UNIVERSIDAD DE LAS PALMAS
DE GRAN CANARIA

ESCUELA DE INGENIERÍA DE TELECOMUNICACIÓN Y ELECTRÓNICA

TRABAJO FIN DE MÁSTER

Prototipo de una unidad de apoyo a la conducción eficiente utilizando un sensor inercial.

Titulación: Máster Universitario en Ingeniería de Telecomunicación
Autor: Frank Pérez Paz
Tutores: Roberto Esper-Chaín Falcón
Fecha: Julio 2015

UNIVERSIDAD DE LAS PALMAS
DE GRAN CANARIA

ESCUELA DE INGENIERÍA DE TELECOMUNICACIÓN Y ELECTRÓNICA

TRABAJO FIN DE MÁSTER

Prototipo de una unidad de apoyo a la conducción eficiente utilizando un sensor inercial.

HOJA DE FIRMAS

Alumno	**Tutor**
Fdo.: Frank Pérez Paz	Fdo.: Dr. Roberto Esper-Chaín Falcón

Fecha: Julio 2015

UNIVERSIDAD DE LAS PALMAS
DE GRAN CANARIA

ESCUELA DE INGENIERÍA DE TELECOMUNICACIÓN Y ELECTRÓNICA

TRABAJO FIN DE MÁSTER

Prototipo de una unidad de apoyo a la conducción eficiente utilizando un sensor inercial.

HOJA DE EVALUACIÓN

Calificación: _____

Presidente

Fdo.:

Vocal **Secretario/a**

Fdo.: Fdo.:

Fecha: Julio 2015

Contenido

Índice de Figuras

Índice de Tablas

1.0 Introducción

En este Trabajo Fin de Máster se pretenderá llevar a cabo la realización de un prototipo de medición de la eficiencia en la conducción. Se desarrollarán diferentes algoritmos para llevar a cabo una comparación y ver cuál de ellos tiene un funcionamiento óptimo.

1.1 Antecedentes

El transporte por carretera consume el 42,1% de la energía en España, lo que representa más del 60% del petróleo consumido en nuestro país. El consumo de carburantes en España en el sector del transporte por carretera es de unos 11.000 millones de litros de gasolina y de unos 24.000 millones de litros de gasóleo al año. [1]

Debido a estos altos consumos, en los últimos años el sector automovilístico ha centrado la mayor parte de los esfuerzos en la reducción del consumo de combustibles. Para ello, se han diseñado motores híbridos, se ha reducido el tamaño de los motores (cilindrada) y se han programado las centralitas de los vehículos para obtener la máxima eficiencia. Otras medidas que se han venido desarrollando en el sector del transporte profesional incluyen la inyección de alta presión con sistemas de inyector unitario o sistemas “Common rail”, la sobrealimentación con postenfriado y turbo de paso variable, así como la recirculación de gases de escape (EGR). Todas ellas giran en torno a la parte mecánica de los vehículos.

Con estos avances se ha logrado reducir el consumo de combustible y las emisiones de CO2 a la atmosfera, pero no es suficiente. Ninguna de estas medidas tiene en cuenta el factor humano, es decir, no se presta atención al estilo de conducir de la persona. Las estadísticas señalan que atendiendo a la manera de conducir de los usuarios, se puede ahorrar hasta un 15% de combustible. Si esta cantidad la miramos en el sector del transporte profesional (flotas de vehículos, transporte público, transporte de mercancías, etc.) las cifras de ahorro son muy elevadas.

La conducción eficiente ofrece una serie de ventajas como el ahorro de energía, ahorro económico para las empresas del transporte, reducción de costes de mantenimiento, reducción de emisiones y mejora de la velocidad media.

Para medir el índice de conducción eficiente, se ha pensado en la utilización de sensores inerciales [3] que detecten el movimiento del vehículo. Estos sensores pueden venir integrados en una PCB prefabricada [2]. El siguiente paso es analizar los datos proporcionados por los sensores.

Para interpretar los datos de los sensores, se utilizan algoritmos como las rotaciones de Euler y los cuaterniones [4][5]. Estos cálculos matemáticos, son utilizados frecuentemente en el sector aeronáutico y de videojuegos para seguir el movimiento de un objeto.

1.2 Objetivos

Los objetivos del presente Trabajo de Fin de Máster son desarrollar un prototipo de apoyo a la conducción eficiente, que tras procesar los datos extraídos de una unidad inercial, cumpla con las siguientes condiciones:

- En la instalación no se necesite de calibración mecánica.
- Mida las aceleraciones y frenadas de un vehículo en el que vaya instalado.
- Indique el nivel de economía y confort estimado.
- Transmita por algún puerto de comunicación alertas de exceso de aceleración o frenada.
- Estime un factor de mérito del nivel de conducción.

1.3 Estructura de la Memoria

La memoria de este Trabajo Fin de Máster se ha dividido en 7 capítulos. Asimismo se ha incluido una lista de referencias al finalizar los capítulos y un presupuesto del proyecto; ambos localizados en la parte final del documento.

En los primeros capítulos, además de la introducción, se explica la base matemática que se ha estudiado para desarrollar el proyecto, seguidamente se explican los principios de la navegación inercial, y luego una descripción del Hardware y el Software utilizado.

En mitad de la memoria se empiezan a describir los algoritmos desarrollados, las pruebas realizadas y las conclusiones extraídas de su funcionamiento.

Cada capítulo contiene lo siguiente:

En el segundo capítulo se empieza explicando cómo funcionan las rotaciones de Euler, y las matrices de rotación que se utilizan. Seguidamente se explica en que se basan los cuaterniones, empezando por una explicación de los conceptos básicos hasta como se expresan matemáticamente. Luego se explica, incluyendo un ejemplo, las rotaciones de los cuaterniones. Y para finalizar el capítulo se habla sobre el método SLERP aplicado a los cuaterniones.

En el tercer capítulo se habla en primer lugar de la implementación física de los sensores inerciales. Seguidamente se comenta lo que es una IMU para luego continuar describiendo en qué consisten los sistemas de navegación inercial. Para finalizar se da una breve introducción a los filtros, haciendo hincapié en los filtros digitales.

En el cuarto capítulo se empieza describiendo el microcontrolador que se va a utilizar. Se describe la unidad de medida inercial (IMU) que se emplea y los esquemáticos de conexión entre ambos. También se explica el funcionamiento de la herramienta de desarrollo utilizada para la programación del microcontrolador y las pruebas realizadas antes de la programación, por último se expone el dispositivo programador utilizado.

En el quinto capítulo se explican los diferentes algoritmos que se han ido desarrollando. Se comienza con el primer algoritmo más básico hasta el último algoritmo implementado el cual es el más complejo.

En el sexto capítulo se habla de las pruebas de cada algoritmo. En cada prueba se hace una descripción de si funcionaba adecuadamente el algoritmo y de los problemas que se habían de solucionar.

En el último capítulo se explican las conclusiones extraídas de la realización de este proyecto, así como las posibles mejoras que se pueden incorporar.

Para finalizar la memoria, se incluye el apartado de las referencias utilizadas para realizar los primeros capítulos sobre la base matemática y los principios de la navegación inercial. Y se concluye con un presupuesto en el que se recoge el precio final del prototipo.

2.0 Modelado Matemático

2.1 Introducción

Para el desarrollo de este proyecto, es preciso conocer cómo representar el movimiento de un vehículo en un sistema de coordenadas. En los sistemas de navegación aeronáuticos se han empleado tradicionalmente los ángulos de Euler. Sin embargo, como para determinados ángulos de rotación no siempre son fiables, en los últimos tiempos se empezaron a utilizar los cuaterniones. Los cuaterniones se utilizan sobre todo para la representación gráfica del movimiento de un objeto, pero son una alternativa a los ángulos de Euler al ser más eficientes (matemáticamente) y seguros (no tiene el problema que tienen los ángulos de Euler).

2.2 Rotaciones de Euler

Para ver cómo se calculan las matrices de rotación, se hace una matriz por cada eje que va rotando. Si se empieza una rotación en el eje **Z** con un ángulo ψ, se produce una nueva coordenada donde el eje **Z** se mantiene y los ejes **X** e **Y** son rotados el ángulo ψ. En la siguiente figura se explica el movimiento que realiza el avión al rotar sobre el eje **Z**, en donde se desplazan los ejes **X** e **Y**.

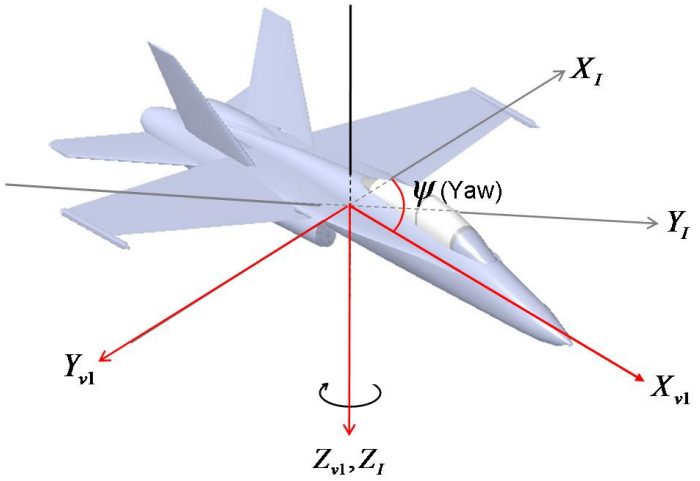


Figura 1: Rotación entorno al eje guiñada (yaw)

Si la rotación que se produce es en este sentido, se obtiene la siguiente matriz:

$$R_I^{v1}(\psi) = \begin{pmatrix} \cos(\psi) & \sin(\psi) & 0 \\ -\sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Ahora si se produce otra rotación pero esta vez en el eje Yaw y cabeceo, donde cabeceo es representado por el eje **Y** con una rotación de θ.

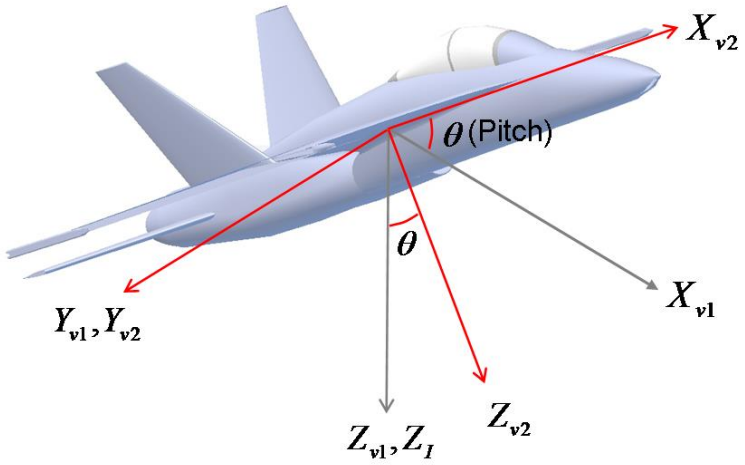


Figura 2: Rotación sobre el eje cabeceo (pitch)

Esta nueva rotación es representada por la siguiente matriz:

$$R_{v1}^{v2}(\theta) = \begin{pmatrix} \cos(\theta) & 0 & -\sin(\theta) \\ 0 & 1 & 0 \\ \sin(\theta) & 0 & \cos(\theta) \end{pmatrix}$$

Y la rotación total realizada hasta el momento es representada por la multiplicación de ambas matrices: $R_I^{v2}(\theta, \psi) = R_{v1}^{v2}(\theta) \cdot R_I^{v1}(\psi)$.

Por último, cuando se tiene una rotación con sobre los tres ejes guiñada, cabeceo y alabeo, se obtiene la matriz de rotación del eje que falta, el eje **X** con un ángulo ϕ.

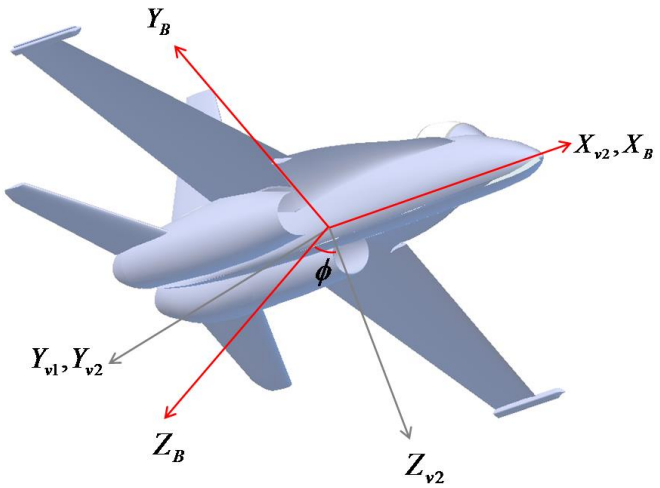


Figura 3: Rotación sobre el eje alabeo (roll)

Se obtiene la matriz de rotación que falta:

$$R_{v2}^{B}(\phi) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi) & \sin(\phi) \\ 0 & -\sin(\phi) & \cos(\phi) \end{pmatrix}$$

Para obtener la matriz de rotación del movimiento completo, se realiza la multiplicación de todas las matrices. Si el movimiento es en sentido opuesto, se cambia de signo al ángulo.

$$R_{I}^{B}\left(\phi, \theta, \psi\right) = R_{v2}^{B}(\phi) \cdot R_{v1}^{v2}(\theta) \cdot R_{I}^{v1}(\psi)$$

El inconveniente que presentan los ángulos de Euler es que puede producirse un efecto conocido como ”Gimbal Lock”. Este efecto ocurre cuando debido a una rotación, dos ejes apuntan al mismo sitio y produce que se pierda uno de los dos ejes. Se produce cuando se realizan rotaciones con un ángulo de ±90°. Si la aplicación en la que se utilicen las rotaciones de Euler, no tiene una rotación de ±90°, no tendrá problemas. Los ángulos de Euler tienen la ventaja de ser fáciles de entender e intuitivos, pero están limitados por este efecto. Por este motivo, para determinadas aplicaciones es necesario utilizar los cuaterniones. [4]

2.3 Álgebra de cuaterniones

La utilización de los ángulos de Euler para la representación de la orientación, es fácil de desarrollar y de visualizar pero presenta problemas a la hora de realizar los cálculos y dependiendo del movimiento, puede dar indeterminaciones. Por este motivo, no es adecuado (efectivo) utilizarlos para la representación de dispositivos que pueden moverse en todas las direcciones.

Para ello se utilizan los caterniones, los cuales se basan en la rotación de Euler pero con estados relativos de orientación los cuales eliminan los problemas que causan los ángulos de Euler. Un cuaternión está formado por una componente escalar, así como un vector situado en el espacio complejo.

Los cuaterniones fueron desarrollados por William Rowan Hamilton el cual empezó a interesarse por el movimiento de los sistemas de cuerpos y por la dualidad que existía entre un sistema dinámico y las coordenadas de su posición. Empezó a trabajar con los números complejos y tras varias teorías, empezó a desarrollar los cálculos matemáticos de los cuaterniones. En 1843 desarrolló los cuaterniones como números hipercomplejos los cuales eran una extensión de los números complejos, donde el tiempo es el escalar y los puntos del espacio son definidos por las tres coordenadas reales.[9] Los cuaterniones tienen las siguientes propiedades: [10]

$$ij = k \quad jk = i \quad ki = j \quad Ciclicidad$$

$$ji = -k \quad kj = -i \quad ik = -j \quad Anticiclicidad$$

$$i^2 = j^2 = k^2 = -1 \quad Antinormalidad$$

En la siguiente tabla, se puede observar las propiedades la multiplicación de los cuaterniones.

	1	i	j	k
1	1	i	j	k
i	i	-1	k	-j
j	j	-k	-1	i
k	k	j	i	-1

Tabla 1: Resultado de la multiplicación de los vectores básicos del cuaternión

Los cuaterniones, se representa de la siguiente manera:

$$q = (w,\vec{v}) = \left(q_w, q_x\hat{\imath}, q_y\hat{\jmath}, q_z\hat{k}\right)$$

En donde $\hat{\imath},\hat{\jmath}\ y\ \hat{k}$ son vectores complejos ortogonales, y debido a su naturaleza compleja, la multiplicación de los cuaterniones no es conmutativa, es decir, $p * q \neq q * p$. La forma de representar un cuaternión en forma de matriz es la siguiente:

$$q = \begin{bmatrix} w \\ \vec{v} \end{bmatrix} = \begin{bmatrix} w \\ v_x \\ v_y \\ v_z \end{bmatrix} = \begin{bmatrix} q_s \\ q_x \\ q_y \\ q_z \end{bmatrix}$$

Como el cuaternión es definido como una rotación de los ejes y del ángulo de rotación, la parte real (el componente escalar) se representa como un $cos\theta$ y la parte vectorial, se representa como un $sen\theta$.

$$q = \left(cos\theta, \left|\left|\vec{v}\right|\right| \cdot sin\theta\right)$$

$$q = \begin{bmatrix} q_s \\ q_x \\ q_y \\ q_z \end{bmatrix} = \begin{bmatrix} cos\theta \\ \left|\left|\vec{v}\right|\right| \cdot sin\theta \end{bmatrix}$$

Donde $\left|\left|\vec{v}\right|\right|$es el cuaternión normalizado y el ángulo θ no es la rotación del ángulo sino una transformación de este.

Al ser el cuaternión de naturaleza compleja, el cual se puede definir como un complejo conjugado, se puede expresar el complejo conjugado del cuaternión con la notación * y se define como:

$$q^* = (w,-\vec{v}) = \left(q_w, -q_x\hat{\imath}, -q_y\hat{\jmath}, -q_z\hat{k}\right)$$

$$q^* = \begin{bmatrix} q_s \\ -q_x \\ -q_y \\ -q_z \end{bmatrix} = \begin{bmatrix} cos\theta \\ -\left|\left|\vec{v}\right|\right| \cdot sin\theta \end{bmatrix}$$

Cambiándole el signo a la parte vectorial y manteniendo la parte escalar; esto permite definir la inversa del cuaternión como q^{-1}, tal que $qq^{-1} = 1$.

$$q^{-1} = \frac{q^*}{|q|}$$

La norma (módulo) del cuaternión se calcula de la siguiente manera:

$$|q| = \sqrt{{q_w}^2 + {q_x}^2 + {q_y}^2 + {q_z}^2}$$

Un cuaternión con $|q| = 1$ es conocido como cuaternión unidad. Es necesario conocer cómo se calcula la norma, debido a que se utiliza para normalizar el cuaternión. La forma de normalizar un cuaternión es:

$$||q|| = \frac{q}{|q|}$$

Un cuaternión puede ser sumado y restado de manera similar a la de los números complejos. Es decir, sumando o restando la parte real de un vector con la parte real del otro e igualmente con la parte imaginaria.

Para multiplicar los cuaterniones, se realiza de manera semejante a la multiplicación de números complejos. Suponiendo dos cuaterniones q y p, estos se multiplican de la siguiente manera:

$$\begin{aligned} q \cdot p &= \left(q_w, q_x\hat{\imath}, q_y\hat{\jmath}, q_z\hat{k}\right) \cdot \left(p_w, p_x\hat{\imath}, p_y\hat{\jmath}, p_z\hat{k}\right) \\ &= \left(q_w p_w - q_x p_x - q_y p_y - q_z p_z\right) + \left(q_w p_x + p_w q_x + q_y p_z - p_y q_z\right)i \\ &+ \left(q_w p_y + p_w q_y + q_z p_x - p_z q_x\right)j + \left(q_w p_z + p_w q_z + q_x p_y - p_x q_y\right)k \end{aligned}$$

2.4 Rotación de cuaterniones

Para comprender la rotación de los cuaterniones, la mejor manera es a través de un ejemplo, en donde se empiece por una rotación de un cuaternión puro, es decir, con la parte escalar igual a cero. También se ha de considerar un caso especial en el que el cuaternión **p** es perpendicular a **q**. [5]

Por ejemplo, rotando el vector **p** en 45º sobre el eje **z**, tenemos el siguiente cuaternión:

$$q = [cos\theta + sin\theta] = \left[\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}k\right]$$

Y para que se dé el caso especial mencionado anteriormente, hacemos que el cuaternión **p** sea perpendicular a **k**, es decir:

$$p = [0,2i]$$

Haciendo el producto de ambos, se obtiene un cuaternión puro rotado 45º sobre el eje **k**.

$$p' = qp = \left[\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}k\right][0,2i] = [0, \sqrt{2}i + \sqrt{2}j]$$

En la figura siguiente se observa cómo obtenemos el vector p' el cual tiene las coordenadas "i" y "j" y el módulo de p, se mantiene para p'.

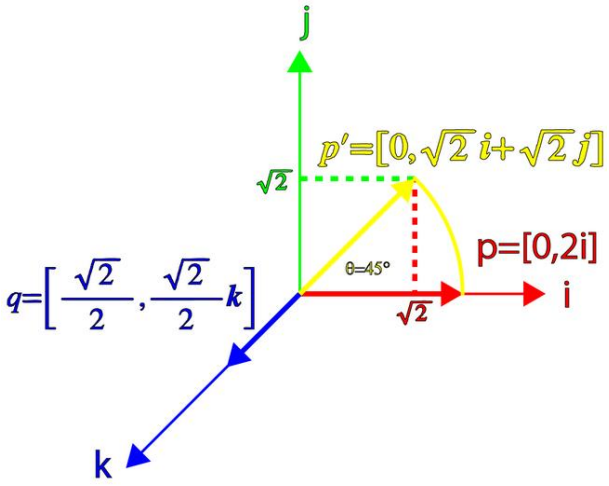


Figura 4: Rotación de 45º sobre el eje k

Ahora si se considera que el cuaternión **q** no es ortogonal a **p**. Es decir, el cuaternión **q** ahora tiene valores de dos coordenadas $q = \frac{\sqrt{2}}{2}i + \frac{\sqrt{2}}{2}k$.

$$p' = qp = [cos\theta, sin\theta\hat{q}] = [-sin\theta\hat{q}p, cos\theta p + sin\theta\hat{q}p] \rightarrow \theta = 45° \rightarrow$$

$$= \left[-\frac{\sqrt{2}}{2}\left(\frac{\sqrt{2}}{2}i + \frac{\sqrt{2}}{2}k\right)\cdot(2i), \frac{\sqrt{2}}{2}2i + \frac{\sqrt{2}}{2}\left(\frac{\sqrt{2}}{2}i + \frac{\sqrt{2}}{2}k\right)\cdot 2i\right]$$

$$= [-1, \sqrt{2}i + j]$$

Con un cuaternión no puro, no se obtiene una rotación de 45º y tampoco se consigue el mismo módulo del vector. Se puede observar más detalladamente en la siguiente imagen, donde el cuaternión resultante p' no tiene semejanza alguna con el original, es decir, no se ha realizado correctamente la rotación del cuaternión.

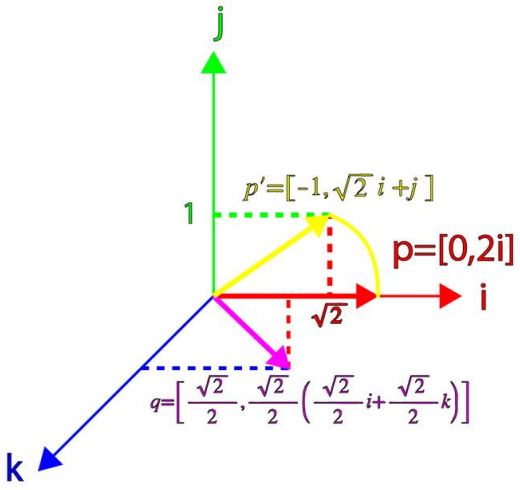


Figura 5: Rotación de un cuaternión sobre otro cuaternión no puro

Debido a que si rotamos un cuaternión con un cuaternión no puro, no se obtiene la rotación esperada, lo que se hace es multiplicar el resultado anterior por la inversa del cuaternión **q**, dando como resultado un cuaternión puro.

$$q^{-1} = \left[cos\theta, -sin\theta\left(\frac{\sqrt{2}}{2}i + \frac{\sqrt{2}}{2}k\right)\right] \rightarrow \theta = 45° \rightarrow = \left[\frac{\sqrt{2}}{2}, -\frac{\sqrt{2}}{2}\left(\frac{\sqrt{2}}{2}i + \frac{\sqrt{2}}{2}k\right)\right]$$

$$= \frac{1}{2}[\sqrt{2}, -i - k]$$

Y si multiplicamos esto por el valor anterior, obtenemos:

$$\boldsymbol{p'} = \boldsymbol{qpq^{-1}} = [-1, \sqrt{2}i + j] \cdot \frac{1}{2}[\sqrt{2}, -i - k] = [0, i + \sqrt{2}j + k]$$

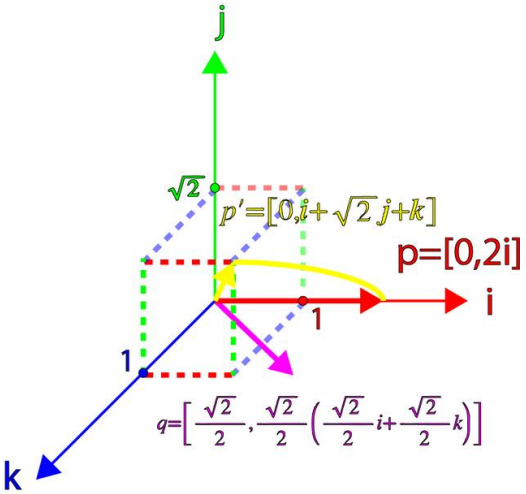


Figura 6: Visualización del resultado de la rotación de un cuaternión

Donde el vector resultante tiene el mismo módulo que el vector inicial, aunque está rotado 90º en vez de los 45º que se habían utilizado. Para corregir esto, la manera correcta de girar un vector sobre un cierto ángulo θ de un cuaternión es dividir el ángulo a la mitad, quedando el cuaternión de la siguiente manera:

$$q = \left(cos\frac{\theta}{2}, ||\vec{v}|| \cdot sin\frac{\theta}{2}\right)$$

2.5 SLERP

El método SLERP (stands for **S**pherical **L**inear Int**erp**olation) se utiliza para interpolar un punto sobre dos orientaciones. Si se tiene un punto inicial q1 con un t = 0, el segundo punto será un q2 con t = 1 y el punto que se interpola lo denominamos por P. El principio de interpolación lineal es [5]:

$$p' = p_1 + t(p_2 - p_1)$$

donde $p_2 - p_1$es la diferencia entre los puntos.

El principio de interpolación lineal se puede extrapolar a los cuaterniones con unos sencillos pasos. El primero es recordar que la diferencia de dos cuaterniones es $q_1^{-1}q_2$, el segundo paso es obtener el exponente de un cuaternión para lo cual se utiliza la fórmula: $q^t = \exp(t \cdot log(q))$. Para t = 0, se obtiene que $q^0 = \exp([1,0])$ y para t = 1, se obtiene $q^1 = \exp(q) = q$.

Utilizando estos pasos, el principio de interpolación lineal para cuaterniones se obtiene con la fórmula general:

$$q' = q_1(q_1^{-1}q_2)^t$$

Para utilizarla cuando se estén manipulando los vectores de los cuaterniones, se utiliza de la siguiente manera, aplicando la siguiente fórmula:

$$v_t = \frac{\sin(1-t)\theta}{\sin\theta}v_1 + \frac{\sin(t)\theta}{\sin\theta}v_2$$

Si en vez de utilizar los vectores, se utilizan los cuaterniones sin modificar, la expresión no se modifica, en donde estaba el vector, ahora estará el cuaternión.

$$q_t = \frac{\sin(1-t)\theta}{\sin\theta}q_1 + \frac{\sin(t)\theta}{\sin\theta}q_2$$

Y se puede obtener el ángulo θ haciendo el producto entre los cuaterniones y despejando el ángulo.

$$\cos\theta = \frac{q_1 \cdot q_2}{|q_1||q_2|} = \frac{s_1 s_2 + x_1 x_2 + y_1 y_2 + z_1 z_2}{|q_1||q_2|}$$

$$\theta = \cos^{-1}\left(\frac{s_1 s_2 + x_1 x_2 + y_1 y_2 + z_1 z_2}{|q_1||q_2|}\right)$$

En ocasiones, puede que el resultado del producto sea negativo. Para solucionar este problema lo que se hace es modificar uno de los dos cuaterniones y convertir la parte escalar en negativa, dejando la parte vectorial como estaba.

Otro problema que puede surgir es cuando la diferencia entre los cuaterniones es muy pequeña y hace que el $\sin\theta$ sea 0. Cuando esto pasa, hay que volver a interpolar nuevamente.

3.0 Principios de Navegación Inercial

3.1 Introducción

Los principios de la navegación inercial están basados en las leyes del movimiento de Newton. Concretamente la segunda ley de Newton dice que la aceleración de un cuerpo es proporcional al resultado de la fuerza que se le aplica y en el mismo sentido de la fuerza. Esto quiere decir que la fuerza que se le tiene que aplicar a un cuerpo para desplazarlo es directamente proporcional a su masa y a la aceleración que se le quiera dar. Expresado de forma matemática se obtiene [6]:

$$F = ma$$

Dónde:

F = fuerza aplicada

m = masa del cuerpo

a = aceleración del cuerpo debida a la fuerza aplicada sobre él.

La física permite convertir el valor de la aceleración *a* en velocidad *v* ya que la velocidad es un desplazamiento *s* derivado de la aceleración. Es decir la velocidad y la aceleración pueden ser estimadas por la diferencia de desplazamiento existente.

$$v = \frac{ds}{dt}; a = \frac{dv}{dt} = \frac{d^2s}{dt^2}$$

Si se hace la inversa del diferencial se obtiene el proceso de integración:

$$v = \int adt;\ \ s = \int vdt = \iint adtdt$$

Un sistema de navegación inercial consiste en un sensor detector de movimientos de aceleración y giro. Estos movimientos son integrados para obtener la velocidad, y si son integrados nuevamente, se obtiene el espacio recorrido respecto al tiempo. Además se necesita un sensor de rotación para transformar el sistema inercial en el sistema de navegación y para observar la forma de giro del vehículo en donde esté colocado.

3.2 Implementación Física de un Sistema Inercial

Hay dos maneras de implementar el Sistema Inercial: la primera manera es estableciendo una plataforma en donde los sensores estén colocados en anillos que giran en ángulo recto sobre el cuerpo del vehículo; la segunda es colocando directamente los sensores sobre el cuerpo del vehículo. En la siguiente figura se observa la diferencia entre las dos formas de colocación.

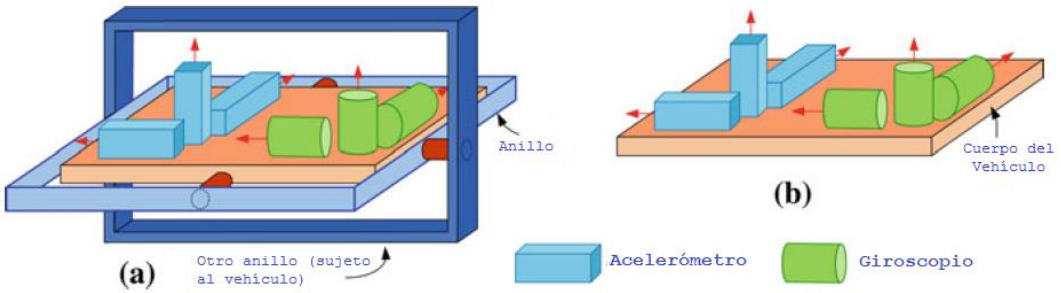


Figura 7: a) En Colocación de Anillos b) Directamente sobre el Vehículo

La colocación en Anillos, se utiliza para que la plataforma en donde están los sensores rote al mismo tiempo que el torque de motor y sea captado por los giroscopios, los cuales siempre estarán continuamente girando. Por este motivo la salida del acelerómetro es directamente integrada para obtener la velocidad y la posición o el espacio recorrido. El problema de estos sensores es que son más complejos, más caros, más grandes y pesados por lo que su utilización está más limitada que los sensores colocados directamente sobre el vehículo.

En los sistemas colocados directamente sobre el vehículo, el sistema de movimiento de la colocación en anillos es remplazado por un software que simula la rotación de la plataforma. La rotación es continuamente medida y actualizada por el giroscopio al igual que la aceleración.

Un sistema inercial está formado por dos módulos principales: un sistema de medida inercial conocido como IMU (Inertial Measurement Unit), una unidad de pre-procesamiento. Una IMU utiliza normalmente un acelerómetro y un giroscopio que son capaces de medir los 3 ejes ortogonales (x,y,z).

3.3 Unidad de Medida Inercial (IMU)

La unidad de medida inercial (IMU) está compuesta por los sensores inerciales montados dentro de la unidad inercial, normalmente un acelerómetro y un giroscopio, ambos capaces de medir los tres ejes. En la siguiente figura se visualiza una IMU típica.

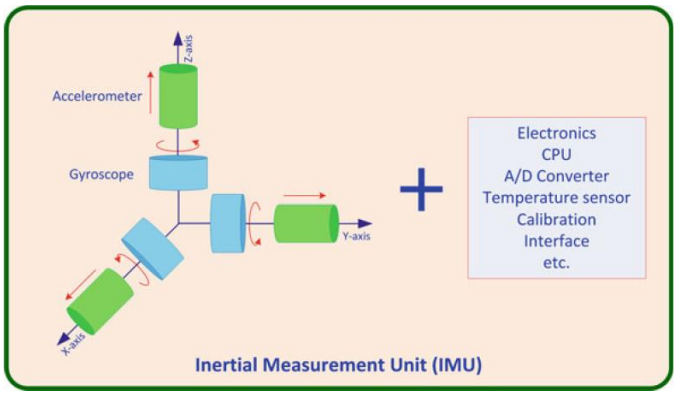


Figura 8: Componentes de una IMU

Los acelerómetros miden el movimiento lineal, mientras que los giroscopios miden el movimiento angular, ambos en las tres direcciones ortogonales. Normalmente y como se observa en la figura 8, los sensores están en paralelo en cada uno de los ejes.

También una IMU puede contener la electrónica necesaria para la calibración de los sensores, dispositivos convertidores analógicos digitales e interfaces de comunicación.

3.4 Sistema básico de navegación

Como se ha mencionado anteriormente, el posicionamiento inercial está basado en la diferencia de la posición que se obtiene con una doble integración de la aceleración en función del tiempo. Al tener tres dimensiones, no es tan simple obtener la aceleración que lleva el vehículo, por este motivo, se consideran tres casos diferentes: en una dirección, en dos direcciones y tridimensional.

3.4.1 Navegación basada en Una dirección

Para comprender el sistema tridimensional es más fácil empezar explicando el sistema unidireccional. En él, se considera que el vehículo se desplaza en una línea recta como muestra la figura 9.

