UCUENCA

Universidad de Cuenca

Facultad de Ingeniería

Carrera de Electrónica y Telecomunicaciones

Desarrollo de un sistema IoT para la adquisición de datos biomecánicos de un patinador de velocidad, utilizando dispositivos ESP32

Trabajo de titulación previo a la obtención del título de Ingeniero en Electrónica y Telecomunicaciones

Autores:

Miguel Alejandro Beltrán Saavedra

Freddy Medardo Tacuri Merchán

Director:

Darwin Fabián Astudillo Salinas

ORCID: 0000-0001-7644-0270

Cuenca, Ecuador

2023-05-24

Resumen

En los deportes de competición, considerar pequeños detalles en la técnica puede significar mejores resultados. Actualmente, los estudios que relacionan el patinaje de velocidad con *Internet of Things* (IoT) y los dispositivos *Wearables* son limitados, siendo la experiencia de atletas de élite o entrenadores la principal referencia de los patinadores al momento de mejorar sus técnicas. Es por ello, que la incorporación de sistemas de medición con sensores inerciales que describan los movimientos del deportista de manera cuantitativa, daría paso a entrenamientos más controlados y eficientes. En el presente proyecto de titulación, se desarrolla un sistema IoT *Wireless* que incorpora módulos ESP32 y sensores inerciales para la adquisición de datos de aceleración de las cuatro extremidades de un patinador de velocidad; dichos datos se almacenan en el servidor Web *Thingsboard Cloud*, utilizando el protocolo *MQ Telemetry Transport* (MQTT). El sistema de medición tiene una arquitectura tipo estrella que consta de un nodo central y cuatro nodos sensores, mismos que se comunican mediante el protocolo ESP-NOW. Para comandar la captura de datos en los nodos sensores y visualizar los datos inerciales, se desarrolla una aplicación móvil para *Android*. Para verificar el funcionamiento del sistema de adquisición, se realizaron pruebas en tres patinadores con distintos niveles de experiencia y bajo las mismas condiciones, pudiendo caracterizar los datos de aceleración de las cuatro extremidades en los tres ejes cartesianos, tanto para un tramo recto como curvo de la pista.

Palabras clave: patinaje de velocidad, IoT y tecnología *Wearable*, módulos ESP32, ESP-NOW, Thingsboard

Abstract

In sports of competition, considering small details in technique can mean better results. Currently, studies that relate speed skating with IoT and wearable devices are limited, being the experience of elite athletes or coaches the main reference for skaters when improving their techniques. Therefore, the incorporation of measurement systems with inertial sensors that describe the athlete's movements in a quantitative way would lead to more controlled and efficient training. In this project an IoT Wireless system is developed that incorporates ESP32 modules and inertial sensors for the acquisition of acceleration data from the four extremities of a speed skater; these data are stored in the Thingsboard Cloud Web server, using the MQTT protocol. The measurement system has a star-like architecture consisting of a central node and four sensor nodes, which communicate using the ESP-NOW protocol. To command the data capture at the sensor nodes and visualize the inertial data, a mobile application for Android is developed. To verify the operation of the acquisition system, tests were performed on three skaters with different levels of experience and under the same conditions, being able to characterize the acceleration data of the four extremities in the three cartesian axes, both for a straight and curved section of the track.

Keywords: speed skating, IoT and Wearable technology, ESP32 modules, ESP-NOW, Thingsboard

Índice de contenidos

Índice de figuras

Índice de tablas

Índice de extractos de código

Dedicatoria

A mis padres, Miguel y Luisa, por confiar en mí y apoyarme a continuar mi carrera universitaria sin importar los obstáculos presentados; este logro es también para ellos.

A mis hermanos, pues me han apoyado directa e indirectamente a continuar mis estudios.

A Michelle, porque su apoyo y cariño han sido incondicionales en todo momento. Es un motivo para seguir adelante.

A mi querida Sammy, ese ser auténtico que cambió mi forma de ver la vida para siempre.

Miguel Beltrán S.

Dedicatoria

A mis padres, Miguel y Elvia, por el apoyo incondicional a lo largo de mi carrera universitaria. Su sacrificio es la inspiración y motivación para seguir adelante frente los obstáculos.

A todos mis familiares, quienes me brindaron sus constantes consejos y palabras de aliento. Más aún, a mis abuelos: Lucrecia, Ernesto (+), Isaura y Luis (+), quienes con su sencillez me enseñaron que todo esfuerzo tendrá sus frutos tarde o temprano. También resalto a mis tíos quienes fueron un pilar importante para los logros alcanzados.

A mis amigos, que me acompañaron durante mi etapa académica. En especial a Paola, quien fue un gran soporte desde que apareció en mi vida universitaria.

Freddy Tacuri M.

Agradecimientos

A nuestro director de tesis, Ing. Darwin Fabián Astudillo Salinas, quien con compromiso nos brindó su tiempo y conocimiento durante el desarrollo de este trabajo de titulación. Un agradecimiento también a la Facultad de Ingeniería de la Universidad de Cuenca y a sus docentes, por la gran formación académica brindada. Varios de aquellos docentes han sido una inspiración para nuestra vida profesional.

A nuestros amigos y compañeros, a quienes hemos conocido a lo largo de nuestra vida universitaria y con quienes seguiremos compartiendo en un futuro. Un agradecimiento especial a Fabricio y Felipe por el apoyo y los consejos, sus aportes fueron importantes para cumplir este objetivo.

LOS AUTORES

Abreviaciones y Acrónimos

ADC *Analog-to-Digital converter*. 36

AP *Access Point*. 29, 50, 92

API Interfaz de programación de aplicaciones. 32, 33, 60, 93, 94

CCMP *Counter Mode Cipher Block Chaining Message Authentication Code Protocol*. 28

CoAP Protocolo de aplicación restringida. 32

CPU Unidad central de procesamiento. 26

DB *Data Base*. 56

DLPF *Digital low pass filter*. 35

EID Extremidad inferior derecha. 70–72, 90, 92

EII Extremidad inferior izquierda. 68, 70, 71, 90, 92

EMG *Electromiografía*. 76

ESD Extremidad superior derecha. 70, 71, 90, 92

ESI Extremidad superior izquierda. 68, 70, 71, 90, 92

GPIO *General Porpouse Input Output*. 26

GPS Sistema de Posicionamiento Global. 17

HTTP Protocolo de transferencia de hipertexto. 32, 93, 94

I2C *Inter-Integrated Circuit*. 34

IDE Entorno de desarrollo integrado. 49

IHM *Interacción humano-máquina*. 23

IoT *Internet of Things*. 2, 3, 18, 20–23, 26, 30, 32, 37–40, 54, 87, 88

JWT *JSON Web Token*. 61, 94–96

Li-Po *Litio y polímero*. 41–46, 88

LwM2M *Lightweight Machine to Machine*. 32

M2M *Machine-to-Machine*. 30

MAC *Media Access Control*. 27, 50, 89

MEMS sistemas microelectromecánicos. 37

MPU *Multiple Process Unit*. 53, 76

MQTT *MQ Telemetry Transport*. 2, 3, 20, 30–32, 36, 37, 39–41, 54, 67, 88, 89, 92

NTP *Network Time Protocol*. 52

OMS Organización Mundial de la Salud. 17

OTA *Over The Air*. 89

PCB *Printed circuit board*. 6, 18, 41–44, 46, 47, 87, 89

RAM *Random Access Memory*. 26

ROM *Read Only Memory*. 27

RPC Llamada a procedimiento remoto. 6, 7, 10, 32, 33, 51, 54, 56–58, 60, 61, 88, 91–93

RSSI *Received Signal Strength Indicator*. 30

RTC *Real Time Clock*. 27

SDK *Software development Kit*. 59

SMD Dispositivo de montaje superficial. 43

SMP *Multiprocesamiento simétrico*. 49

SPI *Serial Peripheral Interface*. 35

SQL Lenguaje de consulta estructurada. 34

SRAM *Static Random Access Memory*. 27, 35, 42

TCP *Protocolo de Control de Transmisión*. 93

TTL *Time to live*. 56

UDP *User datagram protocol*. 50

UI Interfaz de usuario. 33

URL *Uniform Resource Locator*. 94

Introducción

Este capítulo presenta la identificación del problema, justificación, alcance y los objetivos del presente proyecto de titulación.

1.1. Identificación del problema

Según la Organización Mundial de la Salud (Organización Mundial de la Salud (OMS)), los movimientos generados por los músculos esqueléticos se definen como una actividad física; y un deporte consiste en la práctica de alguna actividad física [8, 9]. Dicha actividad en general, es indispensable para combatir el sedentarismo y tener una mejor calidad de vida. Varias publicaciones explican que los deportes no solo brindan beneficios físicos o emocionales, sino también repercuten en el ámbito económico [10, 11].

En las disciplinas deportivas prima la competitividad, por lo que cualquier mínimo detalle, marca una gran diferencia [12]. La evolución de la exigencia en el deporte profesional, ha sido un disparador para incorporar nuevas ciencias y tecnologías aplicadas a la evaluación del rendimiento deportivo [13]. Estas permitirían obtener datos más precisos que contribuyan a la mejora del rendimiento deportivo ya sea de manera individual o colectiva [14].

Uno de los deportes que va sumando seguidores en países del primer mundo, como Estados Unidos, es el patinaje. A nivel de Latinoamérica, Colombia es uno de los países más representativos. En el caso de Ecuador, la práctica del patinaje deportivo va en aumento [15]. A pesar de ser un deporte en crecimiento, se han realizado escasos estudios cuantitativos de la ejecución técnica de los movimientos de estos atletas; la medición y caracterización de los movimientos dentro de este deporte permiten evaluar el rendimiento de los deportistas para mejorar sus técnicas o evitar lesiones [10].

1.2. Justificación

En el ámbito deportivo, la utilización de dispositivos inteligentes y *wearables* equipados con sensores como Sistema de Posicionamiento Global (GPS), acelerómetros y giroscopios, junto con el análisis de los datos que estos generan, ofrecen a entrenadores y atletas datos cuantitativos que valoran sus progresos [12, 16]. El uso de estas tecnologías aportan una ventaja competitiva extra en los deportistas que utilizan estas herramientas, con respecto de aquellos

que no evalúan sus técnicas. Partiendo de este enfoque, la implementación del *Internet of Things* (IoT) permite métodos de entrenamiento más controlados y eficientes. De esta manera, los diferentes actores vinculados al deporte pueden mejorar sus capacidades de decisión y, en consecuencia, los resultados [12].

El uso de dispositivos IoT en la actualidad abarca multitud de campos como la logística, el ámbito sanitario, el desarrollo de ciudades inteligentes, la gestión de residuos, el sector educativo, o el deporte [17]. A pesar de la escasez de chips y el impacto del COVID-19 en la cadena de suministro en los últimos dos años, el mercado de IoT continúa creciendo. En 2021 se estima que el número global de dispositivos IoT conectados creció un 9 % respecto al 2020, a 12.300 millones de puntos finales activos. Para 2025, es probable que haya más de 27 mil millones de conexiones de IoT [1].

Las exigencias en las distintas disciplinas deportivas son cada vez mayores, por lo que se requiere incorporar nuevas herramientas y tecnologías. La integración de sensores inerciales, en un dispositivo IoT aplicado al patinaje, permite la adquisición de datos de los movimientos de los patinadores. Los movimientos adquiridos pueden ser analizados en lo posterior, con el fin de mejorar la técnica y el rendimiento del deportista [18, 19]. Cabe mencionar que este deporte puede ser caracterizado por los movimientos de, al menos, las 4 extremidades [20].

1.3. Alcance

En la presente propuesta de trabajo de titulación, se desarrollará un prototipo de un sistema IoT para la adquisición de datos biomecánicos de un patinador de velocidad, utilizando módulos ESP32 y acelerómetros. Los nodos sensores serán colocados en las cuatro extremidades y trabajarán de manera sincronizada en el proceso de captura y almacenamiento de los datos de aceleración. Para cada una de las extremidades, se desarrollará una *Printed circuit board* (PCB) prototipo que integre el módulo ESP32, un acelerómetro y una tarjeta MicroSD, alimentados con una batería.

Dentro del sistema propuesto, se utilizará un quinto nodo colocado en el torso del deportista, el cual se encargará de sincronizar la captura de datos en los distintos nodos. Posteriormente, los datos adquiridos se enviarán a un servidor web, desde cada nodo directamente. La información del servidor podrá ser consultada mediante una aplicación desarrollada para Android. El alcance del presente proyecto no cubre el análisis e interpretación de los datos de aceleración almacenados en el servidor web desde el punto de vista deportivo.

1.4. Objetivos

1.4.1. Objetivo general

Diseñar e implementar un sistema IoT para la adquisición de datos biomecánicos de un patinador de velocidad, utilizando dispositivos ESP32.

1.4.2. Objetivos específicos

El presente trabajo tiene los siguientes objetivos específicos:

- Revisar el estado del arte relacionado con el uso de dispositivos ESP32 en el deporte.
- Diseñar e implementar un prototipo *wearable* para la toma de datos de aceleración, con al menos un nodo por extremidad.
- Implementar un mecanismo de comunicación y sincronización entre los dispositivos ESP32.
- Desarrollar una aplicación móvil para mostrar los datos almacenados en el servidor web.

Marco teórico

En este capítulo se realiza una revisión teórica de los conceptos más relevantes relacionados con el patinaje de velocidad y su biomecánica, así como la importancia de las mediciones cuantitativas dentro de este deporte. También, se aborda información acerca de la relevancia que tiene tanto el IoT como los *Wearables* en el mundo moderno (Sección 2.2) y como su uso favorece al ser humano en distintos ámbitos como en el deporte (Sección 2.2.7). Además, se recalca la utilidad que tienen los microcontroladores, como el ESP32 (Sección 2.3), dentro de las aplicaciones IoT. Por su parte, en la sección 2.4, se mencionan algunos protocolos de comunicación enfocados a ESP32, los cuales son muy utilizados en las redes inalámbricas de sensores. Se abordan algunos puntos relevantes sobre el protocolo de transporte *MQ Telemetry Transport* (MQTT) (Sección 2.5) y las facilidades que brinda la plataforma *Thingsboard* (Sección 2.6) para la gestión de la información censada en determinados nodos. La relevancia del uso de sensores inerciales para la adquisición de datos en los dispositivos IoT también es mencionado. Del mismo modo, la incorporación de una memoria externa, como una tarjeta microSD, para expandir la capacidad de almacenamiento es un punto importante a ser expuesto. Finalmente, se revisan varios artículos científicos relacionados con el uso de dispositivos ESP32 en el deporte (Sección 2.9).

2.1. Patinaje de velocidad

El patinaje de velocidad es un deporte que ha venido desarrollándose a lo largo del tiempo y va sumando seguidores. Esta disciplina deportiva abarca varias pruebas, ya sea a corta o larga distancia, individuales o en grupo, en pista o en circuito [21, 22]. Por la ejecución de los movimientos, este deporte es considerado de tipo cíclico y demanda una gran preparación física y mental. Debido a la variedad de pruebas que se manejan, se requieren ritmos constantes de oxígenación y explosividad, por lo que es considerado un deporte aeróbico y anaeróbico [23, 21].

El patinaje consiste en la incorporación de patines en los pies, y así un individuo puede deslizarse sobre una superficie plana [24]. El patinador velocista sustenta sus movimientos en: la acción inicial, la velocidad en línea recta y la destreza que se tenga en las curvas. Este deporte es potenciado mediante técnicas como: análisis de la puntualidad de los empujes, la velocidad y aceleración absolutas, el seguimiento fisiológico, entre otros. La medición y caracterización de los movimientos dentro de este deporte permiten evaluar el rendimiento de los atletas para mejorar sus técnicas de ejecución o evitar lesiones [10]. Cabe mencionar que este deporte

puede ser caracterizado por los movimientos de al menos las 4 extremidades [20].

Los parámetros de caracterización cuantitativos pueden ser de tipo cinemático o cinético. En este contexto, el movimiento cinemático se refiere a un movimiento determinado, la posición del cuerpo y su evolución. Mientras que el movimiento cinético se asocia a las fuerzas que producen el movimiento. El análisis cuantitativo de la técnica de patinaje basada en la adquisición de datos biomecánicos, permite una descripción objetiva de un gesto técnico, la determinación del rendimiento, la comparación entre datos recopilados, la visualización de errores y la objetividad de las características cualitativas [22].

No obstante, el estudio actual del patinaje de velocidad involucra escasa información cuantitativa y de poco valor científico. Por lo cual, los distintos modelos de entrenamiento están sustentados en el error y la corrección; así como también en la observación de grandes patinadores o la experiencia de entrenadores que han sido anteriormente patinadores. El hecho de no haberse podido confirmar dichas suposiciones teóricas con un análisis cuantitativo minucioso ha generado gran controversia y un sinnúmero de opiniones acerca de las técnicas de patinaje, marginando en muchos casos a gran cantidad de competidores debido a su formación técnica [22].

2.2. IoT y la tecnología *Wearable*

El Internet de las cosas o *Internet of Things* (IoT) permite que objetos, sensores y elementos cotidianos generen, intercambien y consuman datos [12]. Esto brinda la posibilidad de aprovechar la información recolectada, y a su vez, que los sistemas puedan reaccionar de forma autónoma a determinados eventos y cambios [25].

Este conjunto de tecnologías supone “la evolución de Internet desde una red de ordenadores interconectados hasta una red de objetos interconectados”, formando parte de uno de los agentes de la llamada “cuarta revolución industrial”. El hecho de que IoT sea sensorial (que procese variables como temperatura, humedad, estrés, etc.) y ubicua (que esté presente en todas partes gracias a Internet y las distintas tecnologías inalámbricas), permite que la implementación masiva de estas tecnologías sea factible [25].

2.2.1. Arquitectura de IoT

La arquitectura de los sistemas IoT puede ser dividida en cuatro capas fundamentales [26], mismas que se describen brevemente en la Tabla 2.1.

Tabla 2.1: Arquitectura IoT de cuatro capas

Capa	Descripción
Detección	Constituida por los sensores, los objetos físicos y la obtención de datos.
Intercambio de datos	Corresponde a la transmisión transparente de los datos a través de redes de comunicación.
Integración de la información	Procesamiento de la información adquirida, filtrado de datos no deseados e integración de la información en conocimiento útil para los servicios y los usuarios finales.
Servicio de aplicación	Brinda servicios de contenido a los usuarios.

2.2.2. El mercado de IoT

El uso de dispositivos IoT en la actualidad puede ser aplicado en multitud de campos [17]. De acuerdo a la segunda edición del informe *Things Matter*, elaborado por Telefónica en 2019, el uso de dispositivos IoT y sus aplicaciones aumentó un 66 % entre el año 2017 y 2019, con un 87 % de usuarios que declaran que, una vez probados dichos dispositivos, ya no renuncian a sus beneficios [27].

A pesar de la escasez de chips y el impacto del COVID-19 en la cadena de suministro en los últimos dos años, el mercado de IoT continúa creciendo. En 2021 se estima que el número global de dispositivos IoT conectados ha crecido un 9 % respecto al 2020, a 12.300 millones de puntos finales activos. Para 2025, es probable que haya más de 27 mil millones de conexiones de IoT [1]. En la Figura 2.1 se observa el pronóstico del mercado global de IoT de acuerdo con los últimos informes presentados en la página de *IoT Analítics*.

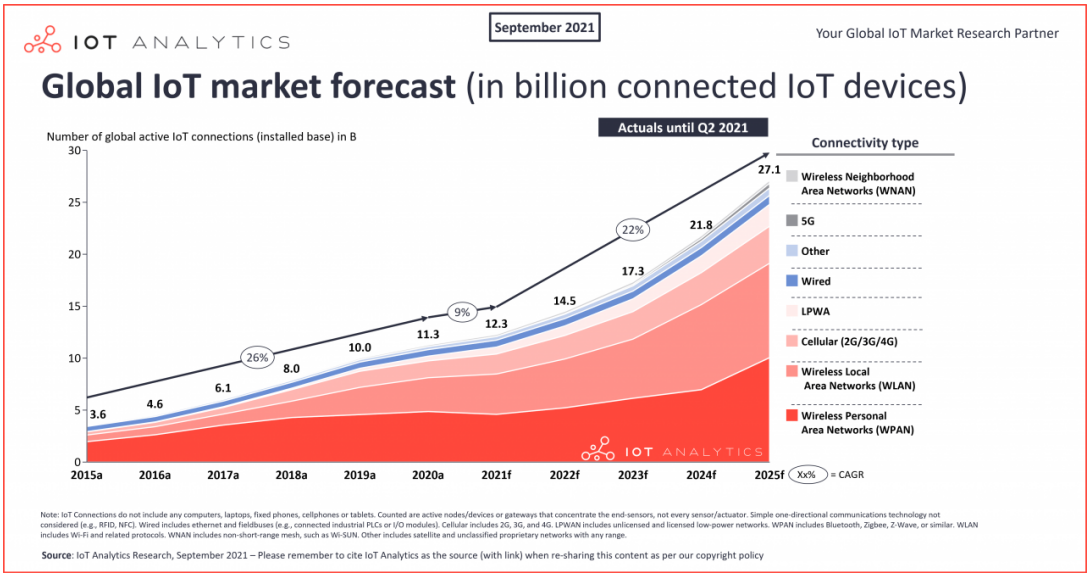


Figura 2.1: Pronóstico del mercado global de IoT [1]

2.2.3. Aplicaciones de IoT

IoT permite un número ilimitado de aplicaciones y servicios, pudiéndose adaptar a diversos campos de la actividad humana y brindar mayor confort[26]. En efecto, los usos de IoT se encuentran en la gran mayoría de sectores industriales [25]. A continuación, se enumeran los campos de aplicaciones y servicios basados en IoT; sin embargo, este es solo un limitado conjunto del basto abanico de posibilidades que el IoT puede proporcionar [26]:

1. *Smart cities* y transporte.
2. Electrónica de consumo.
3. Educación.
4. Casas y edificios inteligentes.
5. Agricultura y medio ambiente.
6. *Smart cars*.
7. Salud y deportes.

2.2.4. Tecnología *Wearable*

La llegada de Internet y otras tecnologías han permitido que hoy en día sea posible la comunicación entre los elementos de vestir y las bases de datos públicas o privadas. Esta comunicación es una forma de establecer una *Interacción humano-máquina* (IHM).

El concepto de *Wearable* puede ser traducido desde el inglés como “llevable” o “usable”. Esta tecnología hace referencia a dispositivos electrónicos como microprocesadores, sensores y transductores diseñados para ser vestidos, siendo un complemento en la ropa o algún accesorio y que permiten obtener información de manera continua [28].

Las características más importantes de la tecnología *Wearable* son la capacidad de una conexión inalámbrica y el propósito de acceder a la electrónica de una manera fácil, transparente y constante; lo que permite que el usuario pueda obtener la información incluso en tiempo real [28].

2.2.5. Atributos de la tecnología *Wearable*

En 1977, Steve Mann planteó ocho atributos que debe cumplir un dispositivo para ser considerado *Wearable* [28]:

- **Constante:** El dispositivo debe estar activo la mayor parte del tiempo de interés.
- **Sin restricciones para el usuario:** El dispositivo no debe interferir en las diferentes actividades que el usuario pueda estar realizando.
- **No monopoliza la atención del usuario:** El uso del dispositivo no debe aislar al usuario del mundo exterior.
- **Observable por el usuario:** Se puede configurar para generar alertas y notificaciones al usuario.
- **Controlable por el usuario:** El usuario puede controlar el dispositivo de manera total cuando así lo desee.
- **Atento al ambiente:** Es amigable con el entorno.
- **Comunicativo:** Puede ser usado como medio de comunicación, si así se lo desea.
- **Personal:** Debe ser inseparable del cuerpo y puede ser controlado por una persona diferente al dueño cuando este último lo autorice.

2.2.6. El mercado de los *Wearables* y la tecnología portátil

En el mercado existe gran variedad de dispositivos *Wearables*, como las *Google Glass* o los relojes Fitbit [28]. Hasta hace poco, la tecnología *Wearable* se había limitado a dispositivos usados en la muñeca, como las pulseras inteligentes, mismos que en el año 2018 representaron alrededor del 60 % del gasto del consumidor en este sector. Sin embargo, en los años posteriores han ganado popularidad otros dispositivos como los auriculares inteligentes [29]. Del mismo modo, la creciente prevalencia de enfermedades crónicas y obesidad han contribuido al uso de dispositivos de monitoreo corporal con información en tiempo real [30].

El número de dispositivos portátiles conectados en el mundo se duplicó entre el año 2016 y el año 2019, pasando de 325 millones a 722 millones. Se pronostica que el número de dispositivos conectados supere los mil millones en el presente año 2022, tal como se indica en la Figura 2.2 [2].

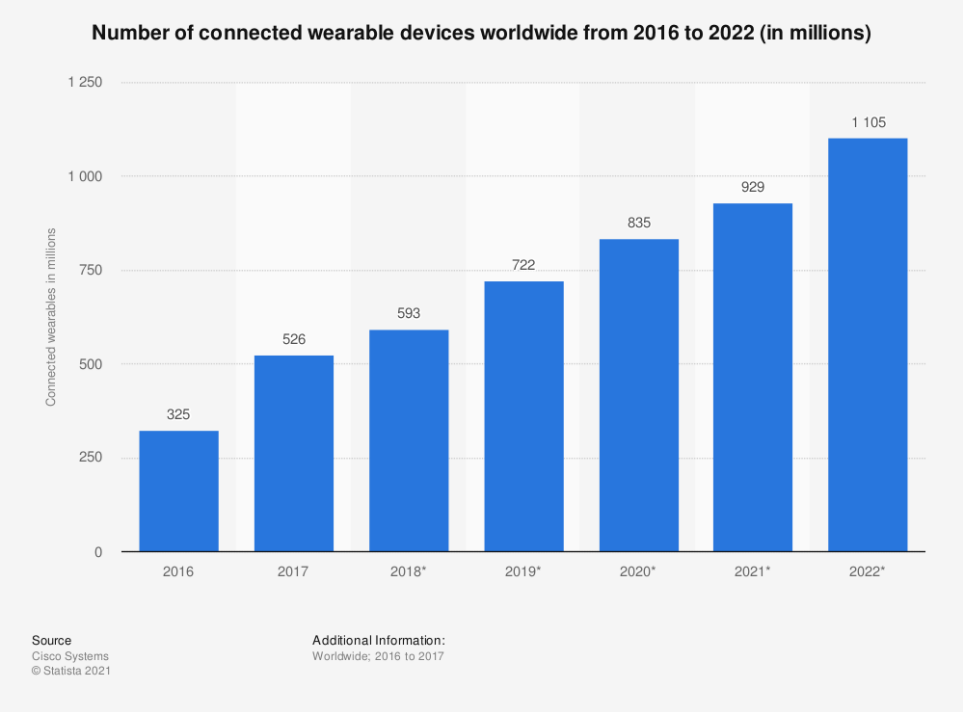


Figura 2.2: Dispositivos portátiles conectados en todo el mundo entre 2016 y 2022 [2]

2.2.7. *Wearables* en el deporte

Hace pocos años, se pensaba en la tecnología *Wearable* como una idea del futuro. Sin embargo, esta tecnología ha avanzado rápidamente permitiendo la integración de múltiples disciplinas [28, 31]. En el deporte, muchas personas utilizan diferentes *Wearables* como complemento dentro de sus entrenamientos. El mercado de dispositivos enfocados al deporte ha aumentado sus ventas debido a que nuevos fabricantes ofrecen innovadores accesorios para acompañar la práctica deportiva. Dichos dispositivos permiten llevar un mayor control y seguimiento del ejercicio físico y son un componente motivador para fijar metas de una manera simple y atractiva [31, 32].

Actualmente, los *Wearables* son la fuente más fiable y precisa que se tiene para el control sistemático de las actividades deportivas. Esta tecnología no solo permite tener el control sobre la actividad física, sino que también brinda soporte y seguridad personal. Se infiere que, la tecnología vestible y portátil es de importancia fundamental para la mejora física de un deportista, permitiendo la corrección de posturas y técnicas a través del estudio de la biomecánica del cuerpo. Por otro lado, según un estudio realizado en 2016, en [32] se menciona que la tecnología portátil incita a niños y adolescentes a la práctica de alguna actividad deportiva, permitiendo una formación en la cultura física a temprana edad y combatiendo a largo plazo el índice de población sedentaria.

2.3. Microcontrolador

En el corazón de los dispositivos IoT se encuentran los microcontroladores, los cuales están pensados para interactuar con el medio ambiente a través de sensores y actuadores, así como realizar tareas de control. Estos semiconductores contienen entre otras cosas, Unidad central de procesamiento (CPU), *Random Access Memory* (RAM) y memoria persistente, pines *General Porpouse Input Output* (GPIO), interfaces de comunicación y toda la electrónica necesaria para que funcione [33, 34].

