



UJAT

UNIVERSIDAD JUÁREZ
AUTÓNOMA DE TABASCO

“ESTUDIO EN LA DUDA. ACCIÓN EN LA FE”



Universidad Juárez Autónoma de Tabasco

Tesis de Maestría

Sistema IoT de sensores corporales para la captura del movimiento de la marcha humana

Que presenta

Cristian Matías Alberoni

Para obtener el grado de

Maestro en Ciencias de la Computación

Director

Dr. José Adán Hernández Nolasco

Dr. Francisco Diego Acosta Escalante

Cuerpo Académico

Sistemas Distribuidos

Línea de Generación y Aplicación del Conocimiento

Computación móvil y ubicua

Cunduacán, Tabasco, México

Marzo 2020



UJAT

UNIVERSIDAD JUÁREZ
AUTÓNOMA DE TABASCO

"ESTUDIO EN LA DUDA. ACCIÓN EN LA FE"



Universidad Juárez Autónoma de Tabasco

Tesis de Maestría

Sistema IoT de sensores corporales para la captura del movimiento de la marcha humana

Que presenta

Cristian Matías Alberoni

Para obtener el grado de

Maestro en Ciencias de la Computación

Director

Dr. José Adán Hernández Nolasco

Dr. Francisco Diego Acosta Escalante

Jurado: **Dr. Miguel Antonio Wister Ovando**
Dr. Pablo Pancardo García
Dr. Pablo Payró Campos

Cuerpo Académico
Sistemas Distribuidos

Línea de Generación y Aplicación del Conocimiento
Computación móvil y ubicua

Cunduacán, Tabasco, México

Marzo 2020

CARTA AUTORIZACIÓN

El que subscribe, autoriza por medio de la presente escrito a la Universidad Juárez Autónoma de Tabasco par que utilice tanto física como digitalmente le Tesis de Maestría "Sistema IoT de sensores corporales para la captura del movimiento de la marcha humana", de la cual soy autor y titular de los Derechos de Autor.

La finalidad del uso por parte de la Universidad Juárez Autónoma de Tabasco de la tesis antes mencionada, será única y exclusivamente para difusión, educación y sin fines de lucro; autorización que se hace de manera enunciativa más no limitativa para subirla a la Red Abierta de Biblioteca Digitales (RABID) y a cualquier otra Red Académica con la que la Universidad tenga relación institucional.

Por lo antes mencionado, libero a la Universidad Juárez Autónoma de Tabasco de cualquier reclamo legal que pudiera ejercer respecto al uso y manipulación de la Tesis mencionada y para los fines estipulados en este documento.

Se firme la presente autorización en la Ciudad de Villahermosa, Tabasco a los 6 días del mes de marzo del año 2020.


AUTORIZO
Cristian Matías Alberoni



UNIVERSIDAD JUÁREZ
AUTÓNOMA DE TABASCO

"ESTUDIO EN LA DUDA. ACCIÓN EN LA FE"



DIVISIÓN ACADÉMICA DE
CIENCIAS Y TECNOLOGÍAS
DE LA INFORMACIÓN



Oficio No. 452/2020/DACYTI/D
06 de marzo de 2020

C. Cristian Matías Alberoni
Matrícula 172H13003

En virtud de que cumple satisfactoriamente los requisitos establecidos en el Reglamento General de Estudio de Posgrado vigente en la Universidad, informo a Usted que se autoriza la impresión del trabajo recepcional **"Sistema IoT de Sensores Corporales para la Captura del Movimiento de la Marcha Humana"**, para presentar examen y obtener el Grado de Maestro en Ciencias de la Computación bajo la modalidad de Tesis.

Sin otro particular, aprovecho la oportunidad para saludarle.

Atentamente

UNIVERSIDAD JUÁREZ AUTÓNOMA DE TABASCO


MTE Oscar Alberto González González
Director



DIVISIÓN ACADÉMICA DE INFORMÁTICA Y SISTEMAS

C.c.p. MASI, Arturo Corona Ferreira.- Encargado del Despacho de la Coordinación de Posgrado.
Archivo.
Consecutivo.

Miembro CUMEN desde 2008
Consortio de
Universidades
Mexicanas
ORGANISMO DE CALIDAD PARA LA EDUCACIÓN SUPERIOR

Carretera Cunduacán-Jalpa Km. 1, Colonia Esmeralda, C.P. 86690.
Cunduacán, Tabasco, México.
Tel: (993) 358 1500 ext. 6727; (914) 336 0616; Fax: (914) 336 0870
E-mail: direccion.dacyti@ujat.mx
<http://ujat.mx/dacyti>



UNIVERSIDAD JUÁREZ AUTÓNOMA DE TABASCO

"ESTUDIO EN LA DUDA. ACCIÓN EN LA FE"

DIVISIÓN ACADÉMICA DE CIENCIAS Y TECNOLOGÍAS DE LA INFORMACIÓN

Cunduacán Tabasco 05 de marzo de 2020

En la Universidad Juárez Autónoma de Tabasco, de acuerdo al Reglamento de Estudios de Posgrado vigente, se revisó el trabajo de investigación titulado **"Sistema IOT de Sensores Corporales para la Captura del Movimiento de la Marcha Humana"**, realizado por el **C. Cristian Matías Alberoni**, para obtener el Grado de Maestro en Ciencias de la Computación bajo la modalidad de Tesis.

Los integrantes del jurado, después de revisar el trabajo, lo declararon aceptado. Firmando la presente a los 05 días del mes de marzo de 2020.


Dr. Miguel Antonio Wister Ovando


Dr. Pablo Pancardo García


Dr. Pablo Payró Campos



Índice general

Tabla de contenido	I
Índice de Figuras	IV
Índice de Tablas	VII
Resumen	1
1. Generalidades	2
1.1. Introducción	2
1.2. Planteamiento del problema	3
1.2.1. Definición del problema	3
1.2.2. Delimitación de la investigación	4
1.3. Pregunta de investigación	5
1.4. Hipótesis	5
1.5. Objetivo general	5
1.6. Objetivos específicos	5
1.7. Justificación	6
1.8. Metodología utilizada	6
2. Marco teórico	8
2.1. Conceptos y teorías fundamentales de la investigación	8
2.1.1. Biomecánica de la marcha	8
2.2. Literatura relacionada	12
2.2.1. Arquitectura de BSN y Nodo de sensores	13
2.2.2. Transmisión de la información, tipología de redes	14
2.2.3. Algoritmos, filtros, fusión y almacenamiento de datos	14
2.3. Marco tecnológico	16

2.3.1.	MEMS	16
2.3.2.	Computadora de placa única	17
2.3.3.	Redes de sensores inalámbricas	18
2.3.4.	Representación y manipulación de los datos obtenidos	20
3.	Modelado de la biomecánica de la marcha	22
3.1.	Modelo matemático	22
3.1.1.	Péndulo invertido simple	22
3.1.2.	Péndulo invertido doble	22
3.1.3.	Péndulo invertido simple de dos ejes	23
3.1.4.	Descripción de modelo de péndulo invertido simple de dos ejes	24
3.1.5.	Modelo de simulación en Matlab	25
3.2.	Modelado de filtro digital y obtención de ángulos de Euler	26
3.2.1.	Representación usando cuaterniones	26
3.2.2.	Obtención de matriz de rotación	28
3.2.3.	Obtención de ángulos de Euler	28
3.2.4.	Obtención de orientación a partir de la tasa angular	29
3.2.5.	Obtención de orientación a partir observaciones vectoriales	30
3.2.6.	Algoritmo de fusión	33
3.2.7.	Cálculo de ganancia de Filtro	34
4.	Red IoT de sensores corporales	36
4.1.	Propuesta de modelo	36
4.2.	Arquitectura general	37
4.3.	Hardware	41
4.3.1.	Nodos	41
4.3.2.	Esquemas de circuito de nodos	43
4.3.3.	Hardware de Servidor	50
4.4.	Firmware	51
4.4.1.	Funcionamiento	52
4.4.2.	Arquitectura de Firmware	53
4.4.3.	Implementación de la Aplicación	55
4.4.4.	Servicios y características Bluetooth LE	59
4.4.5.	Rutinas de bajo consumo	63
4.5.	Software	63
4.5.1.	Servidor	63

5. Experimentos y Resultados	72
5.1. Prototipo de nodo	72
5.1.1. Características físicas	73
5.1.2. Testeo de servicios	74
5.2. Prototipo de servidor	74
5.3. Análisis pérdida de paquetes en función de distancia	77
5.4. Ensayo comparativo usando test 6MWT	80
5.4.1. Análisis de la marcha usando iGait	82
6. Contribuciones, conclusiones y trabajos futuros	87
6.1. Conclusiones	87
6.2. Contribuciones	88
6.3. Trabajos futuros	88
Bibliografía	90

Índice de figuras

2.1. Fases de la marcha humana. Fuente: [22]	10
2.2. Parámetros de la marcha humana	11
2.3. División de fase de apoyo. Fuente: [22]	11
2.4. División de fase de oscilación. Fuente: [22]	12
3.1. Modelo del péndulo simple invertido. Fuente: [38]	23
3.2. Modelo del péndulo doble invertido. Fuente: [12]	23
3.3. Esquema de péndulo invertido de 2 ejes	25
3.4. Modelo 3D de msra-walking-robot simulado con Simscape Multibody en Simulink de Matlab. Fuente: [47]	26
3.5. Orientación del marco de referencia B mediante una rotación θ alrededor del eje ${}^A\hat{r}$ respecto del marco A.	27
3.6. Ángulos de Euler denotados con ϕ , ψ y θ .	29
3.7. Diagrama de bloque de la implementación del Filtro	35
4.1. Modelo general propuesto	38
4.2. Detalle de modelo general propuesto	39
4.3. Arquitectura del sistema propuesto	40
4.4. Broker MQTT: Publicación y suscripción en multiples topics	41
4.5. Circuito esquemático de nodos	44
4.6. Módulo de batería y batería LIR2032H	45
4.7. PinOut Arduino Nano 33 BLE	46
4.8. Módulo CPU y Bluetooth 5	47
4.9. Módulo de sensores inerciales MPU-9250	48
4.10. Fotos de modelos de sujeción con neopreno comercial	49
4.11. Prototipado de gabinete para nodo diseñado en SolidWorks 2018	50
4.12. Servidor montando en Raspberry Pi y fuente de alimentación portátil	51

4.13. UML Diagrama de máquina de estado de cada Nodo	53
4.14. Arquitectura del firmware	54
4.15. Diagrama de flujo de ejecución de estados para la aplicación NodoSensor.	56
4.16. Diagrama de flujo de configuración de la aplicación NodoSensor	57
4.17. Diagrama de flujo del estado LPM	57
4.18. Diagrama de flujo del estado Baja Batería	58
4.19. Diagrama de flujo del estado de Enciende BLE	59
4.20. Diagrama de flujo del estado Apagar BLE	60
4.21. Diagrama de flujo del estado SM	60
4.22. Datos generados por sensores con su marca de tiempo.	63
4.23. Arquitectura general del sistema operativo Linux	64
4.24. Arquitectura de BlueZ	65
4.25. Estructura jerárquica de los tópicos MQTT	66
4.26. Nodo Virtual	67
4.27. Diagrama de flujo de integración de nodo virtual	68
4.28. Diagrama de flujo de algoritmo de fusión de datos	69
4.29. Diagrama de Base de datos	71
4.30. DB Browser for SQLite DB	71
5.1. Fotografías de nodos montados en gabinetes finales.	73
5.2. Comprobación de peso del hardware de los nodos.	73
5.3. Verificación de servicios BLE disponibles en los nodos.	75
5.4. Servidor Raspberry PI con fuente de alimentación portátil de 20000mAh	76
5.5. GUI interfaz gráfica	77
5.6. Escaneo de los sensores encontrados	77
5.7. Muestra de gráficos en Tiempo Real	78
5.8. Gráfico de comportamiento de la media de paquetes perdidos en porcentaje.	80
5.9. Ubicación de nodos sensores en el cuerpo de la persona para medición.	81
5.10. Ubicación de nodos sensores en el cuerpo de la persona para medición.	82
5.11. Gráficos de mediciones en crudo de Aceleración y Velocidad angular en 3D.	83
5.12. Gráficos de trayectoria lineal por cada eje y en tres dimensio- nes en función del tiempo	84

5.13. Análisis de parámetros usando Kinovea	85
5.14. Resultado del análisis del set de datos usando iGait.	86

Universidad Juárez Autónoma de Tabasco.
México.

Índice de tablas

3.1. Valores de $\tilde{w}_\beta$ para el sensor MPU-9250.	35
4.1. Hardware de Nodo	42
4.2. Comparativa de microcontroladores nrf2840 y atmega328p . .	43
4.3. Característica de la batería	45
4.4. Tabla de consumo energético por componente y duración de carga de batería	49
4.5. Hardware de servidor	51
5.1. Características técnicas físicas de los nodos	74
5.2. Resultado de prueba de distancia	79
5.3. Tabla comparativa de parámetros de la marcha obtenidos usan- do iGait y Kinovea	84

Resumen

En este trabajo se desarrolló un prototipo de red IoT de sensores corporales para la captura de información biomecánica de la marcha utilizando sensores inerciales, como acelerómetros y giroscopios.

El prototipo hace uso de software y hardware abierto como Arduino y Raspberry Pi para adquisición de la información. El objetivo es obtener datos referentes a la biomecánica de la marcha mediante y generar un conjunto de datos que puedan ser generados.

Como propuesta para resolver el problema de adquisición, procesamiento y visualización de la información biomecánica se utilizaron sensores inerciales MPU-9250 y Arduino Nano 33BLE para formar nodos Bluetooth. La información capturada se almacenó en un servidor formado por una Raspberry Pi.

Se implementaron dos maneras de visualización de la información en tiempo real, mediante un programa de visualización dentro del servidor y otra para acceso remoto mediante la implementación de MQTT.

Se construyó un set de datos con personal que no presentan alteraciones de la marcha y se procesaron los datos usando el programa iGait, para posteriormente comparar características de la marcha obtenida con sistemas ópticos de análisis de la biomecánica.

Capítulo 1

Generalidades

1.1. Introducción

El número de dispositivos conectados está creciendo exponencialmente formando la llamada Internet de las Cosas (IoT), una gran red de redes que conecta dispositivos inteligentes como sensores y actuadores. Estos dispositivos cubren aplicaciones en varias áreas tales como salud, redes inteligentes, transporte inteligente, gestión de residuos, casas inteligentes, ciudades inteligentes, agricultura, gestión de la energía, etc [8].

Los desarrollos recientes en sensores biomédicos, sistemas de comunicación inalámbrica y redes de información están transformando los sistemas de atención médica convencionales. Estos sistemas de atención médica están poniendo a disposición servicios de atención distribuidos para pacientes que pueden no estar ubicados junto con los proveedores de atención médica, brindando diagnósticos tempranos y reduciendo el costo en la sección de atención médica. Los desarrollos en la Internet médica de las cosas (m-IoT) permitirían una gama de aplicaciones, incluida la monitorización remota de la salud a través de dispositivos portátiles de grado médico para brindar atención domiciliaria a personas mayores; interacción virtual médico-paciente para tener acceso en cualquier momento y lugar a los profesionales médicos, examen endoscópico inalámbrico y cirugía robótica operada a distancia para ampliar el acceso a cirujanos altamente calificados [18].

La monitorización remota de la salud es un componente clave en los modelos actuales de salud pública y es definida dentro del eSalud o eHealth en inglés, como una herramienta esencial para disminuir los costos sanitarios y

aumentar la calidad de vida de la población [33]. Es en este contexto toma relevancia el monitoreo y la obtención continua de datos para un pos procesamiento de información y la integración con otros sistemas inteligentes. Actualmente la evolución en la integración con otros objetos, miniaturización, autonomía y algoritmos de los denominados dispositivos *wearable* permite que se desarrollen campos como el reconocimiento de actividades para el cuidado de la salud de pacientes [13][39][49].

1.2. Planteamiento del problema

1.2.1. Definición del problema

El crecimiento de la población a nivel mundial, no se traduce de manera directamente proporcional en un aumento en la calidad de vida de la misma. Esto, aunado a otros factores como el envejecimiento de la población, el desigual acceso a oportunidades, la pobreza, entre otros; se traduce en un reto para las disciplinas encargadas de proveer soluciones tecnológicas a este tipo de problemas sociales. Por tanto, impactar en la calidad de vida de las personas es un reto y una responsabilidad, donde el rol de la tecnología es esencial. Desde este punto de partida, es que en esta propuesta, se plantea la relevancia de contribuir, mediante el entendimiento profundo de los sensores aplicados, a la mejora de la calidad de vida de la población. En este sentido, son las pruebas y los procedimientos de diagnóstico, las herramientas vitales que ayudan a los médicos a confirmar o descartar la presencia de un trastorno neurológico u otra enfermedad [46].

Hace un siglo, la única manera de hacer el diagnóstico positivo de muchos trastornos neurológicos era realizar una autopsia después de la muerte del paciente. En la actualidad, las décadas de investigación básica sobre las características de las enfermedades de este tipo, el desarrollo de técnicas que permiten a los científicos observar dentro del cerebro humano y monitorear la actividad del sistema nervioso a medida que ocurre; le han dado a los médicos herramientas para diagnosticar enfermedades y estudiar la eficacia y eficiencia de una terapia en particular[42].

Sin embargo, la mayoría de los procedimientos de medición y/o monitoreo, se realizan en entornos controlados y acondicionados que requieren de instalaciones hospitalarias y personal de la salud altamente capacitado, lo cual

requiere de una inversión del sistema de salud para la adquisición, operación y mantenimiento, incrementando los costos del sistema de salud. Según el informe de la OMS¹, OMC² y OMPI³ [5] , la distribución de los recursos humanos capacitados para tal fin, como así también la tecnología para la detección, se encuentran concentrados en los grandes centros urbanos desfavoreciendo a las poblaciones geográficamente alejadas y reproduciendo el modelo de desigualdad social al obligar a trasladarse para obtener el servicio de salud. Esto presenta el primer problema detectado.

Por otro lado, actualmente el análisis cuantitativo de la marcha humana es reconocido como una herramienta de investigación y enseñanza, además de tener grandes aplicaciones en el campo clínico para el estudio y tratamiento de sus alteraciones [19].

1.2.2. Delimitación de la investigación

Alcances

- El prototipo desarrollado consta de un conjunto de hardware. Por un lado la red corporal consta de un grupo de 3 nodos Bluetooth los cuales contienen sensores del tipo acelerómetros, giroscopios y magnetómetros.
- El hardware se complementa con un servidor que utiliza una SBC ⁴.
- Se desarrolla un firmware prototipo que se ejecuta en los nodos para las tareas de captura, almacenamiento y transmisión de la información.
- Se desarrolla una aplicación prototipo que se ejecuta en el servidor para gestión de la comunicación Bluetooth, gestión de dispositivos IoT, procesamiento de la información, almacenamiento y visualización de los datos.
- El prototipo muestra los datos en tiempo real y genera un conjunto de datos exportable en formato C3D, inglés *Coordinate 3D file*.

¹Organización Mundial de la Salud

²Organización Mundial de Comercio

³Organización Mundial de la Propiedad Intelectual

⁴SBC: del inglés computadora de placa única.

Delimitaciones

- La validación se realiza con personas que no presentan patologías que alteren la marcha normal con edades de entre 20 a 63 años.
- El conjunto de dato contiene información de 3 nodos colocados en diferentes partes del tren inferior de los sujetos analizados.
- Se valida la biomecánica de la marcha de las personas software de análisis visual de biomecánica y filmaciones de los pacientes del plano sagital.
- Como el proyecto es de código abierto bajo licencia Apache 2.0 con fines no comerciales, el uso de APIs, librerías y Software que forman parte del desarrollo son de código abierto.

1.3. Pregunta de investigación

¿Como capturar información de la biomecánica de la marcha usando sensores inerciales y lograr la integración de los mismos con otros sistemas IoT?

1.4. Hipótesis

Es posible medir la biomecánica de la marcha mediante un sistema IoT de arquitectura abierta que utilice sensores inerciales.

1.5. Objetivo general

Desarrollar un sistema IoT que sea capaz de capturar la biomecánica de la marcha humana utilizando sensores inerciales.

1.6. Objetivos específicos

- Desarrollar un conjunto de sensores corporales de plataforma abierta.
- Construir una red inalámbrica que conecte los sensores corporales entre ellos.

- Construir un set de datos que contenga mediciones inerciales de los sensores.

1.7. Justificación

El desarrollo de investigaciones en el campo de sensores corporales para la captura de movimiento de uso en aplicaciones médicas, permite conocer y evaluar aplicaciones existentes. Adicionalmente, el uso de arquitecturas abierta en sistemas *wearables* de hardware y software, permite la exploración de nuevos algoritmos o mejora de los existentes orientados a crear dispositivos cada vez más inteligentes, optimizar los recursos y evaluar nuevas aplicaciones.

La captura de información biomecánica de la marcha para uso médico se circunscribe a espacios de laboratorios de la marcha ubicados en hospitales y centros de rehabilitación de los centros urbanos, donde se debe disponer de equipamiento de captura y de personal entrenado. Por esto, es relevante contar con un sistemas autónomos de captura que no dependa de personal técnico especializado y de un entorno con instalaciones especiales.

Actualmente los profesionales especializados en la valoración de la marcha tienen diferentes manera de realizar esta tarea, siendo la principal herramienta la valoración visual. Se basan en tablas y estándares promedio para generar información de diagnóstico o evaluación de un paciente.

El presente trabajo propone el desarrollo de un prototipo de red de sensores corporales para la captura de los movimientos de la marcha de manera autónoma y de arquitectura abierta.

1.8. Metodología utilizada

Esta investigación es de tipo descriptiva, narrativa, experimental y cuantitativa.

- Estudio inicial
 - Caso de estudio: análisis de la biomecánica de la marcha y marcha normal.
 - Tecnologías de redes corporales, determinar arquitectura a usar.

- Caracterización de sensores
 - Construcción de prototipo - Red de sensores y Gateway
 - Evaluar y calibrar sensores con patrones.
 - Pruebas y test sobre Gateway simulando sensores y conexiones.
 - Medición de rendimiento de algoritmos de filtrado usados en la fusión de datos.
 - Ensayo comparativo con sistemas ópticos open source como Kinovea, Skillspector, KWON 3d, usando test de la marcha 6MWT en pacientes sanos.

Capítulo 2

Marco teórico

2.1. Conceptos y teorías fundamentales de la investigación

En la sección 2.1.1 se comienza con la descripción de la biomecánica de la marcha con el fin de comprender la problemática.

2.1.1. Biomecánica de la marcha

La biomecánica de la marcha es un campo que describe la cinemática y dinámica de la caminata de una persona. Es importante describir el problema para poder comprender y actuar sobre el sistema. En esta sección se hace una descripción de la biomecánica de la marcha normal, es decir que el movimiento no se ve afectado por factores extrínsecos y intrínsecos como pueden ser superficie del suelo, inclinación o patologías que la persona pueda tener.

Marcha humana normal

El hombre se desplaza por medio de sus extremidades inferiores, la marcha es el modo de locomoción en el cual el sujeto nunca deja el suelo y mantiene un equilibrio dinámico. Cuando una persona camina el cuerpo puede asimilarse a una masa sometida a traslaciones y que sufre la acción de la gravedad,

inerencia y aceleración.

Una persona necesita tener una postura estable de bipedestación antes de comenzar la marcha. Una vez conseguido el equilibrio de bipedestación puede iniciarse la marcha o capacidad de mantener un paso rítmico y estable.

La marcha es el resultado de una sucesión de pasos, siendo un paso un conjunto de movimientos que se producen entre el apoyo de un talón y el sucesivo apoyo del talón contra lateral.

Ciclo de la marcha

El ciclo de la marcha comienza cuando se apoya el talón en el suelo y termina con el siguiente contacto de talón del mismo pie en el suelo. También se puede decir que es el intervalo comprendido entre dos choques de talón sucesivos del mismo pie.

En la figura 2.1 se muestra el ciclo de la marcha. Este se comprende de dos fases, la fase de apoyo o portante que abarca el 60 % del ciclo y la fase de oscilación o balanceo que es el 40 % restante. Un ciclo está compuesto de dos pasos donde dos pasos forman una zancada, por lo que en una zancada cada pie pasa por una fase de apoyo y una fase de oscilación, tal como se muestra en la figura 2.2.

Fase de apoyo

La fase de apoyo se inicia con el contacto o choque de talón con el suelo y termina con el despegue del antepié.

A su vez la fase de apoyo se puede subdividir en diferentes subfases que son las siguientes:

- Fase de contacto inicial (CI): Se presenta del intervalo del 0 % al 2 % de la marcha. Conlleva al contacto del pie con el suelo para iniciar el apoyo a través del talón produciendo así la rodadura del pie hacia abajo apoyado en el talón. La cadera está flexionada, la rodilla en extensión y el tobillo está en posición neutra.
- Fase inicial de apoyo o de respuesta inicial a la carga (AI): Mantiene una progresión suave manteniendo la estabilidad mediante la amortiguación del descenso del cuerpo, y abarca del 0 % al 10 % del ciclo de la marcha. Se presenta una desaceleración de la masa corporal controlada por la flexión de la rodilla y la flexión plantar del tobillo.
- Fase media de apoyo (MA): Comienza con el despegue de dedos del

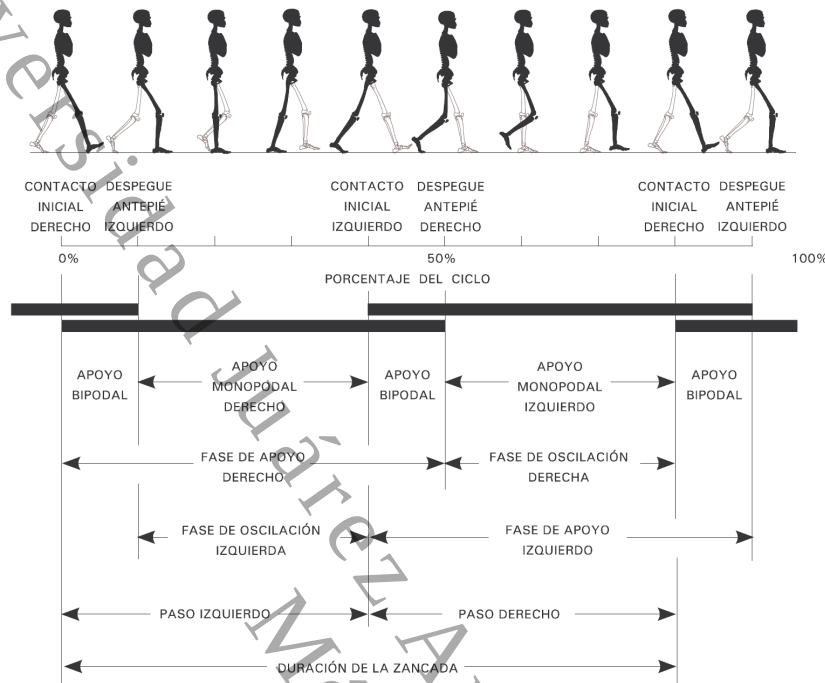


Figura 2.1. Fases de la marcha humana. Fuente: [22]

miembro contralateral y se prolonga hasta el despegue de talón se presenta en el intervalo comprendido entre el 10 % y el 30 % de la marcha como se muestra en la figura 2.3. En esta fase el cuerpo avanza sobre un pie estacionario, efectuándose en el tobillo una dorsiflexión, para rotar en torno a él como si fuera un rodillo, manteniendo la estabilidad de la cadera y rodilla.

