Estimulación y monitorización de enfermos UCI
Stimulation and monitoring of patients in the ICU

Trabajo de Fin de Máster
Curso 2021–2022

Autor
Guillermo García Mansilla

Director
Carlos García Sánchez

Colaborador

Máster en Internet de las Cosas
Facultad de Informática
Universidad Complutense de Madrid

Estimulación y monitorización de enfermos UCI
Stimulation and monitoring of patients in the ICU

Trabajo de Fin de Máster en Internet de las Cosas
Departamento de Informática

Autor
Guillermo García Mansilla

Director
Carlos García Sánchez

Colaborador

Convocatoria: *Junio 2022*
Calificación: *8*

Máster en Internet de las Cosas
Facultad de Informática
Universidad Complutense de Madrid

24 de Junio de 2022

Agradecimientos

A todo el personal del hospital de Fuenlabrada dispuesto a ayudar con este trabajo y a mi abuelo Clemente que también ha sido de gran ayuda poder contar con su apoyo.

Resumen

Estimulación y monitorización de enfermos UCI

El servicio de la Unidad de Cuidado Intensivos del hospital Universitario de Fuenlabrada viene trabajando en los últimos años en el Proyecto InnovaHUCI[1] (de cuidados intensivos, 2018) cuyo principal objetivo es la mejora de la calidad asistencial en los pacientes críticos. Una de las tareas que consideran relevantes para la mejora de la calidad en aquellos pacientes que están en el tránsito de la UCI a la recuperación cotidiana es la evaluación y monitorización de los avances relativos a la movilidad de dichos pacientes. Los pacientes habitualmente han perdido gran cantidad de masa muscular debido al largo periodo de inactividad, siendo el trabajo de los fisioterapeutas la única forma de la estimulación muscular. El interés del servicio UCI radica en disponer de una herramienta que permita al paciente de forma autónoma realizar una serie de ejercicios de rehabilitación.

Este Trabajo de Fin de Máster evalúa el dispositivo Nicla Sense ME para este propósito mediante el cuál se realiza una monitorización de una serie de ejercicios planteados para la estimulación y la recuperación de los enfermos que han pasado por la UCI.

La primera parte de la investigación consiste en comparar las mediciones de los diferentes sensores para selecionar las medidas que más se ajustan al objetivo del proyecto. De todas las mediciones realizadas finalmente fueron extraídas la orientación del sensor y la aceleración lineal, ya que son las medidas que ofrecen la representación más fiel a los ejercicios planteados, así como proporcionar los datos necesarios para los cálculos de que son realizados para la monitorización de los mismos, por ejemplo, el número de repeticiones realizadas de un ejercicio en concreto.

Finalmente ha sido desarrollada una aplicación que facilita la visualización de las medidas de los sensores empleados proporcionando una interfaz que permite al usuario seleccionar el ejercicio a realizar, así como unos indicadores visuales para la correcta realización de los ejercicios planteados. Esos ejercicios son guardados en una base de datos en la nube y pueden ser revisados en la aplicación, así como interactuar con las gráficas obtenidas a partir de esos datos.

Palabras clave

Ejercicio, rehabilitación, monitorización, sensores, IoT, movimiento, UCI, recuperación.

[1] https://twitter.com/innovahuci

Abstract

Stimulation and monitoring of patients in the ICU

The Intensive Care Unit service at the Fuenlabrada University Hospital has been working in recent years on the InnovaHUCI Project [2](de cuidados intensivos, 2018), the main objective of which is to improve the quality of care for critically ill patients. One of the tasks that it is considered relevant for improving the quality of those patients who are in transit from the ICU to daily recovery is the evaluation and monitoring of the advances related to the mobility of these patients. Patients have usually lost a large amount of muscle mass due to the long period of inactivity, with the work of physiotherapists being the only way of stimulation. The interest of the ICU service lies in having a tool that allows the patient to autonomously carry out a series of rehabilitation exercises.

This Master's Thesis evaluates the Nicla Sense ME device for this purpose, through which a series of exercises proposed for the stimulation and recovery of patients during the ICU can be monitored.

The first part of this project consists of comparing the measurements of the different sensors to select the measurements that best fit the objective of the project. Finally, the orientation of the sensor and the linear acceleration were seen as the most suitable fit measurements, since they offer the most faithful representation of the proposed exercises, as well as provide the necessary data to count the number of repetitions performed of a specific exercise.

Finally, an application has been developed that facilitates the monitoring of the measurements of the sensors used, providing an interface that allows to a single user to select the exercise to be performed, as well as visual parameters for the correct performance of the proposed exercises. It is important to note that these exercises are stored in a database in the cloud allowing to medical staff the offline revision of patient exercises. In addition, the tool stores the patient's exercise history, which allows evaluating progress and adapting the exercise table.

Keywords

Exercise, rehabilitation, monitoring, sensors, IoT, movement, ICU, recovery.

[2]https://twitter.com/innovahuci

Índice

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

La Unidad de Cuidados Intensivos o UCI es una sección de un hospital dónde se le proporciona a pacientes con problemas de salud potencialmente mortales una monitorización y una asistencia intensiva de forma constante. Tras haber sufrido un accidente, operación o enfermedad lo suficientemente grave sería necesaria la estancia de forma temporal en una de estas unidades.

Tras varios años bajo la presencia del Covid-19 en nuestras vidas, cabe destacar la importancia de la Unidad de Cuidados Intensivos, ya que en muchas ocasiones el virus ha supuesto un peligro potencialmente mortal para las personas. Al haber aumentado significativamente el número de casos de personas que acaban internas en la UCI es interesante mejorar los métodos de recuperación de los pacientes. La tecnología que proporciona el mundo del Internet de las Cosas abre la puerta a la monitorización a distancia de pacientes, para así evitar colapsos en las salas de estas Unidades de Cuidados Intensivos, además de agilizar y mejorar el trato del personal sanitario con el paciente. De esta manera, aquellos que ya están fuera de riesgo mortal pueden adquirir una recuperación en sus casas sin la ocupación de una sala mientras pueden seguir siendo monitorizados por los médicos. Además, al no estar utilizando material sanitario se estarían reduciendo costes tanto en los servicios médicos como el mantenimiento de los equipos mirando a largo plazo.

Las estancias en la UCI suponen un gran desgaste tanto físico como mental para el paciente debido a los prolongados intervalos de tiempo que debe estar en una camilla sin realizar apenas movimientos. Estos tiempos pueden llegar de unos días o un par de semanas a varios meses donde el músculo, entre otras cosas, queda deteriorado.

El resultado del deterioro es tal que el paciente no puede realizar los movimientos básicos sin ayuda externa, por lo que requiere de un proceso de rehabilitación para recuperar de nuevo la movilidad y las capacidades motoras que le permitan realizar las funciones básicas del día a día. A día de hoy esta tarea recae en el servicio de fisioterapia, pero en algunos casos los propios pacientes puede comenzar a realizar el proceso de rehabilitación por ellos mismos si se dispusiese de una herramienta para la motivación, y el control de los avances de los ejercicios.

En este Trabajo de Fin de Máster se ha realizado la investigación necesaria para cubrir las necesidades básicas de un programa el cual realiza un seguimiento del paciente con el fin de determinar si está realizando los ejercicios propuestos de forma correcta o no. Además, el sistema propuesto permite al médico llevar un seguimiento del paciente con resultados en tiempo real, de manera que puede observar la constancia y la mejoría de los ejercicios realizados.

1.1. Motivación

En los últimos años hemos podido observar como se utilizan diferentes sensores para la creación de relojes inteligentes que monitorizan todo tipo de ejercicios, desde correr hasta levantar pesas. Esos mismos sensores cada vez son más compactos y con muy bajo coste, siendo una gran herramienta para la monitorización de cualquier tipo de ejercicios, incluidos aquellos relaccionados con la rehabilitación.

La aplicación de estos sensores en usos relacionados con la medicina es un fenómeno en auge, siendo la telemedicina uno de los ejemplos más conocidos Bojan Milosevic (2020). Los usos de sensores en telemedicina son muy variados, por ejemplo el uso de IMU (Unidad de Medición Inercial) en la identificación de catacterísticas motoras patológicas Saber-Sheikh et al. (2010); Shull et al. (2014). Un caso concreto es el uso de estas IMU para evaluar la progresión de ciertas enfermedades motoras como puede ser la enfermedad de Párkinson, pudiendo diferenciar pacientes con temblor dominante a pacientes sin ese temblor dominante. "Durante el estudio se encontró una correlación entre la gravedad de la enfermedad y la movilidad que tenía el paciente, indicando así que la enfermedad se había desarrollado"(PT y more, 2018). Otro ejemplo de uso de IMUs en medicina es la detección de lesiones deportivas en corredores Ricci et al. (2016), dónde se puede observar en los corredores con algún tipo de lesión que los índices de carga vertical son mucho mayores en aquellos con lesión respecto a los que no están lesionados.

La telerehabilitación también está en el punto de mira de médicos y fisioterapeutas, de manera que puedan ayudar a los pacientes de manera constante en su fase de rehabilitación. El uso de IMUs abre una puerta a este gran mundo de la telerehabilitación donde permite la constante monitorización de los pacientes en lugar de las visitas periódicas al centro médico.

Durante el desarrollo de este Trabajo de Fin de Máster ha sido posible contar con la ayuda del servicio de UCI del hospital de Fuenlabrada, el cual propone disponer de un sistema IoT que cumpla dos objetivos:

- Recogida de información asociada a ejercicios de rehabilitación. Empleando sensores en los pacientes colocados en sus extremidades para recoger esa información que permite al personal médico evaluar los progresos del paciente y adecuar la tabla de ejercicios de rehabilitación.

- Feedback-motivación al paciente para avanzar en el proceso de rehabilihación. Ofreciendo una interfaz interactiva para el usuario donde pueda ver su progreso en todo momento, así como el resto de ejercicios de la tabla propuesta por el personal médico con el fin de observar su mejoría y motivándolo a continuar con los ejercicios.

Los ejercicios de rehabilitación propuestos por el personal médico del hospital de Fuenlabrada son los siguientes:

- El paciente flexiona el codo llevando su mano al hombro.

- El paciente encoge los hombros llevándolos hacia su cabeza.

- El paciente eleva el brazo hacia el techo, de un lado o de ambos.

- El paciente eleva el pie hacia la rodilla, de un lado o de ambos de forma alterna.

- El paciente eleva la pierna de un lado o de ambos de forma alterna.

- El paciente cierra el puño comprimiendo un objeto.

Alternativamente, ofrecer la posibilidad de personalizar los ejercicios a base de las necesidades de los pacientes para optimizar la recuperación de los estos. De esta manera en caso de no poder seguir los ejercicios propuestos y evitar así lesiones accidentales, o en caso de haber obtenido un buen rendimiento con los ejercicios también sería posible el ligero endurecimiento de estos. Además para aquellos pacientes con enfermedades progresivas también permite que se adapte el ritmo de aquellos ejercicios con el fin de frenar estas enfermedades.

Además, la elección de utilizar el sensor Nicla Sense ME se encuadrada en el acuerdo UCM-Bosch cuyo resultado más tangible es la creación de la Cátedra de Bosch(BOSCH, 2022) lo que ha permitido la donación de las placas para la realización del Trabajo de Fin de Máster.

1.2. Estado del arte

Como ya se comentaba en la introducción, existen numerosas aplicaciones para combinar los beneficios que traen los sistemas del Internet de las Cosas con la medicina.

El sistema propuesto por la Universidad de Edge Hill (McHale, 2020) sugiere la posibilidad de crear un sistema IoT para la monitorización remota de personas que sufren epilepsia. Debido a la complejidad que este problema supone ya que cada uno de los casos de epilepsia son diferentes propone un sistema completamente personalizable que se adapte a cada individuo para que las tecnologías IoT sean mucho más precisas en la detección de estos problemas. La forma de detectar estos ataques epilépticos es mediante el uso de técnicas de clasificación y el análisis de clústers con el fin de crear grupos para cada tipo de paciente de párkinson. La forma de capturar estos datos es mediante sensores colocados en diferentes partes del individuo, que a su vez están conectados a un servicio IoT dónde serán analizados esos datos. Debido a la variedad de los focos de los ataques epilépticos, cada individuo tendrá un plan personalizado de monitorización donde contará con sensores propios que serán diferentes a otros pacientes que sufran diferentes tipos de ataques.

Otro de los usos de sistemas IoT integrados en medicina es el uso de Kinect (Bojan Milosevic, 2020) para la monitorización de programas de rehabilitación de forma remota. Kinect es una cámara de video de Microsoft de muy bajo coste, la cual serviría como dispositivo de entrada para sensorizar los movimientos que realizan los pacientes. Se propone el uso de diferentes algoritmos, como un algoritmo que es proporcionado en el SDK del propio Kinect para determinar la posición del cuerpo al ser grabados por la videocámara y así determinar si las posturas realizadas son las correctas. También plantean el potencial uso de diferentes sensores como la orientación calculada a partir de giróscopos y acelerómetros, así como el uso de diferentes filtros con el fin de mejorar la precisión de estas medidas. Al contar con un sistema IoT es posible la rehabilitación completa desde casa obteniendo resultados positivos para una gran cantidad de ejercicios realizados Chiang et al. (2017).

1.3. Objetivos

Los objetivos propuestos para cumplir con los requisitos del sistema son los siguientes:

- Explorar las herramientas de programación de Arduino para volcar el código en la placa de desarrollo Nicla Sense ME.
- Explorar capacidades de la placa de desarrollo Nicla Sense ME.

- Analizar las diferentes medidas que ofrece la placa de desarrollo Nicla Sense ME para localizar las medidas necesarias para realizar la monitorización de los ejericicios.
- Investigar como poder realizar la comunicación entre la placa de desarrollo y el programa principal.
- Desarrollar una aplicación que permita la correcta monitorización de los ejercicios propuestos por los fisioterapeutas.
- Creación de un registro de ejercicios para el seguimiento de la evolución de los pacientes en el tiempo.

1.4. Plan de trabajo

Este Trabajo de Fin de Máster ha sido estructurado siguiendo los objetivos planteados en la sección anterior, de manera que para el análisis o el desarrollo de cada objetivo fue dedicada una cantidad de tiempo determinada.

1.4.1. Exploración Arduino IDE

Los primeros días fueron dedicados a la instalación y exploración de Arduino IDE. Este programa permite la subida de código de manera sencilla a las placas de desarrollo. La instalación del programa llevó más tiempo de lo esperado porque había varias versiones del mismo y no todas funcionaban correctamente. Tras encontrar e instalar la última versión, fue necesaria la instalación de diversos módulos de librerías que permiten la programación sobre la placa Nicla Sense ME. Además, también fue dedicado tiempo extra a la exploración de la visualización de los datos, pues pueden ser visualizados mediante el puerto serie del cual está conectada la placa de desarrollo directamente, donde se muestran los datos como un terminal, o alternativamente utilizar el "serial plotter" para visualizar los datos mediante gráficos de líneas.

1.4.2. Exploración Nicla Sense ME

Las siguientes dos semanas fueron una primera toma de contacto con la placa de desarrollo Nicla Sense Me. Para ello fueron replicados algunos ejercicios que están disponibles en la página de documentación de la placa. El primero de los ejercicios a replicar fue modificar el LED que viene integrado en la placa de desarrollo alternando colores aleatorios en intervalos de 1 segundo. El siguiente ejercicio para entender un poco más el Nicla Sense ME fue imprimir tanto por el puerto serie como en el serial plotter los datos de varios sensores. Los primeros datos a mostrar fueron la orientación del sensor, los datos del acelerómetro y los datos obtenidos del giroscopio, para sus respectivos ejes X, Y y Z.

Finalmente la última prueba a realizar con el sensor fue establecer una conexión mediante Bluetooth con un dashboard o tablero de datos cuyo código está disponible en la documentación de la placa de desarrollo, así como un enlace al propio dashboard para la visualización de estos datos. Tras la conexión de la placa con el dashboard fue posible la visualización de todos los datos del Nicla Sense ME, así como una representación con un modelo 3D de la placa que reproducía los movimientos de la propia placa en tiempo real, la cual fue la principal inspiración para el desarrollo de la aplicación de monitorización.

1.4.3. Análisis de los datos de los sensores

Las siguientes tres semanas fueron dedicadas íntegramente a realizar pruebas sobre los diferentes datos obtenidos para determinar las mediciones de los diferentes sensores que pudieran resultar útiles, descartando aquellos sensores que no aportaban información útil para el desarrollo de esta práctica. Entre esas pruebas destacan el cálculo de los ángulos de orientación a partir de los sensores de acelerómetro y giroscopio, añadiendo filtros adicionales para evitar datos erróneos y la prueba de todos los sensores que tuvieran relación con el movimiento o la orientación de la placa de desarrollo.

1.4.4. Comunicación con la aplicación

Los siguientes dos días fueron dedicados a la creación de un pequeño programa escrito en el lenguaje de programación Python para establecer una conexión vía Bluetooth con la placa de desarrollo.

1.4.5. Desarrollo de la aplicación

Para el desarrollo de la aplicación fue asignado el mayor porcentaje de tiempo respecto con los otros objetivos, ya que era la tarea que más trabajo y más pruebas requería para su correcto funcionamiento. Esta etapa tiene una duración aproximada de un mes y medio o dos meses, donde se trabajó paralelamente en el análisis de los datos de los sensores y en la creación de un registro de ejercicios. Durante este tiempo fueron hechas diversas pruebas con Dash, la librería de Python en la que está basada principalmente la aplicación final. Estas pruebas consistían en probar las diferentes herramientas que ofrece Dash, como la posibilidad de crear un dashboard completamente interactivo con diferentes formas de interactuar con los datos y visualizarlos, así como la posibilidad de añadir modelos 3D para la representación del movimiento del sensor.

1.4.6. Creación de un registro de ejercicios

La exploración de una base de datos para guardar el registro de todos los ejercicios realizados fue llevada a cabo durante aproximadamente tres días. El primer día fue dedicado a la investigación de una base de datos SQL alojada de forma local en el ordenador. Tras varias pruebas y volver a estructurar los objetivos del proyecto, la decisión de utilizar esa base de datos cambió por la posibilidad de utilizar una base de datos en la nube utilizando MongoDB, la cual supondría una decisión más acertada para el desarrollo de un sistema IoT. Dos días fueron necesarios para la investigación, y la creación de la base de datos en la nube, así como su integración en la aplicación.

1.4.7. Extras

Los últimos días dedicados al desarrollo de este Trabajo de Fin de Máster fueron dedicados a la investigación de posibles mejoras o añadidos a la aplicación final. Las investigaciones realizadas fueron la posibilidad de añadir una inteligencia artificial a la cámara encargada de grabar los movimientos de los pacientes al realizar los ejercicios, de manera que esa inteligencia artificial pueda indicarnos si la postura del paciente es errónea o es la adecuada para el ejercicio en cuestión. Además, fue investigada la posibilidad de añadir una segunda placa de desarrollo Nicla Sense ME con el fin de monitorizar varias extremidades al mismo tiempo.

Capítulo 2

Nicla Sense ME

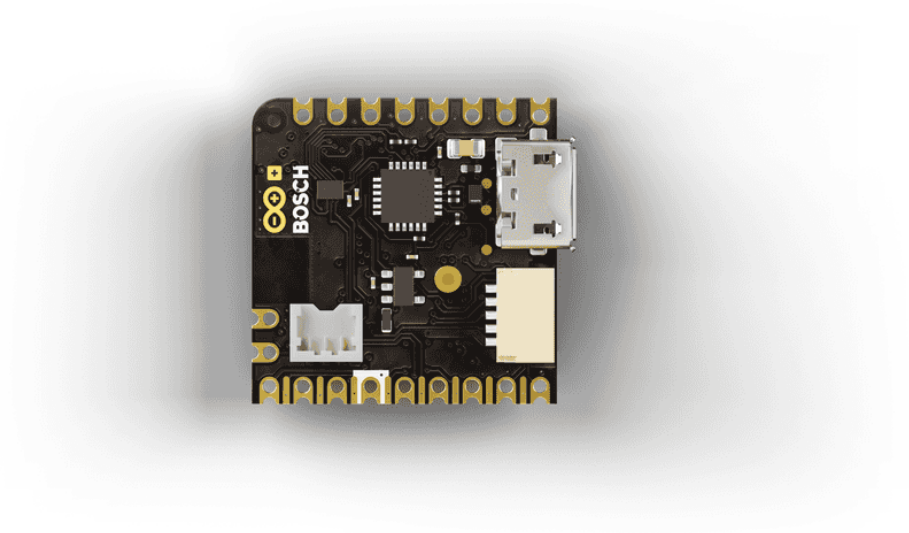

Figura 2.1: Placa de desarrollo Nicla Sense ME (Romero, 2022)

Nicla Sense ME (cuyas siglas significan Movimiento y Entorno) (Romero, 2022) es una de las placas de desarrollo más pequeñas hechas hasta la fecha. Esta pequeña placa permite a los usuarios la creación de sistemas IoT de forma sencilla, ya que está integrada en el ecosistema Arduino y posee varios sensores para la recolección de la información. La placa es muy compacta y uno de sus puntos fuertes es el ahorro de energía, permitiendo su uso continuado durante muchas horas sin costes energéticos elevados. Con tan solo 23 milímetros cuadrados y 2 gramos de peso la convierten en el candidato perfecto para su colocación en una pulsera y cumplir así los objetivos propuestos en este trabajo. Además, cuenta con la posibilidad de conectar una batería portátil, ya sea por el puerto micro USB (USB-B) o por el conector de una batería de litio (Li-ion/Li-Po), permitiendo su portabilidad completa sin necesidad de estar conectado a ningún ordenador. Esta placa de desarrollo cuenta con los siguientes sensores integrados de Bosch Sensortec:

- **BHI260AP:** sensor inteligente de autoaprendizaje de IA con acelerómetro y giroscopio integrados
- **BMP390:** sensor de presión digital

- **BMM150:** Sensor geomagnético
- **BME688:** sensor digital de gas, presión, temperatura y humedad de bajo consumo con IA

De todos esos sensores el que más se ajusta para cumplir los objetivos previstos es el sensor **BHI260AP**, ya que contiene los sensores inteligentes con IMU integrada de 6 ejes (acelerómetro de 3 ejes + giroscopio de 3 ejes) para detección de actividad, alimentado por una CPU Synopsys DesignWare ARC™ EM4™ de 32 bits (electronics, 2021). Este sensor está principalmente diseñado para su uso en aplicaciones de seguimiento de fitness, el seguimiento de la posición o ubicación de una persona, estimación de la orientación e incluso posee una inteligencia artificial que permite el análisis de aprendizaje automático. Además este sensor tiene una solución dedicada para el seguimiento de ejercicios de natación, abriendo la posibilidad de emplearlo en ejercicios de rehabilitación en piscina.