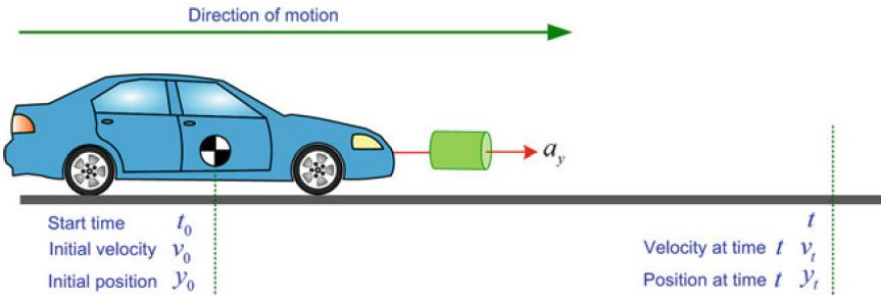


Figura 9: Navegación en una dimensión

De esta forma, el cálculo de la velocidad y el desplazamiento del vehículo se realiza de la siguiente manera:

$$v_t = \int a_y dt = a_y t + v_0$$

$$y_t = \int (a_y t + v_0) dt = \frac{1}{2} a_y t^2 + v_0 t + y_0$$

3.4.2 Navegación en Dos Dimensiones

Al tener un sistema bidimensional, el cálculo matemático se hace más complejo. Se necesitan los datos de la aceleración en dos direcciones y además es necesario un giroscopio para detectar el movimiento de rotación en la dirección perpendicular al plano de movimiento.

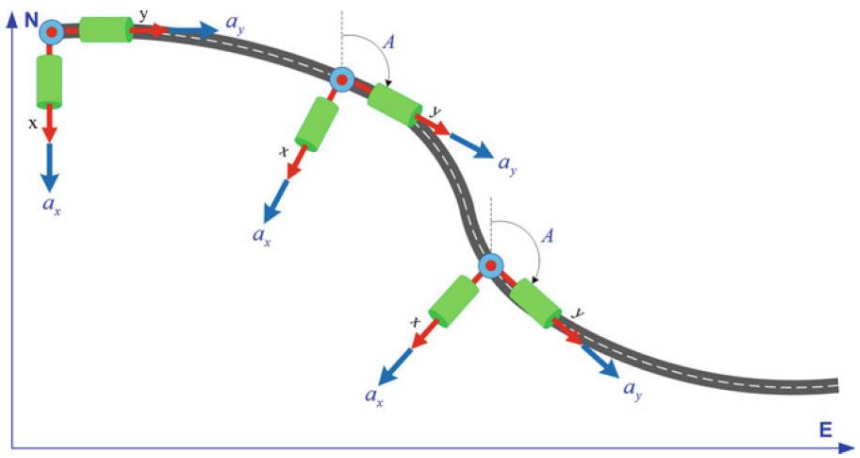


Figura 10: Ejemplo de la Navegación en un Sistema bidimensional

Como ahora hay que tener en cuenta la aceleración de los ejes x-y, se necesita hacer una transformación para dejar únicamente una aceleración válida para cada uno de los ejes. Esto se consigue de la siguiente manera:

$$a_E = a_y sinA + a_x cosA$$

$$a_N = a_y cosA - a_x sinA$$

Por lo tanto, la velocidad que se obtiene es la integral de cada aceleración por separado. Para determinar la distancia recorrida, se realiza la doble integral como en casos anteriores. La velocidad en un sistema bidimensional se realiza de la siguiente manera:

$$v_E = \int (a_y sinA + a_x cosA) dt$$

$$v_N = \int (a_y cosA - a_x sinA) dt$$

3.4.3 Navegación en Tres Dimensiones

La navegación tridimensional requiere un acelerómetro que mida tres ejes (x, y, z) y un giroscopio que mida tres ángulos de giro (cabeceo, alabeo, azimut). En este modelo, interviene la gravedad, por lo que se hace más complejo el cálculo de la aceleración.

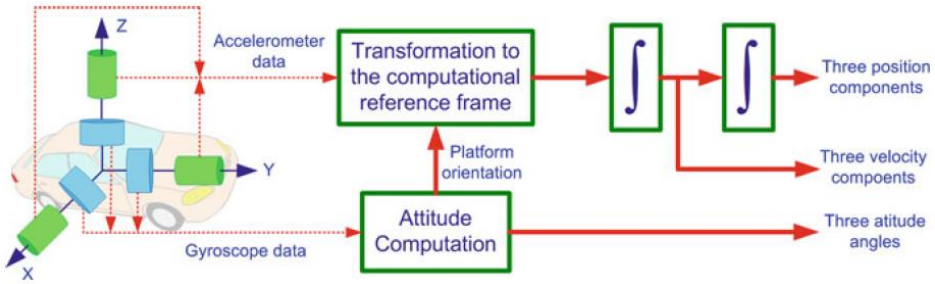


Figura 11: Principio general de la navegación en 3D

En el caso que el acelerómetro se encuentre en la posición en donde el eje **Z** indique la gravedad, se puede considerar de la siguiente manera:

$$f_x = 0; f_y = 0; f_z = g$$

Pero como el eje **Z** no siempre estará paralelo a la gravedad, este puede inclinarse un ángulo llamado cabeceo. De este modo, las medidas tomadas por los ejes **X** e **Y** del acelerómetro son:

$$f_x = 0; f_y = gsin(p); f_z = gcos(p)$$

Como se observa en la siguiente figura, al rotar el cabeceo (pitch p) un cierto ángulo, las mediadas se ven afectadas por este ángulo. Dependiendo de sobre qué eje ha sido la rotación, se verán afectados los otros ejes.

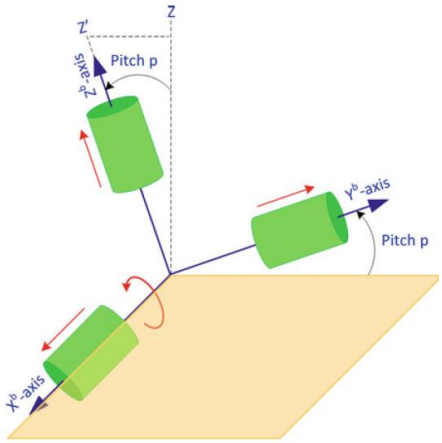


Figura 12: Medidas tomadas con un cabeceo p (Pitch p)

Ahora si se rota el eje **Y** un ángulo r, se produce el mismo efecto anterior. Esto de manera matemática se expresa como:

$$f_x = -gcos(p)\sin(r)\,; f_y = gsin(p); f_z = gcos(p)\cos(r)$$

A medida que va rotando sobre los diferentes ejes, los demás se ven afectados por la g en función de senos y cosenos del ángulo de rotación. Para determinar estos movimientos, existen las denominadas rotaciones de Euler, las cuales parametrizan estas rotaciones.

3.5 Introducción a los filtros

Los filtros son elementos normalmente pasivos y con bajas pérdidas al paso de señales en las frecuencias a las que deja pasar, mientras que otras frecuencias son atenuadas. La función de transferencia del filtro se caracteriza por varios parámetros como su respuesta en amplitud y a la fase, las bandas de paso, el rizado en la banda de paso, la pendiente de atenuación, etc.

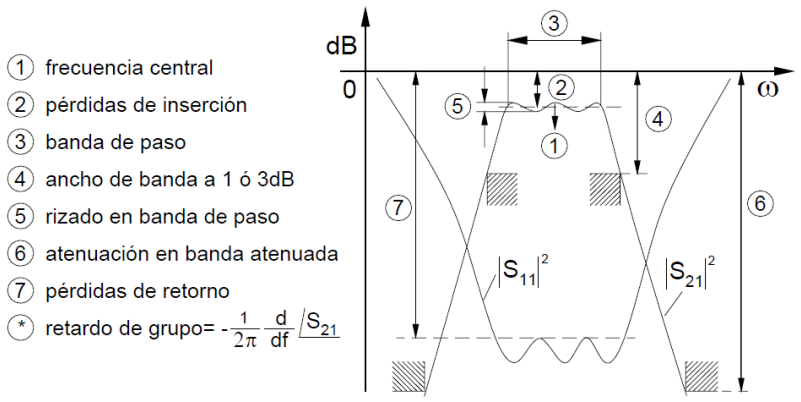


Figura 13: Parámetros básicos de un filtro

Dependiendo de su funcionamiento, y de cómo se desee filtrar, se pueden tener varios tipos de respuesta al filtrado. Las denominaciones de los filtros se pueden separar en 4 tipos. Los filtros paso-bajo que dejan pasar las señales que estén por debajo de la frecuencia de corte marcada. Los filtros paso-banda que permiten el paso de las señales que se encuentren en el ancho de la banda del filtro. Los filtros de banda-eliminada que funcionan justo al contrario que los paso-banda y dejan pasar las frecuencias que no estén dentro de su ancho de banda. Y por último los filtros paso-alto, que dejan pasar las señales más altas a partir de la frecuencia de corte.

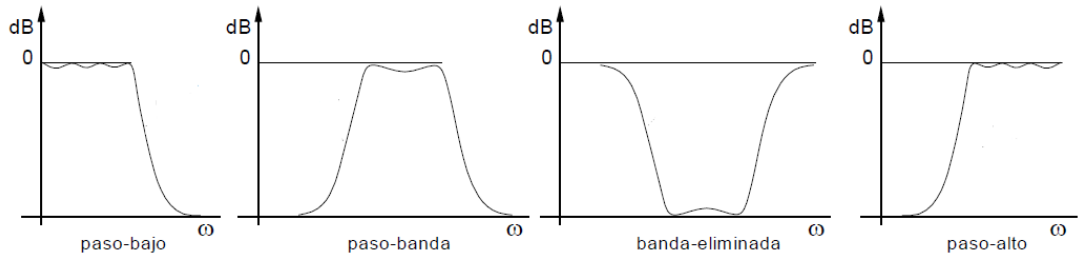


Figura 14: Respuesta típicas de los Filtros

Los filtros son utilizados en innumerables aplicaciones como en audio para tener un rango de frecuencia en la zona audible, en los dispositivos de transmisión donde se utilizan filtros paso-banda a varios megahercios para separar los canales, e incluso son utilizados para eliminar los 50Hz de la red eléctrica. La desventaja de utilizar los filtros es que añaden un retraso o un cambio de fase a la señal de entrada.

3.6 Filtros digitales

Los filtros digitales tienen la ventaja de que se pueden implementar algorítmicamente por microprocesadores, lo cual facilita acondicionar señales una vez

digitalizadas. Además, dado que la característica de transferencia y tipo dependen de unos simples coeficientes, estos filtros son altamente configurables dando una alta flexibilidad a su diseño, frente a los filtros analógicos clásicos que se basan en componentes electrónicos que tienen que ser reemplazados si se desea cambiar la característica del filtro.

El método tradicional de diseño de filtros IIR en tiempo discreto se basa en la transformación de un filtro en tiempo continuo en un filtro discreto que cumpla las especificaciones preestablecidas. El hecho de que los diseños de filtros en tiempo continuo se puedan trasladar a diseños de filtros en tiempo discreto es totalmente independiente de que el filtro en tiempo discreto se vaya a utilizar en una configuración de un sistema lineal e invariante en el tiempo para procesar las señales en tiempo continuo. [7]

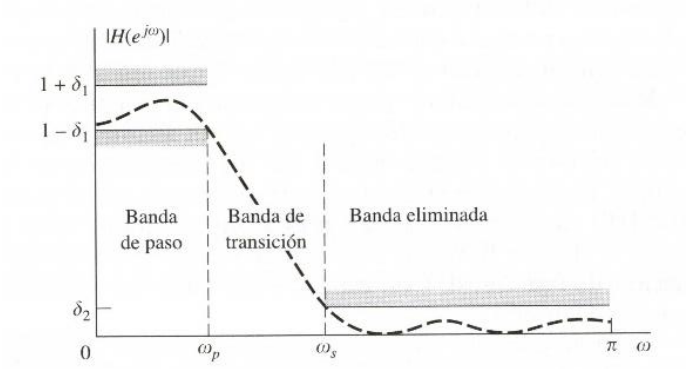


Figura 15: Especificaciones correspondientes del filtro en tiempo discreto.

Al diseñar un filtro en tiempo discreto transformando un filtro prototipo en tiempo continuo, las especificaciones del filtro en tiempo continuo se obtienen mediante la transformación de las especificaciones del filtro en tiempo discreto deseado. Se obtienen a continuación la función de transferencia $H_c(s)$ o la respuesta al impulso $h_c(t)$ del filtro en tiempo continuo utilizando uno de los métodos de aproximación establecidos para el diseño de filtros en tiempo continuo. Seguidamente se obtiene la función de transferencia $H(z)$ o la respuesta al impulso $h(n)$ del filtro en tiempo discreto aplicado a $H_c(s)$ o a $h_c(t)$.

Al realizar esta transformación se desea normalmente que la respuesta en frecuencia del filtro en tiempo discreto resultante mantenga las propiedades de la respuesta en frecuencia del filtro en tiempo continuo. Esto implica que el eje imaginario del plano s se transforme en la circunferencia unidad del plano z. Y además si el filtro en tiempo continuo es estable, ha de ser estable en tiempo discreto. Esto significa que si un sistema en tiempo

continuo sólo tiene polos en el semiplano izquierdo s, el sistema en tiempo discreto sólo debe tener polos en el interior de la circunferencia unidad del plano z.

3.6.1 Diseño de filtros IIR en tiempo Discreto

La manera normal de diseñar un filtro IIR en tiempo discreto se basa en la transformación de un filtro en tiempo continuo en un filtro en tiempo discreto que cumpla con las especificaciones establecidas. Esta solución es razonable por varios motivos:

- El arte del diseño de filtros IIR en tiempo continuo está muy avanzado, por lo cual, resulta una ventaja utilizar los procedimientos de diseño que ya se han implementado para los filtros en tiempo continuo.
- Hay gran cantidad de métodos útiles de diseño de filtros IIR en tiempo continuo que son fáciles de calcular con fórmulas de diseño simple. Por tanto, los métodos para calcular los filtros discretos a partir de estas fórmulas en tiempo continuo, son fáciles de realizar.
- Los métodos de aproximación estándar que funcionan para calcular los filtros en tiempo continuo, no producen el mismo efecto cuando se aplican directamente a un filtro IIR en tiempo discreto.

Hay dos métodos de realizar los filtros IIR en tiempo discreto a partir de filtros en tiempo continuo. Se puede utilizar el diseño de filtros mediante la invarianza al impulso y mediante la transformación bilineal.

El procedimiento de diseño mediante **la invarianza al impulso** para transformar un filtro en tiempo continuo a discreto consiste en que la respuesta al impulso del filtro en tiempo discreto es proporcional a muestras equiespaciadas de la respuesta al impulso del filtro en tiempo continuo. Cuando se utiliza el método de invarianza al impulso para diseñar un filtro en tiempo discreto con una respuesta en frecuencia especificada, el interés se centra especialmente en la relación entre las respuestas en frecuencia de los filtros en tiempo discreto y tiempo continuo.

Las frecuencias se relacionan mediante un cambio de escala lineal de la frecuencia, concretamente $\omega = \Omega T_d$ para $|\omega| < \pi$. Como a ningún filtro en tiempo continuo se le puede limitar la banda, se producen interferencias entre los términos causando solapamiento. Sin embargo si el filtro en tiempo continuo se aproxima a cero a altas

frecuencias, el solapamiento puede ser razonablemente pequeño y se puede obtener un filtro útil en tiempo discreto.

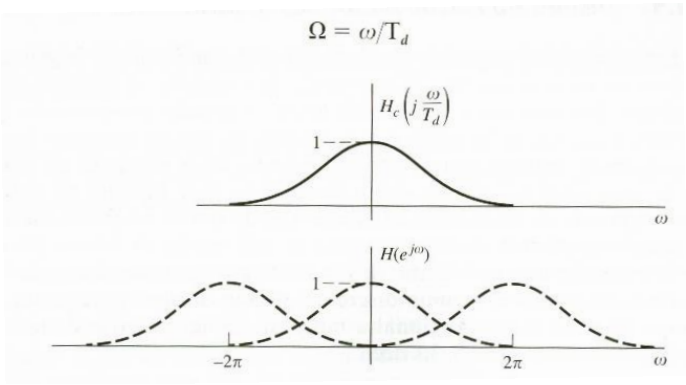


Figura 16: Ilustración del solapamiento en la técnica de diseño de invarianza al impulso.

La técnica de **transformación bilineal** evita el problema de solapamiento utilizando una transformación algebraica entre las variables *s* y *z* que transforma el eje $j\Omega$ del plano *s* en una revolución de la circunferencia unidad del plano *z*. Como $-\infty \leq \Omega \leq \infty$ se transforma en $-\pi \leq \omega \leq \pi$, la transformación entre las variables de frecuencia en tiempo continuo y en tiempo discreto ha de ser no lineal. Por tanto el uso de la transformación bilineal está limitado a situaciones en las que la correspondiente modificación del eje de frecuencias es aceptable.

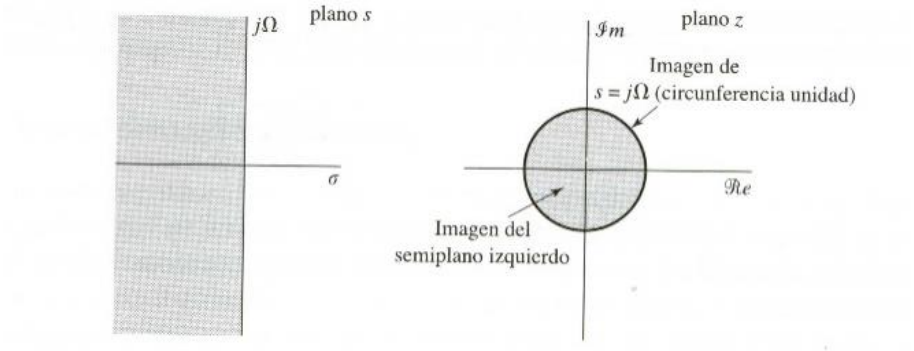


Figura 17: Transformación del plano s en plano z mediante la transformación bilineal

Para realizar los filtros digitales, se utiliza la biblioteca dsp-collection [11], la cual será comentada más adelante. Esta biblioteca ya viene implementada en Java y hace que el desarrollo del proyecto se realice de manera más rápida, evitando realizar tediosos cálculos para cambiar las configuraciones de los filtros.

4.0 Descripción de la Unidad Inercial

4.1 Introducción

Este capítulo va a detallar los recursos hardware y software que se utilizarán en el proyecto. Se describirá el microcontrolador, sus características y los sistemas utilizados como son los buses de comunicación. Seguidamente se explicará la unidad inercial (IMU), los pines de conexión y el bus de comunicación que se emplea para comunicarse con el microcontrolador. También se detallan los esquemáticos que establecen las conexiones entre el microcontrolador y la IMU, además de otras conexiones importantes. Para finalizar se describirá el programa que se ha empleado para desarrollar la programación en C y las pruebas de programación.

4.2 Microcontrolador Atmega324A

El microcontrolador que se utilizará para el desarrollo de este proyecto es el Atmega324A. Este microcontrolador pertenece a la familia AVR de ATMEL, utiliza la tecnología de alto rendimiento basada en RISC de 8 bits.

4.2.1 Características

Este microcontrolador tiene una memoria ISP flash de 32 KB, 1KB de memoria EEPROM y 2 KB de SRAM. Cuenta con 32 líneas de I/O y 32 registros de propósito general. Además de 2 Timer/Counter de 8 bit y otro de 16 bit, ambos con Prescaler y modos de comparación separados, con funciones de PWM (hasta seis canales PWM).

Dos puertos USART orientados para la conexión serie por medio de 2 líneas (TX y RX). Dispone de una Interfaz Serie de dos cables (I^2C) que ofrece una velocidad de transferencia de datos de hasta 400 kHz. Dos buses SPI para la comunicación serie, uno de ellos es compartido con la USART

Además cuenta con 8 canales para el convertidor analógico digital de 10bit de resolución. Su rango de funcionamiento va desde los 1.8 v hasta los 5.5 v. Para su programación, cuenta con la Interface JTAG

4.2.2 Diagrama de Bloques

El núcleo del microcontrolador AVR combina instrucciones RISC con 32 registros de propósito general conectados directamente a la Unidad Aritmética Lógica (ALU), permitiendo que dos registros independientes puedan acceder a una instrucción en un único ciclo de reloj. El resultado de esta arquitectura es que se tiene un código más eficiente siendo hasta 10 veces más rápido que un microcontrolador CISC.

Como se puede observar en la figura 18, la generación de reloj, los circuitos osciladores y el Watchdog funcionan con la frecuencia del cristal externo. El conversor analógico digital se encuentra en el puerto A. El comparador analógico, el SPI y uno de los Timer/Counter de 8 bits están conectados al puerto B. El JTAG y la interfaz I^2C se encuentran en el puerto C. La USART y tres Timer/Counter están conectados al puerto D.

Para maximizar el rendimiento y paralelismo, el AVR utiliza una arquitectura Harvard en donde la memoria y los buses de datos y de programa están separadas. Permitiendo de esta manera, ejecutar una instrucción con un único nivel de pipeline. Mientras una instrucción está siendo ejecutada, la siguiente instrucción se pre-carga desde la memoria de programa.

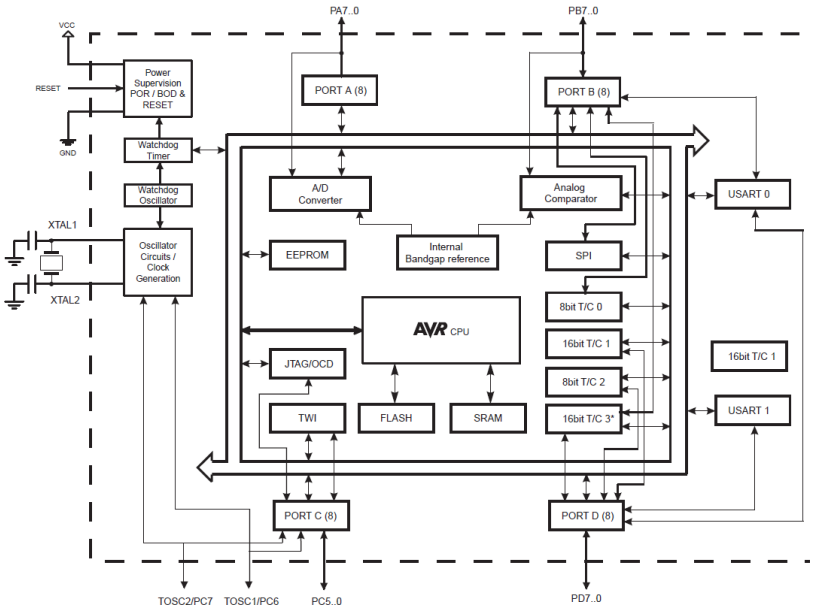


Figura 18: Diagrama de bloques del microcontrolador

Al tener los registros de propósito general conectados a la ALU, permiten un rápido acceso y manipulación de los registros con un único ciclo de reloj. Lo que permite realizar una operación de la ALU por cada ciclo de reloj. En una operación típica de la ALU, dos operandos son leídos de los registros, se ejecuta la operación y el resultado se guarda en el registro con un solo ciclo de reloj.

La memoria de programa Flash está dividida en dos secciones, el programa de arranque y el programa de aplicación. Ambas secciones bloquean determinados bit contra la escritura para protegerlos. Las instrucciones de SPM (Store Program Memory) que escribe dentro de la memoria de aplicación Flash, debe de estar dentro de la sección de arranque del programa.

Durante una interrupción o un salto en el programa, la dirección de retorno se escribe en el contador de programa (PC) en la pila. La pila se encuentra en la memoria SRAM y por este motivo está limitada al tamaño total de la SRAM y al espacio utilizado. Todos los programas deben de inicializar el SP (Puntero de Pila) en la rutina del Reset (antes de que se llame a una subrutina o una interrupción sea ejecutada). El Puntero de Pila es leído y escrito accediendo al espacio de E/S.

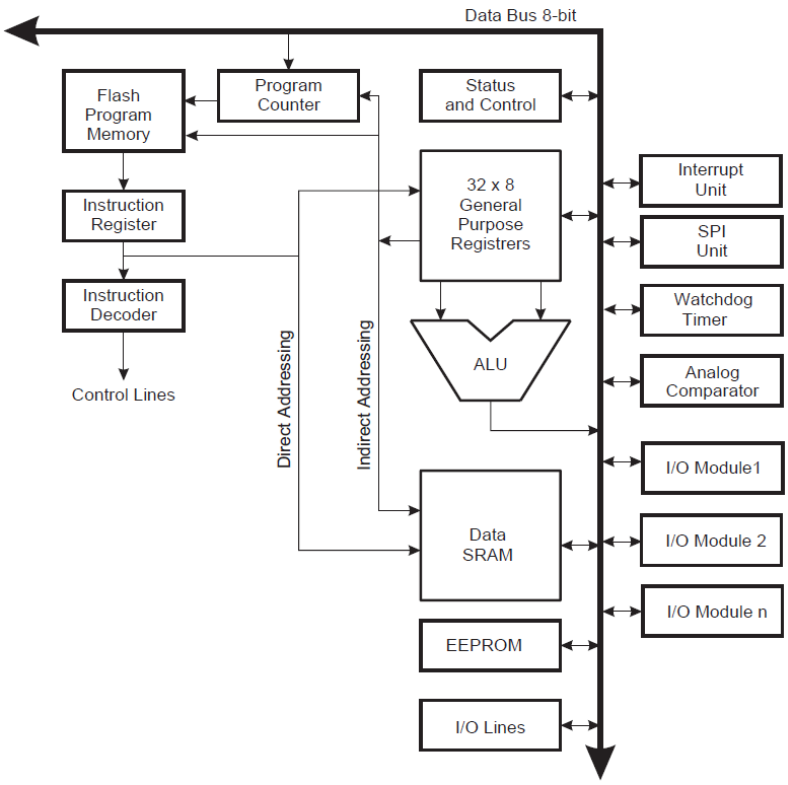


Figura 19: Arquitectura AVR

4.2.3 Sistema de reloj

En la figura 20 se puede observar cómo se realiza la distribución de reloj. Los relojes no necesitan estar activados al mismo tiempo, cada reloj se configura por separado y se pueden desactivar para ahorrar consumo.

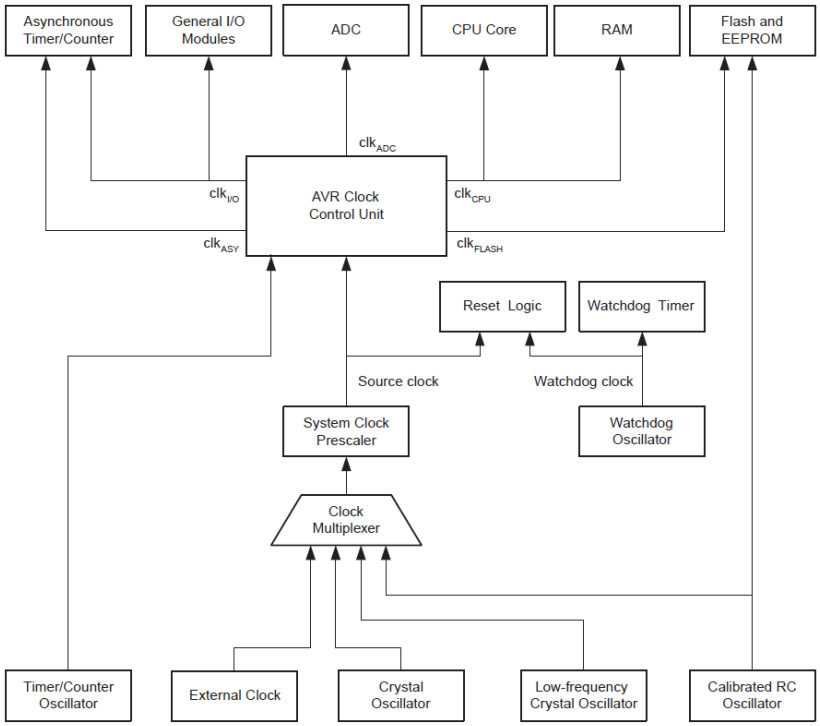


Figura 20: Distribución del sistema de reloj

Los relojes externos como el cristal oscilador, el cristal oscilador de baja frecuencia, el oscilador RC son introducidos a un multiplexor en donde se elige la frecuencia de trabajo deseada. Después del Multiplexor, se introduce la señal de reloj en un Prescaler programable para dividir la frecuencia. La frecuencia del Prescaler, se utiliza para la unidad de control y el watchdog. La unidad de control reparte la frecuencia del Prescaler hacia los Timer/Counter, los módulos de E/S generales, los convertidores analógicos digital, la CPU, la RAM y la Flash.

4.2.4 Puertos de E/S

Todos los puertos son bi-direccionales con opción de conectarlos a pull-up internamente. Para configurar los pines hay tres registros, el registro DDxn, el PORTxn y el PINxn.

El DDxn es un registro que sirve para configurar la dirección de los pines. Si el DDxn es escrito a nivel alto, el pin queda configurado como salida, por el contrario si el DDxn es escrito a nivel bajo, es configurado como entrada.

El PORTxn se utiliza para escribir un nivel alto o bajo a la salida del pin cuando está configurado como salida. Si el pin está configurado como entrada, el PORTxn se utiliza para activar/desactivar un interno pull-up. Los pines de los puertos están en tri-estado cuando se activa el Reset, si el reloj no está funcionando.

El PINxn se emplea para leer el valor que tiene el puerto.

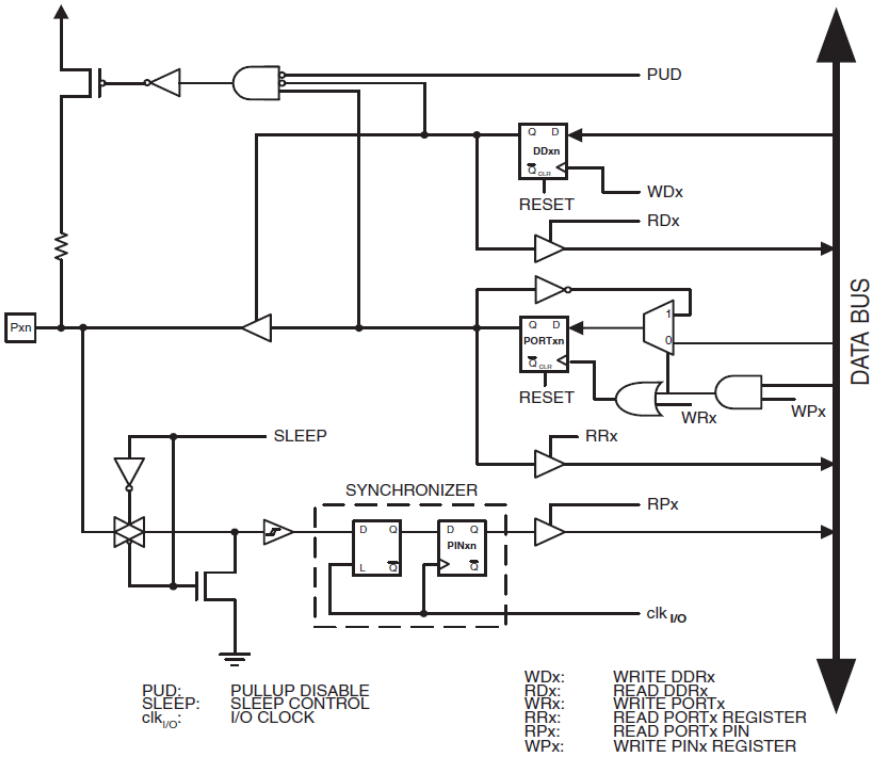


Figura 21: Esquema de configuración de los puertos de E/S

4.2.5 Timer/Counter

El Timer/Counter tiene dos registros de comparación y da soporte a la señal PWM. Permite mantener el programa en ejecución mientras realiza la generación de pulsos. Como se observa en la figura 22, el Timer/counter utiliza el registro TCNTn y el Comparador utiliza los registros OCRnA y OCRnB.

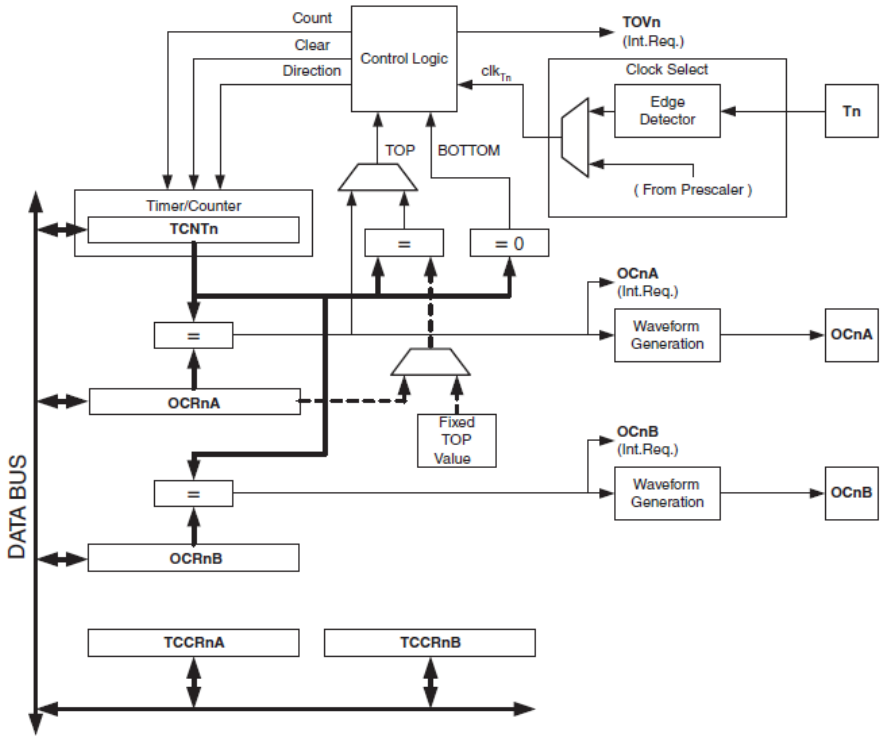


Figura 22: Diagrama del Timer/Counter de 8 bit

El Timer/Counter puede utilizar el reloj interno generado por el Prescaler, o utilizar una fuente de reloj externa conectada al pin Pn. La fuente de reloj se utiliza para incrementar o decrementar el valor de puesto en el TCNTn. El Timer/Counter estará deshabilitado cuando no se seleccione ninguna fuente de reloj (TCCR0B).

La solicitud de interrupción se hace cuando el Flag de interrupción TIFRn se activa. Todas las interrupciones tienen una máscara de interrupción (TIMSKn) para habilitar o deshabilitar las interrupciones. El Timer tiene varios modos de funcionamiento, pero el que se va a utilizar será el modo CTC (Limpia el Timer al ser igual la Comparación).

El funcionamiento se puede ver en la siguiente figura, en donde se aprecia que el TCNTn empieza a contar y cuando alcanza el valor del registro OCRn se borra y empieza a contar nuevamente. Cada vez que la comparación del TCNTn sea igual al OCRn se producirá una interrupción que hay que desenmascarar en el registro OCIEnA.