- Fase final de apoyo (AF): En la figura 2.3 se observa como la AF comienza con el despegue de talón y finaliza con el contacto de talón del miembro contralateral. Está comprendida en el intervalo del 30 % -50 % del ciclo. Se despega el talón y se comienza a rotar sobre el antepié y el cuerpo sobrepasa al pie de soporte y la rodilla termina su extensión y comienza a flexionarse, la cadera continua extendiéndose y finalmente el miembro contralateral contacta con el suelo con el talón.
- Fase previa de oscilación (OP): En esta fase se inicia el segundo periodo de doble apoyo, en esta fase el miembro se prepara para la oscilación

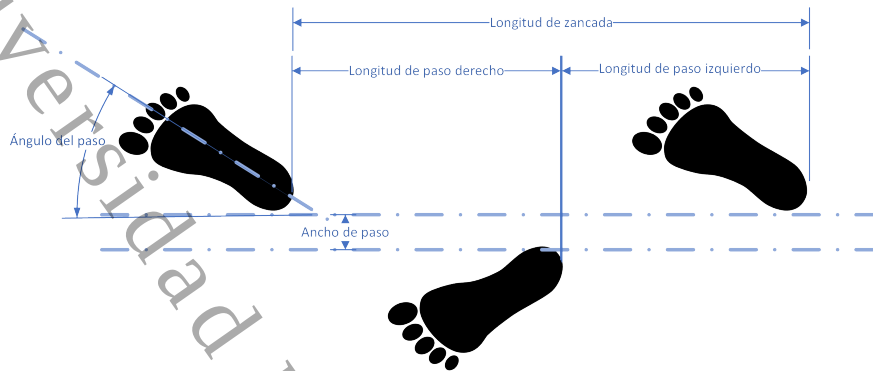


Figura 2.2. Parámetros de la marcha humana

o balanceo, la cual es suscitada por la entrada en carga del miembro contralateral, hacia el cual se transfiere la carga. Se presenta una flexión plantar del tobillo y de rodilla que caracterizan a esta fase, que se presenta en el 50 % al 60 % de la marcha.

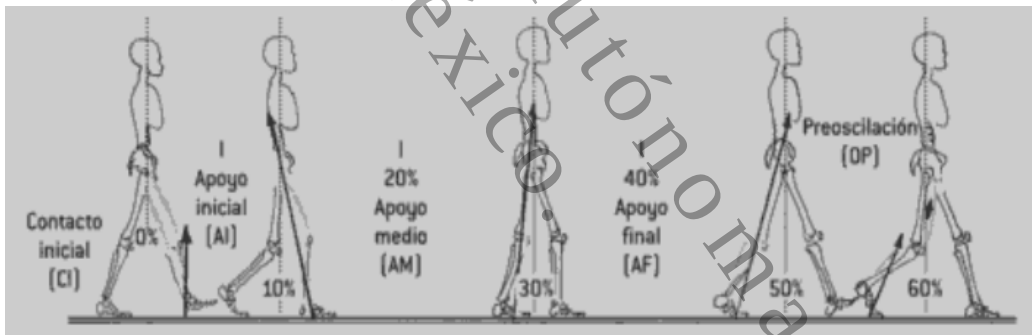


Figura 2.3. División de fase de apoyo. Fuente: [22]

Fase de Oscilación

En esta fase el miembro que se despegó del suelo se desplaza hacia adelante. La fase de oscilación se subdivide en las siguientes subfases:

- Fase inicial de oscilación (OI): Comienza con el despegue del antepié, el pie está en el aire y el avance del pie se produce por la flexión de cadera y rodilla, para lograr una cadencia adecuada y finaliza cuando el pie que oscila alcanza al miembro contralateral que se encuentra en

fase media de apoyo. Esta fase está comprendida entre el intervalo del 60 % al 73 % del ciclo de la marcha.

- Fase media de oscilación (OM): Comprendida entre el 73 % al 87 % del ciclo. Se inicia cuando los dos miembros se cruzan y finaliza con la posición vertical de la tibia oscilante, una vez sobrepasado el miembro de apoyo, como se muestra en la figura 2.4. Al final de esta fase, la cadera alcanza su flexión máxima y la posición de la pierna es casi vertical y la extensión de la rodilla continúa.
- Fase final de oscilación (OF): Con el contacto del miembro con el suelo finaliza el ciclo, dando paso a una zancada, como se muestra en la figura 2.4. La flexión de rodilla y cadera se frenan, para así cumplir con los objetivos de decelerar la pierna y preposicionar correctamente el pie para contactar con el suelo.

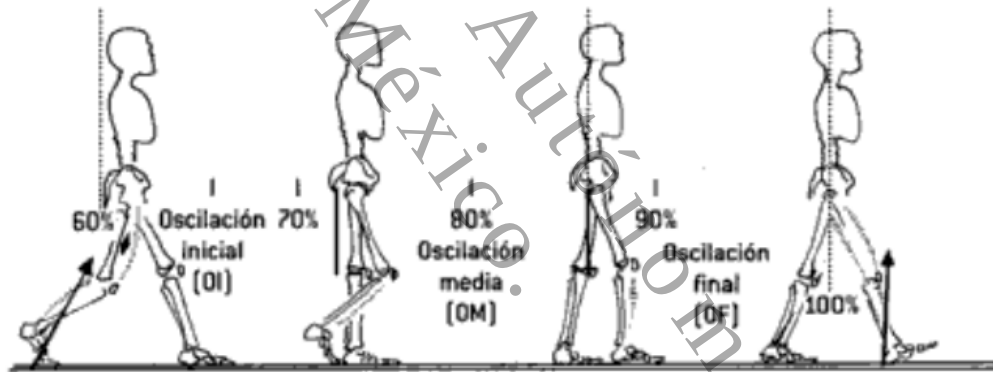


Figura 2.4. División de fase de oscilación. Fuente: [22]

2.2. Literatura relacionada

El análisis del movimiento humano (HMA¹) utilizando redes corporales (BSN²) de sensores inerciales (IMU³), se ha convertido en un área de investigación indispensable para aplicaciones de uso médico [29]. Uno de los

¹HMA: Human Movement Analysis

²BSN: Body Sensor Networks

³IMU: Inertial Measurement Unit

objetivos principales en esta área de investigación se enfoca en la creación de nuevos algoritmos que permitan tener mejor calidad de datos capturados. Otro objetivo a destacar es el estudio de nueva tecnología inalámbrica de conexión entre sensores orientados al incremento de la velocidad de comunicación e internet de las cosas (IoT) [45].

2.2.1. Arquitectura de BSN y Nodo de sensores

Con el fin de aumentar la precisión de la información capturada, los autores han explorado distintas arquitecturas con sensores como son: nodos con solo un sensor de aceleración [50], nodos con un sensor de aceleración, velocidad angular y/o campo magnético [41] [14] nodos con arreglo de sensores de aceleración, velocidad angular y campo magnético [43][44] [32]. Los nodos con solo un sensor de aceleración generan información insuficiente para calcular parámetros precisos de posición angular. Análogamente, los nodos con arreglo de sensores presentan errores significativos tales como no linealidades, sensibilidad g y errores de escala, que son dependientes de la dinámica y requieren de algoritmos mas complejos para obtener un resultado [43]. Actualmente las investigaciones se han enfocado en la utilización de sistemas con un sensor de aceleración, velocidad angular y/o campo magnético.

En el trabajo “A Wearable Inertial Measurement Unit for Long-Term Monitoring in the Dependency Care Area” [41], los autores presentan un sistema de adquisición de dato utilizando el conjunto de sensores inerciales tales como, acelerómetro triaxial, girómetro triaxial y magnetómetro triaxial. El sistema captura toda la información y la almacena en una memoria uSD⁴. Mediante un módulo Bluetooth 2.0 genera una conexión punto a punto con un dispositivo, como puede ser una PC, tablet o smartphone, para transmitir los datos capturados. La frecuencia de muestreo se estableció variable desde 1Hz hasta 200Hz. Se logró una autonomía energética del sistema de 17,94hs para una frecuencia de muestreo de 200Hz y transmisión continua inalámbrica hasta 45hs para una frecuencia de captura de 40Hz sin transmisión por Bluetooth. En el trabajo “A Wearable System for Gait Training in Subjects with Parkinson’s Disease” [14], los autores describen la plataforma de sensores EXL-s1⁵

⁴uSD: micro Security Card

⁵url: <http://www.exelmicroel.com/download/EXLs1.pdf>

utilizada en el proyecto CuPID⁶. Esta plataforma comercial cuenta con un conjunto de sensores inerciales comprendidos por un acelerómetro triaxial, un girómetro triaxial y un magnetómetro triaxial. Este sistema contiene una memoria NAND⁷ interna de 1Gb para almacenar la información capturada y comunicación inalámbrica con Bluetooth 2.1. Los resultados arrojaron una autonomía de 6hs típica para una frecuencia de muestreo de 100Hz y transmisión continua de datos por Bluetooth. Adicionalmente se evaluó el sistema con la plataforma MTw⁸, recolectándose datos de ambos sensores de manera estática durante 10 segundos con el fin de medir el ruido producidos por los sensores.

2.2.2. Transmisión de la información, tipología de redes

Los estándares más utilizados para transmitir los datos capturados [23]: IEEE 802.15.4, ANT+ y Bluetooth.

Los autores de “A Survey on Wearable Sensor-Based Systems for Health Monitoring and Prognosis” [34] realizan una revisión del estado del arte de las diferentes arquitecturas y sistemas de sensores para monitoreo de la salud. Se analizaron 19 plataformas de monitoreo de la salud donde se describieron los módulos de comunicación, el tipo de señales que se miden y la aplicación médica de cada sistema. Se establecieron 16 características de evaluación y tres grupos (paciente, médico y manufactura) que se calificaron estableciéndose una categorización según los pesos asignados. Las características de las 16 analizadas con mayores pesos son: seguridad de la información en la comunicación de las redes BAN, miniaturización, eficiencia energética, validaciones clínicas y estandarización.

2.2.3. Algoritmos, filtros, fusión y almacenamiento de datos

Con el fin de eliminar ruido y error de deriva, es necesario que las mediciones obtenidas de los sensores sean preprocesadas fusionando los valores

⁶CuPID: Close-loop system for personalized and at-home rehabilitation of people with Parkinson’s disease. <http://www.cupid-project.eu>

⁷Memoria NAND: Es un tipo de memoria construida con compuestas NAND

⁸MTw de la firma: Xsens, Enschede, Netherlands

de distintos sensores. Diferentes estudios se caracterizan por abordar alguna de las siguientes estrategias mas utilizadas [10]: filtros complementarios (CF)[25], filtro de Kalman (KF)[26], filtro de Kalman extendido (EKF)[28]. Todos estos métodos se caracterizan por ser utilizados en procesamiento en tiempo real, sin embargo presentan diferente costo computacional y calidad de resultados obtenidos.

En el trabajo “Benchmarking Foot Trajectory Estimation Methods for Mobile Gait Analysis” [24], los autores utilizan tres métodos: filtro de Madgwick(MF), filtro de Euston (EF) e Integración de giroscopio con el fin de obtener valores de estimación de la orientación del pie. Para todos los casos se utiliza representación de cuaterniones y se comparan los resultados con mediciones con el sistema VICON ⁹. Los resultados obtenidos muestran los valores óptimos de configuración de los parámetros para cada filtro y a su vez se observa que el EF es menos sensible al movimiento comparado con el MF. Basado en los resultados se establece un ranking de performance para cálculo de la estimación de la orientación, teniendo a MF como el mejor y siguiendo el método de integración de giroscopio y por último EF.

Finalmente, en los trabajos “A Mobile Kalman-Filter Based Solution for the Real-Time Estimation of Spatio-Temporal Gait Parameters” [20] y “An Extended Kalman Filter to Estimate Human Gait Parameters and Walking Distance” [9], los autores aplican filtros estocásticos como KF y EKF respectivamente.

En el primer trabajo, se utiliza el algoritmo de KF como método de fusión y cálculo de parámetros espacio-temporales. En este, se implementa un sistema de análisis de la marcha de zero-velocity-update basado en KF. Los autores desarrollaron una aplicación de Android para análisis de la marcha en tiempo real. A su vez, el sistema es ensayado en tres velocidades distintas (4,1Km/h, 5,6Km/h y 2,9Km/h) obteniendose una diferencia de RMS del 2,9 % respecto del patrón tomado de GAITRite¹⁰.

Para el segundo caso, se utiliza el algoritmo de EKF para el cálculo de ángulo de articulación y distancia recorrida obtenidos de la captura por medio de sensores IMU. En este trabajo, se implementa un EKF capturando los datos de dos sensores colocados en la tibia y femur. Los datos son comparados con el sistema VICON y se observa una precisión del 97 % para el desplazamiento

⁹VICON: Sistema óptico de captura de movimiento para biomecánica y deportes. url: <https://www.vicon.com/motion-capture/biomechanics-and-sport>

¹⁰url: <https://www.gaitrite.com/>

lineal y un RMSE promedio de 7cm. Con esta información es posible decir que el algoritmo implementado de EKF brinda alta precisión para la configuración de sensores ensayadas.

El almacenamiento de datos en sistemas IoT es especialmente relevante por el volumen de datos que se generan en fracciones de tiempo y porque se debe garantizar la integridad de los mismo. A los tradicionales gestores de base de datos como MySQL¹¹, SQLite¹² y PostgreSQL¹³ surgen soluciones MongoDB¹⁴, InfluxDB¹⁵ o DynamoDB¹⁶ que se las conoce como base de datos NoSQL. En el artículo “MySQL and NoSQL database comparison for IoT application” [40] se analiza el rendimiento de una base de datos MySQL y una MongoDB con datos generados por una aplicación de Internet de las cosas. En el artículo se observa que MongoDB requiere menos tiempo de respuesta en comparación con MySQL. Por otro lado, las respuestas de MySQL fueron estables en comparación con MongoDB. Finalmente se concluye que es fundamental determinar que consulta se usará principalmente en la aplicación IoT porque de eso determinará que sistema seleccionar para una aplicación determinada.

2.3. Marco tecnológico

En la presente sección se desarrollan los conceptos de Sistemas micro electromecánicos, sensores inerciales, redes de sensores y algoritmos para representar y manipular los datos de sensores inerciales.

2.3.1. MEMS

Sistemas micro electromecánicos (MEMS¹⁷) son dispositivos de tamaño microcópico del orden de 0.001 a 0.1 mm, que contienen partes móviles [48]. Los MEMS construidos en tecnología de semiconductores y sus aplicaciones son variadas desde sensores hasta actuadores microscópicos y micromotores.

¹¹MySQL: <https://www.mysql.com>

¹²SQLite: <https://www.sqlite.org/index.html>

¹³PostgreSQL: <https://www.postgresql.org/>

¹⁴MongoDB: <https://www.mongodb.com/>

¹⁵InfluxDB: <https://www.influxdata.com/>

¹⁶DynamoDB: <https://aws.amazon.com/es/dynamodb>

¹⁷MEMS: Microelectromechanical systems

Acelerómetro

Los acelerómetros MEMS son dispositivos semiconductores transductores, capaces de medir aceleraciones lineales y entregar proporcionalmente una señal analógica o digital, según el modelo y fabricante [52]. Actualmente los fabricantes ofrecen un dispositivo que integra los 3 ejes (x, y, z) en una sola pastilla de silicio facilitando su integración en los desarrollos.

Girómetro

Los girómetros o giróscopo MEMS, utilizado indistintamente en la literatura, son dispositivos semiconductores transductores, capaces de medir velocidades angulares y entregar proporcionalmente una señal analógica o digital [52]. Según el modelo y fabricantes son las características técnicas que ofrecen al mercado pero actualmente es común encontrar un girómetro que integre en una misma pastilla de silicio los 3 ejes (x, y, z).

Magnetómetro

Los magnetómetros MEMS son transductores contruidos con tecnología de semiconductores de tamaño reducido, capaces de tomar la intensidad de un campo magnético y entregar una señal proporcional a este [52]. Actualmente se encuentran disponibles en el mercado magnetómetros capaces de medir en los 3 ejes (x, y, z) integrados en un solo chip.

2.3.2. Computadora de placa única

Las computadoras de placa única (SBC¹⁸) son sistemas cómputo completos integrados en una sola tarjeta de circuito impreso (PCB¹⁹). Las SBC integran microprocesador/s, memoria/s, puertos de entrada y salida con características que permiten aplicar esta tecnología al desarrollo de sistemas y prototipos, educación y como un sistema embebido. Los ejemplos de SBC más conocidos de hardware de fuente abierta son los proyectos Arduino²⁰,

¹⁸SBC: Single-board computer

¹⁹PCB: Printed circuit board

²⁰url: <https://www.arduino.cc/>

Raspberry Pi²¹, BeagleBoard²² y Banana Pi²³, entre otros.

2.3.3. Redes de sensores inalámbricas

Es posible definir una red de sensores inalámbrica (WSN²⁴) como una cantidad de dispositivos (nodos de sensores), autónomos, distribuidos físicamente e instalados alrededor de uno o varios fenómenos que se desea monitorear. Estos dispositivos tienen la capacidad de capturar, almacenar y comunicar datos en una red en forma inalámbrica[7].

Redes inalámbricas corporales

Las redes inalámbricas corporales (WBAN²⁵) son de especial interés en el ámbito de tecnología médica. Se pueden definir como dispositivos que se interconectan y que se encuentran físicamente en el cuerpo con el objeto de generar un beneficio al usuario[35]. El objetivo principal de estas redes es disminuir la carga en los hospitales y proveer a los paciente un monitoreo remoto de la salud. Existen una variedad de tecnologías actuales para la comunicación como Bluetooth[1], Zigbee[4], WiFi[2], etc., pero independiente de estas, desde el año 2012 existe la norma IEEE 802.15.6[3] creada para la estandarización de las WBAN.

Protocolo Message Queuing Telemetry Transport

El protocolo Message Queuing Telemetry Transport (MQTT) desde la versión 3.1.1 forma parte como estandar OASIS²⁶ y desde 2016 es estandar ISO/IEC 20922²⁸. MQTT es un protocolo de conectividad M2M²⁹/IoT.

²¹url: <https://www.raspberrypi.org>

²²url: <https://beagleboard.org>

²³url: <http://www.banana-pi.org>

²⁴WSN: Wireless sensor networks

²⁵WBAN: Wireless body area network

²⁶OASIS: Por sus siglas en Ingles de Organización para el avance de los estándares de información estructurada. Es un consorcio sin fines de lucro que impulsa el desarrollo, la convergencia y la adopción de estándares abiertos para la sociedad global de la información.

²⁷<https://www.oasis-open.org/news/announcements/mqtt-version-3-1-1-becomes-an-oasis-standard>

²⁸ISO/IEC 20922: <https://www.iso.org/standard/69466.html>

²⁹M2M: Protocolo Máquina a Máquina por sus siglas en ingles.

Se diseñó para transporte de mensajes de publicación/suscripción extremadamente ligero. Actualmente se encuentra en la versión v5³⁰. El estandar MQTT presenta implementaciones, tanto para sensores como para servidores o *Brokers* que son propietarias y otras que son open source. En el caso de implementaciones de *Brokers* de código abierto se identifican a los principales: Mosquitto de Eclipse Foundation³¹, EMQ (Erlang MQTT Broker)³², ActiveMQ y ActiveMQ Artemis de Apache Foundation³³ [36].

Tecnología Bluetooth

Bluetooth actualmente es un estándar IEEE (IEEE 802.15.1[1]) de comunicaciones inalámbricas desarrollado por Bluetooth Special Interest Group (SIG) y diseñado para intercambio de información en distancias cortas. El espectro de frecuencias de transmisión se encuentra entre 2.402 GHz to 2.480 GHz con 79 canales de 1MHz de ancho de banda por canal. Dependiendo de la clase de cada dispositivo, las potencias de transmisión se encuentran entre 0,5mW para clase 4 hasta 100mW para dispositivos clase 1 y alcances entre 0,5m a 100m respectivamente. En Abril del 2016 se presenta el release 5.0 que incluye mejoras orientadas a IoT. A partir de la versión 4.0 se incluyen modificaciones como la reducción a 40 canales y ampliación a 2MHz el ancho de banda de cada canal y se incorpora la característica de LE³⁴ o de bajo consumo, haciendo factible su utilización en WBAN de uso médico según[35].

Topología de red en estrella - BLE

Bluetooth Low Energy (BLE) utiliza la topología de red de tipo estrella, donde los dispositivos Maestros pueden tener varias conexiones de capa de enlace con periféricos Esclavos y simultáneamente realizar búsquedas de otros dispositivos. Los dispositivo en rol de Esclavo solo puede tener una conexión de capa de enlace con un único Maestro.

Bluetooth LE define una pila para gestionar los dispositivos, la conexión y la

³⁰Estandar OASIS MQTT v5: <https://docs.oasis-open.org/mqtt/mqtt/v5.0/mqtt-v5.0.html>

³¹Mosquitto: <https://mosquitto.org/>

³²EMQ: <http://emqt.io/>

³³ActiveMQ y ActiveMQ Artemis: <https://activemq.apache.org/>

³⁴Low energy por sus siglas en Inglés

interfaz de las aplicaciones, para esto se divide en tres partes básicas según el estándar: Controller, Host y Applications.

2.3.4. Representación y manipulación de los datos obtenidos

La representación y manipulación de los datos se refiere a la forma con la que los valores obtenidos de los sensores son modificados aplicando diferentes métodos o herramientas matemáticas para facilitar y minimizar errores en la implementación. Las principales herramientas que se utilizan son la representación en cuaternios o cuaterniones, matriz de rotación, representación en Ángulos de Euler y filtros digitales.

Matrices de rotación

Las matrices de rotación o Matrices de cosenos directores no son más que un conjunto de matrices que permiten hacer la rotación de vectores. Formalmente una matriz de rotación, es una matriz que representa la rotación en el espacio euclideo. Un ejemplo de esto son las siguientes matrices que representan la rotación en un espacio de tres dimensiones, de un vector a través de los ejes x , y , z :

Cuaterniones

Los cuaterniones o cuaternios son números complejos de cuatro dimensiones. Pueden ser utilizados como alternativa para representar y realizar operaciones como integración e interpolación, que son incompatibles al usar una matriz de rotación[17].

Ángulos de Euler

La representación de cantidades espaciales en forma de ángulos de Euler es una manera intuitiva que describe la orientación como ángulos de rotación ϕ , θ y ψ , respecto a los ejes x , y y z . Utilizadas originalmente para describir la orientación de un sistema de referencia solidario con un sólido rígido en movimiento.

Filtro digitales

Son ecuaciones matemáticas traducidas en algoritmos que permiten manipular señales discretas de forma digital. En general se utiliza para atenuar o amplificar un frecuencia determinada o rango. También son utilizados para la fusión de datos o sensores como son el caso del Filtro complementario[25], Filtro de Kalman[53], Filtro de Kalman extendido[53], Filtro de Mahony-Madgwick[30], entre otros.

Capítulo 3

Modelado de la biomecánica de la marcha

El presente capítulo tiene por objetivo obtener el modelo del filtro digital sintonizado, es decir las ecuaciones y parámetros que serán utilizados en el capítulo 4.5 para ser implementado en el servidor. Para realizar las simulaciones del filtro es necesario tener un modelo matemático cinemático de la marcha para luego introducirlo en la herramienta Simulink de MATLAB versión 2019b.

3.1. Modelo matemático

3.1.1. Péndulo invertido simple

El modelo más simple de caminata es un péndulo invertido que oscila (cae y se eleva); si éste se coloca sobre un carro que pueda moverse en dirección horizontal, se obtiene una primera analogía sencilla de la caminata. El centro de masas del cuerpo será análogo a la masa del péndulo, y el centro de presiones del pie será análogo al pivote del péndulo (3.1).

3.1.2. Péndulo invertido doble

Durante la fase inicial de apoyo o de respuesta a la carga, la masa corporal se desacelera controlando la flexión de la rodilla y el tobillo realiza una flexión

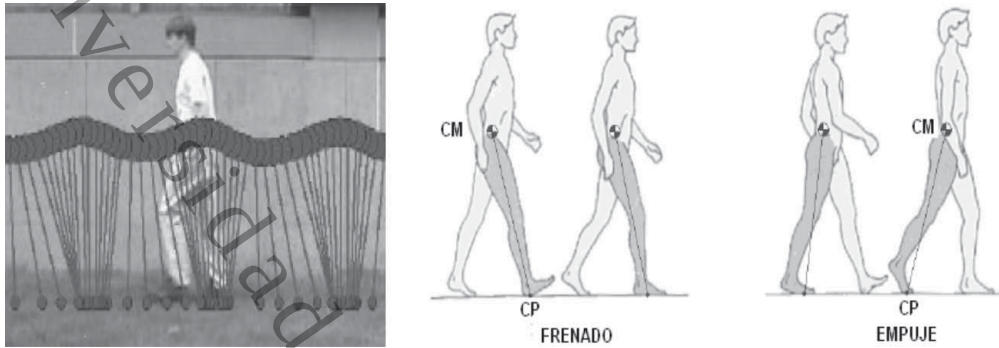


Figura 3.1. Modelo del péndulo simple invertido. Fuente: [38]

plantar; se continúa después con la fase media de apoyo, en la cual ambos péndulos se encuentran en posición vertical (extensión de rodilla y cadera); se culmina con la fase final de apoyo, en la cual se produce una nueva flexión de la rodilla; en esta última fase, la función principal del miembro es prepararse para la oscilación o balanceo (3.2).



