UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

CARRERA DE INGENIERÍA EN MECATRÓNICA

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

TEMA:

"***Dispositivo para la medición de ángulos de las extremidades superiores por medio de sensores inerciales.***"

AUTOR: Jonathan Andrés Criollo Mediavilla

DIRECTOR: MSc. Iván Iglesias Navarro

IBARRA – ECUADOR

UNIVERSIDAD TÉCNICA DEL NORTE
BIBLIOTECA UNIVERSITARIA

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

1. IDENTIFICACIÓN DE LA OBRA

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

DATOS DE CONTACTO			
CÉDULA DE IDENTIDAD:	100385896-4		
APELLIDOS Y NOMBRES:	Criollo Mediavilla Jonathan Andrés		
DIRECCIÓN:	Ibarra, Carlos Proaño 2-20 y C1		
EMAIL:	jacriollom@utn.edu.ec		
TELÉFONO FIJO:	N/A	**TELÉFONO MÓVIL:**	0983516592

DATOS DE LA OBRA	
TÍTULO:	"Dispositivo para la medición de ángulos de las extremidades superiores por medio de sensores inerciales."
AUTOR (ES):	Jonathan Andrés Criollo Mediavilla
FECHA: DD/MM/AAAA	12/05/2022
SOLO PARA TRABAJOS DE GRADO	
PROGRAMA:	■ **PREGRADO** ☐ **POSGRADO**
TITULO POR EL QUE OPTA:	Ingeniero en Mecatrónica
ASESOR /DIRECTOR:	MSc. Iván Iglesias

2. CONSTANCIAS

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

Ibarra, a los 29 días del mes de junio del 2022

EL AUTOR:

Criollo Mediavilla Jonathan Andrés

CERTIFICACIÓN

En calidad de tutor del trabajo de grado titulado: “Dispositivo para la medición de ángulos de las extremidades superiores por medio de sensores inerciales”, certifico que el presente trabajo fue desarrollado por el egresado Criollo Mediavilla Jonathan Andrés, bajo mi supervisión.

IVAN IGLESIAS NAVARRO

Firmado digitalmente porIVAN IGLESIAS NAVARRO
Motivo:Apruebo este documento
Fecha:2022-06-28 14:54-05:00

MSc. Iván Iglesias

DIRECTOR DEL TRABAJO DE GRADO

AGRADECIMIENTO

Agradezco a mi madre que está en cada paso de mi vida apoyándome incondicionalmente y que siempre ha confiado en mí, a Erika quien se convirtió en mi compañera de vida y la cual me impulsa cada vez que yo quiero rendirme. También agradezco a todas las personas que aportaron en mi vida para estar en donde estoy: familia, amigos y conocidos.

Criollo Mediavilla Jonathan Andrés

DEDICATORIA

Este trabajo de titulación me lo dedico por todos los años de estudio y por no rendirme a pesar de todo el sacrificio que representó en mi vida.

A mi madre porque se todo el esfuerzo que tuvo que hacer para poder darme el estudio.

Resumen

Las lesiones en las extremidades superiores ocurren con mucha frecuencia por practicar algún deporte, una caída o un accidente dejando parcial o totalmente inmovilizado al individuo. En el siguiente documento se describe la investigación para la creación de un dispositivo capaz de medir ángulos mediante sensores inerciales, ideal para especialistas en el área de terapia física que requieren tomar ángulos en procesos de rehabilitación de extremidades superiores, el propósito de este proyecto es presentar una herramienta tecnológica y funcional que sea de fácil manipulación.

En la metodología se detalla la investigación utilizada para realizar la investigación del dispositivo además de los requisitos de diseño se obtuvieron a partir de la casa de la calidad.

El registro de la evolución del paciente se realiza de manera digital a través de una aplicación de escritorio que puede ser instalado en cualquier computador con sistema operativo Windows. Cuenta además con una base de datos integrada que asegura la perpetuidad de la información en la nube.

Abstract

Injuries to the upper extremities occur very frequently due to sports, a fall or an accident, leaving the individual partially or totally immobilized. The following document describes the research for the creation of a device capable of measuring angles using inertial sensors, ideal for specialists in the area of physical therapy that require taking angles in rehabilitation processes of upper extremities, the purpose of this project is to present a technological and functional tool that is easy to handle.

The methodology details the research used to carry out the investigation of the device in addition to the design requirements were obtained from the house of quality.

The registration of the patient's evolution is done digitally through a desktop application that can be installed on any computer with Windows operating system. It also has an integrated database that ensures the perpetuity of the information in the cloud.

ÍNDICE DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

CAPÍTULO I

INTRODUCCIÓN

1.1. Problema

Las lesiones en las extremidades superiores ocurren por practicar algún deporte, una caída o un accidente dejando parcial o totalmente inmovilizado al individuo. Los especialistas una vez determinado el diagnóstico, tratan al paciente con técnicas de rehabilitación y terapia, manteniendo controles clínicos periódicos y frecuentes según la complicación de la lesión.
La confederación Mundial de la Fisioterapia [1], menciona que el conjunto de métodos, actuaciones y técnicas mediante la aplicación de medios físicos, curan, previenen, recuperan y adaptan a personas afectadas por disfunciones somáticas permitiendo mantener independencia en las actividades.

Los expertos encargados en la rehabilitación registran datos de la evolución del paciente mediante la lectura de ángulos articulares, los cuales pueden ser tomados con un margen de error amplio, esto sea por fallas humanas o por carencia de instrumentos [2].

La ciencia y la tecnología se desarrolla con el objetivo de generar avances al servicio de la humanidad, brindando mayor eficiencia, eficacia y bienestar, lo que concede al hombre la posibilidad de mejorar su calidad de vida. Utilizando equipos y herramientas que contribuyan al trabajo de los profesionales de esta área [3], abordando como solución la recepción de señales mediante sensores, procesamiento de imágenes mediante cámaras los cuales tiene costos elevados.

Es por lo que surge la necesidad de construir un dispositivo capaz de recolectar, procesar datos y así contribuir con los profesionales involucrados en el área de terapia física, aportando herramientas de estudio para posteriores investigaciones.

1.2. Objetivos

1.2.1. Objetivo General

Construir un dispositivo para la medición de ángulos de las extremidades superiores por medio de sensores inerciales.

1.2.2. Objetivos específicos

- Investigar el comportamiento de las extremidades superiores del ser humano al formar ángulos en procesos de terapia y rehabilitación física.
- Diseñar un dispositivo que reciba, procese y envíe señales proporcionadas por el sensor inercial.
- Implementar una aplicación en la cual se visualicen los datos y movimiento que realiza la extremidad.
- Validar el funcionamiento del sistema.

1.3. Alcance

Se realizará un dispositivo formado por sensores inerciales para la lectura de ángulos articulares de las extremidades superiores, estos datos serán procesados por medio de una tarjeta de control y podrán ser almacenados en una PC.
Los datos recibidos y el movimiento de la extremidad serán mostrados en directo mediante una aplicación de escritorio de PC.

1.4. Justificación

Las enfermedades o lesiones de las extremidades de movimiento del cuerpo humano pueden ser causadas de manera directa por agentes de riesgo tales como movimientos bruscos, fricción y compresión continua, dichas lesiones pueden causar incapacidad temporal o permanente [4].

Existen herramientas muy precisas en el proceso de medición de variables, que brindan al fisioterapeuta, la posibilidad de trabajar con profesionales de diferentes áreas como la ingeniería, con la finalidad de diseñar herramientas como hardware y software que se puedan emplear en rehabilitación funcional de personas [5].

La contribución principal de esta investigación radica en la construcción de un dispositivo constituido por sensores inerciales que permitan representar el movimiento del brazo con los respectivos ángulos articulares. Los datos obtenidos deberán ser válidos y fiables sobre el movimiento del brazo y servirán al profesional en terapia física a establecer las condiciones de rehabilitación que deberá cumplir una persona.

En la Universidad Técnica del Norte por parte de la carrera de Ingeniería en Mecatrónica se ha venido trabajando previamente en el tema y se han utilizado sensores inerciales para monitorear la ergonomía de la muñeca y para medir ángulos en la rodilla. Por lo que este proyecto pretende seguir aportando en la línea de investigación.

1.5. Antecedentes

David Pozo muestra en su investigación [6] el análisis de los movimientos de cabeceo y alabeo mediante sensores inerciales para luego realizar una simulación de movimiento a través de servomotores cuya función es emular los movimientos realizados por el usuario, para lograr esta investigación el autor utilizó el filtro de Kalman como principal característica para controlar los datos y realizar la programación en un microcontrolador DSPIC30F4011 y como último paso para la interacción con el usuario se realiza la presentación de los resultados en una pantalla LCD ubicada en la parte exterior del módulo didáctico que se muestra en la figura 1.

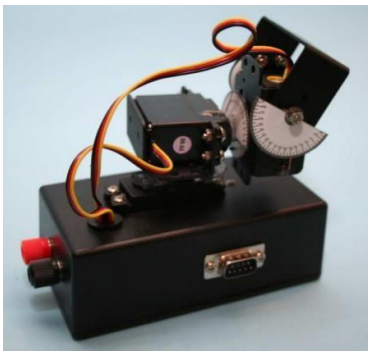


Fig. 1. Plataforma fija del módulo didáctico. [6]

Por otro lado, Cuesta [7] realiza una investigación para determinar la capacidad de obtención de datos de la cinemática cervical de un paciente mediante sensores inerciales,el estudio se inició poniendo un sensor inercial sobre el hueso de la cabeza y se realizaronvarios movimientos repetitivos obteniendo las características con un mínimo de error, como resultado se presenta que los sensores inerciales tienen una alta capacidad de obtención de datos con una alta fiabilidad.

Cristina Roldán realizó el estudio cinemático [8] de los miembros superiores e inferiores mediante el uso de sensores inerciales y los comparó con los datos de la población general, de esta forma se puede obtener diferencias entre un paciente y una persona sana,facilitando la obtención del diagnóstico, el proceso consiste en adherir el dispositivo a lapiel del usuario sea superior o inferior en el caso de este estudio e indicarle al usuario unos movimientos específicos como se puede observar en la figura 2 para realizar la recopilación de datos.

Fig. 2. Movimiento articular superior. [8]

José Jarrin [9], en su trabajo de grado describe el desarrollo de un sistema diseñado para obtener los ángulos articulares de miembro superior mostrado en la figura 3, con la utilización de una cámara web y un servidor para la recolección y procesamiento de la información. El hardware implementado consistía en una CPU para el tratamiento de imágenes, con conexión USB entre la cámara web y el servidor, así como un disco duro para el almacenamiento de datos. Para el software se usó bibliotecas de visión artificial OpenCV, bibliotecas de aprendizaje automático, software de base de datos y el uso de una interfaz para el usuario.

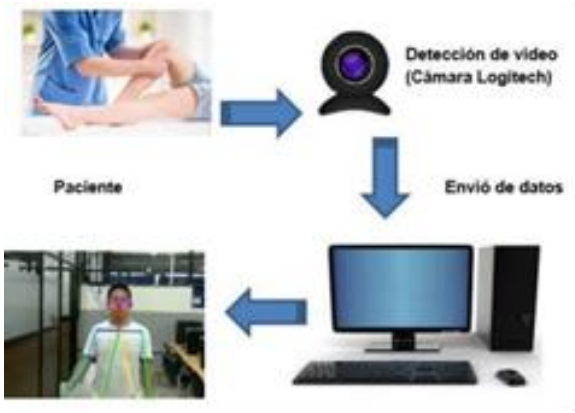


Fig. 3. Arquitectura del funcionamiento del sistema. [9]

Alejandro López [10], presenta en su trabajo de titulación una interfaz para la medición de ángulos de las extremidades inferiores por medio de sensores inerciales de la serie MPU6050, con el procesamiento de datos por medio de un controlador *Arduino Nano* y el software para la interfaz denominado TMOVE, obteniendo resultados no óptimos, pero dentro de los rangos propuestos, concluyendo que al realizar un ajuste a su sistema se podría obtener datos precisos y se los podría aplicar en el área médica.

Fig. 4. Módulo Inercial. [10]

CAPÍTULO II

MARCO TEÓRICO Y REFERENCIAL

2.1. Miembros Articulares superiores

La unión que forman dos o más huesos que permiten la movilidad de una de las extremidades se conoce como articulación. Las articulaciones son controladas por los ligamentos y la función de movilidad la cumple el músculo [11].

El esqueleto del miembro articular superior está compuesto de brazo, antebrazo, mano y dedos. Las articulaciones originan movimiento desde el triángulo formado en la clavícula como se muestra en la figura 5 [12].

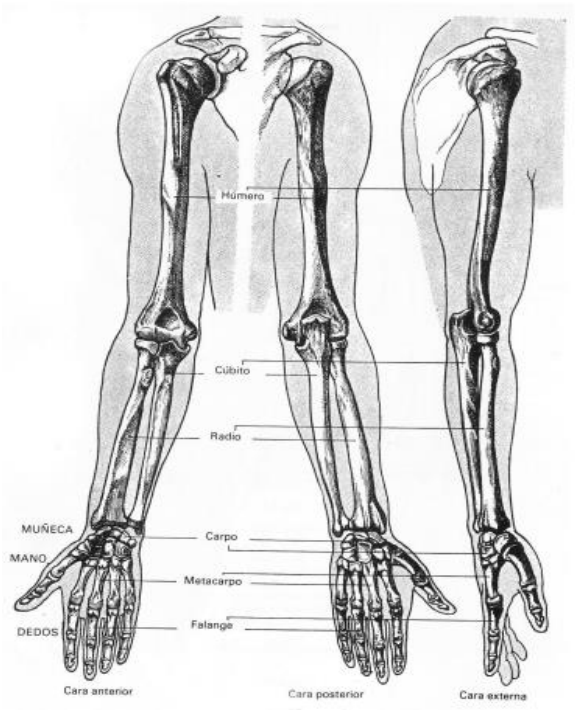


Fig. 5. Miembro articular superior [12].

2.1.1. Anatomía miembro superior

La extremidad superior como se muestra en la figura 6 está formada por: la cintura escapular, el brazo, el antebrazo y la mano. La conexión entre el tronco humano y el miembro superior se contacta por medio de la cintura escapular la cual está formada por la clavícula y la escapula. El húmero es el único hueso del brazo, el antebrazo está formado por cubito y el radio los cuales forman parte de las articulaciones del codo y la muñeca [13].

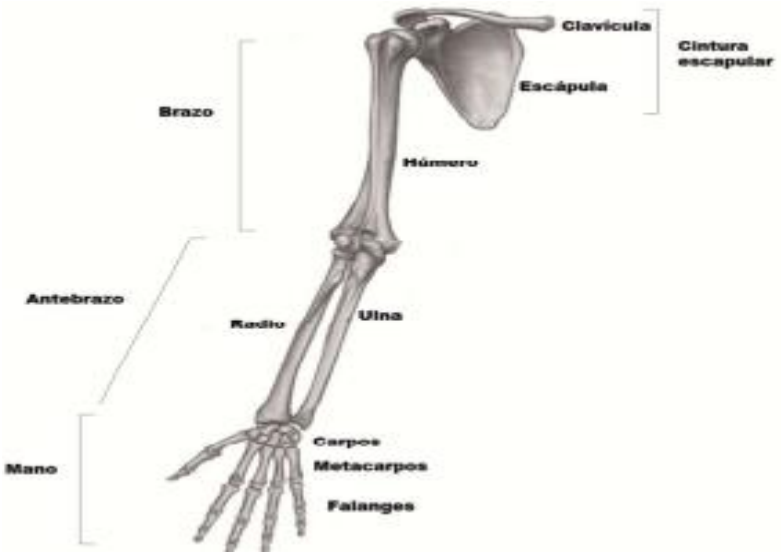


Fig. 6. Anatomía de la extremidad superior con sus diferentes regiones anatómicas.

2.1.2. Ángulos de movilización de las extremidades superiores

Los diferentes miembros del cuerpo humano forman ángulos articulares los cuales pueden ser medidos indistintamente con relación a la postura del individuo, ilustrado en la figura 7. Las mediciones se pueden realizar mediante transportadores de ángulos mediante dispositivos que permitan la toma de datos angulares, también es posible medir ángulos por medio de fotografías, realizando ensayos con un número suficiente de tomas desde diferentes puntos de vista.

