ESCUELA TÉCNICA SUPERIOR DE INGENIEROS INDUSTRIALES Y DE TELECOMUNICACIÓN

UNIVERSIDAD DE CANTABRIA

Trabajo Fin de Grado

DISEÑO E IMPLEMENTACIÓN DE UNA BOYA OCEANOGRÁFICA PARA LA PREDICCIÓN DEL OLEAJE

(Design and implementation of an oceanographic buoy for surf forecast)

Para acceder al Título de

Graduado en Ingeniería de Tecnologías de Telecomunicación

Autor: Pablo Novo Cañizares

Septiembre - 2018

GRADUADO EN INGENIERÍA DE TECNOLOGÍAS DE TELECOMUNICACIÓN

CALIFICACIÓN DEL TRABAJO FIN DE GRADO

Realizado por: Pablo Novo Cañizares
Director del TFG: Luis Valle López
Título: "Diseño e implementación de una boya oceanográfica para la predicción del oleaje"
Title: "Design and implementation of an oceanographic buoy for surf forecast"

Presentado a examen el día:

para acceder al Título de

GRADUADO EN INGENIERÍA DE TECNOLOGÍAS DE TELECOMUNICACIÓN

Composición del Tribunal:
Presidente (Apellidos, Nombre):
Secretario (Apellidos, Nombre):
Vocal (Apellidos, Nombre):

Este Tribunal ha resuelto otorgar la calificación de: .....

Fdo.: El Presidente	Fdo.: El Secretario
Fdo.: El Vocal	Fdo.: El Director del TFG (sólo si es distinto del Secretario)
Vº Bº del Subdirector	Trabajo Fin de Grado Nº (a asignar por Secretaría)

Resumen

El presente trabajo describe el desarrollo e implementación de varios sensores en una boya oceanográfica que permitan la toma de medidas de algunas magnitudes físicas, y en base a ellas predecir el momento y comportamiento del siguiente tren de olas.

El dispositivo contará con un acelerómetro-giroscopio, un barómetro, GPS y un módulo LoRa. Todos estos sensores serán gestionados con un microcontrolador Arduino UNO, que se encargará de procesar todos los datos recogidos por los sensores y enviarlos a través el modulo LoRa.

Para la adquisición y transmisión de datos en la boya, se utilizara el software nativo de Arduino, y para el procesado de los datos en la estación de tierra, se utilizará el software Matlab, que permitirá realizar numerosas operaciones al mismo tiempo.

Abstract

This project describes the developement and implementation of various sensors into an oceanographic buoy which aims to measure the surge.

The device will count with an accelerometer-gyroscope, a barometer, GPS and one LoRa module. All of these sensors will be managed by an Arduino UNO microcontroller, wich will take care of processing all of data collected by the sensors, and they will be sent through the LoRa module.

For the acquisition and transmission of data in the buoy, the native Arduino software will be used, and for the processing of the data in the ground station, the Matlab software will be used, which will allow numerous operations at the same time.

Índice general

Índice de figuras

Acrónimos y simbología

IMU	unidad de medición inercial
hPa	hecto Pascales
Ax	aceleración en el eje X
Ay	aceleración en el eje Y
Az	aceleración en el eje Z
Gx	velocidad angular sobre el eje X
Gy	velocidad angular sobre el eje Y
FPB	filtro paso bajo
FPA	filto paso alto
fc	frecuencia de corte
E/S	entrada y salida
PCB	placa de circuito impreso
CPU	Unidad Central de Proceso

1. Motivación

Hace aproximadamente un año, el realizador de eventos y torneos de surf para la BBC, me planteó el diseño de algún tipo de dispositivo capaz de anticiparse al momento preciso en el que comenzará una serie de olas y cuál sería su tamaño, con un margen de error no demasiado alto.

Los torneos de surf se organizan en mangas, lo que significa que, durante aproximadamente 20 minutos, hay una tanda de 4 surferos en el agua, éstos cogen olas sin parar, y les puntúan las mejores olas. La publicidad en los torneos se ponen de forma preprogramada, por lo tanto el espectador puede estar viendo anuncios justo cuando haya un buen tren de olas, y viceversa.

El propósto final, sería diseñar un sistema capaz de predecir el oleaje con 10 o 15 minutos de antelación. Así se podrían organizar de una forma mucho más eficiente las mangas, los cortes publicitarios, etc…

Además, la creación de un sistema boyas transportables cuya funcionalidad fuese predecir el oleaje en ocasiones puntuales, podría tener muchas más aplicaciones, como la ayuda para pequeñas embarcaciones al salir de puerto, cuando hay mar agitada.

2. Fundamentos teóricos

2.1 Propagación de las olas

Las olas marinas son generalmente un fenómeno eólico, es decir, que se forman y mueven en dependencia de la intensidad del viento, aunque también existen las olas de marea o mar de fondo generadas por terremotos o grandes movimientos telúricos en el océano. [1]

Una ola regular de viento se crea por el roce del viento sobre la superficie, luego ésta pequeña ola retorna parte de su energía al mar contribuyendo a formar olas de mayor tamaño. Estas olas crecen en forma proporcional a la fuerza del viento, por lo que, más viento, más fortaleza y altura de la ola.

Cuando las olas vienen en forma periódica y estable se les llama un tren de olas. La altura de una ola se determina por la distancia entre la cresta y el valle.

Para este proyecto, se tratarán las olas del mar como movimiento armónico simple (M.A.S) [2], en la figura se pueden observar los parámetros más importantes de una ola.

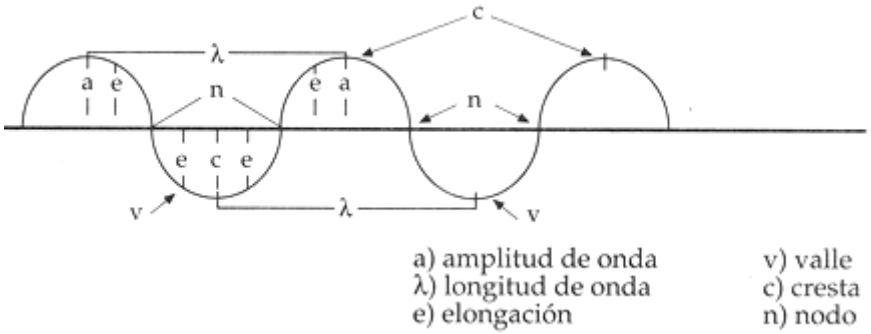

Figura 2.1: Parámetros de una ola

Interesa conocer la velocidad de propagación de la ola, que se calcula:

$$c = \frac{\lambda}{T} = \frac{gT}{2\pi} Ta \frac{2\pi}{\lambda} \qquad (2.1)$$

Aguas profundas es cuando $a > \frac{\lambda}{2}$, por tanto:

$$\lambda = \frac{gT^2}{2\pi} \tag{2.2}$$

$$T = \frac{2\pi v}{g} \tag{2.3}$$

$$v = \frac{\lambda}{T} = \sqrt{\frac{g\lambda}{2\pi}} = f(\lambda) \tag{2.4}$$

Al final se obtiene una ecuación de la velocidad que depende de la longitud de onda y de la aceleración de la gravedad. [3]

2.2 Sistema GPS

El Sistema de Posicionamiento Global (GPS) es un servicio propiedad de los EE.UU. que proporciona a los usuarios información sobre posicionamiento, navegación y cronometría. Está considerado como un sistema de navegación casi ideal que ha desplazado la mayor parte de los sistemas de radionavegación que proliferaron tras la Segunda Guerra Mundial.

El Departamento de Defensa de los EEUU desarrolló el proyecto en 1973. El primer satélite fue lanzado en 1978, aunque la fase de producción no comenzó hasta 1985 y sólo en 1992 el número de satélites en órbita permitió la utilización operacional del sistema. En 1994 se lanzó el último satélite. [4]

La constelación completa de GPS está formada por 24 satélites (21 satélites operativos y 3 de reserva) en seis planos orbitales. Los satélites orbitan alrededor de la Tierra con un periodo de 12 horas con trayectorias aproximadamente circulares a 20180 Km de altura (operar a tal altitud permite a las señales cubrir un área muy grande) y con una inclinación de 55º respecto al Ecuador. Cada satélite pasa sobre el mismo emplazamiento alrededor de dos veces por día (viajando a aproximadamente 7000 millas por hora). Los satélites están posicionados de forma que en cualquier punto de la Tierra se pueden recibir señales de 6 de ellos cerca del 100% del tiempo.

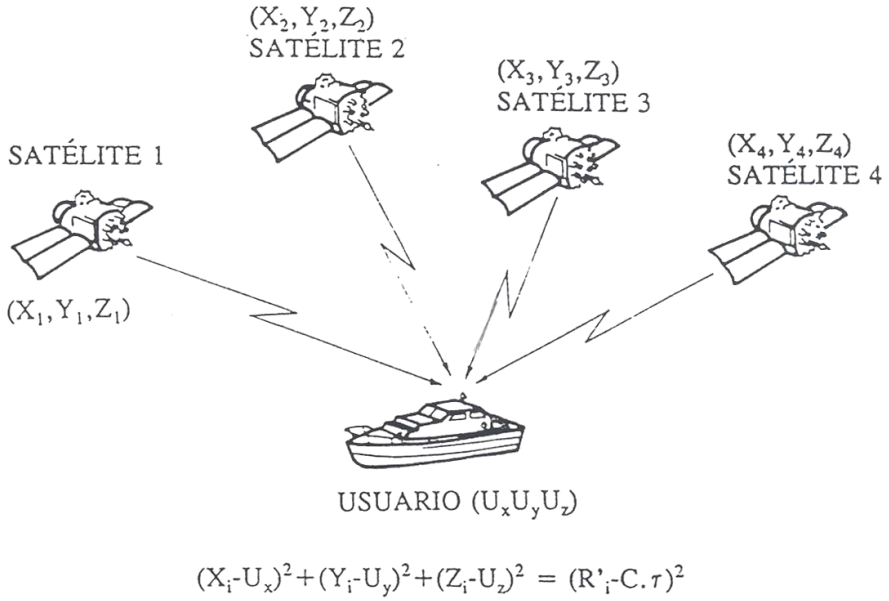

Figura 2.2: Medida de la distancia basada en la medida de retardos temporales

Cuando se desea determinar la posición tridimensional, el receptor que se utiliza para ello localiza automáticamente como mínimo cuatro satélites de la red, de los que recibe unas señales indicando la identificación y hora del reloj de cada uno de ellos, además de información sobre la constelación. Con base en estas señales, el aparato sincroniza el reloj del GPS y calcula el tiempo que tardan en llegar las señales al equipo, y de tal modo mide la distancia al satélite mediante el método de trilateración inversa, el cual se basa en determinar la distancia de cada satélite al punto de medición. Conocidas las distancias, se determina fácilmente la propia posición relativa respecto a los satélites. Conociendo además las coordenadas o posición de cada uno de ellos por la señal que emiten, se obtiene la posición absoluta o coordenadas reales del punto de medición. Los satélites transmiten señales radio de muy poca potencia del orden de 20-50 W en varias frecuencias. [5]

La precisión con que puede determinarse la posición depende de la exactitud con que sean determinadas las pseudodistancias y de la geometría que en dicho momento tengan los satélites.

