UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

Facultad 2

Implementación de algoritmos para la estimación de la orientación en Vehículos Autónomos Subacuáticos

Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas.

Autores: Liliet de la Caridad González Polanco

Hansel Sanabia Padrón

Tutores: MSc. Mirta Bertrandez Sardiñaz

Ing. Darvis Dorvigny Dorvigny

Ing. Lester González López

“(…) *Cuando se quiere vencer, cuando existe la voluntad de vencer, se vence. No hay obstáculos, no hay dificultades que puedan interponerse a la voluntad indoblegable de los hombres* (…)” *

Fidel Castro Ruz

*Discurso pronunciado en el acto central por el XXX Aniversario de los CDR, el 28 de septiembre de 1990. *Granma,* 1ro de octubre de 1990, página 3.

DECLARACIÓN DE AUTORÍA

Declaramos que somos los únicos autores del trabajo para optar por el título de Ingenieros en Ciencias Informáticas titulado “Implementación de algoritmos para estimación de orientación en Vehículos Autónomos Subacuáticos” y reconocemos a la Universidad de las Ciencias Informáticas y la Facultad # 2 los derechos patrimoniales del mismo con carácter exclusivo.

Para que así conste firmamos el presente a los ___días del mes _____ del año 2016.

_____	_____
Firma del Autor	Firma del Autor
Liliet de la Caridad González Polanco	Hansel Sanabia Padrón
_____	_____
Firma del Tutor	Firma del Tutor
Ing. Darvis Dorvigny Dorvigny	Ing. Lester González López.

_____

Firma del Tutor

MSc. Mirta Beltrández Sardiñas

❖ **Agradecimientos**

Debo agradecer en primerísimo lugar a mi siempre líder Fidel Castro Ruz, por ser mi guía y enseñarme lo que son los principios morales que enaltecen a los hombres.

A mi Revolución Cubana y el Partido Comunista de Cuba, por permitirme, aun siendo hija de obreros con pocos recursos, el haber podido estudiar en una universidad tan importante como lo es la UCI y tener acceso a las tecnologías a pesar de que no pudiera costearme la carrera.

Sería ingrato de mi parte no agradecer eternamente a la propia universidad que me ha formado como profesional y joven comunista porque ha puesto a mi servicio a profesores preocupados, dedicados a su profesión y ejemplo en todo momento. Gracias UCI: me has brindado un claustro de lujo.

Agradeceré en todo momento a mis tutores Mirta y Lester por la preocupación y seriedad con que han respaldado mi trabajo, así como la paciencia y amabilidad con la que han sabido responder a todas a mis preguntas y dudas. Al profe Darvis, en primer lugar, por haberme seleccionado y confiar desde un primer momento en mí para desarrollar una investigación con el nivel de complejidad como la que se presenta; pero más que nada por el apoyo incondicional que he sentido me ha brindado aun cuando la distancia nos dificulta la comunicación, aun así, he sabido que he podido contar con él en todo momento. Gracias a mis tutores han hecho un trabajo excepcional, por tanto tiempo que me han dedicado los considero, más que tutores, compañeros de tesis.

De la misma forma debo agradecer a Hansel, porque ha posibilitado que todo el trabajo sea menos complicado y también por su sacrificio por lograr juntos este sueño. La verdad hemos hecho un buen equipo.

Y por último debo agradecer a la razón fundamental por la que me he esforzado 5 años de carrera: mi familia: mi mamá Magalys, mi papá Javier, Mi hermana Liset, mi cuñado Yadian, mi novio Leonardo, porque han hecho a un lado todas las cosas, incluidas aquellas muy importantes, para darme a mí la prioridad en todos los aspectos materiales y espirituales, porque son los que sin importar nada han estado ahí para apoyarme, cuidarme, comprenderme, darme la mano cuando he caído y ayudarme a levantar.

Muchas gracias a los buenos amigos que me acompañan, me han ayudado a ser feliz estos años de carrera.

Liliet de la Caridad González Polanco.

❖ **Agradecimientos**

Gracias a ti Darvis, a ti Mirta, a ti Lester, a ti UCI y a todos aquellos que me ayudarían con gusto pero el momento no los hizo necesarios. Gracias a Liliet. Gracias a ti que lees esto.

Hansel Sanabia Padrón

❖ **Dedicatoria**

A mi familia, a Fidel Castro Ruz y mi Cuba bella.

Liliet de la Caridad González Polanco

❖ **Dedicatoria.**

Está dedicado a ti mami. Es para ti mi amor, seamos eternos. Está dedicado a ustedes, sangre de mi sangre, hermanos. Está dedicado a esos que no dudarán ni un segundo en pasar conmigo los momentos más críticos de mi vida. Todos esos son mi familia y mi familia son todos esos.

RESUMEN

Con la creciente demanda y evolución de los Vehículos Autónomos Subacuáticos se hace cada vez más imperante para Cuba adentrarse en el conocimiento sobre este tipo de tecnología, a pesar de su alto costo en el mercado. Estas tecnologías para que realicen las tareas para las que son programadas es necesario que estimen la orientación.

Esto a su vez es posible a través de las estimaciones de las mediciones de los sensores: giróscopos, acelerómetros y magnetómetros los cuales suelen tener precios elevados en el mercado, razón por la cual es necesario utilizar los de bajo costo, aunque por su parte, estos tienen marcadas desventajas.

En la presente investigación se implementan tres algoritmos de estimación de orientación, con el fin de evaluar su eficacia ante mediciones realizadas con sensores de bajo costo. Los algoritmos siguen las dos tendencias principales, la basada en Filtro de Kalman, y la basada en filtro complementario. Se realizaron pruebas estadísticas para evaluar los resultados, de lo que resulta que es el Filtro Complementario propuesto por Valenti (2015) el que mejor estimación de la orientación realiza con sensores de bajo costo.

Palabras claves: Estimación de Orientación, Filtro Extendido de Kalman, Filtro Complementario, Vehículos Autónomos Subacuáticos

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE PRUEBAS

INTRODUCCIÓN

Los grandes avances que han ocurrido en el área de la Robótica e Informática han impulsado la evolución de los Vehículos Autónomos Subacuáticos (*Autonomous Underwater Vehicles (**AUVs**))* hasta niveles cada vez mayores de sofisticación. Han superado la fase experimental y han sido incorporados por las principales marinas del mundo como plataformas de todo tipo de sensores submarinos. El aumento de su fiabilidad y prestaciones, a la par de la reducción de coste, los sitúan al alcance de la mayor parte de las marinas de guerra. (Sesto, 2009)

Los Vehículos Autónomos Subacuáticos, son robots submarinos con capacidad de movimiento y desarrollo de misiones bajo el agua sin que sean comandados directamente por una persona u operador, por tanto, estos disponen de autonomía energética e inteligencia suficiente para llevar a cabo las tareas programadas. (Guerrero González, López Maestre, Gilabert Cervera, García-Vidal Simón, & González Reloid, 2010). También los Vehículos Autónomos Subacuáticos extraen información de su entorno empleando una amplia variedad de sensores, y luego emplea dicha información para tomar decisiones sobre la navegación (García, 2011) es por ello las amplias aplicaciones a nivel mundial de los mismos.

En Cuba existen el Centro de Investigación y Desarrollo Naval (CIDNAV) y el Grupo de Automatización, Robótica y Percepción de la Universidad Central “Marta Abreu” de Las Villas, estas entidades han desarrollado múltiples investigaciones sobre esta tecnología. Los trabajos más recientes se realizan en el modelado dinámico de los vehículos para el guiado y la navegación. La Universidad de las Ciencias Informáticas ha comenzado a incursionar en el tema de los Vehículos Autónomos fundamentalmente en la implementación de algoritmos para la navegación integrada INS/GPS en AUVs.

Especial interés tiene Cuba por adentrarse en el conocimiento de estos equipos por la importancia que representa para la economía y defensa del país (Navarro, 2014) pero este tipo de tecnología requiere gran cantidad de recursos, por lo que los países desarrollados son los que mayormente tienen el liderazgo en la utilización de estas plataformas, a su vez, son vendidas a muy altos precios a los países interesados en su explotación. (Fossen, 2011)

Para el propósito de la navegación, la estimación precisa de la orientación de un cuerpo rígido ha empleado sensores inerciales y magnéticos de alta precisión, pero el reciente desarrollo de sistemas micro-electro-mecánicos (MEMS) de bajo costo y peso ligero ha permitido adoptar sensores inerciales más pequeños y más baratos para una gama más amplia de aplicaciones e incluso en el uso diario de la electrónica de consumo, tales como consolas de juegos y dispositivos móviles. (Valenti, Dryanovski, & Xiao, 2015)

Los AUVs, para el cumplimiento de misiones, requieren de información sobre los parámetros de navegación en cada instante de muestreo. Estos parámetros son la orientación, expresada como una secuencia de rotaciones; y la posición, expresada generalmente en latitud, longitud y altura (Weston & Titterton, 2004). Existe dependencia entre estos parámetros. No es posible determinar con precisión la posición del vehículo si no se conoce la orientación en el instante de muestreo para la conversión de las mediciones inerciales de sensores de bajo costo, de un marco de referencia a otro.

Asegurar la correcta estimación de la orientación juega un papel fundamental en un sinnúmero de áreas tales como: el espacio aéreo, la robótica, la navegación, el análisis motor humano y la interacción con las máquinas (Madgwick, 2010). Por ello, un problema crítico para la navegación es la estimación de la orientación, ya que no se dispone de un método que garantice la precisión necesaria para estimar posteriormente la posición del vehículo.

Uno de los principales instrumentos de navegación en los Vehículos Autónomos Subacuáticos es la unidad de medición inercial (IMU, por sus siglas en inglés), la cual al adicionarse un magnetómetro se le conoce también como “Sistema de medición de gravedad, velocidad angular y magnetismo” (MARG, por sus siglas en inglés). La información que proporcionan los sensores que componen un MARG brinda la posibilidad de obtener la orientación y rumbo de cualquier vehículo, pero estos sensores suelen ser muy susceptibles al ruido y a las perturbaciones. (Rodriguez & Chérigo, 2016)

Una IMU está compuesta por acelerómetros y giróscopos de forma que habilita el seguimiento de movimientos rotacionales y transitorios en orden para medir en tres dimensiones. Solo con instrumentos inerciales se puede estimar con determinada precisión la orientación a la dirección de la gravedad, que es suficiente para muchas aplicaciones. Los sistemas MARG, también conocidos como AHRS (*Attitude and Heading Reference Systems*) proveen una medición completa de la orientación relativa a la dirección de la gravedad y el campo magnético de la Tierra. (Madgwick, 2010)

Los acelerómetros y giróscopos son los llamados sensores inerciales, dado que miden aceleración y velocidad angular, respectivamente. Conocidas las condiciones iniciales, la doble integración de la aceleración permite conocer la posición; la integración de la velocidad angular proporciona la orientación del vehículo. Los magnetómetros tienen la capacidad de medir la intensidad del campo magnético local. (Rodriguez & Chérigo, 2016)

Es válido mencionar que los sensores que permiten esta estimación de la orientación son sensores que se caracterizan por tener precios elevados para países subdesarrollados, y la mayoría, por carácter táctico o estratégico, simplemente no están disponibles en el mercado. Por otra parte, existe una gama de sensores para la navegación de carácter comercial, de bajo costo, que son más accesibles y se caracterizan por ser poco fiables, ruidosos, imprecisos, de mala calidad, con altos niveles de sesgo, tanto dinámico como estático, que empeoran el cálculo de los parámetros de navegación en cuanto a precisión y exactitud.

También, los algoritmos usados para la mejora de la medición, que incorporan los sensores de medición inercial, implementan formulaciones matemáticas del álgebra de cuaterniones, álgebra lineal, procesos estocásticos y estadística inferencial, lo cual aumenta el costo computacional y el tiempo de desarrollo de los proyectos. (Rodriguez & Chérigo, 2016)

Las soluciones de navegación con estos sensores implican un costo enorme en precisión y exactitud en la estimación. A esto se agregan otros criterios de la verificación empírica, que permitieron obtener un grupo de insuficiencias y quedaron formuladas a manera de situación **problemática** de la siguiente forma:

- Cuba no cuenta con los recursos financieros necesarios para invertir en sensores de alta calidad para controlar la orientación de un AUV, teniendo que acceder a los de bajo de costo.
- Los sensores de bajo costo que pueden utilizarse para el control de la orientación, se caracterizan por ser poco fiables, ruidosos, imprecisos, de mala calidad y con altos niveles de sesgo.
- El desempeño de los algoritmos está condicionado por las características de los sensores utilizados, por lo que, a mayor precisión de los sensores, mayor precisión en los resultados que arrojen los algoritmos.
- No es posible acceder a los algoritmos implementados en los sensores de alto costo por ser propietarios y de código cerrado.

A partir de la situación problemática se identifica como **problema a resolver:** ¿Cómo determinar la orientación de un Vehículo Autónomo Subacuático con las mediciones de sensores de bajo costo?

En consecuencia, con lo anterior se determinó como **objeto de estudio**: Los vehículos Autónomos Subacuáticos.

Además, se identifica como **campo de acción:** Estimación de la Orientación en Vehículos Autónomos Subacuáticos.

Para darle solución al problema planteado se define el siguiente **objetivo general**: Implementar tres algoritmos con las mediciones de sensores de bajo costo, para estimar la orientación de un Vehículo Autónomo Subacuático.

Para darle cumplimiento al objetivo general se proponen los siguientes **objetivos específicos:**

- Fundamentar los referentes teóricos sobre los algoritmos para estimación de orientación de un Vehículo Autónomo Subacuático.
- Implementar algoritmos que permitan la correcta estimación de la orientación con sensores de bajo costo.
- Validar la funcionalidad de los algoritmos para la estimación de orientación con sensores de bajo costo.
- Comparar los algoritmos implementados y seleccionar el que mejor estimación de la orientación proporcione.

A partir de formular los objetivos planteados se trazaron las siguientes **tareas de investigación:**

- Fundamentación teórica sobre los algoritmos que permiten la estimación de la orientación en Vehículos Autónomos Subacuáticos.
- Caracterización de la cinemática de la rotación de un cuerpo en el espacio.
- Caracterización de la representación y estimación de la orientación para la navegación.
- Caracterización de los algoritmos previstos para la propuesta de solución.
- Implementación de los algoritmos para la estimación de la orientación con sensores de bajo costo.

- Construcción de escenario de prueba con datos reals para evaluar el desempeño de los algoritmos de estimación de orientación con respecto a los datos registrados como referencia.
- Validación por el método estadístico de los resultados de los algoritmos.

Los métodos de investigación realizados se describen a continuación:

Métodos teóricos:

- **Análisis–Síntesis:** posibilitó esencialmente la obtención de información teórica acerca de los conceptos sobre: Vehículos Autónomos Subacuáticos, representación de la orientación, cinemática de la rotación de un cuerpo en el espacio, algoritmos con mejores resultados en la estimación de la orientación de Vehículos autónomos.

- **Modelación:** permitió seguir la lógica interna de los algoritmos que utilizan sensores de bajo costo para estimar la orientación de un Vehículo Autónomo Subacuático.

Métodos Empíricos:

- **Análisis Documental**: permitió el análisis de los contenidos en los documentos disponibles en Internet y las bibliografías consultadas sobre los Vehículos de Autónomos Subacuáticos, representación de la orientación, sensores y cinemática de la rotación de los cuerpos en el espacio.
- **Medición**: permitió evaluar los resultados, así como la aceptación de la implementación de los algoritmos garantizando la selección del algoritmo que mejor estimación de la orientación representa con sensores de bajo costo.
- El **método estadístico** utilizado viabilizó la selección de la muestra, el procesamiento de datos y la validación de la propuesta de solución.

Estructura del documento

El presente documento está organizado en tres capítulos, a continuación, se muestra una breve descripción de cada uno de ellos.

Capítulo 1. Estudio y fundamentación de los referentes teóricos sobre el cálculo de la orientación en Vehículos Autónomos Subacuáticos, cinemática de la rotación de un cuerpo en el espacio, sensores, representación de la orientación, estado actual de los AUVs y herramientas y leguajes de programación utilizados.

Capítulo 2. Se caracterizan e implementan los algoritmos seleccionados para la estimación de la orientación con sensores de bajo costo, como propuesta de solución al problema de la investigación.

Capítulo 3. Se realizan las pruebas y se validan los resultados obtenidos en la implementación de los algoritmos para la estimación de orientación con sensores de bajo costo en Vehículos Autónomos Subacuáticos.

CAPÍTULO 1: FUNDAMENTOS TEÓRICOS PARA ESTIMAR ORIENTACIÓN EN VEHÍCULOS AUTÓNOMOS SUBACUÁTICOS.

1.1 Introducción

En el presente capítulo se abordan los elementos teóricos necesarios para dar solución a la situación problemática. Comprende los principales conceptos que serán tratados durante el desarrollo de la investigación. Se muestra una caracterización de las técnicas y tecnologías propuestas para el desarrollo del sistema.

1.2 Antecedentes y características de los AUVs en el contexto tecnológico

Los AUVs tienen sus orígenes en los Vehículos Operados Remotamente (*Remotely Operated Vehicles).* Fueron los primeros vehículos de este tipo, cuyo desarrollo se inició en los años 60 aunque existe la referencia del robot subacuático operado remotamente nombrado POODLE, desarrollado por Dimitri Rebikoff, en Francia, en la década del 50 del siglo pasado (Moreno, 2014). Estos eran controlados por operadores humanos a través de un cable que une al vehículo con su plataforma de control.

El desempeño de estos vehículos permitió una mayor autonomía de operación. Las profundidades en que pueden ser utilizados varían, según los materiales con que fueron diseñados y la longitud del cable. Sin embargo, un modelo en particular superó todas las capacidades de operación consideradas actualmente, el *Víctor-6000* construido por *Ifremer.* (Valencia Niño & Dutra, 2010)

Inmediatamente estos mini-submarinos encontraron importantes clientes en la industria del petróleo y el gas, de forma que son imprescindibles para la inspección, mantenimiento y reparación de cables, tuberías y otras estructuras subacuáticas. (Sesto, 2009)

En la esfera militar, diversas marinas los utilizan también para la caza de minas o la inspección de cascos de buques en busca de posibles explosivos adosados. Otras importantes aplicaciones son el salvamento marítimo, la hidrografía o la arqueología submarina, solo por citar algunos ejemplos.