Figura 3.2. Modelo del péndulo doble invertido. Fuente: [12]

3.1.3. Péndulo invertido simple de dos ejes

Los modelos presentados son descriptos mediante ecuaciones diferenciales y cuanto mayores grados de libertad es mayor la complejidad de las ecuaciones que describen el movimiento. En el caso del modelo de péndulo simple

invertido permite una sencilla manipulación pero es una aproximación imprecisa. Por otro lado el modelo del péndulo invertido doble se aproxima a la biomecánica de la marcha pero el conjunto de ecuaciones son mas complejas. Un modelo que puede generar una aproximación de mejor calidad que el péndulo invertido simple y de complejidad menor que el modelo del péndulo invertido doble es el modelo del péndulo invertido simple de dos ejes. En el presente trabajo se optó por utilizar este modelo para realizar las simulaciones y estimar los valores de sintonía del filtro.

3.1.4. Descripción de modelo de péndulo invertido simple de dos ejes

Un péndulo de 2 ejes fue el tema elegido para la simulación y los estudios experimentales, ya que se aproxima a varias articulaciones de manipulación humana y robótica y luego el movimiento oscilatorio es análogo a la marcha y excita los 6 DoF¹ del cuerpo suspendido. El movimiento amortiguado da como resultado tasas de magnitud baja y alta y el movimiento complejo de la articulación de 2 ejes da como resultado una cinemática rotacional en los tres ejes del cuerpo suspendido. El péndulo está representado por el esquema que se muestra en la figura 3.3, donde θ_1 y θ_2 representan el ángulo en cada eje de la articulación. Las ecuaciones de movimiento pendulares de 2 ejes se han derivado como las ecuaciones 3.1 y 3.2 donde c_1 y c_2 son la amortiguación de cada eje de la articulación.

$$\ddot{\theta}_1 = 2\dot{\theta}_1\dot{\theta}_2 \tan \theta_2 - \frac{g \sin \theta_1}{l \cos \theta_2} - \frac{c_1 \dot{\theta}_1}{m \cos \theta_2^2} \quad (3.1)$$

$$\ddot{\theta}_2 = -\dot{\theta}_1^2 \cos \theta_2 \sin \theta_2 - \frac{g}{l} \cos \theta_1 \sin \theta_2 - \frac{c_2 \dot{\theta}_2}{m} \quad (3.2)$$

La cinemática del cuerpo está calculada como la cinemática directa descrita por las ecuaciones 3.3 y 3.4 donde ${}^B a$, ${}^B \omega$ y ${}^B \dot{\omega}$ son la aceleración traslacional, la velocidad angular y aceleración angular respectivamente del marco del cuerpo en dirección relativa al marco del cuerpo. ${}^E R$ describe la orientación del marco del cuerpo relativo al marco de la tierra.

¹DoF: Grados de libertad por sus siglas en inglés

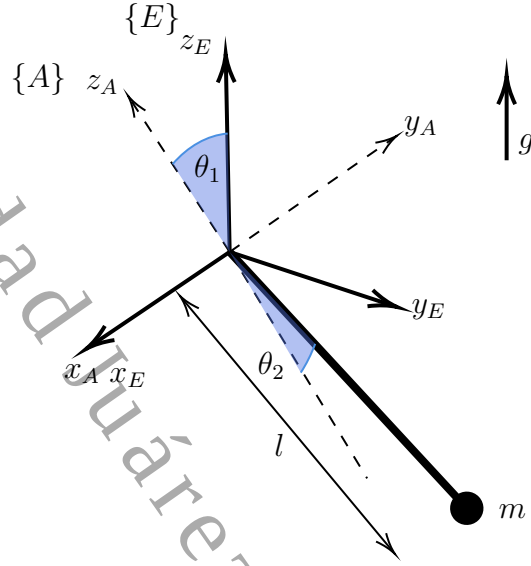


Figura 3.3. Esquema de péndulo invertido de 2 ejes

$${}^B a = g \begin{bmatrix} -\sin \theta_2 \\ \sin \theta_1 \cos \theta_2 \\ \cos \theta_1 \cos \theta_2 \end{bmatrix} \quad (3.3)$$

$${}^B \omega = \begin{bmatrix} \dot{\theta}_1 \cos \theta_2 \\ \dot{\theta}_2 \\ \dot{\theta}_1 \sin \theta_2 \end{bmatrix} \quad (3.4)$$

$${}^B \dot{\omega} = \begin{bmatrix} \ddot{\theta}_1 \cos \theta_2 - \dot{\theta}_1 \dot{\theta}_2 \sin \theta_2 \\ \ddot{\theta}_2 \\ \ddot{\theta}_1 \sin \theta_2 + \dot{\theta}_1 \dot{\theta}_2 \cos \theta_2 \end{bmatrix} \quad (3.5)$$

$${}^E_B R = \begin{bmatrix} \cos \theta_2 & 0 & \sin \theta_2 \\ \sin \theta_1 \sin \theta_2 & \cos \theta_1 & -\sin \theta_1 \cos \theta_2 \\ -\cos \theta_1 \sin \theta_2 & \sin \theta_1 & \cos \theta_1 \cos \theta_2 \end{bmatrix} \quad (3.6)$$

3.1.5. Modelo de simulación en Matlab

Los modelos mostrados en los puntos anteriores sirven de base teórica para modelos mas complejos. Para el modelizado de la marcha se decidió explorar herramientas utilizadas por equipos de investigación en el área de

robots bipedestadores y trabajos relacionados como modelado de la marcha [6] [27] [37].

Se decidió utilizar la herramienta de simulación Simscape Multibody para Simulink de Matlab R2019b y utilizar el modelo de la marcha *msra-walking-robot*² provisto por Mathworks. Este modelo se basa en el un sistema de péndulo doble invertido con la particularidad que incluye grados de libertad adicionales para el movimiento de los pies, como se muestra en la figura 3.4.

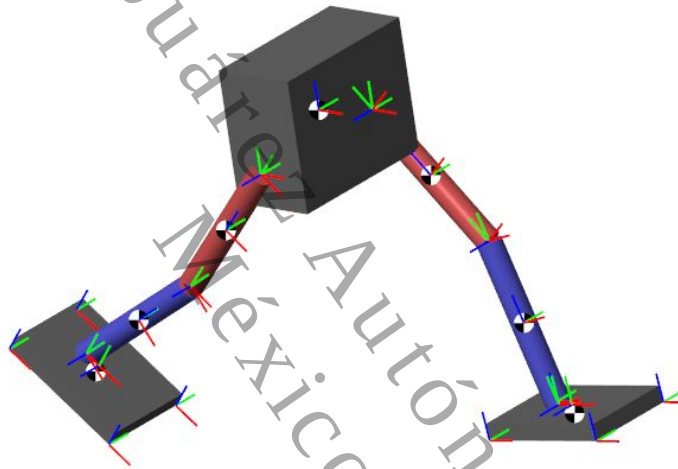


Figura 3.4. Modelo 3D de *msra-walking-robot* simulado con Simscape Multibody en Simulink de Matlab. Fuente: [47]

3.2. Modelado de filtro digital y obtención de ángulos de Euler

3.2.1. Representación usando cuaterniones

El presente trabajo no intenta explicar la teoría matemática detrás los cuaterniones, sino mas bien se busca hacer uso de esta herramienta. Se hace

²<https://github.com/mathworks-robotics/msra-walking-robot>

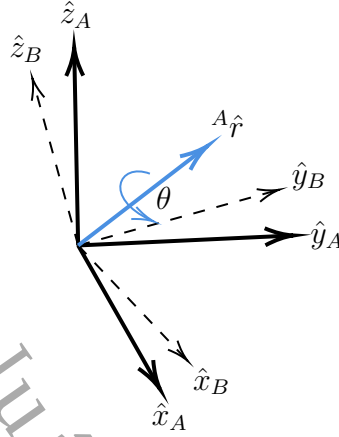


Figura 3.5. Orientación del marco de referencia B mediante una rotación θ alrededor del eje ${}^A\hat{r}$ respecto del marco A.

una introducción a la nomenclatura que se usará y a las operaciones que se utilizarán en este trabajo.

La representación usando un cuaternión permite trabajar con la orientación de cuerpos rígidos o espacios tridimensionales en un marco coordinado. Un cuaternión es un número complejo de cuatro dimensiones como se ve en la ecuación 3.7, donde ${}^A_B\hat{q}$ es la notación utilizada para indicar una orientación arbitraria del marco B respecto del marco A cuando se realiza una rotación en un ángulo θ como se muestra en la figura 3.5. Donde r_x , r_y y r_z definen las componentes del vector unitario ${}^A\hat{r}$.

$${}^A_B\hat{q} = [q_1 \quad q_2 \quad q_3 \quad q_4] = [\cos \frac{\theta}{2} \quad -r_x \sin \frac{\theta}{2} \quad -r_y \sin \frac{\theta}{2} \quad -r_z \sin \frac{\theta}{2}] \quad (3.7)$$

Los vectores $\hat{x}_A$, $\hat{y}_A$ y $\hat{z}_A$ que se muestran en la figura 3.5 son los vectores unitarios referenciados al marco A y los vectores $\hat{x}_B$, $\hat{y}_B$ y $\hat{z}_B$ son los vectores unitarios referenciados al marco B.

Es importante destacar que en el presente trabajo se utilizará la notación de Craig [31][15] para describir los vectores y las orientaciones de los marcos relativos.

Las operaciones de adición y sustracción de cuaterniones se encuentran definidas y se opera de análogamente a como se hace con los complejos. Por otro lado también se puede definir el conjugado y tiene aplicación en el cambio de los marcos relativos, es decir que ${}^B_A\hat{q}$ es el conjugado de ${}^A_B\hat{q}$ y se denota como se muestra en la ecuación 3.8 donde el símbolo $*$ denota el conjugado.

$${}^B_A\hat{q}^* = [q_1 \quad -q_2 \quad -q_3 \quad -q_4] \quad (3.8)$$

El producto de dos cuaterniones se denota con el símbolo $\otimes$ y se resuelve usando la regla Hamilton.

La rotación de un vector en 3 dimensiones se puede realizar aplicando la ecuación 3.9, donde ${}^A v$ y ${}^B v$ son el mismo vector descrito en el marco A y marco B respectivamente y cada vector contiene un cero colocado como primer elemento para tener cuatro vectores fila de elementos.

$${}^B v = {}^A\hat{q} \otimes {}^A v \otimes {}^A\hat{q}^* \quad (3.9)$$

3.2.2. Obtención de matriz de rotación

Es importante definir la matriz de rotación que describe la rotación de ${}^A\hat{q}$ como se expresa en la ecuación 3.10 y simbolizada como ${}^A_B R$.

$${}^A_B R = \begin{bmatrix} 2q_1^2 - 1 + 2q_2^2 & 2(q_2q_3 + q_1q_4) & 2(q_2q_4 - q_1q_3) \\ 2(q_2q_3 - q_1q_4) & 2q_1^2 - 1 + 2q_3^2 & 2(q_3q_4 + q_1q_2) \\ 2(q_2q_4 + q_1q_3) & 2(q_3q_4 - q_1q_2) & 2q_1^2 - 1 + 2q_4^2 \end{bmatrix} \quad (3.10)$$

3.2.3. Obtención de ángulos de Euler

Los ángulos de Euler ϕ , ψ y θ describen una orientación del marco B logrado por las rotaciones secuenciales, desde la alineación con el marco A, de ψ alrededor de $\hat{z}_B$, θ alrededor de $\hat{y}_B$ y ϕ alrededor de $\hat{x}_B$. En las ecuaciones 3.11 3.12 3.13 se muestran las expresiones que definen los ángulos de Euler para ${}^A_B\hat{q}$.

$$\psi = \arctan\left(\frac{2q_2q_3 - 2q_1q_4}{2q_1^2 + 2q_2^2 - 1}\right) \quad (3.11)$$

$$\theta = -\arcsin(2q_2q_4 + 2q_1q_3) \quad (3.12)$$

$$\phi = \arctan\left(\frac{2q_3q_4 - 2q_1q_2}{2q_1^2 + 2q_4^2 - 1}\right) \quad (3.13)$$

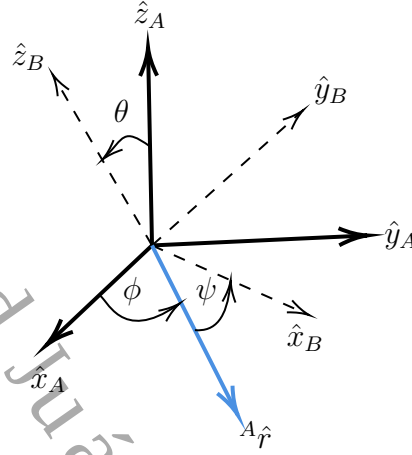


Figura 3.6. Ángulos de Euler denotados con ϕ , ψ y θ .

3.2.4. Obtención de orientación a partir de la tasa angular

Un giroscopio de 3 ejes mide las velocidades angulares sobre los ejes x , y y z y se pueden definir como w_x , w_y y w_z respectivamente. Se pueden introducir estos parámetros en un vector ${}^s w$ como se muestran en la ecuación 3.14. Es posible definir también la tasa de cambio de orientación de la tierra relativo al marco del sensor como un cuaternión derivado ${}^S_E \dot{q}$ como se muestra en la ecuación 3.15.

$${}^s w = [0 \quad w_x \quad w_y \quad w_z] \quad (3.14)$$

$${}^S_E \dot{q} = \frac{1}{2} {}^S_E \hat{q} \otimes {}^s w \quad (3.15)$$

La orientación de la tierra relativa al marco del sensor en un tiempo t , es decir ${}^S_E q_{w,t}$, puede ser calculada mediante calculo numérico, integrando la derivada del cuaternión ${}^S_E \dot{q}_{w,t}$ como se muestran en las ecuaciones 3.16 3.17.

$${}^S_E \dot{q}_{w,t} = \frac{1}{2} {}^S_E \hat{q}_{est,t-1} \otimes {}^s w_t \quad (3.16)$$

$${}^S_E q_{w,t} = {}^S_E \hat{q}_{est,t-1} + {}^S_E \dot{q}_{w,t} \Delta t \quad (3.17)$$

Donde:

${}^s w_t$ es la medición de la velocidad angular en un tiempo t .

Δt es la periodo de muestreo.

${}^S_E\hat{q}_{est,t-1}$ es la orientación estimada previa, es decir en el tiempo $t - 1$.

3.2.5. Obtención de orientación a partir observaciones vectoriales

Un acelerómetro de tres ejes medirá la magnitud y la dirección del campo de gravedad en el marco del sensor compuesto por aceleraciones lineales debido al movimiento del sensor. En el contexto de un filtro de orientación, inicialmente se supondrá que un acelerómetro medirá solo la gravedad.

Si la dirección del campo de la tierra se conoce en el marco de la tierra, una medición de la dirección del campo dentro del marco del sensor permitirá calcular una orientación del marco del sensor en relación con el marco de la tierra. Sin embargo, para cualquier medición no habrá una solución única de orientación del sensor, sino infinitas soluciones representadas por todas esas orientaciones logradas por la rotación, la orientación verdadera alrededor de un eje paralelo al campo. Una representación de cuaternión requiere que se encuentre una solución completa. Esto puede lograrse mediante la formulación de un problema de optimización en el que la orientación del sensor, ${}^S_E\hat{q}$, es la que alinea una dirección de referencia predefinida del campo en el marco de tierra, ${}^E\hat{d}$, con la dirección medida del campo en el marco del sensor, ${}^S\hat{s}$, usando la operación de rotación descrita por la ecuación 3.9. ${}^S_E\hat{q}$ se puede encontrar como solución de la ecuación 3.18 donde la ecuación 3.19 representa la función objetivo. Las componentes de cada vector están definidas por las ecuaciones 3.20, 3.21 y 3.22.

$$\min_{{}^S_E\hat{q} \in \mathbb{R}^4} f({}^S_E\hat{q}, {}^E\hat{d}, {}^S\hat{s}) \quad (3.18)$$

$$f({}^S_E\hat{q}, {}^E\hat{d}, {}^S\hat{s}) = {}^S_E\hat{q}^* \otimes {}^E\hat{d} {}^S_E\hat{q} - {}^S\hat{s} \quad (3.19)$$

$${}^S_E\hat{q} = [q_1 \quad q_2 \quad q_3 \quad q_4] \quad (3.20)$$

$${}^E\hat{d} = [0 \quad d_x \quad d_y \quad d_z] \quad (3.21)$$

$${}^S\hat{s} = [0 \quad s_x \quad s_y \quad s_z] \quad (3.22)$$

Algoritmo de descenso del gradiente

Se utiliza el algoritmo del descenso del gradiente para obtener solución a la ecuación 3.19 porque es de bajo costo computacional para los sistemas embebidos de limitadas prestaciones de hardware. En la ecuación 3.23 se muestra el algoritmo del descenso del gradiente para n iteraciones, donde ${}^S_E\hat{q}_{n+1}$ es la estimación de la orientación resultante, ${}^S_E\hat{q}_0$ es la orientación inicial y μ es el paso. La ecuación 3.24 calcula la solución gradiente de la superficie formada por la función objetivo y su Jacobiano, donde la ecuación 3.25 es la función objetivo y la ecuación 3.26 es el Jacobiano desarrollado.

$${}^S_Eq_{k+1} = {}^S_E\hat{q}_k - \mu \frac{\nabla f({}^S_E\hat{q}_k, {}^E\hat{d}, {}^S\hat{s})}{\|f({}^S_E\hat{q}_k, {}^E\hat{d}, {}^S\hat{s})\|}, k = 0, 1, 2 \dots n \quad (3.23)$$

$$\nabla f({}^S_E\hat{q}_k, {}^E\hat{d}, {}^S\hat{s}) = J^T({}^S_E\hat{q}_k, {}^E\hat{d}) f({}^S_E\hat{q}_k, {}^E\hat{d}, {}^S\hat{s}) \quad (3.24)$$

$$f({}^S_E\hat{q}_k, {}^E\hat{d}, {}^S\hat{s}) = \begin{bmatrix} 2d_x(\frac{1}{2} - q_3^2 - q_4^2) + 2d_y(q_1q_4 + q_2q_3) + 2d_z(q_2q_4 - q_1q_3) - s_x \\ 2d_x(q_2q_3 - q_1q_4) + 2d_y(\frac{1}{2} - q_2^2 - q_4^2) + 2d_z(q_1q_2 - q_3q_4) - s_y \\ 2d_x(q_1q_3 + q_2q_4) + 2d_y(q_3q_4 - q_1q_2) + 2d_z(\frac{1}{2} - q_2^2 - q_3^2) - s_z \end{bmatrix} \quad (3.25)$$

$$J^T({}^S_E\hat{q}_k, {}^E\hat{d}) = \begin{bmatrix} 2d_yq_4 - 2d_zq_3 & 2d_yq_3 + 2d_zq_4 & -4d_xq_3 + 2d_yq_2 - 2d_zq_1 & -4d_xq_4 + 2d_yq_1 + 2d_zq_2 \\ -2d_xq_4 - 2d_zq_2 & 2d_xq_3 - 2d_yq_2 + 2d_zq_1 & 2d_xq_2 + 2d_zq_4 & -2d_xq_1 - 4d_yq_4 + 2d_zq_3 \\ 2d_xq_3 - 2d_yq_2 & 2d_xq_4 - 2d_yq_1 - 4d_zq_2 & 2d_xq_1 + 2d_yq_4 - 4d_zq_3 & 2d_xq_2 + 2d_yq_3 \end{bmatrix} \quad (3.26)$$

Aplicación del algoritmo del descenso del gradiente

Una convención apropiada puede ser asumir que la dirección de la gravedad está definida en el eje z como se muestra en la ecuación 3.27. Sustituyendo ${}^E\hat{g}$ y normalizando mediciones de aceleración ${}^S\hat{a}$ por ${}^E\hat{d}$ y ${}^S\hat{s}$ en las ecuaciones 3.25 y 3.26, se obtienen las ecuaciones 3.29 y 3.30.

$${}^E\hat{g} = [0 \quad 0 \quad 0 \quad 1] \quad (3.27)$$

$${}^S\hat{a} = [0 \quad a_x \quad a_y \quad a_z] \quad (3.28)$$

$$f_g({}^S_E\hat{q}, {}^S\hat{a}) = \begin{bmatrix} 2(q_2q_4 - q_1q_3) - a_x \\ 2(q_1q_2 + q_3q_4) - a_y \\ 2(\frac{1}{2} - q_2^2 - q_3^2) - a_z \end{bmatrix} \quad (3.29)$$

$$J_g({}^S_E\hat{q}) = \begin{bmatrix} -2q_3 & 2q_4 & -2q_1 & 2q_2 \\ 2q_2 & 2q_1 & 2q_4 & 2q_3 \\ 0 & -4q_2 & -4q_3 & 0 \end{bmatrix} \quad (3.30)$$

Un enfoque convencional para la optimización requeriría que se calcularan múltiples iteraciones de la ecuación 3.23 para cada nueva orientación y las mediciones correspondientes del sensor. Algoritmos eficientes podrían requerir un ajuste del tamaño de paso μ en cada iteración para lograr el valor óptimo. Este valor se obtiene usualmente con la derivada segunda de la función objetivo pero en términos computacionales se incrementa su complejidad de calculo. Es por esto que para la presente aplicación es aceptable calcular una iteración por tiempo de muestreo, por lo tanto el tamaño de paso queda en función del tiempo μ_t .

En la ecuación 3.31 se calcula la orientación estimada para un instante t mediante valores previos de orientación estimada y el gradiente de la función objetivo. El gradiente de la función objetivo está definido en la ecuación 3.32 donde, ${}^S\hat{a}_t$ es la medición de aceleración para el instante t .

El valor óptimo de μ_t puede ser definido como aquel que garantice la tasa de convergencia de la 3.31 se limite a la tasa de orientación física, ya que esto evita el sobrepaso debido a un tamaño de paso innecesariamente grande. En la ecuación 3.33 se describe como calcular μ_t donde, Δt es el periodo de muestreo y ${}^S_E\dot{q}_{w,t}$ es la tasa de orientación física medida por el giroscopio y α es una orientación de μ para tener en cuenta el ruido de las mediciones del acelerómetro.

$${}^S_Eq_{\nabla,t} = {}^S_E\hat{q}_{est,t-1} - \mu_t \frac{\nabla f}{\|\nabla f\|} \quad (3.31)$$

$$\nabla f = J_g^T({}^S_E\hat{q}_{est,t-1}) f_g({}^S_E\hat{q}_{est,t-1} {}^S\hat{a}_t) \quad (3.32)$$

$$\mu_t = \alpha \|{}^S_E\dot{q}_{w,t}\| \Delta t, \alpha > 1 \quad (3.33)$$

3.2.6. Algoritmo de fusión

El filtro de fusión de datos consiste en obtener la orientación estimada del marco del sensor con relación al marco de la tierra ${}^S_E q_{est,t}$. Esto se obtiene de fusionar la orientación ${}^S_E q_{w,t}$ y ${}^S_E q_{\nabla,t}$ calculadas en las ecuaciones 3.28 y 3.31. En la ecuación 3.34 se muestra el cálculo donde, γ_t y $(1 - \gamma_t)$ son pesos aplicados para el cálculo de cada orientación. Como se observa esos valores son complementarios.

$${}^S_E q_{est,t} = \gamma_t {}^S_E q_{\nabla,t} + (1 - \gamma_t) {}^S_E q_{w,t}, 0 \leq \gamma_t \leq 1 \quad (3.34)$$

El objetivo es encontrar un valor óptimo de γ_t puede definirse como aquel que asegure la divergencia ponderada de ${}^S_E q_w$ sea igual a la convergencia ponderada de ${}^S_E q_{\nabla}$. en la ecuación 3.35 se observa esta relación donde $\frac{\mu_t}{\Delta t}$ es la tasa de convergencia de ${}^S_E q_{\nabla}$ y β es la tasa de divergencia de ${}^S_E q_w$. Despejando el valor de γ_t de la ecuación 3.35 se obtiene la ecuación 3.36.

$$(1 - \gamma_t)\beta = \gamma_t \frac{\mu_t}{\Delta t} \quad (3.35)$$

$$\gamma_t = \frac{\beta}{\frac{\mu_t}{\Delta t} + \beta} \quad (3.36)$$

El valor de α no tiene limite superior porque de las ecuaciones 3.35 y 3.36 aseguran la fusión óptima de los valores suponiendo que la tasa de convergencia de ${}^S_E q_{\nabla}$ gobernada por α es igual o mayor que la tasa física de cambio de orientación. La ecuación 3.33 se simplifica si se elige un α grande tal que haga muy grande al valor de μ_t . Esto hace que en la ecuación 3.31 el valor de ${}^S_E \hat{q}_{est,t-1}$ sea despreciable pudiendo reescribirse como 3.37.

$${}^S_E q_{\nabla,t} \approx -\mu_t \frac{\nabla f}{\|\nabla f\|} \quad (3.37)$$

El valor de β en el denominador en la ecuación 3.36 se hace despreciable frente al término de μ_t pudiendo reescribirse como se muestra en la ecuación 3.38.

$$\gamma_t \approx \frac{\beta \Delta t}{\mu_t} \quad (3.38)$$

En la ecuación 3.34 se sustituyen los valores calculados y se obtiene la ecuación 3.39. Es importante notar que el valor de γ_t para el primer término se escribe tal como se calculó en la ecuación 3.38 y para el segundo término se asume que su valor tiende a cero, quedando como $(1-0)({}^S_E\hat{q}_{est,t-1} + {}^S_E\dot{q}_{w,t}\Delta t)$.

$${}^S_E\dot{q}_{est,t} = \frac{\beta\Delta t}{\mu_t}(-\mu_t \frac{\nabla f}{\|\nabla f\|}) + (1-0)({}^S_E\hat{q}_{est,t-1} + {}^S_E\dot{q}_{w,t}\Delta t) \quad (3.39)$$

Es posible simplificar la ecuación anterior para obtener la ecuación 3.40, donde ${}^S_E\dot{q}_{est,t}$ es la tasa estimada de cambio de orientación definida en la ecuación 3.41. ${}^S_E\hat{q}_{\epsilon,t}$ es la dirección del error de ${}^S_E\dot{q}_{est,t}$ definida en la ecuación 3.42

$${}^S_E\dot{q}_{est,t} = {}^S_E\hat{q}_{est,t-1} + {}^S_E\dot{q}_{est,t}\Delta t \quad (3.40)$$

$${}^S_E\dot{q}_{est,t} = {}^S_E\dot{q}_{w,t} - \beta {}^S_E\hat{q}_{\epsilon,t} \quad (3.41)$$

$${}^S_E\hat{q}_{\epsilon,t} = \frac{\nabla f}{\|\nabla f\|} \quad (3.42)$$

La figura 3.7 muestra el diagrama de la implementación completa del filtro de orientación. Las ecuaciones 3.40 3.41 3.42 se observa que el filtro calcula la orientación integrando numéricamente la tasa de orientación estimada ${}^S_E\dot{q}_{est,t}$. El filtro calcula ${}^S_E\dot{q}_{est,t}$ como la tasa de cambio de orientación medida por los giroscopios, ${}^S_E\dot{q}_w$, con la magnitud del error de medición del giroscopio, β , eliminado en la dirección del error estimado, ${}^S_E\hat{q}_{\epsilon}$, calculado a partir de las mediciones del acelerómetro.