Entre los aspectos a considerar al momento de seleccionar un microcontrolador se encuentra la compatibilidad de *Hardware*, la capacidad de procesamiento y el tamaño de memorias requerido, el consumo de energía y la documentación disponible para el aprendizaje. La elección del microcontrolador no debe requerir modificaciones importantes del entorno donde se vaya a instalar [33].

El microcontrolador ESP32 es un chip versátil y de bajo consumo con doble núcleo, desarrollado por Espressif Systems, con una capacidad computacional de hasta 240Mhz y opciones de conectividad Wi-Fi en la banda de 2.4Ghz y Bluetooth v4.2. Esta capacidad de procesamiento y conectividad, junto con la gran cantidad de contenido Open Source disponible, permite crear prototipos IoT de manera rápida y económica. Debido a sus características, ESP32 es cada vez más popular en proyectos de IoT [35, 36].

La placa ESP32 Wemos D1 mini de la Figura 2.3, posee las mismas características que la placa de desarrollo ESP32, incluyendo en su interior el mismo chip ESP32-WROOM-32, pero con la particularidad de tener un menor tamaño [3]. De acuerdo con [37], esta placa integra una memoria flash de una capacidad de 4Mb.

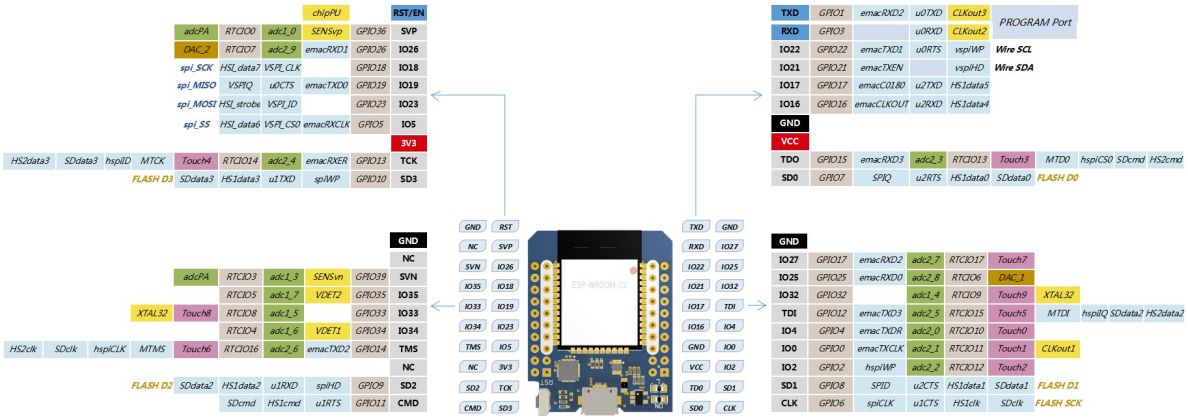


Figura 2.3: Pinout del ESP32 Wemos D1 mini [3]

De acuerdo con [38], el chip de ESP32-WROOM-32 posee las siguientes características:

- *Read Only Memory* (ROM): 448KB orientados al inicio del programa y el llamado a la función del kernel.
- *Static Random Access Memory* (SRAM): 520KB destinados al almacenamiento de datos e instrucciones.
- Memoria rápida *Real Time Clock* (RTC): 8KB de la SRAM pueden ser utilizados cuando el ESP32 se encuentre en modo de suspensión profunda.
- Memoria lenta RTC: 8KB de la SRAM pueden ser utilizados cuando el ESP32 se encuentre en modo de suspensión profunda.
- 1 Kbit eFuse: 256 bits están destinados a la dirección *Media Access Control* (MAC) y configuraciones del chip; y 768 bits están orientados a los programas.

2.4. Redes inalámbricas de sensores

El uso de dispositivos inteligentes es una tendencia cada vez mayor, por lo que muchas tecnologías se enfocan en conectar dispositivos entre sí, utilizándose por ejemplo una red tipo estrella o tipo *Mesh*. Cualquier solución de red pretende ser autoorganizada (la red forma su conexión sin configuración previa) y autorreparable (es posible agregar nodos a la red existente y esta se reorganizará en caso de fallas de los nodos) [5]. En relación con dispositivos ESP32, se tienen soluciones como ESP-NOW, ESP-WIFI-MESH Y PainlessMesh para crear redes WiFi entre dichos dispositivos, estas alternativas se describen brevemente a continuación.

2.4.1. ESP-NOW

ESP-NOW es un protocolo de comunicación inalámbrica Wi-Fi basado en la capa de enlace de datos y cuya característica es reducir cinco capas del modelo OSI a solo una. Es decir, los datos no necesitan transmitirse a través de la capa de red, de transporte, de sesión, de presentación y de aplicación. Además, no se requieren encabezados de paquetes o desempaquetadores en cada capa, permitiendo una respuesta rápida y reduciendo el retraso causado por la pérdida de paquetes en redes congestionadas [39]. En la Figura 2.4 se expone las diferencias entre el modelo OSI y el modelo ESP-NOW, también se indican los diferentes protocolos y codificaciones soportadas en cada una de las capas de los dos modelos [4].

ESP-NOW aplica la tecnología descrita en *IEEE Std. 802.11-2012*, junto con la función IE

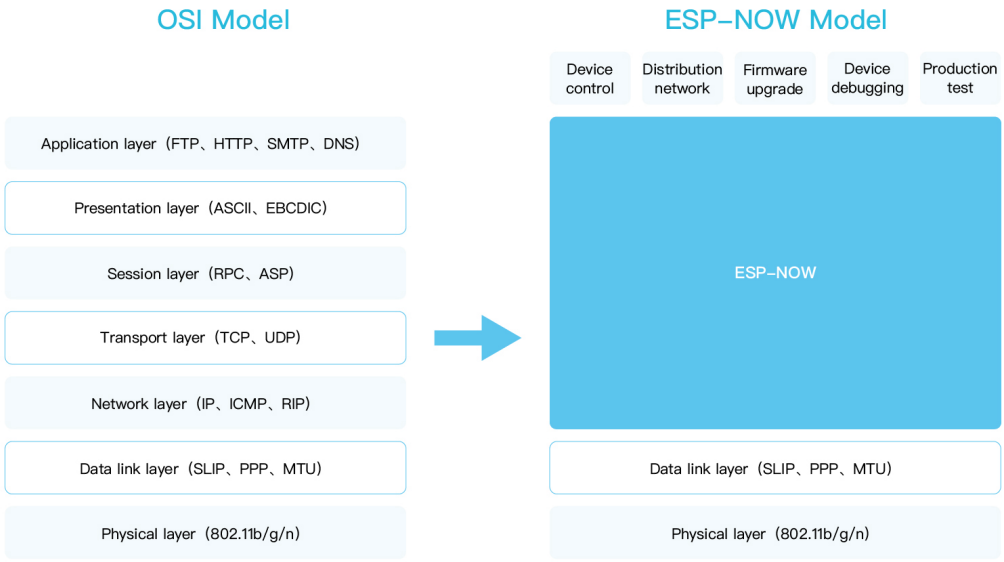


Figura 2.4: Comparativa entre el modelo OSI y el modelo ESP-NOW [4]

desarrollada por Espressif y la tecnología de encriptación con *Counter Mode Cipher Block Chaining Message Authentication Code Protocol* (CCMP), logrando una solución de comunicación segura y sin conexión [40]. ESP-NOW ocupa menos recursos de CPU y flash, pudiendo funcionar con Wi-Fi y Bluetooth LE. Es compatible con la serie de ESP8266, ESP32, ESP32-S y ESP32-C [4].

ESP-NOW tiene las siguientes características [40, 41, 4]:

- Comunicación unidifusión cifrada y no cifrada.
- Se puede transportar una carga útil de hasta 250 bytes por paquete.
- La función de devolución de llamada de envío se puede configurar para informar a la capa de aplicación del éxito o fracaso de la transmisión.
- La velocidad de bits ESP-NOW predeterminada es de 1 Mbps.
- El número máximo de dispositivos emparejados soportados es 20.
- Permite una comunicación bidireccional Unicast y Broadcast.
- Admite la conexión y el control de dispositivos de uno a muchos y de muchos a muchos dispositivos.

2.4.2. ESP-WIFI-MESH

Es un protocolo de red desarrollado y patentando por la compañia Espressif. Se construye sobre el protocolo WiFi y permite que los nodos se extiendan por grandes áreas, ampliando la conexión a Internet más allá del alcance del *Access Point* (AP). Es autónomo e implementa funciones de autoorganización y autorreparación.

Los nodos ESP32 combinan simultáneamente el modo AP y el modo estación base. De acuerdo con la Figura 2.5, la topología de ESP-WIFI-MESH forma un árbol que tiene un nodo raíz (*Root Node*), con información completa sobre la malla y que es el único que conecta la red externa a la malla y transmite paquetes dentro y fuera de ella. El nodo raíz tiene varios nodos secundarios (*Intermediate Parent Nodes*) y a estos se pueden conectar varios nodos hoja (*Leaf Nodes*), los cuales solo pueden transmitir y recibir paquetes, sin tener a su vez nodos secundarios. Los nodos inactivos (*Idle Nodes*) son nodos que aún no se han unido a la malla [5].

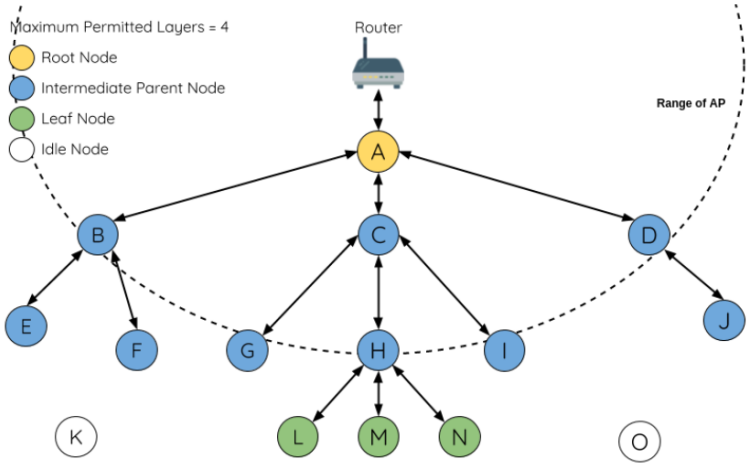


Figura 2.5: Topología de la red de malla ESP-WIFI-MESH [5]

2.4.3. PainlessMesh

PainlessMesh es una librería desarrollada en C++ y que usa JSON para el intercambio de mensajes, permitiendo una fácil depuración de los mismos. La configuración de la red en malla se la puede realizar de una forma sencilla, permitiendo la sincronización en el tiempo en el caso de ser necesario. La topología de la red se ilustra en la Figura 2.6.

La topología de esta red en forma de estrella puede cambiar de acuerdo a las necesidades, gracias a que los nodos pueden actuar como estación o como punto de acceso (AP). A di-

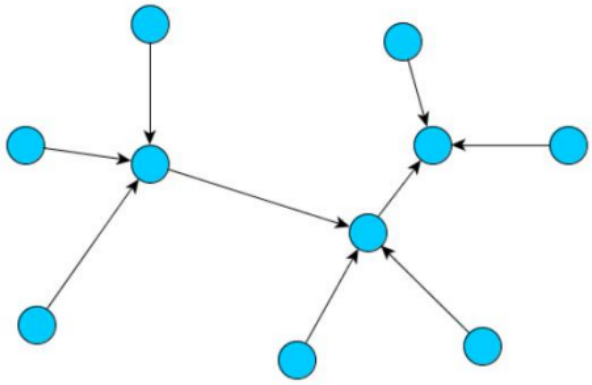


Figura 2.6: Topología de la red de malla Painlessmesh [5]

ferencia de ESP-WIFI-MESH todos los nodos son iguales, es decir, no existe jerarquía entre ellos. Con la finalidad de que cada nodo conozca la topología completa, cada uno de estos nodos intercambia sus conexiones y subconexiones con los demás. Esta topología se actualiza cada 3 segundos. Los nodos que aun no se encuentren dentro de la malla buscan un punto de acceso que no se encuentre en la lista de conexiones y con la mejor señal *Received Signal Strength Indicator* (RSSI); con esto se garantiza que no exista bucles entre los nodos [5].

2.5. Protocolo MQTT

MQTT es un protocolo de transporte de mensajería de publicación/suscripción cliente-servidor. Es liviano, libre, simple y diseñado para ser fácil de implementar. Estas características lo hacen ideal para su uso incluso en entornos restringidos. Es muy utilizado en la comunicación *Machine-to-Machine* (M2M) e IoT donde se requiere una pequeña huella de código y/o el ancho de banda de la red es escaso. MQTT es utilizado en una gran variedad de industrias, como la automotriz, la manufactura, las telecomunicaciones, etc. [6].

2.5.1. Arquitectura de publicación/suscripción

El modelo publicación/suscripción de MQTT elimina la comunicación directa entre el editor del mensaje y el suscriptor. La actividad de filtrado del intermediario permite controlar qué cliente/-suscriptor recibe determinado mensaje. El desacoplamiento se considera en tres dimensiones [42]:

- **Desacoplamiento espacial:** el editor y el suscriptor no necesitan conocerse.
- **Desacoplamiento de tiempo:** no es necesario que el editor y el suscriptor se ejecuten al mismo tiempo.

- **Desacoplamiento de sincronización:** no es necesario interrumpir las operaciones en ambos componentes durante la publicación o la recepción.

MQTT utiliza el filtrado de mensajes basado en temas. Cada mensaje contiene un tema (asunto) que el *broker* puede usar para determinar si un cliente suscriptor recibe el mensaje o no. También puede configurar el filtrado basado en contenido con un sistema de complemento personalizado [42].

2.5.2. Establecimiento de conexión entre cliente y *broker*

Un cliente MQTT es cualquier dispositivo (desde un microcontrolador hasta un servidor completo) que ejecuta una biblioteca MQTT y se conecta a un agente MQTT a través de una red. Las bibliotecas de cliente MQTT están disponibles para una gran variedad de lenguajes de programación.

Por su parte, el *broker* es responsable de recibir todos los mensajes, filtrarlos, determinar quién está suscrito a cada mensaje y enviar el mensaje a estos clientes suscritos. También sostiene las sesiones de todos los clientes persistentes, incluidas las suscripciones y los mensajes perdidos. Dependiendo de la implementación, un *broker* puede manejar hasta miles de clientes MQTT conectados simultáneamente.

El protocolo MQTT se ejecuta a través de TCP/IP y tanto el cliente como el *broker* deben tener una pila TCP/IP. Para iniciar una conexión, el cliente envía un mensaje de CONNECT al *broker*. El *broker* responde con un mensaje CONNACK (*Acknowledgement*) y un código de estado. Una vez que se establece la conexión, el *broker* la mantiene hasta que el cliente envía un comando de desconexión o se interrumpe la conexión [42]. El envío de los mensajes CONNECT y CONNACK entre el cliente y el *broker* se representa en la Figura 2.7.

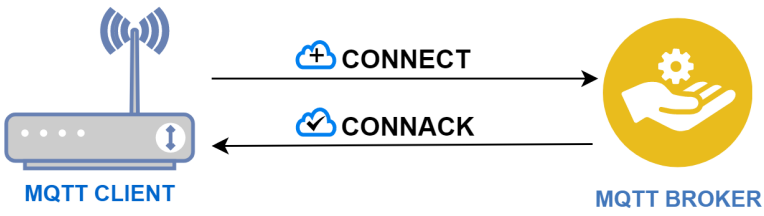


Figura 2.7: Mensajes de conexión entre cliente MQTT y *broker* MQTT

2.6. Thingsboard

Thingsboard es un plataforma IoT de código abierto que permite un desarrollo rápido, gestión y escalado de proyectos IoT. Hace posible la conectividad de dispositivos a través de protocolos estándar de la industria como: MQTT, Protocolo de aplicación restringida (CoAP) y Protocolo de transferencia de hipertexto (HTTP); además admite implementaciones locales y en la nube. ThingsBoard combina escalabilidad, tolerancia a fallos y rendimiento para evitar la pérdida de datos. El usuario puede seleccionar la edición *Community*, *Cloud* o *Professional* dependiendo de sus necesidades [43].

Entre sus funciones permite [43]:

- Aprovisionar dispositivos, activos o contenedores para determinados dispositivos, y clientes; pudiendo definir las relaciones entre ellos.
- Recopilar y visualizar los datos de dispositivos y activos.
- Analizar la telemetría entrante y activar alarmas con procesamiento de eventos complejos.
- Controlar los dispositivos mediante Llamada a procedimiento remoto (RPC).
- Crear flujos de trabajo basados en un evento del ciclo de vida del dispositivo, como un evento de Interfaz de programación de aplicaciones (API) REST, una solicitud de RPC, etc.
- Diseñar paneles dinámicos y receptivos. Así como presentar telemetría de dispositivos o activos a los clientes.
- Habilitar características específicas de casos de uso mediante cadenas de reglas personalizables.
- Enviar los datos de dispositivos a otros sistemas.

Thingsboard está diseñado para ser escalable horizontalmente, tolerante a fallos, robusto, eficiente, duradero y personalizable. En el diagrama de la Figura 2.8 se muestran los componentes clave del sistema y las interfaces que proporcionan; los activos brindan la posibilidad de reorganizar los dispositivos (*devices*) dentro de contenedores para una mejor estructura y organización del sistema.

Como se observa en la Figura 2.8, Thingsboard proporciona APIs basadas en MQTT, CoAP, HTTP y *Lightweight Machine to Machine* (LwM2M) que están disponibles para las aplicacio-

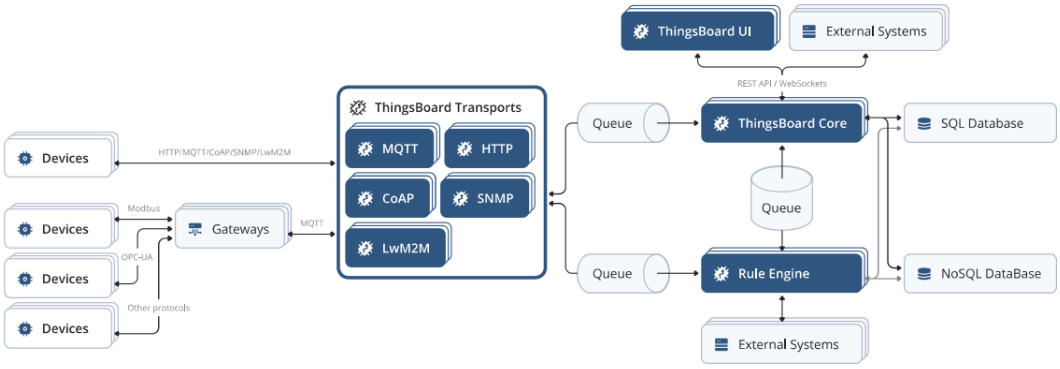


Figura 2.8: Arquitectura de componentes e interfaces de Thingsboard [6]

nes/*firmware* de los dispositivos. Cada una de las APIs de protocolo es proporcionada por un componente de servidor independiente y forma parte de la capa de transporte de Thingsboard.

El núcleo de Thingsboard es responsable de manejar las llamadas a la API de REST y las suscripciones a *WebSocket*. También es responsable de almacenar información actualizada sobre las sesiones activas de los dispositivos, así como monitorear el estado de conectividad de los mismos.

El motor de reglas de Thingsboard (*Rule Engine* traducido al inglés) es el corazón del sistema y es responsable de procesar los mensajes entrantes. Los nodos de *Rule Engine* pueden unirse al clúster, donde cada nodo es responsable de ciertas particiones de los mensajes entrantes. El motor de reglas se suscribe a la fuente de datos entrante y reconoce el mensaje una vez que se procesa.

El motor de reglas permite crear flujos de trabajo basados en eventos. Tiene tres componentes principales [44]:

- **Mensaje:** es cualquier evento entrante. Puede ser un dato de un dispositivo, el ciclo de vida de un dispositivo, un evento API REST, una solicitud RPC, etc.
- **Nodo de regla:** una función que se ejecuta en un mensaje entrante. Existen nodos de filtrado, de transformación o de ejecución de una determinada acción.
- **Cadena de reglas:** conecta a los nodos entre sí mediante relaciones, por lo que el mensaje saliente de un nodo se envía a los siguientes nodos conectados.

En cuanto a la Interfaz de usuario (UI) web, Thingsboard proporciona componentes ligeros escritos con *Express.js* para alojar contenido estático de UI. Sin embargo, esos componentes no permiten muchas configuraciones. Además, Thingsboard admite varias implementaciones de colas de mensajes (*QueUe*): *Kafka*, *RabbitMQ*, *AWS SQS*, *Azure Service Bus* y *Google Pu-*

b/Sub. El uso de colas duraderas y escalables permite implementar contrapresión y equilibrio de carga.

Thingsboard utiliza una base de datos para almacenar entidades (dispositivos, activos, clientes, paneles, etc.) y datos de telemetría (atributos, lecturas de sensores de series temporales, estadísticas, eventos). Se recomienda usar *PostgreSQL* como base de datos de Lenguaje de consulta estructurada (SQL) principal; sin embargo, también es posible utilizar almacenamientos híbridos de *PostgreSQL y Cassandra* o de *PostgreSQL y TimescaleDB*, con la capacidad de elegir dónde almacenar las entidades principales y dónde almacenar datos de telemetría [6].

2.7. Adquisición de datos

En la actualidad varias instituciones deportivas están apostando a tecnologías que permitan obtener una retroalimentación de la biomecánica. Confían que con la ayuda de datos cinemáticos y cinéticos se mejore la técnica deportiva. La adquisición de datos cinemáticos a través de unidades inerciales o acelerómetros brindan una retroalimentación biomecánica relacionada con el control a través de los cambios lineales y angulares [20].

El sensor inercial MPU6050 es una solución de procesamiento de movimiento que presenta 6 ejes, compuesta por un acelerómetro de 3 ejes y un giroscopio de 3 ejes. Con el procesador digital de movimiento incorporado y *Motion Fusion* es capaz de brindar soluciones a algoritmos complejos de sensores de 9 ejes. Este módulo sensor, permite comunicaciones *Inter-Integrated Circuit* (I2C) de hasta 400kHz e integra un regulador de voltaje de 3.3V, por lo que se lo puede alimentar con 5V a través del pin VCC. En la Figura 2.9, se describe el *Pinout* del módulo en cuestión.

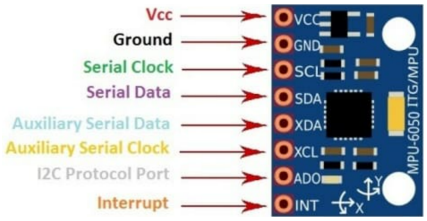


Figura 2.9: Definición de pines del sensor MPU6050 [7]

Para un mejor rendimiento, el MPU6050 permite regular el rango de la escala del giroscopio, que puede estar entre ±250, ±500, ±1000 y ±2000 °/seg (dps). De manera similar, el rango del acelerómetro puede ser ajustado entre ±2g, ±4g, ±8g y ±16g. En el caso del acelerómetro

integrado, la frecuencia de muestreo es de 1kHz/s, mientras que del giroscopio es de 8kHz/s cuando el *Digital low pass filter* (DLPF) se encuentra desactivado [7].

2.8. Almacenamiento

A medida que la duración de la adquisición de datos incrementa, una mayor capacidad de almacenamiento es requerida. En esta situación, la memoria interna de módulos como la del ESP32 se ve limitada, por lo que se requiere la incorporación de memoria externa, tal como una tarjeta microSD.

Las tarjetas microSD se han convertido en un estándar, desplazando a otros medios de almacenamiento de datos debido a su gran capacidad, compatibilidad y pequeño tamaño; siendo integradas en una gran cantidad de dispositivos. Si bien el ESP32 soporta la adición de SRAM externa, esta puede ser de hasta 8MB [45]. Por otro lado, con ESP32 también se puede utilizar memorias NOR Flash *Serial Peripheral Interface* (SPI), que son un medio electrónico de almacenamiento de memoria no volátil que se puede borrar y reprogramar eléctricamente [46]. Sin embargo, a pesar de que una memoria Flash SPI es más rápida y tiene un consumo de energía más bajo que una microSD, su disponibilidad es escasa en nuestro medio.

2.9. Trabajos Relacionados

Son muchas las posibilidades que brindan los módulos ESP32 en la creación de dispositivos de bajo costo, mismos que pueden estar al alcance de un amplio grupo de usuarios. En el deporte, dichos módulos se aplican en varios e interesantes temas de investigación. En el trabajo presentado en [47], el autor propone el diseño de un sistema sensorial para la medición de la presión en el pie al realizar la práctica de patinaje de velocidad. Las pruebas realizadas se llevan a cabo utilizando sensores de presión en siete puntos distribuidos en los metatarsos, el pie medio y el talón, los datos capturados son enviados en tiempo real hacia una aplicación *Android* utilizando comunicación Bluetooth. El módulo ESP32 es el que mejor se adaptó a los requerimientos del proyecto debido a su baja demanda de consumo energético, a sus reducidas dimensiones y menor peso comparado con otras tarjetas.

De manera semejante, en [48] se presenta el diseño e implementación de un dispositivo que utiliza módulos ESP32 para la medición de diferentes variables relacionadas con el comportamiento del deportista al momento de realizar la actividad física. Se implementan acelerómetros, giroscopios, sensores de presión y flexómetros para caracterizar el movimiento del

pie del atleta. Posteriormente, se envía dicha información a un dispositivo móvil. El módulo ESP32, gracias a su antena transceptora que incorpora WiFi, junto a la inclusión de 18 pines con conexión al *Analog-to-Digital converter* (ADC) de la tarjeta, permite obtener un dispositivo portable y de bajo coste que se incorpora al calzado deportivo sin afectar significativamente el uso de este.

El autor de [49] utiliza módulos ESP32 para crear dispositivos *Wearables* con sensores inerciales que apoyan la adquisición y mejora de las habilidades deportivas. El dispositivo de grabación de movimiento desarrollado se compone de un microcontrolador ESP32, acelerómetro y giroscopio, brújula digital y GPS. Los datos de movimiento y ubicación se almacenan en una tarjeta micro SD, para posteriormente ser transmitidos a un dispositivo de borde utilizando tanto Bluetooth como WiFi. La facilidad de programación del ESP32 en el entorno de Arduino con muchas bibliotecas compatibles, incluidos los dispositivos inalámbricos, es una razón adicional por la que se utilizan estos módulos en dicho proyecto.

En [50] se propone el diseño e implementación de sistema de sensores inerciales de bajo costo para análisis biomecánico de un sistema vivo. El sistema consta de una placa maestra y tres sensores esclavos individuales que incluyen una unidad de medición inercial, un microcontrolador ESP32 y una batería. El objetivo es que mediante el uso de dos o más sensores, como acelerómetros y giroscopios, se pueda obtener la cinemática y forma del cuerpo en movimiento. Una vez recolectada la información con los sensores, se envía a una computadora donde se procesa los resultados transformándolos en animación de la captura de movimiento. Gracias al uso de ESP32, el sistema es portatil, móvil y con conexión inalámbrica.

Un sistema portable de reconocimiento del patrón de marcha basado en *Web* se presenta en [51]. Se utilizan siete sensores de medición inercial en diferentes partes del cuerpo del deportista, cada uno conectado a un microcontrolador ESP32, el cual se encarga de procesar los datos recopilados. Los datos procesados dentro de ESP32 se envían mediante WiFi a un servidor local facilitado con Raspberry Pi 3. Se diseña una interfaz basada en *Web* dentro del servidor local para almacenar y mostrar en gráficas individuales los datos recibidos.

Dentro del trabajo presentado en [52], se desarrolla un sistema de medida de distancia para competiciones de ciclismo de ruta. Se diseña un dispositivo detector del incumplimiento de la distancia de *drafting* que sirva para informar tanto al deportista como a la organización acerca de la infracción que se está llevando, dicho dispositivo se coloca en la bicicleta del deportista y se comunica con una aplicación que interpreta los datos y verifica la información. Para la comunicación se utiliza WiFi y el protocolo MQTT mediante ESP32, el cual aporta una gran

capacidad de conectividad y es de bajo coste, lo que representa una ventaja significativa.