2.2.1 Distancia entre coordenadas geográficas

Para calcular la distancia entre dos puntos de los que se conocen sus coordenadas para ello utilizaré la fórmula del Haversine. [6]

$$R = radio\ de\ la\ Tierra$$

$$\Delta lat = lat2 - lat1$$

$$\Delta long = long2 - long1$$

$$a = sin^2(\Delta lat/2) + cos(lat1) \cdot cos(lat2) \cdot sin^2(\Delta long/2) \quad (2.5)$$

$$c = 2 \cdot \text{atan2}(\sqrt{a}, \sqrt{(1-a)}) \quad (2.6)$$

$$d = R \cdot c \quad (2.7)$$

A pesar de que la fórmula del Haversine es de las más utilizadas para el cálculo de distancias entre dos puntos, la fórmula asume que la Tierra es completamente redonda por lo que cabe esperar una pequeña tasa de error.

2.3 Fórmula barométrica

Para este proyecto, se utilizará un sensor de presión atmosférica, que ayudará a estimar la altura relativa respecto a la de reposo, para ello se implementará la librería de Arduino "SFE_BMP180.h" que incluye la fórmula barométrica [7].

La altura tiene una relación inversa con la altura, es decir, cuanto más ascendemos, menos aire queda encima y el peso del aire es menor, por lo tanto un punto alto soporta menos presión que uno bajo.

A nivel del mar, el valor de la presión es de 1.013 mbar. En este proyecto solo será necesario conocer la diferencia de presión entre dos puntos para estimar la altura desplazada.

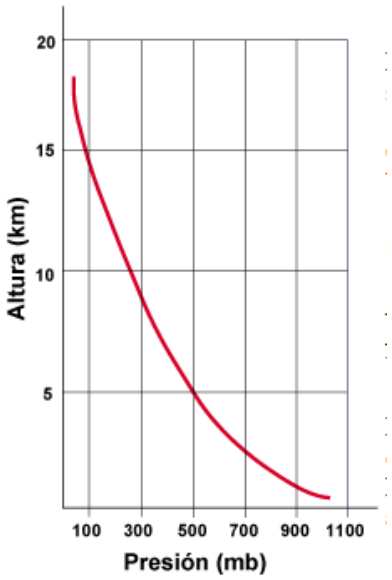

Figura 2.3: Gráfica altura respecto a la presión

2.4 Sistema de navegación inercial

Una unidad de medición inercial o IMU (del inglés inertial measurement unit), es un dispositivo electrónico que mide e informa acerca de la velocidad, orientación y fuerzas gravitacionales de un aparato, usando una combinación de acelerómetros y giroscopios. [8]

Los acelerómetros están colocados de tal forma que sus ejes de medición son ortogonales entre sí. Éstos miden la aceleración inercial en unidades de fuerza tt (fuerzas basadas en la aceleración producida por la gravedad).

Los tres giroscopios están colocados en un patrón ortogonal similar, midiendo la posición rotacional referida a un sistema de coordenadas seleccionado de forma arbitraria.

Mediante el seguimiento de la velocidad angular actual del sistema y de la aceleración lineal medida con el sistema de movimiento, es posible determinar su aceleración lineal en el sistema de referencia.

2.4.1 Giroscopio

El giroscopio es un dispositivo que mide o mantiene el movimiento rotacional. Los giroscopios MEMS (sistema micro electromecánico) son sensores pequeños que miden la velocidad angular [9]. Las unidades de velocidad angular se miden en grados por segundo (°/s) o revoluciones por segundo (RPS). La velocidad angular es simplemente la medición de la velocidad de rotación.

Miden la velocidad angular (ω) usando el efecto Coriolis. Cuando se hace girar el giroscopio, una pequeña masa se desplaza con los cambios de velocidad angular. Un sensor capacitivo mide esa velocidad angular que es proporcional al recorrido de dicha masa.

A partir de la variación del ángulo de giro es posible determinar el grado de inclinación del dispositivo en el que se encuentra montado.

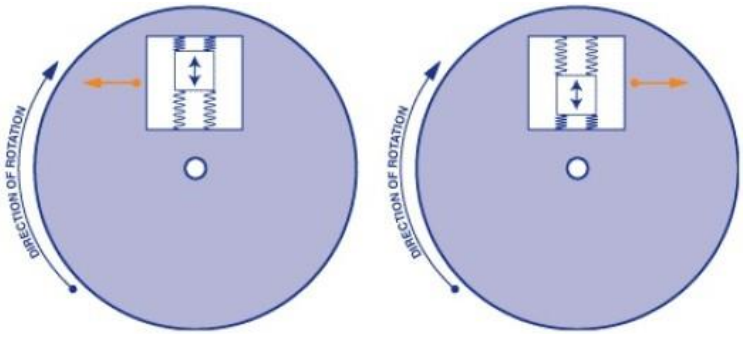

Figura 2.4: Funcionamiento interno de un sensor giroscópio MEMS.

2.4.2 Acelerómetro

El acelerómetro es un dispositivo capaz de medir aceleraciones, es decir, la variación en la velocidad por unidad de tiempo, si se tiene en cuenta la segunda ley de Newton, que dice: $F = m\ a$ y suponemos una masa (m) constante, resulta que la aceleración medida (a) será proporcional a la fuerza (F) que se le aplique al acelerómetro. [10]

Esas fuerzas pueden ser estáticas o dinámicas, las estáticas incluyen la gravedad, mientras que las dinámicas pueden incluir vibraciones y/o movimiento, con el empleo de filtros paso bajo o paso alto se podrá discriminar la componente no deseada.

Generalmente, los acelerómetros contienen placas capacitivas internamente, algunas de estas placas son fijas, mientras que otras están unidas a resortes minúsculos que se mueven internamente conforme las fuerzas de aceleración que actúan sobre el sensor. Como estas placas se mueven en relación el uno al otro, la capacitancia entre ellos cambia. A partir de estos cambios en la capacitancia, se puede determinar la aceleración. [11]

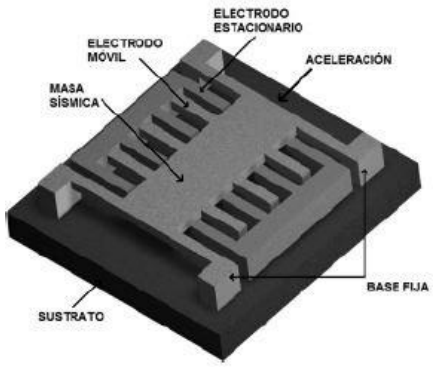

Figura 2.5: Sistema microeléctromecánico para la aceleración en un eje.

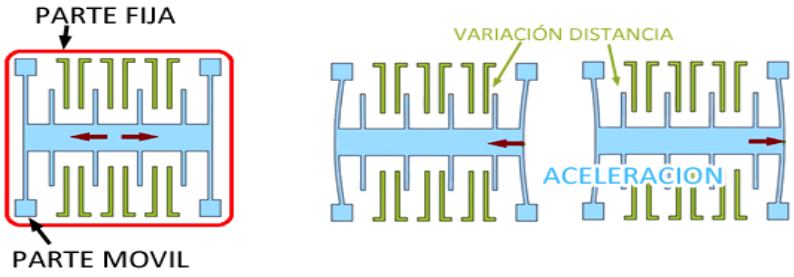

Figura 2.6: Fundamento físico de un acelerómetro

2.5 Transmisión por radiofrecuencia

Para enviar la información del GPS de la boya a tierra se ha optado por un sistema de comunicación de radiofrecuencia. Por radio se entiende la transmisión de señales a través del espacio, mediante ondas electromagnéticas, sin que haya conexión física entre transmisor y receptor. El medio de propagación de las ondas electromagnéticas es, en este caso, el aire o el vacío.

En el trabajo con sistemas radioeléctricos es frecuente emplear el término radiofrecuencia (RF), y por tal, se entiende la frecuencia a la que la radiación de energía electromagnética es útil para propósitos de comunicación. Así, las radiofrecuencias abarcan desde unos pocos KHz hasta más de 100 GHz.

El envío de la información vía radio tiene algunas ventajas frente a las comunicaciones por línea:

- Movilidad: si el transmisor o el receptor se mueven las comunicaciones por línea no sirven.
- Accesibilidad: permite llegar a zonas poco accesibles con el uso de satélites o propagación a bajas frecuencias.
- Menos obras: no son necesarias las obras del canalizado de los cables ni los permisos que esto requiere.

En este proyecto, se utilizará una frecuencia de transmisión de 433 MHz que corresponde a UHF (Ultra High Frecuency) y una potencia transmitida de 1 W.