Así es como una evolución del ROV ha sido el AUV, en el que el control pasa a tomarlo un ordenador instalado en el propio vehículo, lo cual le permite operar de forma independiente. De esta forma, un AUV es capaz de tomar decisiones en función de cambios del medio o de los distintos eventos que sus sensores detecten y su software intérprete. (Sesto, 2009)

En la fig. 1 se presenta una clasificación de los robots submarinos. La principal manera de clasificarlos es de acuerdo a su nivel de autonomía. (Moreno, 2014)

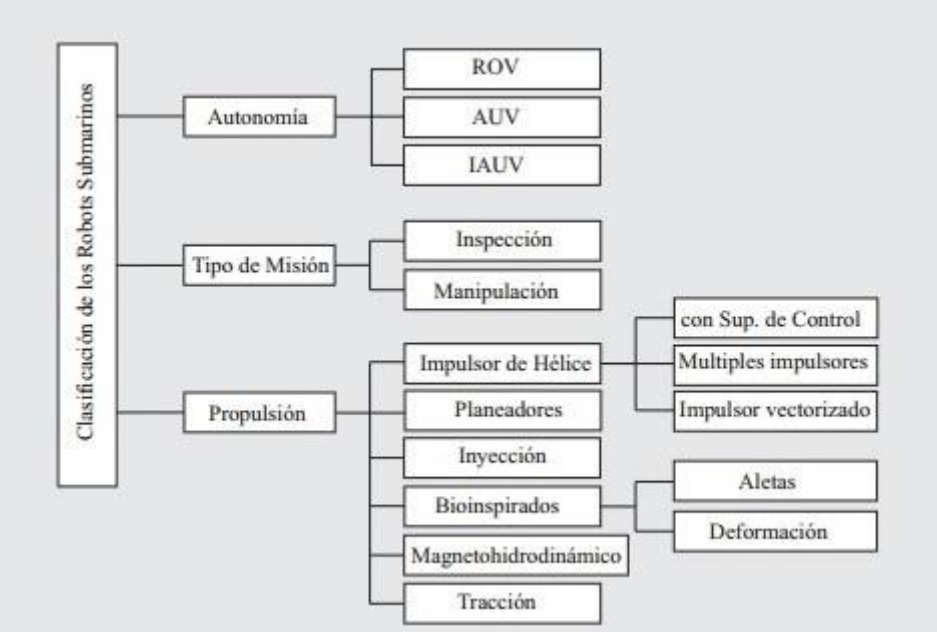

Figura 1: Clasificación de los robots submarinos.

El autor Héctor A. Moreno en conjunto con otros autores han definido en el artículo: "Robótica *Submarina: Conceptos, Elementos, Modelado y Control"* del año 2014 las principales características que distinguen a los vehículos autónomos subacuáticos:

- Los robots subacuáticos autónomos poseen una arquitectura de control que les permite realizar misiones sin la supervisión de un operador.
- Contienen su propia fuente de energía generalmente basada en baterías recargables, en dependencia de estas será el tiempo que podrán pasar estos explorando el fondo marino.
- Por lo general no hay una línea de comunicación entre el vehículo y la superficie debido a que se le suele programar con tareas y misiones predefinidas, pero en caso de que sea requerido, el intercambio de información con la superficie, la comunicación se puede realizar a través de dispositivos acústicos.
- Los AUVs se utilizan actualmente para tareas de exploración científica, muestreo oceanográfico, arqueología submarina y exploración debajo del hielo.
- Los datos que recopila el vehículo se almacenan en su memoria interna para luego ser analizados.
- Por otro lado, también se han utilizado para operaciones militares (ejemplo: la detección de minas), y se están desarrollando aplicaciones más elaboradas como lo son vigilancia submarina.

1.3 La navegación inercial

El problema de la navegación consiste en conocer la posición de un vehículo en cada instante de tiempo. Antiguamente los movimientos por tierra se basaban en puntos de referencia conocidos, y para los movimientos en el mar no se podía perder de vista la costa. Existieron diferentes métodos para dar solución a la necesidad de conocer posiciones en la superficie, el más tradicional era la observación de los astros para obtener referencias espaciales. Los métodos antiguos carecen de precisión para aplicaciones de vehículos autónomos. (Weston & Titterton, 2004)

Con la llegada del siglo XX, aparecieron nuevos sistemas de posicionamiento. La principal fuerza de desarrollo provino de los intereses militares, que buscaban determinar la posición de sus unidades de ataque para guiarlas hacia sus objetivos. Por esta necesidad se desarrollaron métodos de navegación inercial (Weston & Titterton, 2004). Los sistemas de referencias considerados en navegación son bases ortonormales, o sea, formadas por marcos de vectores de módulo unitario mutuamente ortogonales en el espacio euclidiano de dimensión 3: E^3 (España, 2010)

La navegación inercial se basa en la integración de aceleraciones registradas por acelerómetros y velocidades angulares registradas por giróscopos. Se sustenta en el principio de la cinemática que plantea que : "*conocidos en un instante inicial la velocidad, la orientación y la posición de un móvil, así como los valores instantáneos presentes y futuros de su aceleración lineal y su velocidad angular relativas a un sistema de referencia dado, es posible calcular la posición, la velocidad y la orientación del vehículo en todo instante futuro"* de ahí que para poder navegar sea necesario conocer la orientación. (España, 2010) .

La ecuación fundamental de la navegación inercial se deriva directamente de la segunda ley de Newton

$$\frac{d^2r}{dt^2} = C_b^i f + \mathrm{g} \qquad (1)$$

donde *r* es el vector de posición inercial, C_b^i Esta matriz es la matriz de cosenos directores, es la que expresa la orientación desde el marco del vehículo al marco de referencia inercial. (España, 2010), *f* es el vector fuerza específica total (registrada por los acelerómetros), y *g* es la aceleración provocada por la gravedad. Esta ecuación se integra para obtener la velocidad y posición, pero relativos al marco de referencia inercial. (España, 2010)

Los giróscopos medirán el movimiento angular del cuerpo del vehículo respecto al marco inercial de tal manera que permita calcular C_b^i. Además, el sistema inercial utilizará un modelo gravitacional para estimar la aceleración debido a la gravedad basándose en el cálculo de la posición actual. (Ramírez-González & Rubio, 2003)

En la navegación, se utilizan como marcos de referencia a los sistemas de coordenadas, que no son más que un conjunto de vectores y números que no tienen sentido si no están relacionarlos con una referencia conocida. Los marcos fundamentales son: marco del cuerpo, marco de navegación, marco inercial y marco terrestre. (Rogers, 2003)

1.4 Formas de representar la orientación

Para describir y caracterizar matemáticamente la orientación relativa entre marcos se utilizan ciertos conjuntos de parámetros, llamados "*parametrizaciones",* cuyos valores numéricos son actualizados permanentemente por el algoritmo de navegación. Al igual que su posición, la orientación de un cuerpo en el espacio es relativa al sistema de referencia elegido.

Para representar la orientación de un cuerpo existen varias representaciones matemáticas que pueden ser usadas para definir su orientación relativa a un marco de referencia (Weston & Titterton, 2004) los más empleados son los siguientes:

Ángulos de Euler:

Corresponden con los ángulos convencionales de alabeo (φ), cabeceo (θ), rumbo (ψ), donde alabeo, es la rotación alrededor del eje x; cabeceo, es la rotación alrededor del eje y; y rumbo es la rotación alrededor del eje z, conocidos en inglés como *roll*, *pitch* y *yaw* respectivamente.

El empleo de los ángulos de Euler tiene como desventaja la aparición de singularidades trigonométricas debido a que su cálculo se basa en el uso de este tipo de funciones. Un ejemplo de singularidad es el Bloqueo de *Gimbal*. (España, 2010)

Matriz de Cosenos Directores*:*

Las matrices de cosenos directores (DCM, en inglés *Direction Cosine Matrix*) son conocidas como matrices de rotación y son la elección clásica para la implementación de los ángulos de Euler. Tiene su base teórica en el álgebra lineal. Son matrices cuadradas, que sirven para transformar un vector expresado en un marco de referencia, en otro marco específico, mediante la multiplicación habitual de matrices. Cada elemento de la matriz es una función trigonométrica de uno o varios ángulos de Euler.

En el campo de la navegación, tiene su origen en la composición de rotaciones de un marco de referencia a otro. (Weston & Titterton, 2004)

Cualquier matriz de cosenos directores **C** debe cumplir con la condición:

$$\boldsymbol{C}_i^T \boldsymbol{C}_j = \begin{cases} 0, i \neq j \\ 1, i = j \end{cases} \qquad (2)$$

Entre dos marcos positivos con un mismo origen existe una única DCM $\boldsymbol{C}_b^a$ que describe la rotación relativa entre ambos marcos (España, 2010), por lo cual representa la rotación del vehículo de manera única.

Cuaterniones:

Un cuaternión es un número complejo de una componente real, y tres componentes imaginarias, que puede usarse para representar la orientación de un cuerpo en el espacio tridimensional. Una orientación arbitraria de un marco B relativa a un marco A se puede conseguir mediante una rotación del ángulo θ alrededor de un eje ${}^A\hat{\boldsymbol{r}}$ definido en el marco A como se muestra en la fig.2:

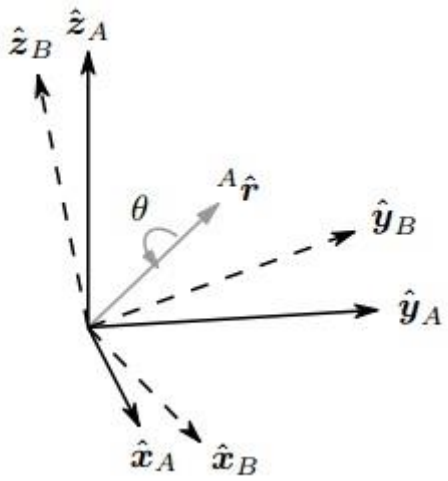

Figura 2: Orientación del marco B relativa al marco A rotando θ alrededor de ${}^A\hat{\boldsymbol{r}}$.

El cuaternión que describe esta orientación ${}^A_B\mathbf{q}$, está definido por la ecuación

$${}^A_B\boldsymbol{q} = [q_0 \quad q_1 \quad q_2 \quad q_3] = [\cos\tfrac{\theta}{2} \quad -r_x \sin\tfrac{\theta}{2} \quad -r_y \sin\tfrac{\theta}{2} \quad -r_z \sin\tfrac{\theta}{2}] \qquad (3)$$

Donde r_x, r_y y r_z definen las componentes del vector unitario ${}^A\hat{\boldsymbol{r}}$ en los ejes **x**, **y**, **z** del marco A, respectivamente. La aritmética de cuaterniones requiere a menudo que un cuaternión que describe una orientación sea normalizado por primera vez.

El conjugado del cuaternión, denotado por *, se puede usar para intercambiar los marcos descritos por una orientación. Por ejemplo, ${}^{B}_{A}\boldsymbol{q}$ es el conjugado de ${}^{A}_{B}\boldsymbol{q}$ y describe la orientación del marco A relativa al marco B. El conjugado de ${}^{A}_{B}\boldsymbol{q}$ se define por la ecuación

$$ {}^{A}_{B}\boldsymbol{q}^{*} = {}^{B}_{A}\boldsymbol{q} = [q_0 \; -q_1 \; -q_2 \; -q_3] \qquad (4) $$

El producto de cuaternión, denotado por $\otimes$, se puede usar para definir orientaciones de compuestos. Por ejemplo, para dos orientaciones descritas por ${}^{A}_{B}\boldsymbol{q}$ y ${}^{B}_{C}\boldsymbol{q}$, la orientación compuesta ${}^{A}_{B}\boldsymbol{q}$ puede definirse por la ecuación

$$ {}^{A}_{C}\boldsymbol{q} = {}^{B}_{C}\boldsymbol{q} \otimes {}^{A}_{B}\boldsymbol{q} \qquad (5) $$

A continuación, se muestran las características principales de los cuaterniones descritas por (Daniel Alpaya, 2014)

- Ofrecen una parametrización muy eficaz de la orientación debido que sólo tienen un parámetro redundante.
- No presentan singularidades para ningún valor de sus parámetros que deba resolverse con información suplementaria.
- Se observa dificultad para interpretar el sentido de sus parámetros.

Relación entre las formas de representación

Estas tres formas están relacionadas entre sí ya que la *DCM* puede estar expresada en términos de ángulos de Euler o cuaterniones y viceversa. También el cuaternión puede estar expresados en términos de los ángulos de Euler o de la *DCM* (Weston & Titterton, 2004) por lo que una manera de representar orientación puede estar en función de otra.

Es posible obtener **los Ángulos de Euler una vez estimada la orientación en forma de cuaternión** a partir de la fórmula:

$$ \begin{bmatrix} pitch \\ roll \\ yaw \end{bmatrix}(\boldsymbol{q}) = \begin{bmatrix} atan2\,(2(q_0q_1 + q_2q_3), 1 - 2\,({q_1}^2 + {q_2}^2)) \\ arcsen\,(2\,(q_0q_2 - q_3q_1)) \\ atan2(2(q_0q_3 + q_1q_2), 1 - 2\,({q_2}^2 + {q_3}^2) \end{bmatrix} \qquad (6) $$

Los cuaterniones, aunque no poseen una interpretación intuitiva rotan los ejes utilizando operaciones algebraicas que implican un menor tiempo de ejecución de los algoritmos.

1.5 Cinemática de la orientación de un cuerpo en el espacio

Un giróscopo de tres ejes mide la velocidad angular alrededor de los ejes **x**, **y**, **z** denominados w_x,w_y,w_z respectivamente. Si estos parámetros están dispuestos en el vector $\boldsymbol{w}$ definido por la ecuación

$$\boldsymbol{w}_t = [0\ w_x\ w_y\ w_z] \qquad (7)$$

La derivada del cuaternión que describe la tasa de cambio de orientación $\dot{\boldsymbol{q}}_t$ puede calcularse siempre y cuando se conozcan las condiciones iniciales, como

$$\dot{\boldsymbol{q}}_t = \frac{1}{2}\boldsymbol{q}_{t-1} \otimes \boldsymbol{w}_t \qquad (8)$$

Si Δt es el período de muestreo y $\boldsymbol{q}_{t-1}$ es la estimación previa de la orientación, entonces es posible estimar orientación en un tiempo t integrando numéricamente la derivada de cuaternión $\dot{\boldsymbol{q}}_t$, utilizando la ecuación

$$\boldsymbol{q}_t = \boldsymbol{q}_{t-1} + \dot{\boldsymbol{q}}_t \Delta t \qquad (9)$$

Los acelerómetros miden el cambio en la orientación con respecto al vector de la fuerza gravitacional, sin embargo, el acelerómetro puede ser utilizado sólo para medir la rotación alrededor de los ejes x, y, no alrededor del eje z, debido a que el vector gravitacional es paralelo al eje z. El magnetómetro se utiliza para medir el ángulo de rotación alrededor del eje z, a partir del vector campo magnético.

Un algoritmo de fusión de datos también llamado algoritmo de estimación de orientación es el encargado de utilizar las mediciones de los acelerómetros y/o magnetómetros para mejorar la estimación de la orientación obtenida a través de las mediciones del giróscopo.

1.6 Tipos de algoritmos existentes para la estimación de la orientación en un AUVs.

Los algoritmos existentes que permiten el cálculo de la orientación siguen dos estrategias distintas, una basada en el filtro de Kalman y otra basada en filtros complementarios.

Filtro de Kalman

El Filtro de Kalman (KF, en inglés *Kalman Filter*) es un algoritmo de estimación de estados que caracterizan un proceso o sistema. Es considerado un estimador de error cuadrático medio recursivo lineal mínimo que produce una estimación estadísticamente óptima de un estado del sistema subyacente representado por una corriente de datos ruidosos. Es el método *go-to* (ir-a) para fusionar datos de sensores en aplicaciones AHRS, ya que puede diseñarse para estimar tanto la orientación como otras variables tales como sesgos de sensores. (WATSON, 2013)

El Filtro de Kalman se puede dividir en dos pasos. El paso de predicción produce una estimación de las variables de estado actuales, así como sus incertidumbres. En el paso siguiente, se compara esta estimación previa con una medición externa, de modo que se pueda corregir por medio de una combinación ponderada de la predicción y la medición, con mayor peso a la que mayor certidumbre tenga. Esto es la matriz de ganancias de Kalman. Luego de la corrección, se actualiza la incertidumbre de la nueva estimación, que juntas constituirán la entrada en el próximo período de muestreo. (WATSON, 2013)

Filtro Complementario

Un filtro complementario utiliza un análisis en el dominio de la frecuencia de las señales para combinarlas para obtener una mejor estimación de una cantidad particular. Si las señales son afectadas por ruidos con diferente frecuencia, se pueden aplicar dos filtros, con un ancho de banda apropiado, de modo que la suma de las dos señales filtradas cubra toda la gama de frecuencias útiles (Valenti, Dryanovski, & Xiao, 2015). En esencia, el método consiste en el empleo de una colección de filtros $f_i(s), \text{con } i = 1, \ldots, n$ encargados de filtrar las señales obtenidas de cada uno de los instrumentos de medida.

Los filtros complementarios son muy empleados en sistemas de navegación inercial. Aplicaciones típicas son la combinación de las medidas de aceleración vertical y velocidad barométrica vertical para obtener una estimación de la velocidad vertical o mediciones de unidades inerciales y sistemas de visión. La idea básica del filtro complementario es combinar la salida del acelerómetro y del giróscopo para obtener una buena estimación del ángulo de orientación de la plataforma. (Gaydou, Redolfi, & Henze, 2011)

1.7 Algoritmos para la estimación de la orientación

Son varios los algoritmos que permiten la estimación de orientación. A continuación, se caracterizan los algoritmos a implementar como propuestas de solución.

Filtro Extendido de Kalman

El Filtro Extendido de Kalman (EKF, en inglés *Extended Kalman Filter*) es la versión no lineal del Filtro de Kalman regular, que utiliza el Jacobiano de las funciones de predicción y medición para linealizar alrededor de una estimación del estado actual y la covarianza. Esto permite utilizar un nuevo conjunto de modelos de predicción y de medición para la estimación. (WATSON, 2013)

Filtro Complementario basado en gradiente descendente

Es un algoritmo de estimación de orientación propuesto por Sebastian O.H. Madgwick en abril del 2010 que se basa en el método del gradiente descendente. El algoritmo está diseñado para sensores de bajo costo, prevé distorsión en las lecturas de los magnetómetros, y ofrece compensación al sesgo de los giróscopos. El filtro utiliza una representación utilizando los cuaterniones permitiendo la utilización de datos de los acelerómetros y magnetómetros para calcular la dirección del error de la medición del giróscopo como un derivado del cuaternión. Los beneficios del filtro incluyen: (Madgwick, 2010)

- Computacionalmente barato.
- Eficaz a bajas frecuencias de muestreo.