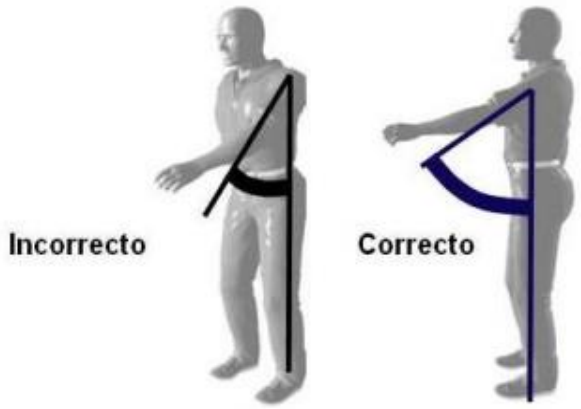


Fig. 7. Medición de Angulo de forma incorrecta frente a una medición correcta.

A. Ángulos del brazo

El ángulo por medir se forma con respecto al eje del tronco, se medirá en 4 etapas las cuales el paciente tendrá el tronco extendido como se muestra en la figura 8.

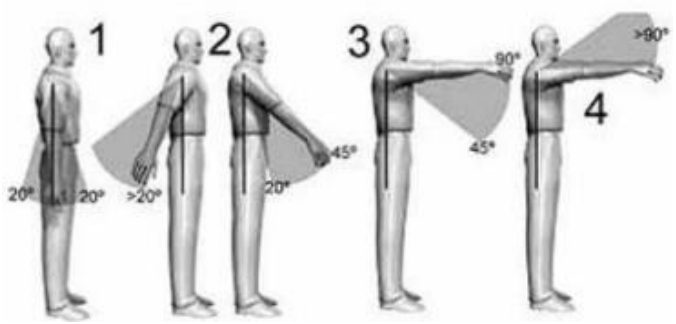


Fig. 8. Posiciones del brazo.

Es importante recordar que si el tronco esta flexionado los ángulos deben medirse desde el eje del tronco como se muestra en la figura 9 y la puntuación del brazo se indica en la tabla 1.

Tabla 1. Puntuaciones del brazo.

Puntos	Posición
1	Desde 20° de extensión hasta 20° de flexión
2	Extensión>20° o flexión entre 20° y 45°
3	Flexión entre 45° y 90°
4	Flexión >90°

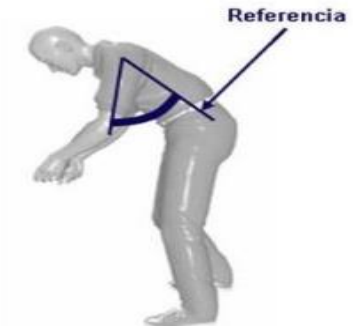


Fig. 9. Eje de referencia para la medición del ángulo de los brazos.

Existen modificaciones sobre las puntuaciones del brazo la cual se observa en la figura 10, la puntuación extra se observa en la tabla 2.

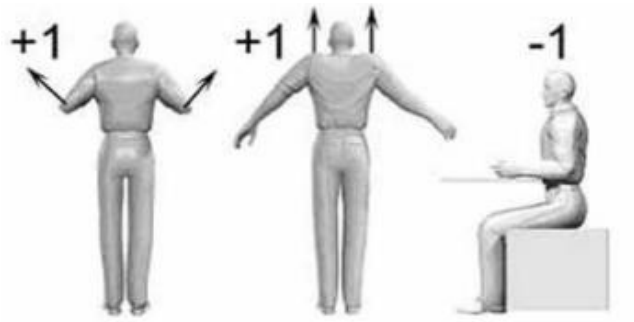


Fig. 10. Posiciones que modifican la puntuación del brazo.

*Tabla 2. Modificaciones de la puntuación del brazo***.**

Puntos	Posición
+1	Si los brazos están abducidos
+1	Si el hombro esta elevado
-1	Si el brazo tiene un punto de apoyo

B. Ángulos del antebrazo

La medición se realizará en dos pasos como se muestra en la figura 11 y la puntuación del antebrazo se indica en la tabla 3.

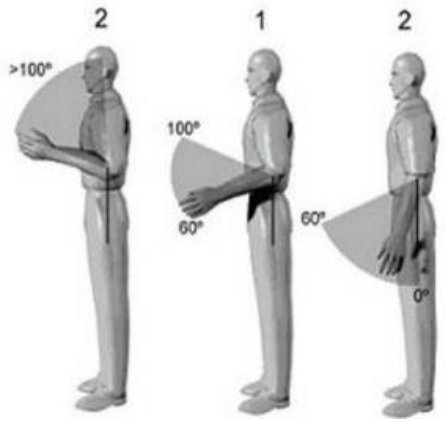


Fig. 11. Posiciones del antebrazo.

Tabla 3. Puntuaciones del antebrazo.

Puntos	Posición
1	Flexión entre 60° y 100°
2	Flexión < 60°o > 100°

Existen modificaciones sobre las puntuaciones del antebrazo la cual se observa en la figura 12, la puntuación extra se observa en la tabla 4.

Fig. 12. Posiciones que modifican la puntuación del antebrazo.

Tabla 4. Modificación de la puntuación del antebrazo.

Puntos	**Posición**
+1	Si la proyección vertical del antebrazo se encuentra más allá de la proyección vertical del codo.
+1	Si el antebrazo cruza la línea central del cuerpo.

C. Ángulos de la muñeca

Para la medición de ángulos de muñeca se analizará la posición de la muñeca. Se determinará el grado de flexión como se muestra en la figura 13 y las puntuaciones se indican en la tabla 5.

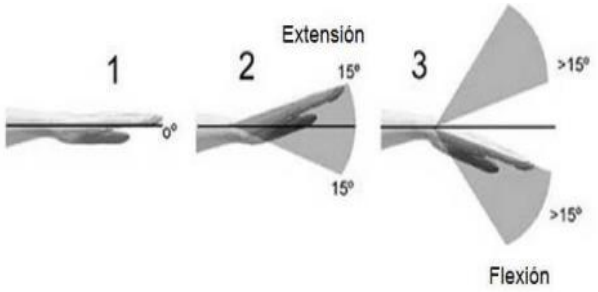


Fig. 13. Posiciones de la muñeca.

Tabla 5. Puntuación de la muñeca.

Puntos	**Posición**
1	Si está en posición neutra respecto a flexión
2	Si esta flexionada o extendida entre 0° y 15°
3	Para flexión o extensión mayor de 15°

Existen modificaciones sobre las puntuaciones de la muñeca la cual se observa en la figura 14, la puntuación extra se observa en la tabla 6.

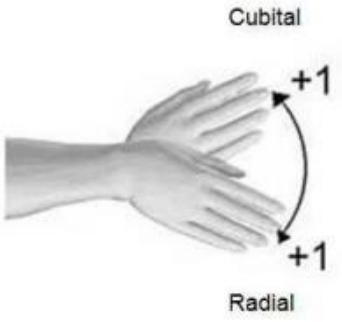


Fig. 14. Posiciones que modifican la puntuación de la muñeca.

Tabla 6. Puntuación de la desviación de la muñeca.

Puntos	Posición
+1	Si está desviada radial o cubitalmente

2.2. Rehabilitación para los ángulos articulares

La presencia de enfermedades articulares que se van originando con el paso del tiempo es común, al igual que las causas por las cuales se generan tales como la vejez, accidentes laborales entre otros. Es necesario un examen médico para valorar el estado de la enfermedad donde el experto de salud realiza varios exámenes al paciente de los cuales hace parte un examen físico donde se determinar el grado de afectación tomando en cuenta el reposo, movimiento activo y pasivo de cada articulación [14].

2.2.1. Problemas articulares frecuentes

Dentro de los problemas articulares que se producen en el miembro superior, están los que se describen a continuación:

A. Tendinitis del manguito de los rotadores

Es la principal enfermedad que se presenta al comienzo de las extremidades superiores y se debe a la inflamación de los tendones que rodean el hombro llamados manguito de rotadores, su presencia es la responsable de la disminución en la funcionalidad de esta articulación [15].

B. Epicondilitis

Es el daño de los tendones que causa un grado alto de dolor en el codo, es el resultado de movimientos repetitivos e intensos que debilitan y disminuyen la movilidad de la articulación [16].

C. Síndrome de túnel carpiano

El principal trastorno que se presenta en nivel de la muñeca es el síndrome del túnel carpiano, el cual se origina por una presión excesiva en el nervio mediano en la articulación, el cual puede llegar a ser el causante de la parálisis parcial de la mano [17].

Los problemas articulares se los puede determinar con la ayuda de instrumentos manuales que los profesionales de terapia física emplean, como es el caso del goniómetro para los miembros articulares superiores.

2.2.2. Goniómetro

El goniómetro ilustrado en la figura 15 es un instrumento en forma de semicírculo que sirve para la medición de ángulos [18]. En el área de la rehabilitación médica se realiza la medición de ángulos entre los huesos tomando como punto cero la articulación en cuestión, con esta medición se puede definir si el paciente sufre algún tipo de trastorno musculoesquelético y encontrar la terapia apropiada [19].

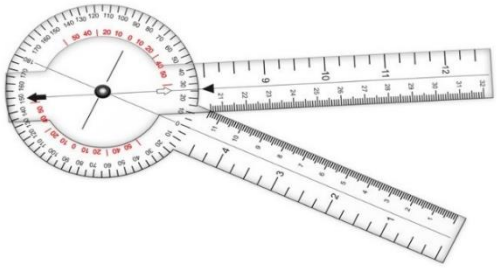


Fig. 15. Goniómetro empleado en procesos de rehabilitación.

2.2.3. Goniometría del miembro superior

Con el fin de tener un método de medición clínica simple y estandarizado se presentó el cero neutro con datos recopilados por la Asociación para el Estudio de la Osteosíntesis (AO), siendo el método de la medición de ángulos más objetiva y de mayor fiabilidad que las radiografías [20].

En la tabla 7 se muestra un resumen de los datos recopilados en la investigación antes mencionada para la articulación del hombro.

Tabla 7. Goniometría de la articulación escapulohumeral. [19]

Goniometría de la articulación escapulohumeral			
Movimiento	**Posición de Inicio**	**Ángulo de inicio**	**Ángulos de fin normales**
Abducción	Decúbito dorsal	hombro=0, codo=0, antebrazo=0, muñeca=0	160°/180°
Aducción	Decúbito dorsal	hombro=0, codo=0, antebrazo=0, muñeca=0	30°
Flexión	Decúbito dorsal	hombro=0, codo=0, antebrazo=0, muñeca=0	150°/170°
Extensión	Decúbito ventral	hombro=0, codo=0, antebrazo=0, muñeca=0	40°
Rotación externa - interna	Decúbito dorsal	hombro=90, codo=90, antebrazo=0, muñeca=0	90° 70°

En la tabla 8 se encuentran los datos comunes para el movimiento de la articulación del codo y en la tabla 9 de la muñeca.

Tabla 8. Goniometría del codo.

Goniometría del codo			
Movimiento	**Posición de Inicio**	**Ángulo de inicio**	**Ángulos de fin normales**
Flexión	Decúbito dorsal	hombro=0, codo=0, antebrazo=0, muñeca=0	150°
Extensión	Decúbito dorsal	hombro=0, codo=0, antebrazo=0, muñeca=0	10°
Supinación	Paciente Sentado	hombro=0, codo=90, antebrazo=0, muñeca=0	90°
Pronación	Paciente Sentado	hombro=0, codo=90, antebrazo=0, muñeca=0	90°
Goniometría de la muñeca			
Movimiento	**Posición de Inicio**	**Ángulo de inicio**	**Ángulos de fin normales**
Flexión	Paciente sentado	codo=90, muñeca=0	60°
Extensión	Paciente sentado	codo=90, muñeca=0	60°
Desviación radial	Paciente sentado	codo=90, muñeca=0	30°
Desviación cubital	Paciente sentado	codo=90, muñeca=0	30°

2.3. Sensores Inerciales

Los sensores inerciales o unidades de medida inercial son sistemas importantes desarrollados para el diagnóstico y la monitorización de pacientes que sufrieron lesiones en uno de sus miembros ya sea superior o de pacientes que sufren enfermedades como el Parkinson.

Los IMUs (*Inertial Measurement Units*) miden la función motora de forma sencilla y que sea fácil de trasmitir los datos. Están formados por un acelerómetro y un giroscopio triaxial [21].

2.3.1. ADXL345

El ADXL345 mostrada en la figura 16 es un acelerómetro micro mecanizado capacitivo de 3 ejes que funcionan independientemente, puede ser conectado de manera directa a un autómata [22].

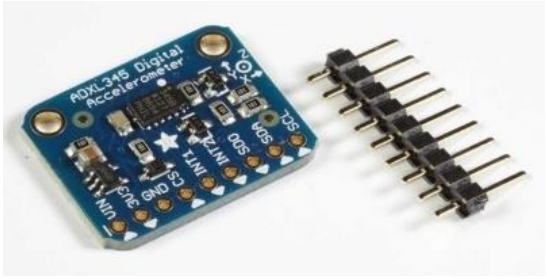


Fig. 16. Acelerómetro serie ADXL345.

2.3.2. HMC5883L

El HMC5883L es una brújula digital como se observa en la figura 17 la cual mide el valor del campo magnético en tres ejes. Hace posible la estimación de la orientación con respecto al campo magnético de la tierra [22].

Fig. 17. Giroscopio serie HMC5883L.

2.3.3. MPU-6050

El módulo MPU-6050 mostrado en la figura 18 es una unidad de medición inercial de 6 grados de libertad, en el cual están incluido el acelerómetro y un giroscopio [22].

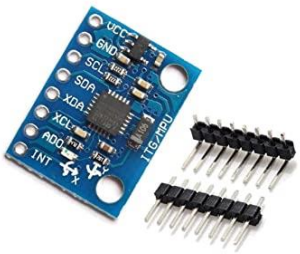


Fig. 18. Módulo inercial MPU-6050.

2.3.4. MPU-9250

El módulo MPU-9250 que se indica en la figura 19 es una unidad de medición inercial en la cual incluye un acelerómetro, giroscopio y magnetómetro [22].

Fig. 19. Módulo inercial MPU-9250.

2.4. Software de programación para sensores inerciales

Existen varias formas de programar sensores inerciales, como también diversos softwares que facilitan la interacción entre el usuario y el lenguaje de programación a emplea, es por eso por lo que a continuación se menciona alguno de ellos:

2.4.1. Matlab

Matlab es una herramienta *software* que permite la resolución de cálculos matemáticos a partir de valores matriciales, resulta ser un aliado si de programación se habla, debido a su entorno sencillo y el uso de lenguajes comunes como C o C++, gracias a su amplia gama de librerías se puede utilizar esta herramienta para la recepción de las señales de sensores para posteriormente realizar su análisis [23].

2.4.2. Arduino

Arduino es una placa electrónica *hardware* muy popular en el mercado por su fácil adquisición, su microcontrolador incorporado y una serie de puertos permiten la conexión entre distintos tipos de sensores para realizar la programación necesaria en tiempo real [24].

2.4.3. LabVIEW

LabVIEW es un entorno de programación gráfica amigable que se compone por dos ventanas: un panel frontal y un panel de bloques. *LabVIEW* permite la conexión de módulos a través del cual se registrará el ingreso de las señales de los sensores y con el panel de bloques realizar la programación de manera rápida e intuitiva [25].

2.4.4. Unreal Engine

Unreal Engine es un entorno abierto y gratuito para el desarrollo y lanzamiento de juegos o simulaciones, que gracias a su versatilidad se ha convertido en una herramienta clave en distintas profesiones, tiene dos tipos de programación, por bloques o en C++ lo cual lo convierte en el mejor *software* aliado [26].

2.5. Tarjetas de adquisición de datos

2.5.1. Arduino NANO

Arduino NANO es una plataforma electrónica gratuita, está compuesta por una serie de pines los cuales generan entradas y salidas de señales. En la figura 20 se muestra la tarjeta de adquisición Arduino Nano formado por microcontroladores CI que son capaces de realizar varias tareas al mismo tiempo grabadas en la memoria. Están constituidas de tres elementos fundamentales, la unidad central de proceso (CPU), la memoria y los periféricos de entrada y salida. [27]

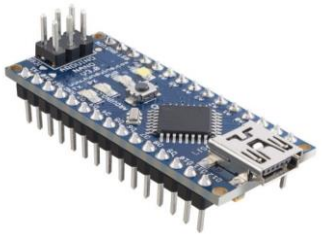


Fig. 20. Arduino Nano.