3 Material

3.1 Sensor de presión atmosférica

Para incrementar la precisión de las mediciones, se ha incorporado un sensor de presión atmosférica, que tomará medida de la presión en reposo y así se podrá calcular la diferencia de presiones y por tanto la altura relativa.

El sensor elegido es el BMP180 de tecnología piezo-resistiva; el BMP180 es un sensor barométrico digital de alta precisión y baja potencia, que incorpora un sensor de temperatura para compensar sus efectos en la medición de la presión barométrica.

El rango de medición es de 300hPa a 1110 hPa, equivalente a una altitud de -500m a 9000m sobre el nivel del mar. La precisión es configurable, desde 0.06hPa (0.5 metros) en el modo de bajo consumo, a 0.02hPa (0.17 metros) en el modo de alta precisión. [13]

Sus características principales son:

- *Consumo promedio de 0.1µA en stand-by, y 650µA durante la medición, lo que supone un consumo promedio de 5µA tomando 1 muestra por segundo en precisión estándar.*
- *El tiempo de respuesta es de unos 5ms en resolución estándar, y 17ms en alta resolución.*
- *Tensión de alimentación: 1,8 – 3,6 V.*
- *Conexión I2C.*
- *0,25m de resolución mínima.*

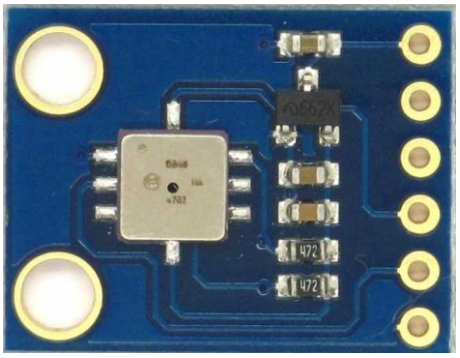

Figura 3.1: Sensor bmp085 usado en el proyecto.

3.2 Acelerómetro MPU6050

El MPU-6050 es una unidad de medición inercial (IMU) de seis grados de libertad (6DOF) fabricado por Invensense, que combina un acelerómetro de 3 ejes y un giroscopio de 3 ejes.

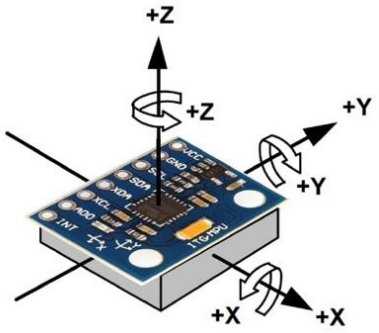

Figura 3.2: Sensor MPU6050 usado en el proyecto.

Las principales características por las que se usó este sensor fueron:

- *Tensión de alimentación: 2,4 – 3,6 V*
- *Rango del acelerómetro ajustable entre ±2g, ±4g, ±8g, y ±16g.*
- *Rango del giroscopio ±250, ±500, ±1000, y±2000°/sec.*
- *Consumo de 3.5mA, con todos los sensores y el DMP(Digital Motion Processor) activados.*
- *Conexión I2C*
- *Conversor ADC de 16 bit.*
- *Corriente en standby 5μA.*

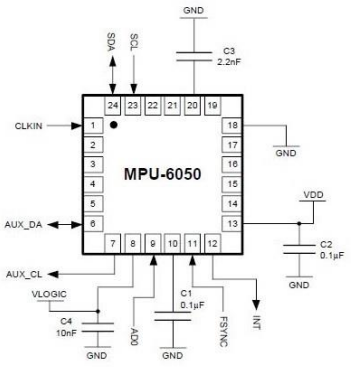

Figura 3.3: Diagrama de conexión MPU6050.

3.3 El GPS

Es importante conocer la ubicación de la boya o posibles boyas en el océano, ya que nos servirá para calcular la distancia a la que están de la estación terrena y a qué velocidad se propagan.

El sensor que utilizaré para la boya es el módulo GY-GPS6MV2, que viene con un módulo de serie U-Blox NEO 6M equipado en el PCB, una EEPROM con configuración de fábrica, una pila de botón para mantener los datos de configuración en la memoria EEPROM, un indicador LED y una antena cerámica. [14]

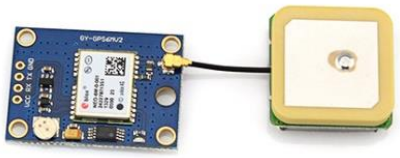

Figura 3.4: Sensor GPS NEO VI usado en el proyecto.

Sus principales características son:

- *Ultra sensibilidad: -165dBm*
- *Tensión de alimentación: 2,7 – 3,6 V*
- *Precisión de 2,5m en velocidad 0,1m/s*
- *Soporta estándares WAAS/EGNOS/MSAS/GAGAN*
- *Frecuencia de actualización 5Hz*
- *Velocidad de desplazamiento máxima: 500m/seg*
- *Protocolo NMEA (a 9600bps)*
- *Rango de temperatura: -40º - 85º C*
- *Cumple estándar RoHS*
- *Tamaño reducido 30mm x20mm x 11.4mm*
- *Conexión I2C*

El GPS proporciona datos según el protocolo NMEA (National Marine Electronics Asociation), las cuales son sentencias estándares para la recepción de datos GPS.

Una de ellas y la más usada son las sentencias $GPRMC, las cuales tienen la siguiente estructura:

$GPRMC,044235.000,A,4322.0289,N,00824.5210,W,0.39,65.46,020615,,,A*44

Donde si analizamos la trama de este ejemplo y basándose en el protocolo NMEA, podríamos determinar las siguientes variables:

- **044235.000** representa la hora GMT (04:42:35)

- "**A**" es la indicación de que el dato de posición está fijado y es correcto. "**V**" sería no válido

- **4322.0289** representa la longitud (43º 22.0289´)

- **N** representa el Norte

- **00824.5210** representa la latitud (8º 24.5210´)

- **W** representa el Oeste

- **0.39** representa la velocidad en nudos

- **65.46** representa la orientación en grados

- **020918** representa la fecha

De estos datos sólo se utilizará de forma práctica la latitud, longitud y la velocidad.

3.4 Módulo LoRa

Para la transmisión por radiofrecuencia, se ha elegido el módulo LoRa (Long Range) E32. La elección de este modelo en concreto ha sido porque ofrecía una potencia de 1W con un bajo consumo, y eso podría dar mucha versatilidad ya que se podrían conseguir distancias teóricas de hasta 8 Km en línea de visión directa. [15]

El módulo transmitirá a una frecuencia de 433 MHz aunque también existe una versión que funciona a 868 MHz.

Sus características principales son:

- *Rango de frecuencia 410-441 MHz*
- *Voltaje 2,3 – 5,5 V*

- *Potencia de transmisión 30dBm*
- *Velocidad de transmisión 0,3 – 1,2 – 2,4 – 4,8 – 9,6 – 19,2 Kbps*
- *Consumo standby 2,0 μA*
- *Corriente transmitida 120mA@20dBm*
- *Buffer de transmisión de 512 bytes*
- *Sensibilidad -138dBm @0,3 Kbps*
- *Antena tipo SMA*
- *Temperatura de operación -40 a 85ºC*

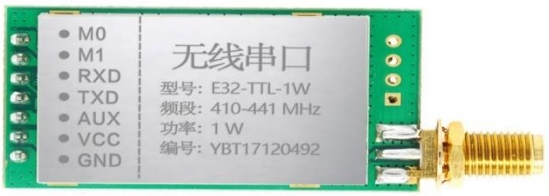

Figura 3.5: Módulo LoRa E32 TTL

El módulo cuenta con dos pines de configuración M1 y M0, que sirven para cambiar el modo de operación del módulo.

Mode (0-3)	M1	M0	Mode introduction	Remark
Mode 0 Normal	0	0	Los canales UART e inalámbricos se abren, transmisión transparente activada.	El receptor debe funcionar en el modo 0 o 1.
Mode 1 Wake-up	0	1	Los canales UART e inalámbricos son abiertos. La diferencia entre el modo normal y el "wake up" es que este añadirá código de preámbulo automáticamente antes del paquete de datos, por lo que podrá despertar al receptor que está funcionando en el modo 2.	El receptor puede funcionar en modo 0, modo 1 o modo 2.
Mode 2 Power-saving	1	0	Comunicación UART desabilitada. El módulo inalámbrico funciona en modo WOR (wake on radio). Abrirá la comunicación UART y transmitirá, después de recibir los datos inalámbricos.	El transmisor debe funcionar en el modo 1, transmitir no está permitido en este modo
Mode 3 Sleep	1	1	Ajuste de parámetros.	

Figura 3.6: Modos de operación LoRa

Para este proyecto, se configurará el módulo con los dos pines a tierra, lo que inducirá el modo 0, aunque en un futuro, si el dispositivo tiene un consumo elevado, podría ponerse el modo 2 "power-saving"

3.5 Arduino UNO

Arduino es una plataforma computacional física open-source basada en un microcontrolador Atmega328 [16] y un entorno de desarrollo que implementa el lenguaje Processing/Wiring.

El hardware de la platadorma es una placa compuesta por entradas y salidas analógicas/digitales que permiten la programación de sensores y actuadores.

La placa fue elegida por varios motivos.

- Simple, ya que hay otros microcontroladores con características parecidas pero no son tan universales.
- Bajo coste, comparada con Raspberry Pi o Arduino Due por ejemplo.
- Varias entradas analógicas, para conectar todos los sensores era necesario que tuviese varias ya que todos los sensores son analógicos.
- Bajo consumo energético.

Sus características principales son:

- *Voltaje operativo: 5V.*
- *Voltaje de entrada: 7 – 12 V.*
- *Pines I/O digital: 14 (de las cuales 6 son salidas PWM).*
- *Pines de entradas analógicas: 6.*
- *Memoria Flash: 32KB (ATmega328) de los cuales 0,5KB son usados por Bootloader.*
- *SRAM: 2KB (ATmega328).*
- *Velocidad del reloj: 16MHz.*

Figura 3.7: Placa Arduino Uno R3

4 Software y diseño

4.1 Diseño

La idea del proyecto es una boya que recopila datos provenientes de los sensores, los envía la placa Arduino por radiofrecuencia mediante el módulo LoRa a un receptor que se encontrará en la estación de tierra, una vez allí, la placa enviará mediante una conexión puente Arduino – Matlab los datos al ordenador.