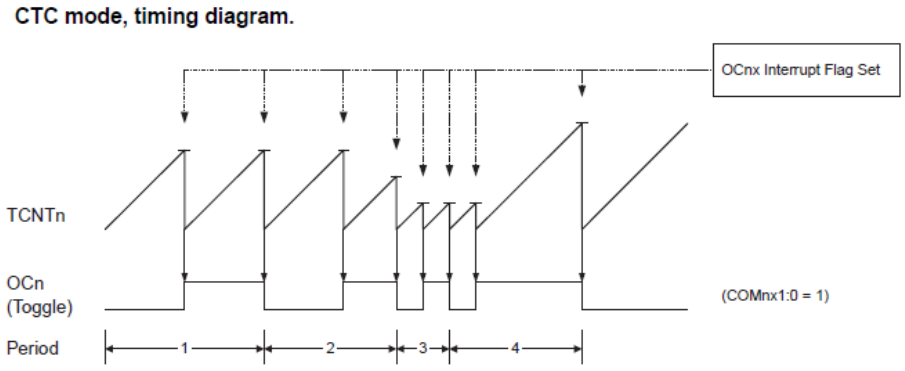


Figura 23: Modo de funcionamiento CTC

El Timer/Counter de 16 bits es muy parecido sólo que utiliza registros de 16bits. Como los registros son de 16 bits, hay que acceder a ellos de manera especial. Estos pueden acceder al bus de datos de 8 bits de la CPU almacenando en un registro temporal la parte alta del registro. Este registro temporal, es compartido entre todos los registros de 16bit que tiene el timer.

Cuando se escribe la parte baja del registro de 16 bits, la parte alta es almacenada en el registro temporal y cuando se produce un ciclo de reloj, la parte baja y la parte alta que se encuentra en el registro temporal, son escritos al mismo tiempo en el registro de 16 bit. Para leer, cuando la CPU lee la parte baja, la parte alta se copia en el registro temporal en el mismo ciclo de reloj. Luego la CPU lo que hace es leer este registro temporal para conocer los 16bit.

4.2.6 Bus de comunicación SPI

La Interface Periférico Serie dota al AVR de una alta velocidad de transferencia de datos síncrona entre el microcontrolador y los dispositivos periféricos. El bus USART, al ser una interfaz de comunicación serie, puede ser usado como SPI maestro.

En el diagrama de bloques se puede observar como el SPI contiene un bloque de control encargado de generar las señales de interrupción, habilitar el SPI, configurar si actúa como maestro o esclavo, etc. Otro bloque dedicado a configurar el sistema de reloj el cual contiene un Prescaler que divide la frecuencia del XTAL. Un bloque lógico para controlar las características de las señales transmitidas y recibidas. Por último un bloque de almacenamiento de dato el cual actúa como un registro de desplazamiento.

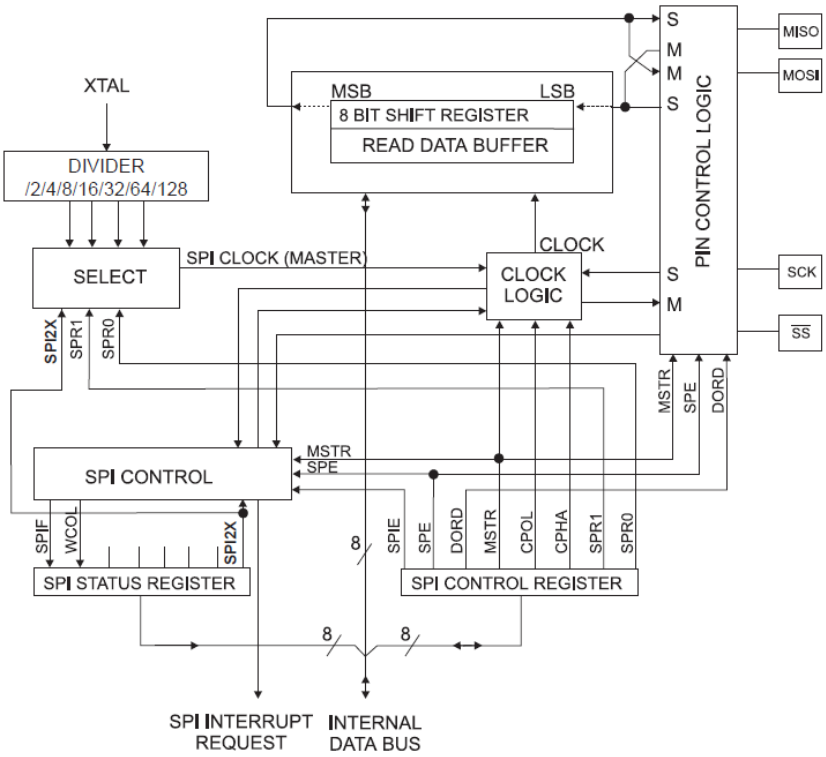


Figura 24: Diagrama de bloques del SPI

La conexión entre el Máster y el Esclavo se muestra en la figura 25 en donde se ven los dos registros de desplazamiento y la señal de reloj que controla el Máster. El máster genera la señal de reloj y envía los datos por la señal MOSI. Al esclavo ir recibiendo los datos, va enviando los datos que contiene su registro por su salida MISO.

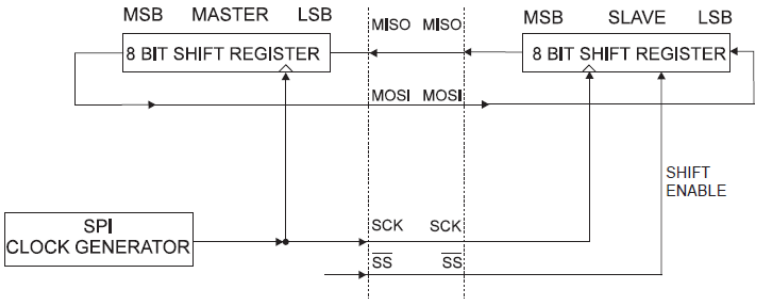


Figura 25: Conexión entre el Máster y el Esclavo del SPI

Para sincronizar esta transferencia de datos, el maestro controla la señal de selección ($\overline{SS}$), la cual activa para indicarle al esclavo que va a empezar a enviarle datos. Cuando termina la transmisión, desactiva la señal $\overline{SS}$. El control de la línea $\overline{SS}$ se ha de hacer mediante el software, debido a que no es generado automáticamente. Esto permite que un máster pueda comunicarse con varios esclavos activando o desactivando la señal $\overline{SS}$ de

cada uno de ellos por separado. No puede haber más de un dispositivo esclavo activado al mismo tiempo.

4.2.7 Bus de comunicación USART

El bus Universal Síncrono y Asíncrono de Recepción y Transmisión serie es uno de los más flexibles y altamente integrado en los dispositivos. En el diagrama de bloques se divide las partes de la USART en tres partes: Generador de reloj, Transmisor y Receptor.

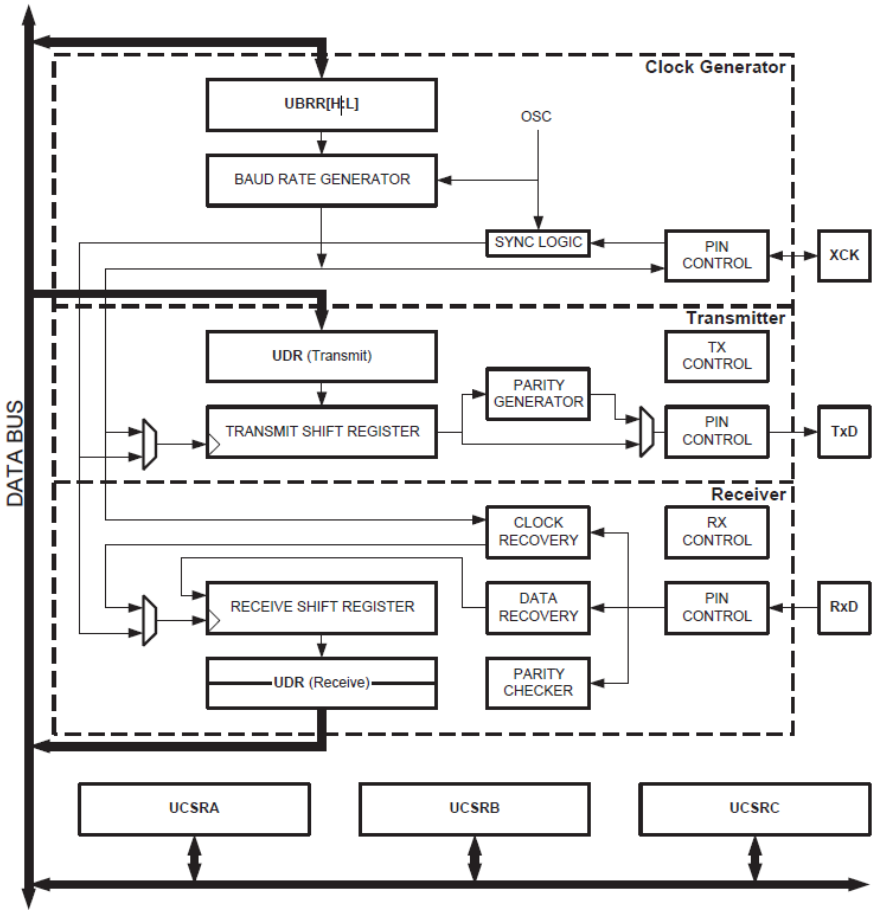


Figura 26: Diagrama e Bloques de la USART

El sistema generador de reloj se encarga de la sincronización de reloj y de generar la Tasa Binaria a la que serán trasmitidos los datos. Soporta cuatro modos de funcionamiento: normal, doble, sincronismo maestro y sincronismo esclavo. Estos modos son controlados por los registros de control de la USART.

El registro Baud Rate (UBRRn) se programa dependiendo de cómo esté configurado el Prescaler. El funcionamiento es simple, hay un contador que se va decrementando según la frecuencia del reloj, cuando se carga un valor en el registro UBRRn el contador

decrementa esa cantidad hasta llegar a cero y vuelve a cargar el valor del registro. Cada vez que realiza esta operación, genera a la salida la señal de reloj a esta frecuencia.

Cuando se programa para que funcione en modo Asíncrono normal (U2Xn = 0):

$$BAUD = \frac{f_{osc}}{16(UBRRn + 1)}$$

Si se programa para que funcione en modo Asíncrono con doble velocidad (U2Xn = 1):

$$BAUD = \frac{f_{osc}}{8(UBRRn + 1)}$$

Cuando funciona como modo maestro síncrono:

$$BAUD = \frac{f_{osc}}{8(UBRRn + 1)}$$

El formato de la trama es muy flexible, pudiéndose configurar varios parámetros de sincronización y de verificación de errores. Se pueden realizar hasta 30 combinaciones posibles modificando los bits de stop, los bits de datos, el bit de Start y el bit de paridad.

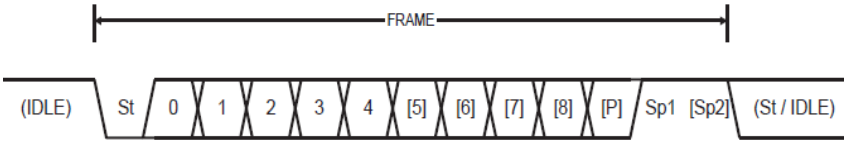


Figura 27: Formato de una trama USART

Antes de empezar la comunicación se tienen que configurar cómo es el formato de la trama que se va a utilizar, la velocidad de transmisión, habilitar la transmisión y/o recepción. Esta configuración se realiza manipulando los siguientes registros:

- UBRRn: Registro de la Tasa Binaria de la USART (calculado con la fórmula del BAUD).
- UCSRnA: Registro de Control y Estado de la USART (contiene los flag de recepción recibida, transmisión completada, registro de datos vacío, error en la recepción, overrun, error de paridad y además es donde fijamos si se trabaja al doble de velocidad).

▪ UCSRnB: Registro de Control y Estado de la USART (se utiliza para activar la transmisión y recepción, habilitar interrupciones y configurar el tamaño de los datos).

4.3 Unidad de Medida Inercial LSM330DLC

En este apartado se explica el modelo de IMU utilizada para el proyecto. La unidad elegida ha sido el dispositivo LSM330DLC el cual contiene un sensor giroscópico triaxial y un acelerómetro triaxial.

4.3.1 Características principales

- Voltaje de alimentación analógico: 2,4 - 3,6V
- Voltaje de alimentación digital: 1.8V
- Modo bajo consumo
- 3 canales de aceleración y 3 de giro independientes
- ±2g/±4g/±8g/±16g
- ±250/±500/±2000 dps
- Interfaz serie SPI/I^2C (16 bit)
- Interrupciones programables

4.3.2 Diagrama de bloques

En el diagrama de bloques se puede observar cómo los sensores miden la aceleración $\vec{I}(\vec{a})$ y los giros $\vec{I}(\vec{\Omega})$ en el bloque sensor, el cual separa cada medida en los ejes **X,Y,Z**. Después de este bloque, los ejes son introducidos en dos multiplexores y amplificados. La salida del amplificador del giroscopio es demodulada y pasada por un filtro paso-bajo, antes de ser introducida en el convertidor analógico/digital. Del convertidor pasa a la unidad de control para ser enviada al microcontrolador a través de los buses de comunicación.

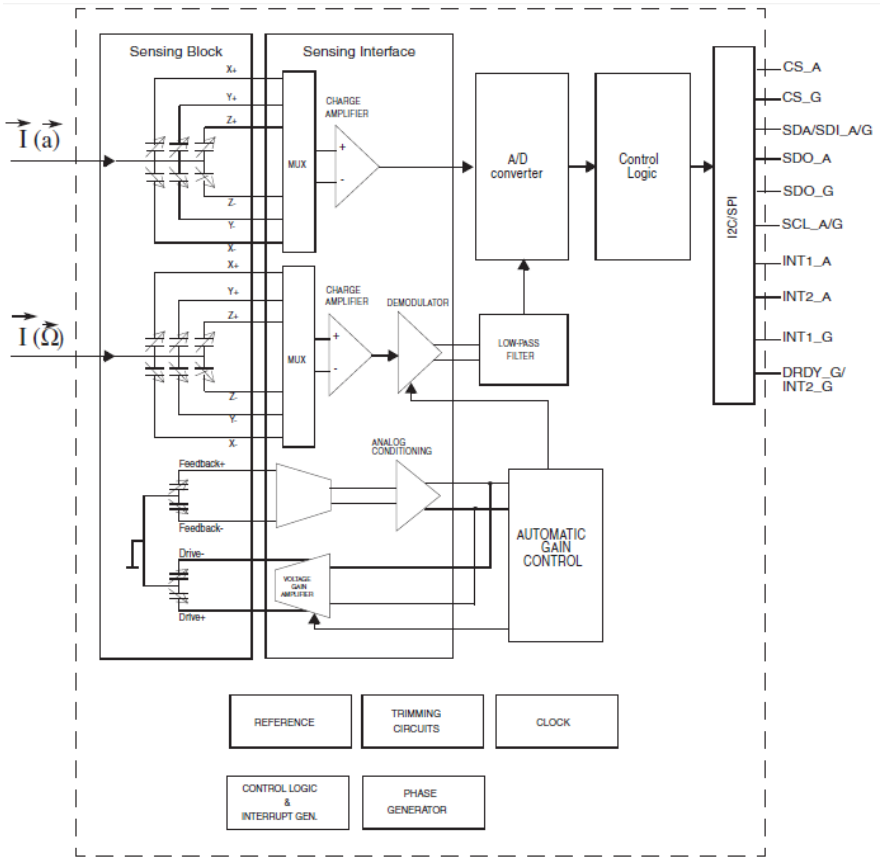


Figura 28: Diagrama de Bloques

4.3.3 Descripción de los pines

La dirección de medida de los ejes está dispuesta como indica la siguiente figura en donde el eje **X** está en paralelo al eje más largo del sensor, el eje **Y** está en paralelo al eje más corto del sensor y el eje **Z** está en dirección ortogonal a los otros dos ejes. Los pines se conectan como muestra la figura 29, recordando siempre que es una vista de la cara inferior del sensor *(bottom view)*.

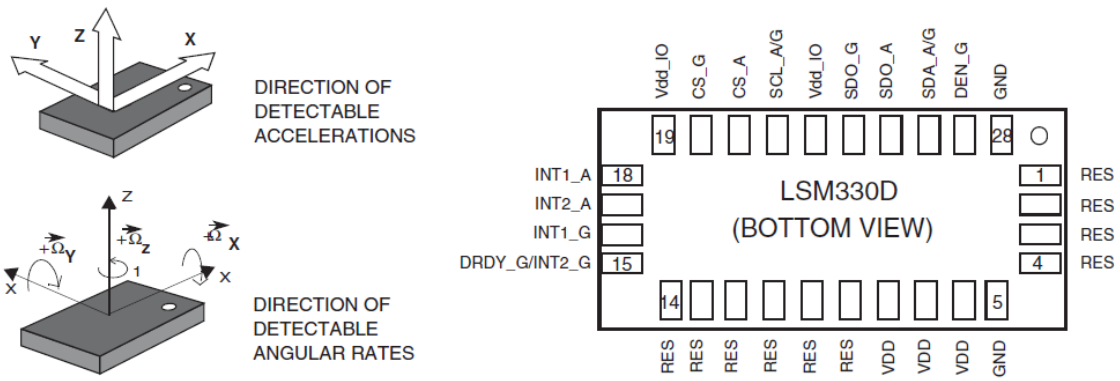


Figura 29: Pines de conexión

La conexión al bus SPI se realiza por los pines 20, 22, 24 y 26 para el giroscopio y los pines 21, 22, 25 y 26 para el acelerómetro. Los pines 22 y 26 son utilizados tanto para el acelerómetro como para el giroscopio puesto que son los pines de SPC y de entrada de datos (SDI) al LMS330D. El pin 20 es el de selección de Giroscopio y el pin 21 es el de selección de acelerómetro. Y por último se tienen los pines 24 y 25 que son de salida de datos SDO para el giroscopio y el acelerómetro en este orden.

Las demás conexiones que necesita son las alimentaciones digitales y analógicas, dos interrupciones utilizadas para el giroscopio y otras dos interrupciones utilizadas para el acelerómetro. Los pines en donde se conectan las interrupciones y las alimentaciones, se pueden observar en la Figura 29.

Dependiendo de cómo se configuren los registros del LSM330D, a la salida se obtendrá diferentes tipos de sensibilidad, esto quiere decir que se puede tener sensibilidad controlada con valores de ±2g/±4g/±8g y ±16g. Las medidas del giroscopio también se pueden controlar mediante los registros de control pudiéndose obtener a la salida valores de sensibilidad del orden de ±250, ±500 y ±2000 dps.

4.3.4 Bus de comunicación SPI

La secuencia de comunicación con el bus SPI que se necesita para acceder a los registros del LSM330D es la siguiente:

1. Señal de CS a nivel bajo.
2. El reloj SPC empezara con la frecuencia del reloj maestro.
3. En el flanco de bajada del reloj se recibe el bit más significativo que envía el máster.
4. Mientras se recibe el bit de máster, se envía el dato que estaba en el buffer.
5. Consecutivamente se sigue realizando la secuencia 3 y 4 hasta que el CS sea puesto a nivel alto.

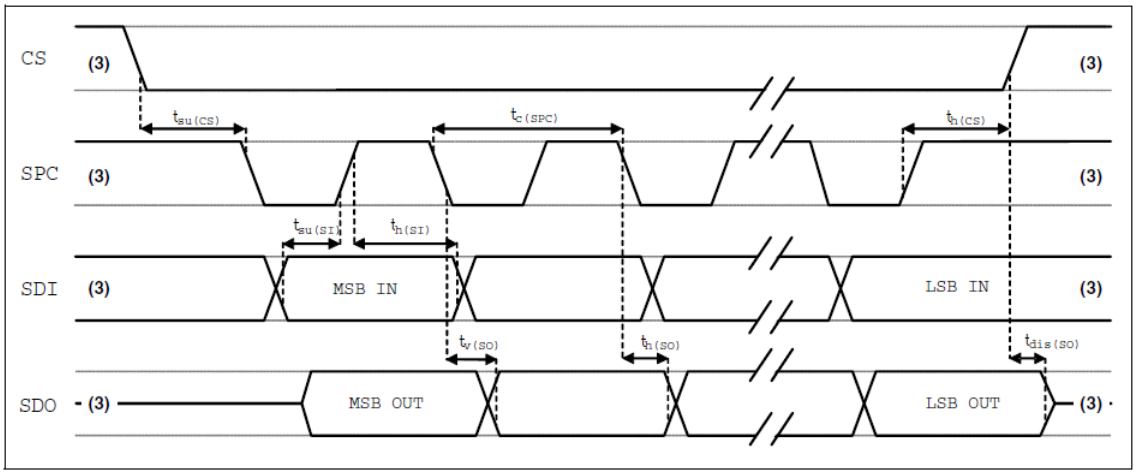


Figura 30: Diagrama de tiempo del SPI en modo esclavo

Cuando se quiere escribir en un registro del LSM330D, primero se envía la dirección del registro que se quiere escribir con el bit $R\bar{W}$ a nivel bajo para que realice un acceso de escritura. El segundo bit $M\bar{S}$ indica si es un acceso múltiple o simple, así que si se quiere acceder a un sólo registro, ha de tener un nivel bajo. Y los otros 6 bits restantes indican la dirección del registro. Una vez enviada la dirección del registro, se envía el dato que se desea escribir.

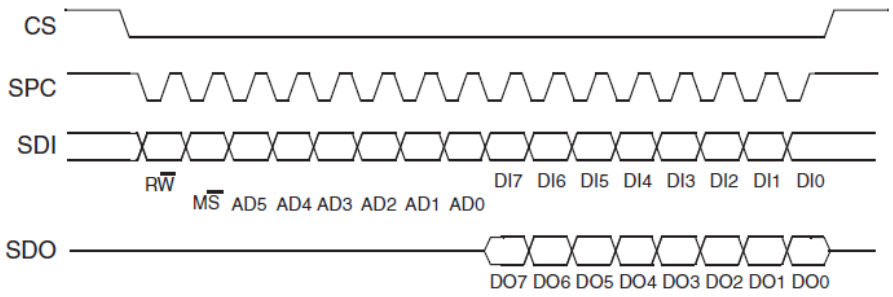


Figura 31: Acceso de escritura en el bus SPI

Cuando se desea leer un dato del sensor, se lee a través del bus SPI el registro que contiene el dato, para ello se envía primero la dirección del registro, con el bit más significativo $R\bar{W}$ a nivel alto para indicar que el acceso es de lectura. Una vez enviada la dirección del registro, se envía otro dato cualquiera, puesto que como el SPI funciona como un registro de desplazamiento, al escribir la segunda vez es cuando se recibe el dato que había sido pedido leer.

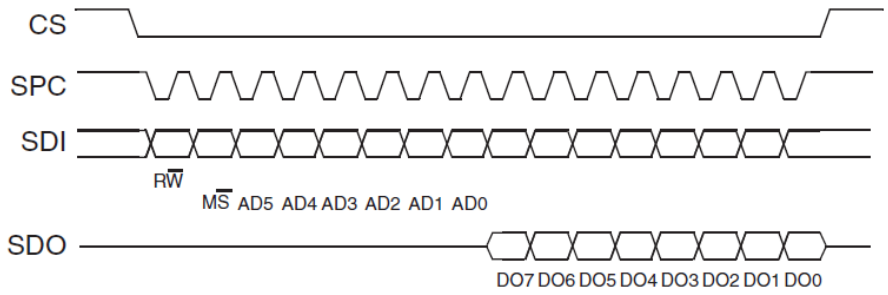


Figura 32: Acceso de lectura en el bus SPI

Cuando se quiere leer varios registros seguidos, se realiza la misma operación utilizada en la lectura normal pero con el bit $M\bar{S}$ a nivel alto. De esta manera, solo con poner la dirección del registro el cual se quiere empezar a leer, es suficiente. Mientras el CS tenga un nivel bajo, la señal SDO irá enviando los datos de los registros contiguos uno tras otro. Cuando el CS es puesto a nivel alto, se acaba la lectura y se tiene que empezar nuevamente escribiendo si el acceso es de lectura o escritura y si es acceso múltiple o simple.

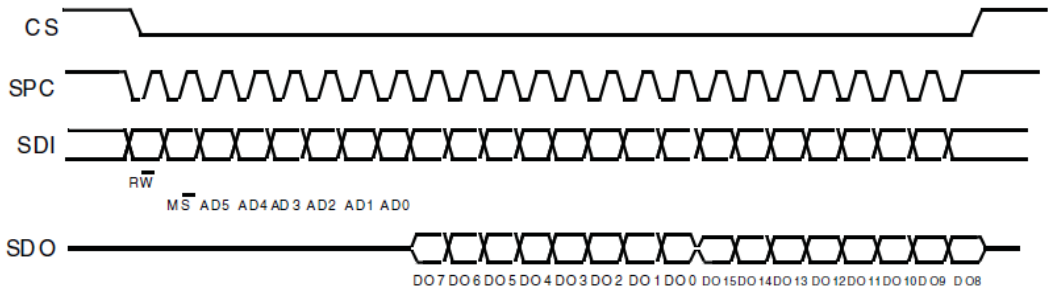


Figura 33: Acceso múltiple de lectura en el bus SPI

Los registros de configuración y control se encuentran en el datasheet del dispositivo en la página web del fabricante:

http://www.st.com/web/en/resource/technical/document/datasheet/DM00043695.pdf

4.4 Esquemáticos de conexión

4.4.1 Introducción

En este Trabajo Fin de Máster se ha utilizado un módulo inercial para aplicaciones generales, de diseño propio y suministrado por el tutor de este trabajo. A continuación se presentarán los aspectos más relevantes del diseño para el desarrollo de este trabajo.

4.4.2 Alimentación

El circuito de alimentación de la PCB es el siguiente:

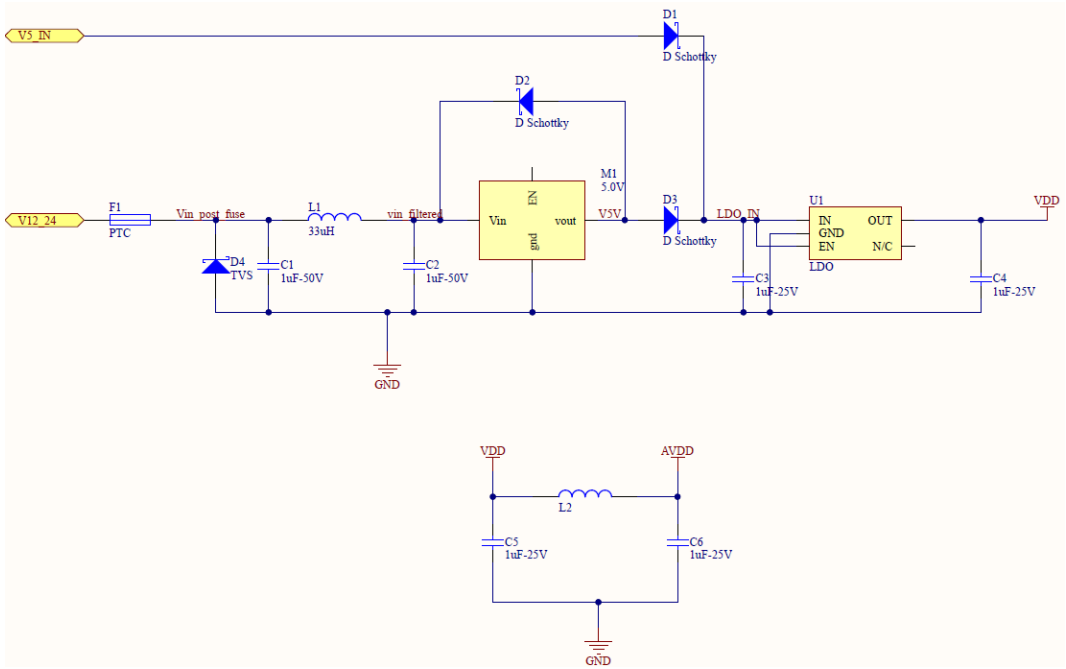


Figura 34: Esquemático de Alimentación

Se puede observar como tiene varios diodos Schotky de protección por si el circuito es conectado incorrectamente. Cuando se alimenta con 5 V, esta tensión entra directamente al regulador (de la serie TLV702xx) en donde se obtiene una salida constante de 2,8 V. Si se conecta a una tensión de 12-24 V, pasa por un convertidor DC/DC (ROF-78E) que la convierte a 5 V para que a continuación pase por el regulador TLV702xx. Siempre se tendrá a la salida una tensión de 2,8 V. Para pasar esta tensión digital que sale de regular a una tensión analógica, se realiza el esquema de la parte inferior de la figura 34.

4.4.3 Microcontrolador

Las conexiones al microcontrolador se pueden observar en la siguiente figura en donde se puede comprobar que el puerto SPI se conecta al puerto B del micro. Este puerto será el que se utilice para la comunicación con la IMU. También se puede apreciar como el bus USART está conectado al puerto D. Además de la conexión a los buses de comunicación, también se puede ver donde se conecta el cristal externo, donde se encuentran conectadas las interrupciones de la IMU, las conexiones para el JTAG y los pines de propósito general, los cuales no están conectados a ningún sitio.

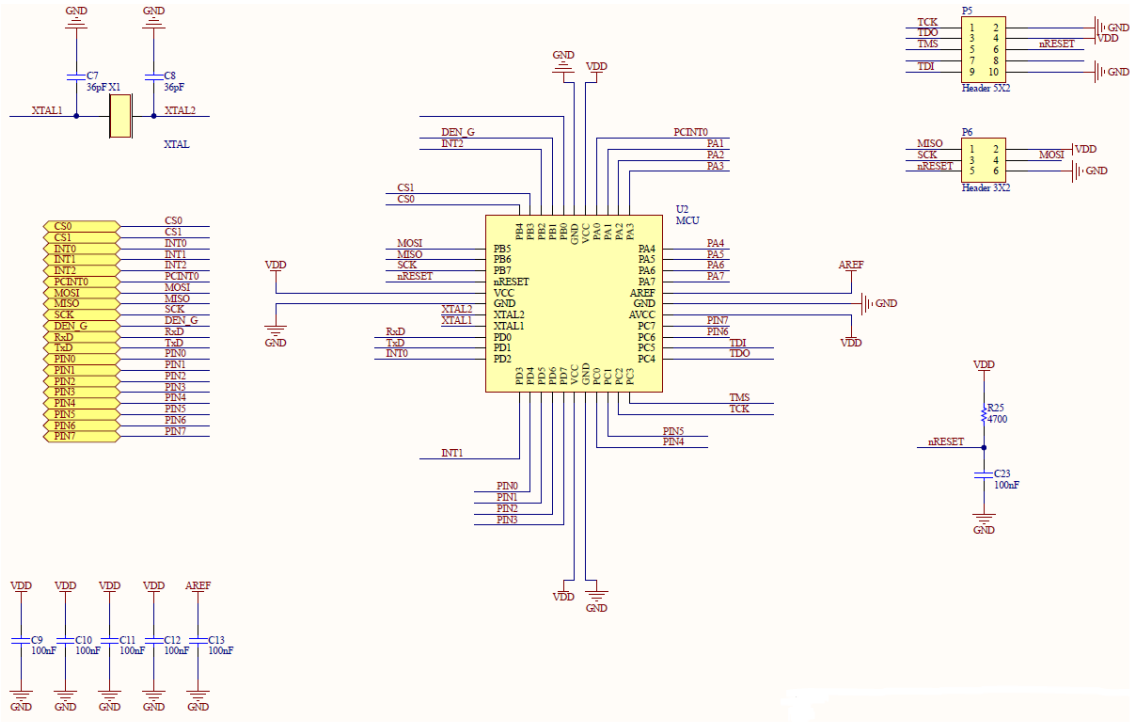


Figura 35: Esquemático del Microcontrolador

4.4.4 IMU (Unidad de Medida Inercial)

En el esquemático de la conexión de la unidad sensora, se puede observar los pines de conexión que van hacia el microcontrolador. Además se puede ver como se realizan los pull-ups a los CS y como deja la opción para soldar las resistencias R19 y R20 y hacer que SDO/SDA se utilicen con pull-ups. De la misma manera, se deja la opción de trabajar con pull-ups o pull-down la señal de habilitación de la IMU.

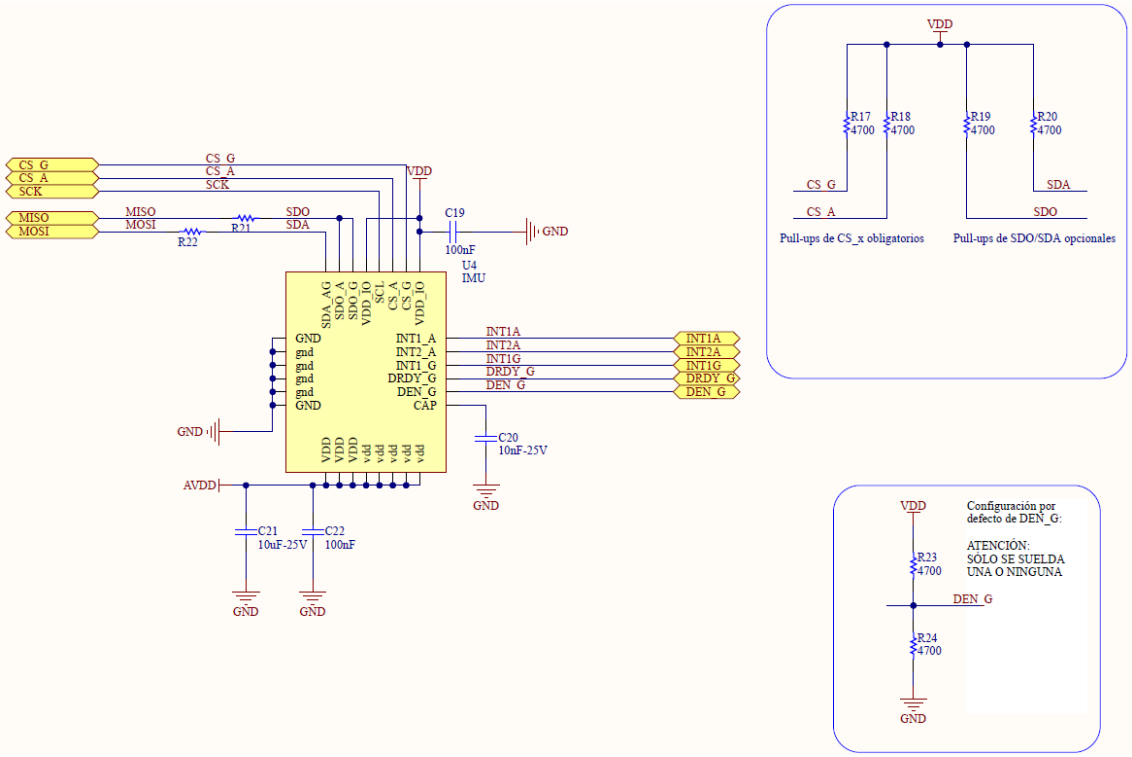


Figura 36: Esquemático de la IMU (LSM330D)

4.4.5 Conexión RS232

Para comunicar el microcontrolador con un dispositivo externo de comunicación se utiliza el puerto serie de comunicación RS232. Este puerto tiene la peculiaridad de trabajar con señales de ±12 V. La comunicación se realiza a través de dos hilos, uno de transmisión y otro de recepción. Para interconectar el bus de comunicación de la unidad con un puerto de un PC es necesario utilizar un conversor que reciba una señal de ±12 V y a su salida se obtenga una señal entendible por un puerto USB del PC. Para ello es necesario que tenga un adaptador que convierta señales del puerto USART serie al protocolo USB.

