Trabajo Fin de Máster

Máster en Ingeniería Electrónica, Robótica y Automática

Diseño de un sistema electrónico para recolección de datos de golpes en pádel

Autor: Mario Cruz Luque

Tutor: Daniel Gutiérrez Reina

Dpto. Ingeniería Electrónica
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2023

Trabajo Fin de Máster
Máster en Ingeniería Electrónica, Robótica y Automática

Diseño de un sistema electrónico para recolección de datos de golpes en pádel

Autor:

Mario Cruz Luque

Tutor:

Daniel Gutiérrez Reina

Profesor titular

Departamento de Ingeniería Electrónica

Escuela Técnica Superior de Ingeniería

Universidad de Sevilla

Sevilla, 2023

Proyecto Fin de Carrera: Diseño de un sistema electrónico para recolección de datos de golpes en pádel

Autor: Mario Cruz Luque

Tutor: Daniel Gutiérrez Reina

El tribunal nombrado para juzgar el Proyecto arriba indicado, compuesto por los siguientes miembros:

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2023

El Secretario del Tribunal

A mi familia

A mis maestros

Agradecimientos

A mi director de proyecto, Daniel Gutiérrez, por ayudarme en todo lo posible, por su plena implicación, por sacar tiempo de su día a día para guiarme en la realización del proyecto y por su inmejorable trato.

Al Grupo de Ingeniería Electrónica de la ETSI, en especial a mi compañero y amigo Jorge Jiménez, por ser un gran apoyo, dedicándome tiempo, teniendo paciencia y siendo un pilar fundamental a nivel anímico para el presente trabajo.

A la Universidad de Sevilla, por permitirme continuar mi formación de estudiante aceptándome en el máster y concediéndome la oportunidad a través de la cual se ha podido llevar a cabo este proyecto.

Y a mi familia y amigos, por acompañarme en este camino duro, pero a la vez bonito, y por ser un elemento clave y necesario para convertirme en la persona que soy a día de hoy.

Mario Cruz Luque

Sevilla, 2023

Resumen

El sector del deporte debe evolucionar y adaptarse a los avances tecnológicos actuales. Esto implica la integración de herramientas y dispositivos innovadores que permitan mejorar la precisión, la seguridad y la experiencia de los atletas y los espectadores. Para ello, es necesario estar al día con los avances para poder seguir siendo competitivo y atractivo en el mundo moderno, siendo el tamaño, el consumo y la falta de cobertura las principales dificultades para su implementación.

El presente Trabajo Fin de Máster es un proyecto de ejecución de un dispositivo de bajo consumo capaz de proporcionar datos de los golpes dados en el pádel a través de la tecnología de comunicación Bluetooth, con el fin de poder clasificarlos y conocer estadísticas sobre un jugador en cuestión o sobre un partido.

La estructura de este trabajo consistirá en una memoria con el objeto y alcance del proyecto, una introducción acompañada de algunos antecedentes, la fundamentación teórica útil para entender la solución propuesta, los requisitos de diseño e implementación tanto a nivel de hardware como de software, los resultados y conclusiones obtenidas y las propuestas de futuro, así como los anexos necesarios para el adecuado desarrollo de este.

Abstract

The sports industry must evolve and adapt to current technological advancements. This involves the integration of innovative tools and devices that allow for improved accuracy, safety, and the overall experience for both athletes and spectators. To do so, it is necessary to stay up-to-date with these advancements in order to remain competitive and attractive in the modern world. However, the main challenges for implementing these advancements include size, power consumption, and lack of coverage.

This Master's Thesis project aims to develop a low-power device capable of providing data on paddle hits through Bluetooth communication technology. The device is intended to classify these hits and provide statistics on a particular player or match.

The structure of this project will consist of an introduction that includes the project's objectives and scope with some background information, the theoretical framework necessary to understand the proposed solution, the design requirements and the implementation for both hardware and software, the results and conclusions obtained, and proposals for future developments, as well as the necessary appendices for proper development of the project.

Índice

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

1 INTRODUCCIÓN

El fracaso es simplemente la oportunidad para comenzar de nuevo, en esta ocasión con más inteligencia.

- Henry Ford -

En esta parte inicial del documento, primero se presentarán los diferentes objetivos marcados de necesario cumplimiento para que el proyecto tenga éxito, acto seguido se analizará el estado actual de la tecnología dentro del ámbito del deporte en términos generales, luego se presentan diversos trabajos realizados anteriormente los cuales tienen relación con este proyecto, y por último, tras exponer el origen de la motivación para comenzar el estudio, se comentará la estructura que seguirá el documento a grandes rasgos.

1.1. Objeto y alcance del Proyecto

El presente proyecto, realizado en la Escuela Técnica Superior de Ingeniería de Sevilla [1], busca desarrollar e implementar un dispositivo dedicado al pádel a modo de wearable que recoja diferentes características del golpe dado y las guarde para un posterior análisis. Para ello se hará una ruta entre un dispositivo que funcione como cliente, el cual será el encargado de captar y mandar la información; y otro que trabaje de servidor, que guardará dichos datos en el formato deseado. Por tanto, se seleccionarán una serie de dispositivos previamente estudiados, los cuales conformarán el reloj en cuestión, como son el sensor, el microcontrolador y la alimentación. Esto permitirá implementar una solución de bajo coste, poco consumo y con una larga vida útil que beneficie al sector del deporte.

Para cumplir estas especificaciones, se plantea un objetivo general:

- Diseñar e implementar un dispositivo *wearable* que recoja información de los golpes dados en el pádel mediante equipos de bajo consumo y los envíe a través de bluetooth para su posterior clasificación.

Y varios objetivos específicos:

- Evaluar y seleccionar los diferentes instrumentos a utilizar tras su estudio.
- Adquirir conocimiento sobre las características buscadas en el diseño de un *wearable*.
- Estudiar el comportamiento de dos dispositivos al configurarse como cliente y servidor.
- Aprender sobre las prestaciones y limitaciones de una comunicación bluetooth dedicada al envío, recepción y almacenamiento de datos.

1.2. Evolución de la tecnología en el deporte

El avance de la tecnología en la última década ha permitido que, a día de hoy, la mayor parte de los dispositivos estén conectados, tales como electrodomésticos, wearables, muebles o vehículos, entre otros muchos. Tanto es esto cierto que hace algunos años se confirmó que el número de instrumentos interconectados ya sea por Internet, radiofrecuencia o bluetooth, habían superado a la población mundial [2]. Además, transcurre una época en la cual se trabaja frecuentemente con la recopilación masiva de datos que recoja cualquier dispositivo para, posteriormente, poder tratar dicha información, analizarla y darle un uso específico.

Esto es posible gracias a conceptos como son el Internet de las Cosas o el Bluetooth [3], cuyos objetivos se centran en crear amplias redes de sensores y dispositivos conectados entre sí a los que se pueda acceder desde cualquier punto con internet o mediante conexión en área local, con el fin de modificar sus propiedades o extraer sus datos almacenados [4]. Anteriormente se vio que son de diversas categorías los instrumentos que se pueden encontrar estos días formando estas redes, lo que permite implementar estas tecnologías en diferentes campos como, por ejemplo, en viviendas y ciudades inteligentes, vehículos, industrias, en el sector sanitario, agrícola y deportivo, en el cual está centrado este trabajo [5]. En la Figura 1 puede observarse las aplicaciones previamente mencionadas.

Figura 1. Red de dispositivos interconectados [6]

El deporte lleva en auge ya varios años, cada vez existen más tipos, categorías, modalidades y competiciones, es decir, sigue evolucionando con el paso del tiempo. A su vez, las innovaciones en el campo tecnológico han sabido adaptarse rápido a este movimiento ofreciendo materiales e indumentarias que mejoran el rendimiento del usuario, instrumentos capaces de recoger estadísticas que pueden resultar muy útiles para entrenadores y equipo técnico e interesantes para el espectador, dispositivos que ayudan en la prevención de lesiones o en rápidas recuperaciones, o incluso programas que ayudan al cuerpo arbitral en comunicaciones o decisiones complicadas [7].

Dichos dispositivos son en su mayoría sensores, es decir, instrumentos que últimamente han experimentado una gran evolución tecnológica y que permiten recoger información de diversos tipos con el fin de analizarla y mejorar. Esto trae grandes ventajas al sector, como, por ejemplo, un cuidado al detalle de la salud y seguridad del deportista, una toma de decisiones más justa y precisa, una mejora general del rendimiento y un almacenamiento de datos útiles. A su vez, aunque sean de menor impacto también existen ciertos inconvenientes, como puede ser una prolongación de los tiempos de espera en algunas situaciones, o un intento de hacer trampas por parte de los atletas aprovechándose de ventajas brindadas por estas tecnologías, lo que se conoce como dopaje tecnológico.

Ejemplos reales más concretos de la tecnología deportiva son: la termografía, que da información sobre la temperatura corporal de los atletas; las aplicaciones deportivas, que suelen ir acompañadas de pulseras capaces de monitorizar y evaluar el entrenamiento; los softwares en sistemas de vídeo, que permiten detectar cosas que el propio ojo no ve; y la nanotecnología, la cual hace del deporte algo más cómodo y ligero.

Por último, es necesario introducir el deporte para el cual va destinado este proyecto, el pádel, deporte el cual hoy en día se encuentra en su mejor momento. La tecnología cobra en este un papel fundamental, pues se buscan los mejores materiales para que la pala sea óptima y los datos más precisos sobre el ritmo cardiaco, las pisadas o los golpes ganadores de los jugadores durante los partidos. Esto puede conseguirse a través de sensores y chips implementados en wearables, zapatillas e incluso en la propia pala [8].

1.3. Antecedentes

Son varios los proyectos, artículos y noticias que se han publicado y que tratan sobre las implementaciones tecnológicas parar la mejora de este deporte. En 2021, Guillermo Cartes Domínguez, realizó el proyecto fin de máster a partir del cual nacería este trabajo. El título es "Comparación de algoritmos de aprendizaje automático para clasificación de golpes de pádel" y su objetivo consiste en crear una red neuronal capaz de clasificar los golpes de pádel a través de datos recogidos con una unidad de medida inercial (IMU). En el presente trabajo se buscará optimizar la toma de estos datos a través de dispositivos de menor tamaño [9].

Más anteriormente, en 2015, fue publicado otro trabajo fin de máster con título "Dispositivo de captura de movimiento basado en sensores inerciales con comunicación inalámbrica". En este, Miguel Ángel Aparicio Pérez propone un prototipo capaz de capturar datos de movimiento y enviarlos de modo inalámbrico gracias a la tecnología Bluetooth 4.0, para que posteriormente sean analizados y estudiados [10].

También es interesante hablar sobre la empresa española Kaitt, la cual ha lanzado una pala con carga de batería al mercado con multitud de características accesibles mediante bluetooth. Entre ellas encontramos el recuento de golpes totales y por partido, la velocidad de los impactos, el número de metros recorridos y las calorías gastadas por sesión o la temperatura interna de la pala, seleccionando previamente si se está jugando a nivel usuario o profesional.

Figura 2. Pala de pádel inteligente de la empresa Kaitt [11]

Por último, en 2020, Clara Menduiña Fernández realizó un proyecto fin de máster con título "Desarrollo de un sistema de monitorización de estrategias de pádel en tiempo real usando unidades de medición inerciales y algoritmos de Deep Learning". El objetivo de este es desarrollar un sistema que, gracias a una unidad de medición inercial, una arquitectura MQTT y un algoritmo, sea capaz de deducir la estrategia de juego, ya sea ofensiva o defensiva, de cada jugador o pareja en un partido completo [12].

1.4. Motivación del estudio

Como bien se ha comentado anteriormente, el presente proyecto es una continuación de un trabajo anterior [9], el cual se centraba en poder clasificar golpes de pádel gracias al uso de algunos de los principales algoritmos de aprendizaje como pueden ser redes neuronales, árbol de decisión o K vecinos más próximos. Para ello, se necesitó un dispositivo electrónico capaz de recoger y almacenar datos de velocidad y aceleración, es decir, que contara con una unidad de medida inercial (IMU). En la Figura 3 puede observarse que finalmente se optó por una Raspberry Pi 4 junto a la IMU LSM9DS1 incorporada en el Sense Hat, todo anclado sobre una plataforma impresa en 3D y alimentado mediante una batería portátil anclada al pantalón.

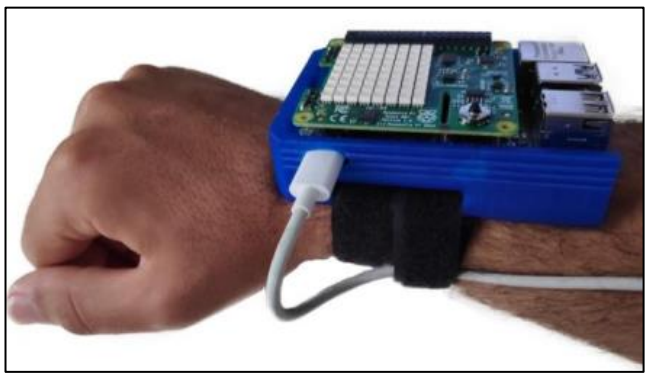

Figura 3. Raspberry Pi + Sense Hat + Plataforma

Es a partir de este dispositivo cuando nace la motivación que da lugar a esta investigación, pues, aunque es cierto que el pádel es un deporte de raqueta en el que la muñeca no suele tener mucho movimiento y que la mayoría de las pruebas pudieron completarse sin mucha dificultad, sería interesante poder optimizar el tamaño del conjunto y hacer que se pareciera aún más a un reloj. Es por ello por lo que a lo largo del presente proyecto se tienen muy presentes tanto las dimensiones de los diferentes instrumentos, como su peso y comodidad.

Respecto a la parte más enfocada al software de la recogida y almacenamiento de datos, la solución adoptada consistía en ir tomando datos durante un periodo de tiempo de tal forma que, cuando los valores recogidos por el acelerómetro y el giroscopio de la IMU sobrepasaran un umbral, se guardaran una cierta cantidad de datos registrados antes y después de ese momento. De esta forma al final se contaba con que un golpe estaba compuesto por diferentes muestras de la IMU, siendo la detección la muestra central.

Es por esto por lo que, el objetivo a perseguir en la parte de software de esta investigación será desarrollar varias opciones para poder almacenar la información de cada golpe manteniendo en todo momento el formato propuesto para que, una vez procesados los datos, puedan ser introducidos de forma directa en la red neuronal para su posterior análisis.

1.5. Estructura del documento

La estructura que se plantea para este escrito pretende realizar un recorrido desde el estudio y la elección de los diferentes dispositivos hasta la simulación de una situación de envío y recogida de datos, junto con su almacenamiento para un posterior análisis. El texto se dividirá de la siguiente forma:

1. En este primer apartado se han establecido las bases del proyecto, presentando los objetivos, el alcance y la relevancia de la investigación. Además, se ha contextualizado el estudio al proporcionar una visión histórica de la evolución tecnológica del deporte y revisar los antecedentes existentes. Finalmente, se ha explicado la motivación detrás del estudio y se presenta la estructura general del documento.
2. En el segundo apartado se exploran aspectos clave de la tecnología aplicada al deporte. Se abordan conceptos como el IoT, el Deporte 4.0 y los sensores. Además, se examina la tecnología Bluetooth en sus versiones 4.0 y 5.0, y se destacan las ventajas de los dispositivos de bajo consumo energético. Esta parte proporciona la base teórica necesaria para comprender cómo se aplican estas tecnologías en el contexto deportivo.
3. El tercer apartado se divide en dos partes principales: una parte de hardware y una parte de software. En la parte de hardware, se aborda la selección y los requisitos del microcontrolador, la IMU y la alimentación. En la parte de software, se explora la topología de red utilizada en el proyecto. Estos requisitos de diseño son fundamentales para el desarrollo y la implementación exitosa del sistema propuesto.
4. El cuarto apartado también se divide en dos partes al igual que el anterior. En la parte de hardware, se describe la implementación de los dispositivos gracias a una PCB y a una carcasa. En la parte de software, se aborda la implementación del software para la recogida

de datos y el envío y almacenamiento de estos. Estas implementaciones son cruciales para asegurar el correcto funcionamiento y la utilización efectiva del sistema desarrollado.

5. Los siguientes apartados se centran en los resultados del estudio, las conclusiones obtenidas a partir de estos resultados y las líneas futuras de investigación que podrían derivarse del proyecto. Estos capítulos son fundamentales para sintetizar y comunicar los hallazgos del estudio, así como para proporcionar orientación sobre cómo avanzar en el campo de estudio en el futuro.
6. Por último, se incluyen las referencias bibliográficas, donde se enumeran todas las fuentes consultadas y citadas a lo largo del trabajo, y los anexos, que contienen material adicional y relevante que respalda el estudio realizado, proporcionando así una base sólida para la verificación y comprensión del proyecto.

2 FUNDAMENTACIÓN TEÓRICA

De vez en cuando vale la pena salirse del camino, sumergirse en un bosque. Encontrarás cosas que nunca habías visto.

- Alexander Graham Bell -

Como se ha explicado anteriormente, en este apartado de fundamentación teórica se proporciona un breve resumen de los diferentes conceptos necesarios para que cualquier lector pueda entender los desarrollos propuestos, yendo desde lo más general hacia lo más específico. Primero se dará una pequeña introducción al llamado *Internet of Things* (IoT) comentando su historia, bases y diferentes aplicaciones en el día a día. Acto seguido se comentará el estado actual en el que se encuentra la tecnología aplicada al ámbito del deporte y lo que se espera en un futuro. Después se explicará cómo funcionan y qué fin tienen los sensores, los cuales son los protagonistas a la hora de captar cualquier tipo de información específica y de transmitirla ya sea de forma inalámbrica o física. Por último, se profundizará sobre la tecnología Bluetooth hablando de sus diferentes versiones y su funcionamiento, y sobre los diferentes dispositivos de bajo consumo, junto con sus ventajas e inconvenientes.

2.1. Internet of Things

Para hablar de los inicios de "Internet of the Things" se debe retroceder al año 1990, cuando, tras muchos estudios y análisis se diseña el que es considerado el primer objeto conectado a Internet [2]. Este fue una tostadora diseñada por John Romkey cuya conectividad funcionaba a través de TCP/IP y a la cual se podía acceder desde ordenadores conectados a la web para controlar su encendido, apagado y tiempo de funcionamiento. A partir de este suceso tuvieron lugar diferentes avances hasta que en 2008 ya se apreciaba un mundo en el que los dispositivos conectados a internet habían superado al número de personas conectadas [2]. Además, en este año se crea la IPSO Alliance, un grupo de empresas cuyo objetivo era crear redes de dispositivos inteligentes.

Así pues, tras todas estas mejoras, en 2009 aparecería de la mano de Kevin Ashton el término “Internet de las cosas”, más conocido como IoT. Este concepto se define como la interconexión de numerosos dispositivos en una red, que puede ser pública o de uso privado, los cuales pueden interaccionar entre sí sin necesidad de manipulación humana [4]. Lo peculiaridad de IoT es que los objetos o dispositivos que sean conectados pueden ser cualquier cosa que se pueda imaginar, como sensores, electrodomésticos, incluso camas o estanterías, lo que permite mucha flexibilidad a la hora de realizar proyectos.

Entre todos los campos en los que se encuentra implementada y en desarrollo esta tecnología destacan la domótica, mediante la cual es posible automatizar la mayoría de dispositivos de una

vivienda y así mejorar las condiciones de la vida cotidiana; la salud, sector en el que se pueden controlar algunas constantes de pacientes mediante sensores incluso cuando no estén en el hospital; la gestión de tráfico y flotas, que permite mediante la geolocalización y el uso de diversa cartografía controlar y monitorizar las vías de circulación; y el deporte, sector en el cual se centra el presente trabajo [5].

Figura 4. IoT en el sector de la salud [13]

2.2. Deporte 4.0

El deporte ha evolucionado a lo largo de la historia, adaptándose a los avances tecnológicos y demandas sociales de tal forma que hoy en día se habla del deporte 4.0, una nueva etapa en la que este se puede disfrutar, practicar y gestionar de una manera diferente gracias a la tecnología. Este cambio se ha conseguido gracias a la incorporación de diversos dispositivos inteligentes, la conectividad en tiempo real y el análisis de datos durante o después del ejercicio. Así pues, se crea poco a poco un entorno en el que los deportistas entrenan, compiten y se relacionan con los aficionados de formas distintas a las convencionales.

Entre las diferentes tecnologías que han sido clave para este movimiento, destacan las siguientes:

- Dispositivos portátiles y *wearables*: estos permiten monitorizar en tiempo real la actividad física, el rendimiento y la salud de los deportistas con el fin de recopilar información para optimizar los entrenamientos, prevenir lesiones y mejorar el rendimiento.
- Internet de las cosas (IoT): como ya se comentó anteriormente, la tecnología IoT también ha llegado al deporte, permitiendo la recolección masiva de información en tiempo real gracias a una amplia red de sensores y dispositivos interconectados. Estos se pueden encontrar tanto en estadios como en equipaciones o pelotas, proporcionando datos detallados sobre el Partido en cuestión, el estado y posición de los jugadores y otros aspectos relevantes.

- Realidad virtual (RV) y realidad aumentada (RA): estas tecnologías están mayormente dedicadas a los aficionados. Mientras que con la realidad virtual se puede disfrutar de eventos deportivos como si se estuviera presente en el estadio a través de uso de unas gafas especiales, con la realidad aumentada se puede visualizar información gráfica superpuesta al partido en tiempo real.