Filtro Complementario basado en cuaterniones para IMUs y MARGs

Es un algoritmo de estimación de orientación propuesto por Valenti (2015). Corrige la estimación de la orientación realizada con giróscopos, a partir de la estimación del error en el vector gravedad medido por los acelerómetros y el vector de intensidad del campo magnético medidas por los magnetómetros. Se basa en cuaterniones, y utiliza métodos de interpolación. (Valenti, Dryanovski, & Xiao, 2015)

Algoritmo combinado de Filtro Complementario y Filtro de Kalman propuesto por Navarro (2014).

Este algoritmo implementa una estructura en cascada de dos Filtros de Kalman y un Filtro Complementario. El primer FK se encarga de suavizar las mediciones obtenidas de nueve sensores (acelerómetros, giróscopos y magnetómetros en **x,y,** y **z**).

El segundo FK, implementado con un modelo cinemático lineal del movimiento circular uniforme tiene como entradas las mediciones suavizadas de los giróscopos en sus tres ejes y los ángulos de navegación dados por el FC (φ, θ, ψ), a partir de los cuales, y basado en modelo mencionado, corrige las mediciones de los giróscopos, las cuales serán realimentadas al FC, siendo este el que calcula los tres ángulos finales. (Navarro, 2014)

Algoritmo de Filtro de Kalman propuesto por Li Wang, Zheng Zhang & Ping Sun (2015).

Algoritmo basado en Filtro de Kalman que utiliza cuaterniones para la estimación en tiempo real de la orientación de un helicóptero. El Filtro de Kalman combina el giróscopo de 3 ejes y el cuaternión calculado para determinar los ángulos de alabeo y cabeceo. Esta combinación supera el error de linealización de las ecuaciones de medición y reduce el coste de cálculo. El procesamiento de un acelerómetro de tres ejes es a través del Gradiente Descendente. Este algoritmo AHRS es capaz de eliminar el impacto de distorsión magnética. (Wang, Zhang, & Sun, 2015).

Filtro Extendido de Kalman propuesto por Heikki Hyyti & Arto Visala (2015).

Consiste en un algoritmo de estadística de actitud que utiliza un filtro adaptativo ampliado de Kalman que consiste en la fusión de acelerómetros y giroscopios triaxiales de MEMS de bajo costo y se propone como solución al problema de utilizar estos sensores con precisión. (Hyyti & Visala, 2015)

Este algoritmo para garantizar la robustez frente a las aceleraciones temporales no gravitacionales utiliza una covarianza variable. También permite la calibración en línea exacta del giróscopo usando solamente un giróscopo triaxial y un acelerómetro. (Hyyti & Visala, 2015)

1.8 Herramientas informáticas y lenguajes de programación a utilizar

La selección de las herramientas y lenguaje de programación a utilizar se realizó teniendo en cuenta las características de los algoritmos a implementar y las pruebas a realizar.

MATLAB

Es una potente herramienta informática para el cálculo científico e ingenieril que ofrece un entorno de desarrollo integrado (IDE) con un lenguaje de programación propio (lenguaje M). Está disponible para las plataformas Unix, Windows, Mac OS X y GNU/Linux. Entre sus prestaciones básicas se hallan: la manipulación de matrices, la representación de datos y funciones, la implementación de algoritmos, la creación de interfaces de usuario (GUI) y la comunicación con programas en otros lenguajes y con otros dispositivos hardware. Es muy utilizado en universidades y centros de investigación y desarrollo. Permitió implementar y probar los algoritmos, además de graficar sus estimaciones y compararlas.

RStudio

RStudio es una herramienta informática multiplataforma para el procesamiento estadístico de datos que ofrece un IDE para su lenguaje de programación R. Permitió la validación de los resultados utilizando test estadísticos.

Lenguaje de programación M

Las aplicaciones de MATLAB se desarrollan en un lenguaje de programación propio. Este es interpretado, y puede ejecutarse tanto en el entorno interactivo, como a través de un archivo de *script* (archivos *.m). Permite operaciones de vectores y matrices, funciones, cálculo lambda, y programación orientada a objetos.

Lenguaje de programación R

R es un entorno y lenguaje interpretado de programación con un enfoque al análisis estadístico.

R es una implementación de software libre del lenguaje S, pero con soporte de alcance estático. Se trata de uno de los lenguajes más utilizados en investigación por la comunidad estadística. Tiene la posibilidad de cargar diferentes bibliotecas o paquetes con funcionalidades de cálculo y gráficas.

Al igual que S, se trata de un lenguaje de programación, lo que permite que permite extender sus propias funciones. La gran mayoría de las funciones de R fueron escritas en R, pero por cuestiones de rendimiento existen funciones escritas en lenguajes de más bajo nivel como C o Fortran.

1.9 Conclusiones del capítulo.

- El análisis de las características de los AUVs implica el estudio de los sistemas de navegación inercial sus componentes y los conceptos asociados al problema.
- Las representaciones de la orientación a utilizar son: la matriz de cosenos directores, los ángulos de Euler y los cuaterniones.
- Los algoritmos seleccionados a implementar como propuesta de solución al problema de la estimación de la orientación en AUVs son: el Filtro Extendido de Kalman según Watson (2013), Filtro Complementario según Madgwick (2010) y el Filtro Complementario según Valenti (2015) teniendo así representación de los dos tipos de algoritmos que existen para la estimación de la orientación.
- Se selecciona la herramienta Matlab y su lenguaje propio M para la implementación de los algoritmos, y diseño del escenario de prueba.
- Se selecciona la herramienta RStudio y su lenguaje propio R para las pruebas estadísticas que validen los resultados y permitan compara algoritmos.

CAPÍTULO 2: IMPLEMENTACIÓN DE LOS ALGORITMOS PARA LA ESTIMACIÓN DE LA ORIENTACIÓN EN VEHÍCULOS AUTÓNOMOS SUBACUÁTICOS

2.1 Introducción

En este capítulo se aborda como propuesta de solución la implementación de los algoritmos Filtro Extendido de Kalman según Watson (2013), el Filtro Complementario propuesto por Madgwick (2010) y el Filtro Complementario propuesto por Valenti (2015) para la estimación de orientación en Vehículos Autónomos Subacuáticos.

2.2 Presentación de la propuesta de solución.

Para dar solución al problema de la investigación se implementan tres algoritmos para la estimación de la orientación en vehículos autónomos subacuáticos. Para mantener las condiciones de bajo costo se hace uso de los sensores que se utilizan para la navegación (giróscopos y acelerómetros), añadiendo magnetómetros, cuya mayor utilidad reside en la mejora de la estimación del rumbo (Rodriguez & Chérigo, 2016). Los algoritmos a implementar utilizan diferentes estrategias para corregir la orientación estimada con los giróscopos con la cinemática clásica. Se eligieron tres algoritmos representativos de las dos grandes familias de algoritmos de estimación de orientación.

2.3 Descripción de los algoritmos a implementar

Existen una gran cantidad de algoritmos que estiman la orientación, por lo que compararlos todos y escoger el mejor para situaciones determinadas es bien difícil e implicaría costos elevados en el tiempo de selección y ejecución; por esta razón los algoritmos a implementar han sido seleccionados de manera no probabilística y aleatoria.

Todos los algoritmos existentes tienen ventajas y desventajas, algunos funcionan perfectamente para sensores de alta gama, sin embargo, puede que no sean recomendados cuando se trabaja con sensores de bajo costo, pero que cuentan con menos de 8 años de implementados desde su surgimiento. Uno de estos algoritmos pertenece a la clasificación de Filtro de Kalman y los otros dos son de Filtro Complementario.

Algoritmo de Filtro Extendido de Kalman

El Filtro Extendido de Kalman es la versión no lineal del filtro de Kalman regular, que utiliza el Jacobiano de las funciones de predicción y medición para linealizar alrededor de una estimación del estado actual y la covarianza. Esto permite utilizar un conjunto completamente nuevo de modelos de predicción y medición para la estimación, con covarianzas $\boldsymbol{Q}_k$ y $\boldsymbol{R}_k$.

$$\boldsymbol{x}_k = \boldsymbol{f}(x_{k-1}, u) + N(O, \boldsymbol{Q}_k) \qquad (10)$$

$$\boldsymbol{z} = \boldsymbol{h}(x_k) + N(O, \boldsymbol{R}_k) \qquad (11)$$

Las variables del vector de estado usado se traducen en el cuaternión de la estimación de orientación y tres componentes del sesgo del giróscopo, como se muestra a continuación.

$$\boldsymbol{x} = [q_0\ q_1\ q_2\ q_3\ \omega_{xb}\ \omega_{yb}\ \omega_{zb}]^T \qquad (12)$$

En esta implementación, el modelo de predicción representa una función que describe la próxima estimación de orientación en términos de la anterior estimada y el vector de proporción angular del giróscopo. El próximo cuaternión se halla simplemente a través de integración numérica usando

$$\boldsymbol{q}_k = \boldsymbol{q}_{k-1} + dt * \dot{\boldsymbol{q}}_k \qquad (13)$$

Donde:

$$\dot{\boldsymbol{q}}_\omega(\boldsymbol{q}, \boldsymbol{\omega}) = \frac{1}{2}\begin{bmatrix} -q_1 & -q_2 & -q_3 \\ q_0 & q_3 & -q_2 \\ -q_3 & q_0 & q_1 \\ q_2 & -q_1 & q_0 \end{bmatrix}\begin{bmatrix} \omega_x - \omega_{xb} \\ \omega_y - \omega_{yb} \\ \omega_z - \omega_{zb} \end{bmatrix} \qquad (14)$$

Los componentes del sesgo del giróscopo no se alteran durante la etapa de predicción. Esto permite que $\boldsymbol{f}$ se construya como:

$$\boldsymbol{x}_k = \boldsymbol{f}(\boldsymbol{x}_{k-1}, u_k) = \begin{bmatrix} q_0 + \frac{dt}{2} * \left(-q_1(\omega_x - \omega_{xb}) - q_2(\omega_y - \omega_{yb}) - q_3\ (\omega_z - \omega_{zb})\right) \\ q_1 + \frac{dt}{2} * \left(q_0(\omega_x - \omega_{xb}) - q_3(\omega_y - \omega_{yb}) - q_2\ (\omega_z - \omega_{zb})\right) \\ q_2 + \frac{dt}{2} * \left(-q_3(\omega_x - \omega_{xb}) - q_0(\omega_y - \omega_{yb}) - q_1\ (\omega_z - \omega_{zb})\right) \\ q_3 + \frac{dt}{2} * \left(q_2(\omega_x - \omega_{xb}) - q_1(\omega_y - \omega_{yb}) - q_0\ (\omega_z - \omega_{zb})\right) \\ \omega_{xb} \\ \omega_{yb} \\ \omega_{zb} \end{bmatrix} \qquad (15)$$

Luego se calcula el Jacobiano para producir la matriz **F**

$$\frac{\partial f}{\partial x} = \boldsymbol{F} =$$

$$\begin{bmatrix} 1 & -\frac{dt}{2}(\omega_x - \omega_{xb}) & -\frac{dt}{2}(\omega_y - \omega_{yb}) & -\frac{dt}{2}(\omega_z - \omega_{zb}) & \frac{dt}{2}q_1 & \frac{dt}{2}q_2 & \frac{dt}{2}q_3 \\ \frac{dt}{2}(\omega_x - \omega_{xb}) & 1 & -\frac{dt}{2}(\omega_z - \omega_{zb}) & \frac{dt}{2}(\omega_y - \omega_{yb}) & -\frac{dt}{2}q_0 & -\frac{dt}{2}q_3 & \frac{dt}{2}q_2 \\ \frac{dt}{2}(\omega_y - \omega_{yb}) & \frac{dt}{2}(\omega_z - \omega_{zb}) & 1 & -\frac{dt}{2}(\omega_x - \omega_{xb}) & \frac{dt}{2}q_3 & -\frac{dt}{2}q_0 & -\frac{dt}{2}q_1 \\ \frac{dt}{2}(\omega_z - \omega_{zb}) & -\frac{dt}{2}(\omega_y - \omega_{yb}) & \frac{dt}{2}(\omega_x - \omega_{xb}) & 1 & -\frac{dt}{2}q_2 & \frac{dt}{2}q_1 & -\frac{dt}{2}q_0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

(16)

El modelo de medición sirve para correlacionar la estimación del estado actual con el vector de medición, permitiendo que la predicción sea comparada con las mediciones del mundo real. En este caso, esto implica describir lo que tanto el acelerómetro como el magnetómetro deben leerse teniendo en cuenta la predicción del cuaternión actual, de modo que el vector de medición puede definirse como:

$$\boldsymbol{z} = \begin{bmatrix} \mathrm{a_x} & \mathrm{a_y} & \mathrm{a_z} & \mathrm{m_x} & \mathrm{m_y} & \mathrm{m_z} \end{bmatrix}^{\mathrm{t}} \qquad (17)$$

- **Mapeo del acelerómetro:**

El vector de gravedad se hace girar en el marco del cuerpo, representado por el cuaternión *q*, trazándola en el vector de aceleraciones. Teniendo en cuenta solo la dirección del vector de gravedad y no su magnitud, el vector se normaliza antes de ser operado.

$$\begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} = \boldsymbol{h}(\boldsymbol{x}_k) = R(\boldsymbol{q}) * \vec{\boldsymbol{g}} = R(q) * \begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix} = \begin{bmatrix} -2(q_1 q_3 - q_0 q_2) \\ -2(q_2 q_3 - q_0 q_1) \\ -{q_0}^2 + {q_1}^2 + {q_2}^2 - {q_3}^2 \end{bmatrix} \quad (18)$$

- **Mapeo de magnetómetro:**

El mapeo del magnetómetro se realiza de la misma manera que con el acelerómetro, haciendo girar el vector de fuerza de campo magnético del marco de referencia en el marco del cuerpo representado por el cuaternión de estimación

$$\begin{bmatrix} m_x \\ m_y \\ m_z \end{bmatrix} = \boldsymbol{h}(\boldsymbol{x}_k) = R(\vec{\boldsymbol{q}}) * \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix} =$$

$$\begin{bmatrix} b_x(q_0^2 + q_1^2 - q_2^2 - q_3^2) + 2b_y(q_1 q_2 - q_0 q_3) + 2b_z(q_1 q_3 - q_0 q_2) \\ 2b_x(q_1 q_2 - q_0 q_3) + b_y(q_0^2 - q_1^2 + q_2^2 - q_3^2) + 2b_z(q_2 q_3 - q_0 q_1) \\ 2b_x(q_1 q_3 - q_0 q_2) + 2b_y(q_2 q_3 - q_0 q_1) + b_z(q_0^2 - q_1^2 - q_2^2 + q_3^2) \end{bmatrix} \qquad (19)$$

A continuación, se presentan las ecuaciones utilizadas en el filtro extendido de Kalman en cada una de sus etapas.

- **Predicción**

Predicción del estado estimado $\boldsymbol{x}_k = f(\boldsymbol{x}_{k-1}, \boldsymbol{u})$ (20)

Predicción de la covarianza estimada $\boldsymbol{P} = \boldsymbol{FPF'} + \boldsymbol{Q}$ (21)

- **Actualización**

Medición residual $\boldsymbol{y} = \boldsymbol{z} - h\,(\boldsymbol{x}_k)$ (22)

Covarianza residual $\boldsymbol{S} = \boldsymbol{HPH'} + \boldsymbol{R}$ (23)

Ganancia de Kalman $\boldsymbol{K} = \boldsymbol{PH'S^{-1}}$ (24)

Actualización del estado estimado $\boldsymbol{x}_k = \boldsymbol{x}_k + \boldsymbol{K}_y$ (25)

Actualización de la covarianza estimada $\boldsymbol{P} = (\boldsymbol{I} - \boldsymbol{KH})\boldsymbol{P}$ (26)

Con $\boldsymbol{F} = \frac{\partial \boldsymbol{f}}{\partial x}$ (27) y $\boldsymbol{H} = \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}}$ (28). Se representa la matriz de identidad.

El siguiente diagrama representa el flujo del Filtro de Kalman y fue presentado en (Leccadito, Bakker, Niu, & Klenke, 2015)

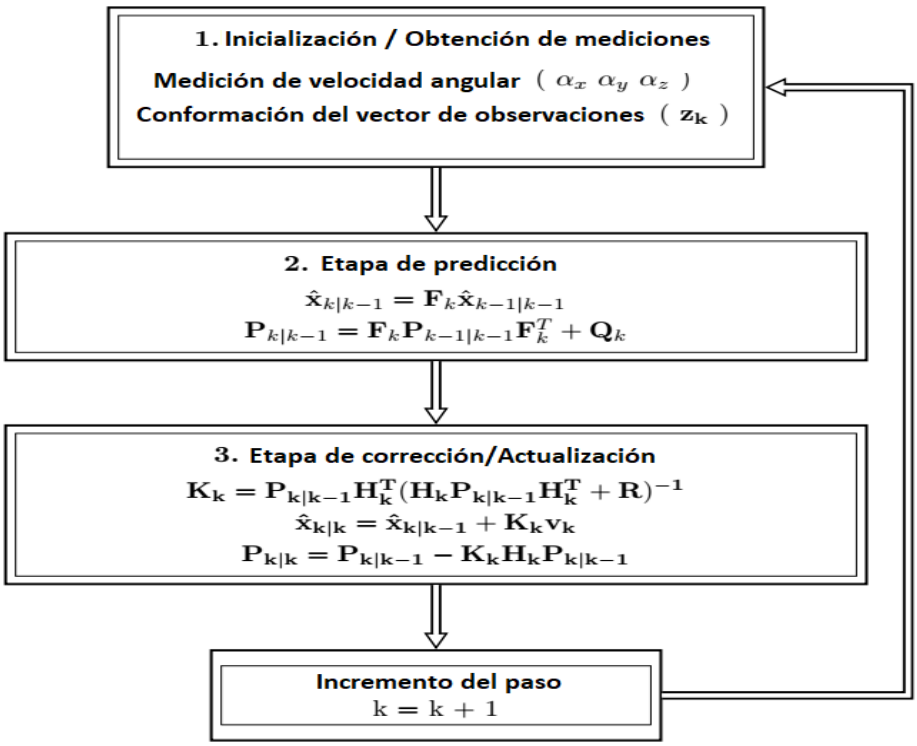

Figura 3: Diagrama del Filtro Extendido de Kalman

El cuaternión estimado luego se convierte en los valores de los ángulos de orientación alabeo, cabeceo y rumbo usando

$$\begin{bmatrix} pitch \\ roll \\ yaw \end{bmatrix}(q) = \begin{bmatrix} atan2\,(2(q_0q_1 + q_2q_3), 1 - 2\,({q_1}^2 + {q_2}^2)) \\ arcsen\,(2\,(q_0q_2 - q_3q_1)) \\ atan2(2)(q_0q_3 + q_1q_2), 1 - 2\,({q_2}^2 + {q_3}^2) \end{bmatrix} \quad (29)$$

Implementación del algoritmo de Filtro extendido de Kalman según Watson en lenguaje .m (anexo 1).