Siempre se intentará evitar al máximo cualquier procesado de datos en las placas, ya que esto ayudará a que no fallen en la transmisión. Además, es mejor procesar los datos con Matlab ya que al ser una gran cantidad, se necesita un software potente que permita hacer todos los cálculos en tiempo real.

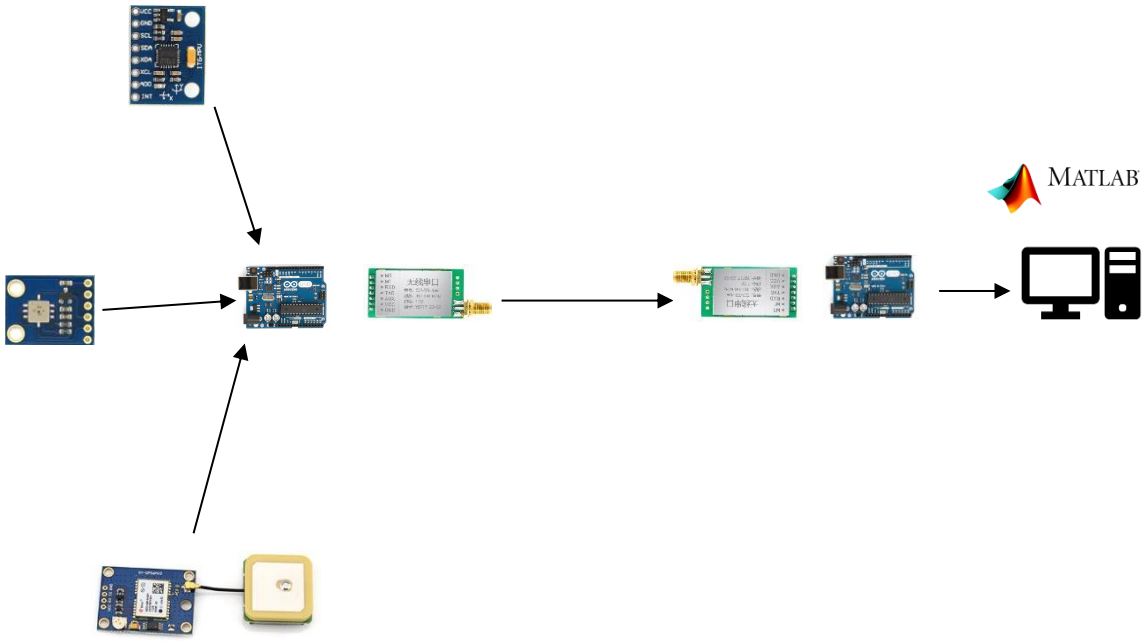

Figura 4.1: Esquema general del sistema

4.2 Arduino

El software de la plataforma Arduino se programa mediante un lenguaje propio (de extensión .ino) basado en el lenguaje de alto nivel Processing y en C, soportando muchas de las funciones de este.

El código implementado en Arduino, tendrá la funcionalidad de recoger los datos de los sensores, y enviarlos a través del módulo LoRa, hasta la otra placa.

Para poder llevar el código a cabo, ha sido necesario implementar las librerías correspondientes a cada sensor, además de las librerías comunes para la comunicación serie, etc…

4.2.1 Comunicación I2C

I2C es un bus de comunicaciones en serie, su nombre viene de Inter-Integrated Circuit (Inter-Circuitos Integrados). I2C usa solo 2 cables, uno para el reloj (SCL) y otro para el dato (SDA). Esto significa que el maestro y el esclavo envían datos por el mismo cable, el cual es controlado por el maestro, que crea la señal de reloj. I2C no utiliza selección de esclavo, sino direccionamiento. La velocidad es de 100 kbit/s en el modo estándar, aunque también permite velocidades de 3.4 Mbit/s. [17]

La principal característica de I^2C es que utiliza dos líneas para transmitir la información: una para los datos y otra para la señal de reloj. También es necesaria una tercera línea, pero esta sólo es la referencia (masa). Como suelen comunicarse circuitos en una misma placa que comparten una misma masa esta tercera línea no suele ser necesaria.
Las líneas se llaman:

- SDA: datos
- SCL: reloj
- GND: tierra

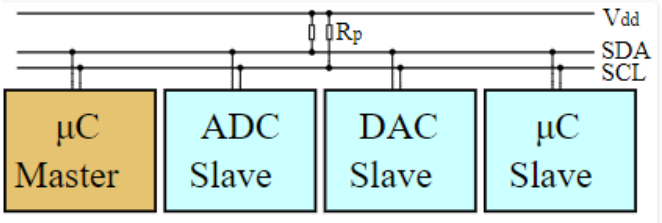

Figura 4.2: Esquema conexión SDA SCL

Los dispositivos conectados al bus I2C tienen una dirección única para cada uno. También pueden ser maestros o esclavos. El dispositivo maestro inicia la

transferencia de datos y además genera la señal de reloj, pero no es necesario que el maestro sea siempre el mismo dispositivo, esta característica se la pueden ir pasando los dispositivos que tengan esa capacidad. Esta característica hace que al bus I2C se le denomine bus multimaestro.

El proceso de comunicación en el bus I2C es:

- El maestro comienza la comunicación enviando un patrón llamado "start condition". Esto alerta a los dispositivos esclavos, poniéndolos a la espera de una transacción.
- El maestro se dirige al dispositivo con el que quiere hablar, enviando un byte que contiene los siete bits (A7-A1) que componen la dirección del dispositivo esclavo con el que se quiere comunicar, y el octavo bit (A0) de menor peso se corresponde con la operación deseada (L/E), lectura=1 (recibir del esclavo) y escritura=0 (enviar al esclavo).
- La dirección enviada es comparada por cada esclavo del bus con su propia dirección, si ambas coinciden, el esclavo se considera direccionado como esclavo-transmisor o esclavo-receptor dependiendo del bit R/W.
- Cada byte leído/escrito por el maestro debe ser obligatoriamente reconocido por un bit de ACK por el dispositivo maestro/esclavo.
- Cuando la comunicación finaliza, el maestro transmite una "stop condition" para dejar libre el bus. [18]

Figura 4.3: Protocolo transmisión de datos I2C

En cada placa Arduino los pines SDA y SCL varían, en este caso son los pines analógicos de Arduino A4 y A5.

4.2.2 Conexionado

Para el conexionado, hay que tener en cuenta que el acelerómetro y el barómetro, funcionan con comunicación I2C, por tanto es necesario conectar los dos módulos a los mismos pines (A4 y A5).

El módulo LoRa y GPS utilizan comunicación serie del tipo transmisor receptor. Para no saturar los mismos puertos, en el código se han definido los pines 10 y 3 como receptores (RX) y los pines 11 y 4 como transmisores (TX), de esta manera sólo hay que conectar los sensores cruzando cables de transmisión y recepción.

En el Arduino transmisor (fig 4.4) están conectados el barómetro BMP180, acelerómetro MPU6050, el módulo GPS y el módulo LoRa.

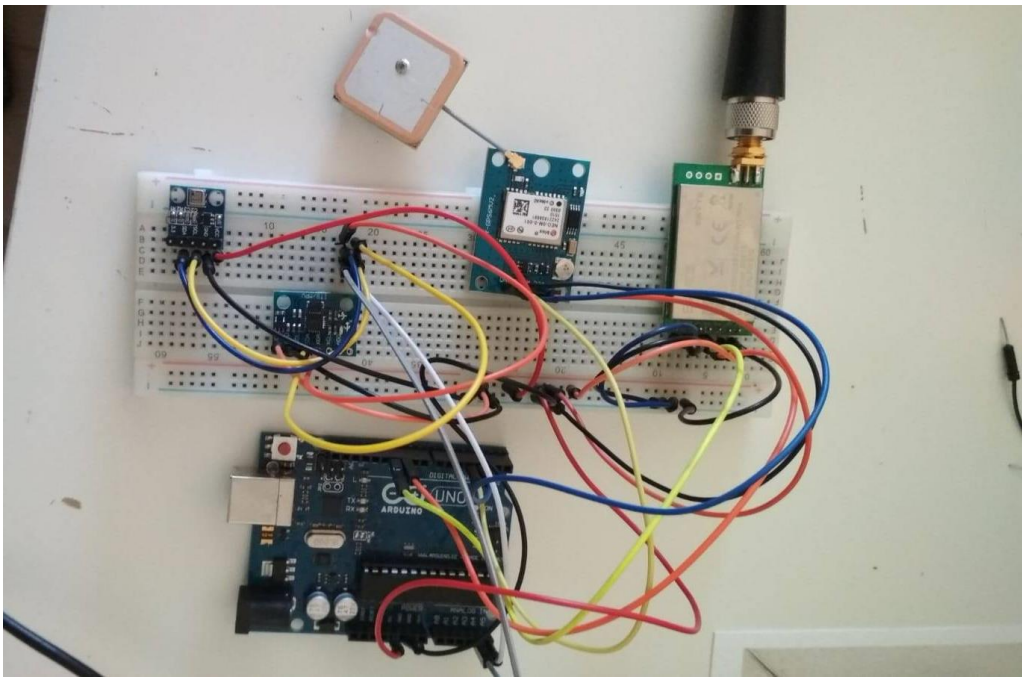

Figura 4.4: Conexión Arduino transmisor

En la placa receptora, quedaría conectado el módulo LoRa para recibir los datos, y el módulo GPS, que proporciona coordenadas de referencia para calcular la distancia.