Figura 5. Realidad virtual y aumentada combinadas [14]

A partir de estas tecnologías el deporte 4.0 ofrece una serie de ventajas como la posibilidad de crear entrenamientos personalizados, los cuales están adaptados a las necesidades de cada deportista a partir de datos recogidos. También ofrece la oportunidad de prevenir lesiones a través de la monitorización de indicadores de salud o minimizarlas detectando con antelación posibles problemas. A los aficionados les ofrece de forma directa mayor interacción y participación gracias a las redes sociales y de forma indirecta permitiendo gestionar eventos deportivos de manera eficiente, desde la seguridad hasta la distribución de las entradas.

Así mismo, también existen una serie de inconvenientes como puede ser la completa dependencia de la tecnología, ya que si su funcionamiento no es correcto puede fallar todo el sistema. También pueden ser un problema la preocupación por la privacidad y seguridad de los datos o la brecha digital existente, la cual dificulta la accesibilidad en algunos lugares a estas tecnologías. Por último, no se puede obviar ni el costo que supone toda esta modernización, el cual obliga a disponer de unos recursos financieros, ni la saturación que pueden provocar la multitud de datos recogidos en un solo evento.

Por último, el pádel también ha experimentado este gran cambio, recogiendo información como la velocidad de la pelota, el número de golpes en un intervalo de tiempo o la distancia recorrida por cada uno de los jugadores. Estos datos son usados para crear o ajustar estrategias de juego o para aumentar el rendimiento de los deportistas, siendo analizados previamente. De la misma manera las diferentes tecnologías han facilitado la gestión de torneos y competiciones, automatizando el proceso de registro, programación de partidos y seguimiento de resultados en tiempo real a partir de

aplicaciones que están al alcance de la mayoría.

2.3. Wearables

LOs dispositivos *wearables*, conocidos también por el nombre de dispositivos vestibles o ponibles, son instrumentos electrónicos que pueden llevarse puestos como puestos como prendas de vestir o accesorios. Estos han experimentado un crecimiento significativo en popularidad en los últimos años gracias a la capacidad que poseen para mejorar y monitorizar la vida diaria de las personas. Dichos dispositivos pueden ser desde relojes inteligentes o pulseras de actividad hasta gafas de realidad aumentada o dispositivos de seguimiento de salud, y con su aparición han revolucionado la forma en la que la humanidad interactúa con la tecnología y se mantiene conectada al mundo.

Los comienzos de los dispositivos *wearables* se remontan a principios del siglo XXI, momento en el cual empezaban a surgir los intentos de fusionar la tecnología con la moda. Uno de los primeros dispositivos fue el reloj inteligente SPOT (*Smart Personal Objects Technology*) lanzado por Microsoft en el año 2004, el cual, aunque no tuvo un gran éxito, pudo definir las bases para una futura tecnología vestible. Así pues, en el año 2013 Samsung sacaría el Galaxy Gear, uno de los primeros relojes inteligentes con gran popularidad.

Figura 6. A la izquierda Microsoft SPOT [32], a la derecha Galaxy Gear [33]

Hoy en día, los dispositivos *wearables* han evolucionado bastante y han pasado a ser una parte integral de la vida moderna. Esto ha sido gracias a relojes inteligentes como el Apple Watch y otros dispositivos con sistema operativo Android, los cuales han conseguido tener una gran aceptación por parte de la sociedad debido a su capacidad para mostrar notificaciones, realizar un seguimiento de la actividad física y permitir monitorizar la salud. Además, las pulseras de actividad, como las de Fitbit o Garmin, son también populares ya que fomentan un estilo de vida más saludable.

Entre las características clave que debe poseer un dispositivo *wearable*, destaca la capacidad

para recopilar y analizar datos. Estos dispositivos están equipados con gran cantidad de sensores como acelerómetros, giroscopios o sensores de frecuencia cardiaca, los cuales recopilan información sobre actividades diarias, rendimiento físico o patrones de sueño. Dichos datos son accesibles a través de aplicaciones móviles o plataformas en la nube, permitiendo un mayor control al usuario.

Una de las ventajas del uso de estos dispositivos es el acceso rápido y conveniente a la información, como mensajes, llamadas telefónicas o actualizaciones en redes sociales sin tener que sacar el móvil del bolsillo. También, mediante la monitorización puede ayudar a las personas a establecer metas y mantenerlas motivadas a llevar una vida más activa y saludable. Por último, permiten a los profesionales del campo de la salud tener acceso a información útil para el seguimiento de enfermedades.

Aun así, también existen inconvenientes, como la dependencia de estos dispositivos, que puede llevar a una distracción constante o incluso a generar algo de ansiedad en el usuario. Además, existe el riesgo de que los datos almacenados puedan ser robados o utilizados de manera indebida si no se toman las medidas adecuadas para protegerlos, por lo que la privacidad de estos es una preocupación notable.

Por último, el uso de dispositivos *wearables* puede verse en diversos campos y sectores de la vida moderna, como por ejemplo en el deporte y la actividad física, en la salud y el bienestar, en la industria de la moda y el estilo personal, en la realidad virtual y aumentada, en la industria empresarial, en la industria de la logística y el transporte, en la educación y en la industria del entretenimiento y los videojuegos.

2.4. Unidades de medición inercial (IMU)

Las unidades de medición inercial, conocidas por las siglas IMU, son dispositivos electrónicos utilizados para medir la orientación, la velocidad angular y la aceleración de un objeto en el espacio tridimensional. Estos constan de tres componentes principales: un giroscopio, que mide la velocidad angular y el cambio de orientación en los ejes X, Y y Z; un acelerómetro, que recoge la aceleración lineal; y un magnetómetro, el cual detecta el campo magnético terrestre para determinar la orientación absoluta en relación con el norte magnético.

Para el funcionamiento del acelerómetro utiliza la fuerza ejercida sobre una masa interna con el fin de medir la aceleración lineal. Dicha fuerza se convierte en una señal eléctrica proporcional a la aceleración. El giroscopio en cambio detecta la rotación midiendo la deflexión causada por la fuerza de Coriolis generada por la rotación del objeto. Por otra parte, el magnetómetro utiliza sensores de efecto Hall o tecnología similar para medir el campo magnético y determinar la orientación. Es por esto que, como cada uno de ellos cuenta con unos ejes con 3 grados de Libertad, las IMU del mercado se suelen clasificar por cuántos de ellos cuentan, que generalmente son 3, 6 o 9.

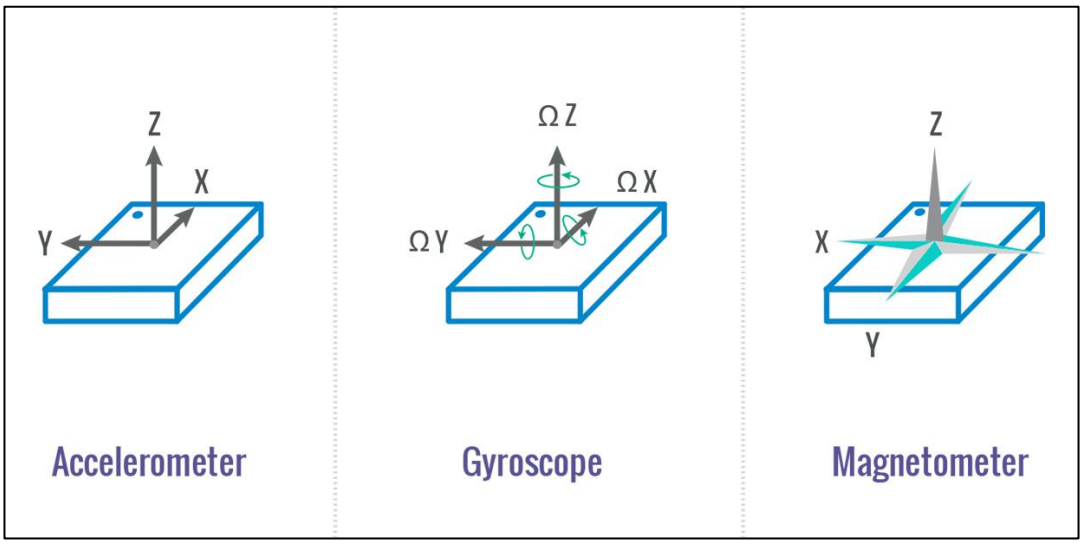

Figura 7. Diferentes ejes de los componentes principales en una IMU [15]

Al utilizar este tipo de sensores hay que tener en cuenta la calibración de los mismos para reducir errores, al igual que tras usarlos es recomendable realizar un filtrado de los datos recogidos para eliminar ruido y vibraciones indeseadas. Además, a la hora de escoger una para trabajar, se deberá decidir si se busca una que se conecte de forma inalámbrica o de forma física mediante cableado, como, por ejemplo, por I2C. La primera usa tecnologías como Bluetooth, Wi-fi o radiofrecuencia para trasmitir la información a un receptor, por lo que es muy útil en trabajos que requieran una mayor movilidad, como es el caso de drones. En cambio, la segunda opción ofrece mayor fiabilidad en la transmisión de datos, por lo que destacan en aplicaciones donde la precisión y la latencia reducida son importantes, como la robótica industrial.

Entre las ventajas que presentan las unidades de medición inercial destacan su pequeño tamaño y su bajo consumo de energía, por lo que son adecuadas para aplicaciones portátiles; su medición en tiempo real que les permite dar respuesta rápidas y precisas; su bajo costo, que se traduce en una mayor accesibilidad para quien precise de ellas; su uso en gran variedad de aplicaciones; y su funcionamiento autónomo, siendo muy útiles en lugares donde la señal es débil o nula, ya que no depende de ninguna señal externa como sí lo hace por ejemplo el GPS.

Por contra, también existen inconvenientes en el uso de estas como puede ser el error acumulativo dado por errores sistemáticos, los cuales suelen requerir una recalibración periódica. Además, son unos dispositivos sensibles al ruido y las vibraciones ambientales, como por ejemplo los campos magnéticos externos o los obstáculos metálicos, y suelen tener un rango dinámico limitado, por lo que existen dificultades a la hora de medir de manera precisa movimientos muy rápidos o lentos.

Por último, entre los campos en los que se pueden encontrar aplicados estos dispositivos destacan: la robótica, para el control de orientación y estabilización de robots; la navegación; la realidad virtual, permitiendo seguir diferentes movimientos; la salud; y la industria aeroespacial.

2.5. Bluetooth 4.0 & 5.0

Bluetooth es una tecnología inalámbrica utilizada para la comunicación de corto alcance entre dispositivos, la cual opera en la banda (ISM) sin licencia industrial, científica y médica, en el rango de 2.4 a 2.485GHz, utilizando un espectro ensanchado, con salto de frecuencia, señal completa dúplex a una tasa nominal de 1600 saltos/s. Las versiones 4.0 y 5.0 son dos generaciones importantes de Bluetooth que presentan mejoras significativas en términos de velocidad, distancia de dispositivos conectados y eficiencia energética.

Bluetooth 4.0, también conocido como Bluetooth de baja energía (Bluetooth Low Energy), se creó con el objetivo de mejorar la eficiencia energética y proporcionar conectividad para dispositivos de baja potencia. Esto se consiguió asumiendo una velocidad de datos menor en comparación a versiones anteriores a cambio de consumir mucha menos energía. Además, es una versión que es compatible con versiones anteriores pero que tiene menos alcance que las posteriores.

Por otra parte, el Bluetooth 5.0 es una versión más reciente y ofrece mejoras significativas en términos de velocidad, alcance y capacidad de transmisión de datos en comparación con Bluetooth 4.0, aunque puede consumir más energía. Esta versión ofrece velocidades de transferencia de datos hasta cuatro veces más rápidas y un alcance extendido en comparación con sus predecesores. También proporciona una capacidad de transmisión de datos mejorada y mayor capacidad de carga útil, lo que permite la transmisión de datos más grandes y complejos. Bluetooth 5.0 también es compatible con versiones anteriores, lo que significa que puede conectarse con dispositivos Bluetooth más antiguos, aunque es una compatibilidad limitada ya que quizás no puedan soportar las mejoras específicas dedicadas a esta versión.

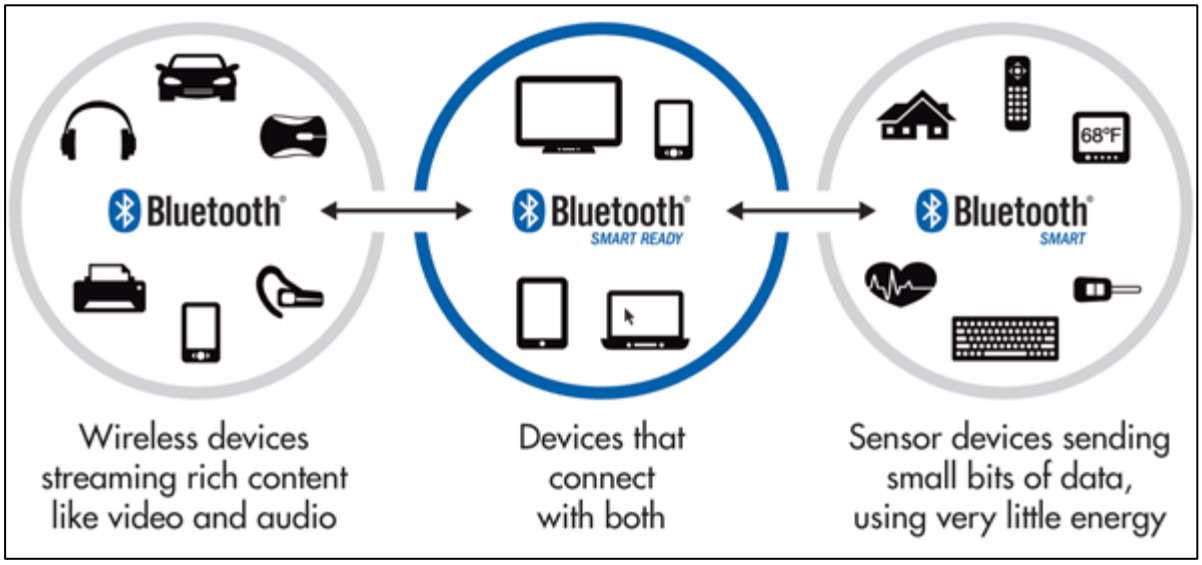

Figura 8. Red de dispositivos interconectados por tecnología Bluetooth [16]

Esta tecnología cuenta con diferentes protocolos que forman las bases para establecer y gestionar conexiones. Estos son un conjunto de reglas y formatos de datos que se utilizan para establecer la comunicación, manejar la transferencia de datos, la autenticación, la seguridad y otros

aspectos relacionados con la transmisión de información. Algunos ejemplos son el protocolo L2CAP (*Logical Link Control and Adaptation Protocol*), el cual proporciona servicios de multiplexación, segmentación y reensamblaje de datos para aplicaciones Bluetooth, y el Protocolo RFCOMM (*Radio Frequency Communications),* que un puerto serie RS-232 sobre Bluetooth y permite la comunicación con aplicaciones que utilizan puertos serie estándar.

Por último, se hablará sobre los perfiles Bluetooth, los cuales definen un conjunto de características y funcionalidades específicas que permiten a los dispositivos Bluetooth comunicarse y realizar tareas particulares entre sí. Los perfiles Bluetooth describen la manera en la que los diferentes dispositivos interactúan y qué tipo de servicios pueden ofrecer o utilizar. Cada uno de estos perfiles está diseñado para satisfacer unas necesidades específicas, como el perfil *Hands-Free Profile* (HFP), se utiliza para establecer una comunicación manos libres entre un teléfono móvil y un dispositivo de manos libres en un automóvil. También existe el perfil GATT (*Generic Attribute Profile*), que define cómo los dispositivos Bluetooth de baja energía intercambian datos utilizando una arquitectura cliente-servidor, por lo que es muy útil cuando se trabaja en entornos de IoT. Para terminar, uno de los más conocidos es el perfil A2DP (*Advanced Audio Distribution Profile*), el cual permite la transmisión de audio estéreo de alta calidad desde un dispositivo fuente a un dispositivo de reproducción de audio.

2.6. Dispositivos electrónicos de bajo consumo

El interés por los dispositivos electrónicos de bajo consumo ha crecido significativamente en las últimas décadas. Con el avance de la tecnología, el uso de componentes cada vez de menor tamaño y la necesidad de dispositivos portátiles, ha surgido la necesidad de diseñar dispositivos que consuman la menor cantidad de energía posible durante su funcionamiento. Este objetivo se puede perseguir de varias formas, como por ejemplo realizando diseños de bajo voltaje, implementando modos de suspensión y ahorro de energía, los cuales reducen el consume cuando el dispositivo no está en uso, o mejorando la eficiencia energética de los componentes individuales como pantallas, sensores o procesadores.

Entre las ventajas que aparecen junto al uso de esta tecnología se encuentran: una mayor duración de la batería, lo cual es clave por ejemplo en el mercado de móviles o relojes inteligentes; un menor costo de operación a largo plazo, ya que se utiliza menos energía para alimentarlos; una mayor sostenibilidad Ambiental, contribuyendo a disminuir la huella de carbono; y una tendencia de los dispositivos a generar menos calor durante su funcionamiento, lo cual puede mejorar su vida útil y evitar métodos de enfriamiento complejos.

Aunque sean escasos, también existe algunos inconvenientes como el rendimiento limitado debido a la limitación de recursos o la complejidad de los diseños, ya que implica equilibrar la eficiencia energética con el rendimiento y la funcionalidad, cosa que en los dispositivos convencionales no es necesario.

Para terminar, las aplicaciones en las cuales se hace un mayor uso de esta tecnología hoy en día son las siguientes:

- Dispositivos portátiles: en teléfonos, relojes inteligentes, auriculares inalámbricos o instrumentos para hacer deporte.
- Internet de las cosas (IoT): en sensores, sistemas de monitorización remota, dispositivos de domótica o cualquier aparato que requiera de una vida útil prolongada.
- Energía renovable: en paneles solares, sistemas de almacenamiento de energía o medidores de parámetros inteligente.
- Electrónica médica: en dispositivos como marcapasos, controladores de glucosa y otros dispositivos de seguimiento de la salud, donde es importante garantizar un funcionamiento fiable y prolongado.

3 REQUISITOS DE DISEÑO

La verdadera felicidad radica en la finalización del trabajo utilizando tu propio cerebro y habilidades.

- Soichiro Honda -

Este apartado se divide en dos partes principales, una de Hardware y otra de Software. En la primera se plasma un análisis realizado de las diferentes alternativas existentes para cada uno de los componentes que conforman el dispositivo, estudiando las ventajas e inconvenientes de cada uno y diversas características a partir de las cuales se ha podido seleccionar uno de ellos. La segunda parte del capítulo se centrará más en la topología escogida para la transmisión y el análisis de los datos recogidos.

3.1 Hardware

En esta sección se verán los candidatos que fueron estudiados para las tres partes físicas que componen el sistema que son: el microcontrolador, para el cual se analizaron varias posibles opciones; la IMU, que se decidió una que fuera compatible a partir del micro y que cumpliera con las necesidades del proyecto; y la alimentación, para la que se tuvo que decidir entre diversas alternativas.

3.1.1 Microcontrolador

Primordialmente se buscaba una placa de desarrollo que incluyera la propia IMU junto al microcontrolador, opción que se descartó ya que en el mercado no había placas lo suficientemente pequeñas para la aplicación del presente proyecto. Es por esto la decisión de separar estos dispositivos, buscando una placa de desarrollo que ofreciera las siguientes características:

- Consumo considerablemente bajo.
- Tamaño reducido para minimizar a ocupación de espacio.
- Comunicación inalámbrica preferentemente o con capacidad para almacenar datos mediante memoria interna o externa.
- Frecuencia de funcionamiento no elevada, ya que para este proyecto no es necesaria y aumentaría el consumo.
- Varias interfaces de comunicación para tener más opciones a la hora de escoger el sensor.
- Número de pines reducidos, debido a que solo habrá que conectar una IMU y la alimentación

externa.

- Compatible con una fuente de energía externa, es decir, una batería.
- Preferiblemente compatible con Arduino IDE por experiencia en la programación en dicho entorno.

Como primera opción se pensó en Arduino LilyPad [17], una plataforma de desarrollo basada en microcontroladores diseñada específicamente para proyectos de electrónica de *wearables*. Estos ofrecen dispositivos que pueden ser cosidos a la misma ropa mediante un hilo conductor, por lo que es una buena posibilidad para una muñequera, aunque precise de un buen mantenimiento y unas posibles reparaciones.

La placa estudiada es la LilyPad Arduino Main Board, la cual está montada sobre el microcontrolador ATmega328P, se alimenta mediante una fuente de energía externa y se programa a través del entorno de Arduino. Esta placa de tamaño compacto cuenta con varios puertos de entrada/salida analógicos y digitales que permiten la conexión con sensores y otros componentes mediante protocolos como I2C, UART y SPI.

Esta placa, cuyo microcontrolador funciona a 16 MHz y tiene 32KB de memoria Flash y 2KB de RAM, se alimenta en un rango de voltaje entre 2.7V y 5.5V y cuenta con 14 pines digitales y 6 analógicos. Sus dimensiones son aproximadamente de 50 milímetros de diámetro y para su programación es necesario utilizar un adaptador USB-TTL externo u otro dispositivo compatible. Además, esta placa cuenta con un LED de encendido y apagado, un botón de reinicio y conectores específicos para comunicación.