2.5.2. Raspberry PI

La placa *Raspberry PI* es un minicomputador comprendido de algunas partes, un procesador, una memoria RAM y un chip gráfico mostrados en la figura 21. Dispone de varios pines para entradas y salidas ya sean analógicas o digitales, también posee puertos USB, HDMI, puerto ethernet, ranura para tarjeta SD, salidas de video y de audio.

Fig. 21. Raspberry PI.

2.5.3. Módulo ESP32

El módulo ESP32 que se muestra en la figura 22, integra los servicios de Wi-fi, Bluetooth y procesamiento de datos, contiene varios periféricos que facilitan la conexión con interfaces externas [28].

Fig. 22. Módulo ESP32.

2.6. Protocolo de comunicación

Los protocolos de comunicación permiten que dos dispositivos se comuniquen de forma correcta en una red [29].

2.6.1. I2C

La comunicación I2C requiere dos señales para lograr la conexión: línea de datos (SDA) y el reloj (SCLK), esta conexión logra que el número de cables sea menor en caso de necesitar más de un esclavo [30].

2.6.2. SPI

La comunicación SPI necesita cuatro señales para lograr la conexión entre el máster y el esclavo: CS, SCK, MOSI Y MISO. En caso de necesitar más de un esclavo se añade una nueva señal “CS” por lo cual la cantidad de cables conectados aumentan [30].

2.6.3. UDP

El Protocolo de Datagrama de Usuario o también conocido como UDP es un sistema de conexión de datos que no necesita una conexión previa para realizar el envío de información desde un emisor a un receptor, simplemente se envía los datos a un puerto de destino dentro de la dirección IP especificada en la red [31].

CAPÍTULO III

METODOLOGÍA

3.1. Investigación

La información necesaria sobre la manera de realizar la medición de los ángulos de las extremidades superiores, las enfermedades más frecuentes debido a lesiones, así como los sensores inerciales y *software* de programación se obtiene a través de la consulta de varias fuentes científicas en la web con la finalidad de establecer un modelo conceptual para proponer la construcción de un dispositivo para la medición de ángulos de las extremidades superiores por medio de sensores inerciales, es importante recalcar que la investigación de diferentes métodos estuvo presente durante todo el desarrollo de este trabajo [32].

3.2. Análisis de requerimientos

El análisis de los requerimientos para la propuesta del sistema de medición de ángulos a través de sensores inerciales inició con la identificación de los puntos esenciales para el uso del dispositivo por parte de los pacientes, como el tamaño, seguridad y fácil limpieza esto requerido por una especialista en rehabilitación física. Se continuó con la identificación de los deseos, como es el bajo costo y una vida útil elevada para tener la posibilidad de aplicarlo en el servicio médico.

3.2.1. Requerimientos del sistema

El sistema debe cumplir con los requerimientos mostrados en la tabla 10, para desempeñar un funcionamiento adecuado en el cual en la parte izquierda se detallara si es un deseo o un requerimiento dependiendo las variables a emplear en el proyecto, en la derecha se observa las descripciones de cada una de ellas.

Tabla 10. Tabla de requerimientos para el sistema

	D/R	Descripción
Dimensión	R	Las dimensiones deben ser pequeñas para que no incomode al paciente.
Costo	D	El costo debe ser bajo.
Durabilidad	D	La vida útil debe ser alta.
Seguridad	R	Al estar en contacto con el paciente debe ser seguro eléctricamente.
Limpieza	R	Se debe permitir la fácil limpieza entre pacientes.

D = deseo R = requerimiento

3.2.2. Selección de tarjeta de adquisición de datos

Se utilizará el módulo Esp32 al ser compatible con el sensor inercial seleccionado y asegurar una fácil conexión para el envío de datos, en comparación con las demás alternativas se asegura reducir el costo y las dimensiones del sistema.

3.2.3. Selección de sensores inerciales

El sensor inercial MPU-9250 será el seleccionado para el sistema debido a que cuenta con giroscopio, acelerómetro y magnetómetro en el mismo módulo, ayudando a disminuir el costo, las dimensiones del sistema y a que la funcionalidad aumente.

3.2.4. Selección de protocolos de comunicación

Para realizar la comunicación entre los sensores inerciales y la tarjeta controladora se utilizará la comunicación UDP debido a la fácil conexión para el envío de la información, además de la característica de convertirlo a un dispositivo inalámbrico.

3.2.5. Voz del Cliente

La voz del cliente son los puntos más importantes, ya que se reciben directamente del cliente el cual hará uso del sistema y marcará la diferencia en el mercado al cubrir necesidades visibles.

3.2.6. Voz del Ingeniero

La voz del ingeniero es la traducción técnica de todos los puntos mostrados por el cliente además de incluir nuevos requerimientos o deseos que el diseñador crea necesarios para mejorar el sistema y de esta manera lograr eficiencia, una mejor consolidación y un correcto funcionamiento.

3.3. Diagramas Funcionales

Los diagramas funcionales ayudan de manera gráfica a representar y consolidar las ideas para establecer los pasos que estarán presentes en el funcionamiento final, así se visualizan puntos importantes y se clasifican los sistemas de procesamiento. Se tiene los niveles cero, uno y dos.

- El nivel cero representa el funcionamiento básico del sistema.
- En el nivel uno se especifican los sistemas de procesamiento necesarios de manera general.
- Al finalizar se obtiene el nivel dos, en el que se detallan todos los pasos que serán necesarios para consolidar el funcionamiento total del sistema.

En la figura 23 se muestra el diagrama funcional de nivel cero que representa el trabajo principal del dispositivo para la medición de ángulos de las extremidades

superiores que inicia con el ingreso de energía eléctrica y la recepción de las señales enviadas por los sensores y termina con la representación gráfica de la información.

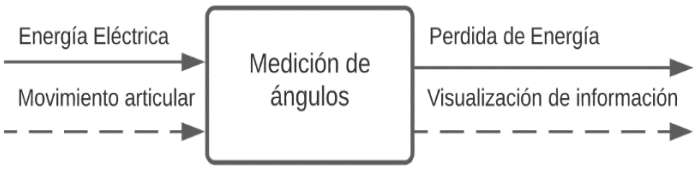


Fig. 23. Diagrama funcional de nivel cero.

El nivel uno de los diagramas funcionales muestra los módulos que componen la estructura principal del sistema para la medición análisis y visualización de la información, tal como se muestra en la figura 24.

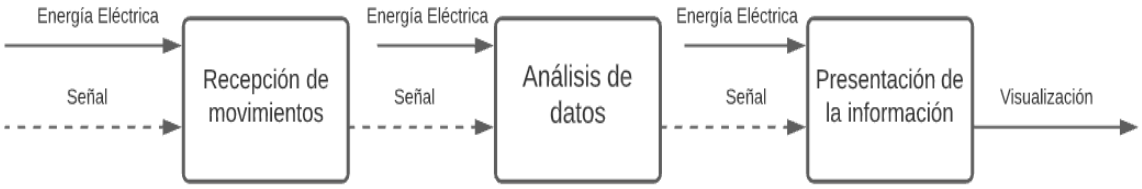


Fig. 24. Diagrama funcional de nivel uno.

Todos los pasos para el correcto funcionamiento del sistema de medición de ángulos se encuentran detallados en la figura 25 que muestra el nivel dos de los diagramas funcionales.

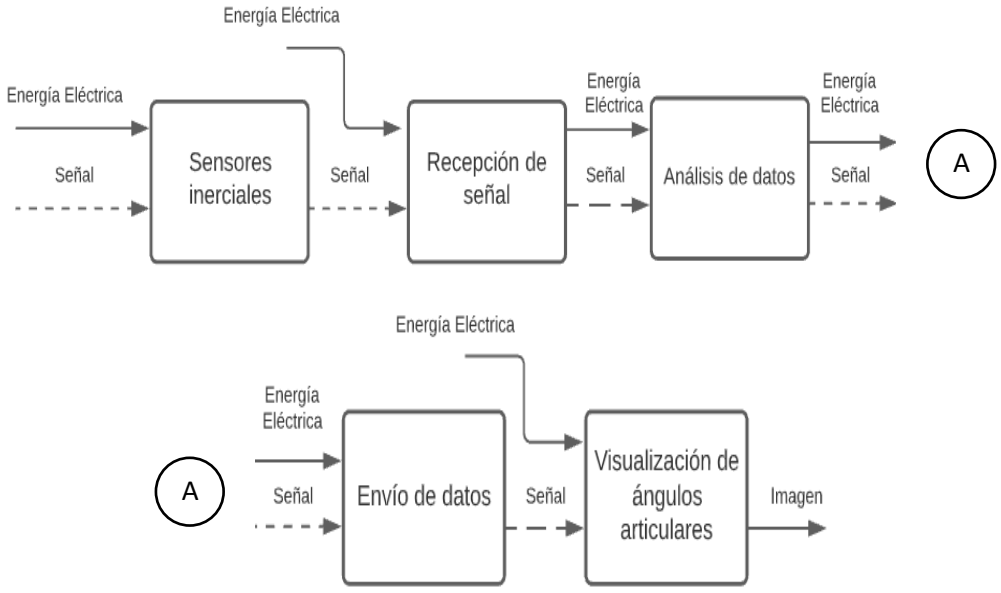


Fig. 25. Diagrama funcional de nivel dos.

3.3.1. Diagrama de flujo

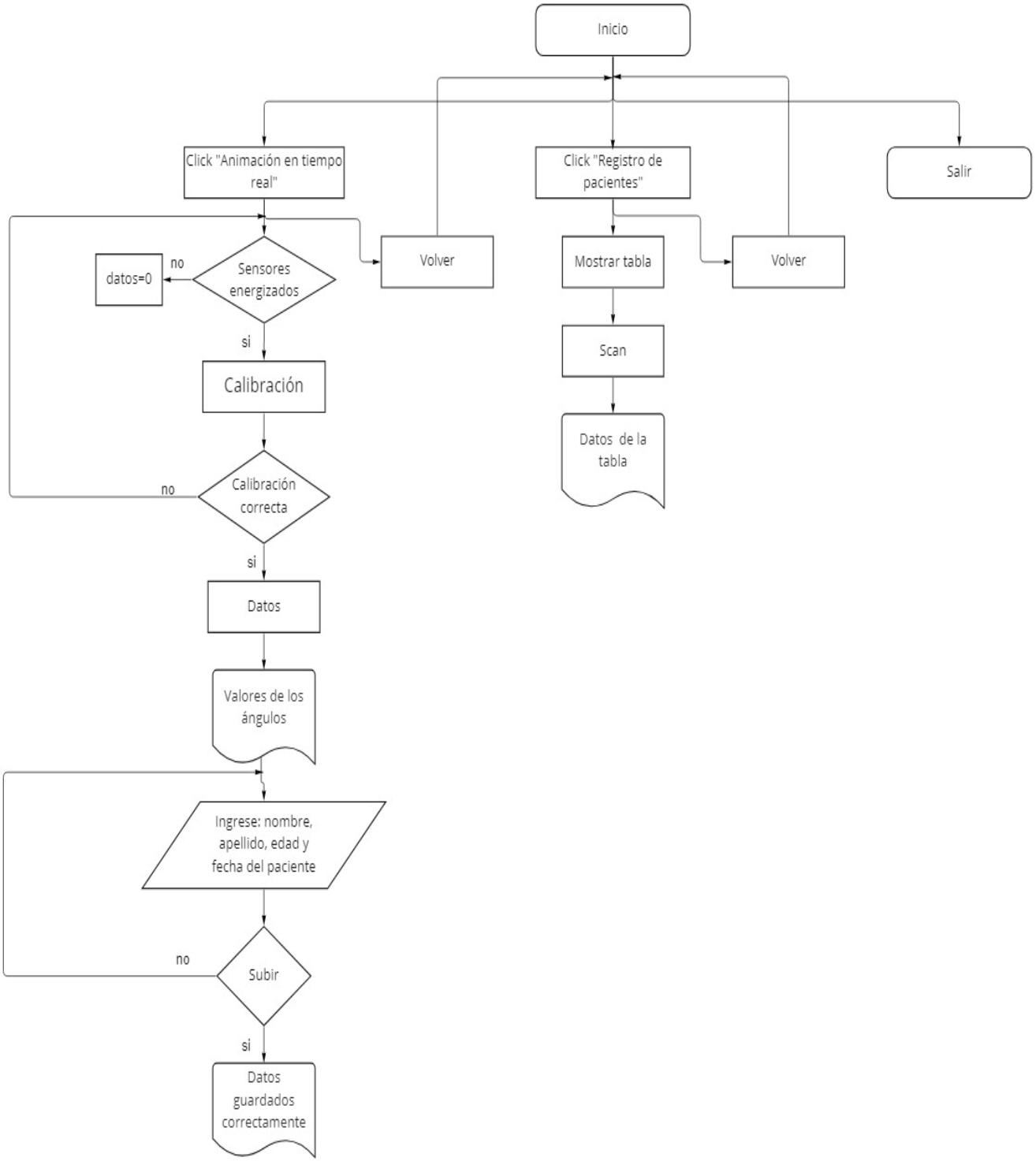


Fig. 26. Diagrama de flujo.

3.4. Módulos

La tabla 11 muestra el detalle de funcionamiento para cada módulo propuesto.

Tabla 11. Detalle del funcionamiento de los módulos.

N°	Módulo	Detalle
1	Recepción de señal	Es el módulo encargado de la recepción de los movimientos articulares a través de sensores inerciales.
2	Análisis de datos	Se realiza la recepción de las señales y el análisis de la información.
3	Presentación de la información	Los datos analizados se presentarán a través de una interfaz en tiempo real.

3.5. Cálculo de Ángulos

Para el cálculo de los ángulos articulares se hará referencia al método tradicional de descomposición de rotaciones alrededor de 3 ejes cartesianos, usando algunos sistemas como son los giroscopios. Los ángulos de Euler para que sean entendibles se pueden representar como cuaterniones y las rotaciones de objetos en tres dimensiones. Tienen componentes i,j,k que representan los ejes alrededor del cual se producirá una rotación. También tiene componente q0 o eje referencial el cual representará la cantidad de rotación que se producirá alrededor del mismo.

3.5.1. Ángulos de Euler

Euler probó que para especificar la rotación de un sistema de coordenadas en el espacio se necesitan 3 parámetros reales, y que toda rotación en R3 puede obtenerse como una composición de tres rotaciones elementales consecutivas: se selecciona el orden de los ejes coordenados y se efectúa la rotación alrededor de ese eje de forma consecutiva. Los tres ángulos correspondientes forman los ángulos de Euler representados en la figura 27.

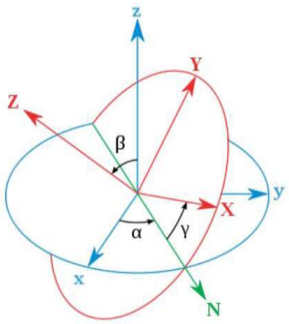


Fig. 27. Representación gráfica de los ángulos de Euler.

3.5.2. Cuaterniones

Para la presente investigación se decidió que la forma más simple de representar el movimiento de las extremidades superiores mediante la plataforma Unreal Engine es con la utilización de código de programación de cuaterniones los cuales son números hipercomplejos de cuatro componentes. Se los puede denotar de la forma $q = w + x.i + y.j + z.k$ con (x, y, z) vector en el espacio, w parte real y (i, j, k) parte imaginaria. Los cuaterniones tienen una notación compacta y son muy fáciles de componer. Las operaciones usando cuaterniones son muy eficientes en comparación con las que usan matrices, ya que requieren menor cantidad de operaciones básicas y menor espacio de almacenamiento. Conforman una representación no 40 singular y solucionan el problema de "pérdida de dimensionalidad". Gracias a todo esto se usa en aplicaciones que requieren rotaciones o ubicaciones en el espacio bajo demanda en tiempo real.

3.5.3. Comandos de programación de Unreal Engine para el uso de cuaterniones

En la tabla 12 se muestran los comandos de programación que se pueden tomar en cuenta para la programación de la aplicación realizada en el *software Unreal Engine* para la movilización en tiempo real de las extremidades superiores.