Algoritmo de Filtro Complementario basado en gradiente descendente

- **Orientación desde la velocidad angular:**

El vector ${}^{S}\boldsymbol{w}$ definido por la ecuación ${}^{S}\boldsymbol{w} = [0 \quad \omega_x \quad \omega_y\, \omega_z]$, donde ω_x, ω_y y ω_z representa la velocidad angular medida por el giróscopo alrededor de los ejes **x**, **y**, **z.** La orientación del marco terrestre relativo al marco del sensor en el tiempo t ${}^{E}_{S}\dot{\boldsymbol{q}}_{w,t}$, puede ser calculada integrando numéricamente la derivada del cuaternión ${}^{E}_{S}\dot{\boldsymbol{q}}_{w,t}$, como se describe a continuación:

$${}^{\mathrm{S}}_{\mathrm{E}}\dot{\boldsymbol{q}}_{w,t} = \frac{1}{2}\,{}^{\mathrm{S}}_{\mathrm{E}}\mathbf{q}_{\mathrm{est,t-1}} \otimes {}^{S}\boldsymbol{w}_t \quad (30)$$

$${}^{\mathrm{S}}_{\mathrm{E}}\mathbf{q}_{\mathrm{w,t}} = {}^{\mathrm{S}}_{\mathrm{E}}\mathbf{q}_{\mathrm{est,t-1}} + {}^{\mathrm{S}}_{\mathrm{E}}\dot{\boldsymbol{q}}_{w,t}\Delta\mathrm{t} \quad (31)$$

- **Orientación desde el vector de mediciones**

Un acelerómetro de tres ejes medirá la magnitud y la dirección del campo de gravedad en el marco del sensor combinado con aceleraciones lineales debido al movimiento del sensor. Del mismo modo, un magnetómetro de tres ejes medirá la magnitud y la dirección del campo magnético de la Tierra en el marco del sensor combinado con el flujo magnético local y las distorsiones. En el contexto de un filtro de orientación, se asumirá inicialmente que un acelerómetro solo medirá la gravedad en cada eje, un magnetómetro solo medirá el campo magnético de la Tierra.

La orientación ${}^{S}_{E}\boldsymbol{q}$ se podrá calcular utilizando:

$$\min_{{}^{S}_{E}\boldsymbol{q}\in\mathbb{R}^4} \boldsymbol{f}\left({}^{S}_{E}\boldsymbol{q}, {}^{E}\boldsymbol{d}, {}^{S}\boldsymbol{s}\right) \quad (32)$$

Donde el objetivo de la función es

$$\boldsymbol{f}\left({}^{S}_{E}\boldsymbol{q}, {}^{E}\boldsymbol{d}, {}^{S}\boldsymbol{s}\right) = {}^{S}_{E}\boldsymbol{q}^* \otimes {}^{E}\boldsymbol{d} \otimes {}^{S}_{E}\boldsymbol{q} - {}^{S}\boldsymbol{s} \quad (33)$$

La dirección del campo en el marco terrestre es ${}^{E}\boldsymbol{d}$ y la dirección de la medición del campo en el marco terrestre es ${}^{S}\boldsymbol{s}$.

La siguiente ecuación:

$$ {}^{S}_{E}\boldsymbol{q}_{k+1} = {}^{S}_{E}\boldsymbol{q}_{k} - \mu \frac{\nabla \boldsymbol{f}({}^{S}_{E}\boldsymbol{q}_{k}, {}^{E}\boldsymbol{d}, {}^{S}\boldsymbol{s})}{\|\nabla \boldsymbol{f}({}^{S}_{E}\boldsymbol{q}_{k}, {}^{E}\boldsymbol{d}, {}^{S}\boldsymbol{s})\|}, k = 0,1,2 \ldots . n \quad (34) $$

Describe el algoritmo de gradiente descendente para n iteraciones resultando una estimación de la orientación de ${}^{S}_{E}\boldsymbol{q}_{n+1}$.

La siguiente ecuación define la solución del gradiente y la función objetivo con su Jacobiano:

$$ \nabla \boldsymbol{f}({}^{S}_{E}\boldsymbol{q}_{k}, {}^{E}\boldsymbol{d}, {}^{S}\boldsymbol{s}) = \boldsymbol{J}^{T}({}^{S}_{E}\boldsymbol{q}_{k}, {}^{E}\boldsymbol{d})\boldsymbol{f}({}^{S}_{E}\boldsymbol{q}_{k}, {}^{E}\boldsymbol{d}, {}^{S}\boldsymbol{s}) \quad (35) $$

Las siguientes ecuaciones definen la forma general del algoritmo aplicable a un campo determinado en cualquier dirección:

$$ \boldsymbol{f}_{g}({}^{S}_{E}\boldsymbol{q}, {}^{S}\boldsymbol{a}) = \begin{bmatrix} 2(q_1 q_3 - q_0 q_2) - a_x \\ 2(q_0 q_1 + q_2 q_3) - a_y \\ 2\left(\frac{1}{2} - q_1^2 - q_2^2\right) - a_z \end{bmatrix} \quad (36) $$

$$ \boldsymbol{J}_{g}({}^{S}_{E}\boldsymbol{q}) = \begin{bmatrix} -2q_2 & 2q_3 & -2q_0 & 2q_1 \\ 2q_1 & 2q_0 & 2q_3 & 2q_2 \\ 0 & -4q_1 & -4q_2 & 0 \end{bmatrix} \quad (37) $$

Normalizando las mediciones del magnetómetro:

$$ \boldsymbol{f}_{g}({}^{S}_{E}\boldsymbol{q}, {}^{E}\boldsymbol{b}, {}^{S}\boldsymbol{m}) = \begin{bmatrix} 2b_x(0.5 - q_2^2 - q_3^2) + 2b_z(q_1 q_3 - q_0 q_2) - m_x \\ 2b_x(q_1 q_2 - q_0 q_3) + 2b_z(q_0 q_1 + q_2 q_3) - m_y \\ 2b_x(q_0 q_2 + q_1 q_3) + 2b_z\left(\frac{1}{2} - q_1^2 - q_2^2\right) - m_z \end{bmatrix} \quad (38) $$

$$ \boldsymbol{J}_{g}({}^{S}_{E}\boldsymbol{q}, {}^{E}\boldsymbol{b}) = \begin{bmatrix} -2b_z q_2 & 2b_z q_3 & -4b_x q_2 - 2b_z q_0 & -4b_x q_3 + 2b_z q_1 \\ -2b_x q_3 + 2b_z q_1 & 2b_x q_2 + 2b_z q_0 & 2b_x q_1 + 2b_z q_3 & 2b_x q_0 + 2b_z q_2 \\ 2b_x q_2 & 2b_x q_3 - 4b_z q_1 & 2b_x q_0 - 4b_z q_2 & 2b_x q_1 \end{bmatrix} \quad (39) $$

La siguiente ecuación:

$$ {}^{S}_{E}\boldsymbol{q}_{\nabla,t} = {}^{S}_{E}\boldsymbol{q}_{est,t-1} - \mu_t \frac{\nabla \boldsymbol{f}}{\|\nabla \boldsymbol{f}\|} \quad (40) $$

Calcula la estimación en un tiempo t basada en una previa estimación de la orientación ${}^{S}_{E}\boldsymbol{q}_{est,t-1}$. La forma de $\nabla \boldsymbol{f}$ es seleccionada de acuerdo al sensor en uso como se muestra a continuación:

$$ \nabla \boldsymbol{f} = \begin{cases} \boldsymbol{J}_{g}^{T}({}^{S}_{E}\boldsymbol{q}_{est,t-1})\boldsymbol{f}_{g}({}^{S}_{E}\boldsymbol{q}_{est,t-1}, {}^{S}\boldsymbol{a}_{t}) \\ \boldsymbol{J}_{g,b}^{T}({}^{S}_{E}\boldsymbol{q}_{est,t-1}, {}^{E}\boldsymbol{b})\boldsymbol{f}_{g,b}({}^{S}_{E}\boldsymbol{q}_{est,t-1}, {}^{S}\boldsymbol{a}, {}^{E}\boldsymbol{b}, {}^{S}\boldsymbol{m}) \end{cases} \quad (41) $$

se calcula μ_t a partir de la ecuación:

$$\mu_t = \alpha \left\| {}^{S}_{E}\dot{\boldsymbol{q}}_{w,t} \right\| \nabla t, \alpha > 1 \qquad (42)$$

Esto es siempre que se conozcan las condiciones iniciales. En estas ecuaciones, ${}^{S}\boldsymbol{w}_t$ es la velocidad angular medida en el tiempo t, Δt es el período de muestreo y ${}^{S}_{E}\boldsymbol{q}_{est,t-1}$ es la estimación previa de orientación. El subíndice ω indica que el cuaternión se calcula a partir de la velocidad angular.

Una orientación estimada del marco del sensor relativa al marco terrestre ${}^{S}_{E}\boldsymbol{q}_{est,t}$ es obtenida a través de la fusión de los cálculos de la orientación ${}^{S}_{E}\boldsymbol{q}_{\omega,t}$ y ${}^{S}_{E}\boldsymbol{q}_{\nabla,t}$ como se muestra a continuación:

$${}^{S}_{E}\boldsymbol{q}_{est,t} = \gamma_t {}^{S}_{E}\boldsymbol{q}_{\nabla,t} + (1-\gamma_t) {}^{S}_{E}\boldsymbol{q}_{\omega,t}, 0 \leq \gamma_t \leq 1 \qquad (43)$$

Puede ser visto por las siguientes ecuaciones que el filtro calcula la orientación ${}^{S}_{E}\boldsymbol{q}_{est}$ por la integración numérica del ángulo de la estimación de la orientación ${}^{S}_{E}\dot{\boldsymbol{q}}_{est}$. El filtro calcula ${}^{S}_{E}\dot{\boldsymbol{q}}_{est}$ como el ángulo de cambio de orientación medido por el giróscopo ${}^{S}_{E}\dot{\boldsymbol{q}}_{\omega}$ con la magnitud de medición de error del giróscopo β, eliminado en la dirección de la estimación del error, ${}^{S}_{E}\dot{\boldsymbol{q}}_{\epsilon}$ es las medidas del acelerómetro y el magnetómetro

$${}^{S}_{E}\boldsymbol{q}_{est,t} = {}^{S}_{E}\boldsymbol{q}_{est,t-1} + {}^{S}_{E}\dot{\boldsymbol{q}}_{est,t}\Delta t \qquad (44)$$

$${}^{S}_{E}\dot{\boldsymbol{q}}_{est,t} = {}^{S}_{E}\dot{\boldsymbol{q}}_{\omega,t} - \beta {}^{S}_{E}\dot{\boldsymbol{q}}_{\epsilon,t} \qquad (45)$$

$${}^{S}_{E}\dot{\boldsymbol{q}}_{\epsilon,t} = \frac{\nabla \boldsymbol{f}}{\|\nabla \boldsymbol{f}\|} \qquad (46)$$

Implementación en M del algoritmo propuesto por Madgwick (2010) (anexo 2).

Algoritmo de Filtro Complementario basado en cuaterniones para IMUs y MARGs

Este algoritmo describe un nuevo enfoque que proporciona una estimación en forma de cuaternión como la solución algebraica de un sistema a partir de mediciones inerciales/magnéticas. Se separan los problemas de encontrar el cuaternión de "inclinación" y el cuaternión de rumbo en dos sub-partes del sistema. Este procedimiento es la clave para evitar el impacto de las perturbaciones magnéticas en los componentes alabeo y cabeceo de la orientación cuando el sensor está rodeado por un flujo magnético no deseado. Se aborda el problema de las perturbaciones magnéticas separando las correcciones del cuaternión en dos secuencias de corrección diferentes y adoptando el método presentado en (Madgwick, 2010), que también permite que el algoritmo sea independiente de los parámetros externos. Además, se adopta un enfoque de ganancia adaptativa para reducir el error de estimación durante el movimiento dinámico alto.

- **Predicción**

En la etapa de predicción, el vector de velocidad angular, medido por el giróscopo de tres ejes, se utiliza para calcular una primera estimación de la orientación en forma de cuaternión. Asumiendo que son conocidas las condiciones iniciales, es posible obtener la orientación del marco global relativa al marco local en el tiempo t_k puede calcularse finalmente integrando numéricamente la derivada de cuaternión utilizando el período de muestreo $\Delta t = t_k - t_{k-1}$.

$$ {}^{L}_{G}\dot{\boldsymbol{q}}_{\omega,t_k} = {}^{L}_{G}\boldsymbol{q}_{t_{k-1}} + {}^{L}_{G}\dot{\boldsymbol{q}}_{\omega,t_k}\Delta t \qquad (47) $$

Donde:

$$ {}^{L}_{G}\dot{\boldsymbol{q}}_{\omega,t_k} = -\frac{1}{2}{}^{L}\boldsymbol{w}_{q,t_k} \otimes {}^{L}_{G}\boldsymbol{q}_{t_{k-1}} \qquad (48) $$

- **Corrección**

El cuaternión predicho ${}^{L}_{G}\dot{\boldsymbol{q}}_{w}$ se corrige por medio de otros dos cuaterniones:

$$ {}^{L}_{G}\boldsymbol{q} = {}^{L}_{G}\boldsymbol{q}_{w} \otimes \widehat{\Delta\boldsymbol{q}}_{acc} \otimes \widehat{\Delta\boldsymbol{q}}_{mag} \qquad (49) $$

Se divide la etapa de corrección en dos pasos. En el primer paso se corrige el cuaternión previsto sólo en los componentes de alabeo y balanceo mediante la aplicación de $\widehat{\Delta\boldsymbol{q}}_{acc}$ calculado con las mediciones del acelerómetro. En el segundo paso se utilizan las lecturas del campo magnético y se corrige el componente de rumbo del cuaternión aplicando $\widehat{\Delta\boldsymbol{q}}_{mag}$.

- **Corrección basada en el acelerómetro**

Primeramente, se obtiene el vector gravedad predicha a partir de la fórmula

$$ \boldsymbol{R}\left({}^{G}_{L}\boldsymbol{q}_{w}\right){}^{L}\boldsymbol{a} = {}^{G}\boldsymbol{g}_{p} \qquad (50) $$

La gravedad prevista tiene una pequeña desviación del vector de gravedad real; por lo tanto, se calcula el cuaternión $\Delta\boldsymbol{q}_{acc}$, que gira ${}^{L}\boldsymbol{a}$ en ${}^{G}\boldsymbol{g}_{p}$, utilizando el siguiente sistema:

$$ \boldsymbol{R}(\Delta\boldsymbol{q}_{cc}) \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} g_x \\ g_y \\ g_z \end{bmatrix} \qquad (51) $$

Donde

$$ {}^{L}\boldsymbol{a} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \qquad {}^{G}\boldsymbol{g}_{p} = \begin{bmatrix} g_x \\ g_y \\ g_z \end{bmatrix} \qquad (52) $$

Obteniendo

$$\boldsymbol{q}_{acc} = \begin{cases} \left[\sqrt{\frac{g_z+1}{2}} \quad -\frac{g_y}{\sqrt{2(g_z+1)}} \quad \frac{g_x}{\sqrt{2(g\ \ z+1)}} \ 0\right]^T, g_z \geq 0 \\ \left[-\frac{g_y}{\sqrt{2(1-g_z)}} \quad \sqrt{\frac{1-g_z}{2}} \quad 0 \ \frac{g_x}{\sqrt{2(1-g_z)}}\right]^T, g_z < 0 \end{cases} \qquad (53)$$

Luego se le aplica una interpolación con el cuaternión identidad $\boldsymbol{q_I} = [1\ 0\ 0\ 0]$.

Si $\Delta q_{0acc} > \epsilon$, donde ϵ es el valor del umbral ($\epsilon = 0.9$), se utiliza Interpolación Lineal (LERP) como en la siguiente ecuación:

$$\overline{\Delta \boldsymbol{q}_{acc}} = (1-\alpha)\boldsymbol{q}_I + \alpha \Delta \boldsymbol{q}_{acc} \qquad (54)$$

Como LERP no mantiene la norma unitaria del cuaternión $\Delta\, \boldsymbol{q}_{acc}$, este se normaliza inmediatamente después:

$$\widehat{\Delta \boldsymbol{q}_{acc}} = \frac{\overline{\Delta \boldsymbol{q}_{acc}}}{\|\Delta \boldsymbol{q}_{acc}\|} \qquad (55)$$

Si $\Delta q_{0acc} < \epsilon$, se utiliza la Interpolación Lineal Esférica (SLERP). La fórmula SLERP que se utiliza es la siguiente:

$$\widehat{\Delta \boldsymbol{q}_{acc}} = \frac{\sin([1-\alpha]\Omega)}{\sin\Omega} q_I + \frac{\sin(\alpha\Omega)}{\sin\Omega} \Delta \boldsymbol{q}_{acc} \qquad (56)$$

Donde α ∈ [0,1] es la ganancia que caracteriza la frecuencia de corte del filtro.