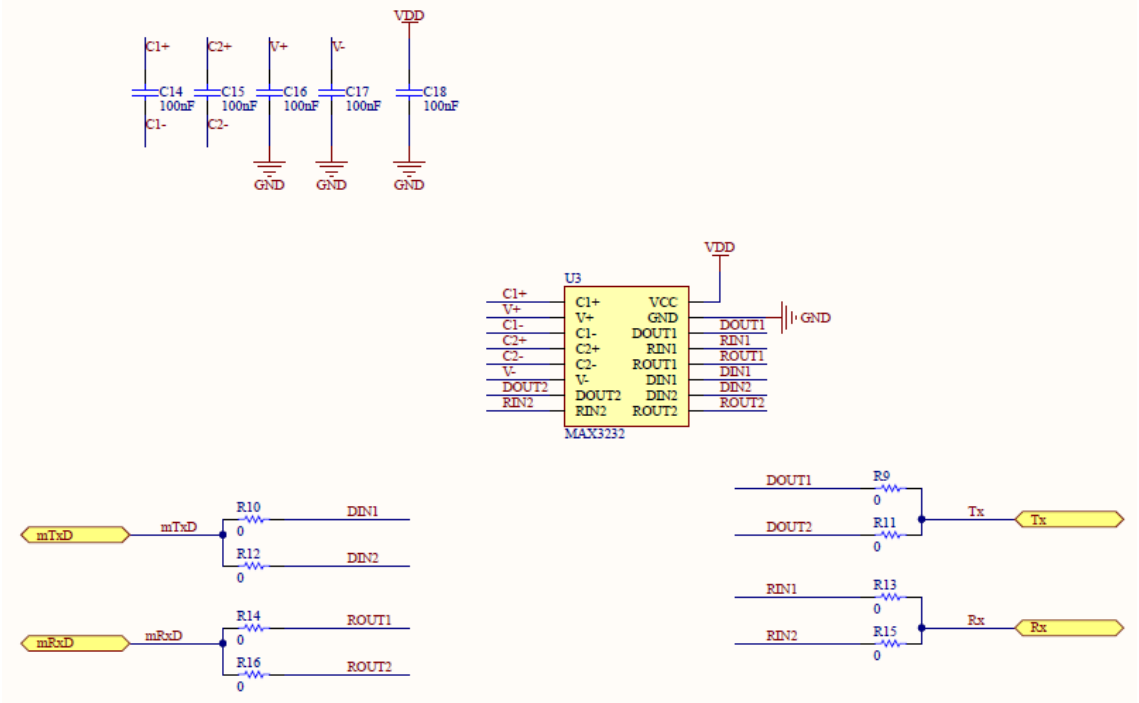


Figura 37: Conexión mediante el puerto RS232

4.4.6 Conexión General

Por último se tiene el esquemático de la conexión de todos los esquemáticos anteriores. Se puede apreciar cuáles son las líneas de conexión entre los diferentes bloques. El bloque de alimentación se comunica con el bloque de conectores para recibir la tensión de alimentación. A su vez el bloque de conectores se comunica con el adaptador RS232 y este al puerto UART del microcontrolador. El bloque de conectores se conecta a un bloque que tiene transistores NMOS para conectar externamente leds.

Por el otro lado de la imagen se pueden ver todas las conexiones entre el microcontrolador y la unidad inercial. Estas líneas de conexión son las de comunicación por el puerto SPI y las de control de la IMU. Entre las líneas de control se encuentran las de aviso de interrupciones cuando es el sensor quien las controla y la de habilitación en la cual el microcontrolador en determinados casos puede apagar la unidad inercial para ahorrar energía.

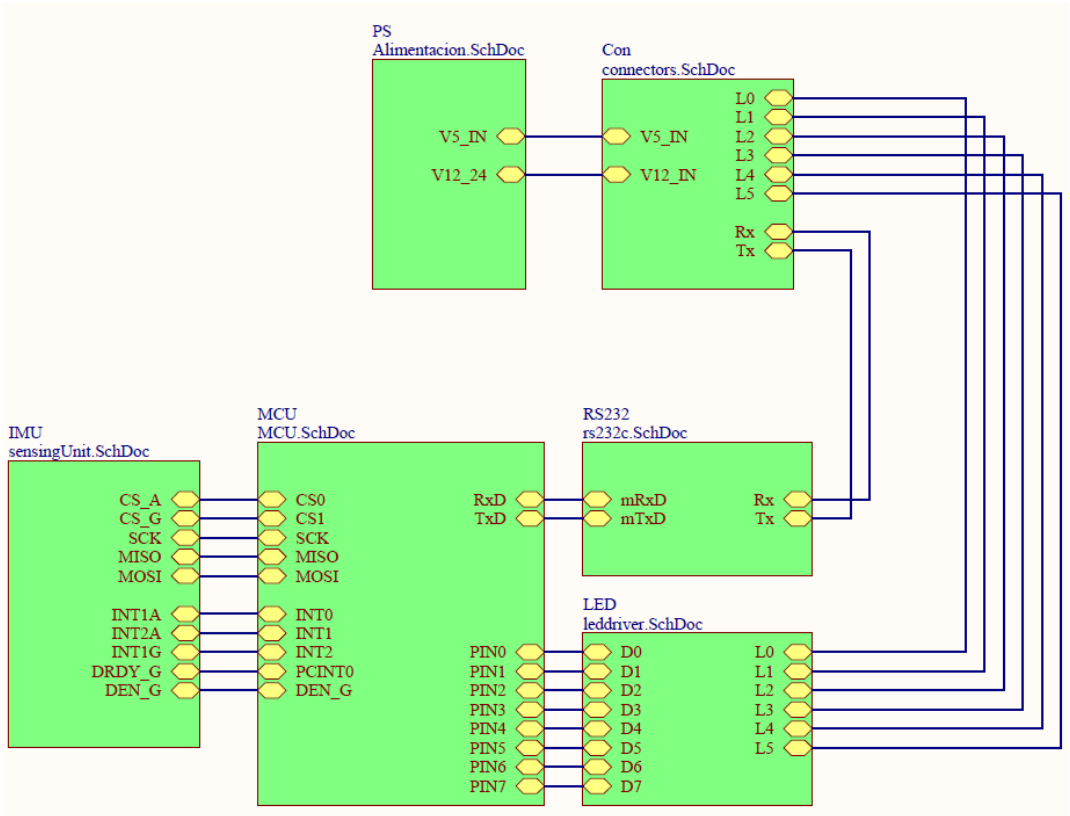


Figura 38: Conexión General de todos los esquemáticos

4.4.7 PCB

Una vez estén los esquemáticos realizados, se pasan a posicionar y conexionar en una PCB para mandar a fabricar la placa. Una vez la placa se ha fabricado y soldado sus componentes, el resultado final de la PCB es el que se muestra en la siguiente imagen.

Figura 39: PCB fabricada

4.5 Herramienta de desarrollo (AVRStudio 6)

El software que se utilizará para crear la programación y controlar el microcontrolador, va a ser el AVRStudio 6. Este programa ha sido elegido debido a que lo proporciona el mismo fabricante del microcontrolador gratuitamente para programar sus chips. Como es software propietario, ya contiene las bibliotecas propias de todos los micros comerciales de Atmel.

Para la instalación, solo hay que descargárselo desde la página web de Atmel. Es completamente gratuito y en la instalación te pide descargar un compilador de C/C++. También se descargan los drivers de los micros para que sean reconocidos por el programa al utilizar el JTAG.

A continuación se ofrece el enlace de descarga: http://www.atmel.com/microsite/atmel_studio6/

Una vez realizada la instalación, se verá la siguiente pantalla:

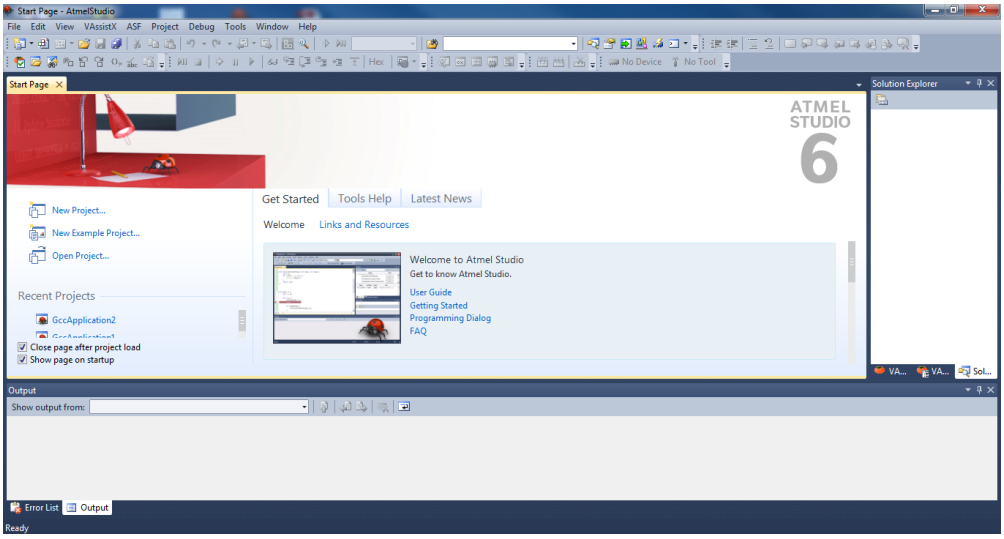


Figura 40: Pág. de inicio del AVRStudio

Como se observa en la imagen, da la opción de seguir una guía de usuario para saber cómo funciona y la manera de crear un proyecto. Además tiene una ayuda que informa sobre qué herramientas se pueden utilizar, cómo son los programadores y depuradores compatibles.

Para crear un nuevo proyecto sólo hay que hacer clic en New Project y aparecerá un menú para elegir el tipo de proyecto que se desea empezar. Para el proyecto que se va a realizar hay que diferenciar entre el GCC C ASF Board Projects y el GCC C Executable Project que es el que se utilizará debido a que no se necesita la arquitectura ASF.

La diferencia de usar la arquitectura ASF es que se tienen partes de módulos desarrollados por Atmel que hacen que se reduzca el tiempo empleado para la programación al ya venir hechos. Brinda un nivel de abstracción de la placa que permite simplificar el uso de los microcontroladores. ASF es una biblioteca de código libre y abierto muy utilizado para la evaluación, creación de prototipos, diseño y producción.

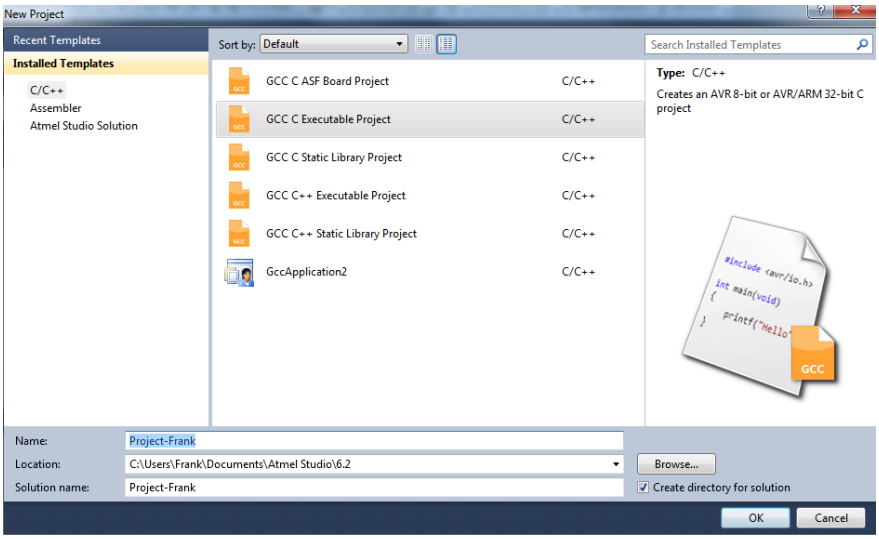


Figura 41: Ventana para elegir tipo de Proyecto

El siguiente menú que saldrá es para elegir el microcontrolador que se utiliza. Para elegirlo hay opciones de filtrado y cuando se encuentre el empleado, proporciona información sobre la alimentación a la que funciona y la familia de microcontroladores a la que pertenece. Además tiene un link para acceder directamente al datasheet y proporciona las herramientas de programación que soporta.

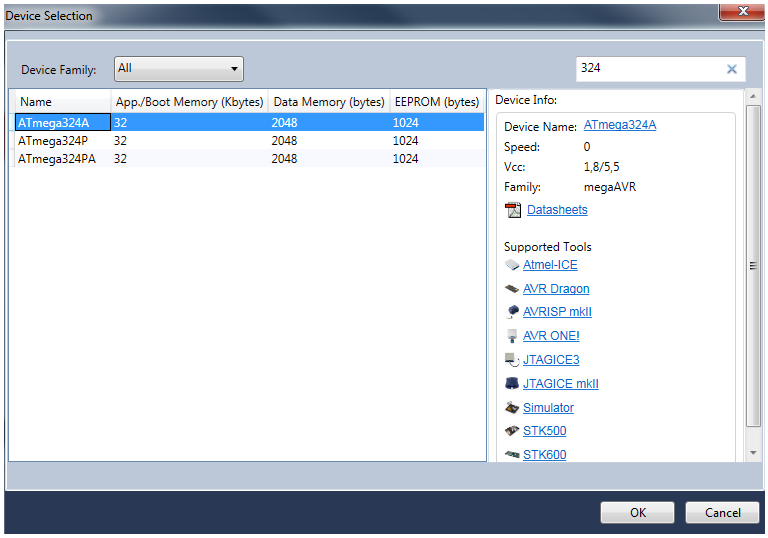


Figura 42: Ventana de elección del microcontrolador

El entorno de programación que se abrirá se puede observar en la siguiente figura, donde se tienen 3 bloques distintos. El bloque más grande ubicado en el centro se utiliza para escribir el código que se utilizará. A su derecha está el bloque en donde se encuentran los archivos que se han creado y los que se necesitan y que ya están creados por AVR como las bibliotecas propias de AVR, las de I/O, las de interrupciones, etc. Y en la parte inferior

está el bloque que informa de los errores, avisos, breakpoints cuando se ejecuta la compilación o depuración del código.

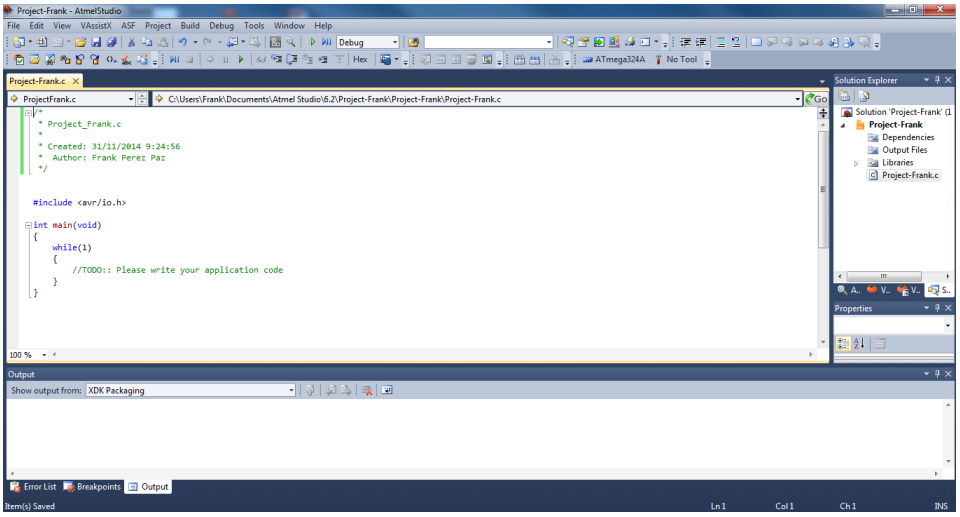


Figura 43: Entorno de Programación

4.6 Pruebas de pre-programación del microcontrolador

Antes de conectar el programador a la placa PCB del dispositivo sensor se comprobaron las conexiones de las partes integrantes de la placa. Para ello se utilizó un polímetro y se midió la continuidad y el aislamiento de las pistas y planos de alimentación.

A continuación se realizaron las conexiones de alimentación y comunicación. Esto es debido a que la placa PCB utiliza dos conectores RJ45 de 10 cables y hay que realizar unos cables específicos para su conexión. Para el cable de alimentación se utilizó un cable Ethernet de 8 hilos en los que solo se utilizaron los dos de masa y alimentación. Para el cable de comunicaciones, se utilizó otro cable Ethernet en donde se utilizaron 3 hilos, uno de masa, otro para la recepción de datos y el otro para la transmisión, este cable se soldó a un conector DB9 macho para poder conectarlo a un convertidor DB9 to USB. La conexión al conector DB9 se realizó como se muestra en la siguiente figura.

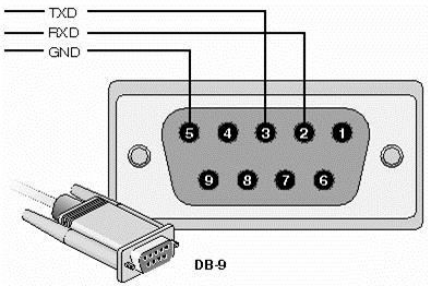


Figura 44: Conexión del conector DB9

Una vez realizadas las conexiones se pasó a conectar la placa a la una fuente de alimentación para comprobar con un polímetro la tensión que se le suministraba al microcontrolador. Con esto se comprobó la correcta alimentación del microcontrolador y se prosiguió a conectar el dispositivo programador para ver si había conexión y poder así empezar a realizar el software de control.

4.7 Depurador/programador JTAG

El programador que se utilizará para introducirle al microcontrolador el software realizado será el JTAGICE3. Este dispositivo sirve de depurador de código para el microcontrolador que se utiliza (Atmega 324A).

Figura 45: Programador JTAGICE3

Para programar el micro, se utiliza el programa AVRStudio conectado al JTAGICE3 instalando los driver necesarios para su funcionamiento. Este depurador se conecta al microcontrolador por medio del puerto JTAG. Para comprobar su conexión hay que ir a Tools → Device Programming y aparecerá una ventana donde seleccionar el dispositivo y configurar la velocidad de conexión con el micro.

En la siguiente imagen se observa la ventana de configuración que aparece. La configuración normal que se ha de utilizar es la siguiente:

Para configurar la velocidad de reloj del JTAG hay que ponerle normalmente una frecuencia 8 veces inferior a la frecuencia de reloj del microcontrolador. Como el microcontrolador funcionará a una frecuencia de 8 MHz, la frecuencia de reloj del JTAG se configura a 1 MHz.

Para ver si está conectado correctamente, se lee el voltaje del dispositivo y los tres bytes del código de firma del micro. Este código de firma se encuentra en el datasheet y sirve para identificar el dispositivo. En este caso, el código de firma lo encontramos en la siguiente tabla:

	Signature bytes address			JTAG	
	0x000	0x001	0x002	Part number	Manufacture ID
ATmega164A	0x1E	0x94	0x0F	940A	0x1F
ATmega164PA	0x1E	0x94	0x0A	940A	0x1F
ATmega324A	0x1E	0x95	0x15	9511	0x1F
ATmega324PA	0x1E	0x95	0x11	9511	0x1F
ATmega644A	0x1E	0x96	0x09	960A	0x1F
ATmega644PA	0x1E	0x96	0x0A	960A	0x1F
ATmega1284	0x1E	0x97	0x06	9705	0x1F
ATmega1284P	0x1E	0x97	0x05	9705	0x1F

Tabla 2: Dispositivos e identificador

En la siguiente figura se observa como en el recuadro Device Signature aparece el código 0x1E9515 el cual coincide con el que aparece en la tabla. Además se puede ver como el voltaje es el correcto con el que alimentamos el microcontrolador. Con estas comprobaciones ya se puede pasar a programar el dispositivo.

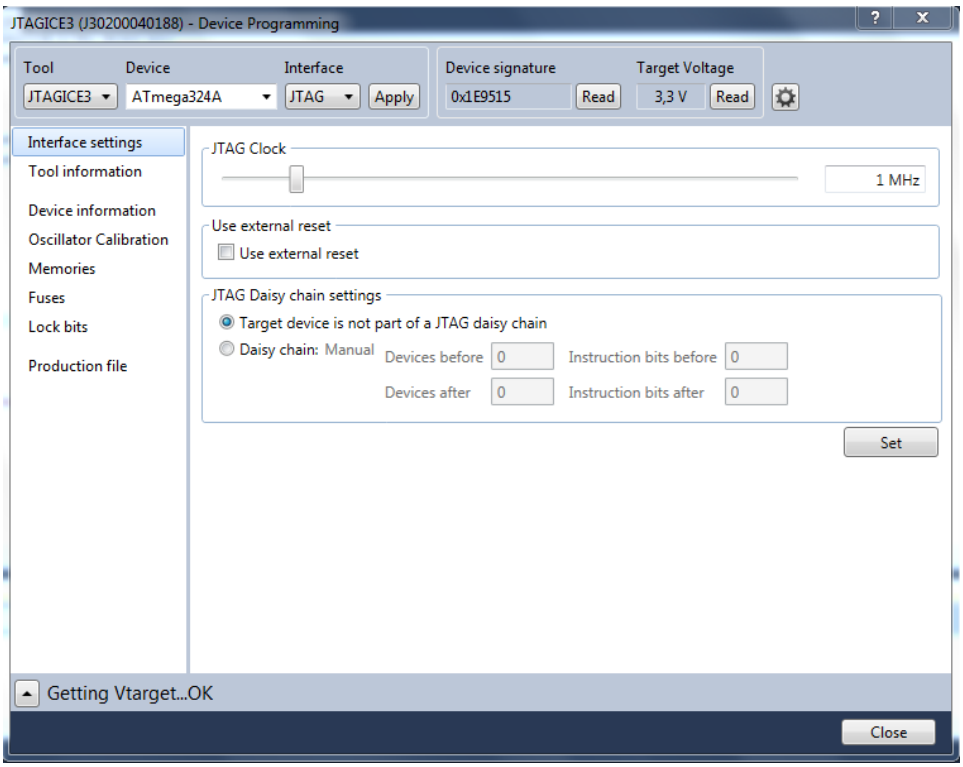


Figura 46: Configuración del JTAG

5.0 Implementación del Sensor de Movimiento

5.1 Introducción

En este apartado se explican los diferentes algoritmos que se han desarrollado para lograr realizar las medidas de aceleración de un vehículo. Los diferentes algoritmos se han ido mejorando, empezando por el algoritmo más simple. Se han ido realizando nuevos algoritmos a medida que se fueron realizando las pruebas y se comprobaba la funcionalidad.

Los tres problemas fundamentales que se intentaron solventar con el desarrollo de los sucesivos algoritmos son los que se citan a continuación:

- El primer problema se encuentra en eliminar el factor gravedad. Este parámetro causa un error debido a que su valor es mucho más elevado que cualquier aceleración normal de un vehículo. Si este factor se introduce en la señal que se desea medir (aunque sea un pequeño acoplo), causa que la medida sea mucho más fuerte que el movimiento realizado. Por ejemplo, la aceleración de un vehículo de 0 a 100 km/h en 10 segundos es de 2,77 m/s^2, valor que no es comparable con el de la gravedad que es de 9,81 m/s^2.
- El segundo problema es que el sistema no requiera de una alineación mecánica a la hora de su montaje, uno de los requisitos que se plantearon como objetivo para el Trabajo de Fin de Título. Por lo tanto, el sensor tiene que detectar por sí mismo la orientación en la que está colocado con respecto al movimiento del vehículo.
- Por último, el tercer problema que se intentó solucionar fue el de las fuerzas centrífugas de los giros. Dichas aceleraciones son muy fuertes y no están relacionadas con la eficiencia en la conducción, por lo que deben ser discriminadas.

5.2 Primer algoritmo

El primer algoritmo que se planteó consiste en eliminar el factor de la aceleración de la gravedad de la medida del acelerómetro y así obtener los datos de las aceleraciones del sensor sin verse influenciada por la constante gravitatoria. La manera más básica para este propósito es realizar el módulo de las tres medidas de los sensores.

$$|Modulo| = \sqrt{X^2 + Y^2 + Z^2}$$

Para calcular el módulo, se hace la raíz cuadrada de la suma de los ejes elevados al cuadrado. El resultado es el módulo del vector que forman ambos, el cual tiene la aceleración a la que es sometido el acelerómetro y la aceleración de la gravedad.

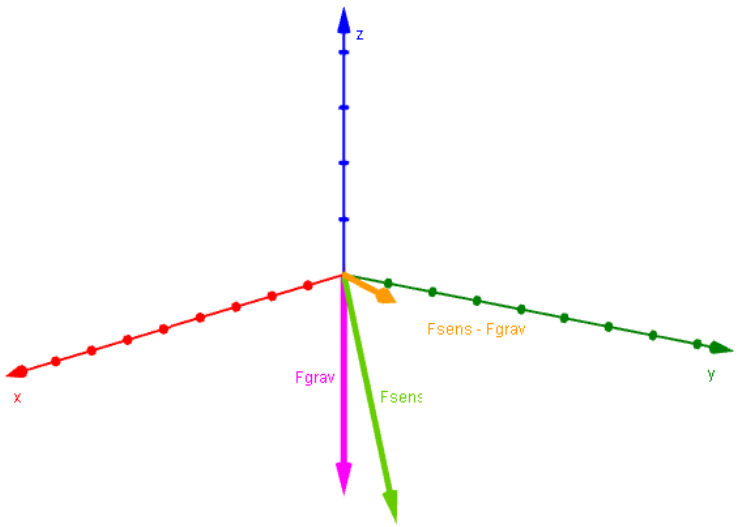


Figura 47: Vector gravedad (Rosa), vector medido del sensor (Verde) y vector diferencia (dato real, Amarillo)

El proceso para obtener la aceleración del acelerómetro y eliminar la producida por la gravedad consiste pasarla por un filtro paso-bajo (10 Hz) que contiene la gravedad y los datos útiles (flecha verde en la figura 47), a continuación pasarla por un filtro paso-alto que elimina la gravedad (ya que es una señal constante, similar a un nivel de continua) y obtiene los datos (flecha amarilla de la figura 47). Esto quiere decir que al filtro que contiene los datos y la gravedad (filtro paso-bajo), se le elimina la gravedad en el siguiente filtrado (paso-alto). Esta combinación de filtros funciona como un filtro de banda de paso el cual solo contempla los datos de aceleración.

En el siguiente diagrama de flujo se explican los pasos que tiene el algoritmo y como se realiza el proceso. Primero se empiezan a tomar valores del sensor. Cada valor es

introducido en un filtro de gravedad + datos (paso-bajo de 10 Hz), posteriormente estos valores van a un filtro paso-alto que elimina los valores continuos, es decir, la gravedad. A la salida de estos filtros se le realiza el módulo de sus ejes para obtener el valor de la aceleración sin gravedad.

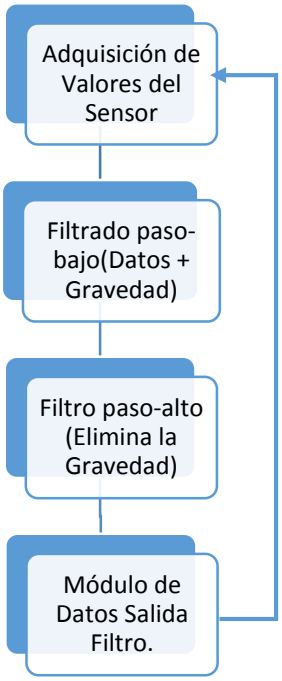


Figura 48: Flujograma del primer Algoritmo

Como el sensor no tiene filtrado de las señales, el cálculo de restar al módulo la aceleración hay que pasarlo por una etapa de filtrado. Este filtrado se hace mediante filtrado digital en el cual, primero ha de calcularse los coeficientes adecuados. Tras varias pruebas con distintos valores de filtros, se dedujo que el filtrado más adecuado era un filtro paso-banda. Este filtro tiene un ancho de banda de 9.8 Hz puesto que la banda de paso inferior empieza en 0.2 Hz y la banda de paso superior está limitada a los 10 Hz.

Para implementar este filtro, lo que se realizó fue la creación de dos filtros, un filtro paso-bajo y un filtro paso-alto. Mezclando estos filtros se obtiene el filtro paso-banda deseado. Se utilizan dos filtros separados en vez de utilizar directamente el paso-banda, porque queremos fácilmente cambiar las frecuencias de corte, están separados suficientemente y las características del paso-alto y paso-bajo no tienen por qué ser iguales.

El filtro paso-bajo se diseñó para que fuera de 5º orden con una frecuencia de corte de 10 Hz. Como la Tasa Binaria del acelerómetro estaba programado para 100 Hz, se diseñó un filtro Butterworth en donde se obtuvieron los siguientes coeficientes:

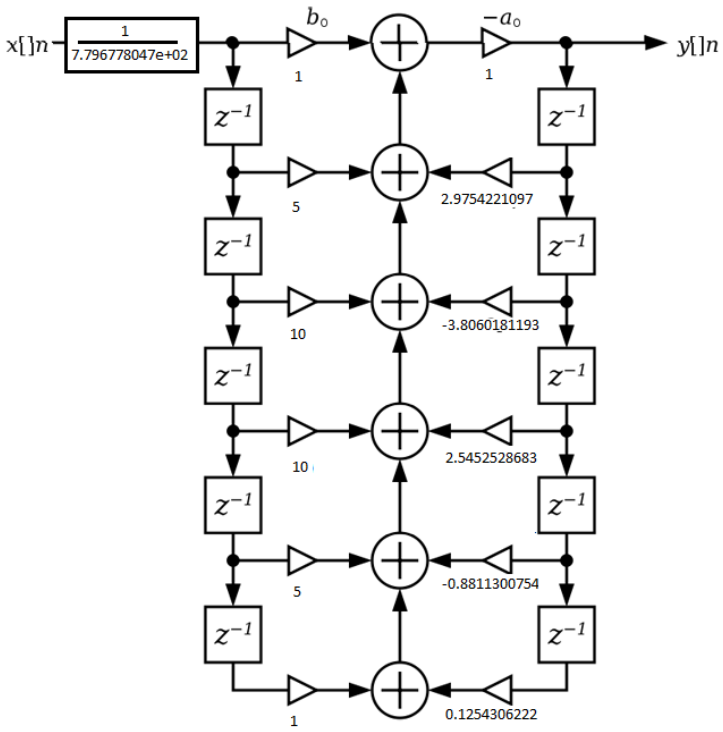


Figura 49: Coeficientes filtro paso-bajo de 10 Hz

Estos coeficientes son introducidos en el siguiente algoritmo de filtrado para obtener el valor filtrado a la frecuencia especificada. Para que el filtro empiece a funcionar adecuadamente, los valores de xv[] tienen que contener los primeros valores del sensor. Además los primeros valores de yv[], serán incorrectos porque hasta que no haya realizado unas cuantas mediciones, el filtro tendrá una oscilación producida por estos valores iniciales.

```
xv[0] = xv[1];
xv[1] = xv[2];
xv[2] = xv[3];
xv[3] = xv[4];
xv[4] = xv[5];
xv[5] = Nuevo Valor del Sensor / Ganancia;
yv[0] = yv[1];
yv[1] = yv[2];
yv[2] = yv[3];
yv[3] = yv[4];
yv[4] = yv[5];
yv[5] = b0(xv[0] + xv[5]) + b1 * (xv[1] + xv[4]) + b2 * (xv[2] + xv[3])
        + (a5 * yv[0]) + (a4* yv[1])+ (a3 * yv[2])
        + (a2* yv[3])  + (a1 * yv[4]);
Valor de Salida Filtrado = yv[5];
```

Obteniendo la siguiente grafica de cómo responde el filtro. Se puede observar cómo a 8 Hz la señal empieza a atenuarse hasta llegar a atenuarse hasta que en 10 Hz se obtiene -3 dB de atenuación del filtro.

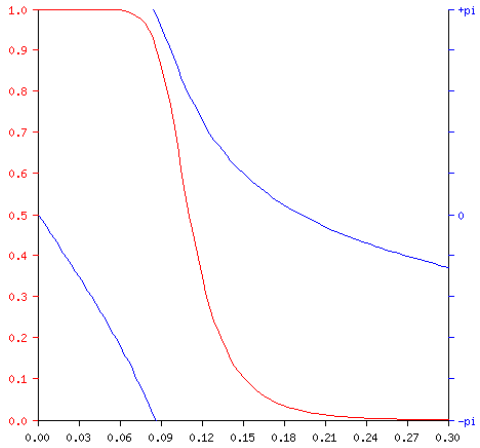


Figura 50: Respuesta del filtro paso-bajo (Magnitud (rojo) y fase (azul))

Para el filtro paso-alto, se realizó para una frecuencia de corte de 0,2 Hz y también de 5º orden. Utilizando el mismo algoritmo pero cambiando los coeficientes por los siguientes:

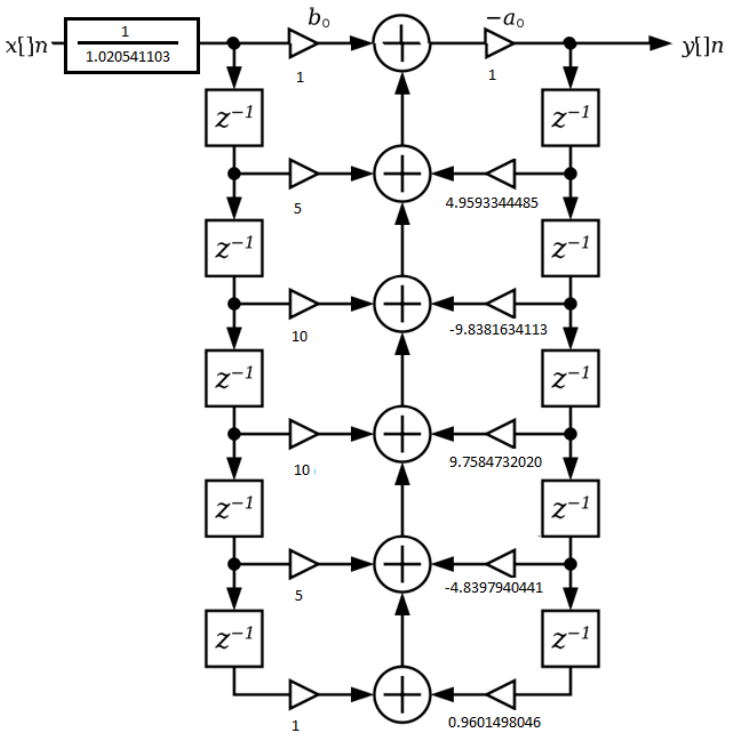


Figura 51: Coeficientes filtro paso-alto de 0,2 Hz

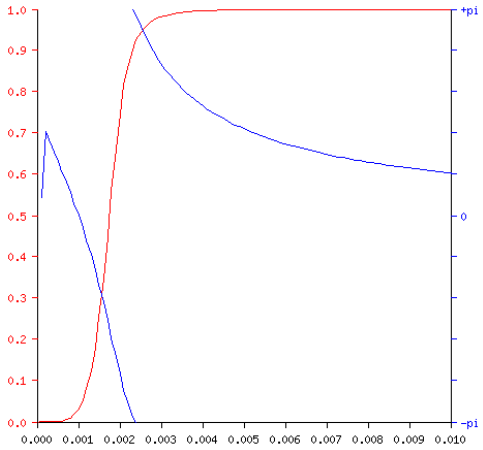


Figura 52: Respuesta del filtro paso-alto (Magnitud (rojo) y fase (azul))

Para saber si el ángulo de inclinación de un vehículo cuando está en llano o cuando tiene que iniciar una subida en las pendientes no afecta a la medida tomada por el sensor, se decidió realizar las siguientes comprobaciones. Normalmente las carreteras por la que circulan los vehículos tienen una pendiente de aproximadamente el 10% de inclinación.