Tabla 12. Lista de comandos para programar cuaterniones.

Comando	Descripción
Ángulo (Quat)	Obtenga el ángulo de este cuaternión
Distancia angular (Quat)	Encuentre la distancia/diferencia angular entre dos cuaterniones de rotación.
Eje X (Quat)	Obtenga la dirección de avance (eje X) después de que este Cuaternión lo haya rotado.
Axis Y (Quat)	Obtenga la dirección correcta (eje Y) después de que este Cuaternión lo haya rotado.
Eje Z (Quat)	Obtenga la dirección hacia arriba (eje Z) después de que este Cuaternión lo haya rotado.
Asegure el arco más corto a (Quat)	Modifique el cuaternión para asegurarse de que el delta entre él y B represente el ángulo de rotación más corto posible.

Igual (Quat)	Devuelve verdadero si Cuaternión A es igual a Cuaternión B (A == B) dentro de una tolerancia de error especificada
Euler (Quat)	Convierta un cuaternión en ángulos de Euler de coma flotante (en grados).
Invertida (Quat)	Devuelve una copia invertida de este cuaternión.
Es Identidad (Quat)	Comprueba si este cuaternión es un cuaternión de identidad. Supone que Cuaternión probado está normalizado.
Es no finito (Quat)	Determine si hay valores no finitos en este Quat.
Está Normalizado (Quat)	Devuelve verdadero si este cuaternión está normalizado
Registro (Quat)	Cuaternión con W=0 y V=theta*v. Usado en combinación con Exp ().
Hacer de Euler (Quat)	Convierta un vector de ángulos de Euler de coma flotante (en grados) en un cuaternión.
Normalizar (Quat)	Normalice este cuaternión si es lo suficientemente grande en comparación con la tolerancia suministrada. Si es demasiado pequeño, configúrelo en el cuaternión de identidad.
Normalizado (Quat)	Obtenga una copia normalizada de este cuaternión. Si es demasiado pequeño, devuelve un cuaternión de identidad.
No es igual (Quat)	Devuelve verdadero si Quat A no es igual a Quat B (A! = B) dentro de una tolerancia de error especificada
Quat - Quat	Devuelve la resta del Vector B del Vector A (A - B)

cuaternario * cuaternario	Obtiene el resultado de multiplicar dos cuaterniones (A * B). El orden importa al componer cuaterniones: $C = A * B$ producirá un cuaternión C que lógicamente aplica primero B y luego A a cualquier transformación posterior (primero a la derecha, luego a la izquierda).
cuaternario + cuaternario	Devuelve la suma del Vector A y el Vector B (A + B)
Identidad cuaternaria	Constante de cuaternión de identidad El objetivo es la biblioteca de matemáticas de Kismet
Girar vector (Quat)	Gira un vector por este cuaternión. El objetivo es la biblioteca de matemáticas de Kismet
Eje de rotación (Quat)	Obtenga el eje de rotación del Cuaternión. Este es el eje alrededor del cual se produce la rotación para transformar el sistema de coordenadas canónicas en la orientación del objetivo. Para la identidad Cuaternión que no tiene tal rotación, se devuelve FVector (1,0,0). El objetivo es la biblioteca de matemáticas de Kismet
Siete Componentes (Quad)	Establezca los componentes X, Y, Z, W de Cuaternión.
Juego de Euler (Quat)	Convierta un vector de ángulos de Euler de coma flotante (en grados) en un cuaternión.
Tamaño (Quat)	Obtenga la longitud del cuaternión.
Tamaño al cuadrado (Quat)	Obtenga la longitud al cuadrado del cuaternión.
Unrotate Vector (Quat)	Girar un vector por el inverso de este cuaternión.

Avance vectorial (Quat)	Obtenga la dirección de avance (eje X) después de que este Cuaternión lo haya rotado.
Vector derecho (Quat)	Obtenga la dirección correcta (eje Y) después de que este Cuaternión lo haya rotado.
Vector arriba (Quat)	Obtenga la dirección hacia arriba (eje Z) después de que este Cuaternión lo haya rotado.

3.6. Transformación de cuaternión a ángulos de rotación

Para el movimiento de las extremidades superiores que se observa en pantalla se emplea los cuaterniones, por otro lado, para que el usuario pueda interpretar estos datos, los cuaterniones serán transformados en ángulos rotacionales, que giran alrededor de los ejes: x, y, z. La transformación se la realizó con la fórmula 1 [33]:

$$q_x = \sin(\frac{\alpha}{2}) \quad (1)$$

Donde:
qx: es el cuaternión generado alrededor del eje x.
α: es el ángulo de rotación alrededor del eje x. [°]

Para despejar el ángulo nos apoyaremos de la identidad trigonométrica mostrada en la fórmula 2.

$$\sin(\frac{\alpha}{2}) = \pm\sqrt{\frac{1-cos\alpha}{2}} \quad (2)$$

Remplazamos (2) en (1):

$$q_x = \pm\sqrt{\frac{1-cos\alpha}{2}}$$

Despejamos el ángulo α:

$$\alpha = \cos^{-1} 1 - 2(q_x)^2 \quad (3)$$

Para encontrar los ángulos rotacionales alrededor de los ejes "*y*" y "*z*", se emplea la fórmula 3, remplazando el valor de "*qx*" por "*qy*" o "*qz*" respectivamente.

3.7. Diseño de la interfaz Gráfica

La aplicación debe constar de una interfaz amigable para sus usuarios, mostrando datos claros y con una manipulación sencilla, es por tanto que se ha realizado una aplicación bajo la norma ISO 9241.

3.7.1. Norma ISO 9241

Esta norma internacional trabaja en la ergonomía de la presentación de información en pantallas de visualización de datos, características y propiedades que deben tener interfaces y graficas [34].

A. Visualización de elementos

La visualización de información en pantallas de visualización de datos PVD debe estar bien definida con las dimensiones correctas y disponer del espacio suficiente en la organización de elementos que esta contenga.

En el tema de caracteres alfanuméricos propone:

- La matriz que representa a los caracteres debe tener una mínima medida de 5*7 pixeles, cabe recalcar si existe una lectura frecuente se recomienda tener un mínimo de 7*9 pixeles.
- El tamaño de los caracteres alfanuméricos está basado en la distancia de la visión y debe tener un mínimo de 22 minutos de arco, con el caso de que la distancia no sea menos a los 400 mm.
- El ancho entre caracteres debe ser igual a la anchura de trazo; el espacio entre palabras debe ser mínimo a un caracter y la distancia entre líneas debe tener un píxel.
- La imagen debe tener estabilidad espacial; en el caso de un punto, el movimiento de oscilación debe de ser menor al 0,02 por cien de la distancia nominal de la visión.
- En el ámbito de luminancia de la pantalla y contraste de caracteres, la pantalla debe tener un mínimo de luminancia de 35 Cd/m^2, aparte de estas características existen dos formas admisibles de representar caracteres alfanuméricos en las PDV:
- Polaridad Positiva. - se identifica por caracteres oscuros en fondo claro, los reflejos son menos perceptibles y se recomienda para personas de mayor agudeza visual.
- Polaridad Negativa. - Se tiene a caracteres claros en fondo oscuro, el parpadeo es menos y se recomienda para personas de menor agudeza visual.

CAPÍTULO IV

RESULTADOS

Para la estructura del dispositivo para la medición de ángulos de las extremidades superiores a continuación se selecciona la mejor alternativa para cada parte que lo conformará:

4.1. Desarrollo del hardware

El modelo del dispositivo se concentrará en las tres articulaciones del brazo superior siendo muñeca, codo y hombro, para asegurar la adquisición de información según la goniometría del miembro superior, los sensores se ubicarán en una carcasa asegurada por correas al paciente, las cuales cumplen con los requerimientos seleccionados: fácil limpieza entre pacientes, mayor durabilidad y bajo costo.

4.2.Conexión del sistema

En la figura 28 se muestra el esquema de la conexión entre el módulo Esp32 y los 3 sensores inerciales MPU-9265 que se comunicarán mediante el protocolo UDP, este sistema se comunicará a través de conexión Wifi al computador para la recepción de información.

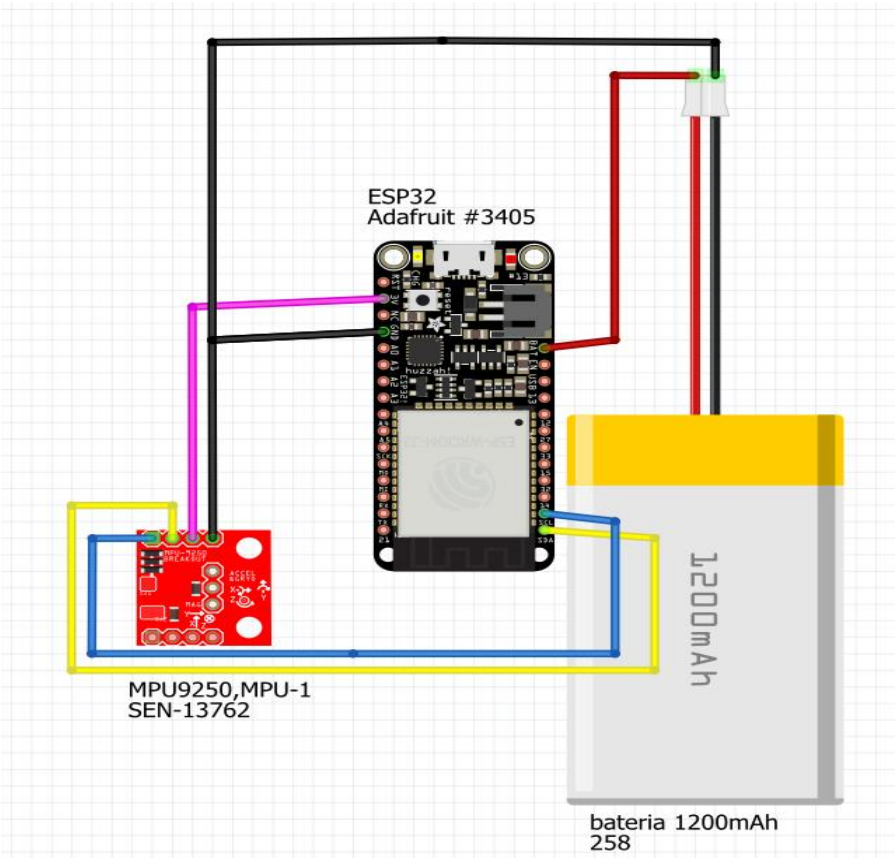


Fig. 28. Esquema de conexión.

Donde la conexión del circuito esta distribuido de la siguiente manera:

La batería proporciona un voltaje de 3.7 a 1500 mAh, el cable rojo representa el polo positivo, el cable negro representa GND o tierra.

MPU9250 -- ESP32

3.3V ------ 3.3V Cable rosa

SDA --------- 21 Cable amarillo

SCL --------- 22 Cable azul

GND -------- GND Cable negro

4.3. Desarrollo de la aplicación

La aplicación de escritorio se realizó mediate la herramienta *Unreal Engine* de la plataforma *Epic Games*, en la cual mediante una interfaz amigable con el usuario es capaz de recolectar datos, procesarlos y transmitirlos en tiempo real.
Para la construcción de la aplicación se requiere del *software* Arduino, *Visual Studio* y *Unreal Engine*, la cual requiere del siguiente proceso:
El código de programación realizado en Arduino establece la configuración necesaria para que el módulo ESP 32 pueda conectarse con una red de internet local.

```
#include <Wire.h>
#include "I2Cdev.h"
#include "MPU9250_9Axis.h"
#include "EEPROM.h"
#include <WiFi.h>
#include "AsyncUDP.h"

#define EEPROMUtils
#define SettingsUtils
#define SerialPortHelper
#define WiFiHelper
#define UDPClient
#define ProtocolHelper
#define DataUtils
```

Fig. 29. Librerías necesarias para programar el módulo ESP32.

En la figura 29 se muestran las librerías necesarias para la correcta compilación de la programación.

GyroSensor_MPU9250_ESP32 - ProtocolHelper.ino | Arduino 1.8.19 (Windows Store 1.8.57.0)

Archivo Editar Programa Herramientas Ayuda

GyroSensor_MPU9250_ESP32 | DataUtils | EEPROMUtils | ProtocolHelper | SerialPortHelper | SettingsUtils | UDPClient | WiFiHelper

```
#include "ArduinoJson.h"

String GetStatusJSONString(int wifiStatus, String localIP)
{
    StaticJsonDocument<1024> staticJsonDocument;
    staticJsonDocument["type"] = "getStatusResult";
    staticJsonDocument["wifi_status"] = wifiStatus;
    staticJsonDocument["local_ip"] = localIP;

    char docBuf[1024];
    serializeJson(staticJsonDocument, docBuf);

    return String(docBuf);
}

String GetGyroDataJSONString(String sensorName, float qX, float qY, float qZ, float qW)
{
    StaticJsonDocument<1024> staticJsonDocument;
    staticJsonDocument["type"] = "gyroData";
    staticJsonDocument["sensorName"] = sensorName;
    staticJsonDocument["qX"] = String(qX, 2);
    staticJsonDocument["qY"] = String(qY, 2);
    staticJsonDocument["qZ"] = String(qZ, 2);
    staticJsonDocument["qW"] = String(qW, 2);

    char docBuf[1024];
    serializeJson(staticJsonDocument, docBuf);

    return String(docBuf);
}
```

Fig. 30. Protocolo de comunicación y parámetros de los sensores inerciales.

GyroSensor_MPU9250_ESP32 - SettingsUtils.ino | Arduino 1.8.19 (Windows Store 1.8.57.0)

Archivo Editar Programa Herramientas Ayuda

GyroSensor_MPU9250_ESP32 | DataUtils | EEPROMUtils | ProtocolHelper | SerialPortHelper | SettingsUtils | UDPClient | WiFiHelper

```
void InitSettings()
{
    if (CheckIsEmptyEEPROM(0)) WriteStringEEPROM(0, "wifi_ssid");
    if (CheckIsEmptyEEPROM(1)) WriteStringEEPROM(1, "wifi_pass");
    if (CheckIsEmptyEEPROM(2)) WriteStringEEPROM(2, "GyroSensor00");
    if (CheckIsEmptyEEPROM(3)) WriteStringEEPROM(3, "127.0.0.1");
    if (CheckIsEmptyEEPROM(4)) WriteStringEEPROM(4, "7755");
}

String ResetSettingsJSONString()
{
    WriteStringEEPROM(0, "wifi_ssid");
    WriteStringEEPROM(1, "wifi_pass");
    WriteStringEEPROM(2, "GyroSensor00");
    WriteStringEEPROM(3, "127.0.0.1");
    WriteStringEEPROM(4, "7755");

    delay(100);

    StaticJsonDocument<1024> staticJsonDocument;
    char docBuf[1024];
    staticJsonDocument["type"] = "resetSettingsResult";
    staticJsonDocument["error"] = 0;

    serializeJson(staticJsonDocument, docBuf);

    return String(docBuf);
}

String LoadSettingsJSONString()
{
    StaticJsonDocument<1024> staticJsonDocument;
    staticJsonDocument["type"] = "loadSettingsResult";
    staticJsonDocument["wifi_ssid"] = ReadStringEEPROM(0);
```

Fig. 31. Configuración necesaria para establecer una conexión de internet en el módulo ESP32.

Una vez configurado el módulo ESP32 en Arduino se procede a conectar a la red de internet local mediante el software Visual Studio, la cual permitirá ingresar el nombre de la red, la contraseña y que sensor se conectará, en este caso se puede observar en la figura 32 el sensor a conectar será el del antebrazo (*sLeftForeArm*). Una vez llenos esos datos se presiona *save settings* y después en *get status*.