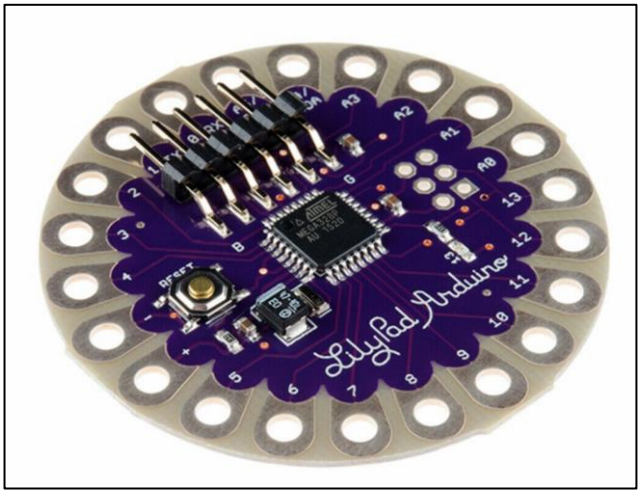

Figura 9. Arduino LilyPad Main Board [17]

Como segunda opción se estudió la placa de desarrollo WMW101 [18], basada en ESP32 y compatible con Arduino. Esta cuenta con tecnología WiFi y Bluetooth y funciona con un consumo muy bajo gracias a su modo de espera *deep sleep*. Además, cuenta con varios sensores ya integrados, un botón de reinicio y algunas interfaces como SPI, I2S, I2C y UART. Sus pequeñas dimensiones

son de 56 x 50 milímetros, junto a un finísimo grosor que no llega a los 8 milímetros. También proporciona 17 pines GPIO, varios LEDs y un conector JST para una batería externa.

Respecto a sus características más técnicas, esta placa funciona con un procesador Xtensa 32 bit LX6 doble núcleo, que funciona a 160 o 240 MHz con una capacidad máxima de 600 DMIPS, y con un coprocesador de muy bajo consumo. Su memoria cuenta con 520 KB de SRAM y con 448 KB de ROM. Respecto a su conexión inalámbrica, su versión de WiFi es la 802.11 b/g/n y de Bluetooth la 4.2 BR/EDR y BLE. Para terminar, la gestión de la energía se consigue gracias a un consumo de corriente de 300 mA máximo (10 uA si se encuentra en modo de espera), a una máxima tensión de entrada de 6V y a una máxima corriente de carga de la batería de 450 mA.

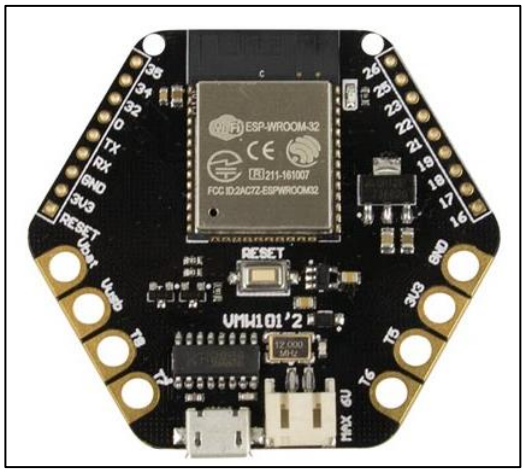

Figura 10. Placa de desarrollo ESP32 modelo WMW101 [18]

Otra opción basada en ESP32 y compatible con Arduino ha sido la placa de desarrollo TinyPICO [19]. Esta al igual que la anterior cuenta con WiFi y Bluetooth y un funcionamiento de bajo consumo. Su nombre viene dado por sus dimensiones, que son de 18 x 32 milímetros, contando con 14 pines GPIO, varios leds, botón de reinicio y un puerto micro USB para su programación, la cual puede hacerse mediante MicroPython, Arduino IDE o Espressif IDF. Además, la placa cuenta en su parte inferior con dos pads dedicados a una posible batería externa

Entre sus características técnicas destacan un procesador 32 bit de dos núcleos operando a 240 MHz, una memoria SPI *flash K*de 4 MB y una memoria extra PSRAM de otros 4MB. Su versión de WiFi es la 802.11 b/g/n y de B1luetooth la 4.2. Por último, el bajo consumo que proporciona este dispositivo es posible gracias a un regulador LDO de 3.3 V de 700 mA y a una ruta de alimentación optimizada para el uso de la batería.

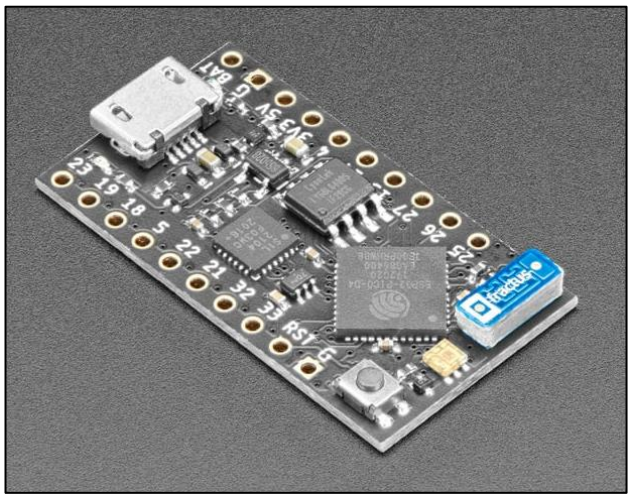

Figura 11. Placa TinyPico v2 [19]

También se estudió la opción de una placa de desarrollo de Adafruit compatible con Arduino llamada GEMMA v2 [20]. Este dispositivo destaca por sus pequeñas dimensiones (28 milímetros de diámetro y 7 milímetros de grosor) y por su bajo costo. Además, cuenta varios LEDs, un botón de reinicio, un interruptor de apagado/encendido, 3 pines de entrada/salida, capacidad de comunicación I2C, y un conector micro-USB.

Dicha placa cuenta con el microprocesador Attiny85 funcionando a 8 MHz, el cual cuenta con una memoria flash de 8K de los cuales se pueden usar 5.25K, ya que los restantes se utilizan para el *bootloader*. Por otra parte, este dispositivo de bajo consumo (9 mA durante el funcionamiento) cuenta con 512 bytes de SRAM y otros 512 de EEPROM. También hay un regulador de voltaje incorporado de 3.3 V con una capacidad de salida de 150 mA y una caída de voltaje extremadamente baja, soportando una entrada de hasta 16V. Esta placa cuenta con un consumo bastante bajo que ronda los 9 mA en funcionamiento.

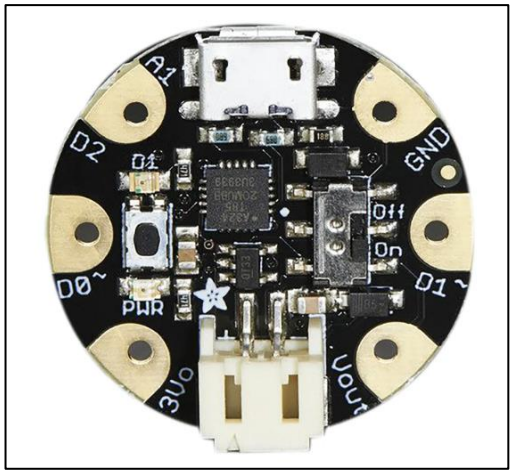

Figura 12. Placa de desarrollo Adafruit GEMMA v2 [20]

Por último, se investigó acerca de la placa Beetle BLE de DFRobot, un dispositivo de tamaño reducido basado en Arduino Uno y con Bluetooth 4.0. Esta placa de solo 10 gramos y de 28.8 x 33.1 milímetros, permite comunicarse hasta una distancia de 50 metros en campo abierto aguantando temperaturas entre -10ºC y +85ºC. Cuenta con 4 pines digitales y 4 analógicos, con 2 salidas PWM y con interfaces I2C, UART y Micro-USB.

El microcontrolador empleado en el dispositivo es el ATmega328, que trabaja a 16 MHz como se vio anteriormente. Respecto al Bluetooth usa el chip CC2540, el cual se caracteriza por tener una sensibilidad de -93dBm, lo que permite tener una buena capacidad de recepción en entornos con señales débiles o distancias más largas. Su consumo es aproximadamente de 34.3 mA en modo IDLE y de 35.4 mA cuando se conecta a otro dispositivo. Además, el dispositivo puede alimentarse hasta con 8 V de máximo, siendo compatible con diversas fuentes de energía.

Figura 13. DFRobot Bluno Beetle BLE [21]

Se recogen las características más significativas acorde con las necesidades que se vieron al inicio de la sección para decidir:

	LilyPad	**WMW101**	**TinyPico**	**Gemma V2**	**Beetle BLE**
Dimensiones (mm)	50 de diámetro	56 x 50 x 7.5	18 x 32	28 diámetro x 7 grosor	28.8 x 33.1
Comunicación Inalámbrica	No	WiFi y Bluetooth	WiFi y Bluetooth	No	Bluetooth
Frecuencia de funcionamiento	16 MHz	160 – 240 MHz	240 MHz	8 MHz	16 MHz
Consumo	15-20 mA	300 mA máx 10 uA espera	80-260 mA 15uA espera	9 mA	34-35 mA

	LilyPad	WMW101	TinyPico	Gemma V2	Beetle BLE
Pines	20	17	14	3	8
Interfaces de comunicación	I2C, UART y SPI	SPI, I2S, I2C Y UART	I2C, UART y SPI	I2C	I2C y UART
Admite batería	Sí	Sí	Sí	Sí	Sí
Compatible con Arduino	Sí	Sí	Sí	Sí	Sí
Precio en €	19.60	28.20	18.67	9.29	13.91

Tabla 1. Comparación entre las opciones para microcontrolador

A partir de la Tabla 1 se ha decidido trabajar en torno al dispositivo Beetle BLE, ya que cumple con todas las especificaciones necesarias para poder llevar a cabo el presente proyecto.

3.1.2 IMU

Para la unidad de medición inercial (IMU) se partió del microcontrolador escogido, es decir, se hizo una búsqueda de los diferentes dispositivos que ofrecía la misma empresa para ahorrar problemas de compatibilidad. En base a esto se buscó un dispositivo de bajo consumo, que fuera de tamaño reducido, con gran precisión en las medidas y que tuviera una comunicación compatible con las interfaces de la placa escogida anteriormente.

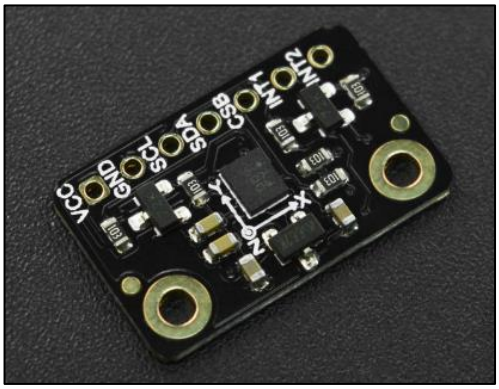

Figura 14. Fermion: BMX160 sensor de 9 ejes [22]

Así pues, como puede observarse en la Figura 16, se decidió usar el sensor de 9 ejes BMX160 [22], el cual está diseñado para integrarse en *wearables* como relojes o gafas. Sus características se recogen en la siguiente tabla:

Tensión de funcionamiento	3.3V – 5.5V
Interfaz de comunicación	I2C
Acelerómetro	±2g / ±4g / ±8g / ±16
Giroscopio	Rango: ±125°/s ~ 2000°/s
Magnetómetro	Rango: ±1150uT(x-,y-axis) ; ±2500uT(z-axis) Resolución: ~0.3
Dimensiones	20 x 12.5 mm
Precio	12,99 €

Tabla 2. Características de la IMU de 9 ejes BMX160 [22]

3.1.3 Alimentación

En la alimentación de la placa se necesitaba una tensión menor a 8V, ya que es lo máximo que aguanta la placa del microcontrolador, pero sin llegar a ser muy baja ya que se podría ver empeorado el funcionamiento del dispositivo. Es por ello, que se empezó buscando una batería recargable rectangular de tamaño reducido la cual debía ser de litio ya que estas baterías son idóneas para proyectos de este tipo porque el voltaje en su descarga es bastante lineal, lo que permitirá un mayor rendimiento que si se usa una batería alcalina en la que la pendiente en su descarga es mayor.

Al no encontrar ninguna batería que fuera lo suficientemente pequeña en tamaño para el presente dispositivo se optó por la opción de buscar un dispositivo que permitiera integrar en serie dos pilas de botón de 3.3 V, dando como resultado una tensión de unos 6.6 V. Investigando se encontró el dispositivo de la Figura 15, el cual podía cumplir las necesidades del sistema añadiendo un interruptor que permite apagar y encender el dispositivo en cualquier momento.

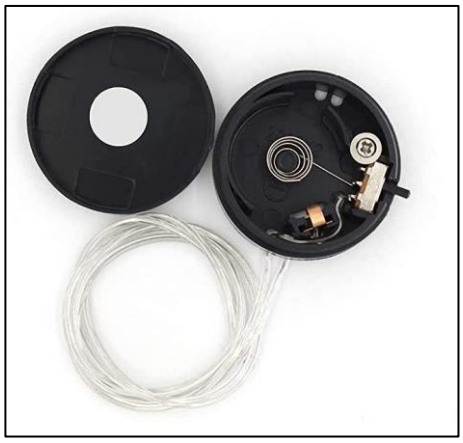

Figura 15. Soporte para pilas de botón en serie RUNCCI-YUN [23]

3.2 Software

Esta sección se dividirá en dos partes, una de ella dedicada a la topología de red, es decir, de qué manera llegan los datos desde que son recogidos hasta la unidad que se encargará de almacenarlos; y una segunda parte dedicada a la forma que tendrán dichos datos almacenados.

3.2.1 Topología de envío de datos

Este apartado se centra en el flujo de información desde la unidad de medición inercial (IMU) hasta el dispositivo de recepción y análisis de los datos, que será un ordenador. Una vez que el sensor recoge los parámetros de sus diferentes ejes los manda al microcontrolador a través de I2C y es aquí cuando existen dos métodos de transmisión hacia el ordenador, los cuales son:

1. Inicialmente, se consideró utilizar la comunicación Bluetooth para transmitir los datos desde el microcontrolador al ordenador de forma directa. Sin embargo, se encontró una limitación en esta opción, ya que el ordenador no era compatible con dispositivos Bluetooth de baja energía (BLE) tal y como se muestra en la Figura 16, donde en la misma página se avisaba de ello. Como solución, se evaluó el uso de un dispositivo adicional conocido como *BLE link*, el cual actúa como un puente entre el microcontrolador y el ordenador, pero se descartó para no añadir otro dispositivo más al sistema y hacerlo más complejo.

Figura 16. Advertencia sobre la comunicación del dispositivo al ordenador por Bluetooth [24]

2. La segunda opción ha sido crear una red configurando uno de los microcontroladores como cliente y el otro como servidor. Así pues, cuando los datos son recogidos por el sensor y transmitidos al cliente a partir de I2C se mandan al servidor a través de la tecnología Bluetooth para, posteriormente, recogerlos con el ordenador haciendo uso del puerto serie para su almacenamiento y análisis.

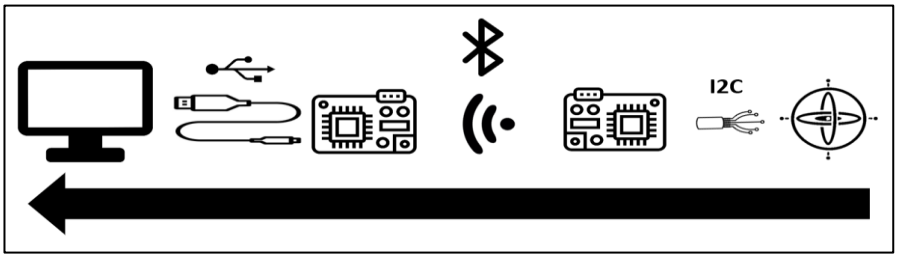

Figura 17 Flujo de la información en red cliente-servidor

3.2.2 Procesamiento de la información

Como se comentó en la motivación de este estudio, el presente proyecto parte de un trabajo anterior [9], en el cual los datos recogidos se almacenan en un '.csv ' con un formato específico. Por lo tanto, lo que se buscará en esta sección será que, tras procesar la información, el resultado tenga un formato compatible para que pueda ser introducido en la red neuronal ya creada. Para ello, se describirá de forma breve y general el proceso que se llevó a cabo en el sistema previo y se planteará una solución para el actual.

En dicho trabajo el autor realizó pruebas de aproximadamente minuto y medio en las que un sujeto devolvía bolas con diferentes golpes. En dichos experimentos, la IMU iba registrando los datos recogidos por el acelerómetro y el giroscopio con una frecuencia de 20 Hz, ya que en [25] se lleva a cabo un estudio donde se observa que una tasa correcta para reconocer actividades humanas se encuentra en torno a dicha frecuencia.

Acto seguido se procedía con la detección de un golpe, para lo cual se definieron ciertos umbrales en cada grado de libertad de forma que si alguno de los ejes del acelerómetro y giroscopio lo superaran en la misma muestra se daba por hecho que el golpe había sido realizado. Además, se estableció una duración de golpe de 2 segundos (40 muestras), por lo que se guardarían todas las muestras tomadas en ese rango de tiempo quedando la detección justo en el centro.

Así pues, el conjunto de datos final tendrá el formato que se muestra en la Figura 18. Este se divide en tres bloques principales, siendo el primero los 40 parámetros recogidos por el sensor en cada uno de los ejes. En el siguiente bloque se encuentra: el tipo de golpe, que vendrá dado por un número entero de acuerdo con un código; el número de golpe; y el tiempo transcurrido desde que inició la prueba hasta el golpe en segundos. En el último bloque se encontrarán algunas características que definen al deportista que realiza la prueba.

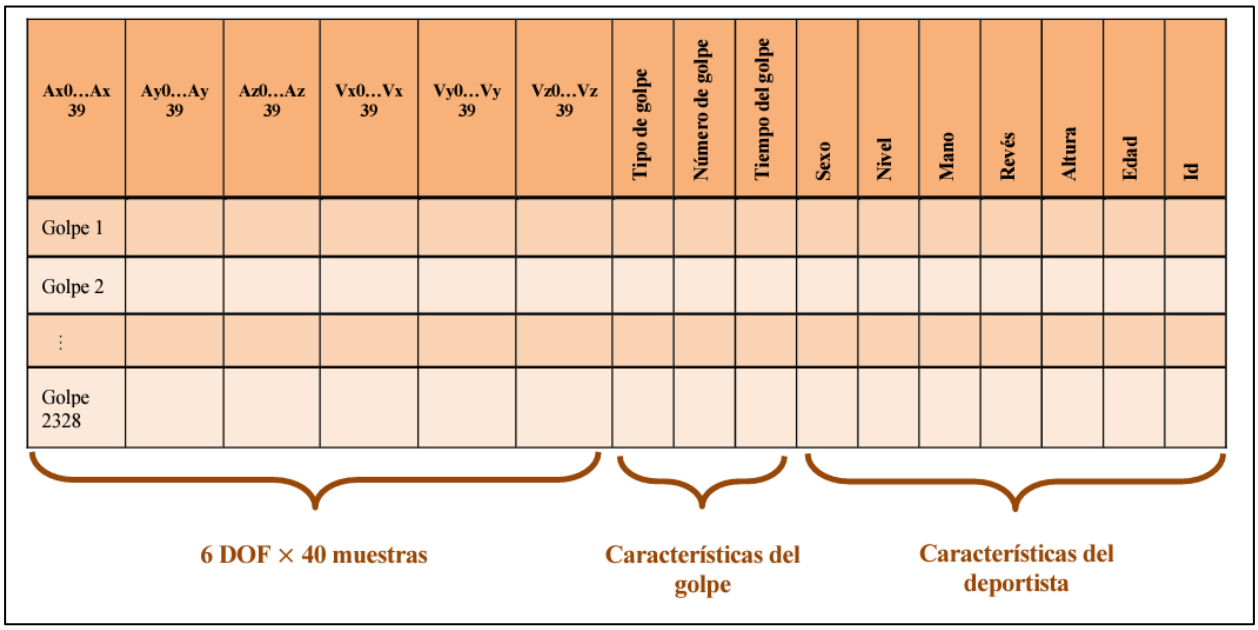

Figura 18. Formato final buscado de los datos recogidos [9]

Para verificar y validar el comportamiento del sistema completo, es decir, para comprobar que no existen fallos, al ser una primera versión se recibirá el *Dataset* completo en el ordenador sometiéndolo a un procesamiento independiente del microcontrolador. Aquí nace una de las líneas futuras que se comentarán posteriormente, que consiste en trasladar el procesamiento y realizarlo en el mismo micro antes de que los datos sean recogidos.

Así pues, la solución software que se ha decidido seguir para el presente proyecto consistirá en enviar todos los datos mediante Bluetooth del cliente al servidor a la vez que se van recibiendo por puerto serie. Dicha información se guardará en un archivo '.csv ' el cual, una vez haya terminado el experimento, será trasladado a Matlab para realizar la detección de golpes con su correspondiente filtrado.

4 IMPLEMENTACIÓN

"La gente normal cree que si no está roto, no lo arregles. Los ingenieros creen que si no está roto es que aún no tiene suficientes características".