En un escenario semejante, en [53] se propone un sistema de monitorización de señales fisiológicas aplicado a ciclistas. Se pretende que, a través de un equipo de bajo costo y consumo de energía, los ciclistas puedan compartir la vía con vehículos y peatones sin poner en riesgo su integridad. El prototipo tiene dos partes, una parte fija ubicada en la bicicleta y dos dispositivos portátiles que utiliza el ciclista. En la parte móvil se procesan datos fisiológicos del ciclista. Por otro lado, la parte fija se encarga de gestionar las alertas que genera el sistema y enviar esta información a través del protocolo MQTT a un servidor en la nube. Tanto la parte móvil como fija del prototipo utilizan un microcontrolador ESP32 para la comunicación inalámbrica. Adicionalmente, una aplicación *Android* trabaja como cliente MQTT y está suscrito a los mismos tópicos que publica el ESP32 para recibir notificaciones de las alertas.

En [54], se tiene un sistema de guía para atletas con discapacidad visual en la carrera olímpica en pista. Se desarrolla un prototipo equipado con comunicación inalámbrica y un código de lenguaje que utiliza estímulos vibratorios para generar comandos que guían al atleta con discapacidad visual en la carrera olímpica en pista. El guía opera un teléfono inteligente para pasar comandos de forma remota al atleta y orientarlo durante la carrera. En un módulo ESP32, se desarrolló un programa para controlar la intensidad de vibración de dos pulseras vibratorias y dos motores vibradores colocados en el pecho y la espalda del atleta. El microcontrolador ESP32, recibe la información transmitida por el celular y reenvía los comandos para activar el dispositivo vibrador correcto.

El trabajo presentado en [55] corresponde a un sistema para la determinación automática de la puntuación en combates de Karate. A través de este sistema se puede clasificar los movimientos del brazo del juez central y obtener la puntación de combate correspondiente. Se diseña e implementa dos brazaletes que reconocen doce movimientos de relevancia para la puntuación y que son cómodos para el usuario; dichos dispositivos utilizan placas de desarrollo ESP32 y acelerómetros tipo sistemas microelectromecánicos (MEMS). Los datos son transmitidos desde cada brazalete hacia dos tarjetas clasificadoras ESP32 mediante WiFi. Dichas tarjetas clasifican la información recibida y se procede a visualizar en una interfaz del computador las faltas y puntos en tiempo real.

La creación de un dispositivo IoT denominado “Workout Box” se presenta en [56]. Este dispositivo permite que las rutinas de entrenamiento sean más eficientes gracias a información y estadísticas en tiempo real. Puede ser colocado en distintas zonas del cuerpo y en aparatos de gimnasio, dependiendo del ejercicio a realizar. Su funcionamiento consiste en la captura de

datos de un determinado ejercicio a través de múltiples sensores, lo que permite la medición del número de repeticiones realizadas, la medición de la distancia recorrida o la determinación de la velocidad y tiempo de ejecución. Posteriormente, se envían los datos a un dispositivo móvil a través de Bluetooth. La aplicación móvil tiene la capacidad de mostrar rutinas de acuerdo a los objetivos de entrenamiento. Una vez más, el uso de un módulo ESP32 permite crear un dispositivo óptimo de tamaño reducido que se ajuste a todos los requerimientos de diseño.

El trabajo presentado en [57] corresponde a la implementación de un sistema de monitoreo de señales biomédicas para cuantificar el rendimiento físico de futbolistas. Un dispositivo cliente ESP32 es colocado en un chaleco deportivo, mismo que se encarga de gestionar la medición de la frecuencia cardiaca, la temperatura del deportista, la distancia que recorre, la velocidad y la aceleración. Adicionalmente, se realiza una base de datos para almacenar la información emitida por los sensores, permitiendo que dichos datos sean posteriormente revisados por un médico deportivo. La transmisión de datos se realiza mediante WiFi desde el dispositivo cliente al dispositivo servidor y, a su vez, el servidor envía estos datos por medio de comunicación serial a la base de datos local implementada en Matlab.

2.10. Conclusiones

El patinaje de velocidad es un deporte que va sumando seguidores con el pasar del tiempo. Es una disciplina aeróbica y anaeróbica que demanda gran preparación física y mental, siendo necesaria la medición y caracterización de los movimientos del deportista con el objetivo de mejorar sus técnicas de ejecución. Dichas técnicas pueden ser evaluadas de manera cualitativa y cuantitativa, siendo esta última la menos estudiada. Debido a la falta de disponibilidad o altos costos de dispositivos que permitan medir el movimiento de los deportistas.

El uso de IoT y los dispositivos *Wearables* hoy en día son aplicados en multitud de campos de la ciencia con un mercado en constante crecimiento, incluso en el ámbito deportivo. Es en este contexto donde la integración de sensores inerciales en dispositivos IoT, permitiría la adquisición de datos de los movimientos de los deportistas, pudiendo ser analizados en lo posterior con el fin de mejorar cuantitativamente la técnica y el rendimiento. Un sensor de interés es el MPU6050, que se comunica a través de I2C a una frecuencia de 400kHz, logrando adquirir los 3 ejes del acelerómetro y los 3 ejes del giroscopio.

Thingsboard es un plataforma de código abierto que permite el rápido desarrollo, gestión y escalado de proyectos IoT. Esta plataforma permite habilitar características específicas de

casos de uso mediante cadenas de reglas personalizables. Además, permite conectividad de dispositivos a través de protocolos estándar de la industria como MQTT; este es un protocolo de transporte de mensajería entre un cliente y un *broker* responsable de recibir todos los mensajes, filtrarlos y gestionarlos. MQTT cuenta con las características de ser liviano, libre, simple y diseñado para ser fácil de implementar; además, es muy utilizado en el mundo de IoT.

Entre los módulos de gran versatilidad para el desarrollo de dispositivos *Wearables* se encuentra ESP32. El microcontrolador ESP32 es un chip versátil y de bajo consumo, que permite la conectividad a través de Wi-Fi en la banda de 2.4GHz, cuenta con doble núcleo y puede trabajar a una frecuencia de 240MHz. El protocolo de comunicación ESP-NOW permite el transporte de paquetes pequeños a una velocidad de 1Mbps, ya sean cifrados o no. También es importante recalcar que el protocolo ESP-NOW permite realizar varias configuraciones entre los nodos, lo cual le hace un protocolo ideal para el desarrollo de aplicaciones.

Finalmente, en cuanto a la literatura revisada se corrobora la relevancia que tienen los módulos ESP32 en el desarrollo de dispositivos IoT y la cuantificación de la información de diversas disciplinas deportivas, en donde el uso de sensores inerciales es de suma importancia. Sin embargo, la implementación de tecnologías, que midan el desempeño de los deportistas y que permitan comparar los datos obtenidos con las diversas técnicas aplicadas, sigue siendo un campo con escasos estudios. Es por ello que el desarrollo de un sistema para la adquisición de datos biomecánicos de las extremidades de patinadores de velocidad es un interesante aporte. Enfocar la propuesta a un deporte como el patinaje de velocidad permite tener una visión más específica de los objetivos a lograr; no obstante, un sistema de este tipo podría aplicarse a multitud de disciplinas y ser una base desde la que realizar mejoras de acuerdo una necesidad específica.

Diseño e implementación

En este capítulo se indican las diferentes secciones involucradas dentro del desarrollo del sistema de toma de datos de aceleración. Se describen los procesos y las herramientas utilizadas.

3.1. Arquitectura del sistema

En esta sección se presenta la arquitectura general del sistema IoT para la adquisición de datos biomecánicos de un patinador de velocidad. Como se observa en la Figura 3.1, se propone un sistema que trabaja de manera inalámbrica y que puede ser dividido en tres secciones: estructura de los nodos sensores, servidor Web y aplicación móvil.

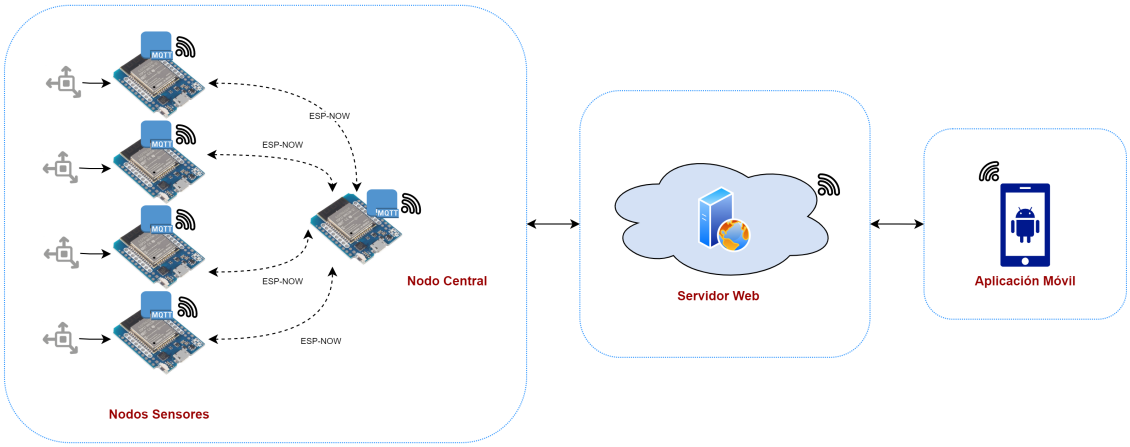


Figura 3.1: Arquitectura del sistema de adquisición de datos biomecánicos

En el proceso de captura y almacenamiento de los datos de aceleración del deportista se tiene cuatro nodos sensores involucrados (ver Figura 3.1). Estos nodos se comunican con un quinto nodo central utilizando el protocolo de comunicación WiFi ESP-NOW. Este nodo central permite la sincronización del tiempo de captura de los datos de aceleración. Dichos datos son tomados desde acelerómetros conectados a los nodos sensores, mismos que son colocados en las cuatro extremidades del deportista. En la Figura 3.2 se indican con letras las ubicaciones del cuerpo del patinador en donde se colocan los distintos nodos, el nodo A es el encargado de sincronizar a los demás. Tanto lo nodos sensores como el nodo central se comportan como clientes MQTT.

Además, se dispone de un servidor Web en la plataforma de *Thingsboard Cloud*, mismo que

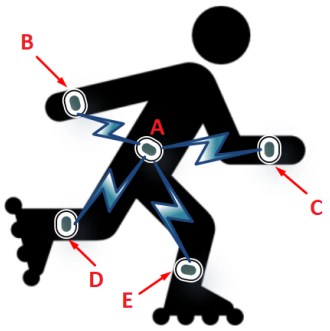


Figura 3.2: Ubicación de los nodos en el cuerpo del patinador

recibe los datos almacenados en cada uno de los dispositivos una vez terminada la práctica deportiva. Este servidor se comunica tanto con los nodos como con una aplicación móvil desarrollada para *Android* e integra el broker MQTT responsable de recibir y filtrar los mensajes enviados por los clientes MQTT. La aplicación móvil envía la orden de inicio y final de captura de los datos de interés, así como también permite la visualización de la información de aceleración y distancia.

Mientras se lleva a cabo la actividad deportiva, los nodos sensores mantienen una comunicación con el nodo central en modo local. Una vez terminada la práctica, dichos nodos utilizan internet para enviar los datos recolectados directamente al servidor Web; el nodo central mantiene una conexión tanto local como a internet, para comunicarse con los nodos sensores y con el servidor Web. Igualmente, se requiere de acceso a internet para la comunicación entre la aplicación móvil y *Thingsboard Cloud*.

3.2. Diseño y construcción de los nodos

Dentro del diseño y construcción de los nodos, se desarrolló una PCB en la que se conectan los diferentes módulos electrónicos involucrados. Posteriormente, se diseñó una caja contenedora, dentro de la cual se fijan la PCB junto con un batería tipo *Litio y polímero* (Li-Po) para su respectiva alimentación, brindando mayor seguridad y comodidad en el uso de los dispositivos.

3.2.1. PCB y módulos utilizados

Debido a su versatilidad, tamaño y bajo consumo; así como su capacidad de procesamiento y gran cantidad de contenido *Open Source* disponible, se ha decidido utilizar el chip ESP32-

WROOM-32 para la creación de los dispositivos. Del mismo modo, para la captura de los datos de aceleración se opta por los sensores inerciales MPU6050, ya que su escala de medición y su frecuencia de muestreo se ajusta a lo que se busca en el presente proyecto. Además, la relativa facilidad con la que el sensor MPU6050 se vincula con el chip ESP32 es una ventaja adicional.

Para el diseño de la PCB de los nodos se utiliza el software *Altium Designer* en su versión $19.0.15$. A partir del esquemático de la Figura 3.3, se obtuvo la PCB ilustrado en la Figura 3.4 y la Figura 3.5, en donde se indican las capas superior e inferior de la misma. El esquemático incorpora el módulo ESP32, el sensor MPU6050 y el regulador de voltaje $AMS1117-3.3$V. El regulador de voltaje se encarga de transformar la tensión de la batería Li-Po de 7.4V a 3.3V para la alimentación del ESP32 y los módulos utilizados, entregando una corriente máxima de 1A [58].

En la Figura 3.3 también se considera la implementación de un lector microSD [59], mismo que permite la inserción de una tarjeta del mismo tipo para el almacenamiento de los datos biomecánicos adquiridos. Esto debido a que la memoria interna del ESP32 brinda únicamente 8KB de SRAM para el almacenamiento de datos. Por lo que la implementación de una memoria microSD brinda mayor capacidad de almacenamiento de una manera relativamente económica y sencilla, debido a la información disponible en la comunidad.

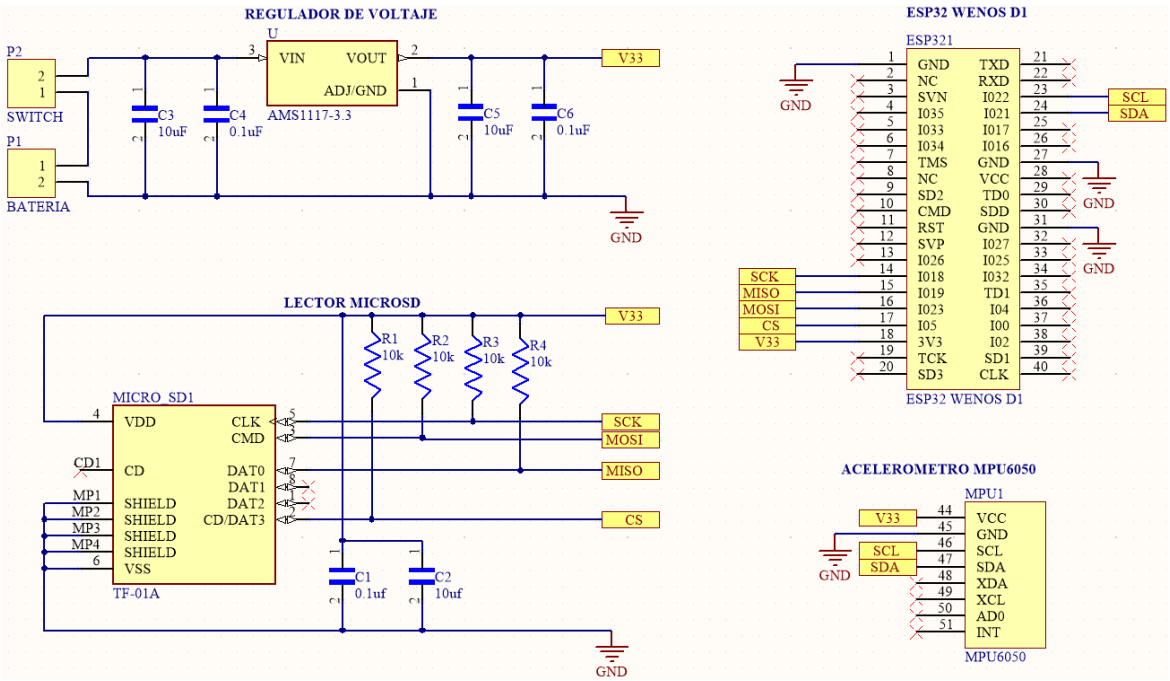
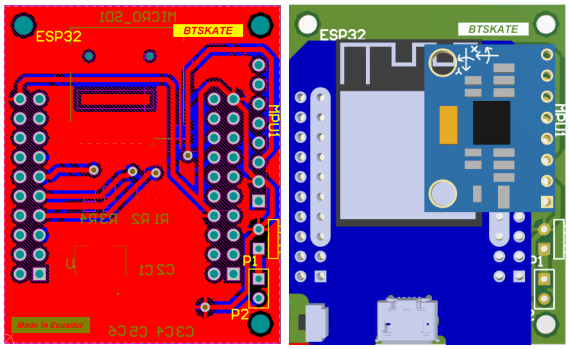


Figura 3.3: Esquemático de la PCB diseñado en *Altium Designer*

Se utiliza una batería Li-Po de 7.4V debido a que el regulador de voltaje $AMS1117-3.3$V,

requiere un voltaje de entrada de al menos 1.5V por encima del valor de salida [58]. Se podría pensar en utilizar una batería Li-Po de 3.7V directamente, pero el inconveniente es que cuando esta se encuentra completamente cargada puede llegar a entregar voltajes cercanos a los 4V y el puerto de alimentación del ESP32 tiene una tolerancia de voltaje de alimentación de hasta 3.7V.

Los componentes que mayor corriente requieren son el ESP32, el sensor inercial MPU6050 y la tarjeta microSD. La batería Li-Po tiene un voltaje de 7.4V y una corriente de 400mAh. El módulo ESP32 puede consumir 225mA de corriente en los niveles de consumo máximos [60], el sensor MPU6050 puede llegar a consumir 3.8mA cuando se está censando en sus 6 ejes [61] y una microSD típica consume hasta 100mA a 3.3V [62], lo cual daría como resultado un consumo de corriente de alrededor de los 330mA. Por lo tanto, con la batería utilizada se garantizaría que los dispositivos puedan funcionar de manera ininterrumpida por alrededor de 1 hora.



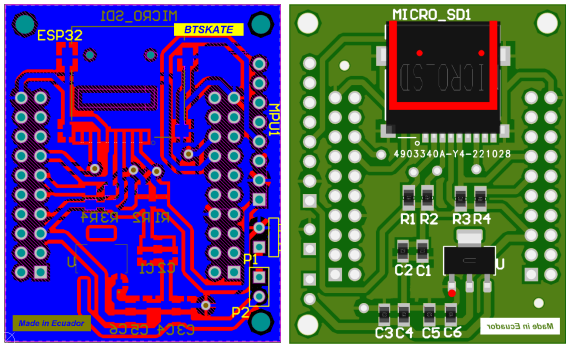
(a) 2D (b) 3D

Figura 3.4: Capa superior de la PCB

La PCB resultante tiene unas dimensiones de 44mm $\times$ 36mm, de largo y ancho respectivamente. En la Figura 3.4 se presenta el trazado de las pistas en la capa superior de la placa (3.4a), así como su modelo en 3D (3.4b). Sobre esta capa serán colocados el módulo ESP32 Wemos D1 y el sensor MPU6050.

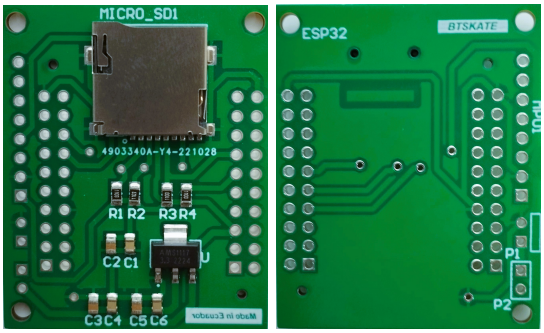
De manera similar, en la Figura 3.5a se muestra el trazado de las pistas en la capa inferior de la placa, y en la Figura 3.5b se indica el modelo en 3D correspondiente. En esta capa, se encuentra integrado como Dispositivo de montaje superficial (SMD) el lector microSD y el regulador de voltaje, con sus respectivos capacitores y resistencias.

En la Figura 3.6 se muestra el resultado impreso de la placa descrita anteriormente. En la Figura 3.7 se indica la placa con los módulos ESP32 y MPU6050 soldados a la misma, así como con la memoria microSD colocada en su respectiva ranura.



(a) 2D (b) 3D

Figura 3.5: Capa inferior de la PCB



(a) Capa superior (b) Capa Inferior

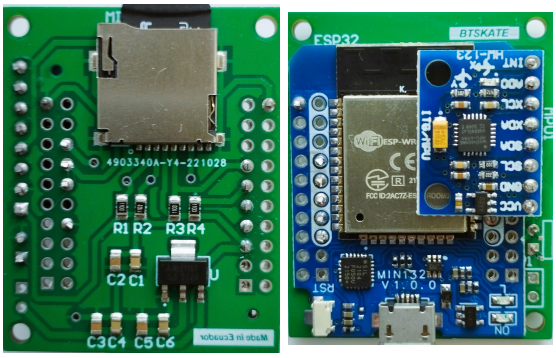
Figura 3.6: PCB impresa

3.2.2. Modelado de cajas contenedoras 3D

Con el objetivo de brindar mayor comodidad al usuario en el uso de los dispositivos y que estos sean más resistentes y estéticos, se diseñaron unas cajas 3D, dentro de las cuales se colocan las PCBs, la batería tipo Li-Po y un interruptor para encender el dispositivo.

En la plataforma online de *Tinkercad* se realizó el diseño de una caja contenedora y luego se imprimieron 5 cajas semejantes, para cada uno de los nodos del sistema. En la Figura 3.8 se indican las dimensiones de largo, ancho y alto de dichas cajas, las cuales son de 4.7cm, 3.9cm y 2.6cm respectivamente. Como se puede observar, en el diseño se dispone de una tapa, misma que se sujeta con tornillos al resto de la caja, permitiendo que esta se pueda abrir con facilidad en el caso de requerirse el reemplazo o revisión de algún componente.

En la Figura 3.9 se presenta un vista superior tanto del interior de la caja (3.9a) como de la tapa (3.9b). Se incluyen unos orificios en los vértices de la caja, donde se colocan insertos para roscar los tornillos que sujetan la tapa; en el diseño de la tapa se considera una ranura, por donde pasará una tira velcro con la que se ajusta el dispositivo al cuerpo del deportista.



(a) Capa superior (b) Capa Inferior

Figura 3.7: PCB impresa, con módulos ESP32 y MPU6050 soldados a la misma

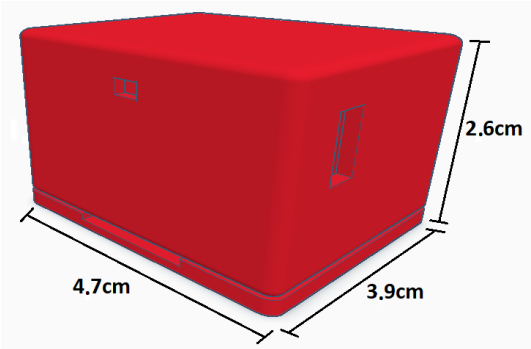
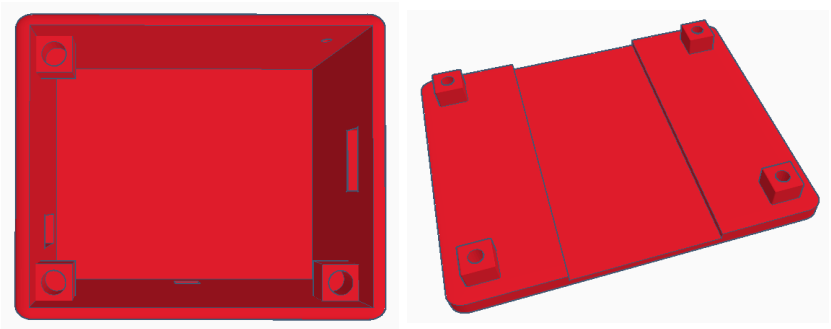


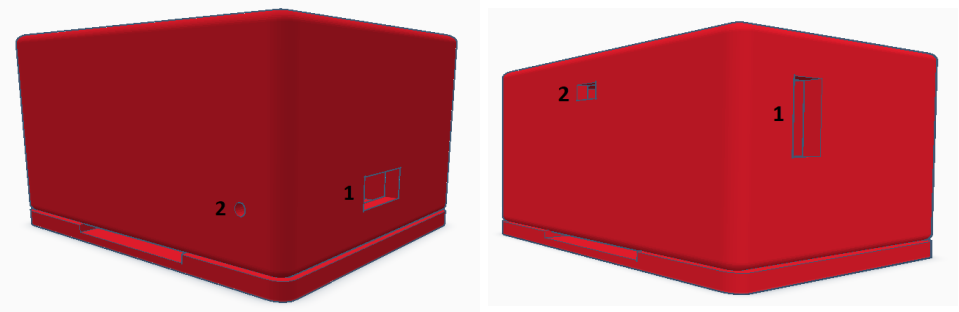
Figura 3.8: Dimensiones de la caja 3D

Por su parte, en la Figura 3.10 se indican las vistas laterales de la caja 3D. De acuerdo con la Figura 3.10a, en el punto **1** se considera un orificio por el que se accede al puerto microUSB del ESP32, en el caso de ser necesario realizar modificaciones en su Software. En **2** se incluye un orificio circular que facilita la acción de reiniciar el nodo. De la misma manera, en la Figura 3.10b, en **1** se indica la ubicación del puerto de carga de la batería Li-Po y en **2** se colocará un interruptor para encender o apagar el dispositivo.



(a) Interior de la caja (b) Tapa

Figura 3.9: Vista interior de la caja y su tapa



(a) (b)

Figura 3.10: Vistas laterales de caja 3D

3.2.3. Prototipo final

En la creación de los dispositivos se tuvieron dos versiones. En la primera versión se utilizaron unas PCB más sencillos, que contienen únicamente pistas, añadiéndose luego los módulos para la tarjeta microSD y para la regulación del voltaje de la batería, tal y como se indica en la Figura 3.11. Las cajas 3D fueron impresas utilizando un plástico rígido. El primer prototipo armado se indica en la Figura 3.12.

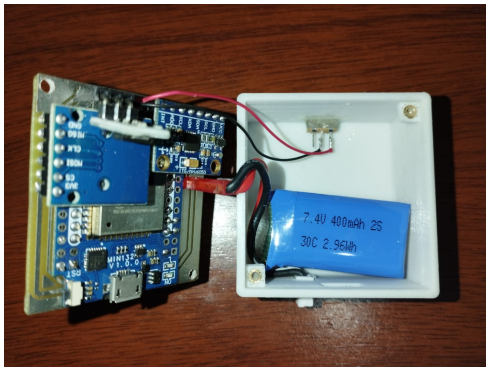
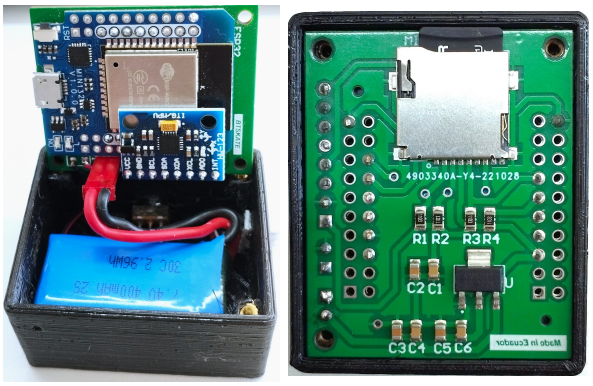


Figura 3.11: PCB y módulos utilizados en el prototipo 1

Para el prototipo final, una vez impresas las cajas contenedoras utilizando un material flexible, se colocaron las PCB, las baterías Li-Po y los interruptores dentro de las mismas, como se muestra en la Figura 3.13a. En la Figura 3.13b se aprecia como todos los componentes son colocados de una manera óptima en pro de que el tamaño de los dispositivos sea reducido. En la Figura 3.14a y 3.14b se presenta una vista superior y lateral de un dispositivo armado; se ha colocado una cinta velcro delgada, misma que puede ser ajustada dependiendo del tamaño de la extremidad o ubicación del cuerpo del deportista en donde se colocará el nodo sensor. Un dispositivo semejante es armado para las demás extremidades y para el torso del patinador, excepto por el hecho de que el nodo del torso no requiere la colocación del sensor inercial ni de la memoria microSD.

Figura 3.12: Prototipo 1 armado



(a) (b)

Figura 3.13: Prototipo final. Batería, interruptor y PCB dentro de la caja 3D

En la Tabla 3.1 se indican las dimensiones de ambos prototipos, así como también su peso. Evidenciando las mejoras que se tiene al respecto en el prototipo final.

Tabla 3.1: Dimensiones y peso de los prototipos

	Prototipo 1	**Prototipo final**
Largo	5cm	4.7cm
Ancho	4.8cm	3.9cm
Altura	3cm	2.6cm
Peso	82g	51g

En el repositorio de GitHub [63], se encuentra el código fuente del presente proyecto. En el directorio “ESP32” se indica el código utilizado en los distintos nodos del sistema. El directorio “SERVIDOR THINGSBOARD” contiene el código en Javascript usado en las cadenas de reglas del servidor web. Por su parte, “APP” contiene todo el código de la aplicación de *Android*.