El resto de sensores de la placa no tienen utilidad ninguna para este proyecto, por lo que no han sido investigados más a fondo.

2.1. Arduino IDE

Arduino IDE (Integrated Development Environment) es el entorno de desarrollo de Arduino (Arduino, 2022b) que permite la creación y depuración de programas además de la subida del mismo código a las placas de desarrollo de la misma marca de manera sumamente sencilla. Está escrita en el lenguaje de programación Java y es un programa de código abierto. Para la creación de programas se utiliza generalmente el lenguaje de programación C o C++ y en este entorno existen multitud de librerías que facilitan enormemente las tareas más complejas. La estructura general de cualquier programa de estas características requiere la preparación y activación de los sensores o partes del hardware que van a ser utilizados y una función principal que va a estar ejecutándose en un bucle infinito realizando las tareas que necesitemos.

Para grabar el código en las placas de Arduino es necesario tener conectado el dispositivo mediante el puerto USB al ordenador, y finalmente verificar y subir el código deseado.

Esta herramienta permite acceder a los diferentes sensores de la placa de desarrollo Nicla Sense ME de una manera muy sencilla. Como se puede observar en el código mostrado a continuación los únicos pasos a seguir para conseguir acceder a los sensores de acelerómetro y del giróscopo son definir los sensores de acelerómetro y giróscopo mediante las macros establecidas por Arduino que contienen las identificaciones de estos sensores. Además es necesario inicializar el sensor BHY2, así como los componentes definidos anteriormente, y finalmente en el bucle principal es necesario mantener actualizado el sensor BHY2. Tras eso es posible acceder a los diferentes valores proporcionados por los sensores giróscopo y acelerómetro.

```
#include ''Arduino_BHY2.h''

SensorXYZ accelerometer(SENSOR_ID_ACC);
SensorXYZ gyro(SENSOR_ID_GYRO);
void setup(){
  Serial.begin(115200);
  BHY2.begin();

  accelerometer.begin();
```

```
  gyro.begin();
}
void loop(){
  BHY2.update();
    Serial.println(accelerometer.toString());
    Serial.println(gyro.toString());
  }
}
```

2.1.1. Bluetooth Low Energy

Bluetooth Low energy (BLE) es una alternativa a Bluetooth que proporciona una serie de protocolos dedicados a la comunicación entre dispositivos pero orientada a conexiones mucho más sencillas requiriendo de esta manera una menor potencia. De esta manera se consigue una conexión eficiente teniendo muy bajo coste pues no va a ser necesario el envío masivo de datos ni la aplicación va a ser lo suficientemente compleja como para necesitar mayor cantidad de datos (ELT, Último acceso 2022; Arduino, Último acceso 2022).

A diferencia de Bluetooth que establece las comunicaciones basadas en conexiones asíncronas por medio de UART (Transmisor-Receptor Asíncrono Universal), BLE utiliza un sistema de publicación y suscripción en forma de tablero de anuncios comunitario. Es decir, si un dispositivo está suscrito a un tema concreto de ese "tablón" recibirá la información en el momento que se ha enviado, de manera que este dispositivo está escuchando a la espera de nueva información de forma continuada. De esta misma manera, todos los dispositivos que estén suscritos al mismo tema recibirán exactamente la misma información que ha sido publicada por el dispositivo central. El funcionamiento es parecido a la comunicación establecida entre un cliente y servidor, siendo el servidor el dispositivo que publica los datos y los clientes los que leen la información disponible.

Alternativamente, los otros dispositivos que están a la escucha también pueden actualizar las características si tienen que aportar algún tipo de información relevante para el dispositivo central.

El tablón metafórico donde van a ser publicados los datos está estructurado en **servicios**, y estos a su vez en **características**. Para entenderlo mejor los servicios son como los capítulos de un libro y las características serían los diferentes párrafos de cada capítulo, como se muestra en la Figura 2.2. Para diferenciar cada una de las características van identificadas con un UUID de 16 bits generalmente aunque este número puede llegar a los 128 bits en caso de tener un sistema más complejo que necesite mayor cantidad de características.

Una cualidad de la que dispone BLE también es el mecanismo de **notificaciones**, por la cual advierte al dispositivo de que han ocurrido cambios en los datos. Cuando las notificaciones están activadas sobre una característica y se escribe sobre esa característica, el valor se envía automáticamente al receptor sin estar esperando a que le llegue la información con un comando de lectura. Este tipo de mecanismos son usados para la recepción de datos de sensores que están continuamente enviándose.

La estructura cliente-servidor de Bluetooth Low Energy, combinada con la característica de notificación, generalmente se denomina modelo de publicación y suscripción.

2.1.1.1. Dashboard Nicla

Para la correcta visualización de los datos en un ordenador, Nicla ofrece un dashboard interactivo como el que se muestra en la Figura 2.3 (Marquínez, 2022) que recibirá los

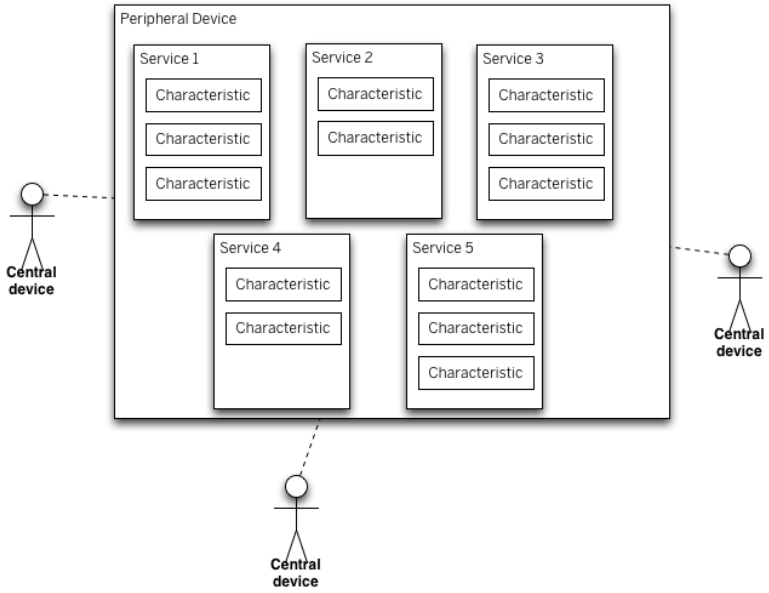

Figura 2.2: Esquema BLE (Arduino, Último acceso 2022)

datos enviados por la placa de desarrollo mostrando en diferentes gráficas los valores de los sensores de la placa, así como un modelo en 3D del Nicla para mostrar su correcta orientación. El dashboard está escrito íntegramente en HTML y Javascript y su código puede ser accedido a través de la página de documentación Arduino (2022a). Al tener acceso al código se abrió la posibilidad de modificarlo con el fin de añadir más recursos o eliminar del dashboard los datos innecesarios. Con fines de prueba el dashboard fue modificado eliminado los datos correspondientes a los sensores de temperatura, CO2, humedad, presión, gas, calidad de aire y el panel para modificar la luz del LED de forma remota, ya que no aportaban nada a la investigación. Adicionalmente fue añadido un espacio para la cámara web y una serie de botones y controles que servirían como primer prototipo de la aplicación. Este dashboard fue la inspiración directa para la creación de la aplicación final, ya que se ajustaba bastante a la idea que se tenía en mente al comenzar con el proyecto. Este dashboard fue solamente utilizado con la intención de hacer pruebas, ya que solamente se podía abrir en Google Chrome y conectar un dispositivo al mismo tiempo, por lo que fue descartada la idea de seguir desarrollando el resto del programa sobre este código.

2.1.2. Explicación del código final subido en la placa

Listing 2.1: Código simplificado subido a la placa

```
#include ''Nicla_System.h''
#include ''Arduino_BHY2.h''
#include <ArduinoBLE.h>
#define BLE_SENSE_UUID(val)(
  ''19b10000'' val ''-537e-4f6c-d104768a1214''
)
BLEService service(BLE_SENSE_UUID(''0000''));
BLECharacteristic orientationCharacteristic(
  BLE_SENSE_UUID(''9001''), BLERead | BLENotify, 3 * sizeof(int32_t)
);
```

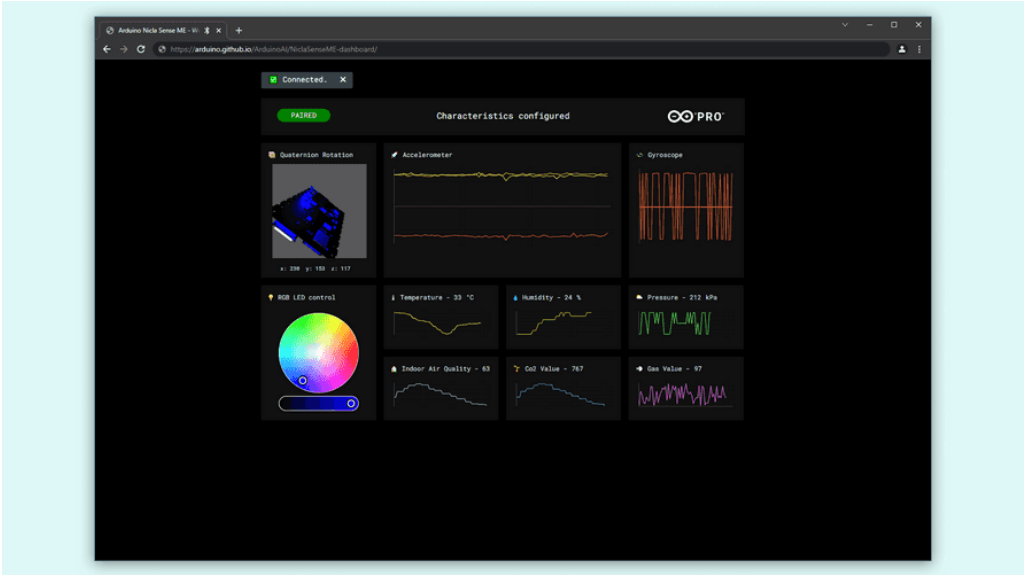

Figura 2.3: Dashboard Nicla Sense ME.(Marquínez, 2022)

```
SensorOrientation orientation(SENSOR_ID_ORI);
void setup(){
    Serial.begin(115200);

    Serial.println(''Start'');

    nicla::begin();
    nicla::leds.begin();
    nicla::leds.setColor(green);

    BHY2.begin();
    orientation.begin();
    if (!BLE.begin()){
      Serial.println(''Failed to initialized BLE'');

      while (1)
        ;
    }

    String address = BLE.address();
    BLE.setLocalName(name.c_str());
    BLE.setDeviceName(name.c_str());
    BLE.setAdvertisedService(service);

    service.addCharacteristic(orientationCharacteristic);

    BLE.addService(service);
    BLE.advertise();
}
void loop(){
    while (BLE.connected()){
      BHY2.update();

     if (orientationCharacteristic.subscribed()){
```

```
            float x, y, z;

            x = orientation.pitch();
            y = orientation.roll();
            z = orientation.heading();

            int16_t oriValues[3] = {x, y, z};

            orientationCharacteristic.writeValue(
              oriValues, sizeof(oriValues));
          }
        }
    }
```

El código mostrado en el Listing anterior muestra el flujo de ejecución de manera simplificada que va a estar siendo ejecutado en la placa de desarrollo Nicla Sense ME. En las primeras líneas de código podemos observar la importación de las librerías explicadas a lo largo de esta sección, además de la declaración del UUID del dispositivo y de la característica a la que va a enviar los datos del sensor de orientación, el cual va a ser definido en la siguiente línea de código. La elección del sensor se decide mediante la Macro "SENSOR_ID_ORI" la cual representa el ID de la dirección correspondiente del sensor de orientación. Cada sensor tiene uno o varios Macros diferentes para cada medición de cada sensor.

Tras las definiciones está la función setup, la cual tiene el objetivo de inicializar el puerto serie, los sensores a utilizar y añadir las características necesarias a los servicios definidos con anterioridad, así como añadir el servicio a la lista de servicios que van a ser publicados. En este caso solamente hay una característica siendo añadida al servicio, pues se trata de una versión simplificada del código original.

Finalmente la función loop es la función principal del programa, que está compuesta por un bucle que se ejecuta infinitamente mientras se haya establecido conexión con BLE. En cada vuelta del bucle el dispositivo va a ser actualizado para obtener valores obtenidos de los sensores más recientes y desglosará los valores en las diferentes características. Cada característica pertenece a un sensor, en este caso para la característica llamada "orientationCaracteristic" serán añadidos los valores correspondientes obtenidos por el sensor orientación. Para enviar los datos de una forma más ordenada se juntan los datos de las coordenadas X, Y y Z en una estructura.

Capítulo 3

Sistema desarrollado

Como ya se ha comentado en la introducción del trabajo, el objetivo principal de este proyecto es la creación de un sistema IoT que permita la monitorización de ejercicios para la rehabilitación de pacientes que acaban de terminar su estancia en la UCI. El objetivo es ser capaces de monitorizar los ejercicios para determinar si han sido completados de forma correcta o no y además poder llevar un seguimiento de los pacientes con el fin de poder observar el progreso de su recuperación.

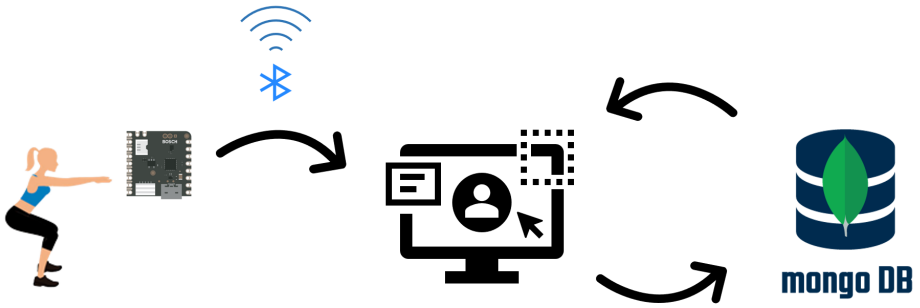

Figura 3.1: Representación del sistema propuesto (GARCÍA, 2017; Jadhav, 2021; ICONS, Último acceso 2022; Router, Último acceso 2022)

La Figura 3.1 muestra la estructura del sistema propuesto, el cual consta de tres componentes principales. Un sensor encargado de captar los movimientos realizados por el paciente, que lo tendrá colocado en las partes del cuerpo que le correspondan para la monitorización de cada uno de los ejercicios. Una aplicación central encargada de ofrecer una interfaz al usuario para que pueda utilizarlo de forma sencilla mostrando los resultados de los ejercicios a realizar y que se comunica directamente con la placa de desarrollo vía Bluetooth. Por último, una base de datos en la nube encargada de almacenar los datos de los ejercicios realizados para la futura visualización de los datos generados en dichos ejercicios.

3.1. Medidas a utilizar

La placa de desarrollo Nicla Sense ME ofrece una gran variedad de sensores dentro de su kit que permiten la recolección de medidas de diferentes tipos de movimientos. Todos los datos que tuvieran relevancia con la captación de movimiento fueron probados con el fin de determinar cuáles son las medidas adecuadas para el desarrollo de este Trabajo de Fin de Máster. Las medidas más relevantes ofrecidas por el sensor Nicla son las siguientes:

- **Acelerómetro:** Sensor que sirve para medir la aceleración gravitacional estática, ya sea la aceleración lineal o angular. Estos dispositivos miden la fuerza de la aceleración en la unidad g, la cual puede ser medida en varios planos. En este caso se medirá en el eje X, Y y Z. Si el acelerómetro no se ve sometido a ningún tipo de aceleración, es decir, está en reposo este mostrará los valores de g en todos sus ejes (TME, 2020). Estos sensores funcionan gracias a contener una gran cantidad de pequeños objetos suspendidos encima de muelles. Al realizar un movimiento sobre el sensor estos objetos se ven afectados y la diferencia de movimiento respecto a su estado inicial es recogida por el sensor.
- **Giróscopo:** Sensor que dectecta la velocidad angular, es decir, el ángulo rotacional por unidad de tiempo. Las unidades de velocidad angular se miden en grados / segundo o en revoluciones por segundo (Gordon, 2021).
- **Orientación:** Medida calculada a través del acelerómetro y el giróscopo que muestra la orientación de los ejes X, Y y Z en ángulos de Euler.
- **Quaternión:** Número complejo de cuatro dimensiones que representa una sola orientación o rotación en el espacio tridimensional. Realizando una serie de cálculos pueden ser convertidos en ángulos de Euler de forma que pueden ser utilizados para la medición de los ejercicios propuestos.

La lectura de los datos de los sensores puede ser obtenida leyendo los datos directamente de la placa a través del modo independiente (standalone mode), mediante Bluetooth Low Energy o mediante UART conectando un cable ESLOV. Para ello es necesaria la instalación de las librerías Arduino_BHY2 y Arduino_BHY2Host. Para localizar cada uno de los sensores a utilizar es necesario el empleo de las Macros que contienen las ID de cada uno de los sensores. Al existir numerosas cantidades de ID diferentes para cada sensor, fue necesaria la comparación de los valores de todas ellas, de modo que finalmente serían utilizadas aquellas que posean una mayor precisión en los datos, o que aporten información relevante. Todas las ID que corresponden con la monitorización del movimiento están representadas en la Tabla 3.1. En esta primera fase de análisis fueron probadas las ID de la 1-16, junto con las ID 43 y 44 para determinar que valores tomar de cada sensor. El resto de medidas se corresponden con el uso de inteligencia artificial para la detección de determinados gestos, como puede ser mover la muñeca, o los pasos que da una persona al andar, que para la monitorización de estos ejercicios no son relevantes aunque para futuras implementaciones pueden tener un uso. Otras de estas medidas, como la aceleración lineal se utilizan más adelante para el desarrollo de un sistema de detección de repeticiones del mismo ejercicio.

3.1.1. Comparación de datos

En esta primera prueba van a ser comparados entre sí por medio de gráficos de líneas todas las posibles variaciones del mismo sensor. Por ejemplo todos los valores del acele-

ID	Descripción	SENSOR ID MACRO
1	Accelerometer passthrough	SENSOR_ID_ACC_PASS
3	Accelerometer uncalibrated	SENSOR_ID_ACC_RAW
4	Accelerometer corrected	SENSOR_ID_ACC
5	Accelerometer offset	SENSOR_ID_ACC_BIAS
6	Accelerometer corrected wake up	SENSOR_ID_ACC_WU
7	Accelerometer uncalibrated wake up	SENSOR_ID_ACC_RAW_WU
10	Gyroscope passthrough	SENSOR_ID_GYRO_PASS
12	Gyroscope uncalibrated	SENSOR_ID_GYRO_RAW
13	Gyroscope corrected	SENSOR_ID_GYRO
14	Gyroscope offset	SENSOR_ID_GYRO_BIAS
15	Gyroscope wake up	SENSOR_ID_GYRO_WU
16	Gyroscope uncalibrated wake up	SENSOR_ID_GYRO_RAW_WU
31	Linear acceleration	SENSOR_ID_LACC
32	Linear acceleration wake up	SENSOR_ID_LACC_WU
43	Orientation	SENSOR_ID_ORI
44	Orientation wake up	SENSOR_ID_ORI_WU
50	Step detector	SENSOR_ID_STD
52	Step counter	SENSOR_ID_STC
53	Step counter wake up	SENSOR_ID_STC_WU
55	Significant motion	SENSOR_ID_SIG
57	Wake gesture	SENSOR_ID_WAKE_GESTURE
59	Glance gesture	SENSOR_ID_GLANCE_GESTURE
61	Pickup gesture	SENSOR_ID_PICKUP_GESTURE
63	Activity recognition	SENSOR_ID_AR
67	Wrist tilt gesture	SENSOR_ID_WRIST_TILT_GESTURE
69	Device orientation	SENSOR_ID_DEVICE_ORI
70	Device orientation wake up	SENSOR_ID_DEVICE_ORI_WU
75	Stationary detect	SENSOR_ID_STATIONARY_DET
77	Motion detect	SENSOR_ID_MOTION_DET
142	Any motion	SENSOR_ID_ANY_MOTION

Tabla 3.1: Tabla de IDs (Romero, 2022)

rómetro captados mientras se realiza un ejercicio concreto. Este ejercicio en cuestión es uno de los ejercicios de rehabilitación más simples que fueron propuestos en la lista de ejercicios a realizar y consiste en el movimiento del brazo partiendo desde una posición inicial de reposo con el cuerpo tumbado, teniendo que mover el brazo hacia arriba hasta llegar a intentar tocar el hombro del mismo brazo con los dedos. Este ejercicio será repetido durante el intervalo de tiempo de 50 segundos a 1 minuto para poder recopilar la mayor cantidad de información disponible. Para ayudar con la recolección de datos se hicieron modificaciones en el dashobard proporcionado por Nicla, de manera que se mostraban los valores de los acelerómetros, así como una cámara para poder grabar los movimientos y un botón que permite guardar los registros de todas las actualizaciones de los sensores que se almacenaban de forma inicial en arrays de Javascript. Ese prototipo de programa para probar los diferentes valores tiene el aspecto que se muestra en la Figura 3.2.