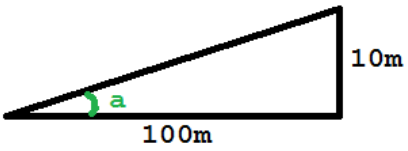


Figura 53: Significado trigonométrico del significado de una pendiente del 10%

Con esta inclinación, el ángulo que forma la pendiente se calcula haciendo el siguiente cálculo matemático:

$$\alpha = \tan^{-1}\frac{Cateto\ Opuesto}{Cateto\ Contiguo} = \tan^{-1}\frac{10}{100} = 5{,}7°\ de\ inclinación$$

Ahora se necesita saber si esta diferencia de inclinación es relevante si tomamos la medida real llamada en la siguiente figura como *m* y la medida que será tomada para los cálculos *x*.

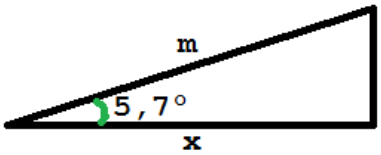


Figura 54: Diferencia entre m y x con 5,7 grados de inclinación

$$x = m * \cos(\alpha) = m * \cos(5{,}7) = m * 0.995 \approx m$$

Ahora como se sabe que el error que se cometerá en la medida por hacer esta aproximación es prácticamente nulo (0.005%) se puede realizar un test de funcionamiento para ver cómo funciona el algoritmo.

Antes de probarlo, para tener una idea de las magnitudes de los valores que se obtendrán, se realizó un pequeño cálculo aproximado del valor de la aceleración que puede producir un vehículo normal si acelera de 0 km/h a 100 km/h (0 m/s a 27.778 m/s) en 10 segundos. El valor de la aceleración que tendrá será de:

$$A = V \cdot t = 27.778\ \frac{m}{s} \cdot \frac{1}{10s} = 2.7778 \frac{m}{s^2}$$

Esta aceleración hay que convertirla a la magnitud de fuerza G, que es la magnitud de medida del acelerómetro. Por lo que utilizando la igualdad de 1G = 9.80665m/s^2, se calcula que la fuerza G ejercida para acelerar a esta velocidad es de:

$$\frac{2.7778 \frac{m}{s^2} \cdot 1g}{9.80665 \frac{m}{s^2}} = 0.28g$$

Con estas magnitudes con las que se trabaja, se ha de esperar poca variación de los datos. Si los datos varían por encima de estos valores, es que está interfiriendo o se está midiendo la fuerza gravitatoria.

5.3 Segundo algoritmo

Tras analizar los resultados del anterior algoritmo, para detectar el motivo del fallo, se averiguo que el motivo por el que se producía el error en la medida era causado por las diferencias entre las magnitudes de la gravedad con respecto a la aceleración. Y aunque los cálculos presentados en donde se mostraba que el coseno (α) al ser un ángulo pequeño no afectaba, si se considera que la gravedad es de 9,8 g y las aceleraciones de los vehículos son de aproximadamente 0,3g, ese pequeño cambio del vector de gravedad se introduce en las medidas de aceleración.

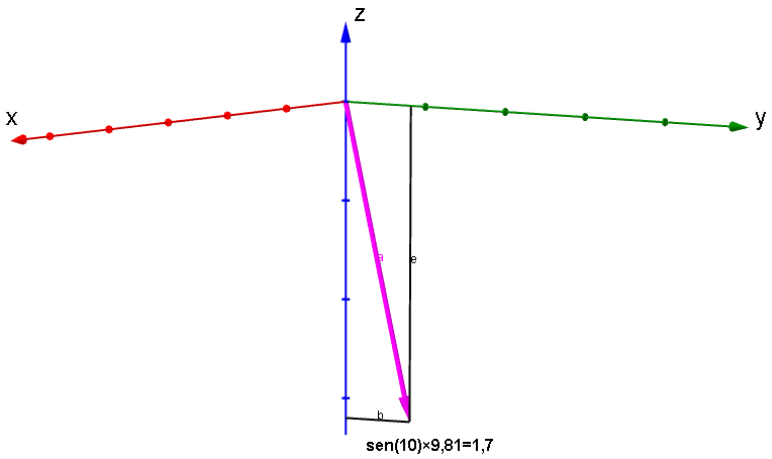


Figura 55: Desplazamiento del vector debido a una pendiente de 10° de inclinación.

Como muestra la figura anterior, la gravedad puede llegar a añadirse a los vectores de aceleración hasta 1,7g en las pendientes más pronunciadas. Por lo que cuando un vehículo se encuentra en una pendiente, lo que está midiendo en realizad es el vector de gravedad, que es significativamente mayor (> 6 veces) que la aceleración producida. Una aceleración de esta magnitud no la puede realizar un vehículo normal y produce un error grave en la medida.

El siguiente algoritmo diseñado consiste en convertir las medidas de los 3 ejes (tri-dimensional) en sólo 2 ejes (bi-dimensional) obteniéndose solo los valores de los ejes **X** e **Y**. Con esto, se elimina el eje **Z** el cual se hará coincidir con la gravedad rotando los ejes de medidas. Como la unidad de medida inercial no se posicionará calibrada y por lo tanto no coinciden paralelamente ninguno de los ejes de medida con el vector de gravedad, se tiene que rotar hasta hacerlo coincidir. Con esta rotación se elimina el problema del algoritmo anterior, puesto que al rotar, cuando se tome una pendiente, los ejes rotaran y el vector gravedad no afectara a la medida.

Además como en el algoritmo anterior se detectó que el filtro no era adecuado (producía mucho retardo por ser de 5º orden), se realizó un filtro de menor orden. Este nuevo filtro se calculó para que fuera de 2º orden. Para este algoritmo lo que se intenta buscar es filtrar el valor de la gravedad, que al ser un valor constante con un filtrado paso-bajo, se puede obtener. Al contrario que en el anterior algoritmo, en este se utilizan dos filtros paso-bajo.

Para diseñar el filtro de gravedad, se utilizó como valor para la frecuencia de corte 2 Hz, de esta manera, se obtienen las señales de los ejes a los cuales les influye la gravedad. El filtro calculado para esta tarea tiene los siguientes coeficientes:

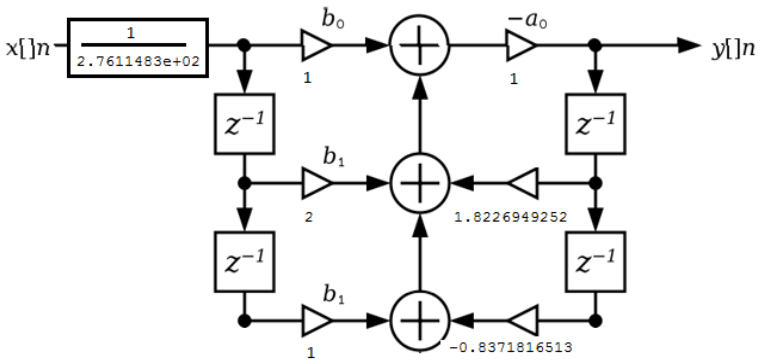


Figura 56: Coeficientes filtro paso-bajo de 2 Hz

En la siguiente gráfica se representa la respuesta de este filtro de gravedad.

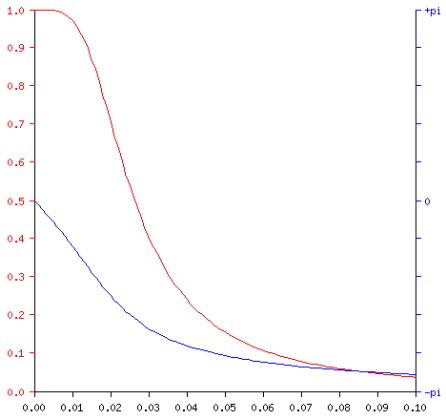


Figura 57: Respuesta del filtro de Gravedad (Magnitud (rojo) y fase (azul))

Cada eje será filtrado por separado en este filtro obteniéndose los valores de la gravedad en cada eje **X**, **Y**, **Z**. A continuación se pasa a calcular el siguiente filtro, el cual se calcula como otro filtro paso-bajo pero con más BW para que no solo se obtengan los valores de la gravedad, sino que deje pasar las señales que cambian debidas a las aceleraciones. Este filtro se calculó para una frecuencia de corte de 10 Hz. Los valores de este filtro paso-bajo son los siguientes:

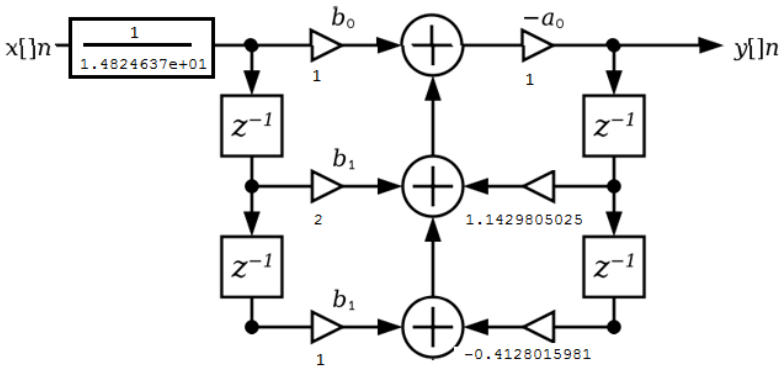


Figura 58: Filtro paso-bajo de 10 Hz

En la siguiente gráfica se observa la respuesta del filtro paso-bajo, representado por la magnitud y fase.

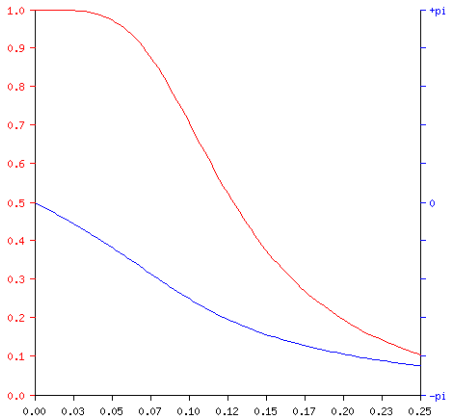


Figura 59: Respuesta del filtro paso-bajo (Magnitud (rojo) y fase (azul))

A continuación, como se tienen las respuestas por separado de los diferentes ejes, se resta el valor de la señal del filtro paso-bajo al valor de la señal del filtro de gravedad. Con esto se obtienen los valores de las aceleraciones de los ejes sin gravedad.

$$Valor\ X\ del\ Sensor = Valor\ Filtro\ Paso\ Bajo\ SensorX - Filtro\ Paso\ Bajo\ GravedadX$$

$$Valor\ Y\ del\ Sensor = Valor\ Filtro\ Paso\ Bajo\ SensorY - Filtro\ Paso\ Bajo\ GravedadY$$

$$Valor\ Z\ del\ Sensor = Valor\ Filtro\ Paso\ Bajo\ SensorZ - Filtro\ Paso\ Bajo\ GravedadZ$$

A continuación, se realiza una rotación de los ejes, para obtener la ventaja de trabajar con sólo dos ejes. Para entender los pasos que contiene el algoritmo, se detalla a continuación un flujograma en donde se explica más claramente el proceso. Como los valores X e Y después de la rotación son los valores de la aceleración, se realiza el módulo de ambos y con esto se obtiene la aceleración en dos dimensiones.

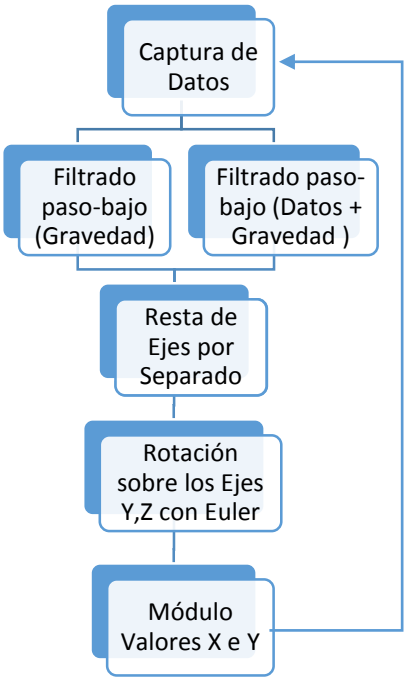


Figura 60: Flujograma del Segundo Algoritmo

Para realizar esta rotación de los ejes se utilizan los valores obtenidos por el filtro de gravedad para ver los ángulos que es necesario rotar los valores de los ejes. A continuación se detalla una imagen en donde se explica la rotación que ha de producirse para hacer coincidir los ejes.

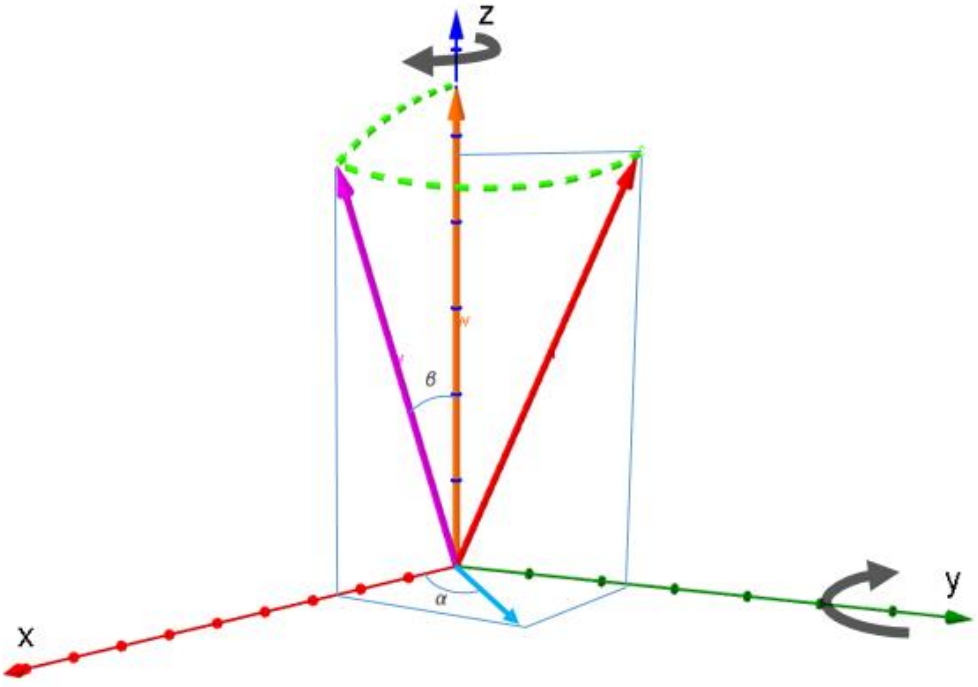


Figura 61: Modelo de rotación vectorial

La primera rotación que se realiza es cogiendo como referencia el eje **Z** y girando sobre él. Una vez se hayan rotado los ejes **X** e **Y**, se realiza otro giro, esta vez cogiendo como referencia el eje **Y**. Para calcular el ángulo de rotación, se utiliza la siguiente formula:

$$\alpha = \tan^{-1}\frac{Filtro\ GravY}{Filtro\ GravX}$$

$$\beta = \tan^{-1}\frac{\sqrt{Filtro\ GravX^2 + Filtro\ GravY^2}}{Filtro\ GravZ}$$

Como se explicó anteriormente en las rotaciones de Euler, si lo que se desea es rotar en función del eje **Z**, se tiene que utilizar la siguiente matriz:

$$R_I^{v1}(\alpha) = \begin{pmatrix} \cos(\alpha) & \text{sen}(\alpha) & 0 \\ -\text{sen}(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Al realizar esta rotación, se obtiene el vector de color rosado que se observa en la figura del ***Modelo de rotación vectorial***. El siguiente paso es hacer la rotación sobre el eje **Y**, utilizando la matriz de rotación siguiente:

$$R_{v1}^{B}(\beta) = \begin{pmatrix} \cos(\beta) & 0 & -\text{sen}(\beta) \\ 0 & 1 & 0 \\ \text{sen}(\beta) & 0 & \cos(\beta) \end{pmatrix}$$

Para realizar esta rotación más eficazmente, y como se trata de ejes móviles, lo que se realiza es una multiplicación entre la matriz de rotación sobre el eje **Z** y la matriz de rotación del eje **Y**.

$$R_I^B(\alpha) = \begin{pmatrix} \cos(\alpha) & \sin(\alpha) & 0 \\ -\sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} \cos(\beta) & 0 & -\text{sen}(\beta) \\ 0 & 1 & 0 \\ \text{sen}(\beta) & 0 & \cos(\beta) \end{pmatrix}$$

$$= \begin{pmatrix} \cos(\alpha)\cdot\cos(\beta) & sen(\alpha) & -\cos(\alpha)\,sen(\beta) \\ -sen(\alpha)\cos(\beta) & \cos(\alpha) & sen(\alpha)sen(\beta) \\ -sen(\beta) & 0 & \cos(\beta) \end{pmatrix}$$

Al realizar esta rotación se consigue que los vectores roten en función del vector de gravedad, con lo que se podría medir de manera correcta las medidas tomas por los vectores **X** e **Y**, los cuales contienen la aceleración en un plano de dos dimensiones. Y según los cálculos de inclinación en pendientes, esa desviación del vector, no debería influir en las medidas.

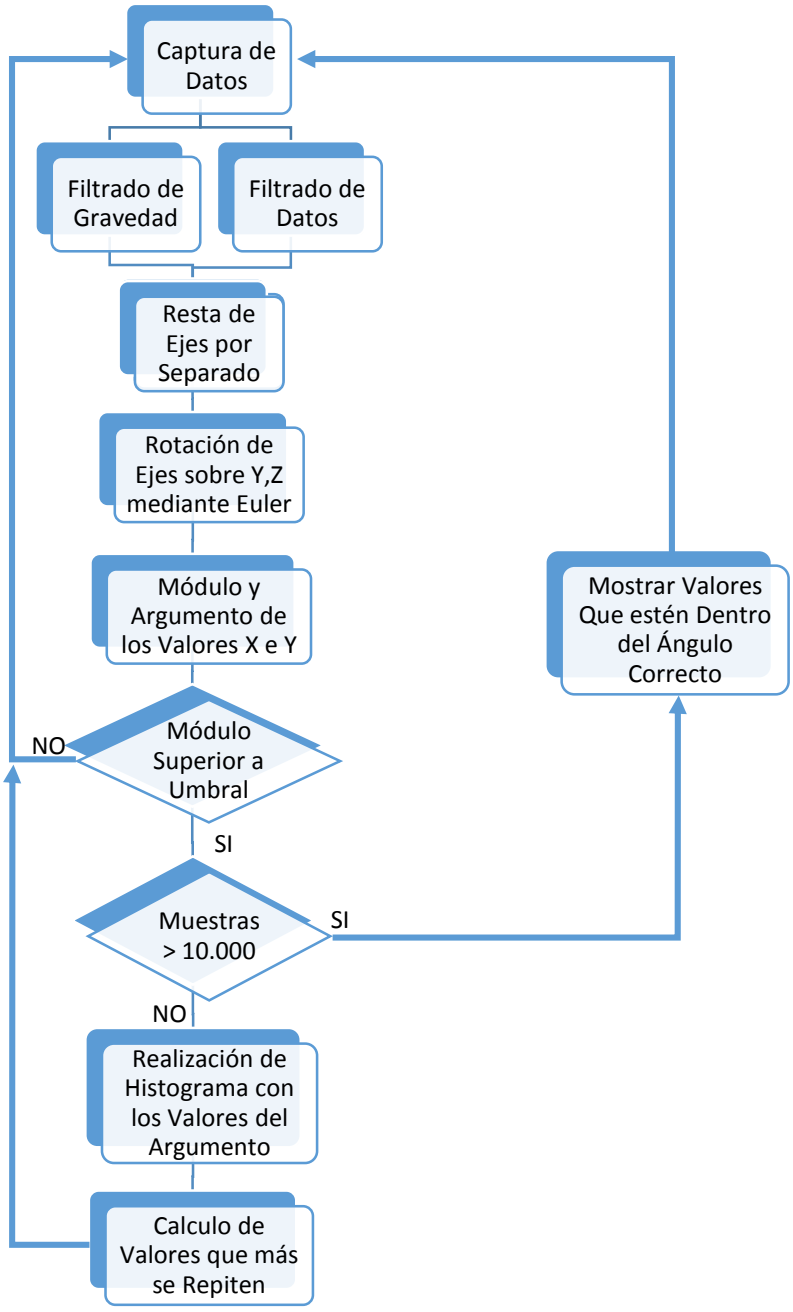
5.4 Tercer algoritmo

En el anterior algoritmo se expuso como se hacía para rotar los ejes de medida del sensor y localizar la gravedad. Este algoritmo funciona correctamente en cuanto a la eliminación de la gravedad y no causa problemas en las medidas cuando hay cambios de nivel en una pendiente. Por lo que se puede utilizar las rotaciones que se realizan para continuar eliminado el factor de la gravedad.

El inconveniente que se produce es que para este proyecto, lo que se busca es medir las aceleraciones y la eficiencia en la conducción durante el trayecto. Y el problema en el algoritmo anterior es que mide todas las aceleraciones en el plano de dos dimensiones y con esto, obtenemos aceleraciones que no son deseadas. Con esto quiere decir que muestra tanto las aceleraciones frontales como las aceleraciones laterales.

Este fallo provoca que cuando se está tomando una curva, el sensor está proporcionando una aceleración, la cual es una fuerza centrífuga, no una aceleración real del vehículo. Incluso cuando el vehículo se encuentra desplazándose pero sin aceleración y se mueve el volante hacia los lados, el sensor produce unas medidas mucho mayores que las aceleraciones frontales cuando el vehículo es acelerado a todo su potencial. Debido a que estas fuerzas laterales son mucho más grandes que una aceleración lineal en un coche. Son más parecidas a la fuerza de desaceleración en el momento de frenado.

Para desarrollar un método que corrigiera los fallos encontrados en los algoritmos anteriores y no mida las aceleraciones laterales, se planteó utilizar el ángulo que forman los ejes X e Y para ver en que coordenadas se produce el movimiento de aceleración frontal. Con este ángulo se podrá observar los movimientos de rotación y diferenciar entre las aceleraciones frontales y los giros. En el siguiente flujograma, se intentará explicar los pasos que se realizan.



Con el anterior algoritmo ya se podía localizar con precisión el eje de gravedad haciendo rotar los ejes de coordenadas cada vez que se capturan los datos del sensor, es decir, de manera continuada. Con esto se conseguía mantener posicionado el eje Z paralelo a la gravedad evitando que se sume a las medidas de aceleración. Al realizar este método, se controla que siempre se muestren las medidas reales de aceleración en los ejes **XY** y cuando se produce un cambio de pendiente, el vector que apunta a la gravedad es corregido para evitar que el ángulo de la pendiente introduzca error.

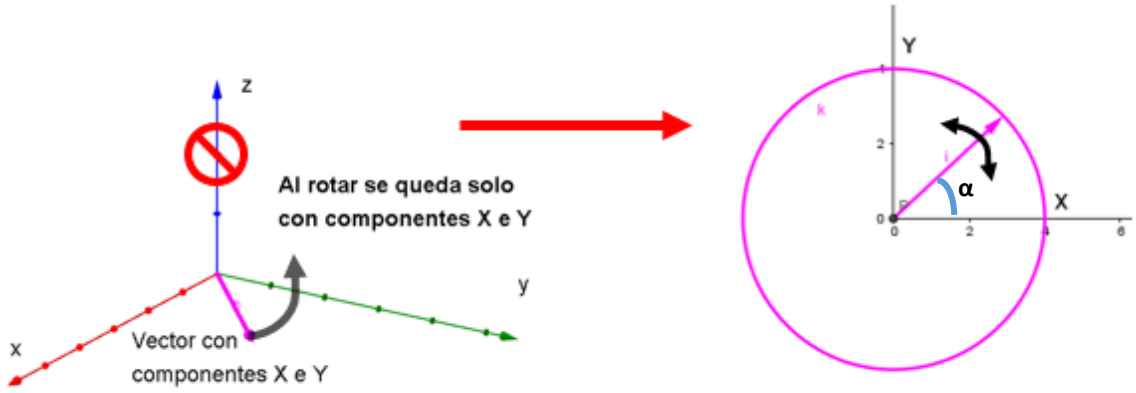


Figura 62: Rotación del vector para obtener un plano en dos dimensiones.

En la anterior imagen se muestra como a partir de la medida del vector de aceleración, el cual está compuesto por X, Y, Z, se pasa a solo tener los ejes **X** e **Y**. Esto se consigue a partir de las rotaciones de Euler, que se mostraron en el algoritmo anterior.

Al tener solo dos ejes se obtiene un vector en un plano, el cual rota dependiendo de la dirección de aceleración. La dirección de esta rotación será la que se utilizará para ver la dirección del movimiento. Este ángulo se obtendrá a partir de siguiente cálculo:

$$\alpha = \tan^{-1}\frac{RotaciónX}{RotaciónY}$$

Para analizar la información del ángulo, se pensó en utilizar una campana de gauss para analizar los valores que más se repiten y los que menos. Con esto se logra averiguar el sentido en donde se acelera o se frena el vehículo debido a que van hacer los datos que más se repitan durante la conducción. Al utilizar este método, se obtiene la dirección hacia donde se dirige el vehículo, es decir, se localiza la parte delantera o trasera de este.

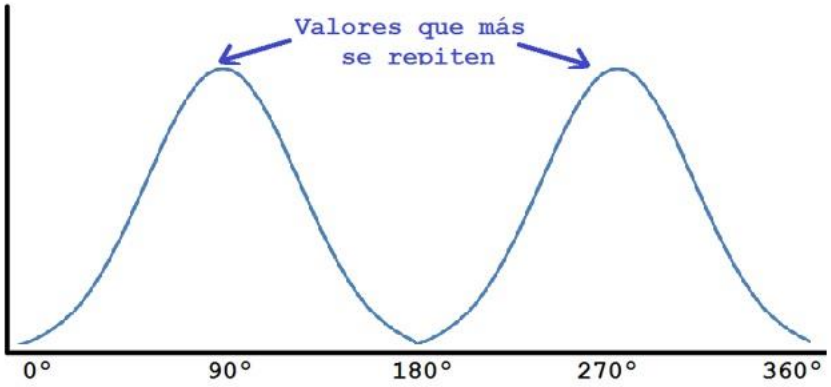


Figura 63: Campana de gauss

Las medidas del módulo del acelerómetro se irán tomando y sumando dependiendo del ángulo que formen. Si por cada ángulo distinto se toma una muestra, habría que tener 360 índices de referencia. Sin embargo esta medida es exagerada debido a que habría que estar mucho más tiempo para diferenciar las muestras tomadas en cada grado. Por ello se optó por dividir los 360º / 4 = 90 para que las medidas que se tomen dentro de esos cuatro grados se acumulen y se pueda diferenciar más rápidamente los valores que más se repiten. Es decir, se tendrá un array con 90 valores que irá acumulando las muestras de cada ángulo.

Cuando se hayan tomado los suficientes valores como para saber que los valores acumulados sean aceptables (se diferencie correctamente los datos de aceleración frontal con desplazamientos laterales), se considera que ya no hace falta seguir tomando muestras y que el algoritmo esta calibrado. Se busca dentro del array el valor acumulado de mayor valor y a partir de ese momento, solo se hace caso a las aceleraciones que tengan esa dirección.

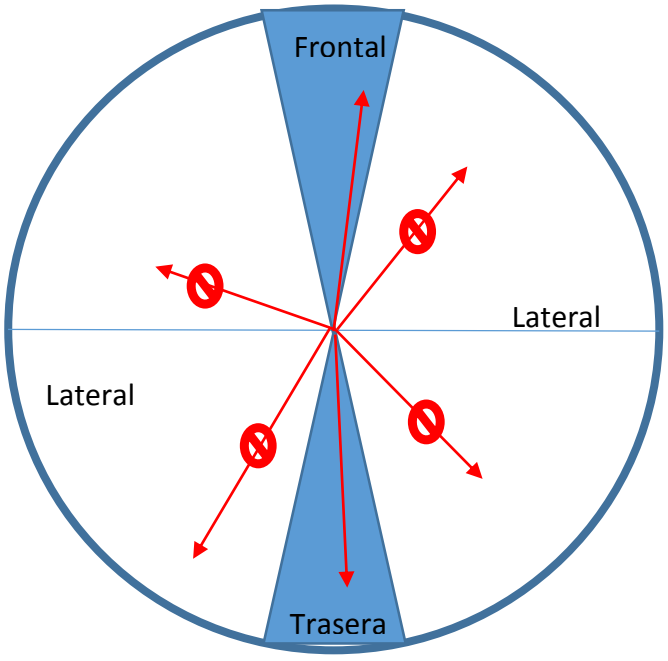


Figura 64: Zona de Aceleración (Azul), fuera de esta zona no adquiere los valores

En la figura anterior se aprecia como los valores que estén fuera de la zona calculada por el algoritmo, serán rechazados, mientras que los que tengan el ángulo correcto, serán medidos y mostrados.

5.5 Cuarto algoritmo

Como los algoritmos anteriores no han sido lo suficientemente precisos se ha decidido continuar con el desarrollo de estos. Para ello se ha considerado utilizar el sensor giroscópico para saber el movimiento giratorio del vehículo. Como el tercer algoritmo funciona, con las limitaciones explicadas en las pruebas, se utilizaran varias funciones de este para avanzar con el cuarto algoritmo.

Para empezar a desarrollar el cuarto algoritmo se planteó no utilizar los filtros que se habían estado utilizando anteriormente. De esta manera se corregiría el retraso que se tiene en la respuesta obtenida del sensor, a las aceleraciones. La manera que se planteó, para no utilizar el sistema de filtros, fue la siguiente:

- Crear una función a la cuál le afecte mínimamente las últimas variaciones de entrada, teniendo en cuenta las anteriores (como si fuera un filtro).
- Hacer que esa función dependa del estado del giróscopo un 99% y del acelerómetro solo un 1%.

En la siguiente figura se observa cómo va a funcionar el modelo planteado. Por la entrada va a recibir los datos del giroscopio, los cuales van a tener el mayor peso y por otro lado se le suma la señal del acelerómetro. A la salida del sistema se obtendrá una señal que se verá poco afectada por las variaciones rápidas del acelerómetro y tendrá más en cuenta al sensor de giro.

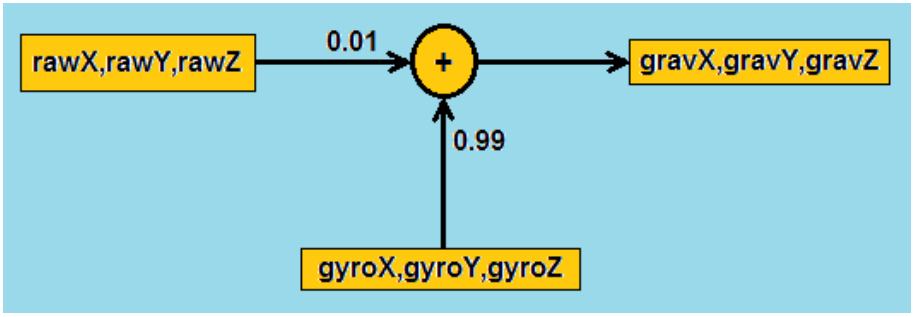


Figura 65: Modelo de filtrado utilizando el giroscopio

Esto se consigue utilizando la siguiente función para cada uno de los ejes:

$$ValorGravEjeX = EjeXAcelerómetroGirado(EjeGiroX°) \cdot 0{,}99 + EjeXAcelerómetro \cdot 0{,}01$$

Para que las señales que le entran al sistema anterior sean adecuadas, primero se han de calibrar ambos sensores y se utilizará una función que cambia los ejes de medidas.

Esta función corrige el que cuando la placa PCB este colocada horizontalmente, el eje Z mostrará la gravedad, sin embargo, cuando el dispositivo es colocado verticalmente, el eje X o el eje Y, serán los que mostrarán la gravedad. Como es necesario saber los ejes que se utilizarán para tomar las medidas y cuál es el eje más afectado por la gravedad, se decidió hacer una función que modificara los ejes y siempre le correspondiera al eje Z la indicación de la gravedad.

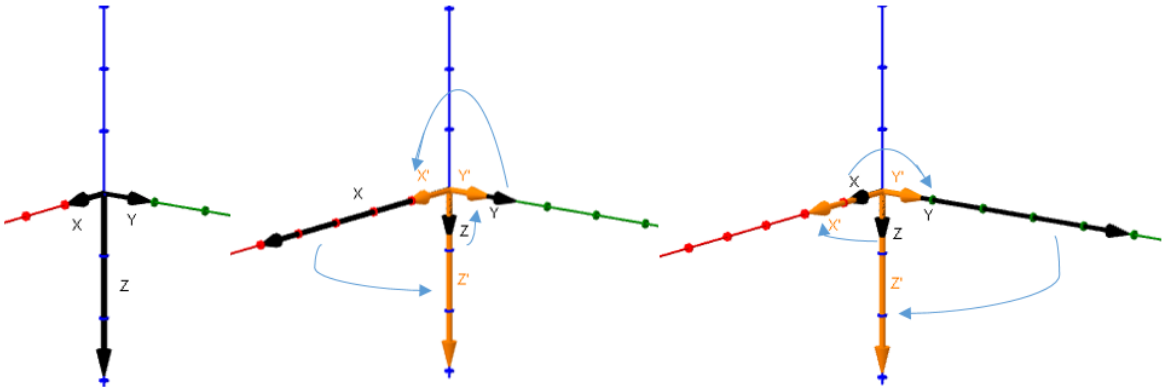


Figura 66: Cambio de Ejes dependiendo de la posición (función AxisSort)

Al no saber cómo estará colocada la PCB, hay que tener en cuenta el paso mostrado en la figura anterior para poder fijar las variables. Este cambio se realiza de la siguiente manera:

1. Se realiza una media con las medidas de los tres ejes del acelerómetro.
2. Se encuentra el eje que tenga el dato más grande (más afectado por la gravedad).
3. Se cambian los datos de los ejes entre si dependiendo de cuál sea el más acusado por la gravedad.