(a) Vista superior

(b) Vista lateral

Figura 3.14: Prototipo final completamente armado

3.3. Proceso de adquisición de datos

En esta sección se expone la secuencia que sigue el sistema propuesto. En la Figura 3.15, se presenta un diagrama que indica las fases generales involucradas en el proceso de adquisición de datos.

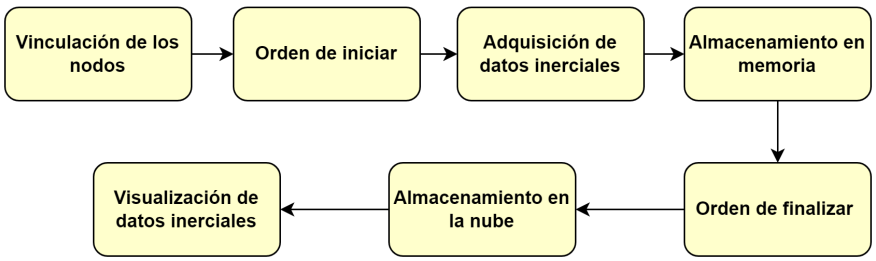


Figura 3.15: Fases involucradas en el proceso de adquisición de datos.

En base al diagrama presentado en la Figura 3.15, el proceso de adquisición de los datos del patinador obedece a la secuencia presentada a continuación:

1. Los diferentes nodos sensores se vinculan con el nodo central.
2. El nodo central informa al servidor Web, y esta a la aplicación móvil, que los nodos sensores se encuentran listos.
3. Desde la aplicación móvil, a través del servidor Web, se envía una orden de inicio para que los nodos comiencen a capturar los datos de aceleración del patinador.
4. Los datos adquiridos se almacenan en la memorias MicroSD conectadas a los nodos.
5. Finalizada la actividad del patinador, desde la aplicación móvil se ordena detener la

captura y almacenamiento de los datos inerciales.

6. Los valores de aceleración se envían al servidor Web desde cada nodo. La información no es enviada en tiempo real.

7. Desde la aplicación móvil desarrollada para *Android*, se visualiza la información de aceleración y distancia recorrida por las extremidades del patinador.

Los procesos de comunicación y sincronización de los nodos, configuración del servidor web *Thingsboard* y desarrollo de la aplicación móvil son abordados con mayor detalle en las secciones 3.4, 3.5 y 3.6 respectivamente.

3.4. Comunicación y sincronización de los nodos

En el proceso de comunicación y sincronización de los nodos sensores con el nodo central, se utiliza el protocolo ESP-NOW debido a su versatilidad y robustez. Además, para tener un sistema en tiempo real y un multiprocesamiento en los 2 núcleos ESP32, se utiliza ESP-IDF FreeRTOS. La programación de los módulos ESP32 se realiza mediante el Entorno de desarrollo integrado (IDE) de *Arduino*, por su facilidad y el contenido disponible en la comunidad.

ESP-IDF FreeRTOS es un sistema operativo en tiempo real, que aprovecha los 2 núcleos idénticos del ESP32 para realizar *Multiprocesamiento simétrico* (SMP). El núcleo conocido como CPU0, se recomienda que esté a cargo de las redes inalámbricas, mientras que el núcleo denominado como CPU1 se encarga del resto de las tareas [64].

Para el sistema propuesto, en los nodos sensores se crearon las 2 tareas que se describen en la Tabla 3.2:

Tabla 3.2: Tareas asignadas a cada núcleo del ESP32

Núcleo	Tarea	Descripción
CPU0	vTaskTbN	Se encarga de la conexión con el servidor de Thingsboard. Lectura y envío de los datos guardados en la memoria microSD
CPU1	vTaskESP_N	Gestiona el protocolo ESP-NOW. Lectura de los datos inerciales del sensor MPU6050. Almacenamientos de los datos inerciales en la memoria microSD.

3.4.1. Comunicación

Como se muestra en la Figura 3.1, el sistema propuesto forma una red de tipo estrella y no de malla, ya que los nodos sensores requieren comunicarse únicamente con el nodo central y no

entre ellos. Por lo que el uso del protocolo ESP-NOW es de interés, ya que permite crear una arquitectura que puede adaptarse a lo planteado.

Además, en [65], los autores realizan una comparativa del retardo de transmisión de un mensaje utilizando ESP-NOW y *User datagram protocol* (UDP). Luego de realizar el envío de 5000 paquetes utilizando los módulos de desarrollo ESP8266, a una frecuencia de trabajo de 80MHz y 160MHz, se concluye que el protocolo ESP-NOW entrega mejores resultados con una precisión de 510ms, por lo que es de esperar que con ESP32 se obtengan resultados similares debido a sus tecnológicas semejanzas.

Para la comunicación bidireccional entre el nodo central y cada uno de los nodos sensores por medio del protocolo ESP-NOW se realiza lo siguiente:

- Los nodos sensores se vinculan con el nodo central mediante la MAC del nodo central.
- De manera similar, el nodo central se vincula a cada nodo sensor conociendo la MAC de cada nodo sensor.
- Se configura la función de la devolución de la llamada, para conocer el éxito del envió de cada uno de los mensajes.

En el Anexo **??** se muestra la estructura del mensaje ESP-NOW que se intercambia entre los nodos sensores y el nodo central para dar a conocer su estado. También, se indica la instrucción *broadcast* que envía el nodo central para indicar a los demás nodos el inicio de la captura de datos.

3.4.2. Mecanismo de sincronización

Para conectar los distintos nodos a internet se utilizan los datos móviles del teléfono celular con una conexión *hotspot*, lo cual facilita el uso del sistema en lugares en donde no se cuente con puntos de acceso (AP) de internet fijo. Este celular es el mismo en donde se instaló la aplicación móvil de control. Las credenciales de acceso a los datos móviles están agregadas previamente en cada uno de los nodos.

En la Figura 3.16 se ilustra la secuencia que sigue el nodo central para sincronizar la toma de datos en los nodos sensores. A continuación se indica dicha secuencia:

1. El nodo central consulta si nodos sensores se encuentren activos y listos para recibir cualquier instrucción.

2. Si todos los nodos se encuentran dispuestos, el nodo central envía un mensaje RPC al servidor notificando que el sistema esta listo. Caso contrario, se vuelve a validar que todos los nodos se encuentren conectados.

3. Se espera la solicitud RPC de inicio por parte del servidor. Cuando recibe la instrucción de iniciar, el nodo central envía un mensaje ESP-NOW a todos los nodos para comenzar a capturar los datos.

4. Mientras los nodos sensores trabajan, el nodo central se mantiene atento al servidor.

5. Si recibe la orden de finalizar, el nodo central envía nuevamente un mensaje ESP-NOW a los nodos sensores.

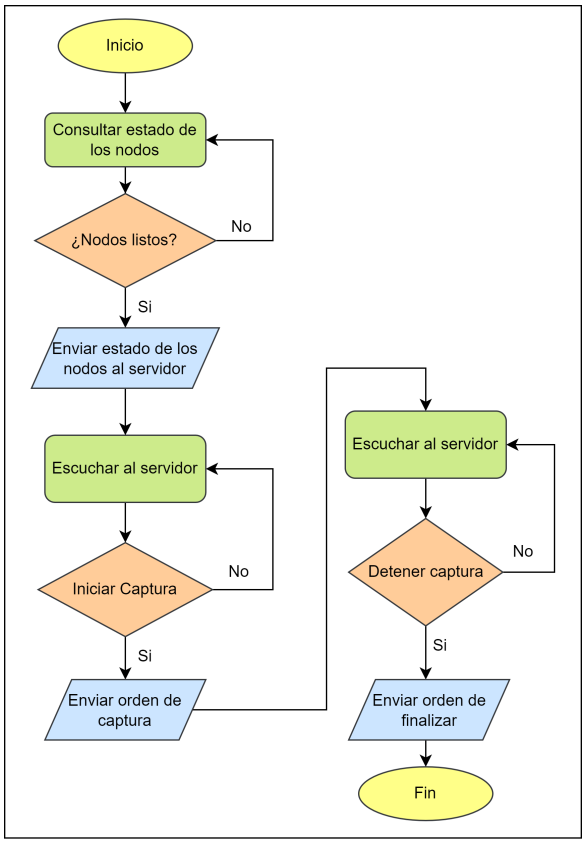


Figura 3.16: Secuencia lógica de la sincronización en el nodo principal.

La Figura 3.17 representa el momento en el que el nodo central envía un mensaje *broadcast* a los nodos sensores. El nodo central consulta la función de devolución de llamada del mensaje enviado y verifica que el mensaje haya sido entregado a todos los nodos; caso contrario, no se realiza la captura de los datos.

En la Figura 3.18 se presenta la secuencia lógica que sigue cada uno de los nodos sensores:

1. Una vez que el nodo sensor verifica que el módulo MPU6050 y la tarjeta microSD inicia-

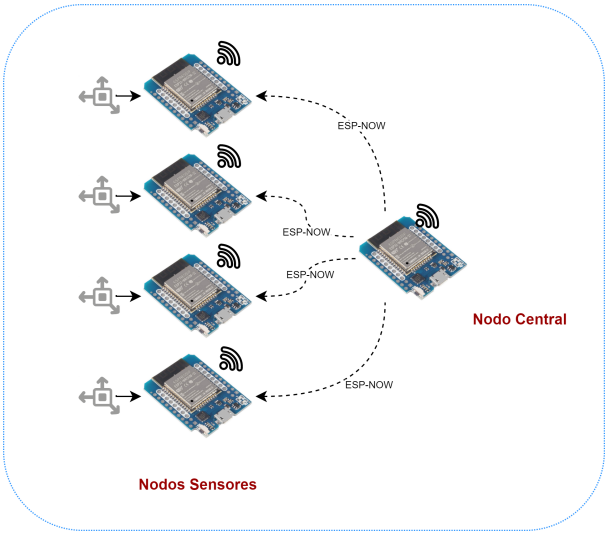


Figura 3.17: Sincronización de los nodos sensores desde el nodo central.

ron correctamente; este informa al nodo central mediante un mensaje ESP-NOW que se encuentra listo.

2. El nodo sensor se mantiene escuchando al nodo central. En cuanto reciba la orden de iniciar, el nodo consulta el tiempo actual en un servidor *Network Time Protocol* (NTP) y lo almacena en una variable `t` en forma de bytes.

3. El nodo empieza a capturar el evento de las aceleraciones en los ejes `x`, `y`, `z` del sensor MPU6050.

4. Almacena las aceleraciones en la memoria microSD en un arreglo de bytes.

5. Verifica si en el nodo central solicita detener la captura; caso contrario, espera que haya transcurrido 20ms y vuelve a tomar el siguiente evento de aceleraciones. Cuando recibe la orden de finalizar, el nodo termina la adquisición de datos y procede al envío de los mismos al servidor.

6. En el proceso de envío, a cada conjunto de datos a enviar se concatena una marca de tiempo relacionada con la variable `t` (*timestamp*).

El servidor NTP utilizado es: `pool.ntp.org`, mismo que se encuentra constituido por un enorme clúster virtual con más de 4000 servidores de tiempo NTP alrededor del mundo, lo cual garantiza su disponibilidad [66]. Si se llega a presentar problemas de disponibilidad del servidor, los nodos del sistema no podrían enviar los datos de aceleración recolectados a *Thingsboard*, pero la probabilidad de que esto suceda es prácticamente nula.

Los patinadores de velocidad tienen una resistencia a la fuerza G superior a la media, experi-

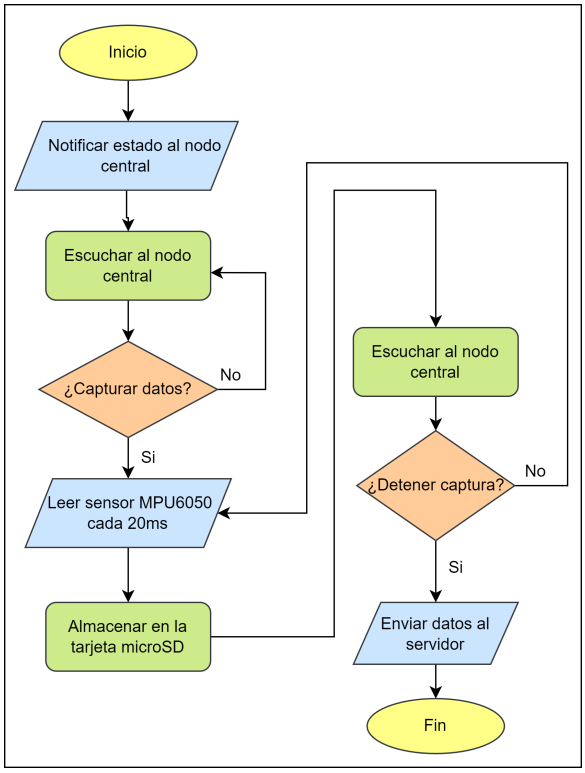


Figura 3.18: Secuencia lógica de la toma de datos en los nodos sensores.

mentando aceleraciones de hasta 2.6G. Sobre todo en los giros, que es donde deben luchar contra la fuerza centrífuga. Lo que les lleva a cambiar su técnica de patinaje en esas partes de la pista [67]. Es por esta razón que en la programación de los nodos, se ha configurado al sensor MPU6050 con una sensibilidad de 4G, que es el valor más cercano al valor máximo que experimentarían los patinadores.

En la fase de empuje del patinaje sobre ruedas se tiene una frecuencia de en torno a los $85 - 90$ empujes por minuto [68]; lo que se traduce en un máximo de 2 zancadas por segundo, o 1 zancada por extremidad. Si, por el teorema de Nyquist, se muestrea al doble de esta frecuencia, se requeriría de al menos 2 muestras por segundo para capturar dichos movimientos en cada extremidad sin inconveniente. Por tanto, obtener los datos de aceleración cada 20ms (o en su defecto, 50 muestras por segundo) brindaría la información suficiente para estudiar el desempeño más relevante del deportista.

Adicionalmente, en [69] se menciona que gran parte de los sensores *Multiple Process Unit (MPU)* utilizados en los sistemas de medición de la literatura revisada, presentan una tasa predefinida de 30 muestras/segundo, por lo que la velocidad de muestreo utilizada en el presente proyecto se encuentra por encima de la velocidad de muestreo promedio en sistemas de medición semejantes.

3.5. Configuración del servidor Web

Con el objetivo de almacenar en la nube los datos de aceleración del patinador, se utiliza la plataforma IoT de código abierto *Thingsboard Cloud*. Esta plataforma permite que la información almacenada sea consultada a través de una RPC, habilitando características específicas de casos de uso mediante cadenas de reglas personalizables [43]. Todo esto hace posible comunicar los dispositivos con la aplicación móvil, por lo que es ideal para ser usada dentro del presente proyecto. *Thingsboard Cloud* es una herramienta administrada, escalable y tolerante a fallos que permite aprovechar todos los beneficios de esta plataforma sin la necesidad de alojar en ella una instancia propia, por lo que el proceso de montaje del servidor se facilita en gran medida.

En el Anexo 5.3.1 se indican los aspectos más importantes de la configuración de los dispositivos dentro del servidor Web. Por su parte, en la sección 3.5.1 se describe la estructura de los paquetes de datos enviados desde los nodos ESP32 al servidor mediante el protocolo MQTT. También, en la sección 3.5.2, se exponen las cadenas de reglas creadas para los diferentes procesos de almacenamiento y consulta de datos.

3.5.1. Estructura de paquetes enviados desde ESP32 a *Thingsboard*

En cada trama MQTT se envía un conjunto de 156 bytes codificados en *base64*. Cada mensaje codificado que envía el nodo sensor al servidor Web, contiene 6 bytes sin codificar de la marca de tiempo y 150 bytes sin codificar de las aceleraciones, como se exhibe en la Figura 3.20. Los 150 bytes están conformados por 25 muestras de 6 bytes pertenecientes a las aceleraciones en los 3 ejes: x, y, z. La aceleración de cada eje está representada en 2 bytes, como se indica en la Figura 3.21. El tamaño del mensaje codificado en *base64* es de 208 bytes; considerando 5 bytes adicionales de la cabecera fija de la trama (1 byte para código de control y 4 bytes para longitud del mensaje [70]) se tiene un paquete resultante de 213 bytes, tal y como se ilustra en la Figura 3.19. El uso de la codificación *base64* asegura que los datos binarios puedan ser transmitidos intactos usando el formato Json [71].

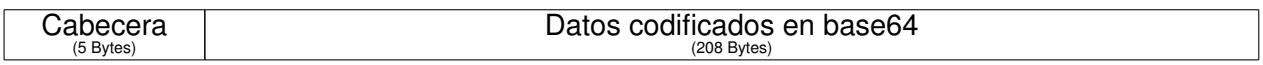


Figura 3.19: Trama MQTT enviada al servidor

El envío de los datos capturados, junto con la marca de tiempo por paquete enviado, se lo realiza desde cada nodo sensor. En el Anexo 5.3.2, se indica la programación requerida en

Timestamp (6 Bytes)	Datos inerciales (150 Bytes)

Figura 3.20: 156 Bytes conformados por el *Timestamp* y los datos inerciales

ax_1 (2 Bytes)	ay_1 (2 Bytes)	az_1 (2 Bytes)	...	ax_{25} (2 Bytes)	ay_{25} (2 Bytes)	az_{25} (2 Bytes)

Figura 3.21: 150 Bytes correspondientes a las aceleraciones: ax, ay y az.

los ESP32 para enlazar cada nodo con *Thingsboard Cloud*.

3.5.2. Motor de reglas

En el presente proyecto se utiliza un motor de reglas para configurar las diferentes acciones de almacenamiento y consulta de datos en el servidor. Se consideran tres cadenas de reglas: *Root RPC*, *Order Reply* y *Nodes State*, tal y como se indica en la Figura 3.22. La cadena de reglas *Root RPC* funciona como raíz y es en ella donde se configuran todos los flujos de trabajo a realizar a partir de los mensajes entrantes. `Order Reply` contiene los nodos de regla para notificar al dispositivo cliente que el envío de la orden para iniciar o finalizar la captura de los datos se ha realizado correctamente. De igual manera, `Nodes State` permite responder al dispositivo cliente cuando este solicita conocer el estado del sistema.

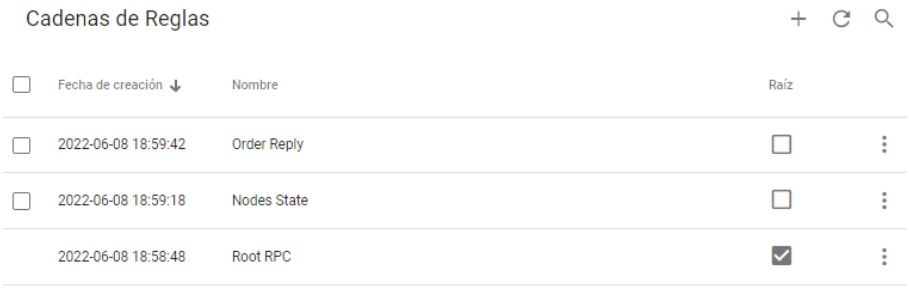


Figura 3.22: Cadenas de reglas creadas

3.5.2.1. Cadena de reglas *Root RPC*

Esta es la cadena de reglas raíz en donde se procesan los paquetes de datos entrantes. En la Figura 3.23 se indican los nodos de regla involucrados, mismos que se describen a continuación:

- **Message Type Switch:** enruta los mensajes entrantes de acuerdo al tipo de mensaje [72]. En este caso, son de interés los mensajes del tipo *Post Telemetry*, ya que estos

corresponden a los paquetes de datos de aceleración enviados desde los dispositivos para ser almacenados en el servidor.

- **JsonTelemetry1:** este *script* contiene una función *JavaScript* que cambia la carga útil del mensaje y el metadato de la marca de tiempo para los datos de aceleración. Es decir, a cada paquete que llega al servidor con varios datos de aceleración codificados, los decodifica, les asigna una marca de tiempo y los almacena en el servidor con su respectiva clave de identificación (`ax`, `ay`, `az`). Adicionalmente, calcula y almacena el módulo de aceleración (`a`) a partir de sus respectivas componentes. El *script* en cuestión se explica con mayor detalle en el repositorio de [63].
- **Save:** almacena en la *Data Base* (DB) los datos de series temporales de los mensajes entrantes y los asocia a la entidad correspondiente. Este nodo dispone del parámetro *Time to live* (TTL), en donde se configura el tiempo en segundos para la expiración de los datos de series temporales [73].
- **Log Other:** registra los mensajes entrantes que sean de un tipo diferente a *Post Telemetry*.
- **start, stop y Ready:** Evalúa el mensaje entrante cuando se trata de una solicitud RPC del dispositivo. Estos *scripts* son utilizados para filtrar la orden de `start` y `stop` en la toma de datos de aceleración, o para conocer el estado de los nodos a través del método `Ready` ; estos métodos llegan desde la aplicación móvil. El *script* devuelve un valor booleano, si es verdadero envía el mensaje a través de la cadena `True` [72].
- **Response Rule:** Reenvía el mensaje con el método `start` o `stop` a la cadena de reglas *Order Reply*.
- **Ready Rule:** Reenvía el mensaje con el método *Ready* a la cadena de reglas *Nodes State*.

3.5.2.2. Cadena de reglas *Nodes State*

Esta cadena de reglas gestiona las peticiones de inicio y final de la captura de los datos de aceleración en los dispositivos mediante mensajes de entrada con el método `start` o `stop`, respectivamente; está compuesto de tres nodos de regla, los cuales se describen a continuación:

- **Related State:** Agrega el valor de telemetría más reciente de la clave booleana `state`

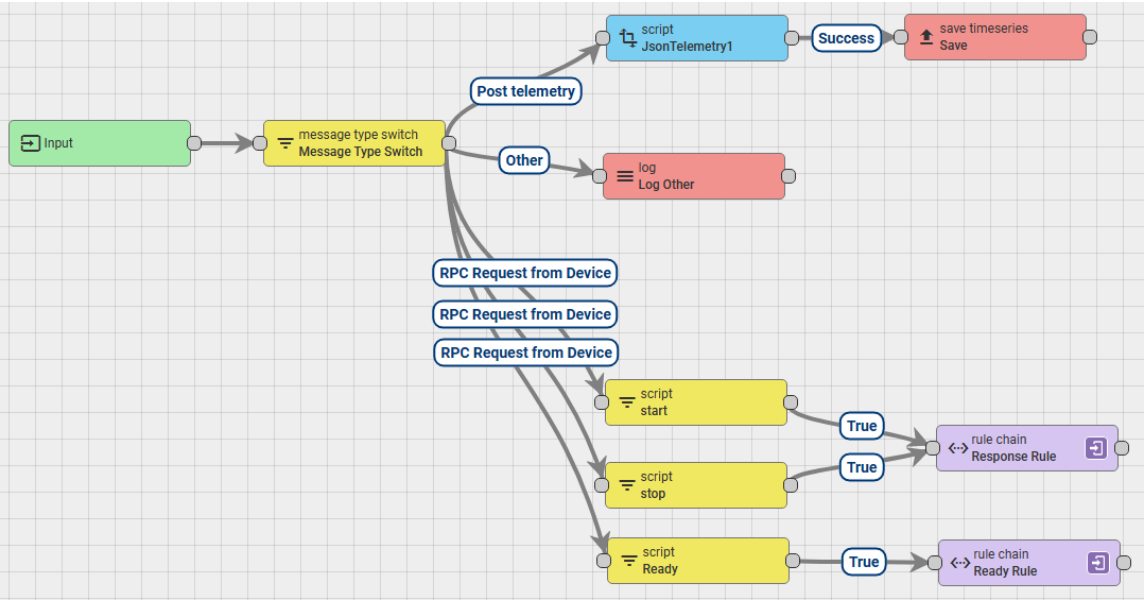


Figura 3.23: Cadena de reglas *Root RPC*.

a los metadatos del mensaje; para ello se debe marcar la casilla de verificación: `Latest telemetry`, así como seleccionar la dirección y nivel de profundidad de la relación con los dispositivos [74], tal y como se observa en la Figura 3.24. De esta manera, a través del dispositivo `Controller A`, se puede saber si los nodos sensores están listos para la captura de datos.

Figura 3.24: Configuración del nodo *Related State*

- **build reply:** Mediante este *script*, una vez consultado el último valor de telemetría de la variable `state`, se responde a la aplicación móvil con la clave booleana `isReady` de acuerdo al estado actual de los nodos sensores.
- **send response:** Envía la respuesta al originador de la llamada RPC [73]; es decir, a la aplicación móvil.

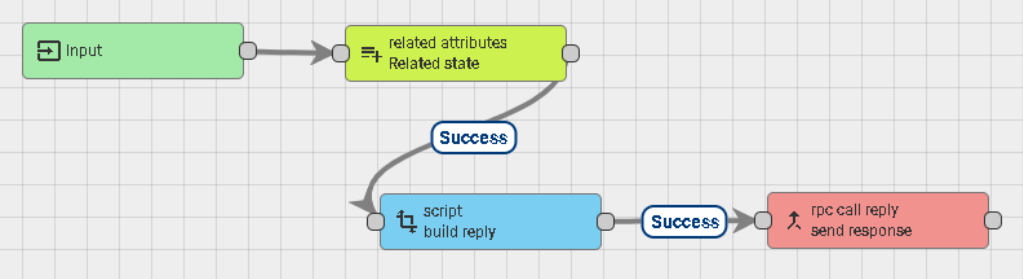


Figura 3.25: Cadena de reglas *Nodes State*

3.5.2.3. Cadena de reglas *Order Reply*

Mediante esta cadena de reglas se gestiona la orden de inicio y fin de la captura de los datos de aceleración en los diferentes nodos sensores. Para ello, se utilizan los cuatro nodos descritos seguidamente:

- **Start Stop:** Mediante este *script*, cuando la aplicación móvil envía al servidor la orden de iniciar o finalizar la captura de datos, este último genera un mensaje con la clave `setOrder` y el valor `1` o `2`, dependiendo de si la orden es `start` o `stop` respectivamente.
- **RPC Call Request:** Envía las solicitudes RPC de `start` y `stop` al dispositivo central. La configuración del nodo tiene el campo: `Timeout`, en donde se especifica el tiempo de espera de respuesta del dispositivo, por defecto se configura en 60 segundos [73].
- **build reply:** Este *script* genera un mensaje de confirmación (`OK`) de que la orden ha llegado correctamente al dispositivo central.
- **send response:** Envía la respuesta de confirmación al originador de la llamada RPC; es decir, a la aplicación móvil.

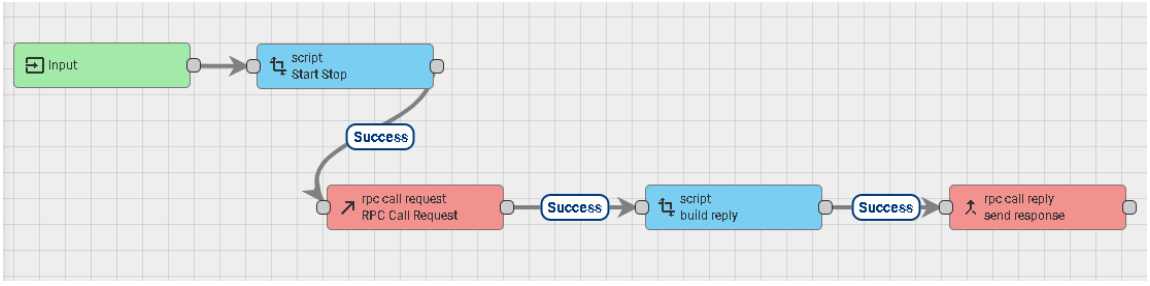


Figura 3.26: Cadena de reglas *Order Reply*

3.6. Aplicación móvil

Con el objetivo de comandar la captura de los datos de aceleración en los diferentes nodos sensores y visualizar dicha información, se ha desarrollado una aplicación móvil para *Android*, utilizando el *Software development Kit* (SDK) *Flutter*. La aplicación móvil trabaja en conjunto con el servidor de *Thingsboard*, siguiendo la lógica presentada en el diagrama de la Figura 3.27.