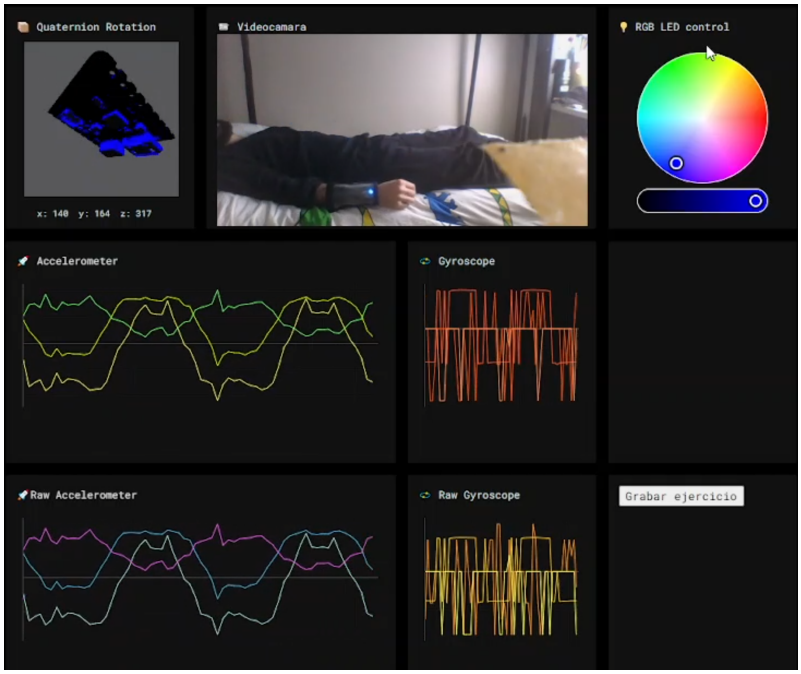

Figura 3.2: Dashboard modificado para el testeo de diferentes sensores

La placa de desarrollo Nicla Sense ME tiene una serie de **limitaciones** que impide la obtención de muchas ID de sensores diferentes al mismo tiempo. Esta limitación se aplica tanto al enviar datos por BLE como al intentar mostralos por el puerto serie. Fueron realizadas varias pruebas para intentar añadir el mayor número de sensores al mismo tiempo y tras varios intentos con diferentes números de ID la conclusión obtenida es que cuando sobrepasan las 8 ID al mismo tiempo los datos recibidos por el puerto serie son erróneos y se muestran a velocidades muy reducidas. De forma normal se pueden llegar a mostrar unas 5 o 6 actualizaciones de datos por segundo mientras que cuando la placa está sobrecargada puede llegar a realizar menos de una actualización por segundo. Estas limitaciones también se aplican al enviar los datos por BLE pero en este caso permite aún menos datos al mismo, bloqueando todo el sistema de publicaciones y suscripciones sin llegar a subir ningún tipo de actualización.

Es por eso que las medidas obtenidas para esta fase de análisis de los datos fueron obtenidas en varias series. La primera serie de sensorización fue la recogida de datos de todos los IDs referentes al acelerómetro, entre los que se encuentran: Accelerometer passthrough, Accelerometer uncalibrated, Accelerometer corrected, Accelerometer offset, Accelerometer

corrected wake up y Accelerometer uncalibrated wake up. Los resultados obtenidos son los que se muestran en la Figura 3.3 las cuales representan todos los valores del acelerómetro por cada eje X, Y o Z (un eje para cada gráfica diferente).

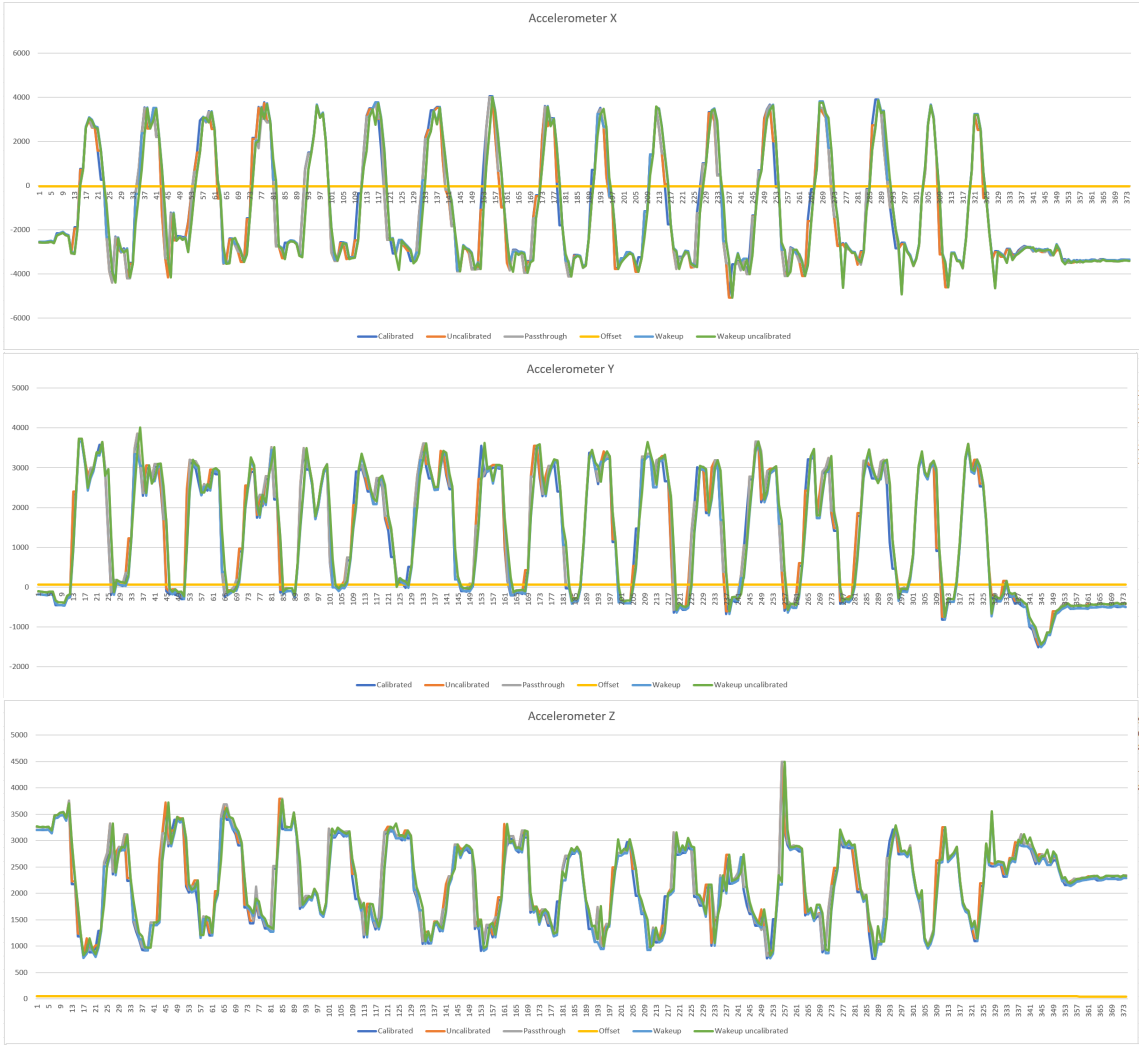

Figura 3.3: Datos acelerómetro obtenidos.

De los resultados obtenidos del acelerómetro podemos observar como todos los datos no poseen desviaciones significativas salvo por el offset del acelerómetro (representado en color amarillo) que mantiene un valor de forma constante y puede ser descartado como dato a tener en cuenta para el trabajo.

Dado que tras observar los resultados no se pueden apreciar diferencias lo suficientemente significativas en cada uno de estos valores la conclusión de este experimento es que se podría utilizar cualquiera de estos valores, pero va a ser utilizado definitivamente el acelerómetro calibrado para los cálculos posteriores, ya que promete ser el que mayor precisión posee. Podemos observar también en estos datos sacados del acelerómetro unos patrones de subida y de bajada que se corresponden con el movimiento del brazo al realizar los movimientos propuestos en el ejercicio de rehabilitación, por lo que este dato es relevante para futuros experimentos como contabilizar el número de repeticiones que se han realizado de ese mismo ejercicio.

Respecto al sensor giróscopo que pueden ser observados en la Figura 3.4, los datos

obtenidos son bastante más confusos, pues cada medida obtenida de los ID del sensor diferentes proporcionaba valores muy distantes entre ellos. De nuevo los valores del ID correspondiente al valor del sensor de giróscopo wakeup es un dato irrelevante, pues este se mantiene a 0 durante todo el recorrido, por lo que en este caso no ha sido incluido en la gráfica y puede ser descartado nuevamente del experimento al no aportar información relevante. Algunos valores estaban invertidos en algunos movimientos, otros se disparaban en momentos donde el sensor estaba prácticamente en reposo y otras veces se mantiene a 0 en casos dónde había mantenimiento. La única medida donde se puede llegar a observar algo de coherencia respecto a los movimientos es el sensor calibrado, pues al realizar la grabación de los datos en la aplicación desarrollada para estas pruebas se podía observar los valores de estos sensores de modo que el que respondía de forma correcta al movimiento realizado del sensor colocado en el brazo era este sensor.

Figura 3.4: Datos giróscopo obtenidos.

El sensor giroscopio sin realizar cálculos adicionales aparentemente no aporta información suficiente como para conocer el estado del sensor para los ejercicios propuestos para este trabajo. Las medidas obtenidas por este sensor serán utilizadas junto a las medidas del sensor acelerómetro para la creación de un filtro complementario para obtener los ángulos de Euler a partir de estos sensores. Todo este procedimiento estará explicado en la Sección

3.2.

La Figura 3.5 representa los valores obtenidos a partir de los datos del sensor con ID correspondiente a la orientación. Los resultados obtenidos corresponden de una manera precisa con el ángulo obtenido al inclinar el sensor en alguno de sus ángulos. Podemos observar los valores del eje X (representado en color azul) como al realizar el ejercicio los valores oscilan entre los valores (medidos en grados) 180 y 90 grados.

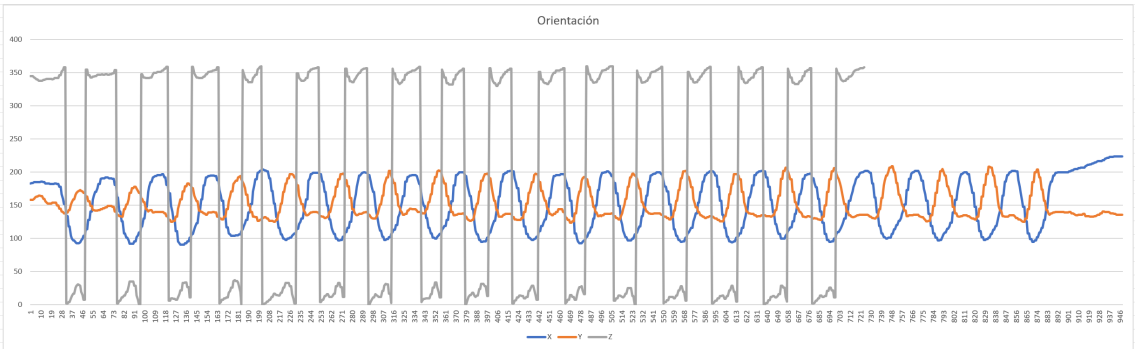

Figura 3.5: Angulos de Euler obtenidos por la ID orientación.

Estos grados podemos decir que son bastante precisos pues se corresponden con los ángulos que deben llegar las puntas de los dedos al realizar el ejercicio propuesto. La Figura 3.6 muestra como los ángulos máximos y mínimos corresponden de forma correcta con los obtenidos en la obtenida por la orientación.

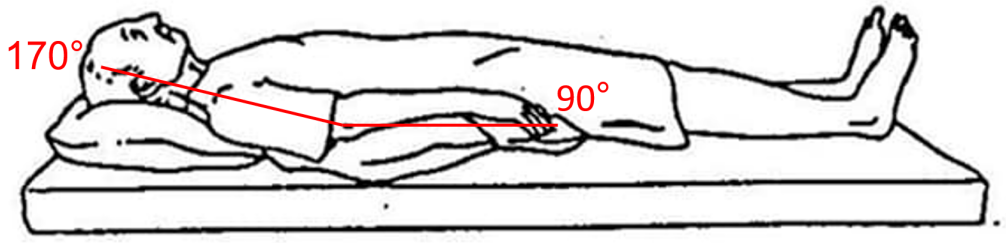

Figura 3.6: Rango de ángulos que debe tomar el ejercicio propuesto. Figura extraida de (mariaguutierrez, Último acceso 2022)

Tras los resultados obtenidos se puede deducir como la medida que más se ajusta para la monitorización de la correcta ejecución de los ejercicios es la orientación del sensor, pues de manera precisa muestra los ángulos obtenidos para cada eje. Estos datos son más precisos y permite la monitorización de manera más sencilla y manejable. Por ejemplo, los valores del acelerómetro que los movimientos de los datos se corresponden con los movimientos del sensor colocado en el brazo, pero al recibir valores que oscilan entre las 15.000 y -15.000 unidades en el eje X, lo convierte en un dato mucho más difícil de tratar que los valores recibidos por la orientación, que son ángulos con valores del 0 al 360.

Determinado esto, es necesario investigar también si los valores obtenidos por la placa de desarrollo correspondientes a la orientación del dispositivo son lo suficientemente óptimos y realistas, o hace falta la implementación de diversos cálculos para obtener de forma similar una orientación más precisa.

3.2. Cálculo de ángulos de Euler

Los ángulos de Euler describen una transformación rotacional al rotar un objeto en sus diversos ejes en cantidades específicas por eje y en un orden de eje específico (three.js, Último acceso 2022). Un ángulo de Euler expresa un ángulo 3D como 3 números, la rotación alrededor de los ejes X, Y y Z (tomvds, Último acceso 2022). Existen diversas formas de obtener los ángulos de Euler, en esta sección serán estudiados dos formas diferentes comparando su resultado con la orientación que ya previamente ha calculado la placa de desarrollo Nicla Sense ME.

3.2.1. Ángulos de Euler a partir de quaternion

Esta primera forma de determinar la orientación del sensor es la que se propone en el código del dashboard proporcionado por el fabricante de Nicla Sense ME. El fragmento de código está escrito en Javascript y es el siguiente:

Listing 3.1: Código conversión quaternion a Euler

```
var rotation = new THREE.Euler().setFromQuaternion(quaternion,'XYZ');

document.getElementById('xQuaternion').innerHTML =
    "x: " + Math.round(rotation.x *180 / Math.PI + 180);
document.getElementById('yQuaternion').innerHTML =
    "y: " + Math.round(rotation.y *180 / Math.PI + 180);
document.getElementById('zQuaternion').innerHTML =
    "z: " + Math.round(rotation.z *180 / Math.PI + 180);
```

Este fragmento de código comienza con la división de los valores obtenidos por el sensor quaternion a una tupla de tres elementos la cual contiene los valores de los ejes x, Y y Z con la conversión ya realizada a los ángulos de Euler. Esta conversión se realiza mediante la función "Euler().setFromQuaternion()" la cual recibe como parámetro el objeto quaternion el cual contiene los valores de los ejes X, Y, Z y W. La rotación obtenida viene dada en radianes, por eso a continuación se multiplica por 180 y se divide por el número π. Finalmente, los datos del sensor están invertidos, por los que le suma 180 para corregir esta inversión.

Los datos mostrados por estos cálculos se comportan de manera fiel a la realidad, indicando con precisión los ángulos que el propio sensor toma. Estos datos obtenidos fueron comparados con los datos de la orientación proporcionados por la placa Nicla mientras se realizaba un fragmento del ejercicio de rehabilitación que consiste en levantar el brazo hacia el techo partiendo de una posición inicial de reposo.

Como se puede observar en la Figura 3.7 los datos generados por la orientación de la placa de desarrollo (representado en color azul) y los ángulos de Euler calculados a partir del quaternion (representado en color rojo) no poseen diferencias significativas entre ellos. Se muestran pequeñas variaciones entre ambas mediciones, pero no llegan a ser lo suficiente significativas para elegir uno por encima de otro. Es por eso que, tras este experimento se sigue manteniendo la consistencia del sensor de orientación dejándolo como el candidato principal para su observación al monitorizar debido a su similitud con los valores calculados de esta orientación, tomando la decisión de elegir la orientación debido a que es una de las herramientas que ofrece la placa de desarrollo Nicla Sense ME utilizada en este trabajo. De esta manera se utilizarían la mayor cantidad de recursos que ofrece la placa para el desarrollo del trabajo.

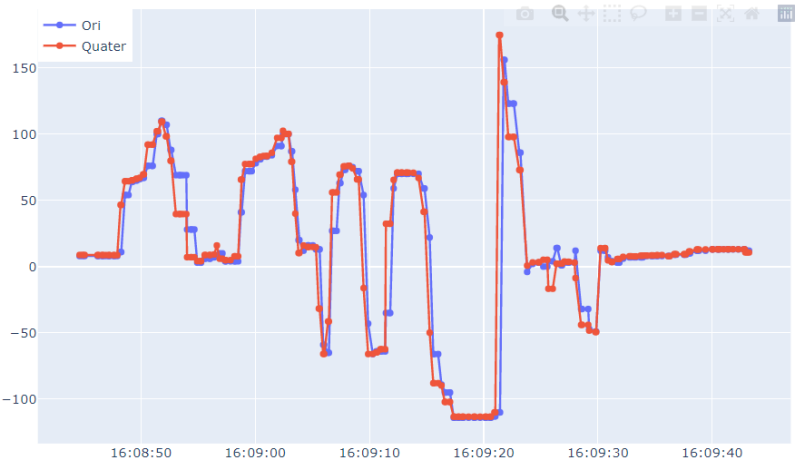

Figura 3.7: Orientación y ángulos de Euler generados a partir del quaternion.

3.2.2. Ángulos de Euler calculados con acelerómetro y giróscopo

La solución propuesta para el cálculo de la orientación del dispositivo en base a las medidas del acelerómetro y giróscopo junto con el uso de varios filtros complementarios fue extraída de un blog llamado Technology Tutorials (McWhorter, 2019).

3.2.2.1. Cálculo de los ángulos con acelerómetro

Estos ángulos calculados a partir del acelerómetro y el giroscopio van a recibir el nombre de Theta (θ) y Phi (ϕ), los cuales representan el ángulo de inclinación del eje X y el eje Y respectívamente.

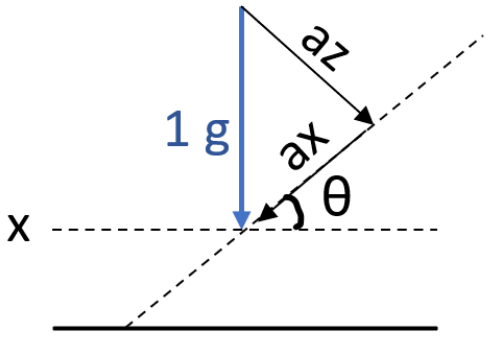

Figura 3.8: Representación del ángulo Theta(θ).

Para entender la naturaleza del ángulo θ podemos observar la Figura 3.8. En ella podemos apreciar como existe un vector de aceleración que es perpendicular al suelo el cual es la gravedad. Ese vector va a ser siempre constante sin importar la inclinación del sensor. El dispositivo estaría representado como el eje X. Al inclinarlo sobre ese mismo eje se forma el ángulo θ, de manera que la gravedad ya no afecta de forma perpendicular al dispositivo. De esta manera, el vector de la gravedad queda dividido en otros dos vectores

que representan la aceleración que sufre el dispositivo tanto en el eje X como en el eje Z. El vector que representa la aceleración del eje Z afecta al dispositivo perpendicularmente, formando un ángulo de 90º con el dispositivo. Con trigonometría básica es posible realizar el cálculo de ese ángulo, empleando la siguiente fórmula para hallarlo:

$$\tan(\theta) = \frac{ax}{az} \tag{3.1}$$

Despejando θ la ecuación quedaría de la siguiente forma:

$$\theta = \tan^{-1}[\frac{ax}{az}] \tag{3.2}$$

Recordemos que los valores del acelerómetro obtenidos del sensor ya vienen divididos en la aceleración de los diferentes ejes del espacio X, Y y Z, por lo que aplicando los valores de las aceleraciones del eje X y el eje Z ya puede ser hallado en ángulo θ.

Los valores de aceleración son divididos por el valor de la fuerza de la gravedad $(9,8m/s^2)$ de modo que los valores de aceleración quedan normalizados para el valor de g, dando como resultado en unidades de g en vez de unidades en metros/segundo. Adicionalmente es necesario convertir el resultado obtenido de la operación del arco tangente de radianes a grados, para ello se divide el resultado anterior por $2/\pi * 360$.

Siguiendo la misma explicación dada en la Figura 3.8 para el cálculo del ángulo θ, es posible el cálculo del ángulo ϕ, pero aplicada al eje Y. De esta forma los vectores de aceleración en los que se dividiría el vector de la gravedad serían la aceleración en el eje Y y la aceleración en el eje Z.