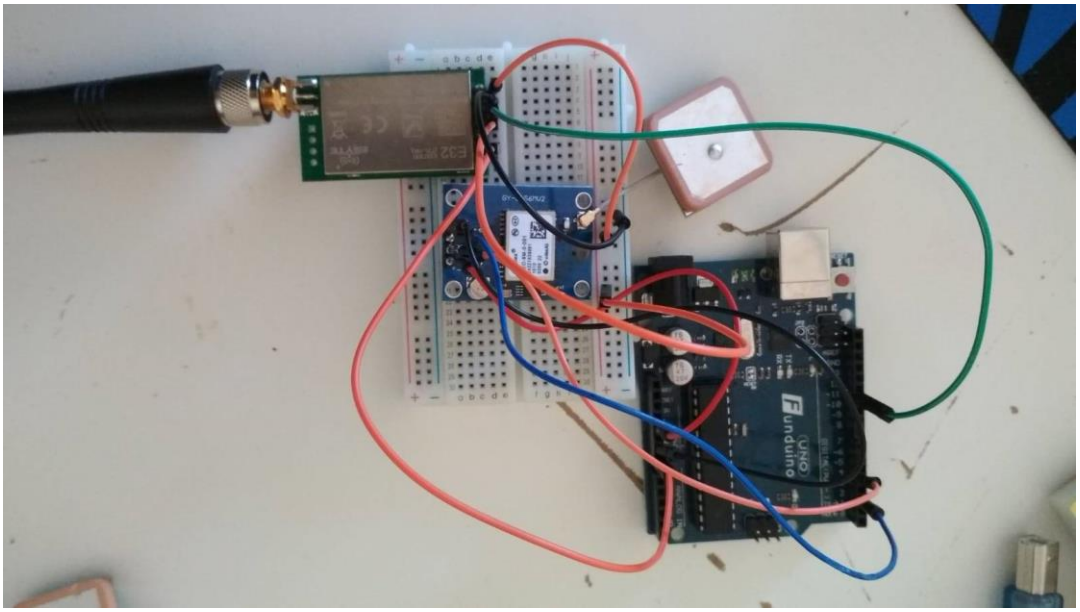

Figura 4.5: Conexión Arduino receptor

4.2.3 Código

Para implementar el código, he necesitado las siguientes librerías:

- Wire.h para la comunicación I2C
- SoftwareSerial.h para la comunicación serial
- SFE_BMP180 para el barómetro
- TinyGPS para la comunicación GPS

Es importante organizar cuando empieza la comunicación serie con cada módulo ya que es necesario que el módulo esté listo para intercambiar información.

```
//MEDIDAS INICIALES DEL BMP180
  if (bmp180.begin()){
status = bmp180.startTemperature();//Inicio de lectura de temper
if (status != 0)
{
  delay(status); //Pausa para que finalice la lectura
  status = bmp180.getTemperature(T);//Obtener la temperatura
  if (status != 0)
  {
    status = bmp180.startPressure(3);//Inicio lectura de presió
    if (status != 0)
    {
      delay(status);//Pausa para que finalice la lectura
      status = bmp180.getPressure(P,T);//Obtenemos la presión
      if (status != 0)
      {
        Po=P; //Asignamos el valor de presión como punto de refe
      }}}
  else
  {
    Serial.println("Error al iniciar el BMP180");
    while(1); // bucle infinito
  }
Wire.begin();              //Iniciando I2C
sensor.initialize();       //Iniciando el MPU6050
Serial.begin(9600);        //Iniciamos el puerto serie
mySerial.begin(9600);      //Puerto serie LoRa
serialgps.begin(9600);     //Iniciamos el puerto serie del gps
```

Figura 4.6: Inicialización comunicaciones serie

En primer lugar, se inicializa el módulo BMP para que tome las medidas en reposo, y después se hace lo mismo con el resto de sensores en función del orden en el que se van a utilizar.

Todos los sensores están configurados a una velocidad de transmisión de 9600 baudios, esta velocidad ha sido elegida porque es suficientemente rápida para el proyecto, además, con velocidades mayores, puede haber problemas con la transmisión, al no estar todos sincronizados.

Por último, se han organizado los datos para que se transmitan en un array, con el siguiente orden:

```
float sensores[12]={A, latitude, longitude, gps.f_altitude(),  gps.f_speed_kmph(), gps.satellites(), gz_si};
```

Figura 4.7: Array de datos enviados por Arduno

4.3 Matlab

MATLAB (abreviatura de *MATrix LABoratory*, "laboratorio de matrices") es una herramienta de software matemático que ofrece un entorno de diseño

integrado (IDE) con un lenguaje de programación propio (lenguaje M). Combina un entorno de escritorio perfeccionado para el análisis iterativo y los procesos de diseño con un lenguaje de programación que expresa las matemáticas de matrices y arrays directamente.

El software cuenta con un elemento básico de datos es el arreglo que no requiere de dimensionamiento previo. Esto permite resolver muchos problemas computacionales, específicamente aquellos que involucren vectores y matrices, en un tiempo mucho menor al requerido para escribir un programa en un lenguaje escalar no interactivo tal como C o Fortran.

Se ha elegido este software debido a su gran capacidad de procesado de datos a alta velocidad.

4.3.1 Comunicación puerto serie

Para transmitir los datos desde la placa Arduino de la estación terrena, hasta el ordenador con MATLAB, es necesario establecer una comunicación puerto serial.

El puerto serial envía y recibe bytes de información un bit a la vez. Aun y cuando esto es más lento que la comunicación en paralelo, que permite la transmisión de un byte completo por vez, este método de comunicación es más sencillo y puede alcanzar mayores distancias. La comunicación serial se utiliza para transmitir datos en formato ASCII. Para realizar la comunicación se utilizan 3 líneas de transmisión: (1) Tierra (o referencia), (2) Transmitir, (3) Recibir. Debido a que la transmisión es asincrónica, es posible enviar datos por una línea mientras se reciben datos por otra [19.]

Para hacerlo, es necesario definir el puerto en el que está conectado el Arduino y la velocidad de transmisión en baudios.

```
delete(instrfind({'Port'},{'COM4'}))
puerto_serial = serial('COM4')
puerto_serial.BaudRate = 38400;

fopen(puerto_serial)
fwrite(puerto_serial, 'a')
```

Figura 4.8: Inicialización comunicación serie

En este caso, Arduino está conectado al puerto 4, y la velocidad de transmisión es de 38400 baudios, para la comunicación entre Arduino y Matlab se ha configurado una velocidad más alta, para que el programa pueda recibir la información más rápidamente.

4.3.2 Estructura del código

El código en MATLAB, recibirá los datos ordenados en un array de 7 columnas y después procesará los datos que necesite.

Para ello, se dividirán los datos según de los sensores de los que provienen:

- BMP180: los datos recibidos son de presión, por lo tanto lo primero es convertirlos en altura sobre el nivel del mar, cada vez que reciba un nuevo dato, calculará la diferencia entre el anterior, y así sucesivamente.

- MPU6050: el sensor proporciona datos de giro y aceleración en todos los ejes, pero yo solo se va a utilizar el valor de aceleración en el eje Z (gravedad). Cada vez que recibe un nuevo dato, un algoritmo se encarga de calcular cuanta distancia ha recorrido en función de la variación de la aceleración.

- GPS: el sensor proporciona variedad de datos, pero para este proyecto solo será necesario la latitud, longitud y velocidad.
  La latitud y longitud servirán para calcular la distancia hasta las coordenadas de la costa.

Una vez hechos los cálculos, se muestran en una GUI de MATLAB, para que sean más fáciles de ver e interpretar.

5 Resultados

A la hora de tomar las primeras medidas, lo primero es asegurarse de que el Arduino transmisor está recibiendo la información de todos los sensores correctamente, para ello, se hace una lectura en bruto de los datos captados por los sensores.

En la siguiente captura se muestran los datos en el siguiente orden: altura relativa, latitud, longitud, altura sobre el nivel del mar, velocidad de GPS, satélites, aceleración en los ejes x, y, z, y ángulo de giro en los 3 ejes también.

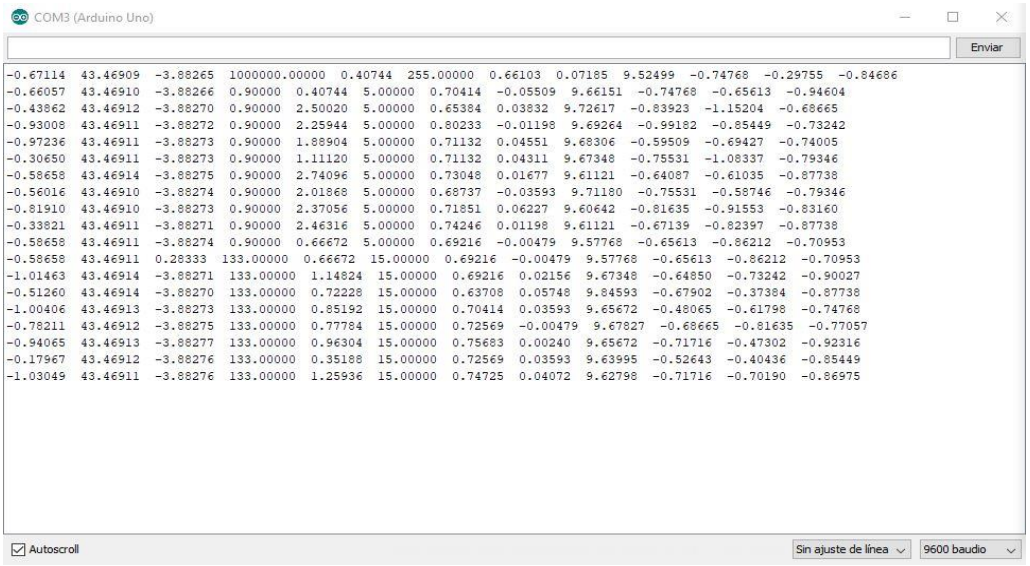

Figura 5.1: Datos en bruto Arduino transmisor

En esta captura, aparecen todos los datos que Arduino está recibiendo de los sensores, pero como ya se ha mencionado anteriormente, al no utilizar todos los datos, sólo se enviarán los necesarios (fig 4.7).