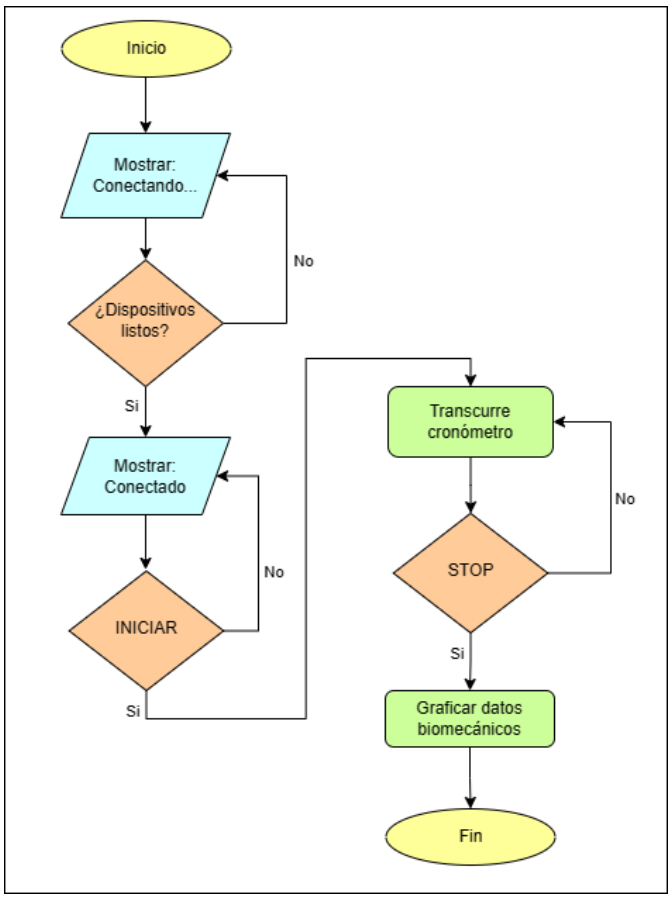


Figura 3.27: Secuencia lógica de la aplicación móvil desde el punto de vista del usuario

De acuerdo al diagrama de la Figura 3.27, se considera la siguiente secuencia desde el punto de vista del usuario:

1. Al iniciar la aplicación, se consulta en el servidor el estado de los dispositivos. También se presenta el mensaje: `conectando...`.
2. Si los nodos del sistema están listos, se cambia el mensaje de la pantalla a: `conectado`, y se permite al usuario iniciar la captura de los datos de aceleración.
3. Al dar clic en el botón INICIAR se envía una orden al servidor para comenzar la captura de los valores de aceleración. También se indica en la pantalla un cronómetro con el

tiempo de acción transcurrido.

4. Al dar clic en el botón STOP se envía una orden al servidor para finalizar la captura de los valores de aceleración.

5. Dependiendo de las componentes de aceleración (`ax`, `ay`, `az`) que el usuario seleccione en la aplicación, se realiza una petición al servidor de los valores respectivos.

6. Se grafica los datos de aceleración y distancia correspondientes.

Para efectuar en la aplicación móvil las tareas de: consultar el estado de los nodos del sistema, comandar la captura de datos inerciales y obtener los datos de aceleración desde el servidor, se utilizan solicitudes RPC con API REST. El proceso seguido por la aplicación móvil para llevar a cabo las tareas antes mencionadas se indica en la Figura 3.28.

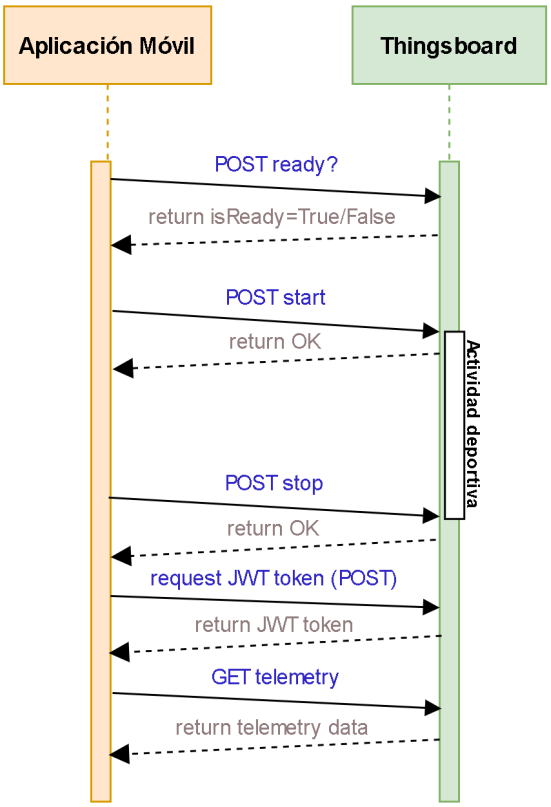


Figura 3.28: Proceso de solicitudes RPC ejecutadas desde la aplicación móvil

De acuerdo con la Figura 3.28, el proceso de solicitudes RPC se explica a continuación:

1. Inicialmente, se envía un mensaje del tipo POST con el método `ready` para consultar el estado de los nodos del sistema; entonces, el servidor retorna un mensaje con una notificación sobre el estado de los mismos. Esta solicitud realiza la aplicación de manera recurrente hasta que reciba una notificación de que el sistema está listo.

2. Si el servidor responde con: `isReady = True`, es posible iniciar la captura de los datos. Para esto, se envía una solicitud del tipo POST con el método `start`. El servidor responde con un `OK` si el mensaje fue recibido correctamente.

3. Para finalizar la captura de datos, se envía un mensaje semejante con el método `stop`, igualmente el servidor responde con un `OK` si la recepción fue exitosa.

4. Una vez concluida la actividad deportiva, para consultar los datos de aceleración almacenados en el servidor se requiere de un token *JSON Web Token* (JWT) temporal de seguridad. Este token se obtiene mediante un mensaje del tipo POST con las credenciales de usuario de *Thingsboard*. Si las credenciales son correctas, el servidor responde con dicho token.

5. Finalmente, para obtener los datos de telemetría y graficarlos en la aplicación móvil, se envía al servidor un mensaje del tipo GET, en donde se especifica las componentes de aceleración requeridas y el token JWT obtenido en el paso anterior.

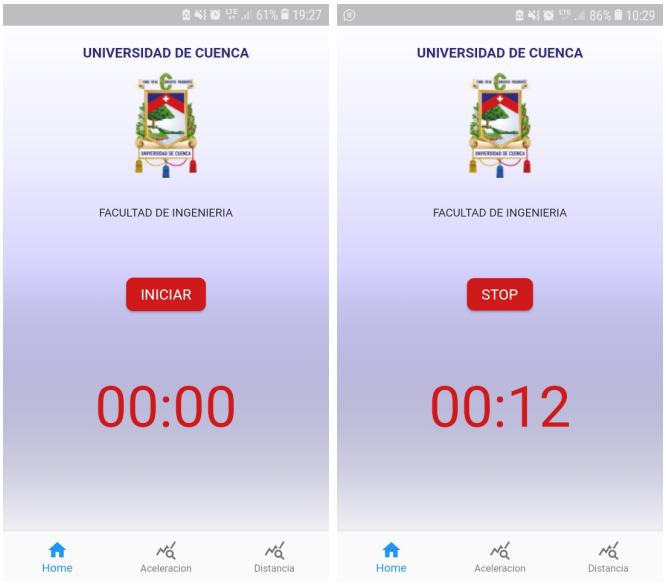
En el Anexo 5.4.1 se detallan las solicitudes RPC enviadas al servidor para ejecutar las tareas requeridas. Por su parte, en la sección 3.6.1 se muestran las diferentes pantallas consideradas en la interfaz gráfica.

3.6.1. Interfaz gráfica

La interfaz gráfica consta de 3 pantallas. En la pantalla *Home*, se dispone de un botón para iniciar o finalizar la captura de datos y un cronómetro que indica el tiempo que ha transcurrido desde el inicio de la actividad deportiva. En la Figura 3.29a se muestra el botón INICIAR y el cronómetro en cero, mientras que en la Figura 3.29b dicho botón aparece como STOP y el tiempo del cronómetro, a manera de ejemplo, ha avanzado 12 segundos.

La segunda pantalla permite visualizar los datos inerciales de aceleración del patinador. De acuerdo con la Figura 3.30, en la parte superior se puede seleccionar la extremidad de la que se quiere obtener la gráfica de aceleración, es posible seleccionar más de una extremidad. Del mismo modo, en la parte inferior de la pantalla se selecciona visualizar una de las componentes de aceleración o el módulo de la aceleración. Estos datos pueden ser visualizados una vez que se termine la actividad deportiva.

En la tercera pantalla de la aplicación, misma que se presenta en la Figura 3.31, se visualizan los datos de desplazamiento o distancia recorrida por cada una de las extremidades del



(a) (b)

Figura 3.29: Pantalla principal de la aplicación móvil

patinador, esta información se obtiene mediante la doble integración de los datos de aceleración utilizando el método del trapecio. La interacción del usuario con la interfaz es semejante a lo indicado anteriormente para graficar las curvas de aceleración. Estos datos pueden ser visualizados una vez que se termine la actividad deportiva.

3.7. Protocolo de uso de los dispositivos

El manejo de los dispositivos presentado en la Figura 3.32, se describen a continuación:

1. Para empezar, se coloca cada uno de los nodos en las extremidades del deportista, y el nodo central en el torso. Estos nodos son ajustados al cuerpo con una cinta velcro y deben estar orientados con el terminal blanco hacia arriba.

2. Posteriormente, se enciende cada uno de los dispositivos empezando por el nodo central ubicado en el torso. Los nodos se comunican entre ellos automáticamente y la conexión a internet también se realiza de manera automática utilizando los datos móviles de un teléfono celular mediante conexión *Hotspot*.

3. Mediante el uso de la aplicación, se verifica que los nodos estén preparados para adquirir los datos. Esto es posible mediante la visualización del botón `INICIAR`, como se ilustra en la Figura 3.29a.

4. El patinador debe colocarse dentro de la pista y alistarse para realizar la actividad de-

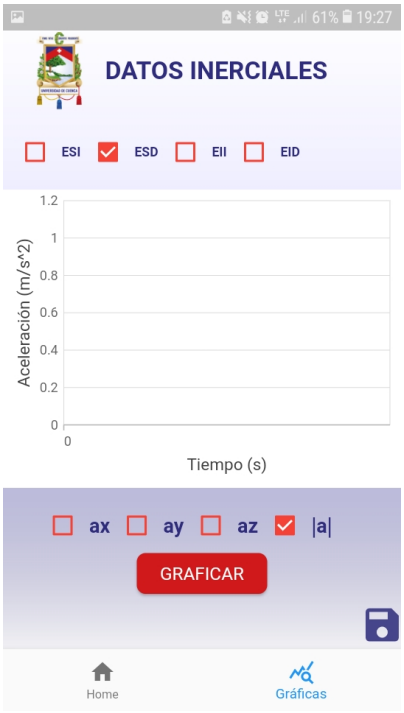


Figura 3.30: Pantalla para visualizar datos de aceleración

portiva.

5. Por medio de la aplicación, se envía a los nodos sensores la orden de iniciar.
6. El deportista realiza la actividad física y todos estos datos son recolectados en cada uno de los dispositivos.
7. Una vez que el deportista finalice su actividad, mediante la aplicación se termina la adquisición de datos y se espera a que la información recolectada se almacene en el servidor.
8. En la aplicación se visualizan las aceleraciones o distancias, seleccionando la extremidad deseada y presionando el botón `GRAFICAR`.

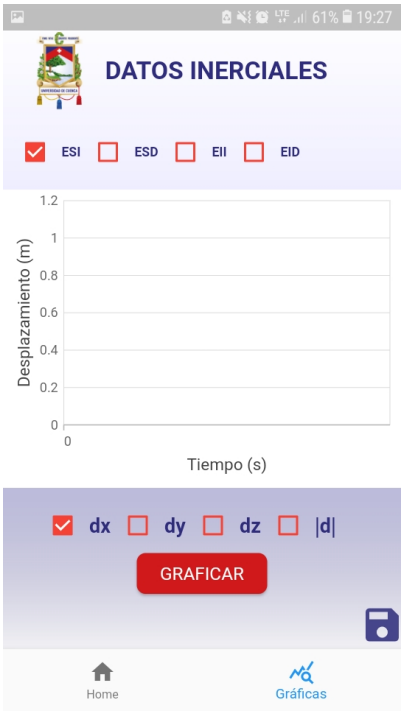


Figura 3.31: Pantalla para visualizar datos de desplazamiento

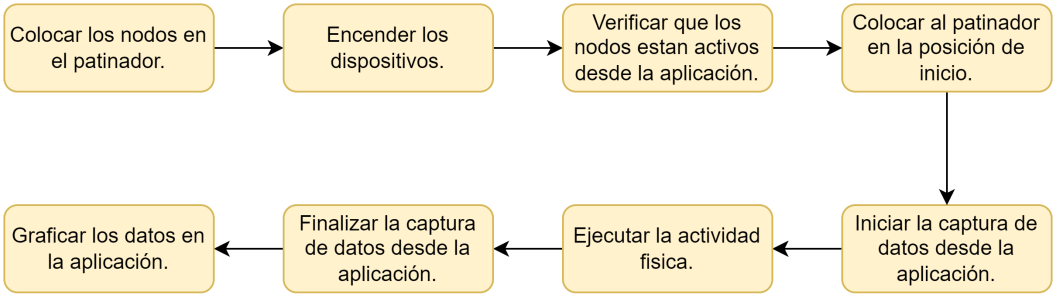


Figura 3.32: Protocolo de uso del sistema

Resultados

4.1. Sincronización de los nodos sensores

Para analizar la precisión con la que el nodo central, mediante ESP-NOW, se envían mensajes a los cuatro nodos sensores. Las pruebas se realizan encendiendo leds aproximadamente cada $50ms$ y visualizando los pulsos de voltaje en un osciloscopio. Esto se evidencia en la Figura 4.1, en donde se observa como los nodos ESP32 accionan dicho pulso de una manera *quasi* síncrona en cuanto reciben el mensaje ESP-NOW. La señal de color verde corresponde al nodo especificado como 1, las señales de los nodos 2, 3 y 4 son las que le suceden a esta de arriba hacia abajo.

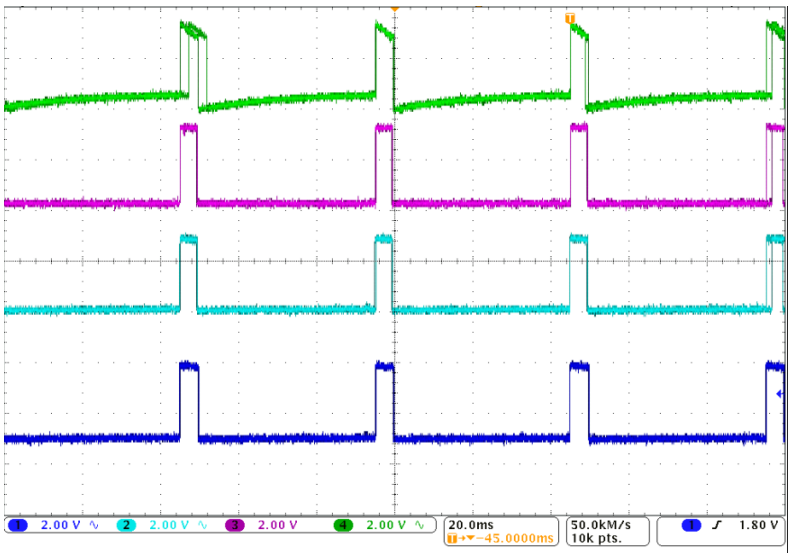


Figura 4.1: Pulsos de voltaje sincronizados en cuatro nodos ESP32

Es de interés que los flancos de subida de los pulsos se encuentren sincronizados lo mejor posible, ya que estos representan el momento en que la orden de iniciar o finalizar la captura de datos de aceleración se ejecuta en cada uno de los nodos sensores. En la Figura 4.2 se ve como el flanco de bajada de los pulsos varía de manera considerable entre las señales de los nodos; sin embargo, los flancos de subida no presentan mucha variación de tiempo, encontrándose la sincronización en una escala de μs.

La prueba indicada en la Figura 4.3 se enfoca en analizar la sincronización en tiempo de los flancos de subida de los pulso de voltaje de los cuatro nodos. La diferencia máxima de tiempo entre pulsos es menor a los 100μs.

Entre las diversas pruebas realizadas, el mayor tiempo de desfase entre pulsos encontrado fue de aproximadamente 406mus, tal y como se indica en la Figura 4.4a, para los nodos iden-

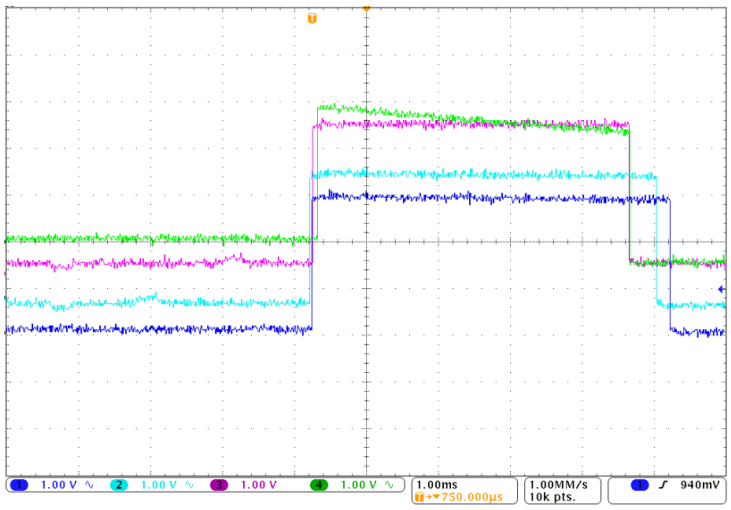


Figura 4.2: Sincronización en el flanco de subida de las señales de pulso

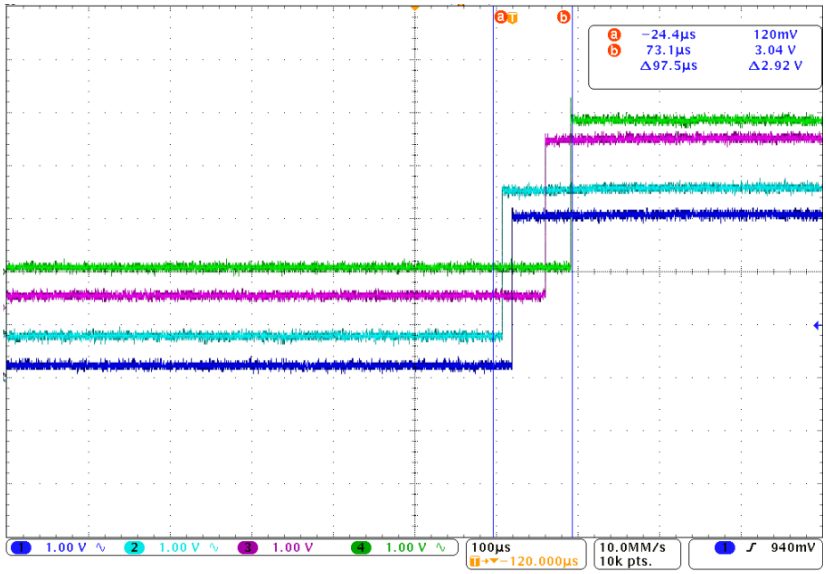
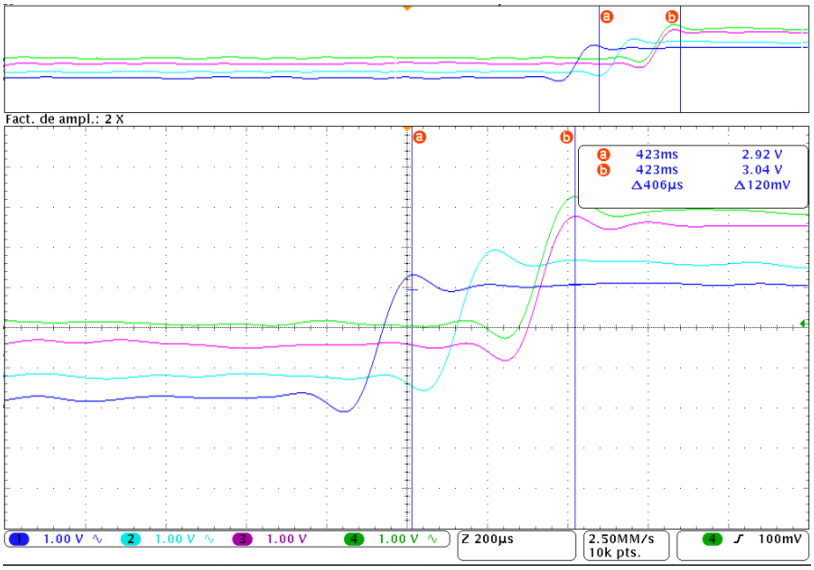


Figura 4.3: Diferencia de tiempo inferior a 100μs entre pulsos

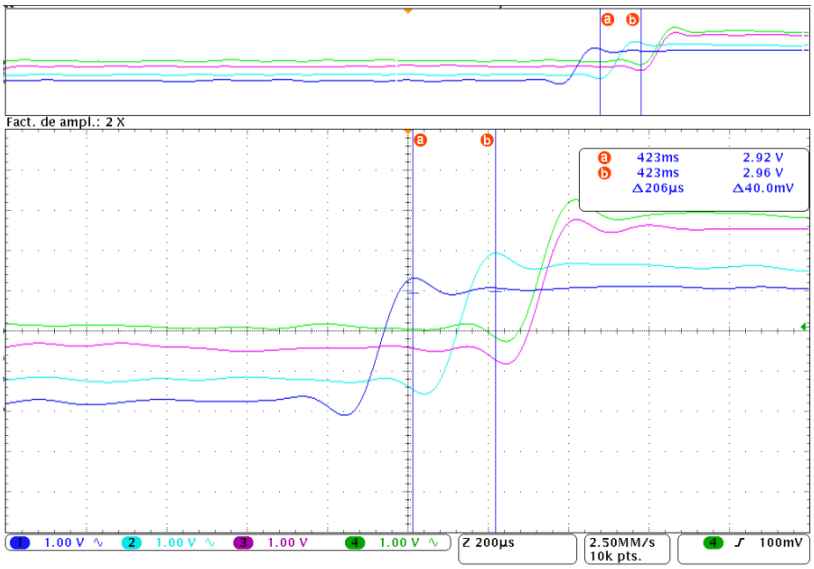
tificados como 1 y 4. Por su parte, en la Figura 4.4b se tiene que el tiempo de desfase entre el flanco de subida del pulso del nodo 3 y nodo 4 es de aproximadamente 200μs. La sincronización en la ejecución de las órdenes enviadas desde el nodo central a los demás nodos utilizando mensajes ESP-NOW tuvo una precisión de μs en todas las pruebas realizadas.

4.2. Envío de paquetes a Thingsboard

ThingsBoars Cloud ofrece planes de suscripción basados en el modelo de pago por uso. Para el presente proyecto se utiliza la suscripción ThingsBoard Cloud Maker, por lo que se tiene un límite de tarifa de mensajes de transporte de telemetría de hasta 10 mensajes por segundo, pero sin exceder los 300 mensajes por minuto y los 7000 mensajes por hora, esto



(a) Diferencia de tiempo de 406μs entre pulso del nodo 1 y nodo 4



(b) Diferencia de tiempo de 200μs entre pulso del nodo 3 y nodo 4

Figura 4.4: Máxima diferencia de tiempo encontrada entre pulsos de los nodos

para cada dispositivo por separado [75]. Es por esta razón que se ha decidido enviar los mensajes de telemetría cada **250 ms**, desde cada nodo ESP32 hacia el servidor; con un total de **240 mensajes** enviados por minuto. Respetando de esta forma el limite oficial impuesto por Thingsboard.

El tamaño de cada paquete enviado desde los nodos sensores al servidor de Thingsboard es de **208 bytes** de datos útiles, ya que al incrementar este tamaño los paquetes no fueron receptados correctamente por el servidor. La longitud de la carga útil de telemetría MQTT para la edición comunitaria de *ThingsBoard* se encuentra entre 100 bytes y 218 bytes aproximadamente, pudiendo modificarse esta restricción al cambiar el valor del parámetro `JSON_MAX_STRING_VALUE_LENGTH`

del archivo `thingsboard.yml` del servidor local; aunque algunos usuarios advierten que no es posible superar una longitud de alrededor de los 306 bytes de carga útil [76, 77]. En la edición de *ThingsBoard Cloud* no se puede realizar las modificaciones antes mencionadas, ya que se utiliza un servidor remoto con configuraciones limitadas.

4.3. Pruebas en patinadores

Para efectuar pruebas del sistema en casos reales, se tuvo la ayuda de tres patinadores con diferentes niveles de experiencia. Se tomaron muestras en el Patinódromo Totoracocha de la ciudad de Cuenca-Ecuador. Las tres personas realizaron el mismo recorrido con las mismas condiciones.

Para que las curvas de aceleración sean presentadas de manera suavizada, se realizó la interpolación de los datos exportados desde el servidor, con la ayuda del *Software* Matlab. En la Figura 4.5 se muestra la comparativa de un tramo de una curva de aceleración sin interpolación vs. la misma señal con interpolación, para la Extremidad superior izquierda (ESI) y Extremidad inferior izquierda (EII). Se puede ver como esta operación no altera el comportamiento de las señales adquiridas, por lo que se lo puede utilizar sin inconvenientes. Tanto en la aplicación móvil como en Matlab, se utiliza una interpolación *spline*.

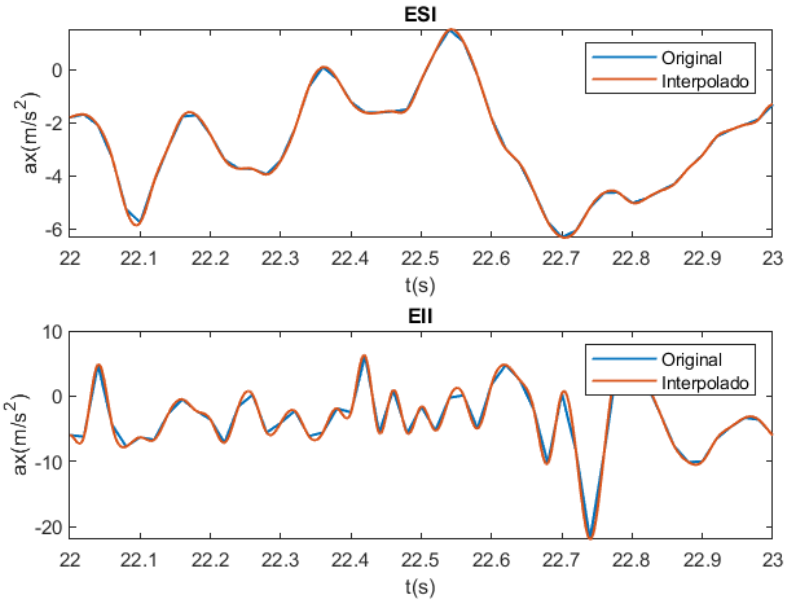


Figura 4.5: Curvas de aceleración con y sin interpolación

El deportista identificado como *"Patinador 1"* es el que se muestra en la Figura 4.6, y de los tres patinadores, es la persona con menor experiencia en este deporte. En dicha Figura,

se indica al patinador en reposo y en acción, con los nodos colocados en las extremidades respectivas, buscando que la ubicación de los dispositivos no afecte en la ejecución de los movimientos.

(a) En reposo (b) En acción

Figura 4.6: Patinador 1.