- Scott Adams -

Esta sección se desarrollan los métodos seguidos para implementar las soluciones estudiadas en el capítulo anterior y así formar el sistema al completo. Al igual que antes, existe una parte hardware dedicada a la parte más física del proyecto en la que se conectarán los diferentes componentes ya escogidos, y una parte software en la que se analizarán de forma general los diferentes códigos que han sido desarrollados en las distintas plataformas.

4.1 Hardware

La subsección de hardware se dividirá en tres partes, la primera dedicada al proceso de diseño de una PCB que permita agrupar los dispositivos y conseguir un tamaño compacto, la segunda dedicada a la parte externa o carcasa del reloj, y la tercera dedicada a la correa escogida para el ajuste a la muñeca.

4.1.1 PCB

Para la PCB se ha hecho uso del software de diseño Altium Designer [26], en el que para empezar se ha realizado un esquemático bastante simple pero para el que se han debido tener en cuenta diversas consideraciones.

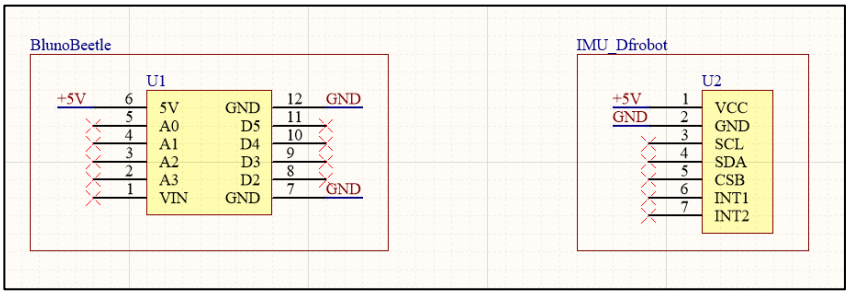

Figura 19. Diseño esquemático de la PCB del sistema

La primera de ellas es que los pines dedicados a la comunicación I2C de la placa del

microcontrolador son pines superficiales, por lo que la orientación de esta debía ser hacia arriba por si hubiera que arreglar cualquier inconveniente en la soldadura. Por lo tanto, las únicas conexiones que están presentes en la placa son las de alimentación al sensor y las de tierra, con un grosor de pista de 1 milímetro. Otra consideración que hay que tener son las dimensiones de los dispositivos, ya que al no encontrar sus modelos en internet se han tenido que crear de forma manual acorde con sus medidas. Además, se ha dejado la entrada micro-USB al borde de la PCB para poder programar el microcontrolador aun estando dentro de la carcasa.

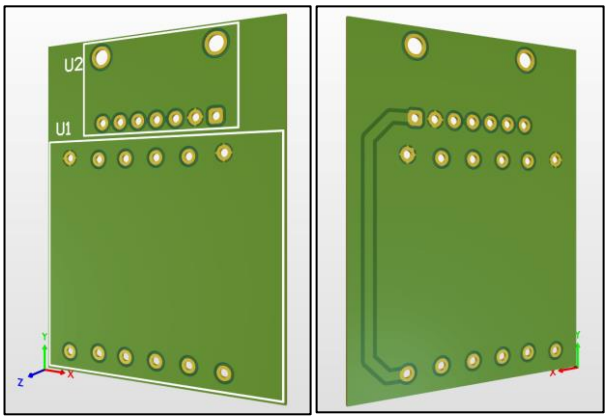

Figura 20. Diseño en 3D de la PCB del sistema

Una vez la PCB ya estaba diseñada, quedando con unas dimensiones de 32.51 x 43.31 milímetros, se mandó a fabricar a la empresa JLCPCB [27]. Tras esto, da lugar todo el proceso de soldadura, en el que primero se sueldan los dos componentes principales de la PCB haciendo uso de algunas tiras de pines macho, siendo el resultado el que puede observarse en la Figura 21. Acto seguido se soldaron los cables necesarios para la comunicación I2C y, por último, se soldó la carcasa de las pilas de botón, la cual se pegó con silicona a la parte baja de la placa, dejando el botón de apagado/encendido de tal forma que fuera accesible. Además, hay que asegurarse que esta quedara con la apertura hacia abajo, haciendo factible un posible cambio de pilas en caso de descarga.

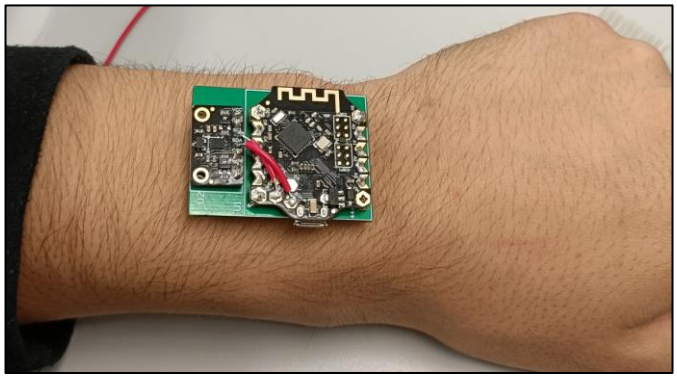

Figura 21. PCB con el microcontrolador y la IMU soldados

4.1.2 Carcasa

La carcasa se ha fabricado con la ayuda de una impresora 3D proporcionada por Alejandro Tapia Córdoba, autor del diseño. Para ello se le comunicaron ciertos requisitos, como que debía haber un hueco por el que acceder al micro-USB del microcontrolador por si hiciera falta una reprogramación; que el interruptor que encendía y apagaba el sistema quedara accesible; o que si hiciera falta un cambio de pilas no fuera un proceso tedioso. Además, esta carcasa debía estar bien ajustada y con espacio para añadirle una correa.

Con estas especificaciones se realizó una primera versión con algunos desajustes e imperfecciones, pero que sirvieron de base para una versión mejorada. El problema del primer prototipo, como se puede observar en la Figura 22, era que las dimensiones de la PCB eran demasiado grandes para encajar, aunque el concepto estaba bien ejecutado y la alimentación sí que coincidía a la perfección.

Figura 22. Primera versión de la carcasa

Todo esto fue corregido en el segundo prototipo, el cual fue asegurado con dos tornillos y tuercas asegurando que la PCB no sufriera daños. Además, este diseño no es completamente cerrado ya que uno su parte inferior se dejó parcialmente descubierta para poder hacer el cambio de pilas del que se habló anteriormente. Así pues, en la Figura 23 pueden observarse varias imágenes de cómo queda el dispositivo en la muñeca sin incorporar la correa, el orificio dedicado a la programación del microcontrolador y la parte inferior habiendo quitado la tapadera del dispositivo de alimentación (únicamente para la fotografía), donde puede apreciarse también el interruptor de apagado/encendido.

Figura 23. Versión final de la carcasa

4.1.3 Correa

Para la correa dedicada al ajuste del dispositivo a la muñeca se escogió una hecha de nailon de la marca Cobee, la cual hubo que retocar de forma mínima para que se ajustara a la perfección a la carcasa completando la implementación hardware del dispositivo. Se estuvo evaluando la posibilidad de escoger una correa elástica, pero, aunque también cumpliera con su función, era menos económica.

Figura 24. Implementación hardware al completo

4.2 Software

Al igual que la sección de hardware esta se dividirá en dos partes principales. La primera se enfoca en el proceso donde la información es recogida por el sensor y enviada por el microcontrolador. En esta parte se explicará cómo se ha realizado la configuración de la red cliente-servidor, cómo se ha enfocado el código de forma general, acompañado de su correspondiente diagrama de flujo, y qué métodos se han usado para que funcione. La segunda parte en cambio estará dedicada al código realizado cuya función es toda la parte del procesamiento de los

datos, partiendo de un enfoque amplio para acabar en sus características más específicas.

4.2.1 Recogida y envío de datos

Para esta parte de la funcionalidad software del sistema, antes de nada, es necesario definir los diferentes nodos que conforman la red, ya que existen dos roles diferentes en los dispositivos BLE, que son central y periférico. Por lo tanto, si se quiere establecer una comunicación inalámbrica, uno de los microcontroladores deberá estar configurado como central y el otro como periférico. Para este caso, el central será el que espera y recibe los datos y el periférico el que los envía.

Para ello, se deberá conectar uno de ellos al ordenador para activar el modo AT en el monitor del puerto serie enviando "+++", habiendo ajustado previamente los baudios a 115200. Este modo activa una interfaz de comandos que permite cambiar la configuración del dispositivo mediando instrucciones AT específicas, dando un mayor control y flexibilidad en la personalización de su funcionamiento. Con este modo activo, se usa uno de los siguientes comandos dependiendo del rol del nodo en cuestión:

```
AT+SETTING=DEFCENTRAL
AT+SETTING=DEFPERIPHERAL
```

Además, existe un comando que es "AT+BLUNODEBUG=OFF", el cual asegura una comunicación inalámbrica más estable. En este caso no se usa ya que impide el uso del puerto serie a través del ordenador.

Una vez se haya configurado el rol del dispositivo, se usará el comando "AT+EXIT" para abandonar el modo AT y se hará lo mismo con el segundo. Para comprobar que todo ha sido ajustado de forma correcta se conectará el central al ordenador mediante cable micro-USB y el periférico se alimentará mediante una batería externa. Si todo ha salido bien, se deberá observar un LED verde en ambos dispositivos que confirmará la comunicación.

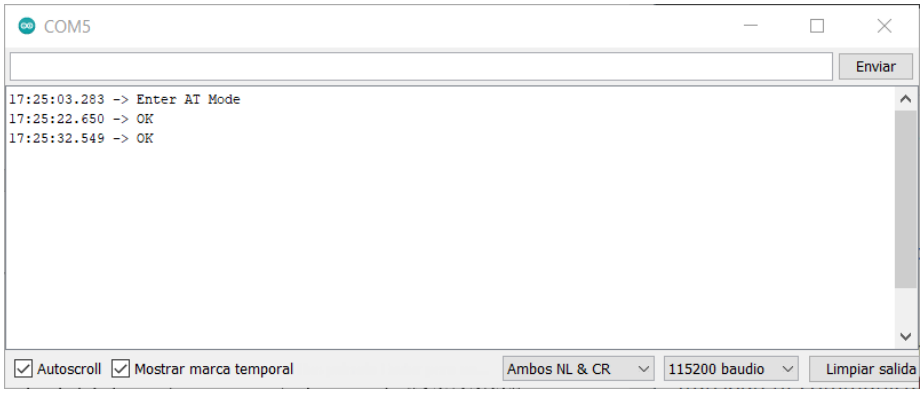

Figura 25. Configuración del nodo central en el monitor del puerto serie

Tras haber configurado los dispositivos es necesario comprobar cuál era la frecuencia de muestreo de la que se dispone en el sistema, ya que, como se comentó anteriormente, se buscaba que

ésta fuera de 20Hz, lo que corresponde a 50 milisegundos entre muestra y muestra. Para ello, se hizo una prueba con un código simple cuya función era la de estar todo el tiempo tomando datos y enviándolos, mostrando junto a ellos el tiempo en microsegundos que pasaba entre muestras.

Tras hacer la prueba se vio que la frecuencia de muestreo era aproximadamente de 33 milisegundos, aunque esta cambiaba ligeramente con la distancia entre dispositivos, con la carga actual de la batería y con los posibles obstáculos del camino. Aun así, nunca se llegaron a sobrepasar el tiempo correspondiente a los 20Hz buscados, por lo que no debería haber problema con la programación.

A partir de esto, la primera opción fue controlarlo a través de pequeños tiempos de espera haciendo uso de la función "delay()", pero como se ha comentado, la existencia de varios factores que hacen que varíe el tiempo de muestreo hicieron que se descartara la idea. Aun así, se llegó a probar este método asumiendo el error que iba a existir para ver su magnitud, consiguiéndose que variara entre 49.8 y 50.2 milisegundos, lo cual era mejor que lo esperado ya que suponía un error del ± 0,4 %.

La segunda opción, que fue la que se implementó debido a que era óptima, consistía en esperar a que el tiempo desde que se iniciara el dispositivo o desde que se había tomado una muestra llegara a ser 50 milisegundos, es decir, se estaba forzando al valor buscado. Al probar este método se obtuvo un tiempo de muestreo fiable ya que no presentaba fallos. En el diagrama de flujo de la Figura 26 puede verse de forma más clara este proceso además del comportamiento completo del código, el cual se comentará a continuación.

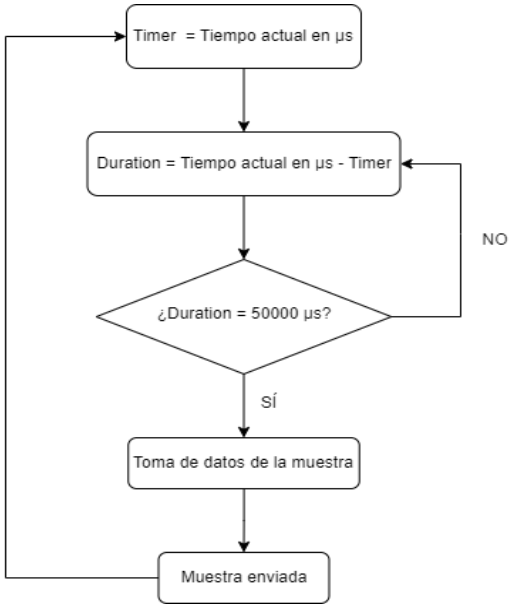

Figura 26. Diagrama de flujo del código de recogida y envío de datos.

El código para la recogida y el envío de datos, el cual ha sido diseñado en Arduino IDE cubriendo todos los posibles casos que se presentan en el diagrama de fujo, se encuentra en el Anexo

III del presente documento. En este, lo primero que se hace es inicializar tanto el puerto serie, ajustando el *BaudRate* en 115200, como el sensor BMX160, mostrando un mensaje si este no es detectado porque, por ejemplo, no haya sido conectado de forma correcta.

Acto seguido se define la función *loop()*, función que se estará repitiendo de forma continua hasta que se rompa el enlace de comunicación. En esta, de primera mano se crea una variable *start* que contiene el instante en microsegundos en el que comienza cada ciclo de la función, junto a otra variable llamada *duration* la cual representa el tiempo que ha pasado desde que ha iniciado el ciclo actual.

Dicha variable se encontrará en constante evaluación y actualización hasta que su valor corresponda a 50 milisegundos, instante en el cual se toman muestras del sensor a partir de la librería proporcionada por el fabricante y se envían. En la sección correspondiente a los resultados, se podrá observar que dichas muestras estarán disponibles en tiempo real al abrir el puerto serie asociado al dispositivo central.

4.2.2 Procesamiento de la información

El código dedicado a procesar la información, cuyo diagrama de flujo puede observarse en la Figura 27, ha sido programado en Matlab. Este corresponde al Anexo IV del presente documento y a continuación se explicará su funcionamiento al detalle junto al por qué de algunas decisiones, para acabar mostrando el resultado final en su sección correspondiente.

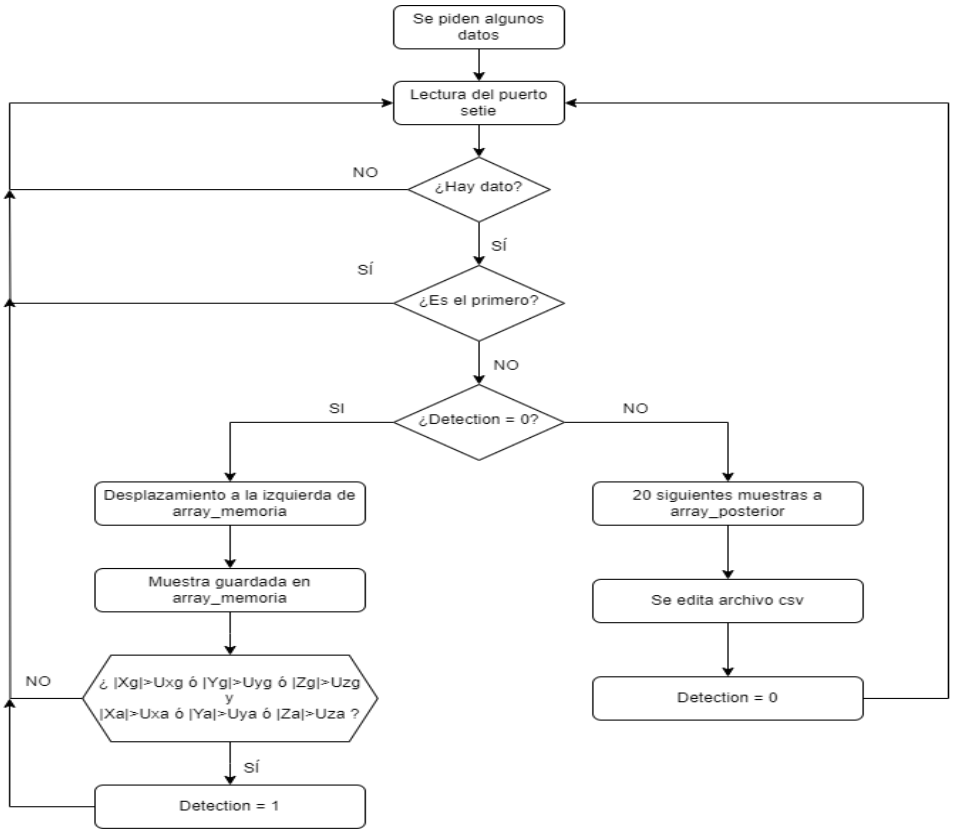

Figura 27. Diagrama de flujo del código de procesamiento de la información

Como se comentó anteriormente en el apartado de requisitos del sistema, el fin de este código es ir guardando en un archivo '.csv' un conjunto de muestras cada vez que se detecte que se ha realizado un golpe, buscando en todo momento mantener el formato de la Figura 18. Para empezar, se pide al usuario que introduzca ciertos datos y características del deportista que realiza la prueba, los cuales servirán posteriormente para rellenar las últimas columnas del archivo final. Una vez que se hayan introducido, se imprimirá un mensaje indicando que el dispositivo que recoge los datos puede encenderse a partir de ese momento. Esta información solicitada es:

- Tipo de golpe: este valor será un número entero que dependerá del golpe con el que se realizará el experimento.

Identificador	Tipo de golpe	Identificador	Tipo de golpe
1	Fondo de derecha	7	Globo de revés con pared
2	Fondo de revés	8	Volea de derecha
3	Revés con pared	9	Volea de revés
4	Globo de derecha	10	Bandeja
5	Globo de revés	11	Remate
6	Globo de derecha con pared	12	Saque

Tabla 3. Identificadores para los diferentes tipos golpes

- Sexo: donde habrá que indicar su género o si prefiere no contestar.
- Nivel: al igual que el tipo de golpe, este parámetro será un número entero atendiendo a la siguiente tabla:

Identificador	Nivel
1	Principiante
2	Amateur
3	Experimentado
4	Muy alto nivel
5	Profesional

Tabla 4. Identificadores para los diferentes niveles

- Mano: para indicar si se es diestro o zurdo. En caso de ser ambidiestro se indicará la más habitual.
- Revés: por si dicho golpe se realiza con una o dos manos.
- Altura: en centímetros.
- Edad: en años.
- ID: este será un identificador que se asignará al deportista por parte del usuario que realiza la prueba. Debe ser un número entero.

Acto seguido, se hará un proceso de configuración en el que se ajustan los parámetros del puerto serie para abrirlo posteriormente y se crean las diferentes variables, vectores y estructuras que serán necesarias durante el transcurso del código, inicializándolas con su valor correspondiente.

Una vez hecho esto, el código entrará en un bucle en el que estará leyendo constantemente lo que llega por puerto serie Es aquí cuando se encontró la primera complicación, ya que en cualquier escenario posible el primer dato recibido no era limpio, es decir, no correspondía con el formato definid. Es por ello que se decidió no trabajar con la primera muestra recibida y empezar a analizar a partir de la segunda, asumiendo que desde que se enciende el dispositivo hasta que se da el primer golpe iban a pasar más de 50 milisegundos.

Partiendo de la segunda muestra y asegurando que la bandera que indica que una detección de golpe esta desactivada, cada dato que se recibe se escribe en la última posición de un vector llamado *array_previo* en el que se encuentran los 19 datos anteriores y el actual. Dicho vector se irá desplazando a la izquierda con cada muestra, olvidando la más antigua. A su vez, el dato actual será evaluado de tal forma que si los parámetros recogidos por el giroscopio o acelerómetro superan cierto umbral se activa la bandera de detección de golpes. Para definir dichos umbrales se analizaron los valores absolutos de diferentes muestras al realizar varios movimientos de golpeo, resultando lo siguiente:

Umbral giroscopio ejes X, Y, Z = 50 g

Umbral acelerómetro ejes X, Y = 15 m/s^2

Umbral acelerómetro eje Z = 20 m/s^2

Cuando ya se ha detectado un golpe, se procederá a guardar las siguientes 20 muestras en el vector llamado *array_posterior*. Terminado este proceso es necesario calcular tanto el número de golpe, el cual se gestiona a través de un contador que se incrementa cada vez que la señal de detección se pone a 1, y el tiempo que ha pasado desde que han empezado a llegar muestras hasta que se ha detectado el golpe. Este último se calcula multiplicando el número de muestras capturadas hasta ese instante por 0.05 segundos, obteniendo finalmente el tiempo en segundos el que ese golpe se ha producido durante el experimento.

