Artificial limbs*, vol. 2, no. 2, pp. 22–35, 1955.

[13] A. Arjun, L. Saharan, and Y. Tadesse, "Design of a 3D printed hand prosthesis actuated by nylon 6-6 polymer based artificial muscles," *IEEE International Conference on Automation Science and Engineering*, vol. 2016-Novem, no. c, pp. 910–915, 2016.

[14] A. Barrientos, L. F. Peñin, C. Balaguer, and R. Aracil, *Fundamentos de robótica*. McGraw-Hill Madrid, 2007, vol. 2.

[15] F. Alkhatib, E. Mahdi, and J. J. Cabibihan, "Design and analysis of flexible joints for a robust 3d printed prosthetic hand," *IEEE International Conference on Rehabilitation Robotics*, vol. 2019-June, pp. 784–789, 2019.

[16] E. F. Shair, A. R. Abdullah, T. N. S. Tengku Zawawi, S. Ahmad, and S. Salleh, "Auto-segmentation analysis of emg signal for lifting muscle contraction activities," *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)*, vol. 8, pp. 17–22, 10 2016.

[17] K. T. Reilly and M. H. Schieber, "Incomplete functional subdivision of the human multitendoned finger muscle flexor digitorum profundus: an electromyographic study," *Journal of neurophysiology*, vol. 90, no. 4, pp. 2560–2570, 2003.

[18] C. Cipriani, C. Antfolk, M. Controzzi, G. Lundborg, B. Rosen, M. C. Carrozza, and F. Sebelius, "Online myoelectric control of a dexterous hand prosthesis by transradial amputees," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 19, no. 3, pp. 260–270, 2011.

[19] S. Micera, J. Carpaneto, and S. Raspopovic, "Control of hand prostheses using peripheral information," *IEEE Reviews in Biomedical Engineering*, vol. 3, pp. 48–68, 2010.

[20] J. Fu, L. Xiong, X. Song, Z. Yan, and Y. Xie, "Identification of finger movements from forearm surface emg using an augmented probabilistic neural network," in *2017 IEEE/SICE International Symposium on System Integration (SII)*, 2017, pp. 547–552.

[21] N. Hogan and R. W. Mann, "Myoelectric signal processing: Optimal estimation applied to electromyography-part i: Derivation of the optimal myoprocessor," *IEEE Transactions on Biomedical Engineering*, no. 7, pp. 382–395, 1980.

[22] K. Englehart, B. Hudgins, P. A. Parker, and M. Stevenson, "Classification of the myoelectric signal using time-frequency based representations," *Medical engineering & physics*, vol. 21, no. 6-7, pp. 431–438, 1999.

[23] K. Veer and T. Sharma, "Extraction and analysis of above elbow semg for pattern classification," *Journal of Medical Engineering & Technology*, vol. 40, no. 4, pp. 149–154, 2016, pMID: 27004618. [Online]. Available: https://doi.org/10.3109/03091902.2016.1153739

[24] Z. Ju, G. Ouyang, M. Wilamowska-Korsak, and H. Liu, "Surface emg based hand manipulation identification via nonlinear feature extraction and classification," *IEEE Sensors Journal*, vol. 13, no. 9, pp. 3302–3311, 2013.

[25] A. Phinyomark, S. Hirunviriya, C. Limsakul, and P. Phukpattaranont, "Evaluation of emg feature extraction for hand movement recognition based on euclidean distance and standard deviation," vol. 2010, 05 2010, pp. 856 – 860.

[26] D. M. Gude, "Automated hand-forearm ergometer data acquisition and analysis system," Ph.D. dissertation, Kansas State University, 2013.

[27] R. O. Duda, *Pattern classification*, 2nd ed., P. E. Hart and D. G. Stork, Eds. New York [etc: John Wiley and Sons, 2001.

[28] M. Ruiz Arahal and M. Vargas Villanueva, *Sistemas de Percepción, Tema 4*. Universidad de Sevilla.

[29] D. S. S. Corporation, "Introducción a Solidworks," *Solidworks*, 2015. [Online]. Available: https://my.solidworks.com/solidworks/guide/SOLIDWORKS{_}Introduction{_}ES.pdf

[30] "Proveedor de servomotores." [Online]. Available: https://opencircuit.shop/Product/TowerPro-SG90-9G-micro-servo-motor-180

[31] C. Cipriani, C. Antfolk, M. Controzzi, G. Lundborg, B. Rosen, M. C. Carrozza, and F. Sebelius, "Online myoelectric control of a dexterous hand prosthesis by transradial amputees," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 19, no. 3, pp. 260–270, 2011.

[32] I. Llop-Harillo and A. Pérez-González, "System for the experimental evaluation of anthropomorphic hands. Application to a new 3D-printed prosthetic hand prototype," *International Biomechanics*, vol. 4, no. 2, pp. 50–59, 2017. [Online]. Available: http://doi.org/10.1080/23335432.2017.1364666