3.2.7. Cálculo de ganancia de Filtro

La ganancia β representan a todos los errores de medición del giroscopio de media zero, es decir, ruido de sensor, señales de *aliasing*, errores de cuantificación, errores de calibración, sensor desalineado, no ortogonalidad en ejes del sensor, características de respuesta en frecuencia.

En la ecuación 3.43 se observa el cálculo de β , donde $\tilde{w}_\beta$ representa el error estimado de medición de media zero del giroscopio, $\hat{q}$ es cualquier unidad de cuaternion.

El valor de $\tilde{w}_\beta$ se obtiene de la hoja de dato del sensor. Para este caso en el que el sensor utilizado es MPU-9250, sus valores se muestran en la tabla

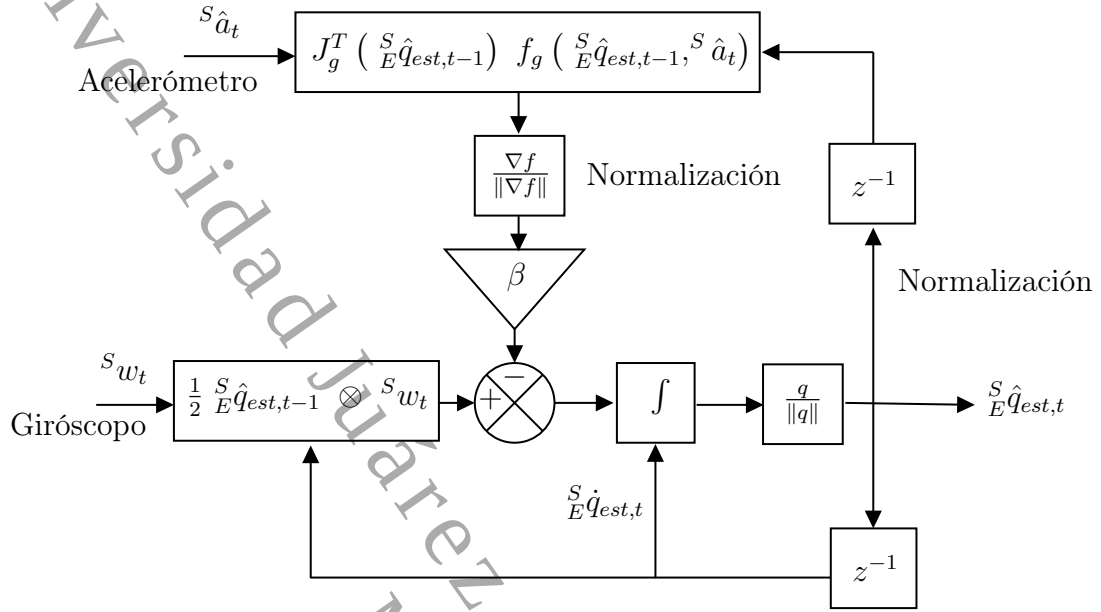


Figura 3.7. Diagrama de bloque de la implementación del Filtro

3.1. Asumiendo la condición ambiental de la hoja de dato, se decide tomar el valor de $\tilde{w}_\beta$ de 5 *grados/seg* para el cálculo β .

$$\beta = \left\| \frac{1}{2} \hat{q} \otimes \begin{bmatrix} 0 & \tilde{w}_\beta & \tilde{w}_\beta & \tilde{w}_\beta \end{bmatrix} \right\| = \sqrt{\frac{3}{4}} \tilde{w}_\beta \quad (3.43)$$

Tabla 3.1. Valores de $\tilde{w}_\beta$ para el sensor MPU-9250.

Sensor	+/- 250 dsp ³	+/- 500 dsp	+/- 2000 dsp
MPU 9250	5 dps	S/D	S/D

Capítulo 4

Red IoT de sensores corporales

En el presente capítulo se desarrolla la implementación del modelo de una red corporal de dispositivos IoT capaces de registrar información inercial de la biomecánica de la marcha.

En la sección 4.1, se propone el modelo de la red de sensores y se explica el funcionamiento general de la misma.

En la sección 4.2 se detalla la arquitectura general del sistema y se explican en detalle su funcionamiento.

En la sección 4.3 se muestra el hardware necesario para implementar el sistema explicado en la sección anterior.

En la sección 4.4 se muestra en detalle el funcionamiento del firmware que contienen los nodos.

Finalmente en la sección 4.5 se describen en detalle el software implementado en el servidor y su funcionamiento, incluyendo la estructura de la base de datos y software de terceros implementados en este servidor.

4.1. Propuesta de modelo

Como se ha explicado anteriormente, para la captura del movimiento de la marcha usando sensores inerciales es necesario de dispositivos que capturen información relativa a este movimiento, es decir sensores de aceleración y velocidad angular. Estos datos son extraídos de diferentes puntos del cuerpo del paciente como se muestra en la figura 4.1.

Tal Como se observa en la figura 4.1 la información generada por los sensores

es transmitida via Bluetooth 5¹ usando GATTs² estándares y no estándares, que fueron necesarios crear específicamente para esta aplicación. Los nodos realizan *streaming* de los datos que son recibidos por un servidor/gateway portátil.

El servidor, que posee conexión a internet, almacena los datos en una base de datos interna y pone a disposición la información para que especialistas visualicen los datos capturados conectándose directamente con el servidor desde la LAN o desde internet. En la figura 4.2 se describe el modelos de captura de datos, procesamiento de los mismos, construcción del set de datos y distribución de la información. Los nodos son dispositivos autónomos de bajo consumo que se encuentran a la espera de una señal para comenzar a capturar y transmitir la información al servidor. El servidor procesa los datos aplicando filtros y generando nueva información que será almacenada junto con los datos en bruto para formar el set de datos. La distribución de la información en tiempo real se realiza mediante este servidor que cumple la función adicional de *Broker* MQTT permitiendo que nodos y clientes se suscriban o publiquen.

En la siguiente sección se describe la arquitectura del sistema en detalle con el fin de exponer en profundidad su funcionamiento.

4.2. Arquitectura general

Como se muestra en la figura 4.3, la arquitectura del sistema propuesto consiste de cuatro componentes principales: nodos sensores, servidor local, router y clientes.

Los nodos sensores son dispositivos de tamaño reducido que, al ser colocados en el cuerpo de la persona a la que se le va realizar la toma de mediciones, permiten la adquisición y transmisión de los datos.

Cada dispositivo permanece en estado de bajo consumo hasta que este se sincronice con el servidor. Una vez realizado el *pairing* permanecerá en modo normal esperando la orden de adquisición y transmisión de las señales por parte del servidor. Cuando recibe la orden de comenzar, los datos capturados serán transmitidos al servidor usando el estándar Bluetooth LE. Es decir

¹Especificaciones Bluetooth 5: <https://www.bluetooth.com/specifications/>

²Especificaciones de GATTs: <https://www.bluetooth.com/specifications/gatt/characteristics/>

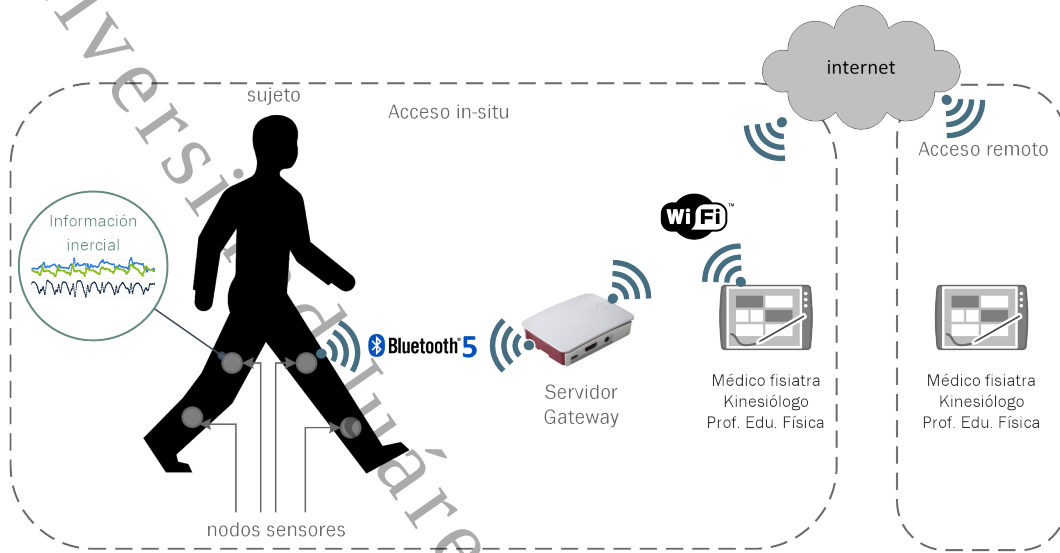


Figura 4.1. Modelo general propuesto

que la detección de dispositivos se hace a través del protocolo de perfil de acceso genérico (GAP) y la comunicación de dispositivo a dispositivo se hace a través del protocolo de atributo genérico (GATT).

Se usaron servicios UUID estándares para por ejemplo: el nombre del nodo, ubicación, versión del firmware, versión del hardware, nivel de batería del mismo. Por el contrario, para los datos propios del sensor, es decir las aceleraciones y velocidades angulares, se decidió usar servicios UUID personalizados. En cada nodo se identificaron tres grupos de datos: (i) información de identificación, (ii) datos de estado y (iii) datos de sensores.

i) Información de identificación: se refiere a los datos que van a ser transmitidos una sola vez y contienen información del dispositivo que solo puede ser leída (*read-only*).

ii) Datos de estado: son aquellos que contienen información del nodo que puede cambiar pero que no es necesaria obtener de manera continua. Es decir, que se obtiene solo cuando el servidor la requiere. Específicamente se identificaron nivel de carga de batería, consumo de corriente de batería, interruptor o *flag* de comienzo de captura y transmisión.

iii) Datos de sensores: son aquellos que contienen información referente a los sensores y que son necesarios tener de manera continua, es decir que el nodo estará haciendo *streaming* de estos datos.

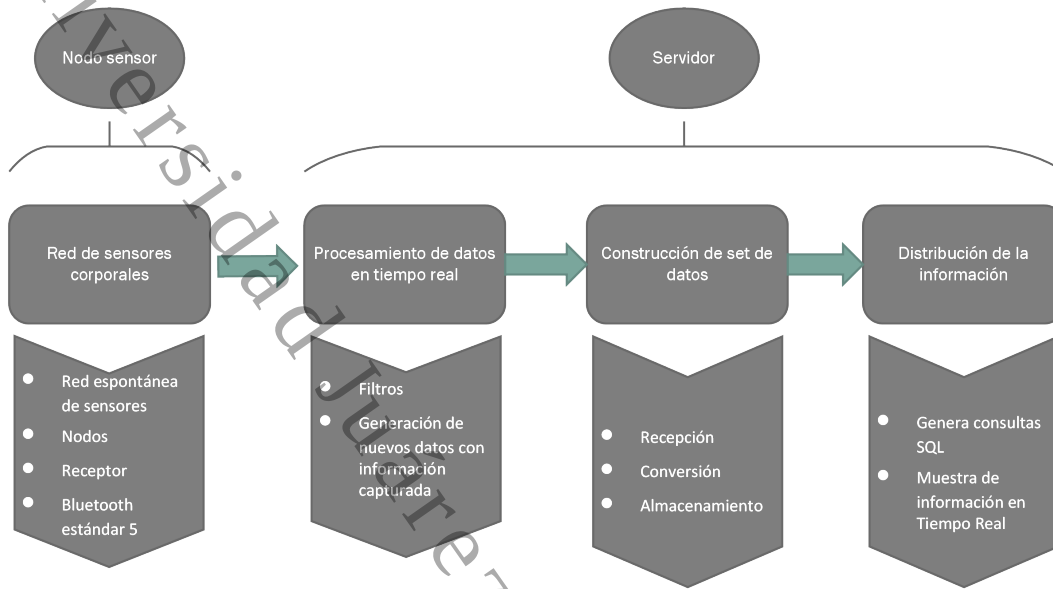


Figura 4.2. Detalle de modelo general propuesto

Como se muestra en la figura 4.3, el servidor RAMserver y el servicio de *Broker* MQTT están alojados en el físicamente en una Raspberry Pi³. RAMserver está compuesto por (i) el módulo de comunicación Bluetooth o Com BLE, (ii) el módulo generador de nodos virtuales, (iii) el módulo de comunicación MQTT o Com MQTT y (iv) la aplicación de filtrado, almacenamiento y configuración de los nodos.

i) Com BLE: Gestiona la comunicación Bluetooth con los nodos.

ii) Generador de nodos virtuales: El servidor genera por cada nodo activo que se encuentre un nodo virtual. Esto se realiza así para generar el vínculo entre los nodo Bluetooth y los clientes que están sobre un red TCP/IP. Adicionalmente los nodos virtuales se conectan al Broker MQTT para publicar o suscribirse a servicios.

iii) Com MQTT: Mediante este módulo los nodos virtuales pueden conectarse al Broker MQTT usando el estandar MQTT v3.1.1.

iv) App: La capa de aplicación o módulo de aplicación es la encargada de servir como interfaz gráfica para la configuración de los nodos, aplicar filtros digitales a las señales, realizar la fusión de datos, almacenar toda esta infor-

³<https://www.raspberrypi.org/products/raspberry-pi-2-model-b/>

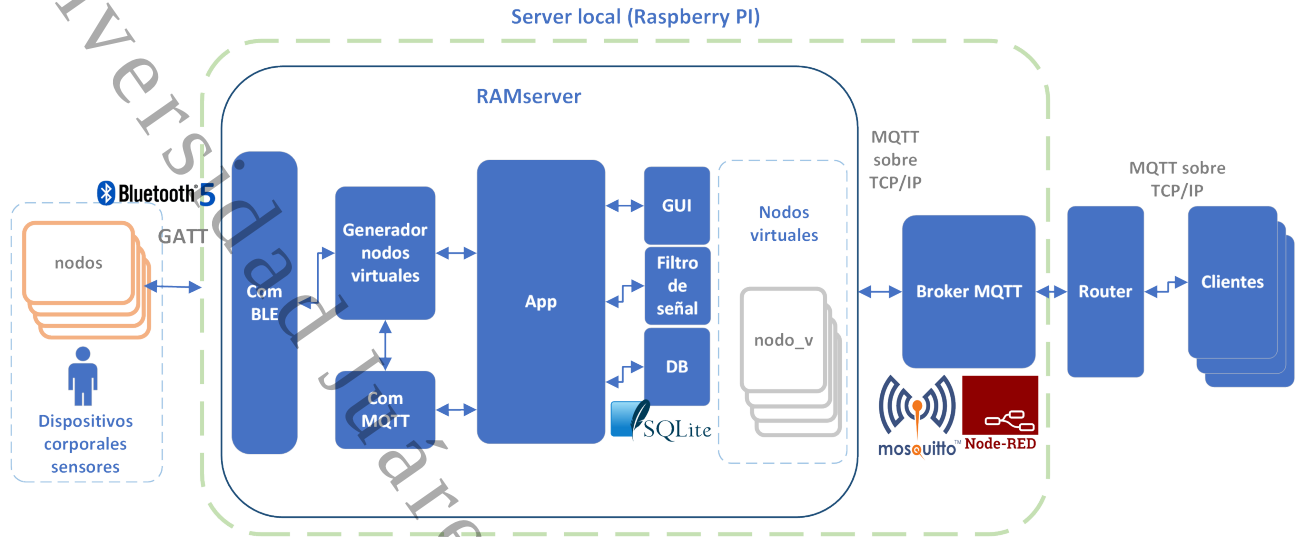


Figura 4.3. Arquitectura del sistema propuesto

mación en una base de datos.

En el bloque *Broker MQTT*, como se muestra en la figura 4.3, se refiere a la implementación de Mosquitto broker. Según el estandar MQTT su funcionamiento se basa en un modelo de suscripción y publicación, es decir que por ejemplo en este caso los nodos publican en el *Broker MQTT*, en un determinado *topic* o tópico, un determinado *message* o *payload* y son los clientes los que se suscriben al *Broker* para recibir todo mensaje que sea publicado en ese determinado tópico.

Como se muestra en la figura 4.4, los nodos virtuales publican en un *topic* específico, en este caso el Paciente A que contiene n nodos virtuales, realiza una publicación usando el tópico `pacienteA/nodoPie/aceleracionX` y al mismo tiempo el Paciente B que contiene m nodos virtuales, publica en el tópico `pacienteB/nodoPie/aceleracionX`. Por otro lado los clientes pueden realizar una suscripción a esos tópicos con el fin de recibir los mensajes que estos están enviando en tiempo real. Como se observa en la figura 4.4, tanto el mensaje o *message* del Paciente A como el del Paciente B son enviados a los clientes que estan suscritos.

La información que los clientes obtienen al suscribirse a un tópico se muestra mediante un *dashboard* o tablero de instrumentos implementado usando

Node-Red⁴. En este caso Node-Red sirve de herramienta de programación gráfica web y permite generar la lógica para la muestra de la información que los nodos virtuales envían y tener acceso mediante un tablero de instrumentos Web. En la sección 4.5 se muestra en detalle el funcionamiento.

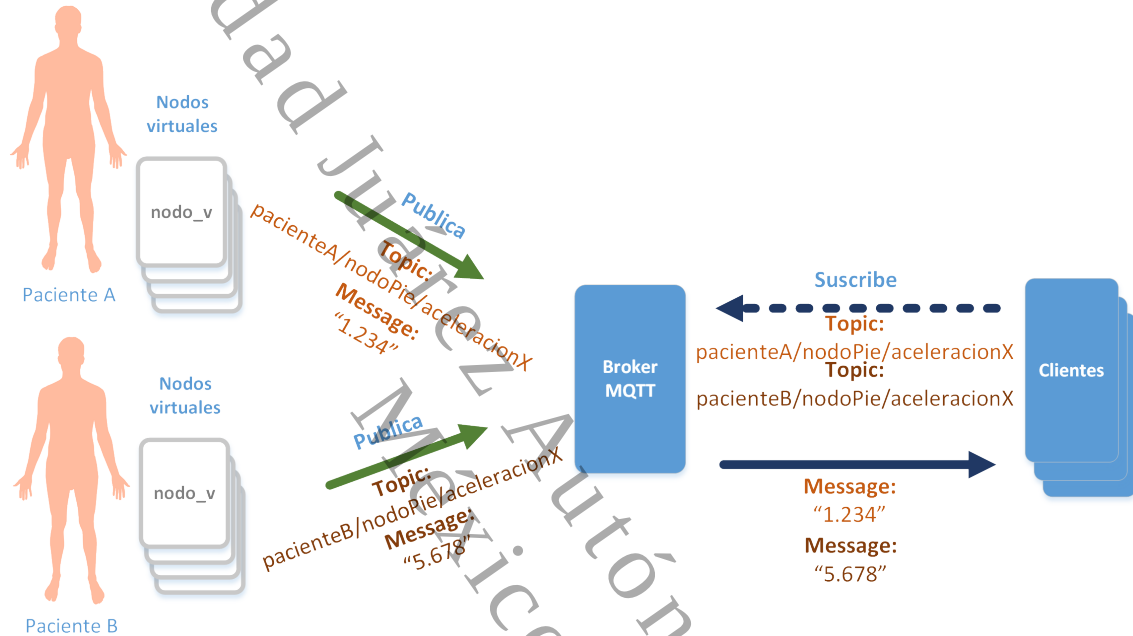


Figura 4.4. Broker MQTT: Publicación y suscripción en multiples topics

4.3. Hardware

4.3.1. Nodos

En la tabla 4.1 se describen los componentes necesarios que forman cada nodo.

Para que cada nodo pueda adquirir, procesar y transmitir la información se necesitan de una serie de componentes como es un microcontrolador, transmisor Bluetooth y sensores. El placa Arduino Nano 33 BLE contiene los componentes necesarios para procesamiento y transmisión de la información

⁴Node-Red: <https://nodered.org/>

Tabla 4.1. Hardware de Nodo

Item	Elemento	Descripción	Cantidad
1	Arduino	Nano 33 BLE	1
2	Sensor	Invensense MPU-9052 Sensor de 9 ejes	1
3	Batería	Litio recargable encapsulado LIR2032 3V	1
4	PCB Batería	Placa con zócalo para batería tipo LIR2032	1
5	Cargador de batería	Cargador de batería de Litio tipo coin cell	1
6	Gabinete	Gabinete impreso 3D	1
7	Sujeción corporal	Banda de Neoprene con cinta Hook and Loop	1

por medio de Bluetooth. Esta nueva placa incorporada recientemente dentro la familia Arduino, conserva el mismo *pinout* que Arduino Nano tradicional con la diferencia que se mejora el microcontrolador al pasar de un micro AT-MEGA328p de 8bits a un NRF52840 de 32bits.

Comparativa nrf52840 vs atmega328p

En la tabla 4.2 se comparan los microcontroladores nrf52840 y el microcontrolador atmega328p. El microcontrolador nrf52840 contiene una CPU de arquitectura ARM Cortex M4 de 32bit pero también es posible ejecutar instrucciones en modo 16bits. Por el contrario el microcontrolador atmega328p es un micro de arquitectura AVR de 8bit. Es importante destacar que para la presente aplicación se hace indispensable contar con un microcontrolador con modos de bajo consumo y FPU⁵ para hacer preprocesamiento de las señales que se obtienen en tiempo real.

El microcontrolador nrf52 posee 8 entradas multiplexadas para un conversor analógico digital del tipo SAR, es decir de aproximaciones sucesivas, que toma 200000 muestras por segundo. Por otro lado, el microcontrolador atmega328p posee 6 entradas multiplexadas para un conversor SAR que toma 15000 muestras por segundo.

En el caso de entradas digitales, ambos poseen pines dedicados a comunicación con protocolos SPI⁶ e I2C⁷.

En el caso de comunicación inalámbrica, solo el microcontrolador NRF52 integra en la misma pastilla de circuito impreso un circuito adicional de radio

⁵FPU: del inglés, unidad de punto flotante.

⁶SPI: Serial Peripheral Interface.

⁷I2C: Inter Integrated Circuit

para comunicación Bluetooth y NFC⁸ tal como se muestra en la tabla 4.2. Son por las razones antes expuestas que la elección del NRF52 es apropiada para la aplicación de la presenta tesis.