Para finalizar, se guardan en el archivo '.csv' tanto los datos guardados en *array_posterior* como los nuevos, junto a todos los demás campos que permiten crear un conjunto final de datos compatible con el acordado en la Figura 18. Transcurrido este proceso, se desactiva la bandera de detección de golpes y el sistema vuelve a seguir comparando las muestras recibidas a partir de ese

instante para detectar otro posible golpeo.

Descrito el comportamiento del código, se comprobará su funcionamiento a través de algunos experimentos en el apartado de pruebas de validación de la sección de resultados.

5 RESULTADOS

En esta sección se verán los resultados obtenidos tras los diferentes experimentos realizados al implementar tanto las soluciones hardware como las software anteriormente desarrolladas. Dichos resultados se dividirán en dos secciones, estando la primera dedicada a las pruebas de comunicación en las que se verificará que los datos son recogidos y enviados a otro dispositivo de manera correcta. La segunda en cambio estará dedicada a las pruebas de validación que buscaran aprobar el buen funcionamiento del sistema, encontrando también las flaquezas de este que puedan ayudar a la propuesta de mejoras constructivas de cara al futuro. Además, se analizará el coste de fabricación del dispositivo, comparándolo con el proyecto anterior.

5.1 Pruebas de comunicación

Este experimento tiene dos objetivos principales, comprobar que el sensor recoge y envía datos correctamente y probar la aplicación móvil que proporciona el fabricante. Esta recibe el nombre de Bluno Remote [28], la cual está disponible tanto en iOS [29] como en Android [30]. Dicha aplicación a través de su interfaz, la cual se puede observar en la Figura 28, permite escanear los diferentes dispositivos Bluetooth que haya por la zona, enviar datos a los mismos y ver cuál es su respuesta. Además, cuenta con una cruceta de direcciones cuya función es mandar comandos de movimiento a un dispositivo cuyo código esté preparado para ello, por lo que en este caso no se hará uso de estas.

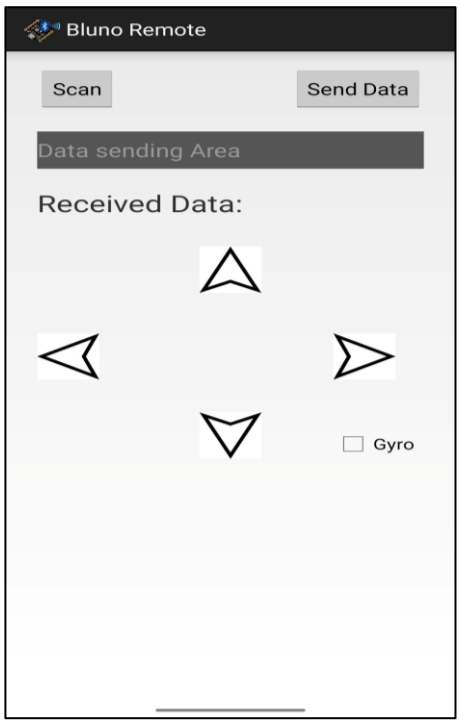

Figura 28. Interfaz de la aplicación Bluno Remote [28]

Para esta prueba se ha precisado del código plasmado en el Anexo I del presente documento. La función del código es enviar un número predeterminado de muestras tomadas por el giroscopio, acelerómetro y magnetómetro. Para ello, al inicio se imprimirá por pantalla una pregunta que hará al usuario decidir si quiere 50 o 100 muestras, las cuales se empezarán a mandar una vez el dispositivo haya recibido una respuesta válida.

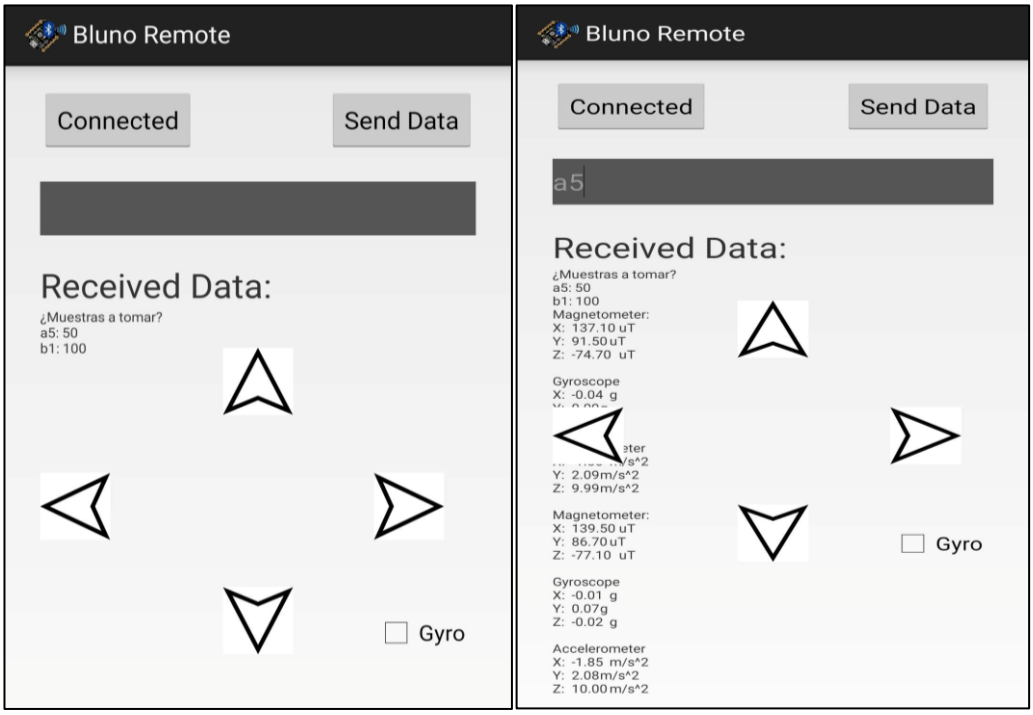

Figura 29. Resultados del experimento de la prueba de comunicación móvil

De cara a las pruebas de comunicación se ha realizado otro experimento que consiste en verificar que el enlace entre el cliente y el servidor funciona correctamente. Para ello, se ha hecho uso del código que se encuentra en el Anexo II de este trabajo, el cual corresponde al diagrama de flujo mostrado en la Figura 26. Es por esto por lo que, como la funcionalidad del código ya ha sido estudiada en profundidad anteriormente, se verán directamente los resultados como así indica el nombre de la sección.

Este experimento, el cual se realizó con una separación de dispositivos de 5 metros aproximadamente, sirve también para demostrar la realidad de la distancia máxima a la que los dispositivos pueden trabajar sin error, pues el fabricante asegura que la esta longitud de transmisión es de 20-30 metros.

Los resultados obtenidos se pueden observar en la Figura 30 tomada desde el visualizador de puerto serie de Arduino**,** donde se observa que todo ha ido como se esperaba, respetando el tiempo entre la toma de las diferentes muestras. Aun así, ha ocurrido un contratiempo y es que, debido a la frecuencia de muestre y transmisión, es decir, a la velocidad a la que las muestras son recogidas y

enviadas, cuando la distancia entre dispositivos superaba los 6 metros la comunicación empezaba a fallar y a cortarse por momentos.

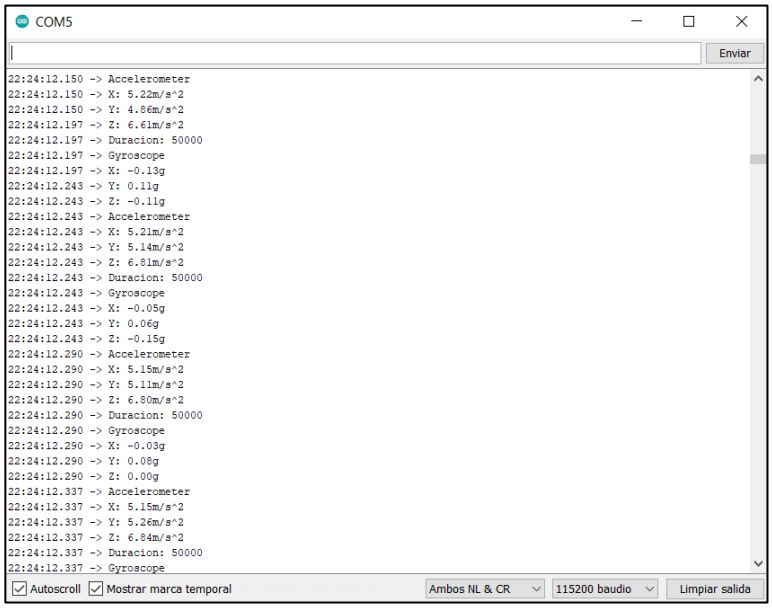

Figura 30. Resultados del experimento de la prueba de comunicación cliente-servidor

Una vez comprobado el funcionamiento de la red del sistema, se cambió el formato en el que los datos eran enviados para facilitar su procesamiento posteriormente. Dicho formato escogido es el siguiente:

TIME = T; Xg=‘’; Yg=‘’; Zg=‘’; Xa=‘’; Ya=‘’; Za=‘’;

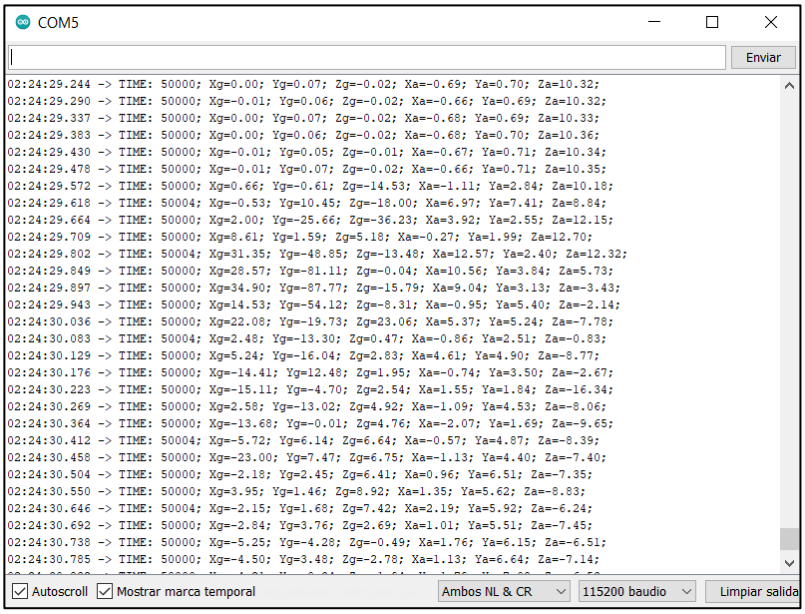

Figura 31. Formato final en la transmisión de datos

5.2 Pruebas de validación

Para validar el sistema al completo se ha realizado un experimento dividido en varios casos haciendo uso del código programado en Matlab, el cual se encuentra en el Anexo IV del presente documento. Dicho experimento ha consistido en seleccionar a cuatro personas de diferentes características y hacer que devuelvan una cantidad determinada de bolas con un tipo de golpe específico. Estas pueden observarse en la siguiente tabla:

ID	SEXO	EDAD	ALTURA	MANO	REVÉS	TIPO DE GOLPE
1	Hombre	27	178	Diestro	1 mano	Fondo de revés
2	Hombre	23	183	Diestro	1 mano	Remate
3	Mujer	56	170	Zurda	1 mano	Saque
4	Hombre	56	175	Diestro	1 mano	Fondo de derecha

Tabla 5. Características de los participantes del experimento

Así pues, se fueron realizando los diferentes casos uno tras otro, guardando todos los datos en un mismo archivo final, que es el que le llegaría a la red neuronal entrenada en [9]. Los resultados se observan a continuación:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	[-3.02,-1.78	[3.24,3.27,2	[10.24,10.1,	[2.63,4.02,2	[-2.73,-4.8,	[4.67,8.49,7	1	1	4.7	Hombre	2	Diestro	1	178	27	1
2	[-0.97,-1.9	[2.02,2.51,4	[11.16,10.94,	[0.06,0.85,4	[-0.66,-2.0	[7.96,11.71,2	1	2	7.85	Hombre	2	Diestro	1	178	27	1
3	[-2.39,-2.4	[4.21,4.32,3	[12.24,12.08	[3.31,2.62,7	[-3.89,-6.5	[9.95,14.17,	1	3	11.1	Hombre	2	Diestro	1	178	27	1
4	[-0.52,0.39	[1.49,2.96,3	[11.08,10.68	[0.69,0.99,3	[0.55,-1.36,	[2.48,5.55,7	1	4	14.35	Hombre	2	Diestro	1	178	27	1
5	[-1.4,-2.84,	[2.79,3.52,3	[11.32,11.52,	[2.9,4.05,7.	[1.87,-0.76,	[11.23,14.5,7	1	5	17.8	Hombre	2	Diestro	1	178	27	1
6	[-1.67,-2.14	[2.98,2.76,2	[10.04,10.66	[3.87,2.93,4	[-1,-3.12,-5	[13.09,14.32	1	6	21.1	Hombre	2	Diestro	1	178	27	1
7	[-0.59,-1.41	[3.52,4.63,4	[9.63,9.46,7	[1.88,2.95,5	[0.77,1.75,3	[4.23,10.72,	1	7	24.05	Hombre	2	Diestro	1	178	27	1
8	[-0.62,-0.7:	[2.95,2.43,2	[11.28,10.12,	[0.61,3.22,5	[-1.01,-0.52	[10.44,15.63	1	8	27.15	Hombre	2	Diestro	1	178	27	1
9	[-4.75,-4.9,	[3.32,3.28,2	[10.94,11.4,7	[1.84,3.73,7	[0.88,-2.55	[16.91,19.52	1	9	30.25	Hombre	2	Diestro	1	178	27	1
10	[-3.39,-4.6	[2.14,2.2,2.4	[9.95,12.94,	[1.87,2.64,4	[-1.72,-3.5	[18.15,20.1,2	1	10	33.4	Hombre	2	Diestro	1	178	27	1
11	[-4.47,-6.2	[3.23,2.29,2	[10.85,13.42	[0.17,3.9,6.8	[-2.59,-4.6	[20.21,23.2	1	11	36.55	Hombre	2	Diestro	1	178	27	1
12	[-4.42,-4.1	[1.9,1.19,2.0	[10.09,12.28	[0.15,1.19,5.	[-0.39,-2.8	[17.66,17.56	1	12	39.5	Hombre	2	Diestro	1	178	27	1
13	[-1.39,-1.45	[2.21,3.54,3	[7.49,9.04,7	[1.84,2.07,0	[0.55,3.55,(	[11.69,15.19,	1	13	42.5	Hombre	2	Diestro	1	178	27	1
14	[-3.81,-2.57	[1.2,1.84,1.1	[11.79,13.73	[-1.23,0.83,	[-2.71,-5.37	[15.65,16.57	1	14	45.6	Hombre	2	Diestro	1	178	27	1
15	[-1.41,-2.7,-	[2.85,3.11,2	[10.48,12,13	[1.37,2.8,2.9	[-5.75,-7.4(	[13.89,17.7,	1	15	48.4	Hombre	2	Diestro	1	178	27	1
16	[-4.47,-4.6	[-2.71,-2.62	[9.18,9.13,9	[1.04,0.2,0.9	[0.18,0.39,(	[0.67,0.07,(	11	1	2.85	Hombre	3	Diestro	1	183	23	2
17	[-2.88,-0.5	[0.49,-1.05,	[10.78,11.62	[0.37,0.33,(	[0,-2.9,2.52	[-2.37,-0.13	11	2	5.3	Hombre	3	Diestro	1	183	23	2
18	[-1.29,-1.47	[-0.36,-1.08	[10.03,10.83	[0.52,-0.24	[-0.62,-0.7	[-0.39,-1.18	11	3	8.5	Hombre	3	Diestro	1	183	23	2
19	[0.18,-1.38,-	[-2,0.82,-0.	[13.14,10.25	[-0.4,2.53,(	[-2.32,3.3,(	[-0.47,0.46	11	4	10.65	Hombre	3	Diestro	1	183	23	2
20	[-0.34,-2.7:	[-0.98,0.02	[9.95,9.59,9	[0.3,2,-2.36	[0.76,8.88,-	[-0.69,4.27	11	5	13.8	Hombre	3	Diestro	1	183	23	2
21	[-1.23,-2.11	[-2.11,-2.66	[9.79,11.02,	[-1,-1.18,-3.	[0.09,-1.52,	[-4.37,-3.73	11	6	16.85	Hombre	3	Diestro	1	183	23	2
22	[-1,-0.69,-0	[0.09,-0.17,	[10.25,9.84,	[0.39,-0.26	[0.66,1.43,-	[0.47,-0.06	11	7	20.2	Hombre	3	Diestro	1	183	23	2
23	[-0.61,-0.65	[-0.03,-0.0!	[10.11,9.79,7	[1.02,-0.98,	[2.16,-0.85,	[1.1,-0.89,0.	11	8	23.15	Hombre	3	Diestro	1	183	23	2
24	[-0.94,-0.4	[0.22,-1.13,-	[10.46,11.32	[-1.02,-0.72	[-1,-0.66,1.8	[-0.63,-1.17	11	9	26.3	Hombre	3	Diestro	1	183	23	2
25	[-0.18,0.04,	[-0.24,0.32	[10.24,10.65	[-1.84,1.15,-	[-2.29,2.74	[-3.05,-1.35	11	10	28.95	Hombre	3	Diestro	1	183	23	2
26	[0.36,-0.83	[-3.29,-3.0(	[11.01,12.53,	[-5.51,-6.86	[-1.3,-4.06,	[-8.64,-14.7	12	1	5.75	Mujer	1	Zurda	1	170	56	3
27	[1.07,0.14,2	[-1.96,-2.08	[12.64,13.25	[-1.18,-6.03	[-1.52,-4.96	[-2.79,-4.68	12	2	10.9	Mujer	1	Zurda	1	170	56	3
28	[0.17,0.21,-1	[-2.42,-2.4(	[10.02,10.23	[-3.19,-4.02	[-0.89,-0.0(	[-7.58,-11.2	12	3	15.35	Mujer	1	Zurda	1	170	56	3
29	[0.66,0.78,(	[0.55,0.09,-	[10.46,10.16	[0.47,0.49,(	[0.51,0.69,(	[-0.19,-0.79	12	4	18.45	Mujer	1	Zurda	1	170	56	3
30	[0.9,-0.31,0	[-1.41,-1.87,	[10.56,11.12,	[-4.79,-6.0(	[0.46,2.61,1	[-3.57,-3.26	12	5	21.5	Mujer	1	Zurda	1	170	56	3
31	[2.84,2.39,4	[-5.19,-4.26	[9.82,9.14,8	[-11.03,-12.1	[-0.55,-1.14	[-18.06,-16.	0	1	6.7	Hombre	2	Diestro	1	175	56	4
32	[1.65,-0.27,	[-3.77,-2.28	[11,8.52,10.2	[-5.97,-10.9	[11.72,6.27,	[-19.28,-21.	0	2	9.3	Hombre	2	Diestro	1	175	56	4
33	[1.11,1.94,0.:	[0.94,-1.57,	[9.04,10.79,	[-7.16,-7.8,	[1.93,3.5,2.5	[-19.69,-19.	0	3	11.55	Hombre	2	Diestro	1	175	56	4
34	[6.1,1.53,4.7	[-7.76,-3.6:	[10.51,9,11.5	[-9.95,-5.9,	[10.9,12.38,	[-10.48,-10.	0	4	13.6	Hombre	2	Diestro	1	175	56	4
35	[2.28,1.49,1	[-5.03,-4.8:	[8.6,8.36,8.	[-6.39,-7.5!	[5.92,6.89,7	[-11.29,-13.3	0	5	16.2	Hombre	2	Diestro	1	175	56	4
36	[-0.64,-1.53	[-3.1,-3.42,	[8.95,9.87,7	[-12.28,-15.	[8.57,9.69,7	[-23.3,-24.3	0	6	18.3	Hombre	2	Diestro	1	175	56	4
37	[-19.6,0.82,	[-19.6,-6.58	[16.44,11.07	[-10.51,-6.3	[9.21,10.02,	[52.74,-20.	0	7	20.25	Hombre	2	Diestro	1	175	56	4

Figura 32. Archivo final tras experimento de validación abierto en Excel

Acto seguido se pasó a comparar las diferentes muestras de un mismo golpe para ver la relación entre ellas. Se escogió hacer para los 5 datos del eje Y del acelerómetro y del eje X del giroscopio para el golpe de saque, siendo estos los resultados:

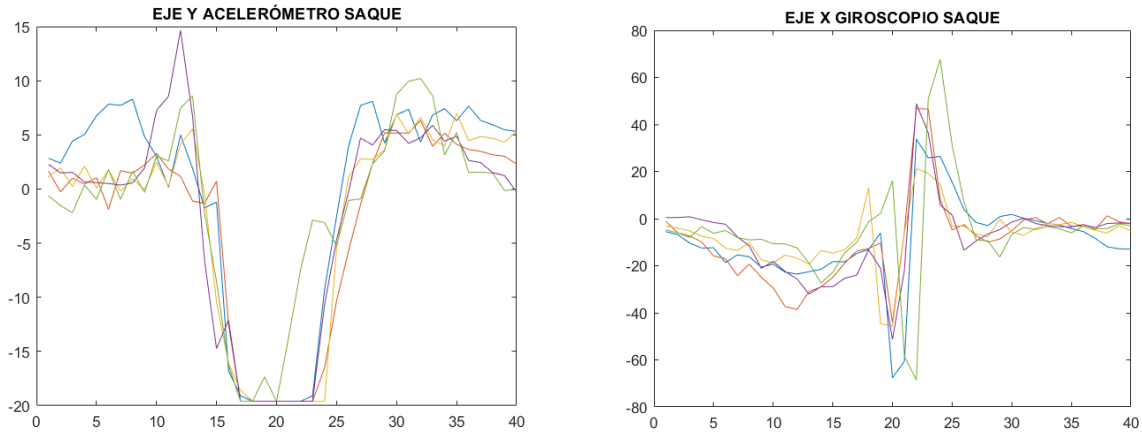

Figura 33. Representación de distintos datos de un conjunto de muestras de saque

Lo siguiente que se debe comprobar es que el dataset obtenido sea compatible con el que se obtuvo en el Trabajo Fin de Máster en el que se basa el presente trabajo. Como se puede observar en la Figura 34, las bases del formato están presentes, pero hay cambios mínimos como por ejemplo que deberían estar todos los datos seguidos unos de otros únicamente separados por comas, sin la existencia de corchetes, o que todas las preguntas previas al experimento deben responderse mediante números.


```
Ax0,Ax1,Ax2,Ax3,Ax4,Ax5,Ax6,Ax7,Ax8,Ax9,Ax10,Ax11,Ax12,Ax13,Ax14,Ax15,Ax16,Ax17,Ax18,Ax19,Ax20,Ax21,Ax22,Ax23,Ax24,Ax25,Ax26,Ax27,Ax28,Ax29,Ax30,Ax31,Ax32,Ax33,Ax34,Ax35,Ax36,Ax37,Ax38,Ax39,Ay0,Ay1,Ay2,Ay3,Ay4,
Ay5,Ay6,Ay7,Ay8,Ay9,Ay10,Ay11,Ay12,Ay13,Ay14,Ay15,Ay16,Ay17,Ay18,Ay19,Ay20,Ay21,Ay22,Ay23,Ay24,Ay25,Ay26,Ay27,Ay28,Ay29,Ay30,Ay31,Ay32,Ay33,Ay34,Ay35,Ay36,Ay37,Ay38,Ay39,Az0,Az1,Az2,Az3,Az4,Az5,Az6,Az7,Az8,Az9,
Az10,Az11,Az12,Az13,Az14,Az15,Az16,Az17,Az18,Az19,Az20,Az21,Az22,Az23,Az24,Az25,Az26,Az27,Az28,Az29,Az30,Az31,Az32,Az33,Az34,Az35,Az36,Az37,Az38,Az39,Vx0,Vx1,Vx2,Vx3,Vx4,Vx5,Vx6,Vx7,Vx8,Vx9,Vx10,Vx11,Vx12,Vx13,
Vx14,Vx15,Vx16,Vx17,Vx18,Vx19,Vx20,Vx21,Vx22,Vx23,Vx24,Vx25,Vx26,Vx27,Vx28,Vx29,Vx30,Vx31,Vx32,Vx33,Vx34,Vx35,Vx36,Vx37,Vx38,Vx39,Vy0,Vy1,Vy2,Vy3,Vy4,Vy5,Vy6,Vy7,Vy8,Vy9,Vy10,Vy11,Vy12,Vy13,Vy14,Vy15,Vy16,Vy17,
Vy18,Vy19,Vy20,Vy21,Vy22,Vy23,Vy24,Vy25,Vy26,Vy27,Vy28,Vy29,Vy30,Vy31,Vy32,Vy33,Vy34,Vy35,Vy36,Vy37,Vy38,Vy39,Vz0,Vz1,Vz2,Vz3,Vz4,Vz5,Vz6,Vz7,Vz8,Vz9,Vz10,Vz11,Vz12,Vz13,Vz14,Vz15,Vz16,Vz17,Vz18,Vz19,Vz20,Vz21,
Vz22,Vz23,Vz24,Vz25,Vz26,Vz27,Vz28,Vz29,Vz30,Vz31,Vz32,Vz33,Vz34,Vz35,Vz36,Vz37,Vz38,Vz39,tipo_golpe,numero_golpe,tiempo_golpe,sexo,nivel,mano,reves,altura,edad,id
-0.010367,0.023032,0.070309,0.063036,0.13286,0.204867,0.388883,0.484649,0.623086,0.666484,0.790616,0.736793,0.4381,0.280267,0.295542,0.67206,1.091249,1.575171,2.041637,1.834831,1.17853,-0.95871,-3.249098,-8.087
323,-6.25679,-7.112322,-8.00414,-8.087323,-8.087323,-8.087323,-5.415328,-3.627497,-2.802325,-1.385488,-1.226279,-0.979691,-0.741741,-0.62326,-0.460842,-0.136747,-0.566881,-0.410566,-0.488,-0.43493,-0.367628,-0.
272585,-0.353878,-0.251116,-0.426246,-0.285611,-0.232783,-0.254011,-0.356291,-0.539864,-0.557956,-0.499096,-0.492342,-0.38572,-0.53914,-0.59028,-0.70848,-0.717406,-0.223134,-1.154266,-1.185143,5.750299,-2.68556
8,-4.118208,-3.828014,0.039575,0.94155,-0.540828,0.555995,0.885222,-0.414185,-0.121337,0.01214,-0.239537,-0.648415,-0.711616,0.497873,0.403318,0.35994,0.358722,0.448402,0.618503,0.747175,1.039854,1.183392,1.150
737,1.0903,1.13319,1.229694,1.188753,1.0613,0.971376,0.644822,-0.232593,-1.415561,-2.551288,-3.766795,-5.30528,-5.322875,-0.721158,-5.424589,-4.176061,-2.02705,-0.714409,0.090168,1.462425,0.89193,0.22542,0.3258
23,0.43378,0.273184,0.265386,0.270991,0.18521,0.159378,0.219327,1.447913,1.208454,0.960137,0.719456,0.639127,0.360573,0.652872,0.257947,0.389894,0.417994,0.617441,0.766493,0.735949,0.773212,0.81078,0.323004,-0.
253958,-1.289374,-3.483297,-5.459446,-7.225763,-8.547981,-5.917595,2.909102,5.322937,4.475667,2.145826,1.625369,9.306083,6.947533,1.715166,1.631478,1.676987,-0.64705,-0.421946,0.179756,-0.236854,0.206635,1.0297
76,0.878586,-1.972886,-1.847964,-1.602701,-1.557497,-1.465257,-1.518402,-1.271307,-1.119201,-1.314067,-1.487859,-1.622554,-1.802149,-2.469825,-2.916367,-3.164073,-3.38093,-3.720876,-4.112441,-4.54585,-4.080982,
-2.669577,0.910093,9.057508,10.002822,10.002822,10.002822,10.002822,10.002822,10.002822,9.91394,5.316569,2.201767,1.159326,0.252191,-0.121659,-0.162281,-0.396853,-0.582556,-0.826597,-0.955795,-0.869358,1.473209,1.447553,
1.472293,1.481151,1.432281,1.068206,1.096611,1.204429,1.236194,1.069122,0.899607,0.878837,0.762162,0.472612,0.018128,-0.436966,-0.939708,-0.854187,-0.628167,-0.276309,1.306749,4.609392,9.005483,9.911396,6.43374
1,-4.810151,-6.894728,-7.707179,-9.792368,-9.770071,-8.458849,-7.415491,-5.522114,-4.587795,-3.891409,-3.276573,-2.097909,-1.074099,0.396865,1.618595,10.0,1.0,2.679375,0.0,4.0,0.0,1.0,176.0,80.0,5.0
```


Figura 34. Dataset de partida

Tras haber detectado estas imperfecciones se pasó a una prueba que consistió en coger diferentes datos de un tipo de golpe de cada Dataset y compararlos para comprobar si los datos que se obtienen serían válidos para entrenar los algoritmos de la inteligencia artificial. Para empezar, se hizo una primera prueba en la que se representaría el mismo tipo de datos que en la figura 34 pero con las muestras de este Dataset, dando como resultado las siguientes gráficas las cuales se asemejan a las ilustradas anteriormente:

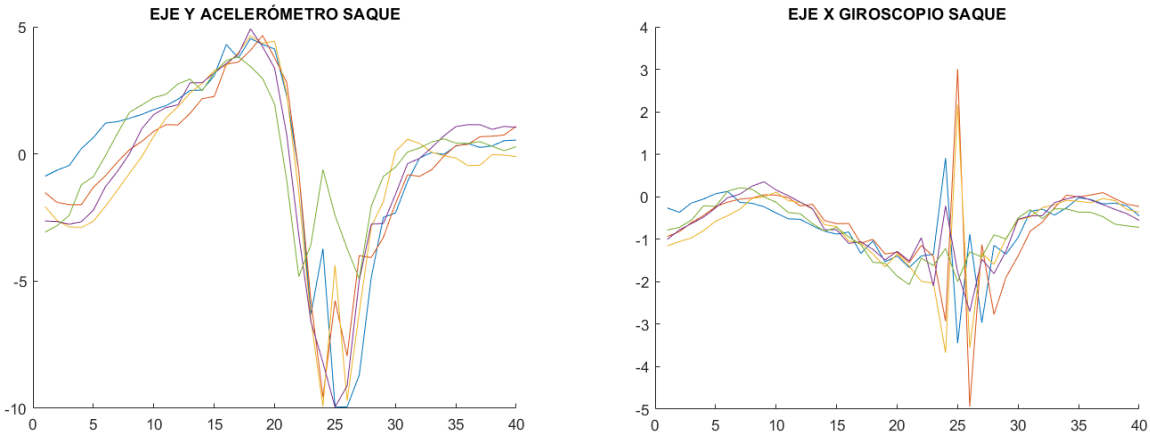

Figura 35. Representación de distintos datos de un conjunto de muestras de saque en proyecto anterior

Fue gracias a esto que se pudo detectar otro error, y es que cada IMU recogía los datos en una unidad de medida diferente. Es decir, mientras que en este trabajo los datos del acelerómetro se miden en m/s^2 y los del giroscopio en º/s, en el proyecto anterior el acelerómetro medía en Gs y el giroscopio en rad/s.

Para la conversión del acelerómetro lo único que hay que hacer es multiplicar los resultados por la inversa de la gravedad, que es de 9,8. En cambio para la del giroscopio primero habría que multiplicar los resultados por $\pi/180$ para convertirlos a rad/s.

Así pues, tras realizar las diversas conversiones, se han obtenido los resultados mostrados en la siguiente figura, en los que puede observarse que la magnitud de estos es coherente, pero difieren un tanto entre sí.

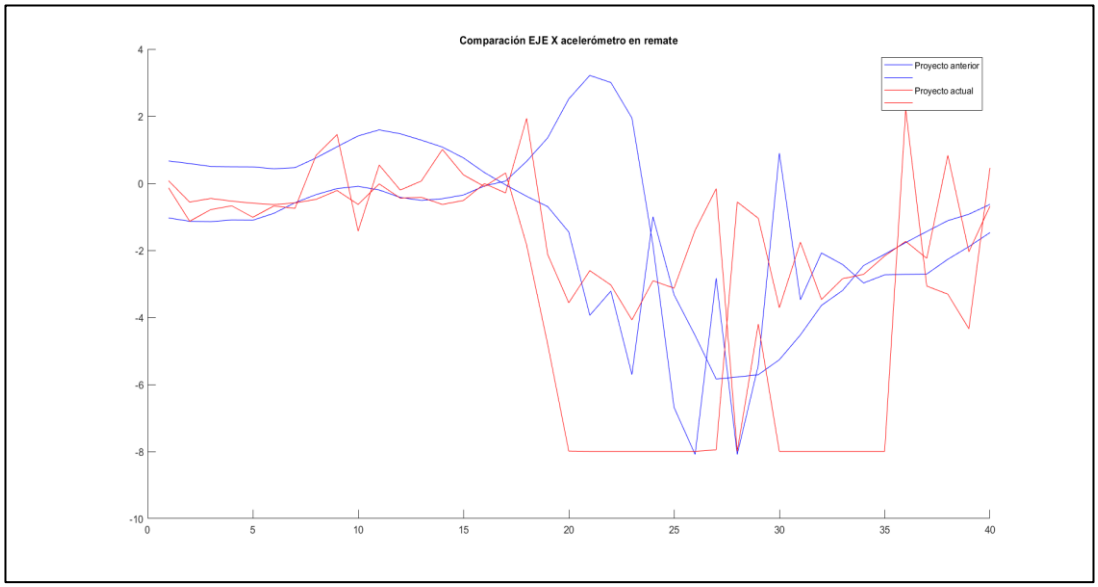

Figura 36. Comparación en eje x del acelerómetro en el remate

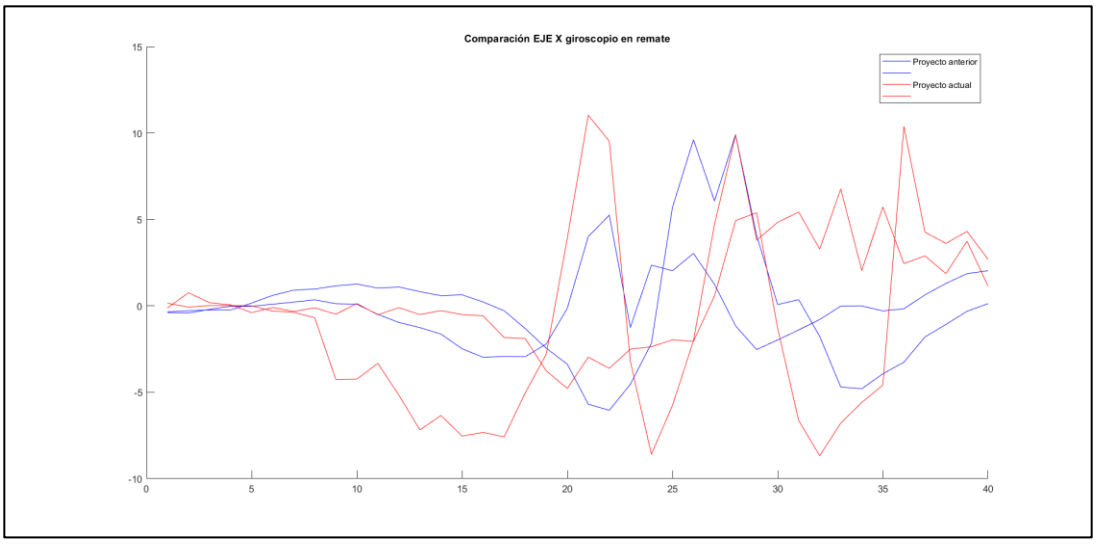

Figura 37. Comparación eje X del giroscopio en el remate

Una vez corregido todos los fallos, se asume que el archivo final contiene todos los conjuntos de muestras de los golpes realizados y en el formato que se buscaba. Aun así, durante estas pruebas hubo que hacer una pequeña modificación en el código que consistió en descartar las cinco primeras muestras que llegan al dispositivo en lugar de solo la primera. Esta decisión fue tomada debido a que había casos en los que la comunicación por Bluetooth no empezaba demasiado bien, dando error en el código. Respecto al código, el cambio es que la bandera dedicada a este descarte se va incrementando durante las cinco primeras muestras, así cuando valga cinco es cuando comienza el análisis de la información.

Para terminar con las pruebas de validación, teniendo ya el formato correcto en el archivo final de muestreas, se probará el funcionamiento de un algoritmo de *machine learning* basado en distancia con dichos datos en varios casos. Para ello se ha escogido uno de los códigos de [36] (knn_padel_st.py) que corresponde al método de K vecinos más próximos, aunque antes de ver los diferentes resultados es necesario explicar algunos conceptos básicos para una mayor comprensión.

Al entrenar y validar un algoritmo de *machine learning*, es común dividir los datos en conjuntos de entrenamiento, validación y prueba. La proporción de datos asignados a cada uno puede variar dependiendo del conjunto de datos y la naturaleza del problema abordado.

- Conjunto de entrenamiento: es el más grande y se usa para entrenar la red neuronal. Generalmente, se asigna aproximadamente el 70-80% de los datos. Cuánto más grande sea, más oportunidades tendrá la red neuronal para aprender patrones y generalizar a partir de dicha información.
- Conjunto de validación: después de cada iteración de entrenamiento, es necesario evaluar el rendimiento de la red neuronal y ajustar sus hiperparámetros si es necesario. Se asigna alrededor del 10-15% de los datos a este conjunto.
- Conjunto de prueba: una vez que se haya entrenado y ajustado la red neuronal, se necesita examinar su rendimiento final en un conjunto de datos independiente. Este contiene

aproximadamente el 10-20% de los datos.

En el caso de la red neuronal escogida para las pruebas de validación, solo cuenta con los conjuntos de entrenamiento y prueba, correspondiéndoles un porcentaje de 70 y 30% respectivamente.

Respecto al método de los K vecinos más próximos, es una técnica utilizada en el aprendizaje automático para clasificar o predecir datos. Se basa en la idea de que los objetos similares suelen estar cerca unos de otros en un espacio de características. En resumen, el método consiste en encontrar los K ejemplos de entrenamiento más cercanos a un nuevo dato y tomar una decisión basada en la mayoría de las etiquetas de clase de esos vecinos. En otras palabras, se "pregunta" a los vecinos más cercanos qué clase pertenece el nuevo dato y se toma esa respuesta como predicción. El valor de K se elige previamente y afecta el rendimiento del modelo: un K pequeño puede ser más flexible pero ruidoso, mientras que un K grande puede suavizar la clasificación, pero perder detalles importantes.

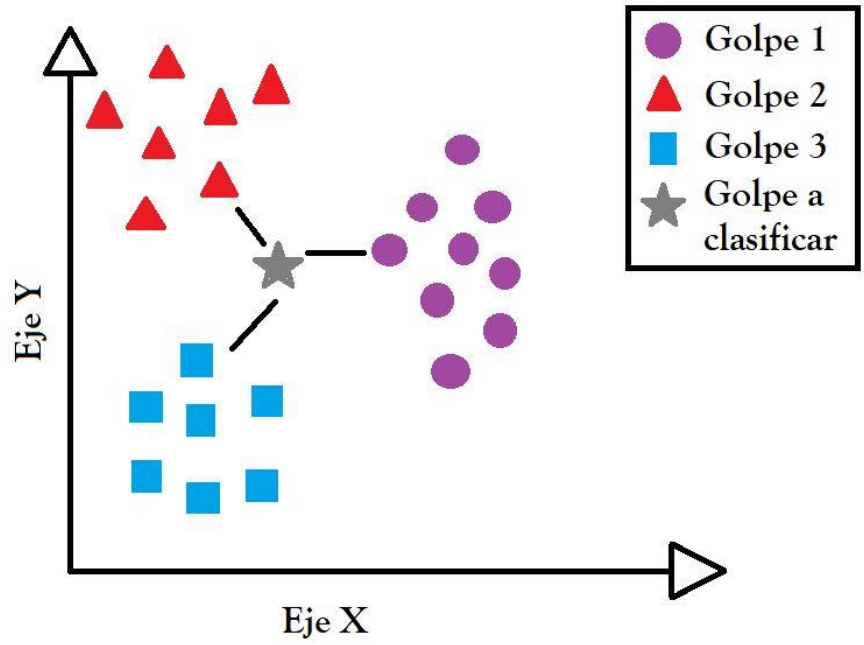

Figura 38. Técnica de clasificación de K vecinos más próximos

Para el experimento se ha tenido que recoger un nuevo *Dataset* con los mismos tipos de golpe que en la Tabla 5, pero con 15 muestras de golpeo en cada uno. Aun así, este no es válido para entrenar y validar la red neuronal, pues al ser un conjunto de muestras tan pequeño, los resultados no serían fiables. Es por ello por lo que lo que se ha hecho es entrenarla con el *Dataset* de partida recogido en el trabajo anterior [9] y usar el actual como conjunto de prueba, viendo si los golpes se clasifican correctamente.

Para ello hubo que modificar algunas líneas del código de Python de [36] para adaptarlo a las necesidades del experimento. Además, surgió un problema debido a que al haber entrenado la red neuronal con 13 tipos de golpes e intentar hacer el *test* con solamente 4, la gráfica de la matriz

resultante suprimía automáticamente las columnas de los que no se encontraban en el conjunto de prueba. Para solucionarlo, se realizó una adaptación de datos en el *Dataset* recogido que consistió en añadir conjunto de muestras de valor 0.0 para cada golpe que falta. Por esto, en la Figura 39 puede observarse que dichos tipos de golpes están clasificados como globos de revés, dando a entender que es el que tiene los valores más próximos a cero.

Respecto a la figura, se observa que se han detectado 11 de 15 golpes de derecha, siendo los 4 golpes restantes confundidos con otros estilos de derecha como son el globo, la volea o la derecha con pared. En el golpe de revés pasa algo similar, pues 13 son detectados, pero 2 son clasificados como globos de revés con pared. Para el golpe de remate se han identificado todos menos uno que se ha confundido con la bandeja, al igual que pasa con el saque y el golpe normal de derecha debido a la similitud de golpeo.

Así pues, de 60 posibles aciertos en la clasificación del conjunto de prueba, 52 han tenido éxito, obteniendo un porcentaje del 86.67%. Esto indica que, aunque no se tienen datos del resto de golpes, se esperan resultados similares.

