ANÁLISIS COMPARATIVO DE ALGORITMOS DE ESTIMACIÓN DE LA LONGITUD DE ZANCADA EN MARCHA HUMANA UTILIZANDO SISTEMAS INERCIALES DE MEDIDA.

ENMANUEL BERRUECOS GÓMEZ
MARLIO ALEJANDRO CHICUE RESTREPO

Universidad del Cauca
Facultad de Ingeniería Electrónica y Telecomunicaciones
Departamento de Electrónica, Instrumentación y Control
Ingeniería en Automática Industrial
Popayán, 2021

ANÁLISIS COMPARATIVO DE ALGORITMOS DE ESTIMACIÓN DE LA LONGITUD DE ZANCADA EN MARCHA HUMANA UTILIZANDO SISTEMAS INERCIALES DE MEDIDA.

Trabajo de grado presentado como requisito para obtener el título de Ingeniero en Automática Industrial

Enmanuel Berruecos Gómez

Marlio Alejandro Chicue Restrepo

Director: M.Sc. Diego Enrique Guzmán Villamarín

Codirector: PhD. Carlos Felipe Rengifo

Universidad del Cauca

Facultad de Ingeniería Electrónica y Telecomunicaciones

Departamento de Electrónica, Instrumentación y Control

Ingeniería en Automática Industrial

Popayán, 2021

Agradecimientos

...A Dios por habernos acompañado y guiado a lo largo de toda la carrera, por darnos fortaleza en los momentos más difíciles y sabiduría cuando más lo necesitamos.

...Le damos gracias a nuestras familias y especialmente a nuestros padres por todo el apoyo brindado, por los buenos valores que nos han inculcado y el ánimo que nos brindaron para cumplir nuestros sueños.

...A nuestro director de tesis M.Sc Diego Enrique Guzmán quien con sus conocimientos y experiencia nos guió y acompaño en todo el proceso de investigación.

...Igualmente, agradecemos al PhD. Carlos Felipe Rengifo, quien como apoyo siempre mostró gran entusiasmo e interés por el proyecto, que, basado en sus conocimientos y habilidades desde el campo de ingeniería, contribuyó al trabajo que hoy presentamos.

...A nuestros amigos, familiares y compañeros que siempre estuvieron dispuestos a ofrecernos su ayuda y colaboración.

...A la universidad del Cauca, especialmente a los profesores del departamento de electrónica, instrumentación y control, por todo el apoyo y conocimientos brindados a lo largo de toda la carrera.

Resumen

El objetivo del presente estudio es comparar la exactitud de diferentes algoritmos, para la estimación de la longitud de zancada en marcha humana, utilizando dispositivos IMU.

En el ámbito médico, el estudio de la marcha ha interesado desde tiempos remotos, haciendo de éste un estudio complejo y de gran utilidad del cual se pueden encontrar diversos sistemas de medición ampliamente utilizados. La variabilidad de la longitud de zancada durante el ciclo de marcha de un ser humano ha sido correlacionada con el riesgo de caída, la existencia de las enfermedades de Parkinson y de Huntington, y la presencia de lesiones cerebrales; lo que justifica la importancia del análisis de la marcha humana en las áreas de la salud y la ciencia en general, por lo cual, se desarrolló un experimento comparativo de la medición de la longitud de la zancada en una población de adultos sanos sin patologías con incidencia en la marcha, mediante dispositivos IMU, para determinar el grado de exactitud de éstos algoritmos de estimación frente a un sistema basado en un algoritmo de visión artificial (sistema óptico de medida). Una vez completado el experimento, se realizó un análisis estadístico comparativo entre los algoritmos IMU y el algoritmo del sistema óptico de medida. Se determinó si estos grupos de datos pertenecían a una misma población mediante el estadístico *Kruskal-Wallis*; y mediante el error RMSE de todos los datos, se determinó el grado de exactitud de los algoritmos IMU frente al algoritmo basado en visión artificial.

Como resultado, el estadístico *Kruskal-Wallis* rechazó la hipótesis nula H_0, y se aceptó la hipótesis alternativa H_1 de que por lo menos 1 grupo de datos de los algoritmos IMU proviene de una población con una distribución distinta al algoritmo basado en visión artificial. La exactitud de los algoritmos IMU tuvo un resultado poco satisfactorio en comparación de los resultados obtenidos mediante el sistema óptico de medida, con errores RMSE de hasta 83,12 % y promedio de zancada de hasta 15,72 centímetros.

Palabras clave: Longitud de zancada, Dispositivos IMU, error RMSE, Visión artificial, Kruskal-Wallis, Sistemas inerciales de medida, ciclo de marcha

Abstract

The objective of the present study is the comparison of different algorithms exactness, for estimating human gait stride length, using IMU devices.

In the medical field, the study of gait has been of interest since ancient times, making it a complex and highly useful study from which various widely used measurement systems can be found. The variability of stride length during the human gait cycle has been correlated with the risk of falling, the existence of Parkinson's and Huntington's diseases, and the presence of brain lesions; which justifies the importance of human gait analysis in the areas of health and science in general, whence, a comparative experiment was developed to measure the length of stride in a population

of healthy adults without pathologies with incidence in gait, using IMU devices, to determine the exactness degree of these estimation algorithms against a system based on an artificial vision algorithm (optical measurement system). Once the experiment was completed, a comparative statistical analysis was performed between the IMU algorithms and the algorithm of the optical measurement system. It was determined whether these data groups belonged to the same population using the statistic *Kruskal-Wallis*; and by means of the RMSE error of all the data, the exactness degree of the IMU algorithms was determined compared to the artificial vision based algorithm.

As a result, the statistician *Kruskal-Wallis* rejected the null hypothesis H_0, and the alternative hypothesis H_1 was accepted that at least 1 group of data from IMU algorithms comes from a population with a different distribution than the algorithm based on computer vision. IMU algorithms exactness had an unsatisfactory result compared with results obtained by using the optical measurement system, with RMSE errors of up to 83.12 % and average stride of up to 15.72 centimeters.

Keywords: Stride length, IMU devices, RMSE error, Machine vision, Kruskal-Wallis, Inertial measurement systems, gait cycle

Tabla de contenido

Lista de figuras

Lista de tablas

Capítulo 1

Introducción.

En éste capítulo se introduce al lector en el contexto del proyecto, trabajos previos de distintos autores, pregunta de investigación y objetivos, mediante el estado del arte, planteamiento del problema y objetivos general/específicos. También se presenta la estructura de la monografía.

1.1. Estado del arte.

La protección de la salud humana, el bienestar y la mejora de la calidad de vida, han sido siempre unos de los factores más acogidos por la ciencia y la tecnología en general. En el ámbito médico, el estudio de la marcha ha interesado desde tiempos remotos haciendo de éste un estudio complejo y de gran utilidad, del cual se pueden encontrar diversos sistemas de medición ampliamente utilizados; pero en la antigüedad, el hombre únicamente disponía de su capacidad de observación [7], y este método presenta limitaciones que pueden generar errores en la toma de la medida, como son la inexperiencia y los errores derivados de la subjetividad del sistema visual humano [8]. En el último siglo, el desarrollo de las técnicas de análisis de marcha y movimiento, han experimentado un desarrollo bastante significativo para su análisis, comprensión, y posterior corrección mediante técnicas y algoritmos que facilitan el cálculo de parámetros específicos en la marcha de manera más exacta y con menor ruido en la medición.

Actualmente existen diferentes métodos para el estudio de la marcha [9, 10] tanto desde la parte médica como algoritmos de análisis de marcha por computador. Concretamente, en el estudio de las variables espacio-temporales que hacen parte como subconjunto de todas las variables de la marcha, se puede observar que se han venido realizando estudios de este tipo de variables desde el siglo XIX cuando recién fabricaron las primeras cámaras fotográficas. En 1836 los hermanos Weber llevaron a cabo el primer análisis mecánico de la marcha humana; en su obra: ”Mechanik der Mensheliehen Gehwerkzeuge Gottingen”, en la cual describen las fases de la marcha humana [7, 11].

Debido a esto se pueden clasificar los métodos de análisis de la marcha humana según el sistema de medición a implementar. Así, tenemos entonces que uno de estos métodos, en el campo de analisis de la marcha, es el método de sistemas ópticos, los cuales consisten en el 'análisis de vídeo', y pueden realizarse tanto empleando marcadores en el sujeto, como prescindiendo de ellos. Durante el estudio puede utilizarse una sola cámara, dando lugar a un análisis en dos dimensiones, o dos cámaras para lograr un análisis en tres dimensiones. La desventaja de un estudio en dos dimensiones es el error que puede producirse por el movimiento en planos no captados por la imagen. Al introducir una segunda cámara se omite este efecto pero es necesario que ambas cámaras capten los puntos de estudio para la reconstrucción del movimiento. También se encuentra dentro de la literatura, el uso de la electromiografía como método para realizar análisis de la marcha; consiste en la utilización de sensores para detectar y medir las pequeñas corrientes eléctricas que se producen durante la contracción de los músculos. La colocación de los sensores puede ser cutánea o intramuscular, siendo esta última opción más exacta. Además de analizarse la intensidad de la señal captada, es posible determinar qué músculos intervienen en el movimiento detectando presencia o ausencia de señal en los mismos. Existen estudios realizados en laboratorios especializados, que implementan diversas técnicas como la electromiografía, videogrametría y la dinamometría [9, 12] para medir los parámetros espacio-temporales de la marcha, y una amplia variedad de sub-parámetros derivados de ellos; estas investigaciones están más orientadas a la parte biomédica y química; además, estos sistemas de análisis han sido utilizados en otros campos como la biomecánica deportiva, ergonomía, diseño y evaluación de calzado, evaluación de riesgo en actividades físicas, o simplemente para el estudio general de la biomecánica [13]. En algunos casos con este tipo de métodos, se realizan estudios complementarios cómo, el registro de la actividad muscular con electromiografía dinámica (EMG), la medición del consumo energético, o el monitoreo del gasto cardio-respiratorio mediante espirometría [6, 10, 12, 14].

Los sistemas optoelectrónicos utilizados para medición y análisis de la marcha captan señales luminosas de marcadores colocados en el cuerpo del sujeto a medir, y las convierten en señales eléctricas [1, 4, 10, 15]. A pesar de que se trata de un completo y minucioso método de análisis de la marcha, no resulta muy práctico en el ámbito del análisis clínico, debido al alto costo y complejidad del equipo; hay que añadir el amplio espacio de trabajo necesario para mantener una línea de visión libre de obstáculos entre el sujeto y los sistemas de medida, además de la complejidad y lentitud a la hora de la toma de datos que suponen un obstáculo en la repetitividad de los resultados experimentales.

Existen también sistemas basados en la medición de las fuerzas, velocidades, aceleraciones o presiones generadas durante la marcha en las extremidades inferiores (sistemas no ópticos), uno de ellos es el sistema basado en plataformas de fuerza, estas son instrumentos utilizados para medir la fuerza de reacción del suelo generada por un cuerpo situado encima [16, 17]. Esta fuerza es medida mediante transductores colocados en las esquinas de la plataforma que convierten la medición de fuerza en una señal eléctrica, pudiendo así calcular el centro de presión. La mayor restricción de este sistema es el reducido espacio en el que el sujeto puede pisar, por lo que el

número de pasos consecutivos a medir es limitado; además, esa restricción del espacio produce una marcha antinatural en los sujetos, alterando el análisis de la marcha normal [13]. Como una mejora del método aparecieron los tapices o alfombras instrumentalizadas, solucionando la restricción del espacio en las plataformas de fuerza. Las ventajas con respecto a los anteriores sistemas, radican en la portabilidad y facilidad de manejo además del ahorro en tiempo debido a la automatización en el cálculo de los parámetros obtenidos. Sin embargo, presentan ciertas limitaciones dado que sólo se obtiene información de la presión ejercida sobre los sensores, sin tener en cuenta la dirección ni las componentes del vector de fuerza.

Otro método similar, son los zapatos que incorporan un sensor de fuerza y una unidad de medición inercial en la punta del pie, un sensor de fuerza y una unidad de medición inercial en el tacón (zapatos instrumentalizados), permitiendo así realizar un análisis tanto cinemático como cinético [18]. Esta característica implica que se pueda realizar un análisis completo de la marcha. Además no restringen el número de pasos ni son excesivamente aparatosos para el paciente. Sin embargo, siguen siendo un sistema de elevado costo y relativamente complejo. Al mismo tiempo, y a pesar de haberse demostrado la utilidad de los zapatos para el estudio de ciertas patologías, la inclusión de los sistemas de medida en la suela del zapato, introduce ciertas anomalías en la marcha natural del paciente que pueden sesgar los resultados; como mejora a este método fueron diseñadas las plantillas de fuerza que, al igual que los zapatos instrumentalizados, permiten realizar un análisis completo de la marcha y no limitan el número de pasos de la medida. Una de las ventajas de su uso es que incluyen sensores colocados a lo largo de su extensión por lo que es posible determinar la presión ejercida en las diferentes zonas del pie [9].

Dentro de lo que son los métodos no ópticos, se encuentra también una gran cantidad de estudios realizados con sensores IMU (del inglés *inertial measurement unit*). Su variación está en el área de colocación de los sensores, y la cantidad de ellos. Generalmente son utilizados como medio de comparación con otros métodos de análisis de parámetros de la marcha, y de la obtención de sus datos pueden derivar las mediciones de distancias, velocidades, alturas del pie, etc. Existen estudios donde se posicionan hasta 8 sensores de este tipo para determinar todo el mecanismo y la cinemática de la marcha en animaciones 3D o sistemas computarizados. Otros también son colocados en un área específica para cumplir una función menos rigurosa y detallada. Evidentemente se realizan cálculos especiales sobre los datos obtenidos por las IMUs (dependiendo de lo que se necesite obtener), para derivar otras variables de estudio, como por ejemplo, los cálculos de los filtros de Kalman, la compensación de la gravedad, matrices de posición, etc. Pero generalmente se hace uso de la longitud de zancada como herramienta fundamental para el cálculo de otros parámetros de la marcha, y de ella se derivan otros parámetros [1, 19, 20].

1.2. Planteamiento del problema

La variabilidad de la longitud de zancada durante el ciclo de marcha de un ser humano ha sido correlacionada con el riesgo de caída [21–23], la existencia de las enfermedades de Parkinson [24] y de Huntington [25], y la presencia de lesiones cerebrales [26], lo que justifica su importancia en el análisis de marcha humana. Dicha variable se mide a partir de sistemas de captura de movimiento que pueden estar constituidos por cámaras infrarrojas [4, 9], plataformas de fuerza [4, 15, 27], o dispositivos inerciales [4, 9]. Los primeros son los de mayor exactitud; sin embargo, son enormemente costosos, requieren de condiciones controladas de iluminación, y de la utilización de marcadores reflectivos que se deben ubicar sobre posiciones específicas del cuerpo del participante, lo que aumenta considerablemente la duración de una prueba.

En cuanto a las plataformas de fuerza, estas son mucho menos costosas que los sistemas de cámaras infrarrojas; no obstante, tienen el inconveniente de que solo permiten estudiar las fuerzas de reacción entre el pie y el suelo, pero no la evolución de las posiciones articulares.

Finalmente, están los dispositivos inerciales, que son de bajo costo pero que requieren de algoritmos de procesamiento complejos, como las diferentes variantes del filtro de Kalman [1, 19, 20]. La principal dificultad con los sistemas de captura de movimiento basados en dispositivos inerciales se presenta cuando se estiman variables espaciales, debido a que se requiere de la integración de señales de aceleración, procedimiento que es afectado por problemas de deriva numérica debidos a componentes espurios de frecuencia cero denominados *offset*. Pese a que en el contexto de la marcha humana se han propuesto numerosos algoritmos para la estimación de la longitud de zancada [1–3, 28–33], la exactitud de los unos con respecto a los otros es aún materia de debate, dado que las comparaciones han sido realizadas por quienes proponen los algoritmos. De lo anterior, surge la presente propuesta de investigación cuyo propósito es dar respuesta al interrogante ¿Qué diferencias se presentan en cuanto a la exactitud de los algoritmos para la estimación de longitud de zancada en marcha humana utilizando unidades inerciales de medida?

1.3. Objetivos

1.3.1. Objetivo general

Comparar la exactitud de diferentes algoritmos para estimación de la longitud de zancada en marcha humana utilizando dispositivos IMU.

1.3.2. Objetivos específicos

1. Implementar tres algoritmos para la estimación de la longitud de zancada en marcha humana basados en dispositivos IMU.
2. Desarrollar un experimento de estimación de la longitud de zancada sobre una población de adultos jóvenes sin antecedentes de patologías con incidencia en la marcha.
3. Determinar la exactitud de cada uno de los algoritmos implementados con respecto a un sistema óptico de medida.

1.4. Estructura de la monografía

Este documento se divide en 5 capítulos cuyo contenido trata las temáticas más relevantes para la realización del proyecto. A continuación, se realiza una breve descripción de los capítulos:

En el **capítulo 1**: Introducción, En este primer capítulo se presenta el contenido preliminar y conceptos base del trabajo de grado, el planteamiento del problema, los objetivos para la realización del mismo y la organización del documento. En el **capítulo 2:** Se realiza la conceptualización, se define la metodología para el desarrollo e implementación de los algoritmos para la estimación de longitud de zancada en marcha humana utilizando unidades inerciales de medida. En el **capítulo 3:** Este capítulo desarrolla la experimentación de los algoritmos para la estimación de longitud de zancada en marcha humana, donde se describe el protocolo de experimentación que se llevó a cabo con los sujetos de prueba. El **capítulo 4:** presenta el proceso de experimentación llevado a cabo y el análisis de los resultados obtenidos haciendo comparaciones estadísiticas entre algoritmos y el sistema óptico de medida. Finalmente, en el **capítulo 5:** se presenta la discusión, conclusiones sobre el análisis de resultados y trabajos futuros.

Capítulo 2

Materiales y métodos

Para la implementación de los algoritmos de estimación de la longitud de zancada en marcha humana mediante sistemas inerciales de medida, es necesario tener conocimientos previos sobre temas cómo: el ciclo de la marcha, tipos de dispositivos IMU, ventajas y desventajas de uso en el proyecto; Además, hay que establecer las herramientas, materiales, métodos y soluciones que se implementaron en el desarrollo del mismo, así como los criterios de selección de los diferentes materiales usados. Por lo cual, en este capítulo se establece: conceptualización teórica, metodología y herramientas empleadas para la selección de algoritmos y, finalmente, la implementación de los mismos.

2.1. Conceptualización Teórica

En ésta sección se abordará la conceptualización de los temas de interés para el proyecto los cuales son el ciclo de la marcha, los sistemas inerciales, dispositivos IMU y teoría de orientación de un objeto en tres dimensiones.

2.1.1. Ciclo de la marcha humana.

El ciclo de la marcha humana inicia cuando el talón de un pie toca el suelo y termina con el contacto en el suelo del mismo talón. La distancia que existe entre el inicio y finalización del ciclo de la marcha se denomina paso completo o longitud de zancada [6].

Dicho ciclo comprende dos fases, la fase de apoyo o descanso que abarca el 60 % del ciclo y la fase de oscilación o balanceo que comprende el 40 % restante, como se observa en la Figura 2.1. En cada ciclo se da la acción de avance de la pierna correspondiente con respecto al eje

de progresión de la marcha, por lo que para dar una zancada cada pierna pasa por una fase de descanso y una fase de balanceo [5] como se observa en la Figura 2.2.

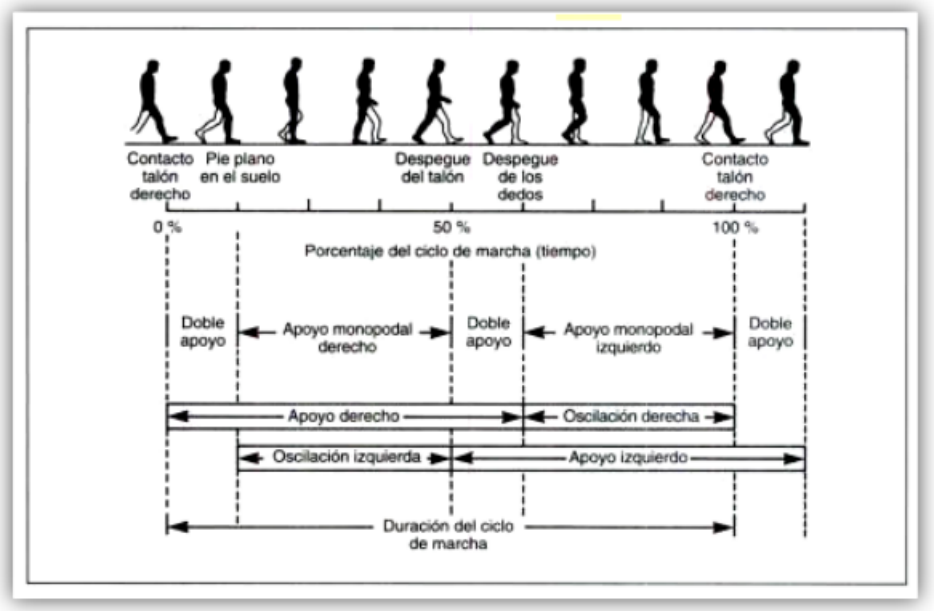


Figura 2.1: Fases de la marcha humana. [4]

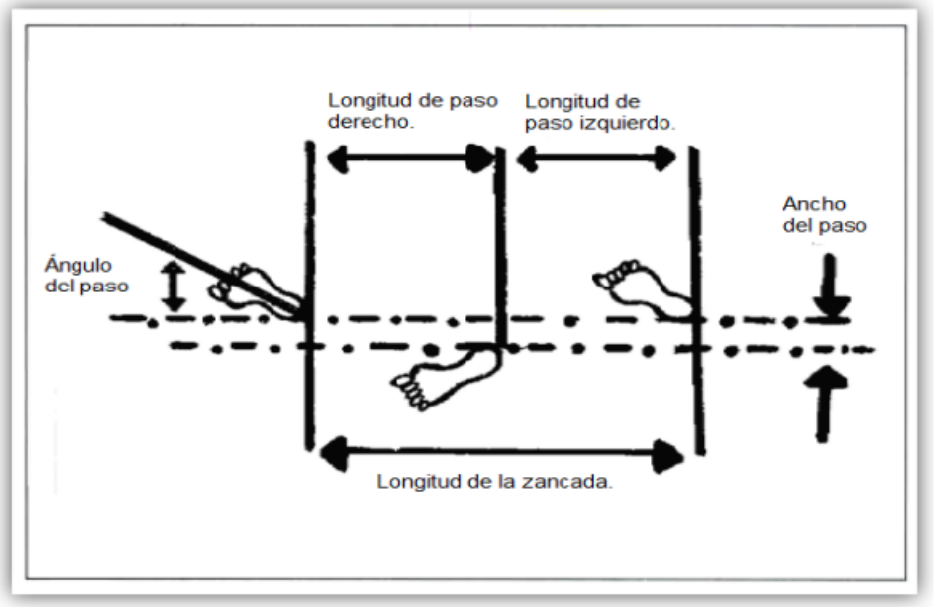


Figura 2.2: Parámetros espaciales de la marcha humana. [5]

De lo anterior se pueden apreciar algunos de los parámetros que se encuentran en la marcha, pero existen otros parámetros que se pueden analizar en la misma [12, 34] cómo parámetros cinemáticos: en el plano frontal cómo oblicuidad de la plevis, aducción-abducción de cadera; en el plano sagital cómo Basculación de pelvis, flexo-extensión de cadera; en el plano transversal cómo rotación de pelvis, rotación interna-externa de cadera; entre otros. Cinéticos cómo: fuerzas de reacción antero-posterior, medio-lateral y vertical, entre otros. La Figura 2.3 muestra los planos de ubicación corporal.

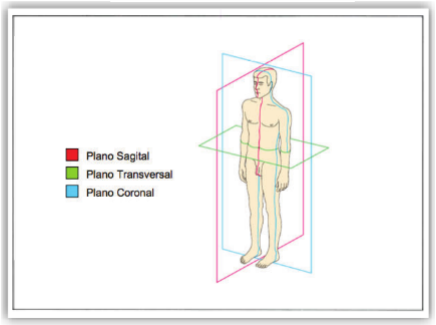


Figura 2.3: Planos observables en el cuerpo humano. [6]

La longitud de zancada es un parámetro espacial de la marcha, existen otros parámetros espaciales, temporales y espacio-temporales [4, 6, 34–36] que se describirán también a continuación para mayor comprensión de la marcha.

Parámetros espaciales:

- **Longitud de zancada:** Distancia lineal entre dos contactos de talón consecutivos de la misma extremidad.
- **Longitud de paso:** Distancia lineal entre el contacto inicial del talón de una extremidad y el de la extremidad contralateral (40 cm aprox. aunque depende de la estatura del individuo).
- **Ancho de paso o Amplitud de base:** La distancia entre ambos pies, generalmente entre los talones.
- **Altura del paso:** El movimiento de las extremidades inferiores otorga una altura de alrededor de 5 cm al paso, evitando el arrastre de los pies.
- **Ángulo del paso o ángulo de la marcha:** Se refiere a la orientación del pie durante el apoyo. El eje longitudinal de cada pie forma un ángulo con la línea de progresión (línea de dirección de la marcha); normalmente, está entre 5° y 8°.

Parámetros temporales:

- **Apoyo:** Porcentaje del ciclo total de la marcha durante el cual el cuerpo se encuentra apoyado sobre una sola pierna.
- **Balanceo:** Porcentaje del ciclo de la marcha durante el cual la extremidad inferior permanece en el aire y avanza hacia adelante.
- **Doble apoyo:** Porcentaje del ciclo de la marcha en el cual ambos pies contactan el suelo.
- **Periodo de zancada:** Lapso de tiempo en el que el transcurren dos eventos idénticos sucesivos del mismo pie, generalmente entre dos contactos iniciales de la misma extremidad inferior.
- **Periodo de soporte o apoyo:** El tiempo que transcurre desde que el pie hace contacto con el piso, hasta el momento de despegue de los dedos del mismo pie.
- **Periodo de balaceo:** Es el tiempo transcurrido entre el instante de despegue de los dedos hasta el punto de contacto inicial de un mismo pie.
- **Cadencia:** Es el número de pasos por unidad de tiempo, generalmente se mide en un minuto. La frecuencia determina el ritmo y rapidez de la marcha.

Parámetros espacio-temporales:

- **Velocidad:** Es la relación entre la distancia recorrida en dirección de la marcha y el tiempo de su duración.
- **Velocidad de Balanceo:** Tiempo en que se demora un miembro inferior desde la aceleración inicial hasta el siguiente paso.
- **Velocidad media:** Producto de la cadencia por la longitud de la zancada expresada en m/s.

Cómo se observó, el entendimiento de la marcha es de vital importancia para estimar cualquier parámetro en ella, y debido a que este proyecto se centra en la estimación de la longitud de zancada, es necesario entender de manera sencilla la complejidad implicita en dicha acción. Para facilitar el análisis, cada uno de los ciclos que la componen se dividen en dos fases, la fase de apoyo y la fase de balanceo, quienes a su vez, se dividen en subfases [4, 5, 11, 37] que a continuación se describen:

2.1.1.1. Fase de apoyo.

Tabla 2.1: Fase de apoyo. [4]

SUBFASE.	DESCRIPCIÓN.	DURACIÓN DURANTE EL CICLO.
Contacto inicial.	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 esta flexionada, la rodilla en extensión y el tobillo está en posición neutra.	Del 0 % al 2 %
Respuesta a la carga.	El pie realiza una flexión plantar hasta quedar en contacto total con el piso, el peso del cuerpo se transfiere a ésta misma extremidad.	Del 0 % al 10 %
Soporte medio.	Comienza con el despegue de dedos del miembro contralateral y se prolonga hasta el despegue de talón del pie estacionario.	Del 10 % al 30 %
Soporte terminal.	El talón se levanta para desplazar el peso hacia los dedos y transferir la carga al pie contralateral, el cual, entra en contacto con el piso.	Del 30 % al 50 %
Pre-balanceo	Transición entre la fase de soporte y la de balanceo. Se entrega totalmente la carga al miembro contralateral.	Del 50 % al 60 %

2.1.1.2. Fase de oscilación o balanceo.

Tabla 2.2: Fase de oscilación o balanceo. [4]

SUBFASE.	DESCRIPCIÓN.	DURACIÓN DURANTE EL CICLO.
Balanceo inicial.	Inicia cuando los dedos del pie se despegan del piso y termina cuando la rodilla alcanza la flexión máxima durante la marcha (60°), el muslo se encuentra directamente debajo del cuerpo y paralelo a la extremidad inferior contralateral que, en este instante, soporta el peso corporal.	Del 60 % al 73 %
Balanceo medio.	El muslo continúa avanzando y la rodilla, que ha alcanzado la flexión máxima, ahora se extiende, de manera que el pie permanece despegado del suelo y termina cuando la tibia se dispone en posición perpendicular al piso.	Del 73 % al 87 %
Balanceo terminal.	Inicia con la posición vertical de la tibia, continúa a medida que la rodilla se extiende completamente y la extremidad se prepara para aceptar la carga durante el contacto inicial.	Del 87 % al 100 %

Al hacer análisis del ciclo de la marcha es importante tener en cuenta cada una de las subfases mencionadas dependiendo del fin de estudio, por esto, es importante clarificar gráficamente dicha acción y en la Figura 2.4, se enseñan las subfases de la fase de apoyo y la fase de balanceo.

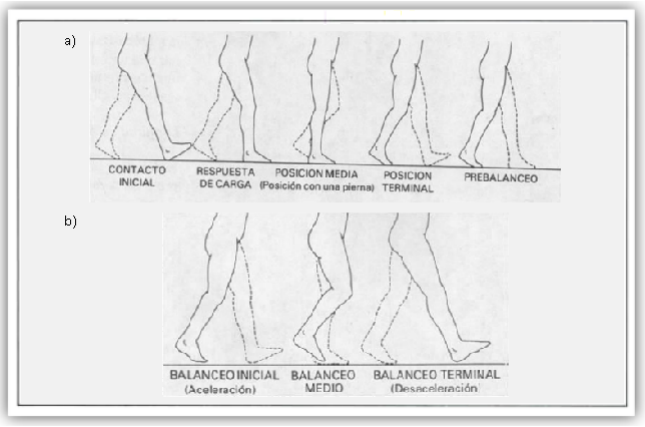


Figura 2.4: **a)** Subfases de apoyo, **b)** Subfases de balanceo. [4]

Teniendo en cuenta lo anterior, la subfase de mayor interés para el proyecto a la hora de estimar la longitud de zancada será la subfase de posición media [1–3] puesto que en éste punto, la energía potencial es la máxima y la cinética es cero.

2.1.2. Unidades de medición inercial.

Un sistema de referencia inercial es un sistema de referencia en el que las leyes del movimiento cumplen las leyes de Newton y, por tanto, la variación del momento lineal del sistema es igual a las fuerzas reales sobre el sistema. Un sensor inercial es aquel capaz de medir valores respecto a dichos marcos de referencia (acelerómetros y giroscopios) y una unidad de medición inercial es un dispositivo que integra varios sensores inerciales junto con un reloj que permite asignar tiempo tanto a los valores medidos por los sensores inerciales como a los valores obtenidos por mecanismos de calibración debida a posibles perturbaciones originadas por cambios de temperatura o a otros agentes externos. Este tipo de unidades implementan internamente ejes ortogonales sobre los cuales se montan los sensores de manera que a cada eje se le asigna un acelerómetro y un giroscopio. La información suministrada por una IMU es la aceleración lineal y la velocidad angular correspondientes a cada uno de los ejes del sistema con el correspondiente valor de tiempo común para estos valores. La frecuencia de salida de datos de una unidad inercial oscila en función de las características *hardware* del dispositivo. [4–6, 15, 38, 39]

2.1.2.1. Acelerómetros.

Los acelerómetros son sensores inerciales que basan su funcionamiento en la ley fundamental de la dinámica o segunda ley de Newton. Proporcionan una medida de la segunda derivada de la posición. Ésta medida se obtiene a partir de la fuerza de inercia que sufre una masa dispuesta convenientemente. Existen diversos tipos de acelerómetros dependiendo de la naturaleza del transductor:

- **Acelerómetros mecánicos:** Emplean una masa inerte y resortes elásticos. Los cambios se miden con galgas extensiométricas. Incluyendo sistemas de amortiguación que evitan la propia oscilación. También se emplean sistemas rotativos desequilibrados que originan movimientos oscilatorios cuando están sometidos a aceleración (servoacelerómetros) o detectan el desplazamiento de una masa inerte mediante cambios en la transferencia de calor (acelerómetros térmicos).