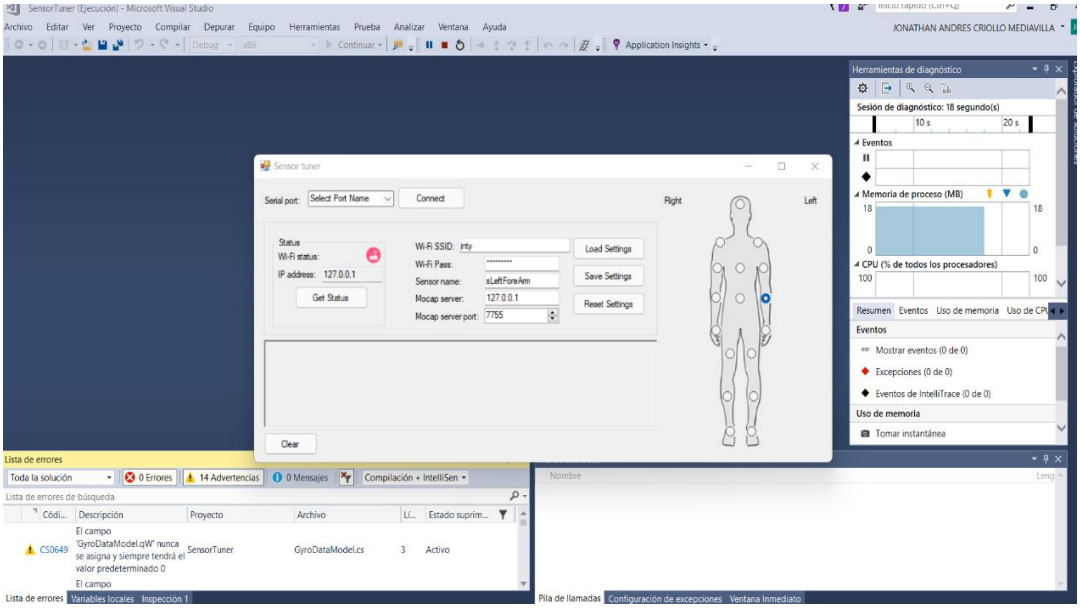


Fig. 32. Ingreso de datos de la red de internet para los diferentes módulos ESP32.

Cuando el indicador Wi-Fi *status* cambia de color de rojo a verde quiere decir que la conexión fue un éxito y está listo para el envío de datos.

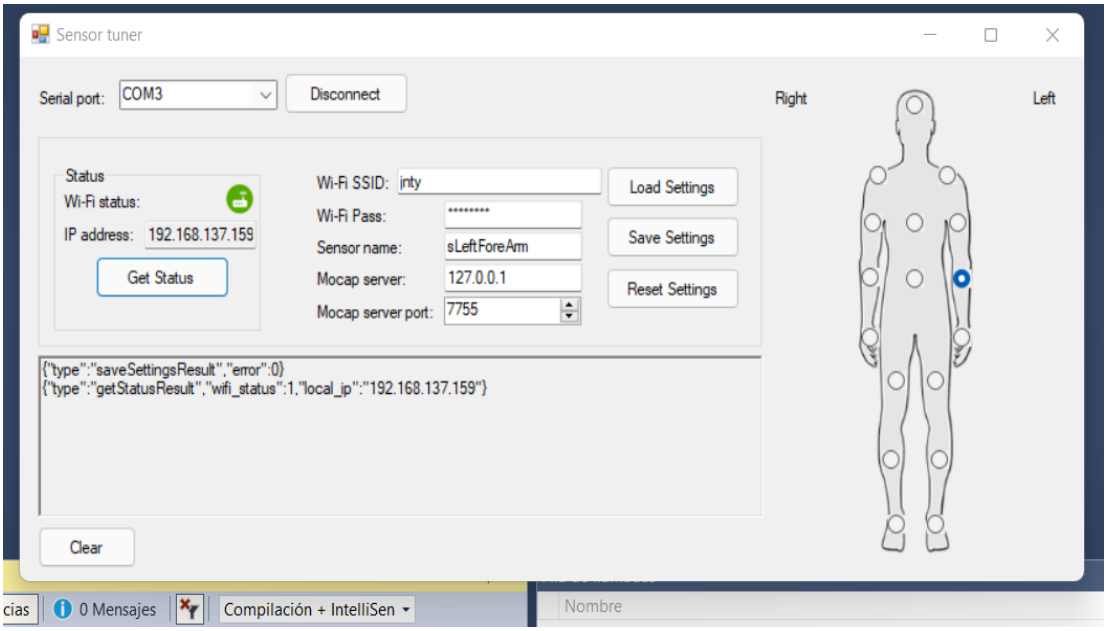


Fig. 33. Indicativo de una conexión con red de internet exitosa.

Una vez conectado a la misma red de internet la computadora y el ESP32 crea un nuevo proyecto en el software *Unreal Engine*, y con la ayuda de una animación que representa a una persona se realiza la configuración.

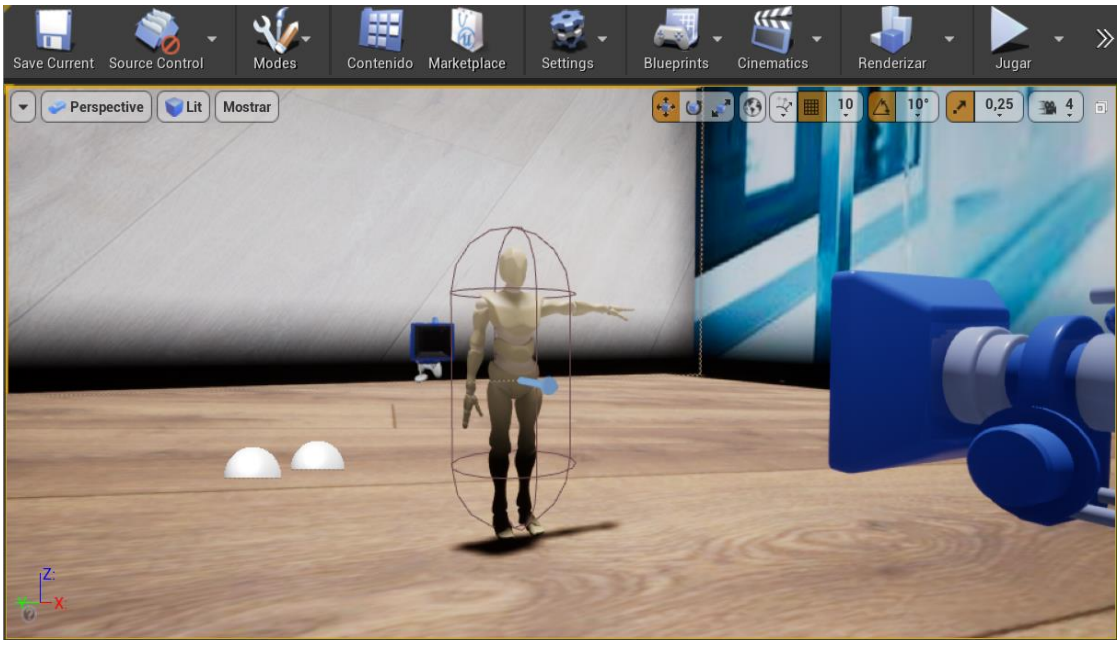


Fig. 34. Representación de un paciente mediante animación en software Unreal Engine.

Se establecen los ángulos de la animación que permitirán la calibración de la simulación como se muestra en la figura 35.

Fig. 35. Configuración de la posición inicial de extremidades superiores.

Si las condiciones de red son las correctas los indicadores cambiarán de color roja a color azul como se muestra en la figura 36.

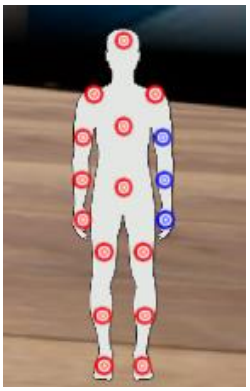


Fig. 36. Indicativos de color azul que demuestran la conectividad exitosa de red.

Se evidenciará en pantalla el movimiento en tiempo real de los sensores como se muestra en la figura 37.

Fig. 37. Movimiento en tiempo real de los sensores mostrados en pantalla.

Al presionar el botón *calibration* los indicadores cambiarán a color verde y la animación se colocará en posición cero como se muestra en la figura 38.

Fig. 38. Indicativos de color verde demuestran la calibración exitosa de los sensores.

Una vez calibrado los sensores se realizó la comunicación para el almacenamiento de los datos. El complemento necesario para poder realizar esta acción es el *AwsCore: DynamoDB* mostrado en la figura 39.

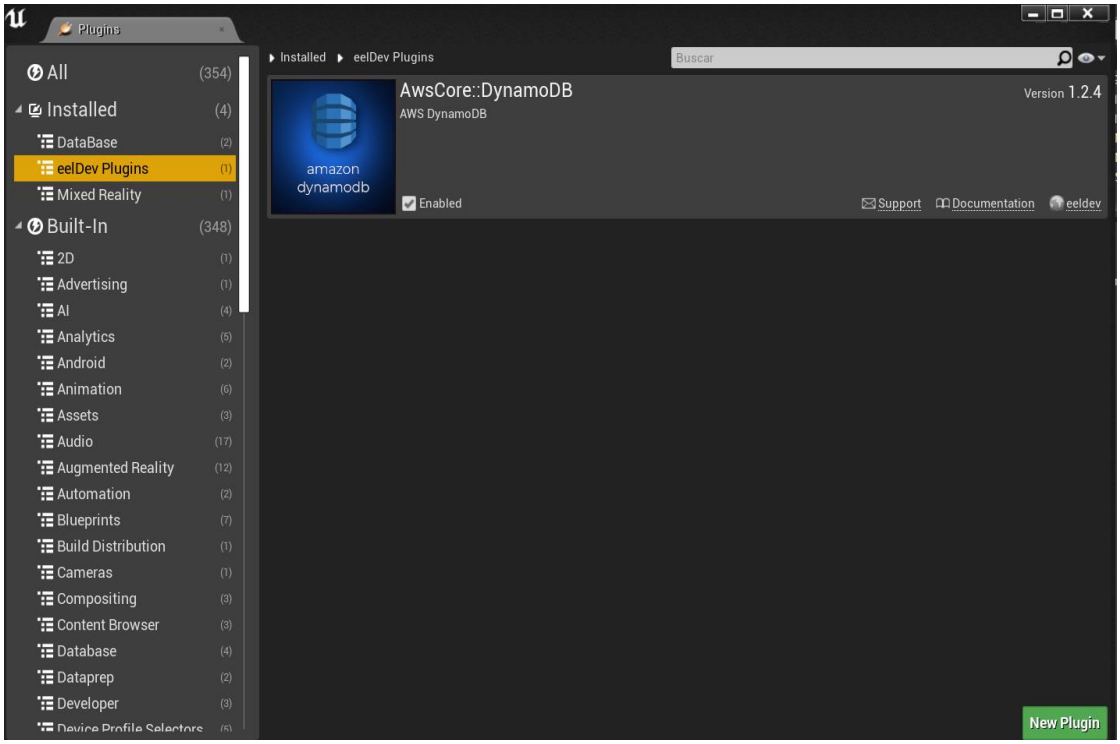


Fig. 39. Complemento necesario para el envío y recepción de datos DynamoDB.

El complemento permite exportar e importar datos desde una base de datos situada en la consola de *Aws de Amazon*.

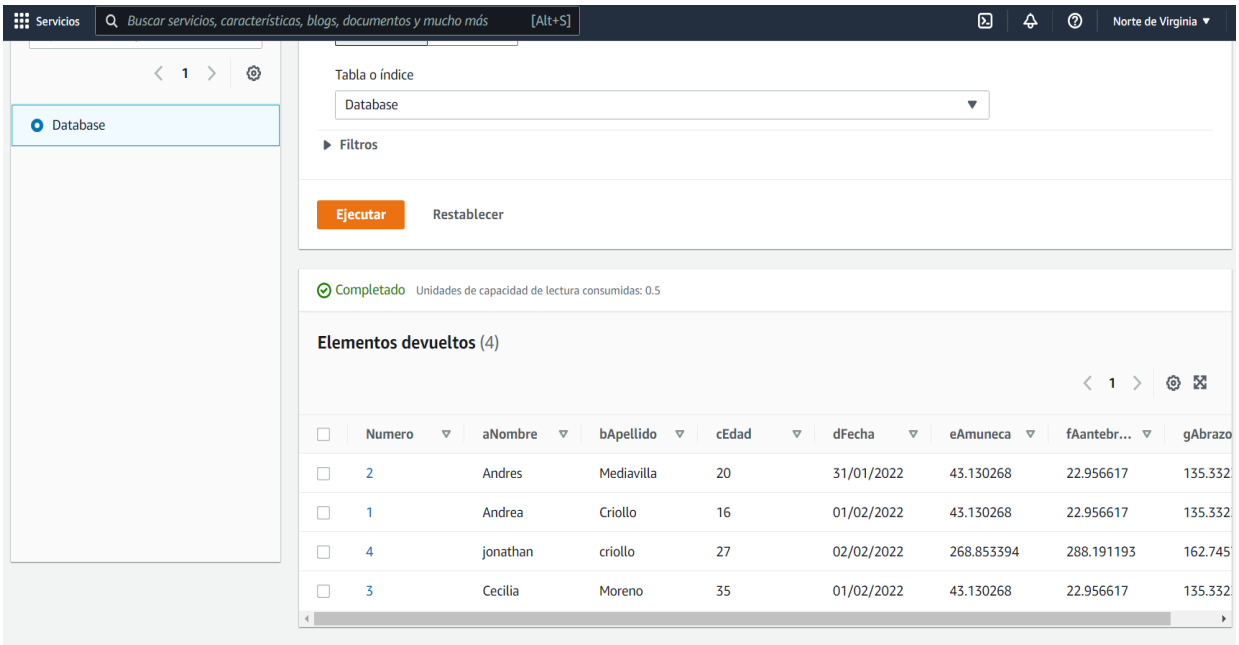


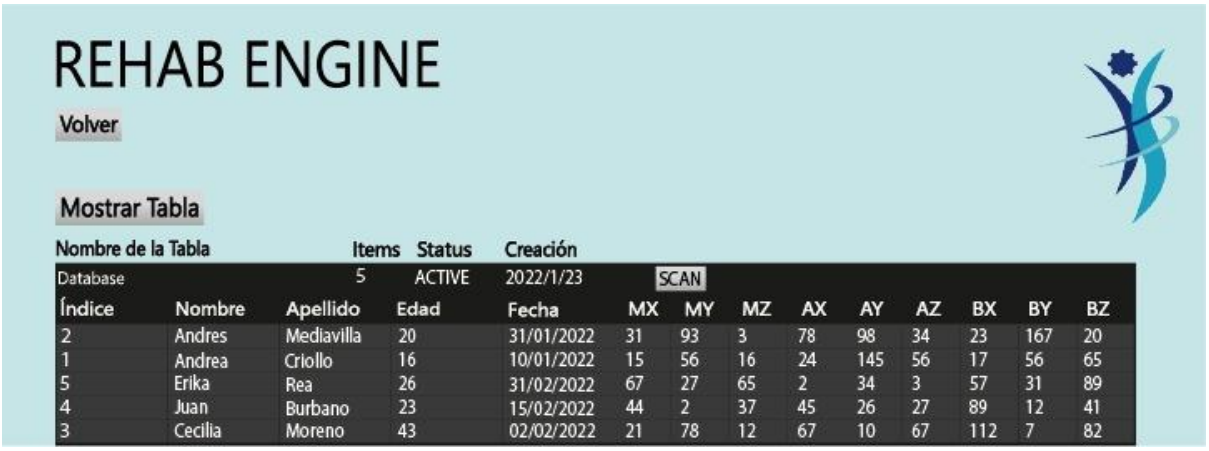
Fig. 40. Consola de base de datos de Aws de Amazon.

Se debe ingresar la información del paciente; nombre, apellido, edad, fecha como se muestra en la figura 41. Al presionar el botón (datos) se mostrarán los valores de los ángulos de la muñeca, antebrazo y brazo. Una vez mostrados esos datos se presiona el botón (subir)

Fig. 41. Ingreso de datos del paciente y visualización de ángulos.

En la tabla que se muestra en la figura 42 se puede observar los datos que fueron subidos de manera exitosa en la aplicación y en la figura 43 los datos en la consola de *Aws*.

En la figura 42 se muestra la tabla donde se guardan los datos del paciente por nombre, apellido, edad y la fecha en la cual se tomó la medida de los ángulos. A continuación, se encuentran las columnas MX, MY y MZ las cuales corresponden al ángulo rotacional generado por la muñeca, seguido las columnas AX, AY, y AZ con la medida del ángulo generado por el antebrazo y las columnas BX, BY y BZ con los ángulos realizados por el brazo del paciente, las letras X, Y y Z en cada nombre hace referencia al eje alrededor el cual se generó la rotación.