El código obtenido para el cálculo del ángulo θ y el ángulo ϕ sería el siguiente:

```
theta =
 atan2(accelerometer.x()/9.8,accelerometer.z()/9.8)/2/3.1416 * 360;
phi =
 atan2(accelerometer.y()/9.8,accelerometer.z()/9.8)/2/3.1416 * 360;
```

Esta implementación funciona correctamente realizando una medida precisa de los ángulos de inclinación de los ejes X e Y, pero al tratarse de cálculos extraídos únicamente del sensor acelerómetro, va a generar datos siempre que se mueva el sensor, sin necesidad de estar inclinado o no. Es decir, si el sensor está completamente estable en una mesa de forma perpendicular a ella los valores que recibe son de 0 grados para ambos ejes. Pero si este sensor es agitado en cualquier dirección sin inclinarlo, siempre en contacto directo con la mesa va a recibir esa información como cambios en los grados θ y ϕ. De momento se trata de una forma poco sofisticada de calcular la inclinación del sensor. Por ello es necesario agregar un **filtro complementario** para evitar este tipo de vibraciones que crean datos erróneos.

3.2.2.2. Filtro de baja frecuencia

El filtro a implementar consiste en un **filtro de baja frecuencia (lowpass)**, que se encarga de dejar pasar solamente las frecuencias bajas, eliminando las frecuencias mucho mayores. Al realizar una inclinación sobre el sensor se produce una frecuencia baja en las mediciones de la inclinación, mientras que al agitar el sensor se obtienen frecuencias mucho mayores.

Para obtener este filtro los valores nuevos de θ y ϕ serán calculados a partir del valor anterior de estos ángulos de la siguiente manera:

$$\theta_{fNew} = \theta_{fOld} + \theta \tag{3.3}$$

Siendo θ_{fNew} los valores nuevos filtrados, θ_{fOld} los valores filtrados de la anterior actualización y θ los valores de las mediciones actuales. Los valores antiguos de estos ángulos van a tener más peso que los valores nuevos de modo que se garantiza una reducción de los valores nuevos cuya diferencia con el dato anteriormente medido sea muy grande. Tras darle más peso a los valores antiguos la fórmula quedaría de la siguiente forma:

$$\theta_{fNew} = \theta_{fOld} * 0{,}9 + \theta * 0{,}1 \tag{3.4}$$

Los valores de θ filtrados que son antiguos tienen un peso de 90 %, mientras que los valores que acaban de ser recogidos por el sensor tienen un peso del 10 %. Al estar recibiendo datos del sensor cada 0.1 segundos aproximadamente los valores de θ se actualizan de forma casi inmediata respecto a los movimientos de una persona al tratar de inclinar el sensor. De esta manera quedan filtrados estos datos generados que representan unos giros agresivos del sensor que podemos considerar erróneos. Los ángulos de ϕ también pueden ser hallados siguiendo el mismo razonamiento.

Tras esta actualización los resultados obtenidos son mejores, ya que los datos que se producen por vibraciones indeseadas quedan suprimidos, pero sigue sin ser adecuado para las mediciones. Esto se debe a que, a pesar de tener unos datos más limpios, la respuesta producida es significativamente más lenta. Por ejemplo, si el sensor parte de una posición inicial en reposo y su inclinación es modificada alcanzando un ángulo de noventa grados, este puede tardar uno o varios segundos en alcanzar el número de forma precisa. Los valores de peso asignados a los valores antiguos filtrados y los nuevos pueden ser modificados aumentando el peso de los valores nuevos en caso de preferir valores que respondan de una forma más rápida pero también más reactivos al ruido generado.

3.2.2.3. Cálculo de ángulos con giróscopo

Alternatívamente a los cálculos realizados anteriormente, también es posible el procesamiento de los datos obtenidos a partir de un giróscopo que recoge velocidades angulares para convertirlas en ángulos. Para el cálculo de los ángulos a partir de un giróscopo es necesaria esta formula:

$$\theta = \theta + W_y * dt \tag{3.5}$$

$$\phi = \phi + W_x * dt \tag{3.6}$$

Siendo el ángulo de rotación obtenido igual al anterior valor de ese mismo ángulo más la velocidad angular recogida (W) multiplicado por la unidad de tiempo. La velocidad rotacional se mide en ángulos o radianes partidos por unidad de tiempo, por lo que en la ecuación se multiplica por unidad de tiempo, de manera que el resultado serán los ángulos directamente. La velocidad rotacional (W) es la obtenida directamente del giróscopo, y el valor del ángulo inicial es igual al 0. Además la unidad de tiempo es el tiempo que se tarda en realizar de nuevo los cálculos, por lo que ya estarían todos los datos necesarios para hallar la orientación del sensor.

Para hallar el valor de dt en el código, es necesario ejecutar la función $millis()$ que devuelve el número de milisegundos transcurridos desde que la placa Arduino comenzó a ejecutar el programa actual (Arduino, 2022c). Al comienzo de cada vuelta del bucle se inicializa una variable que contiene los milisegundos del momento anterior a realizar los cálculos. El valor de dt se obtiene tras realizar una resta de los milisegundos actuales con los milisegundos obtenidos antes de los cálculos. Finalmente ese dato se divide entre 1000 para obtener el resultado en segundos. De esta manera dt contiene el tiempo que se tarda entre una medición y la siguiente.

El código obtenido para el cálculo del ángulo θ y el ángulo ϕ sería el siguiente:

```
dt = (millis()-millosAnterior) / 1000;
millisAnterior = millis();

thetaGyro = thetaGyro + gyroscope.y()*dt;
phiGyro = phiGyro + gyroscope.x()*dt;
```

Esta nueva implementación utilizando solamente los valores del giróscopo ofrece valores correctos para indicar los ángulos de los ejes X e Y de inclinación del dispositivo. Además, al no utilizar el acelerómetro solamente realiza cálculos utilizando velocidad rotacional, por lo que si se agita el dispositivo sin realizar ninguna inclinación no va a verse afectado de ninguna manera. A pesar de ello, tiene un error constante que al realizar varias inclinaciones del sensor, nunca vuelve a la posición inicial de 0, manteniendo valores cercanos a 0 pero nunca volviendo a la posición original. Ese error queda arrastrado a futuras inclinaciones del sensor, pues siempre va a mostrar el ángulo calculado desde la posición inicial.

3.2.2.4. Inclinación del dispositivo calculada con acelerómetro y giróscopo

La implementación de los cálculos de la orientación del dispositivo a partir de los valores del acelerómetro y del giróscopo no es del todo acertada, pues es cierto que ambas implementaciones funcionan mostrando ángulos correctos, pero las implementaciones tienen varios fallos que impiden su uso de forma continuada. La implementación del acelerómetro es susceptible al ruido, aunque con el filtro de baja frecuencia se arregla ese problema, se añade el problema de que requiere un tiempo para calcular el ángulo de forma precisa. Mientras que la implementación del giróscopo ofrece un tiempo de respuesta muy rápido, ignora el ruido percibido, pero tiene un error de desplazamiento de los datos, calculando datos erróneos tras varias inclinaciones del sensor, donde no se puede confiar en los datos a largo plazo.

Es por eso que se puede crear un filtro complementario donde, tanto el acelerómetro como el giróscopo calculan los ángulos de orientación del dispositivo, recogiendo las mejores partes de cada implementación. La forma de estabilizar estas medidas es coger la velocidad de respuesta del giróscopo, mientras que del acelerómetro son seleccionadas las medidas a largo plazo. Finalmente la ecuación para poder calcular eficientemente la inclinación del sensor con ambos sensores es la siguiente:

$$\theta = [\theta + W_y * dt] * 0{,}95 + [\theta_a] * 0{,}05 \tag{3.7}$$

$$\phi = [\phi + W_x * dt] * 0{,}95 + [\phi_a] * 0{,}05 \tag{3.8}$$

Los valores del giróscopo poseen un mayor peso, ya que si se realiza un movimiento más brusco va a tener más veracidad que los datos del acelerómetro que tardaban un tiempo en estabilizarse. De esa misma manera, si el valor de la velocidad rotacional (W) es 0, es decir, el sensor no ha sido inclinado, el término correspondiente del giróscopo tiende a 0, por lo que predomina en el tiempo el término correspondiente con el acelerómetro. En el corto plazo los datos con mayor fiabilidad serán los del giróscopo mientras que en largo plazo serán los del acelerómetro. De esta forma queda creado un filtro de alta frecuencia en los valores del giróscopo y un filtro de baja frecuencia en los valores del acelerómetro.

El resultado obtenido es finalmente el acertado, pues se calcula de manera correcta el ángulo de inclinación del dispositivo, ignorando el ruido percibido por las vibraciones y manteniendo los valores a lo largo del tiempo.

3.2.2.5. Comparación con la orientación proporcionada por Nicla

Como podemos observar en la Figrura 3.9 se muestran los diferentes valores de inclinación del sensor para un mismo ejercicio que consiste en el levantamiento del brazo desde una posición inicial de aproximadamente 0 grados y una posición final de unos 90-100 grados. Los resultados están representados con los colores azul y naranja para los valores de inclinación que han sido calculados con el acelerómetro y el giróscopo (ejes X e Y respectivamente), mientras que los colores gris y amarillo representan los valores obtenidos de la orientación de la placa de desarrollo Nicla Sense ME (ejes X e Y respectivamente).

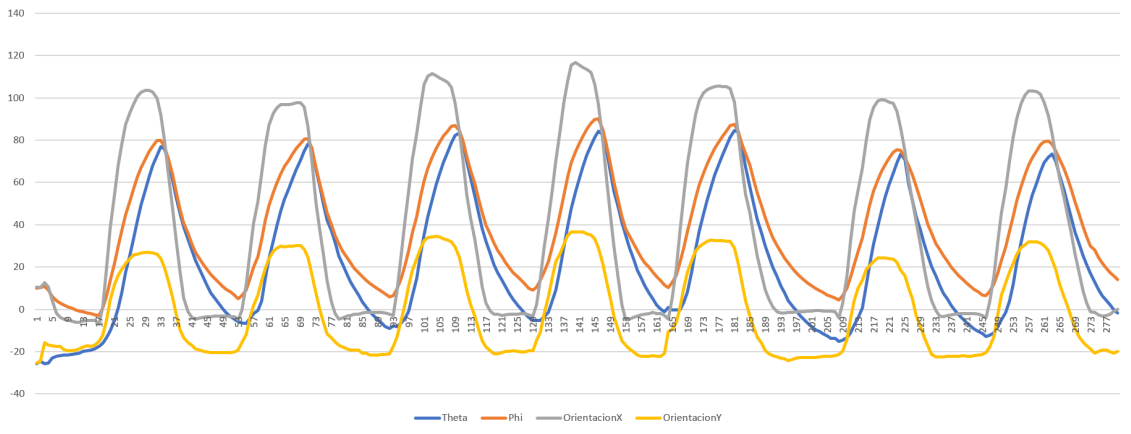

Figura 3.9: Comparación entre los ángulos calculados con los ánguos proporcionados por Nicla.

De los resultados obtenidos para el ejercicio realizado podemos deducir como los valores que proporciona Nicla respecto a la orientación del sensor son más precisos que los calculados anteriormente. Esto se debe a que los valores calculados por el acelerómetro y el giróscopo requiere de un pequeño tiempo para llegar al ángulo apuntado, mientras que la orientación ya calculada por Nicla es precisa y rápida. Además los datos proporcionados por Nicla también ignora los datos percibidos como ruido en vibraciones del sensor. Además los datos del ángulo ϕ al realizar este tipo de movimiento, donde no se mueve el eje X de forma perfecta dejando al eje Y sin moverse, se comporta de forma extraña aumentando su valor a casi el mismo valor que el ángulo θ calculados.

Tras esta experimentación las conclusiones obtenidas son que los datos que van a ser utilizados debido a su fiabilidad y capacidad de respuesta serán los proporcionados por la placa de desarrollo Nicla. No obstante, en caso de no tener una placa que proporcione estas medidas, los cálculos obtenidos actualmente podrían ser válidos para el cálculo de la orientación del dispositivo. Esto se debe a que serán ejercicios que no se ejecutarán a una velocidad demasiado elevada, pues son pacientes tratando de rehabilitarse y las medidas obtenidas no requieren de una precisión milimétrica para llevar a cabo la monitorización de las repeticiones del ejercicio.

3.3. Medidas a transmitir por BLE

Tras los diversos experimentos realizados en las secciones anteriores la **orientación** muestra ser el dato más estable para la monitorización de los ejercicios propuestos. Con ella se podría saber de manera precisa los movimientos de los brazos e incluso piernas para poder deducir y analizar si los movimientos realizados son los correctos.

3.4. Aplicación desarrollada para la monitorización de los ejercicios

Tras realizar las investigaciones dedicadas a elegir las medidas a utilizar para la correcta monitorización de los ejercicios de rehabilitación es necesario aplicar todos estos conceptos en una aplicación que, mediante una interfaz cualquier usuario pueda monitorizar esos ejercicios sin ningún tipo de conocimiento previo. La creación de esta aplicación fue inspirada en el dashboard proporcionado por Nicla (referencia), dónde fue elegido Python como lenguaje de programación debido a la gran cantidad de librerías que posee para la comunicación Bluetooth y el tratamiento y visualización de datos.

3.4.1. Plotly Dash

Dash es una librería de Python especializada en la representación gráfica de los datos (Dash, 2022). Asimismo proporciona las herramientas para la creación de un dashboard completamente interactivo y con interfaces de usuario desde cero. Esta librería de código abierto está escrita sobre Plotly.js y React.js y posee licencia de tipo MIT. Dash recoge todos los protocolos y tecnologías necesarias para la creación de una aplicación web enfocada en la visualización de datos. Al tratarse de una aplicación web es posible acceder a ella desde cualquier ordenador o dispositivo móvil sin descargar nada, ya que puede ejecutarse la aplicación desde una máquina virtual y compartir la URL.

Uno de los puntos fuertes de Dash es que el código utilizado para programar las aplicaciones es declarativo y reactivo, de manera que permite a los diferentes elementos ser interactivos entre sí (referencia). Además, al contener elementos de HTML para el desarrollo de la estructura también permite la personalización completa de todos esos elementos con CSS (plotly, 2017).

Dash lanza sus aplicaciones en servidores utilizando Flask (Flask, Último acceso 2022), el cual es un framework de Python especializado en la creación de aplicaciones web empleando el menor número de líneas de código posibles y las comunicaciones las establece mediante el protocolo HTTP enviando paquetes en formato JSON.

Los elementos interactivos como selectores de fecha, cuadros de opciones, cuadros para introducir texto son sacados de React, otro framework de Python orientado a la creación de interfaces a utilizar en aplicaciones web (Makai, 2022).

Dash permite también la programación de la aplicación web en código HTML gracias a una clase de la librería llamada $dash_html_component$ la cual fue utilizada para el desarrollo de la totalidad de la aplicación. De esta manera se pudieron añadir títulos, botones y selectores de opciones a la aplicación.

3.4.1.1. Funciones dentro de Dash

Las funciones dentro de Dash funcionan en base a **callbacks** que únicamente pueden ser invocadas si se dispara el evento (o eventos) activador asociado a esa función. Los callbacks de las funciones pueden recibir los siguientes elementos:

- **Inputs:** Elemento que va a activar la función de callback. Normalmente suelen ser botones o incluso intervalos de tiempo que se repiten periódicamente.
- **Outputs:** De la misma manera los callbacks tienen asociados unos **outputs** Elementos que van a verse modificados por estas funciones. Estos outputs pueden ser cualquier tipo de componente, desde divs en HTML para añadir contenido HTML o las propias gráficas que puedan verse modificadas o filtrados sus valores.
- **State:** Adicionalmente puede introducir el elemento state en el callback para recibir datos de elementos como ocurría con el input pero sin necesidad de activar la función de callback en caso de interactuar con estos elementos.

Los componentes asignados a estos inputs o outputs son diferenciados utilizando los ID de HTML con los que fueron definidos al crear la aplicación web. Además hay que proporcionarle a la función de callback la propiedad del elemento que va a ser modificada o que se va a tener en cuenta para activar la función de callback. Una función de callback se vería de esta manera:

```
@app.callback(
    Output(component_id='body-div', component_property='children'),
    Input(component_id='button', component_property='n_clicks'),
    State(component_id='selector',component_property='value')
)
def function(n_clicks,valor_selector):
    ...
    return "Output"
```

En el código podemos apreciar como la función llamada *function* tiene un callback asignado con un elemento de output que consiste en un div en HTML, cuya propiedad asignada al callback es *children*. Esto significa que al devolver cualquier tipo de valor, este va a ir asignado directamente a la propiedad children de ese componente div. En este caso la función devuelve la palabra "Output" de manera que en el div cuya id de componente es 'body-div'va a actualizar su contenido. Este va a contener un elemento HTML de tipo párrafo cuyo contenido es la palabra devuelta por la función. El input de la función de callback en este caso es un botón cuya id es 'button'y la propiedad que va a recibir la función es ñ_clicksél cual indica el número de veces que ese botón ha sido presionado. Ese valor de clicks lo recibe la función directamente como parámetro y puede ser utilizado por ella. Además incluye un componente de tipo state que está asociado al elemento de id 'selectorél cual está asignado a un desplegable con varias opciones. De ese selector el callback recible el valor de ese selector en el momento que sea presionado, ya que la propiedad que tiene asignado ese state es 'value'.

Un callback puede recibir varios outputs, varios inputos o varios state siempre y cuando sea declarado en la función de callback como una lista de objetos. En caso de ser inputs o state es necesario agregar un parámetro en la función para cada uno de esos elementos, en el orden que fueron declarados. Y en caso de existir varios outputs o datos de salida es necesario devolverlo de la función como una lista con las componentes de los outputs en el orden correspondiente.

3.4.1.2. Visualización de los datos

Para la visualización de los datos en forma de gráfica se utiliza el módulo de Dash *dcc* (Dash Core Component) el cual incluye el elemento *dcc.Graph*.(referencia) Este componente permite visualizar los datos de varias formas, entre las que se encuentran los gráficos

de líneas, gráficos de barras, gráficos de dispersión, etc. Los atributos de estos gráficos pueden ser modificados dinámicamente con las funciones de callback explicadas anteriormente y permite interactividad completa con los gráficos. Uno de los ejemplos de esta interactividad es poder filtrar los valores en otras gráficas en función de seleccionar unos puntos u otros. Como se muestra en la Figura 3.10 se está seleccionando una serie de puntos con la herramienta de selección, y estos se ven modificados en las otras gráficas.

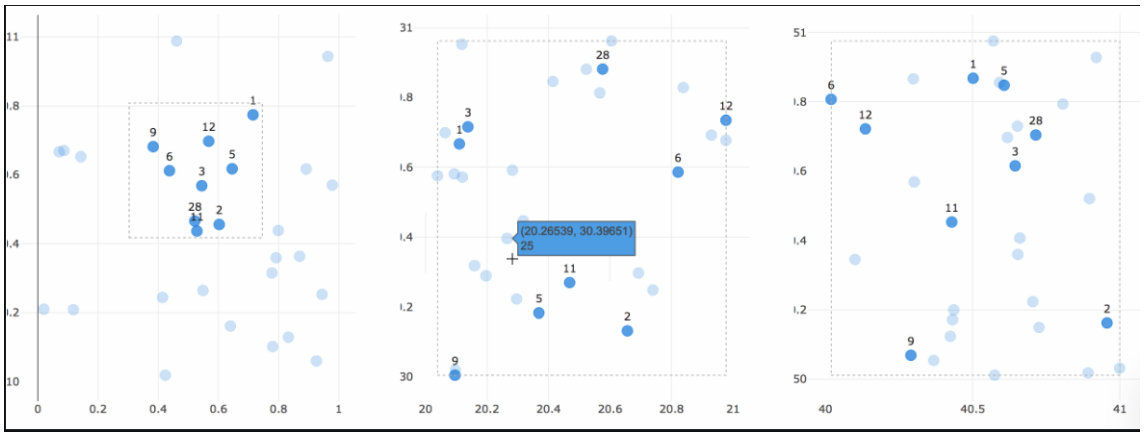

Figura 3.10: Filtrado de datos dinámico en gráfico de Dash.

Estos gráficos interactivos además permiten acciones como mostrar los valores cuando se pasa el ratón por encima de un elemento (hover), al hacer clic en un punto determinado de la gráfica, al seleccionar varios elementos con el lazo o la herramienta de selección, o hacer zoom y redimensionar los datos, entre otras.

Los gráficos que proporciona esta librería (más concretamente los gráficos de líneas) son perfectos para la representación de las medidas recibidas por los sensores del Nicla, por lo que serán usados durante el desarrollo de esta aplicación.

3.4.2. Primera implementación usando Dash

Las primeras implementaciones fueron una toma de contacto con estas herramientas, de modo que programas con funcionalidades reducidas fueron creados al principio. La prueba de contacto fue tratar de mantener el programa recibiendo datos continuamente del sensor mientras se mostraban esos datos por la aplicación generada en Dash. Los resultados obtenidos fueron los que se muestran en la Figura 3.11 dónde son captados los datos que componen la orientación del sensor (pitch, roll y heading).

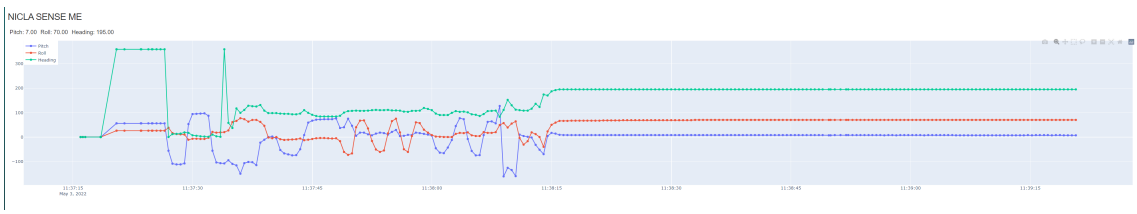

Figura 3.11: Primera visualización de los datos en Dash.