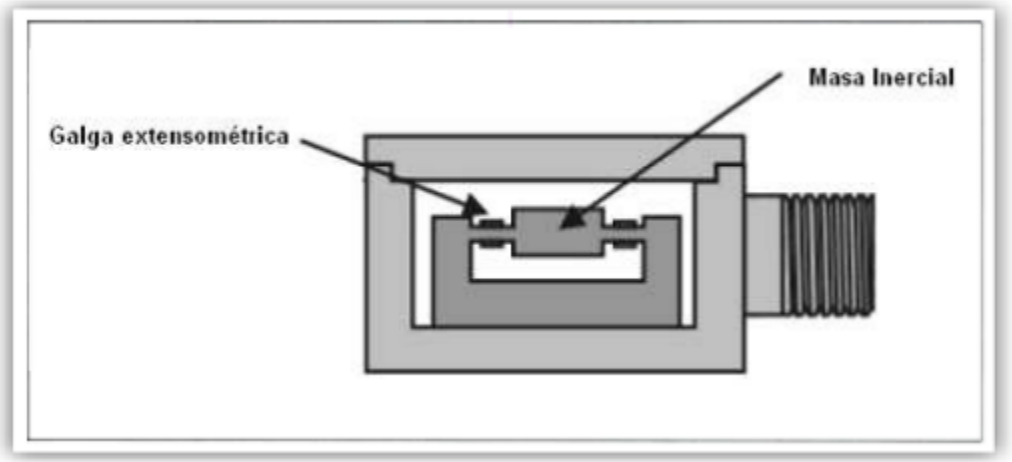


Figura 2.5: Esquema acelerómetro mecánico.

- **Acelerómetros capacitivos:** Modifican la posición relativa de las placas de un microcondensador cuando está sometido a aceleración. El movimiento paralelo de una de las placas del condensador hace variar su capacidad. Los acelerómetros capacitivos basan su funcionamiento en la variación de la capacidad entre dos o más conductores entre los que se encuentra un dieléctrico, en respuesta a la variación de la aceleración.

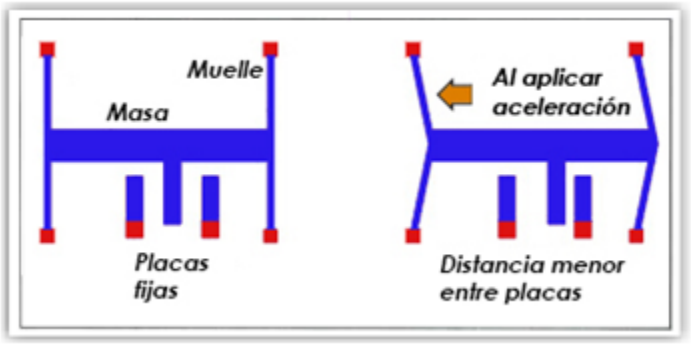


Figura 2.6: Esquema acelerómetro capacitivo.

- **Acelerómetros piezoeléctricos:** El dispositivo emplea una masa en contacto directo con un dispositivo piezoeléctrico (o cristal). Cuando un movimiento variables es aplicado al acelerómetro, el cristal experimenta una fuerza de excitación variable ($F = m \cdot a$) que ocasiona que una carga eléctrica q se desarrolle en él, así:

$$q = dij \cdot F = dij \cdot m \cdot a \tag{2.1}$$

donde q es la carga y dij es el coeficiente piezoeléctrico del material.

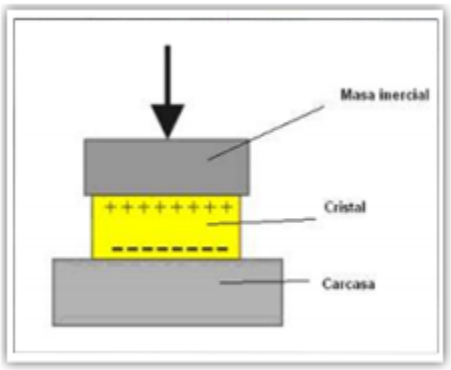


Figura 2.7: Diagrama de un acelerómetro piezoeléctrico.

- **Acelerómetros piezorresistivos:** Este tipo de acelerómetro es esencialmente una galga extensiométrica de alta precisión, donde una deformación física del material cambia el valor de un puente de Wheatsone. Si un conductor es comprimido, su resistencia se altera debido a cambios dimensionales y de sus propiedades piezorresistivas. Esto indica que la resistividad del conductor depende de la presión mecánica a la que es sometido.

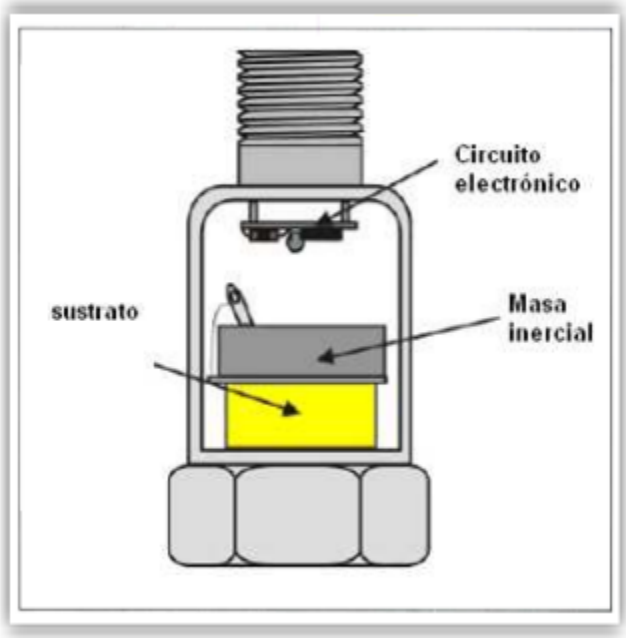


Figura 2.8: Diagrama de un acelerómetro piezorresistivo.

- **Acelerómetros MEMS (del inglés *microelectromechanical systems*):** La micromecanización es la tecnología que permite integrar sobre un mismo sustrato de silicio tanto la parte electrónica como los sensores, actuadores y también diversos elementos mecánicos.

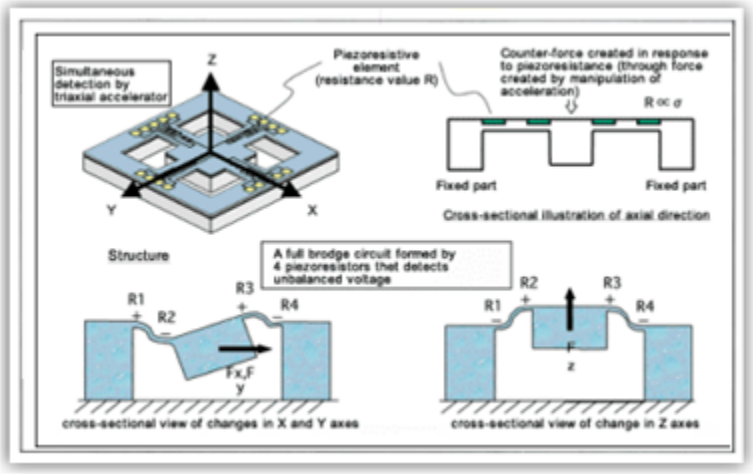


Figura 2.9: Esquemático funcionamiento acelerómetro MEMS Hitachi.

2.1.2.2. Giroscopios.

Los giroscopios miden la velocidad angular de rotación de un objeto. Existen diversos tipos de giroscopios, cada uno de los cuales está regido por principios físicos diferentes.

- **Giroscopios mecánicos:** Están constituidos por un volante o masa, distribuida en la periferia, que gira lo suficientemente rápido alrededor de un eje.

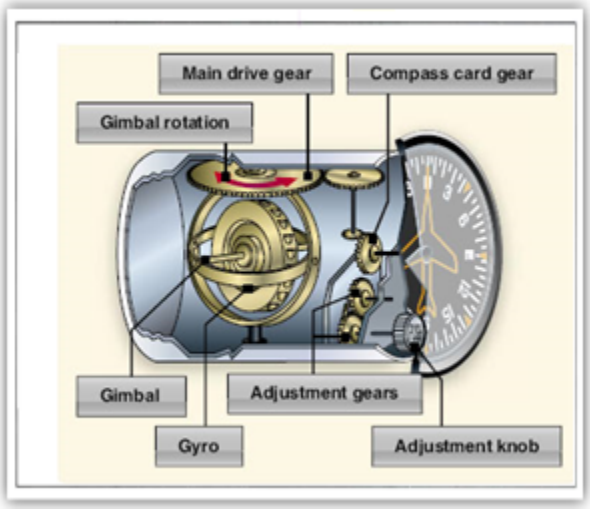


Figura 2.10: Esquema giroscópio mecánico.

- **Giroscopios ópticos:** Se rigen por el efecto Sagnac el cual se basa en la diferencia de camino recorrido por dos haces luminosos dentro de una fibra óptica. A partir de la medida de diferencia de fase, los sensores ópticos obtienen la velocidad de rotación.

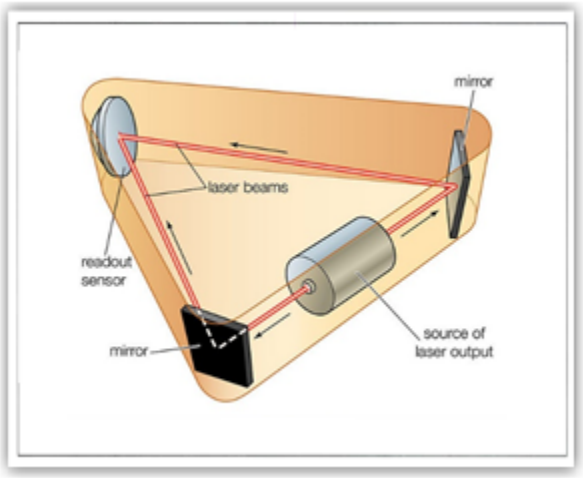


Figura 2.11: Esquema giroscópio óptico.

- **Giroscopios electrónicos:** Este tipo de giroscopios aplican el efecto de la aceleración de Coriolis, que depende de la velocidad de giro. La fuerza de Coriolis es una fuerza ficticia o aparente que sirve para explicar el movimiento anómalo que describe un objeto que se mueve dentro de un sistema de referencia en rotación. Si la velocidad angular de rotación se incrementa, la aceleración de Coriolis aumenta, esto hace que se produzca un mayor

desplazamiento de la masa. Este tipo de sensores son los que más se están usando en la industria debido a su pequeño consumo de potencia y su pequeño tamaño.

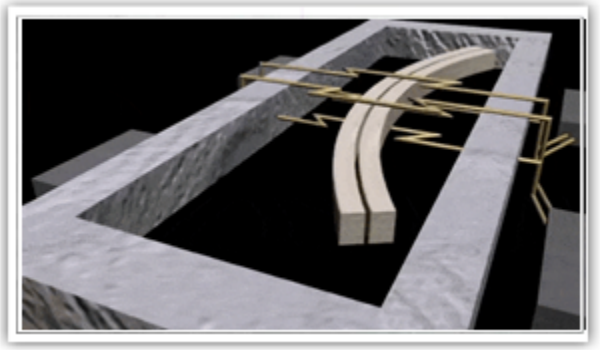


Figura 2.12: Esquema giroscópio electrónico.

Gracias a la micromecanización se dispone de microsensores o sensores miniaturizados, frecuentemente fabricados en tecnologías estándar de integración de circuitos (o bien en versiones ligeramente modificadas de estas tecnologías), que integran acelerómetros y giroscopios, se pueden utilizar éstos sensores en dispositivos compactos por ejemplo, un celular.

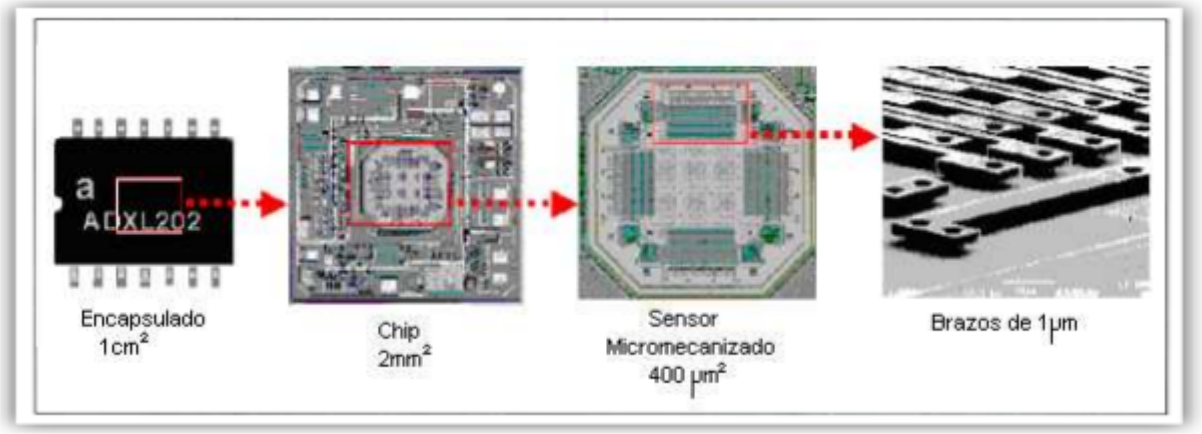


Figura 2.13: Esquemático del interior de un acelerómetro Analog.

2.1.2.3. Ventajas y desventajas de las unidades inerciales de medida.

Las ventajas del uso de sensores inerciales pueden clasificarse en las que repercuten a los sujetos del estudio y las asociadas a los realizadores del estudio. La calibración de los sensores dura unos pocos segundos por lo que los sujetos del estudio no son sometidos a largos tiempos de espera, cuestión especialmente sensible en los sujetos que no puedan permanecer erguidos mucho tiempo. Por otro lado la marcha puede verse afectada por el cansancio de los sujetos tras

tiempos elevados de calibración y, en otros casos, por la incomodidad provocada en los pacientes por algunos sistemas de medida mientras que los sensores inerciales son pequeños, ligeros e inalámbricos. Uno de los beneficios aportados a los realizadores del estudio y al estudio en sí es la sencillez del sistema, mediante una conexión inalámbrica gran cantidad de datos son enviados del sensor a un dispositivo externo como un ordenador ó guardados directamente en el dispositivo para su posterior análisis. Además su peso y tamaño facilitan su movilidad, por lo que es posible llevar el sensor al paciente en lugar del paciente al recinto experimental. Por último, el costo de los sensores inerciales es menor que el costo de gran parte de los métodos mencionados anteriormente. [9]

En cuanto a las desventajas, algunos de los algoritmos de procesamiento son complejos, como las diferentes variantes del filtro de Kalman [1,19,20] para la estimación de la orientación, pero La principal dificultad con los sistemas de captura de movimiento basados en dispositivos inerciales se presenta cuando se requiere integrar señales de inerciales, procedimiento que es afectado por problemas causados por *drift*[1] [38].

2.1.3. Teoría de orientación.

Para identificar la orientación de un objeto en tres dimensiones, es necesario especificar, almacenar y calcular la *orientación* y *subsecuentes rotaciones del objeto* a partir de un marco de referencia. Las cantidades rotacionales son más difíciles de representar que cantidades lineales. Un solo método para expresar esta información no es conveniente para todas las necesidades, por lo tanto, hay diferentes formar de especificar y representar las rotaciones. A continuación se enuncian las más comunes [39,40]:

2.1.3.1. Ángulos de Euler.

Los ángulos de Euler constituyen un conjunto de tres coordenadas angulares que sirven para especificar la orientación de un sistema de referencia de ejes ortogonales, normalmente móvil, respecto a otro sistema de referencia de ejes ortogonales normalmente fijos, como se muestra en la Figura 2.14. Los ángulos de Euler están definidos como tres rotaciones sucesivas sobre los ejes coordenados $\boldsymbol{X}$, $\boldsymbol{Y}$ y $\boldsymbol{Z}$, dadas por los ángulos φ, θ y Ψ respectivamente. Este sistema presenta dos problemas, bloqueo de cardán (*Gimbal Lock*) y una singularidad cuando los ángulos de rotación de los ejes horizontales (φ y θ) se aproximan a 90° debido a la definición del ángulo Ψ [15,40].

[1] Error causado en los sensores inerciales debido a la oposición al cambio repentino de movimiento (inercia). [38]

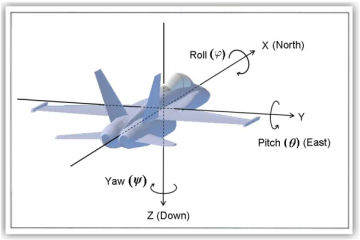


Figura 2.14: Ángulos de euler. [Fuente propia]

2.1.3.2. Cuaterniones.

Los cuaterniones o cuaternios son una representación matemática que sirve como extensión de los números reales, mientras que los números complejos adicionan la unidad imaginaria i, tal que $i^2 = -1$, los cuaternios agregan las unidades imaginarias i, j y k, tal que $i^2 = j^2 = k^2 = ijk = -1$. Dicha extensión ha sido utilizada también para expresar las *rotaciones* dadas por un cuerpo a partir de una orientación anterior fija. Este método no presenta los problemas mencionados en el item anterior. [15, 40]

Como se mencionó, un cuaternión puede representar una rotación tridimensional y expresarse en forma matricial de cuatro componentes:

$$q = (W, V) \text{ Dónde } V = (X, Y, Z) \tag{2.2}$$

Dónde ***W*** es la parte real y ***X***, ***Y*** y ***Z*** son la parte imaginaria.

2.2. Metodología para la selección de herramientas de trabajo.

En esta sección se muestra la ruta metodológica que se utilizó para la realización del proyecto, en lo que respecta a la selección de los tres algoritmos basados en unidades inerciales de medida a implementar, también, la selección de los dispositivos *hardware* y las herramientas *software* que más se adaptaron a las necesidades del proyecto.

Dichos algoritmos fueron tomados de trabajos académicos que fueron encontrados en la revisión sistemática de la literatura con bases de datos como: ScienceDirect, Biblioteca Científica Electrónica en Línea (SciELO), IEEE Xplore, entre otras; Las palabras clave utilizadas fueron: "*walking*", "*swaying*", "*ground reaction force*", "*inertial measurement unit*", "*force plate*", "*treadmill*", "*vibration testing*", "*system identification*", "*human Gait*", "*stride length*", "*IMU devices*", "*RMSE error*", "*machine vision*", "*Kruskal-Wallis*", "*inertial measurement systems*", "*gait cycle*", entre otros.

2.2.1. Revisión Literaria

Como se observó en la Sección 1.1, existen muchos métodos para estimar la longitud de zancada en la marcha humana mediante unidades de medición inercial. Algunos de los trabajos encontrados en la literatura que realizan éste procedimiento (directa o indirectamente) que se consideran son los más representativos y de mayor utilidad para el desarrollo del proyecto, se muestran en la Tabla 2.3.

Tabla 2.3: Trabajos donde estiman la longitud de zancada (directa o indirectamente) en la marcha a partir de unidades inerciales de medida.

Nombre/Artículo	Objetivo de estudio	Características de las pruebas	Variables estimadas	Año	Pais
Estimation of foot trajectory during human walking by a wearable inertial measurement unit mounted to the foot [1]	Desarrollar un sistema para monitorear la marcha humana en un ambiente al aire libre utilizando 1 unidad inercial de medición.	10 participantes saludables 180 ciclos de marcha (10 sujetos $*$ 3 zancadas $*$ 6 repeticiones)	longitud de zancada Altura de paso	2016	Japón
Slip Detection and Prediction in Human Walking Using Only Wearable Inertial Measurement Units (IMUs) [41]	Desarrollar esquema de predicción y detección de deslizamientos en la marcha en tiempo real mediante 5 unidades inerciales de medida portátiles.	1 sujeto masculino jóven saludable. Varias marchas sobre una plataforma de madera.	Trayectoria completa de la pierna. (De manera indirecta longitud de zancada)	2015	Korea
Measurement of foot placement and its variability with inertial sensors [42]	Desarrollar un método para medir la colocación del pie, zancada a zancada en entornos sin restricciones mediante 1 unidad inercial de medición.	9 sujetos saludables jóvenes. Caminata en corredor sin obstáculos.	Trayectoria del pie durante la marcha.	2013	USA
Walking speed estimation using ashank-mounted inertialmeasurement unit [3]	Estudiar la viabilidad de estimación de la velocidad de la marcha usando 1 unidad inercial de medida montada en la pantorrilla.	5 sujetos masculinos y 3 femeninos de entre 22 y 34 años. Varias caminatas sobre una cinta caminadora a diferentes velocidades.	Velocidad de la marcha.	2010	Canadá
Spatio-temporal parameters of gait measured by an ambulatory system using miniature gyroscopes [43]	Desarrollar un sistema ambulatorio para estimación de parámetros espacio-temporales en la marcha humana durante largos periodos de tiempo usando 2 unidades inerciales de medida	9 sujetos jóvenes y 11 adultos mayores Caminatas sobre cinta caminadora.	Parámetros temporales 2.1.1 Parámetros espaciales 2.1.1	2001	Suiza
Diseño e implementación de un sistema para visualizar la marcha humana biomecánica en la afectación de la rodilla ante una ganartrosis [6]	Desarrollar e implementar sistema para visualizar la marcha biomecánica de la rodilla en la zona sagital mediante 2 unidades inerciales de medida para visualizar la afectación de la gonartrosis en la marcha.	Sujetos jóvenes sin afectaciones en la rodilla. Caminatas sobre cinta caminadora.	Posición angular de la rodilla.	2017	Ecuador
Estimation of IMU and MARG orientation using a gradient descent algorithm [2]	Desarrollar algoritmo de orientación para seguimiento de marcha humana mediante 1 unidad inercial de medida para aplicaciones de rehabilitación.	Sujeto adulto sin patologías en la marcha. Caminatas sobre corredor sin obstaculos.	Trayectoria del pie durante la marcha.	2011	Suiza

2.2.2. Selección de trabajos académicos de interés.

En muchos de los artículos analizados en la Sección 2.3, se estimó la longitud de zancada siguiendo la misma escencia , pero hay pequeñas variaciones por ejemplo en la forma cómo se encuentra la matriz de orientación del dispositivo, eliminación del error generado por *drift*, plano de trabajo del algoritmo en el cuerpo humano, entre otros.

Teniendo en cuenta lo anterior, se realizó una matriz de decisión que se ilustra en la Tabla 2.4, donde se hace una selección de tres trabajos para la estimación de la longitud de zancada en marcha humana mediante dispositivos IMU encontrados en la literatura, tomados de la Sección

2.2.1, en la cual se exponen los algoritmos más cercanos a los objetivos del proyecto. Esto con base a unos criterios enfocados a la necesidad de una implementación conjunta entre el *hardware* y el *software* para todos los algoritmos.

Los criterios de evaluación fueron los siguientes:

- **Facilidad de implementación:** menor complejidad con la que puede implementarse el algoritmo, tanto en la parte *hardware* como *software*.
- **Menor cantidad de dispositivos IMU:** Bajo número de dispositivos a colocar en los sujetos de prueba.
- **Adaptabilidad:** Nivel de compatibilidad con diferentes dispositivos IMU y *software* para implementación.

Se evaluó cada criterio en una escala de 1 a 10, donde 1 es la puntuación mas baja y 10 la más alta, luego, se desarrolló la sumatoria y promedio para cada opción.

Tabla 2.4: Matriz de decisión para la selección de trabajos académicos de interés.

Criterios	**Facilidad de implementación**		**Menor cantidad de dispositivos IMU**		**Adaptabilidad**			
Peso	8		9		10			
	Puntos	Sub-Total	Puntos	Sub-Total	Puntos	Sub-Total	**Total**	**Promedio**
Estimation of foot trajectory during human walking by a wearable inertial measurement unit mounted to the foot [1]	10	80	10	90	10	100	**270**	**90,0**
Slip Detection and Prediction in Human Walking Using Only Wearable Inertial Measurement Units (IMUs) [41]	2	16	7	63	5	50	129	43,0
Measurement of foot placement and its variability with inertial sensors [42]	4	32	10	90	7	70	192	64,0
Walking speed estimation using ashank-mounted inertialmeasurement unit [3]	9	72	10	90	10	100	**262**	**87,3**
Spatio-temporal parameters of gait measured by an ambulatory system using miniature gyroscopes [43]	5	40	10	90	8	80	210	70,0
Diseño e implementación de un sistema para visualizar la marcha humana biomecánica en la afectación de la rodilla ante una ganartrosis [6]	8	64	8	72	9	90	226	75,3
Estimation of IMU and MARG orientation using a gradient descent algorithm [2]	9	72	10	90	10	100	**262**	**87,3**

Según los resultados de la matriz de decisión, se seleccionaron los algoritmos propuestos en los trabajos: "*Estimation of foot trajectory during human walking by a wearable inertial measurement unit mounted to the foot*" [1], "*Estimation of IMU and MARG orientation using a gradient descent algorithm*" [2] y "*Walking speed estimation using ashank-mounted inertial measurement unit*" [3].

2.2.3. Selección del *hardware*.

Según los trabajos seleccionados, para el funcionamiento simultaneo de los algoritmos se hace necesario el uso de dos IMUs, uno en el dorso del pie como se menciona en los Artículos [1, 2] y uno a media pantorrilla como se menciona en el Artículo [3], y así obtener las variables de orientación, aceleración y velocidad angular de cada una de éstas ubicaciones en la trayectoria de la pierna durante la marcha. Éstas son necesarias para que los algoritmos puedan estimar la longitud de las zancadas de una persona. Para la comparación de los algoritmos, se deben de usar los datos en bruto de una misma marcha y procesarlos en cada uno de los algoritmos *software* dedicados a la obtención de la estimación de la longitud de cada zancada. De igual forma se hace necesario que los dispositivos se puedan ubicar de manera sencilla, ya que éstos tienen que ir sujetados al cuerpo y deben permitir una marcha natural sin restricciones de movimiento que la afecten [1–3], por el mismo motivo, deben ser inalámbricos, esto conlleva a que tienen que tener una fuente de alimentación y un sistema de almacenamiento de datos propios. (Los datos se analizarán luego de realizar las pruebas)

De acuerdo con esto, la Tabla 2.5 muestra 3 diferentes dispositivos *hardware* encontrados en el mercado colombiano y sus principales características.

Tabla 2.5: Comparación de los dispositivos *hardware*.

	Dispositivos		
Características	Módulo Mpu6050 Gy-52	*Smartphone*	NGIMU
Ilustración			
Sensores requeridos para el proyecto	Si	Si	Si
Conectividad y Comunicación	Transferencia de datos a tarjeta de adquisición. (comunicación Serial-USB)	Wifi, USB, Tarjeta SD, Bluethoot	Wifi, USB, Tarjeta SD, Serie (RS-232)
Software de compatibilidad	GUI y API de código abierto (C #) para Windows • Configurar los ajustes del dispositivo • Trazar datos en tiempo real • C ++ • .Ino	GUI y API de código abierto (C #) para Windows • Registra datos en tiempo real en un archivo (formato de archivo CSV para usar con Excel, MATLAB, etc.) • JavaScript • VisualStudio • C ++ • AndoidStudio • etc.	GUI y API de código abierto (C #) para Windows • Configurar los ajustes del dispositivo • Seguimiento de datos en tiempo real • Registra datos en tiempo real en un archivo(formato de archivo CSV para usar con Excel, MATLAB, etc.)
Uso de otros elementos adicionales	Conexión a una fuente de energía constante de 5 voltios, cables de conexión eléctrica, tarjeta de adquisición de datos (Arduino)	Batería incorporada	Batería incorporada
Tamaño	(2,0 x 1,6 x 0,3) cm	(16,135 x 7,64 x 0,879) cm	(0,56 × 0,39 × 0,18) cm
Peso	40 g	199,8 g	46 g
Precio	$ 10.000,00	$ 1.200.000	$ 1.549.937,13

Para la selección de uno de estos tres dispositivos se realizó una matriz de decisión presentada

en la Tabla 2.6 en la cual se ilustra de manera cuantitativa el *hardware* más adecuado para el propósito del proyecto con base en los criterios de mayor relevancia.

Los criterios de evaluación fueron los siguientes:

- **Facilidad de implementación:** Hace referencia a la sencillez que presenta la interfaz de comunicación con el usuario.
- **Facilidad en la recopilación de datos:** Se refiere al nivel de sencillez que presenta el dispositivo a la hora de extraer los datos tomados.
- **Facilidad de adición al cuerpo:** El dispositivo se adapta a el posicionamiento en el cuerpo.
- **Dispositivo compacto:** El dispositivo no requiere de modulos externos para su funcionamiento.
- **Precio:** Facilidad con la que los desarrolladores del proyecto pueden adquirir el dispositivo.

Se evaluó cada criterio en una escala de 1 a 10, donde 1 es la puntuación mas baja y 10 la más alta. Luego se desarrolló la sumatoria y promedio para cada dispositivo.

Tabla 2.6: Matriz de decisión para los dispositivos hardware

		IMU		**Smartphone**		**NGIMU**	
Criterios	**Peso**	Puntos	Sub-Total	Puntos	Sub-Total	Puntos	Sub-Total
Facilidad de Implementación	7	1	7	10	70	7	49
Facilidad en la Recopilación de Datos	8	3	24	9	72	10	80
Facilidad de adición al cuerpo	6	10	60	7	42	10	60
Dispositivo compacto	10	1	10	10	100	10	100
Precio	10	10	100	10	100	2	20
Total		25	201	46	384	39	309
Promedio		5	40,2	9,2	76,8	7,8	61,8

Los resultados de la matriz de decisión mostraron que el dispositivo *smartphone* fue el de mayor puntuación frente a las otras 2 opciones, con un total de **384** puntos y **76,8** puntos en promedio, haciéndolo el dispositivo seleccionado para la medición y recopilación de datos para el desarrollo del proyecto.

Cabe resaltar que los desarrolladores del proyecto contaban ya con los *smartphone* necesarios para la captura de los datos, por esto se dió un puntaje de 10 en el criterio de precio.

2.2.4. Selección de *software* para adquisición de datos.

Para la implementación de los algoritmos se hizo necesario el uso de un *software* compatible con el formato de los archivos generados por el *software* del dispositivo *hardware* seleccionado (para

este caso *smartphone* Android); ya que se deben extraer los datos para su posterior análisis. Teniendo en cuenta esto, se hizo necesario la selección de un *software* para el dispositivo, ya que éste no posee un programa de fábrica dedicado a la obtención y extracción de datos de sus sensores, y de los cuales se necesita obtener las variables: Aceleración, Orientación y Velocidad angular.

De acuerdo con esto, primero, se presenta la Tabla 2.7 donde figuran tres diferentes alternativas *software* para adquirir los datos de los sensores en los dispositivos.

Tabla 2.7: Comparación de los *software* para el *smartphone*.