Tabla 4.2. Comparativa de microcontroladores nrf2840 y atmega328p

Microcontrolador	nrf52840	atmega328p
CPU	32-bit ARM Cortex™-M4	8-bit AVR
Frec max	64MHz max	20MHz max
FPU	64MHz	No posee
Memoria	1 MB Flash + 256 KB RAM	32 KB Flash + 2 KB RAM + 1KB EEPROM
Interfaces digitales	USB 2.0, 4 x SPI master/slave, 2 x TWI master/slave, 2 x UART, 4 x PWM, QPSI, I ² S, PDM, QDEC	1 x SPI master/slave, 1 x UART, 6 x PWM, 1 x I ² C
Interfaces inalámbrica	Bluetooth 5: 2 Mbps/1 Mbps/500 kbps/125 kbps - 802.15.4: 250 kbps 2.4 GHz proprietary: 2 Mbps/ 1 Mbps	No posee
Interfaces analógicas	8 entradas - 1 x 12-bit 200 ksps ADC	6 entradas - 1 x 10-bit hasta 15 ksps ADC

4.3.2. Esquemas de circuito de nodos

Como se muestra en la figura 4.5, cada nodo está formado por subsistemas o circuitos que se interconectan para dar funcionalidades particulares: *i)* módulo monitor de batería, *ii)* módulo de batería de Lipo, *iii)* módulo Arduino Nano 33 BLE, *iv)* módulo con sensor MPU-9250.

i) Módulo monitor de batería: En la figura 4.5 se muestra el circuito formado principalmente por un divisor resistivo entre la R2 y R1 y del punto medio se conecta al pin A0 del Arduino Nano 33 BLE para medir el nivel de la batería. Los transistores Q1 y Q2 junto con la R3 permiten que la corriente circule por el divisor resistivo para la toma de la medición. Se puede observar que desde el *gate* del transistor Q1 se conecta al pin A1 del Arduino. El pin A1 funciona como salida digital y se pone en alto para que el sistema tome

⁸NFC: Near Field Communication

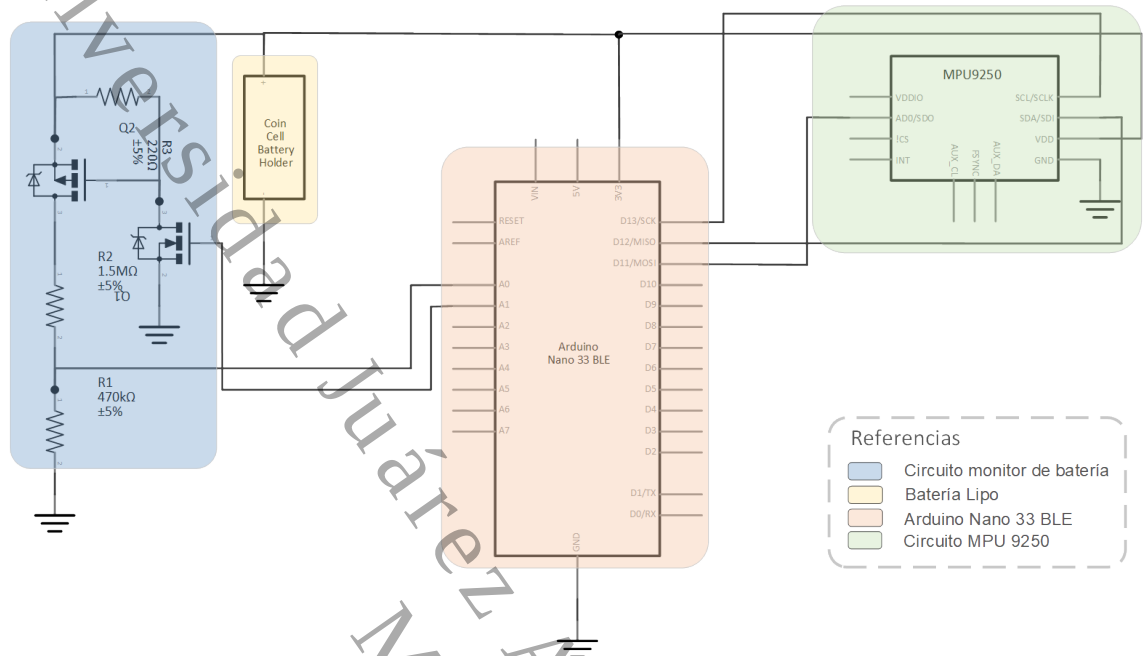


Figura 4.5. Circuito esquemático de nodos

la medición y se pone en bajo para evitar que pase corriente por el circuito divisor resistivo y de esa forma no consumir corriente innecesariamente. En la sección 4.4 se detalla como se realiza el muestreo y algoritmo para medición de la batería.

ii) Módulo de batería de Lipo: En la figura 4.6 se muestra la placa de circuito impreso, el circuito esquemático y batería LIR2032H que cada nodo usa.

La batería LIR2032H es del tipo Litio-ion recargable con voltaje nominal de 3.6v y una capacidad nominal 68mAh. En la tabla 4.3 se detallan las características que presenta la batería.

iii) Módulo Arduino Nano 33 BLE:

Como se observa en la figura 4.7 el módulo de Arduino Nano 33 BLE tiene el mismo pinout que un Arduino Nano tradicional. Este módulo implementa cambios significativos en el hardware como son un nuevo CPU y comunica-

Tabla 4.3. Característica de la batería

Item	Elemento	Descripción
1	Voltaje max	3.7V
2	Capacidad max	70mAh $\pm$ 5
3	Resistencia interna	<600m
4	Diametro/ Espesor/ Peso	20mm/ 3.5mm/ 3g 0.2
5	Temperatura de trabajo	-20C 60C

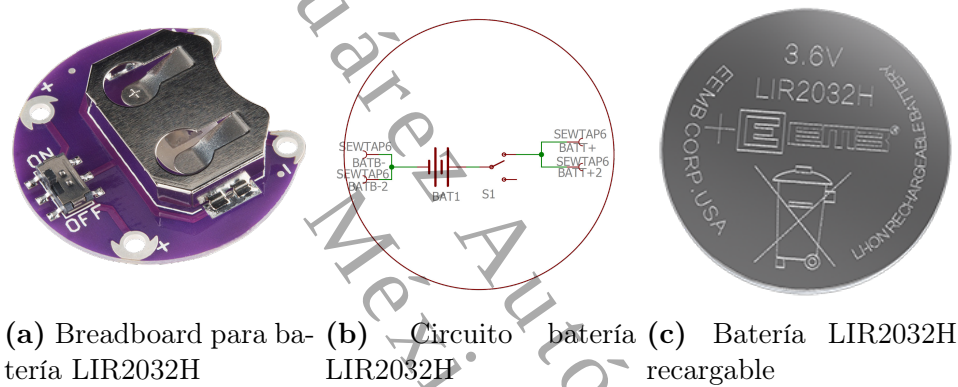


Figura 4.6. Módulo de batería y batería LIR2032H

ción Bluetooth mediante el uso del módulo NINA-B316⁹. En la figura 4.8 se muestra el módulo nina NINA-B316. Este dispositivo es una solución *stand-alone* que implementa el chip nrf52840, como se observa en la figura 4.8b. Adicionalmente se implementan todos los componentes electrónicos necesarios para la comunicación Bluetooth y para el correcto funcionamiento del circuito integrado nrf52. Es decir, en el módulo contiene antena propia y adicionalmente conexiones para múltiples antenas, componentes discretos y circuitos integrados adicionales para la alimentación del chip.

iv) Módulo con sensor MPU-9250:

En la figura 4.9 se muestra el módulo de sensores inerciales MPU-9250 de la

⁹Hoja de datos de NINA-B316: [https://www.u-blox.com/sites/default/files/NINA-B3_DataSheet_\(UBX-17052099\).pdf](https://www.u-blox.com/sites/default/files/NINA-B3_DataSheet_(UBX-17052099).pdf)

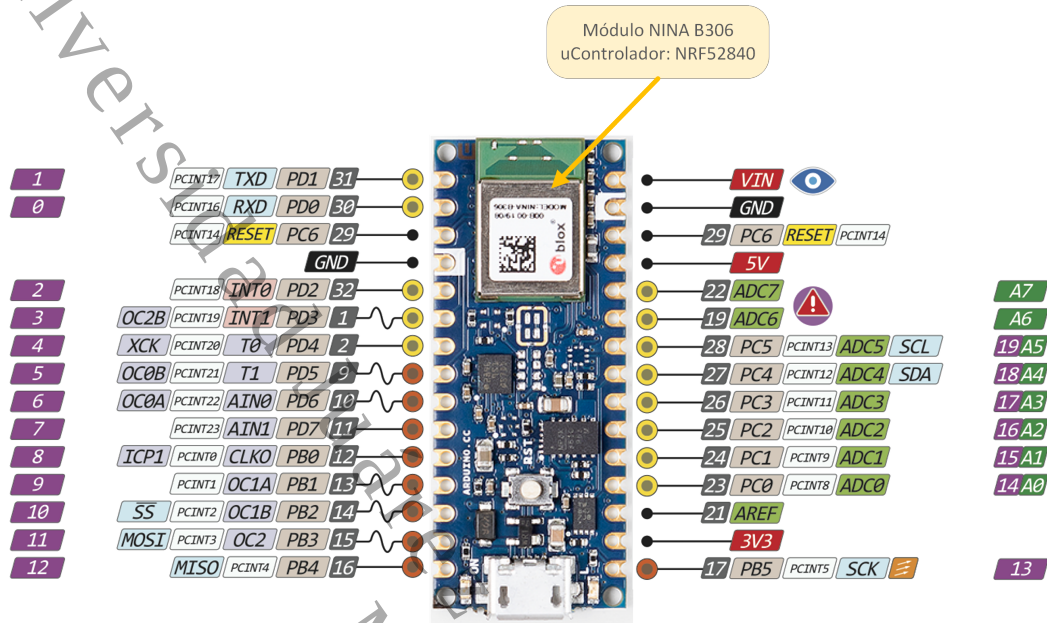


Figura 4.7. PinOut Arduino Nano 33 BLE

firma InvenSense. Este circuito integrado está compuesto en su interior por 3 acelerómetros, 3 giroscopios y 3 magnetómetros. Adicionalmente contiene un sensor de temperatura para compensar las mediciones los sensores internos pero es posible leer esa temperatura de ser necesario. Contiene una unidad interna de procesamiento digital de movimiento (DMP) para el procesamiento interno de reconocimiento de gestos y funcionalidad de podómetro.

Este circuito integrado contiene internamente la electrónica para convertir las señales analógicas en digitales y comunicar estos datos con el microcontrolador. Por lo tanto es importante resaltar que soporta comunicación I2C en *Fast Mode* de 400KHz y SPI a 1MHz.

El acelerómetro tri-axial es un sensor del tipo MEMS que tiene las siguientes características:

- Rango de medición programable para $\pm 2g$, $\pm 4g$, $\pm 8g$ y $\pm 16g$
- Conversor analógico digital interno de 16-bit
- Consumo de corriente en operación normal: 450uA
- Modo de bajo consumo: 8.4uA a 0.98Hz, 19.8uA a 31.25Hz

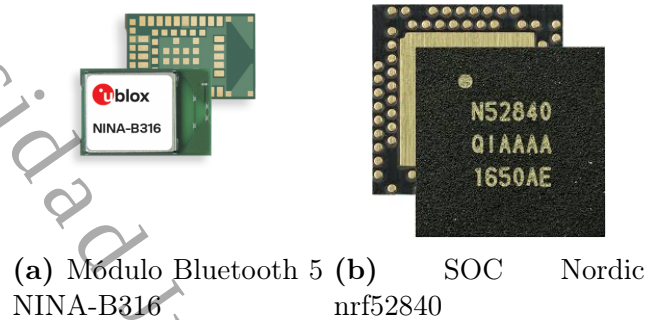


Figura 4.8. Módulo CPU y Bluetooth 5

- Modo sleep: 8uA

El girómetro triaxial es un sensor del tipo MEMS que tiene las siguientes características:

- Rango de medición programable para ± 250 , ± 500 , ± 1000 y ± 2000 grados/seg
- Conversor analógico digital interno de 16-bit
- Consumo de corriente en operación normal: 3,2mA
- Modo sleep: 8uA

El magnetómetro no será utilizado en este trabajo por lo que es posible, mediante un registro específico, apagarlo para que no consuma corriente mientras los otros sensores están en operación.

Consumo de energía

La tabla 4.4 muestra la estimación de consumo energético de cada componente obtenido de las hojas de datos de los fabricantes. En esta tabla se observan todos los estados funcionales calculados con su consumo total en función de los componentes utilizados y con el BLE configurado en modo de alto consumo de energía para asegurar un rango de entre 5 a 10 metros de alcance.

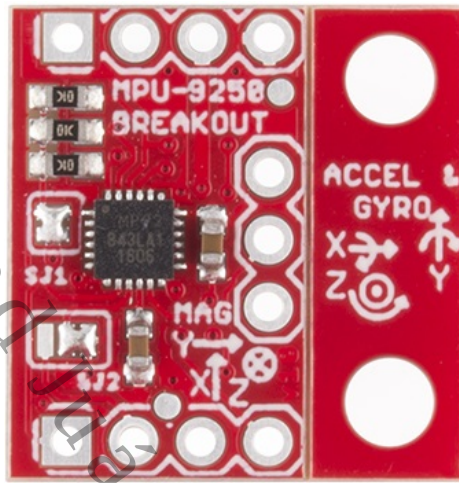


Figura 4.9. Módulo de sensores inerciales MPU-9250

En el Modo KeepAlive, el nodo se enciende y comienza a parpadear un led de color verde. Según la lógica usada el dispositivo puede permanecer en este modo por 19,7 horas con la batería elegida.

En el Modo Bajo Consumo, el nodo no enciende el led. Los sensores se encuentran en bajo consumo por lo que su consumo se encuentra en 16uA. El microcontrolador se encuentra en modo bajo consumo y el BLE se encuentra apagado por lo que el consumo es de 0,4uA. En este modo el nodo puede permanecer con batería por 1690,8 horas.

En el Modo Streaming, el nodo se encuentra funcional y con un led parpadeando de color azul. Como se observa en la tabla 4.4 el CPU y BLE consumen 4900uA y el junto con el resto de los componentes encendidos hacen un total de 8445uA. Con este consumo, la batería permite que el dispositivo trabaje por 8,3 horas.

Sujeción corporal

Los nodos sensores deben tener una sujeción que permita la normal ejecución del movimiento por parte de la persona que lo usa. Por esta razón se decidió estudiar distintas alternativas desde pantalones que contengan los sensores fijos, cintas elásticas, bandas de compresión comerciales y bandas elásticas de neoprene.

Capítulo 4. Red IoT de sensores corporales

Componentes	Consumo por estados [uA]		
	Modo pAlive	Modo de bajo consumo (LPM)	Modo Streaming (SM)
CPU + BLE (NINA-B316)	1.3	0.4	4900
Acelerometro		8	
Giroscopio	3500	8	3500
Magnetómetro		0	
Led rgb	10	0	10
LDOs	35	25	35
Total	3546.3	41.4	8445
Duración con batería de 70mAh [h]			
	19.7	1690.8	8.3

Tabla 4.4. Tabla de consumo energético por componente y duración de carga de batería

Se optó por utilizar neopreno como material textil hipoalergénico (citar: <https://www.ncbi.nlm.nih.gov/pubmed/9693703>).

En la figura 4.10 se muestra como está construido la sujeción corporal para los nodos.



Figura 4.10. Fotos de modelos de sujeción con neopreno comercial

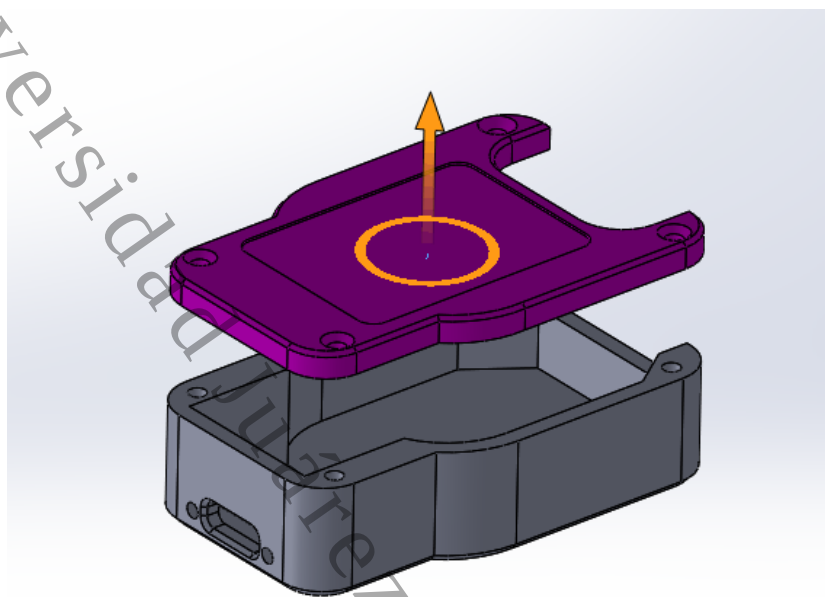


Figura 4.11. Prototipado de gabinete para nodo diseñado en SolidWorks 2018

4.3.3. Hardware de Servidor

En la tabla 4.5 se listan los elementos que se van a usar para terminar. La placa Raspberry Pi 2B es una computadora de tamaño reducido y características que se detallan a continuación:

- CPU: Broadcom BCM2836 900MHz quad-core ARM Cortex-A7 processor
- RAM: 1 GB SDRAM
- Puertos USB: 4 USB 2.0 ports
- Red: 10/100 Mbit/s Ethernet
- Consumo de energía: 600 mA (3.0 W)
- Alimentación: 5V Micro USB
- Dimensiones: 85.60 mm × 56.5 mm
- Peso: 45 g

Tabla 4.5. Hardware de servidor

Item	Elemento	Descripción	Cantidad
1	Raspberry Pi	versión 2B	1
2	Bluetooth Dongle	Whitelabel	1
3	Wifi Dongle	Tp-link Wn 725n Nano	1
4	Batería	Power Bank usb de litio de 20000mAh	1

Es importante destacar que posee comunicación Ethernet pero esta versión carece de conectividad WiFi y Bluetooth. Se decidió utilizar hardware externo con compatibilidad con el Kernel Linux que se usa. Por esta razón se usan los items 2 y 3 de la table 4.5.

Consumo energético: Para el consumo máximo de la Raspberry Pi 2B que el fabricante especifica de 600W y con el pack de batería de 2000mAh, se estima una autonomía del servidor mayor a 30 horas.

El servidor fue montado como se muestra en la figura 4.12.



Figura 4.12. Servidor montando en Raspberry Pi y fuente de alimentación portatil

4.4. Firmware

El firmware contenido en los nodos está formado por diferentes capas como se observa en la figura 4.14. Estas capas permiten la abstracción del hardware, brindando flexibilidad y escalabilidad a la implementación actual.

El firmware implementado está apoyado en la implementación de un sistema operativo de tiempo real, o RTOS por sus siglas en ingles. En este caso, el RTOS usado es Mbed OS¹⁰, un sistema operativo basado en microcontroladores ARM Cortex M pensado para internet de las cosas, con licencia Apache 2.

El firmware contiene la capa que corresponde a la API de Arduino y que soporta la aplicación propiamente dicha llamada NodoSensor. Tanto la capa del sistema operativo como la API de Arduino son provistas por la IDE de este fabricante para la versión del hardware de Arduino Nano 33 BLE.

A continuación se detallará el funcionamiento de la aplicación NodoSensor, se explicará en detalle la arquitectura del firmware y por último se expondrán la implementación de los algoritmos usados.

4.4.1. Funcionamiento

El firmware del dispositivo puede ser actualizado por medio del puerto micro USB que se encuentra en el lateral de la cara del nodo.

En la figura 4.13 se muestra el diagrama de máquina de estado que describe el funcionamiento de los nodos.

El firmware fue diseñado para maximizar la duración de la batería, priorizando la usabilidad del dispositivo para permitir que un técnico de la salud pueda poner en funcionamiento el sistema.

Los nodos se encuentran apagados, una vez que se encienden ingresan al modo de *bajo consumo* o LPM. El dispositivo permanecerá en este estado mientras el nivel de batería sea superior al valor LB y que el movimiento del nodo se encuentre por debajo del umbral de movimiento definido. Tanto el valor LB como el umbral de movimiento se definen como constantes en el código del nodo.

El umbral está definido como un valor en valor absoluto de aceleración tri-dimensional, es decir que se produzca en cualquiera de los 3 ejes, producido por el acelerómetro. En el LPM la frecuencia de muestreo del aceleración se estable en 5Hz.

Cuando el usuario mueve el nodo y supera el umbral, se chequea que el nivel de batería. Si el nivel de batería es inferior a LB, el nodo pasa a un estado de batería baja y lo muestra haciendo parpadear un led rojo por 10 segundos.

¹⁰Mbed: <https://www.mbed.com/en/platform/mbed-os/>

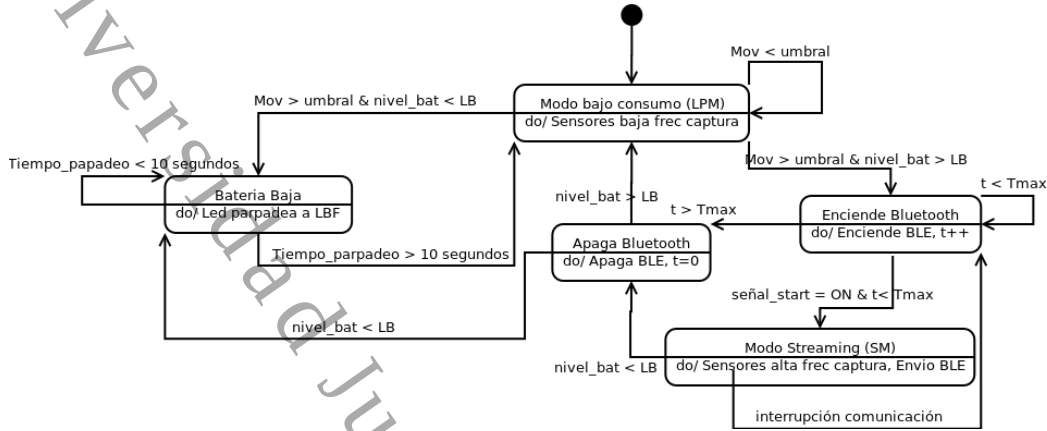


Figura 4.13. UML Diagrama de máquina de estado de cada Nodo

Pasado ese tiempo vuelve al estado LPM. Si el nodo tiene un nivel de batería superior a LB, este pasará al estado *Enciende Bluetooth* donde permanecerá por un tiempo Tmax esperando, dentro de esa ventana de tiempo, que se conecte al servidor para luego pasar al *Modo Streaming* o SM.

En el estado *Enciende Bluetooth* se enciende el BLE y se prepara para transmitir los datos. En el caso en que el nodo no haya podido conectarse al servidor superándose el tiempo Tmax, el nodo para al estado *Apagar Bluetooth* donde apaga el BLE. Si el nivel de batería es superior a LB el nodo regresa al estado LPM. En caso contrario, pasará al estado de batería baja para indicar al usuario la necesidad carga de batería y luego al estado LPM. El nodo solo pasará al estado SM cuando reciba por parte del servidor la *señal start* en ON. Luego, el nodo enciende los sensores y comienza la adquisición de los datos en el estado SM. Si se interrumpe la comunicación o se recibe la *señal start* en modo OFF, el nodo pasa al estado *Enciende Bluetooth* repitiendo nuevamente el proceso.

4.4.2. Arquitectura de Firmware

En la figura 4.14 se describe la arquitectura completa del firmware. En el punto 4.4.1, se describió el funcionamiento del nodo es decir el funcionamiento de la capa de aplicación *NodoSensor*. En esta sección se amplía y se muestra la interacción con todas las capas que componen el firmware en el nodo.

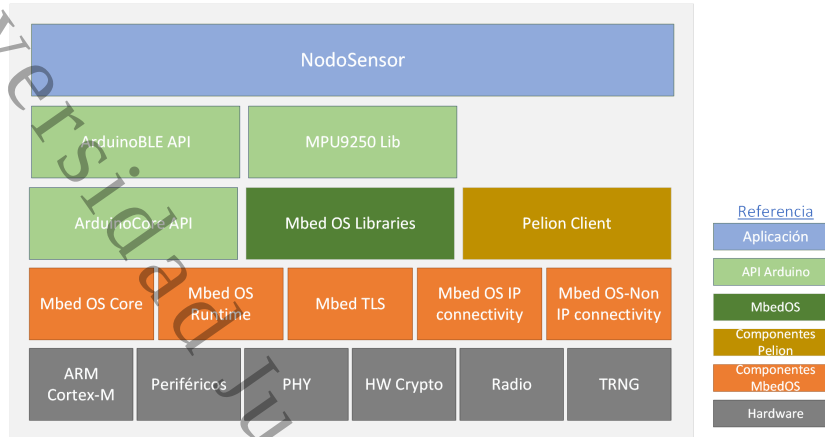


Figura 4.14. Arquitectura del firmware

Como se muestra en la figura 4.14, el firmware está compuesto de las capas de *i)* aplicación, *ii)* API Arduino, *iii)* Mbed OS, *iv)* componentes Pelion, *v)* Componentes Mbed OS y *vi)* Hardware. Cada una de las capas está auto-contenida y permite trabajar de manera independiente. El proyecto Arduino incorpora el RTOS mbed OS en su versión 5.4.0-rc1-15711 y se desprende del uso de la biblioteca *softdevices* de código cerrado provista por Nordic Semiconductor, el fabricante del chip.

A continuación se detalla el funcionamiento de capa:

i) Capa de aplicación: Es la capa que contiene la aplicación NodoSensor, es decir la funcionalidad propia del nodo sensor sin tener en cuenta temas relacionados a la implementación del hardware, drivers, stack de conectividad o gestión de tiempo de ejecución de las tareas.

ii) Capa API Arduino: Es la capa que contiene la API de Arduino necesaria para dar compatibilidad al entorno Arduino y a sus bibliotecas. Esta capa permite usar versiones precedentes o posteriores de Hardware Arduino sin cambios en la lógica principal. Está compuesta por las bibliotecas Arduino-BLE API, necesaria para utilizar BLE, ArduinoCore API que contiene el core de compatibilidad de Arduino y la biblioteca de MPU9250lib para manejo del sensor inercial MPU9250.

iii) Capa Mbed OS: Mbed OS tiene un core de un sistema operativo en tiempo real que es determinista, con capacidad multithreaded y ejecución de software en tiempo real. La capa actual contiene las bibliotecas para llamadas al sistema operativo.

- iv) Capa de componentes Pelion: Componente provisto por el paquete mbed OS para brindar conectividad con la plataforma ARM Pelion IoT.
- v) Capa de componentes Mbed OS : En esta capa se pueden destacar los componentes *Mbed OS Core* que contiene todo el core del sistema operativo y *Mbed OS Non IP connectivity* que contiene el stack de comunicaciones para protocolos como Bluetooth, BLE Stack, LoRa o LoRaWan. Este último componente permite la utilización de la conexión BLE del nrf52840 sin necesidad de utilizar la biblioteca *softdevices* provista por el fabricante del chip.
- vi) Capa de Hardware: Esta capa contiene los drivers específicos para el hardware a utilizar o la llamada HAL por sus siglas en ingles de *Hardware Abstraction Layer*. Es decir, drivers para manejo por ejempli del microcontrolador ARM Cortex o drivers para periféricos digitales/analógicos o puertos de comunicación como USB, SPI o I2C.

4.4.3. Implementación de la Aplicación

La aplicación NodoSensor que se ejecuta en cada nodo responde una lógica que se desarrolla en esta sección. En la figura 4.15 se muestra el comportamiento de la aplicación y como se ejecutan los estados. En el bloque de inicialización que se ejecuta dentro de la función `setup()` se configura e inicializa el hardware necesario que será utilizado como también configuraciones de BLE, puerto digitales/analógicos y configuración de sensores.