Para poder visualizar los datos del sensor y que se actualice la información en tiempo real son necesarios los siguientes elementos:

- **Datos:** Estos se deben estar actualizando de manera constante via BLE. Este proceso

se realiza de forma paralela a la ejecución de la aplicación web con un thread o hilo de ejecución.

- **dcc.Graph:** Componente para poder visualizar los datos. En este caso está formado por un gráfico de 3 líneas.

- **dcc.Interval:** Componente que activa un callback de forma periódica. Este temporizador permite actualizaciones de los diferentes componentes indicando el tiempo de refresco en milisegundos.

Esta gráfica se actualiza cada intervalo regular de 0.2 segundos gracias al componente *dcc.Interval*. Fueron probados intervalos de tiempo más bajos, pero el sistema se desestabilizaba y no actualizaba los datos de forma correcta, o no permitía la interacción con los elementos del dashboard. Por cada activación del componente de intervalos se actualiza el componente *div* actualizando la gráfica con los nuevos valores.

Los valores han de ser almacenados en una lista o, en este caso una estructura que contiene una lista por cada eje del sensor. Es decir, una lista para el pitch, otra para el roll y otra para el heading. Estas listas almacenan todas las medidas recogidas por el sensor desde que se ha iniciado la medición. Si aparece un dato nuevo del sensor orientación, este valor se añade al final de la lista. Estas listas son introducidas en la gráfica mediante el eje X, mientras que el intervalo de tiempo dónde fueron recogidas ocupan el eje Y.

3.4.3. Desarrollo de la aplicación final

Esta aplicación es un prototipo de una posible futura aplicación más completa y que cumpla con los objetivos de una clínica de rehabilitación o un hospital. La aplicación en esta fase está pensada para el tratamiento de los pacientes, pero con poca seguridad, pues no existe un inicio de sesión y los registros de los ejercicios son únicos, por lo que todo el mundo tendría acceso a estos datos dentro de la aplicación. Esos requisitos no cumplen con el objetivo de este Trabajo de Fin de Máster, por lo que queda como trabajo futuro a realizar. La explicación más en detalle será dada en el capítulo de Conclusiones y Trabajo Futuro.

Dado que la mayoría de los ejercicios de rehabilitación consisten en la repetición de movimientos cíclicos simples el sensor elegido de todos los disponibles en la placa de desarrollo Nicla Sense ME es la orientación del sensor. Esto se debe a que si el sensor es colocado en el lugar indicado (normalmente en las muñecas o el tobillo) puede ser obtenido de una manera precisa la inclinación de las extremidades. De esta manera se puede saber si se ha realizado el movimiento completo o se ha quedado con el movimiento a medias. De todos los ejes disponibles en la orientación medida, será utilizado únicamente el pitch u orientación en el eje X. Esto se debe a que prácticamente todos estos ejercicios (salvo dos excepciones) únicamente se realiza movimiento sobre este eje, descartando así información irrelevante.

Esta aplicación cuenta con dos pantallas o pestañas principales: Pantalla de monitorización de movimientos de rehabilitación y pantalla de historial de ejercicios.

3.4.3.1. Pantalla de sensorización de movimientos de rehabilitación

Pantalla principal de la aplicación. En ella se encuentran todos los elementos de configuración del ejercicio a realizar, así como una serie de indicadores visuales para mostrar al usuario si está realizando correctamente el ejercicio.

Esta a su vez se divide en dos partes: Un panel de control y un panel de visualización del ejercicio.

La primera parte consiste en un **panel de control** que contiene la información del ejercicio que va a ser realizado. Todos los parámetros pueden ser configurados acorde a las necesidades del paciente, así como el ejercicio que va a realizar este paciente.

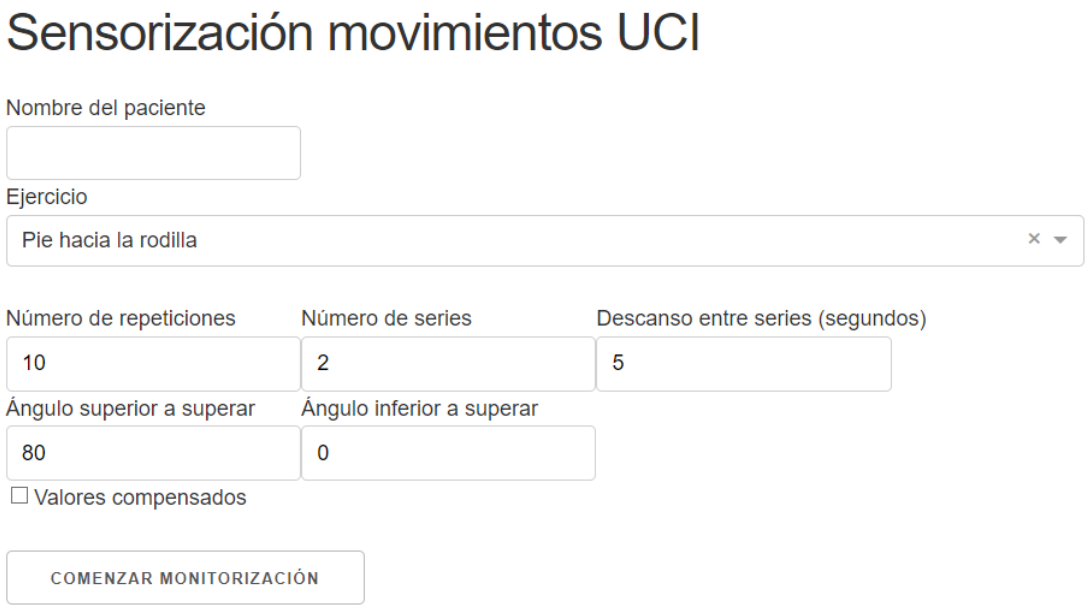

Figura 3.12: Panel de control de la aplicación.

La Figura 3.12 muestra como es el panel de control. Los elementos que componen este panel de control son los siguientes:

- **Nombre del paciente:** Campo de texto donde se introduce el nombre del paciente. Este dato será guardado en la base de datos al terminar el ejercicio y así será posible la distinción entre pacientes.
- **Ejercicio:** Menú desplegable donde puede ser seleccionado el ejercicio a realizar. Las opciones disponibles son: Codo al hombro, brazo al techo, pie hacia la rodilla y levantamiento de pierna. Cada uno de ellos contiene dos versiones, una para la extremidad izquierda o derecha, de modo que se puede llevar también un registro de que extremidades se han ejercitado más o menos.
- **Número de repeticiones:** Campo numérico que indica el número de veces que debe repetirse el mismo movimiento para ser completado.
- **Número de series:** Campo numérico que indica el número de series que debe realizar el paciente para considerar completado el ejercicio. Cada serie implica tener que realizar el mismo número de repeticiones varias veces con un tiempo de descanso entre ellas.
- **Descanso entre series:** Campo numérico que indica el tiempo (en segundos) que van a ser necesitados para comenzar con la siguiente serie.

- **Ángulo a superior superar:** Campo numérico que indica el ángulo de referencia que el paciente debe superar para completar el movimiento superior de la extremidad. Por ejemplo, si el ejercicio indica mover el brazo hacia el techo, para que el movimiento sea correcto el ángulo de inclinación del brazo debe ser mayor a 80 grados, pues apuntar al techo serían aproximadamente 90 grados.
- **Ángulo a inferior superar:** Campo numérico que indica el ángulo de referencia que el paciente debe superar para completar el movimiento inferior de la extremidad. Por ejemplo, si el ejercicio indica mover el brazo hacia el techo, para que el movimiento sea correcto el ángulo de inclinación del brazo debe ser menor a 20 grados, pues volver el brazo a su posición inicial (tumbado) serían de aproximadamente 0 grados.
- **Valores compensados:** Casilla marcable que, en caso de ser marcada, normaliza los datos del sensor partiendo del ángulo 0 independientemente que inclinación tenga el sensor al iniciar el ejercicio. Esto es útil para la monitorización de ciertos ejercicios de menor amplitud.
- **Comenzar monitorización:** Botón que sirve para comenzar la monitorización del ejercicio. Al presionarlo comienza una cuenta atrás y aparece el apartado visual para la monitorización del ejercicio.
- **Intervalo:** Esta componente no es visible, pues se trata de una componente que indica el intervalo de tiempo que requiere para activar ciertos callbacks. Este intervalo inicialmente está desactivado y es el encargado de actualizar todos los elementos de la parte de la pantalla principal relacionada con la interfaz visual. El intervalo de tiempo que requiere cada activación de esta componente es de 0.2 segundos. Fueron probados diferentes valores y este era el más estable, pues con menor intervalo de tiempo el sistema era inestable y se quedaba bloqueado el programa y con mayores intervalos de tiempo no hay suficientes actualizaciones en los datos de la placa Nicla recibidos.

Respecto al funcionamiento interno de este panel de control hay varias funciones que permiten el correcto funcionamiento del sistema:

- La primera función que cabe destacar es la encargada de modificar automáticamente los valores del panel en función del ejercicio. Esta función de callback tiene como parámetro de entrada el valor del selector de ejercicio explicado anteriormente. Cuando se modifica el valor de este desplegable, esta función se activa modificando los valores del ángulo mínimo a superar y el ángulo máximo a superar que tiene definido este callback como parámetros de salida. Los valores establecidos están puestos por defecto según la experimentación de varios ejercicios y comprobar cuáles eran los más adecuados para cada ejercicio. Tras cambiar el ejercicio estos valores pueden modificarse manualmente en los campos indicados del panel de control.
- Tras presionar el botón una vez se activa la función de callback que inicia el ejercicio. Esta función recibe como parámetro el número de veces que se ha presionado el botón que ha activado este callback. Si ha sido presionado una vez se activa el intervalo por lo que comenzaría la monitorización del ejercicio. El valor del texto del botón se modifica adquiriendo el valor de 'PARAR MONITORIZACIÓN'y aparece la componente de la pantalla que es una interfaz visual para el usuario.

  Al pulsar el botón de nuevo se para la monitorización y todo vuelve a su estado inicial. Además, los datos obtenidos así como la información del ejercicio dada en el

panel de control se insertan en la base de datos. Finalmente las variables globales que han sido usadas, como las listas de los valores de la orientación para mostrar en la gráfica, eliminan todo su contenido para la próxima monitorización.

El segundo componente de esta pantalla principal es el **panel de visualización del ejercicio**. Este panel salvo un par de elementos aparecerá tras accionar el botón para comenzar la monitorización del ejercicio. La Figura 3.13 muestra como es este panel al activar la monitorización del ejercicio.

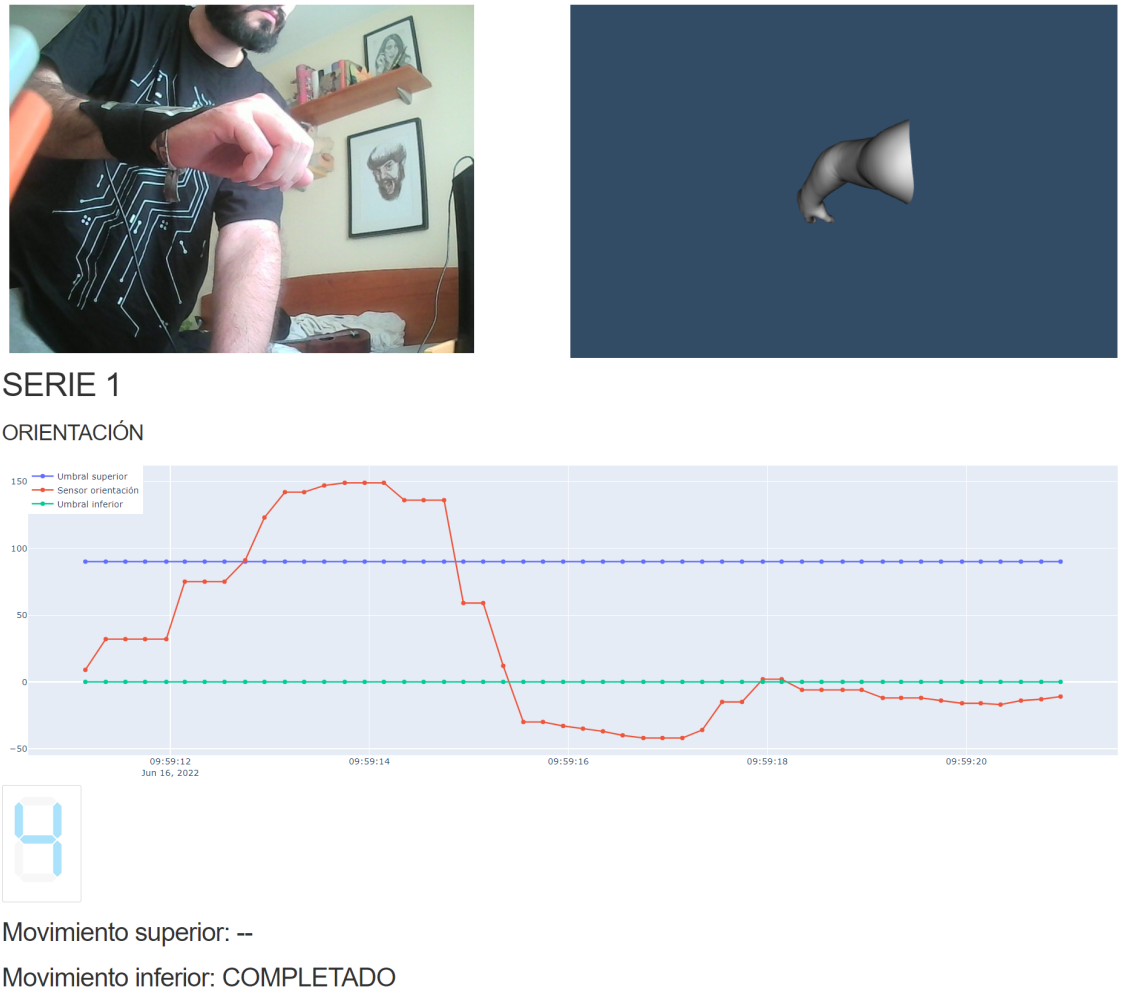

Figura 3.13: Panel de visualización del ejercicio.

Los diferentes elementos que componen este panel son los siguientes:

- **Cámara:** Espacio dedicado para las imágenes captadas por la videocámara.
- **Modelo 3D:** Espacio dedicado para figura 3D de un brazo que se mueve en consonancia con el eje X del sensor, dando feedback al usuario del movimiento realizado.
- **Serie:** Campo de texto donde se indica la serie actual del ejercicio. Al cambiar de serie este valor se actualiza.
- **Gráfica orientación:** Gráfica donde se muestra el movimiento del sensor de una manera visual. En color rojo se puede observar los últimos valores de la orientación

del eje X a lo largo del tiempo de manera que ascienden y descienden con el movimiento del sensor. Además, aparecen dos líneas de colores azul y verde que sirven de limitadores de los marcos superior e inferior que fueron definidos en el panel de control.

- **Contador de repeticiones:** Campo LED que indica visualmente con un número el número de repeticiones que se han realizado en esta serie. Al cambiar de serie este valor se inicializa de nuevo a 0. Para que este valor sea alterado es necesario superar con los valores de orientación del eje X los márgenes superior e inferior respectivamente.

- **Movimiento superior:** Campo de texto que indica si el movimiento del ejercicio que corresponde con subir un brazo o una pierna ha sido completado. En caso de no ser completado se indicará con el valor '–'y una vez se complete se indicará con el valor 'COMPLETADO'. Cuando los movimientos superior e inferior han sido completados volverán a su estado inicial y sumará uno al contador de repeticiones. Para considerar un movimiento como completado es necesario que el sensor supere la barrera superior del ángulo a superar indicada en el panel de control y salir de esa barrera de nuevo entrando en la zona neutral.

- **Movimiento inferior:** Campo de texto que indica si el movimiento del ejercicio que corresponde con bajar un brazo o una pierna ha sido completado. En caso de no ser completado se indicará con el valor '–'y una vez se complete se indicará con el valor 'COMPLETADO'. Cuando los movimientos superior e inferior han sido completados volverán a su estado inicial y sumará uno al contador de repeticiones. Para considerar un movimiento como completado es necesario que el sensor supere la barrera inferior del ángulo a superar indicada en el panel de control y salir de esa barrera de nuevo entrando en la zona neutral.

Respecto al funcionamiento interno de estos componentes hay una función que lleva el funcionamiento completo del programa.

La función de callback principal permite el funcionamiento de toda la sección encargada de la monitorización del sensor, así como la capacidad de hacer aparecer estos elementos en la aplicación en el momento que se inicia este ejercicio. El elemento que sirve de activador para esta función de callback es el intervalo de tiempo. Este intervalo es activado en el momento que el botón de iniciar el ejercicio es presionado y cada 0.2 segundos esta función va a ser activada. Además, esta función recibe varios datos via *State* en el callback, ya que solamente va a activar la función el intervalo. Estos datos son el valor del selector del ejercicio, el valor de los campos numéricos que indican los ángulos máximo y mínimo que el paciente debe alcanzar para superar el ejercicio, el número de repeticiones, si la casilla de compensar valores ha sido marcada, el número de series a realizar, el tiempo de descanso entre series y finalmente el nombre del paciente. Todos esos valores están definidos en la función de callback en forma de lista y la propiedad a recibir de todos estos elementos es 'valueó el valor que contiene ese elemento HTML. Finalmente, como parámetro de salida u *output* es utilizado un div o agrupación de contenido dónde van a ser volcados todos los nuevos componentes.

Tras presionar el botón de comenzar monitorización y se realicen las primeras llamadas a la función, esta mostrará por pantalla una cuenta atrás del 3 al 1 como se muestra en la Figura 3.14, así el paciente tiene tiempo de prepararse. Esta cuenta regresiva en las primeras iteraciones de la función se lleva a cabo gracias al contador del número de veces que se ha activado el intervalo, el cual se pasa como parámetro en la función. Al saber que

los intervalos de tiempo son de 0.2 segundos, cada 5 intervalos es equivalente a 1 segundo. Por esa misma razón, las primeras 5 iteraciones de la función solamente muestran por pantalla devolviendo el valor del componente de dash que genera un LED con el número 3. Asimismo este proceso se repite 2 veces más para el número 2 y el número 1. Durante este proceso también se comprueba si la casilla de compensación está actualizada, y si es el caso se calcula la diferencia del valor del sensor con el número 0 de modo que se obtiene el número encargado de normalizar estos datos.

Figura 3.14: Cuenta progresiva para dar comienzo al ejercicio.

Tras terminar la cuenta atrás los datos del sensor que estaban ubicados en una variable global son recogidos, así como el instante de tiempo actual. Si la casilla que indica que los valores han de ser compensados se realiza una resta del número obtenido en las iteraciones anteriores con el valor del sensor, de modo que los ángulos obtenidos se producen a partir del ángulo del que se partía en el momento de comenzar el ejercicio, contabilizando este ángulo como 0 grados.

Los datos del sensor ya sean con compensación o sin ella son agregados a una lista que a su vez está contenida en una estructura. Esta estructura además contiene los valores de los umbrales máximos y mínimos que debe superar el paciente para la realización del ejercicio (que normalmente serán constantes estos valores, aunque pueden ser modificados mientras se está monitorizando el ejercicio) así como la lista de los instantes de tiempo donde fueron recogidos los valores en cada una de las diferentes iteraciones de la función. Al acumular todos los valores recogidos en listas diferentes permite la representación en la gráfica de líneas de todos estos valores.

Tras la recolección de datos se define la figura del gráfico que va a ser mostrada en la aplicación añadiéndole las medidas de la ventana donde va a aparecer el gráfico, así como la leyenda de ese mismo gráfico. Finalmente se añaden una a una las diferentes líneas que van a aparecer, es decir, una línea para el margen superior, otra línea para los datos del sensor y otra línea para el margen inferior. Cada una de estas líneas son definidas indicando los valores del eje X como la lista con el instante de tiempo donde fueron tomadas las medidas, y el eje Y con cada una de las otras listas.

La actualización del contador de repeticiones se produce contando el posible subidas y bajadas del sensor. Esta solución propuesta consiste en recoger los dos últimos valores de los datos recogidos por el sensor. De este modo que se va a comprobar si el penúltimo registro del valor obtenido no ha superado el umbral y el último valor obtenido si lo supera. En caso de ser un movimiento de subida primero contabiliza la entrada a la zona del umbral superior, indicándolo en una variable global de tipo booleano. Al entrar en la franja superior también tiene que salir de ella para que cuente como completa. Para esto se realiza el mismo procedimiento que para la entrada en esta franja comprobando los dos

últimos valores. Si se ha realizado esta secuencia con éxito se marca como completado el movimiento superior y se actualiza el valor del campo de texto correspondiente. Esta misma fórmula se repite para el umbral inferior.

No existe ningún tipo de orden a seguir para realizar los movimientos planteados, solamente tiene que existir un movimiento superando el umbral superior y otra repetición superando el inferior. De este modo, si se repiten varias veces el movimiento superior sin realizar el movimiento inferior solamente va a ser contada como media repetición.

Cuando ambos valores de movimiento superior e inferior han sido marcados como completados se incrementa en uno el contador de repeticiones del ejercicio en cuestión y se reinician los valores de las variables y de los elementos que indican que se ha completado los movimientos parciales del ejercicio.