	Compatibilidad de la aplicación	**formato del paquete de datos**	**Frecuencia de muestreo**	**Tamaño**	**Tipo de licencia**
MATLAB Mobile	Android, iOS.	datos en raw .csv ó en .mat	10-100 Hz	Depende del dispositivo.	Gratuita/Paga
HyperIMU	Android	datos en raw .csv ó Json	10-100 Hz	3,4 Mb	Gratuita
Physics Toolbox Sensor Suite	Android, iOS.	datos en raw .csv	10-100 Hz	34 Mb	Gratuita

La tabla anterior muestra tres alternativas de *software* para gestionar la toma de datos del dispositivo. Estos fueron los programas que más se acercaron a los requerimientos del proyecto en el apartado *software*, y son preferiblemente accesibles debido al costo que presentan en el mercado colombiano. Se descartaron posibles *software* similares y/o poco documentados, que no resultan eficientes en la toma de datos.

Para la selección de uno de estos tres *software para adquisición de datos*, se realizó una matriz de decisión en la cual se ilustra de manera gráfica y cuantitativa la *aplicación* más adecuada para el propósito del proyecto con base en los criterios de mayor relevancia:

- **Sencillez de interfaz:** Nivel de facilidad con la que se puede poner en marcha la toma de los datos IMU en bruto.
- **Ahorro en recursos de almacenamiento:** Bajo consumo del almacenamiento del dispositivo.
- **Valoración por el público:** Mayor puntuación en la plataforma de descarga.

En la Tabla 2.8 se encuentran los diferentes *software* pre-seleccionados en la parte superior, y los criterios de evaluación en el costado izquierdo de la Tabla. Se evaluó cada criterio en una escala de 1 a 10, donde 1 es la puntuación mas baja y 10 la más alta. Luego se desarrolló la sumatoria y promedio para cada opción.

Tabla 2.8: Matriz de decisión para la selección del *software* para adquisición de datos.

		HyperIMU		**Matlab Mobile**		**Physics Toolbox Sensor Suite**	
Criterios	**Peso**	Puntos	Sub-Total	Puntos	Sub-Total	Puntos	Sub-Total
Sencillez de interfaz	9	9	81	8	72	5	45
Ahorro en Recursos de almacenamiento	6	9	54	8	48	5	30
Valoración por el público	10	8	80	9	90	8	80
Total		26	215	25	210	18	155
Promedio		8,6	71,6	8,3	70,0	6,0	51,6

Los resultados de la matriz de decisión mostraron qué, en promedio, la *aplicación* **HyperIMU** fue la de mayor puntuación frente a las otras dos opciones, con un total de **215** puntos y **71,6** en promedio, haciéndo de ésta, la *aplicación* seleccionada para recolectar datos IMU en el dispositivo para el desarrollo del proyecto.

2.2.5. Selección de *software* para procesamiento de datos.

La Tabla 2.9 muestra cuatro alternativas de *software* para implementar los algoritmos. Estos fueron los programas que más se acercan a los requerimientos del proyecto en el apartado *software*, y son preferiblemente accesibles debido al costo que presentan en el mercado colombiano. Se descartaron posibles *software* similares y/o poco documentados, que no resultan eficientes en la implementación.

Tabla 2.9: Comparación de los *software*.

	Compatibilidad de plataforma	**Lenguaje de programación**	**Análisis Numérico**	**Manejo de Gráficos**	**Tipo de licencia**
Wolfram Mathematica	GNU Linux, Mac OS, Windows	Lenguaje Interpretado	si	si	Paga
GNU Octave	GNU Linux, Mac OS, Ventanas BSD	Lenguaje Interpretado	si	si	Gratuita
Matlab	GNU Linux,, Mac OS, Windows	Lenguaje Interpretado	si	si	Gratuita /Paga
Visual Studio	GNU Linux, Mac OS, Windows	C++, C#, Objetos, .NET, Python, NodeJS, Java script, etc.	si	si	Gratuita /Paga

Para la selección de uno de estos cuatro *software* se realizó una matriz de decisión en la cual se ilustra de manera gráfica y cuantitativa el *software* más adecuado para el propósito del proyecto con base en los criterios de evaluación de mayor relevancia en el, los cuales son:

- **Conocimiento previo del software:** Experiencia con el *software* por parte de los investigadores.

- **Ahorro en recursos computacionales:** No requiere de un computador potente para su funcionamiento.
- **Flexibilidad con el *hardware*:** Se refiere al nivel de facilidad que presenta el *software* frente a la comunicación con el *hardware*.
- **Soporte o licencias:** Indica el tipo de licencias y soporte contra el costo de cada una.
- **Simulaciones:** Hace referencia al nivel que presenta el *software* para la realización de simulaciones en diferentes entornos.
- **Accesibilidad:** Indica la facilidad con la que los investigadores del proyecto pueden adquirir la herramienta *software*.

En la Tabla 2.10 podemos encontrar los diferentes software pre-seleccionados en la parte superior, y los criterios de evaluación en el costado izquierdo de la Tabla. Se evaluó cada criterio en una escala de 1 a 10, donde 1 es la puntuación mas baja y 10 la más alta. Luego se desarrolló la sumatoria y promedio para cada opción.

Tabla 2.10: Matriz de decisión para la selección del *software* para implementación de los algoritmos.

		Wolfram Mathematica		**GNU Octave**		**Matlab**		**Visual Studio**	
Criterios	**Peso**	Puntos	Sub-Total	Puntos	Sub-Total	Puntos	Sub-Total	Puntos	Sub-Total
Conocimiento previo del software	9	2	18	5	45	10	90	9	81
Ahorro en Recursos Computacionales	6	6	36	8	48	5	30	3	18
Flexibilidad con el hardware	10	4	40	5	50	8	80	10	100
Soporte o licencias	8	1	8	7	56	10	80	8	64
Simulaciones	4	7	28	6	24	10	40	10	40
Accesibilidad	10	1	10	10	100	10	100	9	90
Total		21	140	41	323	53	420	49	393
Promedio		3,5	23,3	6,8	53,8	8,8	70,0	8,2	65,5

Los resultados de la matriz de decisión mostraron qué, en promedio, el *software* **Matlab** fue el de mayor puntuación frente a las otras tres opciones, con un total de **420** puntos y **70** en promedio, haciéndolo el *software* seleccionado para el análisis de datos recolectados por el dispositivo *hardware* e implementación de los algoritmos en el desarrollo del proyecto.

2.3. Implementación de los algoritmos.

En la literatura analizada en la Sección 2.2.1, se pudo observar que en general, los algoritmos expuestos que estiman la longitud de zancada en la marcha humana utilizando dispositivos IMU, siguen la misma esencia, que se describe en los siguientes pasos:

1. **Recepción de los datos:** Dependiendo del dispositivo IMU utilizado los algoritmos pueden trabajar *online* u *offline*. De cualquier forma, el *raw* de datos obtenidos debe pre-procesarce antes de ser utilizado en los cálculos matemáticos respectivos.

2. **Detección de las fases de movimiento de la pierna:** En la marcha, cada pierna de manera repetitiva, pasa por dos fases, fase de descanso y fase de balanceo, es importante que el algortimo pueda identificarlas, para ello puede usar los datos del giroscópio ó los datos de la aceleración, teniendo en cuenta que en la fase de balanceo, ambas variables se ven afectadas de manera significativa, muy marcadas con respecto a el comportamiento cuando están en fase de descanso.

3. **Estimación de la orientación del dispositivo IMU:** Para realizar cualquier estimación a partir de las aceleraciones a las cuales fue expuesto el dispositivo durante la marcha, es de suma importancia conocer la orientación del dispositivo en cada instante de tiempo. Es aquí donde se diferencian la mayoría de los algoritmos, ya que para estimar esta orientación utilizan diversos métodos cómo por ejemplo fusión de sensores, modelos cinemáticos de la pierna ó aprovechando una función de los *smartphone*, utilizar la orientación arrojada directamente por el dispositivo, entre otros.

4. **Rotación de la matriz de aceleración:** Cómo se mencionó en el item anterior, conocer la orientación del dispositivo en cada instante de tiempo durante la marcha es de suma importancia, debido a que para poder usar la matriz de aceleración obtenida por el dispositivo para estimar velocidades y/o desplazamientos del dispositivo (trayectorias), debe rotarse dicha matriz, para que a apartir de un marco de referencia se pueda evidenciar la aceleración en cada uno de los ejes. De no rotarse la matriz de aceleración, no se tendría un marco de referencia común, es decir, en cada instante de tiempo se tendría un sistema coordenado distinto (no se tendría en cuenta la trayectoria del dispositivo).

    - **Eliminación de la aceleración de la gravedad:** Luego de tener rotada la matriz de aceleración, sobre el eje que en el marco de referencia está ubicado de forma perpendicular al suelo, se verá afectado por la aceleración de la gravedad, debido a esto, para que no afecte en cálculos posteriores, se debe eliminar.

5. **Cálculo de la matriz de posición:** En este punto se aborda el problema de estimar la matriz de posición del dispositivo a partir de la matriz de aceleración, para ésto, se debe realizar una doble integración de la aceleración, pero aquí hay que tener en cuenta luego de realizar la primer integración que se debe compensar el error en la velocidad causado por *drift*, Luego de tener la matriz de velocidad compensada, se integra por última vez para obtener la matriz de posición, que es el desplazamiento que realizó el dispositivo en cada uno de los ejes a partir de un marco de referencia ya establecido.

6. **Estimación de longitud de zancada:** Para realizar esta estimación se debe tener en cuenta el algoritmo a usar, más precisamente el plano coordenado de trabajo del algoritmo, de allí, sólo se estima la distancia euclidiana entre dos puntos de el plano, relacionado a los puntos de inicio y fin de cada zancada.

En la Figura 2.15 se muestra el digrama de flujo que describe el funcionamiento general de los algoritmos analizados y muestra la ubicación de cada uno de los pasos mencionados.

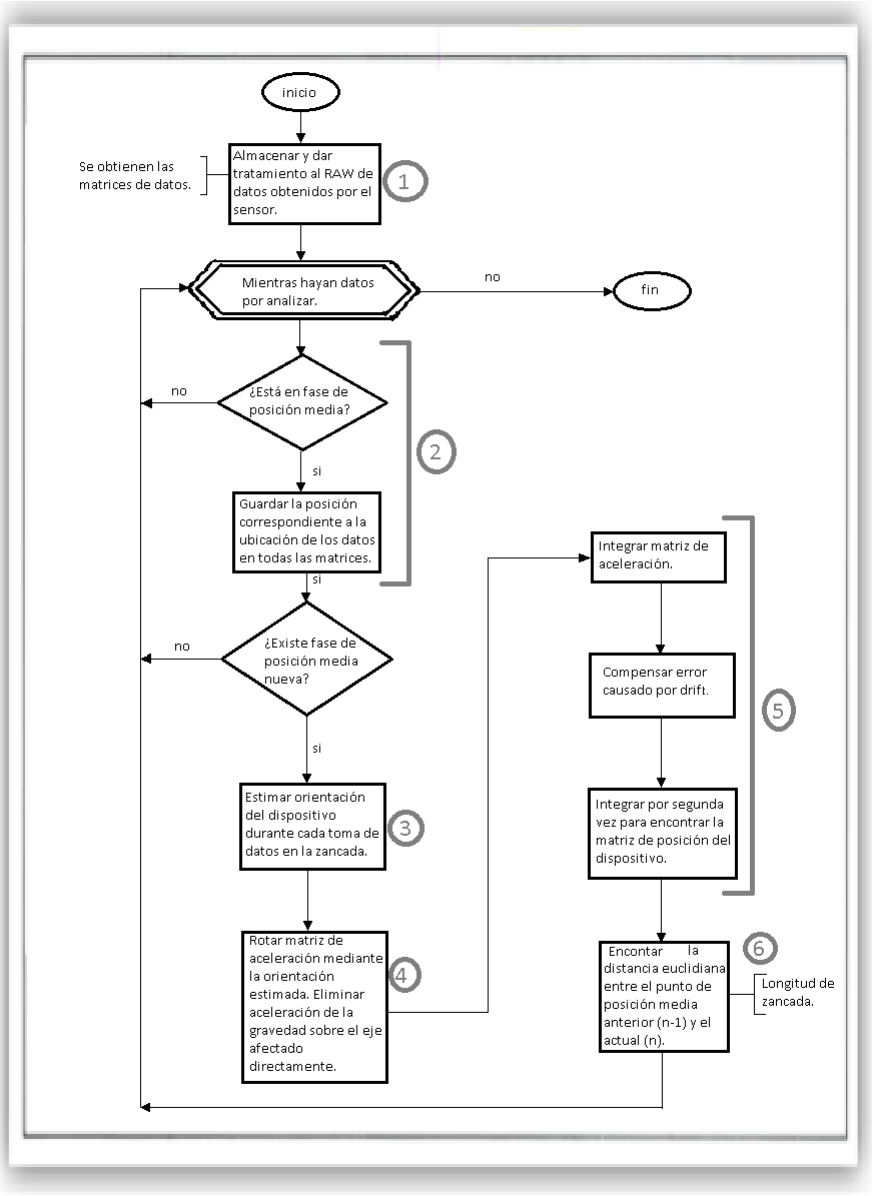


Figura 2.15: Implementación de los algoritmos.

En la Tabla 2.11 se muestra la implementación de manera resumida de los algoritmos IMU con respecto a cada uno de los pasos mencionados, se detallará con más precisión en el Anexo E

Tabla 2.11: Implementación de los de los algoritmos IMU.

Paso	Algoritmo IMU 1 tomado de [1]	Algoritmo IMU 2 tomado de [2]	Algoritmo IMU 3 tomado de [3]
1	Se extrajeron los datos del raw obtenido por HyperIMU y se almacenaron en el espacio de trabajo de Matlab como matrices correspondientes a las variables inerciales de interés.		
2	Se detectaron los inicios y finalizaciones de las fases de descanso de cada ciclo de marcha a partir de la normalización de la velocidad angular (La longitud de zancada está dada por la distancia entre un inicio de fase de descanso actual y el fin de una fase de descanso anterior).	Se detectaron los inicios y finalizaciones de las fases de descanso de cada ciclo de marcha a partir de la normalización de la velocidad angular (La longitud de zancada está dada por la distancia entre un inicio de fase de descanso actual y el fin de una fase de descanso anterior).	Se detectó en la fase descanso, la subfase de posición media a partir de la velocidad angular. En esta subfase, la velocidad angular del eje **Z** tiende a cero, de esta forma se detecta cada posición media, la pierna al encontrarse perpendicular al suelo, la gravedad incide directamente sobre el eje **Y**.
3	El *smartphone* entregó directamente la orientación en ángulos de Euler del dispositivo, a partir del marco inercial de la orientación inicial en la marcha, luego, se convirtió a cuaterniones para evitar cualquier singularidad dada al utilizar ángulos de Euler.	Se implementó el algoritmo de gradiente descendente para estimar la orientación del dispositivo a partir de la aceleración y velocidad angular en el espacio tridimensional, éste algoritmo entregó la orientación del dispositivo en forma de cuaternion.	Se integró la velocidad angular y se estimó la orientación del dispositivo a partir de una matriz de rotación básica bidimensional (la longitud de zancada encontró a partir de la distancia entre dos subfases de posición media).
4	Se rotó la matriz de aceleración de la trayectoria completa de la marcha, se tomó como marco de referencia la orientación inicial del dispositivo. Se rotó mediante el cuaternion encontrado en el paso anterior, luego, se tomó por cada ciclo de marcha la aceleración en fase de descanso para hallar el marco de referencia inercial, es por ello que se estimó la orientación del dispositivo en ésta fase (donde uno de los ejes está ortogonal al suelo) y se rotó la matriz de aceleración en fase de movimiento a partir de ese marco de referencia establecido, luego se eliminó la incidencia de la aceleración de la gravedad sobre el eje en el cual tiene incidencia directa, el cuál se observó en la aceleración de fase de descanso.	Se rotó la matriz de aceleración de la trayectoria completa de la marcha tomando como marco de referencia la orientación inicial del dispositivo, las rotaciones a partir de este marco de referencias fueron descritas por el cuaternion encontrado en el paso anterior (se rotó la matriz de aceleración mediante el cuaternion encontrado).	Se rotó la matriz de aceleración a partir de las matrices de rotación encontradas en el paso anterior y se eliminó la incidencia de la aceleración de la gravedad sobre el eje **Y**.
5	Se compensó error por *drift* en la matriz de acelereación hallada, luego se realizó una doble integración a la matriz de aceleración compensada y se encuentró la posición final del pie (inicio de fase de descanso actual) con respecto al marco de referecia establecido (final de fase de descanso anterior).	Se integró la matriz de aceleración y se compensó error por *drift* en la matriz de velocidad hallada, luego se integró nuevamente a la matriz de velocidad compensada y se encontró la matriz que describió la trayectoria completa del pie durante la marcha.	Se aplicó doble integración a la matriz de aceleración teniendo en cuenta las condiciones iniciales de aceleración y luego velocidad que se mencionan en el artículo, y se encuentró la posición del punto que representa la subfase de posición media actual con respecto a la anterior (marco de referencia origen).
6	Se calculó la distancia que existe entre el marco de referencia (origen) y el punto encontrado en el paso anterior teniendo en cuenta que ésta distancia estaba dada sobre el plano transversal del cuerpo o en otras palabras, plano de progresión de la marcha, para este algoritmo, ejes **X** e **Y**, sólo se estimó la magnitud del vector dado por la posición del punto mencionado en el plano **XY**.	Se calculó la distancia que existe entre el marco de referencia (origen) y cada punto que indica una finalización de zancada, teniendo en cuenta que ésta distancia está dada sobre el plano transversal del cuerpo o en otras palabras, plano de progresión de la marcha, para este algoritmo, ejes **X** e **Y**, se estimó la magnitud del vector dado por la posición de cada punto mencionado en el plano **XY**.	Se calculó la distancia que existe entre el marco de referencia origen y cada punto que indica una finalización de zancada, para lo cual, se estimó la magnitud del vector dado por la posición del punto mencionado en el paso anterior, en el plano **XY**.

Capítulo 3

Experimentación

En esta Sección se describe el desarrollo de todo el proceso relacionado con la obtención de datos en la experimentación del proyecto, y la forma en que trabaja el algoritmo de visión artificial desarrollado por los autores del presente estudio. En la Sección 3.1 se hace la comparación estadística con los algoritmos basados en *Sistemas inerciales de medida*. La Sección 3.2 describe todo lo relacionado con la preparación y condiciones previas al desarrollo del experimento, así como también la explicación detallada de cada procedimiento realizado. Finalmente, la sección 3.3 describe de manera general, el procedimiento que se llevó a cabo, y los protocolos necesarios para la obtención de los datos de la marcha de cada participante en la prueba.

3.1. Sistema óptico de medida.

Para la implementación de este algoritmo se analizaron muchas marchas y se observó que cuando la pantorrilla está completamente perpendicular al suelo coincide con la subfase de posición media como se observa en la Figura 2.4, e indica el inicio de una zancada y cuando la pantorrilla de la misma pierna se encuentra nuevamente en ésta posicion, indica que la zancada finalizó. Teniendo en cuenta ésto, se colocaron en la pantorrilla dos marcadores circulares negros separados 20 cm entre centros y unidos por un segmento de cartón rígido de forma rectangular como se observa en la Figura 3.1 de manera que cuando la pierna se encuentre en subfase de posición media, la línea recta que une los marcadores esté perpendicular al suelo, como se observa en la Figura 3.4, es decir, indique cuando inicia y/o finaliza una zancada.

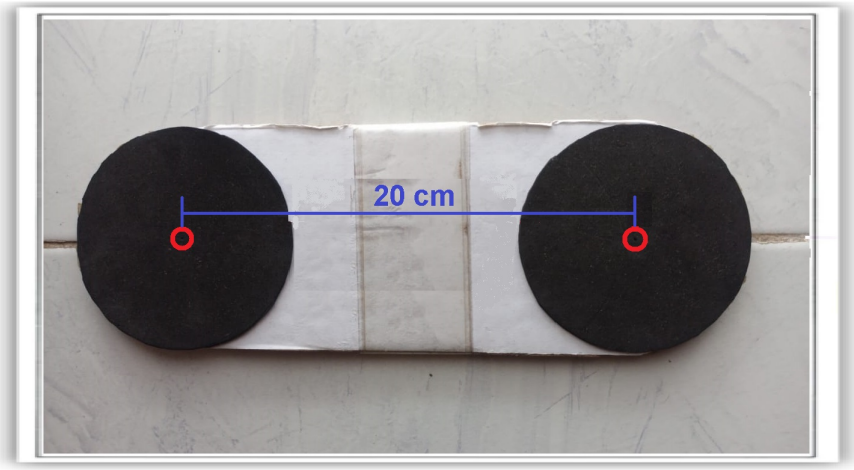


Figura 3.1: Marcador para el sistema óptico de medida.

Los marcadores tienen dos objetivos: el primero es que nos permiten hacer la detección de inicio y/o final de zancada y el segundo es que nos sirven como distancia patrón.

1. **Detección de inicio y/o final de zancada:** Al momento de procesar las imágenes en Matlab, al ser de dos dimensiones se relacionan a un plano coordenado dónde cada pixel es un punto en el plano, los pixeles de la imágen se ordenan de arriba hacia abajo y de izquierda a derecha como se observa en la Figura 3.2 siendo el eje horizontal el eje **X** y el vertical eje **Y**. Con ésto, se puede detectar fácilmente cuando los marcadores están más perpendiculares al suelo, éste momento será cuando la diferencia entre las coordenadas de los centros de los marcadores sea más cercana a cero.

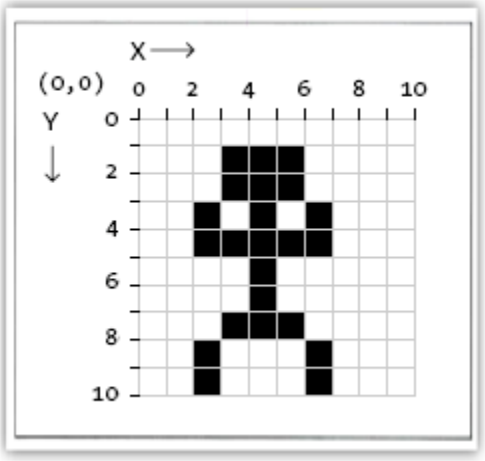


Figura 3.2: Coordenadas en una imágen.

2. **Distancia patrón:** Como se mencionó, la distancia que existe entre los centros de los marcadores es fija de 20 cm, no puede variar. Estimando la distancia euclidianda entre los puntos de los centros de los marcadores se puede relacionar directamente ésta distancia en pixeles en distancia en centímetros, relación que luego servirá para estimar la longitud de zancada.

Teniendo claro la forma en la que se va a detectar y medir cada una de las zancadas de la marcha con el sistema óptico de medida, se dispuso de un sólo dispositivo celular para grabar. Luego de obtener las grabaciones de la marcha se procedió al desarrollo del algoritmo en Matlab.

En la Figura 3.3 se muestra el diagrama de flujo que representa el algoritmo del sistema óptico de medida.

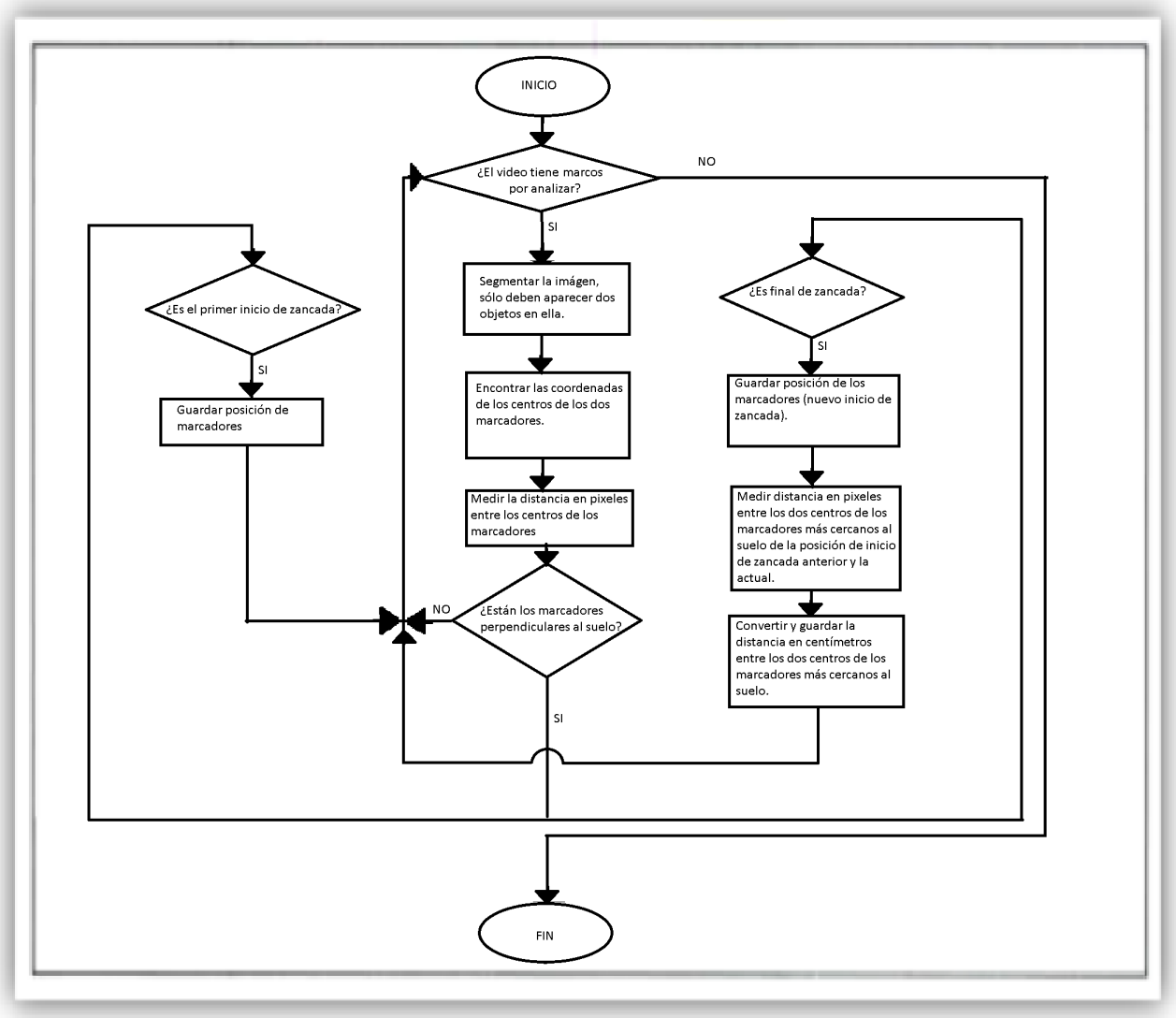


Figura 3.3: Diagrama de flujo que representa al algoritmo de visión implementado.

En la Figura 3.4 se muestra el algoritmo puesto en marcha, se muestra unos marcos antes de la detección de la finalización de la segunda zancada, se puede obsevar cómo recuerda las posiciones de inicio y/o final de zancada anteriores y así mide la longitud de zancada con respecto a la anterior posición de inicio y/o final de zancada.

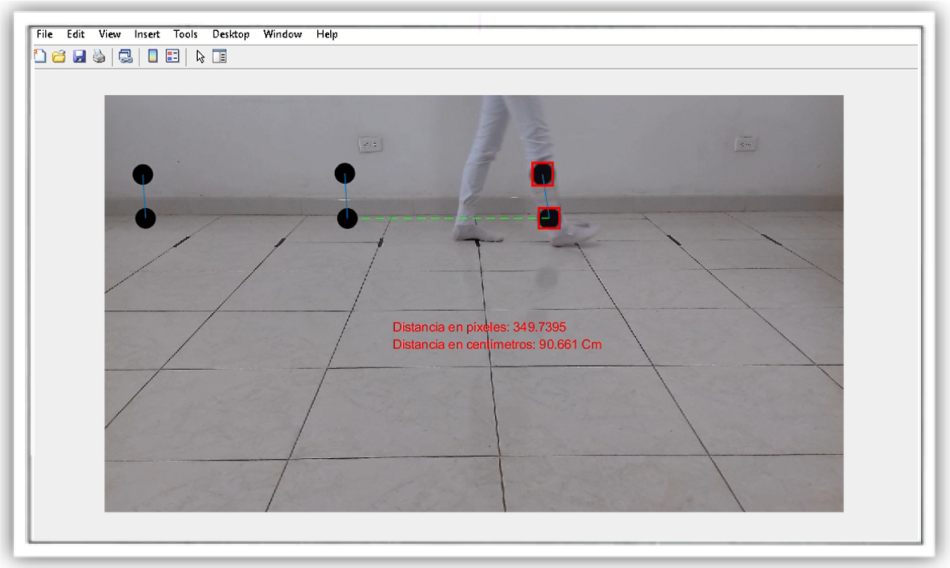


Figura 3.4: Sistema óptico de medida.

3.2. Protocolo Experimental

Inicialmente se procedió a determinar las condiciones necesarias para el buen funcionamiento de los 4 métodos en simultáneo; posteriormente se realizó una comparación estadística de los datos obtenidos en cada marcha. Lo cual significa qué, cada zancada, de cada participante, fue respectivamente comparada en su medición, frente a cada método entre sí. Y esto nos proporcionó la exactitud de cada método frente a la medida real de la zancada.

Debido a la complejidad de obtención de la longitud de zancada por medio del método de visión artificial, y su sensibilidad al ruido frente a la toma de los datos, se decidió adaptar las condiciones de la parte experimental con base en la fidelidad de la toma de datos de este método en particular. Puesto que los demás métodos utilizados en el experimento realizan la obtención de esta información mediante el uso de IMUs integradas al *smartphone*, y estas no presentan mayor sensibilidad ambiental al ruido o datos erróneos en la obtención de los mismos.

3.2.1. Acondicionamiento del espacio

Se asumieron entonces las siguientes condiciones necesarias para una buena toma de datos:

1. Color del fondo de la pared y del suelo, blanco, para que contraste con las circunferencias negras colocadas en el marcado de la pierna.
2. Buena luminosidad en el área de grabación que permita la eliminación de sombras que puedan generar ruido en la detección de las circunferencias.
3. En el área de grabación solo pueden estar, el sujeto de estudio, el marcador colocado en la pierna visible de la cámara, y las marcas de distancia en el suelo; de lo contrario algunos objetos oscuros que también se encuentren en el área podrían generar interferencia con cálculo que obtiene el algoritmo para la medición de longitud de la zancada, mediante visión artificial.
4. Garantizar que el lente de la cámara se encuentra a una distancia de 280 cm con respecto al participante que realiza la prueba, y a una altura de 70 cm del suelo. Esto para asegurar que el lente capte la totalidad de la marcha a una misma distancia de grabación en todas las pruebas.
5. Los dispositivos de obtención mediante IMU para los 3 métodos restantes, deben de estar sujetos a la persona por la misma pierna donde se encuentra el marcador, pero estos no pueden ser visibles frente a la cámara.
6. Se deben colocar los dispositivos de obtención de datos mediante IMUs en las respectivas posiciones que se plantean en cada algoritmo, para realizar un cálculo correcto de cada método empleado en la literatura.
7. Cada participante, durante la prueba, debe caminar de manera natural desde la primera marca colocada en el suelo, hasta la última (que es hasta donde permite grabar la cámara); comenzando con el pie contrario donde se encuentra visible el marcador ante la cámara, y terminar la marcha con los dos pies juntos en la última marca del suelo sin moverse hasta que la persona encargada detenga la grabación de los *smartphone*.
8. Repetir cada marcha en un mismo sentido, luego de finalizar la anterior, para que el método de visión artificial funcione correctamente con el marcador apuntando a la lente de la cámara en una misma posición.