Nombre de la Tabla			Items	Status	Creación									
Database			5	ACTIVE	2022/1/23	SCAN								
Índice	Nombre	Apellido	Edad	Fecha	MX	MY	MZ	AX	AY	AZ	BX	BY	BZ	
2	Andres	Mediavilla	20	31/01/2022	31	93	3	78	98	34	23	167	20	
1	Andrea	Criollo	16	10/01/2022	15	56	16	24	145	56	17	56	65	
5	Erika	Rea	26	31/02/2022	67	27	65	2	34	3	57	31	89	
4	Juan	Burbano	23	15/02/2022	44	2	37	45	26	27	89	12	41	
3	Cecilia	Moreno	43	02/02/2022	21	78	12	67	10	67	112	7	82	

Fig. 42. Datos guardados de pacientes ingresados (Unreal Engine).

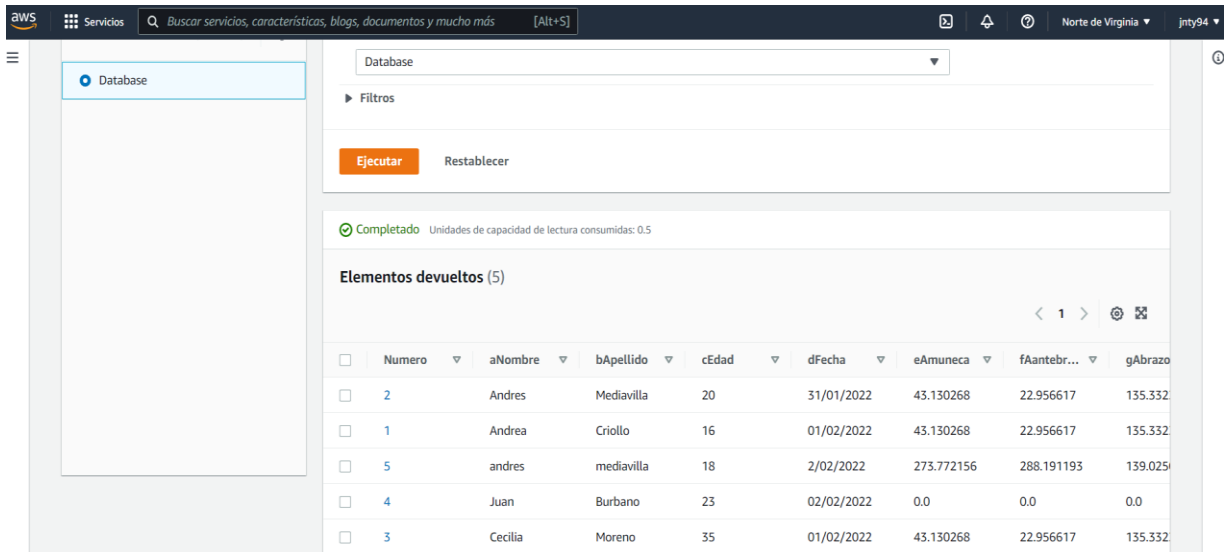


Fig. 43. Datos guardados de pacientes ingresados (AWS).

Por último, se realizó un menú de inicio ilustrado en la figura 44 que permita ingresar a la visualización de la animación en tiempo real, al registro del paciente y un botón salir para finalizar el trabajo con la aplicación.

Fig. 44. Menú principal de la aplicación.

4.2. Análisis de costos

El costo directo de los insumos utilizados para producir el primer ejemplar del sistema para la medición de ángulos en las extremidades superiores se muestra en la tabla 13.

Tabla 13. Costo de insumos para el sistema.

Detalle	Unidades	P. Unitario (USD)	P. Total (USD)
Sensores inerciales MPU-9250	3	9	27
Módulos Esp32	3	12	36
Baterías	3	12	36
Cajas de almacenamiento	3	1	3
Muñequeras	3	2,5	7,5
Plugin (Software)	1	48	48
Otros	1	10	10
		Total	167,5

Este valor disminuirá en el caso de generar el dispositivo en masa, dependiendo de la cantidad a fabricarse.

4.3 Análisis de resultados

Las lesiones se presentan de manera frecuente en la cotidianidad de un centro médico, por lo cual en este trabajo de titulación se presenta un dispositivo que medirá el ángulo de las extremidades superiores a través de la aplicación de sensores inerciales pretendiendo reducir el tiempo empleado en una cita médica para el registro de los datos; que puede ser utilizado en potenciar la rehabilitación y recuperación. Este registro de la evolución del paciente se realiza de manera digital a través de una aplicación de escritorio que puede ser instalado en cualquier computador con sistema operativo Windows. Cuenta además con una base de datos integrada que asegura la perpetuidad de la información en la nube. El costo de la implementación de este sistema para un módulo resulta ser conveniente, ya que se asegura que la vida útil del mismo sea amplia debido al uso de baterías recargables, de un fácil mantenimiento y de disponer de repuestos en el mercado nacional.

Como a cualquier dispositivo se le puede realizar mejoras dependiendo de las necesidades médicas presentes y el software que se utilizó para desarrollarlo garantiza su fácil modificación; estas nuevas ideas serán propuestas para futuros trabajos.

CAPÍTULO V

CONCLUSIONES Y RECOMENDACIONES

5.1 Conclusiones

- El aporte de investigaciones tecnológicas en el campo de la fisioterapia permite a los profesionales del área usar herramientas que faciliten la toma de datos al momento de medir ángulos en procesos de rehabilitación.

- El software Unreal Engine dispone de comandos de programación (Quat) los cuales permiten realizar las diferentes operaciones y mediciones de los ángulos, los cuales a diferencia de los ángulos de Euler (representación de una rotación descomponiéndola en tres rotaciones en los ejes i, j y k) pueden representarse de una manera más sencilla porque a estos cuaterniones se agrega un eje real q0 y tres imaginarios xi, yj y zk, los cuales son más usados en plataformas de juegos o representación virtual en tiempo real.

- El dispositivo diseñado permite establecer una conexión UDP para la obtención de datos proporcionados por los sensores, para transformarlos en ángulos inerciales que el profesional del área pueda manipularlos como convenga.

- La aplicación de escritorio diseñada presenta una interfaz amigable con el usuario de fácil manipulación la cual permite el ingreso y visualización del movimiento en tiempo real de la extremidad superior del paciente.

- El dispositivo y la aplicación que se desarrollaron en este proyecto dan como resultado una herramienta tecnológica funcional en el campo de la rehabilitación física.

5.2 Recomendaciones

- Incluir más sensores inerciales y módulos ESP32 para la toma de ángulos de más partes del cuerpo humano para innovar los procesos realizados por profesionales de fisioterapia.

- Establecer una base de datos que se pueda enviar y recibir datos para el proceso de búsqueda de registro de un paciente mediante su número de identificación o historia clínica.

- Incluir un plugin (complemento) de Unreal Engine que permita el ingreso de datos de forma ordenada, para la correcta manipulación de los datos.

BIBLIOGRAFÍA

[1] C.M.d.l., «Definición de la Fisioterápia,» 1987. [En línea]. Available: https://www.cofiga.org/ciudadanos/fisioterapia/definición. [Último acceso: 12 08 2021].

[2] Hachi Manzur, Uso de sensores inerciales como herramienta complementaria en Estudios de Puestos de Trabajo (EPT) en Chile, para la medición cuantitativa de movimiento repetitivo como factor de riesgo en la calificación de patologías músculoesqueléticos de extremidades., Santiago de Chile: Instituto de Neurociencia Biomédica, 2019.

[3] J. Castellanos, L. Montealegre, B. Martínez, J. Gallo y O. Almanza, Uso de sensores inerciales en fisioterápia: Una aproximación a procesos de evaluación del movimiento humamod., México: Universidad y Salud, 2020.

[4] Vicente Fuster, Diseño y validación de un dispositivo para la cuantificación de la Lordosis Lumbar basado en sensores inerciales, Valencia: Universidad de Valencia, 2017.

[5] C. Ruiz, «Uso de sensores inerciales en fisioterápia,» *Universidad y Salud,* vol. 1, nº 3, p. 57, 2020.

[6] David Pozo, Diseño y construcción de una plataforma didáctica para medir los ángulos de inclinación usando sensores inerciales como acelerómetro y giroscopio, Quito - Ecuador: Escuela Politécnica Nacional, 2010.

[7] Cuesta Vargas, Estudio de la cinemática y fiabilidad de la manipulación vertebral cervical basada en sensores inerciales, España: Asociación Española de Fisioterapeutas, 2011.

[8] C. Roldán, Estudio de la cinemática del miembro superior e inferior mediante sensores inerciales, Málaga: Universidad de Málaga, 2017.

[9] J. Jarrín, Sistema de detección del ángulo articular en los movimientos de miembro superior para evaluación en fisioterapia mediante visión artificial, Ibarra: Universidad Técnica del Norte, 2020.

[10] Alejandro López, Interfaz gráfica para la medición de ángulos de extremidades inferiores del cuerpo humano por medio de sensores inerciales, Ibarra: Universidad Técnica del Norte, 2021.

[11] D. García y V. Germán, Anatomofisioloía de las articulaciones, Curso COT, 2010.

[12] R. Horcajada, Anatomía morfologica aplicada, Madrid: Facultad de bellas artes de San Fernando, 2018.

[13] L. Pérez, Anatomía evolutiva del brazo y el antebrazo, Madrid: Facultad de ciencias biológicas, 2018.

[14] A. Villafonte, «Manual MSD,» febreo 2020. [En línea]. Available: https://www.msdmanuals.com/es-ec/professional/trastornos-de-los-tejidos-musculoesquel%C3%A9tico-y-conectivo/evaluaci%C3%B3n-del-paciente-con-

s%C3%ADntomas-articulares/evaluaci%C3%B3n-del-paciente-con-s%C3%ADntomas-articulares. [Último acceso: 07 11 2021].

[15] M. d. t. e. inmigración, Transtornos Musculoesqueléticos, España: Instituto Nacional de Seguridad e Higiene.

[16] C. o. d. Bizkaia, «Epicondilitis y epitrocleítis,» vol. 25, nº 6, 2011.

[17] G. Fermín, F. Díaz y D. Reis, «Síndrome del Túnel Carpiano,» *Revista Habanera de Ciencias Médicas,* vol. 13, nº 5, 2014.

[18] J. L. R., «¿Cómo funciona un goniómetro?,» [En línea]. Available: https://como-funciona.co/un-goniometro/. [Último acceso: 08 11 2021].

[19] C. Taboadela, Goniometría, Ciudad autónoma de Buenos Aires: ASOCIART, 2007.

[20] Claudio Taboadela, Goniometría : una herramienta para la evaluación de las incapacidades laborales, Buenos Aires: Estela Lafita, 2007.

[21] E. Cubo, P. Mir y Á. Sánchez, Manual SEN de nuevas tecnologias en transtornos de movimiento, MAdrid: SEN, 2021.

[22] L. Llamas, «Arduino,» Ingeniería, Informática y Diseño, 2015. [En línea]. Available: https://www.luisllamas.es/arduino-orientacion-imu-mpu-6050/. [Último acceso: 22 11 2021].

[23] R. Oscar, J. Luis, P. Luis, G. Arturo y P. Adrián, Matlab: Conceptos básicos y representación gráfica, Universidad Miguel Hernández, 2018.

[24] Arduino, «¿Qué es Arduino?,» Arduino, [En línea]. Available: https://arduino.cl/que-es-arduino/. [Último acceso: 08 11 2021].

[25] L. José y Pelegrí José, LabVIEW - Entorno gráfico de programación, Barcelona: MARCOMBO, 2011.

[26] A. Soloaga, «AkademUs,» 19 07 2019. [En línea]. Available: https://www.akademus.es/blog/emprendedores/unreal-engine-que-es-y-para-que-sirve/. [Último acceso: 26 12 2021].

[27] iesalfonsox, «Tecnologia robotica,» 20218. [En línea]. Available: https://iesalfonsox.es/wp-content/uploads/2018/05/TarjetaArduino.pdf. [Último acceso: 08 11 2021].

[28] V. Digi-Key, «Ventas Digi-Key,» [En línea]. Available: https://www.digikey.com/es/articles/how-to-select-and-use-the-right-esp32-wi-fi-bluetooth-module. [Último acceso: 05 01 2022].

[29] R. A. Española, «Diccionario panhistórico del español jurídico,» 2020. [En línea]. Available: https://dpej.rae.es/lema/protocolo-de-comunicaciones. [Último acceso: 25 noviembre 2021].

[30] «Altium,» 30 Noviembre 2020. [En línea]. Available: https://resources.altium.com/es/p/spi-versus-i2c-how-choose-best-protocol-your-memory-chips. [Último acceso: 25 Noviembre 2021].

[31] Infotecs, «Protocolos UDP,» Btob, México, 2020.

[32] E. A. Victor, V. Fernando, M. E. Cosme y Pineda Diego, «Aplicación de ingeniería simultánea en la construcción de máquinas por parte de mipymes metalmecánicas del Ecuador,» Ibarra-Ecuador, 2017.

[33] W. Breckenridge, «Cuaterniones propuestos convenios estándar,» Laboratorio de propulsión a chorro de la NASA. Informe técnico, 1979.

[34] J. A. S. Merinero, Las Normas Técnicas ISO 9241 y EN 29241 sobre pantallas de visualización, Madrid: MAPFRE SEGURIDAD, 1996.

[35] Arduino, «Arduino.cl,» MCI electronics, 2021. [En línea]. Available: https://arduino.cl/arduino-uno/. [Último acceso: 22 11 2021].

[36] Arduino, «Arduino.cl,» MCI electronics, 2021. [En línea]. Available: https://arduino.cl/arduino-nano/. [Último acceso: 22 11 2021].

[37] R. PI, «Raspberry PI,» Raspberry PI, 2021. [En línea]. Available: https://www.raspberrypi.com/products/raspberry-pi-4-model-b/. [Último acceso: 22 11 2021].

[38] J. R. L. J. D. C. Casas Anguitaa, «La encuesta como técnica de investigación. Elaboración de cuestionarios y tratamiento de los datos.,» Escuela Nacional de Sanidad, Madrid-España, 2003.

[39] S. Lorenzoa, J. Miraa, M. Olartea, J. Guerreroa y Silvia Moyanoa, «Análisis matricial de la voz del cliente: QFD aplicado a la gestión sanitaria,» Universidad Miguel Hernández, Madrid-España, 2004.

[40] C. A. Jesus y Hinojosa Adriana, «La Voz del Cliente a través de la Metodologia SERVQUAL / KANO.,» *II Congreso Internacional Virtual de Investigación en Educación Superior y Revista Electrónica de Investigación en Educación Superior,* vol. 2, nº 1, p. 15, 2014.

[41] C. d. N. y. C. d. C. Laboral, «Guía técnica elaboración de mapa funcional,» Sistema Normalizado de Competencia Laboral, Versión: 2.0.

[42] U. Engine, «https://docs.unrealengine.com/,» [En línea]. Available: https://docs.unrealengine.com/4.27/en-US/BlueprintAPI/Math/Quat/. [Último acceso: 25 04 2022].