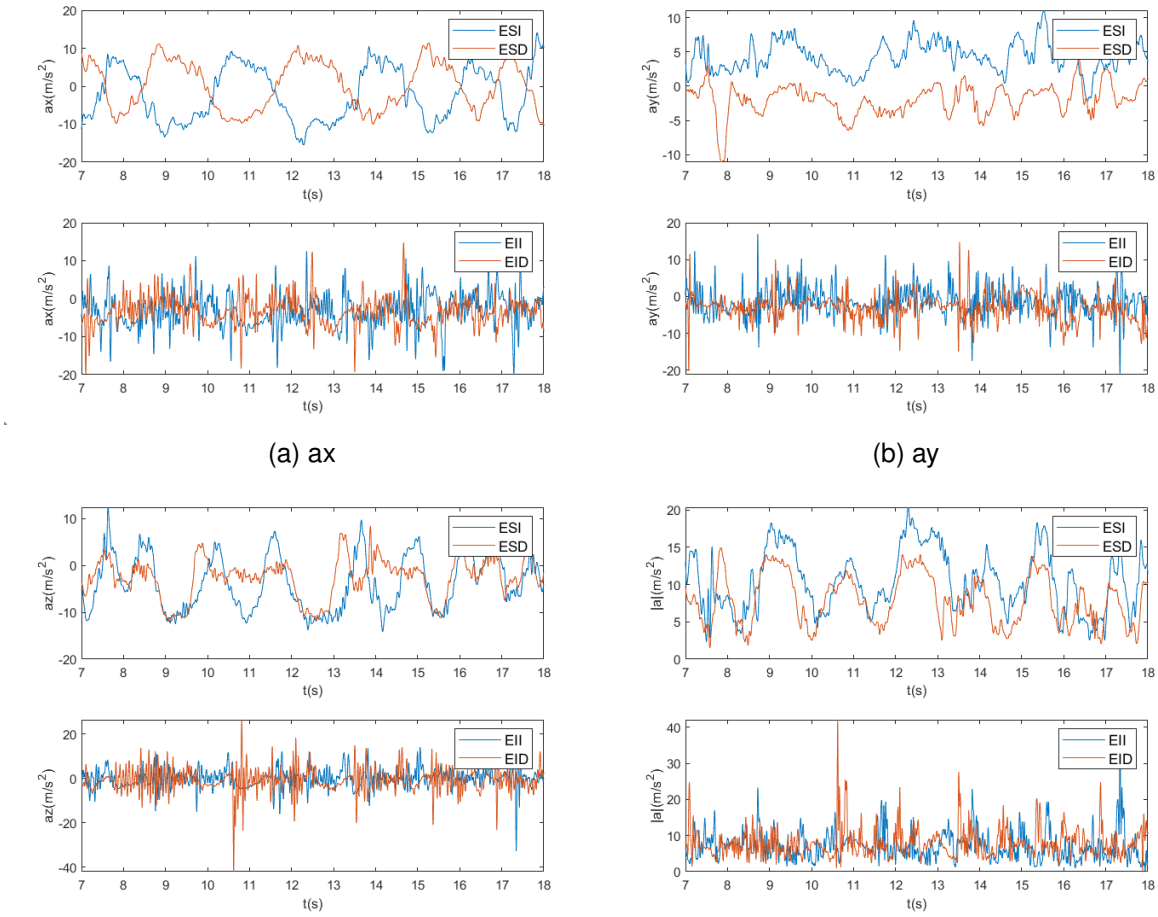
En la Figura 4.7a, 4.7b y 4.7c, se indican las aceleraciones de las cuatro extremidades en los ejes x, y y z respectivamente. Las mismas son generadas por el Patinador 1 en un trayecto en linea recta por la pista de patinaje, en un intervalo de 11 segundos. La Figura 4.7d corresponde al modulo de la aceleración.

Del mismo modo, en la Figura 4.8a, 4.8b y 4.8c se indican las aceleraciones adquiridas en los ejes y, y y z respectivamente, para el Patinador 1 en un tramo curvo de la pista de patinaje; se considera un intervalo de 6 segundos. La Figura 4.8d corresponde al modulo de las aceleraciones adquiridas en los tres ejes.

En la Figura 4.9 se indica al deportista identificada como *"Patinador 2"*, es una persona que tiene una experiencia intermedia en el patinaje. En la Figura 4.9a se muestra la ubicación de los nodos en el cuerpo de la patinadora, en la Figura 4.9b la deportista se encuentra en acción.

Las aceleraciones x, y y z, generadas por el Patinador 2 en un tramo recto de la pista de patinaje, se expone en las Figuras 4.10a, 4.10b y 4.10c respectivamente. La Figura 4.10d representa el modulo de las aceleraciones en los tres ejes. En las gráficas se considera un intervalo de tiempo de 10 segundos.

Para un tramo curvo de la pista, las aceleraciones x, y y z producidas por el Patinador 2, se ilustran en la Figura 4.11a, 4.11b y 4.11c; 4.11d es el módulo de la aceleración. Se considera



(a) ax (b) ay

(c) az (d) |a|

Figura 4.7: Aceleraciones del Patinador 1 en un tramo recto.

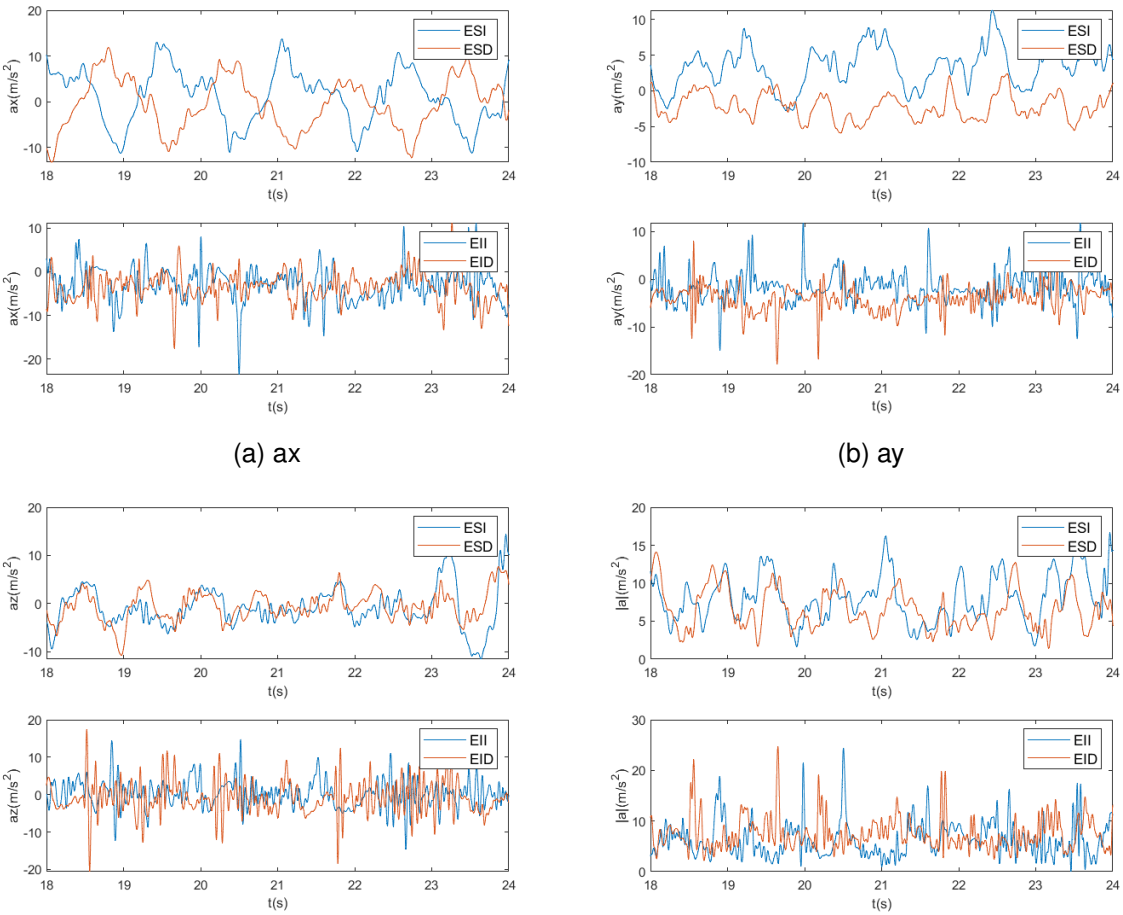
un intervalo de 4 segundos para las gráficas.

Del mismo modo, el deportista identificado como *"Patinador 3"* es el que se indica en la Figura 4.12 y tiene un nivel de experiencia superior a los dos patinadores anteriores.

En las Figuras 4.13a, 4.13b y 4.13c, se muestran las aceleraciones en x, y y z de las extremidades ESI, Extremidad superior derecha (ESD), EII y Extremidad inferior derecha (EID). Estas aceleraciones son tomadas en el tramo recto de la pista. El modulo de la aceleración se indica en la Figura 4.13d.

De manera similar, las aceleraciones de las cuatro extremidades adquiridas en un tramo curvo de la pista se ilustran en las Figuras 4.14a, 4.14b y 4.14c. En 4.14d se muestra el modulo calculado a partir de las aceleraciones en los tres ejes. En este caso se considera un intervalo de tiempo de 4 segundos para la visualización.

De acuerdo con las Figuras 4.7, 4.8, 4.10, 4.11, 4.13 y 4.14, cuando los valores de aceleración se encuentran por arriba de cero, se describe un movimiento de la extremidad en sentido



(a) ax (b) ay

(c) az (d) |a|

Figura 4.8: Aceleraciones del Patinador 1 en una curva

positivo, mientras que una aceleración por debajo de cero representa un movimiento en sentido negativo. Para comprender de mejor manera lo antes mencionado, en la Figura 4.15 se indica el eje de referencia definido para las aceleraciones del patinador, en donde se puede identificar los sentidos positivo y negativo de cada componente. Cabe mencionar que cuando la curva es creciente, se tiene un incremento de aceleración en la extremidad del patinador, mientras que un segmento de curva decreciente representa una desaceleración de dicha extremidad. Las curvas que corresponden a los valores de aceleración de las extremidades inferiores presentan mayor ruido debido a la vibración de los patines en la pista.

Finalmente, en las Figuras 4.16a, 4.16b y 4.16c y 4.16d se presentas la aceleraciones en el eje x, y y z, así como el módulo de la aceleración; graficadas en la aplicación móvil para la ESI y ESD del *Patinador 3* una vez finalizada su actividad deportiva. La curva de color rojo corresponde a la ESI y la curva de color verde corresponde a la ESD. De manera similar, en la Figura 4.25 se indican los componentes y el módulo de desplazamiento para la EII y EID del *Patinador 2*, siendo la curva de color azul la correspondiente a la EII y la de color negro a

(a) En reposo (b) En acción

Figura 4.9: Patinador 2

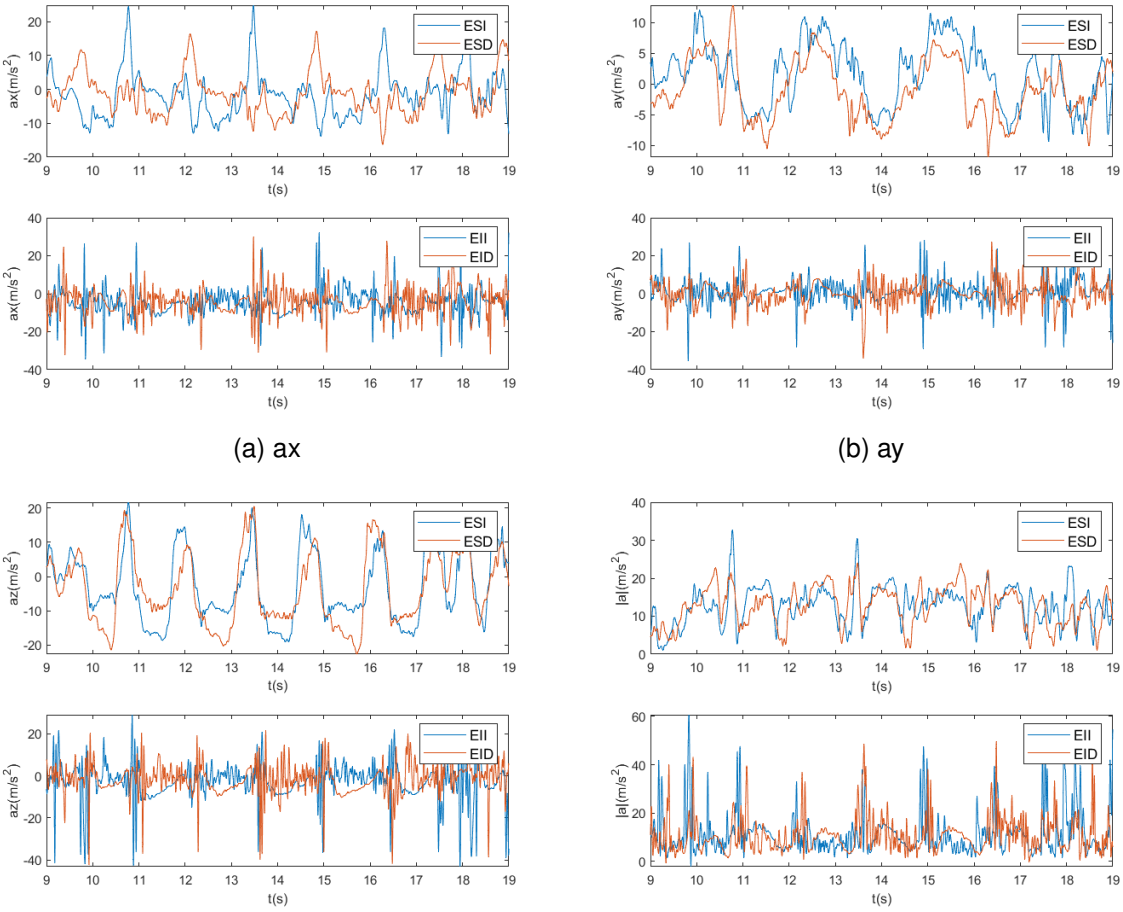
la EID. Estos valores se obtienen a partir de la doble integración de los datos de aceleración mediante el método del trapecio.

En las Figuras 4.7, 4.8, 4.10, 4.11, 4.13 y 4.14 se observa que los patinadores tienen movimientos rítmicos y simétricos de acuerdo a su técnica; así mismo, los patinadores con mayor experiencia presentan picos de aceleración mayores en sus movimientos. Ajustándose a los objetivos del presente proyecto, los datos se presentan en crudo, sin la aplicación de ningún filtro. En un estudio conjunto con un experto en biomecánica deportiva se podría realizar un procesamiento adicional de estos datos de acuerdo a las necesidades.

4.4. Pruebas con un péndulo físico

Como una manera de comprobar que los datos de aceleración obtenidos son correctos, se realizaron pruebas de movimiento pendular con los cuatro nodos sensores. El objeto utilizado para este fin se muestra en la Figura 4.18, está conformado por una barra metálica de `81cm` que cuelga de una base fija y cerca de su extremo inferior se ajusta el nodo sensor; siendo un sistema que correspondiendo a un péndulo físico.

De acuerdo con los ejes de referencia de la Figura 4.15, mismos que están definidos para los cuatro dispositivos sensores, se realizaron tres pruebas con uno de los nodos; estableciendo como coordenada normal del movimiento del péndulo a la coordenada `x`, la coordenada `y` y la coordenada `z` en cada una de las pruebas respectivamente. Con el objetivo de obtener señales de aceleración más limpias, se ha aplicado un filtro pasa bajos sobre las señales obtenidas



(a) ax (b) ay

(c) az (d) |a|

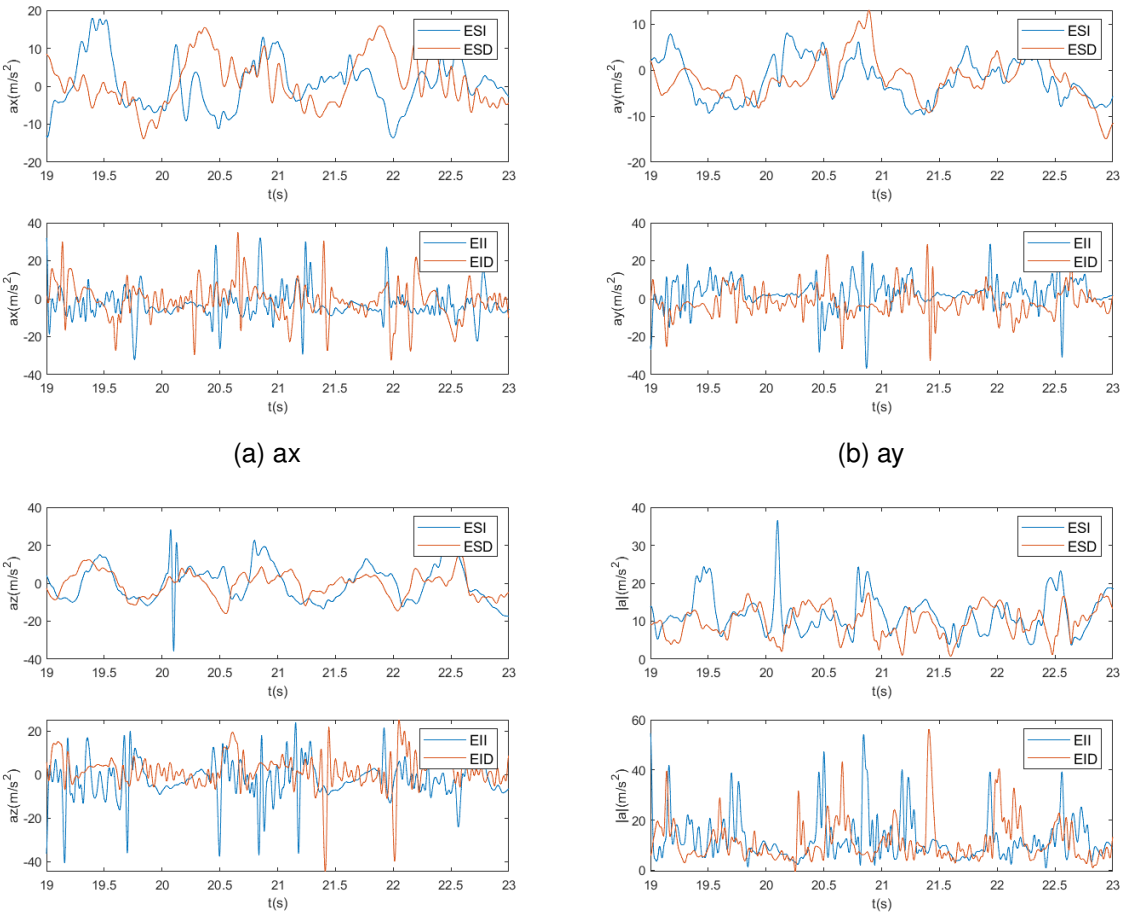
Figura 4.10: Aceleraciones del Patinador 2 en un tramo recto.

por el nodo sensor, utilizando el software de *Matlab*.

Un péndulo simple se comporta como un oscilador armónico cuando oscila con amplitudes pequeñas y si el ángulo de oscilación es pequeño (no más de 15º o 20º) [78]. Para pequeñas oscilaciones de un péndulo físico la amplitud es casi senoidal y el valor de su periodo `T` viene determinado por la expresión 4.1, donde `Icm` es el momento de inercia respecto al centro de masa, `L` es la longitud del péndulo, `l` la distancia desde el pivote hasta el centro de masa, `g` la aceleración de la gravedad y `m` la masa de la barra [79].

$$T = 2\pi\sqrt{\frac{I_{cm} + ml^2}{mgl}} \tag{4.1}$$

Considerando que el momento de inercia respecto al centro de masa de una barra es: $\frac{1}{12}mL^2$, el periodo `T` de la ecuación 4.1 se transforma en la expresión presentada en 4.2 [80].



(a) ax (b) ay

(c) az (d) |a|

Figura 4.11: Aceleraciones del Patinador 2 en una curva

$$T = 2\pi\sqrt{\frac{L^2 + 12l^2}{12gl}} \tag{4.2}$$

Como se observa en la Figura 4.19, cuando el péndulo oscila con un movimiento en x y z la aceleración del nodo muestra un comportamiento senoidal en dichos ejes, la componente y presenta únicamente ruido con valores cercanos a cero. Al ser un péndulo improvisado se presentas algunas distorsiones en las señales de aceleración, esto debido a pequeños errores en la dirección de movimiento del dispositivo y a la fricción del péndulo.

Del mismo modo, cuando el péndulo oscila en y y z (Figura 4.20), las componentes de aceleración son senoidales y en el eje x solo se muestra ruido. Para corroborar el comportamiento de las señales de aceleración se utilizó el acelerómetro de un *smartphone* mediante la aplicación *physics toolbox accelerometer*, colocándolo en la misma posición que el módulo sensor. Los

(a) En reposo (b) En acción

Figura 4.12: Patinador 3

resultados obtenidos son semejantes, tal y como se evidencia en la Figura 4.21.

Algo semejante sucede en la Figura 4.22, donde la aceleración se compone casi en su totalidad por los valores presentes en `z` y `x`. En `y` se tiene la presencia de básicamente ruido.

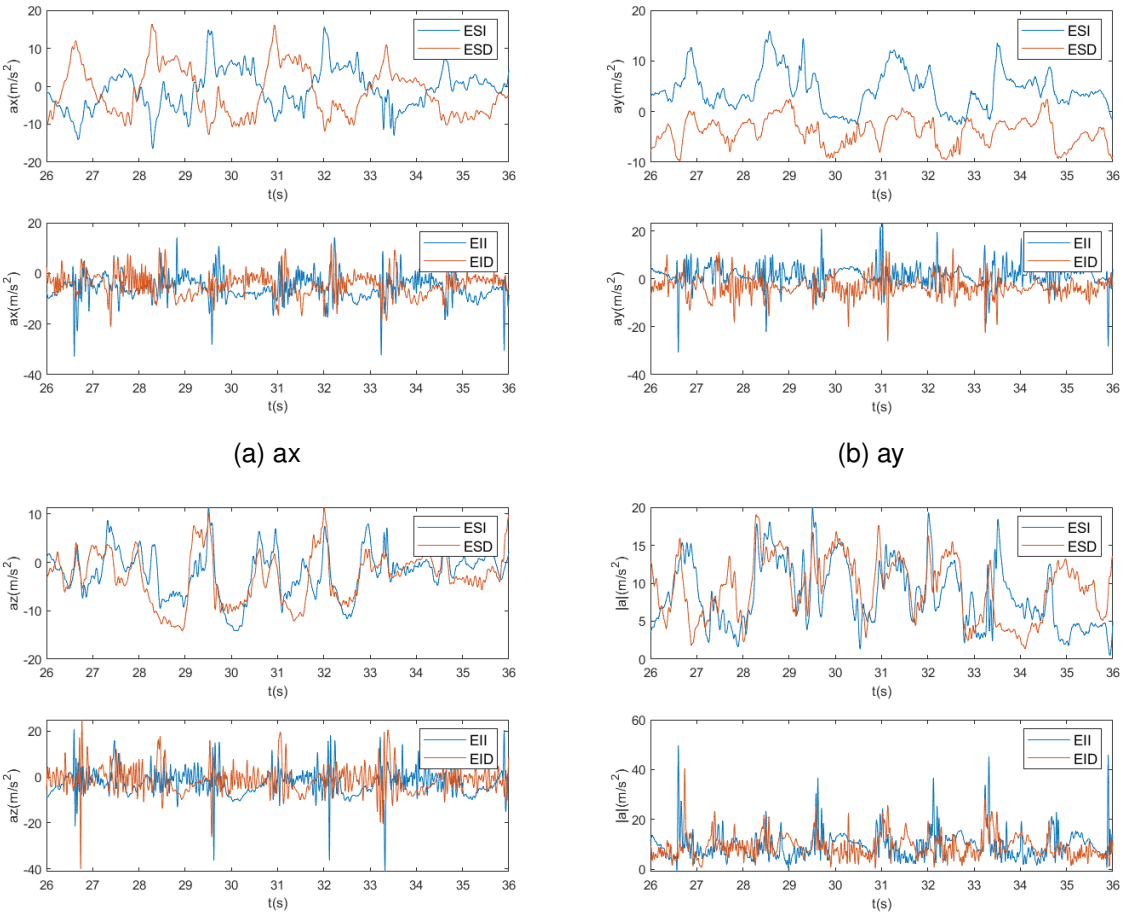
En la Figura 4.23 se indica un segmento de la curva de aceleración de las componentes en `x` y `z` obtenidas al oscilar los cuatro nodos sensores bajo las mismas condiciones. En la Figura 4.24 se aprecia de mejor manera como las señales de aceleración están sincronizadas en el tiempo.

En la Figura 4.25a y Figura 4.25b se indican las componentes de aceleración antes mencionadas para los cuatro nodos sensores, visto desde la aplicación móvil sin la aplicación de ningún filtro.

Asumiendo que el centro de masa del sistema se encuentra en donde se ha ubicado el nodo sensor, se tiene una distancia de `l=47cm`. Aplicando la expresión 4.2 en el péndulo físico utilizado se calcula un periodo de oscilación de `1.53s`; lo cual coincide aproximadamente con el valor de periodo de la componente tangencial `ax` mostrada en la Figura 4.23.

4.5. Aplicación en un exoesqueleto

Como un trabajo adicional para corroborar la utilidad multidisciplinaria que tiene el sistema propuesto, se colaboró dentro de un proyecto de maestría en donde se pretende replicar en un exoesqueleto los movimientos de las extremidades inferiores de una persona con el mínimo



(a) ax (b) ay

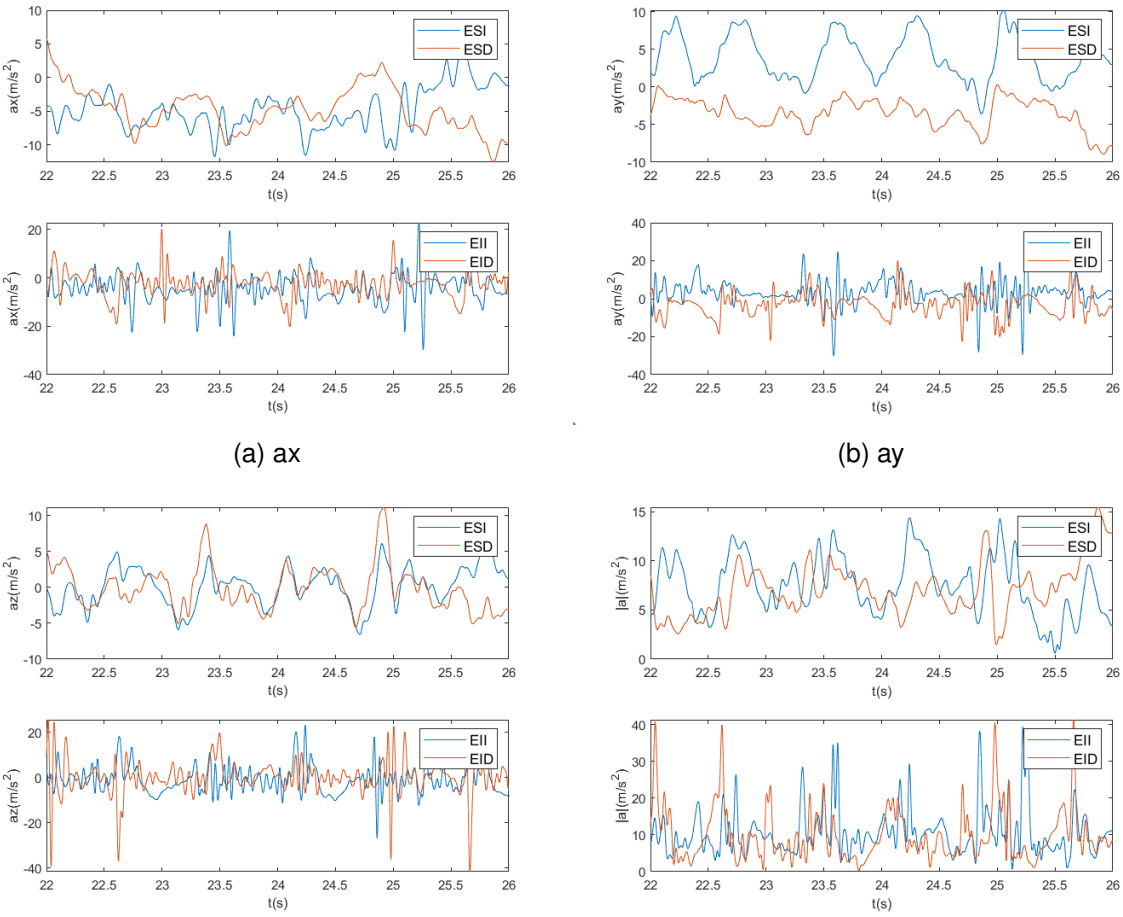
(c) az (d) |a|

Figura 4.13: Aceleraciones del Patinador 3 en un tramo recto.

retardo posible.

El profesional encargado de dicho estudio utiliza como sistema principal de medición un dispositivo colombiano conocido como Imocap, el cual está basado en unidades MPU y de *Electromiografía* (EMG) [69]. Sin embargo, según el responsable del proyecto, este dispositivo presenta algunos inconvenientes en su funcionamiento y los datos de desplazamiento adquiridos no permiten replicar con facilidad un movimiento natural en el exoesqueleto. En la Figura 4.26 se indica un fragmento de una curva de desplazamiento de la rodilla en la coordenada x, adquirida con el dispositivo en cuestión.

Con el propósito de comparar los datos de desplazamiento adquiridos con el dispositivo Imocap y con el sistema propuesto en el presente proyecto de titulación, se realizaron diversas pruebas en una caminadora. Se colocaron los nodos sensores en posiciones como el muslo, la rodilla y el tobillo, realizando trotes en pequeños intervalos de tiempo y a distintas velocidades. En la Figura 4.27 se muestra el exoesqueleto utilizado y la colocación de los nodos



(a) ax (b) ay

(c) az (d) |a|

Figura 4.14: Aceleraciones del Patinador 3 en una curva.

sensores en el muslo y la rodilla de la pierna izquierda. Del mismo modo, en la Figura 4.28 se evidencia una prueba realizada con movimientos de trote sobre la caminadora.