Después de eso se rellena un registro que va a estar disponible en una variable global con el nombre del paciente, el ejercicio, las repeticiones realizadas, la serie en la que se encuentra, los márgenes a superar, una lista con los valores del sensor, la fecha y la hora de la realización del ejercicio. Este registro será insertado en la base de datos en el momento de parar o terminar la monitorización del ejercicio.

Finalmente la función comprueba si el número de veces que se ha repetido el ejercicio concuerda con el número de repeticiones establecido. Si el número es igual a las repeticiones realizadas la función se pone en modo ”reposo” donde lo único que va a mostrar es una cuenta hacia delante desde 0 con un elemento LED. El tiempo que va a estar en modo ”reposo” es el establecido en el campo ”Descanso entre series” del panel de control.

Una vez terminado el tiempo de descanso entre repeticiones volverá a contabilizar las repeticiones a del ejercicio volviendo a contar desde 0. Además el campo de texto que indica la serie ha sido incrementado en uno indicando al paciente que se encuentra en otra serie. Este proceso será repetido hasta que el número de series a realizar concuerde con el número de serie actual. Si esto sucede, el sistema mostrará un mensaje indicando al paciente que ha completado con éxito el ejercicio. Para guardar el progreso es necesario presionar el botón de parar monitorización. La Figura muestra la aplicación tras completar un ejercicio.

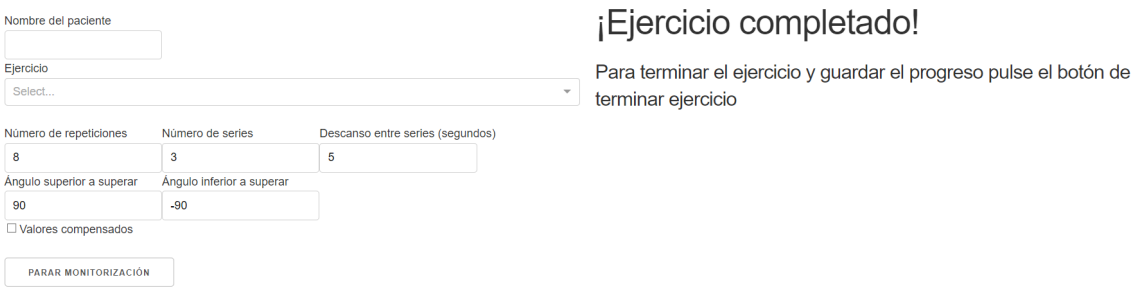

Figura 3.15: Aplicación tras completar el ejercicio.

3.4.3.2. Pantalla de historial de ejercicios:

Esta pantalla secundaria puede ser accedida para revisar el historial de ejercicios que ha realizado el paciente con anterioridad. En ella se puede observar una tabla interactiva, que al seleccionar un ejercicio de la lista se muestra la gráfica de movimiento obtenida durante la ejecución de ese ejercicio. Como se puede apreciar en la Figura 3.16 esta pantalla cuenta con dos componentes principales: una tabla que contiene todos los datos sobre el ejercicio realizado y una gráfica de líneas que muestra los movimientos captados por el sensor en

el momento de la ejecución del ejercicio en cuestión. Al seleccionar una fila presionando sobre cualquiera de sus columnas se modifica el valor del gráfico, de modo de que pueden visualizar los movimientos de todos los ejercicios de los pacientes.

Para la carga de datos desde la base de datos es necesaria una función de callback que va a ser activada en el momento que se cambia de pestaña a la del historial. Esta función recibe como parámetro de entrada el valor del selector de pestañas de la parte superior de la pantalla y como parámetro de salida un elemento de tipo *div* para introducir el código HTML que representa la tabla. Esta función inserta los datos de la base de datos en una tabla (escondiendo el valor de ID de fila ya que es irrelevante) y devolviendo esa tabla a la pestaña asociada.

Para conseguir la interacción de la tabla con el gráfico de líneas es necesaria otra función de callback que va a ser activada cuando se detecta una celda activa, o lo que es lo mismo, presionando sobre una de las celdas. Esta función recibe como parámetros la propia tabla con la propiedad que indica la celda que está activa, y además los datos de toda la tabla. Al tener todos los datos de la tabla y la fila seleccionada solamente hace falta extraer la lista de datos correspondiente a las mediciones del sensor y crear una componente de gráfico de líneas. Este gráfico es devuelto por la función y va a ser mostrado en un elemento de tipo *div* que se encuentra sin contenido inicial en la parte inferior de la tabla.

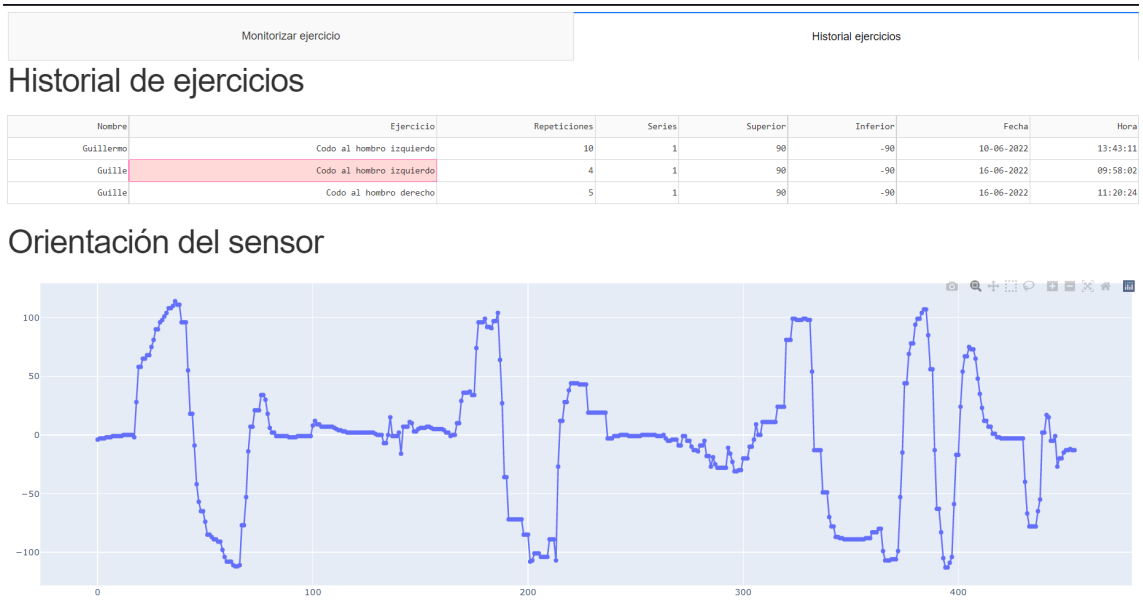

Figura 3.16: Pantalla que contiene la visualización de ejercicios ya realizados.

3.4.3.3. Uso de la cámara y el modelo 3D

Para capturar imágenes de vídeo con la videocámara ha sido utilizado **openCV** (OpenCV, Último acceso 2022). Esta librería de Python de código abierto está especializada en todo lo relacionado con la visión computacional y el aprendizaje automático. Para utilizar esta librería para capturar la cámara es necesario utilizar la clase *VideoCapture*. De esta manera se crea un objeto que contiene el contenido de la cámara. Para que este objeto pueda ser visualizado por la aplicación, los frames del vídeo son recogidos como imágenes con extensión *.jpg*, las cuales se tratan como bytes. Estos datos son devueltos como un generador de Python utilizando *yield* que finalmente será procesado por el propio servidor de la aplicación Dash. Este se reproducirá en un directorio del servidor cuya ruta es */video_feed*.

Para poder mostrarlo en la aplicación, es añadido un elemento HTML de tipo imagen cuya fuente de datos es la ruta del servidor que contiene el vídeo (bcd, 2018).

En cuanto al modelo 3D Dash ofrece la posibilidad de añadir gráficos en 3D que pueden ser rotados e interactuar con ellos. Aprovechando eso es posible cargar un modelo 3D de cualquier figura. En este caso fue seleccionada la figura de un brazo humano para emular los movimientos de los ejercicios. El archivo que contiene el modelo 3D es un archivo con extensión *.obj* y fue descargada un portal gratuito de modelos 3D (turbosquid, 2022). Para interactuar con la orientación del modelo 3D hay que modificar la propiedad del elemento llamada *actor* de forma que se puede acceder directamente a la orientación del objeto. Al añadir los valores de la orientación del sensor al de los ejes no se correspondía al movimiento que tendría que estar realizando realmente, por lo que fueron suprimidas las coordenadas Y y Z de manera que solamente se mueve sobre el eje X.

3.4.3.4. Adicional: primer prototipo de contador de repeticiones

La primera implementación realizada del contador de repeticiones estaba basada en la idea de un proyecto que consistía en una mancuerna o una pesa la cual tenía integrado un acelerómetro (nguyen37, 2017). Ese proyecto sirvió de inspiración para la creación de una forma alternativa de contar los movimientos realizados por el paciente. Esta consiste en utilizar también los datos de la **aceleración lineal**. La aceleración lineal representa la aceleración que ha sufrido el sensor. Este dato serviría para indicar si hay movimiento en el sensor. A diferencia de los datos obtenidos del acelerómetro, la aceleración lineal solamente aumenta su valor cuando existe un movimiento. En cambio los datos obtenidos por el acelerómetro siempre miden la aceleración por parte de la gravedad que reciben las partículas del interior del sensor, de modo que si se encuentra inclinado el sensor, aún no estando en movimiento va a mantener los valores elevados. Esta nueva medida indicará si el sensor está en movimiento o está en reposo. Las medidas de la aceleración lineal tras realizar uno de los ejercicios se muestra en la Figura 3.17. En ella podemos apreciar varios picos de datos que fueron analizados con el fin de obtener un valor mínimo de aceleración para que fuera considerado movimiento o no.

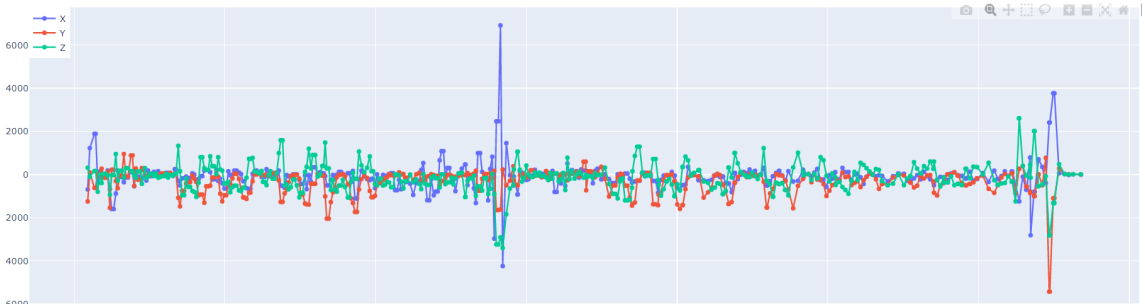

Figura 3.17: Datos de la aceleración lineal tras realizar un ejercicio de rehabilitación.

Los ejercicios propuestos de rehabilitación siguen un patrón básico. Este patrón consisten en que hay movimiento hasta llegar a los extremos superior e inferior dónde el movimiento es mucho más lento. Estos cambios de dirección pueden ser percibidos como si el sensor dejase de estar en movimiento, ya que no superaría el umbral mínimo de aceleración que se haya establecido. Al detectar una pausa en estos ejercicios tras haber estado en movimiento varios instantes de tiempo, podría ser procesado como media repetición.

A esta idea se le sumaba la información del sensor de orientación, detectando patrones de subida y de bajada analizando los valores simultáneos obtenidos, de forma que si había

varios valores seguidos de carácter ascendente se calificaba como una subida y viceversa.

Esta implementación funcionaba, pero era inestable, ya que varios movimientos no los detectaba correctamente. Las pruebas realizadas fueron exitosas, ya que contabilizaba correctamente las repeticiones, salvo alguna repetición de vez en cuando que no la conseguía capturar. Esto se debe al umbral de activación, ya que si estaba establecido con un valor muy alto, hacía falta una velocidad muy alta para realizar los movimientos. Mientras que si se establecía un valor más bajo, en ocasiones lo detectaba como movimiento cuando no lo había o había sido muy leve. Además como se muestra en la Figura 3.17 en ocasiones los valores de la aceleración lineal se disparan dando picos muy elevados que pueden ser detectados como falsos positivos.

Finalmente se dio con una solución mucho más simple y efectiva de contabilizar las repeticiones realizadas que consistía en la colocación de umbrales constantes en la orientación del dispositivo. Al superar esos umbrales es detectado como movimiento superior e inferior y tras realizar ambos movimientos se contabilizaba como una repetición completa.

3.4.4. Bleak

Bleak es la librería de Python utilizada en el desarrollo de la aplicación que se encarga de la comunicación Bluetooth con el dispositivo Nicla. Bleak (Bluetooth Low Energy platform Agnostic Klient) es un software de cliente GATT, capaz de conectarse a dispositivos BLE que actúan como servidores GATT. Está diseñado para proporcionar una API de Python asíncrona y multiplataforma para conectarse y comunicarse, por ejemplo sensores (Blidh, 2022). Tiene una licencia de tipo MIT y su instalación es muy sencilla.

Para conectarse con el dispositivo es necesario conocer su dirección MAC. Para ello con Bleak se puede realizar un escaneo de todos los dispositivos cercanos, filtrando los que incluyan la palabra "Nicla" en el nombre del dispositivo, ya que viene establecido de fábrica en las placas de desarrollo Nicla Sense ME.

Una vez es descubierta la dirección MAC del dispositivo se puede establecer conexión añadiendo la dirección del dispositivo al cliente Bleak y finalmente tratando de establecer conexión. Si no ocurre ningún error la conexión queda establecida y ya pueden ser recibidos los datos.

Para la recepción de datos hace falta realizar una suscripción a las características de los servicios a los que envía los datos vía BLE la placa de desarrollo Nicla. Concretamente para la aplicación los valores que necesita recoger de la placa serían la orientación y la aceleración lineal. Para ello es necesario conocer la UUID de la característica y comenzar las notificaciones sobre esa dirección. Las direcciones utilizadas para la monitorización de la orientación y la aceleración lineal del dispositivo serían las siguientes:

- **Orientación**: 19b10000-9001-537e-4f6c-d104768a1214
- **Aceleración lineal**: 19b10000-6001-537e-4f6c-d104768a1214

Al recibir una notificación de publicación por parte de la placa Nicla, se dispara una función de callback que recibe los datos.

Bleak requiere de su ejecución en un bucle constante, por lo que para que la aplicación funcione de forma continuada y sin bloquearse al recibir datos es necesario el uso de los threads o hilos de ejecución. Los **threads** permiten la ejecución de varias funciones de forma paralela, de manera que se ejecutan en un flujo completamente diferente de ejecución. De esta manera quedan evitados los bloqueos y se garantiza una ejecución fluida de ambos códigos.

3.5. MongoDB

MongoDB es un modelo de bases de datos no relacional y de código libre, donde guarda los datos en estructuras de tipo BSON, permitiendo la creación de bases de datos y la inserción de los datos más fácil y cómodamente. Concretamente fue utilizada la versión "Atlas" de MongoDB el cual es un servicio de bases de datos completamente en la nube. De esta manera el sistema IoT queda descentralizado pudiendo acceder a ella desde cualquier dispositivo siempre se tengan las credenciales de acceso para poder consultar o insertar datos. Posee una versión gratuita y una de pago. La versión gratuita satisfacía las necesidades básicas por lo que no hizo falta la versión de pago, que simplemente añade mayor capacidad para las bases de datos.

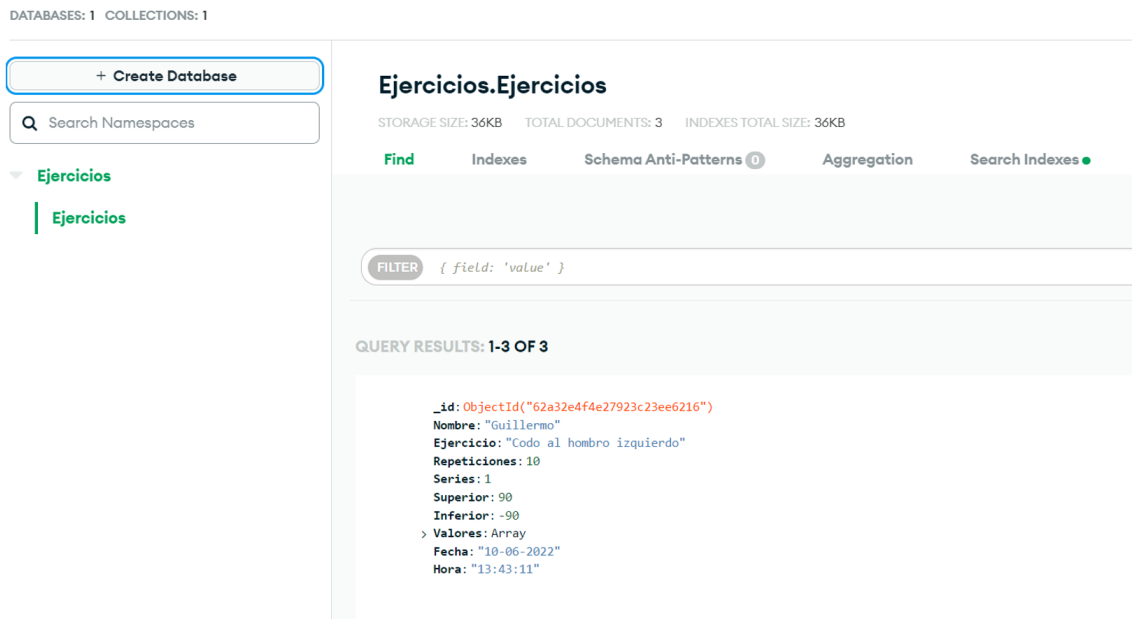

Figura 3.18: Panel de control de MongoDB.

El primer objetivo a realizar es la creación de un **clúster**, el cual engloba diferentes bases de datos. Este servicio se configura automáticamente nada más darnos de alta en MongoDB. Dentro del clúster debe ser creada una **base de datos** la cual va a contener

una **colección**. Las bases de datos pueden contener varias colecciones al mismo tiempo. Las colecciones a su vez almacenan esas estructuras que contienen todos los datos.

La Figura 3.18 muestra el panel de control de MongoDB Atlas dónde se puede ver como este clúster contiene una base de datos llamada "Ejercicios" que contiene una colección llamada "Ejercicios". Dentro de esta colección se encuentran los datos almacenados en formato BSON.

Para establecer conexión con MongoDB desde la aplicación se necesita una cadena de caracteres que es proporcionada directamente por MongoDB en la pestaña de acceso a la base de datos. En ella se configura un nombre de usuario y contraseña que van a ser válidos para la conexión con terceros programas. El código para establecer la conexión y utilizar los datos (Data, 2022) es el siguiente:

```
client = pymongo.MongoClient("cadena de acceso")
db = client["Ejercicios"]
collection = db["Ejercicios"]
```

De esta manera si *collection* es llamada se pueden acceder a los datos ya sea para consultarlos o para introducirlos a la colleción.

Capítulo 4

Caso de Uso

Los resultados tras probar todos los ejercicios añadidos al programa fueron positivos, pues en todos esos casos se pudo realizar la monitorización correcta de estos ejercicios. La respuesta de los sensores tras la comunicación BLE se mantuvo estable durante todas las pruebas realizadas, y los datos obtenidos no tenían ningún tipo de latencia, pudiendo percibir estos resultados en tiempo real.

No obstante estos resultados fueron obtenidos manteniendo el sensor en la posición correcta. Para que el sensor esté en una posición correcta, este debe rotar únicamente sobre el eje X, tratando de mantener siempre el resto de ejes sin demasiadas variaciones. Por ejemplo, si el sensor está colocado en el envés de la mano o la muñeca, la palma de la mano debe mirar en todo caso al suelo.

La forma de colocar el sensor es importante pues si no se coloca en la posición adecuada para la sensorización de cada ejercicio los resultados pueden no ser precisos. Además, es bastante común que los pacientes no realicen los movimientos de manera precisa. Uno de las posibles maneras incorrectas de realizar el ejercicio consiste en que el paciente coloque la palma de la mano apuntando directamente al cuerpo. Varios experimentos fueron realizados colocando la mano de esa manera y los resultados obtenidos difieren un poco respecto a los resultados con una buena posición para la monitorización. En la mayoría de los casos la precisión del sensor baja un poco produciendo resultados levemente distorsionados. Además al tratar de superar el ángulo de 90 grados lo va a detectar siempre como 90. Por ejemplo, si se apunta al techo serían aproximadamente 90 grados, y al tratar de sobrepasar ese ángulo, el resultado en condiciones perfectas sería de aproximadamente 100-110 grados. Con la posición de la mano colocada de forma incorrecta al sobrepasar ese umbral siempre va a marcar 90 grados aproximadamente. Esa condición no supondría un problema, pues los ejercicios no están orientados a llevar al extremo los movimientos de las extremidades, sino tratar de moverlas de manera repetida. Además al ser configurables los ángulos necesarios para que el movimiento sea completado con éxito siempre es posible ajustarlo al comportamiento de cada paciente.

Figura 4.1: Varianzas en resultados según la postura.