En la figura 62 se observa como si el eje Z coincide con la medida mayor, se deja sin cambiar los ejes. Si la medida más grande corresponde al eje X, el valor del eje X se le pasa al eje Z, el del eje Z se cambia al eje Y y el Y es cambiado al eje X. De la misma manera ocurre cuando el valor mayor se encuentra en el eje Y. De esta forma se puede trabajar con los valores X, Y, Z sabiendo que cuando se modifican o se leen, se sabe que Z tiene que contener la gravedad.

El siguiente avance que se va a emplear en este algoritmo es utilizar los ángulos de navegación cabeceo y alabeo. Estos ángulos se utilizan para saber en todo momento que rotación está realizando el vector de gravedad.

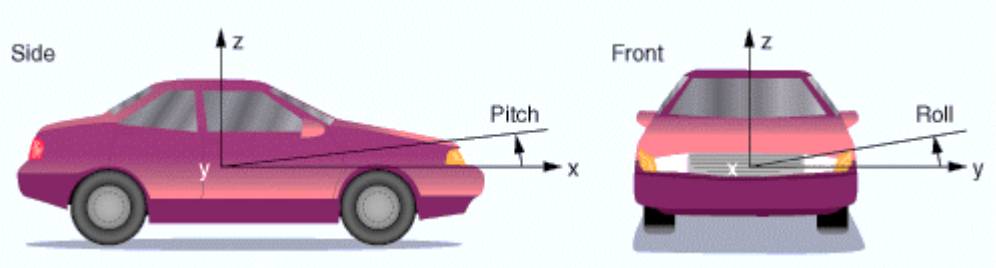
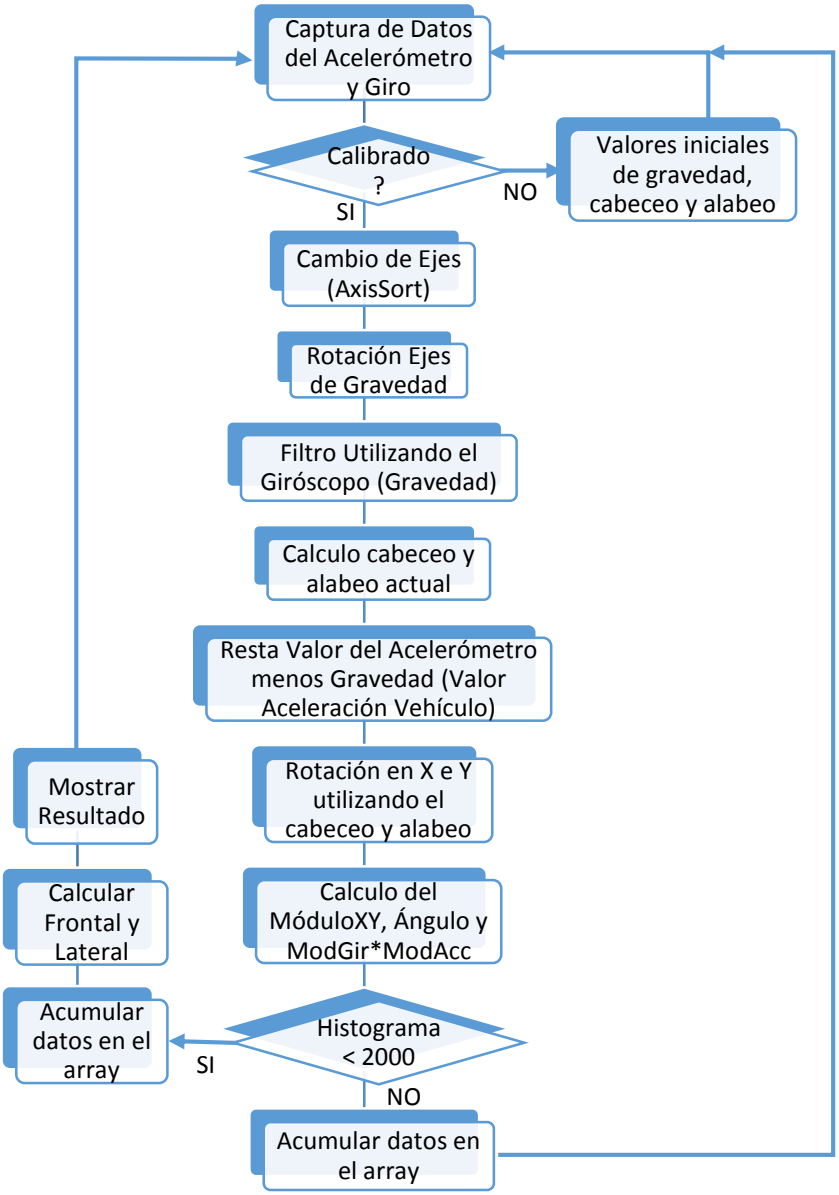


Figura 67: Ángulos cabeceo (Pitch) y alabeo (Roll) en un vehículo

A continuación se detalla un diagrama de flujo con los pasos que realiza el algoritmo.



Al comenzar el programa, y a continuación de calibrar los sensores, se realiza una inicialización para ver en qué posición se encuentra colocado el sensor y se rotaran los ejes con el método comentado haciendo que el eje Z corresponda con la gravedad. Esta primera toma de datos ha de hacerse con el vehículo sin movimiento, para poder calcular la posición de los ejes de gravedadX, gravedadY, gravedadZ, cabeceo y alabeo inicial.

$$PitchInicial = \sin^{-1}\frac{ValorGravEjeX}{Módulo\ de\ los\ Ejes\ de\ Gravedad\ X,Y,Z}$$

$$RollInicial = \tan^{-1}\frac{ValorGravEjeY}{ValorGravEjeZ}$$

A continuación se van tomando nuevos valores los cuales siempre se tratan de la misma manera. Primero se cambian los ejes con la función de cambios de ejes (AxisSort), luego se rotan los valores de gravedad en función del ángulo de rotación proporcionado por el giroscopio (al realizar estos pasos, se tiene localizando permanentemente la gravedad). Para esta rotación de los ejes X, Y, Z, se utiliza la siguiente matriz (realizada de multiplicar las 3 matrices de rotación de Euler en el orden X·Y·Z):

$$\begin{matrix} & & Rotación\ X \cdot Y \cdot Z & & \\ & & & & \\ GravX & \begin{pmatrix} cos(y) \cdot cos(z) & cos(y) \cdot -sen(z) & sen(y) \\ sen(x) \cdot sen(y) \cdot cos(z) + cos(x) \cdot sen(z) & sen(x) \cdot sen(y) \cdot -sen(z) + cos(z) \cdot cos(x) & -sen(x) \cdot cos(y) \\ cos(x) \cdot -sen(y) \cdot cos(z) + sen(x) \cdot sen(z) & cos(x) \cdot -sen(y) \cdot -sen(z) + sen(x) \cdot cos(z) & cos(x) \cdot cos(y) \end{pmatrix} \\ = GravY & & & & \\ GravZ & & & & \end{matrix}$$

Luego se utiliza el filtrado que se explicó anteriormente el cual se basa en los datos del giróscopo para modificar en menor o mayor medida el valor de la gravedad.

```
gravedadX = ValorGravedadRotadoPorElGiroX *0,99 + ValorAceleromX*(1-0,99);
gravedadY = ValorGravedadRotadoPorElGiroY *0,99 + ValorAceleromY*(1-0,99);
gravedadZ = ValorGravedadRotadoPorElGiroZ *0,99 + ValorAceleromZ*(1-0,99);
```

A esta altura del algoritmo, se tiene un valor del vector de gravedad correcto debido a que siempre va a estar localizado por el giroscopio. Ahora lo que se necesita es conseguir el valor actual de la aceleración. Para ello se coge cada valor del sensor y se le resta el valor anterior de gravedad. A continuación se rota este valor con el cabeceo y el alabeo del movimiento del vehículo para obtener el valor de las componentes X e Y en un plano de dos dimensiones. Esta rotación se realiza multiplicando las matrices $Z \cdot Y$ de los ángulos de Euler.

$$MovVehículo = \begin{pmatrix} DatoX - GravX \\ DatoY - GravY \\ DatoZ - GravZ \end{pmatrix} \begin{pmatrix} \cos(roll) \cdot \cos(pitch) & -sen(roll) & \cos(roll) \cdot sen(pitch) \\ sen(roll) \cdot \cos(pitch) & \cos(z) & sen(roll) \cdot sen(pitch) \\ -sen(pitch) & 0 & \cos(pitch) \end{pmatrix}$$

El valor del eje Z es desechado (porque indica la gravedad) y solo se aprovecha las componentes X e Y, se realiza el mismo proceso que en el anterior algoritmo para calcular el ángulo de dirección de desplazamiento rellenando un array de muestras. Además se va a utilizar otro array con el valor de la multiplicación del acelerómetro y del giroscopio. En este array van a acumular los valores que se obtengan de esta multiplicación y va a servir para localizar las aceleraciones frontales.

Es decir, se tendrán dos arrays, uno que es el mismo que en el algoritmo anterior y va a ir acumulando los ángulos en donde se produzcan la mayor cantidad de movimientos y otro que acumulará el valor de las aceleraciones multiplicado por los giros. Como ya se ha mencionado en anteriores ocasiones, los valores del acelerómetro serán valores pequeños, por lo tanto, el giro es el que proporcionara mayor o menor aumento en el valor del acumulador.

Aprovechando este hecho, para ver cuál es la parte frontal del vehículo, se busca en el array el valor más alto que contenga. Este valor corresponderá a un valor en el que el giroscopio proporcionó un valor elevado, por lo tanto se corresponde con una curva. Si a partir de este valor de mayor intensidad, el cual tendrá un ángulo determinado, se podrá encontrar el frontal o la parte trasera.

Para que la medida sea más eficaz y veras, no solo se utiliza el ángulo en donde se tenga el número más alto, sino que se utiliza un acumulado de 4° alrededor de este valor, tal y como se realizaba en el algoritmo anterior. Como las frenadas van a acumular un mayor valor que las aceleraciones, el valor acumulado menor es el que coincide con la parte frontal.

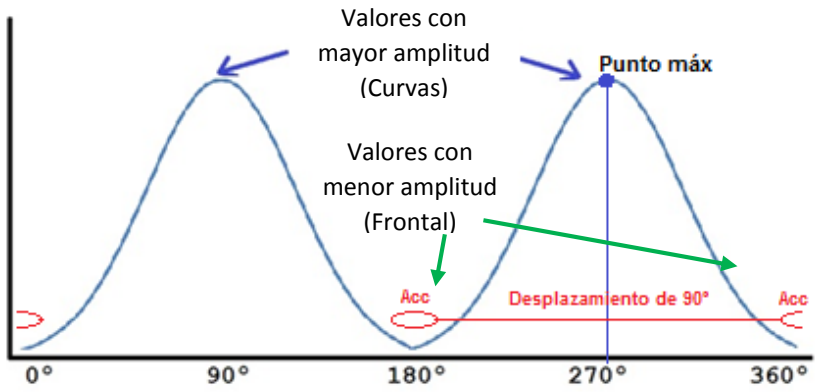


Figura 68: Ejemplo de una gráfica del array de la multiplicación del giro y el acelerómetro

En la figura 64 se observa como Acc es una acumulación de la suma de la multiplicación y como los puntos de máxima amplitud se encuentran en 90° y en 270° pertenecientes a las curvas. Al realizar el desplazamiento de ±90° se encuentra el frontal o la parte trasera y al comparar el Acc, se tendrá si la parte frontal se encuentra en 0° o en 180°.

Al saber el ángulo en donde se produce la aceleración frontal, y el ángulo del vehículo cuando se desplaza, se pueden restar ambos. Si el resultado de la resta es 0, quiere decir que el desplazamiento se está produciendo en la dirección frontal (cos(0)=1 no modifica el módulo y da aceleración frontal, mientras que sen(0) = 0 proporciona aceleración lateral nula), si el resultado es de 90°, quiere decir que se está girando el vehículo.

```
AceleraciónFrontral = ModuloXY * cos(ÁnguloXY - ÁnguloFrontal);
AceleraciónLateral  = ModuloXY * sin(ÁnguloXY - ÁnguloFrontal);
```

Para completar el algoritmo y mostrar el resultado de las aceleraciones, se coge la aceleración frontal y se hace una suma cuando este pase de un valor umbral de aceleración determinado. Y por pantalla se escribe el tiempo que duro dicha aceleración y el pico máximo de aceleración que alcanzó durante este periodo.

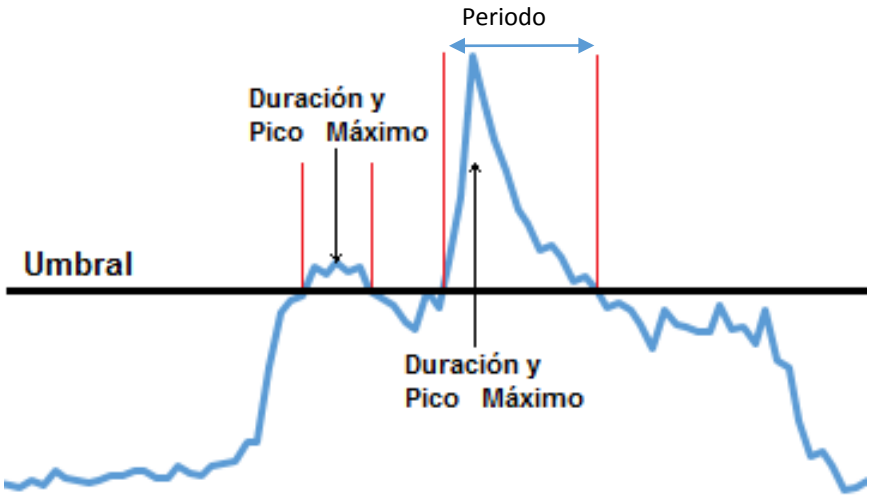


Figura 69: Información proporcionada por pantalla

Esto se tiene que realizar cuando el algoritmo tenga las suficientes muestras en el array, para estar seguro de que los datos de aceleración frontal son los correctos. Además de esto, como interesa también saber también cuando se frena, el algoritmo proporciona este dato en forma de aceleración negativa. Esto es debido a que el cos(ángulo) da negativo y cambia el signo, siendo perfecto para distinguir entre aceleración y frenado.

En los desplazamientos laterales ocurre lo mismo, distinguiendo un lateral positivo y otro lateral negativo. Es debido al mismo motivo, el sen(ángulo) se convierte en negativo y cambia el signo del módulo.

6.0 Programación y Pruebas

6.1 Introducción

Todas las pruebas se realizaron en un recorrido de aproximadamente 9 km, los cuales se recorrían por carretera urbana y por carretera local, con rotondas y cambios de nivel. Se partía desde la zona de Las Torres en Siete Palmas hasta la estación de guaguas de Tamaraceite, ida y vuelta. Se utilizaron dos dispositivos para realizar las medidas, el módulo inercial descrito en el capítulo 4 y un teléfono móvil Samsung Galaxy S3.

Para realizar las pruebas de los algoritmos con el módulo inercial, se utilizó un ordenador portátil para ver el resultado de los test. La placa PCB iba conectada con un adaptador Serie a USB para poder depurar los datos. Debido a la dificultad que planteaba este proceso, se optó por emplear el teléfono móvil para realizar los primeros test de cada algoritmo.

Se seleccionó el Galaxy S3 debido a que utiliza el mismo giróscopo y acelerómetro (LSM330DLC) que el utilizado por el módulo inercial. El Galaxy S3 cuenta con un procesador Samsung Exynos 4412, a una frecuencia de 1,4 GHz, 1 GB de memoria RAM y una pantalla de 4,8”. La desventaja de utilizar un dispositivo Android, es que por su carga de sistema operativo, opera a una tasa más baja que el modulo inercial y con tiempos de muestreo no constantes, lo que supone una imprecisión en las medidas. Por el contrario, ofrece la ventaja de una mayor comodidad, la capacidad de registrar los recorridos, y la facilidad de visualizar los datos en pantalla en tiempo real. Los algoritmos desarrollados en Android, fueron proporcionados por el tutor de este trabajo.

El sensor, tanto el del dispositivo móvil como el de la PCB, iba fijado al salpicadero del vehículo con cinta adhesiva para que no tuviera movimiento una vez calibrado. Los datos extraídos de los algoritmos se visualizaban por pantalla y se guardaban en un documento Excel (.csv) para posteriormente depurarlos y extraer conclusiones.

Antes de empezar con las pruebas de los algoritmos desarrollados, primero se realizó un test de funcionamiento del microcontrolador y de las conexiones. Para comprobar que

funcionaba correctamente se envió por el puerto serie una cadena de texto. Con ello se consigue acreditar que se accede correctamente a los registros y que es posible obtener la información que se desea en el terminal del PC.

La primera cadena de texto enviada fue "Ecosensor Version: 01", la cual fue recibida en el hyperTerminal del PC. Para ello se utilizó la Tasa de Binaria de 57.600 bps, la cual tiene velocidad suficiente para enviar los datos recogidos. Y la configuración del puerto serie elegida fue 8N1, es decir, se estableció a 57.600/8N1.

```
Ecosensor Version: 03
Author: Frank Perez Paz
```

Figura 70: Texto mostrado por el hyperTerminal

A continuación se configuró el puerto SPI para comunicarse con el dispositivo LSM330DLC. A través de este puerto se recibirán los datos del sensor que contendrán la información de aceleración y giro, los cuales serán utilizados para desarrollar el proyecto. Para comprobar la accesibilidad al dispositivo, lo primero que se hace después de configurarlo es leer el registro identificador del sensor. En el datasheet se encuentra el registro, que se observa en la siguiente figura.

WHO_AM_I_G (0Fh)

Table 67. WHO_AM_I_G register

1	1	0	1	0	1	0	0

Figura 71: Valor del registro Identificador del sensor LSM330DLC

Como se aprecia, el valor que tiene este registro de lectura es 11010100 en binario; si se traduce este código a hexadecimal, se obtiene el valor de D4. Para comprobar que el puerto SPI se comunica correctamente con el sensor, el dato recibido al leer el registro identificador se envía por el puerto serie.

```
Ecosensor Version: 03
Author: Frank Perez Paz
Identificador del Sensor: D4
```

Figura 72: Muestra el valor del registro del identificador del sensor

Se puede comprobar que el valor recibido es igual al valor que se esperaba, por lo tanto, esto demuestra que el puerto SPI se ha configurado correctamente y que el sensor está conectado y funcionado adecuadamente. A continuación se desarrolla el software que se desea implementar.

6.2 Prueba del 1º algoritmo de desarrollo

Una vez pensado cómo se tiene que programar para observar los resultados, se empezó a programar en Java sobre Android para visualizar el resultado de los ejes del acelerómetro y la respuesta obtenida al paso de los filtros. Se empezaron las pruebas utilizando el sensor del móvil (LSM330D mismo sensor que utiliza el dispositivo) debido a las ventajas que ofrece el poder recopilar datos, visualizar los datos y los resultados en tiempo real y además se tiene la opción de guardar los datos del sensor y los resultados, en la tarjeta SD del móvil para posteriormente analizarlos adecuadamente.

Al realizar las pruebas, se determinó que el algoritmo no ofrecía una medida real de las aceleraciones. Lo primero que se observó fue que la etapa de filtrado era demasiado lenta, el orden de los filtros era muy alto lo cual retrasaba y producía un sobre-amortiguamiento en la señal.

En la siguiente imagen se observa el aspecto que tiene la aplicación realizada en donde se muestran las medidas de los ejes **X**, **Y**, **Z** y el resultado de la salida de cada filtro. Las siglas LP son las que indican el resultado del filtro paso-bajo. Las siglas BP son el resultado del filtro paso-banda y por último las siglas HP son asignadas al filtro paso-alto.

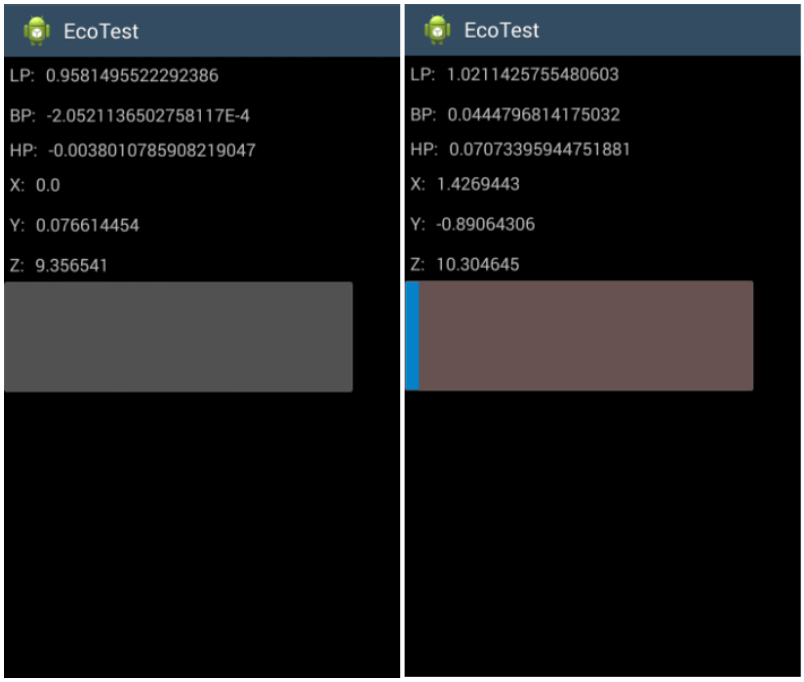


Figura 73: Medidas realizadas por la aplicación

Analizando los datos, se detectó que el acelerómetro con el que se trabaja tiene diferentes medidas para los diferentes ejes. Esto quiere decir, que al posicionar el acelerómetro en una posición en donde el eje **Z** se encuentre vertical respecto a la gravedad, dará un valor diferente a si colocamos el eje **X** o **Y** en la misma posición. Al tener esta diferencia en los ejes de medida, se produce una medida falsa del vector que produce la aceleración, y no se puede restar directamente a este vector, el valor de la gravedad (9.8 m/s^2).

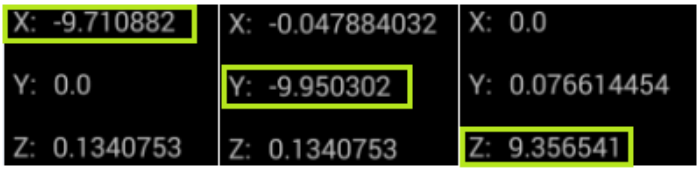


Figura 74: Medidas de los ejes en posiciones ortogonales respecto a la gravedad

El offset que se produce en los ejes **X**, **Y** es como máximo un 1.5% con lo cual se puede despreciar porque el valor de error que se va a producir es despreciable. El eje **Z** en cambio, produce un error mayor. Este error es debido a la calibración, y viene desde fábrica sin posibilidad de corregirlo modificando sus registros. Para solucionar este error, habría que cambiar de acelerómetro y buscar uno de un precio superior que no diera este tipo de errores, con lo que se incrementarían los costes.

6.3 Prueba del 2º algoritmo de desarrollo

Para realizar las pruebas y ver el funcionamiento del algoritmo, se utilizó, como en el algoritmo anterior, la ayuda del dispositivo móvil. Se desarrolló la aplicación teniendo en cuenta los filtros que se iban a utilizar y cómo se tenían que rotar los ejes para obtener el resultado esperado. A continuación se muestra una captura de pantalla del dispositivo en reposo y otra con una aceleración.

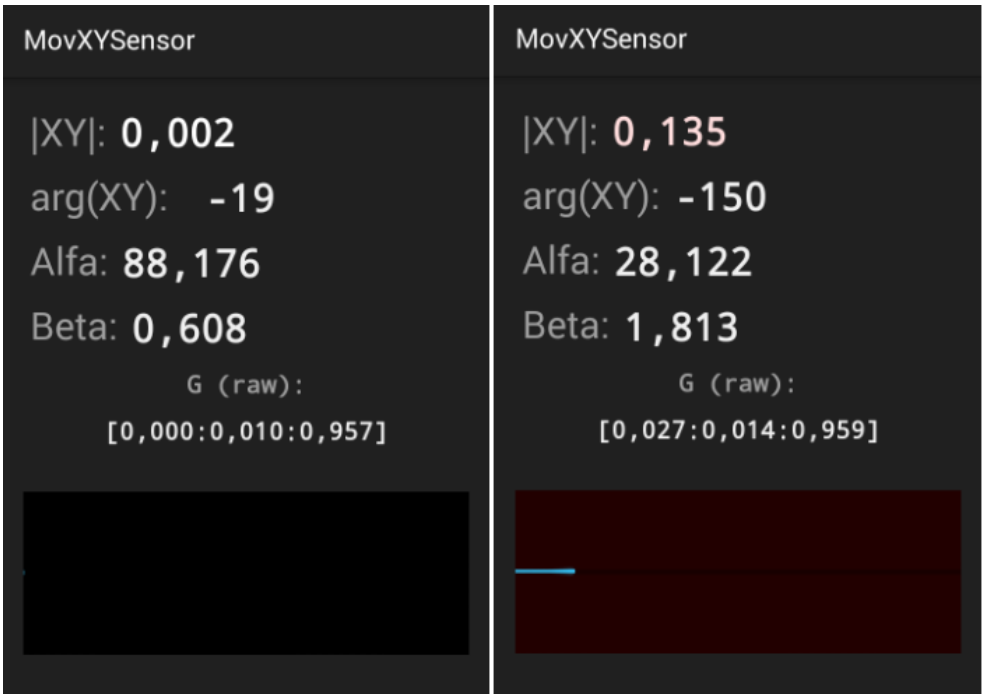


Figura 75: Medidas realizadas aplicando el algoritmo explicado

Se puede observar el modulo resultante |XY| de las aceleraciones sin que influyan las aceleraciones gravitacionales, el ángulo formado por los ejes **X** e **Y**, los ángulos Alfa y Beta utilizados para la rotación de los ejes y las medidas de las fuerzas G medidas al pasar por el filtro de gravedad.

Para poder analizar los datos obtenidos y procesarlos en un PC y además para visualizar en todo momento las medidas proporcionadas por los distintos ejes se desarrolló conjuntamente otra aplicación que guardaba los datos de los acelerómetros y además los representaba gráficamente como se muestra en la siguiente figura.

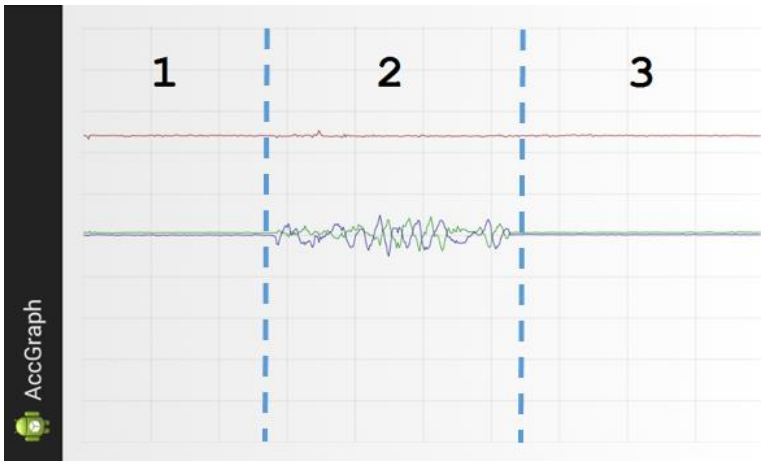


Figura 76: Gráfica de los ejes del acelerómetro cuando el dispositivo está alineado al eje de la gravedad

En color rojo se muestra el eje **Z**, en color verde se representa el eje **X** y por último se puede observar el eje **Y** en color azul. En estado de reposo (representados por los números 1 y 3 en la figura 72) el eje **Z** adquiere el valor de la gravedad y los demás ejes son prácticamente 0. Esto es debido a que el móvil se encuentra en una superficie plana y el eje **Z** coincide con el de la gravedad. Al realizar un desplazamiento sobre la superficie, los ejes **X** e **Y** empiezan a tomar las medidas de la aceleración.

Al estar el dispositivo alineado, es decir, el eje **Z** es paralelo al eje de la gravedad, es más sencillo realizar la medida, pero para este proyecto la idea es que el dispositivo no esté alineado. Por tanto, la aceleración de la gravedad interfiere en todos los ejes y es más difícil realizar las conclusiones a la hora de observar la gráfica. Esto se muestra a continuación.

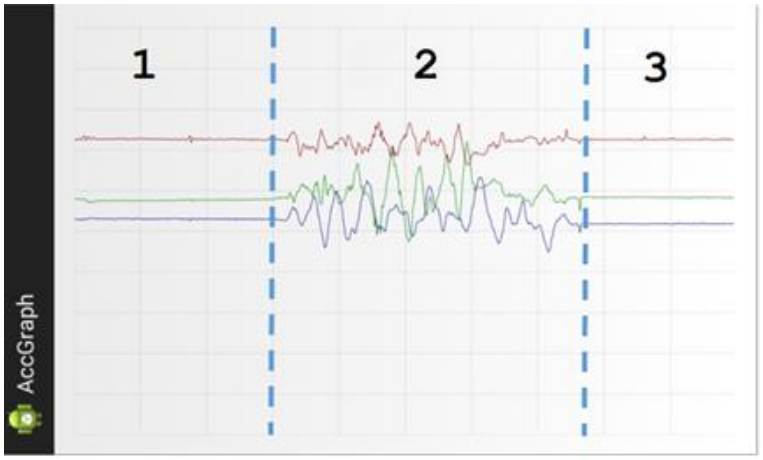


Figura 77: Gráfica de los ejes del acelerómetro cuando el dispositivo está no se encuentra alineado al eje de la gravedad

Otra opción es la de mostrar el resultado de los ejes al pasar por los distintos filtrados como el filtrado paso-bajo, el filtrado paso-alto y el paso-banda. En la siguiente figura se muestra la gráfica en donde se activa el filtrado de los ejes del acelerómetro. Se puede ver como el filtro (paso-alto) realiza el trabajo de compensar la señal del eje **Z** el cual pertenece a la señal de la gravedad (señal continua).

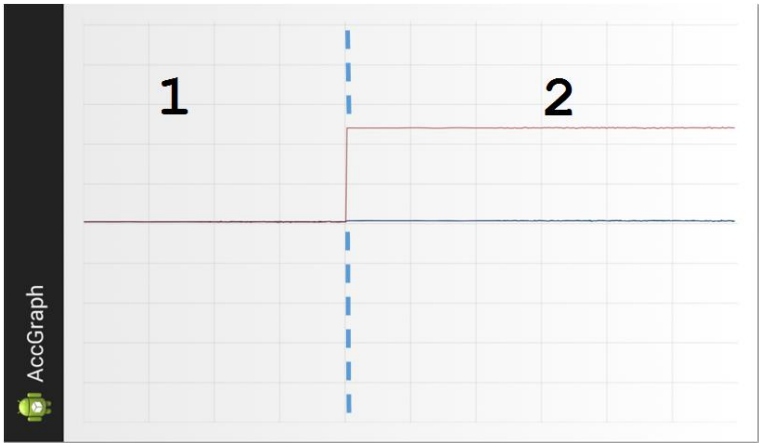


Figura 78: Gráfica de valores de los ejes con filtros activados (1) y desactivados (2)

Como se quería ver el espectro de la señal de los ejes, se realizó una gráfica que mostrara los datos de los acelerómetros sin filtrar. Para mostrar el espectro de la señal, se utilizó la transformada rápida de Fourier (FFT), la cual muestra cómo afecta la gravedad a los ejes de medida.

Figura 79: Realización de la FFT sobre los ejes del acelerómetro

En esta figura se observa cómo se tiene un valor que destaca sobre el resto en el origen, es decir, un valor continuo. Este valor es el perteneciente a la gravedad y con él se puede hacer una estimación sobre los valores del filtro de gravedad. Como se mencionó anteriormente, este filtro paso-bajo se fijó a 2 Hz, frecuencia suficiente para eliminar este valor continuo.

Al hacer las pruebas de funcionamiento con este algoritmo, se detectaron algunos problemas que causaban errores en las medidas reales de aceleración. El problema más significativo se encontró al realizar maniobras de giro, en donde sin acelerar, el sensor

proporcionaba una medida de aceleración. Esto es debido a que el sensor es más sensible a las fuerzas centrifugas que a las aceleraciones producidas.

Con este algoritmo los resultados obtenidos eran válidos solo en determinadas circunstancias siendo ineficaz para realizar mediciones óptimas. Un ejemplo de malas medidas tomadas por el sensor es cuando el vehículo se encuentra en movimiento sin aceleración, es decir, con aceleración inercial y se gira el volante para tomar una curva. El sensor da una medida de aceleración bastante aguda, del orden de 0,3 g. Esta medida es incluso mayor que cuando se produce una aceleración normal.

Otro aspecto que se detectó es que cuando se realiza una aceleración constante, el sensor dejaba de detectar esta aceleración. Esto es debido al filtrado que se realiza para la gravedad. Esta aceleración constante es tomada por el filtro como un valor continuo y la toma como si fuera el valor de la gravedad.

Para solucionar este inconveniente, se decidió crear un tercer algoritmo el cual corrigiera las aceleraciones en giro y no eliminara las aceleraciones continuas. Aunque las aceleraciones continúas no suponen un grave problema puesto que si se trata de detectar la conducción eficiente, una aceleración continua es una buena manera de conducir, en vez de ir produciendo acelerones.

6.4 Prueba del 3º algoritmo de desarrollo

Para realizar los cambios de orden de filtros y frecuencias, se utilizaron bibliotecas que implementan directamente los tres tipos de filtros digitales más comunes: CHEBYSHEV, BUTTERWORTH y BESSEL. La biblioteca que se importa para filtrar las señales es: dsp.filter.IirFilter, la cual ya viene integrada en las bibliotecas de java por defecto.

Ahora queda configurar adecuadamente los filtros para que las medidas del sensor sean reales y no sean eliminadas señales importantes. Cuando se utiliza un filtro de gravedad estrecho, se mejora la cantidad de datos obtenidos del sensor, pero se empeoran los resultados a causa que interfiere la gravedad en la medida. Si se utiliza un filtro de gravedad ancho, se asegura el eliminar la gravedad, pero también se eliminan datos del sensor que son útiles y no se deben eliminar. Por lo que hay que llegar a un punto de concurrencia para obtener los mejores resultados.

Para comprobar el filtro que mejor funciona para detectar las aceleraciones y utilizar para el proyecto, se probaron tres filtros de gravedad diferentes. El primero que se probó fue un filtro paso-bajo Bessel de 0,4 Hz, luego se probó con 0,6 Hz y por último se realizó la misma prueba con un filtro de 0,8 Hz.

El resultado obtenido con los distintos filtros se puede apreciar en las siguientes figuras. En todas las pruebas se aceleró lo máximo posible el vehículo en las tres primeras marchas. En el filtrado con el filtro a 0,4 Hz se pueden apreciar perfectamente las tres marchas del coche pertenecientes a los tres picos de la gráfica. Sin embargo con el filtro a 0,6 y a 0,8 Hz en la segunda marcha aparece un corte en la aceleración que no se llegó a producir, es más en la gráfica de 0,8 Hz se divide completamente la segunda marcha.