3.2.2. Condiciones adicionales del experimento

- Se ubicaron marcas en el suelo con cinta negra, a una misma distancia entre cada marca, para corroborar la exactitud de las mediciones de los diferentes algoritmos implementados, procurando una marcha similar (en distancia recorrida) por cada zancada, en general, de todos los sujetos participantes en el experimento. Y así comparar los datos reales (90 cm) observables, con cada uno de los métodos utilizados. Ya que estas marcas garantizan una comparación de rendimiento en cuanto a exactitud del sistema óptico, y por ende, una medida más fiable frente a los demás métodos IMU con los cuales se comparó el sistema óptico.

Estas marcas se realizaron promediando la longitud de paso de los participantes para garantizar una marcha natural y no forzada en cada uno de ellos al momento de pisar las marcas; lo cual resultó en un promedio de 45 cm entre cada marca del suelo.

- Se debe garantizar que cada paso del sujeto llegue a la siguiente marca tocando la cinta del suelo con el talón del pie.
- La distancia de grabación se adecuó con respecto al espacio disponible en el área, de modo que la lente no grabara otros objetos del lugar. Esto nos dió como resultado 280 cm de distancia entre el participante y la cámara, y 70 cm de altura desde suelo hasta el soporte de grabación. Lo cual permitió una grabación máxima de 3 zancadas por cada marcha (limitante física del espacio en la cantidad de zancadas por marcha).

A continuación se muestra una lista con el material necesario seleccionado y utilizado para el desarrollo del experimento, con base a el acondicionamiento necesario para el buen funcionamiento y toma de datos de los algoritmos.

Los materiales necesarios para la ejecución del experimento se resumen en la siguiente lista:

- Marcador elaborado a base de cartón y papel *foamy*, con dos circunferencias de color negro y la base pintada de blanco. Circunferencias de 10 cm de diámetro, 20 cm de distancia de centro a centro de cada circunferencia, y 30 cm de extremo a extremo del marcador como longitud total.
- Uniforme de grabación:
  - 1 par de calcetínes blancos.
  - 1 pantalón blanco.
  - 1 camisa blanca.
- 2 *smartphone* Android con la aplicación HyperIMU
- 1 *smartphone* Android con cámara, a una resolución de 420 píxeles
- Cinta negra (para las marcas guía en el suelo)
- Cinta blanca (para sujetar los teléfonos inteligentes a cada participante)

Teniendo definidas estas condiciones necesarias previas a la toma de datos, se procedió a realizar el protocolo para la obtención de datos en el área de experimentación, mediante los siguientes pasos:

3.2.3. Protocolo del experimento

1. Proporcionar a los participantes la ropa de color blanco necesario para el desarrollo del experimento (camisa, pantalón y calcetínes)
2. Sujetar el marcador a la pierna visible del participante
3. Sujetar los *smartphones* a la pierna donde se encuentra el marcador en el participante, por el lado contrario al marcador
4. Ubicar al participante con ambos pies en la primera marca del suelo, y poner en funcionamiento los teléfonos con la aplicación HyperIMU a 20 ms de muestreo
5. Poner en funcionamiento la cámara de grabación, en una resolución de 420 pixeles
6. Iniciar la marcha del participante con el pie contrario del lado visible del marcador
7. Finalizar la marcha y detener en orden consecutivo: la grabación del video, la aplicación HyperIMU de los *smartphones* y repetir los pasos 4,5 y 6 de este protocolo, hasta completar la cantidad de marchas requeridas por persona para la prueba

Figura 3.5: Marcas guía del suelo a 45 cm de distancia y espacio de grabación de las pruebas

3.3. Fase de Desarrollo

En esta fase tuvo lugar el desarrollo del experimento con los sujetos de prueba, teniendo previamente ajustadas las condiciones y protocolos experimentales establecidos en la sección anterior. Se procedió a medir la marcha de cada participante siguiendo los protocolos establecidos con 3 personas sanas, sin antecedentes con patologías en la marcha. Cabe aclarar que se realizaron las pruebas con este número de personas según lo estipulado en el artículo *Estimation of foot*

trajectory during human walking by a wearable inertial measurement unit mounted to the foot [1]. Además, el número de participantes no afecta los cálculos estadísticos, en cuanto a cantidad de datos, puesto que las estadísticas se calcularon a partir del total de zancadas medidas por cada algoritmo, es decir, cada algoritmo calculó un total de 30 marchas por participante para un total de 90 datos estadísticos por cada algoritmo.

A continuación se presenta la secuencia de pasos que se realizó con cada participante para obtener las mediciones de longitud de zancada que calcularon los algoritmos implementados, en esta fase experimental.

- Se realizó la firma del consentimiento de cada persona para el uso y tratamiento de sus datos personales de marcha, recolectados en el experimento con fines exclusivamente académicos.
- Se tomaron las medidas antropo-métricas de cada participante.
- Se procedió a realizar los pasos descritos en la sub-sección 3.2.3 del protocolo experimental, con cada participante individualmente para obtener los resultados de las mediciones.
- Finalmente fueron regresados a los investigadores, los artículos prestados a cada participante de prueba para el desarrollo del experimento, y se concluyó la toma de datos con cada participante.

Ejemplo de vestimenta y ubicación de los dispositivos IMU para un participante:

(a) *Vestimenta* (b) *Ubicación de las IMU*

Figura 3.6: Vestimenta y ubicación de IMU's

En la Figura 3.6 se muestra un ejemplo de la vestimenta proporcionada por los investigadores a cada participante para realizar las pruebas con los requerimientos de cada algoritmo. En la Figura 3.6b se muestran las ubicaciones para los dispositivos IMU según la literatura de cada

algoritmo. La ubicación de la pantorrilla correspondiente al algoritmo n.º 3; y la ubicación en el pie correspondiente a los algoritmos IMU 1 e IMU 2.

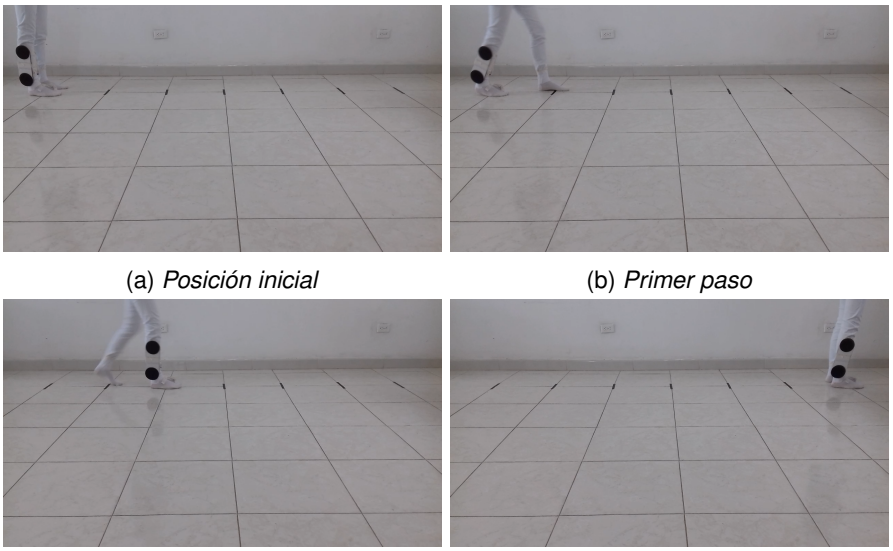
Participante n.º 1:

El primer participante fue una persona de género masculino con 56 años de edad, un promedio de zancada de 94 cm y 47 cm de longitud de paso. Presentó una marcha natural y sin antecedentes patológicos, pero con una velocidad de movimiento algo acelerada en su marcha. En la Tabla 3.1 se muestran algunas de las medidas antropo-métricas del participante, y otras características especiales.

Tabla 3.1: Ficha de caracterización P1

Característica	**Valor**	**Unidad de medida**
Edad	56	Años
Género	M	-
Altura	164	cm
Peso	56	kg
Long. Rodilla	49	cm
Long. Pierna	82	cm
Long. Paso	47	cm
Long. Zancada	94	cm

A continuación se presentan algunas imágenes de la marcha n.º 6 (seleccionada arbitrariamente, y con el único fin de ilustrar su marcha) del participante n.º 1, en la Figura 3.7.



(a) *Posición inicial* (b) *Primer paso*

(c) *Primera zancada* (d) *Posición final*

Figura 3.7: Secuencia de marcha 6 de la persona 1

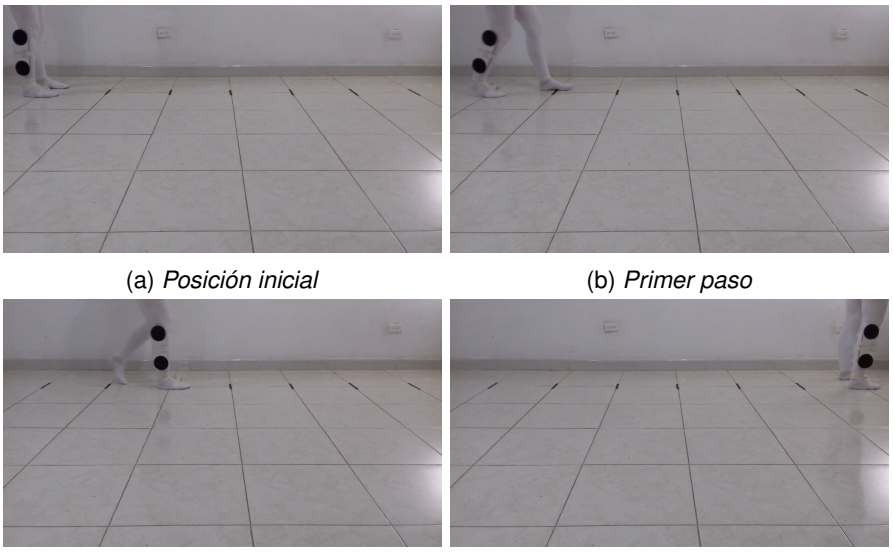
Participante n.º 2:

El segundo participante fue una persona de género femenino con 58 años de edad, un promedio de zancada de 76 cm, y 38 cm de longitud de paso. Presentó una marcha natural y sin antecedentes patológicos. En la Tabla 3.2 se muestran algunas de las medidas antropo-métricas del participante, y otras características especiales.

Tabla 3.2: Ficha de caracterización P2

Característica	**Valor**	**Unidad de medida**
Edad	58	Años
Género	F	-
Altura	155	cm
Peso	60	kg
Long. Rodilla	47	cm
Long. Pierna	85.5	cm
Long. Paso	38	cm
Long. Zancada	76	cm

A continuación se presentan algunas imágenes de la marcha n.º 8 (seleccionada arbitrariamente, y con el único fin de ilustrar su marcha) del participante n.º 2, en la Figura 3.8.



(a) *Posición inicial* (b) *Primer paso*

(c) *Primera zancada* (d) *Posición final*

Figura 3.8: Primer Secuencia de marcha 8 de la persona 2

Participante n.º 3:

El tercer participante fue una persona de género masculino con 23 años de edad, con un promedio de longitud de zancada de 100 cm, y 50 cm de longitud de paso. Presentó una marcha natural

y sin antecedentes patológicos. En la Tabla 3.3 se muestran algunas de las medidas antropométricas del participante, y otras características especiales.

Tabla 3.3: Ficha de caracterización P3

Característica	**Valor**	**Unidad de medida**
Edad	23	Años
Género	M	-
Altura	166	cm
Peso	70	kg
Long. Rodilla	45	cm
Long. Pierna	92	cm
Long. Paso	50	cm
Long. Zancada	100	cm

A continuación se presentan algunas imágenes de la marcha n.º 9 (seleccionada arbitrariamente, y con el único fin de ilustrar su marcha) del participante n.º 3, en la Figura 3.9.

(a) *Posición inicial* (b) *Primer paso*

(c) *Primera zancada* (d) *Posición final*

Figura 3.9: Primer Secuencia de marcha 9 de la persona 3

Con el participante n.º 3 se dió fin al proceso de toma de datos en la fase de experimentación. En general se obtuvo un buen desempeño por parte de los participantes en la prueba para la toma de datos.

Capítulo 4

Resultados, estadísticas y comparación de desempeño

En este capítulo se presenta la comparación estadística de los resultados obtenidos en el Capítulo 3, en el cual se obtuvieron los datos de marcha con 3 personas. De cada persona se obtuvieron datos de longitud de zancada para 10 marchas, y cada marcha de 3 zancadas. Estos datos fueron procesados por los 4 algoritmos implementados en el proyecto, y posteriormente comparados de forma estadística. Por lo cual, en este capítulo se encuentra la Sección 4.1 dedicada a la forma en como se trataron los datos. Luego se encuentra la Sección 4.2 dedicada a la estadística inferencial de los resultados de los algoritmos y la comparación de exactitud de los algoritmos IMU frente al sistema óptico de medida.

4.1. Procesamiento de los resultados

En esta sección se expone la forma como se trataron los datos recolectados por cada dispositivo. En total se hizo uso de 3 *smartphones*; 2 para las ubicaciones del pie y la pantorrilla (obtención de datos algoritmos IMU's), y 1 para la grabación del video (obtención de datos algoritmo de visión artificial del sistema óptico de medida).

De esta manera se obtuvieron dos tipos de información, archivos en formato **.CSV:** el cual es el formato que la aplicación HyperIMU almacena la información de los sensores, y archivos en formato **.MPEG-4 Parte 14:** que es la forma en como se almacena la información para el análisis de visión artificial.

1. **Archivos en formato .CSV:** Del inglés *comma-separated values* es un tipo de documento

para representar datos en forma de tabla, en el cual se separan las columnas por comas y las filas por saltos de línea. Es un tipo de documento sencillo y en formato abierto del cual hace uso la aplicación para android HyperIMU y es donde se obtuvieron los resultados de los dos *smartphones* ubicados en la pantorrilla y pie, puesto que hacen uso de la aplicación para almacenar los datos de aceleración, orientación, campo magnético y giróscopo; los cuales se extraen por medio de dicha aplicación, directamente de los sensores físicos del *hardware* de cada *smartphone*.

2. **Archivos en formato .MPEG-4 Parte 14:** Especificados por ISO/IEC y el grupo MPEG Por sus siglas en inglés (*Moving Picture Experts Group*), más conocido como MP4, gracias al reproductor audiovisual MP4, es un tipo de archivo que contiene información comprimida de audio y video con información especial para la sincronización, de acuerdo a un formato fijado en sus especificaciones técnicas para la reproducción. Este formato fue utilizado para la grabación de video con el *smartphone* restante, y que contiene la información necesaria para el procesamiento de los datos a través del algoritmo del sistema óptico de medida.

A continuación se muestra una imagen de los datos obtenidos en formato .CSV, ya que para el formato .MP4 la información no se encuentra a simple vista, sinó de manera implícita en la sucesión de imágenes del video, y a través de su debido procesamiento junto con otra serie de análisis que realiza el algoritmo del sistema óptico en estos archivos para calcular la información de la marcha.

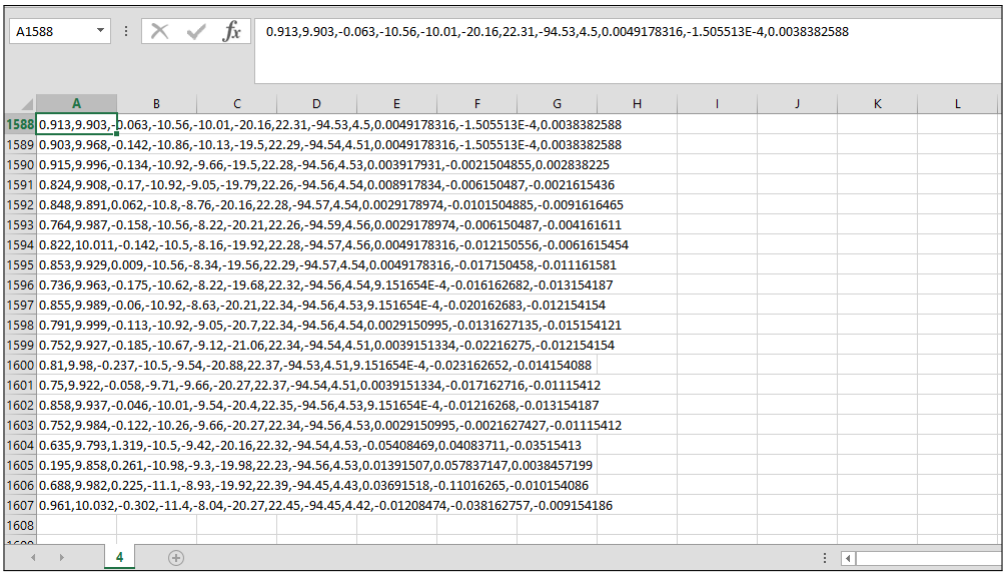


Figura 4.1: Datos en bruto de la marcha 4 en la ubicación de la pantorrilla

En la Figura 4.1 se observa un archivo .CSV obtenido en la aplicación HyperIMU durante una marcha; y se ilustra también un ejemplo de la cantidad de datos que puede tomar la aplicación en una marcha de aproximadamente 3.5 segundos de duración (tiempo determinado según la duración del video correspondiente a esa marcha).

Se tomaron datos de los tres ejes (x,y,z) para cada variable extraída de los sensores, dando como resultado un total de 12 datos por cada marcha como se ilustra en la Figura 4.2, y donde también se pueden apreciar las separaciones por comas de los datos por cada variable y las etiquetas de los sensores en la primera fila del archivo.

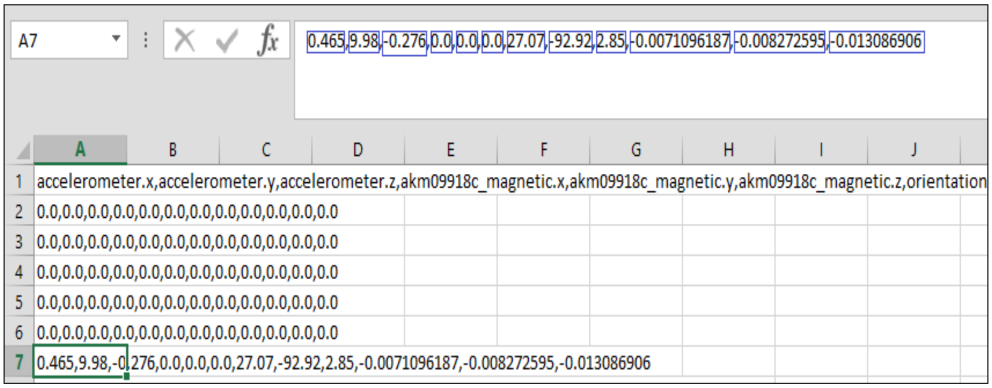


Figura 4.2: 12 datos separados por comas

Posteriormente se procesaron los datos en bruto por cada algoritmo. De este procesamiento se obtuvo un solo archivo **LongZancadas.mat** el cual es una estructura de datos que contiene la información de las 10 marchas de cada persona, y 3 zancadas por cada marcha, dadas en metros. Esta estructura se compone de 4 matrices, *LongZ*, *LongZ1*, *LongZ2* y *LongZ3*. Matrices las cuales tiene un tamaño de *10x3x3* que corresponden a, marchas, zancadas y participantes; respectivamente. Y se puede apreciar en la Figura 4.3.

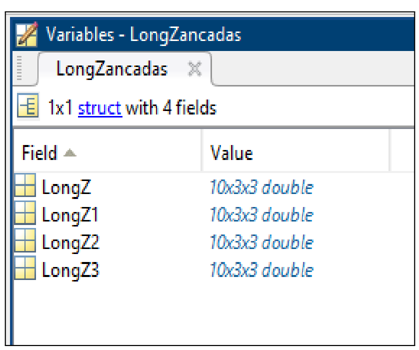


Figura 4.3: Estructura de datos LongZancadas.mat

A continuación se presenta de manera gráfica un ejemplo de cada matriz contenida en el archivo procesado de datos **LongZancadas.mat**, y cómo fue el resultado obtenido para la persona n.º 1 (tomado a modo de ejemplo y de manera arbitraria) en cada una de estas matrices.

- **Matriz de datos LongZ:** Es la matriz correspondiente a los datos del algoritmo basado en visión artificial, y contiene la información de 10 marchas; cada marcha de 3 zancadas por

cada participante. Para el caso de la Figura 4.4, se muestran los datos procesados de las 3 mediciones de longitud de zancada del participante n.º 1 (columnas) en sus 10 marchas (filas), calculadas por el algoritmo del sistema óptico de medida.

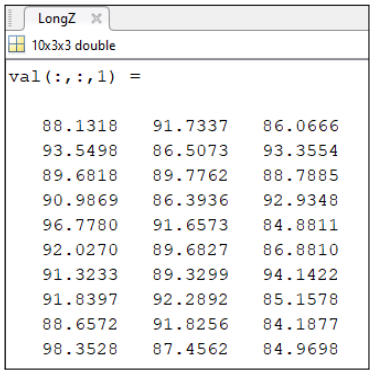


Figura 4.4: Matriz de datos del algoritmo del sistema óptico de medida para el participante n.º 1

- **Matriz de datos LongZ1:** Es la matriz correspondiente a los datos del algoritmo n.º 1, y contiene la información de 10 marchas; cada marcha de 3 zancadas por cada participante. Para el caso de la Figura 4.5, se muestran los datos procesados de las 3 mediciones de longitud de zancada (columnas) del participante n.º 1 en sus 10 marchas (filas), calculadas por el algoritmo IMU 1.

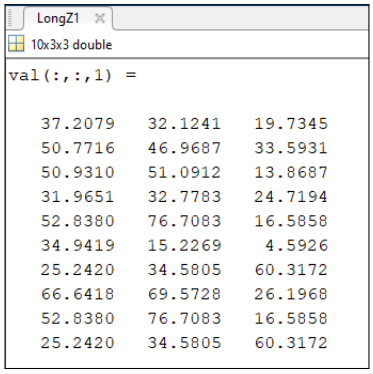


Figura 4.5: Matriz de datos del algoritmo n.º 1 para el participante n.º 1

- **Matriz de datos LongZ2:** Es la matriz correspondiente a los datos del algoritmo n.º 2, y contiene la información de 10 marchas; cada marcha de 3 zancadas por cada participante. Para el caso de la Figura 4.6, se muestran los datos procesados de las 3 mediciones de longitud de zancada (columnas) del participante n.º 1 en sus 10 marchas (filas), calculadas por el algoritmo IMU 2.

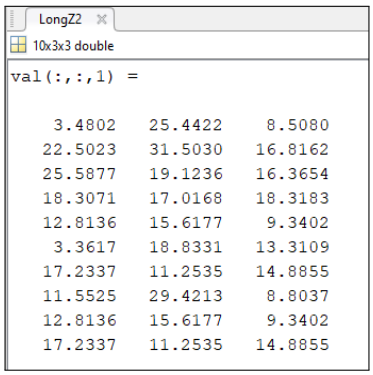


Figura 4.6: Matriz de datos del algoritmo n.º 2 para el participante n.º 1

- **Matriz de datos LongZ3:** Es la matriz correspondiente a los datos del algoritmo n.º 3, y contiene la información de 10 marchas; cada marcha de 3 zancadas por cada participante. Para el caso de la Figura 4.7, se muestran los datos procesados de las 3 mediciones de longitud de zancada (columnas) del participante n.º 1 en sus 10 marchas (filas), calculadas por el algoritmo IMU 3.

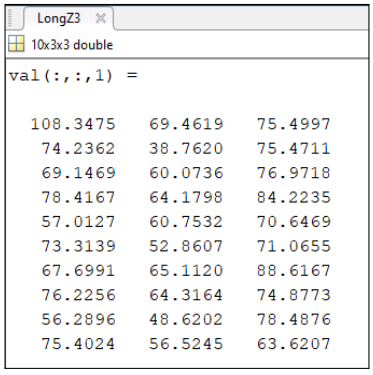


Figura 4.7: Matriz de datos del algoritmo n.º 3 para el participante n.º 1

4.2. Comparación estadística entre algoritmos

En esta sección se muestran las comparaciones estadísticas generales de los 4 algoritmos para los 3 participantes, con el propósito de dar cumplimiento a los objetivos del proyecto y concluir con el rendimiento general de cada algoritmo implementado.

4.2.1. Estadísticas generales de rendimiento

Para dar cumplimiento con el objetivo específico 3 del proyecto, se determinó, en primera instancia, que el algoritmo del sistema óptico de medida tuviese un mejor desempeño de rendimiento en cuanto a la exactitud en la medición de la longitud de zancada, frente a los algoritmos IMU implementados en el proyecto. Esto se determinó mediante el error RMSE relativo por sus siglas en inglés (*Root Mean Square Error*); el cual mide la cantidad de error que hay entre dos conjuntos de datos, normalmente entre un valor pre-dicho y un valor observado por el instrumento de medición. Lo cual lo hace una regla de puntuación cuadrática que también mide la magnitud media del error, y por tanto, se hace susceptible a valores atípicos.

Está dado por la ecuación:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n} \left(Y_i - \hat{Y}_i\right)^2}{n}} \tag{4.1}$$

donde n es numero de datos, Y_i es el valor predicho, y $\hat{Y}_i$ es el valor estimado por el algoritmo. Mediante la ecuación 4.1 obtuvimos el valor del RMSE y RMSE relativo para cada algoritmo implementado, de manera que se realizó una comparación directa de exactitud frente a la medición real de los 90 cm de longitud de zancada marcados en el suelo, para cada algoritmo.

A continuación se presenta la Tabla 4.1 y la Figura 4.8 en las cuales se resumen los datos estadísticos generales de cada algoritmo, a modo de referencia comparativa de su rendimiento y exactitud.

Tabla 4.1: Estadísticas generales de los 4 algoritmos

	Alg. Visión	Alg. IMU 1	Alg. IMU 2	Alg. IMU 3	Unidades
Media Aritmética	89,63	38,22	15,72	97,02	cm
Desviación estándar	6,41	22,12	7,09	35,60	cm
RMSE Relativo	7,09	62,51	82,92	40,11	%

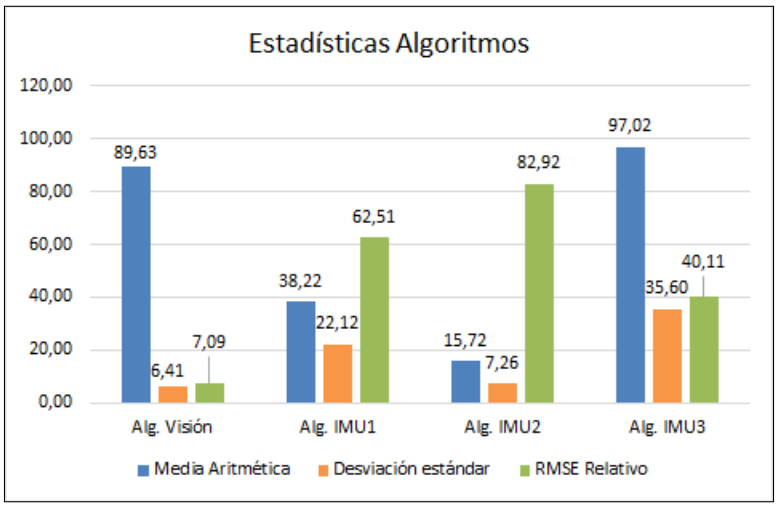


Figura 4.8: Gráfico de estadísticas n.º 1

En la Tabla 4.1 y la Figura 4.8 se muestran los datos: promedio de zancada, desviación estándar y el error RMSE relativo de cada algoritmo; en comparación a la medida real realizada por los participantes en la experimentación del proyecto, correspondiente a 1 zancada de 90 cm (cada 2 marcas guía del suelo de 45 cm por longitud de paso). De estos datos se puede concluir qué:

- El mejor resultado, en cuanto a la exactitud, se obtuvo con el algoritmo del sistema óptico de medida con un porcentaje de error RMSE relativo de 7,09 %. Obtuvo una medición de zancada promedio de 89,63 cm y desviación estándar de 6,41 cm, lo cual lo hace, además, un algoritmo bastante preciso para el proyecto; dando fidelidad de resultados en cada medición que realiza.
- El resultado menos eficiente, en cuanto a la exactitud, se obtuvo con el algoritmo IMU 2, puesto que tiene el porcentaje de error RMSE relativo más elevado de todos (82,92 %) y, en promedio, sus mediciones estuvieron alrededor de los 15,72 cm lo cuál dista mucho de las marca guías en el suelo.
- El segundo mejor resultado, en cuanto a la exactitud, se obtuvo con el algoritmo IMU 3, ya que su error RMSE relativo es de 40,11 % y su promedio de zancada se encuentra en los 97 cm. A diferencia del algoritmo del sistema óptico de medida, este algoritmo tiene una desviación estándar y un porcentaje de error más elevados. Su desviación estándar es de 35,6 cm lo cual puede ocasionar que en ciertas mediciones tengan errores muy elevados, con desviaciones de hasta 35,6 cm. Este tipo de variaciones en el resultado de este algoritmo lo hacen menos confiable y preciso que el algoritmo del sistema óptico de medida.
- El algoritmo IMU 1 se sitúa en el tercer lugar, en cuanto a la exactitud, de los 4 algoritmos implementados durante la fase experimental. De este algoritmo se puede decir que fue poco exacto y preciso en la medición de la longitud de zancada en el proyecto, con error RMSE relativo de 62,51 % y desviación estándar de 22,12 cm. Su promedio de longitud de zancada dista 51,78 cm, por defecto, de la medida real de una zancada (90 cm).

4.2.2. Exactitud de las IMU frente al algoritmo del sistema óptico de medida

Teniendo en cuenta la información de la Sección 4.2.1 se pudo determinar que el algoritmo del sistema óptico de medida es mucho mas exacto y preciso que los demás algoritmos implementados en el proyecto; por lo cual, en esta sección se realizó la comparación del rendimiento, en cuanto a la exactitud, de cada algoritmo IMU frente al rendimiento del algoritmo del sistema óptico de medida. De esta forma se dió cumplimiento con el objetivo específico n.º 3 del proyecto.

A continuación se presenta la Tabla 4.2 y la Figura 4.9 en las cuales se resumen los datos estadísticos generales, y el rendimiento de cada algoritmo frente al rendimiento del algoritmo del sistema óptico de medida. Se estableció entonces los resultados del algoritmo del sistema óptico como punto de comparación de la longitud de zancada para cada dato obtenido por los algoritmos IMU. Se tomó el promedio general del algoritmo del sistema óptico como medida como referencia patrón, y el cálculo punto a punto de las mediciones realizadas por este algoritmo para determinar nuevo valor RMSE relativo de los algoritmos IMU implementados.