El Δq_{acc} puede tener un valor grande cuando la gravedad prevista tiene una desviación significativa de la gravedad real. Si esa condición no ocurre, el delta cuaternión es muy pequeño; por lo tanto, es preferible usar la fórmula LERP porque es computacionalmente más eficiente ya que no utiliza funciones trigonométricas. Finalmente, ${}^{L}_{G}\boldsymbol{q_w}$ se multiplica por $\widehat{\Delta \boldsymbol{q}_{acc}}$

- **Corrección basada en el magnetómetro**

$${}^{L}_{G}\boldsymbol{q}' = {}^{L}_{G}\boldsymbol{q_w} \otimes \widehat{\Delta \boldsymbol{q}_{acc}} \qquad (57)$$

Primeramente, se obtiene el vector campo magnético predicho a partir de la fórmula:

$$R\left({}^{G}_{L}\boldsymbol{q}'\right) {}^{L}\boldsymbol{m} = \mathrm{l} \qquad (58)$$

Luego se encuentra $\Delta \boldsymbol{q}_{mag}$ utilizando el siguiente sistema

$$\mathrm{R}(\Delta \mathbf{q}_{\mathrm{mag}}) \begin{bmatrix} \mathrm{l_x} \\ \mathrm{l_y} \\ \mathrm{l_z} \end{bmatrix} = \begin{bmatrix} \sqrt{\mathrm{l}_{\mathrm{x}}^2 + \mathrm{l}_{\mathrm{y}}^2} \\ 0 \\ \mathrm{l_z} \end{bmatrix} \qquad (59)$$

Obteniendo

$$\boldsymbol{q}_{mag} = \begin{cases} \left[\dfrac{\sqrt{\left(l_x^2 + l_y^2\right) + l_x\sqrt{l_x^2 + l_y^2}}}{\sqrt{2\left(l_x^2 + l_y^2\right)}} \quad 0 \quad 0 \quad \dfrac{l_y}{\sqrt{2}\sqrt{\left(l_x^2 + l_y^2\right) + l_x\sqrt{l_x^2 + l_y^2}}}\right]^T, l_x \geq 0 \\ \left[\dfrac{l_y}{\sqrt{2}\sqrt{\left(l_x^2 + l_y^2\right) - l_x\sqrt{l_x^2 + l_y^2}}} \quad 0 \quad 0 \quad \dfrac{\sqrt{\left(l_x^2 + l_y^2\right) - l_x\sqrt{l_x^2 + l_y^2}}}{\sqrt{2\left(l_x^2 + l_y^2\right)}}\right]^T, l_x < 0 \end{cases}$$

(60)

Entonces se aplica LERP o SLERP al igual que en las ecuaciones 54 y 56, respectivamente donde la ganancia α es reemplazada por β, obteniendo $\Delta\widehat{\boldsymbol{q}_{mag}}$. Por último, el cuaternión final se obtiene de la fórmula

$$\frac{L}{G}\boldsymbol{q} = \frac{L}{G}\boldsymbol{q}' \otimes \widehat{\Delta\boldsymbol{q}}_{mag} \qquad (61)$$

Implementación en M del algoritmo de Filtro Complementario propuesto por Valenti (2015) (anexo 3).

2.4 Generalidades de la implementación en la herramienta Matlab

Para las implementaciones de las soluciones en el lenguaje M se tuvo en cuenta un conjunto de buenas prácticas de la programación que se describen a continuación.

Estilos de código

Los estilos de código no son más que el conjunto de reglas o normas usadas para escribir código y que incluye una gran gama de aspectos dentro del proceso de codificación. Un buen estilo de programación aporta a la eficiencia del proceso de desarrollo, logrando que los programas sean más robustos y comprensibles, en otras palabras, codificar de una manera clara y legible contribuye a la calidad de la implementación.

Independientemente de que las plataformas de desarrollo influyan en cierto modo a la formación de estilos de programación, es responsabilidad de cada equipo determinar el suyo porque no existe un estilo de código definido y mundialmente estandarizado.

El estilo de código utilizado es detallado a continuación:

- El sangrado del texto es de 2 espacios.

- Los nombres de las variables poseen palabras separadas por guiones donde la primera palabra posee caracteres en minúscula y el resto en mayúscula.
- Las constantes se definen en mayúsculas. Cuando cuentan con más de una palabra se separan por guiones bajos.
- Los nombres de los métodos son palabras con letras iniciales mayúsculas y separadas por guiones. Los parámetros, siempre que es posible, tienen valores por defecto. Sin espacio entre el nombre de la función y lo siguiente (paréntesis).
- Los comentarios son añadidos para explicar el flujo del código, el propósito de las funciones y variables y como descripción de trozos de código.
- El espacio en blanco se utiliza con bastante libertad, se separan las líneas un poco para ganar en claridad.

2.5 Conclusiones del capítulo

❖ A partir de la implementación de los algoritmos seleccionados se estima la orientación al utilizar las mediciones de los sensores de bajo costo.

❖ Para la interpretación de resultados se convierte el cuaternión resultante de la estimación de la orientación a ángulos de Euler.

❖ Los algoritmos implementados hacen uso de estilos de códigos puesto que de esta forma se codifica de manera clara y legible y contribuye a la calidad de la implementación.

CAPÍTULO 3: RESULTADOS, COMPARACIÓN Y PRUEBAS

3.1 Introducción

En este capítulo se realizan pruebas para evaluar el desempeño de los algoritmos implementados, así como también la comparación entre ellos para estimar la orientación en Vehículos Autónomos Subacuáticos. También se plantean los resultados obtenidos al ejecutar los algoritmos con datos de experimentos realizados con el Vehículo Subacuático HRC-AUV, por parte del Grupo de Automática y Percepción de la Universidad Central de Las Villas “Marta Abreu” en la Bahía de La Habana en 2014.

3.2 Datos experimentales utilizados

Se dispone de un conjunto de datos obtenidos de dos experimentos realizados con el submarino HRC-AUV 400, de autopiloto y unidad de medición inercial MTI-g que es de gama comercial, aunque es costoso, es considerado de bajo costo. El experimento cuenta con 190000 muestras que brindan mediciones de los acelerómetros, giróscopos y magnetómetros en los ejes **X, Y y Z**. También ofrece los valores de latitud, longitud y altura en toda la trayectoria, además de los ángulos de alabeo, cabeceo y rumbo. Los datos fueron capturados a una frecuencia de muestreo de 100Hz, lo que implica que el período de muestreo sea de 0.01 segundos.

3.3 Gráficas.

A continuación, se presentan en gráficas los resultados arrojados por los algoritmos seleccionados a través de la herramienta Matlab después de la implementación de los mismos. Estos se dan por ángulos para facilitar así la comparación, donde el color negro se reserva para la “referencia” o sea el valor de la orientación real del vehículo, el color “azul” se reserva para el resultado que produjo el algoritmo de Filtro Extendido de Kalman según Watson (2013) al interpretar las mediciones de los sensores de bajo costo. Lo mismo sucede para el color “verde” y “violeta” donde estos representan los resultados de los algoritmos de Filtro Complementario propuesto por Madgwick (2010) y Valenti (2015) respectivamente.

Para facilitar la interpretación de estas gráficas, nótese que entre más se acerque un algoritmo a la referencia mejor estimación de la orientación proporcionará este algoritmo concluyendo que será el más apropiado para la estimación de la orientación en Vehículos Autónomos Subacuáticos con sensores de bajo costo.

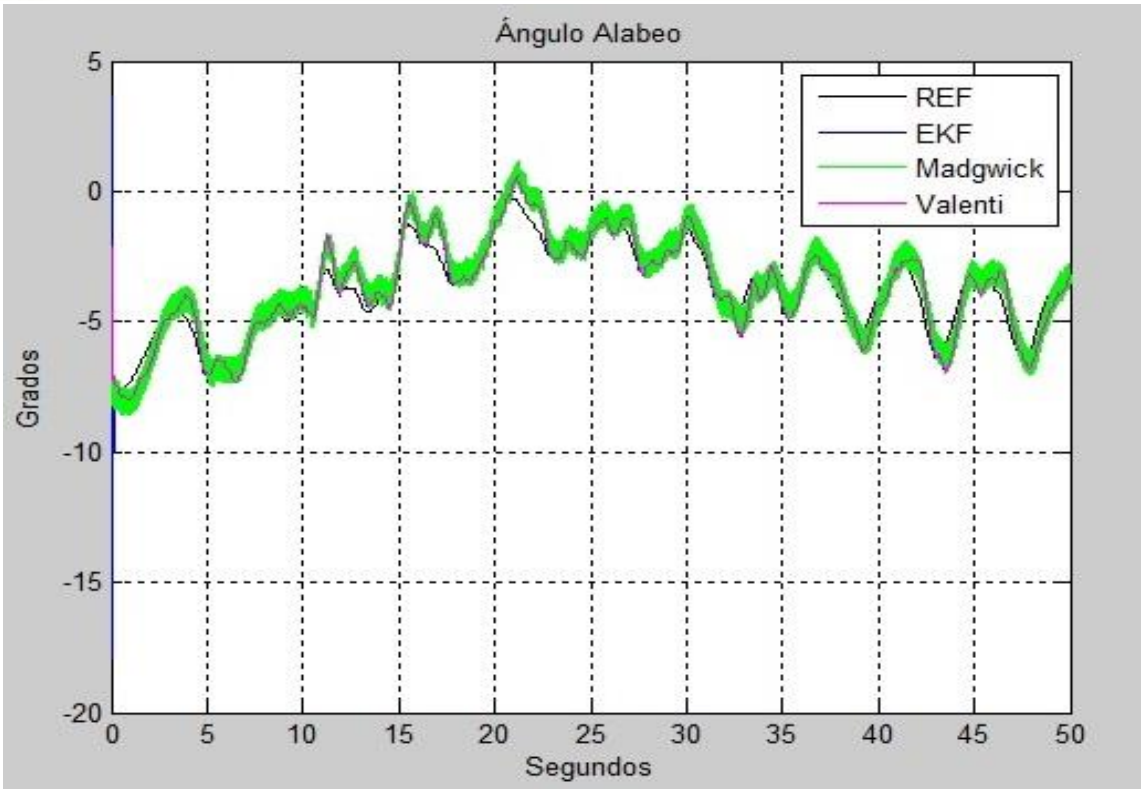

Figura 4: Graficación de resultados por algoritmos para el ángulo alabeo

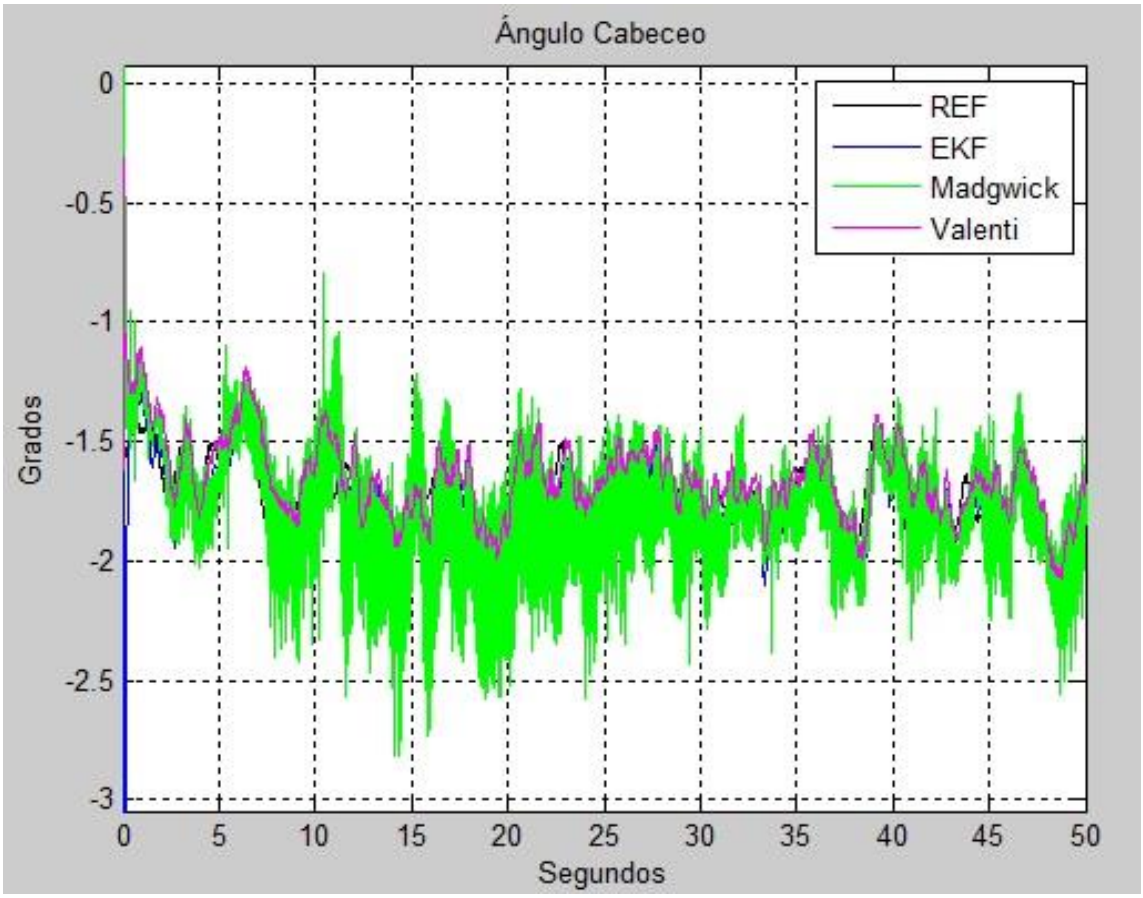

Figura 5: Graficación de resultados por algoritmos para el ángulo cabeceo

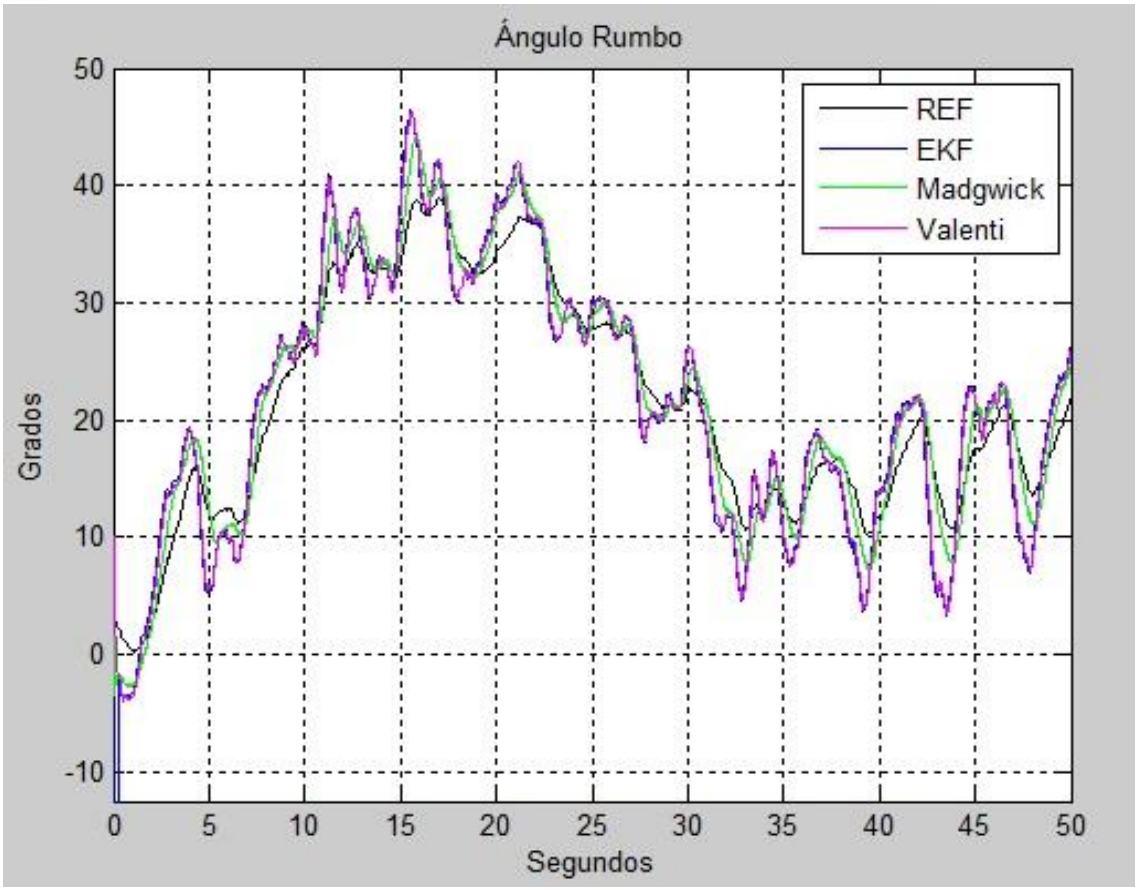

Figura 6: Graficación de resultados por algoritmos para el ángulo rumbo.

Del análisis de las gráficas se puede concluir que este criterio de comparación para el ángulo alabeo y cabeceo sugiere que **los algoritmos Filtro Extendido de Kalman Watson (2013) y Filtro Complementario propuesto por Madgwick (2010) y Filtro Complementario según Valenti (2015)** tienen resultados muy similares entre ellos y también con respecto a la referencia, pero es necesario destacar que para estos ángulos el algoritmo Filtro Complementario según Madgwick (2010) se muestra muy inestable producto a que oscila mucho.

Similares resultados se repiten para el ángulo rumbo, o sea, hay un gran parecido entre los resultados de los propios algoritmos implementados y la orientación real del vehículo y se hace evidente la estabilidad para este ángulo del algoritmo Filtro Complementario según Madgwick (2010), de lo que se concluye que **según el criterio de la graficación los tres algoritmos implementados no representan grandes diferencias entre ellos** de ahí que compararlos a simple vista se torna complicado e insuficiente y se necesita de otros criterios de comparación para arribar a una conclusión final.

3.4 Análisis de Varianza. Test de Friedman

Para la validación de los resultados se hará uso de técnicas de análisis de varianza no paramétricas ya que estas son las que se deben aplicar cuando los supuestos de normalidad, homogeneidad de las varianzas, independencia de los errores, y/o aditividad de los efectos no se cumplen. Específicamente se hará uso de pruebas para k variables relacionadas y dentro de estas la prueba de Friedman.

La **prueba de Friedman** (también conocida como **ANOVA de Friedman**) es la alternativa no paramétrica al modelo ANOVA de una vía con medidas repetidas para comparar más de dos grupos dependientes. (Gardener, 2017)

Fue desarrollada por el economista Milton Friedman. En cuanto a las comparaciones de dos clasificadores, esta prueba tiene teóricamente menos potencia que el ANOVA (paramétrico) cuando se cumplen los supuestos del ANOVA, pero esto no tiene que ser el caso cuando no lo son.

Esta prueba puede utilizarse en aquellas situaciones en las que se seleccionan n elementos y k grupos. El arreglo en bloques consiste en colocar los datos en una tabla de doble entrada de n filas y k columnas, por regla n> 10 y k>5. (Demsar, 2006). El método consiste en ordenar los datos por filas o bloques, reemplazándolos por su respectivo orden. (Gardener, 2017)

Hipótesis:

- **H_0**: No existen diferencias entre los algoritmos.
- **H_1**: Existen diferencias entre los algoritmos.