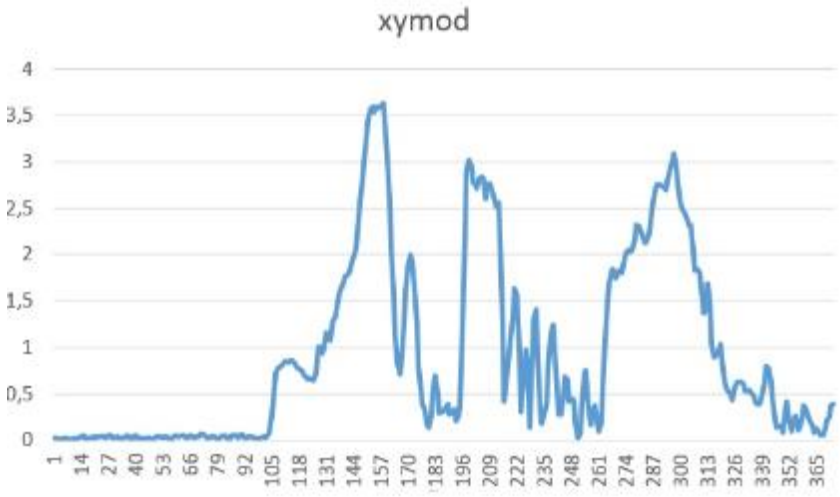


Figura 80: Filtrado a 0,4 Hz

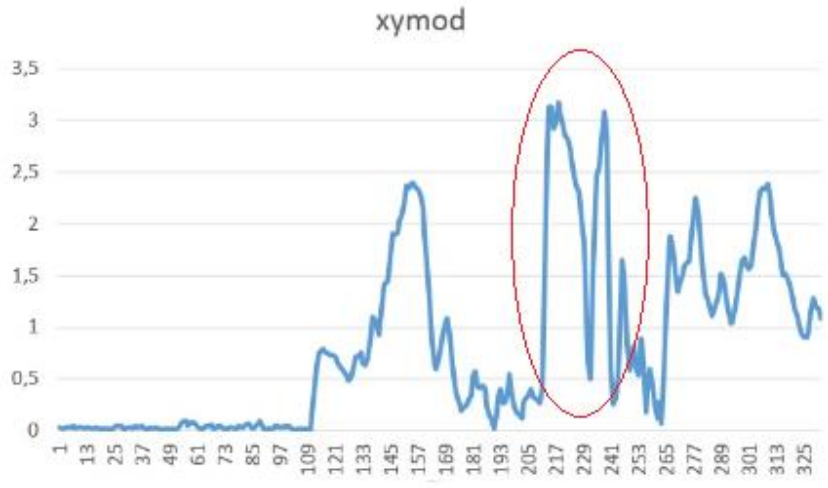


Figura 81: Filtrado a 0,6 Hz

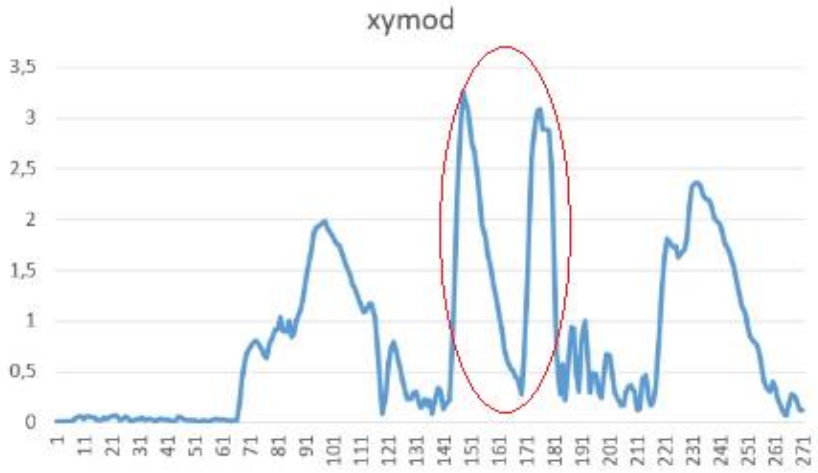


Figura 82: Filtrado a 0,8 Hz

Esto es debido al filtro de gravedad, el cual al ser una aceleración continua, elimina la señal interpretando que forma parte de la señal continua de gravedad. Este hecho es lo que se tiene que evitar, por lo tanto, el filtro más adecuado para utilizar es el de 0,4 Hz. Una vez elegido el filtro de gravedad adecuado, se realizó una prueba para ver si el algoritmo realizado funcionaba correctamente en pendientes y en llano.

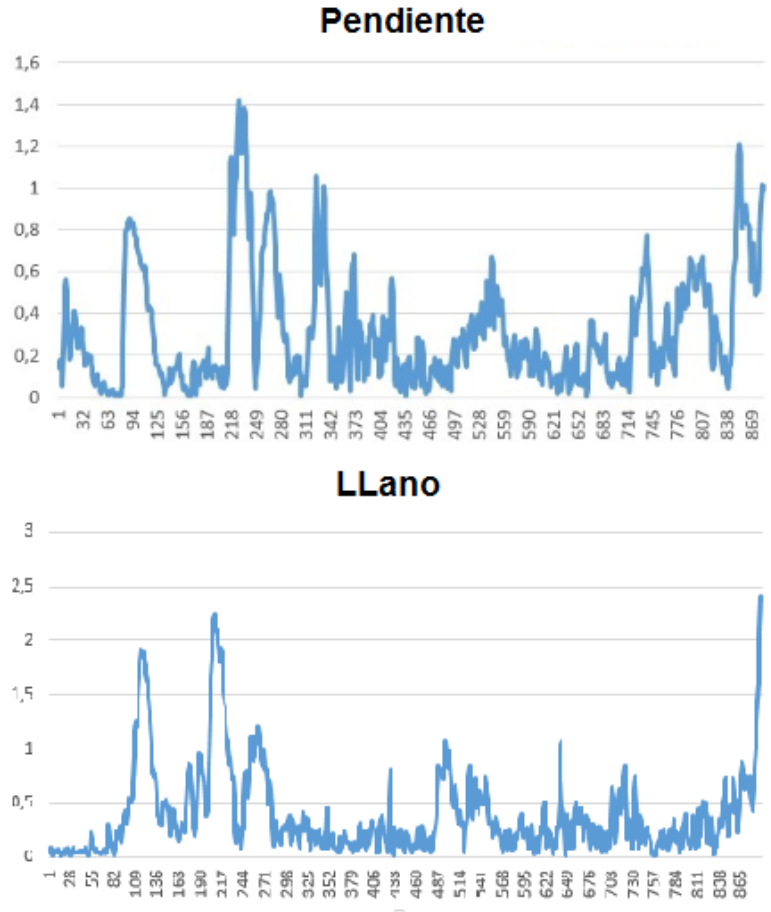


Figura 83: Prueba de Eliminación de gravedad

Como se aprecia, las gráficas son muy similares, coinciden en los picos de aceleración y en su forma. El primer pico pertenece al primer cambio y el siguiente pico y el de mayor amplitud pertenece al movimiento del vehículo al cambiar de primera a segunda marcha, el tercer pico pertenece a la segunda marcha y así sucesivamente. Se cambió hasta la cuarta marcha y se realizó una frenada que es la que se puede ver que finaliza la gráfica.

Después de comprobar que las aceleraciones en pendiente y en llano son iguales en cuanto a las respuestas obtenidas, y que el filtro de gravedad funciona en ambas localizaciones, se pasó a realizar la programación del algoritmo necesario para localizar las aceleraciones del vehículo y así conseguir el resultado deseado.

El resultado de la prueba realizada en el dispositivo móvil es el que se aprecia en la siguiente imagen, donde se puede observar y extraer varias conclusiones:

- Para empezar de la gráfica superior donde se encuentran el módulo de las aceleraciones multiplicado por el modulo del giro se distinguen dos grandes picos que sobrepasan el valor de 10. Estos picos no se encuentran, si son comparados con la gráfica inferior de número de muestras, en ningún momento especial y no aportan información contundente que se pueda extraer. Lo único que se puede interpretar es que el giro debe de ser el causante de introducir tantas variaciones en la medida.

- Si centramos la atención en la gráfica inferior, se pueden apreciar dos picos de gran intensidad y otros dos de menor intensidad. En los picos de más abruptos, superan las 250 muestras, mientras que los picos más pequeños están en el orden de las 150 muestras. Si es verdad lo que se planteaba anteriormente, los picos más intensos han de ser los de aceleración y frenado y los menos intensos los giros.

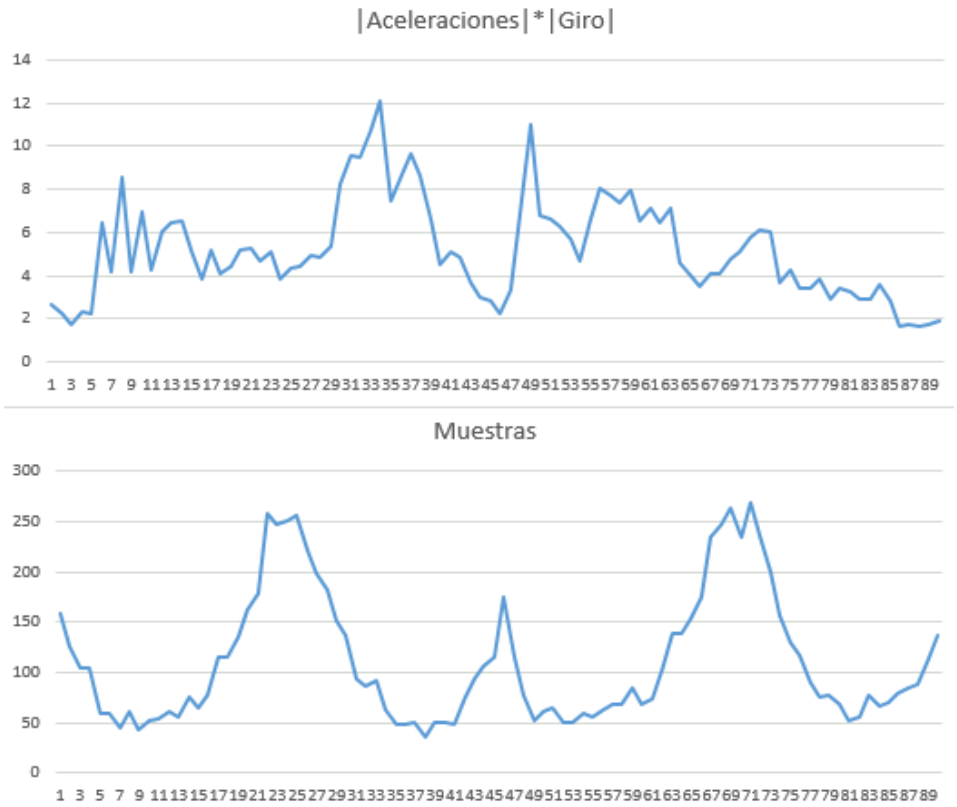


Figura 84: Datos recogidos por el sensor del dispositivo móvil

Si la gráfica de muestras la mostramos en un gráfico radial en donde el punto central es el sensor y los puntos de medida son los 90° en los que se han dividido los 360°, se observa la siguiente figura.

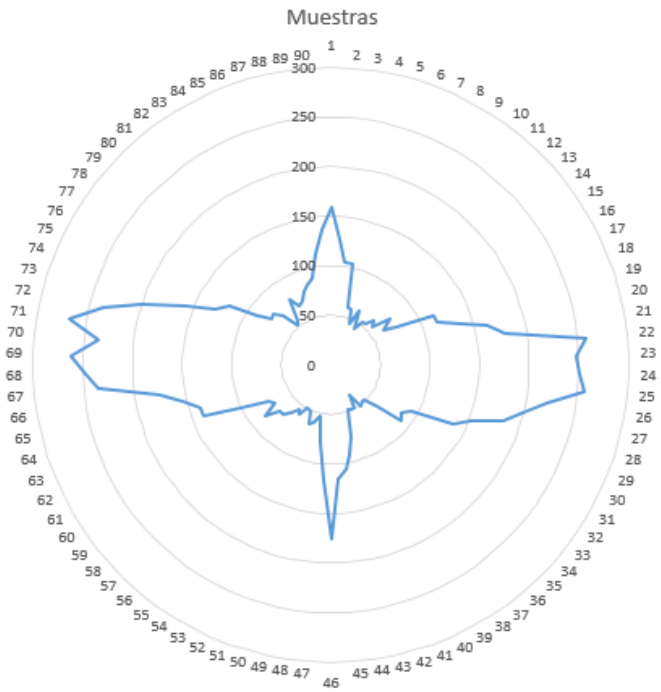


Figura 85: Muestras obtenidas mostradas en un diagrama radial

Los dos picos grandes de la gráfica pertenecen a las aceleraciones y frenazos y los picos más pequeños son los giros. Si utilizamos de referencia el centro de la gráfica, se ve como los picos grandes se encuentran en horizontal y los pequeños en la línea vertical. Con esto podemos confirmar la dirección de movimiento del vehículo referenciado con el sensor. Si el vehículo se desplaza con el ángulo 70°x4 o 25°x4, es decir, 280° o 100° se está produciendo una acción de aceleración o frenado, mientras que si el desplazamiento es en 46°x4 o en 1°x4 (4° y 184°), se produce un giro a derecha o izquierda.

Con las argumentaciones anteriores, extraídas de la gráfica, se puede llegar a la siguiente conclusión: es posible obtener la dirección de avance lineal y desplazamiento lateral del vehículo, realizando una cuenta del número de muestras. Para ello basta con saber el ángulo que se forma cada vez que se produzca una aceleración.

Con las pruebas realizadas en el dispositivo móvil, se empezará a programar el microcontrolador para obtener las medidas anteriores. Una vez se obtengan las mismas medidas válidas, se completará el algoritmo de programación fijando los ángulos de aceleración y giro para empezar a tomar las medidas reales.

Como ya se tiene un modelo funcionando en el dispositivo móvil, se decidió realizar un cambio del lenguaje de programación *Java* a *C*. Como son lenguajes similares, el pasar el algoritmo es una tarea sencilla, lo difícil es cuando se tiene que programar determinados controles que no vienen implementados en C, como por ejemplos determinados registros del microcontrolador y del sensor y los algoritmos de filtrado.

Como se mencionó anteriormente, Java tiene implementadas bibliotecas de filtros, los cuales hacen mucho más fácil el trabajo de programación, basta con elegir el filtro y las frecuencias de corte que se utilizaran. En la programación C, el filtrado hay que hacerlo manualmente, cambiando los coeficientes del filtro.

Además, en el dispositivo móvil, no se controla o no se tiene acceso a modificar los parámetros del sensor LSM330DLC, simplemente cada vez que el sensor toma una medida, realiza una interrupción. En el microcontrolador, esta interrupción hay que programarla escribiendo vía SPI en los registros de control del sensor. La ventaja que tiene el poder programar los registros de configuración del sensor son varias como:

- Control total de las interrupciones.
- Cambiar el acceso a los datos modificando la salida de los ejes del sensor (ByPass, FIFO, Stream, Bypass-to-stream).
- Modificar la frecuencia de captura de datos.
- Modificar su BW.
- Utilizar el modo normal o el modo de bajo consumo.
- Cambiar la escala de medidas y la resolución (8 bits o 12 bits).
- Utilizar umbrales de detección del sensor.
- Habilitar o deshabilitar los ejes por separado.

En un principio se iba a utilizar la fuente de interrupción del sensor, pero se determinó que se controlaba mejor si la interrupción era generada por el microcontrolador a una frecuencia de 100 Hz. Por lo que se eliminaron las fuentes de interrupción del sensor y programó este para que tuviera un *Data Rate* de 100 Hz con *Full Resolution*. Los registros que se utilizaron fueron los que se pueden ver en las siguientes tablas.

CTRL_REG1_A (20h)

ODR3	ODR2	ODR1	ODR0	LPen	Zen	Yen	Xen

ODR3-0	Data rate selection. Default value: 0 (0000: Power-down; Others: refer to *Table 21*, "Data rate configuration")
LPen	Low power mode enable. Default value: 0 (0: Normal mode, 1: Low power mode)
Zen	Z axis enable. Default value: 1 (0: Z axis disabled; 1: Z axis enabled)
Yen	Y axis enable. Default value: 1 (0: Y axis disabled; 1: Y axis enabled)
Xen	X axis enable. Default value: 1 (0: X axis disabled; 1: X axis enabled)

ODR3	ODR2	ODR1	ODR0	Power mode selection
0	0	0	0	Power-down mode
0	0	0	1	Normal / Low power mode (1 Hz)
0	0	1	0	Normal / Low power mode (10 Hz)
0	0	1	1	Normal / Low power mode (25 Hz)
0	1	0	0	Normal / Low power mode (50 Hz)
0	1	0	1	Normal / Low power mode (100 Hz)
0	1	1	0	Normal / Low power mode (200 Hz)
0	1	1	1	Normal / Low power mode (400 Hz)
1	0	0	0	Low power mode (1.620 kHz)
1	0	0	1	Normal (1.344 kHz) / Low power mode (5.376 kHz)

Tabla 3: Registro número 1 del acelerómetro.

CTRL_REG4_A (23h)

0(1)	BLE	FS1	FS0	HR	0(1)	0(1)	SIM

1. This bit must be set to '0' for correct operation.

BLE	Big/little endian data selection. Default value 0. (0: Data LSb @ lower address; 1: Data MSb @ lower address)
FS1-FS0	Full Scale selection. default value: 00 (00: +/- 2G; 01: +/- 4G; 10: +/- 8G; 11: +/- 16G)
HR	High resolution output mode: Default value: 0 (0: High resolution disable; 1: High resolution enable)
SIM	SPI serial interface mode selection. Default value: 0 (0: 4-wire interface; 1: 3-wire interface).

Tabla 4: Registro número 4 del acelerómetro.

Cuando se ha configurado el acelerómetro, se pasa a configurar el microcontrolador para que ejecute una rutina de interrupción cada 10 milisegundos. Cada vez que se produzca la interrupción se leerá el valor del acelerómetro y se realizará el análisis y posterior valoración de la eficiencia sobre la conducción.

Lo primero que se realizó, para comprobar si los valores del acelerómetro eran correctos, fue mostrar los datos por el puerto serie. En la figura 82 se puede comprobar las medidas tomas por el sensor.

```
Acel: X=0.000 Y=0.052 Z=-0.950 ||| Giro: X=1.837  Y=38.938  Z=28.175
Acel: X=0.038 Y=0.052 Z=-0.954 ||| Giro: X=0.962  Y=37.625  Z=28.350
Acel: X=0.000 Y=0.052 Z=-0.956 ||| Giro: X=1.925  Y=37.888  Z=24.325
Acel: X=0.030 Y=0.044 Z=-0.992 ||| Giro: X=0.962  Y=39.900  Z=29.925
Acel: X=0.030 Y=0.050 Z=-0.950 ||| Giro: X=2.275  Y=36.662  Z=28.962
Acel: X=0.030 Y=0.050 Z=-0.992 ||| Giro: X=0.700  Y=37.800  Z=29.400
Acel: X=0.000 Y=0.048 Z=-0.956 ||| Giro: X=3.150  Y=39.638  Z=29.575
Acel: X=0.034 Y=0.050 Z=-0.958 ||| Giro: X=2.450  Y=37.362  Z=29.925
Acel: X=0.000 Y=0.044 Z=-0.992 ||| Giro: X=1.663  Y=37.188  Z=29.050
Acel: X=0.034 Y=0.050 Z=-0.962 ||| Giro: X=4.813  Y=40.862  Z=26.337
Acel: X=0.034 Y=0.040 Z=-0.954 ||| Giro: X=3.675  Y=39.900  Z=28.000
Acel: X=0.034 Y=0.044 Z=-0.952 ||| Giro: X=2.362  Y=38.412  Z=30.450
Acel: X=0.000 Y=0.048 Z=-0.970 ||| Giro: X=4.375  Y=39.638  Z=29.400
Acel: X=0.036 Y=0.048 Z=-0.952 ||| Giro: X=0.875  Y=39.813  Z=28.962
Acel: X=0.036 Y=0.054 Z=-0.952 ||| Giro: X=1.225  Y=36.662  Z=27.387
Acel: X=0.000 Y=0.048 Z=-0.962 ||| Giro: X=-0.875  Y=38.500  Z=26.775
Acel: X=0.034 Y=0.046 Z=-0.952 ||| Giro: X=-0.262  Y=41.650  Z=25.375
Acel: X=0.034 Y=0.050 Z=-0.952 ||| Giro: X=1.575  Y=40.162  Z=26.950
Acel: X=0.036 Y=0.048 Z=-0.958 ||| Giro: X=3.763  Y=37.100  Z=28.788
```

Figura 86: Datos del sensor sin filtrar

Como se puede observar, las medidas con el sensor sin movimiento son correctas debido a que la PCB está colocada para que el eje Z coincida con la gravedad. Y como ya se observó en las primeras pruebas con el dispositivo móvil, el eje Z no mide exactamente los 9,8g de la gravedad. El sensor giroscópico también se comprobó para saber si era correcto el acceso a él, tiene un offset que es normal en la mayoría de los giróscopos.

También para comprobar el funcionamiento del sistema de filtrado, se visualizó por pantalla los datos de salida del filtro de gravedad y del filtro de datos + gravedad. El sensor se encontraba inmóvil, asi que los datos son prácticamente iguales.

```
FiltroGrav: X=0.023 Y=0.051 Z=-0.965 ||| FiltroSens: X=0.007 Y=0.050 Z=-0.968
FiltroGrav: X=0.023 Y=0.051 Z=-0.965 ||| FiltroSens: X=0.019 Y=0.051 Z=-0.972
FiltroGrav: X=0.023 Y=0.051 Z=-0.965 ||| FiltroSens: X=0.029 Y=0.052 Z=-0.973
FiltroGrav: X=0.023 Y=0.051 Z=-0.965 ||| FiltroSens: X=0.032 Y=0.054 Z=-0.972
FiltroGrav: X=0.023 Y=0.051 Z=-0.965 ||| FiltroSens: X=0.032 Y=0.055 Z=-0.968
FiltroGrav: X=0.023 Y=0.050 Z=-0.965 ||| FiltroSens: X=0.034 Y=0.053 Z=-0.963
FiltroGrav: X=0.023 Y=0.050 Z=-0.965 ||| FiltroSens: X=0.036 Y=0.052 Z=-0.959
FiltroGrav: X=0.023 Y=0.050 Z=-0.965 ||| FiltroSens: X=0.035 Y=0.052 Z=-0.958
FiltroGrav: X=0.023 Y=0.050 Z=-0.965 ||| FiltroSens: X=0.030 Y=0.051 Z=-0.958
FiltroGrav: X=0.023 Y=0.050 Z=-0.965 ||| FiltroSens: X=0.022 Y=0.050 Z=-0.958
FiltroGrav: X=0.023 Y=0.050 Z=-0.965 ||| FiltroSens: X=0.016 Y=0.049 Z=-0.959
FiltroGrav: X=0.023 Y=0.050 Z=-0.965 ||| FiltroSens: X=0.018 Y=0.048 Z=-0.960
FiltroGrav: X=0.023 Y=0.050 Z=-0.965 ||| FiltroSens: X=0.025 Y=0.050 Z=-0.961
FiltroGrav: X=0.023 Y=0.050 Z=-0.965 ||| FiltroSens: X=0.030 Y=0.051 Z=-0.965
FiltroGrav: X=0.023 Y=0.050 Z=-0.965 ||| FiltroSens: X=0.033 Y=0.051 Z=-0.972
FiltroGrav: X=0.023 Y=0.050 Z=-0.965 ||| FiltroSens: X=0.035 Y=0.052 Z=-0.977
FiltroGrav: X=0.023 Y=0.050 Z=-0.965 ||| FiltroSens: X=0.036 Y=0.052 Z=-0.979
FiltroGrav: X=0.023 Y=0.050 Z=-0.965 ||| FiltroSens: X=0.036 Y=0.051 Z=-0.979
FiltroGrav: X=0.023 Y=0.050 Z=-0.965 ||| FiltroSens: X=0.034 Y=0.049 Z=-0.978
FiltroGrav: X=0.023 Y=0.050 Z=-0.965 ||| FiltroSens: X=0.029 Y=0.048 Z=-0.972
FiltroGrav: X=0.023 Y=0.050 Z=-0.965 ||| FiltroSens: X=0.022 Y=0.048 Z=-0.965
FiltroGrav: X=0.023 Y=0.050 Z=-0.965 ||| FiltroSens: X=0.021 Y=0.048 Z=-0.960
FiltroGrav: X=0.024 Y=0.050 Z=-0.965 ||| FiltroSens: X=0.026 Y=0.048 Z=-0.959
```

Figura 87: Datos del acelerómetro filtrados

Una vez verificado el funcionamiento de los filtros, se realizan los cálculos para averiguar los ángulos de rotación y se aplican las rotaciones de Euler. El resultado de los cálculos, se envían por el puerto USART hacia el hyperTerminal para poder visualizarlos. En la siguiente figura se observa el valor del módulo de los ejes X e Y rotados, su argumento $[atan2\,(Y\,,X)]$, el ángulo XY que forman utilizando el filtro de gravedad $[atan2\,(Valor\ Filtro\ Gravedad\ Y\,,Valor\ Filtro\ Gravedad\ X)]$ y el ángulo XZ de igual manera pero utilizando el valor del módulo XY de gravedad y el eje Z del filtro de gravedad.

```
Mod:0.01  <-147.71  AngXY:63.07  AngXZ:176.68
Mod:0.01  <-151.66  AngXY:63.10  AngXZ:176.68
Mod:0.01  <-162.82  AngXY:63.13  AngXZ:176.67
Mod:0.01  <-158.24  AngXY:63.15  AngXZ:176.67
Mod:0.01  <-126.67  AngXY:63.16  AngXZ:176.66
Mod:0.00  <-64.54   AngXY:63.15  AngXZ:176.66
Mod:0.01  <40.76    AngXY:63.14  AngXZ:176.65
Mod:0.02  <53.77    AngXY:63.12  AngXZ:176.64
Mod:0.01  <61.34    AngXY:63.10  AngXZ:176.63
Mod:0.01  <69.30    AngXY:63.09  AngXZ:176.63
Mod:0.01  <71.84    AngXY:63.08  AngXZ:176.62
Mod:0.01  <72.03    AngXY:63.08  AngXZ:176.61
Mod:0.00  <-117.34  AngXY:63.09  AngXZ:176.61
Mod:0.01  <-121.61  AngXY:63.11  AngXZ:176.60
Mod:0.01  <-134.84  AngXY:63.14  AngXZ:176.60
Mod:0.00  <130.06   AngXY:63.18  AngXZ:176.60
Mod:0.01  <88.63    AngXY:63.23  AngXZ:176.59
Mod:0.01  <86.07    AngXY:63.29  AngXZ:176.59
Mod:0.02  <80.25    AngXY:63.36  AngXZ:176.59
Mod:0.02  <70.28    AngXY:63.44  AngXZ:176.59
Mod:0.01  <57.86    AngXY:63.54  AngXZ:176.59
Mod:0.00  <45.18    AngXY:63.65  AngXZ:176.59
Mod:0.00  <71.32    AngXY:63.78  AngXZ:176.59
```

Figura 88: Valores del módulo, argumento, ángulo XY y ángulo XZ

Como se aprecia en la figura anterior, el dispositivo esta fijo sin movimiento por lo que el modulo es prácticamente 0. El argumento varía desde valores negativos hasta positivos debidos a que como los valores de X e Y son tan pequeños, cualquier cambio afecta mucho al valor, sin embargo, cuando el sensor está en movimiento, los valores de X e Y se diferencian y el argumento adquiere un valor correcto. Para asegurar que el valor del ángulo del argumento sea el correcto, se pone un límite de no tomar datos hasta que el modulo sea superior a un valor de 0,05g.

Cuando el valor del módulo sea superior al indicado, se pasa a realizar la suma de las muestras del argumento como se explicó en el tercer algoritmo. Para guardar esta suma se utiliza un Array de longitud 90 en donde cada valor representa un grado. Es decir cada valor del Array va a corresponderse con un grado y cada grado equivale a otros cuatro para completar los 360° de la circunferencia. Cada vez que un argumento sea válido, corresponderá a una posición del Array, al cual le será sumado uno para de esta manera ver los valores que más veces se repiten.

Cuando el análisis del movimiento del vehículo ya este realizado (se realizará la primera vez que el vehículo se encuentre en movimiento y se recojan las suficientes

muestras para saber su dirección), solo medirá las intensidades de las aceleraciones o desaceleraciones en la dirección correcta.

Para saber si al encender el vehículo ya se ha realizado el análisis de la dirección correcta de aceleración, se utiliza una dirección de memoria EEPROM en donde se escribirá cuando se realice el análisis. Cuando el microcontrolador se enciende, se lee la dirección de memoria en la EEPROM para saber si ya se ha realizado el proceso de aprendizaje. Cuando se ha realizado, se lee la dirección de aceleración correcta y se empiezan a tomar los valores del sensor con respecto a esta dirección. Si no se ha realizado, se inicializan los valores para realizar el análisis.

```
Ecosensor Version: 03
Author: Frank Perez Paz
Identificador del Sensor: D4

0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;
0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;
0;0;0;0;0;0;0;0;0;0;_
```

Figura 89: Valores iniciales antes de la calibración

Una vez recogidas las muestras necesarias, se determinan los puntos de mayor cantidad de muestras y se deja un margen de 12° para capturar completamente las aceleraciones y que en aceleraciones en curvas rápidas pueda capturar los datos correctamente. También se ha de dejar este margen por el balanceo que tienen los vehículos al acelerar o frenar, puesto que al realizar alguna de estas acciones, se produce un movimiento de rotación del sensor que hay que tener en cuenta.

En la siguiente figura se observa cómo se ha completado el Array de muestras. Se puede ver como es muy similar al recogido por el dispositivo móvil por lo que se puede decir que ambos funcionan de manera similar. Al igual que en el dispositivo móvil, se ven los dos picos más pronunciados de mayor cantidad de muestras y otros más pequeños.

Figura 90: Muestras tomadas desde el Sensor

Si esta misma gráfica se representa de manera radial (tal como se realizó con el dispositivo móvil) se observa cómo las muestras que se repitieron una mayor cantidad de veces se encuentran en 2 y en 47. Esto significa que están a 2x4 = 8° y a 47x4 = 188°, es decir, 180 grados con lo que representan la parte frontal y posterior del vehículo.

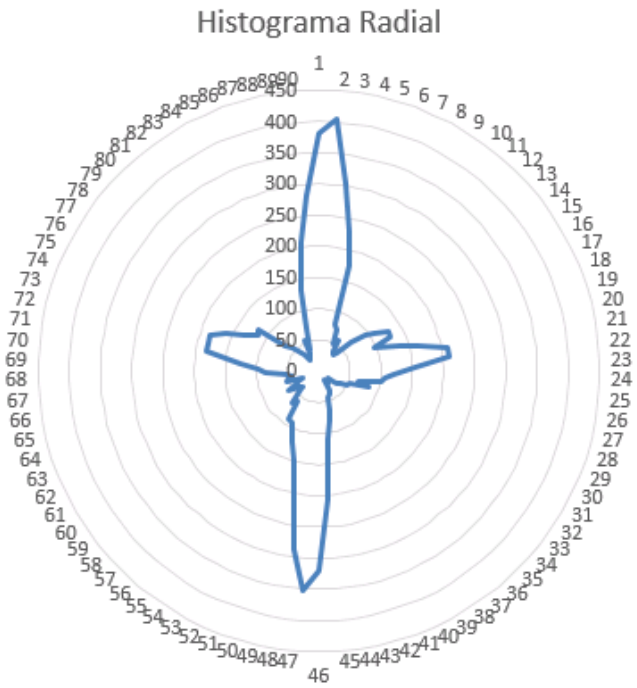


Figura 91: Muestras mostradas en gráfica radial

Para finalizar, se analizaron los datos extraídos de las aceleraciones una vez el sensor fue calibrado, y se observó la siguiente gráfica. Donde todas las aceleraciones sensor estuvo

calibrado para que aceptara las aceleraciones correctas. Las aceleraciones totales tomadas por el sensor se muestran en color azul y las aceleraciones reales en la dirección de desplazamiento, se muestran en color naranja. Se puede observar como muchas veces ambas gráficas coinciden, y como en otras ocasiones las aceleraciones reales quedan por debajo o no son detectadas debido a que no registran datos por estar fuera de los parámetros (dirección de desplazamiento en curva).

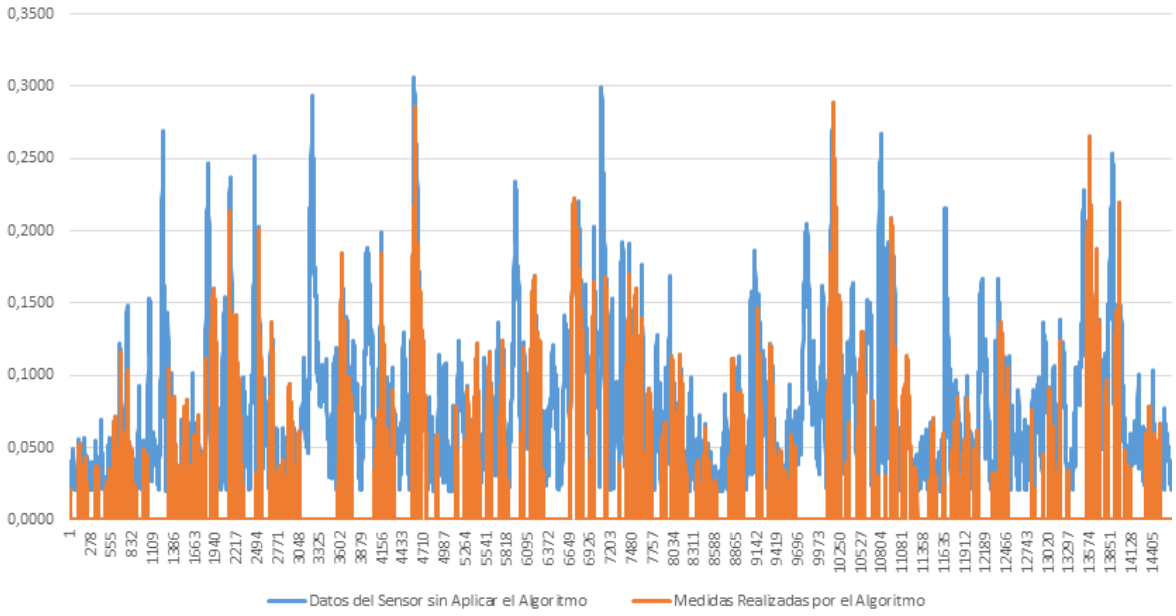


Figura 92: Gráfico de aceleraciones del sensor

Cuando se realizó la prueba para comprobar el funcionamiento del algoritmo y ver cuáles eran las sensaciones que producía, se apreció un retraso en la señal y como en ocasiones no media correctamente cuando tenía que hacerlo. La conclusión a la que se llegó fue que eran dos motivos los que podían producir estos efectos adversos.