Tabla 4.2: Estadísticas de rendimiento frente al sistema óptico

	Alg. Visión	Alg. IMU 1	Alg. IMU 2	Alg. IMU 3	Unidades
Media Aritmética	89,63	38,22	15,72	97,02	cm
Desviación estándar	6,41	22,12	7,26	35,6	cm
RMSE Relativo	7,09	62,9	83,12	40,66	%

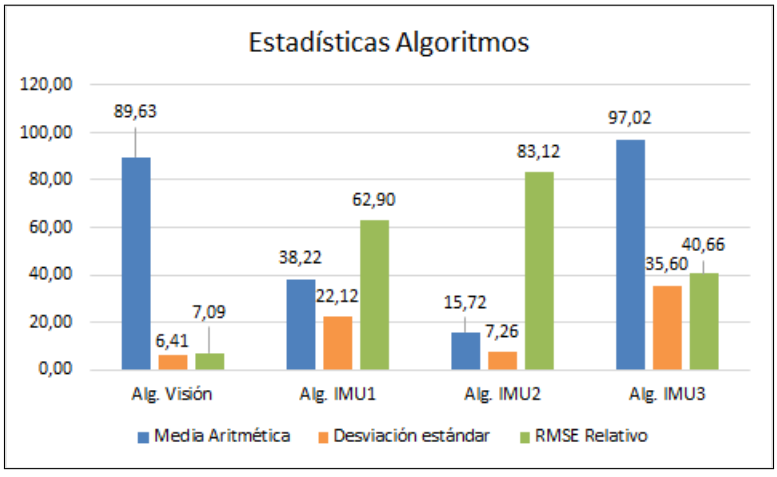


Figura 4.9: Gráfico de estadísticas n.º 2

En la Tabla 4.2 y la Figura 4.9 se muestran los datos: promedio de zancada, desviación estándar y el RMSE relativo para cada algoritmo implementado. Los datos del algoritmo del sistema óptico de medida se obtuvieron a partir de la medida real marcada en el suelo (90 cm); puesto que este fue el algoritmo de menor porcentaje de error frente a la medida real, con un 7,09 % de error y

una media de 89,63 cm. Además, debido a la baja desviación estándar que tiene el algoritmo, se puede afirmar que su precisión es bastante buena, y garantiza una medición fiable en cada zancada que calculó el algoritmo.

Para el cálculo de las estadísticas de error RMSE relativo de los demás algoritmos, se tomó como referencia los resultados del algoritmo del sistema óptico de medida, como ya se mencionó anteriormente. Posteriormente se procesaron los datos con esta nueva referencia y se obtuvieron los resultados que se aprecian en la tabla 4.2. De estos datos se puede concluir qué:

- El mejor resultado de los 3 algoritmos IMU implementados se obtuvo con el algoritmo IMU 3 con un porcentaje de error RMSE relativo de 40,66 %, lo cual lo hace el resultado más exacto frente al algoritmo del sistema óptico.
- El resultado menos eficiente se obtuvo con el algoritmo IMU 2 puesto que su porcentaje de error RMSE relativo es el más elevado de todos (83,12 %).
- El segundo mejor resultado se obtuvo con el algoritmo IMU 1, con un porcentaje de error RMSE relativo de 62,9 %.

La media aritmética y desviación estándar de los algoritmos sigue siendo la misma puesto que esta medida es única e independiente de los resultados obtenidos por cada algoritmo. Cabe aclarar que el análisis de los resultados se profundizará más adelante en la sección de discusión y conclusiones (5).

4.2.3. Estadístico *Kruskal-Wallis*

Para obtener un mejor análisis de los resultados, se procedió a realizar comparaciones estadísticas con las medias aritméticas de las 3 zancadas por marcha, obteniendo así, únicamente 10 datos por persona correspondientes al promedio de zancada por cada marcha. De esta manera, la comparación entre grupos de datos es menos extensa para realizar pruebas de distribución de datos. Esto con el fin de determinar si es posible medir, de manera exacta, la longitud de zancada en marcha humana, indistintamente de cualquiera de los 4 métodos implementados en el proyecto. Mediante la determinación estadística de si los datos provienen o no de una misma distribución.

Para determinar si los datos obtenidos por los algoritmos poseen o no una distribución normal, se realizó la prueba de *Kolmogorov-Smirnov*, la cual plantea la hipótesis nula H_0 de que una muestra de datos proviene de una distribución normal. Esta prueba se realizó en el programa Matlab mediante el comando $h = kstest(x)$, donde x es la muestra de datos y h retorna un valor lógico para determinar si se aprueba o rechaza la hipótesis nula H_0, con un nivel de significancia del 5 %.

Al realizar esta prueba para cada grupo de datos de los algoritmos, se determinó que por lo menos 1 grupo de datos no posee una distribución normal, descartando así el uso del análisis de la varianza (*ANOVA*), ya que para este tipo de comparación estadística, es necesario que todos los grupos de datos se encuentren en una distribución normal.

Por lo cual, se procedió a realizar la comparación de datos entre los 4 algoritmos mediante la prueba estadística de *Kruskal-Wallis*; el cual es un método no paramétrico para probar si varios grupos de datos diferentes provienen de una misma población, cuando el número de grupos de datos es mayor a 2 y sus distribuciones no son normales. Este es un método alternativo al uso de la *ANOVA* cuando el estudio muestral cumple estas condiciones especiales. Se plantea entonces una hipótesis nula y una hipótesis alternativa para definir si los grupos de datos pertenecen o no a una misma distribución no normal, mediante la comparación de las medias de cada grupo de datos. Haciendo uso de rangos que contrastan esta hipótesis, podemos determinar si todos los grupos forman parte de una población más grande ó, por el contrario, pertenecen a poblaciones diferentes con características distintas.

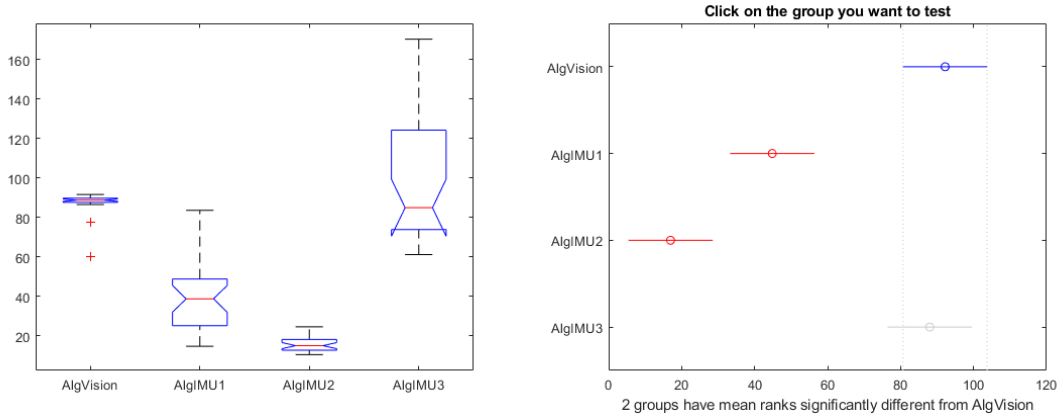
Se planteó como hipótesis nula (H_0): *Las diferencias entre los rangos de medias de los algoritmos, no son estadísticamente significativas y provienen de una misma distribución.*

Se planteó como hipótesis alternativa (H_1): *Las diferencias entre los rangos de medias de los algoritmos, son estadísticamente significativas y no provienen de una misma distribución.*

4.2.4. Comparativo con estadístico Kruskall-Wallis

En esta sección se realizó el comparativo con el estadístico Kruskall-Wallis para los 4 algoritmos, mediante el código **promedioZancadasGen.m** realizado en el programa Matlab, y se encuentra en el Anexo F F.

A continuación se presentan 2 gráficos y 1 tabla de datos obtenidos mediante el estadístico Kruskall-Wallis, al cual se le pasaron los 30 datos (promedio de zancada por cada marcha) de las 3 personas, para obtener la comparación correspondiente entre cada algoritmo.



(a) *Diagrama de cajas y bigotes* (b) *Comparación kruskal-wallis*

Figura 4.10: Comparación estadística para los algoritmos

Source	SS	df	MS	Chi-sq	Prob>Chi-sq
Columnas	117052.1	3	39017.4	96,74	7.80858e-21
Error	26935,4	116	232,2		
Total	143987.5	119			

Tabla 4.3: Tabla ANOVA *Kruskall-Wallis*

La Figura 4.10a se muestra el diagrama de cajas y bigotes de cada uno de los algoritmos implementados en el proyecto. Se puede apreciar que la media del algoritmo del sistema óptico de medida se encuentra cercano al valor de los 90 cm (mayor exactitud), y la mayoría de sus datos se encuentran concentrados al rededor de dicha media. Por tanto, este algoritmo tiene una dispersión de datos bastante pequeña con respecto a su media, lo cual lo hace a su vez, un algoritmo significativamente preciso para el proyecto. Caso contrario sucede con el algoritmo IMU 3, ya que a pesar de que su media se encuentra cercana a los 90 cm de longitud de zancada, la dispersión de sus datos es bastante grande; esto lo hace un algoritmo exacto pero impreciso y genera una sensación de incertidumbre ya que en ocasiones se pueden realizar mediciones bastante alejadas de la realidad. De los algoritmos IMU 1 y 2 se puede decir que sus medias se encuentran distantes del valor real; y el algoritmo IMU 2, a pesar de ser bastante preciso, sus mediciones son inexactas llegando a tener una media de alrededor de los 15 cm.

En la Figura 4.10b se puede apreciar el comparativo de grupos del estadístico Kruskall-Wallis; en el cual se presentan los rangos de las medias de cada grupo de datos de los algoritmos. Si los grupos de datos tienen sus rangos de medias dentro de los mismos valores entre sí, significa que pertenecen a una misma población. Se realizó la comparación con el rango del algoritmo del sistema óptico de medida, y del cual se puede decir qué: 2 grupos de datos tienen los rangos medios significativamente diferentes del algoritmo del sistema óptico de medida (algoritmos IMU 1 y 2), mientras que el algoritmo IMU 3 no tiene diferencias estadísticamente significativas en su rango medio frente al algoritmo del sistema óptico, y se podría decir entonces que estos dos algoritmos sí pertenecen a una misma población.

Así mismo se puede apreciar que en la Tabla 4.3, el resultado de *p* es $7,80858e^{-21}$; lo cual es mucho menor que el nivel de significancia del 5 %, rechazando así la hipótesis nula H_0. Esto indica que la información suministrada para la prueba Kruskal-Wallis por estos grupos de datos, no pertenecen a una misma población; se acepta entonces la hipótesis alternativa H_1 de que por lo menos 1 grupo de datos tiene una distribución distinta al algoritmo del sistema óptico de medida. Esto quiere decir qué, a pesar de que la población de estudio en la prueba fue la misma para todos los algoritmos, los datos de medición calculados por los algoritmos son significativamente diferentes entre cada grupo de datos (cada algoritmo), y por ende, no hay una similitud en los resultados que pueda afirmar que todos los métodos realizaron una medición acertada y confiable de la longitud de zancada en marcha humana, sobre las muestras obtenidas en la experimentación del proyecto.

Capítulo 5

Discusión, conclusiones y trabajos futuros

En este capitulo se se presenta la discusión sobre los resultados obtenidos en las pruebas realizadas, frente a los resultados que se pueden encontrar en la literatura seleccionada para el desarrollo de los algoritmos en este proyecto. Además se presenta también la sección de las conclusiones y los posibles trabajos futuros.

5.1. Discusión

Cabe aclarar que las condiciones del experimento presentadas en este trabajo fueron adaptadas al espacio disponible por los investigadores del proyecto, se intentó aplicar una metodología de experimentación similar a la expuesta en los trabajos de los algoritmos IMU implementados, pero las pruebas aquí realizadas no representan de manera exacta las condiciones expuestas en los mismos. Se optó realizar las pruebas siguiendo la cantidad de zancadas por marcha que se plantea en [1] del cual tomamos el primer algoritmo IMU implementado y en el que se tenían mayores especificaciones de acondicionamiento del lugar y experimentación; además, en sus resultados sí se pudo hallar la estimación de longitud de zancada en valores comparables con los del presente estudio. Teniendo en cuenta lo anterior, se adaptó el espacio de experimentación de tal forma que el sistema óptico de medida no tuviera problemas de iluminación ni interferencia de otros objetos en la imagen.

Inicialmente se discutirá sobre los resultados obtenidos por el sistema óptico de medida, debido a que respecto a éste se evaluó la exactitud de los tres algoritmos IMU implementados en el proyecto. Como se puede apreciar en la Figura 3.4, en el suelo donde se realizó la marcha se colocaron marcas cada 45 cm y, según se define en la Sección 3.2, las personas que realizaron

la marcha debieron procurar terminar la zancada cada dos marcas, es decir, 90 cm, por lo tanto, se tomó ésta como la medida real de cada zancada en la marcha realizada por cada persona. El error RMSE obtenido por el sistema óptico de medida respecto a la medida real, fue de 6,38 cm, lo cual correspondió a 7,09 % de error RMSE relativo. Además el promedio de la longitud de zancada que se obtuvo fue de 89,63 cm que correspondió a un error promedio del algoritmo respecto a la medida real de 0,37 cm que correspondió a un porcentaje de error de 0,41 %. También se obtuvo un error absoluto medio (EAM) de 3,51 cm, que correspondió al 3,9 % de error relativo medio (ERM). Al hacer análisis de trabajos previos donde mediante un sistema óptico de medida estiman: la longitud de la zancada [44] y longitudes antropométricas del cuerpo humano [45], se observó que los resultados pueden ser comparables y, así mismo, el sistema óptico de medida puede ser aceptado como sistema de referencia para comparar respecto a este los resultados obtenidos por los algoritmos IMU. El pequeño error en la exactitud obtenido por el sistema óptico pudo deberse a diversos factores como por ejemplo, distorción radial, efecto de paralaje (errores de perspectiva), error en el cálculo de los centroides de los marcadores, entre otros [44–46].

En dos de los artículos tres artículos seleccionados [2, 3] de dónde se tomaron los algoritmos IMU (2 y 3) para la estimación de la longitud de zancada mediante unidades inerciales de medida, no tenían como objetivo estimar la longitud de zancada de manera directa, puesto que éste parámetro les sirvió para corroborar la estimación de la orientación en el primero [2] y la velocidad de la marcha en el segundo [3], por ésto los resultados de éste parámetro no se presentaron en ninguno de los artículos mencionados, por tal motivo, primero se compararán los resultados obtenidos por el algoritmo IMU 1 tomado de [1] con los expuestos en su literatura y luego se hablará de los resultados obtenidos por los otros dos algoritmos IMU (2 y 3).

En [1] se menciona que el error RMSE obtenido entre la estimación de la longitud de zancada dada por el algoritmo basado en una unidad de medición inercial, frente al sistema de captura de movimiento fue de 4,84 cm, que correspondió a 4,2 % de la media de longitud de zancada que para el trabajo mencionado fue de 115 cm, mientras que en los resultados obtenidos en éste proyecto se encuentra un error RMSE de 56,38 cm que correspondió a un RMSE relativo de 62,9 %, frente a la media de longitud de zancada que para nuestro proyecto fue de 89,63 cm estimado por el sistema óptico de medida. Se puede observar que el resultado obtenido no fue satisfactorio y no correspondió con los resultados mostrados en la literatura.

En cuanto al algoritmo IMU (2) tomado de [2], los resultados obtenidos en la experimentación reflejaron un error RMSE de 74,5 cm que correspondió a un error RMSE relativo de 83.12 %, de la media de longitud de zancada.

Y por último, los resultado obtenidos por el algoritmo IMU (3) tomado de [3] reflejaron un error RMSE de 36,44 cm que correspondió a 40,66 % de error RMSE relativo, de la media de longitud de zancada.

El error RMSE muestra qué, de manera general, los algoritmos IMU no obtuvieron resultados satisfactorios, esto puede deberse a múltiples factores, tales como baja frecuencia de muestreo,

errores en la compensación del *drift*, deslizamientos en la marcha, entre otros [1–3, 41].

5.2. Conclusiones

1. **Implementación de algoritmos:** El análisis de la marcha humana requiere de conocimientos previos sobre todos los parámetros que están implícitos en la complejidad de esta acción. Antes de realizar la implementación de los algoritmos para la estimación de la longitud de zancada propuestos en este proyecto, se tuvo que estudiar muy bien las fases del ciclo de la marcha y los parámetros espaciales en ella, más específicamente la longitud de zancada; se debieron realizar pruebas piloto de marcha para analizar los datos (IMU y captura de video) y encontrar correspondencia de estos con las fases de movimiento en las que se podría encontrar la pierna, esto fue muy importante debido a que en primera instancia en los algoritmos de estimación implementados se debe segmentar la marcha cómo se mencionó en la Sección 2.3 y se deben hallar umbrales en los valores de las variables de aceleración o velocidad angular que determinan el cambio de fase en el ciclo de la marcha.

2. **Experimentación:** Lograr un buen desempeño en las pruebas de experimentación y un resultado comparativo a partir de la teoría de la literatura, no fue una tarea sencilla. Puesto que mucha documentanción acerca de los algoritmos IMU implementados no describe el paso a paso de la manera en que se implementó todo el sistema de medición y, mucho menos, las condiciones exactas de los estudios realizados en ellos. Esto nos llevó a un arduo trabajo dedicado de ensayos previos a la experimentación, en donde se realizaron pruebas de funcionamiento de los algoritmos IMU implementados posteriormente con los participantes. Así como también las pruebas de funcionamiento del sistema óptico de medida, el cual demandó varias pruebas previas de funcionamiento. Ya que este método debe tener unos ajustes de parámetros especiales en sus variables de medición, iluminación y distancia focal del marcador. Cabe resaltar que este método fue desarrollado exclusivamente por los autores del presente documento con base a sus conocimientos sobre visión artificial a lo largo de sus estudios. Debido a lo anterior, podemos concluir que la fase experimental se desarrolló de manera satisfactoria en el presente estudio, puesto que se logró una toma de datos correcta sin mayores contratiempos cumpliendo con todos los requisitos y objetivos que cada algoritmo implementado demandaba, excluyendo el tipo de resultado obtenido en particular.

3. **Análisis de resultados:** En el Capitulo *4* se pudo observar un comportamiento particular de los algoritmos frente a las condiciones previas del experimento y su desarrollo, así como una experiencia de aprendizaje acerca de los dispositivos con IMU integradas para la medición de la longitud de zancada en marcha humana. La determinación de la exactitud de estos dispositivos se hizo mediante un evaluador externo que fuera de gran exactitud frente al valor real de medición. Se pensó de esta forma un evaluador alternativo al uso de la cinta cuando ésta no sea presente en la marcha humana natural, mediante la pregunta: "*¿De qué manera se puede evaluar la exactitud de un sistema IMU frente a otro totalmente diferente?*",

la cual nos llevó a evaluar el desempeño de un sistema óptico de medida con respecto a las marcas de longitud constante en el suelo, para determinar si este método evaluador era lo suficientemente exacto en su medición, para ser un referente de exactitud frente a un sistema inercial de medida. Lo cual nos llevó a las siguientes conclusiones:

- **Algoritmo IMU 1:** Este algoritmo obtuvo un error RMSE relativo de 62,9 % lo cual indica que es un modelo poco eficiente en su exactitud para calcular la longitud de zancada de una persona. Ya que, en poco más de la mitad de sus mediciones, se encontraron valores que no correspondían a una medición acertada de lo que se estaba midiendo en la marcha de los participantes, y este error indica cuán cerca están los datos medidos del valor observado. La raíz del error cuadrático medio (RMSE) es la raíz cuadrada del promedio de errores cuadrados, y el efecto de cada error es proporcional al tamaño del error cuadrado; por lo tanto, los errores mas elevados tienen un efecto desproporcionado en el resultado, lo que lo hace sensible a los valores atípicos. Se puede afirmar entonces que este algoritmo es un modelo que posee una exactitud muy baja para la predicción de la longitud de zancada en marcha humana con respecto al desempeño del algoritmo del sistema óptico de medida desarrollado en el presente estudio.
- **Algoritmo IMU 2:** Este algoritmo obtuvo un RMSE relativo de 83,12 % lo cual indica que es el algoritmo menos eficiente en su exactitud para calcular la longitud de zancada de una persona; ya que su porcentaje de aciertos es de tan solo 16,88 % (según como se realizaron las pruebas en el proyecto). Se puede afirmar entonces que este algoritmo es un modelo que posee la exactitud mas baja de los 3 algoritmos para la predicción de la longitud de zancada en marcha humana, con respecto al desempeño del algoritmo del sistema óptico de medida. Esto puede deberse a múltiples factores, como lo son: la utilización de una baja frecuencia de muestreo, la ubicación de los sensores en este algoritmo, ruidos ambientales que afecten las mediciones, etc.
- **Algoritmo IMU 3:** Este algoritmo obtuvo un RMSE relativo de 40,66 % lo cual indica que es el algoritmo IMU con menor porcentaje de error al calcular la longitud de zancada de una persona; ya que menos de la mitad de las mediciones fueron incorrectas (según como se realizaron las pruebas en el proyecto). Se puede afirmar entonces que este algoritmo es el de mayor exactitud de los 3 algoritmos para la estimación de la longitud de zancada en marcha humana, con respecto al desempeño del algoritmo del sistema óptico de medida. Además, la sensibilidad del error RMSE a valores atípicos, puede ser causante de un ajuste erróneo de un modelo que bien podría desempeñarse mejor en otras condiciones no presentadas en este estudio, o haciendo uso de un comparador de desempeño poco sensible a valores atípicos. Estos valores atípicos pueden incrementar en gran medida el porcentaje de error RMSE, en este caso. Un posible efecto de variación en el rendimiento de estos algoritmos IMU implementados puede deberse a la forma en que se calcula la medición de la longitud de zancada, ya que las variantes de cada algoritmo en particular, como lo son: la ubicación de las IMU en el cuerpo, cálculo de las matrices de rotación y compensación del driff, son elementos fundamentales para

la base da cada algoritmo, y pueden cambiar de uno a otro generando diferencias de exactitud significativas en el resultado cada algoritmo en particular. En este proyecto se procuró realizar una implementación fiel a la literatura encontrada sobre estos algoritmos, y realizar una secuencia de pasos ordenada para obtener mediciones correctas de la longitud de zancada por cada algoritmo, siguiendo un paso a paso general, a pesar de ser algoritmos diferentes desde la ubicación, número de sensores, hasta los cálculos de compensación.

4. **Conclusión General:** A modo general se puede concluir qué, el algoritmo del sistema óptico de medida cumplió con los objetivos comparativos del proyecto, dispuestos mediante la exactitud en su resultado frente a la medición real. De todos los algoritmos IMU implementados, el algoritmo IMU 3 fue el único con el rango de su media dentro del rango de la media del algoritmo del sistema óptico de medida, y con una distribución igual. Los algoritmos IMU 1 y 2 tienen sus rangos de las medias significativamente diferentes del algoritmo del sistema óptico de medida, y poseen una distribución distinta. Por lo cual, se puede decir que los algoritmos IMU implementados en este proyecto no pertenecen a la misma población que el algoritmo del sistema óptico de medida, puesto que por lo menos 1 de estos rangos no pertenece a la misma distribución. Por tanto, la exactitud de dichos algoritmos no cumple satisfactoriamente con los resultados obtenidos por el algoritmo del sistema óptico de medida. Esto puede deberse a múltiples factores que ya se mencionaron anteriormente: baja frecuencia de muestreo, errores en la compensación del *drift*, deslizamientos en la marcha, error de cálculo por parte de los algoritmos, valores atípicos, ruido en la marcha que afecta los sensores de las IMU, entre otros [1–3, 41].

5.3. Trabajos futuros

Al desarrollar este estudio se descubrieron ciertos aspectos que se podrían implementar para mejorar la medición de la longitud de zancada mediante dispositivos IMU, o mejoras al mismo proyecto en cuestión; tales cómo:

1. Realizar mayor número de marchas por cada participante para obtener resultados más significativos estadísticamente.
2. Aumentar la distancia del recorrido de las personas que realizan la prueba para obtener un mayor número de zancadas por persona.
3. Implementar y comparar diferentes algoritmos para la estimación de la orientación de los dispositivos IMU.
4. Utilizar dispositivos IMU dedicados y no el uso de IMUs integradas a los smartphone para una mayor frecuencia de muestreo.

Bibliografía

[1] N. Kitagawa and N. Ogihara, "Estimation of foot trajectory during human walking by a wearable inertial measurement unit mounted to the foot," *Gait & posture*, vol. 45, pp. 110–114, 2016.

[2] S. O. Madgwick, A. J. Harrison, and R. Vaidyanathan, "Estimation of imu and marg orientation using a gradient descent algorithm," in *2011 IEEE international conference on rehabilitation robotics*. IEEE, 2011, pp. 1–7.

[3] "Estimación de la velocidad de marcha usando acelerómetros montados en el vástago," pp. 5096–5101, 2010.

[4] J. C. Hernandez Silva, M. A. Carrero Nuñez *et al.*, *Interfáz gráfica para el análisis de los parámetros cinemáticos y espacio temporales de la marcha*, 2017.

[5] M. Saucedo, "Valoración de la marcha humana," *Universidad Nacional Autónoma de Mèxico, Mexico*, 2009.

[6] T. Pineda and D. Francisco, "Diseño e implementación de un sistema para visualizar la marcha humana biomecánica en la afectación de rodilla ante una gonartrosis," Master's thesis, Quito, 2017., 2017.

[7] S. C. Vázquez, "La marcha: Historia de los procedimientos de análisis." *Biociencias*, vol. 2, p. 13, 2004.

[8] S. Miyazaki and T. Kubota, "Quantification of gait abnormalities on the basis of continuous foot-force measurement: correlation between quantitative indices and visual rating," *Medical and Biological Engineering and Computing*, vol. 22, no. 1, pp. 70–76, 1984.

[9] M. B. Rodero Lasheras, *Procesado de señales proporcionadas por sensores inerciales: Evaluación de la marcha post-ictus*, 2016.

[10] R. Baker, "Gait analysis methods in rehabilitation," *Journal of neuroengineering and rehabilitation*, vol. 3, no. 1, p. 4, 2006.

[11] S. Collado Vázquez, "Análisis de la marcha humana con plataformas dinamométricas: influencia del transporte de carga," Ph.D. dissertation, Universidad Complutense de Madrid, Servicio de Publicaciones, 2004.

[12] A. Villa Moreno, E. Gutiérrez Gutiérrez, and J. C. Pérez Moreno, "Consideraciones para el análisis de la marcha humana. técnicas de videogrametría, electromiografía y dinamometría," *Revista ingeniería biomédica*, vol. 2, no. 3, pp. 16–26, 2008.

[13] C. L. Saltzman and D. A. Nawoczenski, "Complexities of foot architecture as a base of support," *Journal of Orthopaedic & Sports Physical Therapy*, vol. 21, no. 6, pp. 354–360, 1995.

[14] D. Datta, B. Heller, and J. Howitt, "A comparative evaluation of oxygen consumption and gait pattern in amputees using intelligent prostheses and conventionally damped knee swing-phase control," *Clinical rehabilitation*, vol. 19, no. 4, pp. 398–403, 2005.

[15] G. Jerónimo, J. Luis, and A. Sotelo Aguilar, *Sistema opto-inercial para la medición de la cinemática de la marcha del amputado transfemoral*, 2006.

[16] J. M. W. Brownjohn, J. Chen, M. Bocian, V. Racic, and E. Shahabpoor, "Using inertial measurement units to identify medio-lateral ground reaction forces due to walking and swaying," *Journal of Sound and Vibration*, vol. 426, pp. 90–110, 2018.

[17] A. M. Sabatini, C. Martelloni, S. Scapellato, and F. Cavallo, "Assessment of walking features from foot inertial sensing," *IEEE Transactions on biomedical engineering*, vol. 52, no. 3, pp. 486–494, 2005.

[18] C. Liedtke, S. A. Fokkenrood, J. T. Menger, H. van der Kooij, and P. H. Veltink, "Evaluation of instrumented shoes for ambulatory assessment of ground reaction forces," *Gait & posture*, vol. 26, no. 1, pp. 39–47, 2007.

[19] E. Shahabpoor and A. Pavic, "Estimation of vertical walking ground reaction force in real-life environments using single imu sensor," *Journal of biomechanics*, vol. 79, pp. 181–190, 2018.

[20] Q. Li, M. Young, V. Naing, and J. Donelan, "Walking speed estimation using a shank-mounted inertial measurement unit," *Journal of biomechanics*, vol. 43, no. 8, pp. 1640–1643, 2010.

[21] Y. P. Lim, Y.-C. Lin, and M. G. Pandy, "Effects of step length and step frequency on lower-limb muscle function in human gait," *Journal of Biomechanics*, vol. 57, pp. 1–7, 2017.

[22] B. E. Maki, "Gait changes in older adults: predictors of falls or indicators of fear?" *Journal of the American geriatrics society*, vol. 45, no. 3, pp. 313–320, 1997.

[23] A. L. Cerda, "Manejo del trastorno de marcha del adulto mayor," *Revista Médica Clínica Las Condes*, vol. 25, no. 2, pp. 265–275, 2014.

[24] J. Hannink, T. Kautz, C. Pasluosta, J. Barth, S. Schulein, K. G. Gassmann, J. Klucken, and B. Eskofier, "Mobile stride length estimation with deep convolutional neural networks," *IEEE-JOBAHI*, vol. PP, no. 99, pp. 1 – 10, 2017.

[25] M. Danoudis and R. Lansek, "Gait in huntington's disease and the stride length-cadence relationship," *BMC Neurology*, vol. 14, no. 1, p. 161, Oct 2014.

[26] W. Hoogkamer, S. M. Bruijn, S. Sunaert, S. P. Swinnen, F. Van Calenbergh, and J. Duysens, "Toward new sensitive measures to evaluate gait stability in focal cerebellar lesion patients," *Gait & posture*, vol. 41, no. 2, pp. 592–596, 2015.

[27] B. Hu, P. Dixon, J. Jacobs, J. Dennerlein, and J. Schiffman, "Machine learning algorithms based on signals from a single wearable inertial sensor can detect surface-and age-related differences in walking," *Journal of biomechanics*, vol. 71, pp. 37–42, 2018.

[28] J. Hannink, T. Kautz, C. F. Pasluosta, J. Barth, S. Schulein, K.-G. Gassmann, J. Klucken, and B. M. Eskofier, "Mobile stride length estimation with deep convolutional neural networks," *IEEE journal of biomedical and health informatics*, vol. 22, no. 2, pp. 354–362, 2017.

[29] N.-H. Ho, P. Truong, and G.-M. Jeong, "Step-detection and adaptive step-length estimation for pedestrian dead-reckoning at various walking speeds using a smartphone," *Sensors*, vol. 16, no. 9, p. 1423, 2016.

[30] Y. Liu, Y. Chen, L. Shi, Z. Tian, M. Zhou, and L. Li, "Accelerometer based joint step detection and adaptive step length estimation algorithm using handheld devices," *Journal of Communications*, vol. 10, no. 7, pp. 520–525, 2015.

[31] B. Sijobert, M. Benoussaad, J. Denys, R. Pissard-Gibollet, C. Geny, and C. A. Coste, "Implementation and validation of a stride length estimation algorithm, using a single basic inertial sensor on healthy subjects and patients suffering from parkinson's disease," *ElectronicHealthcare*, pp. 704–714, 2015.

[32] V. Renaudin, M. Susi, and G. Lachapelle, "Step length estimation using handheld inertial sensors," *Sensors*, vol. 12, no. 7, pp. 8507–8525, 2012.

[33] A. Peruzzi, U. Della Croce, and A. Cereatti, "Estimation of stride length in level walking using an inertial measurement unit attached to the foot: A validation of the zero velocity assumption during stance," *Journal of Biomechanics*, vol. 44, no. 10, pp. 1991–1994, 2011.

[34] A. I. A. Mendoza, T. J. B. Santamaria, V. G. Urrego, J. P. R. Restrepo, and M. C. Z. García, "Marcha: descripción, métodos, herramientas de evaluación y parámetros de normalidad reportados en la literatura.(gait: description, methods, assessment tools and normality parameters reported in the literature)," *CES movimiento y salud*, vol. 1, no. 1, pp. 29–43, 2013.

[35] F. Martínez, F. Gómez, and E. Romero, "Análisis de vídeo para estimación del movimiento humano: una revisión," *Revista Med*, vol. 17, no. 1, 2009.

[36] J. H. Osorio and M. H. Valencia, "Bases para el entendimiento del proceso de la marcha humana," *Archivos de Medicina (Col)*, vol. 13, no. 1, 2013.