Concluidas las pruebas de trote en la caminadora, se adquieren desde el servidor los datos de aceleración. A estos valores se realiza la doble integración, utilizando el método del trapecio, para obtener los datos de desplazamiento. Posteriormente, se aplica un filtro pasa banda para mantener únicamente la información relevante. En la Figura 4.29 se indica un fragmento de la señal de aceleración y posición de la rodilla izquierda en las coordenadas x y z, ya que estas componentes son de mayor interés para el movimiento del exoesqueleto. Del mismo modo, en la Figura 4.30 se indica un fragmento de la señal de aceleración y de posición del tobillo en las coordenadas x y z.

Según el profesional responsable del proyecto, la información que brinda el sistema desarrollado es de mayor utilidad en relación con la obtenida con el dispositivo Imocap. Considerando interesante la posibilidad de adaptar dicho sistema netamente para adquirir los datos inercia-

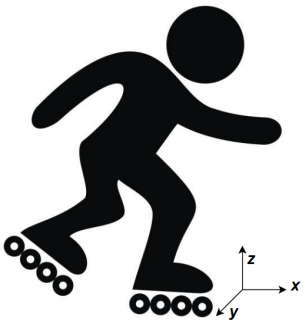
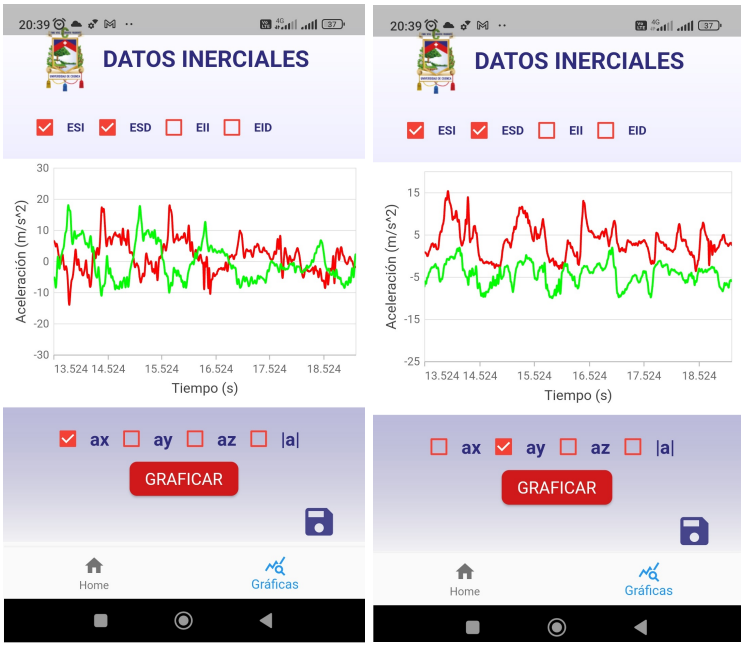
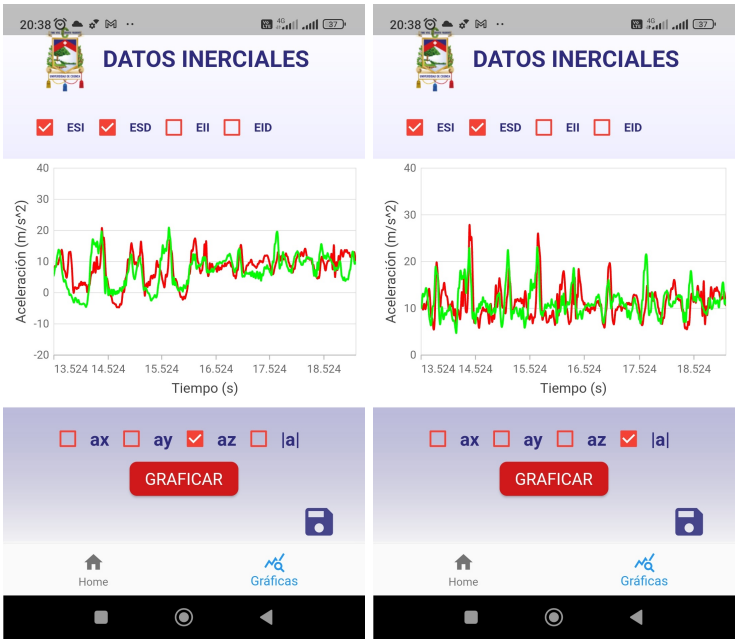


Figura 4.15: Eje de referencia para los valores de aceleración del patinador

les que permitan el movimiento del exoesqueleto en tiempo real.



(a) ax (b) ay



(c) az (d) |a|

Figura 4.16: Aceleraciones del Patinador 3, graficadas en la aplicación móvil

(a) dx (b) dy

(c) dz (d) |d|

Figura 4.17: Desplazamientos del Patinador 2, graficadas en la aplicación móvil

Figura 4.18: Péndulo físico utilizado para pruebas de aceleración

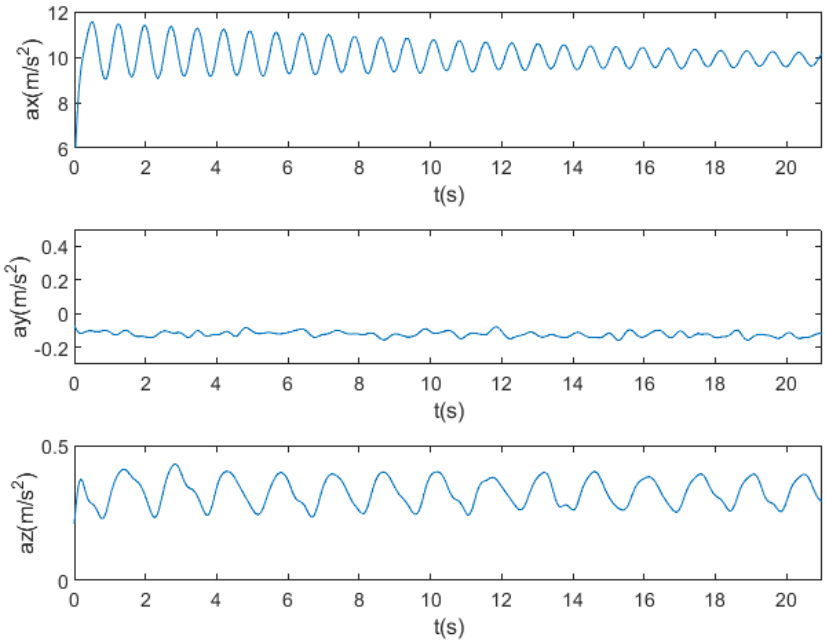


Figura 4.19: Oscilaciones con componentes de aceleración en x y z

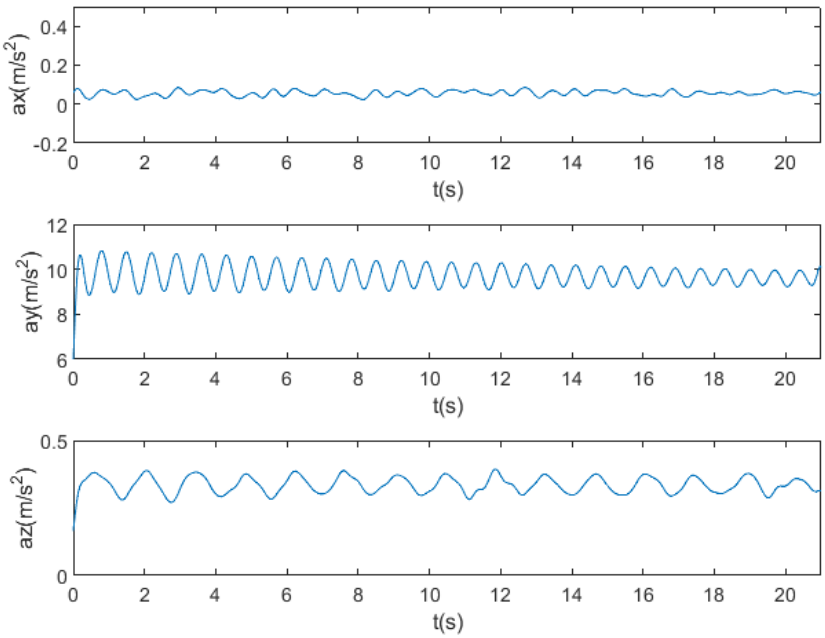


Figura 4.20: Oscilaciones con componentes de aceleración en y y z

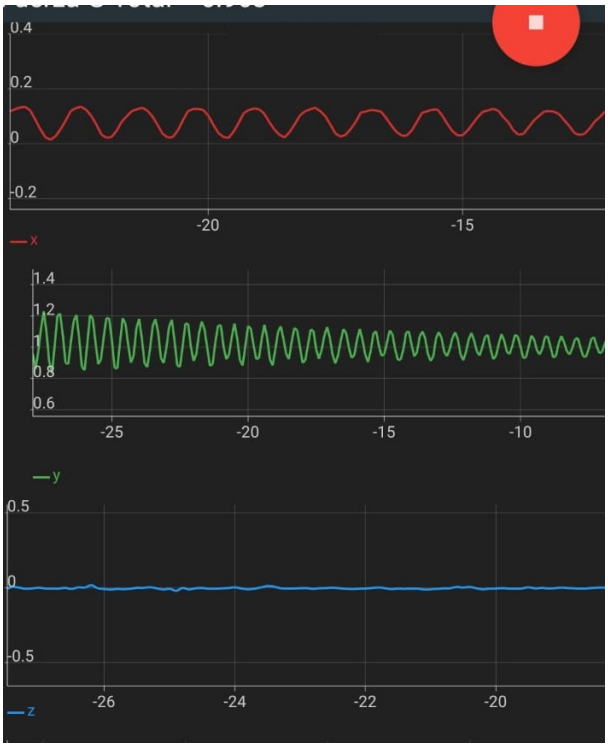


Figura 4.21: Componentes de aceleración de movimiento pendular (*smartphone*)

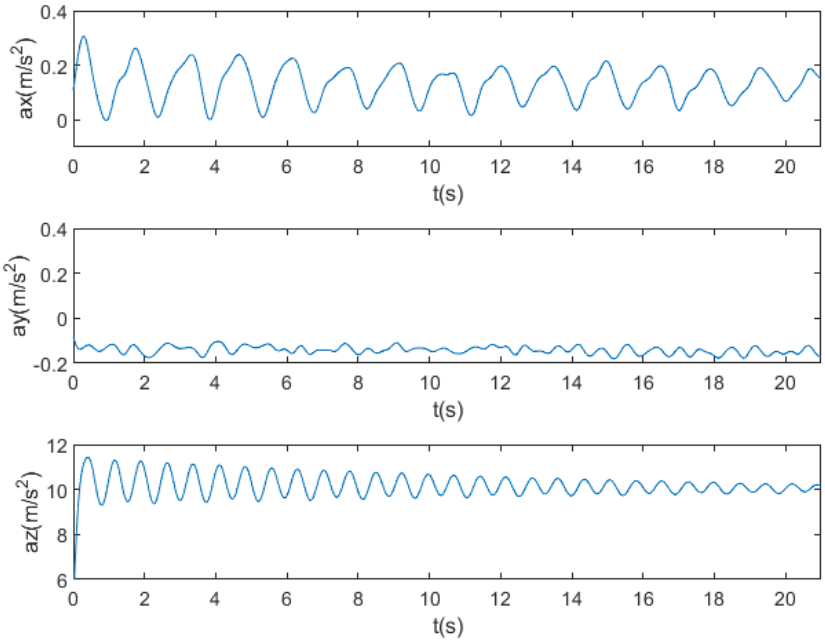


Figura 4.22: Oscilaciones con componentes de aceleración en z y x

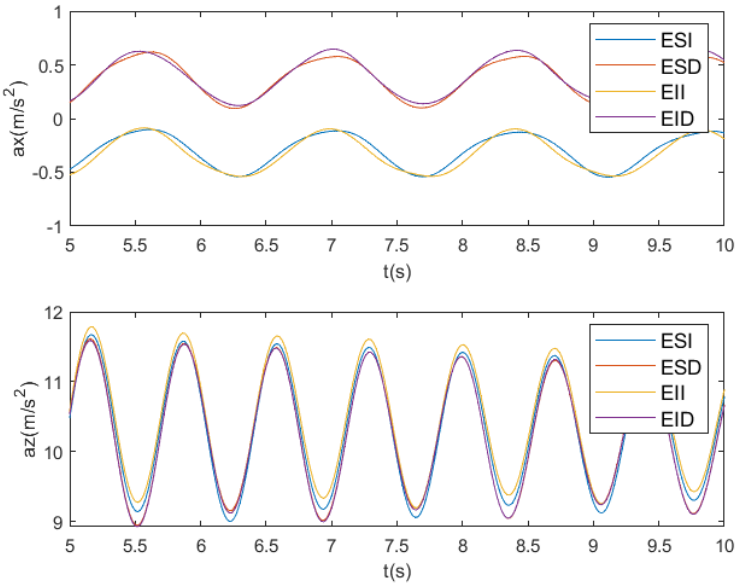


Figura 4.23: Componentes de aceleración z y x en un péndulo, para los cuatro nodos sensores

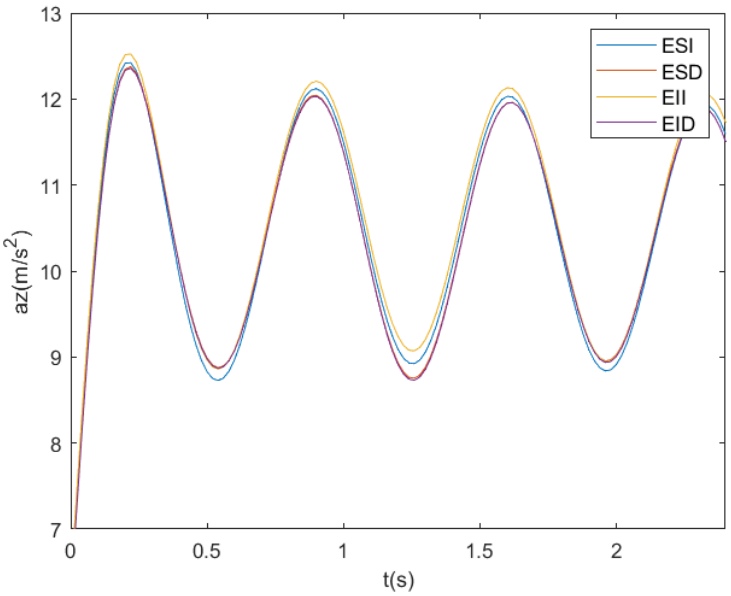
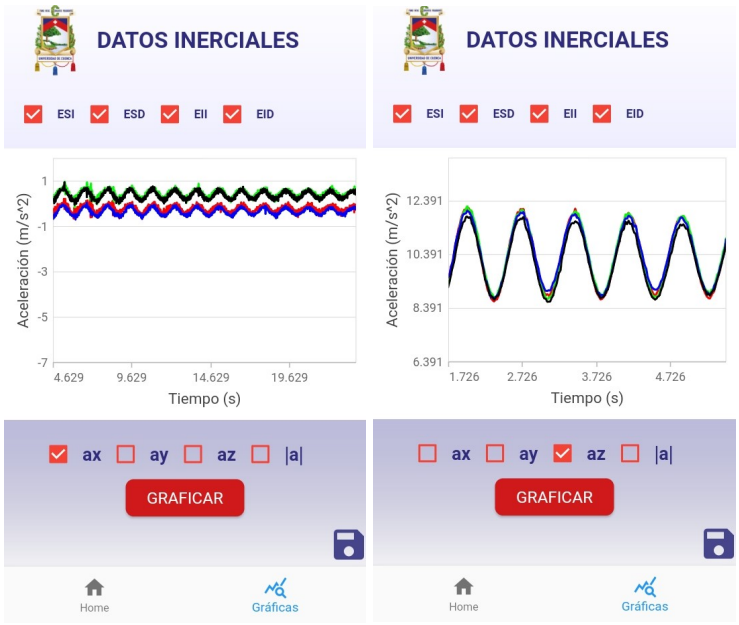


Figura 4.24: Componente z de aceleración en un péndulo, para los cuatro nodos sensores



(a) ax (b) az

Figura 4.25: Componentes de aceleración en un péndulo, para los cuatro nodos sensores (aplicación móvil)

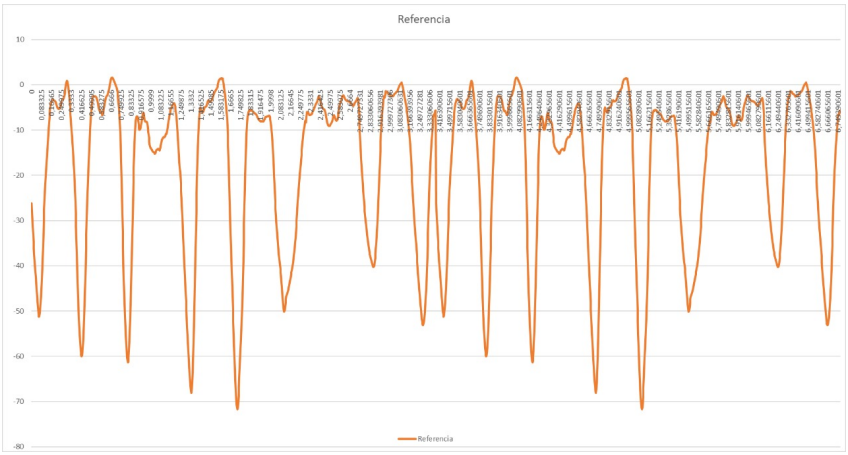
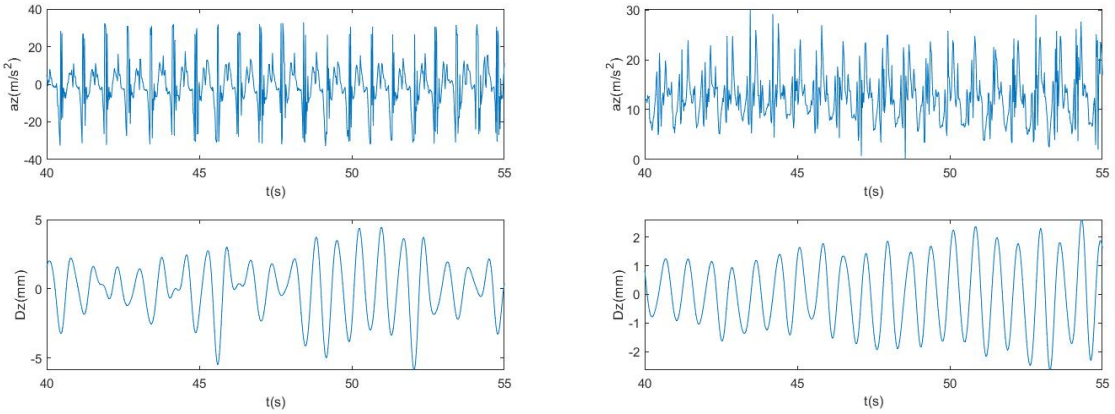


Figura 4.26: Curva de desplazamiento de la rodilla, obtenida con dispositivo Imocap

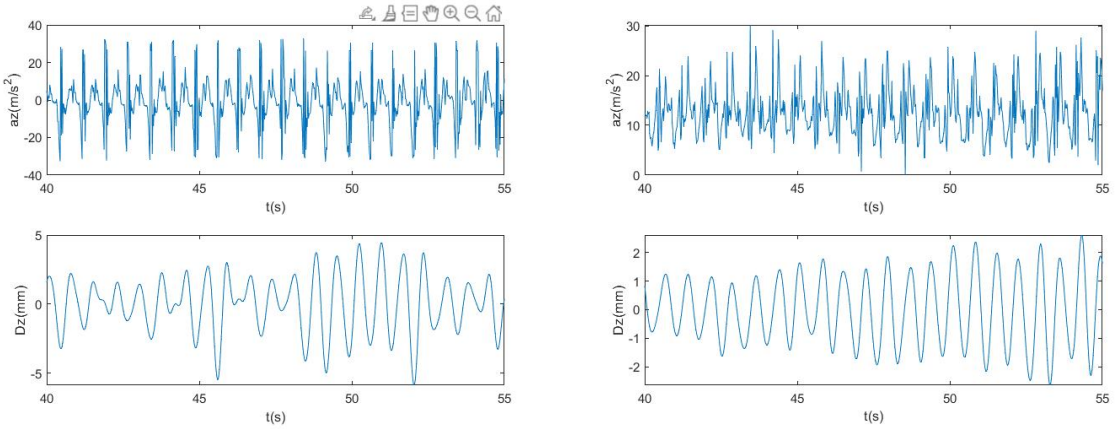
Figura 4.27: Exoesqueleto y colocación de los nodos en la pierna izquierda

Figura 4.28: Pruebas de trote en una caminadora



(a) dx (b) dz

Figura 4.29: Desplazamiento de la rodilla en las coordenadas x y z



(a) dx (b) dy

Figura 4.30: Desplazamiento de tobillo en las coordenadas x y z

Conclusiones y trabajos futuros

En este capítulo se exponen las conclusiones a las que se llegó luego de realizar el presente proyecto de titulación. Se indica también un panorama de trabajos futuros relacionados desde los cuales se puede mejorar el sistema propuesto o adaptarlo.

5.1. Conclusiones

La revisión del estado del arte permitió comprender de mejor manera la importancia del uso de IoT y los dispositivos *Wearables* para el análisis cuantitativo del desempeño de los deportistas. Investigar acerca de los trabajos relacionados con el uso de ESP32 en el deporte, corroboró la utilidad que tienen estos dispositivos sobre todo para la adquisición de datos biomecánicos, en donde son usados sensores inerciales. Sin embargo, en el patinaje de velocidad los estudios son escasos, por lo que haber desarrollado el presente proyecto conlleva un importante aporte a esta disciplina.

En la fase de empuje del patinaje de velocidad, un deportista puede realizar entre 85-90 empujes por minuto, lo que equivale a una zancada por segundo para cada extremidad. Por tanto, muestrear la aceleración de los ejes x, y y z cada 20ms (50Hz) permite adquirir la información de aceleración más relevante del patinador. La posibilidad de visualizar las gráficas de desplazamiento del deportista, contribuye a tener un panorama más amplio para el estudio de la técnica, aunque dicho estudio no forma parte del alcance de este proyecto.

Integrar en una misma PCB tanto los componentes de regulación de voltaje como de almacenamiento en la tarjeta microSD, permitió crear dispositivos ligeros y de pequeñas dimensiones para censar la aceleración de cada una de las cuatro extremidades del patinador, con un nodo adicional que comanda la captura de los datos. El sistema no incomodan a los deportistas en la ejecución de sus movimientos. Los dispositivos se vinculan fácilmente y la batería incorporada brinda una autonomía de alrededor de una hora. El sistema es fácilmente escalable, pudiendo agregar y vincular más nodos de estructura semejante sin inconvenientes. El hecho de ser un sistema *Wireless*, brinda comodidad de uso adicional.

En las pruebas realizadas con los patinadores se obtuvieron varias gráficas de aceleración tanto en un tramo recto como en curva. Los patinadores muestran movimientos rítmicos y simétricos de acuerdo a su técnica y aquellos con más experiencia presentan picos de aceleración mayores en sus movimientos. Ajustándose a los objetivos del presente proyecto, los datos se entregan en crudo, sin la aplicación de ningún filtro.

Para la comunicación entre los dispositivos ESP32 se usó el protocolo ESP-NOW. Este permitió una comunicación fiable entre los nodos sensores y el nodo central, formando una red en estrella. Con este protocolo se transmitieron los mensajes *broadcast* desde el nodo central. La sincronización entre los ESP32 está en el orden de los mus. Por su parte, con la utilización de ESP-IDF FreeRTOS se aprovecharon ambos núcleos del ESP32 para los procesos de captura, almacenamiento y envío de los datos de los nodos sensores.

Debido a las limitaciones de la versión utilizada de *Thingsboard Cloud*, se envían al servidor paquetes MQTT de 208 bytes de datos útiles cada 250ms. Esto equivale a enviar 240 mensajes por minuto, desde cada nodo directamente. Los datos de aceleración son codificados en tramas; esto permite almacenar más información por segmento de memoria, y posteriormente enviar al servidor una mayor cantidad de muestras por paquete. Codificar dichos paquetes en base64 permite que las tramas sean transmitidas usando un protocolo basado en ASCII.

La aplicación móvil desarrollada para Android permite comandar la adquisición de datos de aceleración mediante solicitudes RPC enviadas al servidor. Adicionalmente, permite consultar los datos recolectados y presentarlos gráficamente, tanto para la aceleración como para el desplazamiento; dichos datos también pueden ser descargados directamente desde el servidor para su análisis. La información del deportista puede ser posteriormente filtrada o procesada de acuerdo a las necesidades de estudio requeridas.

Como una aplicación adicional que no está contemplada en el alcance, se realizaron pruebas dentro de un proyecto donde se pretende replicar en un exoesqueleto los movimientos de las extremidades inferiores de una persona en tiempo real. La información de desplazamiento adquirida con el sistema propuesto fue de mucha utilidad para el profesional responsable del proyecto, corroborando de esta manera la utilidad del sistema de medición en múltiples disciplinas.

Con la integración de módulos ESP32, sensores inerciales y baterías Li-Po fue posible la creación de dispositivos *Wireless* IoT que permiten la adquisición datos de aceleración de las cuatro extremidades en las coordenadas x, y y z. Los dispositivos fueron puestos a prueba en tres patinadores de velocidad con diferentes niveles de experiencia y bajo las mismas condiciones; pudiendo ejecutar sus movimientos con comodidad gracias al reducido tamaño de los nodos sensores y su ubicación en el cuerpo. La información recolectada permite evaluar su desempeño tanto en tramo recto como curvo de la pista.

5.2. Trabajos Futuros

Los dispositivos desarrollados en el presente proyecto no generaron molestias a los patinadores en la ejecución de sus movimientos; sin embargo, es posible reducir el tamaño de los dispositivos reemplazando el módulo ESP32 Wemos D1 por únicamente el microcontrolador ESP32 e integrando en el PCB los componentes adicionales, también se puede integrar el sensor inercial MPU6050 en dicha placa.

Para obtener un producto terminado, es posible mejorar el proceso de vinculación de nuevos dispositivos considerando inicialmente una comunicación Bluetooth con el nodo central para registrar automáticamente las direcciones MAC de los nodos. Adicionalmente, se puede considerar la programación *Over The Air* (OTA) para la configuración de los dispositivos.

Debido a las restricciones de la versión de *Thingsboard Cloud* no es posible modificar determinados parámetros. Por lo que se puede realizar pruebas con un servidor propio de *Thingsboard*, buscando reducir al mínimo el tiempo de envío de los mensajes e incrementando el tamaño de los paquetes; todo esto modificando los archivos de configuración *.yml* del servidor. También se puede estudiar la posibilidad de encriptar los mensajes MQTT enviados al servidor.

Al trabajar en conjunto con un experto en biomecánica deportiva es posible realizar un procesamiento adicional de los datos inerciales adquiridos; aplicando determinados filtros que permitan conservar únicamente la información más relevante del movimiento de los patinadores. Relacionando también determinados patrones de movimiento con las técnicas individuales de cada deportista, para encontrar mecanismos que mejoren su desempeño.

Continuar con los estudios de simulación de movimientos con un exoesqueleto, adaptando el sistema a estos objetivos. Para ello, se puede excluir de la arquitectura el uso de un servidor *web* y pretender que el envío de los datos inerciales sea en tiempo real directamente hacia el controlador del exoesqueleto.

Apéndice

Los apéndices 5.3 y 5.4 indican con mayor detalle el proceso de configuración del servidor web para agregar dispositivos y la estructura de los mensajes utilizados para la comunicación, tanto con los nodos sensores como con la aplicación móvil.

5.3. Configuración del servidor Web

5.3.1. Agregación de dispositivos

Una manera de aprovisionar los dispositivos es mediante la interfaz de usuario de *Thingsboard*. Inicialmente, se requiere crear una cuenta en *Thingsboard Cloud*. Dentro de la plataforma, en el apartado “Grupos de dispositivos” se agregan los dispositivos que corresponden a los cinco nodos utilizados en el cuerpo del patinador. En la Figura 5.1 se muestran los dispositivos agregados al servidor, mismos que llevan los nombres: ESI, ESD, EII, EID y Torso. Adicionalmente, se agrega un dispositivo llamado *Controller A*, el cual actúa como un controlador del nodo principal Torso y a través del cual se consulta si los nodos están listos para comenzar la captura de aceleraciones.