En la figura 4.16 se muestra el diagrama de flujo de las rutinas de configuración que el nodo realiza.

Finalizadas las rutinas de configuración, la aplicación ingresa en un bucle infinito de ejecución representado por la función `loop()` como se muestra en la figura 4.15. El programa ingresa en un `switch case` que evalúa los estados y permite la implementación de la máquina de estados en la que la app funciona. Es importante destacar en este punto que la aplicación NodoSensor está implementada como una máquina de estado pero soportada por un sistema operativo de tiempo real permitiendo que con una rutina como `sleep()` se haga una llamada al sistema para que la tarea NodoSensor pase a estado `sleep` y devolver el control de ejecución al sistema operativo. Esto permite que el sistema operativo se ponga en modo de bajo consumo, apagando el hardware para ahorrar energía.

En la figura 4.17 se muestra la implementación del estado de LPM, *low*

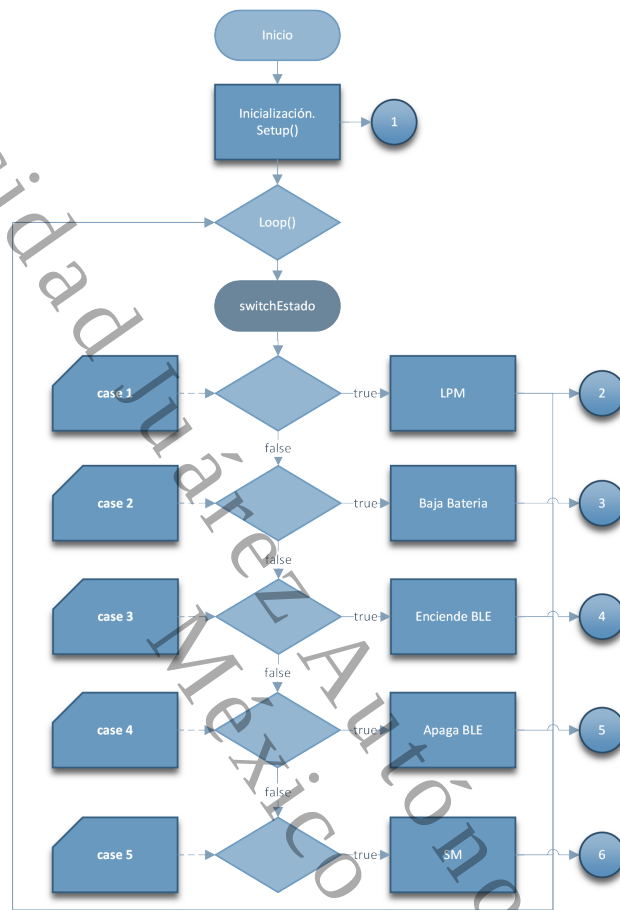


Figura 4.15. Diagrama de flujo de ejecución de estados para la aplicación NodoSensor.

power mode. Se pregunta si el movimiento adquirido por los sensores es superior al umbral y se realiza el cambio de estado correspondiente o por lo contrario se mantiene en el estado actual. También se corrobora el nivel de batería y si es baja se pasa al estado de *baja batería* y si es suficientemente alta para el funcionamiento del nodo, se pasa al estado *Enciende BLE*.

En la figura 4.18 se muestra la implementación del estado de batería baja. En este estado se permanece durante 10 segundos encendiendo un led y transcurrido este tiempo se pasa al estado de LPM.

En la figura 4.19 se muestra el algoritmo utilizado para implementar el



Figura 4.16. Diagrama de flujo de configuración de la aplicación Nodosensor

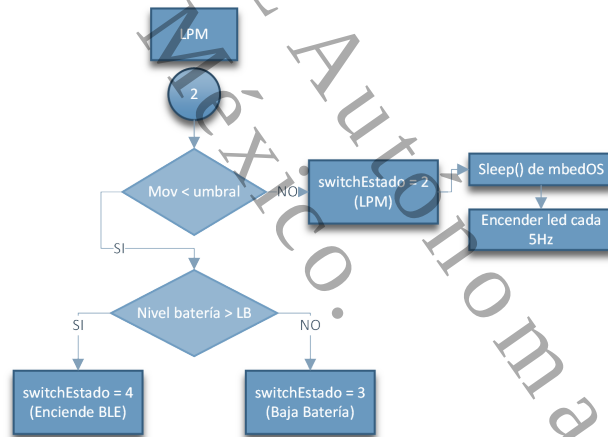


Figura 4.17. Diagrama de flujo del estado LPM

estado de Enciende BLE. En el presente estado se enciende el BLE y se espera que desde el servidor se envíe el mensaje de *start ON*. Si el servidor envía el mensaje de *start ON* se pasa al estado SM, de lo contrario la aplicación queda en el estado actual. Al inicio se cuenta con una instrucción *if* que cumple la función de corroborar si el tiempo de permanencia *t* es mayor al tiempo *Tmax*. Esta rutina es un *timeout* programado que lleva al estado Apagar BLE cuando la condición sea *true*.

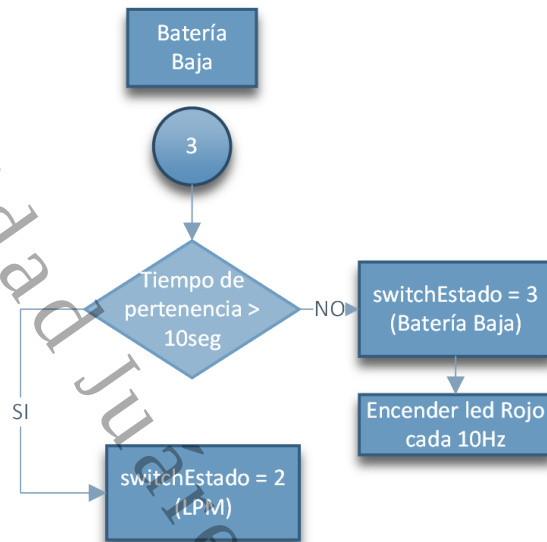


Figura 4.18. Diagrama de flujo del estado Baja Batería

En la figura 4.20 se muestra el diagrama de flujo del estado Apaga BLE. En este estado se apaga el BLE y se lee el nivel de batería para luego ser comparado con LB, es decir el nivel mínimo de batería necesaria para el correcto funcionamiento. Si se cuenta con batería se pasa al estado LPM, de lo contrario se pasa al estado de Baja Batería.

En la figura 4.21 se muestra el diagrama de flujo del estado SM. En este estado se inicia la captura de los sensores y se envía la información por BLE mientras el nodo con el servidor se encuentren vinculados y el nivel de batería se superior a LB. Si durante el envío de los datos se interrumpe la comunicación, se pasa al estado Enciende BLE. Por otro lado, si el nivel de batería es inferior a LB se pasa al estado Apaga BLE. Siempre se hace una verificación del nivel de batería al inicio antes de iniciar con la captura de la información de los sensores.

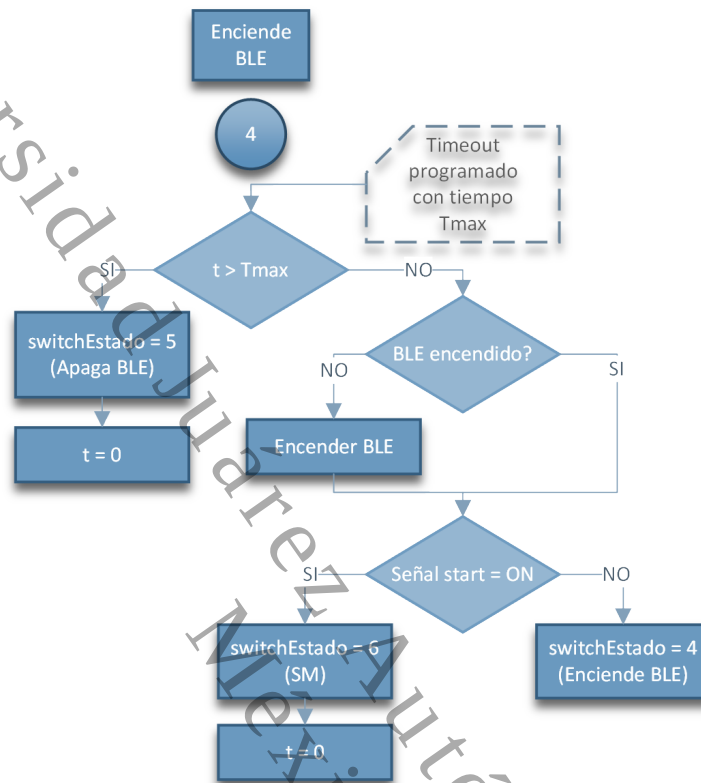


Figura 4.19. Diagrama de flujo del estado de Enciende BLE

4.4.4. Servicios y características Bluetooth LE

En la conexión BLE intervienen dos actores: un dispositivo periférico y un dispositivo central. El periférico (por lo general, un sensor o un monitor de algún tipo) indica su presencia en los canales de anuncio a los dispositivos que podrían estar intentando establecer una conexión. El periférico sigue a un protocolo, que ejecuta un servidor para repartir los datos que está generando. El dispositivo central (por lo general, un smartphone, tablet o similar) explora en los canales de anuncio si hay periféricos a los que poder conectarse. Un dispositivo central es un iniciador de protocolos, que ejecuta un cliente para recibir los datos enviados por el periférico. La interacción de los dispositivos en esta fase se define mediante el perfil de acceso genérico (Generic Access Profile, GAP).

Cuando se ha establecido la conexión, es exclusiva (un periférico sólo pue-

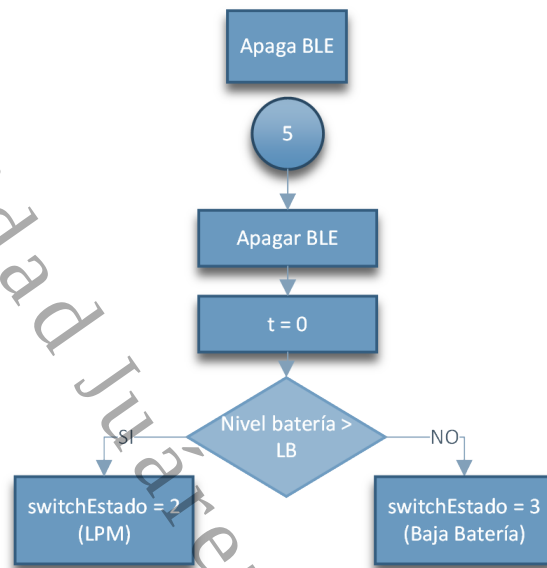


Figura 4.20. Diagrama de flujo del estado Apagar BLE

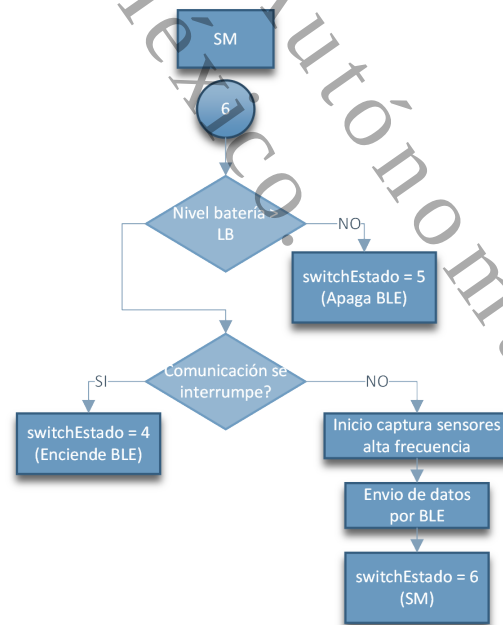


Figura 4.21. Diagrama de flujo del estado SM

de conectarse a un dispositivo cada vez), de modo que el periférico deja de anunciarse. El dispositivo central configurará los parámetros de conexión propios y los del dispositivo periférico. Esta interacción se define por el perfil de atributo genérico (Generic Attribute, GATT) que controla el flujo de datos. Los servicios, las características y sus descriptores se denominan en conjunto como atributos, y se identifican con identificadores únicos universales (Universally Unique Identifier, UUID).

Los UUID son valores de 128 bits pero pueden ser de 16 bits para perfiles estándares Bluetooth SIG.

El dispositivo central averigua los datos que oferta el periférico realizando una detección de servicio y lo que obtiene es un perfil, que es básicamente un conjunto de uno o varios servicios (los tipos de datos en oferta), que a su vez se componen de características (que pueden tener un descriptor). Imagine las características como contenedores de datos, mientras que los descriptores son etiquetas para esos datos. Las características siempre incluyen al menos dos atributos: la declaración de características (con metadatos sobre los datos reales) y el valor de la característica (los datos reales), que se solucionan con sus identificadores. Tranquilo, todo le quedará más claro cuando veamos un ejemplo práctico real más adelante.

Se definen cuatro servicios principales, de los cuales *powerOnServ* y *imuServ* son servicios personalizados y *batteryService* y *informationServ* son servicios estándares según los *GATT Services*¹¹. Para la generación de UUIDs o GUIDs personalizados necesarios para los servicios de *powerOnServ* y *imuServ* se utiliza la herramienta web *GUID online generator*¹².

A continuación se listan los servicios junto con el UUID asignado.

- Lanzar capturas (*powerOnServ*): 19B10000-E8F2-537E-4F6C-D104768A1214
- Estado de batería: (*batteryService*): 180F
- Información básica (*informationServ*): 180A
- Información inercial (*imuServ*): 8be89640-70b3-4f48-9f4b-d0da68cf2465

Para el caso de las características, se utilizan cinco características estándares como *batteryLevelChar*, *manufactureName*, *modelNum*, *hardwareRev*

¹¹GATT Services: <https://www.bluetooth.com/specifications/gatt/services/>

¹²GUID online generator: <https://www.guidgenerator.com/>

y *firmwareRev* según *GATT Characteristics*¹³. Adicionalmente se definen las siguientes características personalizadas: *batteryLevelChar*, *manufactureName*, *modelNum*, *hardwareRev*, *firmwareRev*, *imuTimeStamp*, *imuAcelX*, *imuAcelY*, *imuAcelZ*, *imuGyroX*, *imuGyroY*, *imuGyroZ*. A continuación se listan todas las características implementadas con su correspondiente UUID. Definición de UUID para características:

- Nivel de batería (*batteryLevelChar*): 2101
- Nombre de Fabricante (*manufactureName*): 2A29
- Número de modelo (*modelNum*): 2A24
- Versión de Hardware (*hardwareRev*): 2A27
- Versión de Firmware (*firmwareRev*): 2A26
- *imuTimeStamp*: 8be89642-70b3-4f48-9f4b-d0da68cf2465
- *imuAcelX*: 8be89643-70b3-4f48-9f4b-d0da68cf2465
- *imuAcelY*: 8be89644-70b3-4f48-9f4b-d0da68cf2465
- *imuAcelZ*: 8be89645-70b3-4f48-9f4b-d0da68cf2465
- *imuGyroX*: 8be89646-70b3-4f48-9f4b-d0da68cf2465
- *imuGyroY*: 8be89647-70b3-4f48-9f4b-d0da68cf2465
- *imuGyroZ*: 8be89648-70b3-4f48-9f4b-d0da68cf2465

Las características *imuTimeStamp*, *imuAcel* e *imuGyro* son del tipo *notify*, es decir que el servidor se suscribe a una o varias características a la vez del nodo, generando un canal de streaming directo para el envío de datos.

¹³GATT Characteristics: <https://www.bluetooth.com/specifications/gatt/characteristics/>

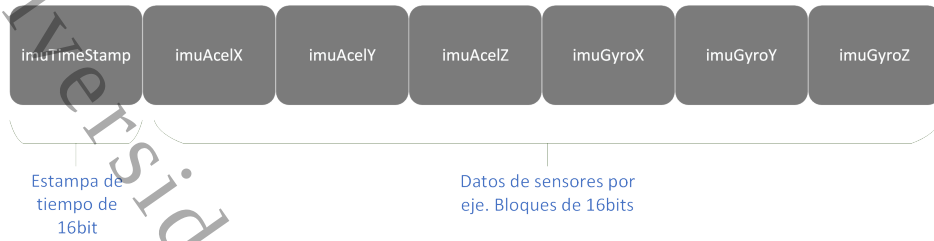


Figura 4.22. Datos generados por sensores con su marca de tiempo.

4.4.5. Rutinas de bajo consumo

Explicar funcionamiento e implementación de las rutinas de bajo consumo implementadas. Para el manejo de las rutinas de bajo consumo, se hace uso de la función de sistema `sleep`. Esta función pertenece a la API de Power Management de Mbed OS. Al hacer la llamada, el clock de sistema del Core se detiene hasta que ocurra una interrupción o un reset. Con esta rutina se deja de alimentar con energía al procesador, memorias del sistema y uso de buses. Cuando el sistema ingresa en este modo mantiene el estado del procesador, periféricos y memoria hasta que una interrupción ocurra y despierte al sistema.

En la aplicación `NodoSensor` la función `sleep` es invocada mientras el nodo tenga que permanecer en el estado de LPM.

4.5. Software

4.5.1. Servidor

El servidor está compuesto por la aplicación `RAMServer`, bibliotecas de comunicación y paquetes complementarios como son el Broker MQTT Mosquitto y una aplicación de visualización que corre sobre Node-Red.

Todo estos sistemas se ejecutan sobre un sistema operativo Linux. La distribución utilizada en este trabajo es Raspbian, un port de Debian para la arquitectura ARM del microprocesador BCM2836 ARM Cortex-A7. La versión de Raspbian usada para este proyecto está basada en Debian Buster que tiene un Kernel Linux versión 4.19.75.

Como se muestra en la figura 4.23 la arquitectura de un sistema Linux está

compuesta el propio Kernel ubicado en el centro entre el hardware y las aplicaciones. Dentro del Kernel, la interfaz de bajo nivel es la capa de conexión con el hardware y la capa abstracción de alto nivel es la encargada de proveer abstracciones comunes a todos los sistemas Unix, incluyendo procesos, archivos, sockets y señales.

Las aplicaciones como RAMServer, Mosquitto, Node-Red y complementos, corren en la capa de aplicación como se muestra en la figura 4.23.

Para la comunicación Bluetooth, el servidor local hace uso de BlueZ¹⁴, la

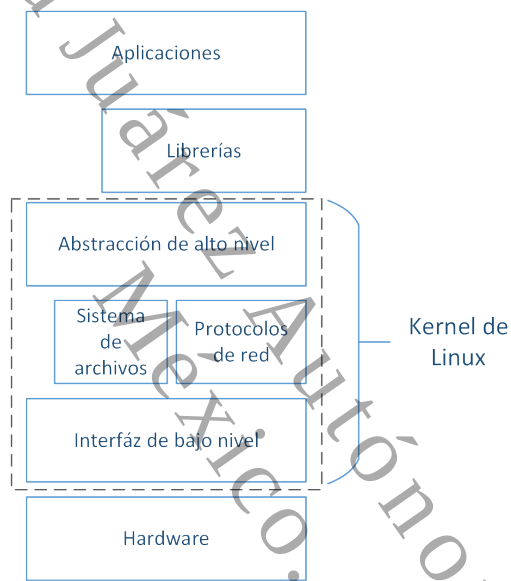


Figura 4.23. Arquitectura general del sistema operativo Linux

pila oficial para el manejo del protocolo Bluetooth en Linux. Actualmente se utiliza la versión 5.52 que no se encuentra en la instalación estándar de Raspbian. En la figura 4.24 se muestra la arquitectura de BlueZ.

Sistema base

Para el desarrollo y funcionamiento del servidor es necesario de una serie de programas y bibliotecas. El servidor se soporta sobre un sistema operativo Linux Raspbian OS versión 9.9 con Kernel Linux versión 4.19.75. Para

¹⁴BlueZ: <http://www.bluez.org/>

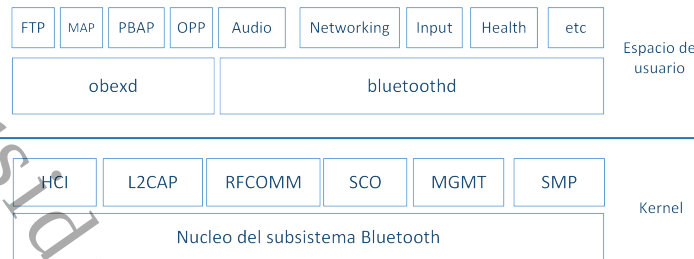


Figura 4.24. Arquitectura de BlueZ

agregar los servicios de MQTT, se utiliza el servidor **Mosquitto Broker** en versión 1.6.8. El programa que provee información a los clientes en tiempo real se programó visualmente usando Node-Red en su versión 1.0.3.

El prototipo de servidor RAMserver fue escrito en Python versión 3.5 usando BlueZ release 5.52, la implementación del stack Bluetooth para Linux. Para el manejo de BlueZ desde Python se utilizó la API bluepy en su versión 0.9.11. Los datos recibidos se guardan en una base de datos implementada en Sqlite3 y se utiliza la API sqlite3 pysqlite3 0.4.1. Para el procesamiento de los datos se utilizaron las bibliotecas **pyquaternion**¹⁵ para la manipulación de cuaterniones, **transformation**¹⁶ para la manipulación de matrices de transformación y se utilizó una biblioteca propia para la aplicación del algoritmo de filtrado y fusión de datos.

Para obtener gráficas en tiempo real es necesario usar la biblioteca de uso científico PyQtGraph versión 0.10.0.

El servidor RAMServer contiene una interfáz visual que se programó en usando las bibliotecas de Qt para Python llamada pyqt5.

Implementación de MQTT

En la figura 4.25 se muestra la estructura jerárquica de los tópicos de la red MQTT implementada. Como se muestra en la figura, se implementan en cuatro niveles. El primer nivel corresponde al **establecimiento**, es decir al lugar físico donde la red va estar funcionando, hospital, centro de rehabilitación, casa, clínica, centro educativo, etc. Este identificador sirve para aglomerar a todos los pacientes que se encuentren usando la red dentro del

¹⁵<http://kieranwynn.github.io/pyquaternion/>

¹⁶<https://www.lfd.uci.edu/~gohlke/code/transformations.py.html>

misimo espacio físico.

El siguiente nivel corresponde a **paciente** donde se identifican a cada pa-

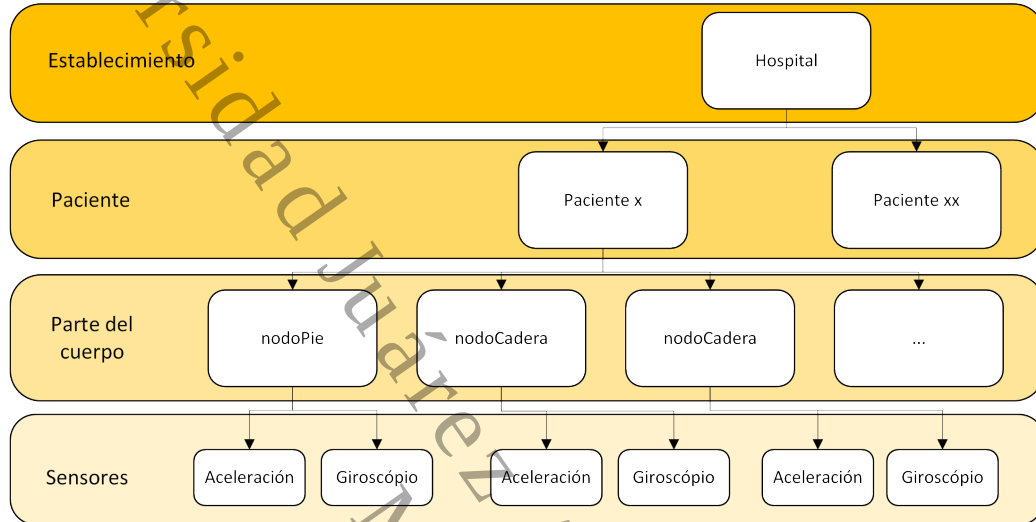


Figura 4.25. Estructura jerárquica de los tópicos MQTT

ciente o usuario de la red como un agente que contiene nodos en su cuerpo. El nivel **parte del cuerpo** corresponde a la ubicación del nodo en un lugar del cuerpo en particular. Esto permite que nodos con hardware heterogéneo envíen parámetros sobre un paciente en particular.

El nivel de **sensores** es la capa que identifica al tipo de sensor que va entregar la medición.

En el siguiente ejemplo se muestra como se aplica un tópico en particular con la información disponible a continuación:

Identificadores para ejemplo:

Establecimiento: ALPI-Córdoba (Asociación de Lucha contra la Parálisis Infantil)

Paciente: Juan Pérez

Parte del cuerpo: pie derecho

Sensores: acelerómetro eje x

Tópico de ejemplo: `alpiCba/juanPerez/nodopiederecho/aceleracionX`

Generación de nodos virtuales

Un nodo virtual es una copia instantánea de un nodo real cuando este se conecta al servidor. En la figura 4.26 se muestra el flujo de datos desde que es generado en los nodos BLE son almacenados en la base de dato y a su vez el dato actual capturado es almacenado en una cola que funciona como un buffer y el nodo virtual usa esta memoria para publicar el dato al Broker MQTT.

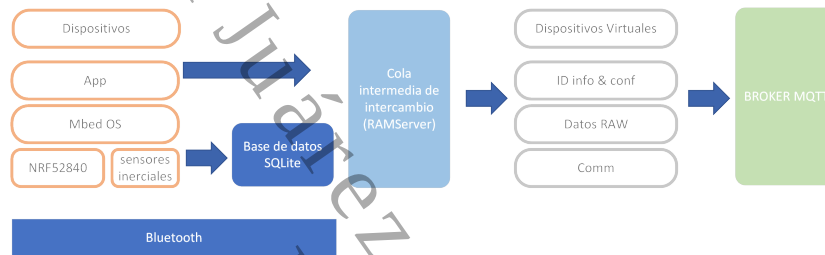


Figura 4.26. Nodo Virtual

En la figura 4.27 se muestra el diagrama de flujo del funcionamiento de generación de nodos virtuales y transmisión de información por MQTT. La aplicación RAMServer realiza un escaneo en búsqueda de dispositivos bluetooth dentro del espacio de cobertura del servidor. Los nodos encontrados se agregan a la pila de dispositivos para luego generar una instancia virtual que será una copia del nodo real con atributos adicionales para la comunicación mqtt. Estos atributos son `topicoMqtt`, `puerto`, `qos` y `mensajeMqtt`. Los nodos virtuales comienzan a publicar o hacer `pub` contra el Broker MQTT. Si la comunicación del nodo real se interrumpe, el atributo de estado o `status` cambia a *no conectado* y el atributo de valor o `value` cambia a cero.