[37] R. Gómez-Ferrer Sapiña, "Estudio biomecánico de la marcha en pacientes con artrosis de cadera." 2005.

[38] J. D. Bernal Iñiguez, “Diseño, construcción e implementación de un sistema de captura de movimiento para análisis ergonómico de riesgo laboral de extremidades superiores,” B.S. thesis, 2014.

[39] G. Ferrer Mínguez, “Integración kalman de sensores inerciales ins con gps en un uav,” 2009.

[40] E. Bachmann, R. McGhee, X. Yun, and M. Zyda, “Inertial and magnetic angle tracking of limb segments for inserting humans into synthetic environments,” *Naval Postgraduate School*, 2000.

[41] M. Trkov, K. Chen, J. Yi, and T. Liu, “Slip detection and prediction in human walking using only wearable inertial measurement units (imus),” in *2015 IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*. IEEE, 2015, pp. 854–859.

[42] J. R. Rebula, L. V. Ojeda, P. G. Adamczyk, and A. D. Kuo, “Measurement of foot placement and its variability with inertial sensors,” *Gait & posture*, vol. 38, no. 4, pp. 974–980, 2013.

[43] K. Aminian, B. Najafi, C. Büla, P.-F. Leyvraz, and P. Robert, “Spatio-temporal parameters of gait measured by an ambulatory system using miniature gyroscopes,” *Journal of biomechanics*, vol. 35, no. 5, pp. 689–699, 2002.

[44] W. Zhu, B. Anderson, S. Zhu, and Y. Wang, “A computer vision-based system for stride length estimation using a mobile phone camera,” in *Proceedings of the 18th International ACM SIGACCESS Conference on Computers and Accessibility*, 2016, pp. 121–130.

[45] H. Marino-Vera, L. E. Mendoza, and O. E. Gualdrón-Guerrero, “Medición automática de variables antropométricas para la evaluación de la respiración usando visión artificial,” *Revista de Investigación, Desarrollo e Innovación*, vol. 8, no. 1, pp. 161–169, 2017.

[46] D. F. Saavedra Lozano *et al.*, “Sistema para el análisis de la marcha humana usando sensores inerciales,” Ph.D. dissertation, Universidad Santiago de Cali, 2019.

[47] J. Diebel, “Representing attitude: Euler angles, unit quaternions, and rotation vectors,” *Matrix*, vol. 58, no. 15-16, pp. 1–35, 2006.

Anexos

Anexo A

Aplicación para la toma de datos HyperIMU.

En éste anexo se documenta el paso a paso que se debe realizar para obtener los datos en bruto mediante la aplicación HyperIMU en el *smartphone*; lo cual incluye desde el cómo se deben calibrar el *smartphone*, hasta la obtención final de los archivos con los datos en bruto.

A.1. Calibración de los sensores.

La marca de los *smartphone* utilizados para la toma de datos de los algoritmos IMU del experimento es *Xiaomi*, la cual posee una aplicación por defecto que se puede utilizar para la calibración de los diferentes sensores que hay en el *smartphone*, dentro de los cuales se encuentran: **acelerómetro, gyróscopo y magnetómetro**; los cuales se deben ajustar previamente a la toma de datos. Para abrir esta aplicación solo es necesario escribir el siguiente código en la pantalla de marcación del teléfono:

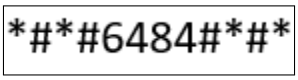


Figura A.1: Código para *smartphone* de la marca *Xiaomi*

Este código abre la siguiente aplicación

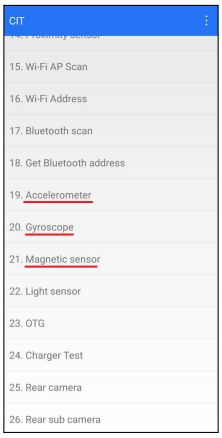


Figura A.2: Menú de calibración de los sensores

En la Figura A.2 se muestra el menú principal para la calibración de los sensores; se marcaron en rojo los sensores calibrados previamente al uso de la aplicación HyperIMU. A continuación se muestra un ejemplo de calibración para el sensor del Gyróscopo

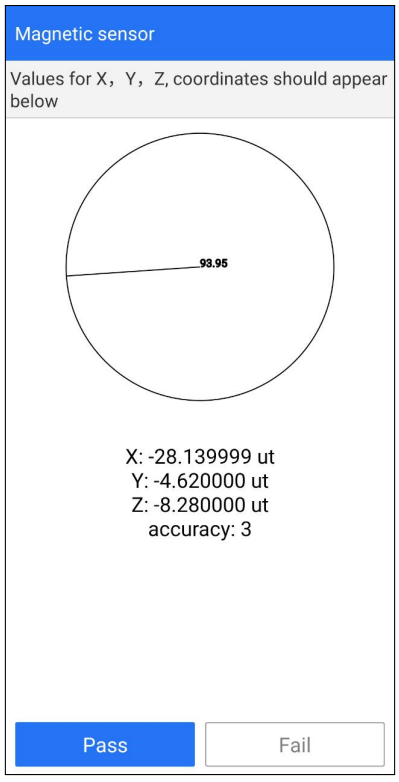


Figura A.3: Calibración Gyróscopo

En la Figura A.3 se muestra un ejemplo de calibración para el sensor del Gyróscopo; el botón

azul que dice **"Pass"** indica que el sensor fue calibrado exitosamente y se puede continuar con el siguiente sensor del *smartphone*.

A.2. HyperIMU

Luego de calibrar todos los sensores, pasamos a descargar y abrir la aplicación de *smartphone* HyperIMU en donde se visualiza la siguiente pantalla principal

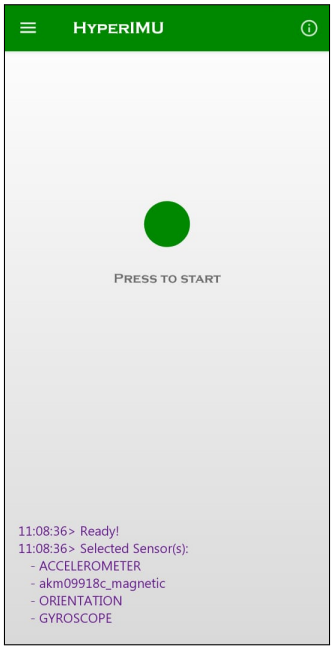
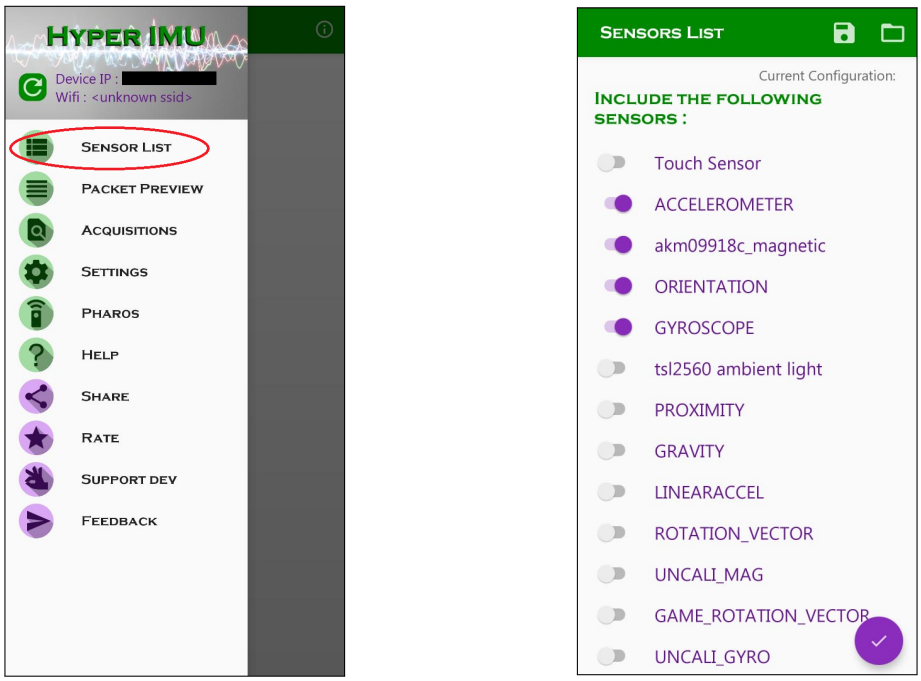


Figura A.4: Pantalla Principal *HyperIMU*

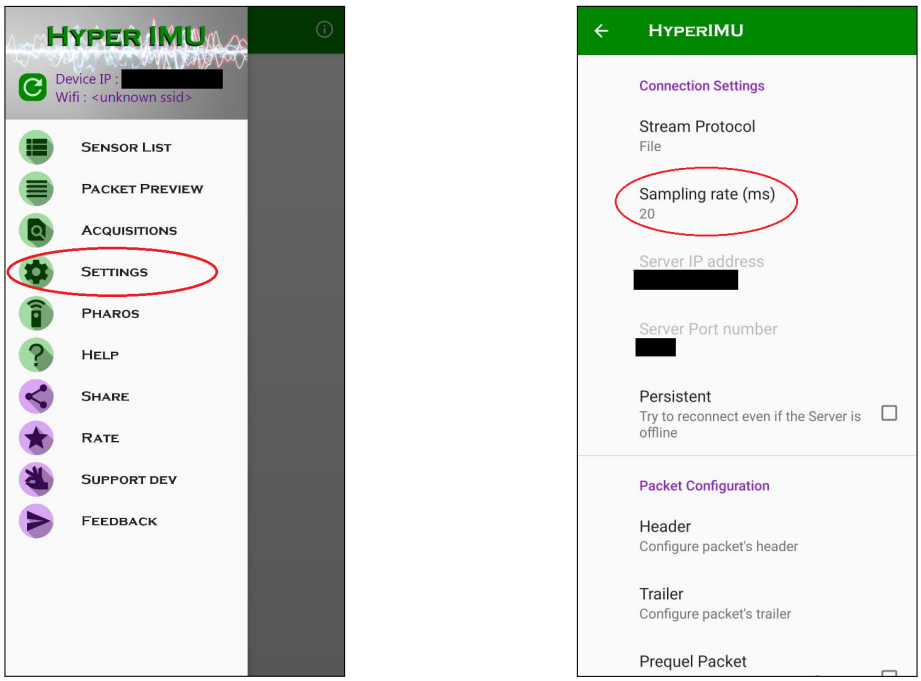
en la Figura A.4 se muestra la pantalla de inicio de la aplicación, donde se puede visualizar el botón central verde para comenzar la grabación de los sensores. Nos dirigimos ahora al apartado *"Sensor List"* correspondiente al listado de los sensores disponibles, en las 3 rayas de la parte superior izquierda de la pantalla principal, para seleccionar los sensores de interés.



(a) *Menú de ajustes HyperIMU* (b) *Selección de los sensores*

Figura A.5: Pasos de configuración

En la Figura A.5 se muestran los pasos para activar los sensores de la aplicación. Una vez activados los sensores se procede a configurar el tiempo de muestreo en 20 milisegundos mediante el menú de ajustes



(a) *Menú de ajustes HyperIMU* (b) *Muestreo*

Figura A.6: Pasos de configuración

En la Figura A.6 se pueden observar los pasos para activar el muestreo a 20 milisegundos en la configuración de la aplicación. El siguiente paso en la aplicación corresponde a la obtención de datos. Una vez esté listo todo el protocolo para la toma de datos con el participante, se procede a presionar el botón verde central *"press to star"* de la pantalla principal (Figura: A.4) para comenzar la adquisición de datos. La aplicación guardará temporalmente los archivos en formato **.CSV** en la dirección donde se encuentra la carpeta de instalación de la aplicación.

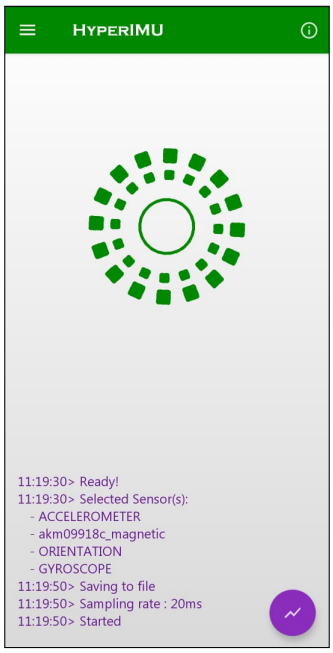
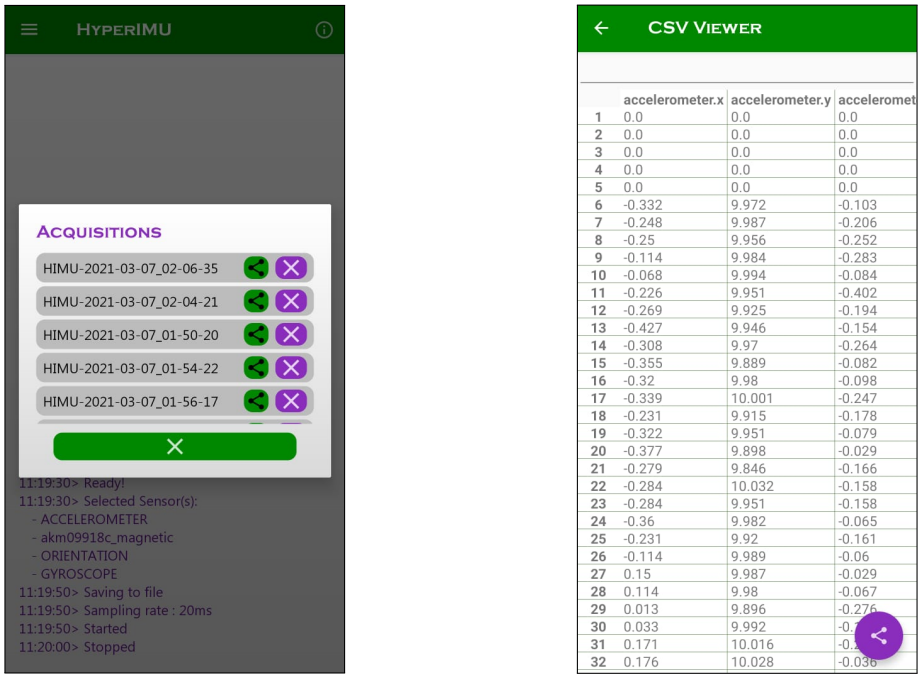


Figura A.7: *Adquisición de datos*

En la Figura A.7 se muestra una toma de datos por medio de la aplicación con sus respectivos parámetros configurados Por último, al finalizar la grabación de una marcha se procede a presionar el botón central verde de nuevo, para finalizar la obtención de los datos y regresar al menú principal. Para visualizar los resultados obtenidos se debe ingresar de nuevo a las 3 rayas de la parte superior izquierda de la pantalla principal, en donde dice *Acquisitions*

Figura A.8: *Adquisición de datos*

A continuación se muestra la forma en como se graban los archivos a través de la aplicación HyperIMU



(a) *Finalización de la toma de datos* (b) *Archivos guardados en formato .CSV*

Figura A.9: Vista previa de los datos adquiridos

En la Figura A.9 se muestran los datos almacenados localmente; estos se pueden visualizar en el entorno de la aplicación. Posteriormente se procede a transferir estos archivos al computador mediante cable USB o también la aplicación posee la opción de compartir directamente los archivos a *google drive*. El nombre de cada archivo (cada marcha), se almacena con el nombre por defecto que le asigna la aplicación, el cual es un prefijo **"HIMU"** seguido de la fecha de grabación (**AA-MM-DD**) y la hora configurada en el *smartphone* (**HH-MM-SS**).

Anexo B

Puesta en marcha código principal.

En éste anexo se documentará el paso a paso que se debe realizar para analizar los datos en bruto obtenidos por la aplicación HyperIMU en el *smartphone*, va desde cómo se deben ordenar los archivos hasta mostrar cuál es el archivo final con todas las estimaciones de longitud de zancada obtenidas por los algoritmos.

Ordenamiento de los archivos.

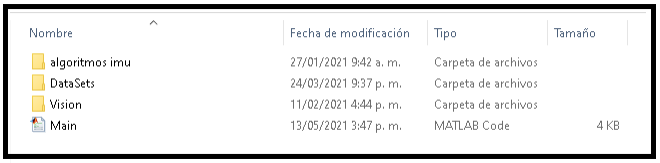


Figura B.1: Carpeta principal.

En la Figura B.1 se muestra la carpeta principal donde se encuentran todos los archivos concernientes a los algoritmos. En las subcarpetas de **algoritmos IMU** y **Vision** se encuentran las implementaciones de los algoritmos correspondientes y se encuentran organizados como se muestra en la siguente Figura B.2.

(a) (b)

Figura B.2: Estructura subcarpetas de algoritmos: IMU **(a)** y Vision **(b)**

En la subcarpeta **Datasets** se deben almacenar los archivos resultantes siguiendo una estructura lógica establecida por los investigadores del proyecto, la cual se detalla en la Figura B.3

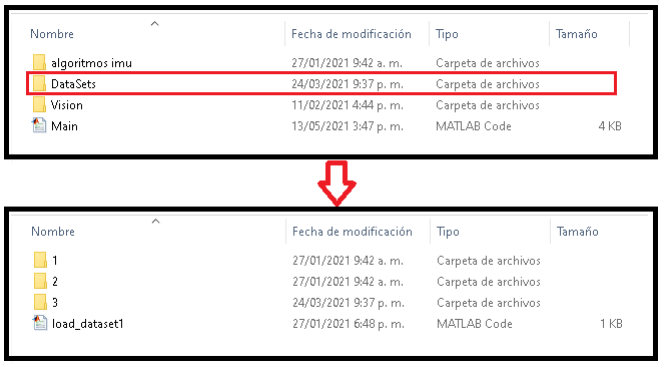


Figura B.3: Estructura Subcarpeta **Datasets**.

las carpetas numeradas almacenan los archivos de cada una de las personas participantes de las pruebas realizadas, dentro de ellas existe también un orden lógico que se mostrará en la Figura B.4.

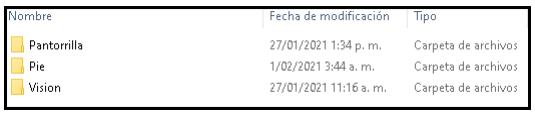


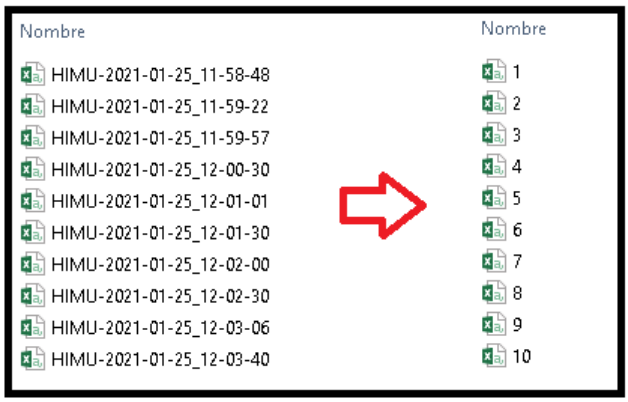
Figura B.4: Orden lógico de almacenamiento de archivos para análisis de datos por persona.

Se observan las carpetas **Pantorrilla**, **Pie** y **Vision** que son las carpetas donde finalmente se almacenan los archivos *.csv* y *.mp4* correspondientes a los archivos obtenidos por la aplicación HyperIMU en cada una de las dos ubicaciones donde se colocaron los dispositivos y por la grabación de la marcha respectivamente.

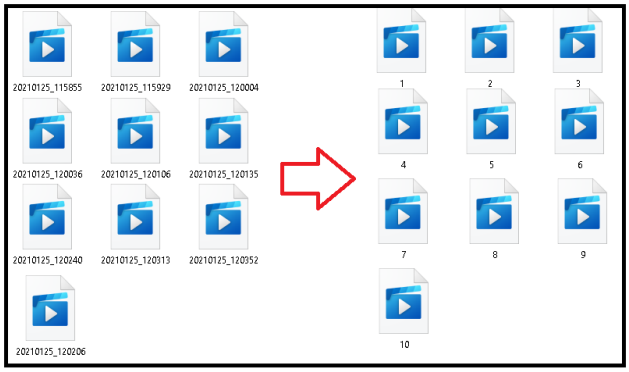
Tanto los archivos IMU cómo las grabaciones de la marcha deben ordenarse de tal forma que correspondan entre sí, debido a que los datos son tomados de manera simultánea para cada una de las marchas de las personas, para conservar el orden, los archivos se enumeran de tal forma que correspondan con la misma marcha en cada una de las carpetas, es decir el archivo **1** en la carpeta **Pantorrilla** debe corresponder los los archivos **1** en las carpetas **Pie** y **Vision** ya que corresponden a los datos tomados de la misma marcha. En éste proyecto se tomaron 10 marchas por persona por tanto en las carpetas **Pantorrilla**, **Pie** y **Vision** se guardarán 10 archivos respetivamente.

Los archivos inicialmente están guardados con nombres dados los la aplicación HyperIMU para los archivos IMU y por el *smartphone* para las grabaciones, estos deben cambiarse manualmente con la enumeración correspondiente. Existe una ventaja y es que se puede comprobar el orden de generación de los archivos mediante el nombre, ya que en éste se escribe la estampa de tiempo

fecha/**hora** en la cual se generó el archivo, con esto, no existe riesgo de enumerar los archivos de manera errónea como se muestra en la Figura B.5.



(a)



(b)

Figura B.5: Enumeración archivos de datos en bruto: IMU **(a)** y Vision **(b)**

En cuanto a los archivos *.csv* debe eliminarse manualmente uno por uno las primeras tres filas del arreglo de datos ya que se encuentra en éstas información no relevante y haciendo ésto disminuimos complejidad a la hora de leer el arreglo en Matlab como se muestra en la Figura B.6.

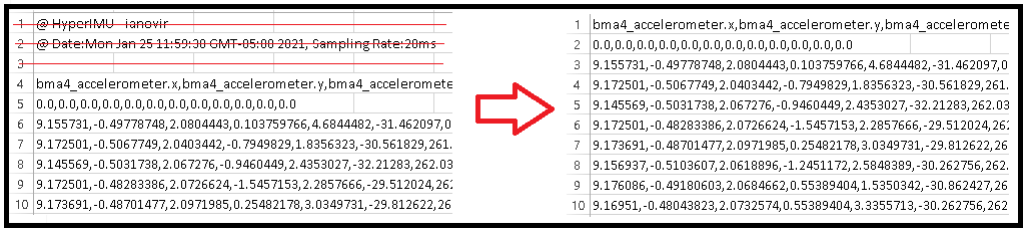


Figura B.6: Eliminación de las primeras tres filas de los archivos *.csv*.

Función implementada en Matlab para cargar los datos al espacio de trabajo.

```
function [A,W,O,Ts]  = load_dataset1(Filename)
%Data set

data                = readtable(Filename,'VariableNamingRule','preserve');
%Se unifican los nombres de las variables dado que la aplicación
%"HyperImu" genera nombres dependientes del hardware de cada teléfono.
if width(data) == 12
    data.Properties.VariableNames = {'Ax','Ay','Az','Mx','My','Mz','Ox','Oy','Oz','Wx','Wy','Wz'};
else
    error('El archivo de datos debe contener 12 columnas');
end

A       = [table2array(data(:,'Ax')) table2array(data(:,'Ay')) table2array(data(:,'Az'))];
O       = [table2array(data(:,'Ox')) table2array(data(:,'Oy')) table2array(data(:,'Oz'))];
W       = [table2array(data(:,'Wx')) table2array(data(:,'Wy')) table2array(data(:,'Wz'))];
Ts          = 20e-3;


end
```

Como se pudo observar, en la función implementada, la variable **Filename** lleva la ruta y el nombre del archivo que se va a cargar, estos se colocan en el código principal **Main.m** que se muestra en el anexo **B**. Luego se cambian los nombres de cada una de las columnas con **Ax**, **Ay**, **Az**, **Mx**, **My**, **Mz**, **Ox**, **Oy**, **Oz**, **Wx**, **Wy**, **Wz** correspondientes a las datos de aceleración, dirección de campo magnético, orientación y velocidad angular en los ejes **X**, **Y** y **Z**, respectivamente.

Luego cada terna de columnas se almacenan en las matrices **A**, **O**, **W** para los datos de aceleración, orientación y velocidad angular, respectivamente, también se coloca la velocidad de muestreo correspondiente a 20 ms.

Compilación del código principal para analizar todos los datos en bruto Matlab

A continuación se describe el código pincipal para el análisis de los datos en bruto, éste es un proceso semi-automático y consta de seis secciones que deben **ejecutarse una a una**[1].

1. **Parametrización inicial de las condiciones del experimento.**

```
1  %%Archivo principal para la ejecución de algoritmos de estimación de la longitud de zancada en marcha ↩
       humana.
2  % Marlio Alejandro Chicue Restrepo
3  % Enmanuel Berruecos Gómez
4
5  % Nota: ejecutar sección por sección.
6  % Al terminar la ejecición de los Algoritmos, las variables de interés
7  % serán LongZ, LongZ1, LongZ2 y LongZ3, donde se guardan los datos de
8  % estimación de longitud de zancada obtenida de los algoritmos de visión,
9  % Algoritmo IMU 1, Algoritmo IMU 2 y Algoritmo IMU 3 respectivamente, dada en centímetros.
10
11 % Los datos se guardarán en un archivo.mat el cual contendrá un arreglo de matrices LongZ,
12 % LongZ1, LongZ2 y LongZ3 donde se guardaran los resultados estimados de cada persona cómo una
13 % matriz por cada uno de los algoritmos.
14
15 % En cada matriz, cada columna a una zancada y cada fila corresponde a una
16 % marcha.
17
18 close all, clear all, clc
19 addpath('algoritmos imu/Algoritmo1');    %Agregar carpeta donde se encuentra el algoritmo IMU 1
20 addpath('algoritmos imu/Algoritmo2');    %Agregar carpeta donde se encuentra el algoritmo IMU 2
21 addpath('algoritmos imu/Algoritmo3');    %Agregar carpeta donde se encuentra el algoritmo IMU 3
22 addpath('Datasets');                     %Agregar carpeta DataSets IMU
23 addpath('Vision');                       %Agregar carpeta donde se encuentra el algoritmo Visión
24 CarpetaV='/Vision/';                     %Agregar ruta datasets visión
25 FormatoV='.mp4';                         %Formato videos
26 CarpetaIMU='Datasets/';                  %Agregar ruta datasets IMU
27 NumMuestras=10;    %Se coloca manualmente el numero de muestras que se van a analizar.
28 NumPersonas=3;     %Se coloca manuelmente el numero de personas que realizaron el experimento.
29 %Promedio zancada de cada Persona (longitud entre dos de las marcas dispuestas en el suelo).
30 %se debe escoger analizando la marcha de cada persona.
31 P=[90 90 90];
```

Desde la línea **19** hasta la **23** se agregan las carpetas donde se encuentran almacenados los algoritmos IMU y del sistema óptico de medida, y la carpeta de Datasets. Las variables **VarpetaV**, **FormatoV** y **CarpetaIMU** se utilizan luego para concatenar la cadena de ubicación de cada uno de los archivos a analizar, y las variables **NumMuestras** y **NumPersonas** almacenan el número de marchas por persona y el número de personas que realizan el experimento, respectivamente. Las últimas dos variables mencionadas sirven para estructurar el ciclo para el análisis de cada uno de los archivos de marcha en el algoritmo correspondiente. La variable **P** corresponde a la longitud de zancada que se tomó como referencia colocando las marcas para validar el sistema óptico de medida, se debe guardar por persona expresado en centímetros.

[1] Para el sistema óptico de medida, es un proceso tardado debido a que analiza y muestra marco a marco la estimación

2. **Ciclo para análisis de los archivos del sistema óptico de medida.**

```
%% Algoritmo Visión
for M=1:NumPersonas
i=0;
while i<NumMuestras
    i=i+1
    LZ=AV(strcat('Datasets/',num2str(M),CarpetaV,num2str(i),FormatoV),P(1,M));
    [a b]=size(LZ);
     if a==3
     LongZ (i,:,M)     = LZ;
     elseif a<3
     LongZ (i,:,M)     = [LZ(1,1) LZ(2,1) 0];
    end
end
end
```

El ciclo para el análisis de los archivos del sistema óptico de media se realizan uno a uno por cada persona y durante el ciclo de cada persona se analiza cada grabación de la marcha de la persona correspondiente. Al algoritmo se le entrega concatenado la dirección de ubicación de cada una de las grabaciones. Como resultado entrega el arreglo **LongZ** que almacena tantas matrices como personas hayan realizado el experimento, cada matriz tiene una dimensión de tantas columnas como zancadas haya dado la persona y por tantas filas como marchas haya realizado la persona; Cada una de las celdas corresponden a la estimación de la longitud de la zancada correspondiente.

3. **Ciclo para análisis de los archivos IMU mediante Algoritmo IMU 1.**

```
%% Algoritmo Distance_Cellphone_100cm
close all
for M=1:NumPersonas
    i=0;
while i<NumMuestras
    i=i+1
    [A,W,O,Ts]      = load_dataset1(strcat(CarpetaIMU,num2str(M),'/Pie/',num2str(i)));
    LongZ1(i,:,M)     = get_stride_length(A,W,O,Ts)*100;
    close all
end
end
```

El ciclo para el análisis de los archivos IMU del pie se realizan uno a uno por cada persona y durante el ciclo de cada persona se analiza cada archivo *.csv* de la marcha de la persona correspondiente. A la función para cargar los datos IMU se le entrega concatenado la dirección de ubicación de cada uno de los archivos *.csv*. Como resultado entrega el arreglo **LongZ1** que almacena tantas matrices como personas hayan realizado el experimento, cada matriz tiene una dimensión de tantas columnas como zancadas haya dado la persona y por tantas filas como marchas haya realizado la persona; Cada una de las celdas corresponden a la estimación de la longitud de la zancada correspondiente.

4. **Ciclo para análisis de los archivos IMU mediante Algoritmo IMU 2.**

```
%% Algoritmo Orientación Mahony
close all
for M=1:NumPersonas
i=0;
while i<NumMuestras
i=i+1
[A,W,O,Ts]      = load_dataset1(strcat(CarpetaIMU,num2str(M),'/Pie/',num2str(i)));
LongZ2(i,:,M)     = Script12(Ts,W,A)*100;
close all
end
end
```

El ciclo para el análisis de los archivos IMU del pie se realizan uno a uno por cada persona y durante el ciclo de cada persona se analiza cada archivo *.csv* de la marcha de la persona correspondiente. A la función para cargar los datos IMU se le entrega concatenado la dirección de ubicación de cada uno de los archivos *.csv*. Como resultado entrega el arreglo **LongZ2** que almacena tantas matrices como personas hayan realizado el experimento, cada matriz tiene una dimensión de tantas columnas como zancadas haya dado la persona y por tantas filas como marchas haya realizado la persona; Cada una de las celdas corresponden a la estimación de la longitud de la zancada correspondiente.

5. **Ciclo para análisis de los archivos IMU mediante Algoritmo IMU 3.**

```
%% Algoritmo Walking speedestimationusingashank-mountedinertialmeasurementunit
close all
for M=1:NumPersonas
i=0;
while i<NumMuestras
i=i+1
[A,W,O,Ts]      = load_dataset1(strcat(CarpetaIMU,num2str(M),'/Pantorrilla/',num2str(i),'.csv'));
LongZ3(i,:,M)     = Algoritmo3(A,W,Ts)*100;
close all
end
end
```

El ciclo para el análisis de los archivos IMU de la pantorrilla se realizan uno a uno por cada persona y durante el ciclo de cada persona se analiza cada archivo *.csv* de la marcha de la persona correspondiente. A la función para cargar los datos IMU se le entrega concatenado la dirección de ubicación de cada uno de los archivos *.csv*. Como resultado entrega el arreglo **LongZ3** que almacena tantas matrices como personas hayan realizado el experimento, cada matriz tiene una dimensión de tantas columnas como zancadas haya dado la persona y por tantas filas como marchas haya realizado la persona; Cada una de las celdas corresponden a la estimación de la longitud de la zancada correspondiente.