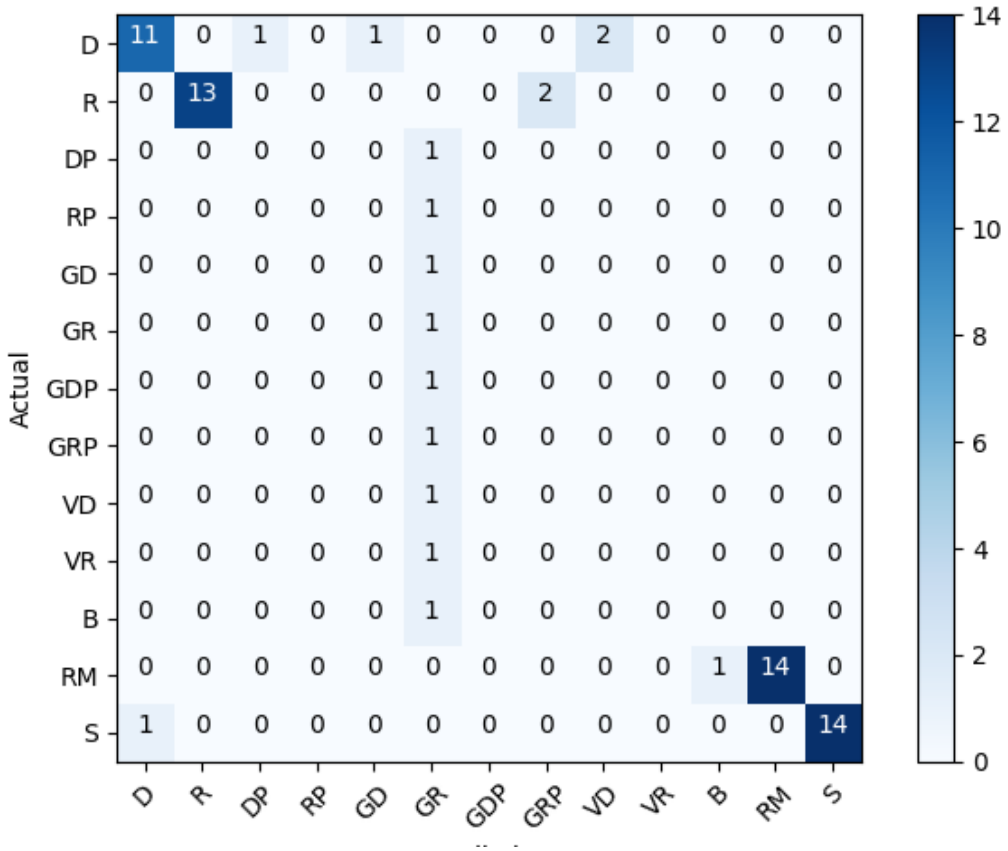

Figura 39. Matriz de confusión del experimento con red neuronal

5.3 Coste de fabricación

Esta sección tiene como fin comprender los costes asociados a cada uno de los componentes y procesos involucrados en el sistema, calculando el coste final de fabricación del dispositivo. Dicho coste es aproximado y puede variar dependiendo de la región, proveedor y otros factores específicos del proyecto, por lo que se recomienda realizar un seguimiento de estos actualizados de cara a posibles líneas futuras.

Para su determinación existen componentes cuyos precios son directos, como pueden ser el microcontrolador, la IMU o el soporte de la batería, y procesos cuyo valor debe estimarse en función de ciertos parámetros. Por ejemplo, para el proceso de soldadura se ha calculado cierto precio pensando tanto en la duración de esta, que es de 1 hora, como en el sueldo medio de un trabajador de este campo. O también para la impresión de la carcasa principal, cuyo valor aproximado se ha calculado mediante [34] introduciendo sus dimensiones.

Así pues, los costes de fabricación del dispositivo se recogen en la siguiente tabla:

Componente / Proceso	Precio en €
Beetle BLE	14.90
IMU BMX160	13.90
Soporte de batería	0.95
Pila de botón 3.3V	1.25
Correa	4.48
PCB	5.48
Soldadura	9.65
Impresión 3D	5.45

Tabla 6. Resumen del coste de fabricación del dispositivo

El coste de fabricación total del proyecto asciende a la cifra de 57.31€. Este se comparará con el del proyecto anterior, el cual se recoge en la siguiente tabla:

Componente / Proceso	Precio en €
Raspberry Pi 4	74.10
Sense Hat	33.38
Plataforma 3D	5.45
Correa velcro	5.15
Batería externa	14.99

Tabla 7. Resumen del coste de fabricación del proyecto previo

El coste de fabricación total del proyecto previo a este asciende a la cifra de 133.07€, por lo que se ha logrado disminuir el mismo a más de la mitad.

6 CONCLUSIONES

En esta sección se exponen las diferentes conclusiones a las que se ha llegado en relación con los objetivos propuestos para la realización del presente Trabajo Fin de Máster, el cual consiste en el diseño de un sistema electrónico para recolección de datos de golpes en pádel.

La modernización del deporte mediante el uso de diferentes tecnologías es un reto de gran calibre hoy en día, pues de esta manera es posible conseguir una gran optimización y mejora de procesos con el fin de mejorar la experiencia tanto a atletas como a aficionados. Una de las soluciones planteadas para alcanzar dicho objetivo, específicamente en el pádel, es el almacenamiento y procesamiento de datos recogidos al dar diferentes golpes durante un partido o entrenamiento. Esto se puede conseguir gracias a la red descrita en este estudio.

Gracias a la adquisición de conocimientos sobre las características buscadas en el diseño de un *wearable,* se han evaluado diferentes dispositivos que podrían formar la red en cuestión, comparando sus características más significativas y seleccionando finalmente los más adecuados conforme a unas necesidades. Además, se ha podido estudiar y comprender el comportamiento de dos dispositivos al configurarse como cliente y servidor. También se ha logrado aprender sobre las prestaciones y limitaciones de una comunicación Bluetooth dedicada al envío, recepción y almacenamiento de los datos.

De esta manera, se ha conseguido implementar una solución de bajo coste, poco consumo y una larga vida útil que beneficie al sector del deporte. Así pues, se puede concluir que se han alcanzado los objetivos perseguidos en el presente trabajo con respecto al diseño general del sistema.

6.1 Líneas futuras

Como continuación al presente proyecto, se proponen una serie de mejoras y extensiones basadas en los hallazgos, resultados y necesidades identificadas durante la implementación y evaluación del sistema, buscando aprovechar las diferentes oportunidades existentes para el crecimiento del mismo. Dichas propuestas son:

- Minimización dimensional: esto podría hacer un sistema aún más compacto y cómodo para el brazo humano. Esto puede implementarse a través de una placa que contenga tanto el microcontrolador como el sensor IMU, o encontrando otro dispositivo de alimentación de menor tamaño.
- Batería recargable: ya que se ha hablado de cambiar el dispositivo de alimentación, lo óptimo sería encontrar alguno que permitiera recargarlo tras su uso contribuyendo al medio ambiente.
- Optimización del consumo: a pesar de que en este proyecto la mayoría de las decisiones se han tomado favoreciendo al bajo consumo del sistema, no se descarta que haya en el mercado nuevos dispositivos o versiones de ellos que ayuden a reducirlo aún más.
- Incorporación de pantalla LCD: esta idea sería interesante ejecutarla mediante una pantalla

colocada en la parte superior del dispositivo, con el fin de poder mostrar en tiempo real al portador alguna estadística durante su uso.

- Cambio en la topología de red: para ello se debería de prescindir de la estructura cliente-servidor formada por los dos microcontroladores y establecer una comunicación directa a través de una tecnología como Bluetooth entre el dispositivo principal y el ordenador.
- Implementación en dispositivo móvil: de alguna forma estaría interesante poder hacer algo con los datos que se reciben en el móvil desde el microcontrolador, no limitándose únicamente poder verlos.
- Optimización del procesamiento de datos: esto sería posible si la información recogida por el sensor se procesa directamente con el mismo microcontrolador, obteniendo un conjunto final de datos que puede ser almacenado en una tarjeta SD si la placa lo permite o enviado directamente de forma inalámbrica.
- Ampliación del rango de comunicación: ya que, en el peor de los casos, teniendo en cuenta que las dimensiones de un campo de pádel son de 20 x 10 metros, si el ordenador se encontrara junto a la red habría una distancia entre dispositivos de 10 metros.

7 REFERENCIAS BIBLIOGRÁFICAS

1. Escuela Técnica Superior de Ingeniería de Sevilla. [Internet]. [Consultado 16 de Mayo 2023]. Disponible en: https://www.etsi.us.es/
2. Paloma R. [Internet]. Breve historia de Internet de las cosas (IoT) [Consultado 16 Mayo 2023]. Disponible en: https://n9.cl/y4e2j
3. Bluetooth Technology [Internet]. [Consultado 16 de Mayo 2023]. Disponible en: https://www.bluetooth.com/
4. García M. [Internet]. IoT-Internet Of Things [Consultado 16 Mayo 2023]. Disponible en: https://www2.deloitte.com/es/es/pages/technology/articles/IoT-internet-of-things.html#
5. Fracttal [Internet]. Las 9 aplicaciones más importantes del Internet de las Cosas (IoT) [Consultado 16 Mayo 2023]. Disponible en: https://www.fracttal.com/es/blog/2018/10/10/9-aplicaciones-importantes-iot
6. Nougoir Tecnológic [Internet]. ¿Qué es IOT o Internet de las cosas y sus aplicaciones? [Consultado 16 Mayo 2023]. Disponible en: https://www.nougir.com/index.php/blog-3/item/13-que-es-iot-o-internet-de-las-cosas-y-sus-aplicaciones
7. Universidad Europea [Internet]. El uso de la tecnología en el deporte [Consultado 16 Mayo 2023]. Disponible en: https://universidadeuropea.com/blog/tecnologia-en-deporte/
8. Antonio José Palacios Álvarez [Internet]. La tecnología llega al pádel [Consultado 16 Mayo 2023]. Disponible en: https://www.padeladdict.com/la-tecnologia-llega-al-padel/
9. Cartes G. Comparación de algoritmos de aprendizaje automático para la clasificación de golpes de pádel. 2021
10. Aparicio M. Dispositivo de captura de movimiento basado en sensores inerciales con comunicación inalámbrica. 2015
11. Kaitt [Internet]. [Consultado 16 Mayo 2023] Disponible en: http://kaitt.es/
12. Menduiña C. Desarrollo de un sistema de monitorización de estrategias de pádel en tiempo real usando unidades de medición inerciales y algoritmos de Deep Learning. 2020
13. CIO [Internet]. Muestran cómo aplicar el IoT en el sector salud [Consultado 17 Mayo 2023]. Disponible en: https://cio.com.mx/118621-2/
14. Ardev [Internet]. Realidad aumentada y deporte, un mundo fascinante [Consultado 17 Mayo 2023]. Disponible en: https://ardev.es/realidad-aumentada-y-deporte/
15. CEVA [Internet]. Dynamic Calibration for IMU Motion Sensors [Consultado 18 Mayo 2023]. Disponible en: https://www.ceva-dsp.com/ourblog/optimize-your-imu-with-dynamic-calibration/
16. Musicovery [Internet]. Bluetooth, ¿cómo funciona? Ventajas y desventajas [Consultado 18 Mayo 2023]. Disponible en: https://musicoveryshop.wordpress.com/2015/04/09/bluetooth-

como-funciona/

17. Arduino [Internet]. Getting Started with the Arduino LilyPad [Consultado 29 Mayo 2023]. Disponible en: https://docs.arduino.cc/retired/getting-started-guides/ArduinoLilyPad
18. Velleman [Internet]. Placa de desarrollo WMW101 [Consultado 29 Mayo 2023]. Disponible en: https://www.velleman.eu/products/view/?id=460454&country=us&lang=es
19. Adafruit [Internet]. TinyPico – ESP32 Development Board – V2 [Consultado 29 Mayo 2023]. Disponible en: https://www.adafruit.com/product/4335
20. Adafruit [Internet]. Adafruit GEMMA v2 – Miniature wearable electronic platform [Consultado 29 Mayo 2023]. Disponible en: https://www.adafruit.com/product/1222
21. DFRobot [Internet]. Bluno Beetle BLE [Consultado 29 Mayo 2023]. Disponible en: https://www.dfrobot.com/product-1259.html
22. DFRobot [Internet]. Fermion: BMX160 9-axis Sensor (Breakout) [Consultado 29 Mayo 2023]. Disponible en: https://www.dfrobot.com/product-2141.html
23. Amazon [Internet]. RUNCCI-YUN 10 pzs CR2032 Soporte de bateria, 3V CR2032 Soporte de Bateria Pilas de Boton con Interruptor de Cable, Portapilas para Pilas Botón [Consultado 29 Mayo 2023]. Disponible en: https://n9.cl/159rl
24. DFRobot [Internet]. Wiki SKU:DFR0267 [Consultado 29 Mayo 2023]. Disponible en: https://wiki.dfrobot.com/Bluno_SKU_DFR0267#Configure_the_BLE_through_AT_command
25. E. K. Antonsson y R. W. Mann, «The frequency content of gait,» Journal of Biomechanics, vol. 18, nº 1, pp. 39-47, 1985.
26. Altium Designer [Internet]. [Consultado 30 Mayo 2023]. Disponible en: altium.com/es/
27. JLCPCB [Internet]. [Consultado 30 Mayo 2023]. Disponible en: https://jlcpcb.com/VGS?utm_source=gg_vgs&utm_medium=cpc&gclid=EAIaIQobChMIsMj32PKu9wIVxAIGAB0Y2wAZEAAYAiAAEgKJ1PD_BwE
28. Google Play [Internet]. Bluno Remote [Consultado 30 Mayo 2023]. Disponible en: https://play.google.com/store/apps/details?id=com.redrobe.robotremote&gl=ES
29. Apple [Internet]. iOS [Consultado 30 Mayo 2023]. Disponible en: https://www.apple.com/es/ios/ios-16/
30. Android [Internet]. [Consultado 30 Mayo 2023]. Disponible en: https://www.android.com/intl/es_es/
31. Amazon [Internet]. Cobee correa de reloj de nailon [Consultado 11 Junio 2023]. Disponible en: https://www.amazon.es/dp/B09TZZJM5K?psc=1&ref=ppx_yo2ov_dt_b_product_details
32. RedUSERS [Internet]. Microsoft estudia la producción de un smartwatch [Consultado 11 Junio 2023]. Disponible en: https://www.redusers.com/noticias/microsoft-estudia-la-produccion-de-un-smartwatch/

33. Clipset [Internet]. Galaxy Gear, el reloj inteligente de Samsung #IFA2013 [Consultado 11 Junio 2023]. Disponible en: https://clipset.com/galaxy-gear-el-reloj-inteligente-de-samsung-ifa2013/
34. Imprimakers [Internet]. [Consultado 11 Junio 2023]. Disponible en: https://imprimakers.com/es/buscador-de-modelos-3d/
35. Guía pádel [Internet]. Pádel: conceptos fundamentales [Consultado 11 Junio 2023]. Disponible en: https://guiapadel.com/padel-conceptos-fundamentales/
36. G.Cartes Domínguez [Internet]. Padel-Shot-Classification-and-Dataset [Consultado 13 Junio 2023]. Disponible en: https://github.com/guillecartes/Padel-Shot-Classification-and-Dataset

ANEXOS

Anexo I – Código prueba de comunicación con móvil

```
char caso = ' ';
#include <DFRobot_BMX160.h>

DFRobot_BMX160 bmx160;
void setup(){
  Serial.begin(115200);
  delay(100);

  //init the hardware bmx160
  if (bmx160.begin() != true){
    Serial.println("init false");
    while(1);
  }
  //bmx160.setLowPower();   //disable the gyroscope and accelerometer sensor
  //bmx160.wakeUp();        //enable the gyroscope and accelerometer sensor
  //bmx160.softReset();     //reset the sensor

  /**
   * enum{eGyroRange_2000DPS,
   *       eGyroRange_1000DPS,
   *       eGyroRange_500DPS,
   *       eGyroRange_250DPS,
   *       eGyroRange_125DPS
   *       }eGyroRange_t;
   **/
  //bmx160.setGyroRange(eGyroRange_500DPS);

  /**
   *  enum{eAccelRange_2G,
   *       eAccelRange_4G,
   *       eAccelRange_8G,
   *       eAccelRange_16G
   *       }eAccelRange_t;
   */
  //bmx160.setAccelRange(eAccelRange_4G);

  Serial.println("¿Muestras a tomar?");
  Serial.println("a5: 50");
  Serial.println("b1: 100");
  delay(100);
}

void loop(){
  if(Serial.available()){
    caso = Serial.read();
  }
  sBmx160SensorData_t Omagn, Ogyro, Oaccel;
  switch(caso){
```

```
49    case 'a':
50      for(int i = 1; i<51; i++){
51        bmx160.getAllData(&Omagn, &Ogyro, &Oaccel);
52        /* Display the magnetometer results (magn is magnetometer in uTesla) */
53        Serial.print("Magnetometer: ");
54        Serial.print('\n');
55        Serial.print("X:"); Serial.print('\t'); Serial.print(Omagn.x);
56        Serial.print('\t'); Serial.print("uT");
57        Serial.print('\n');
58        Serial.print("Y:"); Serial.print('\t'); Serial.print(Omagn.y);
59        Serial.print('\t'); Serial.print("uT");
60        Serial.print('\n');
61        Serial.print("Z:"); Serial.print('\t'); Serial.print(Omagn.z);
62        Serial.print('\t'); Serial.print("uT");
63        Serial.print('\n');
64        Serial.print('\n');
65        delay(100);
66
67        /*Display the gyroscope results (gyroscope data is in g) */
68        Serial.print("Gyroscope ");
69        Serial.print('\n');
70        Serial.print("X:"); Serial.print('\t'); Serial.print(Ogyro.x);
71        Serial.print('\t'); Serial.print("g");
72        Serial.print('\n');
73        Serial.print("Y:"); Serial.print('\t'); Serial.print(Ogyro.y);
74        Serial.print('\t'); Serial.print("g");
75        Serial.print('\n');
76        Serial.print("Z:"); Serial.print('\t'); Serial.print(Ogyro.z);
77        Serial.print('\t'); Serial.print("g");
78        Serial.print('\n');
79        Serial.print('\n');
80        delay(100);
81
82        /*Display the accelerometer results (accelerometer data is in m/s^2) */
83        Serial.print("Accelerometer");
84        Serial.print('\n');
85        Serial.print("X:"); Serial.print('\t'); Serial.print(Oaccel.x);
86        Serial.print('\t'); Serial.print("m/s^2");
87        Serial.print('\n');
88        Serial.print("Y:"); Serial.print('\t'); Serial.print(Oaccel.y);
89        Serial.print('\t'); Serial.print("m/s^2");
90        Serial.print('\n');
91        Serial.print("Z:"); Serial.print('\t'); Serial.print(Oaccel.z);
92        Serial.print('\t'); Serial.print("m/s^2");
93        Serial.print('\n');
94        Serial.print('\n');
95        delay(100);
96      }
97
98      case 'b':
99        for(int i = 1; i<101; i++){
100         bmx160.getAllData(&Omagn, &Ogyro, &Oaccel);
101         /* Display the magnetometer results (magn is magnetometer in uTesla) */
102         Serial.print("Magnetometer: ");
```

```
103           Serial.print('\n');
104           Serial.print("X:"); Serial.print('\t'); Serial.print(Omagn.x);
105           Serial.print('\t'); Serial.print("uT");
106           Serial.print('\n');
107           Serial.print("Y:"); Serial.print('\t'); Serial.print(Omagn.y);
108           Serial.print('\t'); Serial.print("uT");
109           Serial.print('\n');
110           Serial.print("Z:"); Serial.print('\t'); Serial.print(Omagn.z);
111           Serial.print('\t'); Serial.print("uT");
112           Serial.print('\n');
113           Serial.print('\n');
114           delay(100);
115
116           /*Display the gyroscope results (gyroscope data is in g) */
117           Serial.print("Gyroscope ");
118           Serial.print('\n');
119           Serial.print("X:"); Serial.print('\t'); Serial.print(Ogyro.x);
120           Serial.print('\t'); Serial.print("g");
121           Serial.print('\n');
122           Serial.print("Y:"); Serial.print('\t'); Serial.print(Ogyro.y);
123           Serial.print('\t'); Serial.print("g");
124           Serial.print('\n');
125           Serial.print("Z:"); Serial.print('\t'); Serial.print(Ogyro.z);
126           Serial.print('\t'); Serial.print("g");
127           Serial.print('\n');
128           Serial.print('\n');
129           delay(100);
130
131           /*Display the accelerometer results (accelerometer data is in m/s^2) */
132           Serial.print("Accelerometer");
133           Serial.print('\n');
134           Serial.print("X:"); Serial.print('\t'); Serial.print(Oaccel.x);
135           Serial.print('\t'); Serial.print("m/s^2");
136           Serial.print('\n');
137           Serial.print("Y:"); Serial.print('\t'); Serial.print(Oaccel.y);
138           Serial.print('\t'); Serial.print("m/s^2");
139           Serial.print('\n');
140           Serial.print("Z:"); Serial.print('\t'); Serial.print(Oaccel.z);
141           Serial.print('\t'); Serial.print("m/s^2");
142           Serial.print('\n');
143           Serial.print('\n');
144           delay(100);
145       }
146   }
147 }
```