ANEXOS

ANEXO 1: DATASHEET

1.5 Block Diagram

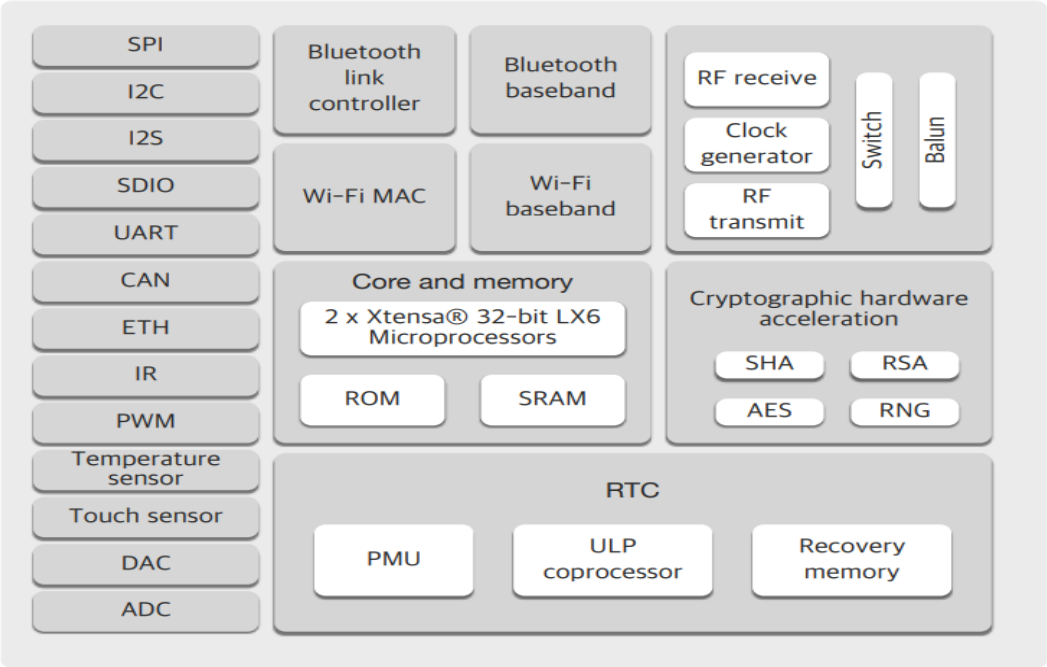


Figure 1: Function Block Diagram

2. Pin Definitions

2.1 Pin Layout

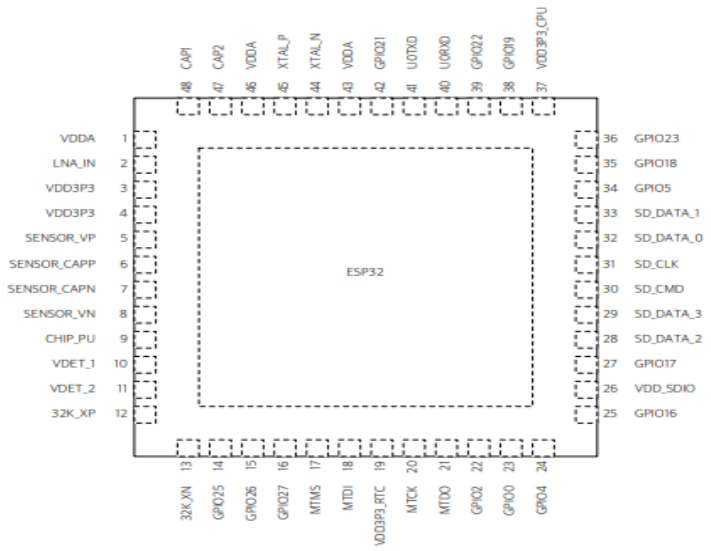


Figure 2: ESP32 Pin Layout

2.2 Pin Description

Table 1: Pin Description

Name	No.	Type	Function
			Analog
VDDA	1	P	Analog power supply (2.3 V – 3.6 V)
LNA_IN	2	I/O	RF input and output
VDD3P3	3	P	Analog power supply (2.3 V – 3.6 V)
VDD3P3	4	P	Analog power supply (2.3 V – 3.6 V)
			VDD3P3_RTC
SENSOR_VP	5	I	GPIO36, ADC1_CH0, RTC_GPIO0
SENSOR_CAPP	6	I	GPIO37, ADC1_CH1, RTC_GPIO1
SENSOR_CAPN	7	I	GPIO38, ADC1_CH2, RTC_GPIO2
SENSOR_VN	8	I	GPIO39, ADC1_CH3, RTC_GPIO3
CHIP_PU	9	I	High: On; enables the chip Low: Off; the chip powers off Note: Do not leave the CHIP_PU pin floating.
VDET_1	10	I	GPIO34, ADC1_CH6, RTC_GPIO4
VDET_2	11	I	GPIO35, ADC1_CH7, RTC_GPIO5
32K_XP	12	I/O	GPIO32, ADC1_CH4, RTC_GPIO9, TOUCH9, 32K_XP (32.768 kHz crystal oscillator input)
32K_XN	13	I/O	GPIO33, ADC1_CH5, RTC_GPIO8, TOUCH8, 32K_XN (32.768 kHz crystal oscillator output)
GPIO25	14	I/O	GPIO25, ADC2_CH8, RTC_GPIO6, DAC_1, EMAC_RXD0
GPIO26	15	I/O	GPIO26, ADC2_CH9, RTC_GPIO7, DAC_2, EMAC_RXD1
GPIO27	16	I/O	GPIO27, ADC2_CH7, RTC_GPIO17, TOUCH7, EMAC_RX_DV
MTMS	17	I/O	GPIO14, ADC2_CH6, RTC_GPIO16, TOUCH6, EMAC_TXD2, HSPICLK, HS2_CLK, SD_CLK, MTMS
MTDI	18	I/O	GPIO12, ADC2_CH5, RTC_GPIO15, TOUCH5, EMAC_TXD3, HSPIQ, HS2_DATA2, SD_DATA2, MTDI
VDD3P3_RTC	19	P	Input power supply for RTC IO (2.3 V – 3.6 V)
MTCK	20	I/O	GPIO13, ADC2_CH4, RTC_GPIO14, TOUCH4, EMAC_RX_ER, HSPID, HS2_DATA3, SD_DATA3, MTCK
MTDO	21	I/O	GPIO15, ADC2_CH3, RTC_GPIO13, TOUCH3, EMAC_RXD3, HSPICS0, HS2_CMD, SD_CMD, MTDO
GPIO2	22	I/O	GPIO2, ADC2_CH2, RTC_GPIO12, TOUCH2, HSPIWP, HS2_DATA0, SD_DATA0
GPIO0	23	I/O	GPIO0, ADC2_CH1, RTC_GPIO11, TOUCH1, EMAC_TX_CLK, CLK_OUT1,
GPIO4	24	I/O	GPIO4, ADC2_CH0, RTC_GPIO10, TOUCH0, EMAC_TX_ER, HSPIHD, HS2_DATA1, SD_DATA1
			VDD_SDIO
GPIO16	25	I/O	GPIO16, HS1_DATA4, U2RXD, EMAC_CLK_OUT
VDD_SDIO	26	P	Output power supply: 1.8 V or the same voltage as VDD3P3_RTC
GPIO17	27	I/O	GPIO17, HS1_DATA5, U2TXD, EMAC_CLK_OUT_180
SD_DATA_2	28	I/O	GPIO9, HS1_DATA2, U1RXD, SD_DATA2, SPIHD
SD_DATA_3	29	I/O	GPIO10, HS1_DATA3, U1TXD, SD_DATA3, SPIWP
SD_CMD	30	I/O	GPIO11, HS1_CMD, U1RTS, SD_CMD, SPICS0
SD_CLK	31	I/O	GPIO6, HS1_CLK, U1CTS, SD_CLK, SPICLK
SD_DATA_0	32	I/O	GPIO7, HS1_DATA0, U2RTS, SD_DATA0, SPIQ
SD_DATA_1	33	I/O	GPIO8, HS1_DATA1, U2CTS, SD_DATA1, SPID
			VDD3P3_CPU
GPIO5	34	I/O	GPIO5, HS1_DATA6, VSPICS0, EMAC_RX_CLK
GPIO18	35	I/O	GPIO18, HS1_DATA7, VSPICLK
GPIO23	36	I/O	GPIO23, HS1_STROBE, VSPID
VDD3P3_CPU	37	P	Input power supply for CPU IO (1.8 V – 3.6 V)
GPIO19	38	I/O	GPIO19, U0CTS, VSPIQ, EMAC_TXD0
GPIO22	39	I/O	GPIO22, U0RTS, VSPIWP, EMAC_TXD1
U0RXD	40	I/O	GPIO3, U0RXD, CLK_OUT2
U0TXD	41	I/O	GPIO1, U0TXD, CLK_OUT3, EMAC_RXD2
GPIO21	42	I/O	GPIO21, VSPIHD, EMAC_TX_EN
			Analog
VDDA	43	P	Analog power supply (2.3 V – 3.6 V)
XTAL_N	44	O	External crystal output
XTAL_P	45	I	External crystal input
VDDA	46	P	Analog power supply (2.3 V – 3.6 V)
CAP2	47	I	Connects to a 3 nF capacitor and 20 kΩ resistor in parallel to CAP1
CAP1	48	I	Connects to a 10 nF series capacitor to ground
GND	49	P	Ground

3.5 Wi-Fi

ESP32 implements a TCP/IP and full 802.11 b/g/n Wi-Fi MAC protocol. It supports the Basic Service Set (BSS) STA and SoftAP operations under the Distributed Control Function (DCF). Power management is handled with minimal host interaction to minimize the active-duty period.

3.5.1 Wi-Fi Radio and Baseband

The ESP32 Wi-Fi Radio and Baseband support the following features:

- 802.11b/g/n
- 802.11n MCS0-7 in both 20 MHz and 40 MHz bandwidth
- 802.11n MCS32 (RX)

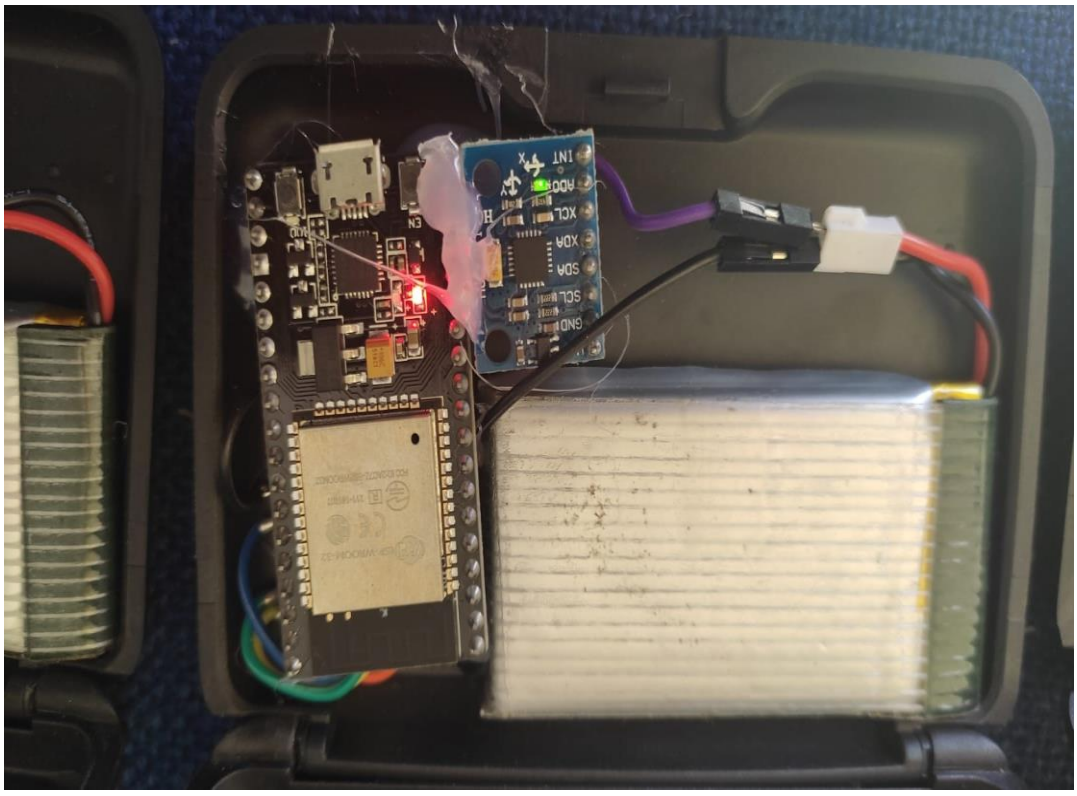
- 802.11n 0.4 μs guard-interval
- up to 150 Mbps of data rate
- Receiving STBC 2×1
- Up to 20.5 dBm of transmitting power
- Adjustable transmitting power
- Antenna diversity
  ESP32 supports antenna diversity with an external RF switch. One or more GPIOs control the RF switch and selects the best antenna to minimize the effects of channel fading.

3.5.2 Wi-Fi MAC

The ESP32 Wi-Fi MAC applies low-level protocol functions automatically. They are as follows:

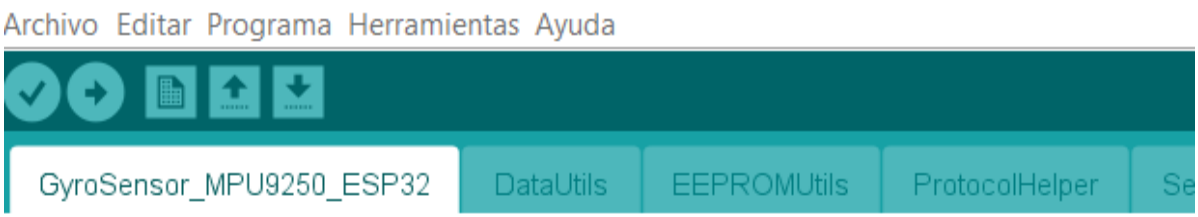
- 4 × virtual Wi-Fi interfaces
- Simultaneous Infrastructure BSS Station mode/SoftAP mode/Promiscuous mode
- RTS protection, CTS protection, Immediate Block ACK
- Defragmentation
- TX/RX A-MPDU, RX A-MSDU
- TXOP
- WMM
- CCMP (CBC-MAC, counter mode), TKIP (MIC, RC4), WAPI (SMS4), WEP (RC4) and CRC
- Automatic beacon monitoring (hardware TSF)

ANEXO 2: FOTOS DEL DISPOSITIVO




Codo

ANEXO 2: PROGRAMACIÓN




```
#include <Wire.h>
#include "I2Cdev.h"
#include "MPU9250_9Axis.h"
#include "EEPROM.h"
#include <WiFi.h>
#include "AsyncUDP.h"

#define EEPROMUtils
#define SettingsUtils
#define SerialPortHelper
#define WiFiHelper
#define UDPClient
#define ProtocolHelper
#define DataUtils

MPU9250 mpu(0x68);

uint8_t mpuIntStatus;
uint16_t packetSize;
uint16_t fifoCount;
uint8_t fifoBuffer[64];
volatile bool mpuInterrupt = false;

Quaternion q;

unsigned long timing;
unsigned long gyroDataIntervalSend = 50;

void setup()
{
    Serial.begin(115200);
    EEPROM.begin(500);

    InitSettings();
```


```
    InitSettings();
    InitWiFi();
    UDPConnect();

    Wire.begin();
    Wire.setClock(400000);

    mpu.initialize();

    delay(1000);

    uint8_t devStatus = mpu.dmpInitialize();

    if (devStatus == 0)
    {
        mpu.setDMPEnabled(true);
        mpuIntStatus = mpu.getIntStatus();
        packetSize = mpu.dmpGetFIFOPacketSize();
    }
}

void loop()
{
    mpuInterrupt = false;

    mpuIntStatus = mpu.getIntStatus();

    fifoCount = mpu.getFIFOCount();

    if ((mpuIntStatus & 0x10) || fifoCount == 1024)
    {
        mpu.resetFIFO();
    }
```


```
    mpuIntStatus = mpu.getIntStatus();

    fifoCount = mpu.getFIFOCount();

    if ((mpuIntStatus & 0x10) || fifoCount == 1024)
    {
        mpu.resetFIFO();
    }
    else if (mpuIntStatus & 0x02)
    {
        while (fifoCount < packetSize)
            fifoCount = mpu.getFIFOCount();

        mpu.getFIFOBytes(fifoBuffer, packetSize);

        fifoCount -= packetSize;

        mpu.dmpGetQuaternion(&q, fifoBuffer);

        if (millis() - timing > gyroDataIntervalSend)
        {
            timing = millis();
            String sensorName = ReadStringEEPROM(2);
            String gyroDataJSONString = GetGyroDataJSONString(sensorName, q.x, q.y, q.z, q.w);
            UDPSendData(gyroDataJSONString);
        }
    }

    SerialPortReceive();
}

void dmpDataReady()
{
    mpuInterrupt = true;
}
```


```
String IPToString(IPAddress ip)
{
    String str = "";
    for (int i = 0; i < 4; i++)
        str += i  ? "." + String(ip[i]) : String(ip[i]);
    return str;
}
```