Aplicación: Fusión de datos

En el capítulo anterior se desarrolló la matemática para la fusión de datos que se utiliza. En esta sección se desarrolla la implementación del algoritmo utilizado para tal fin.

En la figura 4.28 se muestra el diagrama de flujo del algoritmo usado para el cálculo de fusión de datos. Esta función se implementó como una biblioteca externa programa en Python. Como se ve en la figura, el cálculo se ejecuta de

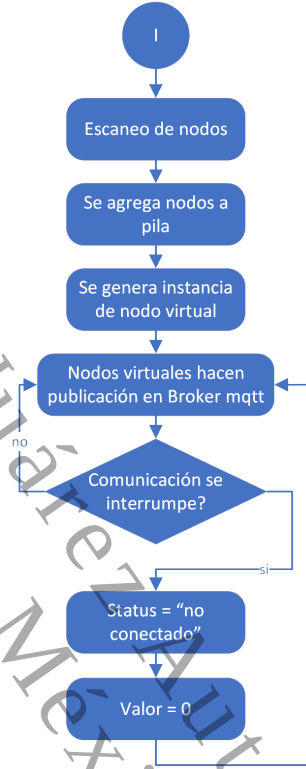


Figura 4.27. Diagrama de flujo de integración de nodo virtual

manera lineal, iniciando con obtener el valor de β y declarando las constantes que pertenecen al sistema sensor. Estas constantes son el periodo de muestreo usado y el error de medición del giróscopo en unidades de radianes/segundos. Se utiliza un periodo de muestreo de 1[ms] y para el error se debe realizar una conversión de unidad de los grados/segundos provisto por el fabricante del sensor usando la ecuación 4.1.

$$ErrorGiroscopio_{rad/seg} = \pi \left(\frac{ErrorGiro_{grados/segundos}}{180} \right) \quad (4.1)$$

La normalización de las mediciones de aceleración ejecutan la ecuación 4.2. Es necesario hacer el cálculo de la norma del vector primero como se observan en las ecuaciones siguientes.

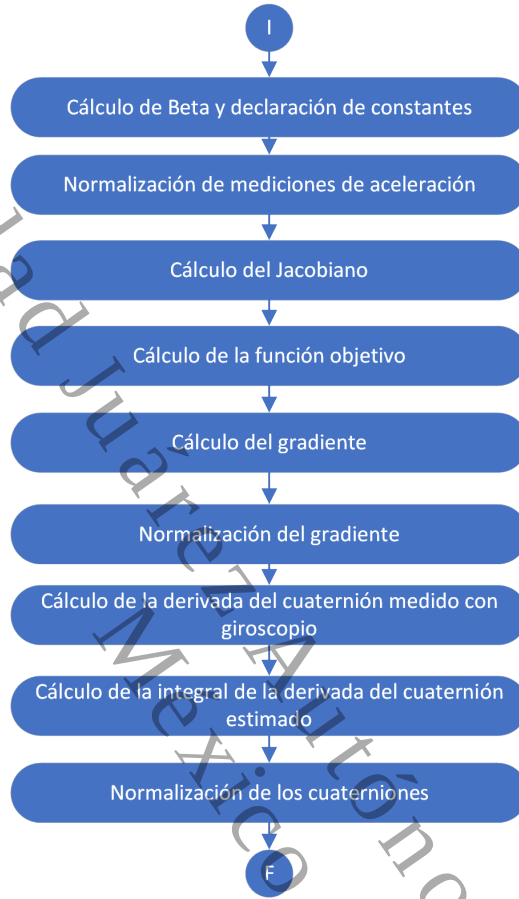


Figura 4.28. Diagrama de flujo de algoritmo de fusión de datos

$$norma = \sqrt{aceleracion_x^2 + aceleracion_y^2 + aceleracion_z^2} \quad (4.2)$$

$$aceleracionNormalizada_x = \frac{aceleracion_x}{norma} \quad (4.3)$$

$$aceleracionNormalizada_y = \frac{aceleracion_y}{norma} \quad (4.4)$$

$$aceleracionNormalizada_z = \frac{aceleracion_z}{norma} \quad (4.5)$$

El cálculo del Jacobiano y de la función objetivo ejecutan las ecuaciones desarrolladas en el capítulo 3.

Para el cálculo de la derivada del cuaternión medido por el giróscopo se utiliza la ecuación 4.6.

$$\dot{\omega}_1 = \frac{1}{2} \text{cuaternion2}_{condicioninicial} w_x - \frac{1}{2} \text{cuaternion3}_{condicioninicial} w_y - \frac{1}{2} \text{cuaternion4}_{condicioninicial} w_z \quad (4.6)$$

La integración del cuaternión que responde a la ecuación 4.7.

$$q = q + (\dot{\omega}_1 - (\beta * \dot{q}) \Delta t) \quad (4.7)$$

Aplicación: Base de datos

Es necesario almacenar los datos capturados por los sensores en red y por esto se ha decidido utilizar un gestor de base de datos SQL como es SQLite. Los datos se almacenan en cinco tablas como se muestra en la figura 4.29.

La tabla **Data** almacena el dato capturado y la información temporal asociada a este, como son el campo **time**, **timestamp** y **date**. Adicionalmente se cuenta con cuatro campos utilizados para relacionar a las otras tablas, como son **id_node**, **id_variable**, **id_sensor** y **id_type**.

En la tabla **Node** se almacenan los datos asociados propiamente de identificación del nodo como son **name** o nombre del nodo, **ble_address** o dirección BLE del nodo y versión de firmware o **firm_version**.

La tabla **Sensor** almacena el tipo de sensor que corresponde a un valor particular de medición, ya que el cada nodo puede contener mas de un tipo de sensor. Por ejemplo, este campo puede contener los valores de **accelerometer**, **gyroscope**, **magnetometer** o **temperature**.

La tabla **Variable** contiene información sobre las variables que se están midiendo. Es decir el nombre de la variable que se mide como por ejemplo, **accel_x** o aceleración en eje x.

La red de sensores está pensada para que contener multiples nodos de diferentes tipos. Es decir, que en este caso particular de este trabajo solo serán nodos tipo IMU pero está prevista para que contenga otros nodos como pueden ser nodos EMG. Por esta razón la tabla **type** contiene datos asociados a los tipos de nodos que la red puede contener.

Para facilitar la creación de la base de datos y la modificación ya sea de las tablas y contenido se utiliza el programa *DB Browser for SQLite* para Linux en la versión 3.11.99.

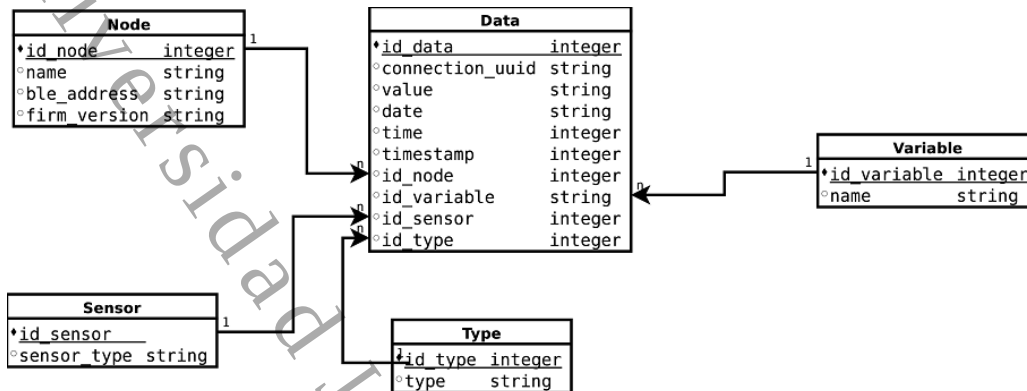


Figura 4.29. Diagrama de Base de datos

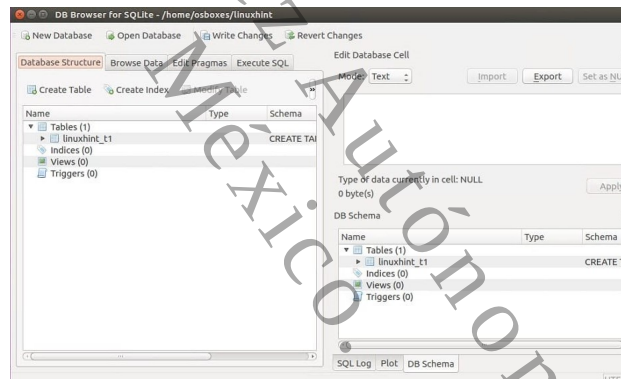


Figura 4.30. DB Browser for SQLite DB

Capítulo 5

Experimentos y Resultados

En el presente capítulo se presentan los resultados de los experimentos realizados. En la sección 5.1 se muestran las características físicas del prototipo de nodo sensor construido, material, dimensiones y peso. Finalmente se realiza una prueba de servicios Bluetooth con diferentes dispositivos móviles para demostrar compatibilidad.

En la sección 5.2 se detalla el prototipo de servidor, su hardware, materiales, consumo de energía y capturas del software servidor.

En la sección 5.3 se realiza un análisis de pérdida de información transmitida por los nodos en función a la distancia.

Finalmente en la sección 5.4 se realiza un ensayo comparativo de la información obtenida por los nodos y distintos programas para el análisis óptico de la biomecánica. Adicionalmente se realiza una prueba de características la marcha como cadencia, velocidad y distancia de pasos usando el programa iGait junto con el set de datos obtenidos por los nodos y se comparan con información obtenida de programas de análisis ópticos.

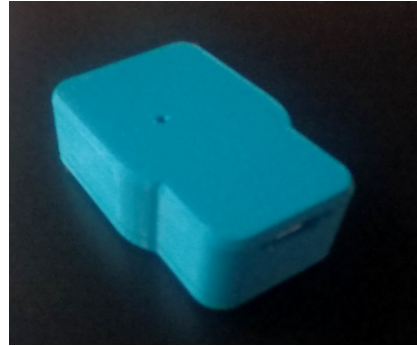
5.1. Prototipo de nodo

En la figura 5.1 se muestran el hardware del nodo montado sobre el gabinete impreso en PLA. En la imagen 5.1a se observa una vista frontal del nodo sensor, donde se puede apreciar la ranura para insertar la batería recargable. Es importante destacar que sobre la tapa se encuentra marcado el sentido en el que los ejes de los sensores se encuentran orientados. Estas marcas se

colocaron con fines prácticos para las pruebas iniciales.



(a) Foto vista frontal del nodo



(b) Foto vista lateral del nodo

Figura 5.1. Fotografías de nodos montados en gabinetes finales.

5.1.1. Características físicas

En la tabla 5.1 se muestran las características físicas que cada nodo presenta, es decir las dimensiones físicas, peso y materiales de construcción. El gabinete del nodo se imprimió usando el polímero biocompatible tipo PLA [16]. En la figura 5.2 se muestra la primera verificación del peso del hardware que compone el nodo junto con su batería desmontable.



Figura 5.2. Comprobación de peso del hardware de los nodos.

Tabla 5.1. Características técnicas físicas de los nodos

Nodos	
Dimensiones	largo: 51 mm ancho: 33 mm alto: 13 mm
Peso	10.7g
Material	PLA

5.1.2. Testeo de servicios

En la siguiente prueba se busca mostrar que los servicios Bluetooth de los nodos se encuentran bien definidos y son capaces de ser visualizados por dispositivos con sistemas Android, IOS y Linux. Para las pruebas se utilizó un teléfono inteligente modelo Motorola XT1575 con Android 8.1.0 y la aplicación *nRF Connect*¹. Por otro lado para las pruebas sobre un dispositivo de la marca Apple se utilizó un Iphone modelo Xs Max con IOS versión 13.3.1 y se usó la aplicación nRF Connect para IOS en la versión 2.2.1 para conectarse al nodo. Finalmente se utilizó el programa *gatttool* en el servidor Raspberry que se describe en el capítulo anterior.

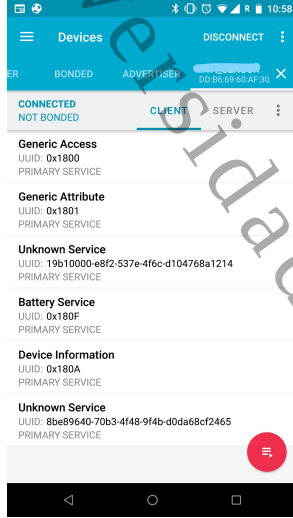
En las figuras 5.3a, 5.3b y 5.3c se muestran las capturas de pantalla de las pruebas de servicios encontrados que demuestra 100 % de compatibilidad con los sistemas Linux, IOS y Android.

5.2. Prototipo de servidor

El prototipo del servidor es montado en un gabinete comercial para protección de sus componentes de material acrílico. En la figura 5.4 se muestra al servidor montado sobre el pack de batería de 20000mAh y cable de alimentación USB correspondiente.

Para ejecutar el prototipo es necesario colocar la sentencia `python3 main.py` en la consola de la Raspberry. La imagen 5.5 muestra la vista inicial de la aplicación. En la parte superior sobre el identificador *Devices* se encuentra un botón destinado a realizar la búsqueda y conexión de nodos cercanos, *Scan*

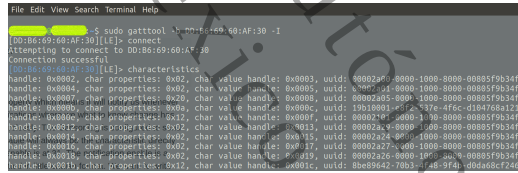
¹<http://www.github.com/NordicSemiconductor/Android-nRF-Connect>



(a) Verificación de servicios usando la aplicación nRF Connect para Android.



(b) Verificación de servicios usando la aplicación nRF Connect para IOS.



(c) Verificación de servicios usando gatt-tool en Raspberry.

Figura 5.3. Verificación de servicios BLE disponibles en los nodos.

y otro para la configuración de los nodos que ya se encuentren conectados, *Config*.

En la figura 5.6 se muestra la ventana de búsqueda y conexión de nodos. Sobre la parte izquierda en *Found devices* aparecerán las direcciones de los dispositivos que se encuentre en el área. Con los botones centrales se selecciona a que nodo se desea conectar. Para finalizar, el botón *Connect* debe ser presionado para que el servidor y los nodos se conecten.

Ahora los sensores conectados transmiten la información de los sensores



Figura 5.4. Servidor Raspberry PI con fuente de alimentación portátil de 20000mAh

constantemente. En la parte superior derecha de la figura 5.5 dentro del identificador *Log information* se encuentran dos botones para comenzar y detener el logeo de la información en la base de datos. Adicionalmente con la opción *Generate C3D* se está indicando que se desea obtener un archivo C3D con la sesión de captura actual para ser analizado posteriormente por programas de análisis de la biomecánica.

En la parte central se visualizan los nodos sensores agregados y se posicionan en función a la información de posición configurada con la que ya cuenta cada nodo sensor.

Dentro de los *labels Address, Status, Battery Level* y *Place* se carga la información referente a un nodo sensor. Es posible obtener mayor información de los nodos sensores como es la información en tiempo real de aceleración en sus 3 ejes y velocidad angular en sus 3 ejes con el uso del boton *+ info*. En la figura 5.7 se muestra la captura de pantalla para tres nodos sensores transmitiendo en tiempo real. Como se observa en la misma figura, las ventanas son individuales por cada nodo sensor por lo que es posible visualizar los nodos que se requieran independientemente uno de otro.

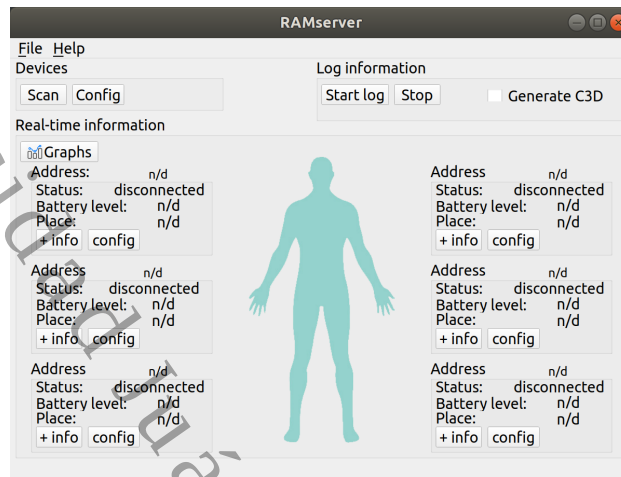


Figura 5.5. GUI interfaz gráfica

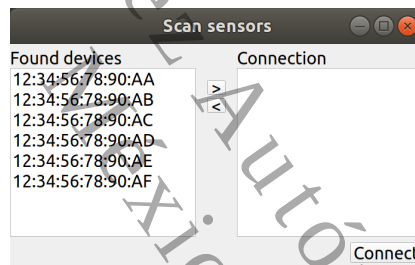


Figura 5.6. Escaneo de los sensores encontrados

5.3. Análisis pérdida de paquetes en función de distancia

La evaluación de paquetes perdidos tiene como fin construir métricas de paquetes perdidos en la transmisión entre un nodo y el servidor a diferentes distancias. Con este ensayo se busca determinar el radio máximo en el que un paciente puede estar del servidor.

Las condiciones iniciales para el ensayo son las siguientes:

- nodo sensor con batería al 100 %.
- ambiente cerrado con visión directa entre nodo y servidor.
- RAMServer corriendo correctamente en Raspberry.

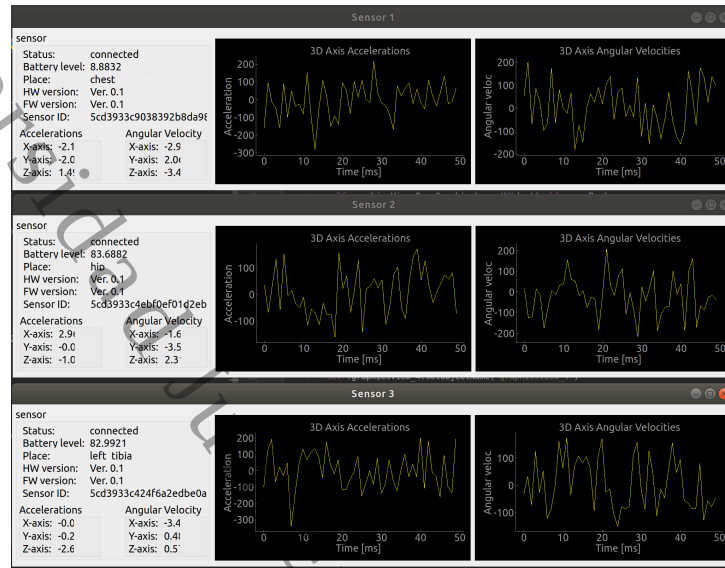


Figura 5.7. Muestra de gráficos en Tiempo Real

Se registrarán 50 muestras por segundo por un tiempo de 30 minutos por cada valor de sensor por eje incluido la marca de tiempo. Es decir que serán 90000 valores por eje de sensor y 90000 marcas de tiempo. En total serán registradas 90000 por 6 sensores, entre acelerómetro eje x , y y z y giróscopo eje x , y y z y adicionalmente 90000 por la marca de tiempo. Esto constituye un total de 630000 paquetes transmitidos por cada distancia.

El registro de mediciones se repite variando la distancia que comienza en 1[m] hasta 10[m] con pasos de 0,5[m].

Se evalúa la perdida de un paquete por cada conjunto capturado de datos.

Un conjunto de datos es la unión de los paquetes individuales de aceleración y giróscopo para cada eje y la marca de tiempo para tal conjunto de datos.

Las marcas de tiempo es un número de 16bit que asignado en el nodo cuando toma un conjunto de datos. Se analiza la diferencia entre la marca de tiempo actual y la anterior. Si esta diferencia es superior a 0,02 segundos o alguno de los paquetes que corresponde a la marca de tiempo analizada no se encuentra, se establece que hubo perdida de algún paquete del conjunto.

En la tabla 5.2 se resumen los resultados obtenidos.

Tabla 5.2. Resultado de prueba de distancia

Distancia [m]	Paquetes perdidos [%]					Media [%]	Desv Std
	Test 1	Test 2	Test 3	Test 4	Test 5		
1	13.21 %	9.95 %	10.73 %	16.59 %	16.64 %	13.42 %	0.0315038
1.5	10.18 %	10.64 %	11.09 %	11.10 %	12.07 %	11.02 %	0.0070084
2	10.44 %	9.40 %	10.03 %	10.35 %	10.30 %	10.10 %	0.0042514
2.5	87.64 %	9.81 %	10.93 %	9.84 %	10.94 %	10.38 %	0.0064556
3	17.85 %	11.46 %	9.84 %	15.28 %	9.84 %	12.86 %	0.0356866
3.5	14.24 %	10.24 %	18.88 %	14.36 %	11.74 %	13.89 %	0.0328621
4	17.28 %	11.43 %	11.11 %	10.06 %	13.63 %	12.70 %	0.0287025
4.5	10.46 %	12.38 %	16.87 %	13.60 %	17.21 %	14.11 %	0.0290866
5	12.27 %	13.45 %	15.88 %	14.34 %	16.73 %	14.53 %	0.0180037
5.5	10.22 %	17.47 %	17.97 %	15.13 %	14.67 %	15.09 %	0.0307462
6	9.57 %	16.64 %	13.17 %	17.49 %	15.76 %	14.53 %	0.0320949
6.5	10.43 %	11.45 %	14.55 %	13.76 %	17.23 %	13.48 %	0.0267794
7	12.77 %	11.26 %	13.32 %	14.35 %	11.65 %	12.67 %	0.0125333
7.5	9.77 %	15.18 %	14.50 %	15.96 %	16.11 %	14.30 %	0.0261566
8	15.94 %	13.44 %	12.32 %	14.92 %	13.04 %	13.93 %	0.0147007
8.5	11.01 %	11.79 %	15.20 %	11.14 %	12.11 %	12.25 %	0.0171118
9	14.17 %	11.37 %	14.25 %	13.42 %	14.26 %	13.49 %	0.0123816
9.5	13.12 %	11.95 %	12.90 %	14.34 %	14.24 %	13.31 %	0.0099745
10	10.98 %	13.74 %	15.53 %	14.98 %	12.26 %	13.50 %	0.018894

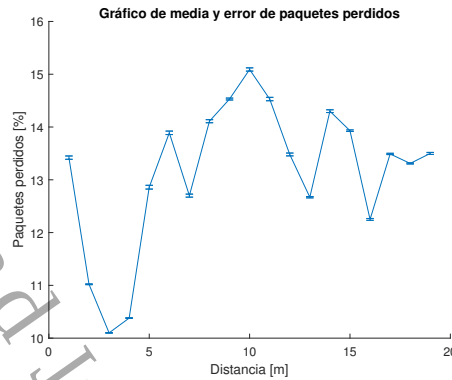


Figura 5.8. Gráfico de comportamiento de la media de paquetes perdidos en porcentaje.

5.4. Ensayo comparativo usando test 6MWT

En la presente sección se evalúa las mediciones obtenidas por los nodos de aceleración y velocidad en la ejecución de la marcha respecto de los datos que entregan herramientas de análisis de la biomecánica ópticos de código abierto como Kinovea v0.8.15, Skillspector v1.2.3 release 2011 y KWON3D. Para el ensayo se decide implementar el test 6MWT² ya que es un estándar para el análisis de la marcha [11] [21].

Para el procedimiento de la prueba se tomarán como referencia los pasos del 6MWT y no se registrarán mediciones de pulso cardíaco ni se registrará SpO2. El procedimiento se describe a continuación:

- Medir y pesar al paciente de forma estandarizada y registrarlo en la hoja de trabajo.
- Solicitar al paciente que permanezca en posición sedente al menos 15 minutos antes de la prueba.
- Contador de vueltas debe estar en cero y el cronómetro programado para seis minutos.

Se tomaron 5 personas de los cuales 2 son mujeres y 3 son hombres. Las edades de los pacientes se encuentran entre los 30 y los 65 años.

Se utilizaron 3 nodos y se posicionaron en pierna, muslo y cadera como se

²6MWT: Prueba de caminata de los minutos

muestra en la figura 5.9.

Todas las ejecuciones del test que los sujetos experimentales realizaron fue-



Figura 5.9. Ubicación de nodos sensores en el cuerpo de la persona para medición.

ron filmadas para el posterior análisis con Kinovea, Skillspector y KWON3D. Para esto se usó una cámara marca GoPro Hero 5 Black montada sobre un trípode de 1 metro de altura.

El ambiente donde se ejecutaron las pruebas mantiene temperatura constante de 26 grados centígrados y la superficie del suelo, de longitud mayor a 30 metros, está perfectamente lisa y nivelada.

Se marcó con cinta de color el suelo para indicar la línea de comienzo como así también la línea de finalización. La actividad se cronometró usando la aplicación de cronómetro por defecto de los dispositivos Android.

Antes de comenzar los sujetos experimentales fueron pesados.

La figura 5.10 muestra la ejecución de un ensayo donde la persona realiza el movimiento con los sensores colocados sobre su cuerpo sobre una línea recta blanca demarcada cada 1 metro.

El análisis de los datos y las gráficas fueron realizadas en MATLAB usando el script de Seb Madgwick desarrollado para su tesis doctoral³.