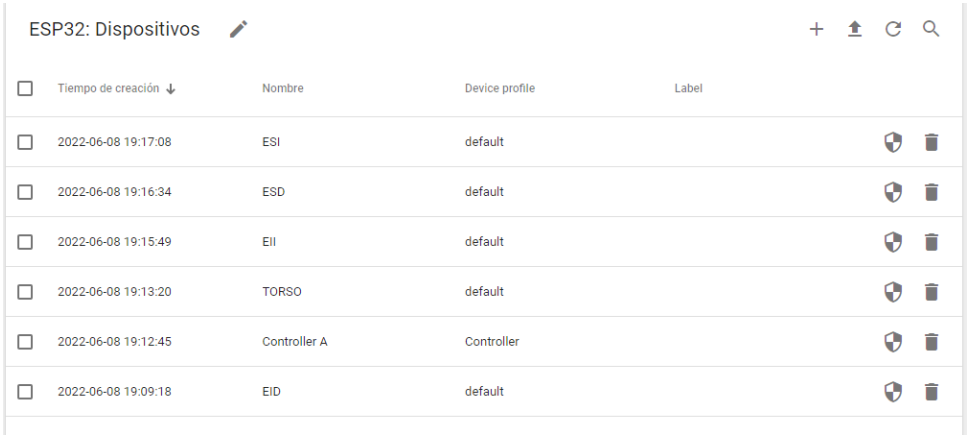


Figura 5.1: Dispositivos agregados al servidor *Thingsboard Cloud*

Cuando el sistema está listo para la captura de datos, el nodo central envía un mensaje: `state = True`, al servidor. A su vez, la aplicación móvil solicita el estado de dicha variable a través de una petición de consulta al dispositivo `Controller A`, notificando al usuario que todo está listo. Para ello, se requiere establecer en el servidor una relación entre el dispositivo principal Torso y el controlador `Controller A`, de tal manera que este último pueda acceder a los datos de telemetría. En la Figura 5.2 se evidencia la relación creada para dichos dispositivos.

Cada uno de los dispositivos agregados en el servidor tiene un código *Access Token* único de

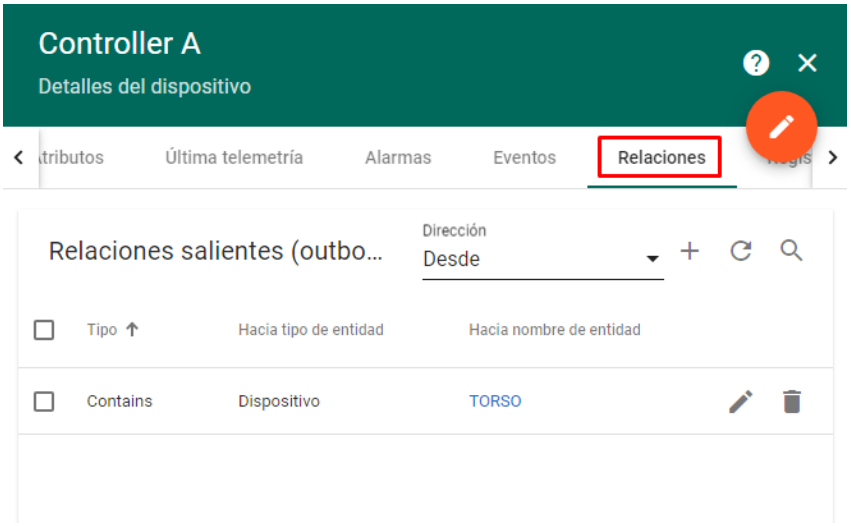


Figura 5.2: Relación creada entre Controller A y Torso

veinte dígitos. Mediante esta credencial los distintos nodos sensores identifican a que dispositivo del servidor deben enviar los datos adquiridos. Además, los códigos Token se utilizan para acceder a la información de telemetría almacenada en el servidor, mediante llamadas RPC enviadas desde la aplicación móvil. Para obtener el código *access token*, se debe acceder a los detalles del dispositivo correspondiente y dar click en el botón “copiar access token”, como se indica en la Figura 5.3 para el dispositivo Torso. En el siguiente apartado se muestra el uso del *Access Token* en los módulos ESP32.

Figura 5.3: Obtención del *Access Token* del dispositivo Torso

5.3.2. Programación de los ESP32 para la comunicación con Thingsboard

En la programación de los ESP32, para enlazar cada nodo con *Thingsboard*, se sigue el siguiente proceso [81]:

1. Incluir en la programación la librería `ThingsBoard.h` y verificar que se disponga de la librería `ArduinoJSON`.
2. Definir la variable `THINGSBOARD_SERVER` con la dirección IP o URL del servidor. En este punto se coloca el *link* del servidor: `"thingsboard.cloud"`.
3. Definir la variable *TOKEN* de acuerdo al *access token* correspondiente de acuerdo a la Tabla 5.1.
4. Inicializar el cliente de Thingsboard `espClient`
5. Inicializar la instancia Thingsboard `tb(espClient)`.
6. Para conectar cada nodo a Internet, se requiere agregar la librería `WiFi.h`. Definir las variables: `ssid = "Nombre de la red"` y `password = "Clave de la red"`. Cabe mencionar que tanto los nodos sensores como el nodo central son definidos como AP y estación WiFi.

Tabla 5.1: Token correspondiente a cada nodo.

Extremidad	Nodo	Access token
Torso	Nodo central	"$vnqG1qqsdQxxxxxxxxx$"
ESI	Nodo 1	"$TOa5jGvjRIxxxxxxxxx$"
ESD	Nodo 2	"$Fer1lDnUZ3xxxxxxxxx$"
EII	Nodo 3	"$cAgpGSTIIcxxxxxxxxx$"
EID	Nodo 4	"$EGbvE4BT6ixxxxxxxxx$"

Con el objetivo de notificar al servidor que todos los nodos sensores están listos, el nodo central por medio de la instrucción: `sendTelemetryBool("state", state)`, envía un mensaje de telemetría del tipo booleano usando MQTT. La variable `state` se mantendrá en `false`, hasta que todos los nodos se encuentre listos. Cuando el sistema este listo se enviara la variable `state = true`.

Para el caso particular del nodo central, mismo que recibe las órdenes de iniciar o finalizar, la captura de datos a través una llamada RPC del servidor, se crea la función de procesos para la solicitud RPC `setOrder` (Listado 5.1) y el manejador RPC `callbacks` (Listado 5.2).

Extracto de código 5.1: Función de procesos para la llamada RPC *setOrder*

```
RPC_Response GpioState(const RPC_Data &data){
  letsgo = data["start"];
  return RPC_Response(NULL, (int)data["start"]);
}
```

Extracto de código 5.2: Manejador RPC

```
RPC_Callback callbacks[] = {
    {"setOrder", GpioState}
};
```

En el nodo central, la variable `letsgo` almacena el valor que retorna la llamada RPC `serOrder`. En el caso que `letsgo` sea `1`, el nodo principal enviará la orden de iniciar a los nodos sensores. Mientras que si `letsgo` es `2`, se enviará la orden de finalizar.

El envío de los datos capturados lo realiza cada nodo sensor a través de la instrucción `sendTelemetryString`.

5.4. Aplicación móvil

5.4.1. Solicitudes RPC

Para efectuar las diferentes solicitudes RPC con API REST, se utiliza HTTP, que está basado en el *Protocolo de Control de Transmisión* (TCP) y utiliza el modelo de solicitud-respuesta. Los nodos de *Thingsboard* actúan como un servidor HTTP, pudiendo acceder a los dispositivos agregados mediante un *token* o un ID, que es la ruta de acceso a cada solicitud HTTP. Cabe mencionar que, de forma predeterminada, *Thingsboard* admite contenido clave-valor en JSON [82]. A continuación, indica la estructura de las solicitudes RPC enviadas desde la aplicación móvil.

5.4.1.1. Consultar estado de los nodos

Inicialmente, desde la aplicación móvil se requiere conocer si los nodos sensores están listos para capturar los datos de aceleración. Para ello, se envía al servidor la solicitud del Listado 5.3.

Extracto de código 5.3: Solicitud para consultar estado de los nodos

```
curl -X POST -d '{"method": "ready", "params":{}}'
https://thingsboard.cloud/api/v1/YjCLy2lNDCivQPn77cax/rpc
--header "Content-Type:application/json"
```

La estructura de dicha solicitud se indica a continuación [83]:

- Se realiza una solicitud POST con el método `ready`.
- Para enviar una solicitud HTTP, se utiliza la utilidad `curl`.
- `https://thingsboard.cloud` es la *Uniform Resource Locator* (URL) base del servidor de *ThingsBoard Cloud*.
- YjCLy2lNDCxxxxxxxxx corresponde al *access token* del dispositivo `Controller A` configurado en el servidor.

Si los nodos están listos, la aplicación recibe la respuesta `isReady = True`.

5.4.1.2. Captura de datos de aceleración

Utilizando una solicitud semejante a la indicada anteriormente, se ejecuta la orden de iniciar o finalizar la captura de datos de aceleración mediante las líneas de código del Listado 5.4.

Extracto de código 5.4: Solicitud para comandar captura de datos

```
curl -X POST -d '{"method": "start/stop", "params":{}}'
https://thingsboard.cloud/api/v1/vnqG1qqsdQkfWT0xBEHd/rpc
--header "Content-Type:application/json"
```

- Se realiza una solicitud POST con el método `start` o `stop`.
- Para enviar una solicitud HTTP, se utiliza la utilidad `curl`.
- `https://thingsboard.cloud` es la URL base del servidor de *ThingsBoard Cloud*.
- vnqG1qqsdQxxxxxxxxx corresponde al *access token* del dispositivo central Torso, configurado en el servidor.

`State = OK` es la respuesta que recibe la aplicación para asegurarse que la orden fue recibida correctamente por el dispositivo central.

5.4.1.3. Obtener *Token* JWT

ThingsBoard utiliza *tokens* JWT para representar solicitudes de forma segura entre el cliente API (navegador, scripts, etc.) y la plataforma. El *token* principal es un *token* de corta duración que se debe usar para realizar las llamadas a la API, como para obtener los datos de telemetría. Los valores de tiempo de caducidad predeterminados del *token* principal son 2,5 horas

[84], por lo que se requiere consultar frecuentemente dicho parámetro, por medio de las líneas de código del Listado 5.5.

Extracto de código 5.5: Solicitud para obtener el *token* JWT

```
curl -X POST --header 'Content-Type: application/json'
--header 'Accept: application/json'
-d '{"username":"your_user@company.com", "password":"secret"}'
'https://thingsboard.cloud/api/auth/login'
```

De esta manera se puede obtener el *token* JWT para el usuario "your_user@company.com".

5.4.1.4. Obtener datos de telemetría

Es posible obtener desde el servidor una lista de valores históricos para un tipo de entidad concreta [85]. La siguiente solicitud del Listado 5.6 permite obtener los datos históricos de telemetría correspondientes a la aceleraciones adquiridas en los distintos nodos sensores:

Extracto de código 5.6: Solicitud para obtener datos de telemetría

```
curl -v -X GET --header 'Content-Type: application/json'
"https://thingsboard.cloud/api/plugins/telemetry/DEVICE/{
   entityId}//values/timeseries?keys=ax,ay,az,a&startTs=
   tiempo_inicial&endTs=tiempo_final&interval=1&limit=100&agg=
   NONE"
--header "X-Authorization: Bearer $YOUR_JWT_TOKEN "
```

Los parámetros admitidos se describen a continuación [85]:

- **keys:** lista separada por comas de claves de telemetría para recuperar. En este caso se consideran las claves: ax,ay,az,a.
- `startTs`: marca de tiempo unix (hora local) que identifica el inicio del intervalo en milisegundos.
- `endTs`: marca de tiempo unix (hora local) que identifica el final del intervalo en milisegundos.
- `interval`: el intervalo de agregación, en milisegundos.
- `agg`: la función de agregación: MIN, MAX, AVG, SUM, COUNT, NONE.

- `limit`: la cantidad máxima de puntos de datos que se devolverán o intervalos que se procesarán.

El parámetro `entityId` corresponde al ID del dispositivo, por lo que se requiere realizar una petición diferente por cada nodo sensor utilizado. De igual manera, debe establecerse el encabezado *X-Authorization* en “Bearer”, utilizando el token JWT principal.

Referencias

[1] S. Sinha, “State of iot 2021: Number of connected iot devices growing 9 % to 12.3 billion globally, cellular iot now surpassing 2 billion.” 9 2021. [Online]. Available: https://iot-analytics.com/number-connected-iot-devices/

[2] F. Laricchia, “Number of connected wearable devices worldwide from 2016 to 2022,” 02 2022. [Online]. Available: https://www.statista.com/statistics/487291/global-connected-wearable-devices/

[3] NETTIGO, “Wifi+bluetooth module esp32 d1 mini,” 2022. [Online]. Available: https://nettigo.eu/products/wifi-bluetooth-module-esp32-d1-mini

[4] lhespress, “Esp-now repository,” 2022. [Online]. Available: https://github.com/espressif/esp-now

[5] B. J. ŠESTÁK, “Dynamic mesh network implemented in micropython on top of esp-now protocol.”

[6] A. Banks, E. Briggs, K. Borgendale, and R. Gupta, “Mqtt versión 5.0,” 2019. [Online]. Available: https://docs.oasis-open.org/mqtt/mqtt/v5.0/mqtt-v5.0.html

[7] *MPU-6000and MPU-6050Register Map and DescriptionsRevision 4.2*, InvenSense Inc, 8 2013, rev. 4.2.

[8] O. M. de la Salud, “Actividad física,” 11 2020. [Online]. Available: https://www.who.int/es/news-room/fact-sheets/detail/physical-activity

[9] J. G. Delgado Martinez, “Revisión sistemática de la literatura sobre el uso de la tecnología en el deporte,” 2020.

[10] J. Taborri, J. Keogh, A. Kos, A. Santuz, A. Umek, C. Urbanczyk, E. van der Kruk, and S. Rossi, “Sport biomechanics applications using inertial, force, and emg sensors: a literature overview,” *Applied bionics and biomechanics*, vol. 2020, 2020.

[11] C. Muñiz and P. Downward, “Una mejor comprensión del impacto del deporte y la actividad física sobre la salud, la integración social, el mercado laboral y el rendimiento académico,” *Papeles de Economía Española*, no. 159, pp. 241–260, 2019.

[12] L. A. Fava, D. G. Vilches Antão, A. Ferraresso, E. Boccalari, and F. J. Díaz, “Inteligencia y tecnologías aplicadas al deporte de alto rendimiento,” in *XXII Workshop de Investigadores en Ciencias de la Computación (WICC 2020, El Calafate, Santa Cruz)*, 2020.

[13] C. Ávila and G. Federico, “Uso de gps y acelerómetro para medir performance en deporte de élite.”

[14] L. B. Robles and C. Barreto, “Uso de dispositivos gps e imu para analizar la performance de deportistas de alto rendimiento,” Ph.D. dissertation, Universidad Nacional de La Plata, 2020.

[15] B. S. Paz Viteri, “La condición física en la aptitud deportiva del patinaje de la categoría infantil en la federación deportiva de chimborazo,” B.S. thesis, Universidad Técnica de Ambato. Facultad de Ciencias humanas y de la . . . , 2016.

[16] G. Aroganam, N. Manivannan, and D. Harrison, “Review on wearable technology sensors used in consumer sport applications,” *Sensors*, vol. 19, no. 9, p. 1983, 2019.

[17] M. del Carmen Rey-Merchán, A. López-Arquillos, and J. M. Soto-Hidalgo, “Integración, aprendizaje y gestión del uso de dispositivos iot para la mejora de la seguridad en obras,” in *XXV Congreso de Ingeniería de Organización*, p. 65.

[18] K. Ishida, “Iot application in sports to support skill acquisition and improvement,” in *2019 IEEE 12th Conference on Service-Oriented Computing and Applications (SOCA)*. IEEE, 2019, pp. 184–189.

[19] Z. Lu, Y. Zhu, C. Jia, T. Zhao, M. Bian, C. Jia, Y. Zhang, and Y. Mao, “A self-powered portable flexible sensor of monitoring speed skating techniques,” *Biosensors*, vol. 11, no. 4, p. 108, 2021.

[20] P. S. Glazier, “Beyond animated skeletons: how can biomechnical feedback be used to enhance sports performance?” *Journal of Biomechanics*, p. 110686, 2021.

[21] J. D. Á. DIAZ and M. O. M. V. ROCHA, “Desarrollo de la fuerza explosiva y potencia en una prueba de velocidad de 100 metros, aplicando una estrategia (innovación del movimiento) en patinadores 10 a 14 años de la escuela de formacion deportiva los delfines de el carmen de bolívar.” 2019.

[22] C. Lugea, “Algunas consideraciones sobre biomecánica, técnica y el modelo técnico en el patinaje de velocidad.” 2007.

[23] R. O. Maria Fernanda, “Skate dreams propuesta para incentivar los deportes agrupados en la liga de patinaje de bogotá mediante el diseño de un escenario deportivo,” 2020.

[24] R. V. Barros Carpintero and J. D. Prieto Diaz, “Fundamentos teóricos que existen respecto al proceso de entrenamiento de la técnica de patinaje de velocidad en los últimos 10 años en el mundo.” Ph.D. dissertation, 2018.

[25] M. B. Andrés, in *Internet de las cosas*. REUS Editorial, 2018, pp. 13–24.

[26] J. Salazar and S. Silvestre, “Internet de las cosas,” *Techpedia. České vysoké učení technické v Praze Fakulta elektrotechnická*, 2016.

[27] europaexpress, “El uso de dispositivos iot aumentó un 66 % en los dos últimos años, según telefónica,” 11 2019. [Online]. Available: https://www.europapress.es/economia/noticia-uso-dispositivos-iot-aumento-66-dos-ultimos-anos-telefonica-20191114114851.html

[28] A. L. Hernandez, M. C. Barrera Cortéz, A. Ávila Barón, L. A. Téllez Tinjacá, and H. A. Guío Ávila, “Tecnología vestible una ventaja competitiva en el entrenamiento deportivo,” pp. 99–105, 2020.

[29] S. C. Naranjo, “El gasto en tecnología wearable se duplicará en 2021,” 11 2019. [Online]. Available: https://es.statista.com/grafico/19984/gasto-estimado-del-usuario-en-dispositivos-ponibles-a-nivel-mundial/

[30] “Wearable technology market size, share & trends,” 2019. [Online]. Available: https://www.grandviewresearch.com/industry-analysis/wearable-technology-market

[31] F. Torralba, “Wearables y movimiento grinder. del deporte a la vida cotidiana,” pp. 228,229, 2018.

[32] E. A. Sullón Atoche and E. A. Moreno Lavaho, “Control de la actividad física con tecnología vestible (wearables). una revisión sistemática.” 2020.

[33] I. Consulting, “Cómo elegir un microcontrolador para iot,” 2020. [Online]. Available: https://iotconsulting.tech/como-elegir-un-microcontrolador-para-iot/

[34] B. S. de Información, “Dispositivos hardware en el iot,” 2021. [Online]. Available: https://www.babelgroup.com/es/Media/Blog/Junio-2021/Dispositivos-hardware-en-el-IoT

[35] A. Kurniawan, *Internet of Things Projects with ESP32: Build exciting and powerful IoT projects using the all-new Espressif ESP32*. Packt Publishing Ltd, 2019.

[36] P. Bertoleti, *Proyectos com ESP32 y LoRa*. Editora NCB, 2019.

[37] bastelgarage, “Esp32minikit wemos,” 2021. [Online]. Available: https://www.bastelgarage.ch/esp32minikit-wemos

[38] programador clic, “Sistema de almacenamiento esp32 y flash,” 2022. [Online]. Available: https://programmerclick.com/article/32461097444/

[39] E. I. Team, “Esp-now: Espressif’s wireless communication protocol,” 2021. [Online]. Available: https://www.espressif.com/en/news/ESP-NOW

[40] ——, “Esp-now user guide,” 2016. [Online]. Available: https://www.espressif.com/sites/default/files/documentation/esp-now_user_guide_en.pdf

[41] ——, “Esp-now programming guide,” 2022. [Online]. Available: https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-reference/network/esp_now.html

[42] H. GmbH, “Mqtt & mqtt 5 essentials,” 2020. [Online]. Available: https://www.hivemq.com/downloads/hivemq-ebook-mqtt-essentials.pdf

[43] T. Authors, “What is thingsboard?” 2022. [Online]. Available: https://thingsboard.io/docs/getting-started-guides/what-is-thingsboard/

[44] ——, “Getting started with rule engine,” 2022. [Online]. Available: https://thingsboard.io/docs/paas/user-guide/rule-engine-2-0/re-getting-started/

[45] E. Inc., “Esp32 technical reference manual,” 2022. [Online]. Available: https://www.espressif.com/sites/default/files/documentation/esp32_technical_reference_manual_en.pdf

[46] R. Mischianti, “Memoria flash spi externa,” 2022. [Online]. Available: https://www.mischianti.org/2022/08/02/arduino-fast-external-spi-flash-memory/

[47] G. Cárdenas Álvarez, “Diseño de un sistema sensorial para la medición de la presión en el pie al realizar la práctica en patinaje de velocidad,” 2020.

[48] S. A. Riaño Bermúdez *et al.*, “Zapatos inteligentes: recopilación y procesamiento de constantes vitales y cinemática de los pies para uso deportivo,” 2018.

[49] K. Ishida, “Iot application in sports to support skill acquisition and improvement,” in *2019 IEEE 12th Conference on Service-Oriented Computing and Applications (SOCA)*. IEEE, 2019, pp. 184–189.

[50] K. E. Rodríguez Zapata *et al.*, “Sistema de sensores inerciales para análisis biomecánico,” B.S. thesis, Quito, 2019.

[51] A. W. Setiawan, F. Muhammad, A. Hidayat *et al.*, “Development of an web-based wearable gait recognition system using gyroscope and accelerometer sensors,” in *2020 International Seminar on Application for Technology of Information and Communication (iSemantic)*. IEEE, 2020, pp. 370–373.

[52] D. Seco Morales, “Desarrollo de sistema de medida de distancia para competiciones de ciclismo en ruta,” 2020.

[53] M. Bonilla, P. Córdova, D. Jiménez, and A. Robayo, “Monitoring system of physiological signals with emergencies management for the road safety of cyclists,” in *2019 Sixth International Conference on eDemocracy & eGovernment (ICEDEG)*. IEEE, 2019, pp. 274–279.

[54] A. Machado and C. Carvalho, “Pro-athlete: A guide system for visually impaired athletes in olympic track race,” *ITEGAM-JETIA*, vol. 7, no. 28, pp. 16–22, 2021.

[55] S. A. Cárdenas Calderón, “Sistema para la determinación de la puntuación en combates de karate de forma automática,” 2020.

[56] A. M. d. Real Rico, “Workout box, dispositivo cibernético para eficientar el aprovechamiento durante las rutinas de entrenamiento,” 2019.

[57] J. J. Puetate Paredes, “Implementación de un sistema para cuantificar el rendimiento físico de futbolistas aplicando el método ahp (proceso analítico jerárquico),” B.S. thesis, Riobamba, Universidad Nacional de Chimborazo, 2021.

[58] O. impulse, “Ams1117-3.3v voltage regulator module,” 2021. [Online]. Available: https://www.openimpulse.com/blog/products-page/product-category/ams1117-3-3v-voltage-regulator-module/

[59] O. S. HardwareLab, “Micro sd card module,” 2022. [Online]. Available: https://oshwlab.com/adrirobot/micro-sd-card-module

[60] L. Llamas, “Comparativa esp8266 frente a esp32, los soc de espressif para iot,” 2018. [Online]. Available: https://www.luisllamas.es/comparativa-esp8266-esp32/

[61] electrohobby.es, “Acelerometro y giroscopio mpu-6050,” 2020. [Online]. Available: https://www.electrohobby.es/posicion/168-acelerometro-y-giroscopio-mpu-6050.html#:~:text=Consumo%20de%20corriente%20de%203.8,rango%20detemperatura%20de%20funcionamiento.

[62] Altium, “Cómo diseñar circuitos de alimentación microsd,” 2020. [Online]. Available: https://resources.altium.com/es/p/how-to-design-microsd-power-circuits-without-destabilizing-on-board-voltage-supply

[63] M. Beltrán and F. Tacuri, “Desarrollo de un sistema iot para la adquisición de datos biomecánicos de un patinador de velocidad, utilizando dispositivos esp32,” 2023. [Online]. Available: https://github.com/freddytac/TT_BELTRAN_TACURI.git

[64] E. Systems, “Esp-idf freertos (smp),” 2020. [Online]. Available: https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-guides/freertos-smp.html#id1

[65] A. V. Zinkevich, “Esp8266 microcontroller application in wireless synchronization tasks,” in *2021 International Conference on Industrial Engineering, Applications and Manufacturing (ICIEAM)*. IEEE, 2021, pp. 670–674.

[66] N. pool project, “Introduction,” 2022. [Online]. Available: https://www.pool.ntp.org/es/

[67] R. Bull, “Los deportes con las aceleraciones más violentas,” 2022. [Online]. Available: https://www.redbull.com/es-es/deportes-con-las-aceleraciones-mas-violentas

[68] D. J. Ruiz Rivera, “Valoración funcional en patinadores de velocidad de alto nivel: determinación de forma directa, mediante una prueba de campo, de la velocidad aeróbica máxima patinando,” p. 31, 2015.

[69] M. Callejas-Cuervo, M. A. Vélez-Guerrero, and W. J. P. Holguín, “Arquitectura de un sistema de medición de bioparámetros integrando señales inerciales-magnéticas y electromiográficas,” *Revista Politécnica*, vol. 14, no. 27, pp. 93–102, 2018.

[70] L. Llamas, “¿que es mqtt?” 2019. [Online]. Available: https://www.luisllamas.es/que-es-mqtt-su-importancia-como-protocolo-iot/

[71] M. contributors, “Base64 codificando y decodificando,” 2022. [Online]. Available: https://developer.mozilla.org/es/docs/Glossary/Base64#:~:text=Los%20esquemas%20de%20codificaci%C3%B3n%20Base64,sin%20modificaciones%20durante%20la%20transmisi%C3%B3n.

[72] T. Authors, “Filter nodes,” 2022. [Online]. Available: https://thingsboard.io/docs/paas/user-guide/rule-engine-2-0/filter-nodes/

[73] ——, “Action nodes,” 2022. [Online]. Available: https://thingsboard.io/docs/paas/user-guide/rule-engine-2-0/action-nodes/

[74] ——, “Enrichment nodes,” 2022. [Online]. Available: https://thingsboard.io/docs/paas/user-guide/rule-engine-2-0/enrichment-nodes/

[75] ——, “Thingsboard cloud subscription plans definition,” 2022. [Online]. Available: https://thingsboard.io/products/paas/subscription/

[76] G. community, “Max length of telemetry payload mqtt,” 2022. [Online]. Available: https://github.com/thingsboard/thingsboard/issues/1363

[77] T. Authors, “Configuration properties,” 2022. [Online]. Available: https://thingsboard.io/docs/user-guide/install/config/

[78] J. Fernandez, “Movimiento armónico simple en péndulos,” 2022. [Online]. Available: https://www.fisicalab.com/apartado/mas-y-pendulos

[79] C. Izquierdo, “Clase 9: Teoría de un péndulo físico,” 2013. [Online]. Available: https://www.youtube.com/watch?v=zPhtXxRTgBw

[80] ——, “Clase 10: Problema de péndulo físico con una barra,” 2013. [Online]. Available: https://www.youtube.com/watch?v=x9kc2nZFD4k&list=WL&index=102&t=167s

[81] T. Authors, “Esp32 pico kit gpio control and dht22 sensor monitor using thingsboard arduino sdk,” 2022. [Online]. Available: https://thingsboard.io/docs/samples/esp32/gpio-control-pico-kit-dht22-sensor/

[82] ——, “Http device api reference,” 2022. [Online]. Available: https://thingsboard.io/docs/paas/reference/http-api/

[83] ——, “Rpc reply with data from related device,” 2022. [Online]. Available: https://thingsboard.io/docs/user-guide/rule-engine-2-0/tutorials/rpc-reply-tutorial/

[84] ——, “Rest api,” 2022. [Online]. Available: https://thingsboard.io/docs/paas/reference/rest-api/#:~:text=When%20you%20login%20to%20the,hours%20and%201%20week%20respectively.

[85] ——, “Working with telemetry data,” 2022. [Online]. Available: https://thingsboard.io/docs/paas/user-guide/telemetry/