Archivo Editar Programa Herramientas Ayuda

GyroSensor_MPU9250_ESP32 | DataUtils | EEPROMUtils | ProtocolHelper | SerialPortHelper | Setti

```
#define EEPROM_SIZE 24

String ReadStringEEPROM(char slot)
{
    int i;
    char data[EEPROM_SIZE];
    int length = 0;
    unsigned char k;
    k = EEPROM.read(EEPROM_SIZE * slot);
    while(k != '\0' && length < EEPROM_SIZE)
    {
        k = EEPROM.read(EEPROM_SIZE * slot + length);
        data[length] = k;
        length++;
    }
    data[length] = '\0';
    return String(data);
}

bool CheckIsEmptyEEPROM(char slot)
{
    bool isEmpty = true;
    for (int i = EEPROM_SIZE * slot; i < EEPROM_SIZE * slot + EEPROM_SIZE; i++)
    {
        unsigned char val = EEPROM.read(i);
        if (val != 255)
        {
            isEmpty = false;
            break;
        }
    }
    return isEmpty;
```

Archivo Editar Programa Herramientas Ayuda

GyroSensor_MPU9250_ESP32 | DataUtils | EEPROMUtils | ProtocolHelper | SerialPortHelper | SettingsUtils

```
        unsigned char val = EEPROM.read(i);
        if (val != 255)
        {
            isEmpty = false;
            break;
        }
    }
    return isEmpty;
}

void WriteStringEEPROM(char slot, String data)
{
    int dataLength = data.length();
    for(int i = 0; i < dataLength; i++)
        EEPROM.write(EEPROM_SIZE * slot + i, data[i]);
    EEPROM.write(EEPROM_SIZE * slot + dataLength, '\0');
    EEPROM.commit();
    delay(50);
}

void EraseSlotEEPROM(char slot)
{
    for (int i = EEPROM_SIZE * slot; i < EEPROM_SIZE * slot + EEPROM_SIZE; i++)
      EEPROM.write(EEPROM_SIZE * slot + i, 255);
    EEPROM.commit();
    delay(50);
}

void EraseEEPROM()
{
    for (int i = 0; i < 500; i++)
      EEPROM.write(i, 255);
    EEPROM.commit();
    delay(50);
}
```

Archivo Editar Programa Herramientas Ayuda

Abrir

GyroSensor_MPU9250_ESP32 | DataUtils | EEPROMUtils | ProtocolHelper | SerialPortHelper | SettingsUtils

```
#include "ArduinoJson.h"

String GetStatusJSONString(int wifiStatus, String localIP)
{
    StaticJsonDocument<1024> staticJsonDocument;
    staticJsonDocument["type"] = "getStatusResult";
    staticJsonDocument["wifi_status"] = wifiStatus;
    staticJsonDocument["local_ip"] = localIP;

    char docBuf[1024];
    serializeJson(staticJsonDocument, docBuf);

    return String(docBuf);
}

String GetGyroDataJSONString(String sensorName, float qX, float qY, float qZ, float qW)
{
    StaticJsonDocument<1024> staticJsonDocument;
    staticJsonDocument["type"] = "gyroData";
    staticJsonDocument["sensorName"] = sensorName;
    staticJsonDocument["qX"] = String(qX, 2);
    staticJsonDocument["qY"] = String(qY, 2);
    staticJsonDocument["qZ"] = String(qZ, 2);
    staticJsonDocument["qW"] = String(qW, 2);

    char docBuf[1024];
    serializeJson(staticJsonDocument, docBuf);

    return String(docBuf);
}
```

Archivo Editar Programa Herramientas Ayuda

Salvar

GyroSensor_MPU9250_ESP32 | DataUtils | EEPROMUtils | ProtocolHelper

```
AsyncUDP udpClient;

void UDPConnect()
{
     String mocapServer = ReadStringEEPROM(3);
     String mocapServerPort = ReadStringEEPROM(4);

     IPAddress ipAddress = IPAddress();
     ipAddress.fromString(mocapServer);

     int port = mocapServerPort.toInt();

     udpClient.connect(ipAddress, port);
}

void UDPSendData(String message)
{
     String mocapServerPort = ReadStringEEPROM(4);
     int port = mocapServerPort.toInt();

     char charBuffer[1024];
     message.toCharArray(charBuffer, 1024);

     if (udpClient.connected())
          udpClient.broadcastTo(charBuffer, port);
     else
          UDPConnect();
}
```

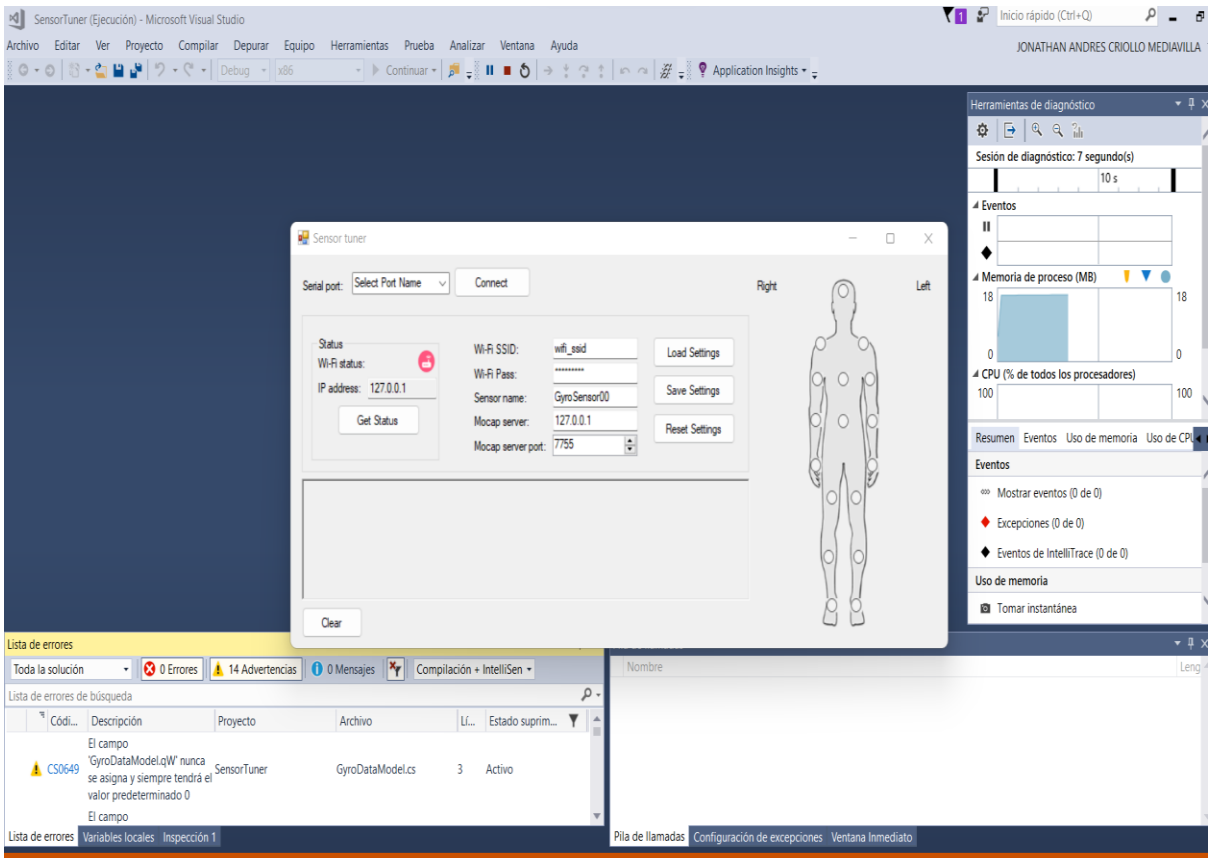
Archivo Editar Programa Herramientas Ayuda

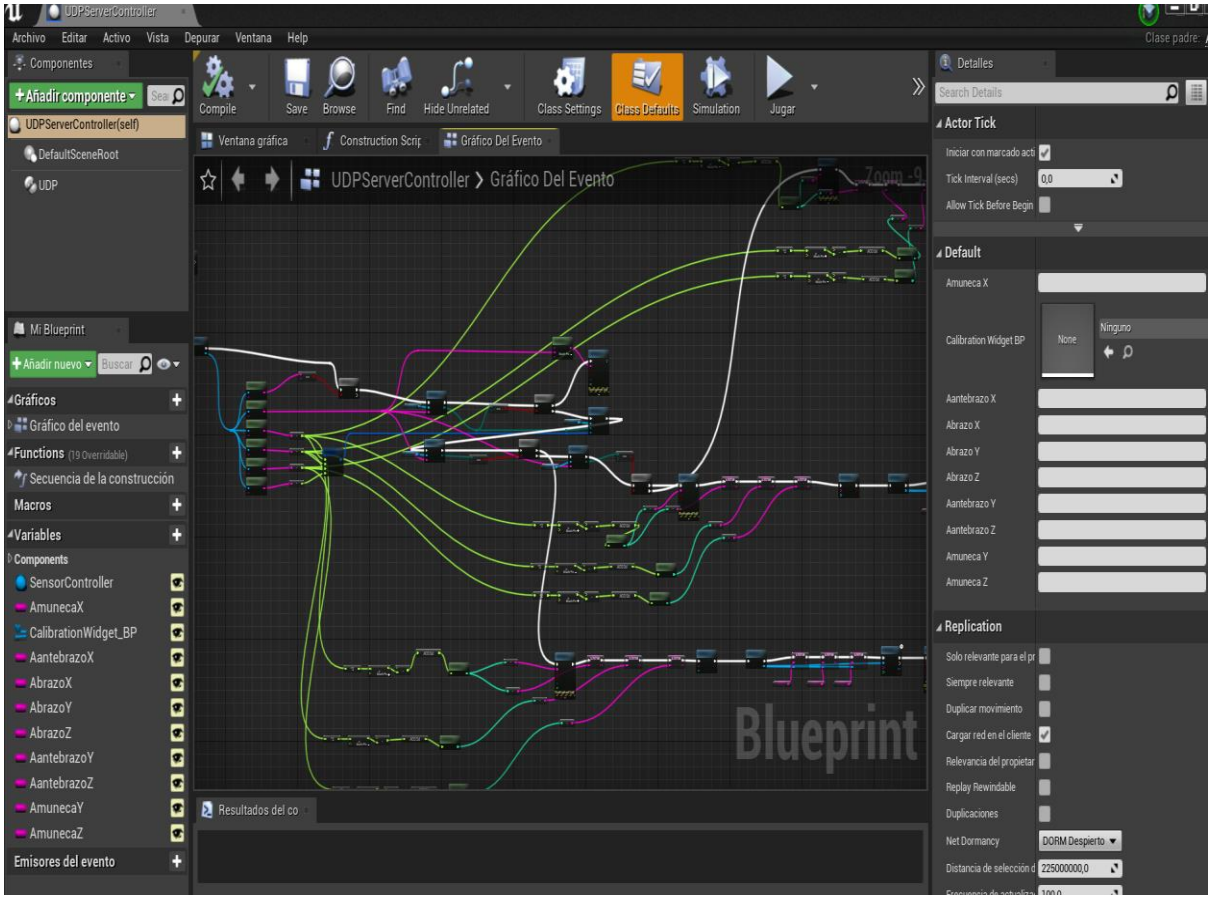
GyroSensor_MPU9250_ESP32 | DataUtils | EEPROMUtils | ProtocolHelper

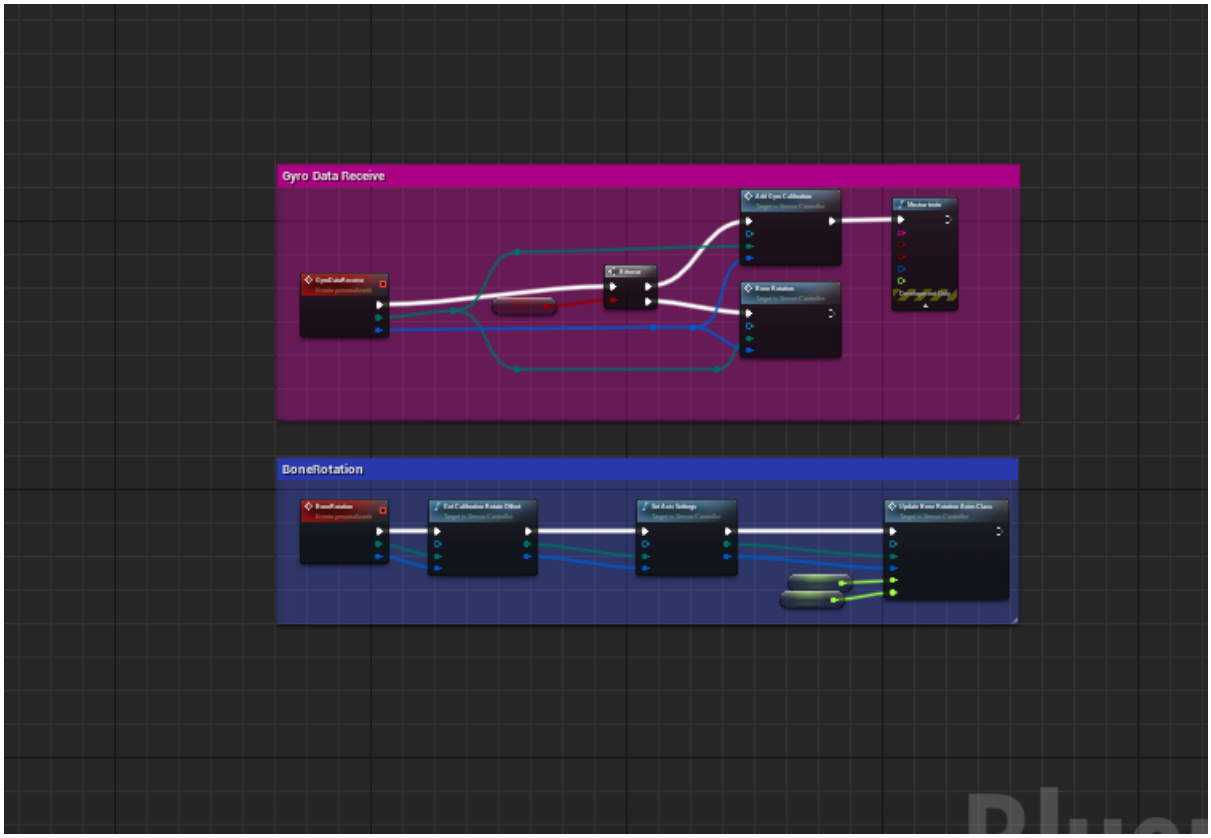
```
void InitWiFi()
{
    String ssidStr = ReadStringEEPROM(0);
    const char *ssid = ssidStr.c_str();
    String passStr = ReadStringEEPROM(1);
    const char *pass = passStr.c_str();

    WiFi.begin(ssid, pass);

    for(int i = 0; i < 5; i++)
    {
        if (WiFi.status() == WL_CONNECTED)
            break;
        delay(1000);
    }
}
```




UDPServerController
Archivo Editar Activo Vista Depurar Ventana Help
Componentes
+Añadir componente
UDPServerController(self)
DefaultSceneRoot
UDP
Compile Save Browse Find Hide Unrelated Class Settings Class Defaults Simulation Jugar
Ventana gráfica Construction Scrip Gráfico Del Evento
UDPServerController › Gráfico Del Evento
Blueprint
Detalles
Search Details
Actor Tick
Iniciar con marcado acti
Tick Interval (secs) 0,0
Allow Tick Before Begin
Default
Amuneca X
Calibration Widget BP
None
Ninguno
Aantebrazo X
Abrazo X
Abrazo Y
Abrazo Z
Aantebrazo Y
Aantebrazo Z
Amuneca Y
Amuneca Z
Replication
Solo relevante para el pr
Siempre relevante
Duplicar movimiento
Cargar red en el cliente
Relevancia del propietar
Replay Rewindable
Duplicaciones
Net Dormancy DORM Despierto
Distancia de seleccion d 225000000,0
Mi Blueprint
+Añadir nuevo
Buscar
Gráficos
Gráfico del evento
Functions (19 Overridable)
Secuencia de la construcción
Macros
Variables
Components
SensorController
AmunecaX
CalibrationWidget_BP
AantebrazoX
AbrazoX
AbrazoY
AbrazoZ
AantebrazoY
AantebrazoZ
AmunecaY
AmunecaZ
Emisores del evento
Resultados del co


Gyro Data Receive
BoneRotation