La Figura 4.1 muestra el comportamiento del sensor tanto en una buena postura como en una postura menos indicada para la buena monitorización. Las cuatro primeras repeticiones representan aquellas con el sensor colocado en la muñeca y la palma de la mano se encuentra mirando al suelo, mientras que las últimas cuatro repeticiones la palma de la mano se encuentra mirando el cuerpo. En ella podemos observar como en las primeras repeticiones hay una respuesta mucho más precisa dónde hay más puntos mientras el movimiento de la mano sube y al encontrarse en los puntos máximos y mínimos, dando como resultado unas curvas más redondeadas. En cuanto a las medidas tomadas con el brazo ligeramente desviado podemos observar como estas medidas son menos suaves, realizando saltos entre las medidas mucho mayores. Además se puede apreciar el problema descrito anteriormente que consiste en que cuando se sobrepasa un ángulo concreto, el sensor lo detecta como un ángulo menor. De esta manera las curvas resultantes son cuadradas, pues cuando se llega al límite ya no puede subir más.

Tras repetir el experimento en varias ocasiones los resultados obtenidos son todos similares haciendo viable la monitorización de estos movimientos si la mano esta ligeramente girada respecto de la posición idónea para la monitorización. También fue observado el caso donde el brazo estaba bastante más inclinado, con la palma mirando prácticamente al techo, pero sin llegar a ser perpendicular con el techo y los resultados obtenidos no eran aptos para la monitorización de los ejercicios. Estos resultados obtenidos se muestran en la Figura 4.2 donde se estaba realizando el ejercicio de llevar la mano al hombro de forma repetida.

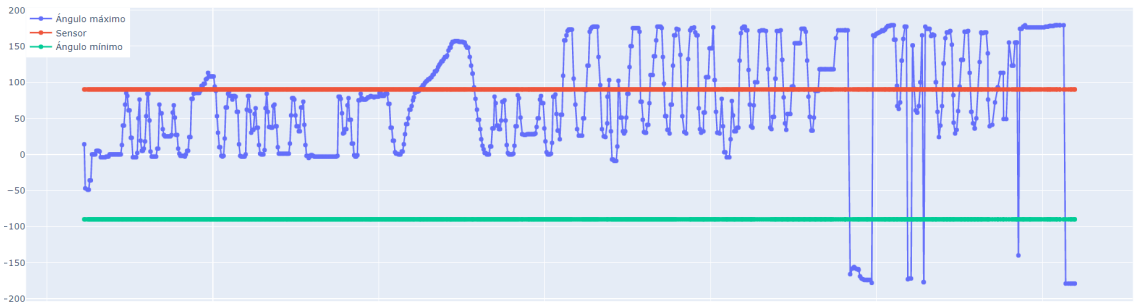

Figura 4.2: Medidas al realizar ejercicio con mala postura.

El ejercicio al ser realizado comienza con la mano mirando al suelo, pero al llegar al hombro se realiza un movimiento natural girando la mano de forma que ahora apunta al hombro. Como resultado el sensor comienza a tomar ángulos mucho mayores pues se encuentra dado la vuelta y se muestran comportamientos extraños como saltos exageradamente grandes como se muestra en las fases finales de la Figura 4.2. Esto se debe a que al estar el sensor dado la vuelta los ángulos de inclinación que detecta superan los 180, por lo que la siguiente medida será de un valor negativo. Los valores de la inclinación cubren los 360 grados, pero divididos en 180 grados positivos y 180 grados negativos. Por esa razón no es posible realizar una monitorización exitosa si se tiene una postura incorrecta, de modo que será necesario explicarle al paciente las posturas y las colocaciones del sensor para que no haya problemas de monitorización. Estos errores no pueden ser corregidos de ninguna forma, ya que son las propias medidas del sensor las que se generan de esta manera, lo cual arreglar ese tipo de problemas requeriría revisar el hardware.

Los ejercicios realizados con las piernas son exitosos pues las piernas no permiten tantos grados de rotación diferentes, ya que siempre van a estar mirando hacia el techo (los ejercicios se realizan con el cuerpo tumbado).

Finalmente caben destacar dos ejercicios que no han sido añadidos a la aplicación. El primero consiste en realizar un movimiento levantando los hombros de forma repetida. Este tipo de ejercicios no utiliza la inclinación del sensor de ninguna manera, por lo que otros sensores son requeridos. Estos ejercicios requieren de un enfoque diferente para ser monitorizados, como puede ser el uso de la aceleración lineal. Pero tras varios intentos por monitorizar este tipo de ejercicios usando estas medidas no fueron conseguidos unos valores que indiquen la correcta monitorización del ejercicio. Finalmente el otro ejercicio en cuestión es apretar un objeto con las manos. La placa de desarrollo Nicla Sense ME no cuenta con los sensores necesarios para medir la presión ejercida sobre un objeto, como puede ser un manómetro, por lo que este ejercicio queda descartado de la lista de ejercicios a realizar propuesta.

Capítulo 5

Conclusiones y Trabajo Futuro

5.1. Discusión personal

La investigación y el desarrollo de este Trabajo de Fin de Máster ha sido muy gratificante, ya que se han podido observar resultados tangibles. A pesar de ser un prototipo que requiere de varias mejoras para cumplir con las expectativas de una aplicación comercial, podemos concluir que el dispositivo Nicla Sense ME es apto para el uso de monitorización completa de varios ejercicios de rehabilitación en pacientes. También ha sido interesante poder haber contado con las opiniones del equipo médico del hospital de Fuenlabrada que pudo aportar una opinión sobre el sistema desarrollado. Estos profesionales de la salud le dieron el visto bueno a la aplicación, ya que consideraban que monitorizar los movimientos del modo que han sido implementados en este proyecto era la adecuada. Para ellos es importante saber que están realizando los movimientos abarcando el rango completo de movimiento que supone hacer el ejercicio y que sea repetido el ejercicio las veces que han acordado con el paciente. Las posibles mejoras que propusieron estarán pensadas como trabajo futuro, ya que, a pesar de ser propuestas muy interesantes para que sea una aplicación completa para el personal médico, por motivos de tiempo se descartaron al no estar dentro de los objetivos iniciales de investigación y del propio desarrollo de este Trabajo de Fin de Máster.

Desde mi punto de vista, la medicina tenderá a prestar servicios desde casa de manera progresiva. Después de todos estos años de COVID-19 la telemedicina está en el punto de mira de muchas empresas y puede ser un avance enorme la posibilidad de tener un servicio completo desde la comodidad de las casas. En situaciones críticas como la vivida en el año 2020 donde las Unidades de Cuidados Intensivos estaban al límite durante un periodo de tiempo muy grande. Durante ese periodo de tiempo muchas vidas se perdieron, ya que no había espacio suficiente para la estancia de todos los enfermos en estas unidades. Con el desarrollo de este Trabajo de Fin de Máster uno de los objetivos es reducir significativamente el tiempo que una persona que ya no está en estado crítico pase en estas unidades. De esta manera se conseguiría liberar espacio en estas unidades durante el tiempo de rehabilitación de los pacientes permitiendo la entrada de nuevos pacientes más graves en estas unidades. Además el periodo de rehabilitación puede ser un proceso de una duración bastante extensa hasta que el paciente se haya recuperado en su totalidad, por lo que si estos ejercicios pueden ser realizados desde la comodidad de su casa será más agradable para el paciente que estar hospitalizado mucho tiempo.

5.2. Conclusiones del sistema desarrollado

El nodo Nicla Sense ME que fue utilizado para el desarrollo de todo este Trabajo de Fin de Máster cumplió con las especificaciones propuestas inicialmente: al tener un tamaño físico muy reducido puede de ser colocarlo a los pacientes post-UCI sin que afecte de alguna forma a sus movimientos cotidianos. Bajo este argumento, el sensor puede ser colocado en una pulsera o tobillera facilitando la recogida de datos de actividad en un conjunto de ejercicios de rehabilitación. Este dispositivo al contar con la IDE de Arduino y una buena documentación permitió que la programación de los sensores fuese una tarea asumible en comparación con otros tipo de sensores con los que se han trabajado durante este Máster. Además al contar con conectividad BLE se pudieron enviar datos del sensor a la aplicación externa de forma casi instantánea, no observando latencias significativas durante la fase de experimentación, considerando la adquisición de datos en tiempo real. Es importante destacar que las comunicaciones inalámbricas facilitar la adopción de estas tecnologías en entornos médicos, ya que puede ser colocado al paciente sin ningún tipo de cable adicional, siendo una solución mucho más elegante sin perder la fiabilidad en la recogida de datos de los sensores.

Por otro lado, la integración de recogida de datos en un dashboard permite acceder a las sesiones de rehabilitación de pacientes de forma distribuida lo que facilita el seguimiento y control del personal médico. De esta forma los ejercicios se pueden realizar de forma semi-autónoma tanto desde la propia UCI como en el propio domicilio del paciente siguiendo las indicaciones del personal médico en todo momento. Este dashboard permite ajustar todos los parámetros que el personal médico ha considerado relevante como son: el número de repeticiones, el número de series y los umbrales de inclinación de los movimientos de los ejercicios de rehabilitación. El equipo de fisioterapia del hospital de Fuenlabrada ha propuesto estos parámetros a medir como los más adecuados para analizar en los progresos de los pacientes, sugiriendo la configuración personalizada de acuerdo al tipo de paciente. La aplicación desarrollada tiene la capacidad de adaptarse a todos los pacientes de acuerdo al tipo de patología y fase de recuperación en la que se encuentre, ya que en algunos casos su movilidad puede verse reducida no pudiendo alcanzar los rangos de movimiento propuestos inicialmente.

Algunas de las pruebas realizadas para comprobar la efectividad de la monitorización de los ejercicios fueron realizadas por mi abuelo Clemente, de 89 años de edad. Tras ajustar los parámetros de movimiento a las capacidades de mi abuelo, que no posee mucha movilidad en los brazos, pudo completar de forma exitosa 2 series de 10 repeticiones de un ejercicio que consistía en flexionar el codo llevando la mano al hombro. Tras comprobar que funciona con terceras personas y que prototipo era configurable a las necesidades de cada uno, se puede concluir que es viable la utilización del prototipo desarrollado en un abanico amplio de pacientes.

Por último, no queríamos dejar de lado el coste del prototipo, siendo su despliegue de bajo coste. El sistema cuenta con la placa de desarrollo Nicla Sense ME de un precio aproximado de 64€, una batería de litio con un precio de 6-10€ y una pulsera para guardar el sensor de unos 4-8€. Aunque el servicio de almacenamiento de datos basado en MongoDB Atlas es gratuito, una versión mejorada con mayor capacidad de almacenaje y procesamiento de datos no suponen gran desembloso. La versión gratuita es de 512Mb a 5Gb de almacenamiento, mientras que la versión Serverless, con hasta 1TB de almacenamiento disponible tiene un coste de 0,10$ por cada millón de lecturas realizadas a los datos, y finalmente la versión Dedicated la cual cuenta con hasta 4Tb de capacidad de almacenamiento 768Gb de RAM está por el precio de 57$ mensuales. Podemos concluir

que el coste total del prototipo es de 76€ lo convierte en una solución con un muy bajo coste y accesible para todo el mundo.

El código desarrollado durante este Trabajo de Fin de Máster puede encontrarse en el repositorio público Github Mansilla (2022).

5.3. Principales dificultades

Las principales dificultades surgidas durante el desarrollo de este trabajo fueron la selección de las diferentes medidas de los sensores que proporciona la placa Nicla Sense ME. Dicho sensor dispone de gran variedad de mediciones, siendo necesario la evaluación de que medida se ajusta mejor al propósito del proyecto. En ocasiones fueron necesarios cálculos externos, como el cálculo de la orientación del sensor en base a las medidas del acelerómetro y el giróscopo. Esos cálculos en ocasiones eran erróneos pues las unidades que eran utilizados por los sensores y los sitios de referencia eran diferentes ya que en ningún lugar expecificaba las unidades de esas medidas (en este caso radianes o ángulos).

Otra de las dificultades encontradas, fue la creación de la aplicación compatible con la librería Dash. Esto se debe a su compleja programación empleando el uso de callbacks como modo de activación de funciones. Los elementos que componen el dashboard pueden ser modificados gracias a estos callbacks, pero únicamente pueden ser modificados por uno de ellos. Al tratar de modificar un elemento desde diferentes callbacks el programa dejaba de responder, dificultando enormemente la tarea de depuración ante ciertas acciones.

Finalmente una dificultad que permaneció hasta el final del proyecto fue la conectividad BLE con la aplicación, ya que la configuración se realiza por un hilo de ejecución independiente, que ante una desconexión puntual supone que la aplicación deje de funcionar adecuadamente.

5.4. Trabajo futuro

En esta sección se propondrán algunas mejoras al sistema, una vez detectadas algunas deficiencias que podrían mejorar el sistema para ser desplegado en el ámbito médico. Sería conveniente implementar un sistema de inicio de sesión con roles asignados por cada usuario, que determine permisos de acceso a configuración y recogida de informes de la aplicación: si el usuario es un personal médico tendrá acceso a los historiales de los pacientes, con todo tipo de filtros para encontrar con facilidad la información que permita ver el desarrollo de cada paciente. El paciente solamente tendría acceso a las funciones de la aplicación que permite la monitorización del ejercicio a realizar, así como alternativamente acceso a su historial de ejercicios realizados. Para ello sería necesaria la implementación de una base de datos diferente a la que lleva el registro de ejercicios (aunque puede estar en el mismo clúster de MongoDB) para llevar todas las credenciales de inicio de sesión así como los roles que tiene asignado cada usuario.

Adicionalmente, la aplicación podría ser configurable permitiendo la incorporación de una lista de ejercicios a desarrollar por los pacientes con los parámetros de los mismos: respectivas repeticiones y series. De esta manera el paciente al entrar en la aplicación encontraría una tabla personalizada a las necesidades de cada paciente. Esta tabla sería personalizada por el personal médico en la aplicación de modo que guardaría un registro personal en la base de datos en la nube de cada paciente.

Otra mejora para el programa es añadir la funcionalidad de reconectar el sensor en caso de desconexión. Tras varios intentos por implementarlo tuvo que dejarse como trabajo

futuro, ya que se estaba invirtiendo demasiado tiempo en esta tarea. Al estar ejecutándose el código de la conexión BLE en una hebra o hilo de ejecución paralelo a la ejecución de la aplicación Dash tendría que detectar cuando ha ocurrido una desconexión y mostrarle esa información al usuario de la aplicación mediante algún mensaje o botón para volver a reconectarlo.

Sería interesante también la incorporación de una segunda placa de desarrollo Nicla Sense ME para permitir la monitorización de varias extremidades al mismo tiempo. La implementación sería relativamente sencilla pues solamente habría que duplicar partes del código y ajustar algunas otras para permitir la interactividad del sistema con ambas placas de desarrollo. Este aspecto permitiría al personal médico evaluar simetrias entre ejercicios y disfunciones motoras.

Tras los experimentos realizados con la aceleración lineal proporcionada por el sensor acelerómetro, sería viable también indicar si el ejercicio se está haciendo con movimientos correctos, o están siendo realizados a una velocidad mayor o menor a la establecida por el personal médico.

También se podría desarrollar otra forma más visual de representar los movimientos recogidos por el sensor, representándolo como si fuera un videojuego. De esta manera se abre un mundo de posibilidades que son bastante más interesantes de representar visualmente los movimientos que debe realizar el paciente. Por ejemplo, podría aparecer un modelo 3D de una silueta humana que sirva de ejemplo para que el paciente siga esos movimientos. Aparte de eso, visualmente la aplicación podría mejorar bastante. Ya que se trata de un prototipo y no es el objetivo de la práctica no se ha invertido tiempo en la parte visual. Al tratarse de una aplicación web cuyos elementos son básicamente HTML se podría añadir todo el código CSS para modificar el estilo de todos los componentes. Además, en caso de preferir una aplicación que vaya a ser descargada en vez de una aplicación web siempre se podrían utilizar otras librerías de Python para la creación de esta, pudiendo mantener las gráficas que proporciona Plotly.

Dash tiene soporte con dispositivos móviles, por lo que el programa podría ser ejecutado desde un móvil sin problemas. Todo el mundo tiene un móvil en su disposición mientras que mucha gente no posee un ordenador personal. Esto facilitaría la portabilidad de la aplicación y añadiendo descentralización al sistema IoT. La aplicación puede estar lanzada en una máquina virtual y el resto de dispositivos pueden acceder a la aplicación mientras se tenga la URL correspondiente.

Respecto al prototipo comercial, sería conveniente algún tipo de correa personalizada que sujete la placa de desarrollo y la batería portátil de forma adecuada mediante una caja o encapsulado con un diseño ergonómico. Para las pruebas realizadas en este proyecto el sensor estaba alojado en una funda para móvil deportiva.

Chapter 6

Introduction

The Intensive Care Unit or ICU is a section of a hospital where patients with life-threatening health problems are constantly monitored and intensively assisted. After having suffered an accident, operation, or illness serious enough, a temporary stay in one of these units would be necessary.

After several years of the presence of Covid-19 in our lives, the importance of the Intensive Care Unit should be highlighted, since on many occasions the virus has posed a life-threatening danger to people. Having significantly reached the number of cases of people who end up in the ICU, it is interesting to improve the recovery methods of patients. The technology provided by the world of the Internet of Things opens the door to remote monitoring of patients, to avoid collapses in the rooms of these Intensive Care Units, in addition to speeding up and improving the treatment of health personnel with the patient. In this way, those who are already out of mortal risk can recover at home without occupying a room while they can continue to be monitored by doctors. In addition, by not using medical materials, costs will be reduced both in medical services and in the maintenance of the equipment looking at the long term.

Stays in the ICU entail great physical and mental wear for the patient due to the long intervals of time that they must be on a stretcher with hardly any movement. These times can range from a few days or a couple of weeks to several months where the muscle, among other things, is deteriorated.

The result of the deterioration is such that the patient cannot perform basic movements without external help, which is why a rehabilitation process is required to regain mobility and motor skills that allow them to perform basic day-to-day functions. Today this task falls to the physiotherapy service, but in some cases, the patients themselves can begin to carry out the rehabilitation process by themselves if a tool for motivation and control of the progress of the exercises is available.

In this Master's Thesis, the necessary research has been carried out to cover the basic needs of a program that monitors the patient to determine if he is performing the proposed exercises correctly or not. In addition, the proposed system allows the doctor to monitor the patient with real-time results, so that he can observe the consistency and improvement of the exercises performed.

6.1. Motivation

In recent years we have been able to observe how different sensors are used to create smartwatches that monitor all kinds of exercises, from running to lifting weights. These same sensors are becoming more compact and very low cost, being a great tool for monitoring any type of exercise, including those related to rehabilitation.

The application of these sensors in uses related to medicine is already occurring, where a trend towards telemedicine is being created, which is growing day by day. The uses of sensors in telemedicine are very varied, for example, the use of IMU (Inertial Measurement Unit) in the identification of pathological motor characteristics. A specific case is the use of these IMUs to evaluate the progression of certain motor diseases such as Parkinson's disease, being able to differentiate patients with dominant tremor from patients without that dominant tremor. "During the study, a correlation was found between the severity of the disease and the patient's mobility, thus indicating that the disease had developed" (PT y more, 2018). Another example of the use of IMUs in medicine is the detection of sports injuries in runners, where it can be observed in runners with some type of injury that the vertical load indexes are much higher in those with injuries compared to those who are not injured.

Telerehabilitation is also in the sights of doctors and physiotherapists so that they can constantly help patients in their rehabilitation phase. The use of IMUs opens a door to this great world of telerehabilitation where it allows the constant monitoring of patients instead of periodic visits to the medical center.

During the development of this Master's Thesis, it has been possible to have the help of the ICU service of the Fuenlabrada hospital, which proposes to have an IoT system that meets two objectives:

- Collection of information associated with rehabilitation exercises. Using sensors on patients placed on their extremities to collect this information that allows medical staff to evaluate the patient's progress and adapt the rehabilitation exercise table.
- Feedback-motivation to the patient to advance in the rehabilitation process. Offering an interactive interface for the user where they can see their progress at all times, as well as the rest of the exercises in the table proposed by the medical staff to observe their improvement and motivate them to continue with the exercises.

The rehabilitation exercises proposed by the medical staff of the Fuenlabrada hospital are the following:

- The patient flexes the elbow by bringing his hand to the shoulder.
- The patient shrugs his shoulders, bringing them toward his head.
- The patient raises the arm towards the ceiling, on one side or both.
- The patient raises the foot towards the knee, on one side or both alternately.
- The patient raises the leg on one or both sides alternately.
- The patient makes a fist by compressing an object.

Alternatively, offer the ability to customize exercises based on patient needs to optimize patient recovery. In this way, in case of not being able to follow the proposed exercises and thus avoid accidental injuries, or in case of having obtained a good performance with the

exercises, it would also be possible to slightly harden them. In addition, for those patients with progressive diseases, it also allows the rhythm of those exercises to be adapted to stop these diseases.

In addition, the choice to use the Nicla Sense ME sensor, which was donated by Bosch to the Faculty of Informatics, was to use Bosch resources in a Master's Thesis and thus be able to access the Bosch Chair awards BOSCH (2022).

6.2. State of the art

As already mentioned in the introduction, there are numerous applications to combine the benefits that Internet of Things systems bring with medicine.

The system proposed by Edge Hill University (McHale, 2020) suggests the possibility of creating an IoT system for remote monitoring of people suffering from epilepsy. Due to the complexity that this problem supposes and that each one of the causes of epilepsy is different, it proposes a completely customizable system that adapts to each individual so that IoT technologies are much more precise in detecting these problems. The way to detect these epileptic seizures is by using classification techniques and cluster analysis to create groups for each type of Parkinson's patient. The way to capture this data is through sensors placed in different parts of the individual, which in turn are connected to an IoT service where that data will be analyzed. Due to the variety of sources of epileptic seizures, each individual will have a personalized monitoring plan where they will have their sensors that will be different from other patients who suffer different types of seizures.