Pero de momento no se necesitan todos los datos, por lo tanto, para ahorrar trabajo de procesamiento y de transmisión, solo se transmitirán los datos útiles. Dejando los datos restantes sin utilizar, pero con fácil acceso para una posible ampliación del sistema.

En segundo lugar, se miden los datos recibidos en el otro terminal Arduino, en el receptor se puede observar que sólo están los datos que interesan:

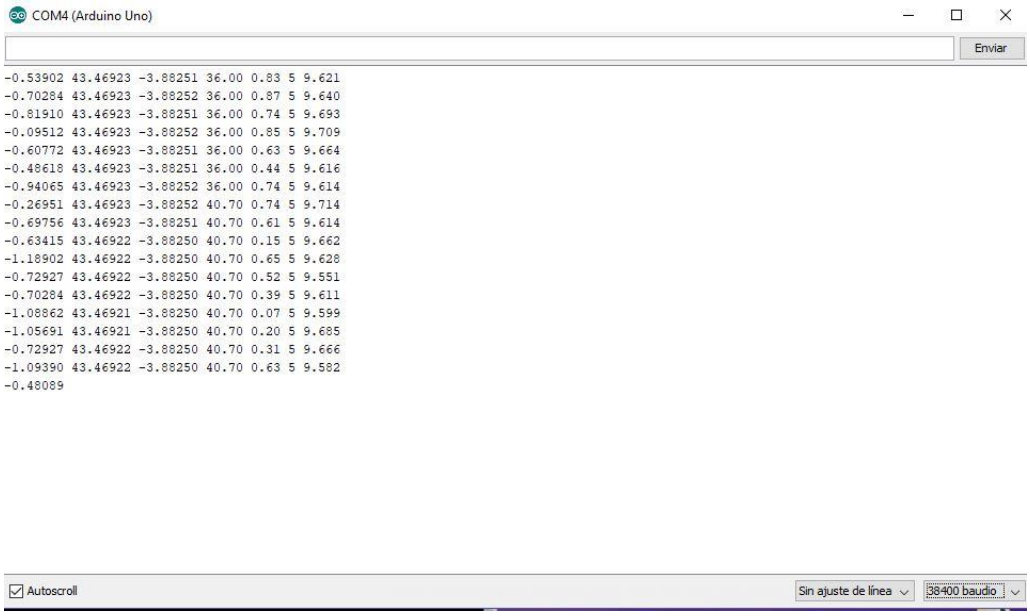

Figura 5.2: Datos en bruto Arduino receptor

El orden de los datos recibidos es: altura relativa, latitud, longitud, altura sobre el nivel del mar, velocidad del GPS y aceleración en el eje z.

Cabe destacar que la placa transmisora envía datos cada 100 ms.

5.1 Error en las medidas

Una vez las placas están funcionando correctamente, se pone en marcha el programa de Matlab de recopilación de datos de puerto serie. Para hacer la primera prueba, se representa la altura relativa medida por el módulo BMP180 respecto al tiempo y le aceleración en el eje Z respecto al tiempo también.

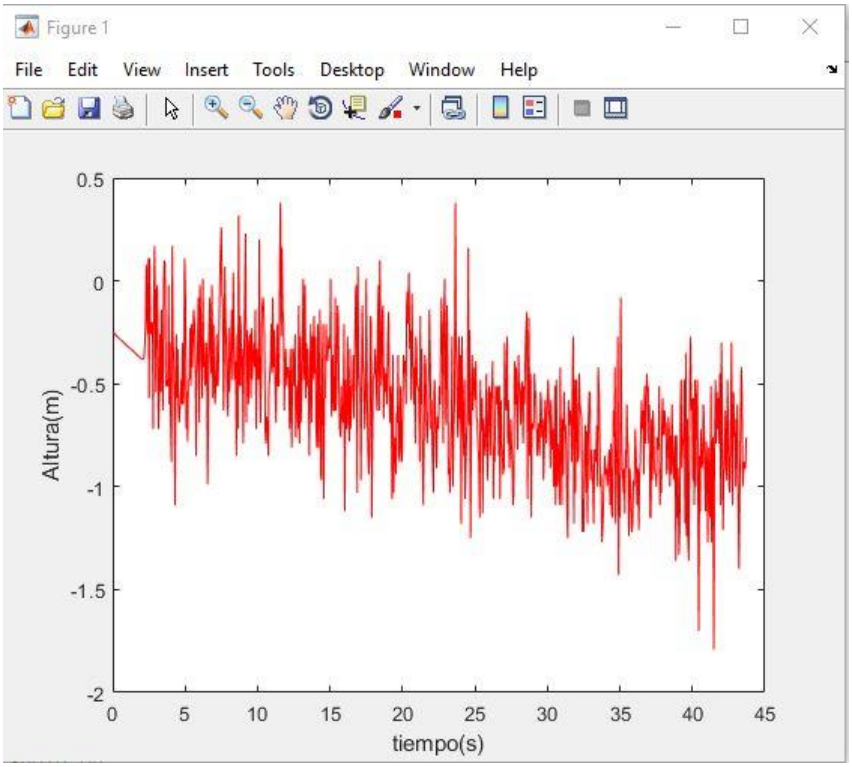

Figura 5.3: Gráfica Matlab altura relativa respecto al tiempo

Los datos proporcionados por el barómetro son bastante ruidosos, esto puede ser debido al cableado, los puertos analógicos, etc… Más adelante se intentará poner una solución para filtrar el ruido.

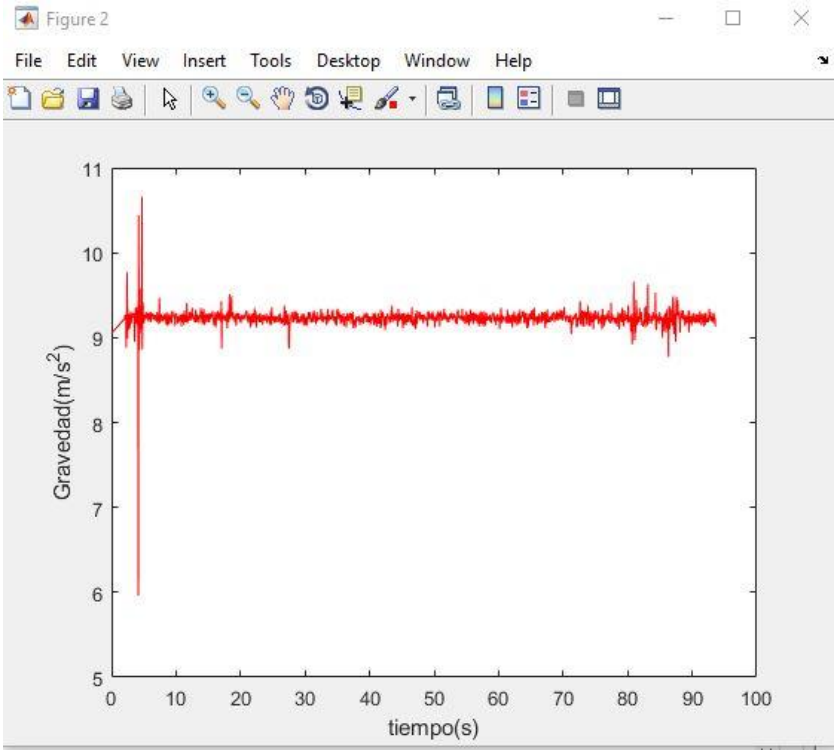

Figura 5.4: Gráfica Matlab aceleración eje Z respecto al tiempo

Los valores proporcionados por el acelerómetro son bastante estables, haciendo pruebas de movimiento se comporta como cabría esperar.

Se puede apreciar a simple vista, que las dos medidas cuentan con ruido, pero las tomadas con el barómetro lo son mucho más. Esto probablemente sea debido a los cálculos que tiene que hacer el barómetro para obtener la altura a partir de la temperatura y la presión, que van acumulando error.

Antes de proceder a realizar los cálculos, sería mucho más preciso utilizar datos limpios de ruido en la medida de lo posible, por lo que procedí a buscar algún filtro que pudiese ser de utilidad.

5.2 Filtro Kalman

El filtro Kalman es un algoritmo recursivo es capaz de calcular el error de cada medida a partir de las medidas anteriores, eliminarlo y dar un valor más aproximado de la aceleración. El algoritmo del filtro es un proceso de dos pasos: el primer paso predice el estado del sistema, mientras que el segundo utiliza las mediciones de ruido para ajustar la estimación del estado del sistema. En cierto modo, es un algoritmo que aprende en cada iteración [20].

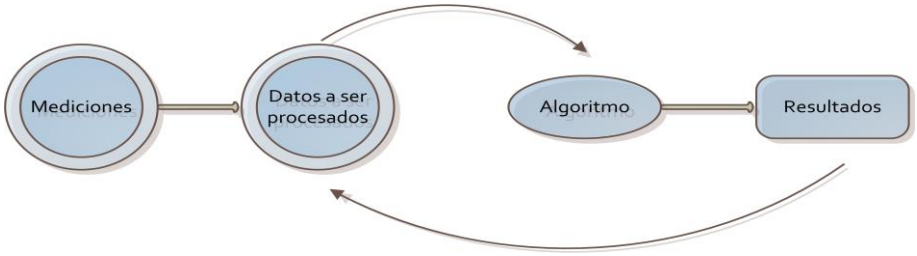

Figura 5.5: Esquema funcionamiento filtro Kalman

Al principio pareció una buena idea, pero éste algoritmo tiene una implementación compleja, y consumía muchos recursos del programa en Matlab, lo que entorpecía su ejecución en gran medida, por lo que se decidió buscar otro filtro.