6. **Guardar los arreglos en un archivo *.mat***

```
%% Guardar Longitud Zancadas en .mat
save LongZancadas.mat LongZ LongZ1 LongZ2 LongZ3 -v7.3;
```

En ésta sección se almacenan los resultados en **LongZancadas.mat**, éste queda almacenado junto al archivo principal **Main.m** como se muestra en la Figura B.7.

Figura B.7: Archivo con los resultados de estimación.

Anexo C

Códigos algoritmos basados en IMU.

En este anexo se presentan los códigos de los algoritmos basados en unidades inerciales de medida

Algoritmo 1.

```
function Step_Length  = get_stride_length(A,W,O,Ts)
%% Step 2: Angular velocity is used to separate motion from rest phases.
%"1" for rest, and  "0" for no rest.
WNorm           = vecnorm(W')';
WNorm           = moving_average_filter(WNorm,20);
A               = moving_average_filter(A,5);
WBinary         = WNorm < 0.95;
Begin_Rest      = find(diff(WBinary) == +1);
End_Rest        = find(diff(WBinary) == -1);
figure; plot(WNorm); hold on; plot(WBinary);
title('Velocidad angular normalizada - vector de detección de fases.');
xlabel('Time [centiseconds]');
ylabel('Angular velocity [rad/sec]');
Comienzo = End_Rest(1);
%This conditional ensures that BFFoot(i) < EFFoot(i), i = 1,2,...
if Begin_Rest(1) >  End_Rest(1)
    End_Rest          = End_Rest(2:end);
end

%% Step 2. Accelerations are expressed in a common reference frame
Q               = angle2quat(O(:,1),O(:,2),O(:,3),'ZYX');
ACommon         = quatrotate(Q,A);

%Calculation loop for each motion phase
N               = length(Begin_Rest);
PGround         = zeros(N,3);
for i = 1:N
    if i==1
```

```
1
2       %Direction of the gravity vector when the foot is at rest. Here is
3       %important to clarify that the rest phase is a subphase of the contact
4       %phase.
5       ARest               = ACommon(Comienzo-20:Comienzo,:);
6       AFoot_t0            = mean(ARest);
7
8       %Calculation of the rotation matrix from the IMU's inertial reference
9       %frame to ground frame. In the ground frame the Z-axis points in the
10      %direction of the gravity.
11      Z                   = AFoot_t0' / norm(AFoot_t0);
12      Y                   = cross(Z,[1;0;0]);
13      Y                   = Y / norm(Y);
14      X                   = cross(Y,Z);
15      X                   = X / norm(X);
16      R_From_F_to_G       = [X,Y,Z]';
17      Q_From_F_to_G       = dcm2quat(R_From_F_to_G);
18
19      %Accelerations during the motion phase are projected to the ground
20      %frame.
21      AMotion             = ACommon(End_Rest(i):Begin_Rest(i+1),:);
22      AGround             = quatrotate(Q_From_F_to_G,AMotion);
23
24      %Gravity acceleration, which is obtained during the rest phase, is
25      %subtracted to the accelerations of the motion phase
26      AGravity            = quatrotate(Q_From_F_to_G,AFoot_t0);
27      AGround             = AGround - AGravity;
28
29      %Integration of the acceleration in the Ground frame.
30      T0                  = End_Rest(i)*Ts;
31      Tf                  = Begin_Rest(i+1)*Ts;
32      %Velocity at the end of the motion phase
33      C                   = Ts*trapz(AGround) / (Tf-T0);
34      AGround_No_Drift    = AGround - C;
35      %Velocity
36      VGround             = Ts*cumtrapz(AGround_No_Drift);
37
38      %Integration of the linear velocity in the Ground frame.
39      PGround(i,:)        = Ts*trapz(VGround);
40      end
41
42      if i>1
43      %%Step 3:
44      %Direction of the gravity vector when the foot is at rest. Here is
45      %important to clarify that the rest phase is a subphase of the contact
46      %phase.
47      ARest               = ACommon(Begin_Rest(i-1)+1:End_Rest(i-1),:);
48      AFoot_t0            = mean(ARest);
49      %%Step 4:
50      %Calculation of the rotation matrix from the IMU's inertial reference
51      %frame to ground frame. In the ground frame the Z-axis points in the
52      %direction of the gravity.
53      Z                   = AFoot_t0' / norm(AFoot_t0);
54      Y                   = cross(Z,[1;0;0]);
55      Y                   = Y / norm(Y);
56      X                   = cross(Y,Z);
57      X                   = X / norm(X);
58      R_From_F_to_G       = [X,Y,Z]';
59      Q_From_F_to_G       = dcm2quat(R_From_F_to_G);
60      %%Step 5:
```

```
    %Accelerations during the motion phase are projected to the ground
    %frame.
    AMotion             = ACommon(End_Rest(i-1):Begin_Rest(i-1+1),:);
    AGround             = quatrotate(Q_From_F_to_G,AMotion);
    %%Step 6:
    %Gravity acceleration, which is obtained during the rest phase, is
    %subtracted to the accelerations of the motion phase
    AGravity            = quatrotate(Q_From_F_to_G,AFoot_t0);
    AGround             = AGround - AGravity;
    %%Step 7:
    %Integration of the acceleration in the Ground frame.
    T0                  = End_Rest(i-1)*Ts;
    Tf                  = Begin_Rest(i-1+1)*Ts;
    %Velocity at the end of the motion phase
    C                   = Ts*trapz(AGround) / (Tf-T0);
    AGround_No_Drift    = AGround - C;
    %Velocity
    VGround             = Ts*cumtrapz(AGround_No_Drift);
    %%Step 8:
    %Integration of the linear velocity in the Ground frame.
    PGround(i,:)        = Ts*trapz(VGround);
    end
end
Step_Length     = sqrt(PGround(:,1).^2 + PGround(:,2).^2);
fprintf('Step length = %.2f cm\n',100*Step_Length);
end

%function Integ_Signal = spline_integrator(Signal,Ts)
%%Dimensions of the signal
%[NRows,NCols]    = size(Signal);
%
%%A time variable is created
%Time            = Ts * (0:NRows-1)';
%
%%Spline approximation
%Integ_Signal    = zeros(1,NCols);
%for i = 1:NCols
%    pp              = spline(Time,Signal(:,i));
%    Integ_Signal(i) = integral(@(T) ppval(pp,T),0,Time(end));
%end
%end
%
%function Integ_Signal = cumulative_spline_integrator(Signal,Ts)
%%Dimensions of the signal
%[NRows,NCols]    = size(Signal);
%
%%A time variable is created
%Time            = Ts * (0:NRows-1)';
%
%%Spline approximation
%Integ_Signal    = zeros(NRows,NCols);
%for col = 1:NCols
%    pp              = spline(Time,Signal(:,col));
%    for row = 1:NRows
%        Integ_Signal(row,col) = integral(@(T) ppval(pp,T),0,Time(row));
%    end
%end
%end
```

Algoritmo 2.

```
function Step_Length=Script12(Ts,W,A)

addpath('algoritmos imu/Algoritmo2/Quaternions');
addpath('algoritmos imu/Algoritmo2/ximu_matlab_library');

%Import data
samplePeriod = 1/100;
A    = moving_average_filter(A,5);
time = (0:length(A))' * (Ts);
gyrX = rad2deg(W(:,1));
gyrY = rad2deg(W(:,2));
gyrZ = rad2deg(W(:,3));
accX = A(:,1)*(1/9.81);
accY = A(:,2)*(1/9.81);
accZ = A(:,3)*(1/9.81);
%-------------------------------------------------------------------
%Detect stationary periods
%Compute angular velocity magnitude
gyr_mag = sqrt(gyrX.*gyrX + gyrY.*gyrY + gyrZ.*gyrZ);
%HP filter angular velocity data
filtCutOff = 0.001;
[b, a] = butter(1, (2*filtCutOff)/(1/samplePeriod), 'high');
gyr_magFilt = filtfilt(b, a, gyr_mag);
%Compute absolute value
gyr_magFilt = abs(gyr_magFilt);
%LP filter angular velocity data
filtCutOff = 5;
[b, a] = butter(1, (2*filtCutOff)/(1/samplePeriod), 'low');
gyr_magFilt = filtfilt(b, a, gyr_magFilt);
%Threshold detection
stationary = gyr_magFilt <35;

Begin_Rest      = find(diff(stationary) == +1);
End_Rest        = find(diff(stationary) == -1);

startTime= (End_Rest(1)-150)*Ts;
stopTime=(Begin_Rest(length(Begin_Rest))+150)*Ts;
indexSel = find(sign(time-startTime)+1, 1) : find(sign(time-stopTime)+1, 1);
gyr_magFilt=gyr_magFilt(indexSel);
time = time(indexSel);
gyrX = rad2deg(gyrX(indexSel, :));
gyrY = rad2deg(gyrY(indexSel, :));
gyrZ = rad2deg(gyrZ(indexSel, :));
accX = accX(indexSel, :);
accY = accY(indexSel, :);
accZ = accZ(indexSel, :);
stationary=stationary(indexSel,:);
Begin_Rest      = find(diff(stationary) == +1);
End_Rest        = find(diff(stationary) == -1);

%-------------------------------------------------------------------
%Plot data raw sensor data and stationary periods

figure('Position', [9 39 900 600], 'NumberTitle', 'off', 'Name', 'Sensor Data');
```

```
ax(1) = subplot(2,1,1);
    hold on;
    plot(time, accX, 'r');
    plot(time, accY, 'g');
    plot(time, accZ, 'b');
    title('Accelerometer');
    plot(time, gyr_magFilt/100, ':k');
    plot(time, stationary, 'k', 'LineWidth', 2);
    xlabel('Time (s)');
    ylabel('Acceleration (g)');
    legend('X', 'Y', 'Z', 'WNorm', 'Stationary');
    hold off;
ax(2) = subplot(2,1,2);
    hold on;
    plot(time, gyrX, 'r');
    plot(time, gyrY, 'g');
    plot(time, gyrZ, 'b');
    title('Gyroscope');
    xlabel('Time (s)');
    ylabel('Angular velocity (^\circ/s)');
    legend('X', 'Y', 'Z');
    hold off;
linkaxes(ax,'x');

%------------------------------------------------------------------
%Compute orientation

quat = zeros(length(time), 4);
AHRSalgorithm = AHRS('SamplePeriod', 1/20, 'Kp', 1, 'KpInit', 20);

%Initial convergence
initPeriod = 2;
indexSel = 1 : find(sign(time-(time(1)+initPeriod))+1, 1);
for i = 1:2000
    AHRSalgorithm.UpdateIMU([0 0 0], [mean(accX(indexSel)) mean(accY(indexSel)) mean(accZ(indexSel))]);
end

%For all data
for t = 1:length(time)
    if(stationary(t))
        AHRSalgorithm.Kp = 0.5;
    else
        AHRSalgorithm.Kp = 0;
    end
    AHRSalgorithm.UpdateIMU(deg2rad([gyrX(t) gyrY(t) gyrZ(t)]), [accX(t) accY(t) accZ(t)]);
    quat(t,:) = AHRSalgorithm.Quaternion;
end
%------------------------------------------------------------------
%Compute translational accelerations
%Rotate body accelerations to Earth frame
acc = quaternRotate([accX accY accZ], quaternConj(quat));
%Convert acceleration measurements to m/s/s
acc = acc * 9.81 ;
%%Remove gravity from measurements
acc(:,3) = acc(:,3) - 9.81;
%Plot translational accelerations
figure('Position', [9 39 900 300], 'NumberTitle', 'off', 'Name', 'Accelerations');
hold on;
plot(time, acc(:,1), 'r');
plot(time, acc(:,2), 'g');
```

```
plot(time, acc(:,3), 'b');
title('Acceleration');
xlabel('Time (s)');
ylabel('Acceleration (m/s/s)');
legend('X', 'Y', 'Z');
hold off;

%-------------------------------------------------------------------------
%Compute translational velocities

%acc(:,3) = acc(:,3) - 9.81;

%Integrate acceleration to yield velocity
vel = zeros(size(acc));
for t = 2:length(vel)
    vel(t,:) = vel(t-1,:) + acc(t,:) * samplePeriod;
    if(stationary(t) == 1)
        vel(t,:) = [0 0 0];     %force zero velocity when foot stationary
    end
end
%Compute integral drift during non-stationary periods
velDrift = zeros(size(vel));
stationaryStart = find([0; diff(stationary)] == -1);
stationaryEnd = find([0; diff(stationary)] == 1);
for i = 1:numel(stationaryEnd)
    driftRate = vel(stationaryEnd(i)-1, :) / (stationaryEnd(i) - stationaryStart(i));
    enum = 1:(stationaryEnd(i) - stationaryStart(i));
    drift = [enum'*driftRate(1) enum'*driftRate(2) enum'*driftRate(3)];
    velDrift(stationaryStart(i):stationaryEnd(i)-1, :) = drift;
end
%Remove integral drift
vel = vel - velDrift;

%Compute translational position

%Integrate velocity to yield position
pos = zeros(size(vel));
for t = 2:length(pos)
    pos(t,:) = pos(t-1,:) + vel(t,:) * samplePeriod;    %integrate velocity to yield position
end

%Plot translational position
figure('Position', [9 39 900 600], 'NumberTitle', 'off', 'Name', 'Position');
hold on;
plot(time, pos(:,1), 'r');
plot(time, pos(:,2), 'g');
plot(time, pos(:,3), 'b');
title('Position');
xlabel('Time (s)');
ylabel('Position (m)');
legend('X', 'Y', 'Z');
hold off;

%-------------------------------------------------------------------------
%Plot 3D foot trajectory

%%Remove stationary periods from data to plot
%posPlot = pos(find(~stationary), :);
%quatPlot = quat(find(~stationary), :);
posPlot = pos;
```

```
quatPlot = quat;

%Extend final sample to delay end of animation
extraTime = 0;
onesVector = ones(extraTime*(1/samplePeriod), 1);
posPlot = [posPlot; [posPlot(end, 1)*onesVector, posPlot(end, 2)*onesVector, posPlot(end, 3)*onesVector]];
quatPlot = [quatPlot; [quatPlot(end, 1)*onesVector, quatPlot(end, 2)*onesVector, quatPlot(end, 3)*onesVector, quatPlot(end, 4)*onesVector]];

%Create 6 DOF animation
%SamplePlotFreq = 4;
%Spin = 120;
%SixDofAnimation(posPlot, quatern2rotMat(quatPlot), ...
%                'SamplePlotFreq', SamplePlotFreq, 'Trail', 'All', ...
%                'Position', [9 39 1280 768], 'View', [(100:(Spin/(length(posPlot)-1)):(100+Spin))', 10*ones(length(posPlot), 1)], ...
%                'AxisLength', 0.1, 'ShowArrowHead', false, ...
%                'Xlabel', 'X (m)', 'Ylabel', 'Y (m)', 'Zlabel', 'Z (m)', 'ShowLegend', false, ...
%                'CreateAVI', false, 'AVIfileNameEnum', false, 'AVIfps', ((1/samplePeriod) / SamplePlotFreq))
%

posi=zeros(length(Begin_Rest),3);

for K=1:length(Begin_Rest)
    if K==1
    posi(K,:)=pos(Begin_Rest(K),:);
    elseif K>1
     posi(K,:)=pos(Begin_Rest(K),:)-pos(Begin_Rest(K-1),:);
    end
end

Step_Length     = sqrt(posi(:,1).^2 + posi(:,2).^2);

fprintf('Step length  = %.2f cm\n',100*Step_Length);

end
```

Algoritmo 3.

```
function stride_length  = Algoritmo3(A,W,Ts)
%Implementación del algoritmo para estimación de longitud propuesto en:
%[1] Li, Q.; Young, M.; Naing, V. & Donelan, J.
%Walking speed estimation using a shank-mounted inertial measurement unit
%Journal of Biomechanics, 2010, 43, 1640 - 1643.

%Se suavisa la velocidad angular con un filtro de media móvil de orden 5.
time    = (1:length(A))' * (Ts);
angular_vel_z       = moving_average_filter(W(:,3),10);
%Busqueda de máximos locales negativos cuya prominencia sea mayor a 10
%grados/segundo.
threshold           = deg2rad(10);
[peak,location]     = findpeaks(angular_vel_z,'MinPeakProminence',threshold);
%Elimina picos con velocidad angular mayor que 10 grados/seg. Con esto se
%asegura que solo se encuentren los picos negativos.
location            = location(peak < threshold);
peak                = peak(peak < threshold);
%Display
figure;
plot(time,rad2deg(angular_vel_z),'b-',location*Ts,rad2deg(peak),'ro'); grid on;
title('Angular velocity medio-lateral axis of shank');
xlabel('Time [seconds]');
ylabel('Angular velocity [deg/sec]');

%Se calcula el número de zancadas como la cantidad de máximos negativos de
%velocidad angular.
nstrides            = length(location)-1;
stride_length       = zeros(nstrides,1);
nn=1;
for nn=1:3
    if nn==1
    %Rango de datos de la zancada actual
    range               = 1:location(1);

    %Se seleccionan datos entre dos picos consecutivos
    wz_stride           = angular_vel_z(range);

    %Se calcula el angulo theta acuerdo con la ecuación (2) de [1]
    theta               = cumsum(wz_stride) * Ts;

    %Aceleraciones tangencial y normal
    accel_tangential    = moving_average_filter(A(range,1),5);
    accel_normal        = moving_average_filter(A(range,2),5);

    %Se convierten las aceleraciones al sistema coordenado de referencia.
    %Ecuación (1) de [1]
    [ax,ay]             = from_sensor_to_world(theta,accel_tangential,accel_normal);
    %Se integran las aceleraciones "ax" y "ay"
    vx                  = cumsum(ax) * Ts;
    vy                  = cumsum(ay) * Ts;
    %Se integran las velocidades "vx" y "vy"
    sx                  = cumsum(vx) * Ts;
    sy                  = cumsum(vy) * Ts;
    %Se resta el factor 0.5 * tstride * vx(T)
```

```
    tstride             = (location(1) - 1) * Ts;
    sx                  = sx - 1/2 * tstride * vx(end);
    sy                  = sy - 1/2 * tstride * vy(end);
    %Longitud de zancada
    stride_length(nn)    = sqrt(sx(end)^2 + sy(end)^2);
    end

    if nn~=1&&nn~=3
     for i = 1:nstrides
     %Rango de datos de la zancada actual
     range              = location(i):location(i+1);

     %Se seleccionan datos entre dos picos consecutivos
     wz_stride          = angular_vel_z(range);

     %Se calcula el angulo theta acuerdo con la ecuación (2) de [1]
     theta              = cumsum(wz_stride) * Ts;

     %Aceleraciones tangencial y normal
     accel_tangential   = moving_average_filter(A(range,1),5);
     accel_normal       = moving_average_filter(A(range,2),5);

     %Se convierten las aceleraciones al sistema coordenado de referencia.
     %Ecuación (1) de [1]
     [ax,ay]            = from_sensor_to_world(theta,accel_tangential,accel_normal);

     %Se integran las aceleraciones "ax" y "ay"
     vx                 = cumsum(ax) * Ts;
     vy                 = cumsum(ay) * Ts;

     %Se integran las velocidades "vx" y "vy"
     sx                 = cumsum(vx) * Ts;
     sy                 = cumsum(vy) * Ts;

     %Se resta el factor 0.5 * tstride * vx(T)
     tstride            = (location(i+1) - location(i)) * Ts;
     sx                 = sx - 1/2 * tstride * vx(end);
     sy                 = sy - 1/2 * tstride * vy(end);

     %Longitud de zancada
     stride_length(nn)    = sqrt(sx(end)^2 + sy(end)^2);
     end
    end
    if nn==3
    %Rango de datos de la zancada actual
    range               = location(2):(length(angular_vel_z)-30);

    %Se seleccionan datos entre dos picos consecutivos
    wz_stride           = angular_vel_z(range);

    %Se calcula el angulo theta acuerdo con la ecuación (2) de [1]
    theta               = cumsum(wz_stride) * Ts;

    %Aceleraciones tangencial y normal
    accel_tangential    = moving_average_filter(A(range,1),5);
    accel_normal        = moving_average_filter(A(range,2),5);

    %Se convierten las aceleraciones al sistema coordenado de referencia.
    %Ecuación (1) de [1]
    [ax,ay]             = from_sensor_to_world(theta,accel_tangential,accel_normal);
```

```
1
2       %Se integran las aceleraciones "ax" y "ay"
3       vx                  = cumsum(ax) * Ts;
4       vy                  = cumsum(ay) * Ts;
5
6       %Se integran las velocidades "vx" y "vy"
7       sx                  = cumsum(vx) * Ts;
8       sy                  = cumsum(vy) * Ts;
9
10      %Se resta el factor 0.5 * tstride * vx(T)
11      tstride             = ((length(angular_vel_z)-30) - location(2)) * Ts;
12      sx                  = sx - 1/2 * tstride * vx(end);
13      sy                  = sy - 1/2 * tstride * vy(end);
14
15      %Longitud de zancada
16      stride_length(nn)    = sqrt(sx(end)^2 + sy(end)^2);
17      end
18  end
19  disp('Longitudes de zancada obtenidas');
20  disp(stride_length);
21  end
```

Anexo D

Código algoritmo sistema óptico de medida.

En este anexo se presentan el código del sistema óptico de medida.

Algoritmo

```
function LongZ = AV(Filename, Prom)
vidObj = VideoReader(Filename);
 MT=0;%Marcos totales.
 x=1;%Contador de marcos procesados por algoritmo.
 z=0;%Contador de las veces que se inicia una zancada.
 conP=0;%Contador de zancadas dadas.
while hasFrame(vidObj)

vidFrame = readFrame(vidObj);
Gy=rgb2gray(vidFrame);
BW=im2bw(Gy,0.2);
BW=imcomplement(BW);
se = strel('disk', 7); %Elemento estructurante tipo disco de tamanno 5.
BW=imfill(BW,'holes'); %Rellenamos los huecos que puedan haber en los objetos cerrados
BW = imopen(BW,se);
[a,b]=size(BW);
stats= regionprops(BW,'Centroid','BoundingBox');
n=length(stats);
%imshow(BW)

if (n==2)
X1=stats(1).Centroid(1);
Y1=stats(1).Centroid(2);
X2=stats(2).Centroid(1);
Y2=stats(2).Centroid(2);
```

```
1
2  distancia1(x,1)=sqrt((X2-X1)^2+(Y2-Y1)^2);%Distancia en pixeles entre los centros de los marcadores.
3
4  if ((abs(X1-X2)>=0)&&(abs(X1-X2)<=5&&z==0))||(z>0&&(abs(X1-X2)>=0)&&(abs(X1-X2)<=5)&&Cent(x-1,1)>(Prom-6))
5   z=z+1;
6   Marca1(z,1)=X1;
7   Marca1(z,2)=Y1;
8   Marca2(z,1)=X2;
9   Marca2(z,2)=Y2;
10  if z>1
11   LongZ(z-1,1)=Cent(x-1,1);
12   conP=conP+1;
13  end
14 end
15
16 imshow(vidFrame)
17 hold on
18 line([X1 X2],[Y1 Y2])
19
20 if z>0
21  line([Marca1(z,1) Marca2(z,1)],[Marca1(z,2) Marca2(z,2)])
22  MX=[X1 X2 Marca1(z,1) Marca2(z,1)];
23  MY=[Y1 Y2 Marca1(z,2) Marca2(z,2)];
24
25  if MY(1,2)>MY(1,1)
26      MX(1,1:2)=[X2 X1];
27      MY(1,1:2)=[Y2 X1];
28  end
29  if MY(1,4)>MY(1,3)
30      MX(1,3:4)=[Marca2(z,1) Marca1(z,1)];
31      MY(1,3:4)=[Marca2(z,2) Marca1(z,2)];
32  end
33  line([MX(1,3) MX(1,1)],[MY(1,3) MY(1,1)],'Color','green','LineStyle','--')
34  distancia3(x,1)=sqrt((MX(1,1)-MX(1,3))^2+(MY(1,1)-MY(1,3))^2);
35
36  Cent(x,1)=(distancia3(x,1)*20)/distancia1(x,1);
37  text(500,400,['Distancia en pixeles: ',num2str(distancia3(x,1))],'Color','r','FontSize',11)
38  text(500,430,['Distancia en centimetros: ',num2str(Cent(x,1)),' Cm'],'Color','r','FontSize',11)
39
40  [A,B]=size(Marca1);
41  r = 17;
42  for L=1:A
43   Cir1 = [Marca1(L,1) Marca1(L,2)];
44   Cir2 = [Marca2(L,1) Marca2(L,2)];
45   pos1 = [Cir1-r 2*r 2*r];
46   pos2 = [Cir2-r 2*r 2*r];
47   rectangle('Position',pos1,'Curvature',[1 1],'faceColor',[0 0 0])
48   rectangle('Position',pos2,'Curvature',[1 1],'faceColor',[0 0 0])
49   line([Marca1(L,1) Marca2(L,1)],[Marca1(L,2) Marca2(L,2)] )
50   hold off
51   pause(0.01);
52  end
53 end
54
55   for l=1:n
```

```
    BB=stats(l).BoundingBox;
    rectangle('position',[BB(1),BB(2),BB(3), BB(4)],'EdgeColor','r','LineWidth',2)
    hold off
    pause(0.01);
   end
   x=x+1;
end
MT=MT+1;
end

fprintf('El numero de zancadas fue %.f \n',conP);
fprintf('Longitud en centimetros %.4f \n',LongZ);

end
```

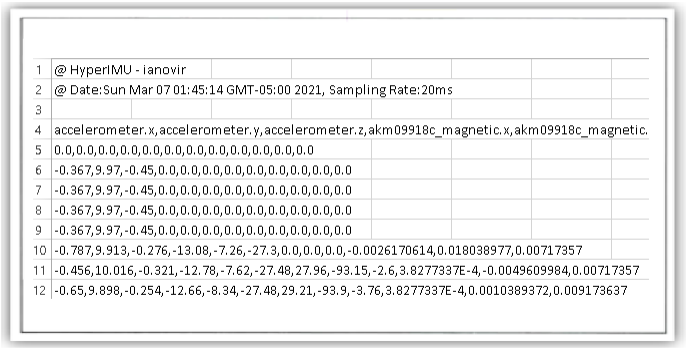
Anexo E

Implementación de algoritmos IMU.

En éste anexo se explicará con más detalle la implementación de los algoritmos IMU.

Para facilidad de implementación, el primer problema se aborda de la misma manera para todos los algoritmos basados en unidades de medición inercial así:

En la Figura E.1 vemos el archivo en formato *.csv* entregado por la aplicación HyperIMU instalada en los *smartphone* destinados a la toma de los datos inerciales en la marcha. Las primeras tres filas de este archivo estan destinadas al encabezado, aquí aparece el nombre de la aplicación, fecha y tiempo de muestreo de los datos (deben ser borradas manualmente). La fila siguiente está destinada a los nombres de cada variable, estos dependen de cada dispositivo, por esto deben cambiarse para estandarizar el nombre de las mismas; El resto de las filas están asociadas a los datos tomados.



1	@ HyperIMU - ianovir
2	@ Date:Sun Mar 07 01:45:14 GMT-05:00 2021, Sampling Rate:20ms
3	
4	accelerometer.x,accelerometer.y,accelerometer.z,akm09918c_magnetic.x,akm09918c_magnetic.
5	0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0
6	-0.367,9.97,-0.45,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0
7	-0.367,9.97,-0.45,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0
8	-0.367,9.97,-0.45,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0
9	-0.367,9.97,-0.45,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0
10	-0.787,9.913,-0.276,-13.08,-7.26,-27.3,0.0,0.0,0.0,-0.0026170614,0.018038977,0.00717357
11	-0.456,10.016,-0.321,-12.78,-7.62,-27.48,27.96,-93.15,-2.6,3.8277337E-4,-0.0049609984,0.00717357
12	-0.65,9.898,-0.254,-12.66,-8.34,-27.48,29.21,-93.9,-3.76,3.8277337E-4,0.0010389372,0.009173637

Figura E.1: Raw de datos entregado por HyperIMU en formato csv.

El siguiente fragmento de código muesta la implementación del paso uno (recepción de los datos).

```
1 data                = readtable(Filename,'VariableNamingRule','preserve');
2 %Se unifican los nombres de las variables dado que la aplicación
```

```
%"HyperImu" genera nombres dependientes del hardware de cada teléfono.
if width(data) == 12
    data.Properties.VariableNames = {'Ax','Ay','Az','Mx','My','Mz','Ox','Oy','Oz','Wx','Wy','Wz'};
else
    error('El archivo de datos debe contener 12 columnas');
end

A       = [table2array(data(:,'Ax')) table2array(data(:,'Ay')) table2array(data(:,'Az'))];
O       = [table2array(data(:,'Ox')) table2array(data(:,'Oy')) table2array(data(:,'Oz'))];
W       = [table2array(data(:,'Wx')) table2array(data(:,'Wy')) table2array(data(:,'Wz'))];
Ts          = 20e-3;
```

En en fragmento de código anterior se puede apreciar la función codificada en Matlab que permite extraer los datos del archivo *.csv* mencionado. Allí se crean las matrices de tres columnas *X, Y* y *Z* de cada una de las variables de interés: *aceleración (A), orientación (O)* y *velocidad angular (W)*, las cuales son almacenadas en el espacio de trabajo de Matlab.

En las subsecciones siguientes, se expondrá cómo son resueltos los cinco problemas siguientes, por cada uno de los algoritmos.

E.0.1. Algoritmo 1 tomado de: *Estimation of foot trajectory during human walking by a wearable inertial measurement unit mounted to the foot* [1]

Para la correcta implementación del algoritmo mencionado en este trabajo, la IMU debe colocarse en el dorso del pie como se muestra en la Figura E.2.

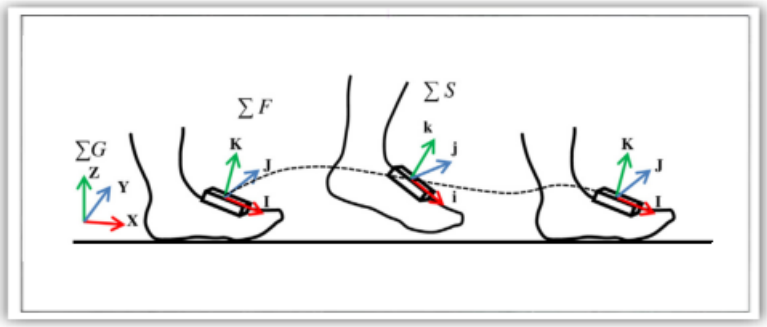


Figura E.2: Colocación de la unidad de medición inercial en el cuerpo [1].

- **Detección de las fases de movimiento de la pierna:**

  En la Figura E.3 se muestra la gráfica de la velocidad angular normalizada (azul) y la detección de inicios y finales de fases en el ciclo de la marcha (rojo).

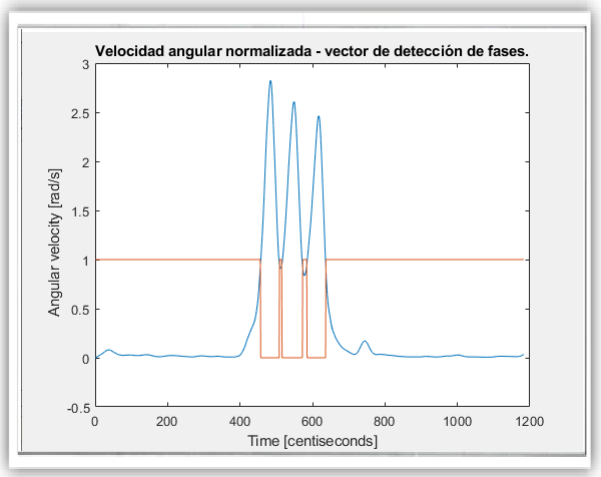


Figura E.3: Velocidad angular normalizada - vector de detección de fase.