Another use of integrated IoT systems in medicine is the use of Kinect (Bojan Milosevic, 2020) for remote monitoring of rehabilitation programs. Kinect is a very low-cost video camera from Microsoft, which would serve as an input device to monitor the movements made by patients. The use of different algorithms is proposed, such as an algorithm that is provided in the SDK of the Kinect itself to determine the position of the body when recorded by the video camera and thus determine if the postures made are correct. They also raise the potential use of different sensors such as orientation calculated from gyroscopes and accelerometers, as well as the use of different filters to improve the accuracy of these measurements. By having an IoT system, complete rehabilitation from home is possible, obtaining positive results for a large number of exercises performed.

6.3. Objectives

As already mentioned in the introduction of the work, the main objective of this project is the creation of an IoT system that allows the monitoring of exercises for the rehabilitation of patients who have just finished their stay in the ICU. The objective is to be able to monitor the exercises to determine if they have been completed correctly or not and also to be able to follow up on the patients to be able to observe the progress of their recovery.

Figure 6.1 shows the structure of the proposed system, which consists of three main components. A sensor in charge of capturing the movements made by the patient will be placed in the corresponding parts of the body to monitor each of the exercises. A central application in charge of offering an interface to the user so that he can use it simply, showing the results of the exercises to be carried out and communicating directly with the development board via Bluetooth. Finally, a database in the cloud in charge of storing the data of the exercises is carried out for the future visualization of the data generated in said exercises.

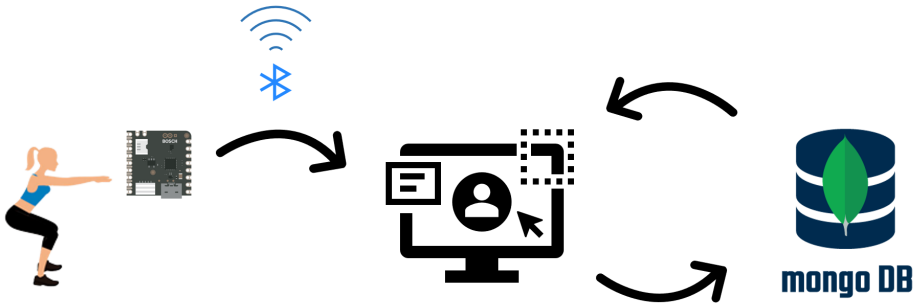

Figure 6.1: Representation of the proposed system (GARCÍA, 2017; Jadhav, 2021; ICONS, Último acceso 2022; Router, Último acceso 2022)

The proposed objectives to meet the system requirements are as follows:

- Explore Arduino programming tools to dump code to the Nicla Sense ME development board.
- Explore capabilities of the Nicla Sense ME development board.
- Analyze the different measures offered by the Nicla Sense ME development board to locate the necessary measures to monitor the exercises.
- Investigate how to communicate between the development board and the main program.
- Develop an application that allows the correct monitoring of the exercises proposed by the physiotherapists.
- Creation of an exercise record to monitor the evolution of patients over time.

6.4. Work plan

This Master's Thesis has been structured following the objectives set out in the previous section so that a certain amount of time was dedicated to the analysis or development of each objective.

6.4.1. Exploring Arduino IDE

The first few days were dedicated to installing and exploring the Arduino IDE. This program allows the easy upload of code to development boards. The installation of the program took longer than expected because there were several versions of the program and not all of them worked correctly. After finding and installing the latest version, it was necessary to install various library modules that allow programming on the Nicla Sense ME board. In addition, extra time was also dedicated to exploring the visualization of the data, since they can be visualized through the serial port to which the development board is directly connected, where the data is displayed as a terminal, or use the " serial plotter" to visualize the data using line graphs.

6.4.2. Exploring Nicla Sense ME

The next two weeks were my first contact with the Nicla Sense Me development board. For this, some exercises were replicated that are available on the documentation page of the board. The first of the exercises to replicate was to modify the LED that is integrated into the development board, alternating random colors at intervals of 1 second. The next exercise to understand the Nicla Sense ME a little more was to print the data from various sensors through both the serial port and the serial plotter. The first data to display was the orientation of the sensor, the data from the accelerometer, and the data obtained from the gyroscope, for their respective X, Y, and Z axes.

Finally, the last test to be carried out with the sensor was to establish a connection via Bluetooth with a dashboard or data panel whose code is available in the documentation of the development board, as well as a link to the dashboard itself for viewing this data. After connecting the plate with the dashboard, it was possible to view all the data from the Nicla Sense ME, as well as a representation with a 3D model of the plate that reproduced the movements of the plate itself in real-time, which was the main inspiration for the development of the monitoring application.

6.4.3. Analysis of sensor data

The following three weeks were devoted entirely to carrying out tests on the different data obtained to determine the measurements of the different sensors that could be useful, discarding those sensors that did not provide useful information for the development of this practice. Among these tests, the calculation of the orientation angles from the accelerometer and gyroscope sensors stands out, adding additional filters to avoid erroneous data and the testing of all the sensors that were related to the movement or orientation of the development board.

6.4.4. Communication with the application

The next two days were spent creating a small program written in the Python programming language to establish a Bluetooth connection with the development board.

6.4.5. Application development

For the development of the application, the highest percentage of time was assigned to the other objectives, since it was the task that required the most work and the most tests for its correct operation. This stage lasts approximately one and a half or two months, where work was carried out in parallel on the analysis of the data from the sensors and the creation of an exercise record. During this time, various tests were carried out with Dash, the Python library on which the final application is mainly based. These tests consisted of testing the different tools that Dash offers, such as the possibility of creating a fully interactive dashboard with different ways of interacting with the data and visualizing it, as well as the possibility of adding 3D models to represent the movement of the sensor.

6.4.6. Creating an exercise log

The exploration of a database to save the record of all the exercises performed was carried out for approximately three days. The first day was dedicated to the investigation of a SQL database hosted locally on the computer. After several tests and re-structuring

of the project objectives, the decision to use that database changed to the possibility of using a database in the cloud using MongoDB, which would be a better decision for the development of an IoT system. . Two days were necessary for the investigation, and the creation of the database in the cloud, as well as its integration in the application.

6.4.7. Extras

The last days dedicated to the development of this Master's Thesis were dedicated to the investigation of possible improvements or additions to the final application. The investigation carried out was the possibility of adding artificial intelligence to the camera in charge of recording the movements of the patients when performing the exercises, so that this artificial intelligence can tell us if the patient's posture is wrong or is the right one for the exercise in question. Furthermore, the possibility of adding a second Nicla Sense ME development board to monitor multiple limbs at the same time was investigated.

Chapter 7

Conclusions and Future Work

7.1. Personal discussion

The research and development of this Master's Thesis has been very rewarding, since tangible results have been observed. Despite being a prototype that requires several improvements to meet the expectations of a commercial application, its use for complete monitoring of various rehabilitation exercises is viable. It has also been interesting to have had the opinions of the medical team at the Fuenlabrada hospital, which could provide an opinion on the system developed. These health professionals gave the application the go-ahead, since they observed that monitoring movements in the way that they have been implemented in this project was adequate. For them it is important to know that they are performing the movements covering the full range of motion involved in doing the exercise and that the exercise is repeated as many times as they have agreed with the patient. The possible improvements that they propose will be thought of as future work, since, despite being very interesting proposals to make it a complete application for medical personnel, they do not fall within the research and development objectives of this Master's Thesis.

I think that medicine will tend to progressively provide services from home. After all these years of COVID-19, telemedicine is in the sights of many companies and the possibility of having a complete service from the comfort of home can be a huge advance. In critical situations such as the one experienced in the year 2020 where the Intensive Care Units were at the limit for a very long period of time. During that period of time, many lives were lost, since there was not enough space for all the patients to stay in these units. With the development of this Master's Thesis, one of the objectives is to significantly reduce the time that a person who is no longer in critical condition spends in these units. In this way, it would be possible to free up space in these units during the rehabilitation time of the patients, allowing the entry of new, more seriously ill patients in these units. In addition, the rehabilitation period can be a process of a fairly long duration until the patient has fully recovered, so if these exercises can be performed from the comfort of your home, it will be more pleasant for the patient who will be hospitalized for a long time. weather.

7.2. Conclusions of the developed system

The Nicla Sense ME node that was used for the development of all this Master's Thesis complied with the specifications initially proposed since it is adjusted to have a very small

size to be able to place it on patients without affecting movements in any way. of these. In this way it can be placed on a bracelet or anklet so that it is very easy to put on and make movements with it on. This device, having the Arduino IDE and good documentation, made the programming of the sensors a very simple task compared to other sensors that have been worked with during this Master. In addition, by having BLE connectivity, data from the sensor could be sent to the application instantly without any type of observable latency, allowing the measurements of these sensors to be reached in real time. This type of wireless communication is also interesting to improve the quality of the final product, since it can be placed on the patient without any additional cable, being a much more elegant solution without losing the reliability of sensor data collection.

Integration in the dashboard allows access to patient rehabilitation sessions in a distributed manner, which facilitates the monitoring and control of medical personnel. In this way, the exercises can be performed semi-autonomously from the ICU itself as at home, following the instructions of the medical staff at all times. This dashboard allows you to adjust all its parameters such as the number of repetitions, the number of series and the inclination of the movement that must be performed to be completed following the recommendations of the health personnel of the Fuenlabrada hospital. This allows the application to have the ability to adapt to all patients, since in some cases their mobility may be reduced, not being able to reach the initially proposed ranges of motion.

Some tests carried out to verify the effectiveness of exercise monitoring were carried out by my grandfather Clemente, 89 years old. After adjusting the movement parameters to the capabilities of my grandfather, who does not have much mobility in his arms, he was able to successfully complete 2 sets of 10 repetitions of an exercise that consisted of flexing the elbow by bringing the hand to the shoulder. After verifying that it works with third parties and that it was adjustable to the needs of each one, I was able to conclude that it is possible for other people to use this implemented system, giving them a brief introduction to how to place the sensors depending on the movement, and how to adjust the parameters for start the exercise.

7.3. Main difficulties

The main difficulties that arose during the development of this work were the selection of the different measurements of the sensors provided by the Nicla board. As there was such a variety of data, a very extensive experimentation phase was necessary with all these measures. External calculations were sometimes necessary, such as calculating the orientation of the sensor based on measurements from the accelerometer and gyroscope. Those calculations were sometimes wrong because the units that were used by the sensors and the reference sites were different since nowhere specified the units of those measurements (in this case radians or angles).

Another difficulty was the creation of the application with the Dash library. This is due to its complex programming employing the use of callbacks as a way of activating functions. The elements that make up the dashboard can be modified thanks to these callbacks, but they can only be modified by one of them. When trying to modify an element from different callbacks, the program stopped responding, greatly hindering the task of modifying the program when performing certain actions.

Finally, a difficulty that remained until the end of development was BLE connectivity with the application, since it is being launched from an execution thread when a disconnection occurs, it is not managed efficiently. After various tests, its development was discarded and proposed as future work because it was taking too long to implement.

7.4. Future work

Starting from the base established at the end of this work, several elements can be modified to create a more complete application-oriented to be used by medical personnel. To meet these expectations, it would be necessary to implement a login system with roles assigned to each user. If the user is a medical staff, they will have access to patient records, with all kinds of filters to easily find the information that allows them to see the development of each patient. The patient would only have access to the functions of the application that allow monitoring of the exercise to be performed, as well as alternatively access to their history of exercises performed. To do this, it would be necessary to implement a different database than the one that keeps the record of exercises (although it may be in the same MongoDB cluster) to keep all the login credentials as well as the roles assigned to each user.

Additionally, so that the application is much more complete, a list of exercises can be added to comply with their respective repetitions and series by the patients. In this way, when entering the application, the patient will find a table personalized to the needs of each patient. This table would be personalized by the medical staff in the application so that it would save a personal record in the cloud database of each patient.

Another improvement for the program is to add the functionality to reconnect the sensor in case of disconnection. After several attempts to implement it, it had to be left as future work, since too much time was being spent on this task. Since the BLE connection code was running in a thread or thread of execution parallel to the execution of the Dash application, it had to detect when a disconnection had occurred and show that information to the application user through a message or button to reconnect it.

It would also be interesting to add a second Nicla Sense ME, development board, to work with several extremities at the same time. The implementation would be relatively simple since it would only be necessary to duplicate parts of the code and adjust some others to allow interactivity of the system with both development boards. During the development phase of the application, a second Nicla development board was available to add to the system, but the idea was passed on to future work because it would not provide much extra information for the development of the work compared to what was achieved with a single board. However, monitoring multiple limbs at the same time could be extremely useful for a real application.

After the experiments carried out with the linear acceleration provided by the accelerometer sensor, it would also be feasible to indicate if the exercise is being done with correct movements, or if they are being performed at a speed greater or less than that established by the medical staff.

Another more visual way of representing the movements picked up by the sensor could also be developed, representing it as if it were a video game. In this way, a world of possibilities opens up that is much more interesting than visually representing the movements that the patient must carry out. For example, a 3D model of a human silhouette could appear to serve as an example for the patient to follow those movements. Other than that, visually the app could use a lot of improvement. Since it is a prototype and it is not the objective of the practice, no time has been invested in the visual part. As it is a web application whose elements are HTML, all the CSS code could be added to modify the style of all the components. Also, if you prefer an application that is going to be downloaded instead of a web application, you could always use other Python libraries to create it, being able to keep the graphs provided by Plotly.

Dash has support for mobile devices, so the program could be run from mobile without

problems. Everyone has a mobile phone at their disposal while many people do not have a personal computer. This would facilitate the portability of the application and add decentralization to the IoT system. The application can be launched on a virtual machine and the rest of the devices can access the application as long as the corresponding URL is available.

Regarding the hardware, some kind of custom strap would be needed to hold the development board and the portable battery properly and if possible, have a nice design. For the tests carried out in this work, the sensor was placed in a case to carry the mobile when doing sports and this solution for a prototype is viable, but not for a serious work tool.

An advantage of the proposed system is also its low cost since the elements that make up this system are not expensive. The system has the Nicla Sense ME development board that costs approximately €64, a lithium battery that costs €6-10, and a bracelet to store the sensor for €4-8. Additionally, the MongoDB Atlas database service can be improved in case you want to access greater data storage and processing capacity, different paid versions can be purchased. Being the free version from 512Mb to 5Gb of storage, the Serverless version, with up to 1TB of available storage and with a cost of $0.10 per million readings made to the data, and finally, the Dedicated version which has up to 4Tb of storage capacity 768Gb of RAM for the price of $57 per month. The total initial cost of approximately €76 makes it a very low-cost solution and accessible to everyone.

The code developed during this Master's Thesis can be found in the following Github Mansilla (2022).

Bibliografía

Y así, del mucho leer y del poco dormir, se le secó el celebro de manera que vino a perder el juicio.
(modificar en Cascaras\bibliografia.tex)

Miguel de Cervantes Saavedra

ARDUINO. Arduino and ai. `https://github.com/arduino/ArduinoAI`, 2022a. [Online].

ARDUINO. Arduino ide. `https://www.arduino.cc/en/software`, 2022b. [Online].

ARDUINO. millis(). `https://www.arduino.cc/reference/en/language/functions/time/millis/`, 2022c. [Online].

ARDUINO. Arduinoble. `https://www.arduino.cc/reference/en/libraries/arduinoble/`, Último acceso 2022. [Online].

BCD. Foro: Does dash support opencv video from webcam? `https://community.plotly.com/t/does-dash-support-opencv-video-from-webcam/11012`, 2018. [Online].

BLIDH, H. bleak documentation. `https://buildmedia.readthedocs.org/media/pdf/bleak/stable/bleak.pdf`, 2022. [Online].

BOJAN MILOSEVIC, A. L. . E. F. Kinect and wearable inertial sensors for motor rehabilitation programs at home: state of the art and an experimental comparison. 2020.

BOSCH. Cátedra extraordinaria bosch-ucm. `https://gaia.fdi.ucm.es/research/catedraBosch/`, 2022. [Online].

CHIANG, C.-Y., CHEN, K.-H., LIU, K.-C., HSU, S. J.-P. y CHAN, C.-T. Data collection and analysis using wearable sensors for monitoring knee range of motion after total knee arthroplasty. *Sensors*, vol. 17(2), página 418, 2017.

DASH, P. Dash python user guide. `https://dash.plotly.com`, 2022. [Online].

DATA, C. Python dash web application connected to live database. `https://www.youtube.com/watch?v=DWqEVpOfYxE`, 2022. [Online].

ELECTRONICS, D.-K. Nicla sense me. `https://www.digikey.es/es/product-highlight/a/arduino/nicla-sense-me`, 2021. [Online].

ELT. Ble (bluetooth low energy). https://www.elt.es/ble-bluetooth-low-energy, Último acceso 2022. [Online].

Flask. Flask documentation. https://flask.palletsprojects.com/en/2.1.x/, Último acceso 2022. [Online].

GARCÍA, L. Cómo realizar el entrenamiento funcional: fases y ejercicios. https://www.webconsultas.com/ejercicio-y-deporte/tablas-de-ejercicios/como-realizar-el-entrenamiento-funcional-fases-y-ejercicios, 2017. [Online].

Gordon, C. Quaternion representation of 3d orientation and rotation for sensor fusion applications. https://sites.tufts.edu/eeseniordesignhandbook/files/2021/05/Gordon_QuaternionRepresentation.pdf, 2021. [Online].

ICONS, C. Bluetooth icon. https://cdn-icons-png.flaticon.com/512/143/143817.png, Último acceso 2022. [Online].

de cuidados intensivos, U. Profesionales del hospital de fuenlabrada ponen en marcha una aplicación para familiares. https://www.comunidad.madrid/noticias/2018/03/26/profesionales-hospital-fuenlabrada-ponen-marcha-aplicacion-familiares, 2018. [Online].

Jadhav, A. Nicla sense me: Arduino's upcoming smallest form factor ai development board. https://www.electronics-lab.com/nicla-sense-me-arduinos-upcoming-smallest-form-factor-ai-development-board/, 2021. [Online].

Makai, M. React. https://www.fullstackpython.com/react.html, 2022. [Online].

Mansilla, G. G. Github proyecto. https://github.com/GuilleGarciaMansilla/IoTTFM, 2022. [Online].

mariaguutierrez. ¿qué posición tiene una persona que está acostada boca arriba? https://es.quizzclub.com/trivia/que-posicion-tiene-una-persona-que-esta-acostada-boca-arriba/, Último acceso 2022. [Online].

Marquínez, P. Displaying on-board sensor values on a webble dashboard. https://docs.arduino.cc/tutorials/nicla-sense-me/web-ble-dashboard, 2022. [Online].

McHale, S. A. An iot approach to personalised remote monitoring and management of epilepsy. 2020.

McWhorter, P. 9-axis inertial measurement unit (imu). https://toptechboy.com/arduino-based-9-axis-inertial-measurement-unit-imu-based-on-bno055-sensor/, 2019. [Online].

nguyen37. Arduino integrated dumbbell. https://github.com/nguyen37/Arduino_Integrated_Dumbbell/blob/master/Arduino_Integrated_Dumbbell/Arduino_Integrated_Dumbbell.ino, 2017. [Online].

OpenCV. Opencv webpage. https://opencv.org, Último acceso 2022. [Online].

plotly. Introducing dash. https://medium.com/plotly/introducing-dash-5ecf7191b503, 2017. [Online].

PT, F. P. y MORE. Wearable movement sensors for rehabilitation: A focused review of technological and clinical advances. *Wiley Online Library*, 2018.

RICCI, L., TAFFONI, F. y FORMICA, D. On the orientation error of imu: Investigating static and dynamic accuracy targeting human motion. *PloS one*, vol. 11(9), página e0161940, 2016.

ROMERO, S. Arduino nicla sense me cheat sheet. `https://docs.arduino.cc/tutorials/nicla-sense-me/cheat-sheet#sensors`, 2022. [Online].

ROUTER, O. What is mongodb? the nosql database explained easily. `https://www.opc-router.com/what-is-mongodb/`, Último acceso 2022. [Online].

SABER-SHEIKH, K., BRYANT, E. C., GLAZZARD, C., HAMEL, A. y LEE, R. Y. Feasibility of using inertial sensors to assess human movement. *Manual Therapy*, vol. 15(1), páginas 122–125, 2010. ISSN 1356-689X.

SHULL, P. B., JIRATTIGALACHOTE, W., HUNT, M. A., CUTKOSKY, M. R. y DELP, S. L. Quantified self and human movement: a review on the clinical impact of wearable sensing and feedback for gait analysis and intervention. *Gait & posture*, vol. 40 1, páginas 11–9, 2014.

THREE.JS. Euler. `https://threejs.org/docs/#api/en/math/Euler`, Último acceso 2022. [Online].

TME. ¿cÓmo funciona y quÉ hace el acelerÓmetro? `https://www.tme.com/ve/es/news/library-articles/page/22568/Como-funciona-y-que-hace-el-acelerometro/`, 2020. [Online].

TOMVDS. Foro unity: Euler, quaternions, radians, degrees... huh? `https://forum.unity.com/threads/euler-quaternions-radians-degrees-huh.53407/`, Último acceso 2022. [Online].

TURBOSQUID. Plataforma modelos 3d. `https://www.turbosquid.com/es/Search/3D-Models/free/arm`, 2022. [Online].

–¿Qué te parece desto, Sancho? – Dijo Don Quijote –
Bien podrán los encantadores quitarme la ventura,
pero el esfuerzo y el ánimo, será imposible.

Segunda parte del Ingenioso Caballero
Don Quijote de la Mancha
Miguel de Cervantes

–Buena está – dijo Sancho –; fírmela vuestra merced.
–No es menester firmarla – dijo Don Quijote–,
sino solamente poner mi rúbrica.

Primera parte del Ingenioso Caballero
Don Quijote de la Mancha
Miguel de Cervantes