Para resolver el contraste de hipótesis anterior, Friedman propuso un estadístico que se distribuye como una Chi-cuadrado; se calcula mediante la siguiente expresión.

Estadístico de prueba:

$$X_r^2 = \frac{12}{nk(k+1)} \sum R_j^2 - 3(k+1) \qquad (62)$$

Donde:

- X_r^2 = estadístico calculado del análisis de varianza por rangos de Friedman.
- n = representa el número de elementos o de bloques (número de hileras o filas).
- k = el número de variables relacionadas
- $\sum R_j^2$ = es la suma de rangos por columnas al cuadrado.

Conforme aumenta la cantidad de bloques en el experimento se puede aproximar el estadístico de Friedman a una distribución $X^2\ con\ k-1$ grados de libertad.

Para la ejecución de la prueba se seleccionó una muestra de 30 observaciones de una población de 65000. También se ejecutó 3 veces la misma prueba para tener los resultados por ángulos (alabeo, cabeceo y rumbo). Para la realización de la prueba, se hizo uso de la herramienta informática RStudio.

Iteración I. Test de Friedman con 30 observaciones

- **Ejecución de la prueba para el ángulo alabeo**

	Ref	EKF	FC.Madgwick	FC.Valenti
1	-7.186317	-7.186317	-7.186317	-7.186317
2	-7.202438	-6.837573	-7.461081	-2.162077
3	-7.220996	-7.273620	-7.566848	-3.674754
4	-7.226152	-5.815696	-7.455746	-4.731032
5	-7.256397	-6.390967	-7.836601	-5.473826
6	-7.266244	3.667316	-7.151791	-5.992552
7	-7.284067	-17.881526	-8.095959	-6.356366
8	-7.303361	-12.108126	-7.191812	-6.611847
9	-7.319078	-9.115445	-8.136786	-6.795500
10	-7.328783	-8.012554	-7.232200	-6.923407
11	-7.355174	-7.312181	-8.177513	-7.026453
12	-7.369491	-7.706358	-7.273660	-7.105226
13	-7.380771	-7.674400	-8.218956	-7.165454
14	-7.392647	-7.947344	-7.315242	-7.211877
15	-7.401275	-8.168186	-8.259683	-7.244331
16	-7.418008	-8.563387	-7.355396	-7.273490
17	-7.434349	-9.052919	-8.297965	-7.298904
18	-7.435395	-9.603592	-7.396343	-7.322114
19	-7.438659	-10.009617	-8.305426	-7.337384
20	-7.440877	-9.967332	-7.414419	-7.348236
21	-7.449509	-9.497875	-7.809387	-7.353165
22	-7.456243	-8.903336	-7.184066	-7.359960
23	-7.469880	-8.407299	-8.127618	-7.372572
24	-7.475124	-8.030384	-7.224168	-7.373822
25	-7.482272	-7.799338	-8.171659	-7.378429
26	-7.478649	-7.647398	-7.265535	-7.381485
27	-7.488189	-7.574441	-8.205009	-7.392852
28	-7.504493	-7.554274	-7.300926	-7.416050
29	-7.507764	-7.535407	-8.248476	-7.436190
30	-7.505186	-7.511720	-7.339839	-7.446846

Figura 7: Tabla de 30 observaciones generada por RStudio del ángulo alabeo.

- **Resultado del Test de Friedman para el ángulo alabeo**

Para el análisis de los resultados se debe conocer que generalmente para este test, por defecto $\alpha = 0.05$ y que para saber si se acepta o rechaza la hipótesis nula se debe comparar el valor de α con el p-value (pv) resultante de la prueba, de la siguiente forma: (Gardener, 2017)

- Si pv es menor que o igual a α. **Rechazar H_0.**
- Si pv es mayor que el nivel de significancia (α). **No se puede rechazar H_0.**

Lo que significa que para el nivel de significancia (α=0.05) si pv es menor o igual a 0.05, se debe rechazar la hipótesis nula.

Al aplicar la prueba de Friedman en la herramienta RStudio (Prueba 1), se obtuvo como resultado al ángulo alabeo $X^2 = 45.0414$, 3 grados de libertad y $\text{pv} = 9.067\text{e} - 10$. Como este valor pv es menor que α se debe rechazar la hipótesis nula. De este resultado se interpreta que se puede asegurar con un 95% de confianza que existen diferencias significativas entre los algoritmos para el ángulo alabeo.

```
> friedman.test(alabeomatrix)

 Friedman rank sum test

data:  alabeomatrix
Friedman chi-squared = 45.0414, df = 3, p-value = 9.067e-10
```

Prueba 1: Resultado del Test de Friedman generado en RStudio para el ángulo alabeo.

Ejecución de la prueba para el ángulo cabeceo

	Ref	EKF	FC.Madgwick	FC.Valenti
1	-1.617591	-1.617591	-1.6175911	-1.6175911
2	-1.620907	-4.425486	0.6975322	-0.3104150
3	-1.626135	-9.924799	-0.2617305	-0.5307861
4	-1.604102	-21.154693	-1.1987996	-0.6828018
5	-1.617347	-42.563593	-0.4489977	-0.7950449
6	-1.600191	-70.856942	-1.0713956	-0.8721001
7	-1.581106	-81.415057	-1.1403399	-0.9236360
8	-1.582847	-53.290463	-1.1134685	-0.9649635
9	-1.578967	-29.192492	-1.1494242	-0.9942866
10	-1.583536	-15.106700	-1.1342659	-1.0217561
11	-1.574339	-8.212601	-1.1725679	-1.0401168
12	-1.573200	-9.539107	-1.1441645	-1.0550650
13	-1.568253	-8.975808	-1.1878146	-1.0666110
14	-1.561138	-9.132212	-1.1548800	-1.0760880
15	-1.548941	-8.970375	-1.1812951	-1.0828299
16	-1.550214	-8.791103	-1.1588088	-1.0909839
17	-1.540793	-8.391232	-1.2486471	-1.0987220
18	-1.530126	-7.619418	-1.1836312	-1.1000348
19	-1.543673	-6.411530	-1.4483044	-1.1174050
20	-1.539690	-4.880355	-1.3035590	-1.1267508
21	-1.543907	-3.509715	-0.4784971	-1.1366709
22	-1.544893	-2.576017	-1.1639029	-1.1491412
23	-1.532529	-2.010820	-1.2749742	-1.1514265
24	-1.519019	-1.689897	-1.2131399	-1.1493675
25	-1.521171	-1.528296	-1.2600086	-1.1549794
26	-1.506483	-1.440649	-1.2227799	-1.1587074
27	-1.511547	-1.417282	-1.3605496	-1.1750895
28	-1.510937	-1.412474	-1.2959020	-1.1912049
29	-1.501966	-1.408409	-1.2746829	-1.2020394
30	-1.495469	-1.415675	-1.2731958	-1.2155295

Figura 8: Tabla de 30 observaciones generada por RStudio del ángulo cabeceo

Resultado del Test de Friedman para el ángulo cabeceo

Al aplicar la prueba de Friedman en la herramienta RStudio (Prueba 2), se obtuvo como resultado al ángulo cabeceo $X^2 = 77.8866$, 3 grados de libertad y $\text{pv} < 2.2\text{e} - 16$. Como este valor pv es menor que α se debe rechazar la hipótesis nula. De este resultado se interpreta que se puede asegurar con un 95% de confianza que existen diferencias significativas entre los algoritmos para el ángulo cabeceo.

```
> friedman.test(cabeceomatrix)

 Friedman rank sum test

data:  cabeceomatrix
Friedman chi-squared = 77.8966, df = 3, p-value < 2.2e-16
```

Prueba 2: Resultado del Test de Friedman generado en RStudio para el ángulo cabeceo.

Ejecución de la prueba para el ángulo rumbo.

	Ref	EKF	FC.Madgwick	FC.Valenti
1	3.471278	3.471278	3.471278	3.4712782
2	3.431478	-3.434736	-3.392845	6.7860174
3	3.377416	4.749625	-3.323241	9.5881052
4	3.382020	-6.101146	-3.253306	10.0769925
5	3.292612	2.544281	-2.839549	9.3435916
6	3.279819	-14.075485	-2.856366	8.0650950
7	3.248708	-166.155895	-2.666553	6.6305692
8	3.217130	-172.670318	-2.725310	5.2485849
9	3.148625	-172.833517	-2.533312	4.0110072
10	3.117646	-171.218940	-2.596367	2.9585265
11	2.995016	-170.691175	-2.409594	2.0718128
12	2.919149	-167.839496	-2.478639	1.3368921
13	2.859061	-164.424848	-2.296859	0.7354983
14	2.809041	-158.791254	-2.369313	0.2477512
15	2.783775	-150.884249	-2.182501	-0.1395511
16	2.726837	-139.453060	-2.257558	-0.4520356
17	2.683581	-123.649086	-2.080044	-0.7070727
18	2.635762	-103.010110	-2.157943	-0.9186181
19	2.607440	-78.993587	-1.976884	-1.0883397
20	2.580240	-55.190510	-2.056020	-1.2219411
21	2.569149	-35.708105	-1.944870	-1.3221124
22	2.525789	-22.278813	-1.987766	-1.4017405
23	2.470048	-13.958668	-1.851017	-1.4757667
24	2.479866	-8.952259	-1.930794	-1.5279806
25	2.435923	-6.025593	-1.793009	-1.5699906
26	2.414265	-4.278870	-1.874620	-1.6024406
27	2.352965	-3.293866	-1.745904	-1.6420156
28	2.275639	-2.796308	-1.833022	-1.7049687
29	2.246380	-2.515430	-1.722254	-1.7789514
30	2.245954	-2.318678	-1.809030	-1.8458200

Figura 9: Tabla de 30 observaciones generada por RStudio del ángulo rumbo

Resultado del Test de Friedman para el ángulo rumbo

Al aplicar la prueba de Friedman en la herramienta RStudio (Prueba 3), se obtuvo como resultado al ángulo cabeceo $X^2 = 71.6069$, 3 grados de libertad y $\text{pv} = 1.933\text{e} - 15$. Como este valor pv es menor que α se debe rechazar la hipótesis nula. De este resultado se interpreta que se puede asegurar con un 95% de confianza que existen diferencias significativas entre los algoritmos para el ángulo rumbo.

```
> friedman.test(rumbomatrix)

 Friedman rank sum test

data:  rumbomatrix
Friedman chi-squared = 71.6069, df = 3, p-value = 1.933e-15
```

Prueba 3: Resultado del Test de Friedman generado en RStudio para el ángulo rumbo.

- **Iteración II. Test de Friedman con 1000 observaciones**

Esta vez se ejecuta el mismo test, pero variando el tamaño de la muestra, para la que se seleccionan esta vez 1000 observaciones.

Ejecución de la prueba para el ángulo alabeo.

Al aplicar la prueba de Friedman en la herramienta RStudio (Prueba 4), se obtuvo como resultado al ángulo alabeo con 1000 observaciones $X^2 = 277.8223$, 3 grado de libertad y $\text{pv} = 2.2\text{e} - 16$. Como este valor pv es menor que α se debe rechazar la hipótesis nula. De este resultado se interpreta que se puede asegurar con un 95% de confianza que existen diferencias significativas entre los algoritmos para el ángulo alabeo.

```
> friedman.test(alabeomatrix1000)

 Friedman rank sum test

data:  alabeomatrix1000
Friedman chi-squared = 277.8223, df = 3, p-value < 2.2e-16
```

Prueba 4: Resultado del Test de Friedman generado en RStudio para el ángulo alabeo con 1000 observaciones.

Ejecución de la prueba para el ángulo cabeceo.

Al aplicar la prueba de Friedman en la herramienta RStudio (Prueba 5), se obtuvo como resultado al ángulo cabeceo con 1000 observaciones $X^2 = 4406.821$, 3 grado de libertad y $\text{pv} = 2.2\text{e} - 16$. Como este valor pv es menor que α se debe rechazar la hipótesis nula. De este resultado se interpreta que se puede asegurar con un 95% de confianza que existen diferencias significativas entre los algoritmos para el ángulo cabeceo.

```
> friedman.test(cabeceomatrix1000)

 Friedman rank sum test

data:  cabeceomatrix1000
Friedman chi-squared = 4406.821, df = 3, p-value < 2.2e-16
```

Prueba 5: Resultado del Test de Friedman generado en RStudio para el ángulo cabeceo con 1000 observaciones.

Ejecución de la prueba para el ángulo Rumbo

Al aplicar la prueba de Friedman en la herramienta RStudio (Prueba 6), se obtuvo como resultado al ángulo rumbo con 1000 observaciones $X^2 = 287.2063$, 3 de grado de libertad y $\text{pv} = 2.2\text{e} - 16$. Como este valor pv es menor que α se debe rechazar la hipótesis nula. De este resultado se interpreta que se puede asegurar con un 95% de confianza que existen diferencias significativas entre los algoritmos para el ángulo rumbo.

```
> friedman.test(rumbomatrix1000)

 Friedman rank sum test

data:  rumbomatrix1000
Friedman chi-squared = 287.2063, df = 3, p-value < 2.2e-16
```

Prueba 6: Resultado del Test de Friedman generado en RStudio para el ángulo rumbo con 1000 observaciones.

Al aplicar el Test Friedman se puede concluir que tanto en la 1era como en la 2da iteración existen diferencias significativas en cada ángulo para los diferentes conjuntos de datos probados en los tres algoritmos implementados, esto se puede asegurar con un 95 % de confianza por lo que es necesario recurrir al análisis post-hoc.

3.5 Test de Nemenyi

Una vez que la prueba de Friedman indica diferencia en al menos 2 algoritmos para los ángulos alabeo, cabeceo y rumbo se está interesado en identificar cuál grupo o grupos son significativamente diferentes. Para esto se hará uso de la prueba de Nemenyi.

En la estadística, la prueba de Nemenyi es una prueba post-hoc destinada a encontrar los grupos de datos que difieren después de que una prueba estadística de comparaciones ha rechazado la hipótesis nula y que el rendimiento de las comparaciones en los grupos de los datos es similar. (Pohlert, 2016).

Sobre la base del rechazo, la prueba post-hoc de Nemenyi se utilizó para comparar todos los clasificadores entre sí. Esta prueba considera que el rendimiento de dos clasificadores se considera significativamente diferente si sus rangos medios difieren al menos por la diferencia crítica (CD por sus siglas en inglés). (Demsar, 2006)

$$CD = q\sqrt{\frac{K(K+1)}{6D}} \quad (63)$$

Donde k y D son el número de algoritmos y observaciones o conjunto de datos respectivamente y el valor “q” es derivado por el rango estadístico estudiado dividido por la raíz cuadrada de dos.

- **Iteración I.**

La implementación de la prueba en la herramienta RStudio muestra los siguientes resultados:

Ángulo alabeo

```
> posthoc.friedman.nemenyi.test(alabeomatrix)

 Pairwise comparisons using Nemenyi multiple comparison test
             with q approximation for unreplicated blocked data

data:  alabeomatrix

            Ref     EKF     FC.Madgwick
EKF         0.09799 -       -
FC.Madgwick 0.97834 0.22798 -
FC.Valenti  0.00024 9.3e-10 4.0e-05

P value adjustment method: none
```

Prueba 7: Resultado del Test de Nemenyi generado en RStudio para el ángulo alabeo. Iteración I.

En la columna que se señala en rojo se puede observar para el ángulo alabeo las diferencias que existen entre la orientación real (columna) y cada una de las orientaciones estimadas por los algoritmos implementados (filas) de ahí que entre menor sea el valor del resultado de un algoritmo mejor estimación de la orientación este brinda puesto que tiene una menor diferencia con respecto a la orientación real.

Para el ángulo alabeo, este test sugiere en su primera iteración que el algoritmo que más se acerca a la orientación real es el **Filtro Complementario según Valenti (2015) y el Filtro Complementario propuesto por Madgwick (2010).**

Ángulo cabeceo

```
> posthoc.friedman.nemenyi.test(cabeceomatrix)

 Pairwise comparisons using Nemenyi multiple comparison test
            with q approximation for unreplicated blocked data

data:  cabeceomatrix

            Ref     EKF     FC.Madgwick
EKF         0.22798 -       -
FC.Madgwick 0.00083 7.2e-08 -
FC.Valenti  2.2e-08 6.0e-14 0.15305

P value adjustment method: none
```

Prueba 8: Resultado del Test de Nemenyi generado en RStudio para el ángulo cabeceo. Iteración I.

Parecido sucede para el ángulo cabeceo, donde este test sugiere en su primera iteración que el algoritmo que más se acerca a la orientación real es **el Filtro Complementario según Valenti (2015) pero el que más se aleja del resultado real es el Filtro Extendido de Kalman según Watson (2013).**

Ángulo rumbo

```
> posthoc.friedman.nemenyi.test(rumbomatrix)

 Pairwise comparisons using Nemenyi multiple comparison test
            with q approximation for unreplicated blocked data

data:  rumbomatrix

            Ref     EKF     FC.Madgwick
EKF         4.1e-13 -       -
FC.Madgwick 5.7e-06 0.0460  -
FC.Valenti  0.4993  6.4e-09 0.0026

P value adjustment method: none
```

Prueba 9: Resultado del Test de Nemenyi generado en RStudio para el ángulo rumbo. Iteración I.

Por su parte, para el ángulo rumbo, este test sugiere que el algoritmo que más se acerca a la orientación real **es el Filtro Extendido de Kalman según Watson (2013) y que es el Filtro Complementario propuesto por Valenti (2015) el que más se aleja del resultado real de la orientación.**

Esta primera iteración concluye que **el algoritmo que mejor estimación representa de los tres implementados es el Filtro Complementario según Valenti (2015)** porque es el que más se acerca a la orientación real del Experimento realizado con el Vehículo Autónomo Subacuático HRC-AUV en el 2014 en la bahía de La Habana.

Para probar con otro conjunto de datos se realizará una segunda iteración del mismo test y se compararán los resultados obtenidos entre iteraciones.

- **Iteración II. Test de Friedman con 1000 observaciones**

Se repite el test, pero esta vez con un tamaño de muestra igual a 1000 observaciones.