La gráfica 5.11 muestra los datos en crudo obtenidos de aceleración en 3D y velocidad angular en 3D para el nodo colocado en la pierna del sujeto. Con la información en cruda, el script de Seb Madgwick realiza los cálculos

³<https://github.com/xioTechnologies/Gait-Tracking-With-x-IMU>

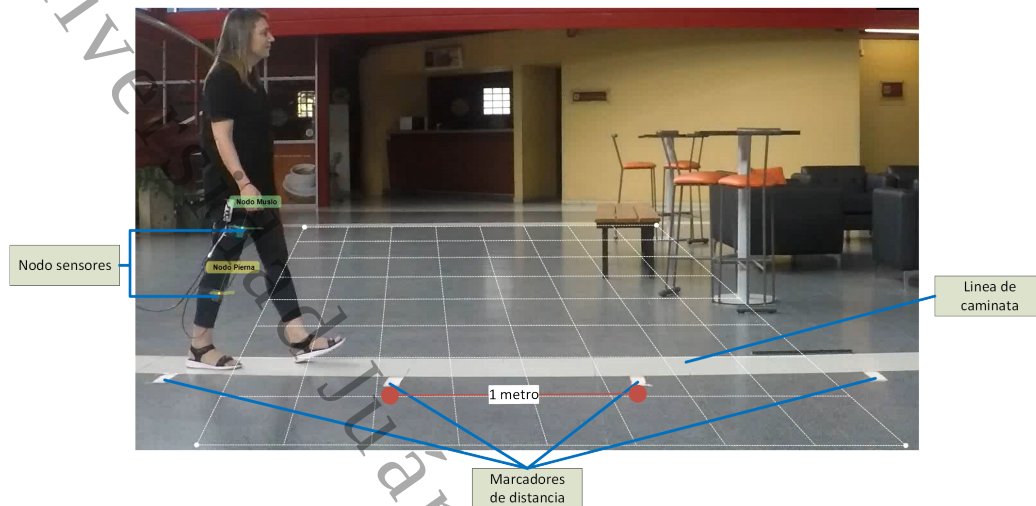


Figura 5.10. Ubicación de nodos sensores en el cuerpo de la persona para medición.

de posición como se muestran en las figuras 5.12a y 5.12b.

En la figura 5.12a se muestra el gráfico de posición en función de tiempo por cada eje x , y y z . En la gráfica 5.12b se muestra la posición en 3 dimensiones obtenida en función del tiempo. Es importante resaltar en el gráfico 5.12a que la posición en el eje x muestra un desplazamiento de 2,79 metros y comparando con la distancia obtenida de la suma de la distancia de pasos mostrado en la figura 5.13, se obtiene 2,80 metros. Esta última medición se obtiene usando Kinovea. Del mismo modo se utiliza los programas Skillspector y KWON3D y se registran distancias de 2,80 metros y 2,78 respectivamente.

5.4.1. Análisis de la marcha usando iGait

iGait es un software escrito para el análisis de la marcha usando conjuntos de datos de aceleraciones. Esta aplicación permite obtener parámetros de la marcha como cadencia, longitud de pasos y velocidad mediante el análisis de las señales de aceleraciones capturadas de una persona [51]. A continuación se realiza un análisis comparativo entre iGait, usando el set de datos obtenido para una persona sin alteración de la marcha de sexo femenino con 1,64[m] de altura, con el análisis visual usando el software Kinovea.

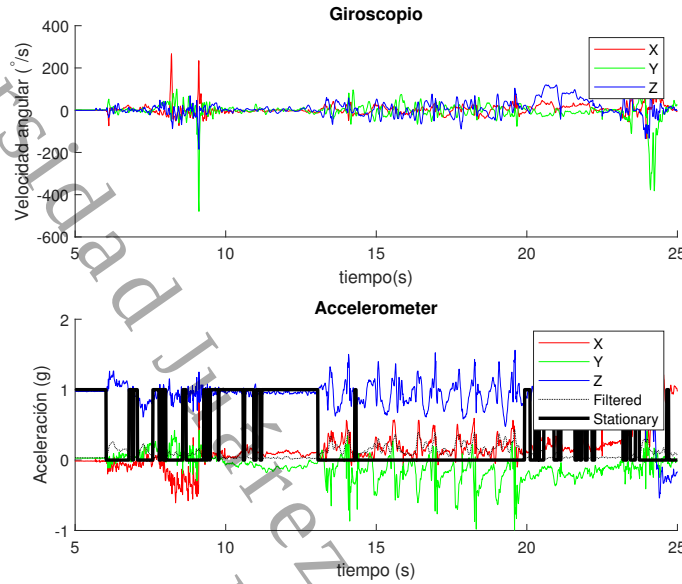
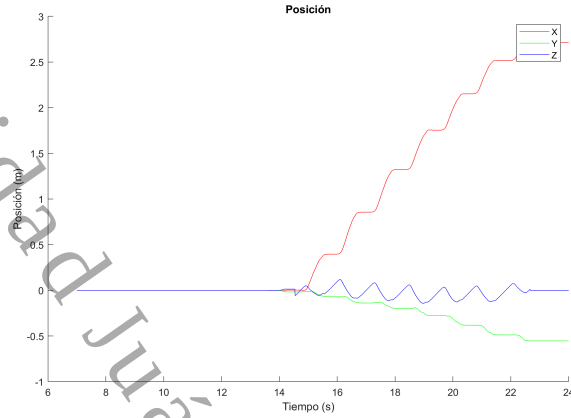


Figura 5.11. Gráficos de mediciones en crudo de Aceleración y Velocidad angular en 3D.

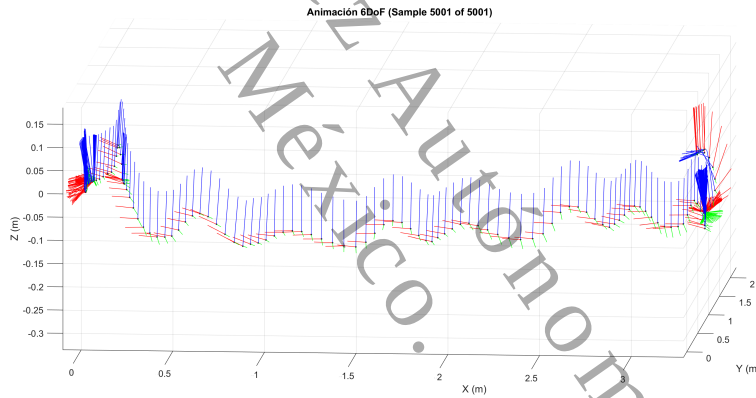
Para esta prueba se compara los parámetros de cadencia, longitud de paso y velocidad de la persona. En las figuras 5.13a, 5.13b, 5.13c, 5.13d, 5.13e, 5.13f se muestran las longitudes de pasos en toda la trayectoria que la persona ejecuta y luego con la información de tiempo transcurrido se calcula la velocidad y cadencia.

Por otro lado, en la figura 5.14 se muestra la gráfica de las señales para el conjunto de datos capturados sobre la misma trayectoria. Es necesario ordenar los datos de manera que iGait pueda identificarlos correctamente. Se configura el *sample rate* en 8 milisegundos es decir 125Hz, la distancia recorrida en 3 metros y el *threshold* se seleccionó en 0,4.

En la tabla 5.3 se muestran los resultados obtenidos usando los dos programas y técnicas para obtener cadencia, distancia de pasos y velocidad.



(a) Gráfico de posición por cada eje x , y y z en función del tiempo

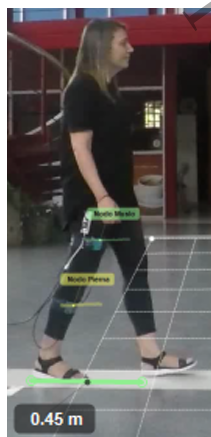


(b) Gráfico de posición 3D en función del tiempo.

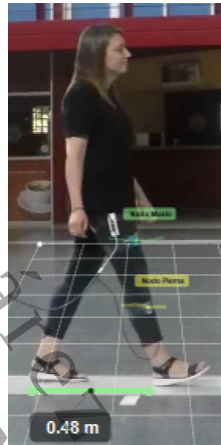
Figura 5.12. Gráficos de trayectoria lineal por cada eje y en tres dimensiones en función del tiempo

Tabla 5.3. Tabla comparativa de parámetros de la marcha obtenidos usando iGait y Kinovea

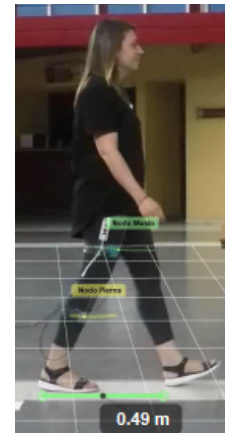
Software	Cadencia [pasos/min]	Distancia media de pasos [m]	Velocidad [m/s]
Kinovea	144,00	0,46	0,91
iGait	143,78	0,45	0,88



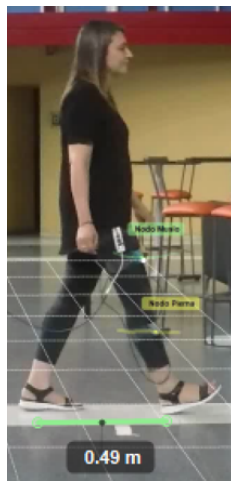
(a) Primer paso



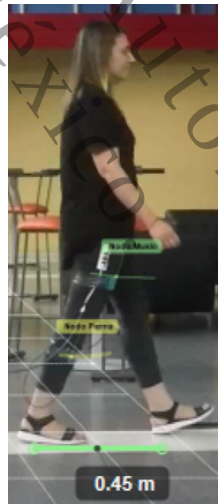
(b) Seg. paso



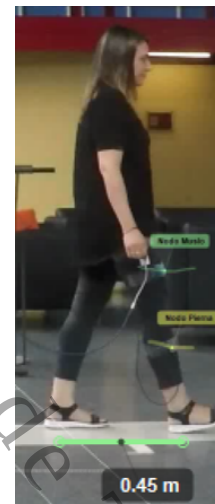
(c) Tercer paso



(d) Cuarto paso



(e) Quinto paso



(f) Sexto paso

Figura 5.13. Análisis de parámetros usando Kinovea

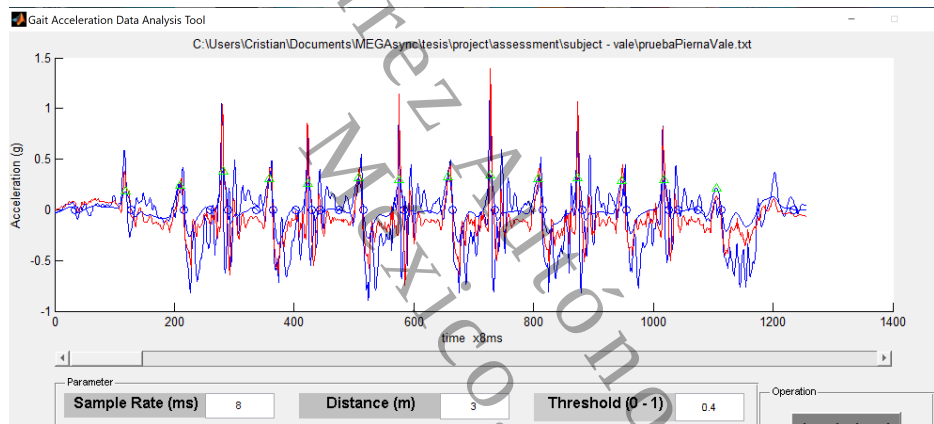


Figura 5.14. Resultado del análisis del set de datos usando iGait.

Capítulo 6

Contribuciones, conclusiones y trabajos futuros

6.1. Conclusiones

En el presente trabajo se identificaron las diferentes tecnologías disponibles en el ámbito de IoT para el sector médico para adquisición, procesamiento de transmisión de la información de datos biomecánicos.

Se realizó un análisis detallado de la problemática y se propuso una arquitectura de sensores corporales de plataforma abierta u *open source* para la captura de información biomecánica de la marcha.

Para realizar las pruebas se desarrolló un prototipo que consta de un servidor implementado en una Raspberry Pi y nodos sensores implementado en Arduino. En el servidor fue necesario desarrollar un programa en *Python* llamado RAMServer para gestión de las comunicaciones entre nodos, almacenamiento y procesamiento de la información y configuración de los nodos. Se evaluó el set de datos construido con las mediciones inerciales de la marcha de las personas comparando las mediciones con sistemas de análisis biomecánicos ópticos. Para la prueba se comparó el set de datos con los programas Kinovea, Skillspector, KWON3D y iGait.

De las pruebas realizadas se registra mediciones comparables de distancia recorrida por la persona, logrando medir 2,79m con los nodos sensores, 2,80m con Kinovea, 2,80m con Skillspecto y 2,78m con KWON3D.

Para el caso de parámetros de la marcha como se obtuvieron mediciones igualmente comparables entre el análisis realizado con el programa iGait ali-

mentado con el set de datos y las mediciones con Kinovea. Se obtuvo una cadencia de 144 pasos por minutos con Kinovea y 143,78 con iGait. Para la distancia de pasos se obtuvo una media de 0,46 metros para Kinovea y 0,45 metros para iGait. Finalmente la velocidad media obtenida es de 0,91 metros por segundo para Kinovea y 0,88 metros para iGait.

Finalmente el prototipo de la red IoT de sensores corporales permite obtener mediciones de la biomecánica de la marcha con la utilización de sensores inerciales de manera efectiva y con una precisión comparable los sistemas ópticos analizados. Esta arquitectura permite la captura de movimiento en entornos naturales para el paciente, fuera de hospitales y laboratorios.

6.2. Contribuciones

Para este proyecto se realizaron investigaciones previas complementarias en el área de IoT, sensores, inteligencia ambiental y captura de información que derivaron en las siguientes publicaciones:

- Alberoni C., Leon E., Wister M.A., Hernandez-Nolasco J.A. (2020) Wireless Sensor Network to Collect Training Data from Cycling. In: Barolli L., Xhafa F., Hussain O. (eds) Innovative Mobile and Internet Services in Ubiquitous Computing. IMIS 2019. Advances in Intelligent Systems and Computing, vol 994. Springer, Cham. doi: https://doi.org/10.1007/978-3-030-22263-5_40
- Leon E., Alberoni C., Wister M., Hernández-Nolasco J. A. (2018). "Flood Early Warning System by Twitter Using LoRa." Proceedings 2, no. 19: 1213. doi: <https://doi.org/10.3390/proceedings2191213>

6.3. Trabajos futuros

Durante la realización de la presente trabajo han surgido oportunidades de investigación y mejora derivadas de los avances y resultados obtenidos. A continuación se muestran los trabajos a futuros derivados:

- Se propone una herramienta de Visualización de la información biomecánica para que incluya un avatar de cuerpo completo.
- Complementar el servidor con nuevos modelos de algoritmos para identificación de actividades usando IA.

- Complementar el servidor con otros algoritmos de fusión y filtrado de señales.
- Mejorar en la distribución de código multiplataforma usando Docker.
- Incorporación de nuevos sensores en los nodos que permitan mejorar la calidad del análisis biomecánico.

Bibliografía

- [1] IEEE standard for information technology– local and metropolitan area networks– specific requirements– part 15.1a: Wireless medium access control (MAC) and physical layer (PHY) specifications for wireless personal area networks (WPAN).
- [2] IEEE standard for information technology–telecommunications and information exchange between systems local and metropolitan area networks–specific requirements - part 11: Wireless LAN medium access control (MAC) and physical layer (PHY) specifications.
- [3] IEEE standard for local and metropolitan area networks - part 15.6: Wireless body area networks.
- [4] IEEE standard for low-rate wireless networks.
- [5] *Promover el acceso a las tecnologías médicas y la innovación - Intersecciones entre la salud publica, la propiedad intelectual y el comercio.* World Intellectual Property, 2012.
- [6] Mario Alviano and Randy Goebel. Modeling and Control Balance Design for a New Bio-inspired Four-Legged Robot. In *Advances in Soft Computing*, pages 728–739, Xalapa, Mexico, 2019. Springer.
- [7] Anjali, Shikha, and M. Sharma. Wireless sensor networks: Routing protocols and security issues. In *Fifth International Conference on Computing, Communications and Networking Technologies (ICCCNT)*, pages 1–5, July 2014.
- [8] Luigi Atzori, Antonio Iera, and Giacomo Morabito. The Internet of Things: A survey. *Computer Networks*, 54(15):2787–2805, oct 2010.

- [9] Terrell Bennett, Roozbeh Jafari, and Nicholas Gans. An extended Kalman filter to estimate human gait parameters and walking distance. *Proceedings of the American Control Conference*, pages 752–757, 2013.
- [10] Elena Bergamini, Gabriele Ligorio, Aurora Summa, Giuseppe Vannozzi, Aurelio Cappozzo, and Angelo Maria Sabatini. Estimating orientation using magnetic and inertial sensors and different sensor fusion approaches: Accuracy assessment in manual and locomotion tasks. *Sensors (Switzerland)*, 14(10):18625–18649, 2014.
- [11] Teresa Beroaza W, Claudia Cartagena S, Ivan Caviedes S, Juan Caspedes G, Monica Gutierrez-Navas, Manuel Oyarzun G, SYlvia Palacios M, and Patricia Schönffeldt G. Prueba de caminata de seis minutos. *Revista chilena de enfermedades respiratorias*, 25:15 – 24, 00 2009.
- [12] Leonardo Emiro Contreras Bravo and Luis Fernando Vargas Tamayo. Generación de modelos de caminata bípeda a través de diversas técnicas de modelamiento, Nov 2006.
- [13] Yekta Said Can, Bert Arnrich, and Cem Ersoy. Stress detection in daily life scenarios using smart phones and wearable sensors: A survey. *Journal of Biomedical Informatics*, 92:103139, apr 2019.
- [14] Filippo Casamassima, Alberto Ferrari, Bojan Milosevic, Pieter Ginis, Elisabetta Farella, and Laura Rocchi. A wearable system for gait training in subjects with Parkinson’s disease. *Sensors (Switzerland)*, 14(4):6229–6246, 2014.
- [15] John J. Craig. *Introduction to Robotics: Mechanics and Control (3rd Edition)*. Pearson, aug 2004.
- [16] Dana da Silva, Maya Kaduri, Maria Poley, Omer Adir, Nitzan Krinsky, Janna Shainsky-Roitman, and Avi Schroeder. Biocompatibility, biodegradation and excretion of polylactic acid (PLA) in medical implants and theranostic systems. *Chemical Engineering Journal*, 340:9–14, may 2018.
- [17] GF Torres del Castillo. La representación de rotaciones mediante cuaterniones. *Miscelanea Matemtica*, pages 43–50, 1999.

- [18] Thierry Edoh. Internet of Things in Emergency Medical Care and Services. *Medical Internet of Things (m-IoT) - Enabling Technologies and Emerging Applications*, i:13, 2019.
- [19] T. Ellis, J. T. Cavanaugh, G. M. Earhart, M. P. Ford, K. B. Foreman, and L. E. Dibble. Which measures of physical function and motor impairment best predict quality of life in parkinson's disease? *Parkinsonism & Related Disorders*, 17(9):693–697, 10 2017.
- [20] Alberto Ferrari, Pieter Ginis, Michael Hardegger, Filippo Casamassima, Laura Rocchi, and Lorenzo Chiari. A mobile Kalman-filter based solution for the real-time estimation of spatio-temporal gait parameters. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 24(7):764–773, 2016.
- [21] Laura Gochicoa-Rangel, Uri Mora-Romero, Selene Guerrero-Zuniga, Monica Silva-Ceron, Sylvia Cid-Juarez, Monica Velazquez-Uncal, Adela Duran-Cuellar, Isabel Salas-Escamilla, Roberto Mejia-Alfaro, and Luis Torre-Bouscoulet. Prueba de caminata de 6 minutos: recomendaciones y procedimientos. *Neumología y cirugía de torax*, 74:127 – 136, 06 2015.
- [22] Ramón Gómez. *Estudio biomecánico de la marcha en pacientes con artrosis de cadera*. PhD thesis, Universidad de Valencia, 2004.
- [23] Raffaele Gravina, Parastoo Alinia, Hassan Ghasemzadeh, and Giancarlo Fortino. Multi-sensor fusion in body sensor networks: State-of-the-art and research challenges. *Information Fusion*, 35:1339–1351, 2017.
- [24] Julius Hannink, Malte Ollenschläger, Felix Kluge, Nils Roth, Jochen Klucken, and Bjoern M. Eskofier. Benchmarking Foot Trajectory Estimation Methods for Mobile Gait Analysis. *Sensors*, 17(9):1940, 2017.
- [25] Walter T. Higgins. A Comparison of Complementary and Kalman Filtering. *IEEE Transactions on Aerospace and Electronic Systems*, AES-11(3):321–325, 1975.
- [26] R E Kalman. A New Approach to Linear Filtering and Prediction Problems 1. *Journal of Fluids Engineering*, 82(Series D):35–45, 1960.
- [27] Conghui Liang, Marco Ceccarelli, and Giuseppe Carbone. Design and Simulation of Legged Walking Robots in MATLAB® Environment.

- MATLAB for Engineers - Applications in Control, Electrical Engineering, IT and Robotics*, 2011.
- [28] L. Ljung. Asymptotic Behavior of the Extended Kalman Filter as a Parameter Estimator for Linear Systems. *IEEE Transactions on Automatic Control*, 24(1):36–50, 1979.
 - [29] César Llamas, Manuel A. González, Carmen Hernández, and Jesús Vegas. Open source platform for collaborative construction of wearable sensor datasets for human motion analysis and an application for gait analysis. *Journal of Biomedical Informatics*, 63:249–258, 2016.
 - [30] Sebastian O H Madgwick. *AHRS algorithms and calibration solutions to facilitate new applications using low-cost MEMS*. PhD thesis, University of Bristol, 2014.
 - [31] S.O.H. Madgwick. An efficient orientation filter for inertial and inertial/magnetic sensor arrays. *Report x-io and University of ...*, page 32, 2010.
 - [32] John-Olof Nilsson and Isaac Skog. Inertial sensor arrays — A literature review. *2016 European Navigation Conference (ENC)*, (ii):1–10, 2016.
 - [33] David (ed.) Novillo-Ortiz. *La eSalud en la Región de las Américas: derribando las barreras a la implementación. Resultados de la Tercera Encuesta Global de eSalud de la Organización Mundial de la Salud*. 2016.
 - [34] Alexandros Pantelopoulos and Nikolaos G. Bourbakis. A survey on wearable sensor-based systems for health monitoring and prognosis. *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, 40(1):1–12, 2010.
 - [35] K. Pavithradevi. History and applications in body area network. *International Journal for Research in Applied Science and Engineering Technology*, V(II):167–170, feb 2017.
 - [36] Giovanni Perrone, Massimo Vecchio, Riccardo Pecori, and Raffaele Giarfeda. The day after mirai: A survey on mqtt security solutions after the largest cyber-attack carried out through an army of iot devices. In *Proceedings of the 2nd International Conference on Internet of Things*,

- Big Data and Security - Volume 1: IoTBDS*,, pages 246–253. INSTICC, SciTePress, 2017.
- [37] Josh Petersen, Pranav Bhounsule, and Andy Ruina. Dynamic walking matlab simulation guide.
- [38] Jerry E Pratt, Gill A Pratt, and Arthur C Smith. *Exploiting Inherent Robustness and Natural Dynamics in the Control of Bipedal Walking Robots by in partial fulfillment of the requirements for the degree of at the by*. PhD thesis, MIT, 2000.
- [39] Adnan Rashid and Osman Hasan. Wearable technologies for hand joints monitoring for rehabilitation: A survey. *Microelectronics Journal*, 88:173–183, jun 2019.
- [40] S. Rautmare and D. M. Bhalerao. Mysql and nosql database comparison for iot application. In *2016 IEEE International Conference on Advances in Computer Applications (ICACA)*, pages 235–238, Oct 2016.
- [41] Daniel Rodríguez-Martín, Carlos Pérez-López, Albert Samà, Joan Cabestany, and Andreu Català. A wearable inertial measurement unit for long-term monitoring in the dependency care area. *Sensors (Switzerland)*, 13(10):14079–14104, 2013.
- [42] T. Sarkodie-Gyan. *Neurorehabilitation Devices: Engineering Design, Measurement and Control*. McGraw-Hill’s biomedical engineering series. McGraw-Hill Education, 2005.
- [43] Isaac Skog, John Olof Nilsson, and Peter Handel. An open-source multi inertial measurement unit (MIMU) platform. *1st IEEE International Symposium on Inertial Sensors and Systems, ISISS 2014 - Proceedings*, pages 0–3, 2014.
- [44] Isaac Skog, John-Olof Nilsson, Peter Händel, and Arye Nehorai. Inertial Sensor Arrays, Maximum Likelihood, and Cram\’er-Rao Bound. pages 1–11, 2015.
- [45] Sebastijan Sprager and Matjaz B. Juric. *Inertial sensor-based gait recognition: A review*, volume 15. 2015.

- [46] E. Sánchez-Delacruz, F. Acosta-Escalante, Miguel A. Wister, J. A. Hernández-Nolasco, P. Pancardo, and J. J. Méndez-Castillo. *Gait Recognition in the Classification of Neurodegenerative Diseases*, pages 128–135. Springer International Publishing, Cham, 2014.
- [47] MathWorks Student Competitions Team. Matlab and simulink robotics arena: Walking robot.
- [48] Waldner. *Nanocomputers and swarm intelligence*. ISTE John Wiley, London Hoboken, NJ, 2008.
- [49] Yan Wang, Shuang Cang, and Hongnian Yu. A survey on wearable sensor modality centred human activity recognition in health care. *Expert Systems with Applications*, apr 2019.
- [50] Che Chang Yang, Yeh Liang Hsu, Kao Shang Shih, and Jun Ming Lu. Real-time gait cycle parameter recognition using a wearable accelerometry system. *Sensors*, 11(8):7314–7326, 2011.
- [51] Mingjing Yang, Huiru Zheng, Haiying Wang, Sally McClean, and Dave Newell. Igait: An interactive accelerometer based gait analysis system. *Computer methods and programs in biomedicine*, 108:715–23, 05 2012.
- [52] N. Yazdi, F. Ayazi, and K. Najafi. Micromachined inertial sensors. *Proceedings of the IEEE*, 86(8):1640–1659, 1998.
- [53] Paul Zarchan and Howard Musoff. *Fundamentals of Kalman Filtering: A Practical Approach (Progress in Astronautics & Aeronautics)*. Amer Inst of Aeronautics &, 2001.