En el siguiente fragmento de código se muestra la implementación de la detección de fase de movimiento de la pierna.

```
%% Step 2: Angular velocity is used to separate motion from rest phases.
% "1" for rest, and  "0" for no rest.
WNorm           = vecnorm(W')';
WNorm           = moving_average_filter(WNorm,20);
A               = moving_average_filter(A,5);
WBinary         = WNorm < 0.95;
Begin_Rest      = find(diff(WBinary) == +1);
End_Rest        = find(diff(WBinary) == -1);
figure; plot(WNorm); hold on; plot(WBinary);
title('Velocidad angular normalizada - vector de detección de fases.');
xlabel('Time [centiseconds]');
ylabel('Angular velocity [rad/sec]');
```

El algoritmo en este punto logra detectar el inicio y finalización de fases del ciclo de la marcha a partir de la velocidad angular del dispositivo, para esto, cómo se ve en el fragmento de código anterior, se calculó la norma euclidiana **(WNorm)** de la velocidad angular presentada en los ejes *X, Y* y *Z* en cada terna de datos tomada y luego se aplicó un filtro de media móvil. Cuando la pierna se encuentra en fase de descanso, el valor de la velocidad angular normalizada es bajo, sabiendo esto, se creó un vector binario **(WBinary)** dónde cuando la velocidad angular normalizada sea menor a un valor de $0{,}95\ rad/s$ [1], en el vector binario se guardará un **1** y se guardará **0** en caso contrario. También cómo se observa en el fragmento de código, en las variables **(Begin_ rest)** y **(End_ rest)** se almacenó la posición del dato donde conmuta el vector binario, cuando el vector **(WBinary)** va de **1** a **0 (End_ rest)** indica que finaliza una fase de descanso o comienza la fase de balanceo, en el otro caso, cuando el vector **(WBinary)** va de un valor de **0** a **1**, **(Begin_ rest)** indica que inició una fase de descanso o lo que es igual, finalizó una fase de balanceo.

[1] Se fija dependiendo de las condiciones del experimento, por ésto se deben hallar experimentalmente en pruebas piloto.

- **Estimación de la orientación del dispositivo IMU:**

  En el siguiente fragmento de código se muestra la implementación del algoritmo para encontrar el cuaternion que describe las rotaciones del dispositivo en la marcha.

```
Q               = angle2quat(O(:,1),O(:,2),O(:,3),'ZYX');
```

  En éste punto, se aprovechó que los *smartphone* pueden entregar directamente la orientación del dispositivo a través de los ángulos de Euler (el teléfono por defecto tiene un marco de referencia fijo y mediante los ángulos *Azimuth, pith y roll*, entrega la posición angular del dispositivo en cada instante de tiempo a partir de ese sistema coordenado ya establecido). Luego, en Matlab se convirtió ésta matriz de orientación expresada en ángulos de Euler a una matriz de cuaterniones procedimiento que se documenta en [47] y se almacenó en la variable **Q**.

- **Rotación de la matriz de aceleración:**

  En el siguiente fragmento de código se muestra la implementación del algoritmo en donde se rotó la matriz de aceleración.

```
ACommon         = quatrotate(Q,A);
```

  En Matlab ya existen funciones para la realización de este procedimiento como se ve en el fragmento de código anterior, donde se rotó la matriz de aceleración a partir del cuaternion encontrado (procedimiento que se documenta en [47]). La aceleración ya rotada a partir del marco de referencia inicial se guardó en **ACommon**.

  A partir de aquí, en el algoritmo se segmenta la marcha y el análisis se realiza por zancada. En éste artículo se menciona que todos los cálculos se deben realizar por zancada para disminuir el error por *drift*.

  En el fragmento de código siguiente, se muestra como se encontró la incidencia de la gravedad en la aceleración del dispositivo y se eliminó posteriormente.

```
    %Direction of the gravity vector when the foot is at rest. Here is
    %important to clarify that the rest phase is a subphase of the contact
    %phase.
    ARest               = ACommon(Comienzo-20:Comienzo,:);
    AFoot_t0            = mean(ARest);

    %Calculation of the rotation matrix from the IMU's inertial reference
    %frame to ground frame. In the ground frame the Z-axis points in the
    %direction of the gravity.
    Z                   = AFoot_t0' / norm(AFoot_t0);
    Y                   = cross(Z,[1;0;0]);
```

```
Y                   = Y / norm(Y);
X                   = cross(Y,Z);
X                   = X / norm(X);
R_From_F_to_G       = [X,Y,Z]';
Q_From_F_to_G       = dcm2quat(R_From_F_to_G);

%Accelerations during the motion phase are projected to the ground
%frame.
AMotion             = ACommon(End_Rest(i):Begin_Rest(i+1),:);
AGround             = quatrotate(Q_From_F_to_G,AMotion);

%Gravity acceleration, which is obtained during the rest phase, is
%subtracted to the accelerations of the motion phase
AGravity            = quatrotate(Q_From_F_to_G,AFoot_t0);
AGround             = AGround - AGravity;
```

Debido a que el análisis se realiza por zancada, debe definirse un nuevo marco de referencia tomado de la fase de descanso, por esto, este marco de referencia se halló mediante la media de las aceleraciones en fase de descanso antes del inicio de la zancada analizada, dónde **(ARest)** y **(AFoot_t0)** son la aceleración en la fase de descanso y la media de dicha aceleración, respectivamente.

El nuevo marco de referencia se encontró, calculó y almacenó en la variable **(R_From_F_to_G)**, luego se convirtió a cuaternion y se almacenó en la variable **(Q_From_F_to_G)**. La aceleración en la fase de balanceo se almacenó en la variable **(AMotion)** y luego, se rotó teniendo en cuenta el marco de referencia establecido, este resultado se almacenó en la variable **(AGround)**.

- **Eliminación de la aceleración de la gravedad:** La eliminación de la aceleración de la gravedad se realizó teniendo en cuenta que ya toda la matriz aceleración ha sido rotada con respecto al marco de referencia establecido, sólo falta determinar cuál es la incidencia de la gravedad en el nuevo sistema coordenado, para esto, se rotó la media de aceleración en fase de descanso **(ARest)** con respecto al sistema coordenado establecido **(R_From_F_to_G)** y se almacenó en la variable **(AGravity)**, cómo último paso se restó **(AGravity)** de **(AGround)** para eliminar la incidencia de la gravedad en la matriz de aceleración durante la zancada.

- **Cálculo de la matriz de posición:**

  En el fragmento de código siguiente, se muestra la implementación del algoritmo donde se calculó la matriz de posición.

```
%Integration of the acceleration in the Ground frame.
T0                  = End_Rest(i)*Ts;
Tf                  = Begin_Rest(i+1)*Ts;
%Velocity at the end of the motion phase
C                   = Ts*trapz(AGround) / (Tf-T0);
AGround_No_Drift    = AGround - C;
%Velocity
VGround             = Ts*cumtrapz(AGround_No_Drift);

%Integration of the linear velocity in the Ground frame.
PGround(i,:)        = Ts*trapz(VGround);
```

Se calculó el tiempo de inicio y final de la zancada (tiempo de integración), se almacenaron en las variables **(T0)** y **(Tf)**, luego, se calculó la estimación del error por *drift* el cual se almacenó en la variable **(C)**, después se resta de **(AGround)** para encontrar la matriz de aceleración compensada. Se calculó la primer integración a la matriz de aceleración, como resultado se obtuvo la matriz de velocidad que se almacenó en **(VGround)**, luego se integró nuevamente para obtener la matriz de poscición del dispositivo, se presentó la ubicación de la pierna con respecto al marco de referencia correspondiente y se almacenó en la variable **(PGround)**.

- **Estimación de longitud de zancada:**

  En el siguiente fragmento de código, se muestra la implementación del algoritmo donde se estimó la longitud de cada una de las zancadas de la marcha.

```
Step_Length     = sqrt(PGround(:,1).^2 + PGround(:,2).^2);
```

Del paso anterior se obtuvieron las ubicaciones del pie en cada finalización de zancada con respecto al marco de referencia correspondiente, se tuvo en cuenta que el plano donde se presenta el movimiento de interés, es el plano **XY**, plano transversal del cuerpo. Luego, sólo fue necesario encontrar la distancia euclidiana entre los puntos de cada uno de los inicios y finales de zancada presentados en la marcha y se almacenan.

E.0.2. Algoritmo 2 tomado de: *Estimation of IMU and MARG orientation using a gradient descent algorithm* [2] .

Para la correcta implementación del algoritmo mencionado en este trabajo, la IMU debe colocarse en el dorso del pie como se muestra en la Figura E.4.

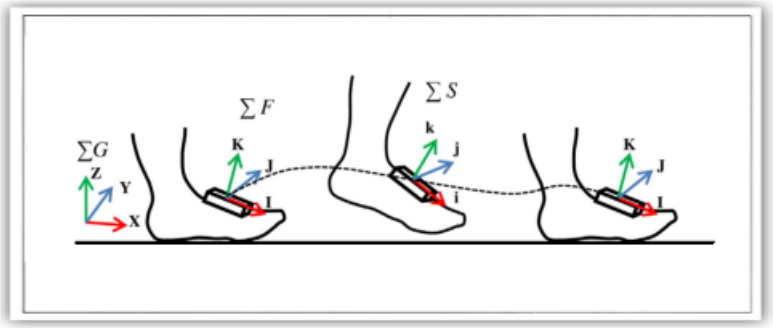


Figura E.4: Colocación de la unidad de medición inercial en el cuerpo [1].

- **Detección de las fases de movimiento de la pierna:**

  En la Figura E.5 se muestran las gráficas asociadas a la velocidad angular en **X**, **Y** y **Z** (colores, rojo, verde y azul, respectivamente), velocidad angular normalizada (línea negra punteada) y vector binario de detección de fase del ciclo de la marcha (color negro); Aceleración en **X**, **Y** y **Z** (colores, rojo, verde y azul, respectivamente).

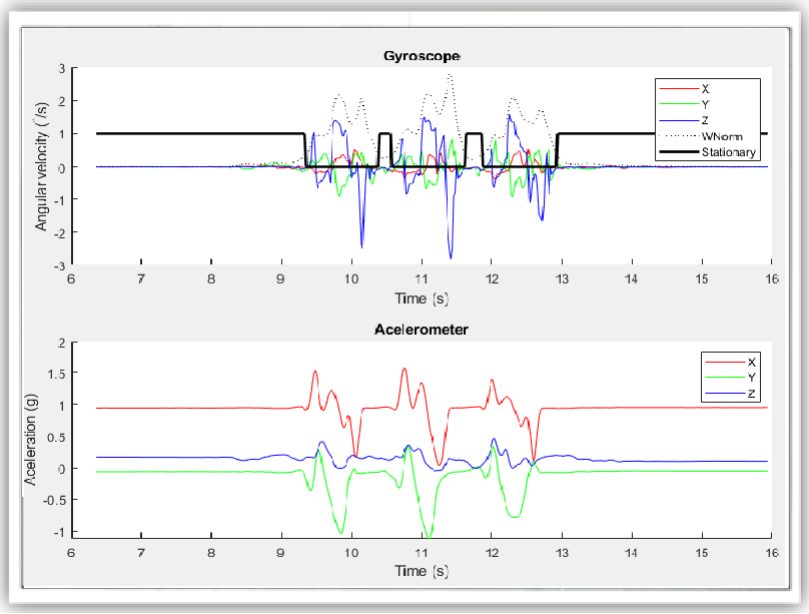


Figura E.5: Velocidad angular & normalizada - vector de detección de fase - Aceleración.

En el siguiente fragmento de código se muestra la implementación de la detección de fase de movimiento de la pierna.

```
%Detect stationary periods
%Compute angular velocity magnitude
gyr_mag = sqrt(gyrX.*gyrX + gyrY.*gyrY + gyrZ.*gyrZ);
%HP filter angular velocity data
filtCutOff = 0.001;
[b, a] = butter(1, (2*filtCutOff)/(1/samplePeriod), 'high');
gyr_magFilt = filtfilt(b, a, gyr_mag);
%Compute absolute value
gyr_magFilt = abs(gyr_magFilt);
%LP filter angular velocity data
filtCutOff = 5;
[b, a] = butter(1, (2*filtCutOff)/(1/samplePeriod), 'low');
gyr_magFilt = filtfilt(b, a, gyr_magFilt);
%Threshold detection
stationary = gyr_magFilt <35;

Begin_Rest      = find(diff(stationary) == +1);
End_Rest        = find(diff(stationary) == -1);
```

Para detectar las fases de la marcha en este algoritmo se usó los datos de la velocidad angular normalizada, para esto, se encuentró la norma euclidiana **(gyr_mag)** de la velocidad angular dada sobre los ejes **X, Y** y **Z** en cada terna de datos tomada, luego se filtró y almacenó en **(gyr_magFilt)**. Cuando la pierna se encuentra en fase de descanso, el valor de la velocidad angular normalizada es bajo, sabiendo esto, se halló un vector binario **(stationary)**, dónde cuando la velocidad angular normalizada sea menor a un valor de $35°/s$ [2], en el vector se guardará un **1** y se guardará **0** en caso contrario. También cómo se observó en el fragmento de código, en las variables **(Begin_ rest)** y **(End_ rest)** se guardó la posición del dato donde conmuta el vector binario, cuando el vector **(stationary)** va de **1** a **0**, **(End_ rest)** indica que finaliza una fase de descanso o comienza la fase de balanceo, en el otro caso, cuando el vector **(stationary)** va de un valor de **0** a **1**, **(Begin_ rest)** indica que inició una fase de descanso o lo que es igual, finalizó una fase de balanceo.

- **Estimación de la orientación del dispositivo IMU:**

  En el siguiente fragmento de código, se muestra la implementación del algortimo de gradiente descendente mediante el cual se encontró el cuaternion que describe la orientación del dispositivo en la pierna durante la marcha.

```
        function obj = UpdateIMU(obj, Gyroscope, Accelerometer)
            %Normalise accelerometer measurement
            if(norm(Accelerometer) == 0)                                    %handle NaN
                warning(0, 'Accelerometer magnitude is zero.  Algorithm update aborted.');
                return;
            else
                Accelerometer = Accelerometer / norm(Accelerometer); %normalise measurement
            end
            %Compute error between estimated and measured direction of gravity
            v = [2*(obj.q(2)*obj.q(4) - obj.q(1)*obj.q(3))
                2*(obj.q(1)*obj.q(2) + obj.q(3)*obj.q(4))
                obj.q(1)^2 - obj.q(2)^2 - obj.q(3)^2 + obj.q(4)^2]; %estimated direction of gravity
```

[2] Se fija dependiendo de las condiciones del experimento, por ésto se deben hallar experimentalmente en pruebas piloto.

```
            error = cross(v, Accelerometer');
            obj.IntError = obj.IntError + error; %compute integral feedback terms
                                                 ...(only outside of init period)
            Ref = Gyroscope - (obj.Kp*error + obj.Ki*obj.IntError)';
            %Compute rate of change of quaternion
            pDot = 0.5 * obj.quaternProd(obj.q, [0 Ref(1) Ref(2) Ref(3)]);%ompute rate of change of ↩
                quaternion
            obj.q = obj.q + pDot * obj.SamplePeriod; %integrate rate of change of quaternion
            obj.q = obj.q / norm(obj.q);             %normalise quaternion
            %Store conjugate
            obj.Quaternion = obj.quaternConj(obj.q);
        end
```

Para encontrar la ortientación del dispositivo durante la marcha, el algoritmo utilizó la fusión de sensores de aceleración y velocidad angular, estos datos pasaron a través del algoritmo de gradiente descendente (*gradient descent algorithm*) con el cual se encontró la matriz de orientación del dispositivo expresada en cuaternion **(quat)**, procedimiento que se decribe en [2].

- **Rotación de la matriz de aceleración:**

  En el siguiente fragmento de código, se muestra la implementación del algoritmo donde se rotó la matriz de aceleración.

```
%Compute translational accelerations
%Rotate body accelerations to Earth frame
acc = quaternRotate([accX accY accZ], quaternConj(quat));
%Convert acceleration measurements to m/s/s
acc = acc * 9.81 ;
%%Remove gravity from measurements
acc(:,3) = acc(:,3) - 9.81;
```

En el fragmento de código mostrado, se rotó la matriz de aceleración a partir del cuaternion encontrado en el paso anterior y se almacenó en **(acc)**, también se convirtió esta a metros sobre segundo al cuadrado *[m/s/s]* ya que se encontraba expresada en gravedad *[g]*. Finalmente se sustrajo la aceleración de la gravedad sobre el eje que tiene incidencia directa, que al haber rotado la matriz de aceleración, sólo tiene incidencia directa sobre el eje **Z**.

- **Cálculo de la matriz de posición:**

  En el siguiente fragmento de código, se muestra la implementación del algoritmo donde se obtuvo la matriz de posición.

```
%Integrate acceleration to yield velocity
vel = zeros(size(acc));
for t = 2:length(vel)
    vel(t,:) = vel(t-1,:) + acc(t,:) * samplePeriod;
    if(stationary(t) == 1)
        vel(t,:) = [0 0 0];     %force zero velocity when foot stationary
    end
end
%Compute integral drift during non-stationary periods
velDrift = zeros(size(vel));
stationaryStart = find([0; diff(stationary)] == -1);
stationaryEnd = find([0; diff(stationary)] == 1);
for i = 1:numel(stationaryEnd)
    driftRate = vel(stationaryEnd(i)-1, :) / (stationaryEnd(i) - stationaryStart(i));
    enum = 1:(stationaryEnd(i) - stationaryStart(i));
    drift = [enum'*driftRate(1) enum'*driftRate(2) enum'*driftRate(3)];
    velDrift(stationaryStart(i):stationaryEnd(i)-1, :) = drift;
end
%Remove integral drift
vel = vel - velDrift;

%Compute translational position

%Integrate velocity to yield position
pos = zeros(size(vel));
for t = 2:length(pos)
    pos(t,:) = pos(t-1,:) + vel(t,:) * samplePeriod;    %integrate velocity to yield position
end
```

Se realizó la primer integración a la matriz de aceleración **(acc)**, como resultado se obtuvo la matriz de velocidad **(vel)** y se forzó a que cuando en pie esté en fase de descanso la velocidad sea cero. En ésta matriz se debió hacer compensación del error por *drift* durante las fases de balaceo de la pierna, para esto se calculó el error por *drift* el cual se almacenó en **(velDrift)** y se restó de la matriz de velocidad **(vel)**. Luego se integró nuevamente para obtener la matriz de poscición **(pos)** del dispositivo, esta representa la trayectoria completa del pie durante la marcha.

- **Estimación de longitud de zancada:** En el siguiente fragmento de código, se muestra la implementación del algortimo donde se estimó la longitud de cada una de las zancadas de la marcha.

```
posi=zeros(length(Begin_Rest),3);

for K=1:length(Begin_Rest)
    if K==1
    posi(K,:)=pos(Begin_Rest(K),:);
    elseif K>1
     posi(K,:)=pos(Begin_Rest(K),:)-pos(Begin_Rest(K-1),:);
    end
end

Step_Length     = sqrt(posi(:,1).^2 + posi(:,2).^2);
```

Teniendo la matriz de la trayectoria del pie, dada sobre los ejes **X, Y** y **Z**, se debió tener en

cuenta el plano en el cual se dió el movimiento de interés, es decir, dónde se presentó la zancada y es posible medir su longitud, para éste caso se pudo observar que el movimiento se da sobre el plano ***XY***, plano transversal del cuerpo. Luego, sólo fue necesario encontrar la distancia euclidiana entre los puntos de cada uno de los inicios y finales de zancada presentados en la marcha.

E.0.3. Algoritmo 3 tomado de: *Walking speed estimation using ashank-mounted inertial measurement unit.* [3]

Para la correcta implementación del algoritmo mencionado en este trabajo, la IMU debe colocarse a media pantorrilla de la pierna como se muestra en la Figura E.6.

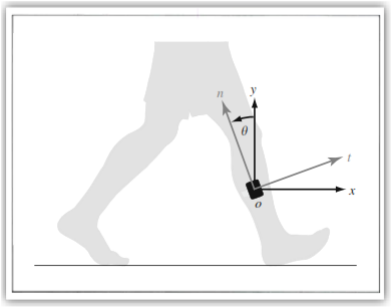


Figura E.6: Colocación de la unidad de medición inercial en el cuerpo [3].

- **Detección de las fases de movimiento de la pierna:**

  En la Figura E.7 se observa la velocidad angular filtrada (azul) y la identificación de los picos más cercanos a cero (círculos rojos).

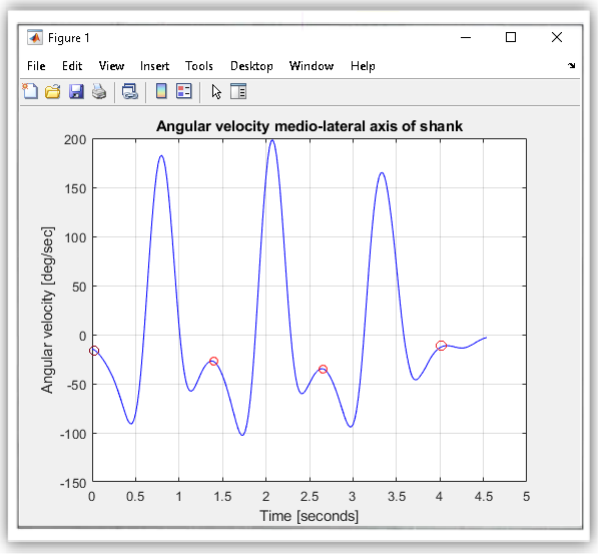


Figura E.7: Velocidad angular - Picos de detección de posición media de la pierna.

El siguiente fragmento de código, muestra la implementación del algoritmo donde se dió la detección de subfase de posición media la pierna.

```
time    = (1:length(A))' * (Ts);
angular_vel_z       = moving_average_filter(W(:,3),10);
%Busqueda de máximos locales negativos cuya prominencia sea mayor a 10
%grados/segundo.
threshold           = deg2rad(10);
[peak,location]     = findpeaks(angular_vel_z,'MinPeakProminence',threshold);
%Elimina picos con velocidad angular mayor que 10 grados/seg. Con esto se
%asegura que solo se encuentren los picos negativos.
location            = location(peak < threshold);
peak                = peak(peak < threshold);
```

En este algoritmo se estimó la trayectoria de la zancada con respecto al plano sagital del cuerpo humano, es decir, sólo se usaron dos datos de aceleración, sobre los ejes ***X*** e ***Y*** y velocidad angular alrededor del eje ortogonal al plano, es decir, alrededor de ***Z***. El dispositivo IMU estaba ubicado a media pantorrilla y para detectar las fases de movimiento de la zancada en la marcha se usó sólo la velocidad angular, se tiene presente que en la subfase de posición media en la fase de descanso *(ver figura 2.4)*, la velocidad angular tiende a cero.

Cómo se observó en el fragmento de código, se detectaron los picos con la función *findpeaks*, los picos de interés deben tener velocidad angular cercana a cero, ya que estos corresponden a la subfase de posición media de la pierna y por esto, entre cada pico se pudo estimar la longitud de la zancada.

- **Estimación de la orientación del dispositivo IMU, rotación de la matriz de aceleración y eliminación de la aceleración de la gravedad:**

  En el siguiente fragmento de código, se muestra la implementación del algoritmo donde se

rotaron las aceleraciones en los ejes ***X*** e ***Y***.

```
function [ax,ay] = from_sensor_to_world(th,accel_tangential,accel_normal)
%Gravedad
g                   = 9.81;

%Ecuación (1) de la referencia [1].
ax                  = zeros(size(th));
ay                  = zeros(size(th));
for i = 1:length(th)
    %Matriz de rotación
    r       = [cos(th(i)), -sin(th(i)); sin(th(i)),  cos(th(i))];

    %Transformación de aceleraciones
    axy     = r * [accel_tangential(i); accel_normal(i)] - [0; g];

    %Separación de las aceleraciones en los componentes X y Y
    ax(i)   = axy(1);
    ay(i)   = axy(2);
end
end
```

Como se mencionó, el análisis de cada zancada se hizo por separado entre los datos correspondientes a los que se encuentran entre dos subfases de posición media consecutivos de la misma pierna. Dicho análisis se realizó mediante la implementación de las ecuaciones mencionadas en [3] y permitieron encontrar la matriz de orientación **(r)** mediante la cual se rotó la matriz de aceleración a partir de la posición angular en el eje ortogonal al plano, ésta fue encontrada integrando los datos obtenidos por el giroscópio, a demás, se eliminó la aceleración de la gravedad que tiene incidencia directa sobre el eje **Y** restando ésta aceleración de la aceleración del eje mencionado procedimiento que se almacenó en **(axy)** dando como resultado las aceleraciones desde el marco de referencia inicial, las cuales se almacenaron en **(ax)** y **(ay)**.

- **Cálculo de la matriz de posición y estimación de longitud de zancada:**

  En el siguiente fragmento de código, se muestra la implementación del algoritmo donde se obtuvo la matriz de posición y longitud de zancada.

```
    %Se convierten las aceleraciones al sistema coordenado de referencia.
    %Ecuación (1) de [1]
    [ax,ay]             = from_sensor_to_world(theta,accel_tangential,accel_normal);
    %Se integran las aceleraciones "ax" y "ay"
    vx                  = cumsum(ax) * Ts;
    vy                  = cumsum(ay) * Ts;
    %Se integran las velocidades "vx" y "vy"
    sx                  = cumsum(vx) * Ts;
    sy                  = cumsum(vy) * Ts;
    %Se resta el factor 0.5 * tstride * vx(T)
    tstride             = (location(1) - 1) * Ts;
    sx                  = sx - 1/2 * tstride * vx(end);
    sy                  = sy - 1/2 * tstride * vy(end);
    %Longitud de zancada
    stride_length(nn)    = sqrt(sx(end)^2 + sy(end)^2);
```

Habiendo rotado ya las matrices de aceleraciones se obtuvieron ***(ax)*** y ***(ay)*** que corresponden a las aceleraciones en el eje de progresión de la marcha y el eje normal, respectivamente, se realizó la primer integración a la matriz de aceleración, como resultado se obtuvieron los vectores de velocidad del pie **(vx)** y **(vy)**, luego se integró nuevamente para obtener los vectores de posición del pie **(sx)** y **(sy)**, estos se corrigieron para reducir errores de estimación de desplazamiento asociados con las compensaciones de aceleración *(offset)* o error por *drift*. Como resultado se obtuvo la trayectoria de la pierna en cada uno de los ejes mencionados. Para estimar la longitud de zancada se calculó la distancia euclidiana entre el punto inicial y final de la zancada que corresponden al espacio entre las dos subfases de posición media de la zancada.

Anexo F

Implementación del código *promedioZancadasGen.m*.

En éste anexo se explicará a detalle la forma en que se calculó el promedio de zancada por marcha para cada participante, y también la manera en que se realizó el comparativo con el estadístico *Kruskall-Wallis*.

A continuación se presenta el código *promedioZancadasGen.m* paso a paso de la forma en que se implementó.

F.1. Promedios de zancada por cada marcha

1. Carga de datos procesados en archivo *LongZancadas.mat* y separación de variables. A continuación se presenta la primera parte del código.

```
%CALCULOS ESTADISTICOS PARA TRABAJO DE GRADO: "ANALISIS COMPARATIVOS DE
%ALGORITMOS DE ESTIMACION DE LA LONGITUD DE ZANCADA EN MARCHA HUMANA
%UTILIZANDO SISTEMAS INERCIALES DE MEDIDA"
%
%UNIVERSIDAD DEL CAUCA - POPAYAN - 2021
%
%ENMANUEL BERRUECOS GOMEZ
%MARLIO ALEJADNRO CHICUE RESTRPO


clear all;
close all;
clc;


%Carga de datos
LongZancadas=load('LongZancadas.mat');    %Datos


%Separacion de datos
LongZ=LongZancadas.LongZ;
LongZ1=LongZancadas.LongZ1;
LongZ2=LongZancadas.LongZ2;
LongZ3=LongZancadas.LongZ3;
```

En la sección de código 1 se muestra la carga de datos del archivo *LongZancadas.mat* en una variable local, y la separación de cada algoritmo en diferentes variables

$$LongZ_n$$

donde

$$n = 0, 1, 2, 3.$$

2. Cálculo del promedio de zancada por cada marcha. A continuación se presenta la segunda parte del código.

```
%Calculo del promedio de zancada por persona

for i=1:3

    for n=1:10

            Z1(n,:,i)= (LongZ(n,1,i) + LongZ(n,2,i) + LongZ(n,3,i))/3;     %Vision
            Z2(n,:,i)= (LongZ1(n,1,i) + LongZ1(n,2,i) + LongZ1(n,3,i))/3; %Alg1
            Z3(n,:,i)= (LongZ2(n,1,i) + LongZ2(n,2,i) + LongZ2(n,3,i))/3; %Alg2
            Z4(n,:,i)= (LongZ3(n,1,i) + LongZ3(n,2,i) + LongZ3(n,3,i))/3; %Alg3

    end
end
```

En la sección de código 2 se muestra el cálculo del promedio por cada marcha mediante dos ciclos *for*.

F.2. Implementación del estadístico *Kruskall-Wallis*

3. Unificación de datos por algoritmos y cálculos estadísticos. A continuación se presenta la tercera parte del código

```
%Unificacion de datos por algoritmos
Vision=[Z1(:,:,1);Z1(:,:,2);Z1(:,:,3)];
IMU1=[Z2(:,:,1);Z2(:,:,2);Z2(:,:,3)];
IMU2=[Z3(:,:,1);Z3(:,:,2);Z3(:,:,3)];
IMU3=[Z4(:,:,1);Z4(:,:,2);Z4(:,:,3)];

ZT= [Vision, IMU1, IMU2, IMU3];

%Calculos estadisticos
close all;
ropes={'AlgVision' 'AlgIMU1' 'AlgIMU2' 'AlgIMU3'};

[p,tbl,stats] = kruskalwallis(ZT,ropes);      %Calcular datos
figure,
c=multcompare(stats);                         %Obtener figuras
```

En la sección de código 3 se muestra la unificación de datos por algoritmos. Las variables Z_n donde, $n = 1, 2, 3, 4.$; almacenan los datos de cada participante en cada algoritmo respectivo; luego, estas variables son pasadas a otras variables con el nombre respectivo de cada algoritmo. Y finalmente se almacenan todos los algoritmos en la variable $ZT = [Vision, IMU1, IMU2, IMU3]$. Posteriormente se realiza el cálculo de la parte estadística, donde se evaluaron los datos de cada participante sometiendo la variable ZT (que contiene los promedios de zancada por macha), como argumento a través de la función **[p,tbl,stats] = kruskalwallis(ZT,ropes)** la cual devuelve como resultado el valor *p* de la hipótesis nula, un diagrama de cajas y bigotes con la mediana de cada grupo de datos, y unos valores estadísticos de relación entre los diferentes algoritmos para determinar si pertenecen o no a una misma población. Los cuales son graficados mediante la función **multcompare(stats)** a la cual se le pasan los datos contenidos en la variable **stats**. Como resultado de esta operación se obtienen 3 gráficos generales con el resultado de los 4 algoritmos, en donde se muestran los datos estadísticos necesarios para su comparación y análisis.