Ángulo alabeo

```
> posthoc.friedman.nemenyi.test(alabeomatrix1000)

 Pairwise comparisons using Nemenyi multiple comparison test
            with q approximation for unreplicated blocked data

data:  alabeomatrix1000

            Ref     EKF     FC.Madgwick
EKF         0.0141  -       -
FC.Madgwick < 2e-16 < 2e-16 -
FC.Valenti  4.7e-14 7.1e-15 0.0066

P value adjustment method: none
```

Prueba 10: Resultado del Test de Nemenyi generado en RStudio para el ángulo alabeo con 1000 observaciones. Iteración II.

El procedimiento para comparar los resultados es el mismo que en la Iteración I, se analizan sólo los resultados de la fila marcada en rojo que es la que tiene las comparaciones con la orientación real que es con lo que se desean compara los resultados estimados de cada algoritmo implementado.

Para el ángulo alabeo, en la segunda iteración, este test sugiere que **el algoritmo que más se acerca a la orientación real es el Filtro Complementario propuesto por Madgwick (2010) y el que más se aleja es el Filtro Extendido de Kalman según Watson (2013).**

Ángulo cabeceo

```
> posthoc.friedman.nemenyi.test(cabeceomatrix1000)

 Pairwise comparisons using Nemenyi multiple comparison test
            with q approximation for unreplicated blocked data

data:  cabeceomatrix1000

             Ref      EKF      FC.Madgwick
EKF          < 2e-16  -        -
FC.Madgwick 7.1e-08  < 2e-16  -
FC.Valenti   < 2e-16  < 2e-16  < 2e-16

P value adjustment method: none
```

Prueba 11: Resultado del Test de Nemenyi generado en RStudio para el ángulo cabeceo con 1000 observaciones. Iteración II.

Para el ángulo cabeceo, en la segunda iteración este test sugiere que **los algoritmos que más se acercan a la orientación real son El Filtro Extendido de Kalman propuesto por Watson (2013) y el Filtro Complementario según Valenti (2015), y por su parte el que más se aleja es el Filtro Complementario propuesto por Madgwick (2010).**

- **Ángulo rumbo**

```
> posthoc.friedman.nemenyi.test(rumbomatrix1000)

 Pairwise comparisons using Nemenyi multiple comparison test
            with q approximation for unreplicated blocked data

data:  rumbomatrix1000

             Ref      EKF     FC.Madgwick
EKF          < 2e-16  -       -
FC.Madgwick < 2e-16  0.2004  -
FC.Valenti   4.2e-14  0.0028  3.0e-07

P value adjustment method: none
```

Prueba 12: Resultado del Test de Nemenyi generado en RStudio para el ángulo rumbo con 1000 observaciones. Iteración II.

En el ángulo rumbo, en la segunda iteración, este test sugiere que **los algoritmos que más se acercan a la orientación real son el Filtro Extendido de Kalman según Watson (2013) y el Filtro Complementario según Madgwick (2010) siendo el Filtro Complementario propuesto por Valenti (2015) el que más se aleja.**

Para esta segunda iteración el comportamiento de los datos se ha visto más inestable indicando que cada algoritmo implementado es el menos indicado en al menos un ángulo, pero el que mejor estimación de la orientación supone es el Filtro Complementario según Madgwick (2010) siendo el seleccionado para los ángulos alabeo y rumbo en esta iteración.

Teniendo en cuenta ambas iteraciones, **se concluye que este criterio estadístico de comparación sugiere que los algoritmos Filtro Extendido de Kalman según Watson (2013) y el Filtro Complementario según Valenti (2015) tienen similaridades y son los que más se acercan a la orientación real**, esto lo demuestra que el primero significa un mejor resultado para la estimación de la orientación en los ángulos rumbo de la iteración I y II y cabeceo en la iteración II. Por su parte, el segundo representa una mejor estimación para los ángulos alabeo en la iteración I y cabeceo en la iteración I y II.

3.6 Tiempo de ejecución

Calcular el tiempo que tarda un programa en ser ejecutado por una computadora puede ser difícil de medir, debido a los Sistemas Operativos Multitarea, que tienen tiempos de respuesta que son independientes de la frecuencia de reloj del ordenador. Por ello es necesario diferenciar entre el tiempo que tarda una CPU en ejecutar el código de un programa, el tiempo que utiliza el Sistema Operativo. para realizar sus tareas, y el tiempo necesario para acceder a los dispositivos de entrada-salida. El tiempo de ejecución de un programa puede quedar dividido como sigue:

- Tiempo de respuesta
- Tiempo de CPU a su vez se divide en:
  - Tiempo de CPU utilizado por el usuario
  - Tiempo de CPU utilizado por el Sistema Operativo

A continuación, se presenta el tiempo de ejecución de cada algoritmo, este criterio de comparación se basa en seleccionar el algoritmo que menos tiempo demore en dar respuesta como el mejor.

Profile Summary

Generated 01-Jun-2017 11:26:07 using cpu time.

Function Name	Calls	**Total Time**	Self Time*	Total Time Plot (dark band = self time)
ahrs_CF_Valenti	4999	1.354 s	0.717 s	
...MadgwickAHRS_	4999	1.133 s	0.668 s	
ahrsEKF	4999	0.723 s	0.723 s	

Figura 10: Tabla con el tiempo de ejecución de cada algoritmo implementado.

Se concluye que **el algoritmo con menor tiempo de ejecución es el Filtro Extendido de Kalman Watson (2013)**, por lo que según este criterio este es el algoritmo con mejor resultado.

3.7 Valor Medio

Otro criterio a tener en cuenta para comparar el desempeño de los algoritmos implementados es el valor medio. Este no es más que calcular el promedio de los resultados arrojados por cada algoritmo.

Para seleccionar qué algoritmo tiene mejores resultados basta con calcular el valor medio de cada uno y escoger el que tenga un valor medio más cercano al valor medio real. La herramienta informática MatLab muestra los siguientes resultados por ángulos:

	1	2	3	4	5
1	'Algorithm'	'REF'	'EKF'	'Madgwick'	'Valenti'
2	'Mean_Roll'	-3.7446	-3.7428	-3.5747	-3.7160
3	'Mean_Pitch'	-1.7270	-1.8377	-1.7868	-1.6538
4	'Mean_Yaw'	21.2361	20.9973	21.7073	21.3916

Figura 11: Tabla con el valor medio de cada algoritmo implementado

De lo que se concluye que para el ángulo alabeo el algoritmo con mejores resultados para la estimación de la orientación según el criterio del valor medio es Filtro Extendido de Kalman propuesto por Watson (2013), no siendo así para el ángulo cabeceo ni rumbo donde el que arroja un resultado más cercano al real para el primero es el Filtro Complementario propuesto por Madgwick (2010) y para el segundo es el algoritmo de Filtro Complementario sugerido por Valenti (2015). Concluyendo de esta forma que no existe un algoritmo que sea recomendado para al menos dos ángulos para este criterio de comparación, lo que significa cualquiera de los algoritmos implementados representa en al menos un ángulo la mejor estimación de la orientación.

3.8 Error medio cuadrático

Una prueba implementada y aplicada para verificar del resultado es el cálculo del error medio cuadrático.

El cálculo del error contenido en la gráfica resulta de la comparación, para el conjunto de los puntos de comprobación empleados, entre sus coordenadas verdaderas (referencia) y las que se obtienen del resultado de los algoritmos. A partir de las discrepancias observadas se puede calcular:

Error medio cuadrático o RMSE (*Root Mean Square Error*):

$$RMSE_x = \sqrt{\frac{\sum_{i=1}^{n} e_{xi}^2}{N}} \quad (64)$$

$$RMSE_y = \sqrt{\frac{\sum_{i=1}^{n} e_{yi}^2}{N}} \quad (65)$$

$$RMSE_Z = \sqrt{\frac{\sum_{i=1}^{n} e_{zi}^2}{N}} \quad (66)$$

El RMSE es un indicador muy empleado, ya que evalúa de modo conjunto la cuantía del sesgo (indicado por el error medio) como de la dispersión (indicado por la desviación típica de los errores) del error de posición en la imagen. Resulta interesante resaltar que cuando el error medio tiende a cero el RMSE se aproxima a la desviación típica de los errores. Por lo tanto, bajo la hipótesis de que el error de la gráfica sigue una distribución normal, puede afirmarse que el RMSE corresponde aproximadamente al 68 % de probabilidad (el 68 % de los errores en la imagen tiene un valor inferior al RMSE). (Rico-Azagra, Rico, Maisterra, & Gil-Martínez, 2015)

Al calcular el error medio cuadrático por ángulos la herramienta MatLab muestra los siguientes resultados:

	1	2	3	4	5
1	'Algorithm'	'REF'	'EKF'	'Madgwick'	'Valenti'
2	'RMSE_Roll'	0	0.5424	0.6792	0.5303
3	'RMSE_Pitch'	0	1.8669	0.2497	0.1281
4	'RMSE_Yaw'	0	8.7770	1.9772	3.4087

Figura 12: Tabla con el error medio cuadrático de cada algoritmo implementado.

Para comparar el desempeño de los algoritmos según este criterio es necesario conocer que entre menor sea el error medio cuadrático mejores resultados supone, ya que es este el que más se acercará a la orientación real, o sea: la referencia, de ahí que se concluya que según el criterio utilizado, **el algoritmo con mejores prestaciones para el ángulo alabeo y cabeceo es el Filtro Complementario propuesto por Valenti (2015) y para el ángulo rumbo es el algoritmo de Filtro Complementario según Madgwick (2010).**

Por mayoría, según este criterio el algoritmo que mejor estimación de la orientación supone es el **Filtro Complementario sugerido por Valenti (2015).**

3.9 Conclusiones de capítulo

- Se conformó el escenario de prueba quedando compuesto por seis criterios para comparar y validar los resultados obtenidos por los tres algoritmos implementados, estos criterios fueron: las gráficas, Test de Friedman, Test de Nemenyi, tiempo de ejecución, valor medio y error medio cuadrático.
- Se utilizó la herramienta RStudio y el lenguaje R para hacer las pruebas estadísticas: Test de Friedman y Test de Nemenyi.
- Se utilizó la herramienta Matlab y su lenguaje propio M hacer las pruebas mediante los criterios: gráficas, valor medio, tiempo de ejecución y error medio cuadrático.
- Los criterios de comparación mediante gráficas, y el valor medio indican que los tres algoritmos implementados representan una solución adecuada al problema de la estimación de la orientación con sensores de bajo costo, a pesar de que el algoritmo de Filtro Complementario según Madgwick (2010) se muestra inestable en las gráficas de los ángulos alabeo y cabeceo, no así para rumbo.
- Se hicieron dos iteraciones de las pruebas estadísticas: Test de Friedman y Test post-hoc de Nemenyi, para comparar los resultados de cada test con diferentes conjuntos de datos.

  a) El Test estadístico de Friedman prueba que existen diferencias significativas entre los resultados de los algoritmos implementados y la orientación real en ambas iteraciones y para cada ángulo.

  b) El Test post- hoc de Nemenyi determina que los algoritmos Filtro Extendido de Kalman según Watson (2013) y Filtro Complementario según Valenti (2015) son los que menos diferencia implican en comparación con la orientación real y por tanto los que mejor estimación de la orientación obtienen.

- El criterio de error medio cuadrático indica que el Filtro Complementario según Valenti (2015) es el algoritmo que mejor estima orientación.
- Según el criterio de tiempo de ejecución el algoritmo que más rápido se ejecuta es el Filtro Extendido de Kalman Watson (2013).
- Teniendo en cuenta el resultado obtenido por el criterio estadístico Test de Nemenyi, y que para la investigación tiene más peso cuál algoritmo estima mejor la orientación a través del que menor error medio cuadrático tenga que a través de su tiempo de ejecución se concluye que es el algoritmo Filtro Complementario propuesto por Valenti (2015) el que ofrece mejores resultados cuando se utilizan mediciones provenientes de sensores de bajo costo.

CONCLUSIONES GENERALES

- El estudio de investigaciones precedentes sobre la estimación de la orientación contribuyó significativamente a la comprensión de los algoritmos de estimación de la orientación en Vehículos Autónomos Subacuáticos.
- A partir de la implementación de los algoritmos seleccionados se estima la orientación al utilizar las mediciones de los sensores de bajo costo utilizando Matlab y su lenguaje de programación.
- El proceso de validación y prueba permitió:
    - Comprobar la capacidad de cada algoritmo al estimar la orientación con mediciones procedentes de sensores inerciales de bajo costo.
    - Propiciar las evidencias matemáticas que permiten la comparación entre los algoritmos implementados
    - Demostrar que el Filtro Complementario según Valenti (2015), es el que mejores resultados tiene a partir de las mediciones de giróscopos, acelerómetros y magnetómetros, así como la referencia absoluta obtenidos del vehículo subacuático HRC-AUV en el 2014.

RECOMENDACIONES

Al término de este trabajo de diploma, se considera que es posible trabajar en las siguientes direcciones:

- Aumentar el número de algoritmos a evaluar, que sigan estrategias de estimación diferentes.
- Identificar un método de ajuste óptimo para los algoritmos, en función de las características de los sensores utilizados.
- Implementar los algoritmos en escenarios de prueba con prototipos reales.
- Incorporar conjuntos de mediciones proveniente de otros tipos de Vehículos Autónomos, con sensores de diferente fabricantes.

BIBLIOGRAFÍA

1. Aravena, I. C., & Vera, J. J. (2012). Modelo de un Control Aproximado de un Cuadricóptero Basado en Equilibrio de un Balancín.
2. Ayub, S., Bahraminisaab, A., & Bahram, H. (2012). A sensor fusion method for smart phone orientention estimation.
3. Baerveldt, A., & Klang., R. (2002). A low-cost and low-weight attitude estimation system for an autonomous helicopter. *IEEE International Conference on*.
4. Baturone, A. O. (2011). Robótica, Manipuladores y Robots Móviles. *Xataka Ciencia*.
5. Bin Fang, W. C. (2011). Attitude estimation of rigid bodies using MEMS inertial sensors. *Fourth International Conference on Intelligent Computation Technology and Automation.* Beijing, China.
6. Buskey, G., Roberts, J., Corke, P., & Wyeth, G. (2004). Helicopter automation using a low-cost sensing system. *Computing & Control Engineering Journal*.
7. Carpio, T. (2003). *Tecomunica.* Obtenido de http://tonicarpio.com/periodismo/submarinos-autonomos
8. Caruso, M. J. (2000). Applications of magnetic sensors for low cost compass systems.
9. CENDEJAS VALDÉZ, J. L. (2014). *Implementación Del Modelo Integral Colaborativo.*
10. Christian Eling, L. K. (2015). Real-Time Single-Frequency GPS/MEMS-IMU Attitude Determination of Lightweight UAVs. *Sensors*.
11. Coad, P., & Lefebure, E. (2000). *Java modeling in color with UML.*
12. Cockbun, A. (2001). *Agile software development. Addison-Wesley.*
13. Daniel Alpaya, F. C. (2014). An extension of Herglotz's theorem to the quaternions.
14. Dawson, R. (2001). Programming in Ansi C. Third edition. *Loughborough University Institutional Repository*.
15. Demsar, J. (2006). Statistical Comparisons of Classifiers over Multiple Data Sets.
16. Derek B. Kingston, R. W. (2004). Real-Time Attitude and Position Estimation for Small UAVs Using Low-Cost Sensors.

17. España, M. (2010). Fundamentos de la Navegación Integrada. *Asociación Argentina de Control Automático (AADECA)*.

18. Esteban, E. v. (2010). El lenguaje de programación C.

19. Fernández, M. O. (2005). Algoritmos de búsqueda heurística en tiempo real.

20. Ferrer Mínguez, G. (2009). Integración Kalman de sensores inerciales INS con GPS en un UAV.

21. Fons, A. B., & Vila, C. A. (2012). Integration of the ins and gps navigation systems using Kalman filter on a mobile phone.

22. Fossen, T. (2011). Handbook of marine craft hydrodynamics and motion control.

23. García de Jalón, J., Rodríguez, J. I., & Vidal, J. (2005). *Aprenda Matlab como si estuviera en primero.*

24. García, J. G. (2011). *Visión Artificial aplicada en vehículos autónomos submarinos.*

25. Gardener, D. M. (2017). Using R for statistical analyses - Non-parametric stats. *GardenerSown*.

26. Gaydou, D., Redolfi, J., & Henze, A. (2011). *Filtro complementario para estimacion de actitud aplicado al controlador embebido de un cuatrirrotor.*

27. Goldstein, H. (2006). *Mecánica Clásica.*

28. Gómez, R. C. (2016). Diseño de un cuadricóptero de bajo coste basado en herramientas de software libre.

29. Guerrero González, A., López Maestre, T., Gilabert Cervera, J., García-Vidal Simón, J., & González Reloid, I. (2010). *Vehículo submarino autónomo para trabajos oceanográficos en aguas costeras.* Obtenido de http://www.jornadassarteco.org/js2012/papers/paper_91.pdf

30. Guilherme S. Terra, L. A. (septiembre 2014). ATTITUDE ESTIMATION USING A TWO-STEP UNSCENTED APPROACH WITH GYRO BIAS ESTIMATION AND ACCELERATION CORRECTION. *congreso brasileño de Automática*.

31. Håvard Fjær Grip, T. I. (2010). Attitude Estimation Based on Time-Varying Reference Vectors with Biased Gyro and Vector Measurements.

32. Higgins, W. (2007). A comparison of complementary and Kalman filtering. *Aerospace and Electronic Systems. IEEE Transactions on*.

33. Highsmith, J., & Orr. (2000). *Adaptive software development: A collaborative to managing complex systems.*

34. Hyyti, H., & Visala, A. (2015). A DCM Based Attitude Estimation Algorithm f or Low-Cost MEMS IMUs. *Hindawi Publishing Corporation*.

35. Javier García de Jalón de la Fuente, J. I., & GoñiLaseheras, R. (1998). Aprenda lenguaje ANSI C como si estuviera en primero.

36. Jesús. (2009). Móviles o Vehículos Robot. *Historia del Arte de la Robótica*. Obtenido de http://robotik-jjlg.blogspot.com/

37. Leccadito, M. T., Bakker, T., Niu, R., & Klenke, R. e. (2015). A Kalman Filter Based Attitude Heading Reference System Using a Low Cost Inertial Measurement Unit. *AIAA SciTech Forum*.

38. Madgwick, S. O. (2010). An efficient orientation filter for inertial and inertial/magnetic sensor arrays.

39. Matthew T. Leccadito, T. B. (5-9 de January de 2015). A Kalman Filter Based Attitude and Heading Reference System Using a Low Cost Inertial Measurement Unit. 12.

40. Moreno, H. A. (2014). Robótica submarina: Conceptos, Elementos, Modelado y Control. *Revista Iberoamericana de Automática e Informática Industrial*, 3-19.