5.3 Filtro exponencial

El filtro exponencial EMA[21] (Exponential Moving Average) consiste en obtener un valor filtrado a partir de una medición mediante la aplicación de la siguiente expresión:

$$S(n) = \alpha M + (1 - \alpha) * S(n - 1) \qquad (5.1)$$

Donde s(n) es el valor filtrado, s(n-1) es el valor filtrado anterior, y M es el valor muestreado de la señal a filtrar, y α es un factor entre 0 y 1.

El filtro presenta un aporte de información "nueva" a través de la medición M, y un efecto suavizado por el valor que aporta el filtrado anterior.

El resultado del filtro es una señal suavizada, donde la cantidad de suavizado depende del factor alpha:

- Un valor de $\alpha = 1$ proporciona una señal sin filtrar ya que prescinde del efecto de filtrado que proporciona la medición anterior.
- Un valor de $\alpha = 0$ provoca que el valor filtrado siempre sea 0 ya que prescinde de la información nueva.

Disminuir el factor, aumenta el suavizado de la señal, pero a costa de poder eliminar componentes frecuenciales significativas, por otro lado, disminuir el factor también aumenta el tiempo de respuesta, lo que produce un retraso entre

señal original y señal filtrada. [22]

Una vez aplicado este filtro en Matlab, graficando las medidas en crudo en color rojo, y las filtradas en azul, los resultados son los siguientes:

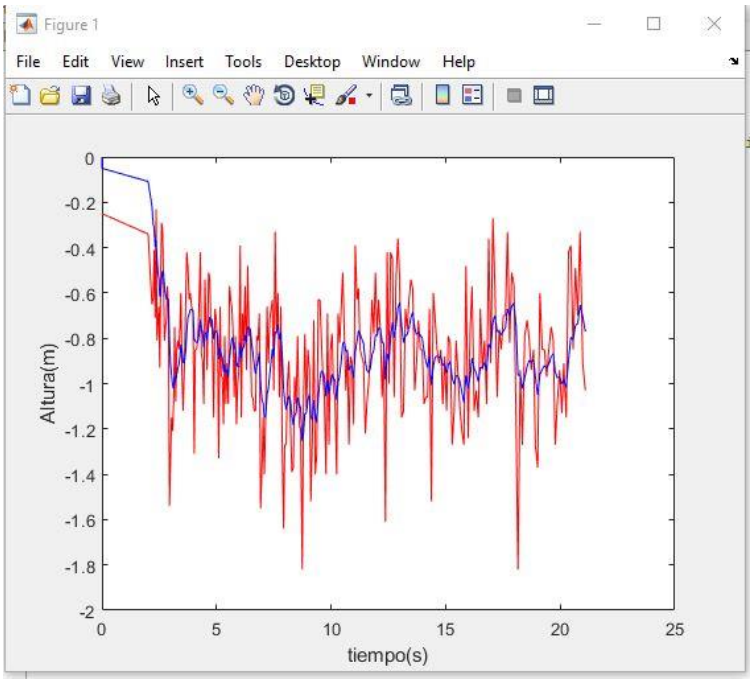

Figura 5.6: Gráfica Matlab altura relativa filtrada respecto al tiempo

En el caso de la altura, se puede apreciar a simple vista como suaviza los valores en gran cantidad. El sensor debería dar una media de 0 en reposo, aunque lo importante no es el valor en sí mismo, sino la diferencia entre el valor y los anteriores.

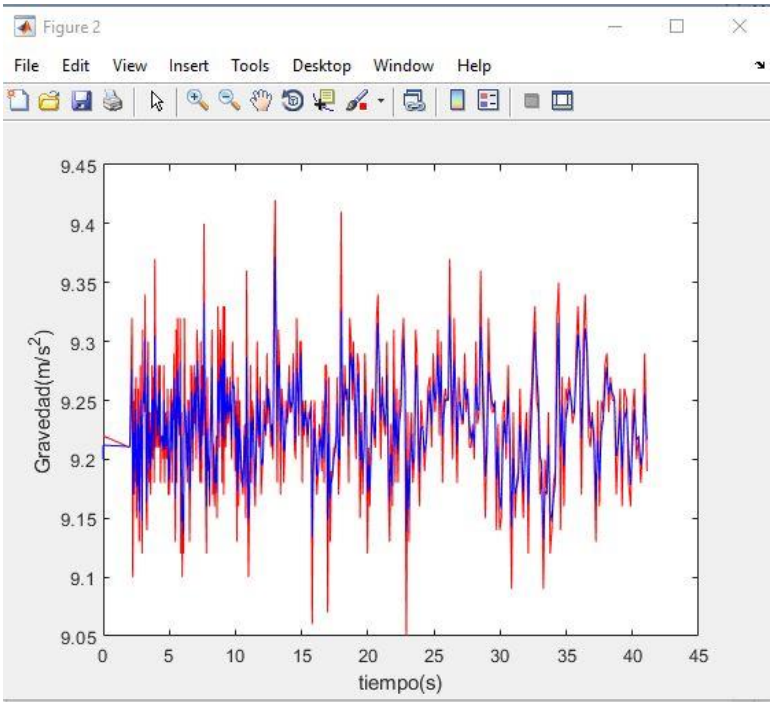

Figura 5.7: Gráfica Matlab aceleración eje Z filtrada respecto al tiempo

En el caso de la aceleración, ya estaba obteniendo unos valores constantes, por lo que el filtro no hace gran efecto en las medidas, aun así, suaviza los picos máximos de interferencia.

En éste caso, se han utilizado valores de alpha diferentes para cada medida ya que la aceleración de la gravedad presentaba una variación menor que la altura, por tanto se ha impuesto un valor de 0.2 para el barómetro y 0.6 para el acelerómetro.

Se puede apreciar una mejora significativa en los resultados, a cambio de una sencillez y eficiencia computacional muy altas.

El cálculo se puede realizar en una única línea de código, lo que no entorpece nada la ejecución.

```
alturabmp_f(j)=0.2*alturabmp(i)+(1-0.2)*alturabmp_f(i); %Filtro exponencial
g_f(j)=0.6*g(i)+(1-0.6)*g_f(i);
```

Figura 5.8: Filtro exponencial en MATLAB

5.4 MATLAB GUI

Con el fin de visualizar los datos de una forma más sencilla y compacta, se diseñó y programó una GUI en MATLAB, que representase los datos más relevantes.

Para programar una GUI, es necesario hacer dos pasos principalmente:

- Diseño de la GUI, es decir, cómo se van a mostrar los datos, las gráficas, botones, y demás componentes interactivos que se mostrarán al usuario final
- Adaptar el código de la boya al código generado por la GUI para que represente los datos donde corresponde.

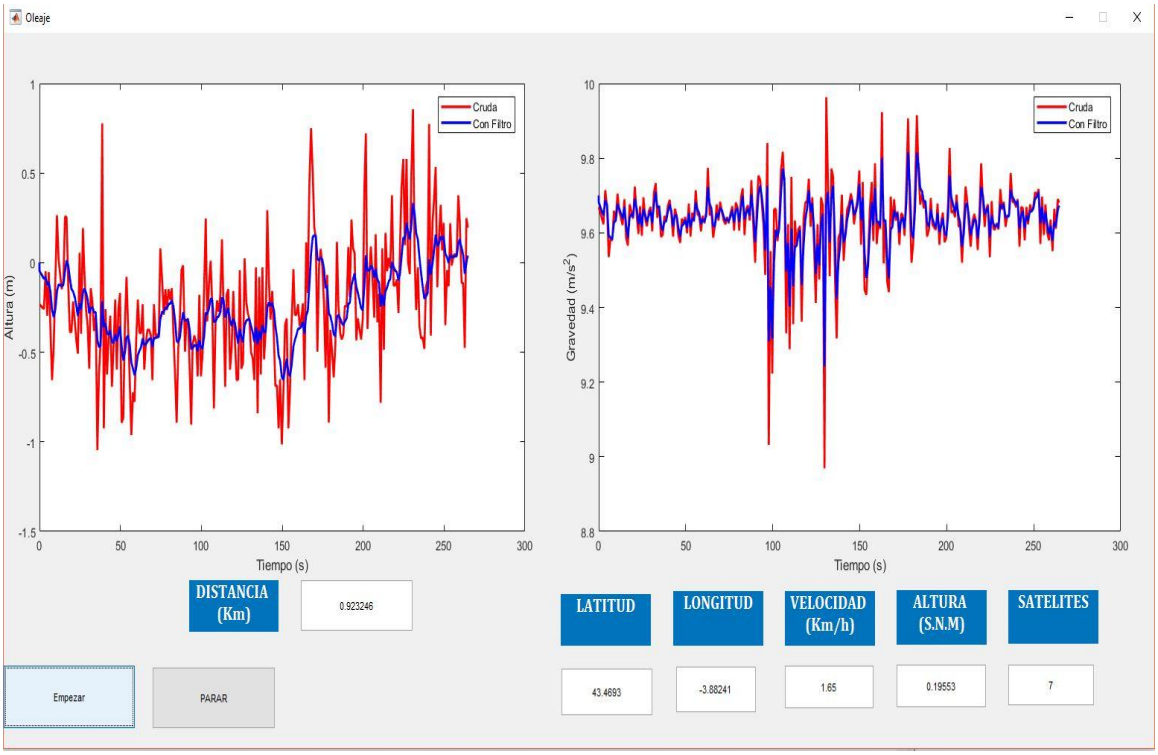

Figura 5.9: GUI MATLAB

Como se puede observar, la GUI tiene un botón para comenzar la toma de medidas y otro para detenerlas. Los valores se obtienen en tiempo real, es decir, que a medida que pasa el tiempo, la gráfica se hace más extensa.

La casilla distancia, se obtiene como resultado del cálculo de la distancia entre las coordenadas de la boya que se encuentra en el mar, y las de la estación terrena.

6 Conclusión y líneas futuras

6.1 Conclusión

El objetivo del trabajo era el desarrollo e implementación de un dispositivo susceptible de ser colocado en una boya marina, capaz de proporcionar datos cuyo tratamiento permitiera la cuantificación y predicción temporal del oleaje.