Anexo II – Código prueba de comunicación cliente-servidor

```
1 #include <DFRobot_BMX160.h>
2
3 DFRobot_BMX160 bmx160;
4 void setup(){
5   Serial.begin(115200);
```

```
  delay(100);

  //init the hardware bmx160
  if (bmx160.begin() != true){
    Serial.println("init false");
    while(1);
  }
  //bmx160.setLowPower();   //disable the gyroscope and accelerometer sensor
  //bmx160.wakeUp();        //enable the gyroscope and accelerometer sensor
  //bmx160.softReset();     //reset the sensor

  /**
   * enum{eGyroRange_2000DPS,
   *       eGyroRange_1000DPS,
   *       eGyroRange_500DPS,
   *       eGyroRange_250DPS,
   *       eGyroRange_125DPS
   *       }eGyroRange_t;
   **/
  //bmx160.setGyroRange(eGyroRange_500DPS);

  /**
   *  enum{eAccelRange_2G,
   *       eAccelRange_4G,
   *       eAccelRange_8G,
   *       eAccelRange_16G
   *       }eAccelRange_t;
   */
  //bmx160.setAccelRange(eAccelRange_4G);
}

void loop(){
  long start = micros();
  if(Serial.available()){
  }
        sBmx160SensorData_t Omagn, Ogyro, Oaccel;
        bmx160.getAllData(&Omagn, &Ogyro, &Oaccel);

        /*Display the gyroscope results (gyroscope data is in g) */
        Serial.println("Gyroscope ");
        Serial.print("X: "); Serial.print(Ogyro.x); Serial.println(" g");
        Serial.print("Y: "); Serial.print(Ogyro.y); Serial.println(" g");
        Serial.print("Z: "); Serial.print(Ogyro.z); Serial.println(" g");
        delay(10);

        /*Display the accelerometer results (accelerometer data is in m/s^2) */
        Serial.println("Accelerometer");
        Serial.print("X: "); Serial.print(Oaccel.x); Serial.println(" m/s^2");
        Serial.print("Y: "); Serial.print(Oaccel.y); Serial.println(" m/s^2");
        Serial.print("Z: "); Serial.print(Oaccel.z); Serial.println(" m/s^2");
        delay(10);

        long duration = micros() - start;
        while (duration < 50000) {
```

```
          duration = micros() - start;
        }
        Serial.print("Duracion: "); Serial.println(duration);
}
```

Anexo III – Código final para la recogida y envío de datos

```
#include <DFRobot_BMX160.h>

DFRobot_BMX160 bmx160;
void setup(){
  Serial.begin(115200);
  delay(100);

  //init the hardware bmx160
  if (bmx160.begin() != true){
    Serial.println("init false");
    while(1);
  }
  //bmx160.setLowPower();   //disable the gyroscope and accelerometer
  //bmx160.wakeUp();        //enable the gyroscope and accelerometer sensor
  //bmx160.softReset();     //reset the sensor

  /**
   * enum{eGyroRange_2000DPS,
   *       eGyroRange_1000DPS,
   *       eGyroRange_500DPS,
   *       eGyroRange_250DPS,
   *       eGyroRange_125DPS
   *       }eGyroRange_t;
   **/
  //bmx160.setGyroRange(eGyroRange_500DPS);

  /**
   *  enum{eAccelRange_2G,
   *       eAccelRange_4G,
   *       eAccelRange_8G,
   *       eAccelRange_16G
   *       }eAccelRange_t;
   */
  //bmx160.setAccelRange(eAccelRange_4G);
}

void loop(){
  long start = micros();
  if(Serial.available()){
  }
        long duration = micros() - start;
        while (duration < 50000) {
          duration = micros() - start;
        }
        sBmx160SensorData_t Omagn, Ogyro, Oaccel;
        bmx160.getAllData(&Omagn, &Ogyro, &Oaccel);
        Serial.print("TIME: "); Serial.print(duration); Serial.print("; ");
```

```
48          Serial.print("Xg="); Serial.print(Ogyro.x); Serial.print("; ");
49          Serial.print("Yg="); Serial.print(Ogyro.y); Serial.print("; ");
50          Serial.print("Zg="); Serial.print(Ogyro.z); Serial.print("; ");
51          Serial.print("Xa="); Serial.print(Oaccel.x); Serial.print("; ");
52          Serial.print("Ya="); Serial.print(Oaccel.y); Serial.print("; ");
53          Serial.print("Za="); Serial.print(Oaccel.z); Serial.println(";");
54  }
```

Anexo IV – Código para el procesamiento de la información

```
 1 clear all
 2 close all
 3 clc
 4 %% SE PREGUNTAN LOS DATOS
 5 golpe = input('0.Fondo derecha 1.Fondo revés 2.Derecha con pared\n3.Revés con
 6 pared 4.Globo de derecha 5.Globo de revés\n6.Globo de derecha con pared 7.Globo de
 7 revés con pared\n8.Volea de derecha 9.Volea de revés 10.Bandeja 11.Remate
 8 12.Saque\n¿Qué golpe va a dar?:', 's');
 9 sexo = input('¿Hombre, Mujer o NC?:', 's');
10 nivel = input('1.Principiante 2.Amateur 3.Experimentado\n4.Muy alto nivel
11 5.Profesional\n¿Cuál es su nivel?:', 's');
12 mano = input('¿Diestro o Zurdo?:', 's');
13 reves = input('¿Con cuántas manos da el revés, 1 o 2?:', 's');
14 altura = input('Altura en cm:', 's');
15 edad = input('Edad en años:', 's');
16 ID = input('Introduzca su ID asignado:', 's');
17 disp('Ya puede encender el dispositivo y comenzar')
18 %% BORRAR PREVIOS
19 delete(instrfind({'Port'},{'COM5'}));
20
21 %% CREAR OBJETO CON EL PUERTO SERIE
22 s = serial('COM5','BaudRate',115200,'Terminator','CR/LF');
23 warning('off','MATLAB:serial:fscanf:unsuccessfulRead');
24
25 %% ABRIR PUERTO
26 fopen(s);
27
28 %% SE DEFINE UNA ESTRUCTURA
29 st = struct('tiempo', 0, 'ax', 0.0, 'ay', 0.0, 'az', 0.0, 'vx', 0.0, 'vy', 0.0,
30 'vz', 0.0);
31
32 %% Crear el array de memoria
33 array_previo = repmat(st, 1, 20);
34
35 %% Crear el array para después del golpe
36 array_posterior = repmat(st, 1, 20);
37
38 %% Se definen banderas y contadores
39 flag_first_sample=0;
40 flag_detection=0;
41 count=0;
```

```
Uv=50.0;
Uaxy=15.0;
Uaz=20.0;
nombre_archivo = 'prueba_final.csv';
cuentagolpes=1;
contadorgeneral=0;

%% Leer datos del puerto serie
while 1
    frase = fgets(s);
    %A partir de la segunda muestra
    if (flag_first_sample==1)
        %Se guardan los datos
        valores = sscanf(frase, 'TIME: %d; Xg=%f; Yg=%f; Zg=%f; Xa=%f; Ya=%f;
Za=%f;');
        contadorgeneral=contadorgeneral+1;
        % Si no estamos en mitad de un golpe se guardan las muestras
        if flag_detection == 0

            array_previo(1:19)=array_previo(2:20);

            array_previo(20).tiempo = valores(1);
            array_previo(20).ax = valores(5);
            array_previo(20).ay = valores(6);
            array_previo(20).az = valores(7);
            array_previo(20).vx = valores(2);
            array_previo(20).vy = valores(3);
            array_previo(20).vz = valores(4);

        % Comparación del umbral
            if (abs(valores(2)) > Uv || abs(valores(3)) > Uv || abs(valores(4)) >
Uv) && (abs(valores(5)) > Uaxy || abs(valores(6)) > Uaxy || abs(valores(7)) > Uaz)
                flag_detection=1;
            else
                flag_detection=0;
            end

        %Si se ha detectado golpe se guardan las siguientes muestras
        else
            array_posterior(count+1).tiempo = valores(1);
            array_posterior(count+1).ax = valores(5);
            array_posterior(count+1).ay = valores(6);
            array_posterior(count+1).az = valores(7);
            array_posterior(count+1).vx = valores(2);
            array_posterior(count+1).vy = valores(3);
            array_posterior(count+1).vz = valores(4);

            if (count<19)
                count=count+1;
            else
                vector1=[array_previo(:).ax array_posterior(:).ax];
                cadena1 = sprintf('[%s]', strjoin(string(vector1), ','));
                vector2=[array_previo(:).ay array_posterior(:).ay];
                cadena2 = sprintf('[%s]', strjoin(string(vector2), ','));
                vector3=[array_previo(:).az array_posterior(:).az];
                cadena3 = sprintf('[%s]', strjoin(string(vector3), ','));
```

```
 98                 vector4=[array_previo(:).vx array_posterior(:).vx];
 99                 cadena4 = sprintf('[%s]', strjoin(string(vector4), ','));
100                 vector5=[array_previo(:).vy array_posterior(:).vy];
101                 cadena5 = sprintf('[%s]', strjoin(string(vector5), ','));
102                 vector6=[array_previo(:).vz array_posterior(:).vz];
103                 cadena6 = sprintf('[%s]', strjoin(string(vector6), ','));
104 
105                 numgolpe=num2str(cuentagolpes);
106                 timegolpe=0.05*contadorgeneral;
107                 tgolpe=num2str(timegolpe);
108 
109                 matriz = cell(1,16);
110                 matriz{1,1} = cadena1;
111                 matriz{1,2} = cadena2;
112                 matriz{1,3} = cadena3;
113                 matriz{1,4} = cadena4;
114                 matriz{1,5} = cadena5;
115                 matriz{1,6} = cadena6;
116                 matriz{1,7} = golpe;
117                 matriz{1,8} = numgolpe;
118                 matriz{1,9} = tgolpe;
119                 matriz{1,10} = sexo;
120                 matriz{1,11} = nivel;
121                 matriz{1,12} = mano;
122                 matriz{1,13} = reves;
123                 matriz{1,14} = altura;
124                 matriz{1,15} = edad;
125                 matriz{1,16} = ID;
126 
127                 fid = fopen(nombre_archivo, 'a');
128                 fprintf(fid, '%s;%s;%s;%s;%s;%s;%s;%s;%s;%s;%s;%s;%s;%s;%s;%s\n',
129 matriz{1, :});
130                 fclose(fid);
131                 count=0;
132                 flag_detection=0;
133                 cuentagolpes=cuentagolpes+1;
134             end
135 
136 
137         end
138     % La  primera muestra no se guarda
139     else
140         frase = fgets(s);
141         flag_first_sample=1;
142     end
143     disp(frase)
144 end
145 
146 %% Cerrar el puerto serie
147 fclose(s);
```

Anexo V – Esquemático PCB

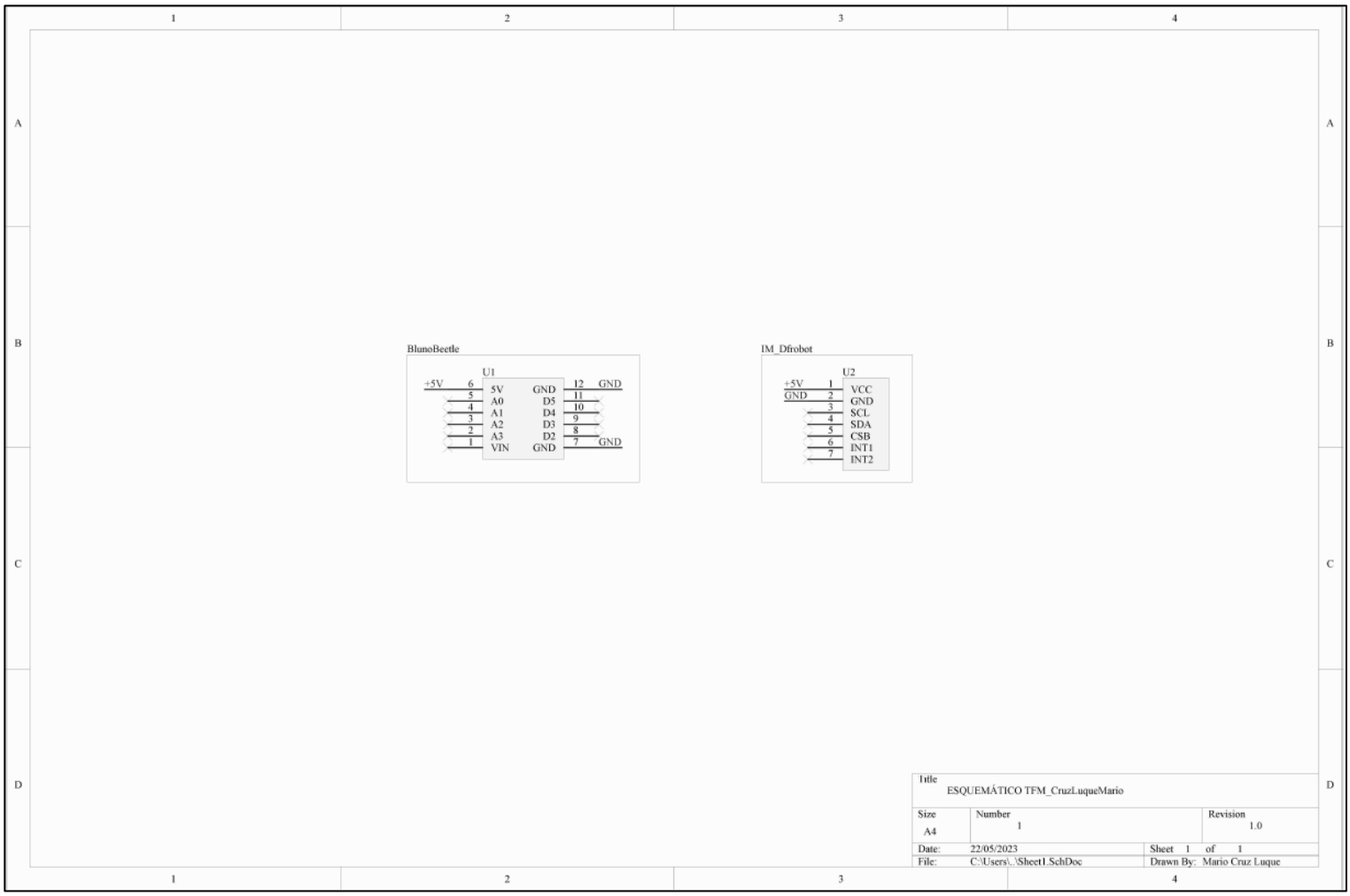

Figura 40. Esquemático de la PCB

Anexo VI – Manual de uso

Para empezar, la persona encargada de monitorizar la prueba, la cual puede ser el mismo deportista, podrá cambiar en el *script* de Matlab (línea 40) el nombre del archivo donde se guardarán todos los datos del experimento. A partir de la primera toma de datos, si este no se cambia se seguirán añadiendo en el mismo archivo las siguientes muestras sin sobrescribirse. También, el encargado deberá comprobar el puerto serie que ocupa el dispositivo que actúa como servidor en la red y cambiarlo en el código si fuera necesario (líneas 15 y 18).

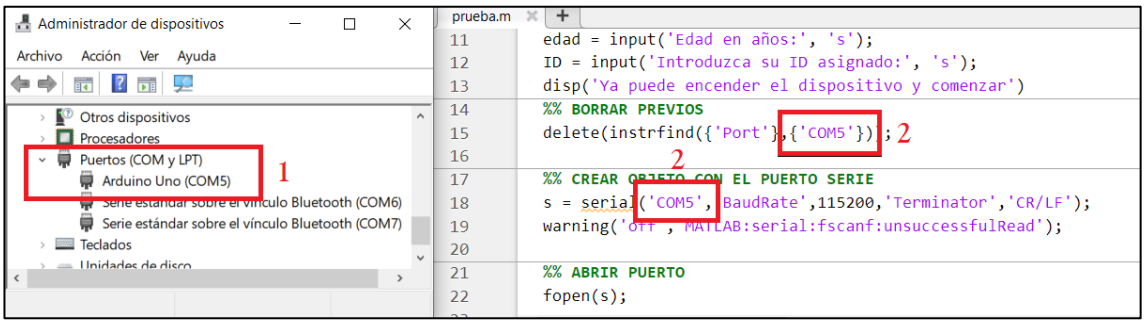

Figura 41. 1. Comprobación del puerto serie a utilizar. 2. Cambio de puerto serie

Una vez hecho esto, se debe dar al botón *RUN* para que se empiece a ejecutar el código, el cual imprimirá ciertas preguntas por pantalla que deberán ser respondidas por el deportista excepto la del ID, que será asignado por el encargado. Cuando se haya respondido a estas aparecerá un mensaje indicando que el dispositivo puede encenderse una vez se esté preparado.

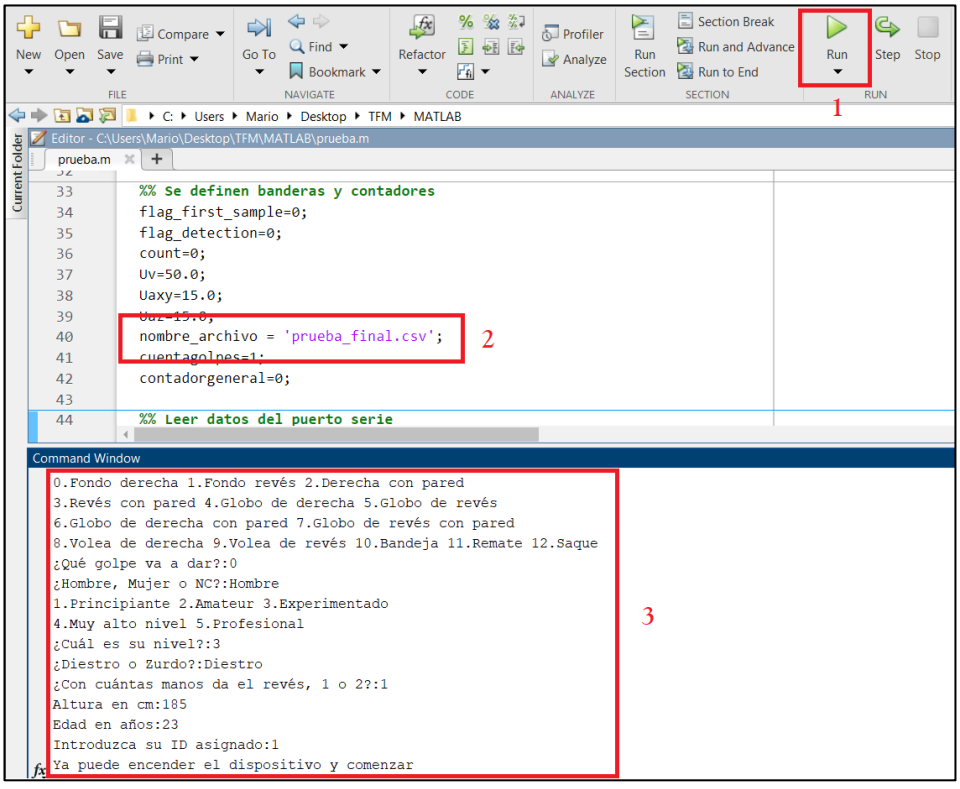

Figura 42. 1. Botón de inicio de código. 2. Cambio de nombre de archivo. 3. Mensajes impresos por pantalla

Acto seguido el deportista deberá encender el dispositivo a través del interruptor situado en la carcasa de las pilas, en la parte inferior del *wearable*. En este momento se deberá permanecer en posición de espera hasta que la comunicación Bluetooth se establezca y se empiecen a observar datos imprimiéndose por pantalla en Matlab.

Figura 43. Posición de espera en pádel [35]

Es en este momento cuando empieza el experimento y se procede a golpear la bola de la manera acordada. El código funciona de tal manera que el entrenamiento puede extenderse en tiempo hasta cuando se desee, por lo que cuando se quiera dar por finalizado habrá dos maneras. Si solamente hay una persona realizando la toma de datos con apagar el dispositivo con el interruptor de las baterías previamente mencionado bastará, ya que cuando Matlab deje de detectar datos por puerto serie parará el código de forma automática imprimiendo por pantalla un error insignificante. Por otra parte, si hay una persona a cargo del ordenador, el experimento se parará desde el mismo programa. Finalmente, si se ha decidido continuar el archivo de toma de datos se incluirán las muestras, o si se ha optado por crear uno nuevo aparecerá en la misma carpeta donde se encuentre el código.

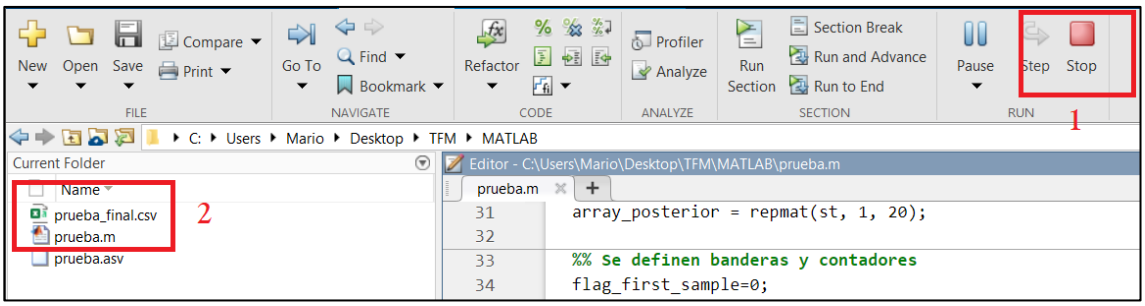

Figura 44. 1. Botón de paro de código. 2. Directorio de almacenamiento