41. Navarro, R. Q. (2014). Acondicionamiento de Mediciones de Sensores Inerciales de Bajo Costo con fines de navegación.

42. Pardo, A., Garrido, J., Ruiz, M. Á., & Martín, R. S. (2007). La interacción entre factores en el análisis de Varianza: errores de interpretación. *Psicothema*, 343-349.

43. Pohlert, T. (2016). The Pairwise Multiple Comparison of Mean Ranks Package (PMCMR).

44. Poppendieck, M., & Poppendiecck, J. (2003). *Software Development:An agile toolkit for software development managers.*

45. Ramírez-González, A., & Rubio, J. A. (2003). ESTUDIO Y MODELADO DE ERRORES EN SISTEMAS INERCIALES PARA NAVEGACIÓN TERRESTRE. 1.

46. Rico, E. C., Docampo, M. G., González, J. A., & Sanmartín, T. R. (2006). La escala cartográfica de la imagen del satélite. Caso particular de las imágenes Ikonos y QuickBird. *Revista de Teledetección*, 18-24.

47. Rico-Azagra, J., Rico, R., Maisterra, P., & Gil-Martínez, M. (2015). Comparación de algoritmos de estimación de actitud. *Actas de las XXXVI Jornadas de Automática*.

48. Rodriguez, H., & Chérigo, C. (2016). Evaluación de Algoritmos de Fusión de Datos de Medición Inercial para Determinar de la Orientación de Vehículos Aéreos no Tripulados. *Laboratorio Especializado de Análisis, Diseño y Simulación, Universidad Tecnológica de Panamá*.

49. Rogers, R. M. (2003). Applied mathematics in integrated navigation systems. 2nd ed.

50. Ruiz, D. V. (2009-2010). *Estado del arte en robótica submariana.* Valencia.

51. Schwaber, K., Beedle, M., & Martin, R. (2001). *Agile software development with scrum.*

52. service, N. A. (2017). *¿Cómo citar con norma APA?* Obtenido de http://normasapa.com/

53. Sesto, J. R. (junio de 2009). Vehículos autónomos submarinos: Nuevos actores en las operaciones navales. *Revista General de Marina, Tomo 256*, 839-842. Recuperado el 26 de octubre de 2016

54. Shin, E.-H. (2001). *Accuracy Improvement of Low Cost INS/GPS for Land Applications.*

55. Stapleton, J. (1997). *Dynamic systems development method:the method in practice.*

56. Uruguay, U. O. (2013). EJEMPLOS DE CITAS EN EL TEXTO Y REFERENCIAS SEGÚN NORMA APA INSTITUTO DE EDUCACIÓN.

57. Valencia Niño, C. H., & Dutra, M. S. (2010). Estado del Arte de los Vehículos Autónomos Sumergibles Alimentados por Energìa Solar. *Iteckne*, 46-53.

58. Valenti, R. G., Dryanovski, I., & Xiao, J. (2015). Keeping a Good Attitude: A Quaternion-Based Orientation Filter for IMUs and MARGs.

59. Wang, L., Zhang, Z., & Sun, P. (2015). Quaternion-based Kalman Filter for AHRS Using an Adaptive-step Gradient Descent Algorithm. *International Journal of Advanced Robotic Systems*.

60. WATSON, M. (2013). The Design and Implementation of a Robust AHRS for Integration into a Quadrotor Platform. *MENG ELECTRONIC ENGINEERING*.

61. Wesley, A. (2000). *Una aplicación de la programación extrema. Aceptar el cambio.*

62. Weston, J. L., & Titterton, D. H. (2004). Strapdown Inertial Navigation Technology. *The Institution of Electrical Engineers.*

63. Woodman, O. J. (2007). *An Introduccion to Inertial navegation.*

64. Zylberberg, A. (2005). *Probabilidad y Estadísticas.*

ANEXO 1. CÓDIGO FUENTE EN LENGUAJE M DEL FILTRO EXTENDIDO DE KALMAN PROPUESTO POR WATSON (2013).

```
function [ q, wb,x,P] = EKF_AHRS( a,w,m,x,P, Q, R,dt )
%%Normalizando las mediciones.
a = a / norm(a);
m = m / norm(m);

q0 = x(1);
q1 = x(2);
q2 = x(3);
q3 = x(4);

wxb = x(5);
wyb = x(6);
wzb = x(7);

wx = w(1);
wy = w(2);
wz = w(3);

% Etapa de prediccion
x = [q0 + (dt/2) * (-q1*(wx-wxb) - q2*(wy-wyb) - q3*(wz-wzb));
 q1 + (dt/2) * ( q0*(wx-wxb) + q3*(wy-wyb) - q2*(wz-wzb));
 q2 + (dt/2) * (-q3*(wx-wxb) + q0*(wy-wyb) + q1*(wz-wzb));
 q3 + (dt/2) * ( q2*(wx-wxb) - q1*(wy-wyb) + q0*(wz-wzb));
 wxb;
 wyb;
 wzb];

% Normalizando el cuaternion
qnorm = sqrt(x(1)^2 + x(2)^2 + x(3)^2 + x(4)^2);
x(1) = x(1)/qnorm;
x(2) = x(2)/qnorm;
x(3) = x(3)/qnorm;
x(4) = x(4)/qnorm;
q0 = x(1);
q1 = x(2);
q2 = x(3);
q3 = x(4);

% Calculo del Jacobiano
F = [    1, -(dt/2)*(wx-wxb), -(dt/2)*(wy-wyb), -(dt/2)*(wz-wzb),
(dt/2)*q1, (dt/2)*q2, (dt/2)*q3;
  (dt/2)*(wx-wxb),    1, -(dt/2)*(wz-wzb), (dt/2)*(wy-wyb), -
(dt/2)*q0, -(dt/2)*q3, (dt/2)*q2;
  (dt/2)*(wy-wyb), (dt/2)*(wz-wzb),     1, -(dt/2)*(wx-wxb),
(dt/2)*q3, -(dt/2)*q0, -(dt/2)*q1;
  (dt/2)*(wz-wzb), -(dt/2)*(wy-wyb), (dt/2)*(wx-wxb),     1, -
(dt/2)*q2, (dt/2)*q1, -(dt/2)*q0;
      0,    0,     0,     0,   1,   0,   0;
      0,    0,     0,     0,   0,   1,   0;
      0,    0,     0,     0,   0,   0,   1];

% Estimar matriz de covarianza del error
P = F * P * F' + Q;

% Vector de observaciones
```

```
z = [a(1); a(2); a(3); m(1); m(2); m(3)];

% Construir la matriz de rotaciones basado en cuaternio.
rq = [q0^2+q1^2-q2^2-q3^2,  2*(q1*q2-q0*q3),  2*(q1*q3+q0*q2);
  2*(q1*q2+q0*q3), q0^2-q1^2+q2^2-q3^2,  2*(q2*q3-q0*q1);
  2*(q1*q3-q0*q2),  2*(q2*q3+q0*q1), q0^2-q1^2-q2^2+q3^2];

% Rotar vector de campo magnético.
rm = rq * [z(4); z(5); z(6)];

bx = sqrt(rm(1)^2 + rm(2)^2);
bz = rm(3);
h = [ -2*(q1*q3 - q0*q2);
 -2*(q2*q3 + q0*q1);
 -q0^2 + q1^2 + q2^2 - q3^2;
  bx*(q0^2 + q1^2 - q2^2 - q3^2) + 2*bz*(q1*q3 - q0*q2);
  2*bx*(q1*q2 - q0*q3) + 2*bz*(q2*q3 + q0*q1);
  2*bx*(q1*q3 + q0*q2) + bz*(q0^2 - q1^2 - q2^2 + q3^2);];

% Medición residual
y = z - h;

% Calculo del Jacobiano
H = [    2*q2,    -2*q3,    2*q0,    -2*q1, 0, 0, 0;
     -2*q1,    -2*q0,    -2*q3,    -2*q2, 0, 0, 0;
     -2*q0,    2*q1,    2*q2,    -2*q3, 0, 0, 0;
  2*( q0*bx - q2*bz), 2*(q1*bx + q3*bz), 2*(-q2*bx - q0*bz), 2*(-
q3*bx + q1*bz), 0, 0, 0;
  2*(-q3*bx + q1*bz), 2*(q2*bx + q0*bz), 2*( q1*bx + q3*bz), 2*(-
q0*bx + q2*bz), 0, 0, 0;
  2*( q2*bx + q0*bz), 2*(q3*bx - q1*bz), 2*( q0*bx - q2*bz), 2*(
q1*bx + q3*bz), 0, 0, 0];

K = P * H' /(H * P * H' + R);

% Actualizacion del estado
x = x + K*y;

% Actualizacion de matriz de covarianza
I = eye(length(P));
P = (I - K * H) * P *(I - K * H)' + K * R * K';

qnorm = sqrt(x(1)^2 + x(2)^2 + x(3)^2 + x(4)^2);
x(1) = x(1)/qnorm;
x(2) = x(2)/qnorm;
x(3) = x(3)/qnorm;
x(4) = x(4)/qnorm;
q = [x(1), x(2), x(3), x(4)]';
wb = [x(5), x(6), x(7)];
end
```

ANEXO 2. CÓDIGO FUENTE EN LENGUAJE M DEL FILTRO COMPLEMENTARIO PROPUESTO POR MADGWICK (2010).

```
classdef MadgwickAHRS < handle

%MADGWICKAHRS Implementation of Madgwick's IMU and AHRS algorithms
%
%
% Date    Author
% 28/09/2011 SOH Madgwick

 %% Public properties
 properties (Access = public)
  SamplePeriod = 1/100;
  Quaternion = [1 0 0 0];  % output quaternion describing the Earth
relative to the sensor
  Beta = 1;      % algorithm gain
 end

 %% Public methods
 methods (Access = public)
  function obj = MadgwickAHRS(varargin)
   for i = 1:2:nargin
    if strcmp(varargin{i}, 'SamplePeriod'), obj.SamplePeriod =
varargin{i+1};
    elseif strcmp(varargin{i}, 'Quaternion'), obj.Quaternion =
varargin{i+1};
    elseif strcmp(varargin{i}, 'Beta'), obj.Beta = varargin{i+1};
    else error('Invalid argument');
    end
   end;
  end
  function obj = Update(obj, Gyroscope, Accelerometer, Magnetometer)
   q = obj.Quaternion; % short name local variable for readability

   % Normalise accelerometer measurement
   if(norm(Accelerometer) == 0), return; end % handle NaN
   Accelerometer = Accelerometer / norm(Accelerometer); % normalise
magnitude

   % Normalise magnetometer measurement
   if(norm(Magnetometer) == 0), return; end % handle NaN
   Magnetometer = Magnetometer / norm(Magnetometer); % normalise
magnitude

   % Reference direction of Earth's magnetic feild
   h = quaternProd(q, quaternProd([0 Magnetometer], quaternConj(q)));
   b = [0 norm([h(2) h(3)]) 0 h(4)];

   % Gradient decent algorithm corrective step
   F = [2*(q(2)*q(4) - q(1)*q(3)) - Accelerometer(1)
    2*(q(1)*q(2) + q(3)*q(4)) - Accelerometer(2)
    2*(0.5 - q(2)^2 - q(3)^2) - Accelerometer(3)
    2*b(2)*(0.5 - q(3)^2 - q(4)^2) + 2*b(4)*(q(2)*q(4) - q(1)*q(3)) -
Magnetometer(1)
    2*b(2)*(q(2)*q(3) - q(1)*q(4)) + 2*b(4)*(q(1)*q(2) + q(3)*q(4)) -
Magnetometer(2)
```

```
    2*b(2)*(q(1)*q(3) + q(2)*q(4)) + 2*b(4)*(0.5 - q(2)^2 - q(3)^2) -
Magnetometer(3)];
   J = [-2*q(3),     2*q(4),     -2*q(1),       2*q(2)
    2*q(2),      2*q(1),      2*q(4),       2*q(3)
    0,       -4*q(2),     -4*q(3),       0
    -2*b(4)*q(3),    2*b(4)*q(4),    -4*b(2)*q(3)-2*b(4)*q(1),  -
4*b(2)*q(4)+2*b(4)*q(2)
    -2*b(2)*q(4)+2*b(4)*q(2), 2*b(2)*q(3)+2*b(4)*q(1),
2*b(2)*q(2)+2*b(4)*q(4),  -2*b(2)*q(1)+2*b(4)*q(3)
    2*b(2)*q(3),    2*b(2)*q(4)-4*b(4)*q(2), 2*b(2)*q(1)-4*b(4)*q(3),
2*b(2)*q(2)];
   step = (J'*F);
   step = step / norm(step); % normalise step magnitude

   % Compute rate of change of quaternion
   qDot = 0.5 * quaternProd(q, [0 Gyroscope(1) Gyroscope(2)
Gyroscope(3)]) - obj.Beta * step';

   % Integrate to yield quaternion
   q = q + qDot * obj.SamplePeriod;
   obj.Quaternion = q / norm(q); % normalise quaternion
  end
  function obj = UpdateIMU(obj, Gyroscope, Accelerometer)
   q = obj.Quaternion; % short name local variable for readability

   % Normalise accelerometer measurement
   if(norm(Accelerometer) == 0), return; end % handle NaN
   Accelerometer = Accelerometer / norm(Accelerometer); % normalise
magnitude

   % Gradient decent algorithm corrective step
   F = [2*(q(2)*q(4) - q(1)*q(3)) - Accelerometer(1)
     2*(q(1)*q(2) + q(3)*q(4)) - Accelerometer(2)
     2*(0.5 - q(2)^2 - q(3)^2) - Accelerometer(3)];
   J = [-2*q(3), 2*q(4), -2*q(1), 2*q(2)
    2*q(2),  2*q(1),  2*q(4), 2*q(3)
    0,   -4*q(2), -4*q(3), 0 ];
   step = (J'*F);
   step = step / norm(step); % normalise step magnitude

   % Compute rate of change of quaternion
   qDot = 0.5 * quaternProd(q, [0 Gyroscope(1) Gyroscope(2)
Gyroscope(3)]) - obj.Beta * step';

   % Integrate to yield quaternion
   q = q + qDot * obj.SamplePeriod;
   obj.Quaternion = q / norm(q); % normalise quaternion
  end
 end
end
```

ANEXO 3. CÓDIGO FUENTE EN LENGUAJE M DEL FILTRO COMPLEMENTARIO PROPUESTO POR VALENTI (2015).

```
function [ q_EST ] = Valenti_AHRS( acc, gyr, mag, q_ANT, ts )
 %% Calculando la ganancia adaptativa alfa para la correccion de la
gravedad.
 G = 9.81;
 em = abs(norm(acc) - G)/G;

 ALFA_CONST = 0.03;
 if em < 0.1
  f = 1;
 else if em>0.2
   f = 0;
  else
   f = -10*em+2;
  end
 end

 alfa = ALFA_CONST * f;

 %% Normalizacion de las aceleraciones
 acc = acc / norm (acc);

 %% Etapa de prediccion
 % En la etapa de prediccion, el vector de velocidad angular, medido
por el
 % giroscopo de 3 ejes, se utiliza para calcular una primera
estimación de la
 % orientacion en forma de cuaternion.

 q_ANT = Quatern_Conj(q_ANT');

 dot_Q = 0.5 * Quatern_Prod(q_ANT, [0 gyr(1) gyr(2) gyr(3)]);
 q_W = q_ANT + dot_Q * ts;
 q_W = q_W / norm(q_W);

 %% Etapa de correcion
 % La corrección adoptada se basa en la técnica multiplicativa. El
 % cuaternion redicho q_W se corrige por medio de dos delta
 % cuaterniones.

 %Calculando delta_Q_ACC
 q_I = [1 0 0 0 ]';
 dcm = Quatern_Rot_Mat(q_W);
 g_P = dcm * acc ;
 g_X = g_P(1);
 g_Y = g_P(2);
 g_Z = g_P(3);
 delta_Q_ACC = [sqrt((g_Z+1)/2), -g_Y/sqrt(2*(g_Z+1)),
g_X/sqrt(2*(g_Z+1)), 0]';
 delta_Q_ACC=delta_Q_ACC/norm(delta_Q_ACC);
 EPSILON = 0.9; % Del autor
 omega_A = acos (Quater_Dot_Product(q_I', delta_Q_ACC')); % Esta
funcion (acos) devuelve el coseno inverso (cos-1) de un elemento X.
 if delta_Q_ACC(1) > EPSILON % LERP
  delta_Q_ACC = (1 - alfa)*q_I + alfa*delta_Q_ACC;
  delta_Q_ACC = delta_Q_ACC / norm(delta_Q_ACC);
```

```
 else % SLERP
  delta_Q_ACC = sin((1 - alfa)*omega_A)/sin(omega_A)*q_I +
sin(alfa*omega_A)/sin(omega_A)*delta_Q_ACC;
  delta_Q_ACC = delta_Q_ACC / norm(delta_Q_ACC);
 end

 % Primera correccion de q_W con delta_Q_ACC
 q_EST = Quatern_Prod(q_W, delta_Q_ACC');
 q_EST = q_EST';

 % Calculando delta_Q_MAG
 l = Quatern_Rot_Mat(q_EST') * mag;
 lx = l(1);
 ly = l(2);

 L = l(1)^2 + l(2)^2;
 delta_Q_MAG = [sqrt((L + lx*sqrt(L)))/sqrt(2*L),0, 0,
ly/sqrt(2*(L+lx*sqrt(L)))]';
 delta_Q_MAG=delta_Q_MAG/norm(delta_Q_MAG);

 BETA = 0.3;
 omega_M = acos (Quater_Dot_Product(q_I', delta_Q_MAG')); % Esta
funcion (acos) devuelve el coseno inverso (cos-1) de un elemento X.
 if delta_Q_MAG(1) > EPSILON % LERP
  delta_Q_MAG = (1 - BETA)*q_I + BETA*delta_Q_MAG;
  delta_Q_MAG = delta_Q_MAG / norm(delta_Q_MAG);
 else % SLERP
  delta_Q_MAG = sin((1 - BETA)*omega_M)/sin(omega_M)*q_I +
sin(BETA*omega_M)/sin(omega_M)*delta_Q_MAG;
 end
 % Segunda correcion de q_W con delta_Q_MAG
 q_EST = Quatern_Prod(q_EST', delta_Q_MAG');
 q_EST = q_EST/norm(q_EST);
 q_EST = Quatern_Conj (q_EST);
 q_EST = q_EST';

 end
```