Se ha conseguido un prototipo que, a pesar del ruido, permite la obtención de medidas que el software analiza y procesa de manera que se puede adelantar el momento en el que acontecerá un tren de olas con un margen de error aceptable.

No se ha llegado a probar cual es la máxima distancia de transmisión alcanzable, porque se necesitaría una antena de recepción de 5dBi de ganancia para lograr los 8Km de las especificaciones, según el datasheet de LoRa. Además, no se ha podido encapsular el sistema para probarlo en mar abierto, esto podría cambiar mucho las medidas tomadas en tierra, ya que el mar en reposo tiene pequeñas oscilaciones que influirían en las medidas del acelerómetro sobretodo.

Después de la construcción del primer prototipo, el sistema permitiría vigilar las zonas cercanas a la costa con numerosas ventajas:

- Costes de producción reducidos. El sistema consta de una placa Arduino, dos sensores y un módulo transmisor, lo que permitiría tener mayor cantidad de boyas y tener una red de éstas, mejorando así la precisión de la predicción.

- Mejora y corrección de errores más efectivo y rápido al ser un sistema en tiempo real.

- Reducido tamaño, lo que permite transportar muchas boyas en un solo viaje, sin necesidad de un anclaje fijo al fondo marino al disponer de módulo GPS.

Respecto a los inconvenientes o desventajas del dispositivo, quisiera destacar los errores en la medición, sistemáticos, de apreciación o accidentales:

La calidad de los sensores utilizados, tienen un porcentaje de error de medición elevado y el prototipo no tiene un acabado final profesional.

Como conclusión final, el sistema puede ser viable, pero necesita un trabajo extra realizado probablemente por un equipo en el que cada uno esté especializado en un campo. Este proyecto abarca muchas materias distintas, como lo son la meteorología, la comunicación por radiofrecuencia, la electrónica… Y es muy complicado que una única persona sea capaz de tener el conocimiento necesario para el proyecto de cada una de ellas.

6.1 Líneas futuras

En un futuro, para hacer de éste un sistema viable, se proponen las siguientes mejoras:

- Algoritmo de predicción. Hasta ahora se ha estimado el tiempo que tardaría en llegar el tren de olas calculando su velocidad, pero ésta forma de hacerlo seguramente sería más óptima utilizando algún modelo de propagación del oleaje en fondo marino que ya esté estudiado y comprobado, y utilizar una técnica de estimación o predicción con sus ecuaciones.

Por el momento el sistema es capaz de medirlos con cierta precisión por lo que se podría probar alguno de los siguientes modelos de propagación en el mar para contrastar la diferencia de resultados.

Fórmula de Carrier-Greenspan:

$$Nc = 2.pi.H \,/\, g.T^2.ß^2 \qquad (6.1)$$

Donde H es la altura de la ola, g la aceleración de la gravedad, T el período de las olas, y ß un factor que describe el fondo del mar y su morfología. El número de Carrier-Greenspan (Nc) debe valer entre 0.09 y 4.8 para asegurar que la ola rompa bien [23].

Fórmula de Carrier-Greenspan 2:

$$a = arcsen\,[1 \,/\, (2n + 1).tanß] \qquad (6.2)$$

Donde n es un número entero, y ß el mismo factor descriptivo de la forma del fondo marino de la formula anterior, y "a" es el ángulo de incidencia óptimo para que la ola rompiente ofrezca una pared surfeable y no se cierre completamente al romper [24].

- Mayor velocidad en los puertos analógicos; actualmente con Arduino UNO tiene un conversor ADC con 10 bits, devolviendo números enteros entre 0 y 1023, como el ADC utiliza la tensión de referencia de la placa de 5V, la resolución sería de

$$5/1023 = 4.88\,mV$$

Con una precisión de +-2.44 mV
Precisión relativa

$$\frac{1}{1023} = \frac{4.88\,mV}{5\,V} = 0.000976\%$$

Mientras que Arduino DUE tiene una resolución de 12 bits, lo que significa mapear valores entre 0 y 4095, por lo tanto la resolución sería en este caso de

$$\frac{5}{4095} = 1.22\ mV$$

Por tanto una precisión de +-0.61 mV y una precisión relativa de

$$\frac{1}{4095} = \frac{1.22\ mV}{5\ V} = 0.000244\%$$

Significaría una cuarta parte de la precisión que se tiene actualmente, y dado que todos los sensores funcionan con puertos analógicos, supondría una mejora importante.

- Implementación de una placa de circuito impreso (PCB)[25] ya que actualmente todo el sistema está cableado, lo que a veces produce errores con la vibración de los cables, falsos contactos, etc…

Además, para una simulación real sería conveniente tener todo el conexionado bien hecho para asegurar que los errores no son debidos a malos contactos, humedad o salitre en el cobre por ejemplo.

- Implementación de un sistema de baterías, el único requerimiento sería que fuesen recargables y tuviesen una duración aproximada de 6 horas, para ello probablemente sería necesario el uso de baterías tipo LiPo y un previo cálculo del consumo real del sistema en funcionamiento. En un futuro otra posible ampliación sería la instalación de paneles solares.

- Sensores de alta calidad, en el mercado se pueden encontrar productos de alta calidad por un precio más, pero que proporcionan medidas de mayor calidad y fiabilidad.

Presupuesto

A continuación se muestra un presupuesto aproximado del coste de los materiales utilizados para el proyecto:

Artículo	Cantidad	Precio/ud (€)	Coste(€)
Placa Arduino R3	2	9,80	20,6
Módulo GPS NEO-6M	2	10,25	20,5
Módulo LoRa E32 TTL	2	7,5	15
Antenas tipo SMA 433MHz	2	3,5	7
Barómetro BMP180	1	3,90	3,9
Acelerómetro MPU6050	1	6,90	6,9
Cableado	varios	3	3
Total			76,9

Es importante tener en cuenta que muchos de estos productos han sido comprados en España, lo que ha aumentado su coste significativamente. Si se produjesen estas boyas de manera más eficiente, comprando los materiales al por mayor a China, se podrían abaratar hasta en un 50%.

Aun así, es un proyecto que con muy pocos recursos, podría obtener grandes resultados. Con un precio inferior a 100 € se ha logrado construir un prototipo capaz de enviar los medir y enviar los datos por radio.

Debido a su bajo coste, el proyecto sería de rápida y barata producción, además de ser fácilmente escalable.

Bibliografía

[1] ”Las Olas marinas”.URL https://mundodelmar.wordpress.com/2010/09/21/las-olas-marinas/

[2] “Movimiento armónico simple (M.A.S) | Fisicalab”. URL https://www.fisicalab.com/apartado/concepto-oscilador-armonico#contenidos

[3] “Teoría de las olas”. URL http://www.masmar.net/esl/Apuntes-N%C3%A1uticos/Oceanograf%C3%ADa/Caracter%C3%ADsticas-de-las-olas.-Longitud-de-onda,-altura,-amplitud,-direcci%C3%B3n

[4] “El sistema de posicionamiento global”. URL https://www.gps.gov/systems/gps/spanish.php

[5] “Navstar: GPS Satellite Network”. URL https://www.space.com/19794-navstar.html

[6] “Calcular la distancia entre dos puntos geográficos”. URL https://www.genbeta.com/desarrollo/como-calcular-la-distancia-entre-dos-puntos-geograficos-en-c-formula-de-haversine

[7] “Fórmula Barométrica”. URL http://hyperphysics.phy-astr.gsu.edu/hbasees/Kinetic/barfor.html

[8]”Unidad de medición inercial”. URL https://es.wikipedia.org/wiki/Unidad_de_medici%C3%B3n_inercial

[9] “Giroscopio MEMS”. URL http://cursos.mcielectronics.cl/giroscopio/

[10] “Acelerómetro sensor de movimiento o vibración – Ingeniería Mecafénix”. URL http://www.ingmecafenix.com/automatizacion/acelerometro/

[11] “Acelerómetro”. URL https://www.luisllamas.es/como-usar-un-acelerometro-arduino/

[12] “Radiofrecuencia – Apuntes Unican”. URL http://personales.unican.es/vallel/RC/

[13] “BMP180 – Arduino”. URL https://www.luisllamas.es/medir-presion-del-aire-y-altitud-con-arduino-y-barometro-bmp180/

[14] “Guide to NEO-6M”. URL https://randomnerdtutorials.com/guide-to-neo-6m-gps-module-with-arduino/

[15] “LoRa Module”. URL https://www.teachmemicro.com/e32-ttl-100-sx1278-lora-module/

[16] “Microcontrolador Atmega328”. URL
https://es.wikipedia.org/wiki/Atmega328

[17] “I2C – PIC comunicación serial - Microcontroladores”. URL
http://microcontroladores-mrelberni.com/i2c-pic-comunicacion-serial/

[18] “I2C – Aprendiendo Arduino”. URL
https://aprendiendoarduino.wordpress.com/2017/07/09/i2c/

[19] “Comunicación puerto serie – Luis Llamas”. URL
https://www.luisllamas.es/arduino-puerto-serie/

[20] “Diseño y utilización de filtros en MATLAB”. URL
https://quantdare.com/filtro-kalman.html

[21] “My UP’S projects – FILTRO DE MEDIA MÓVIL”. URL
http://myupproyects.blogspot.com/2015/08/filtro-de-media-movil.html

[22] “Filtro paso bajo y exponencial – Luis Llamas”. URL
https://www.luisllamas.es/arduino-paso-bajo-exponencial/

[23] “La física de las olas - tresquillas”. URL
http://www.tresquillas.com.ar/fisicadeolas.htm

[24] “Estelas en T.G – La física de las olas”. URL http://estelas-en-tg.blogspot.com/2006/08/la-fsica-de-las-olas.html

[25] “Circuito impreso – Wikipedia”. URL
https://es.wikipedia.org/wiki/Circuito_impreso