El primer motivo y más claro es el retraso que produce el uso de los filtros. Los filtros utilizados son de orden cuatro y producen mucho retraso. El segundo motivo es también debido a los filtros y es que cuando se tiene una aceleración continua o un giro continuo en el tiempo, el filtro va sumando este valor hasta tomarlo como si fuera un valor constante (lo suma a la gravedad). Por estos motivos, las medidas realizadas no son del todo precisas.

6.5 Pruebas del 4º algoritmo de desarrollo

Al igual que en el tercer algoritmo, se probó su funcionamiento en el dispositivo móvil y en la placa de prueba con el microcontrolador. Los resultados obtenidos fueron bastantes

satisfactorios, en el dispositivo móvil se observó cómo al cabo de unos minutos recogiendo muestras encontró la dirección frontal del vehículo. En la placa PCB, se fue escribiendo por el puerto USART y también se observó cómo cambio correctamente a mostrar los datos de aceleración frontal cuando el vehículo aceleraba o frenaba (datos en positivo o negativos de la aceleración producida).

En el dispositivo móvil se insertó una gráfica radial en donde se dibujaban las fuerzas G que se producían en los 360°. También se colocaron unas gráficas de barras para indicar la inclinación a la que está sometido el sensor. Estos valores se inicializan al iniciar la aplicación del dispositivo.

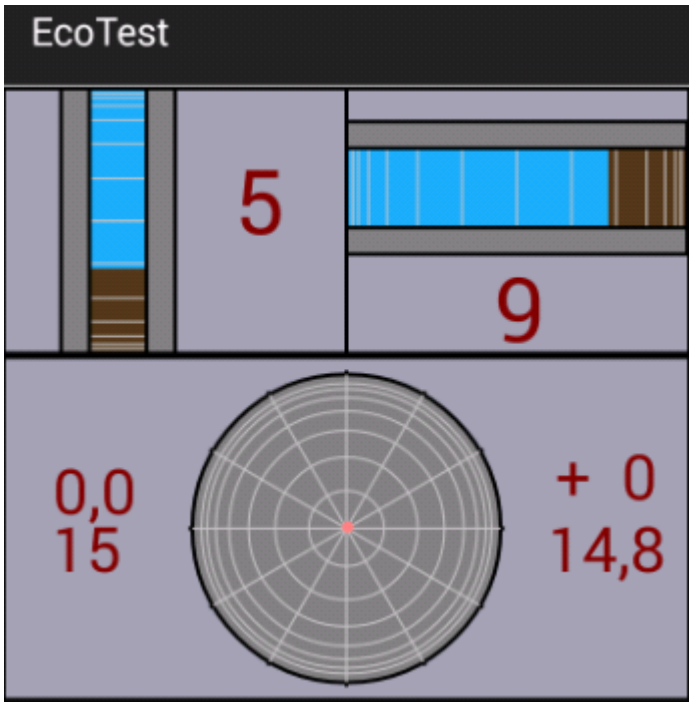


Figura 93: Gráficas mostradas por el dispositivo móvil

Los valores al iniciar la app se fijan al valor 0 para indicar que a partir de ese momento es cuando se cambian las inclinaciones y para tener una referencia de los valores cabeceo y el alabeo en ese instante y saber la rotación que tiene originalmente. En la figura anterior, se inclinó el sensor 5° de cabeceo y 9° de alabeo.

Una vez comprobado en el laboratorio que la app del dispositivo móvil funcionaba, se pasó a probarla en un vehículo para ver su funcionamiento. Antes de probarla en modo real, se le inserto además unas gráficas que mostraran los ejes de aceleración. Esta gráfica se implantó para comprobar que cuando el dispositivo adquiere unas muestras válidas,

aunque sigan indicando que las fuerzas están en un eje determinado, en la gráfica se identifique el frontal.

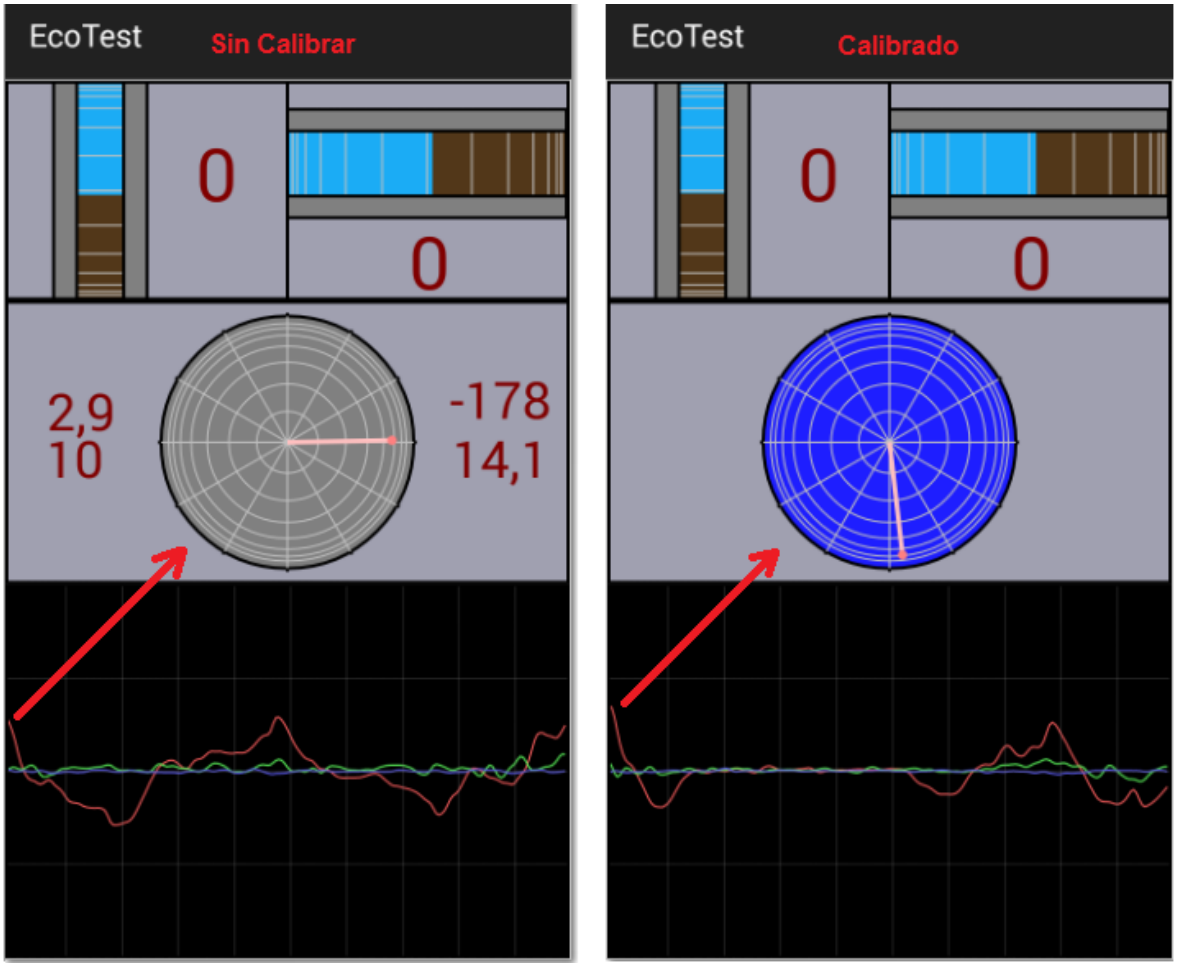


Figura 94: Vista gráfica aplicación móvil. Sin estar calibrado (imagen derecha) y una vez calibrado (imagen izquierda)

Como se puede apreciar en la figura anterior, cuando le entra la misma señal del sensor (línea roja), el dispositivo reacciona de distinta manera una vez se encuentra calibrado. Esto es por causa de la corrección que hace el algoritmo al identificar la parte frontal y lateral del vehículo. Al probarse en el dispositivo móvil y ver los buenos resultados que ofrecía, se prosiguió con las pruebas en el microcontrolador.

Para programar el micro, se pensó en dejar funcionando el tercer algoritmo y comparar los resultados con el cuarto algoritmo para ver si los datos eran semejantes y cuál de ellos era mejor. Se empezó a desarrollar el cuarto algoritmo teniendo en cuenta los cambios que se hacían y se fijó la frecuencia de captura de datos hasta los 60 Hz. Esta frecuencia se estableció así para tener mayor cantidad de datos que procesar y seguir el

movimiento del vehículo eficientemente. Es muy importante este seguimiento debido a que se va a estar continuamente mirando los ángulos de inclinación del giroscopio para saber la dirección del vector de gravedad.

Al programar ambos algoritmos al mismo tiempo, se produjo un problema con el microcontrolador. Este problema fue causado por la memoria de programación flash reprogramable. Debido a que el microcontrolador que se está utilizando contiene una memoria de 16 Kbytes, con las variables que se utilizan y el sistema de filtrado que implementa el tercer algoritmo, se sobrepasa la capacidad de esta memoria como se observa en la siguiente figura.

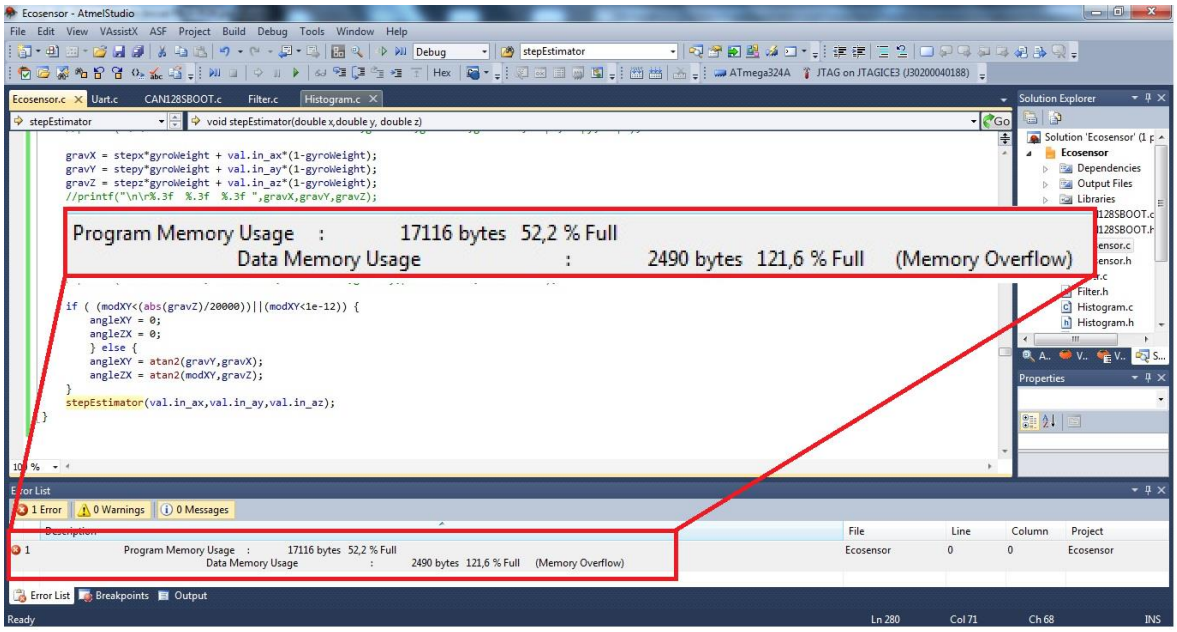


Figura 95: Desbordamiento de la memoria de programación

Al producirse este hecho, se tuvo que eliminar el tercer algoritmo y solo utilizar el último desarrollado. Al programar el microcontrolador se producía un problema en la escritura por el puerto USART y se reiniciaba el micro. Este problema fue ocasionado por la frecuencia para capturar los datos era alta y al tener una mayor cantidad de información a mostrar, no daba tiempo a escribir una cadena de texto completa cuando ya se tenía que escribir otra (mostrando los arrays).

Para solucionar este problema, se utilizó otro Timer/Counter que funcionara como un retardo. Este retardo se utilizaría para que en cada escritura, por el puerto serie, de las cadenas de datos. Un ejemplo son los arrays de las sumas de las muestras capturadas y de los acumuladores de los valores producidos. Al escribir estas cadenas continuas, se utilizó

el retardo para parar la captura de datos del sensor y retener un poco la velocidad de escritura en la USART.

Para ver el funcionamiento del algoritmo se decidió mostrar por el hyperTerminal las siguientes variables: módulo de la aceleración XY, ángulo de esta aceleración, donde se encuentra el punto máximo del acumulador de giros y las fuerzas frontales y laterales. A continuación se muestra una imagen como se muestran los datos.

Ecosensor Version: 05
Author: Frank Perez Paz
Identificador del Sensor: D4
[Calibrando Sensor Giroscopico]
Gravity: 0,963 Pitch:6,098 Roll:-168,697 zSet:2

compMod	compAngle*180/PI	maxIx*2	frontAcc	latAcc
0,11	-84,4	0	0,01	-0,11
0,06	-61,9	136	0,03	-0,05
0,11	100,1	148	-0,02	0,11
0,09	95,4	50	-0,01	0,09
0,05	107,5	50	-0,02	0,05
0,07	-85,7	50	0	-0,07
0,08	-84,3	50	0,01	-0,08
0,06	-87,7	50	0	-0,06
0,05	-75,9	50	0,01	-0,05
0,06	-64,9	50	0,02	-0,05
0,05	-56,3	146	0,03	-0,04
0,05	-60,2	150	0,03	-0,04
0,06	-83,1	148	0,01	-0,06
0,06	-101,6	138	-0,01	-0,06
0,05	129,5	128	-0,03	0,04
0,06	136,9	128	-0,04	0,04

Figura 96: Muestra de la información al inicializar el sensor

Como se puede ver en la figura 92, al inicializar el microcontrolador, este empieza a tomar datos del sensor y realiza una media de los datos para saber el offset que tiene y así poder realizar el proceso de calibración. Cuando el sensor esta calibrado (hay que hacerlo con el vehículo sin movimiento) se realiza una medida de la posición inicial en la que está colocado el sensor. Esta posición se indica en pantalla con los valores de gravedad inicial, el cabeceo y el alabeo. Además se muestra el valor de zSet que contiene un valor del 0 al 2 en el que indica la posición del eje que contiene la mayor cantidad de gravedad.

Los valores de las muestras del ángulo (compAngle) van de 0° a +180° y de -180° a 0° haciendo los 360° de la circunferencia, mientras que los valores del acumulador van de 0° a 90° que son los índices que se tienen de dividir los 360°/4 = 90 valores de índices. Este

índice indica la mayor acumulación de valores y depende en la mayor medida del giroscopio. Con esto quiere decir que no por tener un mayor módulo de aceleración, va tener que coincidir con el índice máximo. Si se observa la figura 64 en donde se explicaba un ejemplo de cómo se realiza la acumulación de valores y a que corresponde el valor máximo, se puede entender mejor.

Como las muestras iniciales no son válidas hasta que se haya realizado un recorrido suficiente para que el algoritmo pueda determinar la parte delantera y lateral del vehículo, se escribieron todos los datos. Cuando el sensor obtuvo 2000 muestras (suficientes para una primera estimación, aunque entre mayor número de muestras mayor precisión), se compraron los resultados de una misma aceleración anterior para ver si se estaba detectando y corrigiendo la orientación del vehículo.

Datos tomados durante la calibración del algoritmo

compMod	compAngle*180/PI	maxIx*2	frontAcc	latAcc
0,07	86,4	100	0	0,07
0,09	70,5	100	0,03	0,09
0,12	68,5	100	0,04	0,11
0,1	86,1	100	0,01	0,1
0,09	57,5	100	0,05	0,07
0,11	54,5	100	0,07	0,09
0,16	76,6	100	0,04	0,16
0,18	82,2	100	0,02	0,18
0,11	86,6	100	0,01	0,11
0,15	88,4	100	0	0,15
0,21	84,7	100	0,02	0,21
0,07	78,4	100	0,01	0,07
0,13	91,3	100	0	0,13
0,12	70,5	100	0,04	0,11
0,12	72,6	100	0,03	0,11
0,15	92,3	100	-0,01	0,15
0,09	83,1	100	0,01	0,09

Datos tomados con el algoritmo funcionando

compMod	compAngle*180/PI	maxIx*2	frontAcc	latAcc
0,2	-80,6	176	-0,18	-0,08
0,2	-85,2	176	-0,19	-0,06
0,18	-90,7	176	-0,18	-0,04
0,19	-92,5	176	-0,18	-0,04
0,2	-92,8	176	-0,2	-0,04
0,2	-98,6	176	-0,2	-0,02
0,18	-97	176	-0,18	-0,02
0,18	-101	176	-0,18	-0,01
0,19	-95,7	176	-0,19	-0,03
0,18	-92,8	176	-0,18	-0,04
0,18	-90,3	176	-0,18	-0,04
0,19	-94,4	176	-0,19	-0,03
0,2	-89,8	176	-0,19	-0,05
0,17	-87,7	176	-0,16	-0,05
0,18	-81,4	176	-0,17	-0,07
0,18	-82,2	176	-0,17	-0,07
0,16	-89,6	176	-0,16	-0,04

Figura 97: Datos tomados con el algoritmo en proceso de calibración (izq.) y una vez completadas las muestras y empezando a funcionar correctamente (dcha.)

Como puede verse en la figura anterior, los módulos tomados en un determinado ángulo (por ejemplo a 80°), son descompuestos en las fuerzas frontales y laterales. En la primera parte, en donde el algoritmo todavía no ha tomado las suficientes muestras como para poder decir que funciona correctamente, está indicando que las aceleraciones que se están produciendo son aceleraciones laterales. Una vez el algoritmo tiene suficientes datos, corrige este mal funcionamiento he indica que para este ángulo de 80°, el mayor valor de la descomposición lo muestra en la aceleración frontal y no en la lateral.

Con esta muestra se comprueba que en el dispositivo microcontrolador está funcionando de igual manera que en el dispositivo móvil. Para ver como se han ido

tomando las muestras y comprobar donde se encuentra el punto máximo del acumulador, se escribió por pantalla todo el Array de muestras y el resultado se muestra gráficamente a continuación.

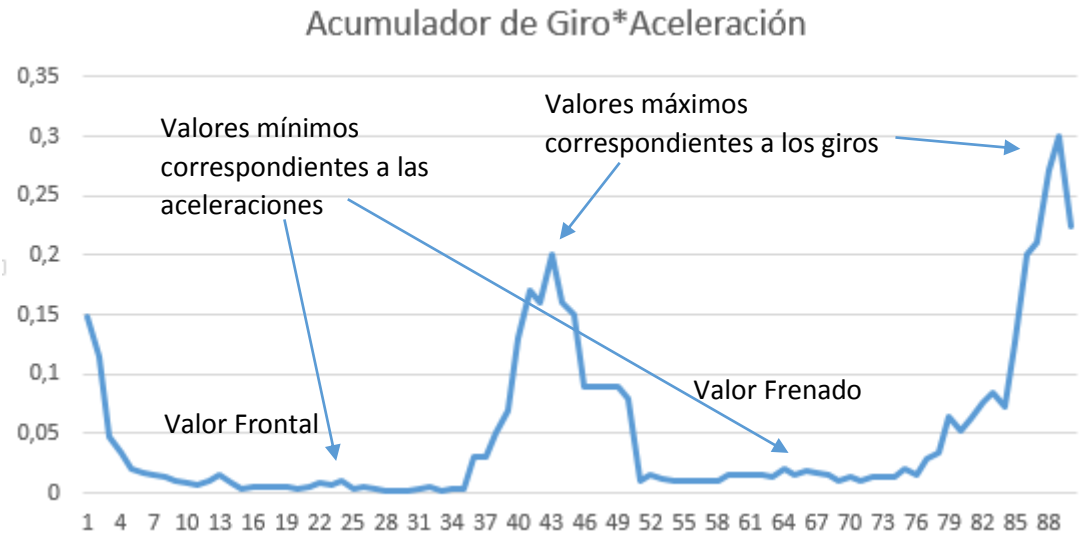


*Figura 98: Array de las muestras de aceleración*giro*

Se aprecia que el pico máximo está en el índice 88 y por tanto esto quiere decir que se encuentra a 88*4=352° o a -8° que es más fácil de comprender. Si se comprueba cómo estaba posicionado el sensor en el vehículo, se aprecia que estos -8° son indicativo del movimiento lateral, por lo que el frontal del vehículo se encuentra a 82° y la parte trasera a -98°.

Por esta circunstancia, se aprecia que en la figura 94 con el algoritmo funcionando correctamente, el valor de -90° aproximadamente (en el array está en la muestra 67*4=268°=-90°) corresponde a una aceleración frontal negativa, es decir, a una frenada. Y como con una aceleración de 80° en el algoritmo sin calibrar la tomaba como una aceleración lateral. En la siguiente figura, se observa la posición del sensor en el vehículo y una explicación de cómo están los ángulos.

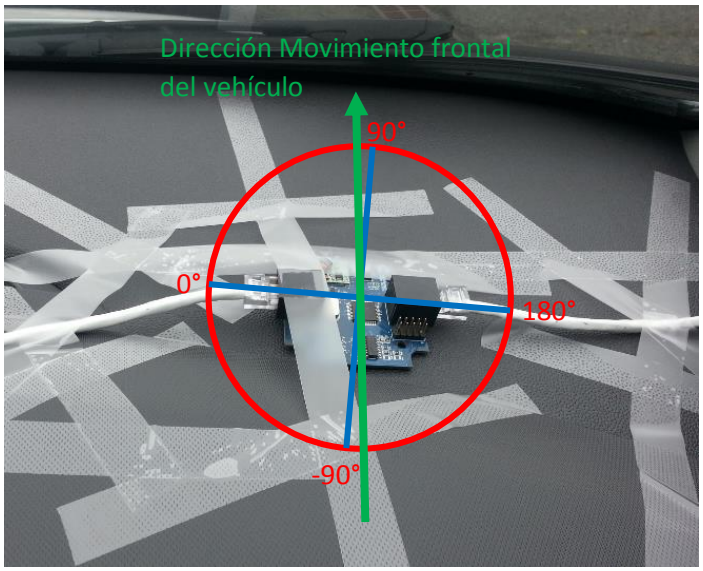


Figura 99: Sensor Fijado en el Vehículo

Para finalizar el algoritmo se implementó además, que una vez calibrado completamente el algoritmo, en vez de mostrar los datos mostrados hasta el momento (son datos para comprobar el funcionamiento), lo que hacía es mostrar los datos que interesaban desde el punto de vista de eficiencia en la conducción. Por este motivo, se mostraba por el hyperTerminal los valores del tiempo de aceleración cuando superaba el umbral de aceleración frontal (acelerando o frenando), el pico máximo de aceleración y las muestras de las aceleraciones producidas durante ese tiempo.

Tiempo	Pico Máximo	Acumulado
T1,25	Fmx0,19	Acc11,379
T0,02	Fmx0,10	Acc0,101
T0,22	Fmx0,22	Acc1,935
T0,02	Fmx0,09	Acc0,086
T0,02	Fmx0,09	Acc0,087
T0,02	Fmx0,09	Acc0,092
T0,15	Fmx0,16	Acc1,130
T0,07	Fmx0,12	Acc0,383
T0,25	Fmx0,13	Acc1,591
T0,02	Fmx0,08	Acc0,075
T0,03	Fmx0,09	Acc0,168
T0,02	Fmx0,08	Acc0,078
T0,02	Fmx0,08	Acc0,078
T0,02	Fmx0,08	Acc0,082
T1,65	Fmx0,35	Acc18,414

Figura 100: Datos de tiempo de aceleración continua, valor máximo y acumulado

En la figura anterior se puede ver cómo y cuándo se produce una aceleración que supere el umbral de 0.075g (de 0km a 100km (27.77 m/s) en 37 segundos con una aceleración de 0.750 m/s^2). El tiempo que esta aceleración está por encima de ese umbral y los valores que se van acumulando durante ese instante. Se eligió ese valor umbral debido a que para aceleraciones inferiores no interesa el dato en lo referente a eficiencia. Si el valor por encima de este umbral supera de manera continua dicho umbral, quiere decir que se está produciendo una aceleración considerable.

Con las pruebes realizadas, cabe decir que este algoritmo realizado funciona adecuadamente y que cumple con las expectativas creadas. Ya no se aprecia el retardo en la toma de valores que se producía en el algoritmo número 3. Además con el seguimiento del vector gravedad, se ha conseguido que las medidas realizadas sean mucho más reales, es decir, con una mayor precisión y exactitud.

Además al no utilizar el sistema de filtrado anterior, sino utilizar este nuevo tipo de filtrado dependiente del giroscopio, se corrige los fallos en las mediciones de este. Esto es debido a que cuando el vehículo se detiene, el offset acumulado por la pérdida de las medidas de rotación que no se han tomado se eliminan, quedando el giroscopio calibrado nuevamente. Este offset que se produce en el giroscopio es imposible de eliminar debido a que por mucho que se suba la frecuencia de muestreo, siempre dejara de tomar algunas medidas, las cuales se van acumulando y causando este problema.

7.0 Conclusiones y Líneas Futuras

Con la realización de este Trabajo Fin de Máster se ha logrado desarrollar teóricamente, realizar y poner a prueba los diferentes algoritmos de programación para conseguir medir de manera eficiente las aceleraciones de los vehículos. Todo se ha conseguido utilizando únicamente un microprocesador y sensor que contiene un acelerómetro y un giróscopo. El sistema diseñado es capaz de medir cada uno de los movimientos recibidos, procesarlos y transmitirlos por el puerto serie disponible, además de contar con la posibilidad de almacenar los valores para un procesado posterior.

Para desarrollar los algoritmos, se partió desde cero y con los métodos más sencillos de aplicar para ver como reaccionaban a la hora de medir las fuerzas de aceleración. A medida que se fueron probando y localizando el motivo por el cual no funcionaba o no era lo suficientemente bueno, se fue mejorando hasta lograr un algoritmo que cumplirá con las expectativas marcadas. El proceso de mejora continua de un algoritmo a otro, al mismo tiempo que se continuaban utilizando los métodos que si funcionaban correctamente, supuso una mejora progresiva a la vez que un aprovechamiento del código.

Con el algoritmo final, se ha conseguido que el dispositivo pueda ser fijado en cualquier posición, sin necesidad de estar alineado o calibrado al acoplarlo, y en cualquier vehículo, tal y como se exigía para desarrollar el proyecto. El dispositivo al iniciarse, es capaz de calibrarse y a partir de su posición inicial, tomar los datos del movimiento. En este último algoritmo, la calibración para saber la localización de las partes del vehículo se realiza en pocos minutos, tan solo necesita recorrer unos pocos cientos de metros en donde se encuentre una carretera con curvas o una rotonda para una primera calibración (aunque para una calibración eficaz es recomendable recorrer una mayor distancia).

Por tanto, se puede considerar que este Proyecto Fin de Máster ha cumplido con creces los principales objetivos iniciales establecidos, al lograr la correcta funcionalidad del sistema, desarrollar teóricamente los algoritmos y realizar las pruebas sobre el terreno.

Con todo esto, se alcanzó un prototipo que logra medir con bastante exactitud las aceleraciones y frenadas, para saber la eficiencia en la conducción, así como los movimientos laterales para saber la confortabilidad de los pasajeros.

Como el Trabajo Fin de Máster está acotado en tiempo, quedaron algunos detalles en los que todavía faltan algunas mejoras. Los detalles a modificar y perfeccionar son:

- La muestra de información al usuario.
- Frecuencia de toma de datos más elevada.
- Mostrar los datos de varios algoritmos al simultáneamente.

Para mejorar la información que se envía por el terminal serie, solo hay que ver qué información mostrar y cuál es la más útil para el posterior post-procesado de los datos. En este caso, se mostró el tiempo durante el cual la aceleración era superior a un valor mínimo. También se transmitió el pico máximo de aceleración que alcanzaba ese movimiento y la integración de las aceleraciones tomadas durante ese periodo. Siempre se puede incrementar la información a mostrar.

En cuanto a mejorar la toma de información, se intentó con este microprocesador y lo máximo a lo que se pudo llegar fue a 62 Hz. A partir de esa frecuencia, cada vez que se escribía por pantalla había que parar la captura de datos, sino, empezaban a aparecer errores en la muestra de la información.

En lo referente a utilizar no solo el último algoritmo realizado, sino utilizar además algún algoritmo anterior para realizar una comparación, se vio limitado por la capacidad de la memoria de programación del microcontrolador. Por lo tanto, para mejorar la frecuencia y mostrar varios algoritmos, es necesario un microprocesador de con mejores características de frecuencia y capacidad.

Bibliografía

[1] Conducción eficiente de vehículos industriales. Revisada el 12/07/2015. http://www.idae.es/uploads/documentos/documentos_10320_Conduccion_eficiente_veh_industriales_A2011_A_982a7098.pdf

[2] Pololu, Robotics and Electronics. Revisada el 12/07/2015. http://www.pololu.com/category/80/accelerometers-gyros-compasses

[3] iNEMO inertial module: 3D accelerometer and 3D gyroscope. Revisada el 12/07/2015. http://www.st.com/web/catalog/sense_power/FM89/SC1448/PF253882

[4] ChRobotics: Understanding Euler Angles. Revisada el 12/07/2015. http://www.chrobotics.com/library/understanding-euler-angles

[5] Understanding Quaternions. Revisada el 12/07/2015. http://3dgep.com/understanding-quaternions/

[6] Fundamentals of Inertial Navigation, Satellite-based Positioning and their Integration, Aboelmagd Noureldin - Tashfeen B. Karamat - Jacques Georgy, ISBN 978-3-642-30465-1, 2013, Páginas 125 – 200.

[7] Tratamiento de Señales en Tiempo Discreto, 2ª Edición. Alan V. Oppenheim, Ronald W. Schafer, John R. Buck. ISBN: 978-84-205-2987-5. Editorial: Pearson Educación.

[8] Chapter 16: Active Filter Design Techniques. Texas Instruments. Revisada el 12/07/2015. http://www.ti.com/lit/ml/sloa088/sloa088.pdf

[9] Historias de Matemática Hamilton y el descubrimiento de los Cuaterniones, José Manuel Sánchez Muñoz, ISSN 2174-0410, 1 de octubre de 2011. Revisada el 12/07/2015. http://www2.caminos.upm.es/Departamentos/matematicas/revistapm/revista_impresa/numero_1/hamilton_y_el_descubrimiento_de_los_cuaterniones.pdf

[10] Introduction into quaternions for spacecraft attitude representation, Dipl. -Ing. Karsten Groÿekatthöfer, Dr. -Ing. Zizung Yoon Technical University of Berlin Department

of Astronautics and Aeronautics Berlin, Germany, May 31,2012. Revisada el 12/07/2015 http://www.tu-berlin.de/fileadmin/fg169/miscellaneous/Quaternions.pdf

[11] The source-code.biz Java DSP collection. Revisada el 12/07/2015 http://www.source-code.biz/dsp/java/

Presupuesto

La cuantía del trabajo elaborado se ha fijado según las indicaciones de contrataciones de la Universidad de las Palmas de Gran Canaria. De este modo, el total del presupuesto se ha desglosado en las siguientes secciones, en las que se han separado los distintos costes asociados al desarrollo del Trabajo Fin de Máster, según su naturaleza.

- Coste de los Recursos Humanos
- Coste de los Recursos Hardware
- Coste de los Recursos Software
- Coste de Edición de la Memoria del Proyecto
- Coste Total

Coste de los Recursos Humanos

El coste de recursos humanos está asociado al tiempo empleado por un ingeniero en la realización del proyecto, sin contar el tiempo de aprendizaje de las herramientas utilizadas. De acuerdo con las últimas tablas de honorarios publicadas el 4 de noviembre de 2010 por la Universidad de las Palmas de Gran Canaria, lo que debe ganar un Licenciado, Arquitecto o Ingeniero con una dedicación de 20 horas semanales, asciende a 1.032,63 €:

Para la realización del presente Trabajo Fin de Máster, se estima que se ha trabajado durante 15 semanas hábiles, trabajando 5 días semanales y 4 horas efectivas diarias, por lo que el número de horas totales trabajadas resulta:

Horas totales = 4 horas/día × 5 días/semana × 15 semanas = 300 horas

El valor final al que ascienden estos honorarios es de:

Honorarios (Coste RRHH) = 12,91 €/h × 300 h = 3.873,00 €

Tabla 5: Costes de los Recursos Humanos

Concepto	Tiempo Empleado	Coste Mensual Aproximado	Importe
Ingeniero en Telecomunicación	15 semanas	1.032,63 €	3.873,00 €
		Coste Total	3.873,00 €

Por tanto, el trabajo tarifado por tiempo empleado asciende a la cantidad de 3.873 €.

Coste de los Recursos Hardware

Para elaborar la memoria así como realizar la programación y las pruebas, se ha utilizado los equipos de la tabla 6. Con los consiguientes costes de los materiales.

Tabla 6: Coste Recursos Hardware

Concepto	Coste
Ordenador Toshiba Satellite-s50-b-131	699 €
Samsung Galaxy S3	150 €
Programador JTAGICE3	130 €
Adaptador USB - Serie	8,50 €
Total	987,50 €

Coste de Recursos Software

El coste de los recursos software se obtiene a partir del valor de las licencias y el mantenimiento de cada uno de los programas utilizados. En la tabla siguiente se resumen estos costes.

Tabla 7: Coste Recursos Software

Concepto	Coste/mes	Coste anual
Altium Designer	195 €	2.340,00 €
Paquete de Microsoft Office 365	8,8 €	105,6 €
Project Pro para Office 365	17,2 €	206,4 €
AvrStudio	0 €	0 €
Coste Total		2.652 €

Coste Edición de la Memoria del Proyecto

Los costes de redacción del proyecto contemplan los gastos generados para la impresión y entrega de la memoria del Trabajo Fin de Máster. Estos gastos se incluyen en la siguiente tabla:

Tabla 8: Coste Edición de la Memoria

Concepto	Coste
Impresión de la Memoria	30 €
Encuadernación	5 €
3xCD	3 €
Total	38 €

Coste Total

En la tabla 9 se recopila el conjunto de los gastos contemplados obteniéndose el importe final.

Tabla 9: Coste Total del Trabajo Fin de Máster

Concepto	Coste
Recursos Humanos	3.873,00 €
Recursos Hardware	987,50 €
Recursos Software	2.652 €
Coste de Edición de la Memoria	38 €
Subtotal	7.550,5 €
IGIC (7%)	528,53 €
Total	8.079,03 €

El presupuesto total del Proyecto de Fin de Carrera asciende a un total de siete mil novecientos dieciocho con cincuenta y tres euros (7.918,53 €).

Frank Pérez Paz declara/certifica que el proyecto “Prototipo de una unidad de apoyo a la conducción eficiente utilizando un sensor inercial”, realizado en calidad de Trabajo Fin de Máster, tiene un coste total de 7.918,53 €, correspondiente a la suma de las cantidades consignadas en los apartados considerados a continuación.

Tabla 10: Coste Total

Concepto	Importe
Recursos Humanos	3.873,00 €
Recursos Hardware	837,50 €
Recursos Software	2.652 €
Coste de Redacción de la Memoria	38 €
Subtotal	7.400,5 €
IGIC (7%)	518,03 €
Presupuesto Total	7.918,53 €

Las Palmas de Gran Canaria

Julio 2015